

개선된 테스트 커버리지를 갖는 서브투어 생성 방법

정회원 강 석 규*, 이 옥 빈*, 김 학 서*, 이 부 호**, 이 상 호*

A Method of Subtour Generation with Improved Test Coverage

Seok Qu Kang*, Ok Bin Lee*, Hak Suh Kim*,
Boo Ho Lee**, Sang Ho Lee* *Regular Members*

요 약

프로토콜을 설계하고 이를 구현하면 IUT가 프로토콜 규격과 일치하는지를 검증하는 적합성 시험을 수행한다. 이 적합성 시험의 결과는 생성된 시험열에 의존한다. 기존의 시험열 생성 방법은 오일러 경로에 의한 방법을 사용하였으나 이 방법은 전체 경로를 다 포함하지 못한다는 단점이 있다. 이 논문에서는 대상 프로토콜의 FSM을 이용한 TPT(Transition Possibility Tree)를 생성하여 확장된 테스트 커버리지를 갖는 서브투어 생성 방법을 제안하였다. 제안 방법에 의해서 생성된 시험열은 기존의 방법보다 길어지나 적합성 시험의 질을 보장한다는데 의미가 있다.

ABSTRACT

When a new protocol designed and implemented, a conformance test should be done. It verifies that an implementation conforms to the protocol specification. The quality of the conformance test depends on the test sequences. Test sequence generation methods based on the Euler path can not cover all paths. In this paper, we propose a new subtour generation method, TPT(Transition Possibility Tree), which has extended test coverage. The length of the test sequences of our method is longer than that of Euler, however it guarantees higher quality of the conformance testing.

I. 서 론

통신프로토콜은 통신 엔티티간의 정보교환을 관장하는 규칙의 집합이다. 일단 프로토콜 규격이 설계되고 논리적/의미적 오류(logical/semantic error)가 검증되면 프로토콜 구현을 시작할 수 있다. 프로토콜 구현이 명세와 일치한다는 것을 확인하는 것은 매우 중요하다. 명세서에 대한 IUT(Implementation Under Test)을 검사하는 절차를 적합성 시험(conformance test)이라 한다[1,2,3,5,6,7,8,9]. 이 적합성 시험의 질은 좋은 시험열에 근거한다.

일반적으로 통신 프로토콜은 FSM(Finite State Machine)으로 모델링되며, 대부분의 적합성 시험방법이

FSM을 기반으로 연구되어 왔다. 프로토콜 FSM에 근거하여 일련의 시험열을 생성하는 가장 기본은 transition tour방법이다. 이 방법은 FSM을 오일러(Euler) 그래프로 변환하여 깊이우선 탐색을 행함으로써 FSM에 포함된 모든 전이(transition)를 적어도 한번 이상 갖는 서브투어를 얻는다[3].

그러나 transition tour를 사용하여 얻은 시험열은 의미있는 경로를 포함하지 못하는 단점이 있다[3,4]. 따라서, 오일러 경로에 의한 서브투어 방법에 근거하여 생성한 테스트 케이스는 IUT의 적합성 시험에 있어 완전하지 못한 결과를 가져올 수 있으므로 보다 넓은 테스트 커버리지(test coverage)를 갖는 서브투어 생성 방법이 요구된다.

* 충북대학교 컴퓨터과학과(kangs@cnlab.chungbuk.ac.kr, lobin@cnlab.chungbuk.ac.kr, tuple@cnlab.chungbuk.ac.kr, shlee@c Bucc.chungbuk.ac.kr)

** 한국전자통신연구원 정보통신표준연구센터 표준기반연구팀(bhlee@pec.etri.re.kr)

논문번호 : 98022-0114, 접수일자 : 1998년 1월 14일

이 논문에서는 대상 프로토콜의 FSM을 이용하여 각 상태에서 전이 가능한 모든 상태로 진행하는 트리를 생성하여 확장된 커버리지를 갖는 서브투어 생성 방법을 제안하며, 이 방법을 TPT(Transition possibility tree) 서브투어 생성 방법이라 부르기로 한다. 제안 방법은 각 노드에서 나가는 모든 간선을 추적하여 경로를 생성함으로써 FSM에서 생성될 수 있는 모든 경로를 포함한다. 따라서 기존의 방법보다 훨씬 더 큰 커버리지를 갖는 만큼 시험열의 길이는 더 길어지게 된다. 또한 이 논문에서 생성된 시험열에는 실제 적합성 시험에서 요구되지 않는 시험열이 포함 될 수 있으나, 이 문제는 시험자의 목적에 따라 선택(selection) 과정을 수행하여 이용함으로써 해결 가능하다.

