SATアルゴリズムを利用した FSMプロトコルに対する試験系列生成の一手法

森 亮憲† 舩曳 信生‡ 東野 輝夫*

† 大阪大学大学院基礎工学研究科

‡ 岡山大学工学部

* 大阪大学大学院情報科学研究科

本稿では，有限状態機械で記述された通信プロトコルに対して，SATを利用して状態確認を行うための試験系列を生成する手法を提案する．状態確認にはUIO系列を利用する．SATとは論理式の充足可能性を判定する問題である．提案手法では，論理式を用いて通信プロトコルの動作や試験系列が満たすべき条件を記述することから，各状態に対して複数のUIO系列が存在する場合や，UIO系列の重複を考慮した場合などに対応して試験系列を生成することができる．提案手法を実装し，DHCP（Dynamic Host Configuration Protocol）に適用して試験系列が生成できることを確認した．

A Method to Generate Test Sequences for FSM Protocols using SAT Algorithm

Takanori Mori†, Nobuo Funabiki‡ and Teruo Higashino*

†Graduate School of Engineering Science, Osaka University

‡Faculty of Engineering, Okayama University

*Graduate School of Information Science and Technology, Osaka University

In this paper, we propose a test sequence generation method for FSM protocols using SAT algorithm. UIO sequences are used for identifying states. SAT is a problem which check satisfiablity of logical formulas. In our method, protocol behaviors and conditions which must be satisfied by test sequences are described as logical formulas. Hence our method can be applied to cases that each state has multiple UIO sequences and/or considering sequences overlapping. We developed a system, and apply it to DHCP (Dynamic Host Configuration Protocol) and generated test sequences.

1 まえがき

SAT問題（satisfiability problem）はNP完全問題[1]として知られている組合せ最適化問題である．SAT問題では，与えられた和積形論理式fに対して，充足可能性の判定や充足可能な場合にf中の変数に対する真理値割当てをできるだけ高速に算出することが要求される．近年，SATアルゴリズムの性能が改善され，Electronic Design Automationの分野などで実用的な問題に適用されている[2]．

本稿では，有限状態機械（FSM：Finite State Machine）でモデル化される通信プロトコルに対してSATを用いた適合性試験系列生成手法を提案する．試験を行う場合には，FSMの状態を識別する入出力系列が使われる．このような系列としてDS（Distinguish Sequence）[3]やUIO系列（Unique Input/Output Sequence）[4, 5]などがある．これらの系列は，FSMのある1つの状態でのみ実行できる入出力系列である．一般に任意のFSMの各状態についてDSやUIO系列が存在するとは限らない．しかしUIO系列については，実際に使われるプロトコルを表したFSMで，ほとんどの場合に存在することが知られている[4, 5, 6]．また，一般に各状態に対して複数のUIO系列が存在する場合もある．本稿ではUIO系列を利用して試験系列を生成する．

FSMプロトコルに対して試験を行う場合は，プロトコル仕様に記述されている状態および状態遷移が実装に存在するのか，また正しく実装されているのかを確認する．ある状態sを確認するには，FSMを初期状態から状態sへ遷移させる入出力系列（先行系列）のあとに，状態sに対するUIO系列を連接した系列を実装に対して適用すればよい．また，状態sから状態s'への状態遷移tを確認するには，

FSM を初期状態から状態 s へ遷移させる先行系列のあとに，状態遷移 t に対応する入出力と状態 s' に対する UIO 系列を連接した系列（部分系列）を実装に対して適用すればよい．すべての状態および状態遷移に対して上記の入出力系列（部分試験系列：先行系列 +UIO 系列または先行系列 + 部分系列）を生成，連接することで試験系列を生成する．

ある部分試験系列の先行系列として，別の部分試験系列を使うことによって試験系列を短くすることができる．また，各状態に対して複数の UIO 系列が存在する場合があるため，UIO 系列の選択によって試験系列の長さが変化する．さらに，UIO 系列や部分系列が共通部分を持っている場合，それらを重複させることでより短い試験系列を生成することができる．一般に，系列の重複を考慮しない場合に最適な試験系列を生成する問題は NP 完全問題であることが知られている．また，系列の重複を考慮した場合も NP 完全問題であることが知られている [7].

