



## Enhanced AODV routing protocol for Bluetooth scatternet

Omar Al-Jarrah\*, Omar Megdadi

Computer Engineering Department, Jordan University of Science and Technology, P.O. Box 3030, Irbid, Jordan

### ARTICLE INFO

#### Article history:

Received 4 September 2007

Received in revised form 14 October 2008

Accepted 15 October 2008

Available online 3 December 2008

### ABSTRACT

Bluetooth is one of the most widespread technologies for personal area networks that allow portable devices to form multi-hop Bluetooth ad hoc networks, so called scatternets. Routing is one of the challenges in scatternets because of its impact on the performance of the network. It should focus on reducing the power consumption in the network because most of the nodes are battery-operated portable devices. In this paper, we propose a routing protocol for Bluetooth scatternets that customizes the Ad hoc On-Demand Distance Vector (AODV) routing protocol by making it power-aware and suitable for scatternets. It enhances the AODV flooding mechanism by excluding all non-bridge slaves from taking apart in the AODV route discovery process. In addition, it improves the AODV route discovery phase by considering the hop count, the predicated node's power, and the average traffic intensity for each node as metrics for best route selection. By removing HELLO packets, our protocol reduces the control packets overhead and the power consumption in network devices. Simulation results show that the proposed protocol achieved considerable improvements over other enhanced AODV protocols by increasing the data delivery ratio by 10.78%, reducing the average end-to-end delay by 8.11%, and reducing the average energy consumption by 7.92%.

© 2008 Elsevier Ltd. All rights reserved.

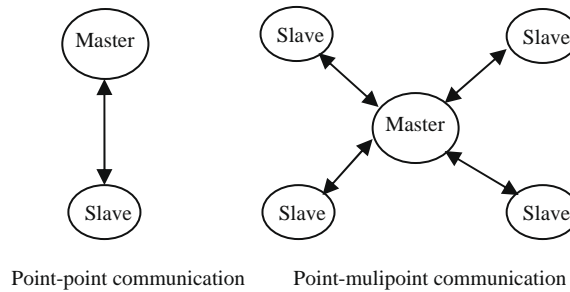
### 1. Introduction

In these days, many people carry handheld portable devices such as Personal Digital Assistants (PDAs), laptop computers, and mobile phones to keep in touch with friends or business partners and to share information using networks of these devices. Bluetooth is a promising wireless technology that allows portable devices to form a temporary short-range multi-hop wireless ad hoc networks or personal area networks (PAN) in order to communicate with each other without any central infrastructures [1]. Key features of the Bluetooth technology are robustness, low complexity, low power, and low cost. It can also be characterized by short-range wireless communication technology, frequency-hopping spread radio technology, time-division-duplex channel model, wireless connection between devices changes frequency at fixed intervals of time, and master-slave connection model where the master determines the timing of hopping and the sequence of used frequencies [1].

Bluetooth devices cannot communicate with each other unless a Bluetooth network is created. When two Bluetooth devices wish to connect to each other, one of them will be a master and the other one will become its slave. When such master and slave pair is established, a simple Bluetooth network, known as a piconet, is formed. A piconet is a collection of devices sharing a common channel. In any piconet, only one device acts as a master for synchronization purposes, while the remaining devices act as slaves for the duration of the network. A master node can have up to seven slaves and it uses a centralized polling scheme to allocate time slots for them. A slave can only transmit when its master has polled it in the

\* Corresponding author.

E-mail addresses: [aljarrah@just.edu.jo](mailto:aljarrah@just.edu.jo) (O. Al-Jarrah), [omar@huson.edu.jo](mailto:omar@huson.edu.jo) (O. Megdadi).



**Fig. 1.** Common topologies of Bluetooth piconet networks.

previous time slot. In the piconet, Bluetooth supports point-to-point and point-to-multipoint networks as illustrated in Fig. 1.

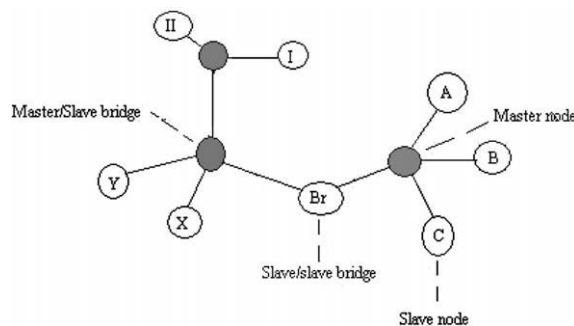
The piconets hop independently; each one has its own channel hopping sequence and phase that is determined by the respective master [2,3]. When several piconets need to communicate, they form a scatternet. If multiple piconets cover the same area, a Bluetooth unit can participate in two or more overlapping piconets at the same time using time multiplexing techniques. This unit can act as a slave in several piconets and a master in only one piconet since the frequency-hopping sequence is based on the address and the clock of its master. The Bluetooth device that participates in more than one piconet is called Bridge node. An example of a scatternet with several bridge nodes is shown in Fig. 2. The bridge node is used to take care of communication between different piconets. It can have a maximum of seven links, which means that it can link up to seven piconets. The choice of bridge devices is very important for routing in Bluetooth scatternet as the device with low power requirements will deliver higher performance/bandwidth to the connected piconets. Therefore, bridges constitute impediments on the bandwidth [4].

One major problem in Bluetooth networks is that slaves within a piconet do not have knowledge of other piconet members except the master. Another problem is that Bluetooth devices are small and resource constrained devices, which means that complicated processing and large routing tables are not suitable for those nodes. Scatternet is a dynamic network topology, in which Bluetooth devices may move from one piconet to another, such that every time a new device enters or leaves a piconet, the master should quickly inform the entire scatternet. In Bluetooth networks, direct slave-to-slave communication is not allowed and a master device must be involved in packet transfer.

In this paper, we address the problem of routing in Bluetooth scatternet network by focusing on reducing the power consumption and enhancing the traffic delivery ratio, while reducing the average end-to-end delay. We propose to modify and customize the AODV routing protocol to make it suitable for the Bluetooth scatternet network and to make it more power-aware. In our work, we introduce new metrics for best route selection between any two nodes in the scatternet.

