Adaptive Role Switching Protocol for Improving Scatternet Performance in Bluetooth Radio Networks

Chih-Yung Chang, Hsu-Ruey Chang

Abstract —*Bluetooth has been considered as a high potential technology for providing wireless communication in a home-networking environment. In a Bluetooth network, it is difficult to control or predefine a scatternet structure because that the scatternet is formed using a distributed procedure, with the master and slave connected at random. A badly structured scatternet exhibits the following characteristics. Firstly, too many bridges in the scatternet will create a guard slot overhead associated with bridge switching among the participated piconets, increasing the probability of packet loss. Secondly, too many piconets in a communicative range will cause packet collision and thus degrade the performance. Unnecessary piconets also lengthen the routing path and transmission delay. This work proposes a distributed scatternet reconstruction protocol for dynamically reorganizing the scatternet topology. By applying the role switching operation, the unnecessary bridges and the piconet can be dynamically removed and hence, improve the packet error rate, save guard slots, and reduce the average routing length. Experimental results reveal that the proposed protocol significantly improves the performance of a Bluetooth scatternet.*

Keywords — Bluetooth, Role Switching, Scatternet, Piconet.

I. INTRODUCTION

Bluetooth [1] is a low-power, low-cost and short-range wireless technology. Bluetooth devices can communicate with each other in the unlicensed ISM band at 2.4GHz. Equipments such as Printer, Desktop PC, home video and audio systems, access point, and cordless phone have embedded with Bluetooth chip to support the wireless communication. Moreover, now-a-days most of the mobile devices, including the mobile phones, mp3 player, digital camera, video camera, PDA, Laptop PC, Tablet PC and so forth, are equipped with Bluetooth, which is highly useful to m-commerce scenarios. Bluetooth has been considered as a high potential technology for providing wireless communication in a home-networking environment and has received extensive attraction in recent years.

[1] This work was supported in part by the National Science Council of the Republic of China under Contract No. NSC 94-2213-E-032-012.

Chih-Yung Chang is with Department of Computer Science and Information Engineering, Tamkang University, 25137 Taiwan (e-mail: cychang@mail.tku.edu.tw).

Hsu-Ruey Chang is with Department of Computer Science and Information Engineering, Tamkang University, 25137 Taiwan (e-mail: hrchang@wireless.cs.tku.edu.tw).

To combat interference and fading, Bluetooth technology applies a fast frequency-hopping scheme which hops over 79 channels 1600 times per second. A *piconet* consists of a master and at most seven active slaves. Each piconet has its own hopping sequence and the master and all slaves share the same channel. In a piconet, the master and slave devices transmit packets in the even and odd slots, respectively. A device that participates in multiple piconets is called a *bridge* which relays packets from one piconet to another. Several piconets connected by common bridges form a connected *scatternet*. Figure 1 demonstrates a scatternet structure in which devices *a* and *c* play the master role and device *b* plays the bridge role.

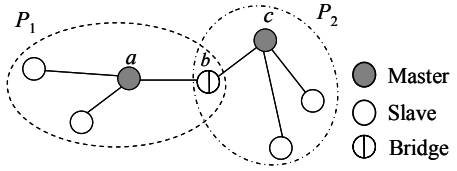


Fig. 1: The scatternet structure of a Bluetooth radio network.

Recently, a comprehensive study proposed scatternet formation protocols [2][3][4] for constructing a connected scatternet. Most protocols seek to reduce the scatternet formation time or increase the probability of constructing a connected scatternet. A device connects to another device at random, according to their 48-bit Bluetooth addresses and clocks, which control the hopping behavior in inquiry or inquiry scan states. The number of piconets and bridges and the role of each device are difficult to control or predefine, because the scatternet is formed by a distributed procedure, with the master and slaves connected at random.

A well structured scatternet has the appropriate number of piconets in a communicative range and the proper number of bridges, and every device has its proper role. Each piconet has its own hopping sequence. Increasing the number of piconets will enhance the channel utilization. However, simultaneous transmissions of two piconets on the same channel will result in the packets collision. As the number of piconets is increased, their hopping sequences are likely to collide with each other [5][6][7] frequently, resulting transmission collision and reducing the performance of the scatternet since more packet retransmissions are required. Besides, increasing the number of piconets also increases the average routing path and hence, increases the power consumption and transmission delay [8][9]. When the packet collision rate exceeds a threshold, appropriately reducing the number of piconets can improve the performance since the packet error rate and the average route length can be improved.

Contributed Paper
Manuscript received August 24, 2006

As well as the number of piconets, the number of bridges is another key determinant of the performance of a scatternet. A bridge can forward data from one piconet to another. The time for bridge switching from one piconet to another is referred to the *guard time*. An excess of bridges cannot improve piconet connections and they generate guard slots. A bridge participating in too many piconets also results in difficult negotiation for scheduling, worsening the packet loss problem and thereby degrading data transmission. Therefore, properly eliminating unnecessary bridges will improve the scatternet performance.

In addition to the number of piconets and bridges, role assignment is another key parameter in determining the performance of a scatternet. Devices with inappropriate roles create new piconets and bridges. *Role switching* is an operation that exchanges the roles of master and slave devices very rapidly. In this paper, each bridge device monitors the parameters including bandwidth requirement, the guard slot overhead and the packet loss rate. While these parameters are considered to be the main reasons that drop the performance, the bridge device will initiate the role switching request to reduce the number of piconets and bridges and adjust its role. This work initially analyzes the effects of role switching on scatternet performance. Then a dynamic role switching protocol is developed to improve the data transmission performance by restructuring the scatternet in a distributive and dynamic manner.

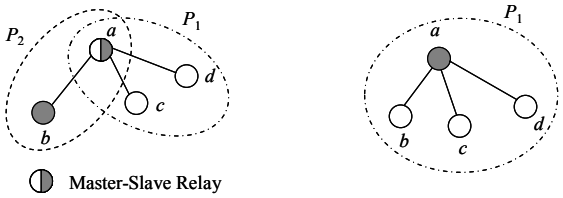
The remainder of this article is organized as follows. Section 2 introduces the basic operations associated with the role switching operation. Section 3 analyzes the usages of role switching operation. Section 4 proposes a distributed role switching protocol useable by devices to determine when and how to gain advantage from executing the role switching operation. Section 5 experimentally examines the proposed protocol. Session 6 draws conclusions.

II. BASIC OPERATIONS OF ROLE SWITCHING

Role switching enables two devices to exchange roles very rapidly, rather then reconnecting by executing the time-consuming inquiry and inquiry scan processes. The role switching operation involves fewer slots than the inquiry/inquiry scan and page/page scan operations in switching the devices' roles. This section proposes three major types of role switching operation and their effects.