文献 [4] では，FSM の各状態に対して UIO 系列を生成するアルゴリズムと状態および状態遷移を確認する試験系列生成手法が提案されている．また，文献 [8] では，FSM の各状態に自己ループがあるような場合，または各状態から初期状態への遷移（リセット遷移）がある場合に，状態遷移を確認する最適な試験系列を Rural Chinese Postman Tours を利用して効率良く解くアルゴリズムが提案されている．文献 [8] の提案手法は各状態に対して UIO 系列を 1 つ選び，部分系列の重複を考慮しない試験系列を生成する．文献 [9] および文献 [10] では，複数の UIO 系列を考慮して状態遷移を確認する試験系列を生成する手法が提案されている．これら文献の提案手法でも FSM の構造に条件を設けている．また，部分系列の重複は考慮していない．一方，文献 [11] では，ヒューリスティックスを導入し，部分系列の重複を考慮して状態遷移を確認する試験系列の生成手法が提案されている．

本稿では，SAT アルゴリズムを用いて，複数の UIO 系列および UIO 系列の重複を考慮して，状態確認を行う最適な試験系列を生成する手法を提案する．提案手法では，FSM の動作や試験系列に対する条件を論理式で記述する．提案手法の有用性を評価するため，提案手法に基づいて試験系列生成プログラムを実装し，DHCP（Dynamic Host Configuration Protocol）[12] に適用した．

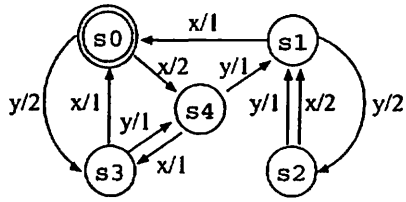

図 1: プロトコル機械の例

以降，2. では適合性試験および SAT 問題について述べる．3. では SAT を用いた試験系列生成手法を説明し，4. では DHCP に対して提案手法を適用した結果について述べる．

2 準備

2.1 プロトコル機械

プロトコル機械は Mealy 型決定性有限状態機械（FSM : Finite State Machine）とし，5 字組 (S, X, Y, H, s_0) で定義する．ここで、S, X, Y はそれぞれ状態の有限集合，入力記号の有限集合，出力記号の有限集合を表す．また，H は状態遷移 (u, v, x, y) の有限集合を表す．ただし，$u, v \in S$ は始状態，終状態，$x \in X$ は入力，$y \in Y$ は出力を表す．s_0 は初期状態を表す．図 1 は，FSM で記述したプロトコル機械の例である．

2.2 適合性試験

本稿では，プロトコル機械の実装（IUT : Implementation Under Test）に対して，プロトコル機械の各状態が正しく実装されているかを確認する試験系列を生成する．IUT に対してある状態 s を確認したい場合は，プロトコル機械が初期状態から状態 s へ遷移する入力系列（先行系列）をプロトコル仕様から生成し，その系列を IUT に与えた後，IUT が状態 s であるかどうかを確認すればよい．このため，FSM の各状態を識別する必要がある．

FSM の各状態を識別する系列として UIO 系列 (Unique Input/Output sequence) がある [4, 5]. FSM M の状態 s に対する UIO 系列とは，FSM M において状態 s 以外のどの状態からも実行することができない入出力系列のことである．ここで，入出力系列 $\alpha = (i_1/o_1)(i_2/o_2)\cdots(i_m/o_m)$ （i_j:入力，o_j:出力）が FSM M の状態 s から実行できるとは，FSM M が状態 s であるときに，α の入力系列 $i_1 i_2 \cdots i_m$ を M に与えて得られる出力系列が α の出力系列 $o_1 o_2 \cdots o_m$ と一致することをいう．