We propose to improve the AODV route discovery phase and to reduce the required power consumption by considering the predicated node's power and the average traffic intensity for each node as metrics for a route selection between a data originator node and a destination node. Also, we propose to enhance the AODV RREQ packets flooding mechanism by excluding all non-bridge slaves from taking part in the AODV route discovery process. Moreover, we suggest avoiding the useless periodic broadcasting of HELLO packets that consume sizeable amount of energy and bandwidth without any need.

The rest of this paper is organized as follows. The next section presents a summary of the related work for routing in Bluetooth scatternets. The proposed enhanced power-aware AODV (P.A. AODV) routing protocol for Bluetooth scatternet network is presented in details in Section 3. Section 4 presents the simulation results. Finally, we conclude this paper in Section 5.



**Fig. 2.** The topology of a Bluetooth scatternet.

## 2. Related work

Scatternets require the use of a routing scheme to find paths in dynamic or static networks. Therefore, a routing protocol must be adopted for the Bluetooth scatternet network, which should take into consideration the formation and reconstruction of scatternets and minimize the negative impact of the overhead introduced due to inter-piconet scheduling.

At any time, a bridge can talk only to one piconet [5]. When it chooses to talk in another piconet, the frequency has to be adjusted to the new piconet, which wastes up to 3 time-slot intervals [6]. By switching between two piconets, a bridge node will establish a communication path between their masters. When a bridge node is switched between piconets for forwarding packets, there is a significant delay since the bridge node has to be adjusted to frequencies and time slots of both piconets. Even a nano-second variation in piconet timing will cost a time-slot wasting for synchronizing to the frequency of the other piconet [7]. This forwarding delay is a problem which has formed the bottleneck in scatternet communications.

Another problem in scatternet communications is the missed traffic. When a bridge node switches from piconet A to piconet B for transferring data, the unavailability of the bridge node causes a pause in transferring data and leads to the problem of missing traffic. Therefore, it is very important for routing in Bluetooth scatternets to choose the suitable scheduling that determines how and when to schedule the presence of a device in two piconets. The scheduling in each bridge device should ensure the fair treatments of passing by routes [8]. Most of the existing scheduling algorithms allocate time slots according to the loads in the bridge queues [9].

Several routing protocols have been proposed for mobile ad hoc networks [10,11], but they cannot be applied directly in Bluetooth networks due to Bluetooth's specific baseband and MAC-layer features. In the Route discovery process, the master needs to find out which of its slaves is a bridge node in order to avoid broadcasting a useless huge number of route request packets to find a route to destination node. Moreover, ad hoc routing protocols should be joined with the scatternet formation functions. Therefore, ad hoc routing protocols need to be adapted to, or interact very closely with, the underlying Bluetooth layers. When applied for scatternets, ad hoc routing algorithms would also have to consider establishment of new Bluetooth connections in order to find efficient paths through the scatternet. Consequently, forming and rearranging the scatternet should be a part of or closely integrated with the scatternet routing algorithm.

A number of protocols based on conventional ad hoc routing protocols have been suggested for Bluetooth scatternet routing [3,5]. The first effort for Bluetooth scatternet routing was introduced by Bhagwat and co-workers who proposed the routing vector method (RVM) based on the concept of source routing [12,13]. Route discovery and packet forwarding in RVM are similar to DSR procedures. However, unlike DSR, RVM does not require Bluetooth nodes to maintain cache information about routes. However, RVM can overload networks to find routes because of its flooding scheme.

In [14], Kargl et al. studied the possibilities of using Bluetooth for building ad hoc networks that are suitable for voice data transmission using synchronous SCO links. They proposed a new routing protocol called Bluetooth scatternet routing (BSR) that is similar to AODV or DSR. However, it keeps additional information about the state of Bluetooth links and fulfils the Bluetooth scatternet restrictions. This protocol is proposed as a cross layer optimization, between link and network layer, to shorten the connection set-up delay.

Huang et al. proposed an optimized AODV routing algorithm for Bluetooth [15]. In this protocol, HELLO packet is only sent after link establishment to exchange node's initial status. In addition, master does not forward routing information to its slaves, but behaves as a cluster-head for its slaves regarding routing information. Cross-layer Optimized Routing protocol for Bluetooth (CORB) is a QoS-extended AODV routing protocol optimized for Bluetooth MAC [16]. It defines a new Load Metric (LM) to reflect the nodes' link bandwidth with respect to Bluetooth nodes' role in the scatternet. This LM metric and some MAC-layer parameters are automatically adjusted according to the changes in radio conditions of scatternet. The main feature of this QoS AODV routing protocol is using LM instead of number of hops as a metric for best route judgment. The propagation of CORB RREQ packet over the scatternet is similar to that one in conventional AODV. The destination node, upon receiving the propagated RREQ, answers by unicasting a RREP packet that contains the LM field. Each relay node updates the LM in the RREP message with the maximum value of LM in the RREP and its link LM. When the originator node receives several RREPs, it selects a route with smaller LM. The path with smallest LM is the path with larger bandwidth.

In most cases, Bluetooth nodes are battery-driven devices, which will make the power invaluable for all the power-consuming components in the device. Limited services and applications can be supported by such devices because batteries carried by mobile nodes in the Bluetooth network have limited power and their processing capabilities are limited. This becomes a bigger issue in Bluetooth scatternet ad hoc networks because they are comprised of systems of autonomous mobile nodes that cooperatively rout packets for each other.