Type 1: *Combing Operations*

In most current scatternet formation algorithms, a device alternates between the inquiry and the inquiry Scan states to connect rapidly with other devices in a distributed manner. As depicted in Fig. 2(a), although device *b* intends to act as a slave, during random switching of its state between inquiry and inquiry scan, device *b* may establish a connection in the inquiry state and play a master role. As presented in Fig. 2(a), after the piconet P_2 has been established, devices *b* and *a* act as master and slave, respectively. Hereafter, device *b* may send a role switching request to device *a* so that it can play a slave role in piconet P_1. The role switching operation will combine two piconets P_1 and P_2 into a single piconet $P_{1,}$ as shown in Fig. 2(b). The role switching operation also eliminates the bridge role of device *a*. This type of role switching can reduce the number of bridges and the number of piconets.

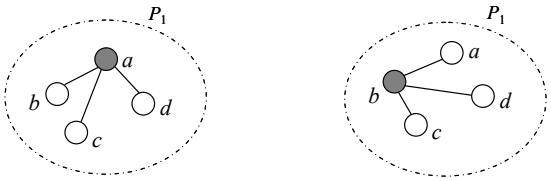


(a) Before executing the combining operation. **(b) After executing the combining operation.**

Fig. 2: The combining operation of role switching.

Type 2: *Splitting Operation*

A role switching operation also can split a piconet into two piconets. As depicted in Fig. 2(b), in piconet P_1, slave device *b* intends to create a new piconet P_2, where device *b* is the master. Device *b* accordingly initiates a role switching request to device *a*. As depicted in Fig. 2(a), device *b* creates a new piconet P_2 and plays a master role in P_2. Device *a* then alternatively participates in two piconets P_1 and P_2 and plays master and slave roles in P_1 and P_2, respectively. This kind of role switching increases the number of piconets and bridge devices and is referred to the *splitting operation*.



(a) Original piconet. **(b) Piconet after executing the *take-over* operation.**

Fig. 3: The execution of *take-over* operation.

Type 3: *Take-Over Operation*

Role switching operation can also be used to enable a slave to take over the resources of its master. For example, as indicated in Fig. 3(a), device *b* initiates a role switching request. Upon receiving the request, device *a* asks all the other slaves including *c* and *d* to stay in the page scan state and transfer their BD_ADDR and clock information to device *b*. Device *b* thus stays in the page state and tries to act as a master, connecting to the slaves that stay in the page scan state. Thereafter, as shown in Fig. 3(b), the master of the constructed piconet will have been changed to device *b*. This type of role switching operation is called the *take-over operation* and enables a slave to take over all slaves that belong to its master.

Since role switching can be applied to Bluetooth devices without entering the inquiry and inquiry scan states, the role switching operation can be cost-effectively applied to dynamically reorganize the scatternet topology.

The following definitions are used in detailing the role switching operation and the proposed protocol.

***Definition*:** Number of Slaves: *NOS*(*m*)

The number of active slaves managed by the master *m* is denoted by *NOS*(*m*). ■

***Definition*:** The role of device *d*: *Role*(*d*)

The function *Role*() returns the role of device *d*. More specifically,

$$Role(d) = \begin{cases} M & \text{if device } d \text{ plays a master role} \\ S & \text{if device } d \text{ plays a slave role} \\ M/S & \text{if device } d \text{ plays the master role in one piconet and plays the slave role in other piconets} \\ S/S & \text{if device } d \text{ plays the slave role in all participated piconets} \end{cases}$$ ■

***Definition*:** Number of Participating Piconets: *NOP*(*d*)

For any device *d*, *NOP*(*d*) refers to the number of piconets in which device *d* participates. In case that *Role*(*d*)=*M* or *S*, value *NOP*(*d*) is one. If *d* is a bridge, *Role*(*d*)=*S*/*S* or *M*/*S*, and *NOP*(*d*)□2.

■

***Definition*:** *Combine*(*b*, *a*), *Split*(*b*, *a*), *TakeOver*(*b*, *a*)

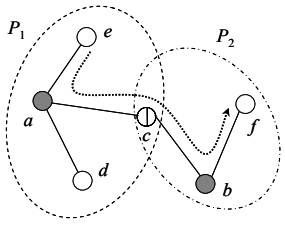
Operations *Combine*(*b*, *a*), *Split*(*b*, *a*), *TakeOver*(*b*, *a*) respectively represent the role switching operations *Combine*, *Splitting*, and *TakeOver*, where *b* represents the device that initiates the request and *a* represents the device that responds to the request. ■

III. Analysis of Role Switching Operations

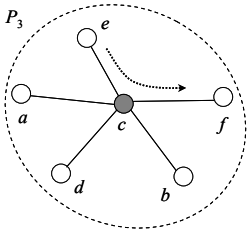
This section analyzes the effect of role switching. Examples are also given to show that applying the role switching operation reduces the packet collision rate, the number of guard slots, and the average length of the routing paths.

Figure 4 shows the advantages obtained from executing a role switching operation. In Fig. 4(a), the bridge device *c* simultaneously participates in piconets P_1 and P_2, which are managed by the masters *a* and *b,* respectively. When bridge *c* initiates a role switching request to devices *a* and *b*, two piconets P_1 and P_2 will be combined into piconet P_3, as shown in Fig. 4(b). The new piconet P_3 contains the new master *c* and the slaves *a*, *b*, *d*, *e*, and *f*. The *TakeOver* operation initiated by device *c* reduces the number of piconets and bridges and hence, reduce the guard time cost and the number of packet collisions.

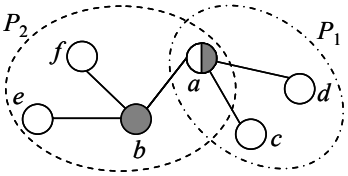
Another purpose of role switching is to reduce the routing length. In a piconet, one slave cannot directly transmit packets to another slave. Rather, the packet should be sent to the master and then be forwarded from the master to another slave. If the source and destination devices belong to different piconets, the packet should pass through a routing path that connects the source and destination piconets. The length of the routing path governs the performance of data transmission along the route. A long path is associated with a long delay and greater power consumption and hence, increases the probability of route breakage and packet loss. Yet another function of role switching operation is to reduce the length of a routing path in a scatternet. As shown in Fig. 4(a), a routing path $e \rightarrow a \rightarrow c \rightarrow b \rightarrow f$ exists between the source device *e* and the destination device *f*. When bridge device *c* initiates *TakeOver* operations *TakeOver*(*c*, *a*) and *TakeOver*(*c*, *b*), the resulting scatternet has the structure depicted in Fig. 4(b), wherein the original routing path has been changed to $e \rightarrow c \rightarrow f$. The number of hops along the route from *e* to *f* has been reduced from four to two. Factors such as power consumption, delay time, packet loss, and routes broken, which impact performance, can be improved.