이 논문의 구성은 2장에서 적합성 시험을 위한 서브투어 생성 방법으로 기존의 오일러 방법을 설명하고 3장에서는 TPT 방법을 제안한다. 4장에서는 기존의 방법과 TPT방법을 실험·평가하고 5장에서 결론과 향후 연구방향을 기술한다.

II. 적합성시험을 위한 서브투어 생성

2.1 FSM

이 논문에서 사용된 프로토콜 모델은 식 (2.1)과 같이 정의되는 결정적 FSM이다. 이와 같은 FSM으로부터 다음상태 함수(next state function) NS는 식 (2.2)와 같이 정의되고, 출력함수(output function) Z는 식 (2.3)과 같다.

$$FSM = \{S, I, O\}, \quad S: \text{상태집합}, \quad I: \text{입력 label}, \\ O: \text{출력 label} \quad (2.1)$$

$$NS : S * I \rightarrow S \quad (2.2)$$

$$Z : S * I \rightarrow O \quad (2.3)$$

FSM은 유한그래프 $G = (V, E)$ 로 표시된다. 각 간선은 입력과 출력으로 레이블링되며, 각 노드는 상태를 표시한다. 프로토콜 FSM에 대한 IUT는 입출력 포트(port)를 갖는 black box로 간주되며, IUT의 내부 행위를 관찰할 수 없으므로 적합성 시험은 오직 출력 포트를 통한 외부 행위(external behaviour)를 관찰함으로써만 가능하다. 또한 모든 상태는 초기상태로부터 도달가능하고 또 어느 상태로부터든지 reset을 통해 출력없이 초기상태에 도달할 수 있다고 가정한다.

2.2 오일러 그래프를 이용한 서브투어 생성방법

오일러 그래프는 반복된 몇 개의 간선을 가지는 그래프이다[3]. 즉 그래프에서 각 노드의 입력 간선의 개수와 출력 간선의 개수가 같도록 간선을 추가함으로써, 원래의 그래프가 모든 간선을 순회하도록 구성하며, 초기 노드에서 시작하여 초기 노드로 되돌아오는 여러 개의 서브투어로 구성되는 transition tour를 생성할 수 있다. 오일러 그래프에 의한 서브투어 생성 방법을 그림 1에 나타내었다.

```

struct myfsm{
    int flag;
    int current_st;
    char *inout;
    int next_st;
} fsm[MAXI]; //입력된 오일러 그래프

Euler_subtour_gen()
{
    struct myfsm euler[MAXI]; // Transition tour방법에 의해 생성된 서브투어 저장
    euler[0] = fsm[0]; // initial tour
    fsm[0].flag= 1; // 이미 탐색된 간선임을 표시
    j = 1;
    while (j < 간선의 수)
    {
        for (i = 1; i<간선의 수; i++)
        {
            if((euler[j-1].next_st == fsm[i].current_st) && (fsm[i].flag == 0))
            {
                fsm[i].flag = 1;
                euler[j] = fsm[i];
                j++;
                break;
            }
            // 아직 탐색되지 않은 간선이면서 간선 j-1에 이어지는 간선을 추가
        }
    } // while
}
    
```

그림 1. 오일러 그래프를 이용한 서브투어 생성 알고리즘
Fig. 1 Subtour Generation Algorithm using Euler Graph

그림 1은 원 그래프에 간선이 추가된 오일러 그래프를 입력으로 사용한다. 각 노드의 상태 정보가 입력되어 있으므로, 이전 노드의 다음 상태와 현재 노드의 현재 상태가 같게 되면 이전 노드와 현재 노드를 연결하는 간선을 서브투어에 추가해 나간다. 이 알고리즘은 간선의 최대 개수에 도달할 때까지 계속된다.

이 알고리즘의 시간 복잡도는 간선의 수를 n 으로 하면 $O(n^2)$ 이다. 또한 공간 복잡도는 $O(n)$ 이다. 그러나 오일러 그래프를 이용하는 transition tour 방법은 서브투어를 생성하기 위한 매우 일반적인 방법이지만 경우에 따라 의미있는 경로를 배제할 수 있는 가능성이 있음이 이미 밝혀져 있다[3].

