

# Search-and-Discover in Mobile P2P Network Databases<sup>1</sup>

Ouri Wolfson<sup>2</sup>, Bo Xu<sup>3</sup>, Huabei Yin<sup>3</sup> and Hu Cao<sup>3</sup>

## Abstract

*In this paper we propose a novel algorithm called Rank-Based Broadcast (RBB) for discovery of local resources in mobile P2P networks. With RBB, each moving object periodically broadcasts the most relevant resource reports and queries it knows to its neighbors, and the contribution is in determining how to rank the reports and queries in terms of their relevance, when to broadcast them, and how many to broadcast. A major difference between RBB and many existing algorithms in the resource discovery and publish/subscribe literature is that RBB does not rely on any pre-established routing structure, and therefore is able to adapt to both high mobility environments. In the paper we experimentally compare RBB with flooding and PSTree, a publish/subscribe algorithm for wireless ad-hoc networks. The results show that RBB by far outperforms the other two algorithms.*

## 1. Introduction

A MOBILE P2P NETWORK (MOPNET) is a set of moving objects that communicate with each other via unregulated, short-range wireless technologies such as IEEE 802.11, Bluetooth, or Ultra Wide Band (UWB). No fixed infrastructure is assumed or relied upon. MOPNET's are different from Mobile-Ad hoc NETWORKS (MANET's) in the sense that objects usually do not know the network-id of other moving objects and therefore do not perform routing in the traditional MANET sense. An important application domain of MOPNET's is local resource discovery. In a local resource discovery application (see e.g. [2, 1]), a user finds local resources that satisfy specified criteria. For example, a driver finds an available parking slot by receiving information generated by the parking meter, or a participant at a convention finds another participant with a matching profile.

The advantages of this method over a central database approach to discovering local information are: eliminating the need for an infrastructure, faster access, and more up to date local information.

The main problem in answering queries about resources in MOPNET's is finding the resource information, since the issuer of a query usually does not know the network-id of the moving objects storing the resource information, or how to reach them. In this paper we propose an algorithm called Rank Based Broadcast (RBB) for resource discovery in MOPNET's. In RBB the moving objects disseminate *resource-information* (namely reports) and *queries* by multi-hop transmissions. Each moving object can be a report producer, consumer (i.e. query issuer), or both.

MOPNET resource discovery arises in many application domains including social networks, transportation, mobile electronic commerce, emergency response, and homeland security. For example, in a large professional, political, or social gathering, the technology is useful to automatically facilitate a face-to-face meeting based on matching profiles. In transportation, the RBB algorithm incorporated in navigational devices can be used to disseminate to other similarly-equipped vehicles information about relevant resources such as free parking slots, traffic jams and slowdowns, available taxicabs, and ride sharing opportunities. In mobile electronic commerce, the RBB algorithm is useful to match buyers and sellers in a mall, or to disseminate information about a marketed product. In emergency response, the RBB algorithm can be used by first responders to support rescue efforts even when the fixed infrastructure is inoperative; it will match specific needs with expertise (e.g. burn victim and dermatologist), and help locate victims. In homeland security, sensors mounted on neighbouring containers can communicate and transitively relay alerts to remote check-points. Although it is unlikely that RBB in a particular device will need to manage data for all these applications simultaneously, it will probably need do so for multiple applications, e.g. mobile electronic commerce and social networks. Thus our experiments do not focus on a particular application, and are parameterized for some "average" application.

<sup>1</sup> Research supported by NASA contract NNA06AA25C, and by NSF grant 0209190.

<sup>2</sup> Pirouette Software Consulting, 1030 N. State St., Chicago IL 60610

<sup>3</sup> Department of Computer Science, University of Illinois at Chicago, 851 S. Morgan (M/C 152), Chicago, IL 60607  
{wolfson, boxu, hyin, hcao2}@cs.uic.edu

Resource-reports and queries correspond to resource descriptions and resource queries in the resource discovery literature, and events and subscriptions in the publish/subscribe literature. Existing literature on resource discovery in MANET's (see e.g. [2, 1]) addresses this problem using a *pull* approach. Specifically, a moving object floods a resource discovery query in the MANET, and when found the network constructs a routing structure from the query originator to the resource. The structure is built by augmenting traditional MANET routing protocols. Existing literature on publish/subscribe in MANET's (see [4, 9, 12]) also has queries (called profiles) and resources (called events), and the objective is to route the events to the matching subscribers. Again, a routing structure is built and maintained, but the events are *pushed* along the structure to the appropriate profiles, as they are produced. In short, both push and pull approaches use routing structures, and consequently they can be inefficient, particularly in high mobility networks which are prone to frequent topology changes.

The RBB algorithm is a hybrid approach between push and pull, in the sense that it disseminates both queries and resource reports. In this approach, a moving object  $O$  periodically selects the  $k-1$  most relevant reports in its database and its native query (the query  $O$  issues) and broadcasts them to its neighbors (i.e. the objects that are within the transmission range of  $O$ ). Upon receiving the broadcast from  $O$ , each neighboring object incorporates the received reports and queries into its own local database, and subsequently broadcasts the top  $k$  reports and queries. Thus reports and queries transitively spread across moving objects.

