

개선된 Partial UIO sequence 생성 방법의 제안

正會員 崔 珍 領* 正會員 洪 范 基*

Improved Partial UIO sequence generation method

Jin Young Choi*, Beom Kee Hong*, *Regular Members*

요 약

프로토콜 적합성 시험은 시험 대상인 구현물을 하나의 black box로서 고려하여 input에 대한 output과 결과 상태를 확인하는 일련의 과정으로 구성된다. 특히 결과 상태를 확인하는 방법에 따라 Unique Input/Output(UIO) Sequence, Distinguishing Sequence(DS) 및 Characterization Set(CS) 등으로 구분할 수 있다. 또한 UIO sequence가 존재하지 않는 상태에 대한 시험을 위해서도 Partial UIO sequence 방법이 제안되었다. 본 논문에서는 이러한 PUIO sequence에 대해 3가지 property를 제시하고 이를 이용한 개선된 PUIO sequence 알고리즘을 제안하였으며, 그 적용 결과로서 PUIO sequence의 수가 55%정도 줄어드는 것을 확인할 수 있었다.

ABSTRACT

Protocol conformance testing consists of procedures to observe an output and to check a transition state of the Implementation Under Test considered as a black box by applying an input. There are several methods to check the transition state such as Unique Input/Output(UIO) sequence, Distinguishing Sequence(DS) and Characterization Set(CS). Particularly, as a test method for a state having no UIO sequence, Partial UIO sequence method can be considered. In this paper, three properties which can be found among Partial UIO sequences and a modified algorithm using these properties are suggested.

* 韓國電子通信研究所
Electronics And Telecommunication Research Institute.

論文番號 : 94198
接受日字 : 1994年 7月 27日

I. 서론

통신 프로토콜들은 복잡도가 증가하고 높은 신뢰도가 요구됨에 따라 체계적이고 규칙적인 방법들을 사용하여 규정하는 것이 일반적이며, 이러한 규격에 따라 구현된 통신 프로토콜 시스템에 대해서는 상용화되기 이전에 규격과 부합되게 구현되어 정상적으로 동작하는지의 여부를 검사하는 적합성 시험(Conformance Test)이 이루어져야 한다. 프로토콜 적합성 시험은 서로 다른 환경에서 구현된 시스템들간의 상호 운용성(Interoperability)의 가능성을 증가시켜주는 시험으로서도 중요한 의미를 갖는다.

적합성 시험은 시험 대상인 구현물 즉 Implementation Under Test(IUT)를 하나의 black box로 간주하고 외부와의 동작을 관찰함으로써 수행된다. 이 때 시험을 위한 test-case는 프로토콜 규격에서 규정하고 있는 통신 프로토콜 시스템의 기능과 동작에 기반을 두고 만들어지게 된다. 프로토콜 규격은 Finite State Machine(FSM)으로 모델링이 되는 제어 부분과 프로그램 세그먼트로 모델링되는 데이터 부분으로 나눌 수 있는데, 본 논문에서는 프로토콜 규격의 근간이 되는 제어 부분에 대한 적합성 시험만을 주대상으로 고려하였다. 이러한 프로토콜 적합성 시험을 위한 접근방법으로서 규격에서 정의된 FSM의 각 상태 transition에 대해 IUT의 동작을 확인하는 Checking experiment를 고려할 수 있는데, 이것은 FSM에서 임의의 시험대상이 되는 상태에서의 각각의 input에 대한 output과 함께 천이 상태까지도 올바르게 이루어졌는지를 확인하는 과정으로 구성된다. 이 때의 test-case는 규격에서 정의된 각각의 상태에 대한 일련의 input과 output으로 이루어진다.

지금까지 Checking experiment에 근거한 많은 test-case 생성 방법이 제안되었는데, transition의 결과 상태를 확인하기 위한 방법에 따라 Unique Input/Output(UIO) sequence, Distinguishing Sequence(DS) 그리고 Characterization Set(CS) 등을 이용한 방법으로 구분할 수 있다.⁽⁸⁾ 특정 상태에 대한 UIO sequence는 다른 상태에서는 이러한 일련의 input/output이 생성될 수 없는 input/output의 연속으로서, FSM에서 정의된 모든 상태에서 특정 상태를 구분하는데 사용될 수 있다. DS는 FSM에서 정의된 모든 상태에 대해 다른 output sequence를 생성하는 일련의 input sequence로서 UIO와 마찬가지로 서로 다른 상태들을 식별하는

데 사용될 수 있다. 반면에 CS는 단지 모든 상태들 중에서 set에 속하는 일부 상태들만이 다른 output sequence를 생성하는 input sequence로 정의된다.

그러나, 실제 프로토콜의 FSM은 모든 상태에 대해서 DS를 갖는 경우가 매우 적으며, 이를 보완할 수 있는 방안으로서 CS를 이용한 시험 방법은 일반적으로 UIO sequence 보다 길이가 길기 때문에 최근에는 주로 UIO sequence를 이용하여 test-case를 생성하는 방법이 많이 사용되고 있다. 하지만 UIO sequence를 가지 않는 상태에 대한 확인이 문제가 되었고, 이를 해결하기 위한 방법으로서 Partial UIO(PUIO) sequence를 이용한 접근방법이 제안되었다.⁽⁴⁾ 제안된 PUIO sequence 알고리즘은 한 상태에 대해 가능한 모든 PUIO sequence를 구한 후, 그 상태를 다른 모든 상태로부터 식별하는데 사용될 수 있는 몇 개의 PUIO sequence를 선택하여 test-case를 구성하였다.

