ICOT Technical Report: TR-132

TR-132

On QFTL and the Refutation Procedure on ω-Graphs

by

A. Fusaoka and K. Takahashi
(Mitsubishi Electric Corp.)

August, 1985

©1985, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

ON QFTL AND THE REFUTATION PROCEDURE ON ω- GRAPHS

Akira FUSAOKA   Kazuko TAKAHASHI

Central Research Laboratory
Mitsubishi Electric Corporation
8-1-1, Tsukaguchi-Honmachi
Amagasaki, Hyogo, 661, JAPAN

ABSTRACT

This paper describes a method for reasoning about a system described in Quantifier Free Temporal Logic (QFTL). Propositinal Temporal Logic is complete and the formulas are represeted in the form of ω- graph which is semantically equivalent to the formulas. We extend the concept to the formulas of QFTL and the mechanical theorem proving method is proposed. QFTL is a subsystem of Temporal Predicate Logic which contains free variables and not contain quantifiers Basic idea of a mechanical theorem proving method is to construct the ω- graph which is semantically equivalent to the formula, and check the emptiness of the graph. It is a refutation procedure. Failure of making such a model verifies the unsatisfiability of the formula. However, this procedure is incomplete , which is due to the incompleteness of QFTL.

CONTENTS

1. INTRODUCTION

Temporal Logic [MP] is an extension of first order logic to include a notion of time. It is a branch of *modal logic* [HC], the relation between worlds is considered as a temporal one. And it is an appropriate methodology to deal with the logical description and reasoning on time.

In the usual temporal framework, some temporal operators are used, whose intuitional meanings are as follows :
$\Box P$ (always P) : P is true in all future instants
$\Diamond P$ (eventually P) : P is true in some future instant
$\circ P$ (next P) : P is true in the next instant
There are a lot of applications of Temporal Logic in hardware description of dynamical systems and concurrent programs, for instance, on the verification of fairness,eventuality, invariance, termination and so on.

However, they mainly treat *Propositional Temporal Logic* (PTL) and do not treat *Temporal Predicate Logic* since the logic is awfully complicated. Especially, there are few researches on the mechanical reasoning method. But the introduction of variables and function symbols can extend the expressive power and also remove the annoyance of writing many similar formulas.

In this paper, PTL is extended to that including free variables and function symbols. We call it *Quantifier Free Temporal Logic (QFTL)*. In QFTL, as for individual variables, only free variables are permitted syntactically. And they are interpreted as universally bound. Therefore, $P(x)$ where x is a variable denotes that $P(x)$ is true for all x.

A mechanical reasoning method called ω- *graphs refutation* is proposed with which unsatisfiability of the (set of) formulas can be checked by means of the *ω-graph*. An ω-graph is introduced a model which is semantically equivalent to the formula of PTL and from it an ω- automaton can be defined. In QFTL, ω-graph is a kind of model scheme, which is also semantically equivalent to the formula. This algorithm is sound but incomplete. It is because of the incompleteness of QFTL. We will show it by simulating Turing Machine in QFTL and using the unsolvability of the halting problem.

2. PRELIMINARIES

2-1 PTL

At first, we will define a syntax and semantics of PTL. Formulas are defined as follows :

(i) an atomic formula (i.e. a propositinal variable)
(ii) $\neg P$, $P \vee Q$, $P \wedge Q$, $P \supset Q$, $P \equiv Q$
where P and Q are formulas
(iii) $\Box P$, $\Diamond P$, $\circ P$ where P is a formula

Axiomatic system is defined as follolws :

Axioms

A1 $\Box(P \supset Q) \supset (\Box P \supset \Box Q)$
A2 $\Box P \supset P$
A3 $\circ\neg P \equiv \neg\circ P$
A4 $\circ(P \supset Q) \supset (\circ P \supset \circ Q)$
A5 $\Box P \supset \circ P$
A6 $\Box P \supset \circ\Box P$
A7 $\Box(P \supset \circ P) \supset (P \supset \Box P)$

Inference rules

R1 (PL tautology)

$$\frac{P \text{ is an instance of a tautology in PL}}{P}$$

R2 (modus ponens)

$$\frac{P \quad P \supset Q}{Q}$$

R3 (necessity)

$$\frac{P}{\Box P}$$

[Definition 2.1]

A formula which does not contain $\Box$, $\Diamond$, $\circ$ as symbols is said to be a *semi-atomic-formula* and abbreviated by SAF.

Semantics is given based on Kripke model. [Kripke]

[Definition 2.2]

(1) *A complete assignment* for P is a function which assigns a truth value (**T** or **F**) to every propositinal variable of P

(2) *A model* M for P is an infinite sequence of complete assignments

$$M = K_0, K_1, K_2, \ldots$$

and M_i is a shifted sequence of M ;

$$M_i = K_i, K_{i+1}, \ldots.$$

We define truth value assignment in the usaul way. Let Q be a subformula of P, and M be a model $M = K_0, K_1, \ldots$. We define K_i as assigning either **T** or **F** to Q, inductively. If Q is propositinal variable, Q has already been defined. $\neg Q$ is assigned **T** (**F**) if and only if Q is assigned **F** (**T**). $Q \wedge R$ is assigned **T** if both Q and R are assigned **T**; otherwise it is assigned **F**. $\Box Q$ is assigned **T** by K_i if every $K_j (j \geq i)$ assigns **T** to Q; otherwise it is assigned **F**. $\circ Q$ is assigned **T(F)** by K_i if and only if K_{i+1} assigns **T(F)** to Q. We define $Q \vee R$ as $\neg(\neg Q \wedge \neg P)$, $Q \supset R$ as $\neg(Q \wedge \neg R)$ and $\diamond Q$ as $\neg\Box\neg Q$.

2-2 QFTL

QFTL is an extension of PTL to include free variables. All free variables are interpreted to be bound by universal quantifiers, which means that a formula $P(x)$ implicitely indicates $\forall x P(x)$. QFTL has more expressive power than PTL. Besides, it is easy to apply it to mechanical proving, since it has neither local variables nor existential quantifiers.

In this case, a concept of term is introduced.