A novel aspect of the RBB algorithm is the strategy used by a moving object  $O$  to rank the reports based on their relevance. Intuitively, the relevance of a report depends on the queries in the local database (which represent the global demand in the network); the more queries it satisfies, the more relevant is the report.

Other novel aspects of the RBB algorithm are strategies for a moving object to determine the following questions. First, when to broadcast the top  $k$  reports and queries, and second, what is the value of  $k$  in each broadcast. Intuitively, RBB answers the first question by triggering a broadcast when new information can be communicated to the neighbors, either because enough new information was received or because the set of neighbors has changed. For answering the second question, in this paper we develop a formula to compute the optimal transmission

amount of each moving object<sup>1</sup>. Using this formula a moving object dynamically adjusts  $k$  based on the length of the time between subsequent broadcasts, such that overall report dissemination is maximized.

In summary, **the novel aspects of the RBB algorithm** are: 1) a hybrid approach between push and pull, in which both queries and resource descriptions (or reports) are disseminated, 2) a novel function to rank reports for the purpose of their dissemination in a MOPNET, 3) a heuristic to determine the broadcast frequency, and 4) based on this frequency and a newly developed mathematical formula, a method to compute the optimal broadcast size.

In order to evaluate the proposed RBB algorithm, we experimentally compared it with periodic flooding, in which a report is periodically flooded by its producer; and with PSTree, a publish/subscribe algorithm designed for wireless ad-hoc networks (see [4]). To put the results in perspective we also evaluated them against an ideal benchmark algorithm. We compared the algorithms in terms of the average number of relevant reports that are delivered to a consumer (i.e. the throughput). We compared the algorithms in high mobility (simulating a vehicular application) and low mobility (simulating a pedestrian application); and for different wireless technologies including 802.11 and Bluetooth. The experimental results show that RBB performs better than periodic flooding and PSTree. The advantage of RBB over PSTree and periodic flooding is greater in Bluetooth (which has a short transmission range) than in 802.11. The reason is that in Bluetooth the network is more prone to disconnection, and the caching of reports and queries that occurs at intermediate nodes in RBB (but not in the other algorithms) better copes with this situation. The throughput of RBB is about 50% of the ideal benchmark algorithm.

The rest of the paper is organized as follows. Section 2 introduces the model. Section 3 describes the RBB algorithm. Section 4 presents the experimental setup, and compares the RBB algorithm with the PSTree and flooding algorithms. Section 5 discusses relevant work, and section 6 concludes the paper.

## 2. Resources and Queries

Our system consists of a finite set of (point, i.e. without an extent) *moving objects*. Their motion occurs in a finite geographic space. Occasionally, a moving object  $O$  obtains information about a *resource* having

---

<sup>1</sup> Observe that if moving objects transmit too much, then many collisions would reduce the number of successfully received reports; and if they transmit too little, report dissemination would suffer.

some unique *resource-id*. For example, a pedestrian passing by an Automatic Teller Machine learns about the location and Bank of the machine from a Wi-Fi transmitter at the station. The information about a learnt resource  $R$  is represented by a *report*, denoted  $a(R)$ ;  $O$  is called the *producer* of that report.

Each moving object  $O$  also issues *queries* that express  $O$ 's interests in certain types of information.  $O$  is called the *issuer* of these queries<sup>2</sup>. An example of a query is a request for the locations of ATM's within 4 blocks of the Sears Tower in Chicago. The queries issued by a moving object  $O$  are *native* to  $O$ . A report  $a(R)$  either satisfies a query  $Q$ , or it does not do so. If it does then  $Q(a(R)) = true$  and  $match(a(R), Q) = 1$ . Otherwise  $Q(a(R)) = false$  and  $match(a(R), Q) = 0$ .

### 3. Rank-Based Broadcast Algorithm

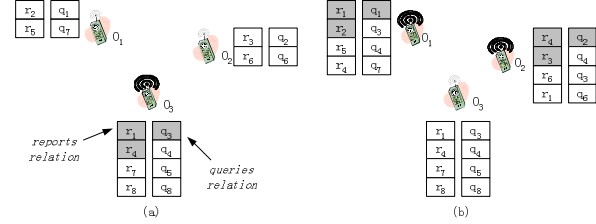
The objective of RBB is to maximize the number of reports delivered to the "right" moving objects, i.e., to the issuers of queries satisfied by these reports. For this purpose, both queries and resources reports are disseminated. The purpose of disseminating the queries is to reflect the demand in the network, such that the reports that are in demand are ranked higher. In other words, at any point in time a moving object's database has reports, the query it issues, and queries issued by other moving objects.

Intuitively, the algorithm works as follows. A broadcast is triggered when a moving object  $O$  receives enough new reports that ensure that new information is communicated to the neighbors, or when it travels for a long enough distance such that the moving objects within its transmission range have changed. In other words, a broadcast is triggered when new information can be communicated to the neighbors, either because new information was received or because the set of neighbors has changed.

