韓國 컴퓨터情報學會 論文誌
第12卷 第6號, 2007. 12.

2007-12-6-2-11

블루투스 네트워크에서 스캐터넷 재구성을 위한 Fast-Recovery 알고리즘

장 범*, 구 명 모**, 김 상 복***

A Fast-Recovery Algorithm for Scatternet Reformation in Bluetooth Networks

Fan Zhang*, Myeong-Mo Gu**, Sang-Bok Kim***

요 약

스캐터넷(Scatternet)에서 피코넷(Piconet)간의 연결을 통하여 블루투스 애드 혹 네트워크를 형성한다. 이러한 스캐터넷을 형성하기 위한 많은 알고리즘이 제안되었다. 하지만 이들은 스캐터넷 재구성 시간이 많이 걸리는 문제가 있다. 본 논문에서는 마스터가 피코넷을 벗어났을 경우 스캐터넷 재구성시 지연시간을 줄이기 위한 Fast-Recovery 알고리즘을 제안한다. 제안한 알고리즘에서는 마스터에 의해 생성된 피코넷 장치 테이블에서 마스터와 비슷한 가중치를 가지는 슬레이브를 Sub-Master로 선택한다. 마스터가 피코넷을 벗어났을 때, Sub-Master가 이전의 마스터 대신에 새로운 피코넷의 마스터가 되고, 페이지 상태에서 피코넷 장치 테이블을 통하여 모든 슬레이브와 브릿지에 직접 연결하도록 한다.

Abstract

A Bluetooth ad hoc network can be formed by interconnecting piconets into scatternet. Many algorithms of scatternet formation have been proposed so far. However, these have a problem that takes a long time to reform scatternet, especially for the case of master moving out. In this paper, we propose a Fast-Recovery algorithm, which aims at reducing the time of scatternet reformation owing to master moving out. In the algorithm, we select a slave with the weigh similar to its master as Sub-Master from a piconet device table created by the master. When a master leaves its piconet is detected by its slaves, the Sub-Master becomes the new master of the piconet instead of the old and directly connect to other slaves and bridge(s) through piconet device table in page state.

▸ Keyword : Scatternet, Piconet, Sub-Master, Reformation, Ad Hoc

• 제1저자 : 장범 • 교신저자 : 김상복
• 접수일 : 2007.12.10, 심사일 : 2007.12.15, 심사완료일 : 2007.12.20.
* 경상대학교 컴퓨터과학과 석사과정, ** 진주산업대학교 컴퓨터공학부 겸임교수,
*** 경상대학교 컴퓨터과학과 교수, 경상대학교 컴퓨터정보통신연구소 연구원, 경상대학교 전산정보원장

Ⅰ. Introduction

Bluetooth (BT) wireless technology is a short-range communication system intended to replace the cables connecting portable or fixed electronic devices such as cell phones, PDAs, laptops, MP3 players, etc. It operates in the 2.4GHz globally available and unlicensed Industrial Scientific Medical(ISM) band, and adopts Frequency-Hopping (FH) scheme to minimize the interferences from other users in the same band. The key features of Bluetooth wireless technology are robustness, low power, and low cost (down to 5 USD/ Bluetooth chip)[1].

According to BT specification, when two BT nodes that come into each other's communication range want to set up a communication link, one of them must assume the role of master of the communication while the other becomes its slave. This simple 'one hop' is called a piconet [2,3]. Each piconet has only one master and up to 7 active slaves. Piconets can also be interconnected via bridge nodes to form a bigger ad hoc network known as a scatternet.

So far, Bluetooth is still an open global specification that has not described in detail the problem of scatternet formation. Therefore, many algorithms of scatternet formation and reformation have been proposed in the literatures [4-10]. However, Due to mobility of Bluetooth devices or limited power supply, the topology of scatternet may be changed occasionally. If the moving device is a slave, the change of topology only occurs in one or two piconet. Whereas, if the role of moving device is either master or bridge, not only piconet topology but also scatternet topology would change. Although in the proposed papers a few may deal with it, only few algorithms cope with master moving out very well. Law et al [8] considered only the static membership. Chang et al [11] executed of scatternet reformation, which may cause unnecessary time consumption and communication cost of scatternet reformation. Sheng et al. [12] randomly selected a slave as the new master from original piconet in optimization of algorithm, which leads to worse performance owing to not consider the suitability of playing the role of master.