(i) an individual variable

(ii) $f(t_1, ..., t_n)$ where f is a function symbol and $t_1, ..., t_n$ are terms

An atomic formula is defined in the form of $p(t_1, ..., t_n)$ where p is a function symbol and $t_1, ..., t_n$ are terms

Formulas of QFTL are defined as same as that of PTL.

Usually a model structure is defined by a non-empty domain and assignments to the individual variables and to the predicate symbols. We take the Herbrand Universe ($\mathcal{H}$) as a non-empty domain.

[Definition 2.3]

For a given formula P and its $\mathcal{H}$, a *complete assignment* for P in $\mathcal{H}$ is defined as a function :

to every n-adic function symbol of P we assign a mapping from $\mathcal{H}$ to $\mathcal{H}$

to every n-adic predicate symbol of P

we assign a set of ordered n-tuples of members in $\mathcal{H}$

(As for constants, since we have the same symbols in $\mathcal{H}$, we assign them, respectively.)

Model and shifted model are defined similarly with the case of PTL.

Let Q be a subformula of P, and let M be a model $M = K_0, K_1, ...$ We define K_i as assigning either **T** or **F**, inductively. If Q is an n-adic atomic formula $p(a_1, ..., a_n)$ where $a_1, ..., a_n$ are elements of $\mathcal{H}$, and if ψ is the set of ordered n-tuples K_i assigns to p, then Q is assigned T by K_i if $a_1, ..., a_n \in \psi$; otherwise Q is assigned **F**. Assume that Q, R are formulas which contain no variables. The assignment of $\neg Q$, $Q \wedge R$, $\Box Q$ and $\circ Q$ is defined as same as that in PTL. If $P(x_1, ..., x_n)$ is a formula which contains variables $x_1, ..., x_n$, $P(x_1, ..., x_n)$ is assigned **T** if and only if $P(a_1, ..., a_n)$ is assigned T for all $a_1, ..., a_n$ in $\mathcal{H}$.

Note that all variables in P are interpreted as global ones. Namely, a variable has a uniform interpretation over P.

2-3 Provability and consistency

[Definition 2.4]

In general, for a formal system A,

(1) We call the finite sequence of formulas in the system A a *proof* such that, each of which is either axiom or the conclusion of an inference rule whose hypotheses procede that formula.

(2) If P is the last formula in a proof, we call the proof a *proof of P*.

(3) If there is a proof of P in A, then P is *provable*, in A and denoted by $\vdash_A P$. When no confusion results, we omit the subscript A.

(4) P is *consistent* iff not $\vdash_A \neg P$.

(5) A finite set of formulas $\{P_1, ..., P_n\}$ is *consistent* iff not $\vdash_A \neg(P_1 \wedge ... \wedge P_n)$.

(6) An infinite set K of formulas is *consistent* iff no finite subset of K' is consistent.

(7) A formula(set) is *inconsistent* iff it is not consistent.

2-4 Satisfiability and validity

[Definition 2.5]

For a formula P of a formal system A and a model

$M = K_0, K_1, ...,$

(1) P is said to be *true(false) in* M if P is assigned **T**(**F**) by K_0.

(2) If P is true in a model M, we say that M *satisfies* P, and denote it by $M \models P$.

(3) P is *satisfiable* if there exists a model which satisfies P.

(4) P is *unsatisfiable* if it is not satisfiable

(5) P is *valid* if it is true in every model and we write $\models P$

3. PTL AND ω- GRAPH

An *ω-graph* is introduced in order to give a graph representation of a formula of Temporal Logic[FST]. To simplify the problem, at first we will consider formulas of PTL. A formula of PTL is decomposed into a sequence of formulas each of which holds at each instant. An ω- graph represents a behavior on time of the formula. It is shown later that an ω- regular automaton equivalent to the formula can be define from the ω- graph.

3-1 Decomposition rule

When a formula P in PTL is given, it is transformed into the logically equivalent form of $\vee_i(A_i \wedge \circ B_i)$ where A_i is a SAF. It is according to the following decomposition rule based essentially upon the *tableau methods*[Wolper].

$\Box P \Longrightarrow P \wedge \circ\Box P \quad ; \quad \Diamond P \Longrightarrow P \vee \circ\Diamond P$

[Definition 3.1]

When a formula P is decomposed into the form of $\vee_i(A_i \wedge \circ B_i)$, B_i is said to be a *state*, and A_i, a *label to a state* B_i. And P is said to be an *initial state*.

Furthermore, two special vectors as flags are added to each state. These vectors are called *r-vector* and *h-vector* both of which guarantee the realization of eventualities. r-vector indicates the current realization of positive eventualities, and h-vector indicates the history of realization. Each vector consists of k elements where k is a number of positive eventualities appearing in the given formula P, and each element has a value of 1 or 0.

Each state is in the form of $[B; r, h]$ where B is a formula, r, h are r-vector and h-vector, respectively. It is said to be a *state-with-condition*.

[Definition 3.2]

A formula P is converted into the logically equivalent form P' by eliminating $\supset$, $\equiv$ and removing $\neg$ to the innermost position. Let $\Diamond Q$ be a subformula of P'. If $\Diamond Q$ does not have $\neg$ on the immediate ahead, $\Diamond Q$ is said to be a *positive eventuality*. If Q is true at a state, it is said that *this eventuality is realized* at a state.

For example, assume a formula $(\Diamond P \supset \Diamond(Q \wedge \neg\Diamond R)) \vee \neg\Box S$ which is equivalent to $\Box\neg P \vee \Diamond(Q \wedge \neg\Diamond R) \vee \Diamond\neg S$. $\Diamond(Q \wedge \neg\Diamond R)$ and $\Diamond\neg S$ are positive eventualities.

The values of r-vector and h-vector are formally defined as follows.

(1) r-vector $\quad r = (r_1, ..., r_k)$
 (i) $r_i = 0 (1 \leq \forall i \leq k)$ for the initial state
 (ii) If B does not include $\Diamond F_i$ as a subformula, then $r_i = 1$
 (iii) If B inculdes $\Diamond F_i$ as a subformula
  if $\Diamond F_i$ is realized at the state, then $r_i = 1$,
  otherwise, $r_i = 0$

(2) h-vector $\quad h = (h_1, ..., h_k)$
 (i) $h_i = 0 (1 \leq \forall i \leq k)$ for the initial state
 (ii) If $r_i = 1$, then $h_i = 1$
 (iii) If $r_i = 0$, h_i is defined according to the h-vector $(h'_1, ..., h'_k)$ of the preceding state-with-condition
  if $h'_i = 0$ or $h'_j = 1 (1 \leq \forall j \leq k)$, then $h_i = 0$

otherwise, $h_i = 1$.