(a) A poor scatternet structure that causes long routing length from device *e* to device *f*.



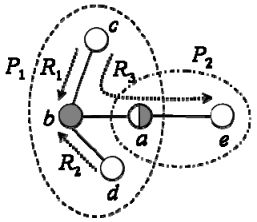
(b) After bridge *c* executing *TakeOver* Operation, routing length from *e* to *f* has been reduced.



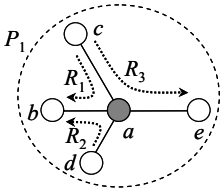
(c) An example that the bridge *a* plays an improper role.

Fig. 4: The advantages obtained from executing a role switching operation.

The role switching operation can also change the relationships among roles in a piconet. For example, in Fig. 4(c), device *a* participates in piconets P_1 and P_2 and plays master and slave roles, respectively. When device *a* switches to piconet P_2, slaves *c* and *d* in piconet P_1 will can not communicate with any other device since their master device *a* is inactive in piconet P_1. The bandwidth utilization falls accordingly. However, the role switching between devices *a* and *c* will solve this problem. Device *c* may take over the piconet and play the master role in P_1. In addition, as the scatternet includes several bridges with poor roles, the role switching operation will save the guard slots and improve the roles.



(a) A scatternet before executing *TakeOver* operation.



(b) A scatternet after executing *TakeOver*(*a, b*) operation.

Fig. 5: An improper role switching may increase the average routing length.

Executing role switching can improve the performance of a scatternet. However, improper execution of role switching

operation negatively impacts the scatternet. Figure 5 shows the problem that is caused by the improper execution of role switching. Figure 5(a) includes three routing paths.

Route R_1: $c \rightarrow b$ R_2: $d \rightarrow b$ R_3: $c \rightarrow b \rightarrow a \rightarrow e$

If bridge *a* initiates a *TakeOver*(*a*, *b*) operation, then the structure of the scatternet will change, as shown in Fig. 5(b). Although the length of R_3 is reduced from three to two, the lengths of both routes R_1 and R_2 increase by one. Therefore, before the role switching operation is executed, it should be estimated whether or not the role switching is appropriate. In the following, an estimation of the route contribution of role switching is discussed.

***Definition*:** *RouteConrtibution*(*d*, *a*)

Let *traffic* T_i of a routing path R_i represent the number of packets that passes through a route in a unit time. Assume that *k* routing paths R_1, R_2, …, R_k pass through device *d*. The route contribution obtained from role switching operations *Combine*(*d*, *a*), *Split*(*d*, *a*), or *TakeOver*(*d*, *a*) is defined as,

$$RouteContribution(d,\ a) = \sum_{i=1}^{k} T_i \Delta_i \quad (1)$$

, where Δ_i denotes the reduction in the length of route R_i obtained by executing the role switching operation. ■

For example, consider Fig. 5 again. After device *a* executes the role switching operation, the lengths of R_1 and R_2 increase by one, which yields Δ_1 = -1 and Δ_2 = -1. The length of R_3 is reduced by 1, and hence Δ_3=1. When the three routes have the same traffic $T_i=\alpha$, for $1 \le i \le 3$, the value of *RouteContribution*(*a*, *b*) can be calculated by, α*(–1) + α*(–1) + α*1= -α, implying that applying role switching on device *d* will have a negative effect.

Role switching not only exchanges the roles of master and slave, but also changes the topology of the piconet, the packet transmission slot and the controller of the piconet. To ensure that role switching improves the performance of the scatternet, each device should analyze the effect of role switching before it initiates the role switching request. Figure 6(a) contains all possible roles including master, slave, *S*/*S* bridge and *M*/*S* bridge and will be used to illustrate the opportunity for applying role switching operation. In the following, different roles are selected to execute the role switching operation to observe their effects.

(1) *Slave executes role switching*

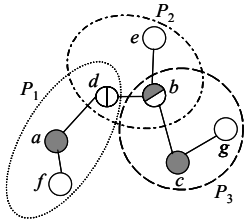
Figure 6(b) shows the scatternet structure obtained from slave *f* executing the *TakeOver*(*f*, *a*) operation. The number of piconets is three, which is the same as that before the *TakeOver* operation was executed. This indicates that the execution of the *TakeOver* operation by a slave device does not change the number of piconets. Executing *TakeOver* operation only changes the master device from device *a* to device *f*. This type of role switching can be applied when a master cannot operate normally, perchance because of power failure or link breakage due to mobility.

(2) *Master executes role switching*

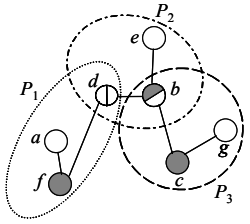
A master initiates a *TakeOver* operation in a manner similar to a slave. For example, in Fig. 6(a), a master *a* initiates a *TakeOver*(*a*, *f*) operation, resulting in the scatternet structure depicted in Fig. 6(b), which is the same as that obtained when slave *f* initiates the *TakeOver*(*f*, *a*) operation. This type of role switching cannot reduce the number of piconets, but the master can select one of its slaves to be the new master.

(3) *M/S bridge executes role switching*

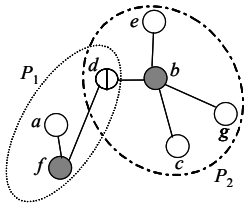
An *M*/*S* bridge plays a master role in one piconet and a slave role in the other piconets. As depicted in Fig. 6(a), device *b* participates in piconets P_2 and P_3 as master and slave, respectively. If device *b* in the slave role initiates a role switching operation to take over all the slaves in piconet P_3, the resultant scatternet structure is as shown in Fig. 6(c). After bridge *b* executes the role switching operation *TakeOver*(*b*, *c*), the number of piconets is reduced from three to two. This example illustrates that an *M*/*S* bridge initiates a role switching operation associate with those piconets wherein the bridges play a slave role can reduce the number of piconets and bridges. The proper execution of this type of role switching operation can alleviate packet loss due to the collision of hopping sequences or save guard time that would otherwise be wasted due to bridge switching among the participated piconets.



