An On-Demand Bluetooth Scatternet Formation and Routing Protocol for Wireless Sensor Networks *

Xin Zhang, George F. Riley
Department of ECE
Georgia Institute of Technology Atlanta, GA 30332-0250

Abstract

Bluetooth is a promising short-range wireless communication technology with the characteristics of interference resilience and power efficiency which are required by wireless sensor networks. As an enhanced sensor node platform, the Intel Mote uses Bluetoth as its radio scheme and MAC protocol. However, most of the existing scatternet formation protocols intend to connect all the nodes within the networks regardless of traffic pattern. For wireless sensor network applications with traffic of low duty cycle, on-demand scatternet formation and routing achieve significant power saving by avoiding to maintain the entire network connectivity. We propose in this paper an on-demand scatternet and route formation protocol for Bluetooth-based wireless sensor networks. It is able to cope with multiple sources initiating traffic simultaneously in densely deployed wireless sensor networks. We also introduce a modified Inquiry scheme using extended ID packet for power efficient propagation of route request messages. Furthermore, we propose a mechanism employing POLL packets in Page processes to transfer scatternet formation and route reply information without extra overhead. Simulation results show that our on-demand scatternet formation and routing protocol can provide multihop channels with reasonable delay for Bluetooth-base wireless sensor networks.

1. Introduction

The collaboration of advanced micro–sensing technology and numerous novel applications have driven lots of research work on wireless sensor networks (WSNs) [1]. Bluetooth [2] is a short-range wireless technology based on time-division multiple access (TDMA) and frequency hopping spread spectrum (FHSS). The property of interference resilience makes Bluetooth an applicable radio and MAC layer candidate for WSNs.

In [3, 4], various advantages and limitations are discussed concerning Bluetooth-based sensor networks. The main advantage of employing Bluetooth is all sensor nodes within radio range can use separate channels to avoid interference instead of competing for a shared channel. In addition, Bluetooth's low power modes allow the radio to enter power saving states when there is no active communication. Given these desirable properties for wireless sensor network applications, BTnodes [5] from ETH Zurich use Bluetooth to prototype WSN applications; and Intel has developed Intel Mote [6] based on Bluetooth as an enhanced WSN node platform.

The applications for WSNs encompass various realms with diverse requirements. The protocols for WSNs are more application-driven rather than universal. An important class of WSN applications have a multi-level network architecture with a large number of sensor nodes dispersed within an area and communicating to the external network through a sink node. The communication between the sensor nodes and the sink node is also multihop. Typical WSN applications with such structure include habitat monitoring [7, 8], civil infrastructure health monitoring [9], data collection [10], etc. In these applications, data transfers appear infrequently with few unpredictable bursts, which are exactly the niche for Bluetooth-base sensor networks [3, 4]. Bluetooth connections for such applications are established on-demand depending on the traffic requirements and torn down when no traffic is going on in order to save power.

One of the open issues for the Bluetooth-based sensor networks to work effectively is to support efficient scatternet formation and routing for a multihop network. This problem becomes even more complicated when multiple sources start route discovery concurrently. In this paper, we propose an on-demand scatternet and route formation protocol including a modified *Inquiry* scheme with extended ID (EID) packets for route discovery, using modified *POLL* packet in *Page* for scatternet formation and route reply information propagation, and constructing cross routes for

* This work is supported in part by NSF under contract numbers ECS-0225417, ANI-9977544.

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

multiple sources initiating traffic at the same time. With our protocol, efficient multihop communication can be achieved by sensor nodes equipped with Bluetooth. The cross route formation for sources with concurrent traffic is essential for densely deployed sensor networks. The data aggregation within the network also benefits from the concurrent process when data from multiple sources are correlated.

The remainder of this paper is organized as follows. Section 2 gives the related work of Bluetooth scatternet formation schemes. In section 3, we describe the detailed on-demand scatternet and route formation protocol. In section 4, we present the results and analysis of the simulation experiments for performance evaluation. Finally, the conclusions and future work are discussed in section 5.

2. Related Work

In the literature of Bluetooth scatternet formation, the major solutions can be categorized as proactive and on-demand mechanisms.

Bluetooth is initially designed as a cable interconnect replacement technology. Thus, connectivity is the concern for most of the existing work on multihop construction (scatternet formation). The performance comparison of the proactive scatternet formation protocols is presented in [11].

For wireless sensor networks with traffic of low duty cycle, maintaining the connection of the entire network is a significant power drain. Hence, on-demand scatternet formation is more feasible. To the best of our knowledge, the only existing work addressing Bluetooth on-demand scatternet formation are presented in [12, 13, 15].