We will show an example of decomposition of $\Box\Diamond P \wedge \Box\Diamond Q$ with these vectors. $[\Box\Diamond P \wedge \Box\Diamond Q; (0,0), (0,0)]$

$$\Longrightarrow \{P \wedge Q \wedge \circ[\Box\Diamond P \wedge \Box\Diamond Q; (1,1), (1,1)]\}$$
$$\vee \{P \wedge \circ[\Box\Diamond P \wedge \Box\Diamond Q; (1,0), (1,0)]\}$$
$$\vee \{Q \wedge \circ[\Box\Diamond P \wedge \Box\Diamond Q; (0,1), (0,1)]\}$$
$$\vee \{\circ[\Box\Diamond P \wedge \Box\Diamond Q; (0,0), (0,0)]\}$$

Assume that the element of h-vector corresponding to the $\Diamond F_i$ is 0. Then, by the definition of h-vector, $\Diamond F_i$ is not realized yet at the state. Therefore, the following condition should be satisfied :

$$\vdash [B; r, h] \supset \Diamond F_i$$

In the above example, the following conditions must be satisfied :

$$\vdash [\Box\Diamond P \wedge \Box\Diamond Q; (1,0), (1,0)] \supset \Diamond P$$
$$\vdash [\Box\Diamond P \wedge \Box\Diamond Q; (0,1), (0,1)] \supset \Diamond Q$$
$$\vdash [\Box\Diamond P \wedge \Box\Diamond Q; (0,0), (0,0)] \supset \Diamond P \wedge \Diamond Q$$

3-2 ω- graph

An ω- graph is a finite digraph whose nodes corresponds to the states-with-condition and edges to the labels. It is constructed as follows.

Construction of an ω- graph

(1) Create a node corresponding to the initial state-with-condition. We call it an *initial node*.

(2) According to the decomposition rule, transfrom P into the form of $\vee_i(A_i \wedge \circ B_i)$. If there exists the node N_i corresponding to the state with a condition $[B_i; r, h]$, make the edge labeled by A_i. If not, make a new node with the edge labeled by A_i.

(3) Take each node N_i as an initial node and execute the above procedure (2).

This procedure terminates in a finite time, since there appear a finite number of states with a condition. The ω- graph of $\Box\Diamond P \wedge \Box\Diamond Q$ is shown in the Fig 1.

[Definition 3.3]

(1) For a sequence $\{N_0, N_1, ..., N_m\}$ of nodes of an ω- graph where N_0 is an initial node, and there exists an edge from N_i to N_{i+1} directly, the sequence is said to be a *path of length m*. (m might be infinite)

(2) If all the elements of the h-vector of a node are 1, then the node is said to be an *ω-node*, otherwise, it is said to be a *t-node*.

3-3 Fundamental theorem

Next, we will show a correspondance between an ω- graph and the formal system of PTL.

[Lemma 3.1]

For any formula P in PTL,

$$P \equiv \vee_{i=1}^{N_m}(A_i \wedge \circ^m B_i)$$

where $A_i = l_0 \wedge \circ l_1 \wedge ... \wedge \circ^{m-1} l_{m-1}$ (each l_i is a SAF), B_i is a formula and m is finite.

Proof) If P is decomposed m times according to the decomposition rule, it is denoted by the finite disjunction of the paths of length m to the graph. If we relate A_i to the path and B_i to the state, the lemma holds.

[Lemma 3.2]

If P is consistent, there exists formulas Q, R which satisfiy the following condition.

(i) $\vdash Q \wedge \circ^m R \supset P$

(ii) $Q, \Box\Diamond R$ are consistent formulas and m is finite

where $Q \equiv q_0 \wedge \circ q_1 \wedge \ldots \wedge \circ^{m-1} q_{m-1}$ (q_i is SAF)

Proof)

Let all the states which appear through the decomposition of P be $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$. If we assume $\varphi \equiv \varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$, then it is trivial, then $\vdash P \supset \Box\varphi$ If we assme that $\vdash \neg\Box\Diamond\varphi_i$ $i(1 \leq \forall i \leq n)$, then $\vdash \wedge_{i=1}^{n} \neg\Box\Diamond\varphi_i$ holds. Therefore, $\vdash P \supset \Box\varphi \wedge \Diamond\Box\neg\varphi_i \supset false$. It means that P is inconsistent, which is a contradiction. Thus, there exists i such that $\Box\Diamond\varphi_i$ is consistent. We can take R as φ_i. Furthermore, let $(q_0, q_1, \ldots, q_{m-1})$ be the path which reaches the node corresponding to φ_i, then Q is consistent. Moreover, $\vdash Q \wedge \circ^m R \supset P$ holds from the lemma 3.1.

From lamma 3.2, P is consistent, there exists a path which visits some node infinitely often on the corresponding automaton.

[Theorem 3.1] (Fundamental Theorem)

If a formula P is consistent, there exists a path which visits an ω- node infinitely often on the corresponding ω- graph.

Proof) Assume that all the paths that satisfy the condition of the lemma 3.2 visit ω- nodes only finite times. From the lemma 3.2, there exists B_k such that $\Box\Diamond B_k$ is consistent. Let $\Diamond F$ be the positive eventuality whose corresponding element of h-vector keeps 0 after some time, then $\vdash B_k \supset \Diamond F_i$ is required. Therefore $\Box\Diamond F_i$ is consistent, namely, not $\vdash \Diamond\Box\neg F_i$. On the other hand, F is not realized, since the value of h-vector is always 0, which indicates that $\vdash \Diamond\Box\neg F$ holds. This is a contradiction. Hence, there exists a path which satisfies the condition.

In the following discussion, it is shown that an ω- automaton an be defined based on the ω- graph, and that there exists a one to one correspondance between PTL and the class of the ω- regular language accpted by the ω- automaton.