그런데, 최소 길이의 test-case 구성을 위해서 필요한 최적의 PUIO sequence들을 구하는 문제는 set cover 문제와 equivalent한 NP-complete로서 대상이 되는 PUIO sequence의 수가 작을수록 최적해를 구하는데 효율적이라고 할 수 있다. 따라서 본 논문에서는 UIO sequence를 갖지 않는 상태로의 transition에 대한 결과 상태를 확인하는 방법으로서 PUIO sequence를 이용한 test-case 생성 방법을 고려할 때, Chun 등이 제안한 PUIO sequence 알고리즘에서 발견할 수 있는 몇 가지 property를 제안하고 이것을 이용하여 전체 test-case의 길이를 최소화할 수 있는 최적의 PUIO sequence set을 구하기 위해서 필요한 최소한의 PUIO sequence set을 구하는 방법에 관해서 다루고자 한다.

본 논문의 구성으로서 2장에서는 결정적(Deterministic) FSM 모델에 대한 정의와 가정 사항들에 대해서 기술하고, 3장에서는 PUIO sequence의 정의와 개선된 PUIO sequence 알고리즘을 제안한다. 4장에서는 3장에서 제시한 알고리즘을 Chun 등이 제안한 알고리즘과 비교한 적용 예를 보이고, 5장에서는 PUIO sequence를 이용한 test-case 생성방법에 관해 기술하고자 한다.

II. Finite State Machine Model

지금까지 제안된 기존의 test-case 생성 방법은 다음과 같이 정의되는 결정적 FSM모델에 그 기초를 두고 있다. 결정적 FSM이란 한 상태에서 같은 input을 갖는 transition이 두 개 이상 존재하지 않는 FSM을

말한다.

(Definition 1) 결정적 FSM 모델

결정적 FSM 모델은 $\langle S, v_0, I, O, T \rangle$ 로 표현되는 transition 시스템이다. 이때 S 는 유한 상태의 set, v_0 는 $v_0 \in S$ 인 초기 상태, I 와 O 는 각각 input과 output을 나타내는 유한 set, 그리고 T 는 transition 매핑을 나타내는 함수로서 $T: S \times I \rightarrow S \times O$ 로 정의된다.

임의의 결정적 FSM에 대한 상태 천이 함수 T 는

$(v_i, v_j; i/o)$ 로 표현될 수 있는데, 이것은 상태 v_i 에서 input i 를 받으면 output o 와 함께 상태 v_j 로 천이함을 나타낸다.

이러한 결정적 FSM 모델은 directed graph $G=(V, E)$ 로 표현될 수 있다. $V=(v_1, \dots, v_n)$ 은 결정적 FSM의 상태를 나타내는 vertex들의 set을 나타내고, $E=(\{v_i, v_j; L\}; v_i, v_j \in V, L=a_k/o_l, a_k \in I \text{ and } o_l \in O)$ 는 상태 천이 함수 T 에 의해 정의되는 상태 천이를 나타내는 edge들의 set을 나타낸다. 이때 각각의 edge는 input과 output을 레이블로 갖는다. 그림 1은 directed graph로 표현된 FSM의 예이다.

이렇게 표현된 directed graph $G=(V, E)$ 를 대상으로 하여 지금까지 다각적인 방법을 이용한 test-case 생성 방법이 연구되어 왔다. 그러나, 이러한 방법들은 모두 결정적 FSM이 만족해야 하는 몇 가지 제한 사항을 근거로 하고 있다.

첫째, 결정적 FSM 모델은 minimal 조건과 strongly connected 조건을 만족해야 한다. 즉, equivalent한 상태가 존재하지 않아야 하며, 임의의 한 상태에서 임의의 다른 모든 상태로 이동할 수 있는 경로가 존재하여야 한다.

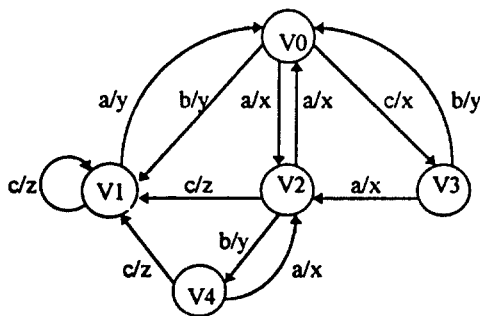


그림 1 DFSM의 예
Fig 1 An Example of DFSM

둘째, 결정적 FSM 모델은 초기 상태를 가지며 임의의 다른 상태로 부터 reset input(r_i)에 의해서 output은 발생시키지 않고 초기 상태로의 천이가 가능해야 한다.

이러한 제한 사항들은 또한 실제 프로토콜이 일반적으로 만족하고 있는 성질이기도 하다. 본 논문에서도 대상이 되는 결정적 FSM 모델이 이러한 제한 사항들을 만족하는 것을 전제로 한다.