In a Bluetooth ad hoc network, nodes would need to periodically enter the inquiry state, which leads to more energy consumption. Furthermore, masters and bridges, other than the packet source and destination, participate in the packets delivery process, which results in extensive energy consumption in these nodes. Shek and Kwok [5] proposed an enhanced AODV (EAODV) algorithm for ad hoc routing in Bluetooth scatternet network. In this proposed algorithm, route control packets have a smaller size than the original AODV versions with 17% reduction. This saving is very important in this type of networks because it can save the power consumptions of intermediate nodes as some nodes may be just a small headset or remote controller that have limited power. In addition, it can allow the control packets to be fitted into smaller packet size in a narrow bandwidth Bluetooth network. Moreover, this packet reduction shrinks the interference probability and allows faster transmission as packets require less time slots. On the other hand, the AODV flooding mechanism in the proposed algorithm

has also been modified by keeping away from involving all non-bridge slaves in the AODV route discovery process. A master node will forward all route packets to its bridge nodes, while a bridge slave will forward the packets to all of its masters that are not the precursor node of the received packet. However, this enhanced AODV, like conventional one, uses only the number of hops as a metric for best path selection. When a node receives duplicate RREQ, it will drop the packet without forwarding. The route reply and HELLO packet broadcasting processes in this enhanced AODV is similar to that in conventional protocols.

Prabhu and Chockalingam proposed a routing protocol, similar to DSR, which uses the available battery power in Bluetooth device as a cost metric to select the best route [17]. In this protocol, the RREQ packet contains a cost field used for path selection. When an intermediate master receives the request packet, it appends its available battery power level to the cost field and then forwards the packet to all bridge nodes of its piconet. Each bridge node receives the packet adds its available battery power level to the cost field and forwards the packet to all piconets that it belongs to. At the destination side, the destination will wait for an amount of time until receiving multiple copies of the same packet from different paths with different cost field. It then selects the path with maximum cost field which represents maximum cumulative battery power in the path. The destination then sends its route reply packet through the selected path, which contains the selected route. After receiving the reply packet, the originator node can send all buffered packets destined for the destination through this selected path.

### 3. Enhanced AODV routing protocol for Bluetooth scatternet

As mentioned previously, the existing ad hoc routing protocols cannot be applied to Bluetooth directly. AODV is used as our core protocol because it is suitable for Bluetooth's narrow bandwidth and small packet size features. However, AODV should be enhanced to improve its performance. Our protocol is based on enhancing the route discovery process of AODV by taking into account the predicated battery power that each node will have in a period of time and the traffic intensity at each node. Two algorithms are used to obtain these two metrics at any time during AODV route discovery phase: traffic intensity calculation algorithm and power predication algorithm.

#### 3.1. Traffic intensity calculation algorithm

In this algorithm, every  $T$  seconds each node counts the number of bits it has received. Therefore, every time the node receives a packet, the number of received bits is incremented by the size of that packet [18]. The next step is to use the smoothed exponential average for traffic intensity calculation. The formula used to calculate the traffic intensity is given by

$$TI = \alpha \times TI + (1 - \alpha) \times R(T), \quad (1)$$

where  $TI$  is the traffic intensity,  $0 \leq \alpha \leq 1$  is a smoothing parameter,  $R(T)$  is the input rate in the last  $T$  seconds, and  $T$  is the sampling time.

We have used this smoothed exponential average because we need to find the average bit rate at any node between the start time and the current moment. The parameter  $\alpha$  is used to control the effect of recent and past situations.

#### 3.2. Power predication algorithm

We have used a power predication algorithm similar to that in [18]. Every node can be in any one of the following three operation modes: Idle, Send, and Receive. When a node sends or receives a control or data packet, some amount of power should be consumed from the node battery power level. In this case, the size of packet determines the power consumption degree. There is also an amount of power consumed when the node is in the idle state. For this purpose, we use the baseband functions that discharge the node battery power depending on the current state of the node and the size of transmitted/received packet. In addition, we have used the smoothed average function to calculate the average amount of consumed power for the sending and receiving processes as follows:

$$AP_s = (1 - G) \times AP_s + G \times Ps, \quad (2)$$

where  $AP_s$  is the average amount of power spent on sending or receiving a packet,  $Ps$  is the amount of spent power, and  $0 \leq G \leq 1$  is a smoothing parameter.

We have used the Markov chain to predict the power [18], where each mode of operation (Idle, Send and Receive) represents a state in the chain. For our prediction, we need to define the following probability matrix:

$$P = \begin{bmatrix} P_{ii} & P_{is} & P_{ir} \\ P_{si} & P_{ss} & P_{sr} \\ P_{ri} & P_{rs} & P_{rr} \end{bmatrix}, \quad (3)$$

where  $P_{kj}$  is the probability that the next state is  $j$  given that the current state is  $k$ . To find  $P_{kj}$ , we divide the number of times in which node was in  $k$  state and went to the  $j$ th state by the number of times node was in  $k$  state. The  $n$ -step transition probability  $P_{ij}(n)$  is the  $(i,j)$  element in the  $P^n$  [18].

The expected amount of power that will be spent in the next  $T$  time steps is given by

$$EP = \sum_{s=1}^3 \left( \sum_{t=1}^T P_{is}(t) \right) \times AP_s, \quad (4)$$

where  $AP_s$  is the average power spent in sending or receiving.

The expected power of any node after  $T$  time steps is found by subtracting  $EP$  from current node battery power. When a node sends or receives a packet, it should do the following:

1. Record the last send or receive time.
2. Update the battery power level of the node.
3. Increment the sent count or receive count.
4. Update the probability matrix.
5. Record the current state as the last state of node.

### 3.3. Route discovery packets

Our RREQ packet is a modified version of the AODV RREQ packet. The reserved bits are not used in our RREQ packet. However, the following fields are added:

- Average traffic intensity as described in Eq. (1).
- Predicted power which is obtained by subtracting the expected power in Eq. (5) from the node battery power.
- Expected reply time: The originator node calculates the value of expected reply time in the RREQ using following formula:

$$RT = 2 \times TT7 \times TTL, \quad (5)$$

where  $RT$  is the expected reply time,  $TT$  is the node traversal time, which is a fixed value that is set by AODV to 40 ms, and  $TTL$  is the value of  $TTL$  field in the RREQ packet:

- *ID-List*: a Boolean array that is used to prevent any possible routing loop that may occur because our protocol does not discard the duplicated RREQ packets. The size of this list should be selected carefully so that the size of RREQ packet stays suitable for Bluetooth networks. Every node that receives a RREQ packet will first check its entry in the *ID-List*, if it finds a *TRUE* value; it will discard the received RREQ packet. Otherwise, it sets its corresponding entry (related to node ID) in this list to *TRUE* before rebroadcasting the received packet.

