Impulsive and switching load balancing model and stability analysis

Tinglei Zhao*, Jianzhong Qiao, Shukuan Lin and Yanhua Wang

School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China

Load imbalance in a distributed system causes low efficiency. Aiming at the dynamics of resource and load running status, we propose an impulsive and switching load balancing model with time delay based on control theory. In order to describe various current states of a node, we construct corresponding sub-systems according to the dynamics of a node's resources. The sub-system switching is triggered by an impulsive signal which can decrease the communication overhead among nodes. The model reallocates loads in light of their real-time running statuses, which improves the efficiency of dynamic load balancing. We deduce the sufficient condition for asymptotic stability of the model by using the Lyapunov–Krasovskii function, and simulation by Linear Matrix Inequality (LMI) to verify effectiveness of the model. Experimental results produced by a shared platform based on block chain demonstrate that the proposed model can make the balanced system speedy, which verifies its feasibility.

Keywords: Asymptotic stability, dynamic load balancing, distributed system, impulsive and switching system, time delay.

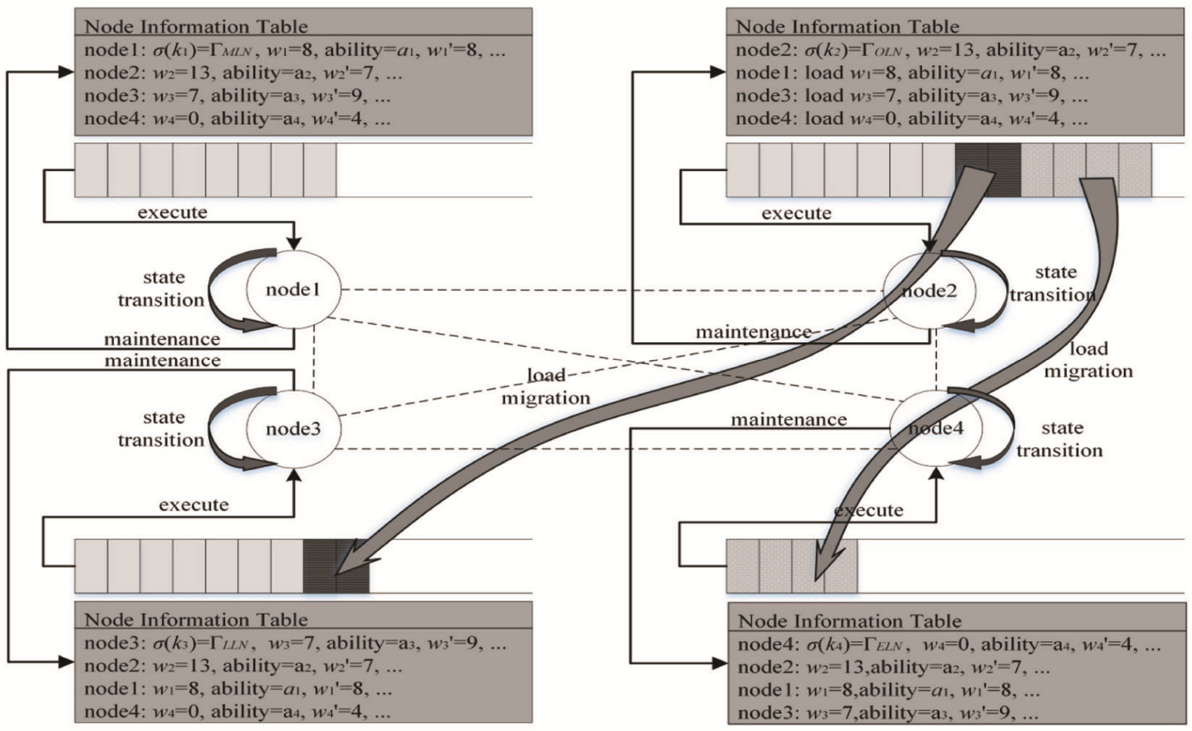
LOAD balance is an important research area in a distributed system, because the load imbalance among nodes seriously affects system efficiency. A distributed system consists of several nodes. Each node enters and exits dynamically and submits different new tasks many times according to the requirement[1]. When dealing with a task on a large scale, first, the system divides it into several sub-tasks and maps them into each node. Then each node returns a computed result, which is mixed together by the system. In the process, if some nodes are too busy to return a result on time, then the whole system will obviously fall into a waiting status. Load balance belongs to non-deterministic polynomial (NP) problem[2], but can reach a better level with various optimization strategies. Dhakal *et al.*[3] proposed a time delay model for distributed dynamic load balance system based on diffusion model. It belongs to stochastic load balance. Tang *et al.*[4] verified the stability of the model in theory. Tang *et al.*[5] deduced the related condition for system stability and obtained the load balance gain of the delay dynamic load balance (DDLB) system. However, the model has shortcomings: the stability condition obtained is only a sufficient condition; the suitable theory load balance gain is obtained by a serial complex calculation.

In order to efficiently solve the load imbalance problem among nodes in a distributed system, after synthesizing the characteristics of the impulsive switching system model[6,7] and time delay system model[8–10], we propose an impulsive and switching load balancing model with time delay. When starting the system, each node preserves the state information and load information broadcast by the other nodes. Each node computes its own state and switches to the corresponding sub-system according to its state. Then each node migrates the load by a load migration rule. Finally, the system reaches balance. Switching to the sub-system of each node is triggered by its own state, which avoids frequent message transmission. During load migration, we consider the processing ability of all nodes comprehensively, so that the system does not generate more over-load nodes after migration. Hence the load balancing efficiency improves. In this study, the contributions of our work can be concluded as: (1) Our model consists of 4 sub-systems which correspond to the states of a node at various moments; (2) A node takes its own state as an impulsive signal. The signal makes it switch to the corresponding sub-system. Migration proportion of a node is calculated according to its ownprocessing ability and that of others, which makes the model local and global. (3) We choose an appropriate Lyapunov–Krasovskii function, deduce the sufficient condition for asymptotic stability of each sub-system, and simulate by LMI to verify the effectiveness of the model; (4) We find that the sufficient and necessary condition of asymptotic stability of system on arbitrary switching is the stability of the each sub-system which shares the same Lyapunov function with others.

Impulsive and switching load balancing model with time delay

The system contains multiple continuous sub-systems. But it is discrete among each sub-system. So it belongs to a continuous-discrete system. Each node corresponds to a different sub-system according to its own state at different times. The sub-system triggers switching when node state changes. Various sub-systems run alternatively and

*For correspondence. (e-mail: zhaotinglei_0928@163.com)


Node Information Table
node1: $\sigma(k_1)=\Gamma_{MLN}$, w_1=8, ability=a_1, w_1'=8, ...
node2: w_2=13, ability=a_2, w_2'=7, ...
node3: w_3=7, ability=a_3, w_3'=9, ...
node4: w_4=0, ability=a_4, w_4'=4, ...

Node Information Table
node2: $\sigma(k_2)=\Gamma_{OLN}$, w_2=13, ability=a_2, w_2'=7, ...
node1: load w_1=8, ability=a_1, w_1'=8, ...
node3: load w_3=7, ability=a_3, w_3'=9, ...
node4: load w_4=0, ability=a_4, w_4'=4, ...