When a broadcast is triggered,  $O$  computes what to broadcast as follows. It sorts its database based on a ranking function, and selects the  $k-1$  highest ranked reports and its native query to broadcast. The natural number  $k$  is determined based on the bandwidth capacity and the length of the time period since the last broadcast. The moving object  $O$  maintains a relation of reports and a relation of queries, and the broadcast issued by  $O$  includes the  $k-1$  highest ranked reports, and the native query. Fig. 1 gives an example of the RBB procedure.

The rest of this section is organized as follows. In subsection 3.1 we discuss the ranking of queries and reports in the RBB algorithm. In 3.2 we develop a

formula that computes the data dissemination throughput of a broadcast in a MOPNET. This formula serves as a basis for determining the broadcast size in the RBB algorithm. In 3.3 we present the RBB algorithm, and in 3.4 we explain it.



**Figure 1.** (a) Moving object  $O_3$  broadcasts its top two reports  $r_1$  and  $r_4$ , and native query  $q_3$ . (b) After receiving from  $O_3$ , object  $O_1$  incorporates the received items, and when its own broadcast is triggered it re-ranks reports, and broadcasts the top two (shadowed) reports and its native query. A similar procedure is executed by object  $O_2$ .

#### 3.1 Ranking of Reports

The rank of a report is defined to be sum of all its relevance values. Let *foreign* queries in moving object  $O$ 's queries relation be queries that are received from other objects, i.e. non-native queries.

**Definition 3:** Let  $Q_1, Q_2, \dots, Q_n$  be the foreign queries in moving object  $O$ 's queries relation. The *rank of a report*  $a(R)$  at  $O$  at time  $t$  is:

$$rank(a(R)) = \sum_{i=1}^n match(a(R), Q_i) \quad (1)$$

Notice that the native queries do not participate in computation of report rank, since they should not increase the probability of a report being propagated.

For simplicity of presentation we assume that all the reports and queries have the same size, denoted  $M$  (although our results easily extend to the different-size case) and that an infinite number of queries and reports fit in the memory of a moving object (to avoid dealing with trivial memory allocation issues).

#### 3.2 Throughput of a Broadcast in a MOPNET

Consider a broadcast of  $k$  reports by a moving object  $x$ . In this subsection we develop and optimize a formula for the expected number of reports that are received without interference (collision) at a neighbor of  $x$ . We target MOPNET's that use a carrier-sense multiple access (CSMA) protocol, e.g. 802.11. In such a network time is divided into slots, moving objects communicate by broadcasts, and each broadcast lasts an integral number of time slots. For example, the length of the 802.11b time slot is  $20\mu s$ .

<sup>2</sup> Note that  $O$  can be both a report producer and a report consumer at the same time.

The reception of a broadcast from a moving object  $x$  at a neighbor  $y$  is said to be *successful* if  $y$  receives the message from  $x$  without incurring any interference generated by transmissions from other neighbors of  $y$  during the broadcast. If another neighbor of  $y$  transmits during some time slot of the broadcast, then a *collision* occurs, and the whole broadcast is considered corrupt at  $y$ . We assume that  $p$  is the probability that a moving object attempts to start transmitting at an arbitrary time slot by sensing the channel. And we assume that  $p'$  is the probability that it actually succeeds, namely that the channel is free at that time. Clearly,  $p' \leq p$ .

Denote by  $k$  the number of reports/queries in the broadcast. Let  $F$  be the number of neighbors that successfully receive the message from  $x$ . The *throughput of the broadcast by  $x$*  (denoted  $Th$ ) is defined to be:  $Th = k \cdot F$ . Intuitively, the throughput is the total number of reports successfully received by neighbors of  $x$ . Remember that all the reports and queries have the same size, denoted  $M$ .

**Table 1.** Summary of symbols used in computing the throughput.

Symbol	Meaning
$\lambda$	The number of moving objects per each unit of the MOPNET area (we assume uniform spatial distribution).
$r$	Transmission range of each moving object in meters.
$b$	Data transmission speed in bits per second.
$Th$	Throughput of a broadcast.
$k$	Number of reports in each broadcast.
$p$	The probability that a moving object attempts (by sensing the channel) to start a broadcast at an arbitrary medium access time slot.
$p'$	The probability that a moving object starts a broadcast at an arbitrary medium access time slot (i.e. the channel was free).
$\tau$	Length of the medium access time slot in seconds.
$h$	Size of Medium Access Control header in bytes.
$M$	The size of each report/query in bytes.