III. Modified PUIO sequence 생성 알고리즘

$\Pi \cup T$ 에 대한 프로토콜 적합성 시험은 2장에서 정의된 directed graph $G=(V, E)$ 의 edge set $E=(\{v_i, v_j; L\}; v_i, v_j \in V, L=a_k/o_l, a_k \in I \text{ and } o_l \in O)$ 에 속해 있는 각각의 edge에 대해 input a_k 에 대한 output o_l 과 결과 상태 v_j 를 관찰함으로써 수행되게 된다. 이 때 앞에서 언급한 바와 같이 결과상태를 확인하는 방법으로 UIO, DS 또는 CS 등을 이용하는 방법이 있지만 대부분 프로토콜들의 상태들은 UIO sequence를 가지며 또한 이 sequence는 DS, CS 보다 길이가 짧다고 알려졌기 때문에 최근에는 test-case를 구성하는데 주로 UIO를 이용하는 연구가 많이 수행되고 있다.

그러나, 이 경우에도 UIO가 존재하지 않는 상태에 대한 시험은 그 결과 상태를 확인하는 것이 문제가 되었고, 이에 대한 해결 방안으로 Partial UIO sequence를 이용한 test-case 생성 방법이 제안되었다.^[4]

(Definition 2) Partial UIO sequence(PUIO)

상태 v_i 에서의 transition $(v_i, v_j; \sigma)$ 에 대해서 set $V - \{v_i\}$ ES에 속하는 임의의 상태 v_j 는 이러한 input/output sequence σ 를 산출하지 않을 때 sequence σ 를 v_i 에서의 PUIO sequence라고 정의한다. 이때 ES는 set $V - \{v_i\}$ 의 subset으로서 ES에 속하는 임의의 상태 v_k 에 대해서는 v_i 와 동일한 input/output sequence σ 가 산출되며, 이를 Exclusion Set이라고 한다. v_i' 은 v_i 에서 sequence σ 에 의해 천이한 결과 상태를 나타낸다.

따라서, 상태 v_i 에서의 PUIO sequence σ 에 의해서 ES에 속하지 않는 상태들에 대해서는 UIO와 마찬가지로 원리에 의해 상태 식별이 가능해지게 된다. 이러한 상태들의 set을 $IS=V-ES-\{v_i\}$ 로 정의하고 Inclusion Set이라고 하자. 이때 IS에 속한 상태들은 PUIO sequence σ 에 의해서 v_i 와 구별될 수 있게 된다. 예를 들면, 그

림 1에서 상태 u_4 는 UIO sequence를 갖지 않으며, 상태 u_4 에 대해 정의된 input/output sequence c/z 는 $ES=\{u_1, u_2\}$ 인 하나의 PUIO sequence이다. 이때 $IS=\{u_0, u_3\}$ 는 이러한 input/output sequence c/z 를 갖지 않지만, ES 에 속한 u_1, u_2 는 input c 에 대한 output z 를 생성한다. 따라서, 이 때의 PUIO sequence c/z 에 의해서 u_4 는 $IS=\{u_0, u_3\}$ 와는 상태 식별이 가능하지만, $ES=\{u_1, u_2\}$ 와는 상태 식별이 될 수 없다. 반면에 UIO sequence는 $ES=\emptyset$ 이고 $IS=V-\{u_1\}$ 인 경우에 해당되게 된다. 이런 점에서 PUIO sequence는 UIO sequence와 비슷하다고 할 수 있다.

이러한 PUIO sequence를 찾기 위한 방법으로서 branch and bound 알고리즘을 이용하여 PUIO sequence와 ES set을 구하는 알고리즘이 제안되었다.¹⁾ 이 알고리즘은 Sabnani 등이 제안한 UIO sequence 알고리즘을 약간 개선한 방법이지만 여기서는 한 상태에 대해 가능한 모든 PUIO sequence를 구한 후 그 중에서 최적의 PUIO sequence들을 찾고 있다. 그런데, 여기서 제안된 PUIO sequence 알고리즘에서는 생성되는 PUIO sequence들간에 몇 가지 property를 발견할 수가 있고, 이를 알고리즘에 적용하면 가능한 모든 노드들을 branch하지 않고 중간 과정에서 property에 의해 일부 노드들을 고려 대상에서 제외시킴으로써 필요없는 PUIO sequence가 일부 제외된 축소된 PUIO sequence set을 구할 수가 있다. 이를 이용하면 상태식별이 가능하면서 최소 길이를 갖는 최적의 test case를 구할 때, Chun 등이 제안한 것과 같이 모든 가능한 PUIO sequence set에 대해 구하는 것보다 매우 효율적이다.

만일, Transition $(v_i, v_k : a_i/o_i)$ 의 결과 상태 v_k 가 UIO가 존재하지 않을 때, 노드 g_k^σ 를 $\langle v_k^\sigma, P^\sigma, ES^\sigma \rangle$ 라고 정의하고, 이때의 v_k^σ 는 상태 v_k 에서 sequence σ 를 적용시킨 후의 결과 상태를 나타내고, P^σ 는 $V-\{v_k\}-ES^\sigma$ 에 속하는 임의의 상태에 대해 sequence σ 를 적용시켰을 때 발생할 수 있는 결과 상태를, 그리고 ES^σ 는 sequence σ 에 의해 v_k^σ 로 transtion을 하는 상태들의 set을 나타낸다고 하면 다음과 같은 property가 성립한다.

