A cache-aware social-based QoS routing scheme in Information Centric Networks

Dapeng Qu\textsuperscript{a}, Xingwei Wang\textsuperscript{b,a}, Min Huang\textsuperscript{c}, Keqin Li\textsuperscript{d}, Sajal K. Das\textsuperscript{e}, Sijin Wu\textsuperscript{b}

\textsuperscript{a} College of Information, Liaoning University, Shenyang, 110036, China
\textsuperscript{b} College of Computer Science and Engineering, Northeastern University, Shenyang, 110819, China
\textsuperscript{c} College of Information Science and Engineering, Northeastern University, Shenyang, 110819, China
\textsuperscript{d} Department of Computer Science, State University of New York, New Paltz, NY 12561, USA
\textsuperscript{e} Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409, USA

\textbf{ARTICLE INFO}

\textbf{Keywords:}
Cache-aware
Named data networking
Information centric network
QoS routing
Social relationship

\textbf{ABSTRACT}

Due to the rapid expansion of Internet and the huge proliferation of users, Internet has evolved from a host-centric model to a content-oriented model. This implies the in-adaptation of current TCP/IP architecture providing the best performance to end-users and the urgency of researching future Internet architecture. In future Internet, the named data rather than traditional IP address may become the thin waist of the hourglass model of networking. Therefore, in this paper, we propose a cache-aware social-based Quality of Service (QoS) routing scheme for Named Data Networking (NDN) in Information Centric Network (ICN). Three kinds of social relationships, namely neighbors (NB), interest friends (IF) and response friends (RF) are devised to describe the relationships among nodes. Thus, when there is a failure in doing Pending Interest Table (PIT) scheme, a forwarding scheme based on social relationships is done before doing Forwarding Information Base (FIB) scheme. Moreover, a caching policy and its corresponding replacement policy based on content popularity, cache space and neighbor caching information are proposed. Results from simulation experiments demonstrate that our proposed scheme has better performance, including a higher routing success ratio, than NDN routing mechanism.

\section{1. Introduction}

Internet, running on top of Transmission Control Protocol/Internet Protocol (TCP/IP), has played a more and more important role in modern society since it was devised in the 1960s. The Internet paradigm is a host-centric model which was developed in accordance with its early usage, such as providing connectivity and sharing resource. However, with the rapid development of telecommunication technology and various networked applications, Internet usage is evolving from the host-centric model to a content-oriented model, and things that people pay attention to are shifting from “where” they can get information to “what” that information actually is. To follow these evolutions, Internet has become more and more complex. For example, peer-to-peer (P2P) technologies and content delivery networks (CDN) have been developed to enable all interested users to access content as efficiently as possible (Passarella, 2012). But they place a lot of pressure on the current TCP/IP model which makes it difficult for Internet to offer the best performance to end-users. Moreover, according to the Cisco report, the number of Internet-connected devices is expected to increase twofold from 22.9 billion in 2016 to 50 billion by 2020, and the global IP traffic is expected to increase nearly threefold from 1.22ZB in 2016 to 3.32ZB by 2021 per year (Cisco, 2017). The prominence of Big Data aggravates the already heavy burden on Internet. Plenty of research communities have been motivated to develop the Information Centric Networking (ICN) paradigm, which has emerged as a promising candidate for the future Internet (Ahlgren et al., 2012). By naming information at the network layer, ICN favors the deployment of in-network cache, thus facilitating the efficient information delivery to end-users. Moreover, some challenges in the current Internet, for example multicast, seem to have been addressed by ICN because one content can be provided by multiple different providers, that is, one entry may contain multiple outgoing faces. The ICN model thus supports multicast by default (Vasilakos et al., 2015). Thus, ICN has become a crucial research focus in future Internet studies.

Nowadays, the research on future Internet attracts large numbers of researchers. As shown in Fig. 1, in the United States, NSF funded five
projects, namely Named Data Networking (NDN), MobilityFirst, eXpressive Internet Architecture (XIA), NEBULA and ChoiceNet under its FIA in 2010, and the former three projects entered the next phrase in 2014. In European Union, FP7 and H-2020 funded some future Internet research, such as Publish-Subscribe Internet Technology (PURSUIT) between 2007-2013 and 2014-2020 respectively. In the above research, NDN has roots in CCN and changes the basic network service semantics from “delivering packet to a given destination” to “retrieving data with a given name”. It reserves the hourglass model of TCP/IP and reshapes Internet architecture by taking “named data” not IP in the thin twist (Jacobson et al., 2009). Therefore, we take NDN as the basic architecture used in this paper.

Although ICN has made significant progress, there are still some problems, such as routing, caching, and Quality of Service (QoS), which need to be addressed (Xylomenos et al., 2014). A conversion from being IP in the thin twist (Jacobson et al., 2009). Therefore, we take NDN as the basic architecture used in this paper.

There are only two kinds of packets in NDN, namely Interest and Data. An Interest packet carrying the name of the desired data is sent by a source node, and a Data packet with the desired data can be returned by any node which holds the corresponding content in network. If the source node cannot receive any Data packet from the same interface where it forwards the Interest packet within a time period, it would get a timeout. Thus, a node should be aware of content distribution, otherwise it can only resort to blind forwarding with network load increasing and performance harmed. Therefore, a key issue of routing mechanism in NDN is how to know which nodes should be preferred when Interest packets are forwarded.

The rapid development of Internet, especially the emergence of web 2.0, changes the interactive way in modern society and strengthens social relationships among people. On one hand, human relationships in real world bring out plenty of online social networking services, such as Facebook and Twitter. On the other hand, virtual relationship in social network and P2P sharing contents and interacting with each other (Yu et al., 2015). In NDN, the friend relationships mainly manifest in interest and content. For example, the probability of two nodes with similar interest and content satisfying the requirements of each other is higher than that of two nodes without any similarity. It should be noted that the ability of a node is considered an important factor in building social relationships. For example, a node with more content, stronger processing capability and higher connection degrees, just like the node with high between-ness in scale-free networks (Wu et al., 2012), would have a higher probability to satisfy the requirements from other nodes. Therefore, we exploit and apply neighbor relationship and friend relationship in network communication.

Meanwhile, in-network caching is an important characteristic of NDN. Caching received content makes a node respond to the subsequent Interest packets directly, thus the routing path can be shorten and transmission efficiency can be improved. Because the cache space of a node is limited, cache-everything-everywhere is unpractical, and thus being cache-aware makes Interest packets be forwarded to more appropriate nodes and routing performance improved (Zhang et al., 2015).

