This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

SCPS: A Social-aware Distributed Cyber-Physical Human-centric Search Engine

Haiying Shen*, *Senior Member, IEEE*, Jinwei Liu, Kang Chen, *Student Member, IEEE*, Jianwei Liu, *Student Member, IEEE*, Stanley Moyer, *Senior Member, IEEE*

Abstract—Advances in ubiquitous sensing, computing and wireless communication technologies are leading to the development of cyber-physical systems (CPS), which promise to revolutionize the way we interact with the physical world. CPS applications, such as healthcare monitoring, may involve many users and objects scattered over a wide area. One critical function of CPS is object search in the physical world through the cyber sphere that enables interaction between the cyber and physical spheres. Some of the previously proposed physical object search engines use RFID tracking, and others collect the information of object locations into a hierarchical centralized server. The difficulty of widely deploying RFID devices, the centralized search, and the need for periodical location information collection prevent CPS from achieving higher scalability and efficiency. To deal with this problem, we propose a Social-aware distributed Cyber-Physical human-centric Search engine (SCPS) that leverages the social network formed by wireless device users for object search. Without requiring periodical location information collection, SCPS locates objects held by users based on the routine user movement pattern. Moreover, using a social-aware Bayesian network, it can accurately predict the users' locations even at the occurrence of exceptional (i.e., non-routine) events (e.g., raining) that break user movement pattern. Thus, SCPS is more advantageous than all previous social network based works which assume that user behaviors always follow a certain pattern. Further, SCPS conducts the search in a fully distributed manner by relying on a distributed hash table (DHT) structure. As a result, SCPS achieves high scalability, efficiency and location accuracy. Extensive real-trace driven simulation results show the superior performance of SCPS compared to other representative search methods including a hierarchical centralized method, a decentralized method, and two social network based methods. The results also show the effectiveness of different components of SCPS.

Index Terms—Cyber-physical systems (CPS), Distributed hash table (DHT), Objection location, Distributed systems, Scalability

1 Introduction

Advances in ubiquitous sensing, computing and wireless communication technologies are leading to the development of cyber-physical systems (CPS), which promise to revolutionize the way we interact with the physical world. CPS are computer systems that monitor and interact with a constantly changing physical environment. While many technologies are important to achieving high performance CPS, perhaps one of the most essential challenges is object search in the physical world through the cyber sphere that enables interaction between the cyber and physical spheres [1]. The problem dealt in this paper is human-centric object search. That is, *how to efficiently search objects carried by people (such as documents, keys and electronic files) in the physical world through a computer system?* Such a search engine could significantly facilitate our work and daily life. Image the search engine helps a student to locate persons who have the key to his lab when he forgot to take the key or locate the textbook he needs in the coming class when he forgets to take the book.

CPS applications, such as healthcare monitoring, are becoming increasingly prevalent, and involve ubiquitous users and objects scattered over a wide area. Large-scale CPS require that a search engine can provide *scalable and efficient* search service. *Low latency*, which enables users to locate objects quickly, and *high location accuracy* are also critical requirements to the search engine design. However, current physical object search methods fail to meet these requirements.

Many existing search methods for physical objects rely on RFID tracking [2], [3], [4]. These methods are not applicable for searching objects carried by people who are moving around in a wide area, since deploying a large number of outdoor sensing devices is prohibitively costly. Some object search methods such as MAX [3], Microsearch [5] and Snoogle [4] utilize a hierarchical centralized structure, where substations are connected to one central station. The substations collect information of its nearby objects and conduct local search for its nearby nodes. Queries cannot be resolved locally are forwarded to the central station, which conducts global search among substations. However, the centralized global search leads to low scalability.

Note that the number of mobile devices with ad hoc wireless communication capacities (e.g., WiFi and Bluetooth) has been increasing at an incredible speed. For example, the worldwide iPhone users increased by 21.1 million in 2009 and by 26.8 million in 2010 [6]. The mobile device users constitute a social network, in which human mobility exhibits certain patterns, and is predictable to a large extent [7]. Also, individuals are tied by one or more specific types of relationship, such as friendship, kinship or trade [8]. In addition, wireless sensors are widely deployed for monitoring environment, such as weather and traffic. The increasing number of mobile users, wireless sensor nodes and base stations (BSs) nowadays promises to enhance the underlying communication capabilities needed in scalable search engines and creates new opportunities for innovations in human-centric object search.

By taking these opportunities, in this paper, we pro-

- * *Corresponding Author. Email: shenh@clemson.edu; Phone: (864) 656 5931; Fax: (864) 656 5910.*
- *Haiying Shen, Ze Li and Jianwei Liu are with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, 29634. E-mail: {shenh, jinweil, kangc, jianwel}@clemson.edu. Stanley Moyer is with the Global Inventures, San Ramon, CA 94583. E-mail: smoyer@inventures.com.*

Digital Object Indentifier 10.1109/TC.2013.211 0018-9340/13/$31.00 © 2013 IEEE

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

pose a Social-aware distributed Cyber-Physical human-centric Search engine (SCPS) which does not specifically depend on additional RFID or sensor devices. Without periodical location information collection, SCPS locates objects held by users based on the user movement patterns in the social network. Moreover, using Bayesian network combined with social network (i.e., social-aware Bayesian network), it can find the users' locations when exceptional (i.e., non-routine) events (e.g., raining, traffic jam and meeting friends) occur, which breaks user routine movement patterns. For example, a person does not play football as usual when it is raining but goes to the gym instead. If two friends[1] bump into each other, they may stay together for a time period. Further, SCPS conducts the search in a fully distributed manner by relying on a Distributed Hash Table (DHT) structure constituted by BSs, which search objects for their nearby mobile users. SCPS is distinguished by its high scalability, efficiency, low latency and high accuracy in object searching.

Social networks also have been leveraged in many routing schemes in Delay Tolerant Networks (DTNs) and Mobile Ad hoc Networks (MANETs) [9], [10], [11], [12], [13], [14]. Assuming that the meeting probability between nodes keeps approximately the same all the time, these schemes exploit the history of contacts and cluster nodes with high meeting frequency or choose the node which has high probability of meeting the destination as the next hop. However, the schemes neglect the external factors (or conditions) in the physical world that break people's routine movement patterns. SCPS is more advantageous than these schemes in that it can handle exceptions in human mobility pattern, thus providing more accurate node location prediction. The social-aware Bayesian network in SCPS can be adopted in these schemes to enhance routing efficiency. Like the social network based schemes and current online social networks (e.g., Facebook), we assume that users joining in the SCPS system are willing to provide social information (e.g., relationship and interests) and enable the system to predict their locations. We leave the privacy protection from malicious nodes as our future work.

The rest of the paper is outlined as follows. Section 2 presents a survey of related work. Section 3 details the SCPS design with an emphasis on its structure, social-aware Bayesian network and object searching process. Section 4 presents the performance evaluation of SCPS in comparison with other methods, and Section 6 concludes this paper with remarks on our future work.

2 Related Work

Physical object search. In recent years, a number of methods for physical object search or localization have been proposed. MAX [3] is perhaps the first search engine for physical objects. It has a centralized hierarchy formed by station and substation for object searching. Descriptions such as "the 2nd shelf on the 1st floor" are utilized to depict the locations of the substations. Substations can sense the RFIDs attached to the objects nearby, and they are responsible for building inverted index[2] of their nearby objects for local search. The central station stores inverted index of substations. Nodes send queries to the substations, which send back the IDs of nodes containing the searched keywords. For the queries that cannot be met locally, they are directed to the central station, which returns the most matched substations. Snoogle [4] extends MAX by adding the schemes for supporting multiple-keyword search and top-k query. It also uses a two-layer hierarchical architecture formed by KeyIP and IPs, functioning as the central station and substations in MAX, respectively. Microsearch [5] further details the design and implementation of the top-k search using wireless sensor nodes with limited memory space. Also, this work presents a memory efficient algorithm and a theoretical model of the search to find the optimized parameters including inverted index size. However, the centralized structure in MAX, Snoogle and Microsearch make them vulnerable to congestions due to many global queries, leading to low scalability. Home-explorer [15] is another physical object localization system. It supports the localization of both "smart nodes" (nodes equipped with sensors) and "hidden objects" (objects without sensors). It detects the hidden objects according to their influence on the smart nodes and further infers their categories by matching the pre-defined attributes of object categories. MASCAL [2] is an operational hospital asset management system. It utilizes active Wi-Fi RFID tags to enable the real-time tracking of patients and assets inside a hospital.

Some works [16], [17], [18] focus on the localization of sensors based on coordination schemes, while others rely on radio frequency or ultrasound to support sensor location. Bruck *et al.* [19] proposed an anchor-free 2D localization scheme by using local angle measurement techniques. Kwon *et al.* [20] proposed a self-localization scheme for a large scale sensor network. The scheme adopts acoustic ranging or radio interferometry to acquire inter-node distance measurements to compute the node locations.

DHT systems. DHT systems, such as CAN [21], Chord [22], Pastry [23], Tapestry [24] and Moore [25], are decentralized file sharing systems which provide scalable key-to-vale search schemes. The DHT systems store files to their mapped nodes and locate files given file keys in a distributed manner. They have optimized complexity of $O(logN)$ for one lookup and $O(log^2N)$ for one node leaving or joining, where N is the number of nodes in the system. However, the DHT overlays cannot be directly applied to MANETs, due to the limited computing and energy capacity of mobile devices, transmission range constraint, and dynamic node mobility. High maintenance overhead for DHTs due to dynamics can easily exhaust the energy of mobile devices.

