Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

End-to-end delay analysis for networked systems*

Jie SHEN†1, Wen-bo HE[2], Xue LIU[2], Zhi-bo WANG[3,4], Zhi WANG†‡1, Jian-guo YAO[5]

([1]*Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China*)

([2]*School of Computer Science, McGill University, Montreal H3A0E9, Canada*)

([3]*School of Computer, Wuhan University, Wuhan 430072, China*)

([4]*Su zhou Institute of Wuhan University, Suzhou 215000, China*)

([5]*School of Software, Shanghai Jiao Tong University, Shanghai 200240, China*)

†E-mail: jshen@iipc.zju.edu.cn; wangzhi@iipc.zju.edu.cn

Received Dec. 4, 2014; Revision accepted July 18, 2015; Crosschecked Aug. 6, 2015

Abstract: End-to-end delay measurement has been an essential element in the deployment of real-time services in networked systems. Traditional methods of delay measurement based on time domain analysis, however, are not efficient as the network scale and the complexity increase. We propose a novel theoretical framework to analyze the end-to-end delay distributions of networked systems from the frequency domain. We use a signal flow graph to model the delay distribution of a networked system and prove that the end-to-end delay distribution is indeed the inverse Laplace transform of the transfer function of the signal flow graph. Two efficient methods, Cramer's rule-based method and the Mason gain rule-based method, are adopted to obtain the transfer function. By analyzing the time responses of the transfer function, we obtain the end-to-end delay distribution. Based on our framework, we propose an efficient method using the dominant poles of the transfer function to work out the bottleneck links of the network. Moreover, we use the framework to study the network protocol performance. Theoretical analysis and extensive evaluations show the effectiveness of the proposed approach.

Key words: Networked system, End-to-end, Delay distribution
doi:10.1631/FITEE.1400414 **Document code:** A **CLC number:** TP393

1 Introduction

Technical advances in sensing, computing, and communication are leading to the development of complicated and heterogeneous larger-scale networked systems, such as Internet systems. For these systems, especially for the systems providing real-time services, end-to-end delay measurement is an essential aspect. When the end-to-end delay distribution is obtained, we are able to verify whether the quality of service (QoS) requirements are met. Thus, to better guarantee QoS, we need useful tools to model and obtain the end-to-end delay distribution.

In networks, generally speaking, the delay distribution of each single link or node follows a random distribution (Almeida *et al.*, 1999), and it can be obtained by theoretical analysis or experiments. However, the end-to-end delay of a networked system which follows a superimposed distribution is extremely complicated considering the system's heterogeneity and complexity. For example, it is difficult to obtain the end-to-end delay distribution of a large-scale system composed of many sub-networks. Traditional methods based on time domain analysis are not convenient here for the following

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 61273079 and 61502352), the Key Laboratory of Wireless Sensor Network & Communication of Chinese Academy of Sciences (No. WSNC2014001), the Open Research Project of the State Key Lab of Industrial Control Technology, Zhejiang University (Nos. ICT1541 and ICT1555), the Natural Science Foundation of Hubei Province, China (No. 2015CFB203), and the Natural Science Foundation of Jiangsu Province, China (No. BK20150383)

ORCID: Jie SHEN, http://orcid.org/0000-0003-4391-813X

reasons: First, they calculate the end-to-end delay distribution by convoluting the delay distributions of individual nodes and links, and the computational cost is extremely high when the network is complicated and its scale is large. Second, they usually find it hard to capture the higher-order delay characteristics, which may help us better design the systems. Third, they may need to rebuild the whole system model when a change in a single network component occurs. It is not scalable. Fourth, in most situations, traditional time-domain-based approaches cannot obtain closed-form solutions. Therefore, there is an urgent demand for novel methodologies to efficiently analyze the end-to-end delay distribution for large-scale networked systems.

In this paper, we propose a new theoretical framework for end-to-end delay distribution analysis of networked systems through frequency control theory. In contrast to our previous work (He *et al.*, 2010) which is related to network structures, in this paper, we use a signal flow graph to model the delay distributions of a networked system, which does not depend on any particular network structure. Moreover, our framework can be used to efficiently model and analyze network protocol performance. We prove that the end-to-end delay distribution in the time domain is indeed the inverse Laplace transform of the transfer function of the signal flow graph. This implies that we could first calculate the system transfer function in the frequency domain, and then obtain the end-to-end delay distribution from the transfer function. Two methods, Cramer's rule-based method and the Mason gain rule-based method, are adopted to efficiently obtain the transfer function. The end-to-end delay distribution is obtained by analyzing the time responses of the system transfer function. Based on our framework, the dominant pole and its related theory are introduced to work out the bottleneck links. By improving the delay distribution of bottleneck links, the end-to-end delay distribution of the networked systems can be efficiently improved. Lastly, we use the proposed framework to study the network protocol performance in terms of end-to-end delay distribution.

To the best of our knowledge, our work is the first applying control theory to end-to-end delay distribution calculation. The contributions of this paper are summarized as follows:

1. We use a signal flow graph to model the delay distributions of a networked system, and prove that the end-to-end delay distribution in the time domain is indeed the inverse Laplace transform of the transfer function of the signal flow graph. This enables us to use the transfer-function-based classical control theory to calculate and analyze the end-to-end delay distribution.

2. We propose a method based on the dominant poles of the transfer function to work out the bottleneck links. By improving the bottleneck links, we significantly improve the end-to-end delay distribution.

3. We use our framework to efficiently study transport protocols with different retransmission mechanisms for reliable communication, i.e., end-to-end vs. hop-by-hop. The proposed analytical framework will be a powerful tool to evaluate various message transmission mechanisms.

2 Related work

End-to-end delay analysis has always been an important research field in networked systems (Wang *et al.*, 2014) with research on many different kinds of networks. For the Internet, the end-to-end packet delay and loss behavior were analyzed using the measured round trip delays of small user datagram protocol (UDP) probe packets (Bolot, 1993); a large-scale experiment was done to study end-to-end Internet packet dynamics (Paxson, 1997). For the Ethernet, an optimized trajectory approach was proposed to improve the worst-case delay of an avionics full duplex switched Ethernet (AFDX) network (Bauer *et al.*, 2010). In wireless sensor networks, a method of using two-hop information to enhance real-time delivery was proposed (Li *et al.*, 2009); an approach which employs a frequency domain analysis for estimating end-to-end delay in multi-hop networks was presented (Despaux *et al.*, 2012); discrete-time queueing theory was employed to analyze the end-to-end delay in wireless multi-hop networks (Xie and Haenggi, 2009). End-to-end delay is also researched in cyber-physical systems (Rao *et al.*, 2010; Xia *et al.*, 2011), network control (Yao *et al.*, 2013), and other networked systems.

