The Design and Evaluation of An Information Sharing System for Human Networks

Yaxiong Zhao, *Member*, *IEEE*, and Jie Wu, *Fellow*, *IEEE*

Abstract—With fast-growing consumer demands and rapidly-developing mobile technologies, portable mobile devices are becoming a necessity of our daily lives. However, existing mobile devices rely on the wireless infrastructure to access Internet services provided by central application providers. This architecture is inefficient in many situations and also does not utilize abundant interdevice communication opportunities in many scenarios. This paper proposes the *human network* (HUNET), a network architecture that enables information sharing between mobile devices through direct interdevice communication. We design *B-SUB*, an interest-driven information sharing system for HUNETs. In B-SUB, content and user interests are described by *tags*, which are human-readable strings that are designated by users. An experiment is performed to demonstrate the effectiveness of this tag-based content description method. To facilitate efficient data dissemination, we invent the *Temporal Counting Bloom filter* (TCBF) to encode tags, which also reduces the overhead of content routing. Comprehensive theoretical analyses on the parameter tuning of B-SUB are presented and verify B-SUB's ability to work efficiently under various network conditions. We then extend B-SUB's routing scheme to provide a stronger privacy guarantee. Extensive real-world trace-driven simulations are performed to evaluate the performance of B-SUB, and the results demonstrate its efficiency and usefulness.

Index Terms—Content-based publish/subscribe, interest-driven information sharing, human network, bloom filter

1 Introduction

With fast-growing consumer demands and the rapid development of wireless technology, *mobile devices* are becoming an indispensable part of our daily lives. Existing wireless networking technologies only allow mobile devices to communicate with each other through wireless infrastructures, for example, GSM/3G/LTE, and so on. This architecture, however, is not ubiquitously applicable. First, it fails in many situations due to limited network resources. For example, in a conference room, the WiFi and cellular connection can be crippled because too many users are competing for the channel simultaneously. Second, this architecture does not take advantage of the abundant interdevice communication opportunities. Again, take the conference room scenario as an example; because of the high density of wireless devices, there can be excellent wireless connections between nearby mobile devices. Existing wireless networks are unable to utilize such communication opportunities.

As a result, this architecture fails to address novel application requirements. Nowadays, most mobile applications are for information sharing; mobile devices are increasingly becoming the end points of information consuming. Evidence is that almost all existing smart phones and tablets are integrated with vendor-supplied music/video streaming services, and social-network-based information sharing services are extremely popular on mobile devices. Given the existing architecture, however, they have to connect with central service providers, which would fail in many situations as described above. Besides, this architecture can be inefficient in many scenarios. For instance, location-based chatting is more natural to implement in a peer-to-peer manner, so that nearby users can talk to each other directly.

Recently, a new architecture of networking portable wireless devices has emerged, which is called the *delay tolerant networks* (DTNs) [1]. DTNs adopt a "store-carry-and-forward" model, which significantly expands the communication capability of mobile device. Driven by the new application demands and the limitations of the existing architecture, we envision a new type of dynamic networking service called *human networks* (HUNETs). Physically, a HUNET is composed of human-carried mobile devices, which have the same structure as DTNs. These devices use short-range wireless communication technologies, such as WiFi or BlueTooth, to communicate with each other. Functionally, HUNETs enable information sharing between users in a completely decentralized manner without the aid of an wireless communication infrastructure. A high-level illustration of this architecture is presented in Fig. 1. The figure shows a HUNET composed of four users, each of which carries a mobile device. Users share information they are interested in with nearby peers through direct inter-device wireless communication.

We present *B-SUB*, an interest-driven information sharing system for HUNETs, which stands for the *bloom-filter-based publish/SUBscribe*. B-SUB is designed for small to medium sized networks composed of dozens of devices restricted in a limited physical area where interdevice communication opportunities are abundant. Typical application scenarios are researchers inside a conference room, students inside a department building,

- *Y. Zhao is with Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043. E-mail: yaxiongzhao@google.com.*
- *J. Wu is with the Department of Computer and Information Sciences, 1805 N Broad St, Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

Manuscript received 6 Feb. 2012; revised 14 Jan. 2013; accepted 1 Feb. 2013; published online 22 Feb. 2013.
Recommended for acceptance by D. Xuan.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2012-02-0083.
Digital Object Identifier no. 10.1109/TPDS.2013.54.

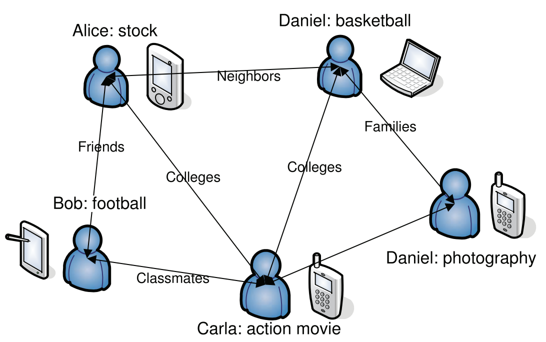


Fig. 1. A high-level illustration of the HUNET. Every user is equipped with a mobile device. Every user has his/her own interests, which is represented as a *name:interest* pair. Messages are forwarded between users via (multihop) store-carry-forward guided by the user interests.

visitors in a recreation center, and so on. The distinctive features of B-SUB are as follows: First, B-SUB employs *content-based networking* [2], [3] to achieve infrastructure-less communication. B-SUB routes and forward messages based on their content instead of addresses, which enables autonomous access to interested information for users without an end-to-end addressing mechanism. Second, B-SUB is much more efficient than traditional content-based publish/subscribe. Mobile devices have weak processors and are powered by batteries. Their computational capability is rather limited. Additionally, the memory capacity and bandwidth of the nodes in a HUNET are also scarce. Traditional content-based networking systems [4], however, are complex and consume excessive memory and bandwidth.

B-SUB employs a *tag-based content description model* and uses Bloom filters [5] to compress content and user interests. We invent the *Temporal Counting Bloom filter* (TCBF), an extension of the Bloom filter, to encode tags, which achieves efficient content routing. However, the TCBF has *false positives* in their queries, which causes useless messages to be forwarded to nodes that are not really interested in their content. We analyze, in theory, several parameters that are related to the *false positive probability* of the TCBF and their impacts on B-SUB's performance. The analysis is verified through extensive simulation studies. To summarize, our contributions in this paper are as follows:

- We propose HUNET, a novel network architecture that facilitates efficient information sharing between portable mobile devices.
- We design B-SUB, an interest-driven information sharing system for HUNETs, a content-based publish/subscribe that achieves infrastructure-less communication between mobile devices.
- We invent the TCBF, an extension to the counting Bloom filter.
- We conduct extensive theoretical analyses and real-world trace-driven simulations to evaluate the performance of B-SUB.