In [12], an extended ID (EID) connectionless broadcast mechanism is introduced. The route discovery delay is greatly reduced compared with traditional Bluetooth broadcast in L2CAP layer. However, ID packet in Bluetooth is designed to be small initially in order to save power since the number of ID packets transmitted in Bluetooth *Inquiry* phase is very large (two ID packets per $625\mu s$ time slot). Substituting all ID packets with much longer EID packets to transfer source information in scatternet formation is unnecessary and power consuming because most of the ID packets are just for neighbor probing and synchronization. In addition, the simulations of route discovery in [12] only consider the scenario of single source. When multiple sources in the network initiate the scatternet formation and route discovery simultaneously involving common intermediate nodes, they will interfere with each other and degrade the performance significantly. In this case, the concurrent instead of consecutive cross routes formation needs to be addressed.

Another on-demand Bluetooth scatternet formation algorithm (ODBT) is presented in [13]. ODBT constructs a scatternet with a tree topology. It is an extension to Bluetree [14] with the ability to cope with Bluetooth devices dynamically joining and leaving the scatternet. However, it still tries to connect all the nodes within the network and can not operate in the presence of multiple sources simultaneously starting the formation of a scatternet involving the same Bluetooth devices.

In [15], a two-phase scatternet formation (TPSF) protocol is introduced to support dynamic topology changes. A control scatternet is constructed in the first phase to support topology changes and route determination while an on-demand scatternet is created in the second phase whenever data communication is needed. Maintaining the control scatternet constructed in the first phase is power consuming and makes TPSF similar to proactive approaches.

The on-demand scatternet formation protocol we proposed in this paper shares some common points with [12]. The Baseband layer broadcast, instead of the L2CAP layer broadcast, is used for the flooding of scatternet and route formation requests in order to achieve greatly reduced delay. However, we propose a modified *Inquiry* in the route discovery phase using EID packets for power saving. Moreover, we employ modified *POLL* packets in *Page* during route reply phase. Most importantly, other than the on-demand scatternet formation work mentioned above, we provide the mechanism to deal with multiple sources initiating the formation of a scatternet simultaneously, which is essential for dense sensor networks and makes in-network data processing simple. The detailed description of the protocol is presented in section 3.

3. On-Demand Scatternet and Route Formation Protocol

As we mentioned in section 1, no protocol in WSNs is universal, but application oriented. We consider a typical WSN architecture as shown in Figure 1 of [7], which is popular in habitat and environment monitoring, data collection, etc. Two classes of Bluetooth nodes exist in the network, high power sink node and low power motes such as Intel motes [6]. Sensor motes communicate with the sink in order to send data to the external network such as Internet. Since the sink may not be in the radio range of all the motes, a Bluetooth scatternet must be formed. Since the traffic from the motes is not continuous, on-demand formation makes the scatternet traffic dependent and power efficient with reasonable formation delay.

3.1. Overview

Bluetooth specification defines a network with *MASTER/SLAVE* structure. To interconnect Bluetooth devices into a scatternet, some devices need to act as bridges and

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

participate two adjacent piconets alternately. In our scatternet formation, *SLAVE/SLAVE* bridges are chosen to reduce the number of piconets within a scatternet. Hence, a structure of strict alternating of *MASTER/SLAVE* is maintained along a route from the source to the sink.

Instead of running a routing protocol after scatternet formation completion, we combine these two processes. The overview of our protocol is depicted in Figure 1. The route requests propagate through *Inquiry* broadcast and are relayed from sources to the sink while the scatternet formation and route replies are delivered in the opposite direction by *Page* messages. Since the common destination is the sink, all route requests arriving at the same intermediate node (e.g. node 3 in Figure 1) are merged, which avoids redundant request transmissions as well as makes nodes on cross routes join the same piconets as often as possible (e.g. node 2 and 6 join the same piconet). In addition, the support for concurrent cross routes formation also makes data aggregation easier, which is important for WSNs. The intermediate nodes buffer all their last hop nodes' device address and clock values in order to *Page* them when route replies come back. After a Bluetooth device discover a route to the sink, the next hop information is cached for a period of time. Before the cached route to the sink expires, if new route requests arrive, the next hop node will be paged first. The neighbor information cached at node 3 in Figure 1 is also shown. The detailed processes of scatternet and route formation are discussed in the following sections.