With the development of a variety of new networked applications, especially multimedia based, and the diversified demands of users, QoS support is more and more important to networks (Meddeb, 2010). For example, a bulk data transfer has the strict requirement on bandwidth and transfer reliability, while a video conference has the strict requirement on bandwidth, delay and delay jitter. Thus, in this paper, our proposed routing scheme considers networked application QoS requirements, specifically on bandwidth, delay, delay jitter, and error rate.

Based on the above discussion, we propose a Cache-aware Social-based QoS Routing scheme (CSQR) for ICN. The contributions of our work are summarized as follows:

1. The CSQR not only exploits social relationships to forward Interest packets, but also is cache-aware and considers QoS constraints of network applications. To achieve the above functions, our routing scheme is built on modules, and each module which can be added or removed flexibly is responsible for the above different function. Thus, CSQR can be integrated into NDN and compatible with current routing mechanism easily.

2. Three kinds of relationships, namely neighbor (NB), interest friends (IF) and response friends (RF) are built. They are based on node proximity, interest similarity, and node capability respectively, which jointly provide the most appropriate forwarding for Interest packets.

3. A content popularity is proposed to denote the probability of a content to be requested later. Moreover, a caching policy and its corresponding replacement policy based on content popularity and neighbor cache information are presented. It makes node cache content reasonably and achieves a tradeoff between content distribution and node space.

The rest of this paper is organized as follows. The related work is reviewed in Section 2. The framework, model and problem formulation are introduced in Section 3. The algorithm description is given in Section 4. Simulation study is presented in Section 5. Conclusion is drawn in Section 6.
2. Related work

Recently, with the rapid development and huge success of social networks, researchers have begun to utilize social relationships among nodes to design applicable routing algorithms (Wei et al., 2014). For example, in Rothfus et al. (2013), a novel routing algorithm which uses similarity metrics from data mining in node contact history is proposed. In Li et al. (2013), social energy, a new metric which is generated via node encounters and shared by the communities of encountering nodes, is introduced to quantify the ability of a node to forward packets to others. In Vendramin et al. (2016), a routing approach based on cultural algorithms, ant colony optimization and operational metrics is proposed. In Wahab et al. (2018), an approach by keeping tagging or nominating intermediate nodes achieving online social network routing is proposed. Most of the existing researches aim at mobile social networks and delay-tolerant networks. There is no related research of applying social relationship to design routing mechanism in ICN. Moreover, the above researches only focus on one single kind of relationships. In fact, the content-centric nature of ICN involves various kinds of relationships, such as geographic position, cached content and interest request. These relationships can help forward Interest packets to some appropriate nodes efficiently with routing performance improved significantly.

To alleviate the pressure from rapidly growing traffic imposing on network bandwidth, ICN provides transparent and ubiquitous in-network caching to speed up content distribution and improve network resource utilization (Abdulahi et al., 2015). It attracts wide attention and takes significant progress in terms of caching policies, and a lot of researchers have begun to exploit caching to develop effective routing mechanism (Zhang et al., 2013). For example, in Eum et al. (2012), a new ICN architecture which contains potential based routing and topology aware content caching policy is proposed. In Lee et al. (2013), a cache capacity-aware CCN consisting of selective caching and cache-aware routing methods that interact with each other to encompass cache management and cache-aware request forwarding is proposed. It utilizes network caches evenly and redirects a content request based on historic forwarding of the desired contents. In Carofiglio et al. (2016), a new approach combining a latency-proportional probabilistic caching policy and a load-aware dynamic forwarding strategy is proposed to reduce end-user experienced latency. Compared with the above researches, our proposed scheme is based on a concept of content popularity which denotes the probability of a content to be requested later. Moreover, a corresponding cache replacement policy based on content popularity and neighbor cache information is also proposed.

Being a NP-hard problem, QoS routing with multiple additive constraints is one of the key challenges in networking. For example, in Wahab et al. (2013), a QoS-based clustering algorithm which considers tradeoff between QoS requirements and high speed mobility constraints is presented for vehicular ad hoc networks. The routing mechanism in Internet mainly use intelligence algorithms or heuristic algorithms to solve complex QoS problems. For example, in Leela et al. (2011), a heuristic multi-constraint QoS unicast routing based on genetic algorithm is proposed. In Wang et al. (2011), a new multicast tree generation method which applies ant colony optimization to control tree growth is proposed. In Yang and Guan (2013), a QoS routing algorithm based on the nonlinear path length measurement, the shortest path, and the principles of non-dominated paths is proposed. With the prompt development and widespread attention of multimedia based networked applications, QoS will play an increasingly significant role in the future Internet. For example, in Hasan and Al-Turjman (2017), a bio-inspired particle multi-swarm optimization routing algorithm is proposed for social Internet of Things. To the best of our knowledge, there are very few QoS routing mechanisms in ICN, although ICN offers native content identification that can be exploited to develop a common elegant framework for supporting QoS based delivery (Al-Nady et al., 2014). This paper is the first work on cache-aware social-based QoS routing scheme in ICN.

3. System framework and model

In this section, we describe a framework of system, a network model proposed for NDN, and the request model dealt with in this paper.

3.1. System framework

To be integrated into NDN easily and be compatible with current NDN routing mechanism (NDNR) (Yi et al., 2012), the proposed routing scheme is designed with modularization. It consists of five modules, namely Caching Management Module (CMM), Forwarding Management Module (FMM), QoS Evaluation Module (QEM), Social Relationship Module (SRM) and Routing Decision Module (RDM), as shown in Fig. 2. CMM is responsible for managing cache, caching contents and executing cache replacement policy. FMM is responsible for forwarding packets and dealing with Pending Interest Table (PIT) and Forwarding Information Base (FIB). QEM is responsible for evaluating QoS of routing paths taken by Interest packets and Data packets for different networked applications. SRM is used to build and maintain social relationships. RDM is responsible for making routing decision based on SRM and QEM. When there is no entry about the received Interest packet in PIT, RDM forwards it to some appropriate nodes by social relationships, namely NB, BF, and IF, before doing FIB scheme.

These modules can be added or removed flexibly. When necessary, QEM, RDM and SRM can be turned off separately or overall. If they are all turned off, and then CSQR would only do the basic caching and forwarding schemes based on PIT and FIB, as NDNR.