(Property 1)

vertex v_k 에 대한 PUIO sequence, $PUIO(v_k)$ 를 구할 때, 레벨 l 에서 생성된 m 번째의 노드 $\langle v_k^{\sigma^l}, P^{\sigma^l}, ES^{\sigma^l} \rangle >^m$ 의 σ^l 이 PUIO sequence가 아닐 때, $v_k^{\sigma^l}$ 에 대해 가

능한 n 개의 input/output $a_j/o_j(j=1,\dots,n)$ 에 대해서 레벨 $l+1$ 에서의 새로운 sequence를 $\sigma^{l+1} = \sigma^l \parallel a_k/o_k$ 라고 하면, 이에 대해 생성된 노드 $\langle v_k^{\sigma^{l+1}}, P^{\sigma^{l+1}}, ES^{\sigma^{l+1}} \rangle^j$ ($j=1,\dots,n$)에 대해서 $ES^{\sigma^l} \subseteq ES^{\sigma^{l+1}}$ 이 성립한다.

(증명)

i) $l=1$ 일때

σ^1 은 Null sequence, $ES^{\sigma^1} = \emptyset$, 그리고 $ES^{\sigma^2} = \emptyset$ 또는 nonempty이다. 따라서 $ES^{\sigma^1} \subseteq ES^{\sigma^2}$ 가 성립한다.

ii) $l=L$ 일때

정의에 의해서 ES^{σ^1} 은 input/output sequence σ^1 에 의해서 $v_k^{\sigma^1}$ 로 transition을 하는 상태들의 set이고, 알고리즘에 의해서 임의의 j 에 대한 a_j/o_j 에 의해서 $P^{\sigma^{l+1}}$ 이 생성될 때 만일 $v_k^{\sigma^{l+1}} \in P^{\sigma^{l+1}}$ 이면 $ES^{\sigma^{l+1}} = ES^{\sigma^1} \cup \{v_k\} \cup \{v_i \in V-\{v_i\} \text{ s.t. } (v_k, v_k^{\sigma^{l+1}}; \sigma^{l+1})\}$ 이므로 $ES^{\sigma^1} \subseteq ES^{\sigma^{l+1}}$ 이다. 그런데, 만일 $v_k^{\sigma^{l+1}} \notin P^{\sigma^{l+1}}$ 이면, $ES^{\sigma^{l+1}} = ES^{\sigma^1}$ 이므로 $ES^{\sigma^1} \subseteq ES^{\sigma^{l+1}}$ 이 성립한다.

따라서, i) ii)로부터 $ES^{\sigma^l} \subseteq ES^{\sigma^{l+1}}$ 이다.

이때 $v_k^{\sigma^l}$ 은 v_k 에서 input/output sequence σ^l 에 의해 transition된 상태를 나타내고 $(v_k, v_k^{\sigma^l}, \sigma^l)$ 로 표현된다. 또한 set ES 는 Chun등이 제안한 알고리즘의 set E 와 동일하다. $\sigma^1 \parallel a_k/o_k$ 는 sequence σ^1 과 a_k/o_k 의 연속을 의미한다.

(property 2)

vertex v_k 에 대한 $PUIO(v_k)$ 를 구할 때, 레벨 l 에서의 노드 $\langle v_k^{\sigma^l}, P^{\sigma^l}, ES^{\sigma^l} \rangle >^m$ 의 σ^l 이 $PUIO(v_k)$ 가 아니고, $v_k^{\sigma^l}$ 에서의 n 개의 input/output $a_j/o_j(j=1,\dots,n)$ 에 대한 레벨 $l+1$ 에서의 새로운 sequence $\sigma^{l+1} = \sigma^l \parallel a_j/o_j(j=1,\dots,n)$ 에 의해 생성된 노드 중에서 $j=t$ 일 때의 노드 $\langle v_k^{\sigma^{l+1}}, P^{\sigma^{l+1}}, ES^{\sigma^{l+1}} \rangle^t$ 의 σ^{l+1} 이 $PUIO(v_k)$ 이고, $j+t$ 인 노드 $\langle v_k^{\sigma^{l+1}}, P^{\sigma^{l+1}}, ES^{\sigma^{l+1}} \rangle^j$ 에 대해 $ES_j^{\sigma^{l+1}} \subseteq ES_j^{\sigma^{l+1}}$ 이면, $\langle v_k^{\sigma^{l+1}}, P^{\sigma^{l+1}}, ES^{\sigma^{l+1}} \rangle^j$ ($j+t$), 에 대한 레벨 $L>l+1$ 에서의 $PUIO(v_k)$ 인 $<$

$v_k^{\sigma L}, P^{\sigma L}, ES^{\sigma L} >^s (\chi=1, \dots, m)$, 에 대해서 $ES_x^{\sigma L} \supseteq ES_j^{\sigma L+1}$ 이 성립한다. 이때, $ES_j^{\sigma L+1}$ 은 노드 $< v_k^{\sigma L+1}, P^{\sigma L+1}, ES^{\sigma L+1} >^l$ 의 exclusion set을 나타낸다.

(증명)

(property 1)에 의해서 $ES_j^{\sigma L+1} \subseteq ES_x^{\sigma L}$ ($j \neq x, L > L+1$)이고, 조건에서 $ES_i^{\sigma L+1} \subseteq ES_j^{\sigma L+1}$ 이므로 $ES_j^{\sigma L+1} \subseteq ES_x^{\sigma L}$ 이 성립한다.