<u>3-4 ω- automata</u>

An ω-regular automaton [MacNauton, CG] is an automaton managing the infinite length words over the input alphabet, while a usual one treats finite length words.

[Definition 3.4]

(1) An *ω-regular automaton* is a five tuple $(S, \Sigma, \delta, S_0, F)$; S is a finite set of states, Σ is a finite input alphabet, δ is the transitive function from $S * \Sigma$ to 2^S, S_0 in S is the initial state, and F, which is a subset of 2^S is the set of designated state sets. And the language accepted by this automaton is said to be an *ω-regular language*.

(2) Let l be an input word (sequence) ; $l = \Pi_{i=0}^{\infty} s_i \in \Sigma^\omega$ where $s_i \in \Sigma(\forall i \geq 1)$, and $In(l)$ be the set of all states the automaton enters infinitely many times in reading the input word. The word l is *accepted* by the automaton iff $\exists H \in F; In(l) \bigcap H \neq \phi$. We call a member of F a *final state*.

(3) For the above l, we denote by l_k the shifted word (sequence) $l_k = \Pi_{i=k}^{\infty} s_i$.

For an ω- graph, let S be a set of nodes. Σ be an input alphabet, S_0 be an initial node, and δ be a set of function such that $N_i = l_{ij}N_j$ where l_{ij} is a label of the arc from the node N_i to the node N_j. And let F be a set of ω- nodes, then $< S, \Sigma, S_0, \delta, F >$ is an ω- automaton.

3-5 Canonical model

The completeness of PTL is proved, for example, by Wolper based upon tableau methods [Wolper]. We will show the completeness of PTL based upon the Henkin's proof of the completeness of first order logic. The basic idea in Henkin's proof is to costruct a maximal consistent set for any consistent formula and show that it can determine the truth value of the formula. As for PTL, we will construct a *canonical set*, namely, a system of maximal consistent sets.

[Definition 3.5]

(1) For a consistent set S, if $S \bigcup \{P\}$ is inconsistent for any formula P not in S, S is said to be a *maximal consistent set* and abbreviated by MCS.

(2) Let G_1, G_2 be MCS's. For any formula P, if $\circ P$ is in G_1 iff P is included in G_2, G_2 is said to be a *successor of* G_1.

[Definition 3.6]

(1) A sequence of MCS's $M^* = K_0, K_1, K_2, ...$ which satisfies the following conditions is said to be a *canonical set* :

[I] For any $i (i \geq 0)$, K_{i+1} is a successor of K_i.

[II] If $\diamond P \in K_i$, then there exists $K_j (j \geq i)$ such that $P \in K_j$.

(2)We will construct a canonical set starting from P similarly to the method in [HC] . But it is insufficient only to construct a sequence of MCS's. We need the condition [II] since the sequence might include a set which is not a model. It is essentially because an axiomatic system of PTL includes an induction on time as an axiom A7.

(3) For a canonical set M^*, $\models_*$ is defined as follows :

$$M^* \models_* P \quad iff \quad P \in K_0$$

[Theorem 3.2]

A canonical set M^* is a model, that is, $\models_*$ is equivalent to $\models$ defined in section 2.

Proof)

(i) If P is a propositional variable, it is trivial from the definition.

(ii) If we assume $M^* \models_* \neg P$, then $\neg P \in K_0$ holds from the definition. Since K_0 is consistent, *not* $M^* \models_* P$. On the other hand, if we assume *not* $M^* \models_* P$, then $P \notin K_0$. Since K_0 is maximal, $\neg P \in K_0$. Therefore, $M^* \models_* \neg P$ holds.

(iii) If we assume $M^* \models_* P \vee Q$, then $P \vee Q \in K_0$. Since K_0 is maximal, either $P \in K_0$ or $Q \in K_0$ holds. (Suppose $P \notin K_0$ and $Q \notin K_0$, then $\neg P \in K_0$ and $\neg Q \in K_0$ since K_0 is maximal. It means K_0 is inconsistent.) Thus, either $M^* \models_* P$ or $M^* \models_* Q$ holds by the induction hypothesis. The only if part can be proved in the same way.

(iv) If we assume $M^* \models_* \circ P$, then $\circ P \in K_0$ holds. Then $P \in K_1$ by the condition [I]. Therefore $M_1^* \models_* P$. The only if part can be proved in the same way.

(v) If we assume $M^* \models_* \Box P$, then $\Box P \in K_0$. Since K_0 is maximal, $\circ^i P \in K_0 (\forall i \geq 0)$.

Therefore, since $P \in K_i(\forall i \geq 0)$ by the condition [I], $M_i^* \models_* P$ holds. On the other hand, if we assume *not* $M^* \models_* \Box P$, $\Box P \notin K_0$. Since K_0 is maximal, $\neg\Box P(\equiv \Diamond\neg P) \in K_0$. Then there exists $K_j(j \geq i)$ such that $\neg P \in K_j$ by the condition [II]. Since K_j is consistent, $P \notin K_j$. Thus, there exists j such that *not* $M_j^* \models_* P$ holds.

From the above discussions, M^* is a model . ∎

3-6 Completeness

[Theorem 3.3]
If a formula P is consistent, there exists a model which satisfies P.

Proof) The given formula is decomposed into the sequence of consistent formulas $(P_0, P_1, ..., P_n, ..., P_m)$ where $\Box\Diamond P_m$ is consistent by the lemma..... Let l_i be the label from $P_i(0 \leq i \leq m)$ to P_{i+1}. At each step, formulas are given in some enumeration. The procedure is divided into 2 steps.

(Step 1) We will construct MCS of SAF's for each i independently by taking $\{l_i\}$ as an initial set.

(1) $S_i^0 = \{l_i\}$

(2) For the k-th formula α_k,

$$S_i^{k+1} = \begin{cases} S_i^k \bigcup \{\alpha_k\} & (\text{ if } S_i^k \bigcup \{\alpha_k\} \text{ is consistent}) \\ S_i^k \bigcup \{\neg\alpha_k\} & (\text{otherwise}) \end{cases}$$

(3) $S_i = \bigcup^k S_i^k$

Finally we can get a sequence $(S_1, S_2, ..., S_n, ..., S_m)$. It is extended to an inifinite sequence $B = (S_1, S_2, ..., S_n, ..., S_m, S_n, ..., S_m, S_n, ...)$

The above B is said to be a *basic structure*, B_i denotes the i-shifted sequence $(S_i, S_{i+1}, ...)$ of B.