Also In the paper, we propose a Fast-Recovery algorithm, which focuses on reducing the time of scatternet reformation caused by the master moving out. In our algorithm, a piconet device table is created by each master of piconets in the procedure of Sub-Master selection, which stores the informations about identity, we ight and role of devices in the piconet. After scatternet formation procedure, every master periodically sends it to all its slaves or bridge(s).

A slave with the weight similar to its master is selected from the piconet device table as Sub-Master. When a master leaves its piconet is detected by its slaves or bridge(s), the slave that acts as Sub-Master goes into page state. On the other hand, others in the old piconet enter page scan state to wait for paging. In additional, we also have considered how to solve the problem that some slave is lost from the recovered piconet.

The rest of the paper is organized as follows. Section 2 discusses the related work on Bluetooth scatternet. Section 3 describes and analyses the Fast-Recovery algorithm that we put forward in great detail. Section 4 evaluates the performance of the algorithm via simulation. Finally, section 5 concludes the paper and discusses further work directions.

Ⅱ. Related Work

2.1 RDSF

Law et al. [8] presented and analyzed a randomized distributed protocol for scatternet formation. The main method is to merge pairs of connected components until one component is left (Fig. 1). In each round, a leader either tries to contact another component or waits to be contacted.

The scatternet formed by this protocol has the

following properties (1) any device is a member of at most two piconets. (2) the number of piconets is close to be optimal. These properties can prevent overloading of any free nodes and reduce the interference between piconets. However, the two protocols only assume static memberships which do not arrive or leave the existing scatternet.

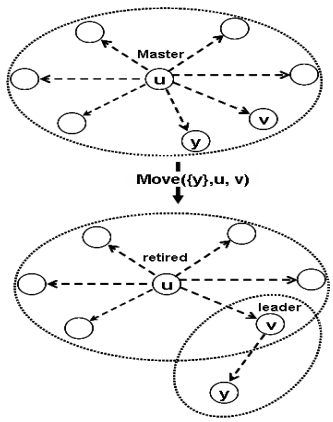


그림 1. 연결된 컨포넌트 병합
Fig. 1. Connected components merging

2.2 Long-lived Scatternet Formation Protocol

Chang et al.[10] proposed a new scatternet formation protocol to construct a long-lived scatternet(Fig. 2).

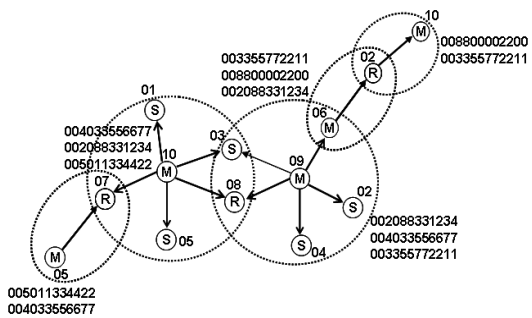


그림 2. 스캐터넷 구성
Fig. 2. Scatternet formation

It consists of three algorithms: device discovery, scatternet construction and scatternet recovery algorithm. In device discovery algorithm, a symmetric mechanism is adopted, which is that each device does inquiry scan first then inquiry. In scatternet construction algorithm, some backup bridges are designated by the masters with the highest weight. The scatternet recovery algorithm is designed to deal with the change of scatternet topology due to the moving of devices and exhaustion of power. There are three cases considered in this section:

(1) a slave moves out of communication range of its master. When a slave has not received the polling packet from its master for a polling time, they assume that it moves out of its piconet.

(2) a bridge moves out of the communicaion range of its master. When a master fails to communicate with the master of neighboring piconet, they assume that it moves out of its piconet.

(3) a master moves out of communication range of its slaves. When a master polls its slave without a respond, they assume that the slave leaves its poconet.

However, in the case of 3, the proposed scatternet recovery algorithm only resets nodes and executes scatternet formation phase, which leads to unnecessary time consumption on reforming scatternet.

III. Fast-Recovery Algorithm

In this section, we describe Scatterenet Fast-Recovery algorithm proposed in this paper. The main aim of our algorithm is to reduce the time of scatternet reformation owing to the master moving out. It mainly consists of four procedures: device discovery, scatternet formation, Sub-Master selection and scatternet re-formation. We adopt the symmetric mechanism similar to introduced in [10] for device discovery procedure. A weight is assigned to each node, which represents the suitability for playing the role of master or bridge.