The workflow of processing a received Interest packet is shown in Fig. 3(a) and described as follows. Firstly, CSQR updates the social relationship by SRM if needed, executes CMM to see whether there is matched content in its Content Store (CS), and checks the corresponding QoS requirements (in terms of available bandwidth, delay, delay jitter, and error rate) of different network applications are satisfied by the routing path from the source node to the current node by QEM (Wang et al., 2018). If yes, a Data packet can be sent back to the incoming interface of the Interest packet; otherwise, it takes the second step, that is, it checks whether the name of this Interest packet is in PIT already. If yes, it adds the incoming interface of this Interest packet to the existing PIT entry; otherwise, it takes the third step, that is, it makes routing decision based on QoS evaluation (QEM) and social relationships (SRM). If there are appropriate nodes with social relationships, the Interest packet is forwarded; otherwise, it takes the last step, that is, it checks FIB to do the longest prefix match. The workflow of processing a received Data packet is shown in Fig. 3(b). It firstly updates the social relationship by SRM if needed, and checks PIT to see whether there are some matched entries. If yes, it executes caching policy, forwards the received Data packet to those interfaces from which the corresponding Interest packets were received, and removes the corresponding entries from PIT; otherwise, it drops the Data packets. The detailed algorithm description are in Section 4.
3.2. Network models

NDN can be modeled as a connected graph $G = (V; E)$, where $V = \{v_i|1 \leq i \leq N\}$ is the set of nodes, $E = \{e_{ij} | v_i, v_j \in V, 1 \leq i \leq j \leq N, i \neq j\}$ is the set of links, $N$ is the total number of nodes in $G$, and $N = |V|$. We consider available bandwidth, delay, delay jitter and error rate for each link, and delay, delay jitter and error rate for each node. For simplicity, delay, delay jitter and error rate of a node are integrated into the corresponding parameters of the nodes’ downstream link along the route respectively. For example, delay of node $v_i$ $d(v_i)$ is integrated into delay of its downstream link $d(e_{ij})$, and it would help simplify the calculation of QoS evaluation in Section 4.1.

Referring to the router structure in IP networks (Tucker, 2008) and NDN, we devise a router structure in ICN, as shown in Fig. 4. There are four tables, namely Content Type (CT) table, NB table, IF table and RF table, in cache module. The CT table is used to maintain information about the number of sending and forwarding Interest packets in each content type, and the number of getting content from other nodes and the size of the got contents, and the latter three tables are used to maintain three different kinds of social relationships respectively. The detailed table structures are described in Section 4.2. It is worth noting, we take hop-by-hop fragmentation and reassembly which is consistent with the data-centric and session-less nature of NDN (Afanasyev et al., 2015). Moreover, this can be performed by a shim layer below the network layer when required, which means that the network layer is unaware of and does not deal with fragments at all.

To support the routing scheme proposed in this paper, referring to the basic packet format in NDN, we devise the structure of an Interest packet and a Data packet shown in Fig. 5(a) and (b) respectively.

Fig. 3. The workflow.

Fig. 4. The structure of a router in NDN.

3.3. Request models

Based on the DiffServ model (Grossman, 2002) and referring to ITU-T Y.1541 (ITU-T 2011), we define $ITS = \{IT_u|1 \leq u \leq |ITS|\}$ is the set of application types where $IT_u$ is an application type with QoS requirement ($QoS_u$) and $|ITS|$ is the number of application types, such as High-quality audio, File transfer, and Video on demand, supported by ICN. Specifically, four QoS parameters, namely bandwidth, delay, delay jitter, and error rate are considered, thus $QoS_u = \{\Delta bw_u, \Delta dl_u, \Delta jtu_u, \Delta er_u\}$, where $\Delta bw_u = [bw_{u, min}, bw_{u, max}]$, $\Delta dl_u = [dl_{u, min}, dl_{u, max}]$, $\Delta jtu_u = [jtu_{u, min}, jtu_{u, max}]$ and $\Delta er_u = \Delta er_u$ represent the requirement intervals of bandwidth, delay, delay jitter, and error rate respectively, permitting flexible QoS (Blanquer et al., 2005).
We use \( R = (v_i, cOn_k, l) \) to denote a routing request in ICN, where \( v_i \) is the source node, and \( cOn_k, l \) denotes the requested content. Moreover, \( cOn_k, l \) can be mapped to content type \( CCon_k \) as the application type \( IT_{app} \) and the minimum satisfaction degree \( Sat_{bw}(rs,d) \) on comprehensive QoS (see details in Section 4.1) automatically.

4. Algorithm descriptions

4.1. QoS evaluation

The actual QoS parameters of a path \( r_{sd} \) from node \( v_i \) to node \( v_d \) are calculated respectively as follows:

\[
\begin{align*}
bw(r_{sd}) &= \min(bw(e_{ij})), \quad \forall e_{ij} \in r_{sd} \\
dl(r_{sd}) &= \sum_{e_{ij} \in r_{sd}} dl(e_{ij}) \\
jt(r_{sd}) &= \sum_{e_{ij} \in r_{sd}} jt(e_{ij}) \\
er(r_{sd}) &= 1 - \prod_{e_{ij} \in r_{sd}} (1 - er(e_{ij}))
\end{align*}
\]

where \( bw(r_{sd}), dl(r_{sd}), jt(r_{sd}), er(r_{sd}) \) and \( bw(e_{ij}), dl(e_{ij}), jt(e_{ij}), er(e_{ij}) \) are the available bandwidth, delay, delay jitter and error rate of \( r_{sd} \) and \( e_{ij} \) respectively.

To better evaluate the QoS along the routing paths for different networked applications, we introduce the concept of user satisfaction degree. Specially, in terms of bandwidth, the user expects to get the largest value, and thus the user satisfaction degree on his actually received bandwidth \( bw(r_{sd}) \) is defined as follows.

\[
Sat_{bw}(r_{sd}) = \begin{cases} 
0 & \text{if } bw(r_{sd}) < bw_{min} \\
\frac{bw_{max} - bw_{min}}{bw_{min} - bw_{max}} + (1 - \frac{bw_{max} - bw_{min}}{bw_{min} - bw_{max}}) \frac{bw_{min} - bw(r_{sd})}{bw_{min} - bw_{max}} & \text{if } bw_{min} \leq bw(r_{sd}) < bw_{max} \\
1 & \text{if } bw(r_{sd}) \geq bw_{max}
\end{cases}
\]

where \( bw_{min} \) is a very small positive number and denotes the satisfaction degree when \( bw(r_{sd}) \) is equal to the lower bound of the corresponding bandwidth requirement. With the increment of \( bw(r_{sd}), Sat_{bw}(r_{sd}) \) also gradually increases. When it is greater than the upper bound of the corresponding bandwidth requirement, \( Sat_{bw}(r_{sd}) = 1 \).