表 1: 図 1 のプロトコル機械に対する UIO 系列

状態 (s_i)	UIO 系列
s_0	(y/2)(x/1)
	(x/2)(y/1)
s_1	(y/2)(x/2)
s_2	(x/2)(y/2)
s_3	(y/1)(y/1)
s_4	(x/1)(y/1)

一般に任意の FSM の各状態に対して UIO 系列が存在するとは限らない．ただし，実際の通信プロトコルでは，ほとんどの場合に各状態に対して UIO 系列が存在することが知られている [4, 5, 6]．また，各状態に対する UIO 系列は複数存在する場合がある．図 1 の FSM の各状態に対する UIO 系列は表 1 のようになる．

各状態を識別するためには，各状態について 1 つの UIO 系列があれば十分であるが，各状態に対して複数の UIO 系列を用意して，それらの UIO 系列から適切な系列を選ぶことによって，試験系列がより短くなる場合がある [9, 10]．また，複数の UIO 系列が重複部分を持つ場合，それらの UIO 系列を重複させた系列を生成し，その系列を利用することでより短い試験系列を生成することができる場合がある [11]．

2.3 SAT 問題

SAT 問題とは，和積形論理式 f に対して充足可能性を判定し，充足可能であれば f 中の変数に対する真理値割当てを算出する問題である．論理式 f は複数個の節の論理積で構成され，それぞれの節は変数 x または変数の否定 $\neg x$ の論理和で構成される．

SAT 問題に対して，多数のアルゴリズムが研究・開発されている [13, 14, 15, 16]．これらのアルゴリズムは Complete 型と Incomplete 型に分類される．Complete 型アルゴリズムは，論理式に対して充足可能となるすべての真理値割当てを算出することができる．しかしこの型の多くのアルゴリズムは解空間のすべてを探索するために，大規模な問題を解くことが難しい．この型のアルゴリズムとして GRASP[13]，SATZ[14] などがある．一方 Incomplete 型アルゴリズムは，論理式が充足可能な場合に充足可能となる 1 つの真理値割当てを算出する．この型のアルゴリズムは解空間の探索範囲を一部分に限定することで，高速に解を算出する．このため Complete 型では解くことが難しい大規模問題を解くことができる．この型のアルゴリズムとして MIPS_SAT[15]，DLM[16] などがある．

```
1  H' = ∅,   I' = ∅
2  for s_ij ∈ S' {
3     if i == 0 then I' = I' ∪ {s_ij}
4     for s ∈ S ∪ S' {
5        if (s, s_i, x, y) ∈ H ∪ H' then
6           H' = H' ∪ (s, s_ij, x, y)
7        if (s_j, s, x, y) ∈ H ∪ H' then
8           H' = H' ∪ (s_ij, s, x, y)
9     }
10 }
```

図 2: H' および I' を構成するアルゴリズム

プロトコル機械の各状態に対して UIO 系列が存在する場合，そのプトロコル機械に対する試験系列は必ず存在するので，本稿では Incomplete 型アルゴリズムを使用する．DIMACS ベンチマーク[*1] [17] の結果から MIPS_SAT を使うことにした．以降，SAT アルゴリズムの実装を SAT solver と呼ぶ．

3 試験系列生成問題から SAT 問題への変換アルゴリズム

提案手法では，プロトコル仕様を表す FSM M と M の各状態に対する UIO 系列から和積形論理式を生成し，その論理式を SAT solver を用いて解くことで試験系列を生成する．提案手法では，各状態に対して論理変数を割当てる．まず，UIO 系列に対応する状態とその状態に関連する状態遷移を M に追加して M' を生成する．その後 M' の動作および試験系列に対する条件を表した論理式を構成する．