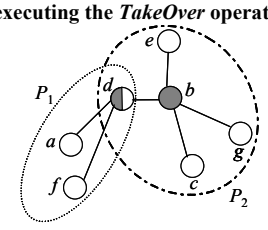
(a) A scatternet that contains combinations of role relations.



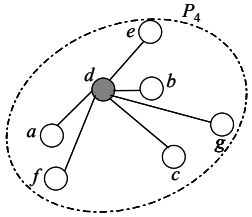
(b) Scatternet structure after slave *f* executing the *TakeOver* operation.



(c) Scatternet structure after *M*/*S* bridge *b* executing the *TakeOver* operation.



(d) Scatternet structure after the *S*/*S* bridge *d* executing the *TakeOver* operation.



(e) Scatternet structure after the *M*/*S* bridge *d* executing the *TakeOver* operation.

Fig. 6: An example to illustrate that different roles initiating the role switching operation will have different effects on scatternet structure.

(4) *S/S bridge executes role switching*

An *S/S* bridge can be changed into an *M/S* bridge by executing the role switching operation *TakeOver*. Consider Fig. 6(c). After device *d* executes a role switching operation *TakeOver*(*d, f*), the resultant scatternet structure is as shown in Fig. 6(d). The device can apply role switching operation *TakeOver*(*d, b*) to take over all of the slaves in piconet P_2. The number of piconets is therefore one, as shown in Fig. 6(e), which indicates that an *S/S* bridge executes the role switching operation to change its role from *S/S* to *M*/*S* can reduce the number of piconets if another role switching operation is performed. Devices with different roles that execute role switching operations will yield differently structured scatternets.

IV. ADAPTIVE ROLE SWITCHING PROTOCOL

The preceding section introduced some good opportunities for executing role switching. Whether or not the execution of the role switching operation is appropriate depends on the role of the initiated device. This section discusses how to obtain full advantages from a role switching operation.

Definition**:** $g_t(\gamma, o)$

Consider a Bluetooth bridge $d \in D$, *Role*(*d*)= *M*/*S* or *S*/*S*. Let γ and o respectively represent the packet error rate and the traffic load for device *d*. A role switching *decision function* $g_t(\gamma, o)$ is defined by the value $g(\gamma, o)$ at time *t*. Each bridge device *d* will maintain $g_t(\gamma, o)$ to determine whether or not it should initiate a role switching request at time *t*. ■

As the packet error rate γ is increased, the role switching operation can effectively reduce the number of piconets. Packet error associated with collisions between hopping sequences can therefore be mitigated. The passing of many packets through a bridge device will cause a bottleneck of inter-piconet packet transmission, therefore another parameter o is included in function g_t. The protocol uses a threshold value δ set to a default value. When $g_t(\gamma, o) > \delta$, the bridge will execute a role switching operation and determine whether or not the packet error rate drops. However, less number of piconets will reduce simultaneous transmission opportunity among piconets. The performance improvement must be monitored to guarantee that the role switching operation effectively promotes the suppression of packet collisions and does not eliminate opportunities for simultaneous transmission of packets. An adaptive value α is also set in the developed protocol for adjusting the weight of packet error rate and traffic load in decision function $g_t(\gamma, o)$. Let

$$g(\gamma, o) = \alpha \times \gamma + (1-\alpha) \times o \qquad (2)$$

A higher value of α implies that the decision of role switching more depends on packet error rate. On the contrary, α with a lower value implies that the role switching operation concerns more about traffic load than packet error rate.

Figure 7 shows the block diagram of *ARSP*. All bridge devices monitor the performance of the scatternet. A function g is used to make decision whether or not to initiate the role switching operation. The value of function g is controlled by the monitoring parameters γ and o. When function g exceeds a threshold, the bridge device initiates a role switching request to its master. Upon receiving the role switching request, the master device transmits to the bridge device the information required for role switching, including the number of slaves and the traffic information. In response to the received information, the bridge device maintains a decision table and initiates appropriate role switching operations accordingly. The details of each block and the relationships between blocks will be illustrated later.

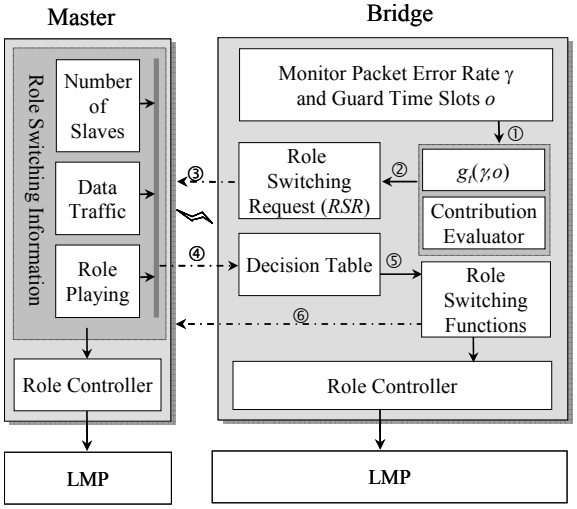


Fig. 7: The negotiation of Master and Slave in *ARSP*.

All devices that intend to execute the role switching operation must follow the following rules to guarantee that the adaptive role switching operation can effectively reduce the number of piconets, guard slots and the length of the routing path.

***Rule*-1**: *Only a bridge device can initiate role switching.* ■

As stated above, a role switching operation initiated by a master or slave cannot reduce the number of piconets or bridges. Rule 1 allows only the bridge device to execute role switching to reduce the packet error rate and guard time overhead. This rule also prevents too many devices from initiating the role switching operation, reducing the complexity of lock operation.

***Rule*-2:** *When* $g_t(\gamma, o) > \delta$, *the role switching operation will be triggered.* ■

One main reason for the high packet error rate is an excessive number of piconets in the scatternet, causing the hopping sequences to collide with each other. Another major reason is that the master polls a bridge that is active in another piconet, causing packets to be lost. The role switching operation can effectively reduce the number of piconets and bridges, helping to suppress the packet error or packet loss. Hence, when the decision function $g_t(\gamma, o)$ maintained by bridge *d* exceeds δ, the role switching operation will be

triggered automatically by device *d*. Details will be presented below.

***Rule*-3:** *TakeOver is the main function of role switching operation.* ■

As discussed in Section 2, the *TakeOver* operation initiated by a bridge device can effectively reduce the number of piconets and bridges and obtain the full advantages. Combining Rules 1, 2 and 3, only bridges are required to monitor the value of $g_t(\gamma, o)$ and maintain the decision table. As the value of $g_t(\gamma, o)$ larger than a threshold δ, the bridge device will select some masters from the decision table and initiate the *TakeOver* operation to the selected masters.