To analyze network delay, there have been a lot of fundamental research works on delay analysis. Some works evaluate performance of queueing systems in terms of their low-order statistics, like the

mean and the variance (Abdelzaher *et al.*, 2004; Bisnik and Abouzeid, 2009; Gupta and Shroff, 2009). However, in many cases, the distribution other than the mean and the variance of the end-to-end delay is required to design networked systems. A method using the maximum variance asymptotic (MVA) upper bound to calculate an upper bound of the buffer tail distribution in a queue with Gaussian input traffic was proposed (Choe and Shroff, 1998). However, the assumption of specific traffic makes these approaches not sufficiently general. Statistical frameworks were proposed (Qiu and Knightly, 1999; Boorstyn *et al.*, 2000; Reisslein *et al.*, 2002; Fidler, 2010), and a few important properties such as statistical multiplexing, end-to-end network delay probability, statistical traffic envelopes, and statistical service envelopes were used to describe the cumulative traffic. However, the computational complexity of statistical methods can be high and sometimes it is hard to gain closed-form solutions. To achieve deterministic performance, the concept of 'network calculus' was proposed to derive bounds for delay through worst case analysis (Cruz, 1991a; 1991b). Further, to analyze the random characteristics of network delay, statistical network calculus can provide statistical delay bounds (Burchard *et al.*, 2006; Koubaa *et al.*, 2006; Schmitt *et al.*, 2007). However, studying the worst case is always conservative.

Computational complexity is one key challenge in end-to-end delay distribution analysis. The computation of the traditional methods based on the time domain is high. Therefore, providing approaches with low computational complexity is the design goal of our work.

3 System modeling

The end-to-end delay distribution is commonly analyzed in the time domain, which, however, incurs heavy computation overhead due to convolution operation of individual nodes and links. He *et al.* (2010) proposed to convert the end-to-end delay distribution analysis from the time domain to the frequency domain, to simply analyze and reduce computation complexity. Let $d(t)$ denote the probability density function of the delay of a system. It is converted into the frequency domain by the Laplace transform as follows:

$$D(s)=\mathcal{L}[d(t)]=\int_0^{\infty}\mathrm{e}^{-st}d(t)\mathrm{d}t, \tag{1}$$

where $\mathcal{L}$ is the operator of the Laplace transform. Meanwhile, if we obtain $D(s)$ by using analysis in the frequency domain, we can easily obtain $d(t)$ by using the inverse Laplace transform as follows:

$$d(t)=\mathcal{L}^{-1}[D(s)]=\frac{1}{\mathrm{j}2\pi}\int_{\sigma-\mathrm{j}\infty}^{\sigma+\mathrm{j}\infty}\mathrm{e}^{st}D(s)\mathrm{d}s. \tag{2}$$

In this paper, we follow the idea of He *et al.* (2010) and analyze the end-to-end delay distribution from the frequency domain. The first problem is how to model the delay distributions of the networked system so that we can obtain $D(s)$ in the frequency domain efficiently. We propose to model the delay distribution as a 'signal flow graph'.

A signal flow graph is a directed graph with vertices and directed edges. Let $G=(V,E,W)$ denote the modeled signal flow graph, where $V=\{x_1,x_2,\cdots,x_n\}$ is the set of vertices. Each vertex x_i represents the cumulative delay distribution from the starting node to node i. For example, in a networked system, we send a message from node 1 to node n. Correspondingly, in the signal flow graph, vertex x_1 is the 'source node', and vertex x_n is the 'sink node'. We always have $x_1=1$ (in the frequency domain), which means no cumulative delay at that point. x_n represents the cumulative delay distribution from node 1 to node n, i.e., the end-to-end delay distribution of the system. $E=\{e(i,j)\}$ is the set of edges. An edge $e(i,j)$ exists if and only if there is a link from node i to node j. $W:E\to\mathbb{R}$ is the set of weights of edges which characterize the delay distribution of each link.

The delay function is the Laplace transform of the delay distribution. Suppose $d(i,j)$ is the delay distribution from node i to node j. Then we define its delay function as follows:

Definition 1 The delay function of an edge $e(i,j)$ is defined as $D(i,j)=\mathcal{L}[d(i,j)]$.

A system has different structures, such as serial, parallel, and circular. In a serial structure, each node has only one incoming link and only one outgoing link. Therefore, the probability of choosing each link is obviously 1. However, in a system which contains parallel and circular structures, one node may have multiple incoming links and outgoing links,

and therefore a link is selected with a certain probabilities. Meanwhile, the sum of probabilities for outgoing links of a node is 1. Let $p(i,j)$ denote the probability of edge $e(i,j)$ being selected. Then we have $\sum_j p(i,j)=1$.

Definition 2 The weight of an edge $e(i,j)$ in a signal flow graph G is defined as the product of its probability and delay function. That is, $w(i,j)=p(i,j)D(i,j)$.

Fig. 1b shows the signal flow graph corresponding to the delay distributions of the networked system shown in Fig. 1a. x_1 is the source node, and x_6 is the sink node. In the signal flow graph, there exist 'paths' from the source node to the sink node. Further, a path in which every node appears only once is called a 'forward path'. For example, in Fig. 1b, $x_1 \to x_2 \to x_3 \to x_6$ is a forward path. The signal flow graph is also able to model the networked system with loops. For example, $x_4 \to x_5 \to x_4$ is a loop. In many cases, higher-level protocols prevent packets from running in loops in the network to avoid congestion. This would have an influence on the probability of a packet using such a link, because the probability would become conditional.

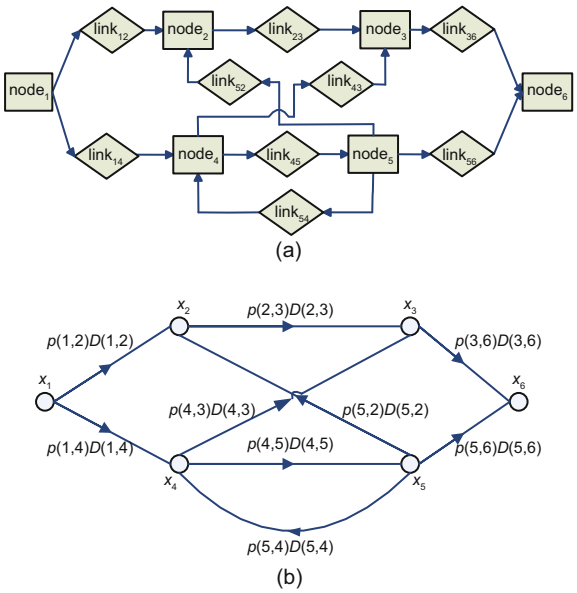

Fig. 1 A networked system architecture: (a) nodes and links; (b) signal flow graph

As shown in Fig. 1b, the signal flow graph is one-way. However, a networked system is always composed of full duplex links, and the link delays are asymmetric (Exel *et al.*, 2014). Thus, in fact, we need two signal flow graphs to model the end-to-end delays of a networked system. For example, we need another signal flow graph to model the end-to-end delay from node 6 to node 1 in our example.