3.1 Device Discovery

In the device discovery, each device is allowed to independently alternate between inquiry and inquiry scan state, remaining in each state for a time

selected randomly and uniformly in a predefined time. The special operation is performed as follows: each device does inquiry scan state first, and waits Tinq(16 time slot). If a node does not hear any IAC packets from others for Tinq, then it switches to inquiry state, sends IAC packets and waits for 1024 time slot to collect responses from its neighboring devices by receiving FHS packets. As mentioned in [4], to allow each pair of neighboring devices to achieve a mutual knowledge of each other ID and weight, the scheme requires that, whenever a device in inquiry (inquiry scan) state receives(sends) an FHS packet, a temporary piconet is set up by means of a page phase and devices exchange their ID and weight together with the synchronization informations required for furtther communication.

As soon as this information has been succes-sfully communicated, the piconet is disrupted. In addition, a random back-off mechanism is used to prevent devices from entering the same state at the same time.

3.2 Scatternet Formation

Scatternet formation is composed of two phases: piconet formation and piconet interconnection.

In the phase of piconet formation, the role (master or slave) of each device depends on the value of weight which is gathered in the device discovery. That is, when the weight of a device is greater than weight of any its nearby devices, it goes into page state as master to connect to its neighboring devices. Otherwise, the device enters page scan state as slave to wait for paging. If a node has been in page scan state for Tp without being paged into a piconet, then the device switches the role to master and enters page state to try to page its neighboring devices with different PID (Piconet Identity). Each piconet in our scheme is assigned with a unique PID, which is an integer and used to identify devices in the same group. The PID is used to avoid constructing too many links.

After forming the piconets successfully, a proper bridge is designated in the view of weight. If a slave has been paged by more than two masters, the slave informs the master with the highest weight to select a bridge. The the master designates the slave with the highest weight as bridge. As a result, a scatternet is constructed by the piconets that interconnect via bridges.

3.3 Sub-Master Selection

To solve the case of the master moving out, a simple approach proposed in [9] is to select a slave randomly in the piconet as the new master in the scatternet reformation. It has the lowest cost, but worse performance due to not consider suitability of playing the role of master well.

표 1. 피코넷 장치 테이블
Table 1. Piconet device table

ID	Role	Weight
1	M	15
2	SUB	13
3	S	12
4	B	11
5	S	10

In the paper, to overcome the drawback of the simple approach, we present an algorithm of Sub-Master selection. The main method is that following scatternet formation, each master in the scatternet creates a piconet device table, which stores the informations about identity, weight and role of devices in its piconet (master self included). The table is made with the device weight by decreasing order. Therefore, each device gets a pseudo sequence number. A slave with the similar weight to its master is selected as Sub-Master from the piconet device table. In other words, a slave which has greater weight than any other slaves in the piconet device table is used as Sub-Master. Therefore, the role of this slave is set to be 'SUB' as shown in Table1. Then master periodically transmits the piconet device table to all of its slaves and bridge(s).

The approach not only saves time but also

reduces communication cost for reconstructing a new master. Figure 3 shows flow of Sub-Master selection.

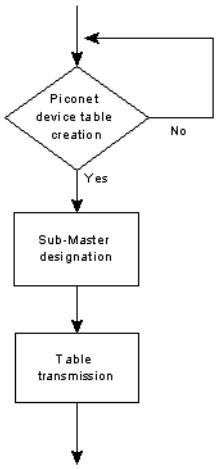


그림 3. Sub-Master 선택 흐름도
Fig. 3. Flow of Sub-Master selection

3.4 Scatternet Reformation

Due to mobility of devices or limited power, the topology of scatternet may be changed on occasion. If the moving device is a slave, the topology change only occurs in the two piconets. However, if the role of moving device is either master or bridge in the scatternet, not only the piconet topology but also the scatternet topology would change. Thus, a Fast-Recovery mechanism is essential for scatternet reformation.

Figure. 4 shows that the flow of scatternet reformation. The first step is device movement detection, which is responsible for detecting the master that loses connections to its piconet. According to BT Specification, a master periodically polls its slaves in the piconet. Let period of the master polling any device in the piconet be Tpoll, if a slave or bridge has not received any polling packets from its master for Tidle, the slave or bridge knows that its master has moved or powered off. Note that the value of Tidle would set be to satisfy Tidle 〉 nTpoll. The average time required to detect the master moving out relies on the above parameters. Moreover, the value of parameters can be adjusted for different device mobility to detect the moving of device quickly.