Notably, the selected threshold δ and value α are keys to improving the performance of the system. A large δ will block the role switching function. A large value of α will cause the execution of role switching operation highly depends on the traffic error rate. The impacts of δ and α will be discussed in the performance study.

Combining Rules 1 and 3 yields the following desired property.

Desired Property: Given a bridge device *d*, $d \in D$, *Role(d)*=*S*/*S* or *M*/*S*, a proper role switching operation has the following desired property:

Min(*NOP*(*d*)) and Max(*NOS*(*d*)) ■

When a bridge device intends to initiate a role switching request to the masters to which it is connected, it will establish a *Decision Table* to help bridge devices determine the order of the masters to which to apply role switching operation. The Decision Table is defined as follows.

***Definition*:** *Decision Table DT*(*d*) stored in bridge *d*

When a bridge *d* tries to execute a role switching operation, a decision table *DT*(*d*) is established by *d* to collect information from all the connected masters about the number of slaves (noted by *NOS* in short) and the traffic. In a *DT*, each row records information about one master to which the bridge is connected. *DT* includes the *NOIS* and the *RouteContribution* fields. The *NOIS* field records the *Number Of Increased Slaves* if the role switching operation is applied to the master, whereas the *RouteContribution* field evaluates the route contribution if the role switching operation is applied to the master.

■

A bridge device *d* will initiate a role switching request to a master device *a,* only if the following three criteria are met.

Criteria for Role Switching Operation:

***Criteria* 1:** device *d* detects that the condition $g_t(\gamma, o)>\delta$ is satisfied.

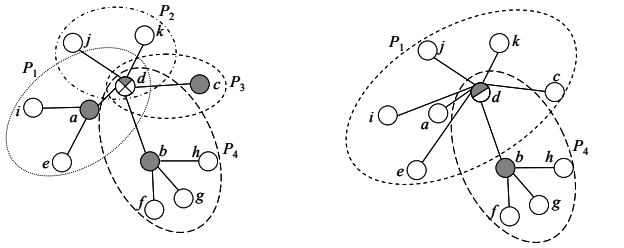
***Criteria* 2:** *RouteContribution*(*d*, *a*) is positive.

***Criteria* 3:** The total number of slaves in the new piconet constructed by role switching should be less than or equal to seven.

Noted that only bridge device can initiate the role switching operation to achieve the goal of Max(*NOS*(*d*)). If the neighbor of device *d* is a bridge, the role switching operation can be extended to 2-hop to eliminate more piconets and bridges. Therefore, in the following, two cases are discussed according to whether or not the device to which the bridge device connects plays a bridge role. The *Nonbridge Neighbor Case* refers to the condition under which no neighbors of device *d* is a bridge while the *Bridge Neighbor Case* refers to the complicated conditions under which at least one neighbor of device *d* is a bridge. In the latter case, two bridges may initiate the role switching request simultaneously, because that both their decision functions $g_t(\gamma, o)$ have a value that exceeds than the threshold δ. Then, two devices will exchange information and determine the order of executing role switching operation to maximize the route contribution.

(1) *NonBridge Neighbor Case*: No neighbor of bridge device *d* is a bridge

In this case, bridge *d* initiates a *TakeOver* request to its connected masters so that all slaves in the neighboring piconets can be combined in a single piconet where bridge *d* plays the master role. Bridge *d* sends a *Role switching Request* (*RSR*) message to each of its master. Upon receiving the *RSR* message, each master sends the *Role switching Information* (*RSI*) message back to *d*. The *RSI* message includes the number of slaves of the master and the traffic load of the master. Bridge *d* will store the received *RSI* message in the *decision table*. In some cases, a bridge device obtains positive route contributions by executing the role switching operation on at least two masters. According to the following formula, bridge *d* can evaluate the total number of slaves in the combined piconet by $\sum NOIS(m) + NOS(d)$, for all masters *m* connected to the bridge *d*. Bridge *d* then initiates a role switching operation *TakeOver*(*d, m*) to each master *m* until the number of slaves combined in the piconet reaches to seven, which is the maximal number of slave a piconet can contain.



(a) A scatternet structure before executing the *TakeOver* operation.

(b) A scatternet structure after executing the *TakeOver* operation.

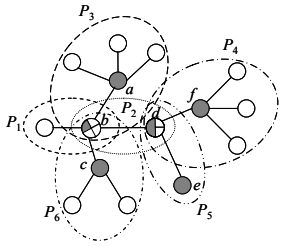
Fig. 8: An example for illustrating Condition *B*.

The example shown in Fig. 8(a) illustrates this case. When bridge device *d* detects that the condition $g_t(\gamma, o)>\delta$ is satisfied and it evaluates the route contributions of executing role switching with masters *a*, *b*, and *c* are positive, device *d* will send a role switching request to masters *a*, *b* and *c*. Upon receiving the request, masters *a*, *b* and *c* reply with information on their *NOS* numbers and traffic loads to device *d*. Device *d* will store this information in its *decision table*, as shown in Table 1, in which *NOIS*(*c*)=1 represents that the

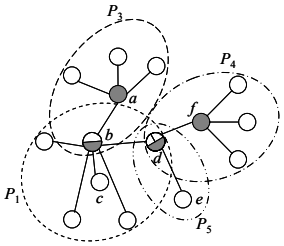
number of slaves is increased by one if device *d* initiates a role switching operation *TakeOver*(*d*, *c*) to device *c*. In such a case, if device *d* applies the role switching operations *TakeOver*(*d*, *a*), *TakeOver*(*d*, *b*) and *TakeOver*(*d, c*) to masters *a*, *b*, and *c*, respectively, then the total number of slaves in the newly combined piconet will violate the condition $NOS(d) \leq 7$. Device *d* will apply the role switching operation to these masters in ascending order of *NOIS* value recorded in the *DT*. Device *d* will initiate role switching operations with those masters until the total number of slaves reaches the upper limit to reduce the number of piconets as much as possible. Hereafter, device *d* will initiate role switching operations *TakeOver*(*d*, *c*) and *TakeOver*(*d*, *a*) with masters *c* and *a*, respectively. Figure 8 (b) shows the resulting structure of the scatternet. The number of piconets has been reduced from four to two.

Table 1: Decision Table of device *d*

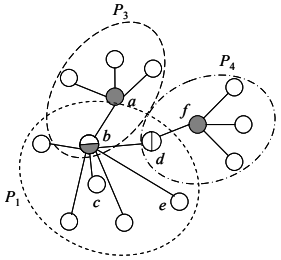
	NOIS	*RouteContribution*
c	1	*RouteContribution* (*d*, *c*)
a	3	*RouteContribution* (*d*, *a*)
b	4	*RouteContribution* (*d*, *b*)



(a) A scatternet structure of bridge-bridge connection.