[Definition 3.5]
For a basic structure B of a consistent formula P, if there exists a path $l = (l_0, l_1, ...)$ in the ω- graph of P that satisfies the following condition, it is said that *l is deduced from B* and denoted by $B \vdash l$.

(i) the path visists ω- node infinitely often

(ii) $K_i \bigcup \{l_i\}$ is consistent for any i.

(Step 2) Next we will add any other formula according to the following rule.

(1) $K_i^0 = S_i$

(2) For the k-th formula β_k,

$$K_i^{k+1} = \begin{cases} K_i^k \bigcup \{\beta_k\} & (B_i \vdash l) \\ K_i^k \bigcup \neg\{\beta_k\} & (\text{otherwise}) \end{cases}$$

(3) $K_i = \bigcup^k K_i^k$

By the above procedure, $M^* = (K_1, K_2, ...)$ can be gotten.

[Theorem 3.4]
M^* is a canonical set

Proof) It is trivial that K_i is MCS for any i. As for the condition [I], it is shown that $\circ P \in K_i$ *iff* $P \in K_{i+1}$ for any P is satisfied in the follwing way. If we assume there exists a formula P such that $\circ P \in K_i$ and $P \notin K_{i+1}$ hold. Then *not* $B_{i+1} \vdash P$. Therefore, *not* $B_i \vdash \circ P$, which means $P \in K_{i+1}$. It is a contradiction. The only if part can be shown in the same way. Furthermore, by the procedure of (Step 2), for any formula, either of eventualities of α nor $\neg\alpha$ is realized at some K_i. Hence, the condition (II) is satisfied. ∎

[Theorem 3.5] (Soundness)
If $\vdash P$, then $\models P$.

Proof) It is proved inductively on the structure of P. If P is an axiom, it is trivially valid. Assume that $\vdash Q$ is deduced by $\vdash P$, $\vdash P \supset Q$. From the induction hypothesis, $\models P$, $\models P \supset Q$. Therefore, $\models Q$ is satisfied by the induction. As for R3, it is proved similarly.

[Theorem 3.6] (Completeness)
$\vdash P$ iff $\models P$

Proof) It is clear from the theorem 3.4 and 3.5.

Note that compactness theorem fails, while completeness theorem holds in PTL. Consider the infinite set of formulas $S = \{\diamond\neg P, P, \circ P, \circ\circ P, ...\}$. Every subset of S has a model, but S doesn't have a model.

3-7 PTL and ω- regular language

We will show that a formula of PTL is semantically equivalent to the language accepted by the corresponding ω- automaton.

[Theorem 3.7]
$M \models P \quad iff \quad \exists l \in L(A_p) \quad M \models l$
where $L(A_p)$ denotes the set of languages accepted by the ω- automaton corresponding to P

Proof)
(1) ($\Rightarrow$) From the lemma 3.1, $P \equiv \vee_{i=1}^{N_m}(A_i \wedge \circ^m B_i)$. Suppose $m = 1$, then $P \equiv \vee_{i=1}^{N_1}(A_i \wedge \circ B_i)$. Since $M \models P$, there exists i such that $M \models \circ B_i$ holds. Therefore $M \models A_i$ and $M \models \circ B_i$. Since it is satisfied with respect to any B_i, $M_i = l_i$ holds for any i. If we assume $l = (l_0, l_i, ...)$, then $M \models l$, $l \in L(A_p)$ holds.

($\Leftarrow$) From the fundamental theorem, there exists l such that $\vdash l \supset P$. From the completeness of PTL, $\models l \supset P$. Thus, if $M \models l$, then $M \models P$. ∎

Fundamental theorem is corresponding to the fact that the class of ω- regular language is represented by the Kleene closure $\bigcup_{i=1}^{n} A_i B_i^{\omega}$ (where A_i, B_i are regular expressions). A path is represented as $l_1 \wedge \circ l_2 \wedge .. \wedge \circ^n l_n \wedge \circ^{n+1}\Box\diamond l_{n+1}$

Furthermore, $P \equiv \vee_{j=1}^{N} l^j$ (where N is finite) holds. Therefore the following correspondance hold between formulas in PTL and ω- regular language.

$$
\begin{array}{lll}
\circ P & \Longleftrightarrow & \Sigma \cdot L(A_p) \\
\diamond P & \Longleftrightarrow & \Sigma^* \cdot L(A_p) \\
P \wedge Q & \Longleftrightarrow & L(A_p) \bigcap L(A_q) \\
\neg P & \Longleftrightarrow & \Sigma^\omega - L(A_p)
\end{array}
$$

4. QFTL

4-1 QFTL and ω- graphs

We will extend the concept of ω- graph to that for QFTL defined before. As for the formulas of QFTL, ω- graph is a kind of model schme which is also semantically equivalent to the formula.

[Theorem 4.1]

$$M \models P(x) \quad iff \quad M \models A_{p(x)}$$

$M \models A_{p(x)}$ denotes that there exists $l(x) \in L(A_{p(x)})$ which is true in the model M

Proof)

For any a in $\mathcal{H}$, left-hide-side of the formula says that $P(a)$ is true in the model M and right-hand-side means that there exists $l(a)$ in $L(A_{p(a)})$, which is true in the model M. Therefore it is sufficient to prove the following condition for any $a \in \mathcal{H}$:

$M \models P(a) \quad iff \quad \exists l(a) \in L(A(a))\ s.t.\ M \models l(a)$

Since $P(a)$ and $l(a)$ are propositional, it indicates that the theorem is reduced to PTL.