The remainder of this paper is organized as follows: Section 2 presents the architecture of HUNET. Section 3 introduces the TCBF. Section 4 presents the design of B-SUB. Section 5 analyzes the impact of the parameters of B-SUB on its performance. Section 6 discusses the simulation results. Section 7 summarizes the relevant work. Conclusion is given in Section 8.

2 The Architecture of HUNET

The objective of HUNET, as its name suggests, is to facilitate efficient information sharing between humans using mobile devices. A HUNET is composed of portable devices that are equipped with wireless communication interfaces, like WiFi or BlueTooth. Constrained by the relatively weak capability, these devices can only do short-range communication. Advanced wireless communication technologies, like directional antenna [6] could be used to expand the communication range. These devices are always carried and operated by human users, which gives the name of Human Network. They are collectively referred to as *nodes* in this paper. It is desirable to have non-human-operated devices to serve as "hot spots" or "offloading stations" [7] to boost the performance, but is not mandatory. In this paper, we assume that there are no such devices in HUNETs.

The processing power of HUNET nodes is weak. Additionally, because battery is the sole energy source, efficiency becomes even more crucial in HUNET, which is an important driving factor of B-SUB's design. Users join a HUNET for sharing information that they receive from others or gather from elsewhere. The most important characteristic of HUNET compared to DTNs is that HUNET exclusively relies on peer-to-peer communication to do forwarding. DTNs, on the contrary, still focus on delivering messages from a source to a destination, although there are generally no end-to-end paths connecting them.

Although HUNETs embody the same network structure as DTNs, DTN routing protocols cannot be directly applied because: 1) DTNs do not support interest-driven communication; 2) DTN routing is based on the end-to-end model, which is not applicable in HUNETs because the information source is unaware of the users who are interested in the information; 3) many existing DTN routing protocols [8] require complex offline processing to achieve optimal performance, which is prohibitive in HUNETs because they consume excessive resources and the needed data are usually impossible to get. B-SUB overcomes these problems by employing content-based publish/subscribe to facilitate infrastructure-less communication in HUNETs, and relies on users' interests to guide content routing.

Fig. 2 depicts the concept of a publish/subscribe system. Fig. 2a shows the architecture of the traditional publish/subscribe (pub/sub) systems, where a central broker connects message producers (publishers) with consumers (subscribers). Fig. 2b depicts the pub/sub system in HUNETs. A swarm of nodes form a mobile broker network. Multiple nodes serve as brokers to carry messages for users. They are distributed and are constantly changing connections due to mobility. These properties make it difficult to implement an efficient pub/sub system in HUNETs. We can still logically treat a swarm of brokers as a central broker, but this abstraction does not provide insights about the structure of HUNETs, nor does it improve the efficiency of information sharing. We take a radically different approach in designing B-SUB to address the unique requirements of HUNETs. B-SUB exploits the peer-to-peer

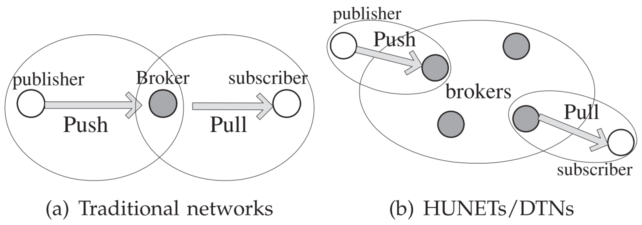


Fig. 2. High-level illustrations of the pub/sub system in traditional networks and HUNETs/DTNs. In traditional networks, A central broker connects publishers with subscribers. However, the "central broker" abstraction is no longer valid in HUNETs, or is too expensive to maintain.

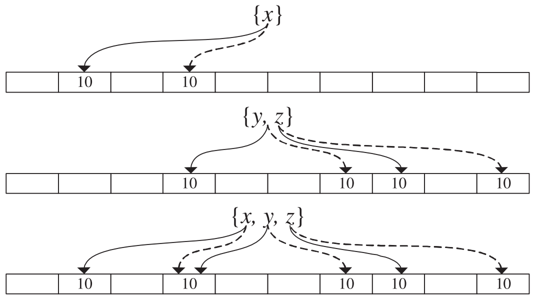


Fig. 4. Illustration of the M-merge of the TCBF. The fourth counter is the maximum of the two original TCBFs.

communication pattern in HUNETs, and lets all users exchange their interests during random contacts. Messages are then forwarded to interested users by following the trails of interest propagation. More details are presented in Section 4.

3 Temporal Counting Bloom Filter

For a preliminary of the Bloom filter and analysis on its properties, please refer to Section 1 of the supplemental file which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2013.54.

3.1 Design of the Temporal Counting Bloom Filter

The *TCBF* is an extension of the counting Bloom filter. Similar to a counting Bloom filter, a TCBF also uses a vector of counters. Insertion of the TCBF increments the associated counters of the inserted key by a fixed value $\mathbb{I}$, called the *initial counter value* (ICV), instead of 1 in the counting Bloom filter. Each time a key is inserted into a TCBF, the counters associated with the key's hashed bits will be set to the ICV. If the counter has already been set by some other keys, we do not change its value. In other words, the results of insertions are always a TCBF with multiple counters of the same value of $\mathbb{I}$.

There are two ways of merging multiple TCBFs: the *additive merging* or *A-merge* and the *maximum merging* or *M-merge*. In the A-merge, as shown in Fig. 3, the counters are set to the sum of the counters of the original filters. In the M-merge, as shown in Fig. 4, the values of the new filter's counters are set as the maximum value of the counters of the original filters. The intentions of these two merging operations will be clear after we present the design of B-SUB in Section 4. We can only insert a key into a filter that has never been merged before. To insert multiple keys into a merged filter, we first insert the keys into an empty TCBF; then, we merge the two TCBFs using A-/M-merge. Note that merging is different from union, the purpose of this design will be clear after introducing the concept of *decay*. The reasons behind forbidding the insertion from increasing counters' values, and to distinguish between A- and M-merge, are related to the semantics of content processing, which will be elaborated on in Section 4.3.