Node Information Table
node3: $\sigma(k_3)=\Gamma_{LLN}$, w_3=7, ability=a_3, w_3'=9, ...
node2: w_2=13, ability=a_2, w_2'=7, ...
node1: w_1=8, ability=a_1, w_1'=8, ...
node4: w_4=0, ability=a_4, w_4'=4, ...

Node Information Table
node4: $\sigma(k_4)=\Gamma_{ELN}$, w_4=0, ability=a_4, w_4'=4, ...
node2: w_2=13,ability=a_2, w_2'=7, ...
node1: w_1=8,ability=a_1, w_1'=8, ...
node3: w_3=7,ability=a_3, w_3'=9, ...

Figure 1. Time delay impulsive switching load balancing model for distributed system with four nodes.

interact with each other, which eventually drives the system into a balanced state.

Description of models

A distributed system that contains 4 nodes is shown in Figure 1. It shows a typical process of impulsive and switching load balancing with time delay. Each node owns a load queue to be dealt with. Each node also maintains a node information table for recording the state information of its resources. The table includes the load queue length of each node, which can read from the Message Queue of operating system. When starting the system, each node broadcasts its own state information of resources to other nodes. Meanwhile, every node calculates its ideal load w_i' and saves the current state of the system. If the system is overloaded, then we make use of the impulsive and switching load balancing model with time delay after processing the ideal load of all nodes. Else we use the model directly.

Each node measures its own load before choosing a suitable sub-system. It switches to a sub-system by an impulsive signal, the process of which obeys the load migration rule. In Figure 1, node 2 is an overload node. The over load is migrated to light-load node 3 and empty-load node 4 by load migration proportion step by step. Finally, every node in the system reaches a state with ideal load w_i'. Then the system reaches a balancing state.

The system consists of n nodes $P_0, P_1, \ldots, P_{n-1}$. The processing ability a_i $(0 \le i \le n-1)$ of node P_i can be calculated in eq. (1).

$$a_i = \begin{cases} Nc_i * (F_i + \varepsilon * M_i) & 0 \le U_i(t) \le \alpha \\ Nc_i * (F_i + \varepsilon * M_i) * \dfrac{1-(\beta-\alpha)^2}{\ln(2-\beta)} * & \\ \ln(2-U_i(t)) & \beta < U_i(t) \le 1 \\ Nc_i * (F_i + \varepsilon * M_i) * (1-(U_i(t)-\alpha)^2) & \alpha < U_i(t) \le \beta, \end{cases} \tag{1}$$

a_i is an aggregative indicator of the number of cores Nc_i, frequency F_i, memory M_i and current utilization $U_i(t)$ of node P_i. The transition points α and β are 0.2 and 0.55 in our nodes for experiment. The coefficient ε is 0.2.

The load of node P_i is recorded as w_i $(0 \le I \le n-1)$. The load which can be processed in the k_ith time interval $[t_{k_i}, t_{k_i+1})$ is recorded as ideal load w_i' and

$$w_i' = (t_{k_i+1} - t_{k_i}) * \frac{\int_{t_{k_i}}^{t_{k_i+1}} a_i(t)\mathrm{d}t}{t_{k_i+1} - t_{k_i}}. \tag{2}$$

$[t_{k_i}, t_{k_i+1})$ is the lingering time of a sub-system.

Definition 1. (Judging criterion for node state). If the load of node P_i is subjected to $w_i < w^{\inf}$, $w^{\inf} < w_i < w^{\sup}$,

$w_i > w^{\sup}$ or $w_i = 0$, then P_i belongs to a light-load node, a moderate-load node, an over-load node or an empty-load node respectively. The load upper bound $w^{\sup}$ and lower bound $w^{\inf}$ are $w^{\sup} = w_i' *(1 + \xi)$ and $w^{\inf} = w_i'*(1 - \xi)$ respectively. $\xi = 0.05$.

When starting the system, node P_i broadcasts its own state and load information to other nodes. Node P_i collects information from other nodes. Node P_i calculates and maintains the current state of the system. The state of system, *state* can be calculated in eq. (3). If *state* = 1, we deal with the ideal load of P_i with $w_i' = \omega * w_i'$ and convert *state* to 0. Then we carry out our model in eq. (4). Else, when *state* = 0, we carry out our model in eq. (4) directly. In a system with 3 nodes, if all the nodes are over-load, calculate the over-load coefficient ω ($\omega > 1$), which can judge that the system is over-load. When the system is over-load, we deal with the ideal load of P_i with $w_i' = \omega * w_i'$ ($i = 1, 2, 3$). Then, $w_1' + w_2' + w_3'$ can match with the current total load of the system. If P_1 is a light-load node, then there is at least one over-load node in P_2 and P_3. If P_1 is an over-load node, then there is at least one light-load node in P_2 and P_3. If P_1 is a moderate-load node, then both P_2 and P_3 maybe moderate-load nodes. Or there is a light-load node and an over-load node in P_2 and P_3. Then make load balancing, this can assure the executing time of the three nodes be similar or same.

$$state = \begin{cases} 1 & \omega > t \\ 0 & \omega \le t \end{cases}, \quad \omega = \frac{\sum_1^n w_i}{\sum_1^n a_i}. \tag{3}$$

In eq. (4), $w(t_i)$ is the state vector of system and $w(t_i) \in R^n$. $\sigma(k_i)$ is the identifier of the sub-system corresponding to the state of P_i at different times. The active sub-system is expressed as $\sigma(k_i)$ in $[t_{k_i}, t_{k_i+1})$. The switching time sequence is subjected to $0 \le t_0 < t_1 < \dots t_{k_i} < t_{k_i+1} < \dots$. The time delay of system τ_i subjects to $\tau_i > 0$. $\Delta w(t_{k_i+1})$ stands for state jumping. The initial condition subjects to $w(\theta_i) = w_i(\theta_i)$.

$$\begin{cases} \dot{w}(t_i) = f_{\sigma(k_i)}(t_i, w(t_i), & t_i \in [t_{k_i}, t_{k_i+1}), \sigma(k_i) \\ \quad w(t_i - \tau_i)) & \in \{\Gamma_{\mathrm{OLN}}, \Gamma_{\mathrm{MLN}}, \Gamma_{\mathrm{LLN}}, \Gamma_{\mathrm{ELN}}\} \\ \Delta w(t_{k_i+1}) = w(t_{k_i+1}) - w(t_{k_i+1}^-) & k_i = 0,1,2,\dots \\ w(\theta_i) = w_i(\theta_i) & \theta_i \in [t_0 - \tau_i, t_0] \end{cases} \tag{4}$$

Each node chooses a suitable sub-system in eqs (6)–(9). An over-load node migrates its over load to adjacent nodes in light of eq. (6). A light-load or empty-load node transforms its state in light of eq. (7) or (9) if it has received a migration load. However, if a light-load node has not received any migration load, it will transform its state in light of eq. (8). An empty-load node which has reveived no load will not make a state transition. A moderate-load node neither migrates load to other nodes nor receives load from others. It makes a state transition according to eq. (8). All load migration processes carried on must obey the load migration rule.