한 상태에 대한 i 번째 PUIO sequence를 PUIO _{i} 라 하고, 이 sequence들 중에서 상태 식별 기능을 가지면서 test-case의 길이를 가장 짧게 하기 위해 필요한 최적의 PUIO sequence들의 index set을 Q라고 하면, set Q는 PUIO sequence PUIO _{i} 와 이에 대응되는 inclusion set IS _{i} , 그리고 $\cup_{i \in R} IS_i = V$ 인 임의의 PUIO sequence index set R에 대해서 $\cup_{i \in Q} IS_i = V$ 이고 $\sum_{i \in Q} |PUIO_i| \leq \sum_{i \in R} |PUIO_i|$ 를 만족해야 한다. 여기서 $|PUIO_i|$ 는 PUIO _{i} 에 포함된 input/output 쌍의 갯수를 나타내고, PUIO sequence를 반복 적용하기 위해서 필요한 test-case의 중간 상태에서 초기 상태로의 transition은 reset input(r_i)에 의해서 이루어진다고 가정한다. 따라서 다음과 같은 property가 성립된다.

(Property 3)

한 상태에 대한 레벨 l 에서의 PUIO sequence에 대응되는 exclusion set을 $ES_k^{\sigma l} (k=1, \dots, n)$ 라고 할때, $ES_j^{\sigma l} \subseteq ES_i^{\sigma l} (L \geq l)$ 이면, 최적의 PUIO sequence index set Q에 대해서 $j \notin Q$ 인 Q가 존재한다.

(증명)

만일 최적의 PUIO sequence index set Q' 에 대해 $j \in Q'$ 라고 가정하고, 이때의 test-case의 길이를 $S(Q') = \sum_{k \in Q'} |PUIO_k|$ 라고 하자. 또한 $Q = Q' - \{j\} \cup \{i\}$ 인 Q를 가정하면 $ES_j^{\sigma l} \subseteq ES_i^{\sigma l} (L \geq l)$ 에서 $IS_i^{\sigma l} \supseteq IS_j^{\sigma l}$ 이고, $|S(Q)| = \sum_{k \in Q} |PUIO_k| < S(Q')$ 와, $\cup_{k \in Q} IS_k = V$ 를 만족한다. 따라서 $j \notin Q$ 인 최적의 PUIO sequence set가 존재한다.

앞에서 언급한 3가지 property를 Chun등이 제시한 PUIO sequence 생성 알고리즘에 적용하면 모든 가능한 PUIO sequence를 구하지 않으면서 최적의 PUIO sequence set을 구하기 위해 필요한 최소한의 PUIO sequence set을 구할 수가 있다. 이러한 PUIO sequence set Z_m 을 구하는 알고리즘은 다음과 같다.

Transition $(v_i, v_j; a_k/o_k)$ 의 결과 상태 v_j 가 UIO가 존재하지 않는다고 가정하고, v_j 에 대한 PUIO sequence set을 $< PUIO sequence, exclusion set >$ 이라고 하자.

- (1) $\lambda = \text{Null sequence}$;
- (2) 노드 $g_j^{\lambda} = \langle v_j, V - \{v_j\}, \phi \rangle$ 를 OPEN과 VISITED 리스트에 삽입;
- (3) TEMP 리스트 = ϕ ;
- (4) PUIO sequence set $Z_m = \phi, \sigma = \lambda$;
- (5) while(OPEN $\neq \phi$) do
- (6) 노드 $g_j^{\sigma} = \langle v_j^{\sigma}, P^{\sigma}, ES^{\sigma} \rangle$ 를 OPEN에서 선택;
- (7) for($(v_j^{\sigma}, v_k; i/o)$ 각각에 대해) do
- (8) $\sigma_{\text{new}} = \sigma \parallel i/o$;
- (9) $P^{\sigma_{\text{new}}} = \{v_l \mid v_l \in P^{\sigma} \text{ such that } (v_l, v_j; i/o)\}$;
- (10) if($v_k \in P^{\sigma_{\text{new}}}$)
- (11) then $ES^{\sigma_{\text{new}}} = ES^{\sigma} \cup \{v_l \mid \exists v_l \in V - \{v_j\} \text{ such that } (v_l, v_k; \sigma_{\text{new}})\}$,
- (12) $P^{\sigma_{\text{new}}} = P^{\sigma_{\text{new}}} - \{v_k\}$;
- (13) else $ES^{\sigma_{\text{new}}} = ES^{\sigma}$;
- (14) $v_j^{\sigma_{\text{new}}} = v_k$;
- (15) $g_j^{\sigma_{\text{new}}} = \langle v_j^{\sigma_{\text{new}}}, P^{\sigma_{\text{new}}}, ES^{\sigma_{\text{new}}} \rangle$;
- (16) if($P^{\sigma_{\text{new}}} = \phi$) and ($ES^{\sigma_{\text{new}}} \neq \phi$)
- (17) then Z_m 에 property 3을 적용하여 새로운 Z_m 을 구성;
- (18) else if($g_j^{\sigma_{\text{new}}} \in \text{VISITED}$) then (7)로;
- (19) else $g_j^{\sigma_{\text{new}}}$ 를 TEMP와 VISITED에 추가;
- (20) done for
- (21) • TEMP 리스트에 대해 property 2,3을 적용
- (22) i) Z_m 의 $ES_{\text{new}}^{\sigma l}$ 에 대해 $ES_{\text{new}}^{\sigma l} \subseteq ES_{\text{new}}^{\sigma l+1}$,