**Theorem 1:** Let a moving object  $x$  execute a broadcast at an arbitrary time slot. Under the assumptions and notations given in Table 1,  $Th$ , the throughput of the broadcast is a random variable with an expected value:

$$\begin{aligned}
& 2 \cdot \pi \cdot \lambda \cdot k \cdot \int_0^r \delta \cdot (1-p)^{\lambda \cdot r^2 \cdot (2 \cdot q(\frac{\delta}{2r}) + (\pi - 2 \cdot q(\frac{\delta}{2r}))(2T+1))} d\delta \\
& \leq E(Th) \leq \\
& 2 \cdot \pi \cdot \lambda \cdot k \cdot \int_0^r \delta \cdot (1-p')^{\lambda \cdot r^2 \cdot (2 \cdot q(\frac{\delta}{2r}) + (\pi - 2 \cdot q(\frac{\delta}{2r}))(2T+1)) - 1} d\delta
\end{aligned} \tag{2}$$

where  $T = (M \cdot k + h) \cdot 8 / b \cdot \tau$ , and for a fraction  $a$ ,  $q(a) = \arccos(a) - a\sqrt{1-a^2}$   $\square$

Theorem 1 gives the lower and upper bounds of the expected number of reports that are received by the neighbors of  $x$  without interference. It can be shown based on [15] that the ratio between the two is close to 1. Therefore, from now on we take the upper bound to be the expected throughput of a broadcast.

Basically, the contribution of Theorem 1 is that it expresses  $F$  in the definition of the throughput in terms of the environmental parameters given in Table 1, i.e.,  $M, h, b$ , etc.

Now consider Eq. (2). If  $\tau, p', \lambda, h, b, M$ , and  $r$  are fixed, then the throughput  $Th$  as a function of the broadcast size  $k$  is a bell curve. Intuitively, when  $k$  is very small, the throughput of the broadcast is small because the wireless channel is underutilized. As the broadcast size increases, the wireless channel is better utilized. But in the meantime the probability of collisions becomes higher, because the broadcast does not use handshakes to avoid or detect collisions as unicast does. Thus there is a value of  $k$  that maximizes the throughput, i.e. achieves the best tradeoff between the channel utilization and broadcast reliability. And this value is computed and used by the RBB algorithm given in the next subsection.

For the rest of this subsection we show that indeed, except for  $k$ , all the parameters of Equation (2) can be determined by a moving object. The parameters  $\tau, h, r$ , and  $b$  depend on the network, and are fixed for a given communication network technology. For example,  $h$  is 47 in 802.11b.  $M$  is fixed (see sec. 3.1). The density  $\lambda$  can be determined by an object  $O$  in various ways. For example, each moving object periodically handshakes with each one of its neighbors and counts the number of neighbors, or  $O$  has a pre-loaded table in which each entry gives the object density at each geographic area at each time period (e.g. rush hour).

The probability  $p'$  is determined as follows. If a moving object starts a broadcast every  $c$  seconds on average, then its broadcast probability in each medium access time slot is  $\tau/c$ . Thus we substitute the broadcast probability  $p'$  in Equation (2) by  $\tau/c$ . For example, if  $c=5$  seconds and  $\tau=20\mu s$ , then  $p'=(20 \times 10^{-6})/5=4 \times 10^{-6}$ .

### 3.3 Description of the RBB Algorithm

At each time unit, a moving object  $O$  executes RBB to determine (i) whether it should broadcast at that time unit, and if so (ii) what to broadcast. Denote by  $|S|$  the cardinality of a set  $S$ . The formal description of the RBB algorithm is given below.

<b>Algorithm:</b> RBB algorithm at moving object $O$
<b>Input:</b> The <i>trigger threshold</i> , $g$ indicates the value of a function beyond which a broadcast is triggered. The

function is defined in step 3.
0. Check if any reports or queries were received in the last time unit. For each received report, if it is not already in $O$ 's database, save it there. For each received query, check for duplicates, and keep only the copy with the smallest rank.
1. Compute $k$ , the number of reports in the current broadcast, using Equation (2).
2. Rank the reports in the reports database using Equation (1). Denote by $S$ the set of top $k-1$ reports.
3. Compute $d$ , the distance between $O$ 's current location and its last broadcast location <sup>3</sup> . Compute $S-S_0$ , where $S_0$ denotes the set of reports in the last broadcast of $O$ . A broadcast is triggered if $d/2r +  S-S_0 / S  > g$ , where $g$ is the <i>trigger threshold</i> and $r$ is the transmission range.
4. If a broadcast is triggered, $O$ broadcasts the sets $S$ and its native query <sup>4</sup> .

### 3.4 Explanation of the RBB Algorithm

First note that the RBB algorithm takes  $g$  as the parameter, and the performance depends on value of  $g$ . Thus, the optimal value of  $g$  will have to be determined before running the algorithm in a real environment. We discuss the tuning of this parameter in section 4.3. Now we discuss steps 1 and 3 of the algorithm.

**Step 1.** This step computes the broadcast size  $k$ , i.e. the total number of reports and queries that can be included in the broadcast if the broadcast is triggered. It is dynamically adapted to the length of the time period since the last broadcast of moving object  $O$ . Equation (2) developed in subsection 3.2 is used to adaptively determine the broadcast size  $k$  as follows. Let  $c$  be the number of seconds since the completion of the last broadcast of  $O$ . As explained in 3.2, using  $p' = \tau/c$  we obtain a formula for the throughput in which the only unknown is  $k$ . Thus we can find the *optimal*  $k$ , i.e. the value of  $k$  for which  $Th$  is maximized; we take this value as the broadcast size.