In terms of the delay, the user expects to get the least value, and the user satisfaction degree on his actually received delay is defined as follows.

\[
Sat_{dl}(r_{sd}) = \begin{cases} 
1 & \text{if } dl(r_{sd}) \leq dl_{min} \\
\frac{dl_{max} - dl_{min}}{dl_{min} - dl_{max}} + (1 - \frac{dl_{max} - dl_{min}}{dl_{min} - dl_{max}}) \frac{dl_{min} - dl(r_{sd})}{dl_{min} - dl_{max}} & \text{if } dl_{min} \leq dl(r_{sd}) < dl_{max} \\
0 & \text{if } dl(r_{sd}) \geq dl_{max}
\end{cases}
\]

where \( dl_{min} \) is a very small positive number and has the similar meaning as \( bw_{min} \) in Eq. (5). With the decrease of \( dl(r_{sd}), Sat_{dl}(r_{sd}) \) gradually increases.

Similarly, as for the delay, we can get the user satisfaction degree on delay jitter \( Sat_{j}(r_{sd}) \) and that on error rate \( Sat_{er}(r_{sd}) \) respectively. By integrating the satisfaction degrees on the above four QoS parameters, we get user satisfaction degree on comprehensive QoS as follows.

\[
Sat(r_{sd}) = \alpha_{bw} \times Sat_{bw}(r_{sd}) + \alpha_{dl} \times Sat_{dl}(r_{sd}) + \alpha_{j} \times Sat_{j}(r_{sd}) + \alpha_{er} \times Sat_{er}(r_{sd})
\]

To reflect the probability of one node getting the required Data packets from other nodes, we introduce the concept of closeness degree. The closeness degree between two IBs is based on sending and
forwarding similarity and physical distance since $IF$ reflects friend relationship in content, and the closeness degree between two $RF$s is based on node capability, connection degree and physical distance since $RF$ reflects friend relationship in node ability.

1) $NB$ descriptions: $NB$ means that two nodes are adjacent, and one node has the required content of the other. When receiving a Data packet, node $v_i$ executes the proposed caching policy (see details in Section 4.3) based on neighbor cache information, and updates $NB$ table, as shown in Table 1, where $S_{con,v_i}(v_j)$ is the neighbor set of node $v_i$ in content $con_{k,m_i}$. Then node $v_i$ adds its own cache information (node ID, cache state) about the content into this Data packet ($Precursor$ $Nodes$ $Info$) and forwards it through PIT.

2) $IF$ descriptions: $IF$ means that two nodes are interested in the same content type. When receiving an Interest packet, node $v_i$ calculates the closeness degree between each precursor node and itself in content type of the Interest packet to determine whether there is a match between them. If yes, node $v_i$ adds the corresponding precursor nodes into $IF$ table, as shown in Table 2. Then node $v_i$ adds its own information, namely $N_{con,v_i}(v_j)$, $N_{con,v_i}(v_j)$, and $N_f(v_i)$, into the Interest packet and forwards it as described in Algorithm 3.

Without loss of generality, we assume that node $v_i$ is a precursor node of node $v_j$, if there is a match between node $v_i$ and $v_j$ in an Interest packet’s content type $con_{k,m_i}$, that is, two nodes both are interested in content type $con_{k,m_i}$. The larger the value of $N_{con,v_i}(v_j)$ and $N_f(v_i)$, the higher the probability of node $v_i$ having contents in $con_{k,m_i}$, and the higher the probability of $v_j$ getting contents in $con_{k,m_i}$ from $v_i$ which was taken as an $IF$. Thus, the closeness degree of node $v_i$ and node $v_j$ as $IF$s in content type $con_{k,m_i}$ is defined as follows:

$$d(v_i, v_j) = \beta \times \frac{N_{con,v_j}(v_i) + N_{con,v_i}(v_j)}{N(v_i) + N(v_j)} + (1 - \beta) \times \frac{N_{con,v_j}(v_i) + N_{con,v_i}(v_j)}{N(v_i) + N(v_j)}$$

where $S_{con,v_j}(v_i)$ denotes the ratio of the number of Interest packets sent by node $v_i$ and $v_j$. Similarly, $S_{con,v_i}(v_j)$ denotes the ratio of the number of Interest packets forwarded by node $v_i$ and $v_j$ in $con_{k,m_i}$ to the number of all Interest packets forwarded by node $v_i$ and $v_j$. The higher the ratio, the higher the probability of node $v_i$ and $v_j$ having contents in $con_{k,m_i}$. $\beta$ is a weight and represents the relative importance of sending and forwarding Interest packets. $d(v_i, v_j)$ is the distance between node $v_i$ and $v_j$ in terms of hop count. Obviously, $d(v_i, v_j)$ increases with the increment of the number of Interest packets sent and forwarded in $con_{k,m_i}$, and the decrement of $d(v_i, v_j)$. If $d(v_i, v_j)$ is larger than some threshold, there is a match between node $v_i$ and $v_j$ in content type $con_{k,m_i}$.

3) $RF$ descriptions: $RF$ is a kind of one-way profit oriented relationship, that is, node $v_i$ is an $RF$ of $v_j$ only means that $v_i$ has more contents, stronger processing capability and higher connection degree, and thus there is a high probability of node $v_j$ getting the required content from node $v_i$.

When receiving a Data packet, node $v_i$ records the source node of this Data packet and those precursor nodes which cached this Data packet, and updates $RF$ table, as shown in Table 3. The closeness degree of node $v_i$ as a $RF$ of $v_j$ in content type $con_{k,m_i}$ is defined as follows:

$$RF_{con,v_j}(v_i, v_j) = \frac{N_{con,v_i}(v_j)}{\sum_{k=1}^{K} \frac{N_f(v_i)}{N_f(v_k)}} \times \frac{N_{con,v_i}(v_j)}{\sum_{k=1}^{K} \frac{N_f(v_i)}{N_f(v_k)}} \times \frac{D_i(v_j)}{D_i(v_k)}$$

where $N_{con,v_i}(v_j)$ is the number of contents of node $v_j$ getting from node $v_i$ in content type $con_{k,m_i}$. $S_{con,v_i}(v_j)$ is the sum of all contents size of node $v_i$ getting from node $v_j$ in $con_{k,m_i}$, and $D_i(v_k)$ is the connection degree of node $v_j$. Obviously, the higher the ratio of contents in content type $con_{k,m_i}$, the larger the gotten contents size, the larger the connection degree of node $v_j$, the smaller the distance, and the higher $RF_{con,v_i}(v_j, v_j)$, that is, the higher the probability of node $v_i$ getting contents in $con_{k,m_i}$ from $v_j$.