3.1 FSM の変更

FSM $M = (S, X, Y, H, s_0)$ に UIO 系列に対応する状態および関連する状態遷移を追加して FSM $M' = (S \cup S', X, Y, H \cup H', \{s_0\} \cup I')$ を生成する．

S' は新しく追加する状態の集合で，UIO 系列 1 つに対して状態を 1 つ追加する．状態 s_i に対する UIO 系列で，UIO 系列を状態 s_i で実行すると状態 s_j へ遷移するとき，状態 s_{ij} を追加するとする．例えば，図 1 の FSM と状態 s_0 に対する UIO 系列 $(y/2)(x/1)$ に対して状態 s_{00} を用意する．この状態は $s_0 \xrightarrow{(y/2)} s_3 \xrightarrow{(x/1)} s_0$ を表している．また，UIO 系列 $(y/2)(y/1)$ に対しては状態 s_{01} を用意する．

[*1] SAT アルゴリズムの性能を評価するベンチマークテスト

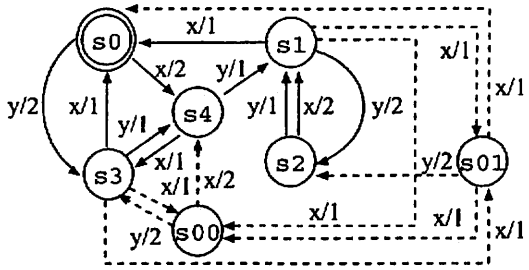

図 3: 状態と状態遷移を追加した FSM

H' は，$s' \in S'$ に関係する遷移を表し，$I' \subseteq S'$ は S' の要素のうち初期状態となり得る状態の集合を表している．図 2 のようにして H', I' を構成する．状態 s_{ij} への遷移は状態 s_i への遷移，状態 s_{ij} からの遷移は状態 s_j からの遷移と同じものにする．

図 1 に状態 s_{00}, s_{01} と関係する状態遷移を加えると図 3 のようになる．図中の破線矢印が追加した状態遷移である．また，状態 s_{00}, s_{01} は I' の要素である．

次に，UIO 系列が重複部分を持つ場合，重複させた系列に対する状態を追加する．例えば，表 1 の状態 s_3 に対する UIO 系列 $(y/1)(y/1)$ と状態 s_4 に対する UIO 系列 $(x/1)(y/1)$ を重複させて系列 $(x/1)(y/1)(y/1)$ を生成し，これに対応する状態として状態 s_{41} を用意する．関係する状態遷移は前と同じ方法で生成する．

この構成法では FSM の構造によって追加した状態間の遷移が生成されないことがある．図 4 (i) の FSM が仕様であり，遷移 a が状態 s_1 の UIO 系列，遷移 b が状態 s_2 の UIO 系列であるとする．先に述べたアルゴリズムで，仕様に状態と状態遷移を追加すると，図 4 (ii) となる．この FSM では，追加した状態 (UIO 系列) を連続して通過することができず，最短の試験系列が生成できない．そこで，図 4 (iii) のように，自己ループを持たない状態に対してループ遷移を追加する (図中の遷移 c, d)．ループを追加しておくことで，図 4 (iv) のような FSM が生成される．生成した試験系列に追加したループ遷移が含まれているときは，これを削除する．

3.2 論理式の構成

3.2.1 論理式を構成する変数

提案手法では，FSM は時刻 0 から動作を開始し，1 単位時間ごとに遷移を 1 つ実行するものとする．論理変数 $X[t][i]$ を用いて和積形論理式を作成する．$X[t][i]$ が真の場合，時刻 t で FSM が状態 s_i であることを表す．例えば，図 1 の FSM に対して 3 単位時間の動作を考える．このときの動作を表すには $X[t][i]$ $(1 \le t \le 3, 0 \le i \le 4)$ の変数が必要である．FSM が状態 s_0, s_4, s_1 の順に遷移することを表すには，$X[1][0], X[2][4], X[3][1]$ を真，これら以外の変数を偽とすればよい．