∎

4-2 ω-graphs refutation

A finite refutation procedure called *ω-graphs refutation* is proposed. It is a procedure to check whether or not a (finite) set of formulas $\{P_1, P_2, ..., P_n\}$ is unsatisfiable by means of ω-graphs. It is analogous to a resolution principle in first order logic. ω-graphs correspond to clauses. Similar to the resolution principle, we repeat a procedure to generate resolvents from the set. We select an appropriate substitution to cut arcs in ω-graph, that is, to make arcs to include a complementary pair as a result of making the product of those graphs, which corresponds to finding a most general unifier. It is a useful method since basically we have only to find a substitution to cut arcs. If ϕ (i.e. the language accepted by the automaton is empty) is generated, it is unsatisfiable. However, different from the resolution principle, it doesn't always generate ϕ from an unsatisfiable set. It is due to the incompleteness of QFTL, shown later.

Let Ω be an ω-graph and σ be a substitution, then $\Omega\sigma$ denotes the ω- graph obtained from Ω by applying a substitution σ.

[Definition 4.1]

Let Ω_1, Ω_2 be two ω-graphs with no variables in common, and σ_1, σ_2 are some substitutions. If Ω_3 be the product ω-graph [FST] of $\Omega_1\sigma_1$ and $\Omega_2\sigma_2$ is Ω_3 Ω_3 is said to be the *resolvent ω-graph of* Ω_1 *and* Ω_2.

Note that if we assume that σ_1, σ_2 are some substitutions and that Ω_1, Ω_2 are the graphs corresponding to the formulas $P_1\sigma_1, P_2\sigma_2$, respectively, then the resolvent ω-graph of Ω_1 and Ω_2 corresponds to the formula of $P_1\sigma_1 \wedge P_2\sigma_2$.

The algorithm is as follows :

ω-graphs refutation

For a (finite) set of formulas $\{P_1, P_2, ..., P_n\}$,

(1) make an ω-graph Ω_i corresponding to P_i for each i $(1 \leq i \leq n)$ and let the set be

$S_0 = \{\Omega_1, \Omega_2, ..., \Omega_n\}$

(2) let $S = S_0$

(3) select two ω-graphs Ω_i and Ω_j $(1 \leq i, j \leq n)$

(4) rename the variables so that Ω_i and Ω_j do not have variables in common

(5) for Ω_i and Ω_j, find appropriate substitutions σ_i, σ_j, to cut arcs, respectively

(6) let the resolvent ω-graph of $\Omega_i\sigma_i$ and $\Omega_j\sigma_j$ be Ω

(7) if Ω is ϕ, then stop

it is unsatisfiable ;

otherwise, let S be $S \bigcup \{\Omega\}$

(8) go to (3)

For example, consider the unsatisfiability of the following formula of QFTL :

$\Box(P(x) \supset \circ P(f(x))) \wedge P(a) \wedge \Box\neg P(f(f(a)))$

We will do the refutation procedure on the set of formulas.

$\{ \Box(P(x) \supset \circ P(f(x))), \quad P(a), \quad \Box\neg P(f(f(a))) \}$

Let each formula be F_1, F_2, and F_3, respectively. At first we will construct the ω-graph Ω_1, Ω_2, and Ω_3 corresponding to F_1, F_2 and F_3, respectively (Fig 2.1). Second, we construct the ω-graph Ω_{11} by substituting a for x in F_1. The resovent ω- graph of $\Omega_{11} \bigcap \Omega_2 \bigcap \Omega_3$ (Fig 2.2) is not empty. Next, we construct the ω-graph Ω_{12} by substituting $f(a)$ for x in F_1, and make the resolvent ω-graph of $\Omega_{11} \bigcap \Omega_{12} \bigcap \Omega_2 \bigcap \Omega_3$ (Fig 2.3). Since this graph is empty, the given formula is unsatisfiable.

4-3 Turing Machine

As for QFTL, the completeness theorem fails. We will show it by using the concept of Turing Machine(TM).

TM is defined in various ways. In this paper, we will consider the deterministic TM together with two tapes, an input tape and an output tape, either of which has the left end and no right end. Initially all the squares of the output tape contain 0. I/O format consists of string of 1's. The string of 1's of length $n+1$ denotes the integer n. At each instance, TM does the following behaviors with its two heads, read head and write head.

(1) read the symbol from the input tape

(2) write the symbol on the output tape

(3) move both heads

TM is formally defined as the 5-tuples

$TM = (A, S, Q, S_0, H)$ where

A : a finite nonempty subset of tape symbols

S : a finite nonempty subset of states

Q : a finite nonempty subset of rules of action written in the form :

$< q, s_1, s_2 > \quad \rightarrow \quad < q', s'_2, d_1, d_2 >$

where $q, q' \in S$ are a present state and a new state of TM, respectively. s_1, s_2 are present recognized symbols of the I/O tape, respectively. s'_2 is a new symbol of the output tape. d_1, d_2 denotes the direction in which the read/write head moves, respectively. It takes the value either of L(left), R(right) or S(still). $S_0 \in S$ denotes the start state and $H \subseteq S$ is a set of final states.

4-4 Simulation of the behavior of TM in QFTL

We will simulate the behavior of TM in QFTL. For example, we will consider the TM which computes the function $f(n) = n + 1$.

$A = \{1, 0\}, \quad S = \{q_0, q_1, q_2, q_3\}$

$S_0 = q_0, \quad H = \{q_3\}$

$$Q = \begin{cases} < q_0, 0, 0 > & \rightarrow & < q_1, 0, L, L > & (1) \\ < q_1, 1, 0 > & \rightarrow & < q_2, 1, S, L > & (2) \\ < q_2, 1, 0 > & \rightarrow & < q_2, 1, L, L > & (3) \\ < q_2, 0, 0 > & \rightarrow & < q_3, 0, S, S > & (4) \end{cases}$$

These rules are represented in QFTL.