($L \geq l$)인 $g_j^{\sigma L}$ 제거

- (23) ii) 남아있는 $g_j^{\sigma new}$ 를 OPEN에 추가
- (24) • TEMP 리스트 = ϕ
- (25) done while
- (26) return PUIO sequence set Z_m

Minimal DFSM에서는 UIO sequence가 존재하지 않는 상태에 대한 PUIO sequence set에 대해서 ES의 intersection이 Null인, 즉 $UIS_i = V$ 인 k 개의 PUIO sequence가 반드시 존재한다는 것이 알려져 있다. 또한, 여기서 제시한 알고리즘에 의해서 구한 set Z_m 에 대해서도 DFSM에 대해서 minimal한 가정을 하였고, 최적의 PUIO sequence set에 대해 redundant한 PUIO sequence만을 제외하였기 때문에 동일한 결과를 보장할 수 있다. 따라서, 여기서 제시한 알고리즘에 의해서 구한 set Z_m 에 대해서 상태 v_i 를 식별할 수 있으면서 최소한의 test-case 길이를 갖도록 하는 PUIO sequence들이 존재한다.

IV. Example

그림 1의 DFSM에서 vertex v_4 는 UIO sequence를 갖지 않는다. 이때 Chun등이 제시한 알고리즘을 v_4 에

적용을 하면 그림 2와 같은 결과를 얻게 된다. 여기서 는 모두 12개의 노드가 생성되었고, 7개의 PUIO sequence들이 구해졌다.

그러나, 동일한 DFSM에 대해서 본 논문에서 제시한 알고리즘을 적용하면 불과 5개의 노드를 생성시키고 2개의 PUIO sequence를 구한다. 이에 대한 결과는 그림 3과 같다. 결국 앞에서 구한 7개의 PUIO sequence들 중에 5개는 최적의 PUIO sequence set을 구하는 데에는 redundant하다고 할 수 있다. 본 논문에서 제시된 알고리즘과 Chun등이 제시한 알고리즘을 랜덤하게 생성된 UIO sequence가 존재하지 않는 DFSM의 상태에 대해 적용하여 각각에 대해서 생성되는 PUIO sequence 수를 비교해 보았을 때 약 55% 정도 줄어든 PUIO sequence set을 구할 수 있었다.

V. PUIO sequence를 이용한 test-case 생성 방법

일반적으로 transition($v_i, v_j; i/o$)에 대해서 상태 v_i 가 UIO sequence를 갖지 않으며, n 개의 PUIO sequence가 존재한다고 가정하면 이를 이용한 test case 생성 방법은 다음의 2가지가 가능하다.

$$\{ Path(v_i) \parallel i/o \} \cup_k \{ PUIO_k \parallel TS(v_j^{PUIO_k}, v_j) \} \parallel PUIO_n \parallel ri, \{ 1 \leq k \leq n-1 \} \quad (1)$$