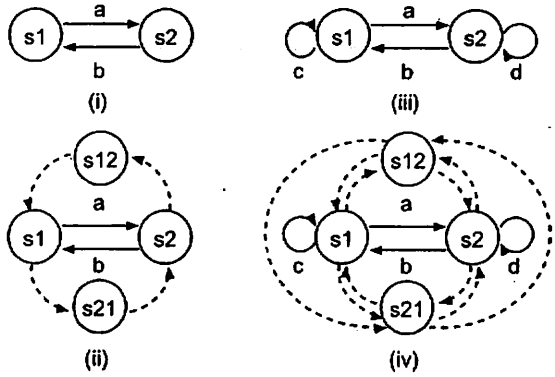

図 4: 図 2 のアルゴリズムで得られる FSM の例

3.2.2 条件を表す論理式

前節で導入した論理変数を用いて，FSM M' の動作を表す論理式，および試験系列に対する条件を表す論理式を構成する．提案手法では論理式を構成するために最大試験系列長 (時刻の最大値) T を与える．

初期状態に対する条件 M' の初期状態は $s_0 \cup I'$ である．そこで，時刻 0 では，M' は $s_0 \cup I'$ のいずれか 1 つの状態であることを表す論理式を構成する．図 3 の FSM では状態 s_0, s_{00}, s_{01} がこの集合に含まれるので以下のようになる．

$$
\begin{aligned}
&(X[0][0] \vee X[0][00] \vee X[0][01]) \\
&\wedge (\neg X[0][0] \vee \neg X[0][00]) \\
&\wedge (\neg X[0][00] \vee \neg X[0][01]) \\
&\wedge (\neg X[0][0] \vee \neg X[0][01]) \qquad (1)
\end{aligned}
$$

各時刻での状態に対する条件 時刻 0 以外の各時刻において，FSM は $S \cup S'$ のうち 1 つの状態であることを表す論理式を構成する．図 3 の FSM では，以下のようになる．

$$
\begin{aligned}
&(X[t][0] \vee X[t][1] \cdots \vee X[t][44]) \\
&\wedge (\neg X[t][0] \vee \neg X[t][1]) \cdots \\
&\wedge (\neg X[t][1] \vee \neg X[t][44]) \quad (1 \le t \le T) \qquad (2)
\end{aligned}
$$

FSM には，初期状態から一遷移 (一単位時間) で到達できない状態がある．そこで，これらの状態を考慮することで，論理式を小さくすることができる．

状態遷移を表す条件　ある時刻 t での状態が決定すれば，次の時刻に遷移することができる状態が決まる．このことから，状態遷移を“状態 → 遷移可能なすべての状態の論理和”で表現することができる．図 3 の状態 s_4 からは，状態 s_1，または s_3 へ遷移することができるので，以下のようになる．

$$X[t][4] \rightarrow (X[t+1][1] \vee X[t+1][3]) \quad (0 \leq t \leq T-1) \qquad (3)$$

UIO 系列を通過する条件　0 から T のいずれかの時刻で，各 UIO 系列（状態 $s' \in S'$）を通過することを表す論理式を構成する．図 3 の状態 s_1 の UIO 系列に対して s_{22} を追加したすると次のようになる．

$$(X[0][22] \vee X[1][22] \vee \cdots \vee X[T][22]) \qquad (4)$$

ある状態に対して複数の UIO 系列がある場合は，それらのいずれかを通過することを表す論理式を構成する．状態 s_0 の UIO 系列に対して状態 s_{00}, s_{01} を追加したとすると次のようになる．

$$(X[0][00] \vee X[1][00] \vee \cdots \vee X[T][00]) \vee (X[0][01] \vee X[1][01] \vee \cdots \vee X[T][01]) \cdots (4')$$