III. TPT를 이용한 서브투어 생성

3.1 개요

이 논문에서 제안하는 TPT는 어떤 상태에서 다음 상태로 갈 수 있는 모든 경로를 조사하여 초기상태까지 다시 돌아오는 경로를 트리형태로 나타낸 것을 말한다. 이러한 TPT를 이용하면 FSM에서 생성할 수 있는 모든 경로가 포함되므로, 오일러 그래프를 이용한 방법보다 훨씬 더 향상된 커버리지를 얻게 된다.

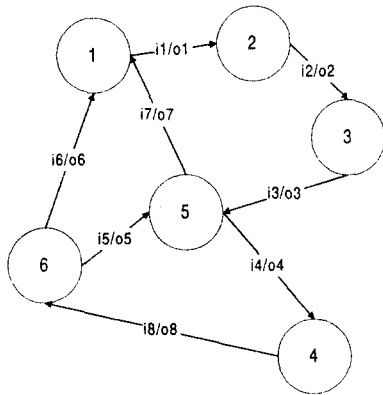


그림 2. 프로토콜의 FSM 예
Fig. 2 Sample of Protocol FSM

그림 2의 상태 1을 프로토콜의 초기상태라 하면 상태 1에서 나가는 간선은 상태 2로 가는 간선 하나 밖에 없다. 이와 같이 나가는 간선이 하나 뿐인 노드는 다음 상태로 진행하면서 간선을 추가해 나가면 된다. 이와 같이 하면 상태 1에서 상태 2로, 상태 2에서 상태 3으로, 상태 3에서 상태 5로 나가는 간선이 차례로 추가되면서 상태 5에 도달하게 된다. 상태 5에서는 나가는 간선이 둘이 있다. 우선 상태 1로 가는 간선을 추가하면 상태 1이 초기상태이므로 하나의 경로가 완성된다. 이제 두 번째 경로를 위해 다시 상태 5까지 도달하는 경로를 만들고 이번에는 상태 5에

서 상태 4로 가는 간선을 추가한다. 상태 4에서는 상태 6으로 가는 간선이 하나 뿐이므로 그 간선을 추가하여 상태 6에 도달하게 되는데 상태 6은 나가는 간선이 둘이다. 우선 상태 1로 가는 간선을 추가하면 상태 1은 초기상태이므로 두 번째 경로가 완성된다. 세 번째 경로는 초기 상태인 상태 1에서 시작하여 상태 5까지 오는 경로에 상태 4와 상태 6으로 가는 간선을 추가하고 상태 6에서는 상태 5로 가는 간선을 택한다. 상태 5에서 상태 1로 가는 간선을 택하면 세 번째 경로가 완성되고, 상태 5에서 상태 4로 가는 간선을 택하면 그림 3과 같은 루프가 발생한다.

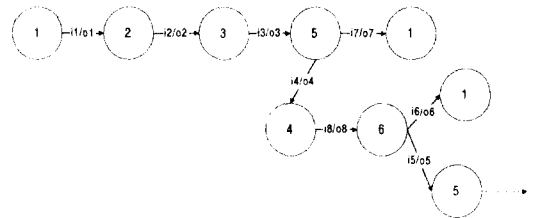


그림 3. TPT방법에 의한 루프 발생
Fig. 3 Loop Detection by TPT Method

이와 같은 방법으로 트리를 증축시키면 TPT를 얻을 수 있다. 그러나 그림 3의 상태 5 → 4 → 6 → 5의 경우와 같이 사이클이 존재하는 경우 트리는 터미널에 도달할 수 없게 된다. 이 경우에서 발생한 루프는 계속해서 같은 경로를 생성하므로 이 방법에서는 1회만 반복하도록 처리한다. 따라서 이 알고리즘에서는 각 노드의 들어오는 간선 수만큼 노드에서 반복할 수 있다.

3.2 알고리즘

3.1절에서 설명한 알고리즘을 그림 4에 나타내었다. 그림 4의 알고리즘에는 입력 프로토콜의 FSM을 나타내기 위한 자료구조와 서브 투어를 얻기 위한 트리의 노드에 대한 자료구조가 기술되어 있다.

```

struct edge{          // FSM의 트랜지션에 대한 정보
    int ini_state;    // 현재상태
    char input[5];   // 입력 label
    char output[5];  // 출력 label
    int next_state;  // 다음상태
} edges[ ];          // 입력된 프로토콜 FSM

struct tree{

