Fast Bluetooth Service Discovery for Mobile Peer-to-Peer Applications

Sidath B. Handurukande Samrat Ganguly Sudeept Bhatnagar
NEC Laboratories America, Inc.
USA

1. INTRODUCTION

Bluetooth is a ubiquitous, low power wireless link which is becoming increasingly available in most mobiles devices. Consequently, Bluetooth has the potential to create neighborhood networks where mobile devices need to communicate with other devices in the neighborhood. Bluetooth based neighborhood network can engender and assist new mobile peer-to-peer (p2p) applications [5, 2, 3, 4]. Such mobile p2p services using Bluetooth bring certain unique characteristics that are not reflected in traditional Bluetooth services (e.g. printing service).

In contrast to the traditional Bluetooth applications, emerging mobile p2p applications have a different interaction scheme: these applications are based on symmetrical interactions for services. In other words, there is no distinction among devices such that one device is a service provider and another one is a service consumer. A given device can act as a service provider for one device while it is looking for other devices to consume services. In addition, user mobility and dynamically changing neighborhood connectivity imposes additional challenges in a deploying a mobile p2p applications. To handle such frequent changes, devices need to constantly search for other devices that offer specific service/s thereby making the service discovery a critical issue.

Service Discovery Problem. Though Bluetooth is a very appealing candidate technology for these p2p applications, there is a fundamental problem making the technology hard to use in the context of p2p mobile applications. The problem is that Bluetooth device and service discovery procedure is very expensive and time consuming in *symmetric* and dynamic settings. In these p2p applications each device needs to constantly search for other devices and services around them. When a given Bluetooth device is searching for other devices and services, it is not discoverable by other devices. Typically, a device discovery is done by hopping through the frequencies in every 1.28s [1]. Longer a device takes to discover a service in the neighborhood, longer it is unavailable for discovery by other devices. Clearly, if this discovery happens in a concurrent manner, the discovery time cannot be bounded in a worst case scenario. An alternative approach, to the concurrent search, is to wait for random period of time in the discoverable mode (so that other devices can discover) and then perform a discovery for devices and services at the end of this random period. However, as the number of devices increases (e.g., when there are 3 devices or more) the time taken to find the neighbors becomes long and intolerable. For example, in [5], authors show an example that takes around 10 minutes to search devices and services with 3 devices.

It is very clear in these p2p settings information exchange is difficult unless fast and power efficient discovery protocols are in place. We propose an algorithm to resolve the Bluetooth discovery problem so that it can be efficiently used in a dynamic setting as required by mobile p2p applications. Our solution is implemented at the application level and it complies with the Bluetooth specification. As a result, the application programmer can easily integrate with any Bluetooth stack implementation without having to change lower level protocol layers or device drivers: this greatly enhances the applicability of our solution in different computing environments.

2. SERVICE DISCOVERY MECHANISM

Our proposed service discovery mechanism is based on each device taking synchronized rounds: for this, the clock information can be obtained infrequently (e.g., from cellular network, Internet, GPS device) to calculate the offset between the global clock and the local clock. Once this offset is known the global time can be inferred at any given time. In each round, a device randomly decides a "role" to perform. The possible roles are to act as a discoverer (by performing a Bluetooth device & service discovery) or to act as a listener (by advertising Bluetooth service & waiting for incoming requests). This scenario is elaborated in Figure 1 where two devices in the discovery state while one is in the listening state.

Each round consists of a number of phases. The exact phase the device goes through in a given round depends on their role. Each device upon moving into a new synchronized round with probability p decides to perform device and service discovery using the Bluetooth functionalities (e.g., Device 1 in Fig. 1): if not the device move into a listener state (e.g., Listening Device in Fig. 1).

The length of this discovery phase is t_1 seconds (t_1 is 10.24 for Bluetooth protocol ver. 1.1). Upon finishing this phase the devices that performed a discovery move into a notification phase and inform its neighbors (that were discovered within the current round, if any) about the neighbors it has discovered (e.g., sending set N_1 by Device 1 in Fig. 1). Once this phase is finished the device then moves into a receiving phase and collects information from neighbors that acted as listeners within the current round (e.g., Device 1 receives N_3 in Fig. 1)