また，重複させた UIO 系列については，重複させる前の UIO 系列または重複させた後の UIO 系列のいずれかを通ることを表す論理式を構成する．状態 s_3 の UIO 系列に対して状態 s_{31}，状態 s_4 の UIO 系列に対して s_{44}，重複させた UIO 系列に対して s_{41} をそれぞれ追加したとすると次のようになる．

$$\{(X[0][31] \vee \cdots \vee X[T][31]) \vee (X[0][41] \vee \cdots \vee X[T][41])\} \wedge \{(X[0][44] \vee \cdots \vee X[T][44]) \vee (X[0][41] \vee \cdots \vee X[T][41])\} \qquad (4'')$$

以上の条件を表す論理式は，それぞれ簡単に和積形に整理することができる．和積形に整理した論理式の論理積をとることによって得られる論理式が，試験系列生成問題に対応する和積形論理式である．この論理式をを SAT solver に適用して解を求める．解が得られた場合，各時刻での FSM の状態が得られる．UIO 系列に対応する状態はその状態が表す状態と状態遷移に展開した後，遷移だけを取り出すことで，試験系列が得られる．

提案手法では，問題の入力として最大試験系列長 T を与える．試験系列では UIO 系列に対応する状態を必ず通過する必要があるので，T の値として UIO 系列数よりも大きな値を与える必要がある．値が小さいことによって解が得られない場合は，より大きな値を与えて論理式を構成する．

表 2: 図 1 の FSM に対する試験系列生成結果

	変数の数	節の数	試験系列長
実験 1	24	77	15
実験 2	41	210	15
実験 3	21	72	13
実験 4	33	171	13

4　試験系列生成システムを用いた実験

提案手法に基づいて，試験系列生成システムを実装し，図 1 の FSM と DHCP（Dynamic Host Configration Protocol）[12] に対して適用した．

4.1　例プロトコルに対する適用結果

図 1 の FSM で表されるプロトコルに対して，試験系列を生成した．試験系列を生成するための論理式に含まれる変数の数，節の数および生成された試験系列の長さを表 2 に示す．実験 1 は，各状態の UIO 系列数が 1 で重複を考慮しない場合，実験 2 は，各状態の UIO 系列数が複数で重複を考慮しない場合，実験 3 は，各状態の UIO 系列数が 1 で重複を考慮する場合，実験 4 は，各状態の UIO 系列数が複数で重複を考慮する場合である．

UIO 系列の重複を考慮した方が，より短い試験系列が生成されている．また，この例では状態 s_0 に対して 2 つの UIO 系列を用意しても生成される試験系列の長さは変わらなかった．

4.2　DHCP に対する適用結果

DHCP のクライアント側プロトコルに対して試験系列を生成した．DHCP とは，ネットワーク上のクライアントにサーバを利用して IP アドレスを動的に割り当てるプロトコルである．DHCP を表す FSM の状態数は 14，状態遷移数は 77 である．

DHCP に対して各状態に対する UIO 系列を作成した．UIO 系列が 1 つの状態が 10，UIO 系列が 2 つの状態が 4 であった．試験系列生成問題を論理式に変換し，SAT solver に適用して試験系列を求めた．得られた試験系列の長さは 21 であった．生成した論理式の変数の数，節の数，生成時間および SAT solver の実行時間は，表 3 の 1 行目のようになった．各時間は 5 回の平均（秒）で，実行環境は CPU PentiumIII 700MHz，メモリ 1GB である．

また，実験のために各状態に対して複数の系列を割り当て，それのいずれかを含む系列を生成した．結果を，表 3 の 2 から 4 行目に示す．2 行目は，各状態に 1 本，3 行目は 2 本，4 行目は 3 本の系列を割当てた結果である．

表 3: DHCP に対する試験系列生成結果

UIO 系列数	変数の数	節の数	論理式 生成時間（秒）	SAT Solver 実行時間（秒）
18	422	6668	0.02	0.37
14	369	5137	0.01	0.23
28	482	9960	0.02	0.70
42	653	17996	0.03	2.40