Observe that in this fashion the broadcast size is dynamically and adaptively adjusted to the length of time between subsequent broadcasts; as the length of the time period  $c$  increases, so does the optimal  $k$ .

<sup>3</sup> This assumes that  $O$  is equipped with a positioning device such as GPS. If not,  $d$  can be estimated by some function of time-since-the-last-broadcast  $t$ , e.g.,  $t \times speed$ .

<sup>4</sup> Observe that there is no handshake, and  $O$  does not incur the overhead of checking the identity of its neighbors. Such overhead includes acknowledgements of neighbors, and the collisions and delays that such acknowledgements would generate. Indeed, the 802.11 protocol specifies an unreliable (rather than reliable) broadcast.

Furthermore, the broadcast size  $k$  can be adjusted to a *bandwidth allocation* parameter. For example, a user may allocate only 10% of the available short-range bandwidth to mobile P2P resource discovery (the rest may be used for internet access, videos download, etc.). Then the broadcast size is taken to be only 10% of the optimal  $k$ .

**Step 3.** This step evaluates the broadcast trigger based on the distance traveled and the change to the database since last broadcast. The general idea is to reduce duplicate transmissions by disseminating only new data to old neighbors (old neighbors, are neighbors that were within transmission range at the last broadcast also) and old data only to new neighbors. Observe that RBB performs a broadcast without a handshake, thus it has no knowledge of who are its neighbors. Thus it is assumed that when the moving object has traveled a distance of  $2r$  or more, its set of neighbors is new (because if all the neighbors were static, then none of them would be a neighbor anymore). Thus the ratio between  $d$  and  $2 \cdot r$  indicates the fraction of new neighbors among all the neighbors of  $O$ . The fraction  $|S-S_0|/|S|$  indicates how much of the broadcast content is new.

## 4. Comparison with PSTree and Flooding

In this section, we compare RBB with three algorithms: 1) periodic flooding, 2) Publish/Subscribe Tree (PSTree), a publish/subscribe algorithm designed for wireless ad-hoc networks (see [4, 9]), and 3) an ideal algorithm that instantaneously delivers reports to the right queries, used for benchmark purposes. First we describe the algorithms (4.1), and then we present the simulation method (4.2) and its results (4.3).

### 4.1 Data Dissemination Algorithms

**Periodic Flooding.** Each report is periodically flooded by its producer. At each round of flooding, the producer broadcasts the report to all neighbors. Each neighbor that receives the report in turn immediately rebroadcasts the report exactly once. Thus the report spreads transitively. A moving object that receives the report and has already broadcast this report during the current round, does not take any action. In the simulations, the flooding period is fixed to be 1 second. In other words, each moving object floods the latest report which is assigned to it every second; increasing the flooding period decreases performance.

**PSTree.** In a PSTree system one moving object is designated as the *root*. Every newly generated report is passed to the root first. Then the root disseminates the report along a multicast tree. In order to maintain the

tree, periodically each moving object  $O$  searches its neighbors to look for the “best” potential parent. Particularly, each one of  $O$ ’s neighbors estimates the overhead of having  $O$  as its child, and the one with the minimum overhead is selected as the parent of  $O$ .

In our simulation system for the PSTree algorithm, initially an arbitrary moving object is selected as the root. When the life span of the root expires, another arbitrary moving object currently in the system is selected as the root. Once a report is generated, it is sent to the root instantaneously, and then disseminated by the root immediately. For each parameter configuration, we tuned up PSTree with different tree update periods ranging from 1 to 2000 seconds. The period that had the best performance (see section 4.2.4) is used for that parameter configuration.

**Discussion.** RBB, flooding, and PSTree are variants of gossiping for mobile environments. Due to disconnections and changing network topology, none of the algorithms guarantees delivery of each resource report to the relevant queries. Collisions further hinder delivery. Thus, these are “best effort” algorithms.

**Benchmark.** In order to put the performance of the algorithms in perspective, we also define an *ideal benchmark* offline algorithm that has complete knowledge and delivers reports instantaneously. Thus, once a moving object enters the system it immediately receives all the reports that have ever been generated in the system, and it instantaneously receives every report that is generated during its life span. This corresponds to an installation which has high performance centralized databases and numerous hotspots (through which these databases are accessed from everywhere).

## 4.2. Simulation Environment

**4.2.1. Mobility Model.** In the simulations, the objects move according to the *random way-point mobility* model. In particular, at the start time of the simulation  $n$  moving objects are randomly placed in a 0.5mile×0.5mile square area. Each object has a destination which is another random point in the space. The object moves to its destination at a constant speed  $s$  along the straight line. After it arrives, it pauses at the destination for a period  $\Delta t$ , randomly chooses another destination, and repeats this procedure. The pause time  $\Delta t$  is uniformly distributed between 120 and 240 seconds. The motion speed  $s$  is randomly picked up from the interval [0.5, 1.5] miles/hour (modeling the pedestrian scenario). This mobility model simulates, for example, the motion of a shopper at a mall. The values of  $n$  (number of objects) are listed in Table 2.