A unique operation of the TCBF is *decay*. The TCBF does not use deletion directly. It supports *temporal deletion* through *decay*. The decay of a TCBF is to constantly decrement its counters' values with a rate given by the *decay factor* (DF). Equivalently, the *decay cycle* is the time for decrementing the counters' value by 1. The following equation holds for these two parameters: $decay\ cycle = \frac{1}{decay factor}$. Tuning DFs and ICVs produces a wide range of decay granularity that are used to represent different frequencies of insertion, A-merge, and M-Merge, which is the key to controlling the performance of B-SUB. The operation is illustrated in Fig. 5, where several keys are put into the filter at different times. The final values of the filter's counters represent the frequency of the keys being inserted into the filter. After 19 units of time, the only keys left in the filter are $\{y, w\}$. If a key is not inserted into the filter frequently enough (by direct insertion or merging with another filter), then it will be removed from the filter eventually. The DF directly determines the scope of

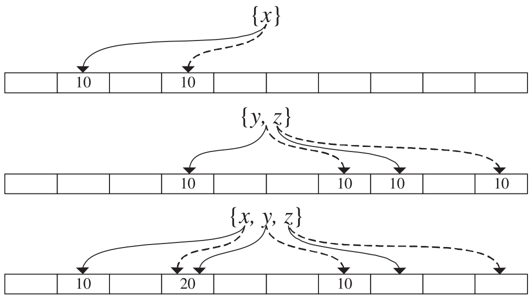


Fig. 3. Illustration of the A-merge of the TCBF. The fourth counter is the sum of the two original TCBFs.

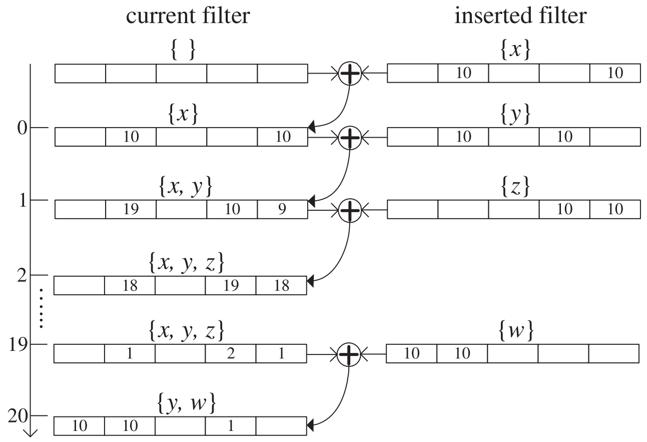
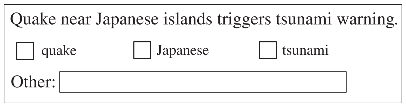


Fig. 5. An illustration of the decay operation of the TCBF. This TCBF uses a vector of five counters and two hash functions. The decay factor is set to 1/unit time.


Quake near Japanese islands triggers tsunami warning.

☐ quake ☐ Japanese ☐ tsunami

Other: ____________

Fig. 6. A sample question used in the experiment of assessing the applicability of the tag-based content description model.

the interest propagation and, in turn, the performance of the message forwarding of B-SUB. These relations are discussed in detail in Section 5.

The query that checks if a key is in a TCBF is called the *existential query* (E-query). TCBFs have the same FPR as the Bloom filters for the E-queries. A difference is that a TCBF's FPR tends to decrease with the time because keys are gradually removed by decay. Another type of query of the TCBF is called the *preferential query* (P-query). P-query is done as follows: For a key k and two TCBFs, F_i and F_j, we get the values of the counters associated with k in F_i and F_j, which are two sets, C_i and C_j. Then, we obtain the minimum values of C_i and C_j, which is denoted as c_i and c_j. We define the *preference* of F_j to F_i against k, $PREF_{i,j}(k)$, as follows:

$$PREF_{i,j}\{k\} = \begin{cases} \frac{c_j - c_i}{c_i} & \text{if } c_i \neq 0 \\ c_j & \text{if } c_i = 0 \end{cases}$$

Note that the preference is c_j when c_i equals 0. This is necessary when comparing the preference of two TCBFs F_j, F_k to F_i, where c_i is 0. The P-query indicates the ratio of the insertion/merging frequency of the queried key in two TCBFs. Its meaning in the context of the interest processing of B-SUB will be discussed in Section 4.4.

3.2 Benefits of Using the TCBF

TCBFs are used to store interests (details are given in Section 4). Basically, TCBFs map, through k hash functions $h_i(1 \leq i \leq k)$, each interest into k bits of the bit-vector. It reduces the space consumption of storing users' interests. The simulation results in Section 6 demonstrates that the TCBF uses half of the space used by the raw strings with an acceptable FPR. It also reduces the bandwidth consumption of the interest propagation. When two interests, I_0 and I_1, of a user cannot be forwarded at the same time due to the bandwidth restriction in the contact, without the TCBF, only one of them gets the opportunity to be served. If we can compress both I_0 and I_1 to fit the bandwidth, both interests might eventually be served. The TCBF does just that: in any situation, a bit-vector of a fixed length is enough to store multiple interests. Besides, content matching using TCBF is also more efficient than string matching.

Using the TCBF's decay operation, we are able to more accurately measure the entering/exiting of a particular interest through the addition/subtraction of counters. The challenge is to set appropriate parameters so that the gain (by compressing more interests) outweighs the cost of handling false positives. An in-depth study is needed on various tradeoffs in selecting parameters, such as the number of interests that can be stored, and the length of the bit-vector.

4 B-SUB Design

4.1 Overview

B-SUB has two components: *content representation* and *pub/sub routing*. B-SUB employs the *tag-based content description model*. The contents of messages and the interests of users are identified by *tags*, which are strings that summarize the topics of the message. They are stored in TCBFs, which are then used as probabilistic hints for forwarding messages. The pub/sub routing provisions two functions: *interest propagation* and *message forwarding*. Both rely on the TCBF to achieve low storage and computational complexities. B-SUB limits the size of messages to a few more than 100 bytes. Large volume content distribution is surly desirable, but is difficult to provision given the exiting infrastructure. It is also true in existing social networking applications that users tend to publish many small-sized messages. For example, Twitter [9], a popular microblogging application, limits the maximum size of each post to 140 bytes.

4.2 Tag-Based Content Description

The tag-based content description model is used in B-SUB. To justify its effectiveness and applicability, we conducted an experiment: users are asked to choose appropriate tags to summarize the content of the given news titles. In this experiment, 10 news titles are extracted from the websites of major news agencies during the last week of December 2010. Fig. 6 shows a sample question.

As depicted in Fig. 6, given a news title, different users may choose distinct tags. In order for the tag-based approach to work, different users shall have a common view, i.e., same tags, for the same message, i.e., a news title in this experiment. We define *consensus* to measure a user's ability to find the common tags. It is calculated as follows: for the tag that is chosen by the most users, denote N as the number of users who chose that tag. The ratio of N to the total number of users is the value of the consensus of this news title. The higher the consensus of a message, the more likely its tags match those of other users who are interested in the message. A consensus of 1 indicates that all users have at least one common tag that describes the content of the news title. For example, if a message has a very low consensus, this means that every user has a different tag for it. If it is labeled as "blue" by user A, and "red" by user B, and user B is interested in "red," then user A should send the message to user B because its label applied by B is "red," which is the same as his/her own interest. However, user A will not forward the message because user A applies a different tag to the message and determines that user B is not interested in it.

