MaCC: Supporting Network Formation and Routing in Wireless Personal Area Networks

Makoto Takizawa  Hiroto Aida  Masato Saito
Graduate School of Media and Governance, Keio University
{makoto, haru, masato}@ht.sfc.keio.ac.jp

Yoshito Tobe
Department of Information Systems and Multimedia Design
Tokyo Denki University
yoshito@unl.im.dendai.ac.jp

Hideyuki Tokuda
Faculty of Environment Information, Graduate School of Media and Governance, Keio University
hxt@ht.sfc.keio.ac.jp

Abstract

This paper presents a novel effective scheme of configuring Wireless Personal Area Networks (WPANs), called Master-driven Connection Control (MaCC). WPANs are based on a new wireless technology, which enables portable and mobile computing devices, and consumer electronic appliances to communicate with each other. Assuming that WPANs operate in a master-slave style, we exploit the information about master-slave relationships for identifying a network topology. Specifically, MaCC provides the minimum-hop paths that cannot be obtained by flooding algorithms. MaCC has several prominent features: self-direction of every node, adaptive formation of networks, and minimization of hop counts for routing control. In this paper, we describe the details of MaCC and analyze its overhead under initialization, routing discovery, and reconfiguration. The results show control messages propagated in a MaCC network at route discovery are reduced to less than one half those in a network utilizing pure flooding.

1. Introduction

Wireless Personal Area Networks (WPANs) are new short-range wireless networks connecting portable and mobile devices, and consumer electronic appliances. These devices can be connected together by providing such a short-range wireless interface without any wired infrastructure. Typical WPANs operate with centralized control of multiple nodes. A master node, one of the nodes in a network has the right of controlling the network. In the network, nodes other than the master operate as slaves. These slaves behave cooperatively by being synchronized with the master. We call such a network a “master-driven” network.

In a master-driven network, the master performs polling; it asks each node in turn whether or not the node owns data to transmit. If many nodes exist in the network, the time interval between two polling messages for a node becomes large. Therefore, WPANs specifies the maximum number of nodes. When the number of nodes in an area exceeds this maximum number, multiple separate networks can be formed. In order to interconnect all the nodes, some nodes are required to be shared nodes, which belong to multiple different networks to bridge them. Respective networks can be inter-connected with each other via shared nodes.

In general, a routing protocol is required to accommodate communication between nodes in a large network. For wireless ad hoc networks, many on-demand routing protocols have been provided such as DSR [5], and AODV [8]. These protocols utilize flooding algorithms with different variations. However, flooding algorithms are not appropriate for WPANs. Because different networks are not synchronized, simultaneous transmissions from different networks may lead to collisions if radio-ranges of these networks are overlapped. Carrier sense channel access scheme such as CSMA can avoid simultaneous transmissions of multiple nodes even by flooding. However, master-driven channel access control scheme increases the possibility of simultaneous transmissions by flooding. Wireless collisions
degrade the quality of communications by invoking retransmissions. Furthermore, many messages are propagated unnecessarily. In addition to the problem of collisions, network topologies affect performance of routing in WPANs. For example, hop counts fluctuate dependently on the topology. Thus it is important to decide which nodes should be masters, slaves, or shared.

We present Master-driven Connection Control (MaCC), which performs WPAN configuration. MaCC provides route discovery function, offering the minimum hop-count routes. It also forms a topology corresponding to the discovered routes dynamically. We analyze its overhead under initialization, routing discovery, and reconfiguration, indicating that at route discovery phase, our scheme prevents simultaneous transmissions and control messages propagated in a MaCC network are reduced to less than one half those in a network utilizing pure flooding.

The rest of this paper is organized as follows. In Section 2, because we have formulated a Bluetooth ad hoc network [14] as WPAN and implemented the prototype of MaCC on the network in this research, we explain the Bluetooth link formation and Bluetooth networks, and discusses some related work on Bluetooth networks. Section 3 describes the design of MaCC, and in section 4 the implementation is mentioned. We show the overhead and discuss the analysis of the overhead in section 5. Section 6 concludes this paper and describes future work.

2. Preliminaries

In this section, we provide background information about Bluetooth technology and discuss the existin formation protocols and routing protocols for Bluetooth networks. Bluetooth emerges as low cost, low power and short-range radio technology based on frequency hopping spectrum spread. We explain two aspects of Bluetooth: Bluetooth link formation and Bluetooth networks.