The whole simulation runs for 3600 time units where each time unit is a second. Each object has a life

span which is set to be a random period between 300 and 900 seconds. When the life span of an object expires, it is removed from the system and a new object is created at a random point in space. Thus the number of *live* moving objects is fixed.

**4.2.2. Generation of Resources and Queries.** For representing resources and queries, we adopted the Number Intervals (NI) subscription model introduced in [4]. Particularly, a resource is represented by a point within the real interval [0, 1]. A query is represented by a range within [0, 1], e.g., [0.2, 0.7]. A report  $a(R)$  matches a query  $Q$  if  $R$ ’s number falls into  $Q$ ’s range.

Resources are generated by a Poisson process with intensity 1, i.e., on average one resource is generated every second. The number that represents a resource  $R$  is randomly picked up from the [0, 1] interval. An arbitrary live moving object becomes the producer of the report  $a(R)$ .

Each moving object has a query, which is generated when the object is introduced to the system, and is fixed for life span. The range of the query is generated by picking a center and a length. The length of the range is selected randomly according to a normal distribution with mean 0.05 and variance 0.002. The query-center falls into the [0, 1] interval following a Zipf distribution. In particular, the [0, 1] interval is divided into 10 disjoint sections ([0, 0.1], [0.1, 0.2], ...). The probability that a query-center falls into the  $i$ -th ( $1 \leq i \leq 10$ ) section is  $(1/i) / \sum_{j=1}^{10} (1/j)$ . In other words, the resources are uniformly distributed, and the queries are distributed according to Zipf’s law.

**Table 2.** Variable system parameters and their values

Parameter name	Symbol	Unit	Values
Number of objects	$n$		100, 250, 500, 750, 1000
Trigger threshold	$g$		0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 2, 3, 4, 5
Transmission range	$r$	meter	10 (Bluetooth), 50 (802.11)
Bandwidth allocation	$a$		0.01, 0.05, 0.1, 0.3, 0.5, 0.7, 0.9

**4.2.3. Communication Model.** We assume that each moving object allocates only a fraction  $a$  of the available short-range bandwidth to mobile P2P resource discovery.  $a$  is a parameter called the *bandwidth allocation* and defined in section 3.4, step 1. Then in RBB the broadcast size is taken to be the fraction  $a$  of the optimal  $k$ . In a simulation for RBB, the RBB algorithm is executed every 10 seconds.

Our simulation system omits detailed representation of protocol layers and radio propagation. It models the reliability of communication affected by signal collisions, using the analytical model introduced in

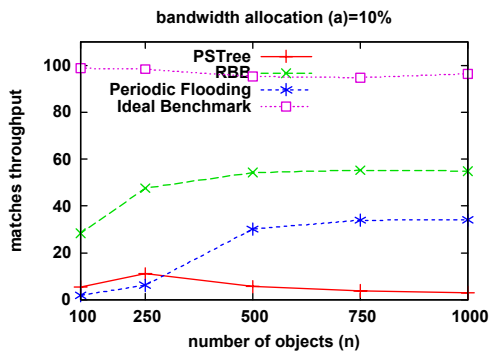
3.2. The other values of the parameters used in the simulation (see Table 2) are as follows. The data transmission speed ( $b$ ) is 1M bits/second. The transmission range ( $r$ ) is either 50 meters or 10 meters. We refer to the 50 meters case as the 802.11 case since 50 meters is a typical transmission range for 802.11. Similarly we refer to the 10 meters case as the Bluetooth case<sup>5</sup>. The size of a report or a query ( $M$ ) is 1000 bytes. This includes the common attributes (type, location, timestamp) and the application specific data (e.g. product description, personal profile, etc.).

**4.2.4 Performance measure.** We consider the average number of matching resource reports received by a moving object. This is called the *matches throughput*. In other words, the matches throughput is the average number of reports matching the native query that are delivered to a moving object. Observe that this follows the publish/subscribe literature, in which a subscriber requests *all* the reports that match the native query.

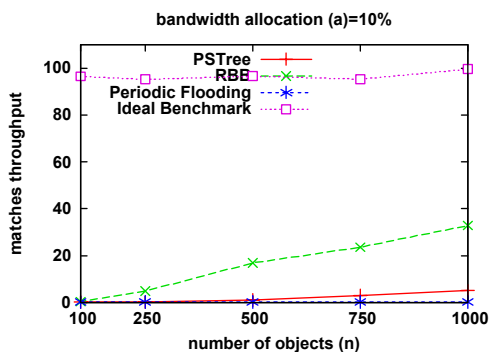
### 4.3. Simulation Results

**4.3.1. RBB Tuning-up.** We tuned up RBB by finding the trigger threshold  $g$  that generates the highest matches throughput. For example, when  $n=750$ , RBB reaches the best performance when  $g=0.1$ . In this case, the simulations reveal that on average 32 reports are transmitted in each broadcast, and the average length of the time period between two successive broadcasts is 20 seconds. Similarly, we tuned up RBB for Bluetooth. The best performance of each configuration is then used when comparing with other algorithms for the same configuration.