Social network based routing. Social networks recently have been leveraged for routing in DTNs [11], [12], [14] and MANETs [9], [10], [13]. These routing schemes derive the routine of each node from statistical analysis of historical data to help in routing. PROPHET [9] always chooses the node with the highest probability of meeting the destination as the next hop and requires nodes to exchange information for the probability cal-

1. Two persons with a direct relationship (e.g., co-workers, family members, classmates and business partners) in a social network are called friends.

2. Inverted index is a data structure, which shows the mapping of objects to their owners or locations.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

culation. To optimize PROPHET, Daly and Haahr [11] exploited ego networks to calculate the centrality metrics of nodes for routing without the requirement of information exchange. MOPS [12] and Bubble Rap [13] divide the nodes into communities based on contact frequency, and select the nodes having high frequency of contacting two communities as brokers for communication between the communities. In SOLAR [10], requesters receive the "hub list" (places one always visits) of the destination node from its acquaintances, and use geographic routing [26] to send the queries to the places in the list. Mtibaa *et al.* [14] assigns weights to nodes according to their centralities for the next hop selection in routing. However, these methods only consider the node movement patterns or routines (i.e., the places people always visit or friends they always meet) in prediction, but neglect exceptions in people movement. SCPS addresses this deficiency by capturing the exceptions in routine human mobility in location prediction.

Bayesian networks. Bayesian network is a probabilistic graphical model that is widely used in machine learning. It uses a structure to connect parameters for probability inference. Inference, parameter learning and structure learning are the main techniques in a Bayesian network. Clique tree propagation [27], [28], [29] is the most commonly used probability inference algorithm in Bayesian networks. For parameter learning in Bayesian networks, if the variables in a Bayesian network are fully observed, the maximum likelihood method (ML) [30] and the maximum a-posteriori (MAP) [31] method can be used. Otherwise, the Expectation-Maximization algorithm (EM) [32] can be utilized. ML and MAP utilize a statistic method to calculate the conditional probabilities in the Bayesian network, while EM gives the estimation of conditional probabilities. There are two main approaches for structure learning. One is based on scoring function and model space searching [33], and the other approach uses constraint based algorithms [34], [35]. Bayesian network has been recently used in various applications such as static indoor ad-hoc sensor location [36], trust model construction in distributed systems [37], [38], [39], intelligent decision rule establishment for routing [40], mobile environment analysis [41] and medical decision support [42].

3 The Design of the SCPS System

3.1 An Overview of SCPS

Distributed hierarchical structure: SCPS can be used in different areas such as business, government or battlefields. In this paper, we use a university campus as an example to explain SCPS. The ubiquitous CPS system that we focus on has three kinds of nodes: mobile nodes (MNs), sensors and base stations (BSs). MNs are mobile devices held by users and connected through an ad-hoc network, and the sensors are in existing sensor networks (e.g., *Sensor Andrew* in CMU [43]) used to monitor environments such as weather and traffic. The BSs has two network interfaces: ad-hoc WiFi interface used to communicate with MNs and sensors, and wired interface used to connect all the BSs.

SCPS builds the nodes into a hierarchical structure, as shown in Figure 1, to efficiently manage object data for search service. In the upper layer of the structure, the BSs

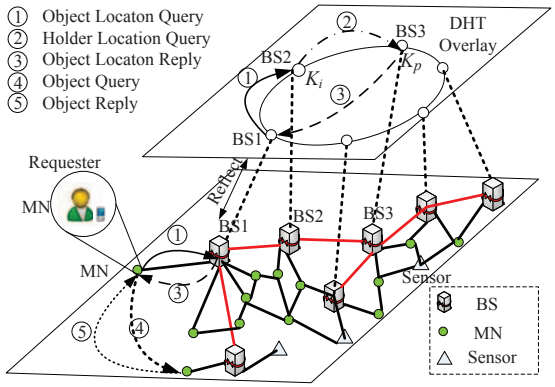


Fig. 1. SCPS structure and object querying.

form a Chord DHT. In the lower layer, MNs and sensors communicate with their closest BSs. We assume each MN has a GPS which can determine its own location. The MNs far away from BSs can communicate with them through other MNs using a MANET routing protocol. We use the geographical routing [26] in our approach. In SCPS, different nodes have different functionalities:

- MNs register with BSs about the objects they would like to share with others. MNs also send queries for object location to BSs. The Bluetooth modules in MNs detect people meeting events.
- Sensors sense and transmit environmental data (e.g., weather and traffic) to BSs, which is used for object location prediction. Sensors can also sense the objects in the interested object list of the SCPS system such as textbooks.
- BSs collect data from MNs and sensors, and use a prediction model to locate objects. They receive requests and respond to requester MNs with object locations. Each BS is responsible for storing inverted indexes of certain objects and calculating the locations of certain MNs (i.e., object holders). We call the BS the *holder indexer* of the objects and *locator* of the MNs, respectively.

An object's name is determined based on a pre-defined name list globally known by all nodes in the system. An object name is for a category of objects that have the same name and same functions. How to define the name list for different applications leaves as our future work. In SCPS, the data is either initially collected or promptly collected. By "promptly collected", we mean that the data of an event is reported to a BS once the event occurs. Table 1 shows all the data needed in SCPS, the way it is collected and its source. We explain the different types of data below.

TABLE 1
Data types and sources

Data	Collecting time	Source
Objects	Initial/Prompt	Users/sensors
MN movement routines	Initial	MNs
Social relationship	Initial	Social network or users
Environmental events	Prompt	Sensors
Social events	Prompt	Bluetooth of MNs

- *Objects.* The data of objects is used to produce inverted indexes for object search. The inverted index of an object can be "obj1: user1, user2" generated from the data initially input by the users or "obj1: loc1, loc2" generated from the data promptly reported by the sensors that sense the objects.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

- *MN movement routines.* Each MN reports its time-location movement information to BSs at the initial phase. BSs then derive the movement routines of MNs (i.e., the location of a MN at a certain time) for object location.
- *Social relationship.* This data can be retrieved from the information system in the organization where the CPS is applied or can be input by the users. For example, Clemson University has a web portal called TigerWeb, from which we can find the users in the same class, the same department and the same lab.
- *Environmental events.* Once unusual events are detected such as raining, traffic jams and high blood pressure, a sensor reports the data to its nearest BS.
- *Social events.* We consider meeting friends as an example for social events in this paper. When a MN's Bluetooth detects the meeting of persons at the same location, the MN reports the event to its nearest BS. SCPS can be easily extended to include other social activities that can be detected.

Social-aware Bayesian network prediction model: Each BS has a social-aware Bayesian network used to calculate the location of queried objects. The BSs use the users' historical mobility, user relationship, the events of meeting friends, and environmental events in constructing a Bayesian network. The social network helps Bayesian network to conduct prediction more quickly and accurately without the need of a long training time period, while the Bayesian network helps social network to capture the external factors of non-routine events in order to accurately locate objects. The social-aware Bayesian network can predict persons' behaviors based on social relationship, social events and environmental events.

Cyber-physical searching: For the cyber-physical search, the inverted index of an object is collected and stored in the object's holder indexer in the DHT-based overlay in the distributed hierarchical structure. To search an object held by a person, a requester sends a query to its nearest BS. Using the DHT routing algorithm, the BS sends the object query to the holder indexer of the object, which further forwards the object query to the locators of the object holders. The locators calculate the locations of the object holders using the social-aware Bayesian network prediction model, and return the predicted object locations to the BS of the requester. To search an object whose data is reported by sensors, the holder indexer directly replies the locations to the requester's BS. Then, the BS forwards the response to the requester, which relies on geographical routing to send its object query to the object holder.

3.2 Data Collection in the Distributed Hierarchical Structure

3.2.1 Data storage and lookups in a DHT

SCPS forms BSs into a DHT-based overlay for data collection and storage. DHTs are a class of decentralized distributed schemes that provide a lookup service similar to a hash table. In DHT, each object (or file) has a key, which is the consistent hash function of the object's name. A DHT has a function *Insert(key, value)* to store the value to a node. We call the node the value's owner. Responsibility for maintaining the mapping from keys to values is distributed among the nodes. Each node has an ID which is the consistent hash value of its IP address. An object is stored in a node whose ID is the closest (or first succeeding) the object's key. For example, an object with key 4 is stored in a node with ID 4. If there is no node having ID 4 and node 5 is a participant, then the object is stored in node 5. Any participating node can efficiently retrieve the value associated with a given key through function *Lookup(key)*. With such object distribution, a change in the set of participants causes a minimal amount of disruption. This allows DHTs to scale to a large number of nodes and to handle continual node arrivals, departures and failures.

3.2.2 Object data collection and storage

The DHT overlay collects and stores the different types of data for efficient search. The holder indexer of an object is the owner of the object's key, and the locator of a MN is the owner of the MN's ID. The object data is either input by users manually using MNs or reported by the sensors when they detect the object. To report object data, a node sends an index report which contains the user name and objects (s)he holds to its nearest BS. For example, in Figure 2, the four persons A-D send their index reports to their nearest BSs. User A sends its index report "*User A has object 1, 2 and 3*" to BS1. After receiving an index report, a BS combines it to its local inverted index (LII), which shows each object's name and their holders. For example, as shown in the figure, in BS1's LII, obj1's owners are user A and user C. Each BS periodically executes *Insert(K, LII)* for each object in its LII, where K is the hash value (i.e., key) of the object's name. Finally, this message arrives at the holder indexer of the object, which stores the holder information of the object in its global inverted index (GII). For example, if BS2 is responsible for storing the GII of obj1, then BS1 and other BSs, which have obj1 in their LIIs, send messages to BS2 by *Insert(K_1, LII)*. Then, BS2 updates the record of obj1 in its GII. The data reported by sensors is collected in the same manner. Thus, the GII can directly indicate the locations of sensed objects.