In control theory, the most important factor of a signal flow graph is the 'transfer function' (D'Azzo and Houpis, 1995). It is a mathematical representation to describe system inputs and outputs.

Definition 3 The transfer function $T(i,j)$ of an edge (or link) $e(i,j)$ is equal to its weight. That is, $T(i,j)=w(i,j)$.

The convolution operation in the time domain is converted into the multiplication operation in the frequency domain. Therefore, we have the following two definitions and theorem:

Definition 4 The transfer function $T(P_k)$ of path P_k is the multiplication of the transfer functions of all the links on the path. That is, $T(P_k)=\prod_{e(i,j)\in E(P_k)} T(i,j)$, where $E(P_k)$ is the set of all the links on P_k.

Definition 5 The transfer function of the signal flow graph G is defined as T_s, which is equal to the sum of the transfer functions of all the paths (D'Azzo and Houpis, 1995). That is, $T_s=\sum_{P_i\in P(G)} T(P_i)$, where $P(G)$ is the set of all the paths of G.

Theorem 1 The end-to-end delay distribution of a system is equal to the inverse Laplace transform of the transfer function of its corresponding signal flow graph G. That is, $d_{\mathrm{e2e}}=\mathcal{L}^{-1}[T_s]$.

Proof First, by Definition 5, we have

$$T_s=\sum_{P_i\in P(G)} T(P_i). \tag{3}$$

Substituting the property $T(P_k)=\prod_{e(i,j)\in E(P_k)} T(i,j)$ in Definition 4, we obtain

$$\begin{aligned} T_s &= \sum_{P_i\in P(G)} \prod_{e(i,j)\in E(P_i)} T(i,j) \\ &= \sum_{P_i\in P(G)} \prod_{e(i,j)\in E(P_i)} w(i,j) \\ &= \sum_{P_i\in P(G)} \prod_{e(i,j)\in E(P_i)} p(i,j)D(i,j) \\ &= \sum_{P_i\in P(G)} \prod_{e(i,j)\in E(P_i)} p(i,j) \prod_{e(i,j)\in E(P_i)} D(i,j). \end{aligned} \tag{4}$$

The inverse Laplace transform of the transfer

function T_s is

$$\begin{aligned}
&\mathcal{L}^{-1}[T_s]\\
&=\mathcal{L}^{-1}\left[\sum_{P_i\in P(G)}\prod_{e(i,j)\in E(P_i)}p(i,j)\prod_{e(i,j)\in E(P_i)}D(i,j)\right]\\
&=\sum_{P_i\in P(G)}\mathcal{L}^{-1}\left[\prod_{e(i,j)\in E(P_i)}p(i,j)\prod_{e(i,j)\in E(P_i)}D(i,j)\right]\\
&=\sum_{P_i\in P(G)}\prod_{e(i,j)\in E(P_i)}p(i,j)\mathcal{L}^{-1}\left[\prod_{e(i,j)\in E(P_i)}D(i,j)\right]\\
&=\sum_{P_i\in P(G)}\prod_{e(i,j)\in E(P_i)}p(i,j)\bigotimes_{e(i,j)\in E(P_i)}d(i,j),
\end{aligned}\tag{5}$$

where $d(P_i)=\otimes_{e(i,j)\in E(P_i)}d(i,j)$ is the end-to-end delay distribution of path P_i, and it is calculated by the convolution of all the delay distributions of links on path P_i. Let $p(P_i)$ denote the probability of choosing path P_i. We have $p(P_i)=\prod_{e(i,j)\in E(P_i)}p(i,j)$. Thus,

$$\mathcal{L}^{-1}[T_s]=\sum_{P_i\in P(G)}p(P_i)d(P_i).\tag{6}$$

Since $\sum_{P_i\in P(G)}p(P_i)d(P_i)$ is the end-to-end delay distribution of the networked system, we obtain

$$d_{\mathrm{e2e}}=\mathcal{L}^{-1}[T_s].\tag{7}$$

Therefore, we convert the end-to-end delay distribution calculation problem into the problem of obtaining the transfer function of the corresponding signal flow graph. After we obtain the transfer function, the end-to-end delay distribution can be easily calculated using the inverse Laplace transform.

4 System transfer function calculation

Since the end-to-end delay distribution is indeed the inverse Laplace transform of the transfer function of the signal flow graph, we need to obtain the transfer function to otain the end-to-end delay. In this section, we present two commonly used methods to obtain the transfer function of the signal flow graph model: Cramer's rule-based method and the Mason gain rule-based method. We use an example here to better explain these methods. Take the signal flow graph model shown in Fig. 1b as an example. The parameters of the system are listed in Table 1. The delay distributions in Table 1 have a form like $a/(s+a)$, which is an exponential distribution. The expected (or mean) delay is given by $1/a$. Thus, a smaller a means a poorer link.

Table 1 Parameters of the example networked system

Link	Delay distribution	Probability
$e(1,2)$	$D(1,2)=9/(s+9)$	$p(1,2)=0.7$
$e(1,4)$	$D(1,4)=7/(s+7)$	$p(1,4)=0.3$
$e(2,3)$	$D(2,3)=5/(s+5)$	$p(2,3)=1.0$
$e(3,6)$	$D(3,6)=1/(s+1)$	$p(3,6)=1.0$
$e(4,3)$	$D(4,3)=4/(s+4)$	$p(4,3)=0.2$
$e(4,5)$	$D(4,5)=7/(s+7)$	$p(4,5)=0.8$
$e(5,2)$	$D(5,2)=0.5/(s+0.5)$	$p(5,2)=0.3$
$e(5,4)$	$D(5,4)=5/(s+5)$	$p(5,4)=0.1$
$e(5,6)$	$D(5,6)=8/(s+8)$	$p(5,6)=0.6$

4.1 Cramer's rule-based method

One advantage of the signal flow graph model is that we may obtain a set of algebraic equations which clearly indicate the flow of signals (delay cumulation) from the model. For example, in Fig. 1b, node x_1 and node x_5 flow into x_2, and they are multiplied by $p(1,2)D(1,2)$ and $p(5,2)D(5,2)$, respectively. So, we obtain $x_2=p(1,2)D(1,2)x_1+p(5,2)D(5,2)x_5$. Once all these algebraic equations are obtained, we may obtain the transfer function by solving them using Cramer's rule.