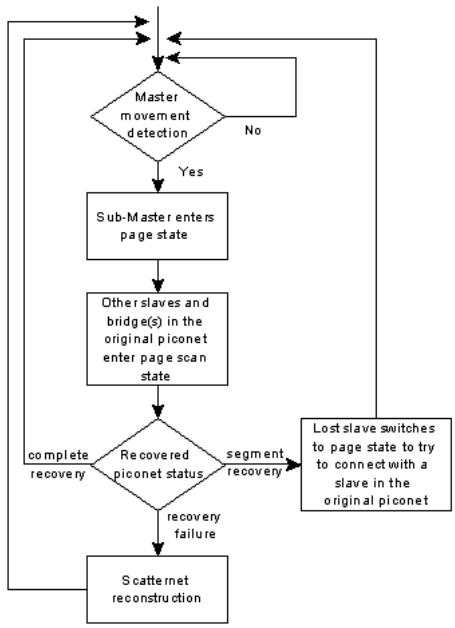


그림 4. 스캐터넷 재구성 흐름도
Fig. 4. Flow of scatternet reformation

If a master is detected by its slaves and bridge(s), according to the stored piconet device table A slave with the role of Sub-Master designated in the previous phase of Sub-Master selection goes into page state. On the other hand, other slaves and bridge(s) of the original piconet enter page scan state to wait for paging. Therefore, the Sub-Master acts as the new master instead of the old to connect with other slaves and bridge(s) of the original piconet. Then after the page process the piconet is recovered.

However, we notice that the communication range of the new master may not cover the same communication range as the old covered. Let the communication range of the original piconet be Ar. △Ar is set to be the communication range that the new does not cover (Fig. 5(a)). Therefore, we also consider about two cases on the status of recovered piconet and adopt different approaches to deal with them respectively.

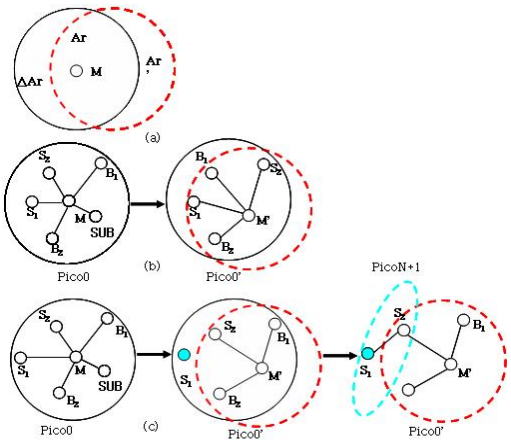


그림 5. 피코넷 재구성
Fig. 5. Piconet reformation

(1) Piconet Complete recovery. If all of the slaves and bridge(s) of the old master are in communication range of the new, then the new in page state may connect with them completely. In other words, the recovered piconet consists of all of the slaves and bridge(s) of the old. Then a new Sub-Master is designated by the new master according to its updated piconet device table. As figure 5(b) shows, Sub-Master M' became the new master, which connected with S1, S2, B1, B2 completely.

(2) Piconet Segment recovery. If some slave in page state is not in communication range of the new master, then Sub-Master could not connect with it. In other words, the original piconet is partitioned due to the lost slave. Sub-Master updates the original piconet device table. Then a new Sub-Master is designated by the new master according to its updated piconet device table and sends it to its slaves and bridge(s) after piconet recovery. On the other hand, the lost slave keeps page scan state for Tp. If it is not paged into piconet during the Tp, then the lost slave switches to page state.

According to its stored original piconet device table, it tries to connect with slaves of the original piconet in sequence till a connection is constructed(considering the degree of bridge, Sub-Master only connects with a slave in the original piconet). Then the lost slave updates its piconet device table as well as. As figure 5(c) shows, S1 is out of the communication range of the new master. After keeping page scan state for Tp, the lost slave enters page state and established a connection link with S2.

(3) Piconet Recovery failure. If the piconet recovery is unsuccessful, all devices in the original piconet are reset, then the scatternet is reconstructed.