Our RREP packet is a modified version of the AODV RREP packet, where we eliminate the reserved bits. Since Bluetooth nodes can rapidly detect any piconet link loss, it is not necessary to send HELLO packets for link loss discovery. Therefore, in our AODV protocol we remove HELLO packets, which reduce the overhead of the routing algorithm.

### 3.4. Check connections algorithm

We have devised an algorithm that used by every master and bridge node to discover the current number of available Bluetooth links connected to the node. Every node can retrieve all information about its current connections using the LMP layer functions as specified in the Bluetooth stack. For a master node, this algorithm returns the number of current non-bridge slaves and number of bridge nodes in its piconet. On the other hand, this algorithm helps every bridge node to get the number of piconets that it belongs to.

Every master node has to check whether the destination of RREQ is a slave member of its piconet. It uses the LMP functions to discover the existence of the destination in its piconet. If the master node finds out that the targeted destination is one of its current non-bridge slaves, it can send a RREP to the source node without forwarding the RREQ to the destination.

### 3.5. Route tables

The route tables are similar to those in the AODV, but with two additional fields: the next hop predicted power and the next hop average traffic intensity. Upon receiving RREQ packet, every node creates a reverse route to the originator and copies the values of the two fields from the RREQ packet into the corresponding fields of the created reverse route entry. The two fields are used by every node during the best route selection process when receiving duplicated RREQ packet from the same originator as it will be discussed later.

### 3.6. Reply cache

Every destination node should wait for a period of time before sending RREP packet to choose the best route to the originator node. Therefore, a reply cache is defined in each destination node to make a list to every originator node. Each reply cache entry contains the following six major fields:

- **Originator address:** the address of the RREQ originator node.
- **Destination sequence number:** the current sequence number of the destination.
- **Timestamp:** the value of first RREQ Timestamp field.
- **Expiry time:** it is used to indicate the validity of the entry. This expiry time is set to the value of the period of waiting time before destination can reply.
- **Hop count:** it can be either 1 or 2.
- **Destination Address:** it depends on the location of the destination because the proposed protocol allows master nodes to send RREP packets when the destination is a non-bridge slave in their piconets. If the destination is the node itself, the Destination Address field is set to the node's address with the hop count set to 1. However, if the node is a master node and the destination is a non-bridge slave member of its piconet, the Destination Address field is set to the slave address with the hop count set to 2.

When a reply cache entry is expired, the RREP generator function is called to create a RREP packet using the information of this expired entry.

### 3.7. Route discovery phase

When an intermediate master node receives a RREQ packet, it will check if the destination device specified in the RREQ is a non-bridge slave within its piconet. If the destination node is one of its non-bridge slaves, the master will create a reply cache entry for the received RREQ and stop forwarding the RREQ packet since any non-bridge slave has only one available link and consequently, there is no possibility to receive a RREQ packet from another node except its master. For this situation, the master plays the role of destination on the behalf of its slave and uses the best route selection algorithm as a real destination to choose the best route to its slave.

If the destination is not a simple slave within its piconet, the master forwards the RREQ packet. Before rebroadcasting a received RREQ packet, each master finds out which of its slaves is a bridge node. Then, the master sends the RREQ packet only to the bridge nodes that are not the same as the source node of the received RREQ. If no slave is found to be bridge node, the master will discard the RREQ. When an intermediate bridge node receives a RREQ packet, it will find out its masters and sends the RREQ packet to all of its masters that are not the same as the source node of the received RREQ packet.

Upon receiving the RREQ packet, each intermediate node caches the RREQ ID and originator address and stores a reverse route toward the originator node with predicted power and average traffic intensity copied from the RREQ packet. Each node calculates its predicted battery power using predicted power algorithm and its average traffic intensity using average traffic intensity algorithm. Before rebroadcasting the RREQ packet, each intermediate node sets the Predicted Power field in the RREQ packet to the minimum of its own predicted power and Predicted Power field in the received RREQ packet. Also, it sets the average traffic intensity field in RREQ packet to the maximum of its own average traffic intensity and average traffic intensity field in the received RREQ packet.

Unlike the conventional AODV, our protocol will not discard any duplicated RREQ that arrives from the same originator through different route. However, initially the intermediate node checks if the duplicated RREQ has been already seen using the ID-List field of the packet. For routing loop avoidance, if the checked RREQ packet previously arrived to the node, it will be discarded. Otherwise, if the duplicated RREQ has arrived through a better route, the intermediate node will update its reverse route to the originator and forward the RREQ packet after updating the average traffic intensity field and Predicted Power field in the RREQ.

Upon receiving RREQ packet, the destination node will create a reply cache entry for the received RREQ packet and wait for a period of time to receive more RREQ of the same originator through different routes. This destination waiting time is calculated as the difference between the expected reply time and the travel time of the RREQ packet.

### 3.8. Best route selection algorithm

When the node receives duplicated RREQ packet from the same originator, it will use the best route selection algorithm to choose the best route to the originator node as shown in Fig. 3. This algorithm works as a mechanism for power saving and timely delivery of data packets into the route discovery phase. To select the path with lower cost to the originator node, the algorithm utilizes two metrics: predicted residual power and average traffic intensity at each node. The cost for a route is given by

$$\text{Cost} = \alpha \times P/\text{MAX\_POWER} + (1 - \alpha) \times \text{TI}/W, \quad (6)$$

where  $P$  is the next hop predicted power,  $\text{TI}$  is the average traffic intensity of the next hop,  $W$  is the bandwidth, and  $0 \leq \alpha \leq 1$  is a smoothing parameter. This algorithm selects a reliable path between the originator and the destination node by using nodes with an appropriate battery power that are identified by the threshold given by