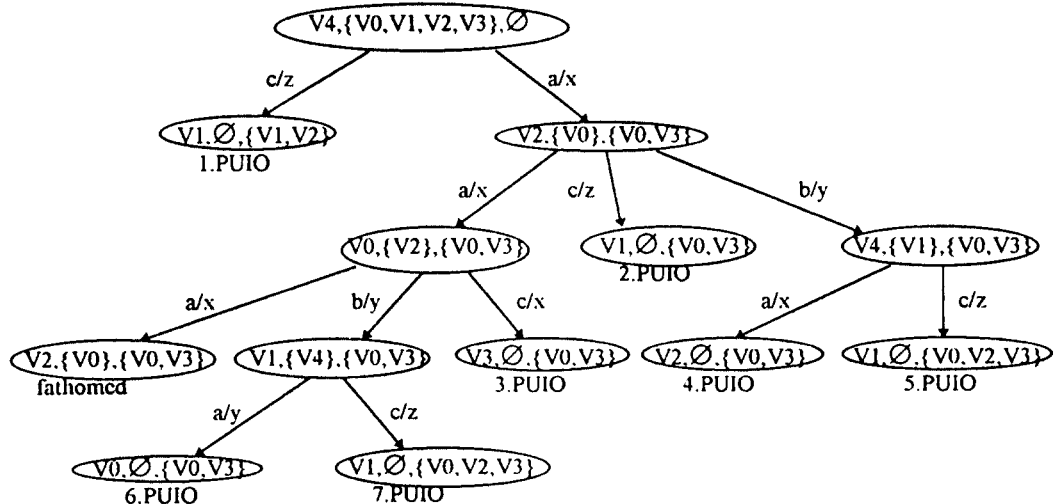


그림 2. PUIO sequence 생성의 search tree

Fig. 2. Search tree of PUIO sequence generation by Chun's algorithm

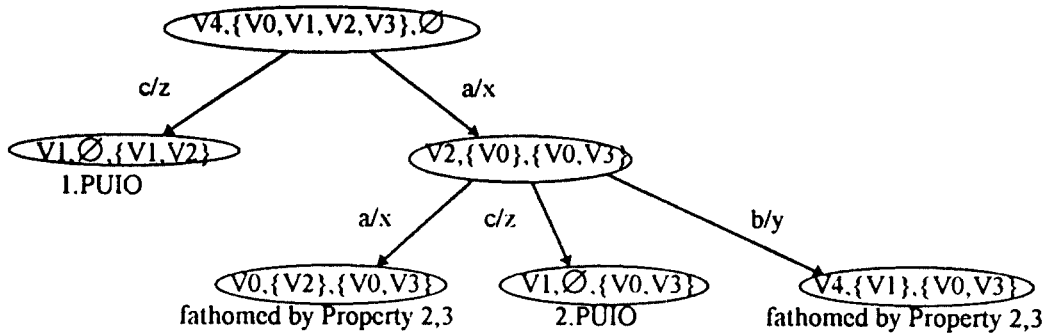


그림 3 개선된 알고리즘의 PUIO sequence 생성 search tree
 Fig 3: Search tree of PUIO sequence generation by modified algorithm

$$\cup_k \{ \text{Path}(v_i) \parallel i/o \parallel \text{PUIO}_k \parallel r_i \}, \{ 1 \leq k \leq n \} \quad (2)$$

여기서 $\text{Path}(v_i)$ 는 초기 상태에서부터 상태 v_i 까지의 transition을 나타내고 $\text{TS}(v_j^{PUIO_k}, v_i)$ 는 상태 v_j 에서 PUIO sequence PUIO_k 를 적용시킨 후의 결과 상태 $v_j^{PUIO_k}$ 에서 v_i 로의 transfer sequence를 나타낸다. 또한 r_i 는 초기상태로의 reset transition을 나타낸다. 그런데, (1)의 방법에 의해 생성된 test case는 그림 4에서처럼 transition $(v_i, v_j ; i/o)$ 가 $(v_i, v_k ; i/o)$ 구현된 IUT에 대해서 만일 PUIO_i 가 상태 v_j 와 v_k 에 대해 모두 존재하고 $\text{TS}(v_j^{PUIO_i}, v_i) = \text{TS}(v_k^{PUIO_i}, v_i)$ 라면, IUT에 존재하고 있는 이러한 구현오류를 발견해내지 못할 가능성도 있으므로 방법(2)에서 생성된 test case보다는 fault coverage가 다소 떨어진다. 또한 방법 (2)의 경우에서

도 PUIO sequence 대신 DS를 사용하는 방법도 가능하겠지만 대부분의 경우에서 하나의 PUIO sequence는 고려대상인 상태를 한 개 이상의 상태로 부터 구별할 수 있으므로 모든 다른 상태에 대한 각각의 DS를 사용하는 경우보다 길이가 짧은 test-case를 얻을 수 있게 된다. 따라서 방법 (2)에 의한 test-case 생성이 가장 합리적이라고 할 수 있다.

또한, $\text{Path}(v_i)$ 로서 초기 상태부터 v_i 까지의 최단거리 input/output sequence를 사용한다고 가정할 때, test case의 길이는 결국 test case에 사용된 PUIO sequence들의 길이의 합에 의해서 결정이 된다. 그러므로 앞에서 제시한 알고리즘에 의해서 구한 PUIO sequence set Z_m 으로 부터 상태 식별 기능을 가지면서 전체 test case의 길이를 최소화 시키는 최적의 PUIO sequence set Z 를 구하는 문제는 다음과 같이 formulation할 수 있다.

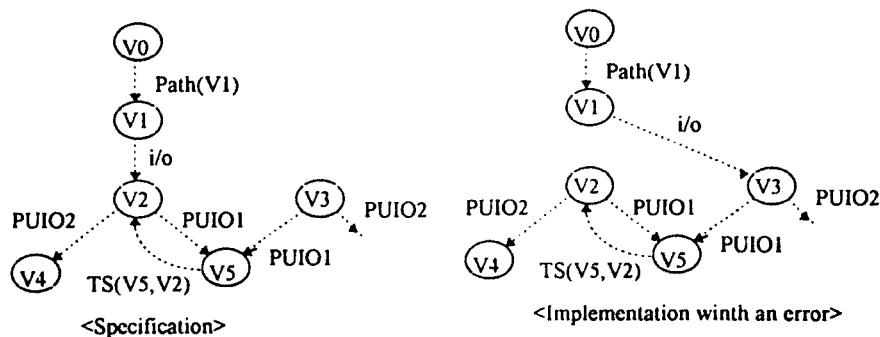


그림 4. 구현 오류가 있는 경우의 예
 Fig. 4. An Example with Implementation error.

$$\begin{aligned} \min \sum_{i=1}^k |PUIO_i| x_i \\ s. t. \cup_{i=1}^k IS(x_i) = V \\ x_i = 0 \text{ or } 1 \quad i = 1, \dots, k \end{aligned}$$

여기서 k 는 set Z_m 에 속하는 PUIO sequence들의 갯수를 나타내고, Z_m 에 속한 i 번째 PUIO sequence에 대해서는 PUIO와 IS _{i} 가 정의된다. 또한 x_i 는 PUIO _{i} 의 set Z 에 대한 포함 여부를 나타내는 0/1 변수이고, IS(x_i)는 $x_i=1$ 일때 IS(x_i)=IS _{i} 이고 $x_i = 0$ 일때 IS(x_i)= ϕ 의 값을 갖는 함수로 정의한다.