5 まとめ

本稿では，FSM 通信プロトコルに対して，状態確認を行うための試験系列を SAT を用いて生成する手法を提案した．提案手法では，各状態に複数の UIO 系列がある場合，および UIO 系列の重複を考慮して最適な試験系列を生成する．DHCP に対して提案手法を適用し試験系列が生成できることを確認した．また，提案手法では仕様に追加する状態を，状態遷移 t と t の遷移先状態に対する UIO 系列を表す状態とすることで，状態遷移を確認する試験系列を生成することができる．今後の課題として，拡張有限状態機械でモデル化される通信プロトコルへの適用を考えている．

参考文献

[1] M. R. Garey and D. S. Johnson, "Computers and Intractability: a Guide to the Theory of NP - Completeness", Freeman, (1979).

[2] J. P. Marques-Silva and K. A. Sakalah, "Boolean Satisfiability in Electronic Design Automation", Proc. 37th Design Automation Conference, pp.675–680, (2000).

[3] G. Gonenc, "A Method for the Design of Fault-detection Experiments", IEEE Trans. Comput., vol.C-19, no.6, pp.551–558, (1970).

[4] K. K. Sabnani and A. T. Dahbura, "A Protocol Test Generation Procedure", Comput. Networks and ISDN Syst., vol.15, no.4, pp.285–297, (1988).

[5] W. Y. L. Chan, S. T. Vuong and M. R. Ito, "An Improved Protocol Test Generation Procedure based on UIOs", Proc. ACM SIGCOMM'89, pp.283–294, (1989).

[6] B. S. Bosik and M. U. Uyer, "Finite State Machine Based Formal Methods in Protocol Conformance Testing: from Theory to Implementation", Comput. Networks and ISDN Syst., vol.22, pp.7–33, (1991).

[7] S. Boyd and H. Ural, "On the complexity of generating optimal test sequences", IEEE Trans. on Softw. Eng., vol.17, no.9, pp.976–978, (1991).

[8] 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", IEEE Trans. Commun., vol.39, no.11, pp.1604–1615, (1991).

[9] Y. N. Shen, F. Lombardi and A. T. Dahbura, "Protocol Conformance Testing using Multiple UIO Sequences", IEEE Trans. Commun., vol.40, no.8, pp.1282–1287, (1992).

[10] M. U. Uyar, "Dual-state Augmentation for Minimizing Conformance Test Costs", Comput. Networks and ISDN Syst., vol.30, no.14, pp.1277–1294, (1998).

[11] B. Yang and H. Ural, "Protocol Conformance Test Generation using Multiple UIO Sequences with overlapping", Proc. ACM SIGCOMM'90, pp.118–125, (1990).

[12] R. Droms, "Dynamic Host Configuration Protocol", RFC2131, Bucknell University, (1997).

[13] J. P. Marques-Silva and K. A. Sakallah, "GRASP : a Search Algorithm for Propositional Satisfiability", IEEE Trans. Comput., vol.48, no.5, pp.506–521, (1999).

[14] C. M. Li and Anbulagan, "Look-Ahead Versus Look-Back for Satisfiability Problems", Proc. 3rd Int'l Conf. Principles and Practice of Constraint Programming–CP97, pp.342–356, (1997).

[15] N. Funabiki, T. Yokohira, T. Nakanishi, S. Tajima and T. Higashino, "A Minimal-State Processing Search Algorithm for Satisfiability Problem", IEEE Proc. Syst. Man. Cybern. Conf., pp.2769–2774, (2001).

[16] Y.Shang and B.W.Wah, "A Discrete Lagrangian-Based Global-Search Method for Solving Satisfiability Problem", J. Global Optimization, vol.12, pp.61–99, (1998).

[17] DIMACS SAT benchmarks,ftp://dimacs.rutgers.edu/pub/challenge/sat/benchmarks/cnf/.