For the system shown in Fig. 1b, we obtain the following equations:

$$\begin{cases}
x_1=1,\\
x_2=p(1,2)D(1,2)x_1+p(5,2)D(5,2)x_5\\
\quad=T(1,2)x_1+T(5,2)x_5,\\
x_3=p(2,3)D(2,3)x_2+p(4,3)D(4,3)x_4\\
\quad=T(2,3)x_2+T(4,3)x_4,\\
x_4=p(1,4)D(1,4)x_1+p(5,4)D(5,4)x_5\\
\quad=T(1,4)x_1+T(5,4)x_5,\\
x_5=p(4,5)D(4,5)x_4\\
\quad=T(4,5)x_4,\\
x_6=p(3,6)D(3,6)x_3+p(5,6)D(5,6)x_5\\
\quad=T(3,6)x_3+T(5,6)x_5.
\end{cases}\tag{8}$$

There are six fundamental equations with six variables $x_1,x_2,...,x_6$, which should be solvable. Rewriting Eq. (8) in the form of a matrix, we obtain

$$\boldsymbol{AX}=\boldsymbol{U},\tag{9}$$

where

$$\boldsymbol{A}=\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ T(1,2) & -1 & 0 & 0 & T(5,2) & 0 \\ 0 & T(2,3) & -1 & T(4,3) & 0 & 0 \\ T(1,4) & 0 & 0 & -1 & T(5,4) & 0 \\ 0 & 0 & 0 & T(4,5) & -1 & 0 \\ 0 & 0 & T(3,6) & 0 & T(5,6) & -1 \end{bmatrix}, \tag{10}$$

$$\boldsymbol{X}=[x_1\ x_2\ x_3\ x_4\ x_5\ x_6]^{\mathrm{T}}, \tag{11}$$

$$\boldsymbol{U}=[-1\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}. \tag{12}$$

The determinant of $\boldsymbol{A}$ is

$$\Delta=|\boldsymbol{A}|=1-T(5,4)T(4,5). \tag{13}$$

By replacing the 6th column of $\boldsymbol{A}$ by the column vector $\boldsymbol{U}$, we obtain $\boldsymbol{A}'$. The determinant of $\boldsymbol{A}'$ is

$$\Delta_6=\begin{vmatrix} -1 & 0 & 0 & 0 & 0 & -1 \\ T(1,2) & -1 & 0 & 0 & T(5,2) & 0 \\ 0 & T(2,3) & -1 & T(4,3) & 0 & 0 \\ T(1,4) & 0 & 0 & -1 & T(5,4) & 0 \\ 0 & 0 & 0 & T(4,5) & -1 & 0 \\ 0 & 0 & T(3,6) & 0 & T(5,6) & 0 \end{vmatrix}. \tag{14}$$

That is,

$$\begin{aligned}\Delta_6=&T(1,2)T(4,3)[1-T(5,4)T(4,5)]\\&+T(1,4)T(4,3)T(3,6)+T(1,4)T(4,5)T(5,6)\\&+T(1,4)T(4,5)T(5,2)T(2,3)T(3,6).\end{aligned} \tag{15}$$

By Cramer's rule, we obtain

$$x_6=\frac{\Delta_6}{\Delta}. \tag{16}$$

Since packages are delivered from x_1 to x_6, we have the input $U_{\mathrm{i}}=x_1=1$ and the output $U_{\mathrm{o}}=x_6$. Thus, the transfer function which represents the relationship between the input and output is obtained as

$$T_s=\frac{U_{\mathrm{o}}}{U_{\mathrm{i}}}=\frac{x_6}{x_1}=\frac{N(s)}{D(s)}, \tag{17}$$

where

$$\begin{aligned}N(s)=&T(1,2)T(4,3)[1-T(5,4)T(4,5)]\\&+T(1,4)T(4,3)T(3,6)\\&+T(1,4)T(4,5)T(5,6)\\&+T(1,4)T(4,5)T(5,2)T(2,3)T(3,6),\end{aligned} \tag{18}$$

$$D(s)=1-T(5,4)T(4,5). \tag{19}$$

Therefore, the transfer function T_s is obtained using Cramer's rule-based method. An advantage of using this method is that we are able to obtain the cumulative delay distribution from the source node to any other node (by solving x_2, x_3, x_4, x_5 in the example) besides the sink node, which is useful in situations where not only the end-to-end delay needs to be considered. However, this method needs matrix operations.

4.2 Mason's gain rule-based method

In control theory, the Mason gain rule is another method for obtaining the transfer function from the signal flow graph. It searches into the graph for different paths and loops to obtain the result. Compared with Cramer's rule-based method, this does not need matrix operations and may be more computationally efficient. The Mason gain formula (D'Azzo and Houpis, 1995) is written as

$$T_s=\frac{\sum T(\mathrm{FP}_k)\Delta_k}{\Delta}, \tag{20}$$

where $T(\mathrm{FP}_k)$ is the transfer function of a forward path between a source node and a sink node. Δ is the graph determinant which is derived from

$$\Delta=1-\sum T_a+\sum T_bT_c-\sum T_dT_eT_f+\cdots, \tag{21}$$

where T_a is the transfer function of each closed path, T_bT_c is the product of the transfer functions of two non-touching loops, and $T_dT_eT_f$ is the product of the transfer functions of three non-touching loops.

In Eq. (20), Δ_k is the cofactor of $T(\mathrm{FP}_k)$. It is the determinant of the remaining sub-graph when the forward path FP_k is removed. Thus, it does not include any loops that touch the forward path in question. Δ_k is equal to unity when the forward path touches all the loops in the graph or when the graph does not contain any loop. Δ_k has the same form as Eq. (21).

In Fig. 1b, there is only one loop: $x_4\to x_5\to x_4$, and its transfer function is $T_a=T(4,5)T(5,4)$. Therefore, $\sum T_a=T_a=T(4,5)T(5,4)$. So, we have

$$\Delta=1-\sum T_a=1-T(4,5)T(5,4). \tag{22}$$

There are four forward paths in Fig. 1b. The forward paths, path transfer functions, and cofactors are listed in Table 2.