$$\text{THRESHOLD} = \text{LEMDA} \times \text{MAX\_POWER}, \quad (7)$$

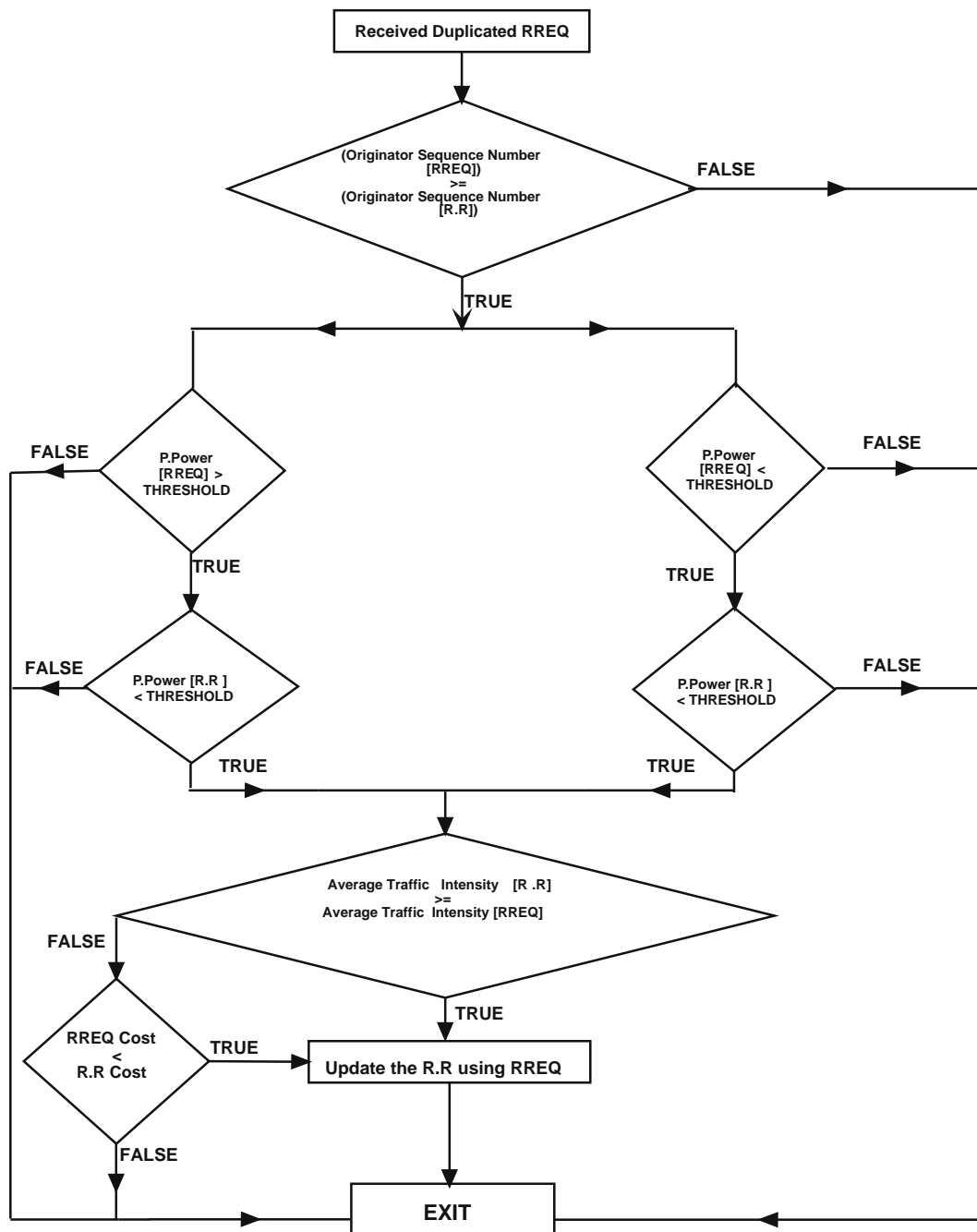


Fig. 3. Enhanced AODV route selection algorithm.

where the parameter LEMDA is used for choosing a percentage of the maximum battery power that will keep the nodes alive during the communication.

If a RREQ arrives through a route with smaller average traffic intensity than the existing reverse route, the algorithm will update the existing reverse route using the information that is carried by the received RREQ packet. This reverse route update helps in avoiding routes that pass through congested nodes, which reduce the data delivery ratio, increase the end-to-end delay of the data traffic, and have a quick dissipated battery power. Otherwise, the algorithm will define a cost function consisting of the predicted power and average traffic intensity parameters, where it will select the optimal path from a power and traffic load characteristics point of view. In order to achieve a balance between the two components of the cost function, the smoothing parameter  $\alpha$  is introduced with value between 0 and 1.

#### 4. Simulation and results

To evaluate the performance of our routing scheme, we have used the University of Cincinnati Bluetooth (UCBT) [21] ns-2 based Bluetooth simulator. Our performance evaluation is examined by comparing the data delivery ratio, average end-to-end delay, and power consumption of our protocol and both the AODV and enhanced AODV in [5].

The simulation environment is made of 50 Bluetooth nodes scattered on an area of  $70 \times 70 \text{ m}^2$ . We have used the algorithm in [19] as our scatternet formation algorithm. We have used the random way point mobility model. In this model, each node is moving to randomly chosen direction and speed. The destination position and speed values are generated in a random fashion. The speed values were chosen randomly from 0 to 1.2 m/s. The position of a node was updated every half slot (the smallest unit of a Bluetooth clock) that is equal to  $312.5 \mu\text{s}$ . To reduce the control overhead of scatternet reconstruction and maintenance, we assume that each piconet master is moving according to the random way point model and slaves are staying in the short range within less than 3 m of their master.

The radio range was assumed to be 11.2 m. Each link has a queue which can buffer 50 L2CAP packets. In our simulation, we have used the DRP algorithm in [20] as our RP bridge scheduling algorithm. The DRP algorithm is implemented using the Sniff mode. During the simulation, Sniff Cycle was 256 slots or 0.16 s. We have used a Priority based Round Robin (PRR) algorithm as a polling algorithm in the piconets to give a bridge higher priority to access the channel and favour inter-piconet traffic. Bluetooth nodes are assumed to have 1 Mbps bandwidth.