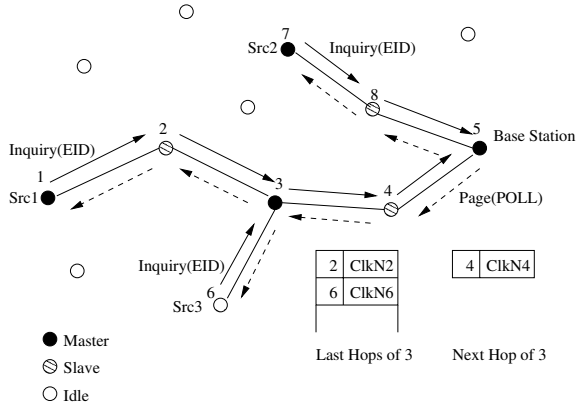

Figure 1. Scatternet and route formation

3.2. Route Request

3.2.1. Extended ID Packet and Modified Inquiry In Bluetooth *Inquiry*, the *MASTER* is able to get the device addresses and clock values of the *SLAVEs* while *SLAVEs* have no information about the *MASTER*. In order to propagate source information in the downstream direction from the source to the sink during scatternet formation and route discovery phase, we propose an extended ID packet (EID) structure. EID packets in our scheme are used in the modified *Inquiry* rather than replacing the original ID packets as in [12, 15]. Each field and their corresponding length in EID packet are shown in Figure 2. The *SrcAddr* is the Bluetooth device address of the node initiating the scatterent and route formation. The fields of *LastHop* and *LastClk* denote Bluetooth device address and native clock of the immediate upstream node respectively. They are useful for route reply transmission. *Seqn* is used to distinguish old and new requests from the same source while *HopCount* limits the number of hops a request traverses.

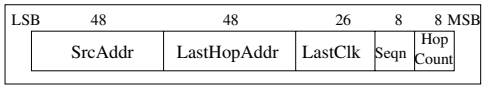

Figure 2. EID packet format

The *Inquiry* process is modified to accommodate the introduction of EID packet. The modified *Inquiry* process is illustrated in Figure 3. We define a new node state type (*ScatType*) for scatternet and route formation to indicate which phase a node is in during scatternet formation process. Initially, all the other nodes are in the *ScatType IDLE* state, excepting the *SOURCE* and *DEST* nodes. An EID packet is sent by the upstream node after receiving the FHS packet. In this case, the downstream node can get the source and last hop information in *Inquiry* process while small size ID packets are still used for the large number of neighbor and synchronization probes. Upon receiving the EID packet, the intermediate nodes go to *ScatType BRIDGE* and function as relays.

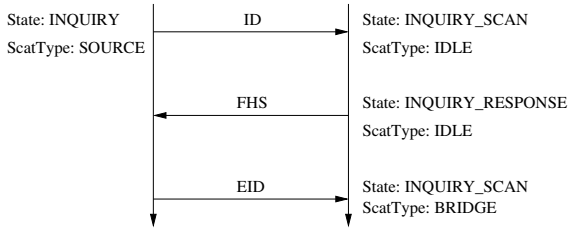

Figure 3. Modified Inquiry process

3.2.2. Route Request Forwarding The pseudocode of route request flooding in the forward direction of scatternet and route formation from the source to the sink is given in Figure 4. When a source has traffic to send to the sink,

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

it starts a scatternet and route formation by transmitting *Inquiry* ID packets to search bridge nodes in order to reach the sink. The modified *Inquiry* process shown in Figure 3 is in effect. A scatternet formation timer *ScatFormTO* is started. This timer is stopped as soon as the scatternet formation and route reply arrives at the source. Otherwise, a new scatternet formation request with increased *Seqn* in the EID packet will be sent upon *ScatFormTO* expiration. The intermediate nodes receiving EID packets save the information about the source and last hop in a structure, *PrecursorList*. The information in this structure is used to relay route reply to the upstream nodes as well as avoid flooding loops.

After the intermediate nodes get the first EID packet, they go into *ScatType BRIDGE* and initiate their own *Inquiry* process to probe the next hop nodes towards the sink. At *Inquiry* timeout, the nodes switch between *PAGE_SCAN* and *INQUIRY_SCAN* states. The state switch enables the nodes to wait for the scatternet and route formation replies, which are *Page* messages, from the downstream nodes. At the same time, the nodes which are in route discovery can also accept new route requests from other sources. This enables multiple sources to start route requests simultaneously. The period that a node stays in *INQUIRY_SCAN* or *PAGE_SCAN* before state switch affects the performance in term of scatternet formation delay, which will be discussed in section 4.