To maintain social relationships, each node maintains a $CT$ table, as

<table>
<thead>
<tr>
<th>Table 1</th>
<th>The NB table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content type</td>
<td>Content name</td>
</tr>
<tr>
<td>CCon_1</td>
<td>con_1,1</td>
</tr>
<tr>
<td>CCon_2</td>
<td>con_2,1</td>
</tr>
<tr>
<td>CCon_k</td>
<td>con_k,1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2</th>
<th>The IF table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content type</td>
<td>IF</td>
</tr>
<tr>
<td>CCon_1</td>
<td>$v_i$</td>
</tr>
<tr>
<td>CCon_2</td>
<td>$v_i$</td>
</tr>
<tr>
<td>CCon_k</td>
<td>$v_i$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3</th>
<th>The RF table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content type</td>
<td>RF</td>
</tr>
<tr>
<td>CCon_1</td>
<td>$v_i$</td>
</tr>
<tr>
<td>CCon_2</td>
<td>$v_i$</td>
</tr>
<tr>
<td>CCon_k</td>
<td>$v_i$</td>
</tr>
</tbody>
</table>
shown in Table 4. It records the number of sending and forwarding Interest packets, the number of getting content from other nodes, and the size of the gotten contents in each content type.

4.3. Caching

As an important characteristic of NDN, caching makes node respond to the subsequent Interest packets directly and achieves effective content distribution. However, the limited CS necessitates designing an appropriate caching and its corresponding cache replacement policy. In particular, the concept of content popularity which is used to measure whether a content should be cached is introduced.

1) Content popularity: Obviously, a content with high probability of being requested later should be cached in the limited CS with high priority. Similarly, a content with low probability of being requested later should be replaced with high possibility when CS is full. Thus, we introduce the concept of popularity to measure the probability of a content being requested later. To reflect content popularity varying with time and space, we take the ratio of the number of received Interest packets of content type $CCon_k$ to the number of all received Interest packets in a time period measured by a node as the popularity of content type $CCon_k$ in this time period measured by a node, and denote it as $N_{CCon_k}^{l-2}(v_i)$. Thus, the popularity variable of content $con_{k,l}$ in current time period got by node $v_i$ is defined as follows.

$$\Delta CP_{con_{k,l}}(v_i) = \begin{cases} N_{CCon_k}^{l-2}(v_i) / N_{(v_i)}� & \text{if node } v_i \text{ receives content } con_{k,l} \\ N_{CCon_k}^{l-2}(v_i) / N_{(v_i)} & \text{otherwise} \end{cases} \tag{10}$$

where $N_{CCon_k}^{l-2}(v_i)$ denotes the number of Interest packets of content type $CCon_k$ received by node $v_i$ and $N_{v_i}(v_i)$ denotes the number of all Interest packets received by node $v_i$. $\gamma$ is used to strengthen the popularity when node $v_i$ receives the corresponding content and is set to 0.5.

To update popularity timely, we apply regular update mechanism. At the end of each time period, the node update content popularity as follows.

$$CP_{con_{k,l}}^{l-1}(v_i) = (1 - \delta) \times CP_{con_{k,l}}^{l-2}(v_i) + \delta \times \Delta CP_{con_{k,l}}^{l-1}(v_i) \tag{11}$$

where $CP_{con_{k,l}}^{l-2}(v_i)$ and $CP_{con_{k,l}}^{l-1}(v_i)$ denote the popularity of content $con_{k,l}$ got by node $v_i$ in the last and current time period respectively. $\delta$ is an update parameter, $0 \leq \delta \leq 1$, and the bigger $\delta$ is, the more the proportion of popularity in current node period is.

Apparently, the bigger the number of content $con_{k,l}$ being visited, and the larger its popularity $CP_{con_{k,l}}$. We divide the content popularity into three levels: popular content (PC), average content (AC), and unpopular content (UC). Thus, two threshold values, $CP^{min}$ and $CP^{max}$, are introduced to bound three different levels. The mapping function is defined as Eq. (12).

$$f_{con_{k,l}}(v_i) = \begin{cases} PC & CP_{con_{k,l}}^{l-1}(v_i) \geq CP^{max} \\ AC & CP^{min} < CP_{con_{k,l}}^{l-1}(v_i) < CP^{max} \\ UC & CP_{con_{k,l}}^{l-1}(v_i) \leq CP^{min} \end{cases} \tag{12}$$

For a newly published content, its popularity is set as the average value of $CP^{min}$ and $CP^{max}$ to give unknown contents an opportunity to live.

2) Cache replacement policy: When there is no enough capacity in CS and some newly arrived contents need to be cached, an appropriate cache replacement policy should be taken. It takes some contents out of CS to accommodate the new contents. To provide enough space and reduce the number of replacements, a cache replacement policy should consider both content popularity and CS capacity, that is, old contents with low popularity and enough CS will be replaced to accommodate new contents. When there are many contents satisfying the above requirements, the least recently used (LRU) method is applied to choose the most appropriate ones to reduce the number of replacements as much as possible. The procedure of cache replacement policy is described in Algorithm 1. Its inputs are CS and RDaP (the received Data packet); its output is CS* (the new CS after cache replacement).

Algorithm 1 Cache Replacement Policy.

**Input:** CS, RDaP

**Output:** CS*

01: if (there exist some UCs which occupy more space than RDaP does) then

02: Choose a UC with LRU;
03: Replace the chosen UC with RDaP;
04: else if (there exist some ACs which occupy more space than RDaP does) then

05: Choose an AC with LRU;
06: Replace the chosen AC with RDaP;
07: else

08: Choose UCs with LRU until they occupy more space than RDaP does;
09: Replace these chosen UCs with RDaP;
10: end if

In Algorithm 1, lines 1–3, lines 4–6, and lines 7–9 are to replace an individual UC, an individual AC, and multiple UCs with RDaP respectively.

3) Caching policy: When a Data packet is received, an effective caching policy should consider content popularity, available space of CS, and NBs. PCs should be cached as far as possible and ACs should be cached when there is no copy in NBs. If there is no enough available space in CS, the cache replacement policy described in Algorithm 1...
would be executed to accommodate contents with high and average popularity. The detailed procedure of the caching policy when receiving RDaP of which content $con_{k,l}$ is described in Algorithm 2. Its inputs are CS, RDaP and NB table; its output is CS** (the new CS after caching).

