2000 년 한국정보처리학회 추계 학술발표논문집 제 7 권 제 2 호

TPT 서브투어 생성에서의 시험열 선택

김학서*, 안병준*, 이상호**
*한국전자통신연구원 네트워크응용팀
**충북대학교 컴퓨터과학과
e-mail : tuple@etri.re.kr

Test Case Selection of TPT Subtour Generation

Hak Suh Kim*, Byungjun Ahn*, Sang ho Lee**
*Multiservice Switching Team, ETRI
**Dept. of Computer Science, Chungbuk Nat'l University

요 약

프로토콜을 설계하고 이를 구현하면 IUT 가 프로토콜 규격과 일치하는지를 검증하는 적합성 시험을 수행한다. 기존의 시험열 생성 방법은 오일러 경로에 의한 방법을 사용하였으나 이 방법은 전체 경로를 다 포함하지 못한다는 단점이 있다. 이 단점을 개선하여 TPT(Transition Possibility Tree)를 생성하여 확장된 테스트 커버리지를 갖는 서브투어 생성방법이 제안되었다. 이 논문에서 기존에 제안된 TPT 서브투어 생성 방법에서 시험열의 갯수가 많아 진다는 단점을 보안하여 TPT 서브투어 생성 방법에서의 효과적인 시험열 선택 방법을 제안하였다. 제안 방법에 의해서 생성된 시험열도 완전한 시험 목적에 부합하지 않는다는 단점은 있으나 시험자가 선택해야 하는 시험열의 양을 최소화하는데 그 의미가 있다.

1. 서론

적합성 시험은 프로토콜 공학의 구현 단계 중 마지막 단계로, 구현된 프로토콜이 명세서(specification)에 기술된 사항을 만족하는가를 테스트 케이스를 이용하여 독립적으로 시험하는 작업이다[1,2].

통신 프로토콜은 통신 엔티티간의 정보교환을 관장하는 규칙의 집합이다. 일단 프로토콜 규격이 설계되고 논리적· 의미적 오류가 검증되면 프로토콜 구현을 시작할 수 있다. 프로토콜 구현이 명세와 일치한다는 것을 확신하는 것은 매우 중요하다. 명세서에 대한 IUT 를 검사하는 절차를 적합성 시험이라 한다[1,2,3,5,6,7,8,9,11]. 이 적합성 시험의 질은 좋은 시험열에 근거한다.

일반적으로 통신 프로토콜은 FSM(Finite State Machine)으로 모델링되며, 대부분의 적합성 시험방법이 FSM 을 기반으로 연구되어 왔다. 프로토콜 FSM 에 근거하여 일련의 시험열을 생성하는 가장 기본은 Transition Tour 방법이다. 이 방법은 FSM 을 오일러(Euler) 그래프로 변환하여 깊이 우선 탐색을 행함으로써 FSM 에 포함된 모든 전이(transition)를 적어도 한 번 이상 갖는 서브투어를 얻는다[3].

그러나 Transition Tour 를 사용하여 얻은 시험열은 의미 있는 경로를 모두 포함하지는 못하는 단점이 있다[3,4,10]. 따라서, 오일러 경로에 의한 서브투어 방법에 근거하여 생성한 테스트 케이스는 IUT 의 적합성 시험에 있어 완전하지 못한 결과를 가져올 수 있으므로 보다 넓은 테스트 커버리지(test coverage)를 갖는 서브투어 생성 방법이 요구된다.

대상 프로토콜의 FSM 을 이용하여 각 상태에서 전이 가능한 모든 상태로 진행하는 트리를 생성하여 확장된 커버리지를 갖는 테스트 케이스 생성 방법을 제안되었다[10]. TPT 서브투어 생성 방법은 각 노드에서 나가는 모든 에지를 추적하여 경로를 생성함으로써 FSM 에서 생성될 수 있는 모든 경로를 포함한다. 따라서 기존의 방법보다 훨씬 더 큰 커버리지를 갖는 만큼 시험열의 길이는 더 길어지게 된다. 또한 생성된 시험열에는 실제 적합성 시험에서 요구되지 않는 시험열이 포함 될 수 있다.