Table 2 Forward paths, path transfer functions, and cofactors of the system in Fig. 1b

Forward path	Path transfer function	Cofactor
FP_1: $x_1 \to x_2 \to x_3 \to x_6$	$T(\mathrm{FP}_1) = T(1,2)T(2,3)T(3,6)$	$\Delta_1 = 1 - T(4,5)T(5,4)$
FP_2: $x_1 \to x_4 \to x_5 \to x_6$	$T(\mathrm{FP}_2) = T(1,4)T(4,5)T(5,6)$	$\Delta_2 = 1$
FP_3: $x_1 \to x_4 \to x_3 \to x_6$	$T(\mathrm{FP}_3) = T(1,4)T(4,3)T(3,6)$	$\Delta_3 = 1$
FP_4: $x_1 \to x_4 \to x_5 \to x_2 \to x_3 \to x_6$	$T(\mathrm{FP}_4) = T(1,4)T(4,5)T(5,2)T(2,3)T(3,6)$	$\Delta_4 = 1$

Thus, by Eq. (20) we obtain the transfer function as follows:

$$\begin{aligned} T_s &= \frac{\sum T(\mathrm{FP}_k)\Delta_k}{\Delta} \\ &= \frac{T(\mathrm{FP}_1)\Delta_1 + T(\mathrm{FP}_2)\Delta_2 + T(\mathrm{FP}_3)\Delta_3 + T(\mathrm{FP}_4)\Delta_4}{1 - \sum T_a} \\ &= \frac{N(s)}{D(s)}. \end{aligned} \tag{23}$$

The result of the Mason gain rule method is the same as that of Cramer's rule-based method. The Mason gain rule method does not need any matrix operation. However, it will become slow as the system's complexity increases (El-Hajj and Kabalan, 1995).

Overall, in this section, we introduced two methods, Cramer's rule-based method and the Mason gain rule-based method, to calculate the system's transfer function (i.e., Laplace transform of the end-to-end delay distribution). Either of the two methods requires multiplication operations only and these are more computationally efficient than convolution operations in time-domain-based approaches. The advantage of the Mason gain rule-based method is that it does not need any matrix operation; i.e., it is more computationally efficient. However, the Mason gain rule-based method can calculate only the end-to-end delay (i.e., delay from the source node to the sink node), while Cramer's rule-based method is able to obtain the delay from the source node to any other node, which is useful in some situations. Requirements will dictate the appropriate method in practice.

5 Delay distribution analysis

In this section, we analyze the time responses of the system transfer function to obtain the end-to-end delay distribution.

We continue to use the example of Fig. 1b. The system transfer function has already been obtained using Eqs. (17) and (23). Given the parameters listed in Table 1, there is

$$T_s = \frac{N(s)}{D(s)}, \tag{24}$$

where

$$\begin{aligned} N(s) =& 44\,814s^6 + 1\,216\,593s^5 + 12\,949\,629s^4 \\ &+ 68\,632\,557s^3 + 186\,397\,407s^2 \\ &+ 231\,016\,800s + 81\,144\,000, \end{aligned} \tag{25}$$

$$\begin{aligned} D(s) =& 500s^9 + 23\,250s^8 + 461\,600s^7 + 5\,097\,450s^6 \\ &+ 34\,151\,700s^5 + 141\,822\,750s^4 \\ &+ 356\,688\,400s^3 + 504\,152\,550s^2 \\ &+ 340\,937\,800s + 81\,144\,000. \end{aligned} \tag{26}$$

The impulse response and step response are two typical time responses. The impulse response equals the probability density function of the end-to-end delay distribution, and the step response equals the cumulative distribution function of the end-to-end delay distribution. We give the explanations as follows:

Let $\delta(t)$ denote the input impulse function, and $y_{\delta(t)}$ the time response for $\delta(t)$. The system transfer function is T_s. We can obtain

$$\mathcal{L}[y_{\delta(t)}] = \mathcal{L}[\delta(t)] \cdot T_s = 1 \cdot T_s = T_s. \tag{27}$$

By the inverse Laplace transform, the impulse response can be obtained as

$$y_{\delta(t)} = \mathcal{L}^{-1}[T_s]. \tag{28}$$

Using Theorem 1, we have $d_{\mathrm{e2e}} = \mathcal{L}^{-1}[T_s]$, and thus $y_{\delta(t)} = d_{\mathrm{e2e}}$; i.e., the impulse response equals the probability density function of the end-to-end delay distribution.

Let $u(t)$ denote the input step function, and $y_{u(t)}$ the time response for $u(t)$. We can obtain

$$\mathcal{L}[y_{u(t)}] = \mathcal{L}[u(t)] \cdot T_s = \frac{1}{s} \cdot T_s. \tag{29}$$

The step response can be obtained by the inverse Laplace transform as

$$y_{u(t)} = \mathcal{L}^{-1}\left(\frac{1}{s} \cdot T_s\right) = \int_0^t d_{\mathrm{e2e}}(\tau)\mathrm{d}\tau = D_{\mathrm{e2e}}, \tag{30}$$

where D_{e2e} is the cumulative distribution function of the end-to-end delay distribution. So, the step response equals the cumulative distribution function of the end-to-end delay distribution.

Therefore, checking the characteristics of the time response is exactly checking that of the end-to-end delay distribution.

Fig. 2 shows the impulse response and step response of our system of Fig. 1b. The impulse response curve describes the relative likelihood for a message delivered at a given time. It will move to the left when the mean end-to-end delay is shorter, and will be much sharper when jitter (or delay variation) is smaller. We find that the end-to-end delay occurs mainly between 0 and 2 s. The probability of the end-to-end delay being larger than 5 s is small.

Also, we see that the cumulative distribution function D_{e2e} increases as the time increases and $\lim_{t\to\infty} D_{\mathrm{e2e}} = 1$. This implies that messages can finally be successfully delivered to the end without a time deadline limit. However, in practice, we have end-to-end deadlines and messages will be lost. Moreover, from the step response curve, we obtain the probability of successful delivery within any deadline. For example, 80% messages can be delivered from node 1 to node 6 within 2 s.

We have developed the theoretical framework to model and analyze the end-to-end delay distribution of networked systems. The procedures are shown in Algorithm 1.

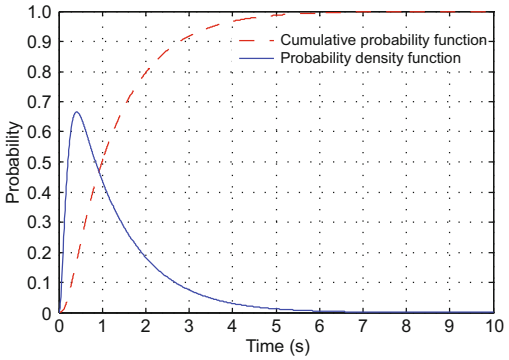

Fig. 2 Probability density function and cumulative distribution function of end-to-end delay

Algorithm 1 Theoretical framework to model and analyze the end-to-end delay distribution of networked systems

1: Obtain the individual delay distributions of nodes and links of networked systems, and convert them into the frequency domain using the Laplace transform.
2: Model the delay distributions using the signal flow graph.
3: Calculate the transfer function of the signal flow graph model by using Cramer's rule-based method or the Mason gain rule method.
4: Analyze the time responses of the transfer function to study the end-to-end delay distribution.