Definition 2. (load migration rule) for two arbitrary nodes P_i and P_j which are in the state of $\sigma(k_i)$ and $\sigma(k_j)$, load can be migrated by the method of $\{P_i \to P_j | \sigma(k_i) = \Gamma_{\mathrm{OLN}}, \sigma(k_j) \in \{\Gamma_{\mathrm{LLN}}, \Gamma_{\mathrm{ELN}}\}\}$.

Because P_i owns its special processing ability a_i, so the load size in migration varies each time. If over load is divided into a fixed size without considerating a_i, then the node state would transform frequently. Because a node with a weak a_i owns a a smaller ideal load w_i' than a node with a strong a_i, a node with a weak a_i should receive fewer loads than a node with a strong a_i. In other words, in order to guarantee the stability of the system, the load migrated from a node with strong a_i should not be excessive. Therefore, we use eq. (5) for an adaptive load migration proportion.

$$p_{ij} = \frac{1}{n} * \left(1 - \frac{a_i(t)}{\sum_1^n a_i(t)}\right). \tag{5}$$

The state of a node is judged by the Definition 1. If a node belongs to an over-load node, then the node should only migrate its overload to others, but not receive load from others, except for processing its own tasks and producing new tasks. If a node belongs to a light-load node, then the node should only receive load from others but not migrate its load to others, except for processing its own tasks and producing new tasks. If a node belongs to a moderate-load node, then the node should neither receive load from others nor migrate its load to others, except for processing its own tasks and producing new tasks. If a node belongs to an empty-load node, then the node should only receive load from others but not migrate its load to others, because it has no tasks of its own and would not produce new tasks. So we construct sub-system models in eqs (6)–(9) separately.

State variable w_i stands for the length of load queue of node P_i. Input variable u_i stands for the over load that is migrated from node P_i to other nodes in unit time. Output variable y_i stands for the over load of node P_i. State variable w_i' stands for the length of ideal load queue of node P_i.

$$\Gamma_{\mathrm{OLN}}: \begin{cases} \dfrac{\mathrm{d}w_i(t)}{\mathrm{d}t} = \lambda_i(t) - \mu_i(t) - \sum_{i \ne j} p_{ij} u_i(t - \eta_{ij}) \\ y_i(t) = w_i(t) - w_i'(t) \\ u_i(t) = k_i y_i(t) \end{cases} \tag{6}$$

$$\Gamma_{\text{LLN}}:\begin{cases}\dfrac{\mathrm{d}w_i(t)}{\mathrm{d}t}=\lambda_i(t)-\mu_i(t)+\sum\limits_{j\neq i}p_{ji}u_j(t-\eta_{ji})\\ y_i(t)=w'(t)-w_i(t)\\ u_i(t)=k_iy_i(t),\end{cases}\tag{7}$$

$$\Gamma_{\text{MLN}}:\begin{cases}\dfrac{\mathrm{d}w_i(t)}{\mathrm{d}t}=\lambda_i(t)-\mu_i(t)\\ y_i(t)=w_i(t)-w_i'(t)\\ u_i(t)=0,\end{cases}\tag{8}$$

$$\Gamma_{\text{ELN}}:\begin{cases}\dfrac{\mathrm{d}w_i(t)}{\mathrm{d}t}=\sum\limits_{j\neq i}p_{ji}u_j(t-\eta_{ji})\\ y_i(t)=w_i'(t)-w_i(t)\\ u_i(t)=k_iy_i(t).\end{cases}\tag{9}$$

λ_i and μ_i stand for the task producing rate and task processing rate of node P_i respectively. We calculate them as follows

$$\lambda_i=r_i*w_i(t)\tag{10}$$

$$\mu_i=\frac{a_i(t)}{\int_{t_{k_i}}^{t_{k_i+1}}a_i(t)\mathrm{d}t}*w_i(t).\tag{11}$$

where $r_i\in R^+$. Its value is fixed by the intrinsic attribute of a load. k_i stands for the closed loop gain in unit time p_{ij} is the load migration proportion calculated by eq. (5). η_{ij} stands for the transmission delay from node P_i to P_j.

Proof of stability

An unstable system cannot complete control tasks as expected. Hence in a control system, stability is an important evaluation index for system performance. Daniel Liberzon concluded the stability problems of switching systems. One problem is to find out the condition for stability of a switching system with arbitrary switching signal on the basis of stability of each sub-system. That is the condition for system stability with arbitrary switching signal. If there exists the same Lyapunov function, then the switching system is always stable with arbitrary switching signal. Therefore, the problem is usually converted to finding or constructing the same Lyapunov function for each sub-system.

Lemma 1. (Stability criterion for switching system). If each sub-system in a switching system: (1) shares the same Lyapunov function with others; (2) is stable, then the switching system is always stable with arbitrary switching signal.

Lemma 2. (Asymptotic stability criterion for Lyapunov function). Assume that the state equation of the system is $\dot{x}=f(x)$, the balancing state is $x_e=0$ and $f(x_e)=0$. If there exists a scalar function $V(x)$ where (1) $V(x)$ has a continuous first-order partial derivative for all x; (2) when $x=0$, $V(x)=0$. $x\neq 0$, $V(x)>0$, if $\dot{V}(x)<0$, then the sub-system in eq. (6) is asymptotic stable at the balancing point.

In the stability description based on state space method, the analysis for system stability occurs in a time domain directly. The sub-system in eq. (6) which mostly adopts frequency domain analysis method aims at the output stability. On the basis of the sub-system in eq. (6), we propose a time domain analysis model Γ_{OLN} in eqs (12–14) for state stability

$$\frac{\mathrm{d}w_i(t)}{\mathrm{d}t}=q_i(t)-\dot{w}_i'(t),\tag{12}$$

$$q_i(t)=\lambda_i(t)-\mu_i(t)-\sum_{i\neq j}p_{ij}u_i(t-\eta_{ij}),\tag{13}$$

$$u_i(t)=k_iw_i(t).\tag{14}$$

In eqs (12)–(14), state variable w_i stands for the overload on node P_i. q_i stands for the changing rate of load queue; λ_i stands for the producing rate of node P_i. μ_i stands for the processing rate of node P_i.

If our model in eqs (12)–(14) is asymptotic stable and the overload equals to zero in the sub-system Γ_{OLN}, then it reaches the state of load balancing.

In order to analyse the stability of system, we transform our model in eqs (12)–(14) into a standard time domain analysis model by the following steps.