(b) Scatternet structure after executing step (5).



(c) Scatternet structure after executing step (6).

Fig. 9: An example for illustrating case of *Bridge-Bridge* connection.

Table 2: Decision Table of bridge device *b*

	NOIS	*RouteContribution*
e(*d*)	2	*RouteContribution* (*b*, *e*)
c	3	*RouteContribution* (*b*, *c*)
a	4	*RouteContribution* (*b*, *a*)
f(*d*)	5	*RouteContribution* (*b*, *d*)

(2) *Bridge Neighbor Case*: Bridge device *d* has a neighbor that is a bridge

This case involves role switching between two bridges. In the following, an example of an *M*/*S* bridge connected to an *M*/*S* bridge will be considered to illustrate the adaptive role switching protocol. Consider the scatternet in Fig. 9(a). It includes two bridge devices *b* and *d* that can initiate role switching. Once bridges *b* and *d* have detected that the condition $g_t(\gamma, o) > \delta$ holds, they evaluate the route contribution and initiate the role switching operation. In this example, device *b* is assumed firstly to detect that the condition $g_t(\gamma, o) > \delta$ holds. In the following, the adaptive role switching protocol will be applied stepwise using this example.

(1) When device *b* detects the condition $g_t(\gamma, o) > \delta$ is satisfied, it initiates a role switching request to masters *a*, *c* and *d.*

(2) On receiving the role switching request, masters *a* and *c* will reply with their information about their numbers of slaves (*NOS*(*a*) and *NOS*(*c*)) and traffic loads. The bridge device *d*, will enter the lock state to prevent two bridges from executing the role switching operations simultaneously.

(3) After entering the lock state, device *d* will collect information from the masters *e* and *f* to which it is connected, and then report to device *b* the information about *NOS*(*e*), *NOS*(*f*) and the traffic loads of *e* and *f*.

(4) According to the *NOS* and traffic information, device *b* will calculate the *NOIS* value and *RouteContribution* value for each master. Then, bridge *b* will sort the masters in order of *NOIS*. Thereafter, device *b* will store the *NOIS* and *RouteContribution* values in a decision table, as presented in Table 2. Under the constraint that the number of slaves in the constructed piconet should be less than or equal to seven, device *b* determines to initiate the role switching operation with devices *e* and *c*, so that the number of piconets can be reduced as much as possible.

(5) Device *b* executes role switching with device *c* by initiating the *TakeOver*(*b*, *c*) operation. However, since device *e* is a neighbor of device *d*, device *b* requests that device *d* initiates the role switching operation, *TakeOver*(*d, e*) to device *e*. Figure 9(b) shows the resulting structure of scatternet.

(6) Finally, device *b* initiates the role switching operation *TakeOver*(*b*, *d*) to take over all device *d*'s slaves. Figure 9(c) shows the resulting scatternet structure.

Adaptive Role Switching Protocol *ARSP*(*d*)
1: Each bridge device monitors γ and *o* parameters
2: If $g_t(\gamma, o) >$ threshold δ
3: For each master *a* to which *d* connected
4: Send *RSR* to *a*
5: If *Role*(*a*) = bridge
6: *ARSP*(*a*)
7: Endif
8: Receive *RSI*(*a*.role, *a*.traffic, *a.NOS*) from *a*
9: *RouteContribution*(*d*, *a*) = $\sum_{i=1}^{k} T_i \Delta_i$
10: If *RouteContribution*(*d*, *a*) > 0
11: Calculate *NOIS*(*a*) according to *NOS*(*a*)
12: Insert (*NOIS*(*a*), *RouteContribution*(*d*, *a*)) into *DT*(*d*)
13: Endif
14: Endfor
15: Sort the *DT*(*d*) in a nondescending order by *NOIS* value
16: While the total number of slaves ≦ 7
17: Select one device from *DT*(*d*)
18: *TakeOver*(*d*, *b*)
19: Endwhile
20: Endif

Fig. 10: The algorithm for Adaptive Role Switching Protocol (*ARSP*).

Figure 10 depict the algorithm of the proposed *ARSP*. In the proposed *ARSP*, steps 1 and 2 depicts that every bridge device d monitors $g_t(\gamma, o)$ and then further detects whether or not the condition $g_t(\gamma, o) > \delta$ is satisfied. If this is the case, bridge d will send a role switching request *RSR* to the masters which connected to the bridge d as shown in steps 3 and 4. On receiving the *RSR* packet, each master a checks whether or not it plays a bridge role. If this is the case, the adaptive role switching algorithm will be recalled as shown in steps 5 to 7. Otherwise, master a replies to d the *RSI* packet that contains the information including its role, traffic, and *NOS* value, as represented in step 8. After that, as shown in step 9, device d calculates the value of *RouteContribution* according to each collected *RSI* packet if the *TakeOver*(d, a) is executed. In case that the route contribution is positive, the bridge d further calculates the *NOIS* value according to the *NOS* value and inserts these information into the *Decision Table DT*(d) as shown in steps 10 to 13. Finally, bridge d sorts the *Decision Table DT*(d) in an nondescending order by *NOIS* value and repeatedly selects one device from *DT*(d) to execute *TakeOver* operation if the number of slaves is less than eight. Steps 15 to 19 depict the abovementioned operations.

V. Performance Study

This section examines the performance of the *Adaptive Role Switching Protocol* (*ARSP*), in terms of the guard time, throughput, route length, and packet error rate. The value of thresholds δ and α in decision function are also investigated. The simulation environment is as follows. The size of the scatternet region is 7×7 units, and the radio transmission range of a Bluetooth device is set to a constant 10 units. Initially, a connected scatternet, namely '*Original*', is randomly constructed. That is, each device alternatively executes an inquiry/page or an inquiry scan/page scan operations to connect with other devices at random. According to the packet error rate and guard time cost, the proposed protocol, namely '*ARSP*', dynamically applies the role switching operations to change the scatternet topology. The number of devices ranges from 10 to 120 and the packet arrival rate is set to 100kbps. Parameters such as the number of devices, the number of piconets, and the average number of bridges are varied to elucidate the performance of *ARSP*.