When a node is in *ScatType BRIDGE* and switches to *INQUIRY_SCAN* state, the arriving EID packets from new source requests will be saved in the *PrecursorList*. However, no *Inquiry* messages are generated for these requests since the node is already in route discovery process waiting for the reply. If an EID packet arrives at a node from a source already in the node's *PrecursorList* but traverse through a different route, the *HopCount* field in the EID packet is compared to the entry buffered in the *PrecursorList*. The last hop information for the source is updated to the route with shorter path. For EID packet with the same source address, but larger *Seqn*, a new *Inquiry* message is generated since the old request gets no response because of packet loss or *Inquiry/Page* failure. The route request flooding loop is also avoided by comparing the address of the source node in the received EID packets and the receiving node's own address.

As the scatternet and route formation request arrives at the sink, the route reply will propagate in the reverse direction of the route request and the scatternet will be formed hop by hop. There is a delay between the first route request (EID packet) arrival at the sink and the initiation of route reply and scatternet formation. This short delay enables multiple requests arrive at the sink and share the same scatternet formation of the sink's immediate hop. In addition, during this delay period, requests from the same source but via different paths can arrive at the sink. In this case, the route with smaller *HopCount* will be chosen, which decreases the number of piconets on a route.

```
Route_Request_Propagate()
  Initialize();
    Source: State = INQUIRY, ScatType = SOURCE;
    Dest: State = INQUIRY_SCAN, ScatType = DEST;
    Others: State = INQUIRY_SCAN, ScatType = IDLE;
  Start ScatFormTO;
  RecvEIDPacket();
    switch(ScatType)
      case IDLE:     //fi rst recv EID packet
          Save EID info in PrecursorList;
          ScatType = BRIDGE;
          Inquiry();  //forward route request
          INQUIRY_SCAN/PAGE_SCAN switch on Inquiry timeout;
      case BRIDGE:     //already in route discovery
          if(new source request)
            Save EID info in PrecursorList;
          else if(existing source with smaller HopCount)
            Update PrecursorList;
          else if(existing source with larger Seqn)
            Update PrecursorList;
            Inquiry();
      case DEST:     //arrive at dest
          if(new source request)
            Save EID info in PrecursorList;
          else if(existing source with smaller HopCount or larger Seqn)
            Update PrecursorList;
          Route_Reply_Propagate();
    end;
end;
```

Figure 4. Pseudocode of route request forwarding

3.3. Scatternet Formation and Route Reply

Upon receiving the route request EID packet and after a short delay, the sink responses with a route reply propagating in *Page* message and initiates the scatternet formation.

3.3.1. Modified Page In our on-demand scatternet formation scheme, a strict *MASTER/SLAVE* role alternate is maintained along any route from a source to the sink. To transfer the scatternet role assignment information along a route without extra route reply messages, we propose a new *Page* procedure with modified *POLL* packets. In a scatternet with alternate *MASTER/SLAVE* role (*ScatRole*), the active member address (*AMAddr*) in the *POLL* packet header assigned by the nodes with *ScatRole* set to *SLAVE* has no meaning since the *Page* from these nodes is only for scatternet formation and route reply information propagation. However, for the nodes with *ScatRole* of *MASTER*, the active member address identifies each active *SLAVE* within the specific piconet. In addition, broadcast packets with active member address of zero are not used in the scatternet connection establishing phase. So when a node with *ScatRole* of *SLAVE*

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

tries to page its last hop nodes to relay the scatternet formation information, the *AMAddr* in the *POLL* packet is set to zero while the *AMAddr* in the *POLL* packet sent from nodes with *ScatRole* of *MASTER* is between 1 and 7. In this case, the upstream nodes in the scatternet route can decide their *ScatRoles* based on whether the *AMAddr* is zero or not. With the modified *Page* scheme, *ScatRole* information is transfered without overhead and there is no effect on the proper operation of the scatternet formation.

3.3.2. Route Reply Propagation and Scatternet Formation The scatternet formation is initiated by the sink. The sink sets its *ScatRole* to *MASTER* and forms its piconet by paging all the last hop nodes in its *PrecursorList* to establish connection channels. The pseudocode of the scatternet formation and route reply propagation is given in Figure 5. The intermediate nodes being paged get their next hop address and clock value through the *FHS* packet in the *Page* process. The *ScatRoles* of the nodes on the scatternet route are determined based on the *AMAddr* field in the received *POLL* packets as stated in section 3.3.1.