Fig. 7 shows the consensus of 10 news titles, all of which were collected from 20 people randomly selected from the authors' email contact lists. In the figure, most titles have high consensus values that are larger than 0.86. Only one has a relatively low value of 0.43. Judging from the fact that the targeted application scenario of B-SUB is to have casual chats about popular topics, the consensus should be better than this experiment; alas, we deem that it suffices to describe contents and interests with the tag-based content description model.

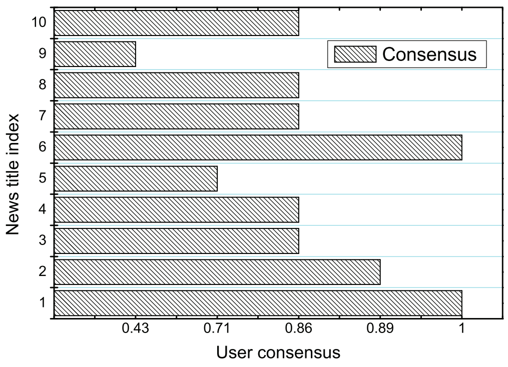


Fig. 7. The user consensus of 10 different news titles.

The conventional attribute-constraint-based content description model [10] consumes more storage and bandwidth resources than the tag-based approach, and it does not provide significant benefits, which makes it inappropriate for our purpose of designing simple and efficient protocols for HUNETs.

4.3 Interest Propagation

In B-SUB, TCBFs are used to compress users' interests. A user stores its own interests in a TCBF, which is called the *genuine filter*. A broker stores the interests collected from other users in another TCBF called the *relay filter*. TCBFs serve as a "compressed" matching hint for delivery, but they are not precise, due to the TCBF's false positives. The false positives of the TCBF causes interests being falsely matched by a message, so that the message will be forwarded to nodes that are not really interested in it. The message is sent according to the guidance of the TCBFs stored in all of the nodes in the network, which will be elaborated on in Section 4.4. Again, false positives may occur. Thus, a delivery tree, instead of a path, will be generated. Nonetheless, it is guaranteed that the original path to the subscriber is embedded in the tree.

When two nodes, say A and B meet, they first exchange the TCBFs that contain their genuine interests and relay interests. A then merges Bs genuine interests with its own relay interests using the A-merge; and merges Bs relay interests with its own relay interests using the M-merge. Note that all operations are performed on the TCBFs instead of the raw strings that are corresponding to the interests.

The intent of applying M-merge, instead of A-merge, in interest propagation between two nodes is to prevent *bogus counters*. Fig. 8 illustrates how the bogus tickets are accumulated in A-merge. Two nodes, B and C, meet frequently, but B only meets with A once-in-a-while. If we allow nodes to use A-merge to combine the filters obtained from others, the counter values of the TCBFs of two frequently-met nodes will be incremented quickly. As a result, B and C will be selected as the forwarders of the messages that A is interested in, by D and E. However, B and C are actually not desirable candidates because they do not frequently meet with A. M-merge significantly limits the impact of bogus counters. In the above example, nodes B and C will not increment their counters using M-merge. Additionally, decay takes effect during the whole process of propagating interests. It limits the range in which a user's

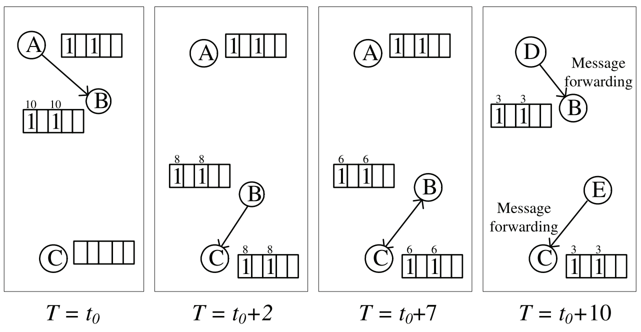


Fig. 8. Illustration of the process that bogus counters are accumulated in A-merge due to the frequent contacts between two nodes. Node B only meets A once. However, B and C obtain bogus counters from each other using A-merge because they meet frequently. Even after 10 units of time, B and C do not remove As interest. Later, B and C are selected by D and E as the forwarders for the messages that A is interested in, which are really poor candidates for forwarding.

interests can propagate and also reduces the impact of the bogus counters.

4.4 Routing and Forwarding

The decay of the TCBF removes, from the nodes' relay filters, the interests from the consumers that they meet infrequently. The preferential query is used by nodes to select forwarders for the buffer messages. The only data that is needed in the forwarding of messages is the TCBFs that encode the interests. The only operations performed are hashing and table lookup.

As presented before, when two nodes meet, they exchange their relay interests and genuine interests encoded in TCBFs. We still use A and B to denote the two nodes. At first, node A queries all of its buffered messages against the genuine filter of B, and then forward all the messages that match the filter to B. To save bandwidth, the genuine filters are represented as Bloom filters without counters.

A then examines Bs relay filter to determine which other messages should be forwarded to B. A maintains a table of the preference values of all of the buffered messages, which is called a *preference table*. For each of the messages that have not been forwarded to B, A performs a preferential query of the message's tag to the relay filter of B, and then compares the obtained preference value to the one associated with the message in the preference table. If Bs preference value is larger, the message is forwarded to B, and the preference value of the message in the preference table is updated to Bs preference value. Otherwise, the message will not be forwarded. The same operations are also performed by B.

The messages are ranked in the message buffer according to their *ages*. The age of a message is the elapsed time since it was generated. B-SUB assumes that every node has a message buffer that can hold at most a fixed number of messages. The above forwarding operations are applied for all messages in the buffer from the newest to the oldest. When the message buffer is overflowed, the oldest message will be removed. The messages' lifetimes are also limited by their TTLs. When a message's age exceeds its TTL, it should be removed from the message buffer.

4.5 Content Routing with Privacy Guarantee

In this section, we present *B-SUB-P*, an extension of B-SUB that provides stronger privacy guarantee. The attack model for this discussion is as follows:

- In our attack model, an attacker disguises as a normal user, and then joins a HUNET and engages in content routing. The attacker collects the interests of other users in doing forwarding. User interests thus are leaked to the attacker and are subjected to privacy infringement.