Figures 11 to 13 compare the *ARSP* with *Original*. The thresholds using in the decision function $g(\gamma, o)$ are δ=50% and α =50%. Packet transmission from one piconet to another relies on a bridge device. The use of the appropriate number of bridges can maintain the scatternet connectivity. However, a bridge device switching among the participated piconets will raise the guard time cost. The bridge switching among numerous piconets also increases the difficulty of scheduling among numerous piconets and hence, increases the probability of packet loss. In Fig. 11, the *BD-Original* and *BD-ARSP* denote the average bridge degrees obtained by applying *Original* and *ARSP* mechanisms. Applying *ARSP* significantly reduces the average bridge degree but maintain the connectivity of scatternet. Furthermore, the *GT-Original* and *GT-ARSP* in Fig. 11 denote the guard time costs of *Original* and *ARSP*, respectively. In general, the guard time cost of *GT-Original* increases with the number of devices but *GT-ARSP* maintains a constant guard time cost and a constant bridge degree. The reduction in bridge degree can largely reduce the guard time cost. As a result, the *GT-ARSP* improves 5% of guard time cost in average and obtains a better performance than *GT-Original*.

In Fig. 12, the *NP-Original* and *NP-ARSP* denote the numbers of piconets obtained by applying *Original* and *ARSP* mechanisms, respectively. In general, *NP-Original* and *NP-ARSP* increase with the number of devices when the number of devices smaller than 90. This indicates that both mechanisms exploit the opportunities of simultaneous transmissions. However, *ARSP* maintains a constant number of piconets when the number of devices exceeds 90 and hence prevents the packet collisions due to simultaneous transmissions from too many piconets. In Fig. 12, the *TH-Original* and *TH-ARSP* denote the throughputs of *Original* and *ARSP*, respectively. Recall that the channel used by each piconet depends on the hopping sequence. A large number of piconets will cause packet collisions due to the overlap of hopping sequences. As the number of devices increases, the *ARSP* constrains the growth in the number of piconets and maintains a constant number of piconets to explore the opportunities of simultaneous transmissions but prevent the packet transmissions from collision. Consequently, the throughput of *ARSP* outperforms that of the *Original* mechanism. The *ARSP* obtains the best performance when the number of devices is 80 and the number of piconets is 16.

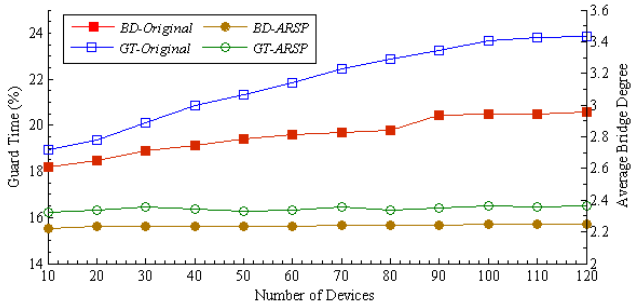


Fig. 11: *ARSP* reduces the average degree of bridge and hence, saves the average guard time cost.

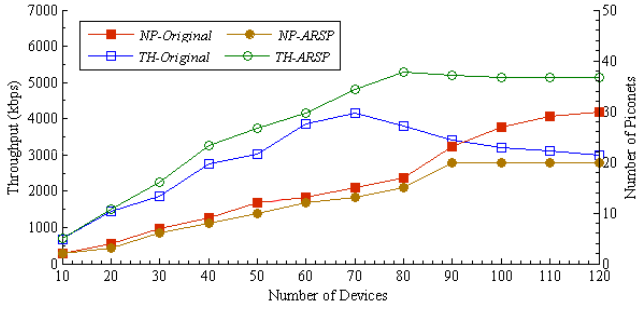


Fig. 12: *ARSP* maintains a constant number of piconets when the number of devices exceeds 90, and hence eliminates the collisions and improves the throughput.

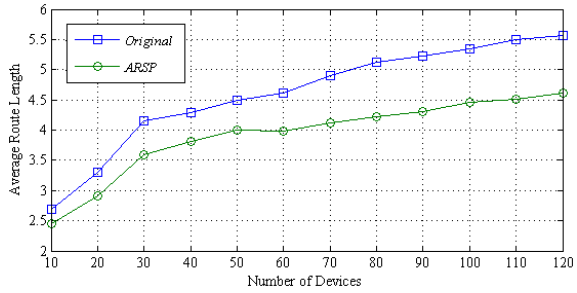


Fig. 13: The average route length of *ARSP* grows slower than *Original* when the number of devices increases.

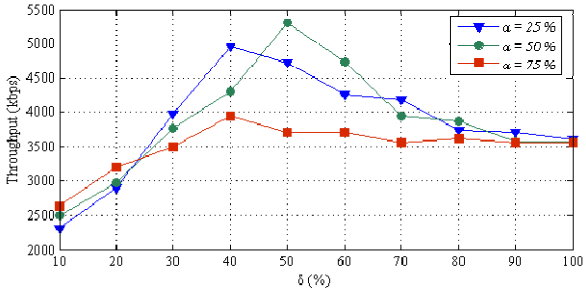


Fig. 16: The comparison of *ARSP* with different α in throughput when δ is varied from 10% to 100%.

The end-to-end transmission delay highly depends on the route length and the guard time associated with the bridge switching among the participated piconets. Figure 13 measures the average route lengths obtained by applying *Original* and *ARSP* mechanisms. The proposed *ARSP* significantly reduces the route length when the number of devices is increased.

Recall that the decision function of the proposed *ARSP* uses thresholds δ and α to determinate whether or not the bridge nodes can perform the role switching operation. Figure 14 compares the performance of the *ARSP* with different thresholds δ while the value of threshold α in decision function is varied from 0% to 100%. The network size is set at 100 devices. A larger value of threshold α implies that role switching more depends on the packet error rate and hence, the execution of role switching operation is more sensitive to the increment in the number of piconets. The role switching operation is executed more frequently as the value of α increases. As a result, the number of piconets decreases as the value of α increases.

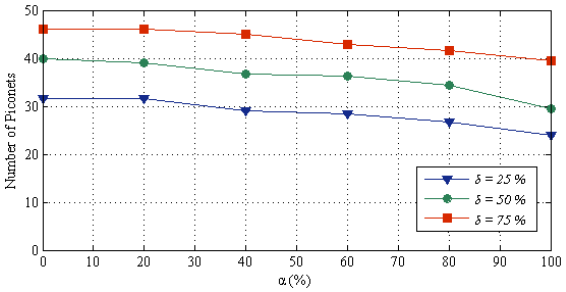


Fig. 14: The number of piconets associated with different δ varying from 0% to 100%.