6 Case study

6.1 Case study 1 to find bottleneck links

In this subsection, we give an example using the proposed framework to efficiently work out the bottleneck links.

The transfer function of the signal flow graph in Eqs. (17) and (23) is written in the form of zero-poles:

$$T_s(s) = \frac{k(s-z_1)(s-z_2)\cdots(s-z_m)}{(s-b_1)(s-b_2)\cdots(s-b_n)}. \tag{31}$$

The zeros of this function, $s = z_h$ $(h = 1, 2, \ldots, m)$, are those values of s for which $T_s(z_h) = 0$. The poles of this function, $s = b_k$ $(k = 1, 2, \ldots, n)$, are those values of s for which $|T_s(b_k)| = \infty$.

Among all the poles of the system, there exist 'dominant poles' which play more important roles than the non-dominant poles. Generally speaking, the time responses due to dominant poles are large in magnitude and die out slowly, while the time responses due to non-dominant poles are small in magnitude and die out quickly. Therefore, the system time response (i.e., end-to-end delay distribution) is determined mainly by dominant poles (D'Azzo and Houpis, 1995). We summarize the features of the dominant pole as follows:

Definition 6 (Dominant pole) Let $B = \{b_i|\mathrm{Re}(b_i) < 0, i = 1, 2, \cdots, n\}$ denote the set of all the poles of T_s, and $Z = \{z_j|\mathrm{Re}(z_j) < 0, j = 1, 2, \cdots, m\}$ the set of all the zeros of T_s. Let B_{d} denote the dominant poles and B_{n} the non-dominant poles. For any $b_i \in B_{\mathrm{d}}$ and $b_j \in B_{\mathrm{n}}$, it must satisfy one of the following conditions:

$$\mathrm{Re}(b_j) < \gamma \cdot \mathrm{Re}(b_i), \tag{32}$$

or

$$|b_j - z_i| < \varepsilon, \tag{33}$$

where $\mathrm{Re}(\cdot)$ denotes the real part, $\gamma \geq 3$, and ε is small enough.

The dominant poles are usually close to the imaginary axis, while the non-dominant poles are usually far away from the left of the dominant poles.

Since the system is composed of links, the poles of the system depend on the delay distributions of individual links. Thus, there exist links named 'dominant links' (bottleneck link) in this paper which decide the dominant poles of the system.

Definition 7 (Dominant link) Let E_{d} denote the set of all the dominant links and E_{n} the non-dominant links. The transfer function $T(i,j)$ of link $e(i,j) \in E_{\mathrm{d}}$ has at least one pole which is the same as (or close to) one of the dominant poles of the system transfer function T_s. When we change link $e(i,j) \in E_{\mathrm{d}}$, there exists at least one dominant pole $b \in B_{\mathrm{d}}$ such that $\mathrm{Re}(b)$ will be changed. However, when we change link $e(l,m) \in E_{\mathrm{n}}$, none of dominant poles is affected.

Take the networked system shown in Fig. 1b as an example. The zeros-poles form of the system transfer function is

$$T_s = \frac{N'(s)}{D'(s)}, \tag{34}$$

where

$$\begin{aligned} N'(s) =& 89.628(s+8.95)(s+3.8)(s+3.5)(s+0.55) \\ & \cdot (s+5.2+0.9\mathrm{j})(s+5.2-0.9\mathrm{j}), \end{aligned} \tag{35}$$

$$\begin{aligned} D'(s) =& (s+9)(s+8)(s+7.9)(s+7)(s+5)(s+4.1) \\ & \cdot (s+4)(s+1)(s+0.5). \end{aligned} \tag{36}$$

Obviously, we obtain the zeros and poles of the transfer function as follows: the zeros are $(-8.95,0)$, $(-3.8,0)$, $(-3.5,0)$, $(-0.55,0)$, $(-5.2,-0.9\mathrm{j})$, and $(-5.2,0.9\mathrm{j})$; the poles are $(-9,0)$, $(-8,0)$, $(-7.9,0)$, $(-7,0)$, $(-5,0)$, $(-4.1,0)$, $(-4,0)$, $(-1,0)$, and $(-0.5,0)$. By Definition 6, we find that $(-1,0)$ is the dominant pole of T_s. The reasons are as follows: first, although pole $(-0.5,0)$ is not to the left of pole $(-1,0)$, there exists a zero $(-0.55,0)$ close to it; second, the other poles are far from the left of $(-1,0)$.

The next step is to work out the dominant links. From Table 1, we find that the transfer function of link $e(3,6)$ is $T(3,6) = D(3,6)p(3,6) = 1/(s+1)$. Thus, its pole is $(-1,0)$, which is the same as the dominant pole of T_s. When we improve $D(3,6)$ from $1/(1+s)$ to $2/(s+2)$, the system transfer function changes accordingly. We find that the dominant pole of the system is changed from $(-1,0)$ to $(-2,0)$. Thus, by Definition 7, we judge that link $e(3,6)$ is the dominant link of the networked system.

It should be pointed out that the dominant link (bottleneck link) does not mean the worst link. In the example system, $e(5,2)$ with delay distribution of $D(5,2) = 0.5/(s+0.5)$ is the worst link. However, $(-0.5,0)$ is not the dominant pole of T_s. Thus, the worst link $e(5,2)$ is not the dominant link. The reason is that the probability of choosing $e(5,2)$ is low in our example. Generally speaking, if the worst link is barely used in the network, then it is not the dominant link.

To evaluate our method, we improve $e(3,6)$ and $e(5,2)$ to $2/(s+2)$, and obtain the corresponding end-to-end delay distribution as shown in Figs. 3 and 4. When we improve $e(3,6)$, the probability density function curve is sharper, and the cumulative distribution function curve increases faster. However, when we improve $e(5,2)$, the probability density function curve and the cumulative distribution function curve are rarely changed. We also improve the other links, and the result is that the end-to-end delay distribution is rarely improved in each situation. Thus, the results indicate that our method based on the proposed framework is effective.