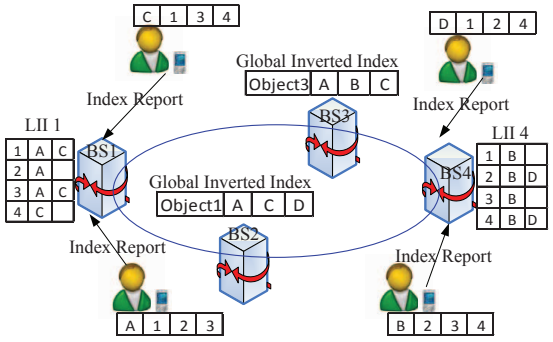


Fig. 2. Inverted indices of objects.

In this paper, we use N to denote the number of BSs in the DHT and use M to denote the number of all nodes in the system.

Proposition 3.1: Suppose there are totally B different objects in the system, then the total number of nodes that must be contacted for object data collection and storage is $O(M + NB \log N)$.

Proof: Since each node needs to send an index report to its nearest BS, $O(M)$ messages are needed for sending index reports for all nodes. Since each BS needs

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

to send its local inverted index for each object to the holder indexer of this object, and the number of nodes that must be contacted to reach a node in an N-node DHT is $O(\log N)$ [22], the total number of hops for all BSs to report local inverted indices for all objects is $O(NB \log N)$. As a result, the total number of nodes that must be contacted for object data collection and storage is $O(M + NB \log N)$. □

3.3 Social-aware Bayesian Network Prediction Model

Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies, which can be used to model complex event-driven casual relationships. In the Bayesian network, nodes represent random variables, and the links between nodes represent the existence of causal relationships among nodes. It can be used to model and infer the probability of an event (i.e., node) according to other observed events (i.e., the node's parent nodes linking to it). Figure 3 shows an example of a Bayesian network for one person. It has variables including time, rain, illness, meeting friends and location. The location has variables such as football field, gym, home and basketball court. The events determine location values. Football field is the value of the routine event when none of the exceptional events happen. Exceptional events such as rain, illness and meeting friends break the person's routine. For example, at time 5:00pm, if it is raining, the probability that the person is in the gym is 1, and the probability that the person is at the football field is 0. If the person is ill, then he is at home. If he meets his friend Bob, then he is playing basketball in the basketball court.

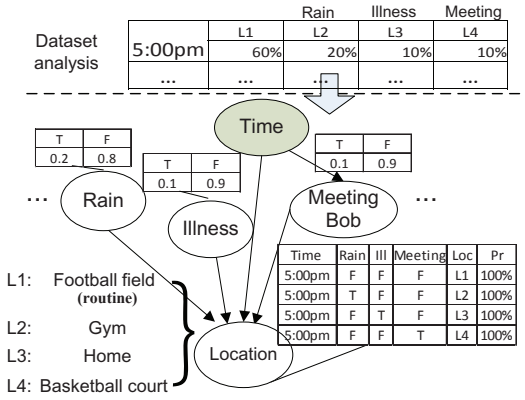


Fig. 3. A Bayesian network for one person.

It was indicated that the movement of humans has certain patterns [7]. Current social network based works in MANETs [10], [9] exploit the meeting probability of nodes in determining the next hop in routing or in community construction. However, this coarse-grained view of node movement pattern cannot be used to accurately predict the locations of people because it fails to consider time and non-routine events (exceptional events). Node A usually has higher probability to meet B than C does not mean it has higher probability to meet B than C at a certain time. Also, a non-routine event such as rain may prevent a person from going to a place (s)he usually goes at a certain time.

To provide a fine-grained view of node movement for more accurate location prediction of people, we consider the routine of a person as a chain through time. Specifically, we find the place each person usually stays during a certain time period. For example, Tom stays in his research lab during 8:00am-12:00pm and stays in the cafeteria during 12:00pm-1:00pm. We further rely on the event-driven Bayesian network to capture the influence of exceptional events. For example, the Bayesian network can make the correct prediction that Tom is in the gym instead of basketball court by considering the raining event through adjusting the probability of variables in the Bayesian network.

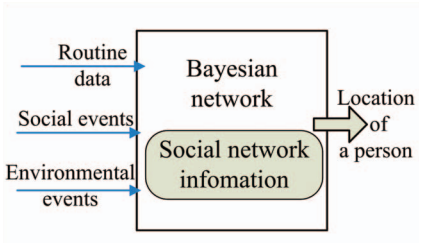


Fig. 4. Social-aware Bayesian network prediction model.

We further consider the events of meeting friends and social network information in the process of Bayesian network prediction. The social network information includes the social relationship or common interests between friends. For example, nodes A and B are friends and they have the common interest of playing basketball. The information of meeting friend events and social relationship, if properly used, can produce more accurate prediction result of people's locations with shorter Bayesian network training time. The intuition behind this scheme is the fact that people with certain social relationships tend to meet at certain places determined by their relationship and common interests in our everyday life. If friends in a basketball club are meeting each other, they are likely to play basketball or watch a basketball game in the basketball court. If two professors are meeting together, they are likely to be in the department building. After training, the Bayesian network can predict the locations of people when they meet each other according to their social relation information.

Figure 4 shows the social-aware Bayesian network prediction model. The inputs of the prediction model are routine data, social events, environmental events and social relationship, while the output is the location of a person. After the model is trained, it can predict the locations based on people's routines. If an exception occurs such as an environmental event or social event, the model makes the prediction accordingly and it needs to consider the social information for social event. For example, when the SCPS system detects a social event that A and B are together, based on their relationship, it predicts that A and B are playing basketball at the basketball court. Prediction based on non-routine events reduces the training time length of the Bayesian network, while raising the prediction accuracy. After the initial training, the Bayesian network can be continuously trained from the feedback of users on the correctness of predictions. The training will progressively increase the prediction accuracy and adaptability of the Bayesian network.

In Section 3.5, we will present the details of building a Bayesian network using a real trace of node mobility. The locator of a MN builds such a prediction model for the MN. To enable each locator to collect the required

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

data for building the Bayesian network, each MN reports its movement routine to its locator at the initial stage, the SCPS stores the social relationship information of each MN to its locator initially, and the data of social events of a MN is promptly sent to its locator. In the DHT-based overlay, *Insert(K,D)* is used to send data to a MN's locator, where K is the ID of the MN and D is the data. Environmental event data is sent to the locators of MNs that are influenced by the events. If most MNs are influenced by an event, say raining, then broadcasting is used to notify all BSs of the event.

3.4 Cyber-Physical Searching

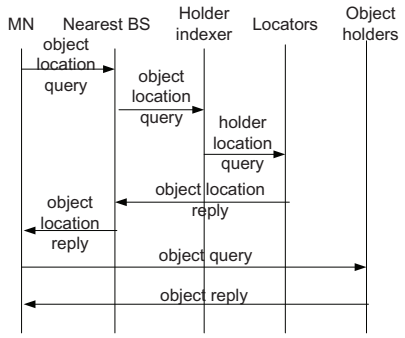


Fig. 5. The process of object querying.

Figure 1 shows the process of object querying in SCPS. The sequence of steps is also illustrated in Figure 5. When one requester wants to search an object named obj1, it sends an *object location query (OLQ)* containing the name of the object to its nearest BS, say BS1. If the requester is not in the transmission range of any BS, it uses geographic routing to send the query to BS1. After receiving the query, BS1 first uses the consistent hash function to hash "obj1" to get its key K_1, and then executes *Lookup(K_1,OLQ)*. Based on the DHT routing algorithm, *OLQ* arrives at the holder indexer of obj1, say BS2, which has the inverted index of obj1. As shown in Figure 2, the inverted index shows that the holders of obj1 are mobile users A, C and D. Then, BS2 hashes the holder names and sends out the *holder location queries (HLQ)* using the functions of *Lookup(K_A,(HLQ,obj1))*, *Lookup(K_C,(HLQ,obj1))* and *Lookup(K_D,(HLQ,obj1))*, where K_A, K_C and K_D are the consistent hash values of the MN IDs. The three requests arrive at the locators of nodes A, C and D, which predict the locations of A, C and D, respectively. Assume BS3 is the locator of user A. It stores the social-aware Bayesian network prediction model of user A. Using the prediction model, BS3 predicts the location of A and responds to BS1 with *object location reply*. BS1 then replies to the requester with the locations. After receiving the locations, the requester sends *object query* to each object holder using geographic routing, and each holder responds to the requester with *object reply*. SCPS can easily adopt the scheme in CYBER [44] to enable multi-term search. Such techniques are orthogonal to our study in this paper. Algorithm 1 shows the pseudo-code of the cyber-physical searching process.

Proposition 3.2: When a requester can directly send a message to its nearest BS, the number of nodes that must be contacted to find the object holder locations of an object is $O(M \log N)$.