We reorganize the equation according to Newton–Leibniz with eq. (13), (14) and (2) being substituted in eq. (12), hence,

$$\dot{w}_i(t)=\lambda_i(t)-\mu_i(t)-k_i\sum_{i\neq j}p_{ij}w_i(t-\eta_{ij})-a_i(t_{k_i+1})+a_i(t_{k_i}).\tag{15}$$

Substituting eqs (5), (10) and (11) in eq. (15) and reorganizing eq. (15),

$$\dot{w}_i(t)=\left(r_i-\frac{a_i(t)}{\int_{t_{k_i}}^{t_{k_i+1}}a_i(t)\mathrm{d}t}\right)*w_i(t)-k_i*\frac{1}{n}$$

$$*\left(1-\frac{a_i(t)}{\sum_1^n a_i(t)}\right)*\sum_{i\neq j}w_i(t-\eta_{ij})-a_i(t_{k_i+1})+a_i(t_{k_i}).\tag{16}$$

We assume $\eta_{ij} = \eta$ and $\eta > 0$ and then deduce the standard equation in eq. (17) for the model in eqs (12)–(14).

$$\dot{\boldsymbol{W}}(t) = \boldsymbol{A}_0\boldsymbol{W}(t) + \boldsymbol{A}_1\boldsymbol{W}(t-\eta) + \boldsymbol{B}. \tag{17}$$

In eq. (17),

$$\boldsymbol{A}_0 = \boldsymbol{L}, \quad \boldsymbol{A}_1 = -\boldsymbol{KS}, \quad \boldsymbol{W}(t) = \begin{bmatrix} w_1(t) \\ \vdots \\ \vdots \\ w_n(t) \end{bmatrix},$$

$$\boldsymbol{L} = \begin{bmatrix} l_1 & & \\ & \ddots & \\ & & l_n \end{bmatrix}, \quad \boldsymbol{K} = \begin{bmatrix} k_1 & & \\ & \ddots & \\ & & k_n \end{bmatrix},$$

$$\boldsymbol{S} = \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_n \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} b_1 & & \\ & \ddots & \\ & & b_n \end{bmatrix},$$

$$l_i = r_i - \frac{a_i(t)}{\int_{t_{k_i}}^{t_{k_i+1}} a_i(t)\mathrm{d}t}, \quad s_i = \frac{1}{n} * \left(1 - \frac{a_i(t)}{\sum_1^n a_i(t)}\right),$$

$$b_i = -a_i(t_{k_i+1}) + a_i(t_{k_i}).$$

Then we propose the condition for asymptotic stability of the sub-system for the model in eq. (17) as follows.

Theorem 1. For the model in eq. (17) with the assumption of $\eta_{ij} = \eta$ and $\eta > 0$, if there exist symmetrical positive definite matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{Z}$, symmetrical matrix $\boldsymbol{X}$ and matrix $\boldsymbol{Y}$, which make

$$\begin{bmatrix} (\boldsymbol{QA}_0 + \boldsymbol{Y})^{\mathrm{T}} + (\boldsymbol{QA}_0 + \boldsymbol{Y}) + \eta\boldsymbol{X} + \boldsymbol{R} & -\boldsymbol{Y} + \boldsymbol{QA}_1 & \eta\boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{Z} \\ * & -\boldsymbol{R} & \eta\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{Z} \\ * & * & -\eta\boldsymbol{Z} \end{bmatrix} < 0,$$

$$\begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ * & \boldsymbol{Z} \end{bmatrix} \geq 0,$$

Then the sub-system in eq. (17) is asymptotic stable.

Proof. In the process of stability analysis, the key point is find a Lyapunov function $V(w)$ which is satisfied by the criterion. In the paper, we choose Lyapunov–Krasovskii functions for various delays in eq. (18).

$$\begin{cases} V(\boldsymbol{W}_t) = V_1(\boldsymbol{W}_t) + V_2(\boldsymbol{W}_t) + V_3(\boldsymbol{W}_t) \\ V_1(\boldsymbol{W}_t) = \boldsymbol{W}(t)^T\boldsymbol{QW}(t) \\ V_2(\boldsymbol{W}_t) = \int_{t-\eta}^{t} \boldsymbol{W}(\theta)^T\boldsymbol{RW}(\theta)\mathrm{d}\theta \\ V_3(\boldsymbol{W}_t) = \int_{t-\eta}^{t}\int_{\alpha}^{t} \dot{\boldsymbol{W}}(\theta)^T\boldsymbol{Z}\dot{\boldsymbol{W}}(\theta)\,\mathrm{d}\theta\mathrm{d}\alpha. \end{cases} \tag{18}$$

$\boldsymbol{W}_t = \boldsymbol{W}(t + \sigma)$, $\sigma \in [\tau, 0]$. $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{Z}$, are symmetrical positive determined matrices. It is obviously that $V(\boldsymbol{W}_t) > 0$.

Our model in eq. (17) equals to eq. (19) according to Newton–Leibniz.

$$\dot{W}(t) = (\boldsymbol{A}_0 + \boldsymbol{A}_1)\boldsymbol{W}(t) - \boldsymbol{A}_1\int_{t-\eta}^{t} \dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha + \boldsymbol{B}. \tag{19}$$

We calculate the derivative of $V_1(\boldsymbol{W}_t)$ for time t along an arbitrary track of sub-system in eq. (19) and get

$$\dot{V}_1(\boldsymbol{W}_t) = \boldsymbol{W}^T(t)\boldsymbol{Q}\dot{W}(t) + \dot{\boldsymbol{W}}^T(t)\boldsymbol{QW}(t)$$

$$= \boldsymbol{W}^T(t)\boldsymbol{Q}\left((\boldsymbol{A}_0 + \boldsymbol{A}_1)\boldsymbol{W}(t) - \boldsymbol{A}_1\int_{t-\eta}^{t} \dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha\right)$$

$$+\left((\boldsymbol{A}_0 + \boldsymbol{A}_1)\boldsymbol{W}^T - \boldsymbol{A}_1\int_{t-\eta}^{t} \dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha)\right)\boldsymbol{QW}(t). \tag{20}$$

Then it is reorganized into

$$\dot{V}_1(\boldsymbol{W}_t) = 2\boldsymbol{W}^T(t)\boldsymbol{Q}(\boldsymbol{A}_0 + \boldsymbol{A}_1)\boldsymbol{W}(t)$$

$$-\int_{t-\eta}^{t} 2\boldsymbol{W}^T(t)\boldsymbol{QA}_1\dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha. \tag{21}$$

Lemma 3. (Moon inequality). For vectors $\boldsymbol{g}$ and $\boldsymbol{h}$ with arbitrary dimension, matrix $\boldsymbol{Y}$, symmetrical matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$, the following inequality relations are established:

$$-2\boldsymbol{g}^T\boldsymbol{Nh} \leq \begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{h} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} - \boldsymbol{N} \\ * & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{h} \end{bmatrix}, \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ * & \boldsymbol{Z} \end{bmatrix} \geq 0,$$

where * stands for the transpose of symmetric block in the matrix.