2.1. Bluetooth link formation

The link formation process specified in Bluetooth baseband specification consists of two processes: inquiry and page. In the inquiry process, a master searches for active neighboring nodes to constructing a network with, to obtain low-level state information about those neighbors. The goal of the page process is to establish a bi-directional communication channel by utilizing the information gathered in during the inquiry process.

During the inquiry process, a node enters either the INQUIRY or the INQUIRY SCAN state. A master, a node in the INQUIRY state repeatedly alternates between transmitting ID packets containing an Inquiry Access Code (IAC) and listening for responses. A slave, a node in the INQUIRY SCAN state constantly listens for packets from nodes in INQUIRY state and responds by sending a Frequency Hopping Synchronization (FHS) packet including its Bluetooth address and other information that are utilized for synchronization. After completing the inquiry process, the master has obtained Bluetooth addresses of the discovered slaves and initiates page process by changing its state to PAGE state. The slave in the INQUIRY SCAN state periodically enters the PAGE SCAN state. Subsequently, the master exchanges unicast ID packets with each slave. When the slave responds back, both nodes proceed to exchange necessary information to establish the connection and eventually enter the CONNECTION state.

2.2. Bluetooth networks

All nodes in a Bluetooth network is organized into piconets, each of them composed of one master and up to seven active slaves which can communicate with the master only. A group of nodes in each piconet shares a common frequency hopping sequence by each piconet master. Within a piconet, a common channel is shared using a slotted time division duplex (TDD) protocol. Piconets can be interconnected via shared nodes to form a bigger ad hoc network called scatternet. Shared nodes alternately switch the hopping sequences of the sharing piconets. A shared node can be a master in one piconet and slave in others, or slave in all piconets. Figure 1 shows two examples of Bluetooth scatternets: a shared node, Z operates slaves in the two piconets, while the node operates a master in a piconet, a slave in the other.

2.3. Related Work

Bluetooth network formation protocols: Bluetooth Topology Construction Protocol (BTCP) [10, 11] creates a scatternet which consists of piconets of the minimum number to cover all nodes. BTCP has three phases: first, all nodes start alternating trying to discover their neighboring nodes, second, one coordinator which will determine a topology is elected, and third, the coordinator informs other nodes how a scatternet should be formed and they create connections following the information. Tree Scatternet Formation (TSF) [12] has constructed a scatternet as tree-structured. TSF does not require to exchange any messages with each node for scatternet formation. We utilize the algorithm of TSF to form a network as tree-structured. Other formation protocols [4, 6, 7] where masters are allowed to have as many slaves as possible are proposed. These protocols, how-
ever, do not address an aspect of routing; these topologies by the protocols may lead to the problem of traffic concentration on some nodes, a root node in a tree-structured network in TSF and master nodes in other protocols.

Routing protocols: Bluetooth scatternet routing protocols [1, 9] utilize pure flooding algorithm for static Bluetooth networks as some routing protocols such as DSR [5] and AODV [8] in wireless mobile ad-hoc networks. In these protocols, query packets for paths store a Bluetooth address or a piconet ID(global in [9], local in [1]) per hop. For Bluetooth networks classified in WPANs, pure flooding degrades the quality of communications and transmits many messages unnecessarily. In addition, these routing protocols assume that networks are already connected, and then they ignore the initializing overhead to constructing networks and to communicate in Bluetooth.

3. Design

In this section, we present the overview and detailed design of MaCC. We assume that WPAN is formed in home or office environments and is composed of some static and some mobile nodes: for example, consumer electronic appliances, and PCs, PDAs, and cellular phones. Some portion of WPAN topology is fixed, and some others vary. MaCC provides topology reconfiguration functions for such a WPAN, in which some static nodes form static topology and some nodes change their locations. Every node’s self-directed operations enable the reconfiguration and achieve robustness.

3.1. Overview

The design goal of MaCC is to discover the shortest path without flooding with less amount of traffic load than flooding, by utilizing a characteristic of master-driven operations in WPAN. MaCC has three phases: first, Parent Child Relationship-net (PCR-net) formation, second, route discovery, and third, connection establishment.

1. **PCR-net formation**: MaCC initially constructs WPAN as a tree-structured network, defined as PCR-net in this paper. A PCR-net does not provide any connections but only PCR, in which one and the other are parent and child, respectively. Figure 2 shows a conceptual image of PCR-net.

2. **Route discovery**: A node queries a route to communicate with another, sending a query to its parent in the PCR. The parent receiving the query sends it to its parent recursively, if the parent does not include the path information.

3. **Connection establishment**: A node sending an original query obtains the path information. For the first time in this phase, the node starts to establish the path which is the minimum hop-count from the source to the destination node.