Algorithm 2
Caching policy.

\begin{verbatim}
Input: CS, RDaP, NB table
Output: CS**
01: switch ($f(Con_{pos,k,l})$)
02: case PC:
03: if (there is enough available space in CS) then
04:     Cache RDaP;
05: else
06:     Execute cache replacement policy as in Algorithm 1;
07: end if
08: break;
09: case AC:
10: if (there is no copy in NBs) then
11:     if (there is enough available space in CS) then
12:         Cache RDaP;
13:     else
14:         Execute cache replacement policy as in Algorithm 1;
15: end if
16: end if
17: break;
18: case UC:
19: if (there is no copy in NBs) then
20:     if (there is enough available space in CS) then
21:         Cache RDaP;
22:     else
23:         end if
24:     break;
25: end if
26: end switch
\end{verbatim}

In Algorithm 2, line 1 is to determine the popularity of the contents in RDaP. Lines 3–8, 10–17, and 19–24 are to deal with RDaP of which content popularity is PC, AC and UC respectively.

The caching policy and corresponding replacement policy based on content popularity and neighbor cache information make node cache content reasonably. Neither does it only consider content distribution and occupy too much node space as leave-everything-everywhere does, nor does it only consider node space saving and abandon intermediate node caching, thus a reasonable tradeoff between content distribution and node space is achieved.

4.4. Routing algorithm

Because there are only two kinds of packets in NDN, namely Interest and Data, we give two algorithms of processing a received Interest packet and a received Data packet respectively.

1) Algorithm of processing a received Interest packet: Based on the Interest packet structure devised in Fig. 5(a) and according to the workflow in Fig. 3(a), after receiving an Interest packet, the current node updates its IFs, and then does CS, PIT, NB, RF, IF and FIB schemes in turn. If a result is returned, the procedure will be terminated. If the built routing path does not satisfy QoS requirements, the Interest packet would be dropped. The detailed procedure of processing a received Interest packet is shown in Algorithm 3. Its input is RlnP (the received Interest packet); its output is a Data packet, forwarding or dropping RlnP.

Algorithm 3
The procedure for processing RlnP.

\begin{verbatim}
Input: RlnP
Output: returning a Data packet, forwarding or dropping RlnP.
01: for each precursor node recorded in RlnP do
02: if it and the current node are matched in content type of RlnP then
03:     Add the precursor node as an IF on this content type;
04:     end if
05: end for
06: Calculate user satisfaction degree $Sat(r)$ on routing path along RlnP from the source node to the current node as in Eq. (7);
07: if $Sat(r) \geq Sat_{min}$ then
08: if there is matched content in CS then
09:     Response a Data packet;
10: else
11: if there is a matched entry in PIT then
12:     Add an interface to the existing PIT entry and drop RlnP;
13: else
14:     Add own content information to RlnP;
15: if there are some NBs in the content of RlnP then
16:     Forward RlnP to the NBs;
17: else if there are some RFs in content type of RlnP then
18:     Forward RlnP to the RF with the highest closeness degree;
19: else if there are some IFs in content type of RlnP then
20:     Forward RlnP to the IF with the highest closeness degree;
21: else
22:     Forward RlnP based on FIB;
23: end if
24: if there is a matched entry in PIT then
25:     Add the precursor node as an IF in content type of RlnP;
26: else
27:     Forward RlnP to the IF with the highest closeness degree;
28: end if
\end{verbatim}

Algorithm 4
The procedure for processing RDaP

\begin{verbatim}
Input: RDaP
Output: caching, forwarding or dropping RDaP.
01: Add the source node and precursor nodes which cached RDaP as RFs;
02: if the adjacent precursor node cached RDaP then
03:     Add it as a NB on the content of RDaP;
04: end if
05: if the current node is the destination of RDaP then
06:     Execute caching policy as in Algorithm 2;
07: else
08: if there is a matched entry in PIT then
09:     Execute caching policy as in Algorithm 2;
10: else
11: Add own cache state information to RDaP;
12: Forward RDaP to all corresponding interfaces in PIT entry;
13: else
14:     Remove the corresponding interface in PIT entry;
15: end if
16: end if
\end{verbatim}

In Algorithm 3, lines 1–5 are used to build IF relationships. Lines 8–25 are to do various operating schemes; among them, Line 9 is to return a Data packet, line 12 is to do PIT scheme, lines 15–20 are to do...
social relationship schemes, and line 22 is to do FIB scheme. Line 27 is to drop RDaP when the QoS requirement is not satisfied.

2) Algorithm of processing a received Data packet: Based on the Data packet structure devised in Fig. 5(b) and according to the workflow in Fig. 3(b), after receiving a Data packet, the current node updates social relationships, that is, RF and NB. If it is the source node of the corresponding Interest packet, caching operation will be done; otherwise, both caching and forwarding operation will be done. The detailed procedure of processing RDaP is shown in Algorithm 4. Its input is RDaP; its output is caching, forwarding or dropping RDaP.

In Algorithm 4, lines 1–4 are used to build RF and NB. Line 6 is to cache RDaP when the current node is the destination. Lines 8–12 are to do the operation schemes when there is a matched entry in PIT, namely executing caching policy, adding cache state information, forwarding RDaP and removing the PIT entry. Line 14 is to drop RDaP if there is no related information.

5. Simulation study

5.1. Experimental setting

We have implemented CSQR by C++ programming language on a personal computer with Intel Core i5-4570 @ 3.20 GHz Dual Core having 4 GB DDR3 RAM (Satish and Raychoudhury, 2016). The operating environment is 64-bit Windows 10 professional. We compare CSQR with NDNR which is the basic routing mechanism in NDN. When a node receives an Interest packet, NDNR firstly checks whether there is matched content in its CS. If yes, it generates a corresponding Data packet and sends back to the incoming interface of this Interest packet, otherwise it checks the name of this Interest packet against the entries in PIT. If the name already exists, which means an Interest for the same content from another node has been received and forwarded earlier, it simply records the incoming interface of this newly received Interest packet in existing PIT entry; otherwise it adds the name of this Interest packet into PIT and further forwards based on information in FIB. To evaluate our proposed scheme further, we integrate our proposed caching policy with NDNR and denote it as NDNR_CP. The simulation topology is CERNET2 of China (CERNET2 2006). Meanwhile, we choose OSPFN (Wang et al., 2012) which is named-based routing capable and currently deployed in NDN testbed as a comparison benchmark.