**4.3.2. Comparison among RBB, Periodic flooding, PSTree, and Ideal Benchmark** (Figures 2-4). In all the cases RBB performs far better than periodic flooding and PSTree on matches throughput. In many cases RBB outperforms the other two by an order of magnitude (see Figure 2). In the PSTree case, mobility causes the tree structure to disconnect. And if the connection between a node and the root is broken, then the whole subtree under the node will not receive new reports. In order to keep the tree connected, the tree update period has to be short. But the short update period increases collisions and loss of messages.



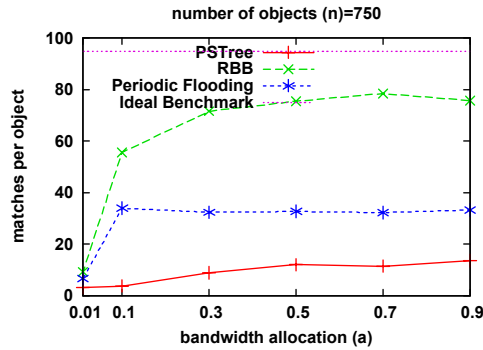
**Figure 2.** Performance versus number of objects (802.11)



**Figure 3.** Performance versus number of objects (Bluetooth)

In periodic flooding a moving object does not dynamically adjust the flooding period. Either the period is too long, which leaves the network idle, or if the period is too short, then there are a lot of collisions. In either case, the effective communication capacity is not optimized. Furthermore, periodic flooding does not smartly accumulate reports into a single broadcast. In other words, each broadcast consists of a single report. And our analytical model for data dissemination indicates that sending  $n$  reports in a single broadcast generates less contention than sending  $n$  broadcasts of a single report each. RBB, on the other hand, adjusts the broadcast period and the broadcast size dynamically such that the effective communication is optimized. Furthermore, in RBB the dissemination of queries and their ranking enables a moving object to broadcast only the most promising reports; this selectivity further increases bandwidth utilization.

<sup>5</sup> The traditional Bluetooth is not a CSMA network, but in this paper we use the term to distinguish between the 10 from the 50 meters transmission range.



**Figure 4.** Performance versus bandwidth allocation (802.11)

**Impact of Object Density.** We varied the object density from 100 to 1000 objects. Given that the simulation area is 640,000 square meters, if 1000 objects participate in the simulation, then the average distance between two objects is about 25 meters, and if 100, then the average distance is 80 meters. This represents a mall or a convention hall (actually, the density in these cases is higher, but remember that not every mobile device is RBB-enabled).

Observe that in the 802.11 case the P2P throughput under RBB is about 40-50% of the ideal benchmark. In other words, installing high performance centralized databases and numerous hotspots (through which these databases can be accessed from everywhere) would double the performance. In the Bluetooth case many more access points would have to be installed (due to the smaller range), but the performance would triple.

The impact of object density on the RBB performance depends on the transmission range. When the transmission range is small as in the Bluetooth case ( $r=10$  meters, Figure 3), the matches throughput of RBB monotonically increases as the object density increases. When the transmission range is large as in 802.11 ( $r=50$  meters, Figure 2), the performance first increases, then it stabilizes. Intuitively, as the transmission range increases, two opposite effects are generated. On the one hand, each successful broadcast reaches more neighbors. On the other hand, there are more collisions and thus the amount of effective communication decreases. The figures show that when the transmission range is small, the positive effect is dominant; when the transmission range is big, the positive effect is dominant for lower densities, but the positive and negative effects cancel each other out for higher ones.

**Impact of Network Type.** The advantage of RBB over periodic flooding is more significant in Bluetooth (Figure 3) than in 802.11 (Figure 2). The reason is that

when the transmission range is small, the network is highly partitioned, and flooding from the producer reaches only a few moving objects. And repeated flooding does not appear to overcome this problem. In contrast, in RBB the moving objects cache the report for later broadcast, creating a replicated environment. Thus, flooding is more sensitive to disconnection.

**Impact of Bandwidth Allocation** (Figure 4). The reports throughput of RBB increases constantly when the bandwidth allocation for mobile P2P resource discovery increases from 0.01 to 0.9. The reason is that more reports can be included in a broadcast message. Further, observe that periodic flooding and PSTree also have a step-wise curve, however, they converge to a much lower value than RBB.

## 5. Relevant Work

In section 1 we compared RBB with the existing work on resource discovery and publish/subscribe in MANET's. In this section we compare RBB with other relevant works. Work has been done on data dissemination in mobile P2P networks [5, 3, 14]. These methods use the gossiping/epidemic communication paradigm to push data to mobile objects, and thus they do not rely on routing structures. However, there are significant differences between them and RBB. For example, in [5] the objective is to disseminate web pages in order to maximize cache hits. There are broadcasts; objects exchange pages (but not queries) when they encounter each other. There is no ranking of information or bandwidth considerations. In [3] the objective is to provide an alternative to MANET source-destination routing; the network-id's of the source and destination nodes are known. When the network is disconnected the destination is found by flooding. In contrast, the objective of the present paper is to deliver each report to all its subscribers. In [14] it is assumed that a message has a unique originator, and the issue is dissemination of this message. The question is to which neighbors to (point-to-point) transmit the message, whereas in our case the question is when and what to broadcast.