After participating the downstream node's piconet and getting information about the next hop's address and clock, the intermediate nodes switch themselves to *Page* state and propagate the scatternet and route information to their upstream nodes. For nodes with *ScatRole* of *SLAVE*, they only page their upstream nodes to transfer scatternet formation and route reply information and then switch to *PAGE_SCAN* state to wait for participating the piconets of their upstream nodes. On the other hand, the nodes with *ScatRole* of *MASTER* page both their last hop nodes and next hop node to form their own piconets.

To avoid multiple nodes with *ScatRole* of *MASTER* paging the same next hop node simultaneously, a random backoff is used. The scatternet formation for these *MASTERs* starts on backoff timeout.

```
Route_Reply_Propagate()
  Dest: ScatRole = MASTER, Page(all nodes in PrecursorList);
  BRIDGE and SOURCE:
    RecvFHSPacket();
      Get NextHop, NextClk from FHS packet;
    RecvPOLLPacket();
      if(AMAddr = 0) //next hop is slave
        ScatRole = MASTER;
      else //next hop is master
        ScatRole = SLAVE;
      SendNULLPacket();
      if(first POLL packet for scatternet formation)
        Random backoff to schedule StartScatForm();
    RecvNULLPacket();
      if(more last hop/next hop nodes need page)
        Page(last hop/next hop nodes);
      else if(ScatRole = SLAVE)
        State = PAGE_SCAN;
    Timeout for random backoff: StartScatForm();
      StartScatForm();
        if(ScatRole = MASTER)
          Page(all nodes in PrecursorList and NextHop);
        else if(ScatRole = SLAVE)
          Page(all nodes in PrecursorList with AMAddr = 0);
end;
(AMAddr: Active Member Address)
```

Figure 5. Pseudocode of scatternet formation and route reply

4. Performance Evaluation

In this section, we provide a quantitative evaluation of our on-demand scatternet formation and routing protocol by means of analysis and simulation.

4.1. EID Power Saving

ID packet in Bluetooth is designed to be small with the size of 68 bits in order to be power efficient because they are transmitted frequently during *Inquiry* and *Page*. Extending the packet length in EID packet increases power consumption for transmission and reception of single packet. Therefore, reducing the number of EID packets and transmitting them only when they are necessary is essential for power saving. Comparing to [12] and [15], which also introduce types of EID packets and substituting ID packets completely with EID packets during *Inquiry*, we keep ID packets for neighbor probing in *Inquiry* and only transmit EID packets when the source and last hop information is necessary for the scatternet formation by a modified *Inquiry*.

The power saving of our proposed *Inquiry* scheme is related to the number of ID packets transmitted during *Inquiry* and the number of nodes participating the scatternet formation in the network.

The length of the period for a node to stay in *Inquiry* state depends on the parameter of *INQUIRY_TIMEOUT*. According to the Bluetooth specification [2], the *MASTER* needs to stay at the *Inquiry* state for 10.24s to collect enough responses from its neighbors. However, the time to get enough neighbors varies significantly depending on the alignments of device clocks. Simulations reveal that 5s is sufficient most of the time.

The current consumption [1] comparison of the modified *Inquiry* with EID packet to the *Inquiry* with EID substituting ID packets for one node is shown in Figure 6. The current consumption goes up linearly as the time for *Inquiry* increases due to more EID packets transmission. However, the current consumption of our scheme achieves great saving for EID packets are transmitted instead of ID packets only when they are necessary to transfer source and last hop information. With *INQUIRY_TIMEOUT* equal to

1 The current drain for a typical Bluetooth device to transmit an ID packet is 26.5mA while it is 39.8mA for an EID packet.

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

10.24s, 33.41% current saving is achieved by our modified *Inquiry* scheme. In wireless sensor networks with a large number of sensor nodes, this saving is significant.