The algorithm of Fast-Recovery is described as figure 6.

```
Fast-Recovery ()
{
  set T_idle be average time of detecting master
   movement;
  while (T_idle >nT_poll){
    for (i=0; i<pico1.number-1; i++){
      if (the role of a slave is 'SUB')
           then slave enters page state;
     else  other slaves and bridge(s) in the
          original piconet enter page scan state; }
      if (state is complete recovery )
           then return;
     else{if (state is segment recovery)
            then lost slave enters page state
       and tries to connect with a slave in the
       recovered piconet;}
     else{if (state is recovery failure)
            then update devices;
          execute scatternet reconstruction;}

 }
}
```

그림 6. Fast-Recovery 알고리즘
Fig. 6. Fast-Recovery algorithm

Ⅳ. Simulation

In this section, we conduct simulation study to compare the proposed algorithm called NEW with Constructing Long-lived Scatternet called CLLS in

the time of scatternet formation and scatternet reformation. A simulation model is presented and the simulation result is discussed.

4.1 Environment

The algorithm proposed in the paper has been implemented by using C++ and UCBT [13](stands for University of Cincinnati-Bluetooth), which is a ns-2 based Bluetooth network module that simulates the Bluetooth network operations in great details. UCBT is operated based on Cygwin.

We assume the radio range of the Bluetooth devices is 10m, Bluetooth devices are randomly deployed in a square region of 10m×10m. The total number of nodes is 50. The maximum number of active slaves is 5 in piconets. A piconet model of simulation is shown in figure 7.

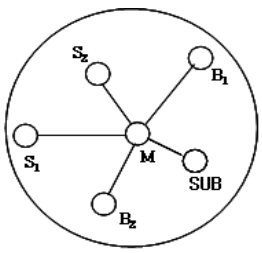


그림 7. 피코넷 모의실험 모델
Fig. 7. A piconet simulation model

A master called M is located at the old of the circle. All of the slaves called S1, S2 and bridge(s) called B1, B2 that are two outermost slaves with respect to the origin are randomly dispersed in the circle area.

4.2 Simulation Results

The proposed algorithm aims at reducing the time of scatternet reformation caused by the master moving out. The table 2 shows the comparison of procedures during running scatternet formation and reformation procedures between CLLS and NEW respectively. Note that the connection is only considered as page process in our algorithm.

표 2. CLLS vs. NEW 처리 비교
Table 2. Comparison of CLLS vs. NEW procedure

procedure / algorithm	Formation	Reformation
CLLS	*device discovery *connection *master selection	*connection *master selection
NEW	*device discovery *connection *master and Sub-Master selection	*connection

Figure 8 shows that the time of scatternet formation in NEW is a few higher than CLLS. After scatternet formation procedure, Sub-Master selection procedure is run. The simulation result indicates the Sub-Master selection has no great influence on the time of scatternet formation.

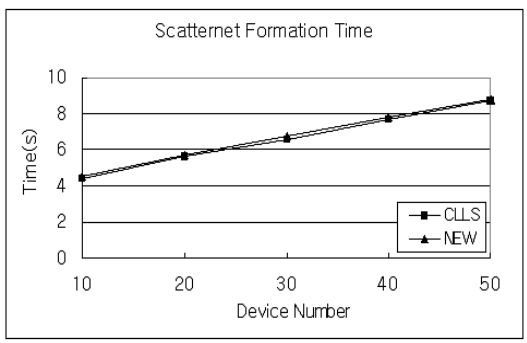


그림 8. 스캐터넷 구성 시간
Fig. 8. Scatternet formation time

Figure 9 shows that our algorithm takes less time than CLLS with respect to the time of scatternet reformation. Note that CLLS more and more increases with the number of devices. However, NEW increases slowly.

The reason is that a Sub-Master is designated in advance in NEW. When the master moves out from its piconet, the slaves and bridge(s) of the original piconet may connect with the Sub-Master directly according to the original piconet device table. Therefore the time of scatternet reformation is reduced.

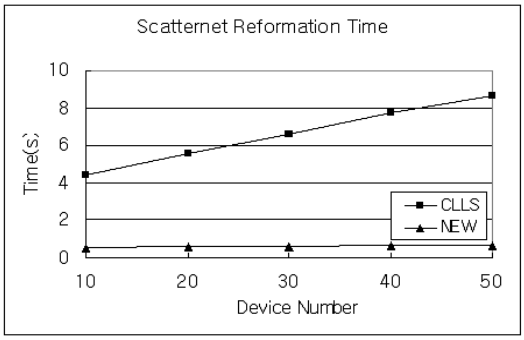


그림 9. 스캐터넷 재구성 시간
Fig. 9. Scatternet reformation time

V. Conclusion and Future Work

In the paper, we have presented an algorithm of Scatternet Fast-Recovery in Bluetooth Networks. A sub-algorithm of selection Sub-Master is designed to select a backup master for scatternet reformation, which not saves the time of scatternet reformation but also reduces communication cost for reconstructing a new master.