The basic privacy guarantee provided by the original B-SUB is called *nondirect linkage*. It means that the attacker cannot obtain direct linkage between a user's identity and his/her interests. This fact could be verified as follows: Given a TCBF that encodes the interests of a user, one cannot discover the real interests of the user because:

- The attacker cannot reverse back the hashed bit-vector to the real interests. This is provided by the security of the one-way hash functions used in the TCBF.
- The attacker can only guess the linkage between the user's identity and his/her interests. B-SUB requires that a user sends a TCBF that encodes its own interests to another user when they encounter. The attacker can guess the real interests by querying the membership of known interests against the TCBF. If an interest is a member of the Bloom filter, then it is assumed to be an interest of the owner of the Bloom filter. This attack achieves the best result when the attacker knows the universal set of interests. Fortunately, this is not true in HUNETs. First, all interests gathered by an attacker are encoded, which means that the attacker cannot directly get the universal set from the users. The attacker can gather other's interests by relaying messages; whenever a message is being delivered to another user, the attacker assumes that the content descriptors of the message are in the universal set. However, this does not establish directly linkage, because the delivery may be caused by a false positive. Without directly examining the raw strings of the user's genuine interests, the attacker cannot associate the interests with users.

The above discussion proves that in the original B-SUB, the attacker cannot know for sure a user's interests. But the attacker can still obtain a guess with computable accuracy, which are calculated based on the parameters of the TCBF using the equations in Sections 3 and 1 of the available online supplemental material. B-SUB-P removes this vulnerability by altering the basic content routing protocol of B-SUB. B-SUB-P mixes the user's own interests and relayed interests when two encountered nodes exchange interests. When two nodes meet, say node A and B, B sends a single TCBF to A instead of two as the original B-SUB does. This filter is obtained by merging Bs genuine interests and relay interests through the A-merge, which is called the *mixed relay interests*. The mixed relay interests do not disclose the linkage between node Bs identity and its interests, because A cannot distinguish between Bs own interests and its relayed interests. That is, A cannot guess Bs interests. A then performs a preferential query to determine what messages should be forwarded to B, which is the same as what the original B-SUB does. After forwarding, A merges Bs mixed relay interests with its own relay interests through the M-merge. Similarly, node B will perform the same operations. This method is thus called B-SUB-P, which stands for *B-SUB with Privacy guarantee*.

5 System Analysis

The DF is the key to adjusting B-SUB's behaviors. We study its impacts on the interest propagation and the FPR. We also analyze the storage complexity of TCBFs and present a TCBF allocation method with the optimal FPR. The details of the analyses are moved to the available online supplemental material. Please refer to the journal's website for accessing the digital copy.

6 Performance Evaluation

Real-world trace-driven simulations are used to evaluate the performance of B-SUB. We present the results of the delivery performance and the overhead. Two human contact data sets: Cambridge/haggle/Infocom06 [11] and MIT/reality [12] are used. We present the results of the former. The results of MIT/reality are in the available online supplemental material. The results indicate that the optimal value of the decay cycle heavily depends on contact patterns.

6.1 Simulation Settings

Algorithms in comparison. We compared B-SUB and B-SUB-P with two other techniques: *PUSH* and *PULL*. PUSH works like *epidemic routing* or *flooding*. A node replicates all of the messages in its message buffer to every node that it encounters without duplication. In the simulation, every message is assigned a 32-bit ID by hashing its content to help nodes detect duplicate messages. In PULL, a node only forward a message to another node that is interested in the message. We use the decay cycle as a changing parameter, which is more intuitive to understand than the decay factor.

Human/device contact trace. We use two real-world human/device contact traces: Cambridge/haggle/Infocom06 [11], denoted as Infocom06 and MIT/reality [12]. The results of the MIT/reality are presented in the available online supplemental material. These two data sets have a moderate number of nodes within a small area, i.e., 100 iMote nodes carried by conference attendees in a hotel building for Infocom06, and 97 cell phones carried by students in the school campus for MIT/Reality, which are the environment that B-SUB is designed for. All contacts of two traces are assumed to be symmetric, and all contact records involving external devices are removed. In the Infocom06 trace, we remove nearly half of the contacts to compensate the assumption of symmetric contacts. The parameters of the two traces after processing are listed in Table 1.

User interests and message tag. We prepare a tag pool of 38 tags from the twitter trends API [9]. Every node is assigned two tags as its interests, which are selected uniformly from the tag pool. Every node generates messages constantly. The interval between the generation of two successive messages is drawn evenly from

TABLE 1
Parameters of Two Data Sets

Data Set	Infocom06	MIT/Reality
Device	iMote	Cell phone
Communication method	Bluetooth	Bluetooth
Duration (days)	3	246
Number of nodes	100	97
Number of contacts	89,701	102,377

$[0, MAX\_INTERVAL]$. The $MAX\_INTERVAL$ is set to 1 hour for the Infocom06 trace and 50 hours for the MIT/Reality trace. These message generation rates produce about 180 and 100 messages for the two traces, respectively. Each message has 1 tag that is also selected from the tag pool based on the distribution in Table 2.

TCBF and message buffer settings. We use a bit-vector of 256 bits and three hash functions in the TCBF. This means that, at most, 4 bytes are needed to encode a tag. The initial counter value is set to 5. The duration of the contact is not modeled, and it is assumed that all messages can be transmitted during a single contact. To compensate for this assumption, we limit the size of the message buffer of every node to 100. Following Twitter's restriction on message size, we limit each message to 140 bytes. This amounts to at most 1.4 KB of memory used in the message buffer. The message buffer is implemented as a priority queue that ranks the messages based on their creation time. When an buffer overflow occurs, the oldest message is dropped.

6.2 Delivery Performance

We present the delivery ratio and the average delay of delivered messages in this section. In these figures, B-SUB-P shows a noticeable performance degradation, which is caused by the imprecision of the processing of the TCBFs.