```

```

int nodenum; // 노드(global state) 번호
int edge; // 부모노드로부터 전이된 트랜
           // 지션 번호
struct tree *par; // 부모노드에 대한 포인터
struct tree *child[10]; // 자식노드들에 대한 포인터
int childcount; // 자식 노드의 수
};

number_of_edge = input_fsm(); // 노드의 개수
number_of_node = count_node(); // edge의 개수
comput_indegree(); // 각 노드의 indegree 계산
root = make_root(); // tree의 root node 생성
mk_child(root); // tree 생성
maketour(); // 터미널 노드로부터 위로 거슬
           // 러 오르면서
           // 서버투어 출력

void mk_child(struct tree *node)
{
    for(i=0; i<number_of_edge; i++)
    {
        if (edges[i].ini_state == node->nodenum)
        {
            n = edges[i].next_state;
            comp = node; count=0;
            while (comp != NULL)
            {
                if (n == comp->nodenum) count++;
                comp = comp->par;
            } //while 부모노드로 거슬러 오르면서 출
            //회수 count
            if (count < indegree[n] )
                temp=(struct tree *) malloc(sizeof(struct tree));
                temp를 node의 child에 추가;
            /* if(edges[i].ini_state == node->nodenum) */
        } /* for */
        if ( node->childcount == 0 )
            return;
        for (i = 0; i < node->childcount; i++)
        {
            xnode = node->child[i];
            if(xnode->nodenum == 1)
            {
                terminal[terminal_count] = xnode;
                terminal_count++;
                return;
            }
        }
        mk_child(xnode);
    }
}

```

그림 4. TPT에 의한 서버투어 생성 알고리즘
Fig. 4 Subtour Generation Algorithm using TPT Method

주요 프로시저인 mk_child()는 노드의 들어오는 간

선의 수인 indegree만큼 자식 노드를 생성하고 추가된 자식노드는 계속해서 자신의 자식 노드를 생성하며 트리를 생성한다.

제한한 알고리즘의 시간 복잡도는 간선의 수를 n이라 하고 임의의 노드의 최대 outdegree가 m일 때 $O(n^m)$ 이고, 공간 복잡도는 기존 방법과 동일한 $O(n)$ 이다.

IV. 실험 및 평가

이 논문에서 실험 및 평가는 트랜스포트 프로토콜(transport protocol)를 사용하여 오일러 경로에 기반한 transition tour방법에 의해 서버투어를 생성하고, 제안한 TPT방법에 의해 생성된 서버투어의 의미를 비교하기로 한다.

그림 5은 트랜스포트 프로토콜의 FSM이다. 그림 5의 FSM을 이용하여 서버투어 생성시 셀프 루프(self loop)는 전체 서버투어 중에 한번만 나타나게 하였고 한 간선에 두 개 이상의 입력과 출력이 존재하면 임의의 입력과 출력을 사용하여 서버투어를 생성하였다 [3].

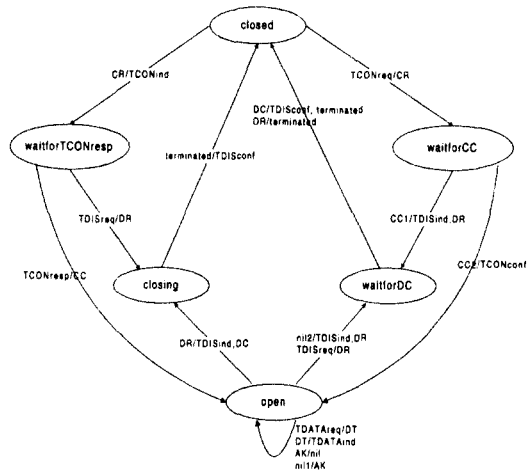


그림 5. Transport Protocol의 FSM
Fig. 5 FSM of Transport Protocol

표 1은 transition tour방법에 의한 서버투어 생성 결과로서 올바른 적합성 시험이 이루어지기 위해서는 표 2의 서버투어가 추가되어야만 한다[3]. 표 3에는 TPT 방법에 의한 서버투어 생성 결과를 나타내었다.

표 1의 서브투어 1, 2, 3, 4, 5는 표 3의 서브투어 1, 2, 3, 4, 5와 각각 동일하다. 또한 반드시 추가되어야 하는 표 2는 표 3의 서브투어 6이다. 이와 같이 TPT 방법에 의한 서브투어 생성은 더 높은 커버리지를 갖는 시험열을 생성할 수 있다.