Magnifying $-2\boldsymbol{W}^T(t)\boldsymbol{QA}_1\dot{\boldsymbol{W}}(\alpha)$ in eq. (21) by Lemma 3,

$$-2\boldsymbol{W}^T(t)\boldsymbol{QA}_1\dot{\boldsymbol{W}}(\alpha) \leq \begin{bmatrix} \boldsymbol{W}(t) \\ \dot{\boldsymbol{W}}(\alpha) \end{bmatrix}^T$$

$$\begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y}-\boldsymbol{Q}A_1 \\ * & \boldsymbol{Z} \end{bmatrix}\begin{bmatrix} \boldsymbol{W}(t) \\ \dot{\boldsymbol{W}}(\alpha) \end{bmatrix}, \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ * & \boldsymbol{Z} \end{bmatrix} \geq 0.$$

Rearranging the above, we get,

$$-2\boldsymbol{W}^T(t)\boldsymbol{Q}A_1\dot{\boldsymbol{W}}(\alpha) \leq 2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\dot{\boldsymbol{W}}(t)$$

$$\boldsymbol{W}^T(t)\boldsymbol{X}\boldsymbol{W}(t)+\dot{\boldsymbol{W}}^T(t)\boldsymbol{Z}\dot{\boldsymbol{W}}(t), \tag{22}$$

$$\begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ * & \boldsymbol{Z} \end{bmatrix} \geq 0. \tag{23}$$

Substituting eqs (22) in (21).

$$\dot{V}1(\boldsymbol{W}_t) \leq 2\boldsymbol{W}^T(t)\boldsymbol{Q}(A_0+A_1)\boldsymbol{W}(t)$$

$$+\boldsymbol{W}^T(t)(\eta X)\boldsymbol{W}(t)+2\boldsymbol{W}^T(t)(Y-\boldsymbol{Q}A_1)$$

$$\int_{t-\eta}^{t}\dot{\boldsymbol{W}}(\alpha)\,\mathrm{d}\alpha+\int_{t-\eta}^{t}\dot{\boldsymbol{W}}^T(\alpha)Z\dot{\boldsymbol{W}}(\alpha)\,\mathrm{d}\alpha. \tag{24}$$

Using Newton–Leibniz on the above one get,

$$\dot{V}1(\boldsymbol{W}_t) \leq 2\boldsymbol{W}^T(t)\boldsymbol{Q}(A_0+A_1)\boldsymbol{W}(t)$$

$$+\boldsymbol{W}^T(t)(\eta\boldsymbol{X})\boldsymbol{W}(t)+2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\boldsymbol{W}(t)$$

$$-2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\boldsymbol{W}(t-\eta)+\int_{t-\eta}^{t}\dot{\boldsymbol{W}}^T(\alpha)\boldsymbol{Z}\dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha. \tag{25}$$

Reorganizing the above, we get,

$$\dot{V}1(\boldsymbol{W}_t) \leq 2\boldsymbol{W}^T(t)(\boldsymbol{Q}A_0+\boldsymbol{Y})\boldsymbol{W}(t)$$

$$+\boldsymbol{W}^T(t)(\eta\boldsymbol{X})\boldsymbol{W}(t)-2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\boldsymbol{W}(t-\eta)$$

$$+\int_{t-\eta}^{t}\dot{\boldsymbol{W}}^T(\alpha)\boldsymbol{Z}\dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha. \tag{26}$$

We calculate the derivative of $V_2(\boldsymbol{W}_t)$ for time t,

$$\dot{V}2(\boldsymbol{W}_t)=\boldsymbol{W}^T(t)\boldsymbol{R}\boldsymbol{W}(t)-\boldsymbol{W}^T(t-\eta)\boldsymbol{R}\boldsymbol{W}(t-\eta). \tag{27}$$

We calculate the derivative of $V_3(\boldsymbol{W}_t)$ for time t,

$$\dot{V}_2(\boldsymbol{W}_t)=\dot{\boldsymbol{W}}^T(t)(\eta\boldsymbol{Z})\dot{\boldsymbol{W}}(t)-\int_{t-\eta}^{t}\dot{\boldsymbol{W}}^T(\alpha)\boldsymbol{Z}\dot{\boldsymbol{W}}(\alpha)\mathrm{d}\alpha. \tag{28}$$

We then synthesize eqs (26)–(28) into

$$\dot{V}(\boldsymbol{W}_t)=\dot{V}1(\boldsymbol{W}_t)+\dot{V}2(\boldsymbol{W}_t)+\dot{V}3(\boldsymbol{W}_t)$$

$$\leq 2\boldsymbol{W}^T(t)(\boldsymbol{Q}A_0+\boldsymbol{Y})\boldsymbol{W}(t)+\boldsymbol{W}^T(t)(\boldsymbol{R}+\eta\boldsymbol{X})\boldsymbol{W}(t)$$

$$-2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\boldsymbol{W}(t-\eta)$$

$$-\boldsymbol{W}^T(t-\eta)\boldsymbol{R}\boldsymbol{W}(t-\eta)+\dot{\boldsymbol{W}}^T(t)(\eta\boldsymbol{Z})\dot{\boldsymbol{W}}(t). \tag{29}$$

We then substitute eq. (17) in eq. (29). Because the changes of $U_i(t)$ on node P_i could be ignored during the lingering time of a sub-system, the processing ability a_i remains unchanged. Therefore, we assume that $a_i(t_{k_i+1}) = a_i(t_{k_i})$. So eq. (29) becomes to

$$\dot{V}(\boldsymbol{W}_t)=\dot{V}1(\boldsymbol{W}_t)+\dot{V}2(\boldsymbol{W}_t)+\dot{V}3(\boldsymbol{W}_t)$$

$$\leq 2\boldsymbol{W}^T(t)(\boldsymbol{Q}A_0+\boldsymbol{Y})\boldsymbol{W}(t)+\boldsymbol{W}^T(t)(\boldsymbol{R}+\eta\boldsymbol{X})\boldsymbol{W}(t)$$

$$-2\boldsymbol{W}^T(t)(\boldsymbol{Y}-\boldsymbol{Q}A_1)\boldsymbol{W}(t-\eta)-\boldsymbol{W}^T(t-\eta)\boldsymbol{R}\boldsymbol{W}(t-\eta)$$

$$+(A_0\boldsymbol{W}(t)-A_1\boldsymbol{W}(t-\eta))^T(\eta\boldsymbol{Z})(A_0\boldsymbol{W}(t)$$

$$-A_1\boldsymbol{W}(t-\eta))-\boldsymbol{W}^T(t-\eta)\boldsymbol{R}\boldsymbol{W}(t-\eta)$$

$$+\dot{\boldsymbol{W}}^T(t)(\eta\boldsymbol{Z})\dot{\boldsymbol{W}}(t). \tag{30}$$

After organization, we get,

$$\dot{V}(\boldsymbol{W}_t) \leq \boldsymbol{\xi}(t)^T\boldsymbol{\Omega}\boldsymbol{\xi}(t), \tag{31}$$

$$\begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ * & \boldsymbol{Z} \end{bmatrix} \geq 0. \tag{32}$$

In eqs (31)–(32),

$$\boldsymbol{\Omega}=\begin{bmatrix} \Theta & -\boldsymbol{Y}+\boldsymbol{Q}A_1+A_0^T(\eta\boldsymbol{Z})A_1 \\ * & -\boldsymbol{R}+A_1^T(\eta\boldsymbol{Z})A_1 \end{bmatrix},$$

$$\xi(t)^T=[\boldsymbol{W}^{\mathrm{T}}(t)\;\;\boldsymbol{W}^{\mathrm{T}}(t-\eta)]$$

and

$$\Theta = (\boldsymbol{Q}A_0+\boldsymbol{Y})^T+(\boldsymbol{Q}A_0+\boldsymbol{Y})+\eta\boldsymbol{X}+\boldsymbol{R}+A_0^T(\eta\boldsymbol{Z})A_0.$$