Algorithm 1: Algorithm run by a BS for object querying.

```
   // abbreviations
1  OLQ: Object Location Query, HLQ: Holder Location Query,
2  OLR: Object Location Reply, OQ: Object Query, OR: Object Reply
3  while (1) do
4      if new packet P arrives then
5          switch P do
6              case P==OLQ
7                  objectName = ExtractObjectName(OLQ);
8                  if !IsHolderIndexer(objectName) then
                       // record the addr. of the OLQ
                          initiator MN
9                      Record(MN_r, requesterList);
10                     K_i = Hash(objectName);
                       // send OLQ to the holder indexer of
                          the object
11                     Lookup(K_i, OLQ);
12                 end
13                 else
                       // check the inverted index and
                          retrieve the holder names
14                     holderNameList = GetHolderNames(objectName);
15                     for every holderName do
16                         K_h = Hash(holderName);
                           // BS_r: addr. of the OLQ sender
                           // send HLQ and BS_r to each
                              holder of the object
17                         Lookup(K_h, (HLQ,BS_r));
18                     end
19                 end
20             endsw
21             case P==HLQ
22                 personName = ExtractPersonName(HLQ);
23                 location= BNPredict(personName);
                   // send the predicted location to BS_r
24                 Send(location) to BS_r;
25             endsw
26             case P==OLR
27                 MN_r = FindRequester(requesterList);
                   // send the location to MN_r
28                 Send(location, MN_r);
29             endsw
30         endsw
31     end
32 end
```

Proof: Suppose a requester wishes to search an object. It sends one message directly to its nearest BS. The number of nodes that must be contacted to reach a node in an N-node DHT is $O(\log N)$ [22]. Therefore, the object location query from the nearest BS travels $O(\log N)$ hops to reach the holder indexer BS. The maximum number of nodes that can hold the requested object is M. Then, the number of BSs must be contacted for the queries from the holder indexer to the locators is $O(M \log N)$, and that for the responses from the locators to the nearest BS is $O(M)$. The nearest BS then sends one message to the requester. As a result, the total number of nodes contacted is $1 + O(\log N) + O(M \log N) + O(M) + 1 = O(M \log N)$. □

3.5 Bayesian Network Construction Using the Real Trace

Real trace data. Reality [45] is a dataset collected at the MIT Reality Group, which contains (1) survey data of 94 users in 10 months, (2) cell phone trace, and (2) Bluetooth trace. The survey includes questions such as "is this person a part of your close circle of friends?" and "do you own a car?" We derived the friend social relationship from the survey data. The testing field at MIT is divided into areas, and each area is divided into cells. Each cell phone tower has a Tower ID (TID) consisting of IDs of the area and cell where it is located.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

Every record of a cell phone trace includes the time and location. The location is represented by the TID of the tower that the user was using at that time. The Bluetooth trace records the meeting time and MAC addresses of Bluetooth devices carried by users, from which we can infer the meeting events of users. Since the time period that users are active is usually during 8:00am-10:00pm, we used the records during this time in the dataset. To make the Reality dataset suitable to our design and testing scenarios, we processed the dataset information as described below.

- *Node movement.* We need location records with a certain granularity, say for every minute, in order to simulate node movement. However, many missing records for days with "no signal" in the dataset prevent us from identifying the exact location of nodes. We went through all records to retrieve users with few missing records and a long time period of records. We identified 19 such users with a relatively long period of records of 11 days. We used the records of the first 10 days for training of the prediction model, and those of the last day for prediction experiments.
- *Social events.* From the Bluetooth trace, we found that the TIDs of friend meetings at different times are always different. This does not necessarily mean that the friends always meet at different locations at different meeting occurrences. This is because different cell phone service providers, or even the same provider, use several towers with different TIDs in the same area. To generate the events of meeting friends, we made modification on the selected dataset. Specifically, we identified the location of the longest accumulated meeting time of two nodes during the 11 days, and changed all meeting locations of them to that location.
- *Environmental events.* The Reality dataset does not provide environmental event information used in the prediction model of SCPS for accurate prediction. We assumed that a TID represents a location, calculated the frequency of each user visiting each location during the 11 days, and considered the infrequent location visiting occurrences as results of environmental events. For example, at 5:00pm, user A visits Loc1 with a probability of 80%, and Loc2 and Loc3 with a probability of 10%, respectively. Then, we considered the occurrences of visiting Loc2 and Loc3 as the result of event1 and event2, respectively. In the dataset, if a user visits a certain place at a certain time corresponding to an event, we assume this event occurs.
- *Physical location.* The Reality dataset does not provide physical locations of towers, while SCPS provides a real physical location (x, y) coordinates for a queried object. Since records of close areas are always recorded together in the dataset, we measured the proximity of two areas according to the frequency that two records of these areas are together. Based on the measured proximity, we then matched the areas to the buildings on the MIT map, and randomly distributed the TIDs of each area around the corresponding building. Then, we measured the (x, y) coordinates to represent each physical location.

Bayesian network structure. In the following, we describe how Bayesian network of each person, as shown in Figure 3, is created. To construct a Bayesian network, we first need to identify the variables and (or) their values. We then build a Conditional Probability Table (CPT) of each variable by parameter learning. We identify three variables: time, event and location. The time variable has 7 different values (14 hours from 8:00am to 22:00pm are divided into 7 intervals with a two-hour granularity). The event variable has two values: true and false, and the location variable has a number of values indicating different locations. The parents of *location* are *time* and *event* because human mobility is influenced by both factors. Accordingly, we constructed the Bayesian network using all retrieved events, locations and time variables from the records in the real trace. In the Bayesian network, there are links from: 1) time to the location (routine), and 2) events to the locations (exceptional events).

Bayesian network parameter learning. After the structure of the Bayesian network is built, it needs to learn the parameters before being able to conduct location prediction. In the real trace, since the variables are all observed, we use the maximum likelihood algorithm [30] for parameter learning. This algorithm estimates a parameter by maximizing the likelihood function of the parameter. The estimation of a parameter is calculated using the following equation:

$$\begin{aligned} Pr(A \mid B) =& Pr(X = x, Y = y)/Pr(Y = y) \\ =& [n(X = x, Y = y)/N]/[n(Y = y)/N] \\ =& n(X = x, Y = y)/n(Y = y) \end{aligned} \quad (1)$$

where X and Y are variables, x and y are values of the variables, $n(X = x, Y = y)$ is the number of records that meet $X = x$ and $Y = y$, and N is the total number of all records.

From the dataset, we calculated the historical frequency (P_r) of a user visiting different locations at different time intervals. For example, if there are 100 records at 5:00pm, in which 60 records are for Loc1, 20 records for Loc2, 10 records for Loc3, and 10 for Loc4, respectively, we retrieve

$$\begin{aligned} & Pr(Loc = Loc1, Time = 5:00pm) \\ & = n(Loc = Loc1, Time = 5:00pm)/n(Time = 5:00pm) \\ & = n(X = x, Y = y)/n(Y = y) = 60\%, \end{aligned} \quad (2)$$

which means that the probability of the person staying at Loc1 at 5:00pm equals 60%. In this way, we can infer that at 5:00pm, the probability of the person staying in Loc1 is 60%, in Loc2 is 20%, in Loc3 is 10%, and in Loc4 is 10%. Recall that since the real trace does not provide data for non-routine events, we assume that a person visits an infrequently visited place due to an exceptional event. Using the above statistic method, we can retrieve the probability distribution of the locations and corresponding events. We then achieved the probability distribution of locations for the Bayesian network structure. Figure 3 illustrates the examples of CPTs for the variables using the calculated probabilities, and the dependency of locations on the events and time. When an exceptional event happens, the probability of the person staying at the corresponding location is 100%. When no exceptional events happen, the probability that the person stay at the routine location is 100%.

4 Performance Evaluation

We conducted real-trace driven simulations to evaluate the prediction accuracy and system effectiveness of the proposed SCPS system based on the MIT Reality dataset [45] on human mobility.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

4.1 Experiment Settings

We conducted experiments on *NS-2* [46]. We used PlanetSim [47] to test the overhead and delay in the DHT-based overlay formed by BSs, and used the Bayesian Network Tools [48] for Bayesian network inference to test the Bayesian network prediction delay. We adopted GPSR [26] for geographic routing in the simulation. GPSR is used when a requester sends a query to its nearby BS not within its transmission range and when a requester sends an object query to the object holder. If the location the requester received is incorrect and the packet arriving at the location cannot find the object holder, GPSR uses perimeter mode routing (also called face routing). The packet is routed along the faces of a planar subgraph until either the destination is reached or greedy forwarding can be performed again. To prevent routing loops in the perimeter routing, the packet records the ID of the first node forwarder in perimeter routing mode. If the packet returns to this node, it means a routing loop is generated and the packet is dropped.

We compared our SCPS system with SCPS without social-aware prediction (SCPSw/oS), SCPS without Bayesian network prediction (SCPSw/oP), Snoogle [4], statistic based method (Statistic) [9], and MOPS [12].

Variants of SCPS. SCPSw/oS is SCPS without considering social events in Bayesian network prediction. SCPSw/oP is SCPS without using the location prediction mechanism. Instead, it uses the periodical reporting from mobile devices to BSs to keep track of the locations of every user. When users are out of the range of BSs, they use multi-hop geographic routing to report their locations.

Snoogle. Snoogle [4] is a centralized cyber-physical search engine with a two-layer hierarchical structure over the sensor nodes. The higher layer is a central server called KeyIP and the sensors in the lower layer are called IPs. Queries for arbitrary objects are firstly directed to the KeyIP, which returns the best matched IPs. Then, nodes send queries to those IPs, which send back the IDs of nodes containing the searched objects by checking their inverted indexes. In the simulation, we used BSs to function as the IPs, and used the BS in the center of the simulation area as the keyIP. To be comparable to SCPS, Snoogle also adopts geographic routing, and relies on periodical location reporting to IPs for destination locations. In Snoogle and SCPSw/oP, mobile nodes report their locations to BSs every 5 seconds, thus the locations retrieved by requesters are always updated.

Statistic. The statistic based method here refers to a scheme similar to SCPS, but only uses routines as the location prediction result. It achieves the probability distribution of variables over time from the training data, and considers the variable with the highest probability as the prediction result. For example, from the visiting probability distribution of all the locations Tom visited at 12:00pm, we find that he visited the cafeteria with the highest probability. Then, the cafeteria will be the prediction result at 12:00pm for Tom in the statistic based method.