이 논문에서는 기존에 제안된 TPT 서브투어 방법의 단점인 시험열 선택 문제를 서브투어 생성시 시험열 선택 정보를 이용하여 해결하는 방법을 제안한다.

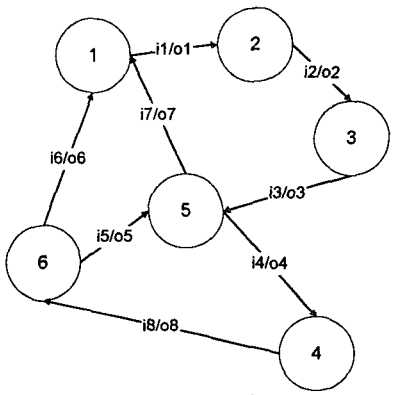
이 논문의 구성은 2 장에서 관련 연구로 오일러 방

법의 전체 테스트 케이스를 포함하지 못하는 단점을 개선한 TPT 방법을 설명하고, 3 장에서 시험열 선택 정보를 이용한 TPT 방법의 성능개선 방안을 제안한다. 4 장에서는 제안된 방법을 실험 및 평가하고 5 장에서 결론과 향후 연구방향을 기술한다.

2. TPT 를 이용한 서브투어 생성 방법

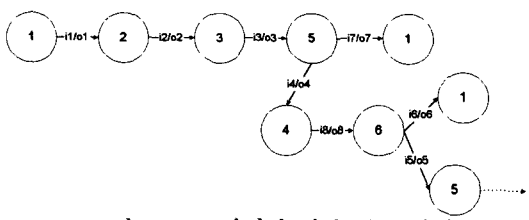
2.1 개요

TPT 는 어떤 상태에서 다음 상태로 갈 수 있는 모든 경로를 조사하여 초기상태까지 다시 돌아오는 경로를 트리 형태로 나타낸 것을 말한다. 이러한 TPT 를 이용하면 FSM 에서 생성할 수 있는 모든 경로가 포함되므로, 오일러 그래프를 이용한 방법보다 훨씬 더 향상된 커버리지를 얻게 된다.



[그림 1] 예제 FSM

[그림 1]의 상태 1 을 프로토콜의 초기상태라 하면 상태 1 에서 나가는 에지는 상태 2 로 가는 에지 하나 밖에 없다. 이와 같이 나가는 에지가 하나 뿐인 노드는 다음 상태로 진행하면서 에지를 추가해 나가면 된다. 이와 같이 하면 상태 1 에서 상태 2 로, 상태 2 에서 상태 3 으로, 상태 3 에서 상태 5 로 나가는 에지가 차례로 추가되면서 상태 5 에 도달하게 된다. 상태 5 에서는 나가는 에지가 둘이 있다. 우선 상태 1 로 가는 에지를 추가하면 상태 1 이 초기상태이므로 하나의 경로가 완성된다. 이제 두 번째 경로를 위해 다시 상태 5 까지 도달하는 경로를 만들고 이번에는 상태 5 에서 상태 4 로 가는 에지를 추가한다. 상태 4 에서는 상태 6 으로 가는 에지가 하나 뿐이므로 그 에지를 추가하여 상태 6 에 도달하게 되는데 상태 6 은 나가는 에지가 둘이다. 우선 상태 1 로 가는 에지를 추가하면 상태 1 은 초기상태이므로 두 번째 경로가 완성된다. 세 번째 경로는 초기 상태인 상태 1 에서 시작하여 상태 5 까지 오는 경로에 상태 4 와 상태 6 으로 가는 에지를 추가하고 상태 6 에서는 상태 5 로 가는 에지를 택한다. 상태 5 에서 상태 1 로 가는 에지를 택하면 세 번째 경로가 완성되고, 상태 5 에서 상태 4 로 가는 에지를 택하면 [그림 2]와 같은 루프가 발생한다.