For there exists

$$A_0^T \eta Z A_0 = A_0^T \eta Z (\eta Z)^{-1} \eta Z A_0,$$

$$A_0^T \eta Z A_1 = A_0^T \eta Z (\eta Z)^{-1} \eta Z A_1,$$

$$A_1^T \eta Z A_1 = A_1^T \eta Z (\eta Z)^{-1} \eta Z A_1.$$

Lemma 4. (Schur complementary property). Assume that for a matrix like

$$\boldsymbol{\Gamma}(x) = \begin{bmatrix} \boldsymbol{\Gamma}(x)_{11} & \boldsymbol{\Gamma}(x)_{12} \\ \boldsymbol{\Gamma}(x)_{21} & \boldsymbol{\Gamma}(x)_{22} \end{bmatrix},$$

$\Gamma(x) > 0$ equals to $\Gamma(x)_{11} - \Gamma(x)_{12}\Gamma^{-1}(x)_{22}\Gamma(x)_{21} > 0$ and $\Gamma(x)_{22} > 0$.

We know that inequality $\boldsymbol{\Omega} < 0$ and the first inequality in Theorem 1 are equal by Lemma 4, so $\dot{V}(\boldsymbol{W}_t) < 0$. The sub-system in eq. (17) is asymptotic stable in light of the asymptotic stability criterion for Lyapunov function in Lemma 1. Therefore, the theorem 1 is proved after synthesizing eq. (23).

We choose the same Lyapunov–Krasovskii for sub-systems in eqs (7)–(9) with that in eq. (6). We prove that sub-systems in eqs (7)–(9) are all asymptotic stable by the same method.

In conclusion, we can prove that our impulsive and switching load balancing model with time delay in eq. (4) is stable according to Lemma 1.

Simulation example and experiment result

We take a system with 4 nodes for example. The parameter matrices $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$ are calculated by the system state equation in eq. (17).

$$A_0 = \begin{bmatrix} -1.5662 & 0 & 0 & 0 \\ 0 & -1.6261 & 0 & 0 \\ 0 & 0 & -1.5066 & 0 \\ 0 & 0 & 0 & -1.6465 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -0.1469 & 0 & 0 & 0 \\ 0 & -0.1493 & 0 & 0 \\ 0 & 0 & -0.0903 & 0 \\ 0 & 0 & 0 & -0.0924 \end{bmatrix}.$$

We then solve the linear matrix inequality given by the Theorem 1 by feasp function in LMI toolbox from Matlab, and get the feasible solution

$$\boldsymbol{Q} = \begin{bmatrix} 583.9947 & 0 & 0 & 0 \\ 0 & 576.4316 & 0 & 0 \\ 0 & 0 & 596.9907 & 0 \\ 0 & 0 & 0 & 577.2324 \end{bmatrix},$$

$$\boldsymbol{R} = \begin{bmatrix} 554.1724 & 0 & 0 & 0 \\ 0 & 554.4720 & 0 & 0 \\ 0 & 0 & 552.0727 & 0 \\ 0 & 0 & 0 & 552.3127 \end{bmatrix},$$

$$\boldsymbol{Z} = \begin{bmatrix} 552.4911 & 0 & 0 & 0 \\ 0 & 552.4607 & 0 & 0 \\ 0 & 0 & 551.5976 & 0 \\ 0 & 0 & 0 & 551.5521 \end{bmatrix},$$

$$\boldsymbol{X} = \begin{bmatrix} 550.3426 & 0 & 0 & 0 \\ 0 & 550.3997 & 0 & 0 \\ 0 & 0 & 550.6907 & 0 \\ 0 & 0 & 0 & 550.7529 \end{bmatrix},$$

$$\boldsymbol{Y} = \begin{bmatrix} -18.2542 & 0 & 0 & 0 \\ 0 & -16.9170 & 0 & 0 \\ 0 & 0 & -12.2932 & 0 \\ 0 & 0 & 0 & -10.1557 \end{bmatrix}.$$

Therefore, we know that there exist symmetrical positive definite matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{Z}$, symmetrical matrix $\boldsymbol{X}$ and matrix $\boldsymbol{Y}$ which guarantee asymptotic stability of the system. Meanwhile, the practical feasibility of Theorem 1 in the paper is illustrated.

With the above design, we experiment in a multi-thread environment for the proposed model under specific initial conditions. The software is win64 multi-thread on VC++, the hardware is Intel Core i7 with 2.30 GHz frequency and 4 GB memory. The conservative synchronization strategy is realized by *mutex*. We use *sleep* to implement the transmission time delay η on the same time scale. The system scale is controlled by setting the global variable. In order to analyse the node state and the change of queue length in the system, we conveniently, take node 6 as an example to explain. For heterogeneous resource, the specific result can be described by Figure 2. The specific initial conditions are: $n = 6$, $\eta_{ij} = \eta = 120$ μs. Initial queue length are: $w_1 = 420$, $w_2 = 980$, $w_3 = 1300$, $w_4 = 0$, $w_5 = 650$, $w_6 = 1600$. Load balancing gain $K = 0.7$.

From Figure 2, we see that the state and queue length of each node change on the time $T_1 = 187.288$ ms, $T_2 = 227.75$ ms, $T_3 = 299.096$ ms, $T_4 = 329.738$ ms, $T_5 = 355.478$ ms, $T_6 = 370.215$ ms. Each node judges whether load balancing is required periodically. On the initial time T_0, the main thread activates other threads. The statistical

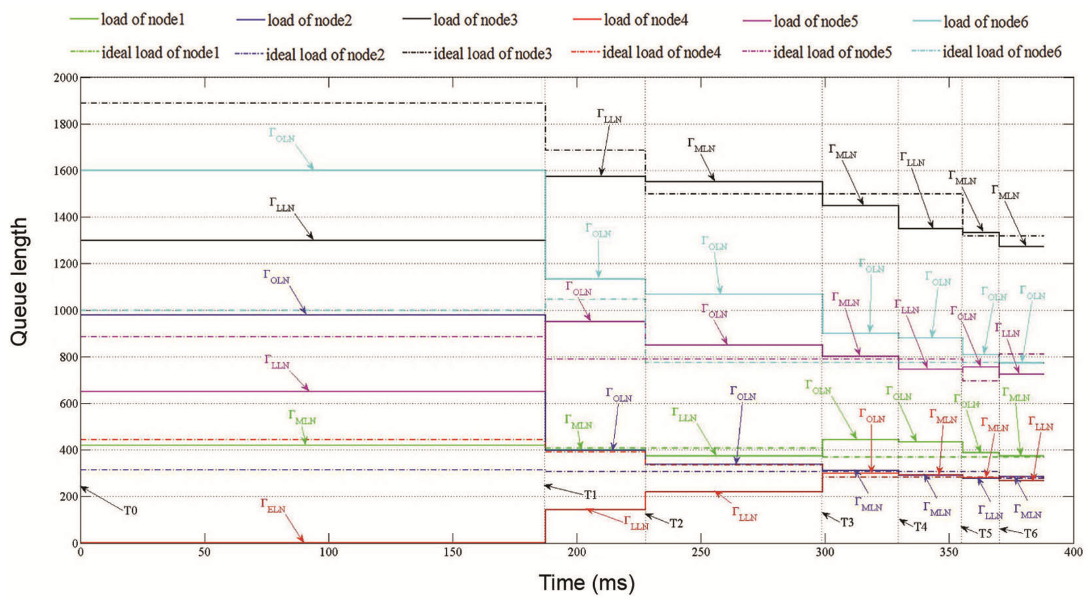


Figure 2. State and queue length change of nodes in process of load balancing.