MOPS. MOPS [12] is a publish/subscribe system based on social network information. It clusters nodes with frequent communications into a community. It uses nodes having frequent contact with different communities as brokers for inter-community communication and uses direct contact between nodes in the same community for intra-community communication. In MOPS, a node carries a message until it meets the destination.

Performance Comparison. SCPS distributes load among BSs, while Snoogle heavily relies on the central server for object searching. Therefore, many requests in Snoogle may not be resolved due to the overload of the central server. The experimental results will show the advantage of the distributed manner of SCPS compared with Snoogle. SCPS is also advantageous with higher location prediction accuracy since it jointly considers routine data, social events and environmental events. We will show this advantage by comparing SCPS with SCPSw/oS and Statistic that only consider part of the factors. SCPS is novel because it leverages the social networks without relying on the periodical location reports from the mobile nodes, which otherwise generates a high overhead. We will show this merit by comparing SCPS with SCPSw/oP that uses periodical location reports. Further, SCPS uses a backbone DHT and geographical routing to actively forward messages, while MOPS completely relies on the node movement to forward messages though it also leverages social networks. Thus, SCPS can achieve faster object querying than MOPS.

Table 2 summarizes the default parameters used in the simulation unless otherwise specified. We conducted our simulations in a 600m$\times$600m region with 69 nodes, among which 19 nodes move according to the real trace and 50 nodes move randomly with a speed randomly chosen from [0,2]m/s. This speed is close to the normal human walking speed. The communication range of wireless ad-hoc nodes is 200m. All results over 2000 queries are averaged for the final results. The node movement and BS distribution are designed based on the real trace data described in Section 3.5.

We randomly assigned 95 items (5 copies of 19 items) to 19 persons. Every node starts to query after 20 seconds initialization time period at a certain query rate for 580 seconds. The simulation then ran 20 seconds before stopping. A query rate of $1/x$ means a query is sent out every x seconds. The item queried is randomly chosen from the 19 items. The length of a query packet is 28 bytes. There is no resending mechanism. If a packet fails to arrive at the destination, the item querying fails.

TABLE 2
simulation parameters

Environment Parameters	**Default Value**
Simulation area	$600m \times 600m$
Node Parameters	**Default Value**
Total number of mobile nodes	69
Total number of base stations	7
Physical layer	IEEE 802.11
Communication range	200m
Movement speed	0m/s - 2m/s
The length of a query	28 bytes
Query Parameters	**Default Value**
Query rate	1/10 (one query every 10 seconds)
Initialization period	20s
Query time	580s

We mainly consider the following metrics:

- **Location query drop rate.** This is the average ratio between the number of *object location replies* and the number of *object location queries*. It reflects the successful rate of the object location queries.
- **Object hit rate.** This is the average ratio between the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

number of *object replies* and the number of queries sent from the requesters to the object holders using geographic routing. This metric can reflect the accuracy of location prediction since a wrong location in geographical routing leads to routing failure.

- **Overhead.** This is the total number of hops passed by all messages in object searching. The messages do not include hello messages.
- **Query delay.** This is the average delay time of all successfully resolved object queries. The delay time of a query is the time elapsed from the time a requester sends out an *object location query* to the time the requester receives the *object reply*.

4.2 Prediction Accuracy

We compare the location prediction accuracy of SCPS with Statistic and SCPSw/oS. The training data for all methods includes the first 8400 records during the first 10 days we selected from the Reality trace. As described in Figure 4, the inputs of the prediction model in SCPS are the routine and environmental events, social events (i.e., people meeting) and social network information. SCPSw/oS does not need the social-related information; while Statistic only needs routine as input. The outputs of all methods are the every-minute locations of persons on the 11th day. Then, we calculated the prediction accuracy by comparing the predicted results with the 11th day real trace data. The *prediction accuracy* is defined as the ratio of the number of predicted locations which are identical to the 11th day records to the total number of the records.

Figure 6 shows the accuracy of the three prediction methods for predicting the locations of 19 persons. We observe that SCPS achieves the highest prediction accuracy among the three methods. It achieves around 82% average prediction accuracy, and even reaches 100% for some persons. SCPSw/oS produces around 70% average prediction accuracy, which is lower than SCPS. The average prediction accuracy of SCPS is 12% higher than that of SCPSw/oS, which indicates that combining social network information into the normal Bayesian network prediction model helps raise the prediction accuracy. Statistic achieves the lowest (around 31%) average prediction accuracy. This shows that Statistic cannot correctly predict the occurrence of the non-routine events without taking exceptional events (environmental and social events) into consideration, resulting in low prediction accuracy. These results verify the high location prediction accuracy of the social-aware Bayesian network in SCPS.

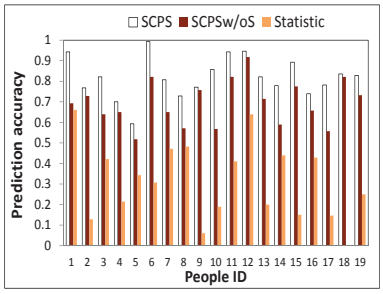


Fig. 6. Prediction accuracy.

4.3 Performance with Different Query Rates

In this experiment, we examined the performance of SCPS and other methods at different query rates. We varied the query rate of nodes over 7 different rates from 1/20 to 1.

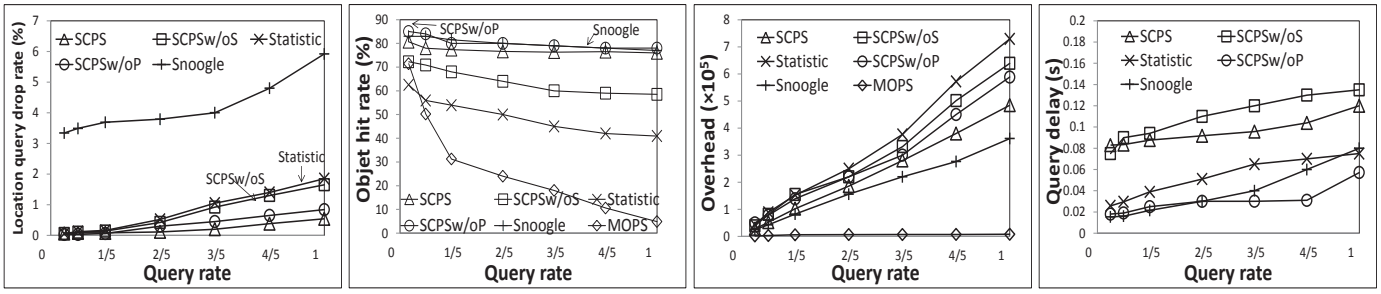
4.3.1 Location Query Drop Rate

Figure 7(a) shows the average location query drop rate of SCPS and other four methods with different query rates. Because nodes in MOPS always carry messages until they meet the destinations, MOPS's location query drop rate is always 0. Therefore, we do not include MOPS into the figure. We observe that the location query drop rate of Snoogle is much higher than the other methods and it increases dramatically as the query rate rises. It increases from 3.3% to 5.9% as the query rate increases from 1/20 to 1. This is because the centralized scheme of Snoogle suffers from congestion when there are a large amount of queries sent to the central KeyIP. The location query drop rates of SCPS and other methods are lower and have smaller increases than Snoogle since they adopt distributed structures. Using a distributed structure, queries are sent to different BSs instead of one central node. The distribution of workload among different BSs reduces the chances of BSs being congested.

We then discuss the drop rates of all methods except Snoogle. From the figure, we see that the varying ranges of the location query drop rates of SCPS, Statistic, SCPSw/oS, SCPSw/oP are [0.06%, 0.54%], [0.06%, 1.85%], [0.06%, 1.65%] and [0.03%, 0.84%], respectively. Their average location query drop rates for different query rates are 0.21%, 0.74%, 0.66% and 0.34%, respectively. We observe that the drop rates of all methods when the query rate is no higher than 1/5 are under 0.2. At the highest query rate, all methods have the highest drop rates. This is because a higher query rate generates more messages and hence more network congestions in routings, producing a higher location query drop rate. It is interesting to find that, at the highest query rate increases to 1, the location query drop rate of SCPSw/oS and Statistic increase more rapidly than SCPS. This is caused by the traffic congestion due to less accurate predicted locations in SCPSw/oS and Statistic. A packet with a wrong location is delivered for many hops until it revisits the first node it met in the perimeter routing mode or its TTL becomes 0. Many of such "hanging packets" generate congestion in the network, leading to high location query drop rate. SCPSw/oP generates higher location query drop rate than SCPS. In SCPSw/oP, the periodical reporting produces many messages, which increases the congestion and drop rate. The distributed structure adopted in SCPS and its higher prediction accuracy help it to reduce congestion, hence reduce the drop rate and enhance its scalability.

4.3.2 Object Hit Rate

Figure 7(b) shows the average object hit rate of the six methods with different query rates. The varying ranges of the average object hit rates of SCPS, SCPSw/oS, Statistic, SCPSw/oP and Snoogle are [76%, 81%], [59%, 72%], [41%, 63%], [78%, 85%] and [77%, 83%], respectively. Their average values of these object hit rates for different query rates are 77%, 65%, 50%, 81% and 80% respectively. We find that the object hit rate of SCPS is higher than the other two prediction based methods (SCPSw/oS and Statistic). This is because it can produce more accurate locations than the other two methods due to its higher prediction accuracy (as shown in Figure 6), so that a query can be sent to the correct destination with higher probability. The higher object hit rate of SCPS



(a) Location query drop rate. (b) Object hit rate. (c) Overhead. (d) Query delay.