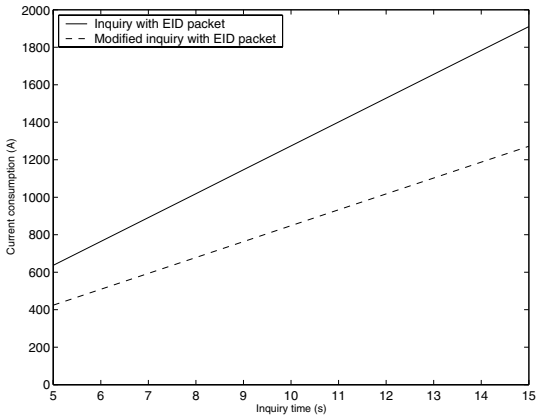

Figure 6. Current consumption of Inquiry

4.2. Scatternet Formation Delay

Besides power efficiency in route request phase, our intention of choosing an on-demand rather than a proactive scatternet formation scheme is to conserve power on connection maintenance of the entire network. The tradeoff of the on-demand scatternet formation scheme is delay. To measure the scatternet formation delay quantitatively, we implement our scatternet formation protocol in *GTNetS* [16, 17], a packet level simulator for large scale network simulation. In our previous work we have designed and implemented a detailed Bluetooth model for *GTNetS* [18]. The network topology we choose for our simulations is shown in Figure 7. This grid topology has one sink and multiple sources. Every node not residing on edges has eight neighbors within its radio range. It is a subnet of a typical monitoring or data collection sensor network topology with all possible source distributions relative to the sink. The source number can be varied.

With sources starting scatternet and route formation simultaneously and sharing some intermediate nodes, the value of the switch timeout (*SwitchTO*), which controls the alternate of *INQUIRY_SCAN* and *PAGE_SCAN* states after routing request forwarding, has significant effect on the scatternet formation delay. In Figure 8, we vary *SwitchTO* from 0.16s to 5.12s to measure the maximum scatternet formation delay of all the sources. Figure 8 shows that the scatternet formation with our simultaneous processes for cross routes achieves greatly reduced formation delay compared to the serial formation process. In a serial formation process, sources sharing the same intermediate nodes have

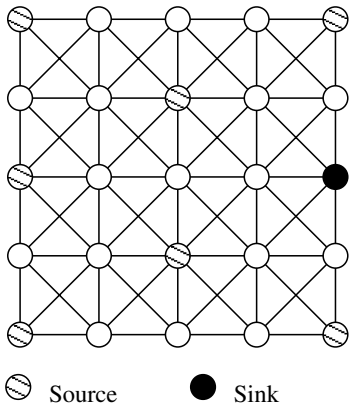

Figure 7. Multihop network topology

to wait for the completion of the scatternet and route formation for other sources, which is a considerable delay. In our scatternet formation protocol, after an intermediate node forwards route request for a source and is waiting for the reply, it switches between *INQUIRY_SCAN* state to listen for new source requests and *PAGE_SCAN* state in case the scatternet formation and route reply comes back. The *SwitchTO* value of 0.64s achieves the best performance for the maximum scatternet formation delay. At that point, the maximum delay for scatternet formation protocol with concurrent process is only 47.94% of the serial process. The optimal *SwitchTO* value occurs at 0.64s. This is because when the timeout value is too small, the node switches frequently between the two states and stays a very short period of time in each state. Bluetooth uses frequency hopping spread spectrum and the switch interval is not enough for the *Inquiry* or *Page* nodes to hop to the frequencies that the scanning nodes are listening to. If the timeout value is too large, when the scatternet formation *Page* message arrives at a node switching to *INQUIRY_SCAN* state, it may take some time for it to switch back to *PAGE_SCAN*, which increases the formation delay. This optimal *SwitchTO* is also coincident with the average page delay, which is half of the *PAGE_SCAN* window (1.28s).

While the maximum scatternet formation delay measures the longest time for one source to finish the sctternet formation, the total delay gives the sum of the formation delay of all the sources. Figure 9 shows that the total delay with concurrent process is less than the case with serial process (75.99% with *SwitchTO* of 0.64s) when the *SwitchTO* value is less than 2.56s, which is the value for *PAGE_TO*. With *SwitchTO* larger than *PAGE_TO*, the state switch detains the scatternet formation. The total formation delay goes up as the *SwitchTO* increases, which is different from the maximum formation delay shown in Figure 8. This is because when the *SwitchTO* value is 0.64s, some sources with small number of hops to the sink sacrifice their

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

own formation delays which increase the total formation delay, but benefit the nodes with large number of hops to the sink to complete the scatternet formation process promptly.