3.2. **PCR-net Formation**

This section illustrates how PCR is established and a PCR-net is shaped in tree structure. To complete establishing PCR, every node alternates two processes in a random interval: searching and scanning. The searching process is
3.3. Route Discovery

In a PCR-net, a node queries a route when starting communications without broadcasting the query through the entire network. Every node keeps its adjacency list which lists the neighboring nodes found in searching process, and all nodes establish PCR, i.e., ROOT - ROOT, Non-ROOT - FREE, and FREE - FREE, to prevent loops in a PCR-net. Figure 3 shows the state transition of nodes in the PCR-net. In this figure, continuous lines represent transitions after establishing PCR, while dotted lines represent transition after disconnection. These rule and state transition that are equivalent of those of TSF allow every node to construct a tree-structured PCR-net. Figure 4 shows an example of PCR-net. Node A, nodes B, D, and H, and nodes C, E, F, G, and I correspond to ROOT, BRANCHs, and LEAVes, respectively. Nodes E and F, and nodes B, C, and D have node B, and node A as parent, respectively. A parent has multiple child nodes, while a child has only one parent.

MaCC allows every node to query a route to its parent and to reply to its child sending the query, recursively, and to execute Dijkstra’s algorithm [13] to acquire the shortest path from its adjacency list. In Figure 5, assuming E sends a query for a path from E to I. First, E tries to find a route from E’s adjacency list using Dijkstra’s algorithm. Since E’s list does not contain I, E sends the query to its parent, B. Similarly, B performs Dijkstra’s algorithm but fails this. Then, B sends the query to its parent, A. A succeeds in finding the route and then reply it to B. B receives the reply and sends it to E. Consequently, E can obtain the path information to I. In general, if the ROOT does not possess the path information, it returns an error message to inform that the route is not found.

3.4. Connection Establishment

A node sends a data packet including the path information which the node obtains. Relay nodes on the path conduct source routing; they look up the information and then forward the packet to the next node. Every node on the path establishes connection when sending the data, and disconnects it after the communication. MaCC establishes the con-
4. MaCC Implementation

This section mentions implementation of MaCC on Bluetooth technology. We have used RedHat Linux 7.3 with kernel 2.4.18 and BlueZ [15] as Bluetooth protocoll stack with version 2.3, and used BL-554 (Brain Boxes) as Bluetooth device which does not construct scatter-net, not capable of multi-hop communications. We have implemented a prototype of MaCC running as daemon process at application layer. Normally, we should implement MaCC as kernel module and will implement MaCC as kernel module in a future task. We have used 48 bit Bluetooth address based on IEEE802 as a node identifier. We have utilized UNIX domain socket as interface between MaCC and a communication application.

4.1. PCR-net Formation on Bluetooth

Bluetooth networks are master-driven and Bluetooth provides inquiry and page operations which allow a node to conduct searching and scanning, and to notify its own Bluetooth address to another. The prototype defines the completion of inquiry and page as the establishment of PCR. Bluetooth specifications, however, do not provide an absolute page. Create Connection command which establishes connection includes page operation. Therefore, the completion of establishing PCR is that of creating connection and child’s transmitting the adjacency list to its parent. The parent disconnects the connection after receiving the list.

The implementation verifies two rules in establishing PCR: permission of ROOT to establish PCR only with ROOT and prohibition of Non-ROOT from establishing it with ROOT and Non-ROOT. The permission is realized by the same way to the TSF algorithm: ROOT propagates and listens to 1D packets containing Dedicated Inquiry Access Codes (DIAC). Bluetooth specifications allow nodes in the INQUIRY SCAN state to listen to a particular list of IAC. ROOT listening to DIAC leads to grouping ROOTs. The prohibition is realized the way different from TSF algorithm: MaCC forbids all nodes from having multiple parents’ BD ADDRs. All the nodes which already keep one parent’s BD ADDR ignore incoming packets requesting their adjacency lists. In TSF, once connected, a node never performs scanning process, which leads to no pair of searching and scanning nodes in a network.

5. Overhead Analysis

In this section, we analyze the cost in three phases of MaCC: initialization, routing discovery, and reconfiguration cost. Initialization cost is to form PCR-net in tree structure from the initial condition in which all nodes are FREE. The route discovery phase is during the period from a node’s sending a query for a path through to the node’s receiving the path. After completing forming PCR-net, the nodes must reform new topology if a node or multiple nodes fails or moves, which accounts for reconfiguration cost.