We have used Constant Bit Rate (CBR) traffic to represent applications that send small sized packets regularly. These applications vary from simple Bluetooth enabled mice control to voice transmissions. Each CBR traffic connection was generated with 0.15 s interval and 512 bytes packet size. The maximum number of packets that can be transmitted during each CBR traffic connection was 30,000 packets. During the simulation, source, destination, and the starting time of each CBR traffic connection were selected randomly, where every node in the scatternet participated in one or more traffic connections. The total simulation time was 1000 s. During the first 90 s, the scatternet was constructed using the selected scatternet formation algorithm. The starting time of the first CBR traffic connection was in the 93 s.

The power consumption model implemented by the UCBT simulator was used to calculate the energy consumption in each Bluetooth node. In the baseband layer, each transmitted/received packet will be queued in the baseband TXBuffer. At the baseband TXBuffer, when the node sends or receives a packet, the node residual energy is updated by the following model:

$$RE = CE - (TOT + WT) \times AECR, \quad (8)$$

where RE is the residual energy, CE is the current energy, TOT is the total operation time, WT is the warm up time, and AECR is the active energy consumption rate. The total operation time depends on the length of the packet received/sent. The warm up time is defined as a time from OFF to ON of the Baseband TXBuffer, and it is set to 0.0002 s in our simulation. The value of active energy consumption rate is  $1 \times 10^{-4} \text{ J/s}$ . Each node will turn off when its residual energy reaches its minimum level of 0.1 J. The total active time of each node is represented as a cumulative sum of each receiving/sending operation time.

In order to evaluate the performance of our proposed protocol, we have implemented and compared three protocols, namely, the original AODV, the enhanced AODV (EAODV) proposed in [5], and our enhanced power-aware AODV (P.A.AODV) under different traffic load with different number of CBR connections. Data delivery ratio, average end-to-end delay, average residual energy, and average energy consumption metrics are used to evaluate the performance of above three protocols. Each experiment was repeated five times and the results are obtained by averaging the results from the five experiments.

We have used the data delivery ratio as the first performance comparison metric. Fig. 4 shows the data delivery ratio of the three protocols at different number of generated CBR traffic connections during simulation time. It is expected that the

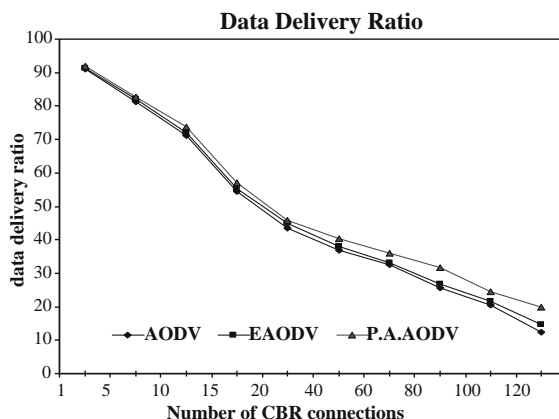


Fig. 4. Data delivery ratio with different CBR connections.

data delivery ratio decreases with the increase of the traffic load in the network because of the uncontrolled congestion, queue overflow at intermediate nodes, interference, and collision. Moreover, as the traffic load increases, more cross traffic passes through all bridge nodes in the network, which results in more scheduling processes at these bridge nodes. When a bridge node is switched to establish a communication path between piconets in the network, the unavailability of the bridge node will cause a pause in transferring data and leads to more packet loss. This figure shows that on the average our protocol outperforms the AODV by 12.85% and the EAODV by 10.78%. In addition, it shows that at light traffic (10 connections) the data delivery ratio of AODV, EAODV and our AODV are almost the same. As the traffic increases (60, 80, 100, 120 connections), it gives higher delivery ratio when comparing with the other two protocols. For example, at 120 CBR connections, our protocol outperforms the AODV by 38.24% and the EAODV by 27.08%.

In the second set of experiments, we have compared the three protocols in term of the average end-to-end delay at different number of generated CBR traffic connections. The average end-to-end delay is the ratio between the total end-to-end delay for all received packets and the total number of received packets. Fig. 5 shows that the average end-to-end delay increases with the increase of the traffic load in the scatternet network. At high traffic load, more nodes will be heavily involved in the network and as a result, their queues will start growing, which will increase the average end-to-end delay of the data packets. When a bridge node is switched between piconets for data packets forwarding, there is a significant delay and waste of time slot since the bridge node has to be adjusted to frequencies and time slots of both piconets. As a result of the increase in the amount of scheduling in the case of heavy traffic, the waiting time of data packets will increase at each intermediate bridge node and consequently, the average end-to-end delay of these data packets will increase. This forwarding delay is a problem that will be repeated at each bridge along the route between the source and the destination of the data traffic session.

Fig. 5 shows that our protocol reduces the average end-to-end delay by 15.52% when compared with AODV and by 8.11% when compared with the EAODV. Again, when the traffic load increases, the average end-to-end delay of our protocol becomes smaller than that of the other two protocols. At 120 CBR connections, our protocol reduces the average end-to-end delay by 16.77% when compared with AODV and by 12.40% when compared with EAODV.

We have also compared the performance of the three protocols in term of power consumption. Figs. 6 and 7 highlight the average residual energy and power consumption of the three protocols at the end of simulation time, respectively. It is not surprising that as the number of CBR traffic connections increased, nodes consume more energy. From Fig. 7, our protocol

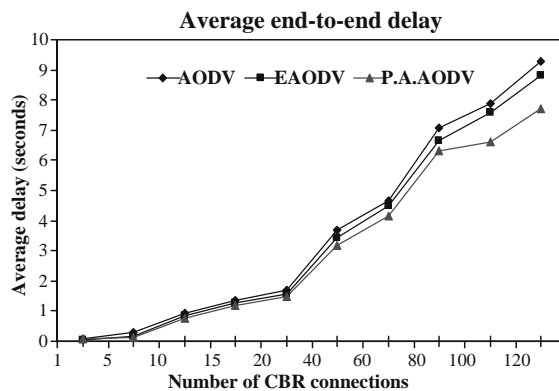


Fig. 5. Average end-to-end delay with different CBR connections.