Delivery ratio. Fig. 9a presents the delivery ratios of the four protocols in the Infocom06 trace. The results of PUSH and PULL are not affected by the decay cycle. B-SUB and B-SUB-P approach PUSH's performance when the decay cycle increases. Both B-SUB and B-SUB-P outperform PULL because they utilize multihop forwarding. B-SUB and B-SUB-P suffers from a performance degradation when the decay cycle grows beyond a threshold. This is mainly caused by the counter overflow of the TCBF. A larger decay cycle allows the interests to be propagated to more nodes, resulting into higher counter values in all of the nodes in the network. When the decay cycle grows beyond a certain threshold, counter overflows will occur because the counters have a fixed value range. Overflows cause many nodes to have very similar preference values, which produce low preference values when nodes are performing preferential queries. When a node that cannot reach the destination gets a message, it will not forward the message to nodes that have a better change to reach the destination, because the node does not have larger preference value. The overall effect is that messages are trapped in nodes that cannot deliver them quickly. We also notice that the variations of the results of the MIT/Reality trace is much smaller than the Infocom06 trace (please refer to the available online supplemental material for the figures). This fact pertains to the characteristics of two traces. The

TABLE 2
The Distribution of the Top 4 Keys in the Simulation (Spaces are Removed)

NewMoon	Twitter'sNew	funnybutnotcool	openwebawards
0.132	0.103	0.0887	0.0739

contact frequency of the MIT/Reality trace is much lower than that of the Infocom06 trace, so the same parameter range only explores a fraction of the entire changing spectrum. However, the results presented in Fig. 9a demonstrate the same changing patterns of B-SUB and B-SUB-P, which is sufficient for verification.

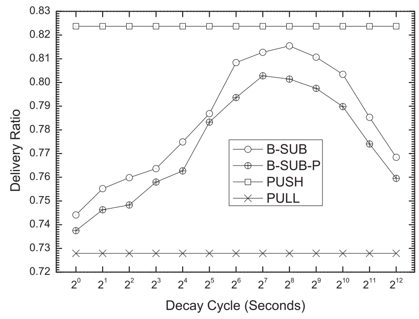
Delay. Fig. 9c shows the average delay of the delivered messages of the Infocom06 trace. The delay of a delivered message is measured as the time interval from the time when the message is generated to the time when it is delivered to a node that is interested in it. The delay behaves similarly to the delivery ratio. Initially, increasing the decay cycle expands the propagation of user interests and helps messages find more paths to reach interested nodes. However, similar to the delivery ratio, messages can be trapped when the decay cycle grows beyond a certain threshold, which results in a longer delay. From the results of Fig. 9a, the delivery ratio and delay achieves the optimal performance at a similar decay cycle, which is between 2^7 and 2^8 seconds. This means that routing messages on the paths that have the minimal delay improve the overall delivery performance.

6.3 Overhead

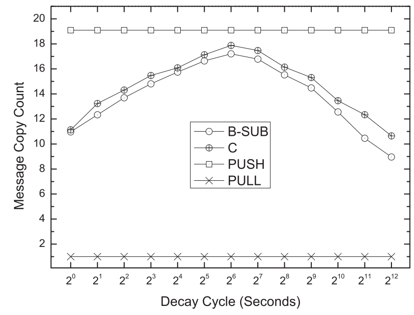
Average message copy count. Fig. 9b presents the message copy counts of the delivered messages of the Infocom06 trace. The message copy count of a delivered message is counted by finding out how many nodes (including the source and the receivers) have relayed the message, i.e., have possessed a copy of the message. PUSH has the worst results. B-SUB can maintain a relatively lower copy count than PUSH. However, B-SUB-P tends to propagate user interests in a wider range compared to B-SUB, which slightly increases message copy count. We also notice that when the delivery ratio reaches optimal, when the decay cycle is around 2^8 seconds, the copy count is about 15.5, which is 20 percent lower than PUSH. Note that dropped messages are not included as message copies.

Dropped message count. Fig. 9e presents the number of dropped messages of the four protocols of the infocom06 trace. The dropped message count is measured as the average number of messages being dropped per node in the entire simulation session. As discussed previously, dropped messages are not treated as message copies. In fact, the sum of the message copies and dropped messages is the total overhead induced by forwarding messages. Dropped messages waste precious wireless bandwidth and energy. As depicted in the figures, PUSH drops more messages than B-SUB and B-SUB-P, which is nearly two orders of magnitude more. Since PULL does not use multihop forwarding, its number of dropped message is always 0, and is not shown in the figures.

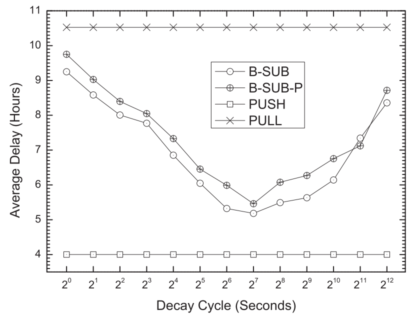
Forwarding false positive rate. Fig. 9d presents the forwarding FPR of the four methods of the infocom06 trace.



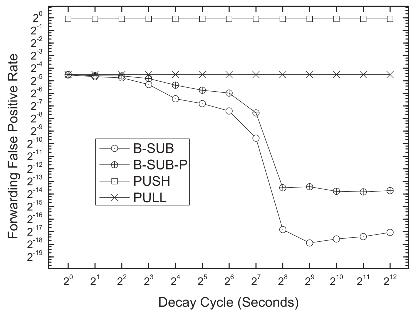
(a) Delivery ratio



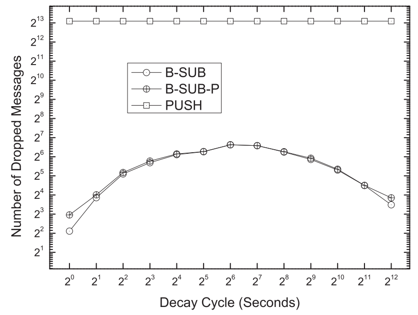
(b) Message copy count



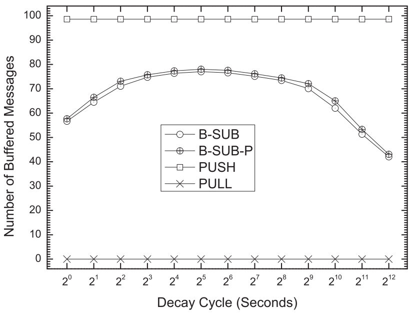
(c) Average delay



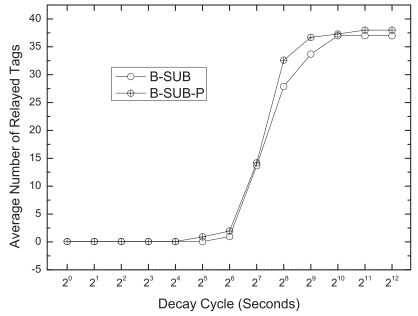
(d) False Positive Rate



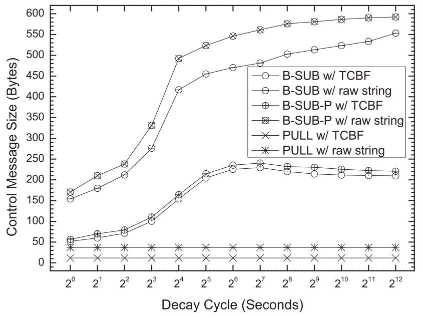
(e) Dropped Mesage



(f) Buffer Size



(g) Relayed tag count



(h) Overhead

Fig. 9. The performance metrics of B-SUB, B-SUB-P, PUSH, and PULL in the Infocom06 trace. The decay cycle in the horizontal axis is in a logarithmic scale.