[그림 2] TPT 방법에 의한 루프 발생

이와 같은 방법으로 트리를 증축시키면 TPT 를 얻을 수 있다. 그러나 <그림 3-4>의 상태 5->4->6->5 의 경우와 같이 싸이클이 존재하는 경우 트리는 터미널에 도달 할 수 없게 된다. 이와 같은 경우에 발생한 루프는 계속해서 같은 경로를 생성함으로, 제안 방법에서는 1 회만 반복하도록 처리한다. 따라서 이 알고리즘에서는 각 노드의 들어오는 에지 수만큼 노드에서 반복할 수 있다[10].

2.2 알고리즘

2.1 절에서 설명한 알고리즘을 [그림 3]에 나타내었다. [그림 3]의 알고리즘에는 입력 프로토콜의 FSM 을 나타내기 위한 자료구조와 서브 투어를 얻기 위한 트리의 노드에 대한 자료구조가 기술되어 있다.

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

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

number_of_edge = input_fsm();  //  노드의 개수
number_of_node = count_node(); //  edge 의 개수
comput_indegree();             // 각 노드의 indegree 계산
root = make_root(); // tree 의 root node 생성
mk_child(root);     // tree 생성
maketour();          // 터미널 노드로부터 위로 거슬러
                     // 오르면서 서브투어 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
                // 출현회수 count
            if (count < indegree[n])
                temp = (struct tree *)malloc(sizeof(struct
tree));
                temp 를 node 의 child 에 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 3] TPT 에 의한 서브투어 생성 알고리즘

주요 프로시저인 mk_child()는 노드의 들어오는 에지의 수인 indegree 만큼 자식 노드를 생성하고 추가된 자식 노드는 계속해서 자신의 자식 노드를 생성하며 트리를 생성한다.

3. TPT 생성 방법에서의 시험열 선택

2 장에서 기술된 TPT 기반의 서브투어 생성방법은 시험열의 개수가 많아지고 길이가 증가하며 시간 복잡도가 증가된다[10].

이 논문에서는 TPT 방법의 단점을 최소화하기 위하여 생성된 시험열을 시험의 목적에 맞게 사전에 선택할 수 있는 방법으로 시험자의 시험 목적에 맞는 시험열 생성을 위하여 미리 시험자의 목적에 따라 시험열 선택 정보를 이용하는 시험열 선택 방법을 제안한다.

시험자는 시험열 생성 전에 자신의 목적에 맞는 시험열 선택 정보를 입력받아 이 정보를 생성된 시험열과 비교하여 시험자의 시험 목적에 최대한 부합하는 시험열을 생성하는 방법이다.

시험자는 시험 목적을 고려하여 노드들의 전이 상태를 파악하고, 시험열에 포함되어야 할 노드들의 정보를 이용하여 시험열을 선택하는 방법이다.

제한된 시험열 선택 방법의 자료구조는 [그림 4]와 같다.

```
struct test_purpose_fsm {
        int repeat_count;       // 반복 횟수
        int current_st;         // 현재 FSM 노드
        char *inout;            // 입력 및 출력 값
        int next_st;            // 다음 FSM 노드
} fsm
```

[그림 4] 제안 방법의 자료 구조

제안 방법의 알고리즘은 [그림 5]에 나타낸다.

```
void extract_test_case_purpose
{
  while(test_case != NULL)
  {
      while(!각각 시험열이 종료 노드)
      {
          if(생성된 시험열의 각 노드 == 시험열 선
          택 정보의 노드 && 생성된 시험열 상태
          반복 횟수 <= repeat_count)
          {
              해당 시험열 선택
          }
      }
  }
}
```

[그림 5] 제안 방법의 알고리즘

시험열 선택 알고리즘은 생성된 전체 시험열중에서 각 시험열의 노드 정보를 시험자가 미리 입력한 시험열 선택 정보의 노드 정보와 비교하고 , 또한 해당 시험열에서 이 노드가 몇 번 반복되어야 하는지를 나타내는 반복 횟수를 이용하여 이 조건에 만족하는 시험열만을 선택하게 된다.

이 제안 방법을 이용하면 TPT 기반의 서브투어 방법에서 발생하는 시험열 증가와 시험열의 길이 증가 단점을 해결할 수 있다.