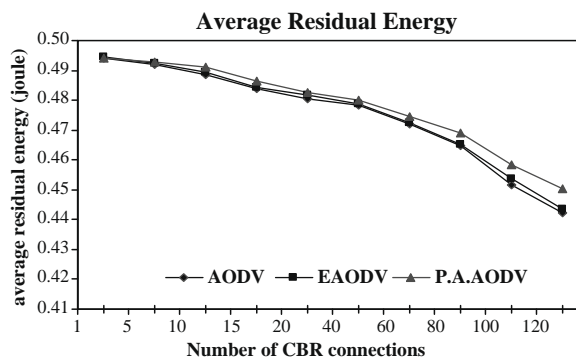


Fig. 6. Average residual energy with different CBR connections.

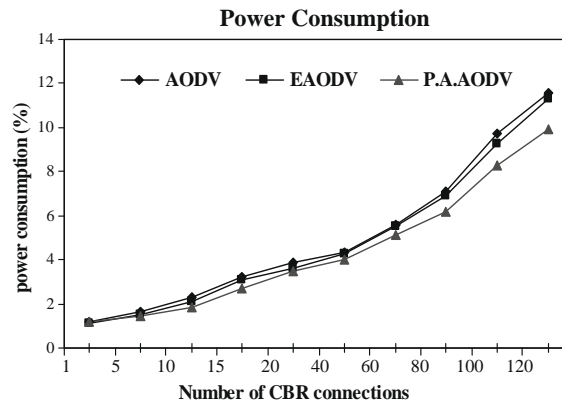


Fig. 7. Power consumption with different CBR connections.

reduces the average power consumption by 11.78% when compared with the AODV and by 7.92% when compared with EAODV. At 120 CBR connections, our protocol reduces the power consumption by 14.60% when compared with AODV and by 12.49% when compared with EAODV.

Referring to simulation results in Figs. 4–7, our protocol works better than both AODV and EAODV in the case of heavy scatternet traffic load. The AODV and EAODV consider the hop count as the metric for best route selection, while our protocol considers the predicated power and average traffic intensity in the best route selection. The shortest route that results from considering only the hop count is not necessary the most reliable path. When we only consider the hop count in route selection, a lot of CBR connections will share the same path simultaneously, which means that the queue at each node across the shared paths will develop uncontrolled congestions that cause many dropping of the data packets and increase the packets buffering time. Furthermore, AODV and EAODV will result in a quick depletion of the battery power of the nodes along the most heavily used routes in the network. The more packets in the heavily used nodes queues will increase the average end-to-end delay and reduce the data delivery ratio.

In Bluetooth networks, there is no need to broadcast HELLO packets are to detect link loss. By removing these HELLO packets, our protocol outperforms the AODV and EAODV protocols. In this way, our protocol can greatly reduce the control packet overhead of routing algorithm and save the energy consumption associated with broadcasting HELLO packets.

On the other hand, our protocol reduces the situation where there is a heavy traffic at some parts of scatternet while low traffic exists at other parts. This balances the number of bridge scheduling through most of network bridge nodes. When a bridge node is switched between piconets for forwarding packets, there is a considerable delay associated with the switching as the bridge node has to adjust to frequencies and time slots of both piconets. A lot of schedules at the same bridge nodes will result in unfair treatments of all routes passing through them and it will cause more packet losses and lower data delivery ratio because traffics at a heavy bridge node need to share the piconet bandwidth and partition its capacity. As a result, scheduling will not be optimal at heavy bridges. Increasing the amount of scheduling at a bridge node will increase the timing and frequency adjustment processes and leads to more energy consumption in the node while it is switching back and forth between piconets. Moreover, the waiting time of packets at each bridge node will be increased and so the average end-to-end delay. Our protocol outperforms the AODV and EAODV because it reduces the number of cross traffics at each bridge node, which leads to reducing the average end-to-end delay and increasing the data delivery ratio.

The proposed approach achieves a good performance under heavy traffic load because it predicts the battery power of each node based on power consumption history and expects how much traffic load is going to pass through each node based on previous traffic loads. Hence based on the predicted information, it protocol can select the most reliable intermediate masters and bridges. This approach improves the bridge scheduling inside the network, reduces the average energy consumption in the scatternet, improves the data delivery ratio, and reduces the average end-to-end delay.

In the next set of experiments, we study the effect of node mobility on the data delivery ratio. We ran our experiments with 10 CBR traffic connections and 0–2.0 m/s nodes speeds. The impact of mobility can be seen in Fig. 8, in which the data delivery ratio decreases when the mobility increases. This is mainly due to the fact that links are more stable at low speeds. At high speeds, frequent link breakage and route recovery due to the node mobility will limit the scalability of networks.

More link breakage will results in more route discovery initializations more control overhead propagation, and more RREQ failure. This repeated route discovery phase will consume a lot of available bandwidth and result in serious collisions. Also, when the route discovery and maintenance increase, more queues at intermediate nodes will develop uncontrolled congestions that cause many dropping of the data packets. Furthermore, reconstruction and maintenance of broken scatternets will lead to more repair control packet propagation that consumes more bandwidth and results in extra overhead with more collision and contention possibilities. Likewise, more data packets will be dropped because of larger probability of buffer overflow in the scatternet with high mobility of nodes. Low available bandwidth for data packet will lead to more packet losses.

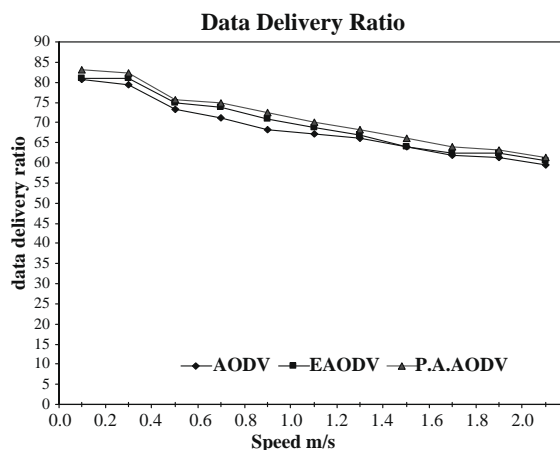


Fig. 8. Effect of node mobility on data delivery ratio with 10 CBR connections.