The simulation results have shown that scaternet formation is constructed as fast as CLLS. The time of scatternet reformation is much lower than CLLS. For the future work, we will go on the study of device mobility and take into account the "quality" of the resulting topology.

참고문헌

〔1〕 http//www.bluetooth.com/.

〔2〕 S. Basagni, C. Petrioli, "Multihop Scatternet Formation for Bluetooth Networks" IEEE VTC, pp. 424-428, 2002.

〔3〕 C. Petrioli,, S. Basagni, and I. Chlamtac, "Configuring BlueStars: Multihop Scatternet Formation for Bluetooth Networks" IEEE Transactions on Computers, Vol. 52, No. 6, pp. 779-790, Jun, 2003.

〔4〕 G.Tan, A. Miu, J. Guttag, and H. Balakrishnan, "An Efficient Scatternet Formation Algorithm for Dynamic Environments" Proc. IASTEDComm. And Computer Networks (CCN), Nov, 2002.

〔5〕 T. Salonidis, P. Bhagwat, L. Tassiulas, and R. L. Maire, "Distributed Topology Construction of Bluetooth Personal Area Networks", 2001.

〔6〕 G. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees-Scatternet Formation to Bluetooth-Based Ad Hoc Networks" IEEE International Conference, pp. 273-277, 2001.

〔7〕 Y. Liu, M. J. Lee, and T. N. Saadawi, "A Bluetooth Scatternet-Route Structure for Multihop ad hoc Networks "IEEE Journal on Selected Areas in Communications, Vol. 21, No. 2, pp. 229-239, 2003.

〔8〕 C. Law, A. K. Mehta, and K. Siu, "A New Bluetooth Scatternet Formation Protocol" ACM/Kluwer J. Mobile Networks and Applications, Vol. 8, No. 5, pp. 485-498, 2003.

〔9〕 C. F. Chiasserini, M. A. Marsan, "A Distributed Self-Healing Approach to Bluetooth Scatternet Formation", IEEE Trans actions on Wireless Communications, Vol. 4, No. 6, pp. 2649-2654, Nov, 2005.

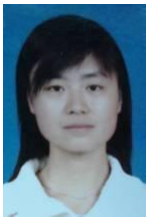
〔10〕 이한욱, 고상근, "노드 형태에 따른 블루투스 스캐터넷 재형성 알고리즘",한국정보과학회논문지, 정보통신, Vol. 32, No. 01, 2005.

〔11〕 H. Y. Chang, C. W. Yu, "Constructing Long-Lived Scatternet in Bluetooth Networks" Proceedings of the Third Internetional Conference on Information Technology and Applications (ICITA'05), 2005.

〔12〕 Z. G. Sheng, L. Qun, W. H. Qiang, and W. Jian, "A New Topology Formation Algorithm for Bluetooh Scatternet" Proceedings of the Second International Conference on Embeded Software and Systems- (ICESS'05), 2005.

〔13〕 UCBT-Bluetooth extension for NS2 at the University of Cincinnati. http://www.ececs.uc.edu/cdmc/ucbt/-ucbt.html

저 자 소 개



장 범

2005년 2월 : University of Science and Technology Anshan 학사

2007년 현재 : 경상대학교 대학원 석사과정

〈관심분야〉 컴퓨터네트워크, 센서 네트워크

구명모

2001년 경상대학교 컴퓨터과학과 석사

2006년 경상대학교 컴퓨터과학과 박사

2006년~현재 진주산업대학교 컴퓨터공학부 겸임교수,(주)센텀 연구소장

〈관심분야〉 멀티미디어통신, 컴퓨터네트워크, 멀티캐스트

김상복

1989년 중앙대학교 전자공학 박사

현재 경상대학교 컴퓨터과학과 교수, 경상대학교 전산정보원장, 경상대학교 컴퓨터정보통신연구소 연구원

〈관심분야〉멀티미디어통신, 한국어정보처리, 컴퓨터프로그래밍