제안 방법을 사용하면 셸프 루프의 처리 조건에 따라 더 많은 서브투어를 생성할 수 있다. 이 논문에서는 [3]에서와 같이 셸프루프를 한번만 사용하여 서브투어를 생성하였으나, 이 제약조건을 배제시키면 표 3의 서브투어 2, 3, 5, 6에 셸프루프가 추가된 서브투어를 더 생성할 수 있다. 또한 transition tour방

표 1. Transition Tour방법에 의한 서브투어
Table 1. Subtour by transition Tour Method

subtour	state	input	output
subtour 1	closed	TCONreq	CR
	waitforCC	CC1	TDISind,DR
	waitforDC	DC	TDISconf, terminated
	closed		
subtour 2	closed	TCONreq	CR
	waitforCC	CC2	TCONconf
	open	TDATAreq	DT
	open	DT	TDATAind
	open	AK	nil
	open	nil	AK
	open	TDISreq	DR
waitforDC	DC	TDISconf, terminated	
closed			
subtour 3	closed	CR	TCONind
	waitforTCONresp	TCONresp	CC
	open	nil2	TDISind, DR
	waitforDC	DR	terminated
closed			
subtour 4	closed	CR	TCONind
	waitforTCONresp	TDISreq	DR
	closing	terminated	TDISconf
	closed		
subtour 5	closed	CR	TCONind
	waitforTCONresp	TCONresp	CC
	open	DR	TDISind, DC
	closing	terminated	TDISconf
	closed		

표 2. 추가해야 할 서브투어
Table 2. Subtour to be added

subtour	state	input	output
subtour 1	closed	TCONreq	CR
	waitforCC	CC2	TCONconf
	open	DR	TDISconf, DC
	closing	terminated	TDISconf
	closed		

표 3. TPT 방법에 의한 서브투어
Table 3. Subtour by TPT Method

subtour	state	input	output
subtour 1	closed	CR	TCONind
	waitforTCONresp	TDISreq	DR
	closing	terminated	TDISconf
	closed		
subtour 2	closed	CR	TCONind
	waitforTCONresp	TCONresp	CC
	open	DR	TDISind,DC
	closing	terminated	TDISconf
	closed		
subtour 3	closed	CR	TCONind
	waitforTCONresp	TCONresp	CC
	open	nil2	TDISind, DR
	waitforDC	DR	terminated
closed			
subtour 4	closed	TCONreq	CR
	waitforCC	CC1	TDISind, DR
	waitforDC	DC	TDISconf, terminated
	closed		
subtour 5	closed	TCONreq	CR
	waitforCC	CC2	TCONconf
	open	TDATAreq	DT
	open	DT	TDATAind
	open	AK	nil
	open	nil1	AK
	open	TDISreq	DR
waitforDC	DC	TDISconf, terminated	
closed			
subtour 6	closed	TCONreq	CR
	waitforCC	CC2	TCONconf
	open	DR	TDISind, DC
	closing	terminated	TDISconf
closed			

법으로 서브투어를 생성하려면 생성된 transition tour를 적절하게 가공해야 하지만 제안 방법을 사용하면 서브투어가 직접 생성된다는 장점이 있다. 그러나, TPT 방법은 가능한 모든 경로를 생성하기 때문에 적합성 시험시 요구되지 않는 경로가 나타날 수 있다. 이러한 경로는 수작업으로 제거하여 의미있는 경로만을 추출해야 한다. 예를 들면, 서브투어 5와 경로는 같지만 셸프루프를 갖지 않는 경로인 closed → waitforCC → open → waitforDC → closed는 셸프루프를 갖는 서브투어 5에 포함되기 때문에 위의 경로표에 나타나 있지 않다.

이 논문에서 제안된 TPT를 이용한 서브투어 생성 방법의 장점은 존재하는 모든 경로를 탐색하므로 커버리지가 높다는 것이다. 물론 시험열의 길이가 길어지고 알고리즘의 복잡도가 복잡해진다는 단점도 존재한다. 하지만 이 단점은 생성된 시험열이 모든 경로

를 다 포함한다는 장점으로 선택될 수 있다.

V. 결 론

프로토콜을 설계하고 IUT를 시험하는 적합성 시험에서의 관건은 질 좋은 시험열을 생성하는데 있다. 기존에 널리 사용되던 transition tour방법에 의한 서버투어 생성방법은 전체 경로를 포함하지 못하는 단점이 있다. 하지만 이 논문에서 제안한 서버투어 생성방법은 비록 transition tour방법에 의한 서버투어 생성 방법보다 시험열의 길이 증가와 알고리즘의 시간 복잡도의 증가, 적합성 시험시 필요한 시험열을 선택해야 된다는 단점은 있으나, 전체 경로를 포함하는 시험열을 생성한다는 장점으로 인하여 보다 정확한 적합성 시험을 보장할 수 있다는데 의미가 있다.