Fig. 7. Performance of object search with different query rates.

than SCPSw/oS verifies the effectiveness of considering social events in prediction. Similarly, the object hit rate of SCPSw/oS is higher than Statistic, which verifies the importance of considering exceptional environmental events in predicting.

Relying on periodical location reporting, SCPSw/oP and Snoogle can always provide the accurate object locations, thus produce the highest object hit rate. It is interesting to see that the object hit rate of SCPS is nearly comparable to SCPSw/oP and Snoogle. This is caused by two reasons. First, SCPS provides relatively highly accurate object locations. Second, nodes in SCPSw/oP and Snoogle periodically send location reporting messages, which greatly increases the channel contention and thereby reduces hit rates, while SCPS does not need the periodical location reporting.

In addition, we see that MOPS exhibits sharp decrease in the object hit rate as the query rate increases. When the brokers of different communities meet each other, due to limited meeting time, they only can exchange limited number of messages. Thus, higher query rate produces more undelivered messages, which are dropped when the test completes, leading to lower object hit rate. These experimental results confirm that SCPS has very high object hit rate even at high query rates, which is comparable to reporting based methods.

4.3.3 Overhead

Figure 7(c) plots the total overhead of the six methods with different query rates. The varying ranges of the total overhead of SCPS, SCPSw/oP and Snoogle are [28000, 484323], [49789, 588306] and [25289, 361349], respectively. Their average total overhead for different query rates are 216214, 260750 and 166302, respectively. We find that the overhead of SCPSw/oP is higher than SCPS. This is because nodes in SCPSw/oP need to periodically report their locations to nearby BSs, while SCPS does not have this requirement due to its capacity of location prediction. It is interesting to see that, though also using location reporting, Snoogle has lower overhead than SCPSw/oP and SCPS. This is due to two reasons. 1) Snoogle has high location query drop rate, which reduces the number of hops traversed by messages in multi-hop transmission. 2) Comparing with Snoogle, SCPSw/oP and SCPS need extra overhead in the DHT layer for looking up the locations of object owners.

We observe that the varying range of the total overhead of MOPS is [1800, 7900], and its average total overhead for different query rates is 5675. The overhead of MOPS is much lower than the other methods. This is because the packets in MOPS are mainly transmitted by brokers carrying the messages instead of hop-by-hop transmission. For Statistic and SCPSw/oS, the total overhead increases from 43000 to 729161, and from 39000 to 639161, respectively. Their average total overheads for different query rates are 316055 and 281073, respectively. We can find that the overhead follows Statistic>SCPSw/oS>SCPS. That is because the location accuracy of the methods follows Statistic<SCPSw/oS<SCPS. Packets with wrong locations result in many "hanging packets", thereby increasing overhead. These observations verify that SCPS has relatively lower overhead compared with methods with similar location query drop rate and object hit rate.

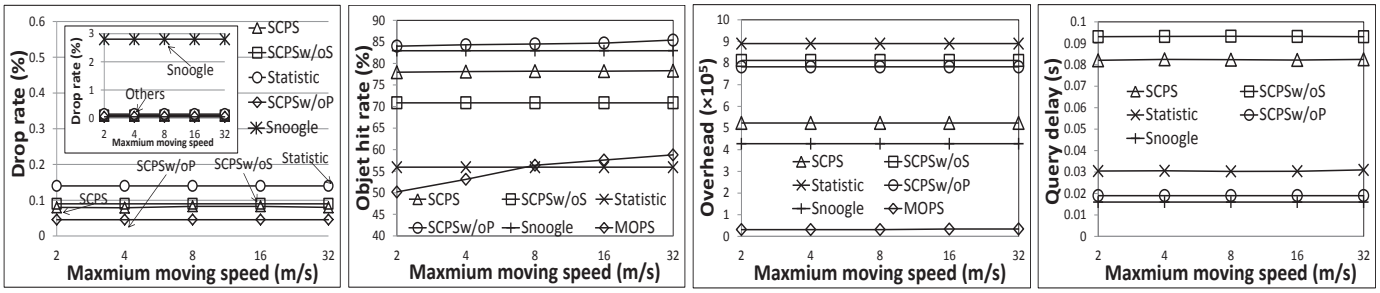
4.3.4 Query Delay

Figure 7(d) illustrates the average query delay of the six methods with different query rates. We find in our experiment result that the average delay of MOPS is much higher than the other methods (about 50s). Thus, we do not include MOPS in the figure. The reason for the high delay is because the inter-community communication is conducted only when brokers meet each other instead of using hop-by-hop transmission.

The varying ranges of the query delay of Snoogle, SCPSw/oP, Statistic, SCPS and SCPSw/oS are [0.02, 0.08], [0.02, 0.06], [0.03, 0.08], [0.08, 0.12] and [0.08, 0.14], respectively. Their average query delays for different query rates are 0.04, 0.03, 0.05, 0.10 and 0.11, respectively. We observe that Snoogle, SCPSw/oP and Statistic generate lower query delay than SCPS and SCPSw/oS. This is because the first three methods do not have the Bayesian network location prediction process, which takes about 0.0675s per prediction. We also see that Statistic produces a slightly higher query delay than Snoogle and SCPSw/oP. Statistic only uses routines and considers the location with the highest probability of a person as the prediction result. Thus, it may generate incorrect location prediction, which results in much longer delay for some packets in routing. Though most of the incorrect location packets are dropped, a few of them can arrive at the destination after long detour in the network, leading to larger average query delay. Without considering social events, SCPSw/oS has lower location prediction accuracy than SCPS, thus it generates higher average query delay than SCPS.

As the query rate increases, the query delay of Snoogle increases faster than SCPSw/oP ([0.02, 0.08] versus [0.02, 0.06]). This is because the centralized KeyIP has a long queue when the query rate is high. We observe that the query delays of all methods grow

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.



(a) Location query drop rate. (b) Object hit rate. (c) Overhead. (d) Query delay.

Fig. 8. Performance of object search with different node moving speeds.

as the query rate increases due to network congestion generated by more queries.

The above observations show that SCPS achieves the lowest location query drop rate, comparatively high object hit rate as the reporting based methods, the second to the lowest overhead, and relatively low query delay. Though Snoogle generates the lowest overhead and high object hit rate, its location query drop rate is extremely high.

4.4 Performance With Different Node Moving Speeds

In this test, we examined the performance of SCPS and other methods at different node moving speeds. The moving speed is randomly chosen from $[0,2^x]$ m/s, and x is varied from 1 to 5 with 1 increment in each step.

4.4.1 Location Query Drop Rate

Figure 8(a) shows the average location query drop rate of SCPS and other four methods. We observe that the drop rates of all methods remain nearly constant as speed increases. Specifically, the drop rates of Statistic, SCPSw/oS, SCPSw/oP, SCPS and Snoogle remain at around 0.14%, 0.09%, 0.05%, 0.08% and 2.80%. This result implies that the node moving speed does not affect the location query drop rate. Essentially, this is due to the reason that the geographic routing can successfully deliver a message to the node that is the nearest to the destination regardless of the node moving speed. The factors affecting the location query drop rate are the traffic congestion and the node query load.

From the small figure, we see that the location query drop rate of Snoogle is significantly higher than the other methods at different node moving speeds. This is due to the centralized scheme of Snoogle as explained for Figure 7(a). From the large figure, we see that in the decentralized methods, the location query drop rate follows Statistic>SCPSw/oS>SCPS>SCPSw/oP. This is because that the location accuracy of the methods follows Statistic<SCPSw/oS<SCPS<SCPSw/oP. Low prediction accuracy generates "hanging packets", which results in severe channel contention and network congestion, increasing query drop rate.

4.4.2 Object Hit Rate

Figure 8(b) shows the average object hit rate of the six methods at different node moving speeds. The average object hit rates of Statistic, SCPSw/oS, SCPS, Snoogle and SCPSw/oP are 55.94%, 70.86%, 78.15%, 82.95% and 84.59% respectively. The average object hit rate of MOPS increases from 50.18% to 58.79% when the maximum moving speed increases from 2m/s to 32m/s. From these results, we can make three observations. First, SCPS and other geographic routing based methods produce almost constant object hit rates with different mobility speeds due to the same reason as in Figure 8(a). Second, the object hit rates of the methods still follow Statistic<SCPSw/oS<SCPS<Snoogle<SCPSw/oP due to the same reason as Figure 7(b). Third, unlike other methods that produce constant object hit rates, the object hit rate of MOPS increases as the node moving speed increases. It is lower than Static at low node moving speeds and then higher than Static at high node moving speeds. With low node moving speeds, brokers in MOPS have fewer chances to meet each other, leading to lower object hit rate. Higher node movement speeds increase the meeting chances of brokers, thus generating higher object hit rate.