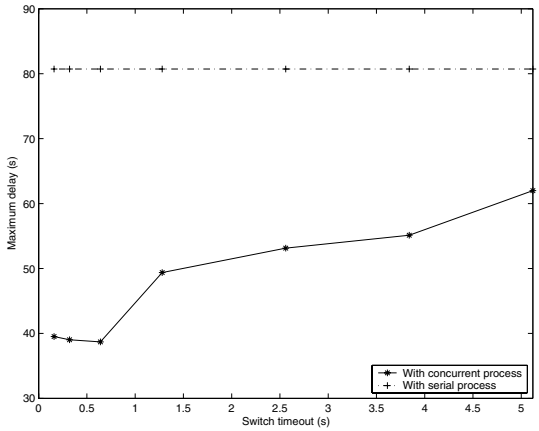

Figure 8. Maximum scatternet formation delay vs. switch timeout

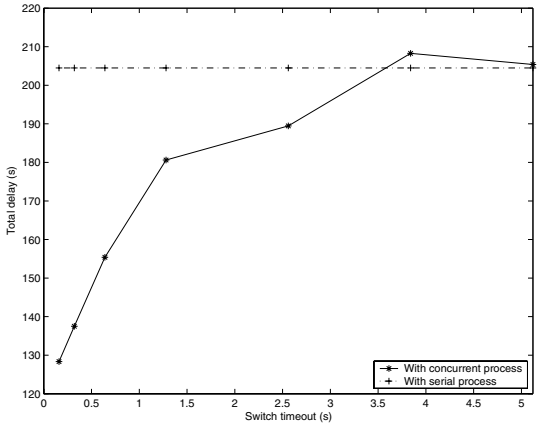

Figure 9. Total scatternet formation delay vs. switch timeout

Another parameter affecting the performance of the scatternet formation which must be tuned carefully is the timeout value for the scatternet formation (*ScatFormTO*). The source nodes initiate new scatternet formation requests if no scatternet formation responses arrive and *ScatFormTOs* expire. If the *ScatFormTO* is set to be too small, new requests are sent out before normal replies come back. Thus, the scatternet formation is initiated repeatedly without any success. On the other hand, setting *ScatFormTO* too large may incur unnecessary delay to wait for the scatternet formation timeout and recover from failures. We vary the *ScatFormTO* from 10s to 30s to look for the optimal selection. The maximum scatternet formation delay and total delay of all sources are presented in Figure 10 and 11 respectively. Both figures show that the delays keep stable when the *ScatFormTO* is less than 20s and increase significantly with the *ScatFormTO* larger than 20s. This is due to the time spent on waiting in vain for a timeout. Although the delays for the *ScatFormTO* less than 20s are small, we found that with the *ScatFormTO* less than 17.5s, there are chances for some sources with large number of hops away from the sink to form the scatternet unsuccessfully due to very small value of *ScatFormTO*. Therefore, 17.5s to 20s is an optimal range for the *ScatFormTO* in this network.

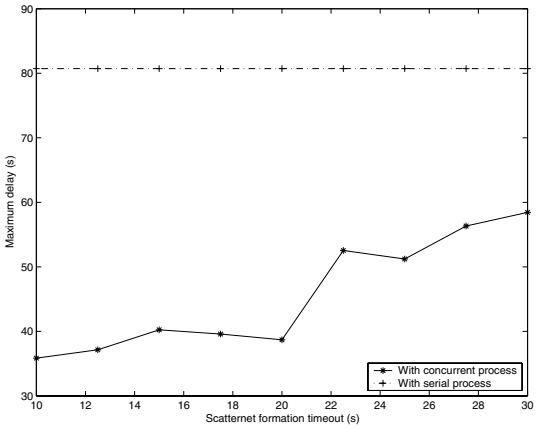

Figure 10. Maximum scatternet formation delay vs. scatternet formation timeout

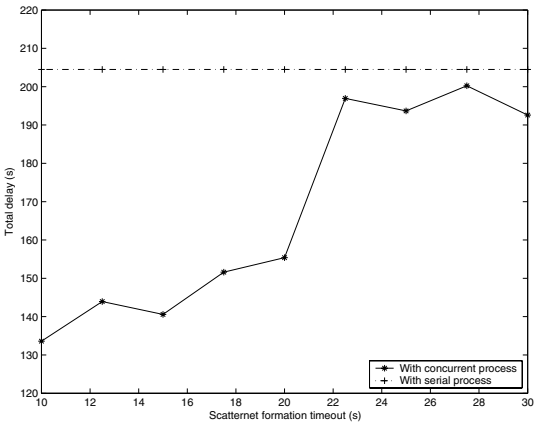

Figure 11. Total scatternet formation delay vs. scatternet formation timeout

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 IEEE

5. Conclusions

We have introduced an on-demand scatternet formation and routing protocol used for Bluetooth-based wireless sensor network applications. We deal with the complicated problem of supporting multiple sources that initiate the scatternet and route formation involving common Bluetooth devices at the same time. In addition, we introduce a modified *Inquiry* scheme with extended ID (EID) packet for route request propagation. We show the power efficiency of this scheme comparing with traditional *Inquiry* with EID packet. Furthermore, we propose a mechanism using *POLL* packet in *Page* to transfer scatternet formation information without extra expense. Simulation results demonstrate that our protocol achieves significant improvement in scatternet formation delay when compared to serial scatternet formation for multiple sources with concurrent traffic. It meets the requirements of Bluetooth-based wireless sensor networks in terms of power efficiency because of on-demand rather than proactive approach in scatternet formation. At the same time, the protocol doesn't incur large scatternet formation delay.