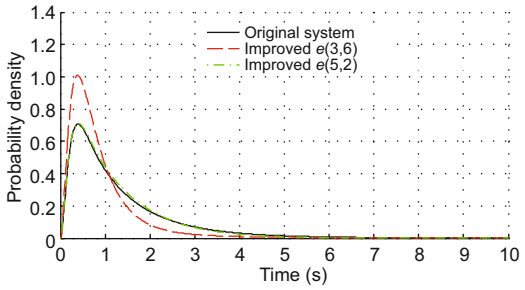

Fig. 3 The impulse response of the example system after improving a few links

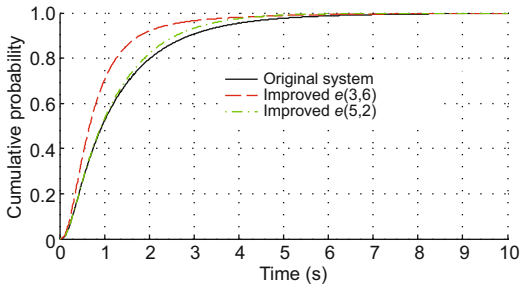

Fig. 4 Step response of the example system after improving a few links

6.2 Case study 2 to investigate packet forwarding protocol performance

Our framework can be used to study the network protocol performance. We will give an example here. There are two types of transport protocols with retransmission mechanisms for reliable communication, end-to-end vs. hop-by-hop, where end-to-end transport protocols (e.g., TCP) (Bakshi *et al.*, 1997; Balakrishnan *et al.*, 1997; Chakravorty *et al.*, 2003) have been widely used. In the end-to-end transport case, lost packets are retransmitted by the source. In the hop-by-hop transport protocol, the retransmission is initiated by the node prior to the hop where the loss occurs. We want to evaluate which type of protocol is better, and how much better. Let us assume: (1) the transport protocols use positive acknowledgment (ACK); (2) the retransmission timeout is selected accurately; (3) the delay over individual links follows an identical exponential distribution, $e(s) = 1/(1+s)$.

Consider a route between nodes x_i and x_j, which includes two links. Assume the packet loss probability per link is p. We use our framework to obtain the signal flow graph model for end-to-end transport protocols and hop-by-hop transport protocols in Figs. 5a and 5b, respectively. For end-to-end transport in Fig. 5b, by Cramer's rule-based method or the Mason gain rule-based method, we have

$$T_{ij}(s) = \frac{(1-p)^2 D^2(s)}{1-[1-(1-p)^2]D^2(s)}. \tag{37}$$

For hop-by-hop transport in Fig. 5b, we have

$$T'_{ij}(s) = \frac{(1-p)^2 D^2(s)}{[1-pD(s)]^2}. \tag{38}$$

We compare the performance of end-to-end transport and hop-by-hop transport protocols in Fig. 6, which shows the end-to-end delay distribution in terms of the percentage of packets that have been received within a given time period when link loss probabilities are 0.05 and 0.5, respectively.

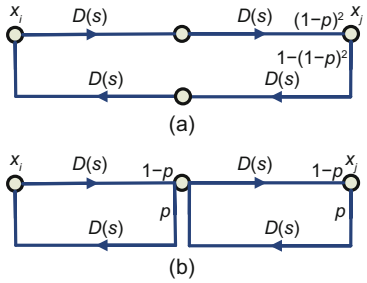

Fig. 5 End-to-end (a) and hop-by-hop (b) transport protocol models via retransmission

Fig. 6 shows that the performance of end-to-end transport is close to that of hop-by-hop transport, if the link loss probability p is small. However, the hop-by-hop transport is superior to its end-to-end alternative if p is large. This result agrees with that found in Heimlicher *et al.* (2007). Since this advantage comes at the price of higher protocol complexity and additional memory and processing requirements, we adopt end-to-end transport protocols in wired Internet applications, where the link loss probability p is small. However, if p is large (e.g., in wireless networks with intermittent connectivity), hop-by-hop transport is preferred.

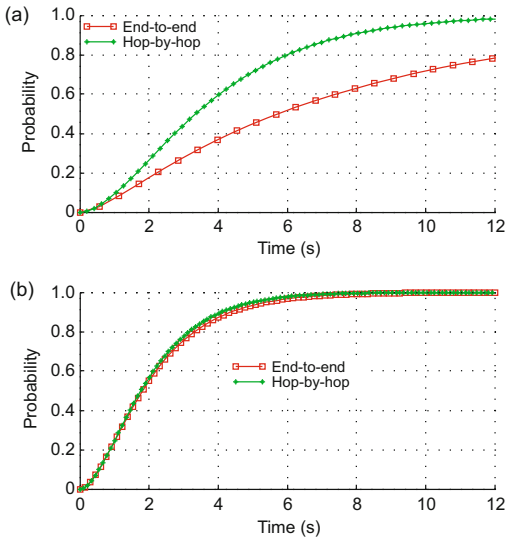

Fig. 6 Delay distribution under link loss probability p=0.5 (a) and p=0.05 (b)

It is important to determine which routes or protocols should be used for applications with end-to-end delay requirements. The above example shows how we adopt the proposed analytical framework to provide insights in protocol design and policy making in networked systems. Due to its flexibility and efficiency, the proposed analytical framework will be a powerful tool to evaluate various message transmission mechanisms.

7 Conclusions

In this paper, we propose a novel framework to analyze the end-to-end delay distribution for

networked systems. In our framework, the delay distributions of individual links are converted to the frequency domain by the Laplace transform. Then we construct a signal flow graph to model the delay distributions of the networked system. The end-to-end delay distribution of the networked system is proved to be the inverse Laplace transform of the transfer function of the signal flow graph. Thereafter, we adopt two computationally efficient methods, Cramer's rule-based method and the Mason gain rule-based method, to obtain the transfer function of the signal flow graph. Finally, we obtain the end-to-end delay distribution by analyzing the time responses of the transfer function. We give two case studies to show the use of our framework. First, we propose an efficient method based on our framework to efficiently work out the bottleneck links. Second, we use our framework to analyze network protocol performance and achieve good results. The examples given in this paper use exponential delay distributions. For the real distributions which are often complex, we can use the flexible combinations of nice delay distributions to represent them. As part of our future work, we will address these problems: (1) individual links are not independent; (2) the link delay distributions are not static; (3) the links are not available all the time.

References

Abdelzaher, T.F., Prabh, S., Kiran, R., 2004. On real-time capacity limits of multihop wireless sensor networks. Proc. 25th IEEE Int. Real-Time Systems Symp., p.359-370. [doi:10.1109/REAL.2004.37]

Almeida, L., Fonseca, P., Fonseca, J.A., *et al.*, 1999. Scheduling and clock synchronization in CAN-based distributed systems. Proc. Int. CAN Conf., p.1-9.