On the other hand, the listeners of a particular round (who

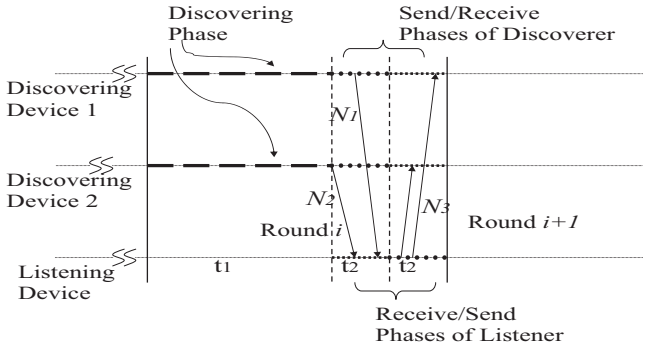

Figure 1: Synchronized Round-based Bluetooth Discovery

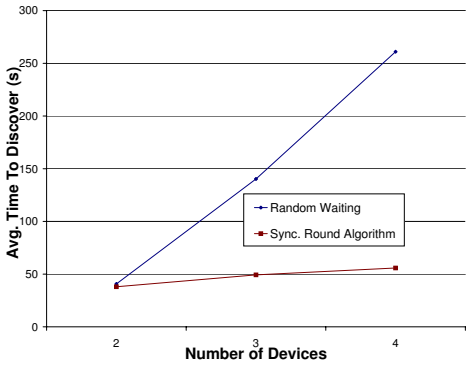

Figure 2: Discovery Time with Random Waiting and Synchronized Round Algorithm

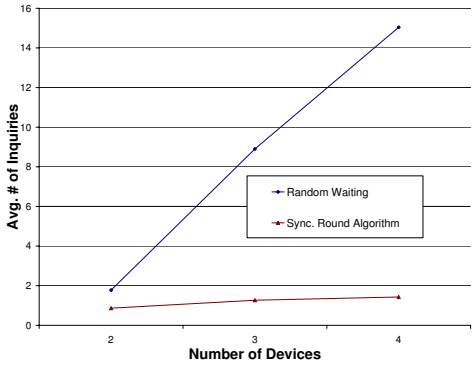

Figure 3: Average Number of Inquiries that is Necessary to Find Neighbors with the Services

decided not to perform discovery) wait to receive notifications from discoverers after t_1 seconds from the start of the round. Within this receiving phase the listeners collect all the received notifications and builds a set of "possible" neighbors N (e.g., Listening Device in Fig. 1 builds N_3): upon finishing this phase (of length t_2 s), the listeners then attempts to send the set N to its neighbors (e.g., Listening Device sends N_3 to Device 1 & 2). Note that all these attempts will not succeed due the hidden node problems. To filter out such hidden devices from the neighbor-set of a device, one can use a "ping" messages at the end of a round to evaluate whether a device is in fact a neighbor. This is not shown in Figure 1.

3. EXPERIMENTAL RESULTS

In the followings we show the results obtained using our real implementation. We used BlueCove an implementation of the JSR-82 Bluetooth API for Java running on top of Microsoft Bluetooth stack. We compared our new algorithm with the random waiting algorithm that we described in Section 1.

Discovery Time Reduction. In our experiments we observed the service discovery time is completed before 19s in general (it takes 10.24s to complete the device discovery). Based on these observations, we used a random value between 0 to 30 seconds as the random waiting time and t_1 and t_2 are selected as 19s and 4s respectively. Then for different number of devices we found the time taken to find all the neighbors (offering a specific service) using (1) random waiting algorithm and (2) our synchronized round based algorithm. Figure 2 shows the results. As can be seen clearly the our synchronized round based discovery algorithm perform significantly better than the random waiting time algorithm. In addition, as the number of devices increases, the performance of the random waiting algorithm decreases rapidly. But our algorithm keeps the discovery time to a manageable level as the number of devices increases.

Energy Consumption Reduction. Performing Bluetooth inquiry to find other devices is very energy intensive process in the Bluetooth communication. For example two Class 1 and 2 USB dongles we tested consume 77mA and 55mA during the device inquiry. On the other hand when waiting for incoming requests these devices consume 11mA and 14mA respectively.

As a result algorithms should be carefully designed to reduce the number of inquiries performed and to discover as many neighbors as possible using one device inquiry. In Figure 3 we show, on average how many inquiries are performed to find out other neighbors offering a specific service using (1) Random waiting algorithm (2) Synchronized round based algorithm. As seen, far too small number of inquires are necessary when using our synchronized round based algorithm: for example with 3 devices our algorithm needs only 1.26 inquiries on average to find other 3 devices with the specific service. On the other hand, the random waiting algorithm needs 8.9 inquiries. As a result of performing less number of device-inquires to find neighbors our algorithm consumes much less amount of energy.

These results show that our algorithm is significantly efficient in terms of time taken to discover neighbors and in terms of energy consumed for discoveries.

4. REFERENCES

[1] Bluetooth-SIG. Bluetooth specification. https://www.bluetooth.org/spec/.

[2] R. Chakravorty, S. Agarwal, S. Banerjee, and I. Pratt. Mob: A mobile bazaar for wide-area wireless services. In *Proc. of ACM MobiCom*, 2005.

[3] P. N. M. Motani, V. Srinivasan. Peoplenet: Engineering a wireless virtual social network. In *Proc. of ACM MobiCom*, 2005.

[4] M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *Proc. of ACM MobiHoc*, 2001.

[5] N. Ravi, P. Stern, N. Desai, and L. Iftode. Accessing ubiquitous services using smart phones. In *Proc. of IEEE PerCom*, 2005.