(1) $\Box\{(Q_0 \wedge P_1(u) \wedge N_1(u) \wedge P_2(v) \wedge N_2(v))$
$\supset \circ(Q_1 \wedge P_1(u+1) \wedge P_2(v+1) \wedge N_2(v+1))\}$

(2) $\Box\{(Q_1 \wedge P_1(u) \wedge E_1(u) \wedge P_2(v) \wedge N_2(v))$
$\supset \circ(Q_2 \wedge P_1(u) \wedge P_2(v+1) \wedge E_2(v+1))\}$

(3) $\Box\{(Q_2 \wedge P_1(u) \wedge E_1(u) \wedge P_2(v) \wedge N_2(v))$
$\supset \circ(Q_2 \wedge P_1(u+1) \wedge P_2(v+1) \wedge E_2(v+1))\}$

(4) $\Box\{(Q_2 \wedge P_1(u) \wedge N_1(u) \wedge P_2(v) \wedge N_2(v))$
$\supset \circ(Q_3 \wedge P_1(u) \wedge P_2(v) \wedge N_2(v+1))\}$

where Q_0, Q_1, Q_2, Q_3 mean that the states of TM are q_0, q_1, q_2, q_3, respectively. $P_1(u), P_2(u)$ denote that read/write head point the u-th square of the I/O tape, respectively. $E_1(u), E_2(u)$ denote that the symbol in the square of I/O tape are 1, and $N_1(u), N_2(u)$ denote 0, respectively.

Furthermore, the rules of action implicitely mean that if neither of the left-hand-sides of (1) ~(4) is satisfied, TM halts at the state Q_3. We describe it in QFTL, too. (It is trivial that we can.) Conjunction of the rules is denoted by D. Initial condition I can also be described. For example, $n = 2$, it is given as follows.

$N_1(a) \wedge E_1(a+1) \wedge E_1(a+2) \wedge E_1(a+3)$
$\wedge N_1(a+4) \wedge N_2(v) \wedge Q_0 \wedge P_1(a) \wedge P_2(b)$
where a, b are constants.

A part of frame axioms F are determined as follows.

$\Box[\, Q_0 \vee Q_1 \vee Q_2 \vee Q_3 \,]$
$\Box[\, N_1(x) \vee E_1(x) \,]$
$\Box[\, N_2(x) \vee E_2(x) \,]$

Moreover, mutual exclusions (e.g. $\neg(Q_0 \wedge Q_1)$) must be added.

At last TM can be represented by $D \wedge F \wedge I$.

4-5 Incompleteness of QFTL

[Theorem 4.1] (halting problem of TM)

For any input w and any TM, there is no algorithm to determine whether or not TM will eventually halt.

As is well known, the halting problem of TM is unsolvable. It is reduced to the unsatisfiability problem of QFTL, that is, the condition that TM will eventually halt is represented by the following QFTL sentence :

$D \wedge F \wedge I \wedge \neg\Diamond Q_3$ is unsatisfiable

and the condition that TM will not halt forever is represented as :

$D \wedge F \wedge I \wedge \neg\Box\neg Q_3$ is unsatisfiable

[Theorem 6.2] QFTL is incomplete

Proof) Since we define TM as a deterministic one, there is a unique state of the ω-graph corresponding the formula $D \wedge F \wedge I$ at an instance t. Let S be the set of models which satisfy $D \wedge F \wedge I$. Then, either (i) $\forall M \in S$, $M \models \neg\Diamond Q_3$, or (ii) $\forall M \in S$, $M \models \neg\Box\neg Q_3$ is satisfied. Therefore, either of the followings holds.

$\forall M \in S$,

$$M \models D \wedge F \wedge I \wedge \neg\Diamond Q_3 \qquad (1)$$

$\forall M \in S$,

$$M \models D \wedge F \wedge I \wedge \neg\Box\neg Q_3 \qquad (2)$$

For any model M, $s.t.$ $M \notin S$, both of the above (1) and (2) hold. Thus,

$\models D \wedge F \wedge I \wedge \neg\Diamond Q_3$ is unsatisfiable

or $\models D \wedge F \wedge I \wedge \neg\Box\neg Q_3$ is unsatisfiable

holds. Assume that QFTL is complete, then

$\neg(D \wedge F \wedge I \wedge \neg\Diamond Q_3)$ is provable

or $\neg(D \wedge F \wedge I \wedge \neg\Box\neg Q_3)$ is provable

holds. It means that there exists a finite procedure for determining whether TM will halt or not. It does not depend on the input, and any TM is described in QFTL similarly. Therefore the halting problem of TM is solvable, which is a contradiction. Thus, QFTL is incomplete. ∎

It follows that there are no decision procedure for unsatisfiability of the formula in QFTL.

5. CONCLUDING REMARKS

5-1 Conclusion

The reasoning method called " ω-graphs refutation" which can check unsatisfiability of the QFTL formulas is presented. An ω-graph is proposed as a tool by which mechanical reasoning is done and it is shown that there exists semantically equivalence between QFTL formula and the ω-regular language accepted by the ω-regular automaton (ω-graph) corresponding to the formula. Termination of the ω-graphs refutation is equivalent to the provability. However, the algorithm is not complete. It is due to the incompleteness of QFTL.

5-2 Additional discussions

Although QFTL is incomplete, we can consideer some significant axiomatic systems. For example, the following sytems $\mathcal{F}$ and $\mathcal{F}'$ can be defined. In these systems, a sentence in the form of $P(x) \longrightarrow Q(y)$ is treated, which menas $\forall x P(x) \supset \forall y Q(y)$>

The Axiomatic System $\mathcal{F}$

Axioms

FA1 (instantiation)

$P(x) \longrightarrow P(t_1) \wedge P(t_2)$

where t_1, t_2 are terms not appearing in P

Inference rules

FR1 (PTL tautology)

$$\frac{P \supset Q \text{ is an instance of a tautology in PTL}}{P \longrightarrow Q}$$

FR2 (modus ponens)

$$\frac{P \longrightarrow Q \qquad Q \longrightarrow R}{P \longrightarrow R}$$

$\mathcal{F}$ has the same power with ω- graphs refutation, which means that a formula is provable in $\mathcal{F}$ iff ϕ is generated by the ω- graphs refutation. $\mathcal{F}$ is a weaker system in which the following formula cannot be proved :

$R(x, y) \equiv \Box C(u, v) \wedge \{ \Box L(x, f(y)) \wedge \Diamond(P(x) \wedge \Box\neg P(f(y))) \}$

where

$$C(u, v) \equiv L(u, f(u)) \wedge \{ L(u, v) \supset L(u, f(v)) \} \wedge \{ L(u, v) \supset L(f(u), f(v)) \} \wedge \neg L(u, u)$$

$\mathcal{F}'$ is an extended system by adding to $\mathcal{F}$ the inference rule of induction such as :

FR3 (Induction)

$$\frac{P \longrightarrow \circ P}{P \longrightarrow \Box P}$$

We will illustrate the proof of the above formula in $\mathcal{F}'$.