Forwarding FPR is measured as the ratio of the number of falsely forwarded messages to the total number of messages being forwarded. Note that this is not the same as delivery FPR, which is the ratio of the falsely delivered messages to all of the delivered messages. Actually, the forwarding FPR of PULL is actually its delivery FPR, because PULL does not use multihop forwarding. As analyzed, its FPR is very close to the worst FPR of storing 38 keys in a Bloom filter of 256 bits and three hash functions, which is about 4 percent. When the decay cycle is small, B-SUB and B-SUB-P perform similarly to PULL. When decay cycle grows larger, the preferential forwarding of B-SUB and B-SUB-P routing reduces the forwarding FPRs. That is, when false positive happens, because of a lower preferential value, the message is prevented from being forwarded. For PUSH, we run B-SUB in the background, and the messages should be forwarded by B-SUB as the denominator. We can see that more than 90 percent of the messages being forwarded by PUSH are considered false positives by B-SUB, which wastes a lot of resources.

Relayed tag count. Fig. 9g presents the average number of relayed tags by each node at the end of the simulation session of the infocom06 trace. The relayed tag count is tracked in the entire simulation to ensure accuracy. The results are measured at the end of the simulation to make sure that it is stable. Since the decay cycle axis is in a logarithmic scale, the increasing rate of the relayed tag count is acceptable. Related to Fig. 9a, the optimal delivery ratio is achieved when there are a moderate number of relayed tags (about 30). This is intuitive: the relayed tag count should be large enough so that B-SUB and B-SUB-P can find sufficient routing paths; and it should not be so large that it causes counter overflows.

Control message size. Fig. 9h presents the average size of all of the control messages being sent in each contact of the infocom06 trace. The control messages only include the ones that are used to represent user interests. Since PUSH does not use user interests in forwarding, its control message is 0. As shown in the figure, using TCBFs reduces nearly 2/3 of the overhead of using raw strings. This results from the fact that the average tag size in the simulation is about 12 bytes, which is 3 times the worst case memory consumption of the TCBF. We can observe a slight decline of the control message size when the decay cycle grows beyond a certain threshold. This is caused by the counter overflow. The overflowed counters' values are the maximum value. When more counter overflows occur, the number of distinctive values of the TCBF is smaller, so is the size of the TCBF.

7 Related Work

DTN [1] enables various novel applications that were deemed impractical before. HUNET follows the same lineage of utilizing the "store-carry-and-forward" communication paradigm to provision a personal information sharing service. An important property of HUNETs that is different from DTNs is its inherent social network structure. The analysis of human contact patterns reveals that community structures are prevalent [13], which is used to facilitate efficient routing [13], [14], [15], [16]. In our previous work [8], [17], we proposed an optimal forwarding rule based on the optimal stopping theory and the long-term relationships between users. All of these routing protocols require a stable contact pattern between nodes, and need a complex and time-consuming preprocessing to gather routing information. However, HUNETs usually exist in short term, which makes these procedures difficult to perform in HUNETs, and makes these protocols unable to work in HUNETs. Privacy in mobile content sharing is studied in [18], [19], [20], [21]. A similar work to ours which studies the user-centric data dissemination in DTN is [22]. This work does not have a content description model.

Content-based networking is used in an interest-driven opportunistic data diffusion application [23]. Other issues in mobile content sharing, including energy efficiency, scalability, and privacy are studied in [24], [25]. TACO-DTN [26] is a pub/sub system for DTNs that utilizes stationary infrastructure to facilitate more efficient content distribution, a similar system for vehicular network is presented in [7], [6]. In [2], a content-based pub/sub system for wireless ad hoc networks is presented. The authors in [27] presented an analysis of the pub/sub systems in wireless mobile networks. Except for the content-based approach, there are topic/channel-based pub/sub systems that support only a limited number of communication channels [3]. Low energy processor units [28], [29] are an important enabling technology to HUNETs. A survey of the application of the Bloom filter in computer networks is in [30].

8 Conclusion

In this paper, we present B-SUB, an interest-driven information sharing system for HUNETs. B-SUB employs content-based networking to achieve infrastructure-less communication between mobile devices. Specifically, B-SUB employs a tag-based content description model. A novel data structure, the TCBF, is invented to compress user interests and guide content routing. The use of TCBF reduces the memory and bandwidth consumption of B-SUB. We systematically analyze the impact of several parameters of B-SUB on its behaviors and performance. An extension of B-SUB called B-SUB-P is then proposed to better protect user privacy. Extensive real-world trace-based simulations are performed to verify the performance of B-SUB and B-SUB-P. The results have proven that B-SUB and B-SUB-P archive similar delivery ratio and delay as the optimal method (PUSH), but consumes much less resources.

References

[1] K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets," *Proc. Conf. Applications, Technologies, Architectures, and Protocols for Computer Comm.*, pp. 27-34, 2003.

[2] A. Carzaniga, D.S. Rosenblum, and A.L. Wolf, "Content-Based Addressing and Routing: A General Model and its Application," 2000.

[3] P.T. Eugster, P.A. Felber, R. Guerraoui, and A.M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114-131, 2003.

[4] A. Carzaniga, D.S. Rosenblum, and A.L. Wolf, "Design and Evaluation of a Wide-Area Event Notification Service," *ACM Trans. Computer Systems*, vol. 19, pp. 332-383, Aug. 2001.

[5] B.H. Bloom, "Space/Time Trade-Offs in Hash Coding with Allowable Errors," *Comm. ACM*, vol. 13, no. 7, pp. 422-426, 1970.

[6] Y. Li, Z. Wang, D. Jin, L. Zeng, and S. Chen, "Collaborative Vehicular Content Dissemination with Directional Antennas," vol. 11, no. 4, pp. 1301-1306, Apr. 2012.

[7] Y. Li, G. Su, P. Hui, D. Jin, L. Su, and L. Zeng, "Multiple Mobile Data Offloading through Delay Tolerant Networks," *Proc. Sixth ACM Workshop Challenged Networks*, pp. 43-48, http://doi.acm.org/10.1145/2030652.2030665, 2011.

[8] C. Liu and J. Wu, "An Optimal Probabilistic Forwarding Protocol in Delay Tolerant Networks," *Proc. MobiHoc '09*, pp. 105-114, 2009.

[9] "Twitter," http://www.twitter.com. 2013.