It is evident from Fig. 8 that our protocol achieves more data delivery ratio when compared with both AODV and enhanced AODV in [18] because of its enhanced route discovery process. Our protocol reduces the risk of queues overflow at each node in the network by reducing the number of transmitted route control packets and considering the number of waiting packets in the queue at each node as a metric for best route selection.

## 5. Conclusions

In this paper, we have developed an enhanced AODV routing protocol for Bluetooth scatternet network. The work was accomplished by introducing a new AODV RREQ flooding mechanism and eliminating the HELLO packets broadcasting. Therefore, we can avoid the flooding of useless huge route request packets that consume considerable amount of energy and bandwidth without any utility in the routing discovery process. Moreover, we have used new metrics for selecting the best route from a set of available routes between any inter-piconet traffic sessions in the scatternet. In addition to hop counts, the metrics are based on node's predicted battery power and the average traffic intensity passing through each node. This will help the AODV in monitoring the remaining energy and how much the traffic load is going to pass through each node in the near future and using the predicted situation in its route decision. Our protocol balances the traffic load inside the whole scatternet network and avoids uncontrolled congestion at every Bluetooth node.

Simulation results show that the proposed protocol improves the performance in the Bluetooth scatternet network by reducing the required energy consumption, increasing the data delivery ratio, and reducing the average end-to-end delay. Our protocol outperforms the original AODV by increasing the data delivery ratio by 12.85%, reducing the average end-to-end delay by 15.52%, and also reducing the average energy consumption by 11.78%. When comparing with the enhanced AODV in [5], our new protocol increases the data delivery ratio by 10.78%, reduces the average end-to-end delay by 8.11%, and also reduces the average energy consumption by 7.92%.

## References

- [1] <<http://www.bluetooth.com>>.
- [2] Bluetooth SIG, Specification of the bluetooth system, Version 1.1; February 2001.
- [3] Choi C, Choi HW. DSR based bluetooth scatternet. Thailand: ITC-CSCC 2002; July 2002.
- [4] Yugandhar D. Dynamically reconfigurable adaptive power-aware clustering and routing for bluetooth network. Master thesis, Arizona State University; December 2002.
- [5] Shek Liza Lai-Yee, Kwok Yu-Kwong. Integrated ad hoc routing and time-slot scheduling in bluetooth networks. In: Proceeding of the ninth ACM annual international conference on mobile computing and networking (MOBICOM'2003), San Diego, California, USA; 2003.
- [6] Cathal McDaid. Routing connections in bluetooth. BS project. Ireland: University of Limerick; 2000. June.
- [7] Topal T. Constructing efficient bluetooth scatternets. Master thesis, Institute of Engineering and Science, Bilkent University; January 2004.
- [8] Liu Y, Lee MJ, Saadawi TN. A bluetooth scatternet route structure for multi-hop ad hoc networks. IEEE J Sel Areas Commun 2003;21:229–39.
- [9] Shek LLY, Kwok YK. Efficient multihop communications in bluetooth scatternets. In: Proceeding of the 14th IEEE international symposium on personal, indoor, and mobile radio communication (PIMRC'2003), Beijing, China, vol. 1; 2003. p. 755–9.
- [10] Johnson David B, Maltz David A. Dynamic source routing in ad hoc wireless networks. In: Imielinski, Korth, editors. Mobile computing, vol. 353(5); 1969. p. 153–81.
- [11] Perkins CE, Royer EM, Das S. Ad hoc on demand distance vector routing (AODV). In: Proceeding of the second IEEE workshop on mobile computing systems and applications, New Orleans; February 1999. p. 90–100.
- [12] Bhagwat P, Segall Adrian. A routing vector method (RVM) for routing in bluetooth scatternets. In: Proceeding of the sixth IEEE international workshop on mobile multimedia communications (MOMUC'99), San Diego, USA; November 1999. p. 375–9.
- [13] L-Garcia A, Widjaja I. Communication networks. McGraw-Hill; 2000. p. 502.
- [14] Kargl Frank, Ribhegge Stefan, Schlott Stefan, Weber Michael. Bluetooth-based ad-hoc networks for voice transmission. In: Proceeding of 36th annual Hawaiian international conference on system sciences; January 2003. p. 9.

- [15] Huang Leping, Sivakumar TVLN, Chen Hongyuan, Kashima Tuyoshi, Hirase Yoshiya, Nakagawa Yoshikatsu. Evaluation of ad-hoc on-demand distance vector routing over bluetooth personal area network. Tokyo, Japan: Wireless Communication Group, Nokia Research Center; 2002. September.
- [16] Huang L, Chen H, Sivakumar T, Sezaki K. Cross-layer optimized routing for bluetooth personal area network. In: Proceeding of 13th international conference on computer communications and networks (ICCCN 2004); October 2004. p. 155-60.
- [17] Prabhu BJ, Chockalingam A. A routing protocol and energy efficient techniques in bluetooth scatternets. In: Proceeding of the IEEE international conference on communication (ICC'02), vol. 5; 2002. p. 3336-40.
- [18] Al-Jarrah O, Sa'deh W. Enhanced AODV routing protocol for mobile ad hoc networks. In: 16th international conference on computer theory and applications (ICCTA); 2006. p. 61-8.
- [19] Law Ching, Mehta Amar K, Siu Kai-Yeung. A new bluetooth scatternet formation protocol. Mobile Networks Appl 2003;8:485-98.
- [20] Wang Qihe, Agrawal Dharma P. A dichotomized rendezvous algorithm for mesh bluetooth scatternets. Ad Hoc Sensor Wireless Networks 2005;1:65-88.
- [21] University of Cincinnati Bluetooth Simulator, UCBT. [Online]. <[http://www.eecs.uc.edu/\\_cdmc/ucbt/](http://www.eecs.uc.edu/_cdmc/ucbt/)>.