Bakshi, B.S., Krishna, P., Vaidya, N.H., *et al.*, 1997. Improving performance of TCP over wireless networks. Proc. 17th Int. Conf. on Distributed Computing Systems, p.365-373. [doi:10.1109/ICDCS.1997.598070]

Balakrishnan, H., Padmanabhan, V.N., Seshan, S., *et al.*, 1997. A comparison of mechanisms for improving TCP performance over wireless links. *IEEE/ACM Trans. Netw.*, **5**(6):756-769. [doi:10.1109/90.650137]

Bauer, H., Scharbarg, J., Fraboul, C., 2010. Improving the worst-case delay analysis of an AFDX network using an optimized trajectory approach. *IEEE Trans. Ind. Inform.*, **6**(4):521-533. [doi:10.1109/TII.2010.2055877]

Bisnik, N., Abouzeid, A.A., 2009. Queuing network models for delay analysis of multihop wireless ad hoc networks. *Ad Hoc Netw.*, **7**(1):79-97. [doi:10.1016/j.adhoc.2007.12.001]

Bolot, J.C., 1993. End-to-end packet delay and loss behavior in the Internet. *ACM SIGCOMM Comput. Commun. Rev.*, **23**(4):289-298. [doi:10.1145/167954.166265]

Boorstyn, R.R., Burchard, A., Liebeherr, J., *et al.*, 2000. Statistical service assurances for traffic scheduling algorithms. *IEEE J. Sel. Areas Commun.*, **18**(12):2651-2664. [doi:10.1109/49.898747]

Burchard, A., Liebeherr, J., Patek, S.D., 2006. A min-plus calculus for end-to-end statistical service guarantees. *IEEE Trans. Inform. Theory*, **52**(9):4105-4114. [doi:10.1109/TIT.2006.880019]

Chakravorty, R., Katti, S., Crowcroft, J., *et al.*, 2003. Flow aggregation for enhanced TCP over wide-area wireless. Proc. 22nd Annual Joint Conf. of the IEEE Computer and Communications, p.1754-1764. [doi:10.1109/INFCOM.2003.1209198]

Choe, J., Shroff, N.B., 1998. A central-limit-theorem-based approach for analyzing queue behavior in high-speed networks. *IEEE/ACM Trans. Netw.*, **6**(5):659-671. [doi:10.1109/90.731205]

Cruz, R.L., 1991a. A calculus for network delay, part I: network elements in isolation. *IEEE Trans. Inform. Theory*, **37**(1):114-131.

Cruz, R.L., 1991b. A calculus for network delay, part II: network analysis. *IEEE Trans. Inform. Theory*, **37**(1):132-141.

D'Azzo, J., Houpis, C., 1995. Linear Control System Analysis and Design: Conventional and Modern. McGraw-Hill Higher Education, USA.

Despaux, F., Song, Y.Q., Lahmadi, A., 2012. Combining analytical and simulation approaches for estimating end-to-end delay in multi-hop wireless networks. Proc. IEEE 8th Int. Conf. on Distributed Computing in Sensor Systems, p.317-322. [doi:10.1109/DCOSS.2012.31]

El-Hajj, A., Kabalan, K.Y., 1995. A transfer function computational algorithm for linear control systems. *IEEE Contr. Syst.*, **15**(2):114-118. [doi:10.1109/37.375319]

Exel, R., Bigler, T., Sauter, T., 2014. Asymmetry mitigation in IEEE 802.3 Ethernet for high-accuracy clock synchronization. *IEEE Trans. Instrum. Meas.*, **63**(3):729-736. [doi:10.1109/TIM.2013.2280489]

Fidler, M., 2010. Survey of deterministic and stochastic service curve models in the network calculus. *IEEE Commun. Surv. Tutor.*, **12**(1):59-86. [doi:10.1109/SURV.2010.020110.00019]

Gupta, G.R., Shroff, N., 2009. Delay analysis for multihop wireless networks. IEEE INFOCOM, p.2356-2364. [doi:10.1109/INFCOM.2009.5062162]

He, W., Liu, X., Zheng, L., *et al.*, 2010. Reliability calculus: a theoretical framework to analyze communication reliability. Proc. IEEE 30th Int. Conf. on Distributed Computing Systems, p.159-168. [doi:10.1109/ICDCS.2010.73]

Heimlicher, S., Nuggehalli, P., May, M., 2007. End-to-end vs. hop-by-hop transport. *SIGMETRICS Perform. Eval. Rev.*, **35**(3):59-60.

Koubaa, A., Alves, M., Tovar, E., 2006. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks. Proc. 27th IEEE Int. Real-Time Systems Symp., p.412-421. [doi:10.1109/RTSS.2006.29]

Li, Y., Chen, C.S., Song, Y.Q., *et al.*, 2009. Enhancing real-time delivery in wireless sensor networks with two-hop information. *IEEE Trans. Ind. Inform.*, **5**(2):113-122. [doi:10.1109/TII.2009.2017938]

Paxson, V., 1997. End-to-end Internet packet dynamics. *ACM SIGCOMM Comput. Commun. Rev.*, **27**(4):139-152. [doi:10.1145/263109.263155]

Qiu, J.Y., Knightly, E.W., 1999. Inter-class resource sharing using statistical service envelopes. Proc. 18th Annual Joint Conf. of the IEEE Computer and Communications Societies, p.1404-1411. [doi:10.1109/INFCOM.1999.752160]

Rao, L., Liu, X., Xie, L., *et al.*, 2010. Minimizing electricity cost: optimization of distributed Internet data centers in a multi-electricity-market environment. Proc. IEEE INFOCOM, p.1-9. [doi:10.1109/INFCOM.2010.5461933]

Reisslein, M., Ross, K.W., Rajagopal, S., 2002. A framework for guaranteeing statistical QoS. *IEEE/ACM Trans. Netw.*, **10**(1):27-42. [doi:10.1109/90.986511]

Schmitt, J.B., Zdarsky, F.A., Thiele, L., 2007. A comprehensive worst-case calculus for wireless sensor networks with in-network processing. Proc. 28th IEEE Int. Real-Time Systems Symp., p.193-202. [doi:10.1109/RTSS.2007.17]

Wang, Z., Liao, J., Cao, Q., *et al.*, 2014. Achieving k-barrier coverage in hybrid directional sensor networks. *IEEE Trans. Mob. Comput.*, **13**(7):1443-1455. [doi:10.1109/TMC.2013.118]

Xia, F., Vinel, A., Gao, R., *et al.*, 2011. Evaluating IEEE 802.15.4 for cyber-physical systems. *EURASIP J. Wirel. Commun. Netw.*, arXiv:1312.6837.

Xie, M., Haenggi, M., 2009. Towards an end-to-end delay analysis of wireless multihop networks. *Ad Hoc Netw.*, **7**(5):849-861. [doi:10.1016/j.adhoc.2008.04.010]

Yao, J., Liu, X., Zhu, G., *et al.*, 2013. NetSimplex: controller fault tolerance architecture in networked control systems. *IEEE Trans. Ind. Inform.*, **9**(1):346-356. [doi:10.1109/TII.2012.2219060]