thread creates statistics for instant load information and state of each node, judges the system state and then deals with the ideal load. Each node thread calculates its own processing ability a_i and ideal load w_i'. The nodes are on the state of moderate-load, over-load, light-load, empty-load, light-load and over-load respectively. Node 2 migrates one unit workload $u_2(T_1)$ to node 3, node 4 and node 5. Node 6 migrates one unit workload $u_6(T_1)$ to node 3, node 4 and node 5. After that on T_1, nodes 4 and 5 change to the states of light-load and over-load. The state of node 3 has not changed for it owns a strong a_3. Node 2 migrates one unit load to nodes 3 and 4 respectively. Node 5 migrates one unit load to nodes 3 and 4 respectively. Node 6 migrates one unit load to nodes 3 and 4 respectively. Because the migration unit of node 2 reduces, w_2 drops slightly on T_2 than on T_1. Over-load nodes continue migrating their overloads to other light-load and moderate-load nodes. At T_3, all nodes are in the state of over-load and moderate-load. So there is no migration in the period and each node deals with its own load. At T_4, nodes 3 and 5 change into the state of light-load. So, nodes 1 and 6 migrate one unit load to them respectively. On T_5, node 5 changes to over-load because it owns weak processing ability. After a new period of migration, there is no over-load on node T_6 and hence the system is balanced.

In Figure 3, equal-division load balancing algorithm and two-division network load balancing algorithm are compared to our algorithm in different scale. In equal-division balancing algorithm, firstly, the workload of each processor is divided by the speed of processor in network. Then, the little part of workload divided in the previous step will be migrated to the corresponding processor, which can balance the load of each processor. In two-division network load balancing algorithm, the system is divided into two sub-networks with the same number of nodes. Then the load is migrated between the sub-networks based on their processing ability. The above process repeats until there is only one node in each sub-network. The system reaches balance after migration.

In the method we proposed, for all over-load nodes, on every migration time, each over-load node divides over workload into n parts according to proportion p_{ij} in eq. (5) (n stands for the system scale). Each over-load node migrates the over load to empty-load nodes and light-nodes on the current time respectively. This assures that the over workload will not be migrated repeatedly. The state of node, processing ability, the length of current load, all of these information will be updated before each migration. This assures load migration based on more accurate information. If the number of empty-load and light-load nodes takes $n/3$, the system needs at least 3 times migration to reach balance. It is known that the balance time has something to do with the proportion which empty-load and light-load nodes take.

For a linear array structure, if the network belongs to a homogeneous net, the average amount of load moved at each step is equal to $\frac{1}{2}\sum_{i=0}^{n-1} w_i$ by using equal-division balancing algorithm. The amount is just half of the initial load. All nodes perform load migration to the parallel neighbour-node, which needs $n - 1$ steps. In two-division network load balancing algorithm, the amount of load migration between a node pair is equal. However, during the process of load migration, the algorithm only migrates the 'load deviation' between nodes, but does not exchange their load. So the amount of load migration is normally small. The number of steps of balancing $n - 1$. If the network belongs to a heterogeneous net, the amount of load migration by the algorithms on each step deals

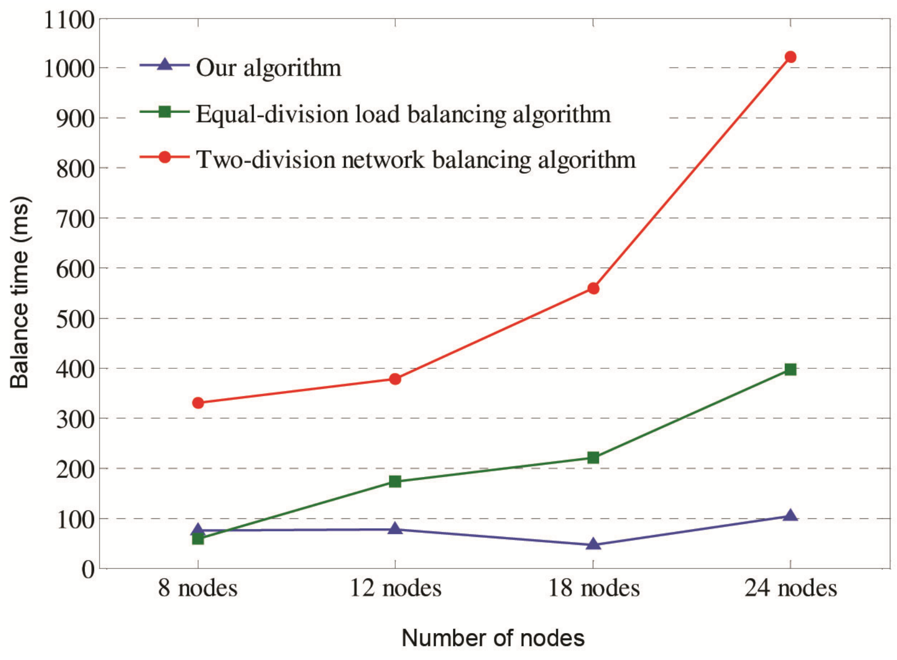


Figure 3. Comparison of three algorithms of load balancing in different scale.