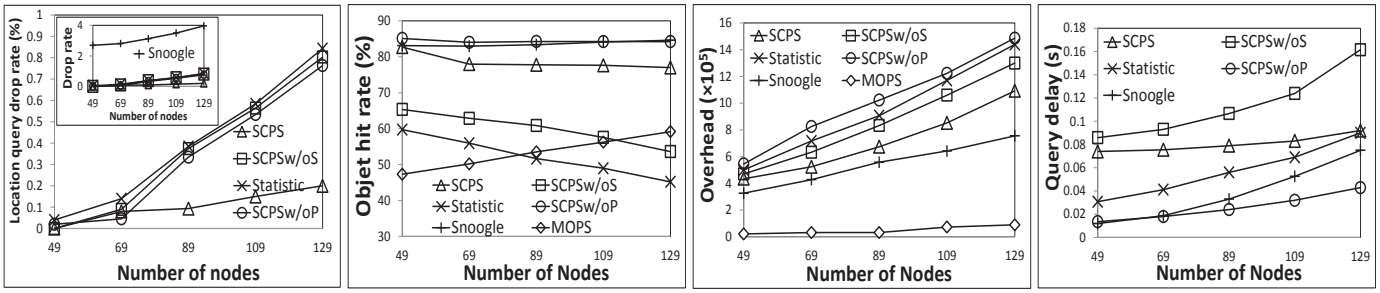
4.4.3 Overhead

Figure 8(c) plots the average overhead of the six methods with different node moving speeds. The average overheads of Snoogle, SCPS, SCPSw/oS, SCPSw/oP, Statistic and MOPS remain at around 42773, 52364, 81214, 78273, 89000 and 3247, respectively. First, we find that all methods have almost constant overheads at different node moving speeds. This is due to the adoption of the geographic routing which always forwards a message along the geographically shortest path, as we explained in Figure 8(a). We also see that the average overhead of MOPS falls in the range [3136, 3426]. MOPS does not have an increase in overhead even though its object hit rate increases as the moving speed grows. This is because the messages in MOPS are carried by brokers, and the maximum hops for one successful delivery is always no more than 2. We can observe that the overheads of the methods still follow the order of MOPS<Snoogle <SCPS<SCPSw/oP≈SCPSw/oS<Statistic due to the same reason as Figure 7(c).

4.4.4 Query Delay

Figure 8(d) illustrates the average query delay of the six methods with different node moving speeds. The average query delays of Snoogle, SCPSw/oP, Statistic, SCPS, SCPSw/oS stay around 0.016, 0.019, 0.031, 0.082 and 0.093, respectively. The average delays of SCPS and other methods remain almost constant over different moving speeds. This is also because the packet forwarding in geographic routing is not affected by the moving speeds of nodes. Also, we can observe

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.



(a) Location query drop rate. (b) Object hit rate. (c) Overhead. (d) Query delay.

Fig. 9. Performance of object search with different network sizes.

that the delays of the methods still follow the order of Snoogle≈SCPSw/oP<Statistic<SCPS<SCPSw/oS due to the same reason as Figure 7(d).

The above experimental results with different node moving speeds show that SCPS is dynamism-resilient. It generates low location query drop rate, high object hit rate, low overhead without compromising query delay at different node moving speeds.

4.5 Performance With Different Network Sizes

In this test, we examined the performance of SCPS and other methods at different network sizes. We varied the network size by adding different number of random nodes that send queries. The number of nodes we added is ranged from 30 to 110 with 20 increase in each step.

4.5.1 Location Query Drop Rate

Figure 9(a) shows the average location query drop rate of SCPS and other four methods with different network sizes. From the small figure, we see that the location query drop rate of Snoogle increases rapidly (from 2.69 to 3.97) as the network size increases, because the centralized scheme and periodical location reporting make Snoogle vulnerable to congestion. When there are more nodes launching queries and reporting locations at the same time, the KeyIP becomes congested, or the interference around KeyIP becomes too severe for it to receive queries correctly. SCPS and other decentralized methods do not have this problem, since the queries are sent to different BSs. Also, we find that the drop rates of Statistic, SCPSw/oS and SCPSw/oP increase much faster than SCPS as the network size increases. Specifically, the varying ranges of the average location query drop rates of SCPS, SCPSw/oP, SCPSw/oS and Statistic are [0, 0.2], [0.02, 0.76], [0, 0.80] and [0.04, 0.84], respectively. In Statistic and SCPSw/oS, the increase of the number of queries leads to the increase of "hanging packets" (as explained for Figure 7(a)) and the traffic congestion and load on BSs. In SCPSw/oP, nodes need to send a large number of location reporting messages to the BSs. Then, the congested and overloaded BSs make the packets with correct locations have fewer chances to be successfully delivered to BSs, leading to sharp growth of drop rates. The above observations confirm the low location query drop rate and high scalability of SCPS.

4.5.2 Object Hit Rate

Figure 9(b) shows the average object hit rate of the six methods with different network sizes. The varying ranges of the average hit rates of SCPSw/oP, Snoogle, SCPS, SCPSw/oS, Statistic, MOPS are [85, 84], [85, 83], [83, 77], [65, 54], [60, 45], [59, 47] when the number of nodes increases from 49 to 129, respectively. First, we find that SCPSw/oP, Snoogle and SCPS achieve high object hit rates. This is because they generate high prediction accuracy, so that the geographical routing can deliver the messages to the destination with high probability. Also, SCPS generates slightly lower hit rate since it uses location prediction rather than periodical reporting, which generates higher accurate prediction but also a large number of messages. We can also see that the object hit rate of SCPS, SCPSw/oS and Statistic drop as the network size increases. This is because more requests generate more incorrect location predictions, leading to higher traffic congestion in the perimeter mode routing in the geographic routing. As the network size scales up, the object hit rate of SCPS decreases at a much slower speed than SCPSw/oS and Statistic, because it has higher prediction accuracy than SCPSw/oS and Statistic. Statistic has low prediction accuracy which leads to "hanging packets", and SCPSw/oP produces many messages due to periodical reporting.

In addition, we find that the object hit rate of MOPS increases as the network size increases. The network area size is fixed in the simulation. As the number of nodes increases, the density of nodes increases. Therefore, the opportunity of nodes meeting each other increases, leading to higher object hit rate. These observations confirm that SCPS can maintain a high object hit rate in different network scales.

4.5.3 Overhead

Figure 9(c) plots the average overhead of the six methods with different network sizes. When the number of nodes increases from 49 to 129, the varying ranges of the average overheads of MOPS, Snoogle, SCPS, SCPSw/oS, Statistic, SCPSw/oP are [2098, 8904], [32555, 75520], [43438, 109203], [47000, 130000], [49999, 143900] and [54495, 148621], respectively. Their average values for different network sizes are 4908, 54165, 71514, 85904, 94639 and 102041, respectively. The average overhead follows MOPS < Snoogle < SCPS < SCPSw/oS < Statistic < SCPSw/oP. From the experimental results, first, we find that the overhead of SCPSw/oP is the highest and increases very fast. This is because it has both the overhead in the DHT layer and in location reporting. Snoogle only has the reporting overhead, while SCPS and other DHT based methods only have the DHT overhead. Second, we observe that the overheads of other prediction based methods are higher than SCPS. This is because as the network size increases, there will be more "hanging packets" resulted from the packets

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

with wrong locations. MOPS always produces the lowest overhead in different network sizes because its message delivery path length is always more than 2.

4.5.4 Query Delay

Figure 9(d) illustrates the average query delay of the six methods with different network sizes. The average query delays of SCPSw/oP, Snoogle, Statistic, SCPS, SCPSw/oS fall in the range [0.01355, 0.042838], [0.012, 0.075], [0.030758, 0.09], [0.073942, 0.092], [0.085885, 0.161512], respectively. Their average values for different network sizes are 0.03, 0.04, 0.06, 0.08 and 0.11. We observe that the query delay of Snoogle exhibits a rapid increase, while the delay of SCPS and SCPSw/oP maintain at a relatively stable level. The reason for Snoogle's sharp performance decrease is the same as Figure 7(d) that the central KeyIP easily becomes congested. However, SCPS and SCPSw/oP do not have this problem due to their adoption of distributed structure and high prediction accuracy. Also, it is interesting to see that the delay of SCPSw/oS and Statistic also increase rapidly as the network size increases though they use a distributed method. This is because their low prediction accuracy leads to more incorrect location responses, which increases the delay in the network due to the same reason as in Figure 7(d). Also, we can still see that the delays of the other methods follow SCPSw/oP<Statistic<SCPS<SCPSw/oS. Though the delays of those methods increase as the network size scales up, the relative performance of the methods remains the same. The reason for this result is the same as in Figure 7(d). These observations confirm that SCPS has high scalability and efficiency in terms of overhead and query delay.

5 Future Study

In order to use SCPS to implement real world applications we will need to address the following issues.

5.1 Naming and Disambiguation

A key feature of the system is the reference to objects by names in the form of user A has object 1, 2 and 3. Some applications may need a search engine that can locate unique objects belonging to a unique user. The issue of creating a unique ID to the object can be complex especially if the user or object names are not unique since the same user may have multiple objects with the same general category name like car keys. Even if the system creates a unique ID for the object, the user may prefer to resort to more descriptive names rather than referring to the object by its unique ID. The system therefore needs to include a facility for disambiguating names using reasoning and contextual information.

5.2 Date and Time based Reasoning

Like other social network based works, the system currently uses knowledge and heuristics about people's routine activities, and schedules and assumes a daily pattern. However, due to a variety of factors including seasonal changes and holidays, this prediction may be wrong. Therefore, the system will need to be able to adjust its predictions based on rules and other knowledge about the users and their space and time context. For example, people often have routines that vary weekly, monthly, and even seasonally. A lot more data must be entered to the Bayesian network to model these types of routines.

6 Conclusion

In this paper, we propose a Social-aware distributed Cyber-Physical human-centric Search engine (SCPS) that provides a scalable, efficient and accurate search service for physical objects carried by moving people. SCPS consists of three components: *distributed hierarchical structure*, *social-aware Bayesian network prediction model* and *cyber-physical searching algorithm*. The distributed hierarchical structure builds BSs into a DHT-based overlay in the higher layer, and connects MNs and sensors to BSs in the lower layer. The BSs collect data from MNs and sensors, and organize data in the DHT for subsequent scalable and efficient searching. BSs employ social-aware Bayesian network prediction models to calculate the locations of the queried objects. By leveraging the node routine mobility pattern in the social network, SCPS does not need periodical location reporting to keep track of objects. Further, SCPS combines social network with the Bayesian network to capture the node mobility changes due to exceptional events, and thereby provides high prediction accuracy. The cyber-physical searching algorithm takes advantage of DHT functions to achieve scalable and efficiency object querying. Extensive real-trace driven simulation results are presented to show the efficiency and location prediction accuracy of SCPS in comparison with existing methods. The results demonstrate the enhancement of efficiency by leveraging social networks and eliminating the periodical location reporting, and the importance of integrating social network with the Bayesian network prediction model to enhance the accuracy of the location calculation in object searching.