향후 이 논문에서 제안한 TPT에 의한 방법으로 생성한 서버투어를 이용하여 대상 FSM의 transition fault를 검출하는 방법에 대한 연구가 필요하다.

참 고 문 헌

1. Keith G. Knightson, "OSI Protocol Conformance Testing(IS 9646 Explained)", McGraw-Hill Inc., pp. 1-14, 1993.
2. OSI Conformance Testing Methodology and Framework : IS 9646.
3. B. Sarikaya. "Principles of Protocol Engineering and Conformance Testing", Ellis Horwood, pp. 31-54, 1993.
4. K. Naik and B. Sarikaya, "Verification of Protocol Conformance Test Cases Using Reachability Analysis", Journal of Systems and Software, Vol. 19 No. 1, Sep. 1992.
5. J. F. Billiard, "Methodology and Tools for Qualitative Protocol Validation," Protocol Specification, Testing and Verification III, 1983.
6. B. Kanugo, L. Lamont, R. L. Probert and H. Ural, "A Useful FSM representation for Test Suite Design and Development," Protocol specification, Testing, and Verification VI, 1986.
7. R. L. Probert and H. Ural, "Requirements for a Test Specification Language for Protocol Implementation Testing," Protocol Specification, Testing, and Verification III, 1983.
8. D. Rayner, "Towards an Objectives Understanding

of Conformance," Protocol Specification, Testing, and Verification, III, 1983.

9. N. H. Sherif, G. L. Hoover and R. P. Wiederhold, "X.25 conformance Testing - A tutorial," IEEE Communications Magazine Vol. 24, No. 1, pp. 16-27, 1986.



강 석 규(Seok Qu Kang) 정회원
1986년 3월~1993년 2월: 청주대학교 전자계산학과 (공학사)
1993년 3월~1995년 8월: 충북대학교 전자계산학과 (이학석사)
1996년 3월~현재: 충북대학교 전자계산학과 박사과정 재학중

1997년 3월~현재: 주성대학 전자계산학과 전임강사
<연구분야> 프로토콜공학, 네트워크 디자인
E-mail : kangs@cnlab.chungbuk.ac.kr



이 옥 빈(Ok Bin Lee) 정회원
1985년 3월~1990년 2월: 조선대학교 전자계산학과 (이학사)
1993년 3월~1995년 8월: 조선대학교 전자계산학과 (이학석사)
1995년 3월~1997년 8월: 충북대학교 전자계산학과 박사과정 수료

<연구분야> 프로토콜 공학, 정보보호
E-mail : lobin@cnlab.chungbuk.ac.kr



김 학 서(Hak Suh Kim) 정회원
1990년 3월~1997년 2월: 청주대학교 전자계산학과 (공학사)
1997년 3월~현재: 충북대학교 전자계산학과 재학중
<연구분야> 프로토콜 공학, ATM 네트워크

E-mail : tuple@cnlab.chungbuk.ac.kr



이 부 호(Boo Ho Lee) 정회원

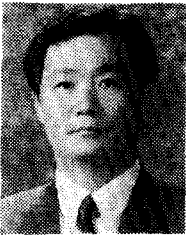
1981년 3월~1986년 2월: 한양대
학교 전자통신공학
과 학사

1989년 3월~1991년 2월: 한양대
학교 전자통신공학
과 석사

1991년 2월~현재: 한국전자통신
연구원 정보통신표
준연구센터, (현)표
준기반연구팀 선임
연구원

<연구분야> 프로토콜 공학

E-Mail : bhlee@pec.etri.re.kr



이 상 호(Sang Ho Lee) 정회원

1972년 3월~1976년 2월: 숭실대
학교 전자계산학과
(공학사)

1979년 3월~1981년 2월: 숭실대
학교 전자계산학과
(공학석사)

1985년 3월~1989년 2월: 숭실대
학교 전자계산학과
(공학박사)

1981년 3월~현재: 충북대학교 컴퓨터학과 교수

<연구분야> 프로토콜 공학, 시뮬레이션, 정보보호, ATM
네트워크

E-mail : shlee@cbucc.chungbuk.ac.kr