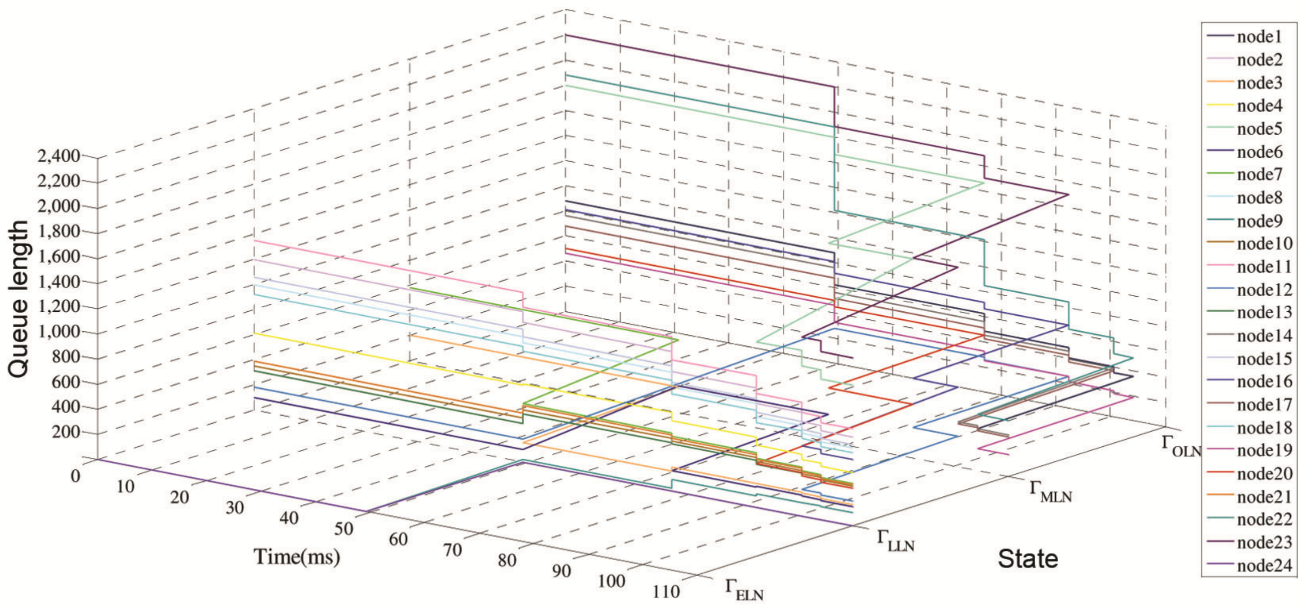


Figure 4. Process of load balancing on *Stream* platform.

with two aspects: the processing speed of each node and the distribution of initial load on each node. The number of steps of load balancing is $n - 1$ in both cases. Our algorithm only migrates the overload of each node; so the whole amount of load migration is small. The number of steps of load balancing is $n - 1$ as well.

For a two-dimensional mesh network structure, if the network belongs to a homogeneous net, the average amount of load moved at each step is equal to $\frac{1}{2}\sum_{i=0}^{n-1} w_i$ by using equal-division balancing algorithm. The number of steps of balancing are $2(\sqrt{n}-1)$. In two-division network load balancing algorithm, the amount of load migration between a node pair is equal. The algorithm only migrates the 'load deviation' between nodes. The number of steps of balancing is $2(\sqrt{n}-1)$. If the network belongs to a heterogeneous net, the amount of load migration by the algorithms on each step deals with two aspects: the processing speed of each node and the distribution of initial load on each node. The number of steps of load balancing is $2(\sqrt{n}-1)$ in both cases. Our algorithm only migrates the over load of each node; so the whole amount of load migration is small. The number of steps of load balancing is $2(\sqrt{n}-1)$.

From the analysis, we know that the performance of a two-dimensional mesh network structure is better than a linear array structure. Our algorithm has the least load migration, followed by the two-division network load balancing algorithm and then the equal-division balancing algorithm.

To verify the effectiveness and feasibility of our model, we take the process of load balancing on *Stream* platform as an example for the analysis. *Stream* is a shared, open, real-time and credible inter-value block chain. By using block chain and intelligent contract techniques, it provides a shared platform of value exchange to end-users which include enterprises and individuals. User, *Stream* port, calculation ability provider, calculation ability demander, market maker, seller and application developer make up the ecological chain of *Stream*. Developers create prosperous applied ecology based on *Stream*. All users share the achievement of the applied ecology. At the moment, there are 24 nodes running on the platform. The balancing process can be seen in Figure 4.

We can see from Figure 4, the changes in nodes state and queue length happen on 49.353 ms, 76.816 ms, 92.289 ms, 100.568 ms and 104.183 ms. When starting the system, there are 9 nodes in the state of over-load. After being adjusted 5 times by the load balancing of the impulsive and switching load balancing model with time delay, all nodes turn into moderate-load or light-load and the system is balanced rapidly.

Conclusions

In this paper, we have proposed an impulsive and switching load balancing model with time delay based on description of state space by control theory. We constructed the corresponding sub-system according to various states of each node in the system. During load migration, the processing ability of all nodes was taken into account, so that there were not too many nodes which change into a state of over-load. Then the efficiency of load balancing improved. We chose suitable Lyapunov functions for each sub-system based on description of the model. After a stability analysis of the sub-system by the method of Lyapunov–Krasovskii function, we achieved the corresponding analysis result. Besides, we verified the effectiveness of the method by LMI toolbox in Matlab. The feasibility of the model was verified by an experimental result which was produced by a shared platform based on block chain. In the future, we intend to further study the theoretical analysis based on our current work and speed up the convergence rate of the system to a state of stability.

1. Meng, Q. Y., Qiao, J. Z. and Lin, S. K., A delay-based dynamic load balancing method and its stability analysis and simulation, Proceedings of European Conference on Parallel Computing, Ischia, 2010, pp. 192–203.
2. Hsiao, H. C., Chung, H. Y. and Shen, H., Load rebalancing for distributed file systems in clouds. *IEEE Trans. Parall. Distr. Syst.*, 2013, **24**, 951–962.
3. Dhakal, S., Hayat, M. M., Pezoa, J. E., Yang, C. D. and Bader, D. A., Dynamic load balancing in distributed systems in the presence of delays: a regeneration-theory approach. *IEEE Trans. Parall. Distr. Syst.*, 2007, **18**, 485–497.
4. Tang, Z., Birdwell, J. D., Chiasson, J., Abdallah, C. T. and Hayat, M., Resource-constrained load balancing controller for a parallel database. *IEEE Trans. Control Syst. Technol.*, 2008, **16**, 834–840.
5. Tang, Z., White, J. and Chiasson, J., Closed-loop load balancing: comparison of a discrete event simulation with experiment. Proceedings of the 2005 American Control Conference, Portland, 2005, pp. 2721–2726.
6. Liu, X., Zhong, S. M. and Ding, X. Y., Robust exponential stability of impulsive switched systems with switching delays: a Razumikhin approach. *Commun. Nonlinear Sci. Numer. Simul.*, 2012, **17**, 1805–1812.
7. Shim, H. and Tanwani, A., Hybrid-type observer design based on a sufficient condition for observability in switched nonlinear systems. *Int. J. Robust Nonlin.*, 2014, **24**, 1064–1089.
8. Liu, X. W., Robust H^{∞} filtering for switched discrete-time systems with time-delays. Proceedings of the 26th Chinese Control Conference, Zhangjiajie, 2007, pp. 660–664.
9. Liu, J., Liu, X. Z. and Xie, W. C., Delay-dependent robust control for uncertain switched systems with time-delay. Nonlinear *Analy. Hybri. Syst.*, 2008, **2**, 81–95.
10. Egorov, A. V. and Mondi, S., Necessary conditions for the exponential stability of time-delay systems via the Lyapunov delay matrix. *Nonlinear Anal. Hybri.*, 2014, **24**, 1760–1771.

ACKNOWLEDGEMENT. This work was supported by the National Natural Science Foundation of China (NSFC) No. 61272177.

Received 7 March 2017; revised accepted 26 February 2018

doi: 10.18520/cs/v115/i2/258-267