그러나 시험자가 대상 프로토콜 FSM 의 전체 상태 노드를 시험열 선택 정보로 입력하게 되면 전체 시험열이 포함되어야 함으로 기존의 TPT 기반의 서브투어 생성 방법과 동일한 시험열을 생성한다. 또한 제안된 선택 알고리즘의 적용으로 전체적인 시간 복잡도가 증가한다는 단점이 존재한다.

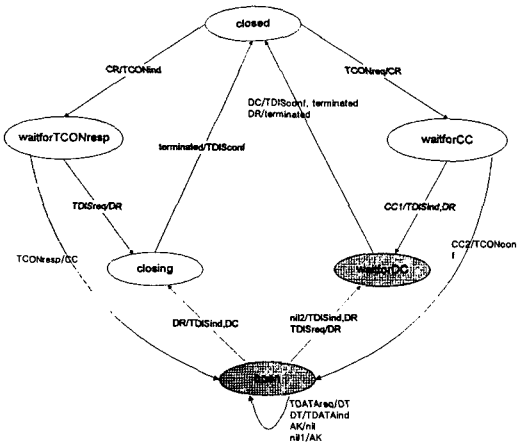
4. 실험 및 평가

이 논문에서 실험 및 평가는 참고문헌[10]과 비교를 위하여 트랜스포트 프로토콜(Transport Protocol)을 이용한다.

[그림 6]은 트랜스포트 프로토콜의 FSM 이다. 서브투어 생성시 루프 발생등의 조건은 참고문헌[10]과 동일 조건으로 실험하였다.

[그림 6]에서 회색으로 표시된 open, waitforDC 두 상태가 각각 한번씩 나타나는 시험열을 선택하기 위하여 시험열 선택 정보를 입력한다.

TPT 방법에 의해서 생성된 서브투어를 [표 1]에 나타내었다.



[그림 6] 트랜스포트 프로토콜의 FSM

[표 1] TPT 방법에 의한 서브투어

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

제안된 시험열 선택 방법을 적용하면 [표 2]와 같은 시험열을 얻을 수 있다.

[표 2] 제안된 방법에 의한 서브투어

subtour	state	input	output
subtour 3	closed waitforTCONresp open waitforDC closed	CR TCONresp nil2 DR	TCONind CC TDISind, DR terminated

[표 2]에 나타난 바와 같이 제안된 방법을 기존의 TPT 기반의 서브투어 생성 방법에 적용하면 시험자의 선택 과정을 최소화할 수 있다. 그러나 시험자의 시험열 선택 정보가 전체 노드를 포함한다면 기존의 TPT 방법과 동일한 결과를 생성하고 이로 인하여 전체적인 알고리즘 시간 복잡도가 증가한다는 단점이 있다.

5. 결론

프로토콜을 설계하고 IUT 를 시험하는 적합성 시험에서의 관건은 질 좋은 시험열을 생성하는데 있다. 이 논문에서는 기존에 제안된 TPT 기반의 서브투어 생성 방법의 단점인 시험열 개수의 증가문제를 해결하고자 시험열 선택 정보를 사전에 입력받아 이를 이용하여 시험열을 생성하는 방법을 제안하였다. 이 논문에서 제안한 방법이 시험자의 선택 정보로 인하여 알고리즘이 복잡하고 또한 최악의 경우 기존의 TPT 방법과 동일한 결과를 초래하는 단점은 있으나, 시험자의 선택 과정을 최소화 시켜 줄 수 있다는 장점을 가지고 있다.

향후 시험의 목적에 맞는 최적의 시험열 선택 과정에 대한 연구가 필요하다.

참고문헌

[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 Confor- mance 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

[10] 강석규, 이옥빈, 김학서, 이부호, 이상호, "개선된 테스트 커버리지를 갖는 서브투어 생성 방법," 한국통신학회 논문지, 23 권, 11 호, 1998

[11] 이상호, S.T. Vuong, "프로토콜 적합성 시험을 위한 통합 환경", 정보과학회논문지, 제 21 권 제 5 호, pp.944-960, 1994