Ongoing research is to investigate the performance under dynamic adjustment of the scatternet formation parameters. Another future effort is to study the effect of route cache in scatternet formation and route requests. We are further extending this protocol with an effective scheduling scheme after the completion of scatternet formation and testing the protocol on real sensor motes.

References

[1] F. Zhao and L. Guibas, *Wireless sensor networks: an information processing approach.* San Francisco, CA: Morgan Kaufmann Publishers, 2004.

[2] Bluetooth SIG, "Bluetooth specifi cation version 1.1," Available HTTP: http://www.bluetooth.com.

[3] M. Leopold, M.B. Dydensborg, and P. Bonnet, "Bluetooth and sensor networks: a reality check," in *Proc. 1st International Conference on Embedded Networked Sensor Systems,* 2003, pp. 103-113.

[4] P. Bonnet, A. Beaufour, M.B. Dydensborg, and M. Leopold, "Bluetooth-based sensor networks," *ACM SIGMOD Record, Special Issue: Special Section on Sensor Network Technology and Sensor Data Management,* vol. 32, no. 4, pp. 35-40, 2003.

[5] J. Beutel, O. Kasten, F. Mattern, K. Romer, F. Siegemund, and L. Thiele, "Prototyping wireless sensor network applications with BTnodes," in *Proc. 1st IEEE European Workshop on Wireless Sensor Networks (EWSN),* 2004, pp. 323-338.

[6] R.M. Kling, "Intel Motes: an enhanced sensor network node," in *Proc. International Workshop on advanced sensors, structural health monitoring, and smart structures,* 2003.

[7] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA),* 2002, pp. 88-97.

[8] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *Proc. 1st European Workshop on Wireless Sensor Networks (EWSN),* 2004, pp. 307-322.

[9] V. Mehta, M.E. Zarki, "A Bluetooth based sensor network for civil infrastructure health monitoring," in *ACM/Kluwer Journal on Wireless Networks (WINET),* vol. 10, no. 4, pp. 401-412, 2004.

[10] M. Handy, J. Blumenthal, and D. Timmermann, "Energy-effi cient data collection for Bluetooth-based sensor networks," in *Proc. IEEE Instrumentation and Measurement Technology Conference (IMTC),* 2004.

[11] S. Basagni, R. Rbuno, G. Mambrini, and C. Petrioli, "Comparative performance evaluation of scatternet formation protocols for networks of Bluetooth devices," in *ACM/Kluwer Journal on Wireless Networks (WINET),* vol. 10, no. 2, pp. 197-213, 2004.

[12] Y. Liu, M.J. Lee, and T.N. Saadawi, "A Bluetooth scatternet-route structure for multihop ad hoc networks," *IEEE JSAC,* vol. 21, no. 2, pp. 229-239, 2003.

[13] E. Pagani, G.P. Rossi, and S. Tebaldi, "An on-demand bluetooth scatternet formation algorithm," in *Proc. 1st IFIP Working Conference on Wireless On-demand Network Systems (WONS),* 2004, pp. 130-143.

[14] G. Zaruba, S. Basagni, and I. Chlamtac, "Bluetree - scatternet formation to enable Bluetooth-based personal area networks," in *Proc. IEEE International Conference on Communications (ICC),* 2001, pp. 273-277.

[15] Y. Kawamoto, V. Wong, and V. Leung, "A two-phase scatternet formation protocol for bluetooth wireless personal area networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC),* 2003, pp. 1453-1458.

[16] G.F. Riley, "The Georgia Tech network simulator," in *Proc. ACM SIGCOMM Workshop on Models, Methods, and Tools for Reproducible Network Research,* 2003, pp. 5-12.

[17] G.F. Riley, "Large-scale network simulations with GTNetS," in *Proc. 2003 Winter Simulation Conference,* 2003, pp. 676-684.

[18] X. Zhang, G.F. Riley, "Bluetooth simulations for wireless sensor networks using GTNetS," in *Proc. 12th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS),* 2004, pp. 375-382.

Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)
0-7695-2294-7/05 $20.00 © 2005 **IEEE**