[10] A. Carzaniga and A.L. Wolf, "Forwarding in a Content-Based Network," *Proc. SIGCOMM '03*, pp. 163-174, 2003.

[11] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD Data Set Cambridge/Haggle (v. 2009-05-29)," http://crawdad.cs.dartmouth.edu/cambridge/haggle, May 2009.

[12] N. Eagle and A.S. Pentland, "CRAWDAD Data Set Mit/Reality (v. 2005-07-01)," http://crawdad.cs.dartmouth.edu/mit/reality, Jul. 2005.

[13] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Pocket Switched Networks: Real-World Mobility and Its Consequences for Opportunistic Forwarding," *IEEE J. Selected Areas Comm.*, vol. 26, no. 5, pp. 748-760, Feb. 2005.

[14] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott, "Impact of Human Mobility on Opportunistic Forwarding Algorithms," *IEEE Trans. Mobile Computing*, vol. 6, no. 6, pp. 606-620, June 2007.

[15] C. Qian and S. Lam, "Greedy Distance Vector Routing," *Proc. 31st Int'l Conf. Distributed Computing Systems (ICDCS '11)*, pp. 857-868, June 2011.

[16] S.S. Lam and C. Qian, "Geographic Routing in D-Dimensional Spaces with Guaranteed Delivery and Low Stretch," *Proc. ACM SIGMETRICS Joint Int'l Conf. Measurement and Modeling of Computer Systems (SIGMETRICS '11)*, pp. 257-268, 2011.

[17] Y. Zhao and J. Wu, "B-Sub: A Practical Bloom-Filter-Based Publish-Subscribe System for Human Networks," *Proc. Int'l Conf. Distributed Computing Systems (ICDCS '10)*, pp. 634-643, 2010.

[18] R. Zhang, Y. Zhang, and Y. Fang, "AOS: an Anonymous Overlay System for Mobile Ad Hoc Networks," *Wireless Networks* , vol. 17, no. 4, pp. 843-859, May 2011.

[19] R. Zhang, J. Shi, Y. Zhang, and J. Sun, "Secure Cooperative Data Storage and Query Processing in Unattended Tiered Sensor Networks," *Selected Areas in Comm., IEEE J.*, vol. 30, no. 2, pp. 433-441, Feb. 2012.

[20] R. Zhang, J. Shi, and Y. Zhang, "Secure Multidimensional Range Queries in Sensor Networks," *Proc. ACM MobiHoc '09*, pp. 197-206, 2009.

[21] Y. Zhang and S. Zhong, "A Privacy-Preserving Algorithm for Distributed Training of Neural Network Ensembles," *Neural Computing and Applications*, pp. 1-14, DOI: 10.1007/s00521-012-1000-8.

[22] W. Gao and G. Cao, "User-Centric Data Dissemination in Disruption Tolerant Networks," *IEEE INFOCOM*, pp. 3119-3127, Apr. 2011.

[23] I. Carreras, D.P. Francesco, D. Miorandi, D. Tacconi, and I. Chlamtac, "Why Neighborhood Matters: Interests-Driven Opportunistic Data Diffusion Schemes," *Proc. Third ACM Workshop Challenged Networks (CHANTS '08)*, pp. 81-88, 2008.

[24] W. Fang, F. Liu, F. Yang, L. Shu, and S. Nishio, "Energy-Efficient Cooperative Communication for Data Transmission in Wireless Sensor Networks," *Consumer Electronics, IEEE Trans.*, vol. 56, no. 4, pp. 2185-2192, Nov. 2010.

[25] Y. Li, D. Jin, L. Su, and L. Zeng, "Stability and Scalability Properties for Dynamic Content Updates over Delay Tolerant Networks," *Proc. 20th Int'l Conf. Computer Comm. and Networks (ICCCN '11)*, pp. 1-6, Aug. 2011.

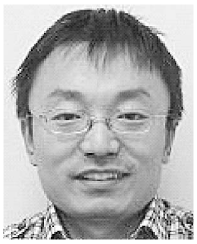
[26] G. Sollazzo, M. Musolesi, and C. Mascolo, "TACO-DTN: a Time-Aware Content-Based Dissemination System for Delay Tolerant Networks," *Proc. ACM First Int'l MobiSys Workshop Mobile Opportunistic Networking (MobiOpp '07)*, pp. 83-90, 2007.

[27] T. Pongthawornkamol, K. Nahrstedt, and G. Wang, "The Analysis of Publish/Subscribe Systems over Mobile Wireless Ad Hoc Networks," *Proc. ACM Fourth Ann. Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous '07)*, pp. 1-8, Aug. 2007.

[28] W. Hu, J. Wang, X. Gao, Y. Chen, Q. Liu, and G. Li, "Godson-3: A Scalable Multicore Risc Processor with X86 Emulation," *Micro, IEEE* , vol. 29, no. 2, pp. 17-29, Mar.-Apr. 2009.

[29] W. Hu, R. Wang, Y. Chen, B. Fan, S. Zhong, X. Gao, Z. Qi, and X. Yang, "Godson-3B: A 1 ghz 40 w 8-Core 128 gflops Processor in 65 nm cmos," *Proc. IEEE Int'l Solid-State Circuits Conf. Digest of Technical Papers (ISSCC '11)*, pp. 76-78, Feb. 2011.

[30] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Math.*, pp. 636-646, 2002.



Yaxiong Zhao received the BE degree from Tongji University, Shanghai, China, in 2005, and the ME degree from Tsinghua University, Beijing, China, in 2008. He is working toward the PhD degree in the Department of Computer and Information Sciences, Temple University. His research interests include distributed computing systems and computer networks. He has published eight first-author papers on wireless sensor networks, delay tolerant networks, and content-based networks. He is a member of the IEEE.

Jie Wu (F'09) received the BS degree in computer engineering, and the MS degree in computer science from Shanghai University of Science and Technology (now Shanghai University), Shanghai, China, in 1982 and 1985, respectively, and the PhD degree in computer engineering from Florida Atlantic University, Boca Raton, in 1989. He is a chair and a professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, Pennsylvania. Prior to joining Temple University, he was a program director with the National Science Foundation, Arlington, Virginia. He has published more than 550 papers in various journals and conference proceedings. His research interests include wireless networks and mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He has served as an IEEE Computer Society Distinguished Visitor and is the chairman of the IEEE Technical Committee on Distributed Processing (TCDP). He serves on the editorial board of the *IEEE Transactions on Computers* and the Journal of *Parallel and Distributed Computing*. He is program co-chair for IEEE INFOCOM 2011. He was also general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, and DCOSS 2009. He is an ACM distinguished speaker and an IEEE Fellow.