We use the following metrics to do performance evaluations and comparisons: Routing Success Ratio (RSR), i.e., the ratio of the number of Interest packets of which source nodes successfully got corresponding Data packets to the number of Interest packets sent by the source nodes; Average Routing Hop (ARH), i.e., the average hop count between the source node of Interest packets and the source node of corresponding Data packets; Cache Hit Ratio (CHR), i.e., the ratio of the number of Interest packets of which source nodes got the Data packets from cache to the number of Interest packets successfully received got the Data packets; Average Running Time (ART), i.e., the average time for routing schemes used from the time point of the first content request received to the time point of the last result got; Forwarding Success Ratio (FSR), i.e., the ratio of the number of Interest packets of which source node successfully got the corresponding Data packets through PIT, social relationship, and FIB to the number of Interest packets sent. Moreover, to demonstrate the effectiveness of forwarding mechanism through different schemes, we divide FSR into three parts, namely FSR-PIT, FSR-SR and FSR-FIB, which denotes FSR through PIT, social relationships and FIB respectively.

5.2. Results analysis

In Fig. 6, we compare RSR of CSQR, NDNR, NDNR_CP and OSPFN. When the number of Interest packets increases from 100 to 1000, RSR firstly increases, then decreases. Specially, NDNR, NDNR_CP and OSPFN get their maximum RSRs at 200 Interest packets, and CSQR gets its maximum RSR at 300 Interest packets. The reason is that caching policy makes nodes cache received Data packets and contributes to content distributed in networks, and thus the four mechanisms get higher RSR when there are relative more Interest packets.

However, with the increment of the number of Interest packets, the available resource becomes less and less. Moreover, nodes would take a cache replacement policy to take out some contents which makes some routing scheme fail. Thus, RSR decreases gradually with the continuous increment of the number of Interest packets.

When the number of Interest packets is small, for example 100, the social relationship is relatively scarce, and RSR of CSQR is the lowest. However, with the increment of the number of Interest packets, the social relationship builds gradually, and it leads Interest packets to some appropriate nodes and makes CSQR get the highest RSR. The cache-everything-everywhere scheme makes NDNR and OSPFN cache more content than CSQR and NDNR_CP do, which makes NDNR and OSPFN get relative poorer performance than the latter two schemes. The ranked list of FIB in OSPFN tries to points to the cache copy with the minimum hop count, thus, OSPFN gets the higher RSR than NDNR when there are relative few Interest packets, such as 300 and 400. However, with the increment of the number of Interest packets, the popular content would take up more space in network, and the regular updating policy of FIB provides improper interfaces for these non-popular content with the same name prefix. Thus, OSPFN gets the lowest RSR with the large number of Interest packets. The RSR of NDNR_CP is a little higher than that of NDNR because the former considers neighbor information when taking caching policy. It demonstrates that our caching policy is more effective than that in NDNR.

Fig. 7 shows ARH of CSQR, NDNR, NDNR_CP and OSPFN. We can...
We observe that ARH of CSQR is relatively lower than that of NDNR, NDNR_CP and OSPFN with a small number of Interest packets. As mentioned previously, when there is no matched entry about the received Interest packet in PIT, CSQR forwards it to the appropriate nodes by doing social relationship scheme before FIB scheme, and NDNR, NDNR_CP and OSPFN just do FIB scheme. In fact, social relationship can provide more effective guidance than FIB, thus CSQR gets the smallest average hop count with a small number of Interest packets. The updating policy of FIB provides OSPFN with the cache copy with the minimum hop count, and it gets the second better performance. With the increment of the number of Interest packets, caching policy makes the cached contents be distributed extensively, thus CSQR, NDNR, NDNR_CP and OSPFN all get less and less ARH. Moreover, the regular updating policy of FIB finds the nearer cache copy for those popular content when the pointing is effective. Thus, OSPFN gets the best performance when there are a large number of Interest packets.

In Fig. 8, we compare CHR of CSQR, NDNR, NDNR_CP and OSPFN. We observe that, in general, with the increase of the number of Interest packets in the beginning, the number of Data packets increases, different caching policies all make CHRs of CSQR, NDNR, NDNR_CP and OSPFN increase gradually. However, when it increases to a critical value, for example, 400 in NDNR, cache replacement policy begins to be taken, and thus CHR begins to decrease. The effective caching policy based on information of content popularity, available space of CS and NBs postpones emergence of the critical value in NDNR_CP and CSQR, which contributes the better performance. Meanwhile, the combination of social relationship and effective caching policy makes CSQR get a higher CHR than NDNR_CP. The regular updating policy of FIB provides more effective pointing for OSPFN, especially for those Interest packets with popular name, so it gets the best performance.

In Fig. 9, we compare ART of CSQR, NDNR, NDNR_CP and OSPFN. We observe that ART of CSQR is longer than that of NDNR and NDNR_CP. As mentioned previously, to generate and maintain social relationships, CSQR handles all received Interest packets and Data packets, and maintains the corresponding NB, IF and RF tables. Moreover, it takes caching policy when Data packets are received. NDNR handles all received Interest packets and Data packets with much simpler strategies, thus it consumes the least time. NDNR_CP needs to determine whether to cache content based on cache state information of neighbor nodes, and thus consumes a little more time than NDNR. Meanwhile, no building and maintaining of social relationships makes NDNR_CP need less time than CSQR. OSPFN needs too much time to take regular updating policy of FIB and gets much longer ART than three others. As shown in Fig. 9, with the increase of the number of Interest packets, the ART increases sharply due to significant increment of computations.

To demonstrate effectiveness of forwarding mechanism through different operating schemes, we compare FSR through PIT, social relationship and FIB in CSQR. As shown in Fig. 10(a), with the increase of the number of Interest packets, each node has built and accumulated more and more social relationships, and the probability of forwarding received Interest packets through appropriate social relationships becomes higher and higher, thus the contribution of social relationships to FSR becomes greater and greater. Moreover, with the increment of the number of Interest packets, the probability of a node receiving multiple Interest packets in a short time interval increases, and thus the contribution of PIT to FSR increases. Because we do PIT, social relationship and FIB scheme in turn, the increment of contribution from the former two schemes decreases that from the last one.