In data broadcasting (e.g. [11]), data is periodically pushed from a server to clients over a broadcast or multicast link. The major issue is how to schedule the broadcast content to minimize the access time and tuning time of the client community. For example, the data disks (see [11]) scheme broadcasts hot (popular) data items with higher frequency than cool (unpopular) data items. We use the same idea in RBB. However, since we have a mobile ad-hoc environment and there are no dedicated servers, we enable awareness of popularity by disseminating queries.

Our work is different from PeopleNet [13] in the following aspects. First, [13] does pair-wise exchange of the data rather than broadcast. Second, PeopleNet concerns the management of main memory, whereas RBB optimizes the bandwidth utilization.

## 6. Conclusion

In this paper we proposed the RBB algorithm for search and discover applications in MOPNET's. These applications include resource discovery, publish/subscribe, and matchmaking. The algorithm may be parameterized to use only a fraction of the available bandwidth.

RBB does not rely on any global routing structure. Every node independently makes a decision on when to broadcast and what resource reports to broadcast to its neighbors. This decision is made based on novel techniques for prioritizing and disseminating mobile information, as well as novel techniques for effective bandwidth utilization in mobile data dissemination. Queries and resource reports are mixed in a broadcast, so in some sense RBB can be viewed as a combination of the push and pull strategies.

We compared the RBB algorithm with PSTree, a previously published algorithm for ad hoc networks, and with periodic flooding. The comparison was in terms of the total number of reports matching the native query that are delivered to an average consumer. The results show that RBB outperforms the other two algorithms, particularly in Bluetooth networks (having a short transmission range), and when the nodes are sparsely distributed in space. The reason is that RBB copes better with a disconnected network. We also compared P2P and the ideal benchmark represented by a zero-delay centralized database approach. In the 802.11 case, the P2P throughput under RBB is about 40-50% of the ideal benchmark. In other words, installing numerous hotspots and high performance centralized databases would double the performance. In the Bluetooth case many more access points would have to be installed (due to the smaller range), but the performance would triple. As far as we know, this is the first general purpose quantitative performance of the two approaches.

## 7. References

- 1 S. M. Das, H. Pucha, and Y. C. Hu. Ekta: An efficient dht substrate for distributed applications in mobile ad hoc networks. WMCSA 2004, English Lake District, UK, 2004.
- 2 C. Frank and H. Karl. Consistency challenges of service discovery in mobile ad hoc networks. *7th ACM int'l symp. on modelling, analysis and simulation of wireless and mobile systems*, 2004.
- 3 H. Hayashi, T. Hara and S. Nishio. Cache invalidation for updated data in ad hoc net-works. In *Proc. Int'l Conf. on Cooperative Information Systems (CoopIS'03)*, 2003.
- 4 Y. Huang and H. Garcia-Molina. Publish/Subscribe Tree Construction in Wireless Ad-Hoc Networks. *MDM*, 2003.
- 5 M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *MobiHoc*, 2001.
- 6 O. Wolfson, B. Xu and H. Yin, Dissemination of Spatio-Temporal Information in Mobile Networks with Hotspots. *DBISP2P'04*, August 2004.
- 7 B. Xu, A. Ouksel, O. Wolfson, Opportunistic Resource Exchange in Inter-vehicle Ad Hoc Networks, in *MDM* 2004.
- 8 J. Choi, J. So, and Y. Ko. Numerical Analysis of IEEE 802.11 Broadcast Scheme in Multihop Wireless Ad Hoc Network, *The International Conference on Information Networking (ICOIN)*, 2005
- 9 Y. Huang, H. Garcia-Molina: Publish/Subscribe in a Mobile Environment. *Wireless Networks*, 10(6): 643-652 (2004).
- 10 L. Wu and P. K. Varshney, Performance analysis of CSMA and BTMA protocols in multihop networks(1). Single channel case, *Elsevier Information Sciences*, Vol. 120, 1999.
- 11 S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast Disks: Data Management for Asymmetric Communication Environments. *ACM SIGMOD* 1995.
- 12 R. Zhang and Y. C. Hu, HYPER: A Hybrid Approach to Efficient Content-based Publish/Subscribe. *Proc. IEEE ICDCS*, 2005.
- 13 M. Motani, V. Srinivasan and P. S. Nuggehalli, PeopleNet: Engineering A Wireless Virtual Social Network. *MobiCom'05*, Aug. 28-Sept. 2, 2005, Cologne, Germany.
- 14 S. Verma and W. Ooi, Controlling Gossip Protocol Infection Pattern Using Adaptive Fanout. *Proc. IEEE ICDCS*, 2005.
- 15 H. Takagi and L. Kleinrock. Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals. *IEEE Transactions on Communications*. Vol. COM-32, No. 3, March 1984.