5.1. Initialization Cost

In MaCC, the initializing cost is composed of the time to construct a PCR-net as tree-structured and the number of messages sent in constructing the PCR-net. Initial state FREE nodes start searching and scanning for active neighboring nodes, attempting to establish PCR with the nodes, and if the PCR is established, child transmits its adjacency list to its parent. The maximum delay of establishing PCR between two nodes is the following: \( I + S + C + S_{AL} \), called PCR delay, where \( I \) is the period for searching, \( S \) is the period for scanning, \( C \) is the connection delay, the average time between two nodes from the inquiry process to the connection complete, and \( S_{AL} \) is the time required to transmit the adjacency list. If a searching node has found \( n \) nodes \((n \geq 2)\), the PCR delay is \( I + S + nC + nS_{AL} \). We set \( I \) as 11 seconds, because Bluetooth specifications specify inquiry operation must last for 10.24 seconds unless the node determines to abort the inquiry earlier when it collects enough responses. While we intend to reduce routing discovery cost, initializing delay is large and some messages at initialization is required: at least \( 2(|V| - 1) \) messages are propagated.

5.2. Routine Discovery Cost

We analyze routing discovery cost, measuring two metrics: the run-time from a node’s transmission of a query to the node’s reception of the corresponding response packet, and the number of control messages by all the nodes in a PCR-net at the route discovery phase.

Run-time: The run-time at the route discovery phase consists of the time of message transmission, a shortest-path algorithms (Dijkstra’s Algorithm), and the connection establishment. We use one hop round trip time (\( rtt \)) as the time of message transmission. Dijkstra’s algorithm is denoted by \( D(|V|) \), where \( V \) and \( |V| \) represents a set of nodes and a size of the set, respectively. It consumes the delay \( C' \) from the PCR state to the CONNECTION state. The average run-time \( T_{\omega} \) is the following: \( T_{\omega} = \sum_{v \in \Theta(\omega)} \left( rtt + C' + \sum_{j \in \Theta(\omega)} D(\Theta(\omega - 1)(v)) \right) \), where \( \Theta(\omega)(v) \) represents the number of nodes in the tree rooted by a node at \( \omega \)-th rank from the node \( v \), and \( \omega \) denotes hop counts of the discovered route.
The number of control messages: In MaCC, the average number \( N_t \) of control messages transmitted in the entire network at the route discovery phase is \( N_t = 2 \sum_{i=0}^{\omega} i+1 \). Through route discovery by broadcasting request packets, flooding, \(|V| - 1(\geq \omega) + \omega \) messages are propagated, if request messages are transmitted without duplication and only one route is discovered. The relation between the number of control messages of MaCC and \( \omega \) is the following: \( \omega > 2 \frac{\sum_{i=0}^{\omega} i+1}{\omega+1} \). Therefore, MaCC transmits messages less than one half those in pure flooding at the route discovery phase. MaCC can decrease the possibility that collisions occur that are caused by many message transmissions for a given short time in wireless radio.

5.3. Reconfiguration Cost

Assuming one PCR-net is divided into \( n \) PCR-nets if some nodes fail or move, reconfiguring delay is \((n-1)(C + S_{\mu})\). Every node already has its own adjacent list and it attempts to establish PCR with a node included in the list. The number of the messages at reconfiguration is \(2(n - 1)\). Assuming a network topology changes at interval \( \alpha \) and one route discovery routine is performed for \( \alpha \). Then, in case \( n < |V|+1 \), for the interval \( \alpha \) the number of the messages at the route discovery phase and reconfiguration is less than that at pure flooding.

6. Conclusion and Future Work

In this paper, we have presented MaCC providing the functions of the route discovery and the adaptive network formation of tree structure in WPAN. In the short-range wireless network environment, propagating many messages unnecessarily affects the performance of other communications. We have presented the analysis of the overheads and showed MaCC reduces control messages at the route discover phase to less than fifty percent compared to pure flooding.

The implementation of MaCC, however, has the problem of connection delay due to the Bluetooth specifications. Bluetooth communications involve considerable delay to create connections. The average delay we measured is 1.4354 second. The implementation consume a large amount of time at the route discovery process. An one-hop query takes about 2 to 3 seconds. We plan to conduct simulations to illustrate the performance achievement except for connection delay. In addition, we will optimize MaCC to reduce overall delay in connection delay with Bluetooth.

In addition, In wireless ad hoc networks, the shortest-path is not always optimal [2, 3]. Therefore, we will introduce other metrics than the hop count to balance the traffic load and to improve system throughput.

References