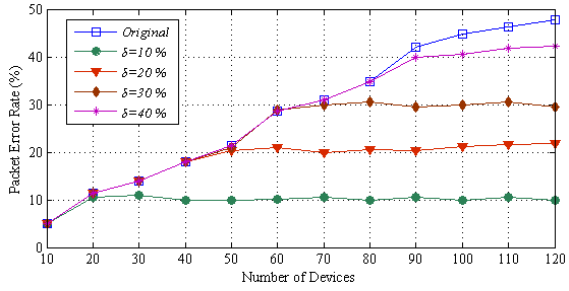


Fig. 15: The effect of δ on packet error rate with α = 50%.

Figures 15 compare the packet error rate of *ARSP* when α = 50%. The number of devices is varied ranging from 10 to 120. In case that the threshold α is 50%, the decision for role switching operation will take into account both the packet error rate and the traffic load. Consequently, the packet error rate increases with the number of devices and is not bounded by the value of δ, as shown in Fig. 15.

Figure 16 shows the effect of threshold values δ and α on throughput. The *ARSP* achieves maximal throughput for α=25%, α=50% and α=75% when δ are 40%, 50%, and 40%, respectively. The maximum throughput is obtained when the value of δ is between 40% and 50%. The decision function works appropriately when the value of δ falls in this range and hence, reduces the number of piconets and eliminates the packet collisions as well as exploits the opportunities for simultaneous transmission. Generally, applying *ARSP* locally changes the topology of the scatternet by reducing the number of piconets, the number of bridges and the route length. Therefore the scatternet performance is significantly improved.

VI. Conclusions

The improper structure of a scatternet will not only cause packet error and guard time overheads but also raise the problem of transmission delay. This work presents an Adaptive Role Switching Protocol (*ARSP*) for reorganizing the structure of an improper scatternet. Three types of role switching operations and their effects on scatternet performance are analyzed. The adaptive role switching protocol uses role switching operations to remove the unnecessary piconets and bridges and shorten the route in a distributed and dynamic manner. By monitoring the packet error rate, the guard time cost, and the traffic, the bridge device can initiate a role switching request to masters in the best order to maximize the route contribution obtained by changing the structure of the scatternet. By maintaining appropriate numbers of piconets and bridges and the role of each device, the proposed protocol reduces the overheads increased by packet loss and guard time cost, as well as the route length. Experimental results reveal that *ARSP* significantly improves the performance of scatternet.

REFERENCES

[1] The Bluetooth Specification, http://www.bluetooth.org/

[2] C. Law, A. K. Mehta, and K.Y. Siu, "A New Bluetooth Scatternet Formation Protocol," *ACM Mobile Networks and Applications Journal*, vol. 8, no. 5, pp. 485-498, 2002.

[3] T. Salonidis, P. Bhagwat, and L. Tassiulas, "Distributed Topology Construction of Bluetooth Personal Area Networks," *Proceedings of the 20rd annual Joint Conference of the IEEE Computer and Communication Societies* (INFORMCOM 2001), pp. 1577-1586, 2001.

[4] G. V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees-Scatternet Formation to Enable Bluetooth Based Ad Hoc Networks," *Proceedings of IEEE International Conference on Communications*, pp. 273-277, 2001.

[5] T. Y. Lin, Y. K. Liu, and Y. C. Tseng, "An Improved Packet Collision Analysis for Multi-Bluetooth Piconets Considering Frequency-Hopping Guard Time Effect," *IEEE Journal on Selected Areas in Communications,* vol. 22, no. 10, pp. 2087-2094, 2004.

[6] S. Zilrbes, W. Stahl, K. Matheus,and J. Haartsen "Radio Network Performance of Bluetooth," *IEEE International Conference on Communications* (ICC 2000), pp. 1563-1567, 2000.

[7] W. Feng, A. Nallanathan, and H. K. Garg, "Performance of PHY and MAC Layers of A Bluetooth Piconet in Multi-Bluetooth Interference Environment," *IEEE Global Telecommunications Conference* (GLOBECOM 2004), vol. 6, pp. 3614-3618, 2004.

[8] S. Zurbes, "Considerations on Link and System Throughput of Bluetooth Networks," *Proceedings of the 11th IEEE International Symposium on Personal, Indoor and Mobile Radio Communication*, pp. 1315-1319, 2000.

[9] Pravin Bhagwat, Adrian Segall, "A Routing Vector Method (RVM) for Routing in Bluetooth Scatternets," *The Sixth IEEE International Workshop on Mobile Multimedia Communications (MOMUC'99)*, pp. 375-379, 1999.

Chih-Yung Chang received the Ph.D. degree in Computer Science and Information Engineering from National Central University, Taiwan, in 1995. He joined the faculty of the Department of Computer and Information Science at Aletheia University, Taiwan, as an Assistant Professor in 1997. He was the Chair of the Department of Computer and Information Science, Aletheia University, from August 2000 to July 2002. He is currently an Associate Professor of Department of Computer Science and Information Engineering at Tamkang University, Taiwan. Dr Chang served as an Associate Guest Editor of *Journal of Internet Technology* (JIT, 2004), *Journal of Mobile Multimedia*(JMM, 2005), and a member of Editorial Board of *Tamsui Oxford Journal of Mathematical Sciences* (2001-2005). He was an Area Chair of *IEEE AINA*'2005, Vice Chair of *IEEE WisCom*'2005 and *EUC*'2005, Track Chair (Learning Technology in Education Track) of *IEEE ITRE*'2005, Program Co-Chair of *MNSAT*'2005 and *UbiLearn*' 2006, Workshop Co-Chair of *INA*'2005, *MSEAT*'2003, *MSEAT*'2004, and Publication Chair of *MSEAT*'2005 and *SCORM*'2006. Dr. Chang is a member of the IEEE Computer Society, Communication Society and IEICE society. His current research interests include wireless sensor networks, mobile learning, Bluetooth radio networks, Ad Hoc wireless networks, and mobile computing.

Hsu-Ruey Chang received the B.S. degree in Computer Science and Information Engineering from Tamkang University, Taiwan, in 2003. Since 2004, he was working toward his PhD degree and currently he is a PhD candidate in Department of Computer Science and Information Engineering at Tamkang University. Mr. Chang is a student member of the IEEE Computer Society, Communication Society and IEICE society. He has won lots of scholarships in Taiwan and participated in many Bluetooth and Wireless Sensor Networking projects. He has published extensively in the wireless networking area and receives the 2005 Workshop on Wireless, Ad Hoc, and Sensor Networks (WASN) Best Paper Award. His research interests are wireless sensor networks, Ad-Hoc wireless networks, and mobile/wireless computing, concerning both theoretic results and algorithms design.