이렇게 formulation 된 최적의 PUIO sequence set을 구하는 문제는 NP complete 문제로 알려진 set cover 문제와 equivalent하고, 이에 대한 접근 방법으로 Branch and bound 알고리즘을 고려할 수가 있다. 이 알고리즘에서는 upper bound로서 UIS _{i} =V인 PUIO sequence 들에 대해서 $\sum |PUIO_i|$ 값을 정의하고 PUIO sequence들의 조합 중에서 UIS _{i} =V를 만족하는 새로운 PUIO sequence들의 set으로서 upper bound를 개선해 나가면 된다. 일반적으로 이 알고리즘은 대상 요소의 갯수가 많을 때는 complexity가 크지만, 여기서는 3장에서 제시한 property에 의해 축소된 PUIO sequence set을 구할 수가 있기 때문에 모든 가능한 경우를 고려하는 것 보다는 훨씬 효율적이라고 할 수 있다.

VI. 결론

지금까지 DFSM에 대한 프로토콜 적합성 시험에서 UIO sequence가 존재하지 않는 상태로의 transition에 대한 결과 상태를 확인하는 방법으로서 PUIO sequence를 이용하는 방법에 관해서 언급하였다. PUIO sequence는 한 상태를 DFSM의 일부 상태로 부터 식별해 줄 수 있는 input/output의 연속으로서 일반적으로 하나의 상태에 대해서 여러 개가 존재한다. 따라서 UIO sequence가 존재하지 않는 상태에 대해서 이러한 PUIO sequence들을 구하고, 이로부터 모든 상태를 식별할 수 있는 PUIO sequence 들을 이용하여 그 상태에 대한 하나의 test case를 구성해야 한다.

본 논문에서는 이를 위해서 Chun등이 제시한 PUIO sequence 알고리즘에 대해 몇 가지 Property를 제시하였고, 이것을 알고리즘에 적용함으로써 최적의 PUIO sequence를 생성하는데 필요한 축소된 PUIO sequence

set을 구하는 알고리즘을 제안하였다. 이 알고리즘을 적용한 결과 기존의 PUIO sequence 알고리즘보다 PUIO sequence수를 55% 정도 줄일 수가 있었고, 이를 적용하면 최적의 test case를 구할 때 효율성을 높일 수 있을 것이라고 생각된다. 또한 이에 대한 한가지 접근 방법으로서 Branch and bound 알고리즘을 고려할 수 있었다.

참고 문헌

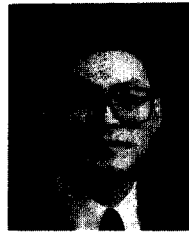
1. K.Sabnani and A.Dahbura, A Protocol Test Generation Procedure, *Computer Networks and ISDN Systems*, 15(4), pp285-297, September 1988.
2. A.V.Aho, A.T.Dahbura, D.Lee, and M.U.Uyar, An Optimization Technique for Protocol Conformance Test Generation Based on UIO Sequences and Rural Chinese Postman Tours, In S.Aggarwal and K.Sabnani, editors, *Protocol Specification, Testing, and Verification VIII*, pp75-86, Amsterdam, 1988, North Holland.
3. Y.N.Shen, F.Lombardo, and A.T.Dahbura, Protocol Conformance Testing Using Multiple UIO Sequences, In E.Brinksma, G.Scollo, and C.A.Vissers, editors, *Protocol Specification, Testing, and Verification IX*, pp131-143, Amsterdam, 1990, North Holland.
4. W.Chun and P.D.Amer, Improvement in UIO Sequence Generation and Partial UIO Sequence, *Protocol Specification, Testing, and Verification X*, pp245-260, 1992.
5. W.Y.L.Chan, S.T.Vuong, and M.R.Ito, An Improved Protocol Test Generation Procedure Based on UIO's, *SIGCOMM '89 Symposium: Communication Architecture and Protocols in Computer Comm.*, Review, 19(4), pp283-294, September 1989.
6. W.Y.L.Chan, S.T.Vuong, and M.R.Ito, On Test Sequence Generation for Protocols, *Protocol Specification, Testing, and Verification IX*, pp119-130, 1990.
7. T.S.Chow, Testing Software Design Modeled by Finite State Machine, *IEEE Transactions on Software Engineering*, 4(3), pp178-187, May 1978.
8. B.S.Bosik and M.U.Uyar, Finite State Machine

based formal methods in protocol conformance testing : From theory to implementation, *Computer Networks and ISDN Systems*, 22(1), pp7-33, 1991.



崔 珍 領 (Jin Young Choi) 정희원
1968년 8월 3일생
1991년 2월 : 한양대학교 산업공학과 (학사)
1993년 2월 : 한국과학기술원 산업공학과(석사)
1993년 2월 ~ 현재 : 한국전자통신

연구소 광대역 통신방식 연구실 연구원



洪 范 基 (Beom Kee Hong) 정희원
1958년 7월 18일생
1982년 2월 : 홍익대학교 전자계산학과(학사)
1984년 8월 : 홍익대학교 전자계산학과(석사)
1982년 3월 ~ 현재 : 한국전자통신

연구소 광대역 통신방식 연구실 선임연구원