To demonstrate effectiveness of social relationships further, we evaluate FSR through different social relationships, namely NB, IF and RF in CSQR. As shown in Fig. 10(b), we observe that NB gets the highest FSR, IF gets the lowest one, and RF is between NB and IF. Moreover, the ratio of three different social relationships was relatively stable with different Interest packets. To exploit influence of social relationships on FSR further, we change sequence of different social relationships in CSQR. As shown in Fig. 11, we observe that, in general, with the increment of the number of Interest packets, social relationship contributes more and more, while FIB contributes less and less, and PIT makes the lowest contribution. Among three social relationships, NB and RF contribute most of FSRs. If NB is taken before RF, it would make more contribution, and vice versa. It is worthy to note that, NB reflects content, and RF and IF reflect content type. Thus, NB usually contributes the maximum FSR. On the other hand, in a transmission process, two nodes become IFs only when they are matched in content type of Interest packets, which does not mean that they cache the specific contents really, and thus the scarce quantity and uncertain caching make IF do the lowest contribution. However, the destination node always becomes a RF of the source node and the intermediate nodes in content type of Data packet for it provides the desired content, and the intermediate nodes who cache the Data packet would become the RFs of their subsequent nodes. Therefore, more quantities and actual caching make RF relationship does a much higher contribution than IF relationship.

Lastly, to test robustness and analyze influence of historical patterns, we compare the performance under router failure and terminal migration. The former means that the related information built by a router is lost, which may happen in some special cases, for example, hardware changes. Then the router would build social relationships again. The terminal migration means that a terminal moving to a new place and accesses to a new router, which would happen frequently, for example, business travel. We choose 800 Interest packets and the related problems happen at the half of experiment time. As shown in Fig. 12, the failure would make the corresponding router restart caching policy, rebuild social relationships and reroute based on new information, thus performances have a little decline. As mentioned in the second paragraph of this subsection, the sequence of RSR from high to low is CSQR, NDNR_CP, NDNR and OSPFN. Since a terminal can use
Fig. 10. Comparison of FSR through different schemes in CSQR.

(a) Comparison of FSR through PTT, social relationships and FIB in CSQR.
(b) Comparison of FSR through different social relationships in CSQR.

Fig. 11. Comparison of FSR with different sequences of social relationship nodes in CSQR.

(a) NB-RF-IF
(b) NB-II-RF
(c) RF-NB-IF
(d) RF-II-NB
(e) IF-NB-RF
(f) IF-RF-NB
social information of the new router under terminal migration, RSR of CSQR under terminal migration is a litter higher than that under router failure.

6. Conclusion

In this paper, a cache-aware social-based QoS routing scheme is proposed for ICN. It builds three kinds of social relationships, namely NB, IF and RF, and deals with the diversified QoS requirements. Moreover, a caching policy and corresponding cache replacement policy based on content popularity and neighbor cache information are presented to achieve a reasonable tradeoff between content copy and node capacity. With the development of information communication technology, real society and virtual world integrate gradually. Thus, exploitation of social relationships would help improve network performance. Compared with NDNR, CSQR mainly adds a social-based routing scheme between PIT and FIB, modifies its caching-everything-everywhere policy to caching policy based on content popularity, and evaluates the corresponding QoS requirements. Thus, CSQR is easy to be integrated into NDN and compatible with NDNR with appropriate modifications of three corresponding modules.

In the future, we plan to carry out our work from the following directions. One is to exploit social relationships deeply to describe relationships among nodes more accurately, and thus each node could have better knowledge about content information distribution and forward routing request more effectively. Another is to do research on privacy protection combined with caching, because some types of data associated with applications, such as economic data and location information, often belong to personal privacy, and thus they should be handled more carefully. The third one is to focus on energy, since current Internet consumes a significant amount of energy. The last one is to implement our proposed scheme in a prototype system in CERNET2 to test it further and make it more practical, meanwhile, we will test it in realistic simulation environment for NDN.

Acknowledgments

This work was supported by the Major International (Regional) Joint Research Project of NSFC under Grant No. 71620107003; the National Science Foundation for Distinguished Young Scholars of China under Grant No. 71325002; the MoE and ChinaMobile Joint Research Fund under Grant No. MCM20160201; the National Natural Science Foundation of China under Grant No. 61572123; the Program for Liaoning Innovative Research Term in University under Grant No. LT2016007; the PhD Science Initiation Foundation of Liaoning Province under Grant No. 2017052032; the CERNET Innovation Project under Grant No. NGII20170613; the PhD Science Initiation Foundation of Liaoning University.

References


**Dapeng Qu** received the B.S. degree and M.S. degree in applied mathematics and computer science from Central South University, Changsha, China, in 2003, and 2006 respectively, and Ph.D. degree in computer science from the Northeastern University, Shenyang, China, in 2012. He is currently an associate professor at the College of Information, Liaoning University. His research interests include computer networking and future Internet, etc.

**Xingwei Wang** received the B.S., M.S., and Ph.D. degrees in computer science from the Northeastern University, Shenyang, China, in 1989, 1992, and 1998 respectively. He is currently a Professor at the College of Computer Science and Engineering, Northeastern University. He has published over 100 research papers. His research interests include cloud computing and future Internet, etc.

**D. Qu et al.**

**Min Huang** received the B.S. degree in automatic instrument, the M.S. degree in systems engineering, and Ph.D. degree in control theory from the Northeastern University, Shenyang, China, in 1990, 1993, and 1999 respectively. She is currently a Professor at the College of Information Science and Engineering, Northeastern University. She has published over 100 research papers. Her research interests include the modeling and optimization for the logistics and supply chain system, etc.

**Keqin Li** received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 1985, and Ph.D. degree in computer science from the University of Houston, Texas, USA, in 1990. He is currently a SUNY distinguished professor of computer science in State University of New York at New Paltz. He has published over 200 journal articles, book chapters, and research papers in refereed international conference proceedings. His research interests include parallel and distributed computing and computer networking, etc.

**Sajal K Das** received the B.S. degree in computer science from Calcutta University, Kolkata, India, in 1983, the MS degree in computer science from the Indian Institute of Science, Bangalore, India, in 1984, and the PhD degree in computer science from the University of Central Florida, Orlando, in 1988. He is currently with the Department of Computer Science, Missouri University of Science and Technology. He is the author of more than 400 published papers and more than 35 invited book chapters. He is a Fellow of the IEEE.

**Sijin Wu** received the B.S. degree in computer science and technology from the Northeastern University, Shenyang, China, in 2015, and M.S. degree in computer application technology from the Northeastern University, Shenyang, China, in 2018. Her research interest is Information-Centric Networking.