In the future, in addition to the two aforementioned issues, we plan to study how to further improve the prediction accuracy by utilizing more information from social networks such as the contact frequency. We also plan to further explore the security issues in SCPS such as using the access control mechanism to protect user privacy, and using the encryption mechanism to avoid man-in-the-middle attacks.

Acknowledgements

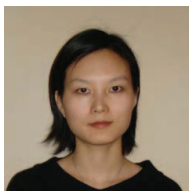
This research was supported in part by U.S. NSF grants CNS-1254006, CNS-1249603, CNS-1049947, CNS-1156875, CNS-0917056 and CNS-1057530, CNS-1025652, CNS-0938189, CSR-2008826, CSR-2008827, Microsoft Research Faculty Fellowship 8300751. We would like to thank Mr. Ze Li and Chenxi Qiu for their help on this work.

References

[1] E. A. Lee. Cyber physical systems: Design challenges. In *Proc. of IEEE ISORC*, 2008.
[2] E. A. Fry and L. A. Lenert. MASCAL: RFID tracking of patients, staff and equipment to enhance hospital response to mass casualty events. In *Proc. of AMIA Annual Symposium*, 2005.
[3] K. K. Yap, V. Srinivasan, and M. Motani. MAX: human-centric search of the physical world. In *Proc. of SenSys*, 2005.
[4] H. Wang, C. C. Tan, and Q. Li. Snoogle: A search engine for the physical world. In *Proc. of IEEE INFOCOM*, 2008.
[5] C. Tan, B. Sheng, H. Wang, and Q. Li. Microsearch: When search engines meet small devices. *Pervasive Computing*, 2008.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

[6] Wall street cheers iphone sales data. http://weblogs.baltimoresun.com/business/appleaday/blog/2007/09/wall_street_cheers_iphone_sale.html.
[7] M. C. Gonzalez, C. A. Hidalgo, and A. Barabasi. Understanding individual human mobility patterns. *Nature*, 2008.
[8] M. Mcpherson. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 2001.
[9] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *Service Assurance with Partial and Intermittent Resources*, 2004.
[10] J. Ghosh, S. J. Philip, and C. Qiao. Sociological orbit aware location approximation and routing (SOLAR) in MANET. *Ad Hoc Networks*, 2007.
[11] E. M. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proc. of MobiHoc*, 2007.
[12] F. Li and J. Wu. MOPS: Providing content-based service in disruption-tolerant networks. In *Proc. of ICDCS*, 2009.
[13] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *Proc. of MobiHoc*, 2008.
[14] A. Mtibaa, M. May, C. Diot, and M. Ammar. Peoplerank: social opportunistic forwarding. In *Proc. of INFOCOM*, 2010.
[15] B. Guo and M. Imai. Home-explorer: Search, localize and manage the physical artifacts indoors. In *Proc. of AINA*, 2007.
[16] G. Mao, B. Fidan, and B. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 2007.
[17] R. Peng and M. L. Sichitiu. Probabilistic localization for outdoor wireless sensor networks. *Mobile Comp. and Comm. Review*, 2007.
[18] H. C. Chu and R. H. Jan. A GPS-less, outdoor, self-positioning method for wireless sensor networks. *Ad Hoc Networks*, 2007.
[19] J. Bruck, J. Gao, and A. Jiang. Localization and routing in sensor networks by local angle information. *ACM Transactions on Sensor Network*, 2009.
[20] Y. M. Kwon, K. Mechitov, S. Sundresh, W. Kim, and G. Agha. Resilient localization for sensor networks in outdoor environments. *ACM Transactions on Sensor Networks (TOSN)*, 2010.
[21] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Proc. of SIGCOMM*, 2001.
[22] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. of SIGCOMM*, 2001.
[23] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. of Middleware*, 2001.
[24] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. *Tech. Rep. Comput. Sci. Div., Univ. California*, 2001.
[25] D. Guo, J. Wu, H. Chen, and X. Luo. Moore: An extendable peer-to-peer network based on incomplete kautz digraph with constant degree. In *Proc. of INFOCOM*, 2007.
[26] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proc. of MobiCom*, 2000.
[27] F. V. Jensen. *An introduction to Bayesian networks*. UCL press London, 1996.
[28] A. P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 1992.
[29] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in recursive graphical models by local computations. *Computational Statistics Quarterly*, 1990.
[30] L. Le Cam. Maximum likelihood an introduction. *ISI Review*, 1990.
[31] H. W. Sorenson. *Parameter estimation: principles and problems*, volume 382. Marcel Dekker New York, 1980.
[32] S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Comp. Statistics & Data Analysis*, 1995.
[33] G. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Proc. of UAI*, 1999.
[34] N. Wermuth and S. L. Lauritzen. Graphical and recursive models for contingency tables. *Biometrika*, 1983.
[35] T. Verma and J. Pearl. An algorithm for deciding if a set of observed independencies has a causal explanation. In *Proc. of UAI*, 1992.
[36] C. Taylor, A. Rahimi, J. Bachrach, H. Shrobe, and A. Grue. Simultaneous localization, calibration, and tracking in an ad hoc sensor network. In *Proc. of IPSN*, 2006.
[37] T. N. Chung, O. Camp, and S. Loiseau. A bayesian network based trust model for improving collaboration in mobile ad hoc networks. In *Proc. of RIVF*, 2007.
[38] Y. Wang and J. Vassileva. Trust and reputation model in peer-to-peer networks. In *Proc. of P2P*, 2003.
[39] Y. Wang and J. Vassileva. Bayesian network-based trust model. In *Proc. of WI*, 2003.
[40] R. Arroyo-Valles, A. G. Marques, J. J. Vinagre-Diaz, and J. Cid-Sueiro. A bayesian decision model for intelligent routing in sensor networks. In *Proc. of ISWCS*, 2006.
[41] K. S. Hwang and S. B. Cho. Modular bayesian network learning for mobile life understanding. In *Proc. of IDEAL*, 2008.
[42] T. Bruland, A. Aamodt, and H. Langseth. Architectures integrating case-based reasoning and bayesian networks for clinical decision support. In *Proc. of Intelligent Information Processing*, 2010.
[43] A. Rowe, M. Berges, G. Bhatia, E. Goldman, R. Rajkumar, L. Soibelman, J. Garrett, and J. M. F. Moura. Sensor andrew: Large-scale campus-wide sensing and actuation. *CMU Tech. Rep.*, 2008.
[44] Y. Li, L. Shou, and K. L. Tan. CYBER: A CommunitY-Based sEaRch engine. In *Prof. of P2P*, 2008.
[45] N. Eagle, A. Pentland, and D. Lazer. Inferring social network structure using mobile phone data. In *Proc. of PNAS*, 2007.
[46] The Network Simulator ns-2. http://www.isi.edu/nsnam/ns/.
[47] PlanetSim. http://projects-deim.urv.cat/trac/planetsim/.
[48] BNT, Bayes Net Toolbox for Matlab. http://code.google.com/p/bnt/.
[49] Joseph L. Hammond and Peter J.P. O'Reilly. *Performance Analysis of Local Computer Networks*. Addison-Wesley Publishing Company, 1988.
[50] Saeed Chahramant. *Fundamentals of Probability with Stochastic Process*. Western New England College, 1986.



Haiying Shen Haiying Shen received the BS degree in Computer Science and Engineering from Tongji University, China in 2000, and the MS and Ph.D. degrees in Computer Engineering from Wayne State University in 2004 and 2006, respectively. She is currently an Assistant Professor in the Department of Electrical and Computer Engineering at Clemson University. Her research interests include distributed computer systems and computer networks, with an emphasis on P2P and content delivery networks, mobile computing, wireless sensor networks, and cloud computing. She is a Microsoft Faculty Fellow of 2010 and a member of the IEEE and ACM.

Jinwei Liu Jinwei Liu received the BS degree in Computer Science from Guangxi University of Science and Technology, China in 2009, and the MS degree in Computer Science from Clemson University in 2012. He is currently a Ph.D. student in Department of Electrical and Computer Engineering of Clemson University. His research interests include wireless sensor networks, data mining and social network.

Kang Chen Kang Chen received the BS degree in Electronics and Information Engineering from Huazhong University of Science and Technology, China in 2005, and the MS in Communication and Information Systems from the Graduate University of Chinese Academy of Sciences, China in 2008. He is currently a Ph.D student in the Department of Electrical and Computer Engineering at Clemson University. His research interests include mobile ad hoc networks and delay tolerant networks.

Jianwei Liu Jianwei Liu received the BS degree in Software Engineering and MS degree in Communication and Information Systems from Xidian University, China in 2007 and 2010, respectively. He is currently a PhD candidate in the School of Computing at Clemson University. His research interests include mobile computing and visual computing.

Stan Moyer Stan Moyer is Vice President and Executive Director at Global Inventures, a company specializing in technology alliance management. He was Executive Director and strategic research program manager in the Applied Research area of Telcordia for over 20 years. He received an M.E. degree in electrical engineering from the Stevens Institute of Technology in 1990, an MBA in Technology Management from the University of Phoenix in 2004, and a B.S. degree in Engineering Physics from the University of Maine in 1987.