Proof) Let $Q(x, y)$ be $\Diamond(P(x) \wedge \Box\neg P(f(y)))$.

We will show a sketch of the proof

$\vdash_{\mathcal{F}'} Q(x,y) \longrightarrow [\, P(x) \wedge \Box\neg P(f(y))\,] \vee \circ Q(x,y)$

$\vdash_{\mathcal{F}'} Q(f(y), f(x))$
$\quad \longrightarrow [\, P(f(y)) \wedge \Box\neg Q(f^2(x))\,] \vee \circ Q(f(y), f(x))$

Therefore,

$\vdash_{\mathcal{F}'} Q(x,y) \wedge Q(f(y), f(x))$
$\quad \longrightarrow \circ Q(x,y) \vee \circ Q(f(y), f(x)) \qquad \ldots.(1)$

On the other hand, since

$\vdash_{\mathcal{F}'} \circ Q(f(y), f(x)) \longrightarrow \circ\diamond[\, P(f(y)) \wedge \Box\neg P(f^2(x))\,]$,

$\vdash_{\mathcal{F}'} \circ Q(f(y), f(x)) \longrightarrow \circ\diamond P(f(y))$ holds.

Therefore,

$\vdash_{\mathcal{F}'} \circ Q(f(y), f(x)) \wedge Q(x,y) \longrightarrow \circ\diamond P(f(y)) \wedge \circ Q(x,y)$,

which indicates

$\vdash_{\mathcal{F}'} \circ Q(f(y), f(x)) \wedge Q(x,y) \longrightarrow \circ Q(x,y) \qquad \ldots.(2)$

From (1) and (2),

$\vdash_{\mathcal{F}'} Q(x,y) \wedge \circ Q(f(y), f(x)) \wedge Q(x,y)$
$\qquad \longrightarrow [\, \circ Q(x,y) \vee \circ Q(f(y), f(x))\,] \wedge Q(x,y)$
$\qquad \longrightarrow [\, \circ Q(x,y) \wedge Q(x,y)\,] \vee \circ Q(x,y)$
$\qquad \longrightarrow \circ Q(x,y)$

On the other hand,

$\vdash_{\mathcal{F}'} Q(x,y) \longrightarrow Q(x,y) \wedge Q(f(y), f(x)) \wedge Q(x,y)$.

Therefore, $\vdash_{\mathcal{F}'} Q(x,y) \longrightarrow \circ Q(x,y)$.

Applying FR3, we get $\vdash_{\mathcal{F}'} Q(x,y) \longrightarrow \Box Q(x,y)$.

On the other hand,

$\vdash_{\mathcal{F}'} \Box Q(x,y) \quad \longrightarrow \quad \Box\diamond(P(x)\Box\neg P(f(y)))$
$\qquad\qquad\qquad \longrightarrow \quad \Box\diamond P(x) \wedge \diamond\Box\neg P(f(y))$
$\qquad\qquad\qquad \longrightarrow \quad false$

Therefore, $\vdash_{\mathcal{F}'} Q(x,y) \longrightarrow false$.

Thus, $\vdash_{\mathcal{F}'} R(x,y) \longrightarrow false$ holds. ∎

It is easily proved that $\mathcal{F}'$ has the same power with Manna's system [MP], if formulas are restricted to the type of QFTL. It means that if $\forall x P(x) \supset \forall y Q(y)$ is provable in Manna's system, then $P(x) \longrightarrow Q(y)$ is provable in $\mathcal{F}'$, and vice versa.

Authors are to improve the ω-graphs refutation so that it is equivalent to $\mathcal{F}'$. It means that if the sentence is deduced by using FR3, we can check it in a finite time by the ω-graphs refutation. For example, the following method can be considered to be adopted. Let Ω be an ω-graph corresponding to the given formula. For a non-final state appearing in the ω-graph of $\Omega(a_1/x) \bigcap \ldots \bigcap \Omega(a_n/x)$ and a non-final state in $\Omega(a_1/x) \bigcap \ldots \bigcap \Omega(a_{n+1}/x)$, if both of every pair of arcs connected those states are the products of instantiations of the same formula, such as $P(a_1)$ and $P(a_1)P(a_2)$, then eliminate that state from $\Omega(a_1/x) \bigcap \ldots \bigcap \Omega(a_{n+1}/x)$.

ACKNOWLEDGMENT

This research was supported by FGCS as a subproject of "Intelligent Programming System ". The authors would like to thank Dr. Yokoi and Dr. Furukawa for the chance of this research.

REFERENCES

[CG] Cohen,R.S. and Gold,A.Y., "Theory of ω-Languages 1: Characterization of ω-Context-Free Languages," Journal of Computer and System Science, No.15, pp.169-184, 1977.

[FST] Fusaoka, A., Seki, H. and Takahashi, K., "Quantifier Free Temporal Logic and ω-regular Automaton," a paper of Technical Group on Automata and Languages, pp.69-79, TGAL83-62, IECE, Japan, 1984.

[HC] Hughes,G.E. and Cresswell,M.J., "An Introduction to Modal Logic," Methuen and Co. Ltd, 1968.

[Kripke] Kripke,S.A., "A Completeness Theorem in Modal Logic," The Journal of Symbolic Logic, Vol.24, No.1, March 1969.

[MacNaughton] MacNaughton,R., "Testing and Generating Infinite Sequences by a Finite Automaton," Informatin and Control, Vol.9, pp.521-530, 1966.

[Manna] Manna,Z., "Verification of Sequential Programs : Temporal Axiomatization," Department of Computer Science, Stanford University, September, 1981.

[MP] Manna,Z. and Pnueli,A., "Verification of Concurrent Programs, Part1 ; The Temporal Framework," Department of Computer Science, Stanford University, June, 1981.

[Wolper] Wolper,L.P., "Synthesis of Communicating Process from Temporal Logic Specifications," Department of Computer Science, Stanford University, August 1982.

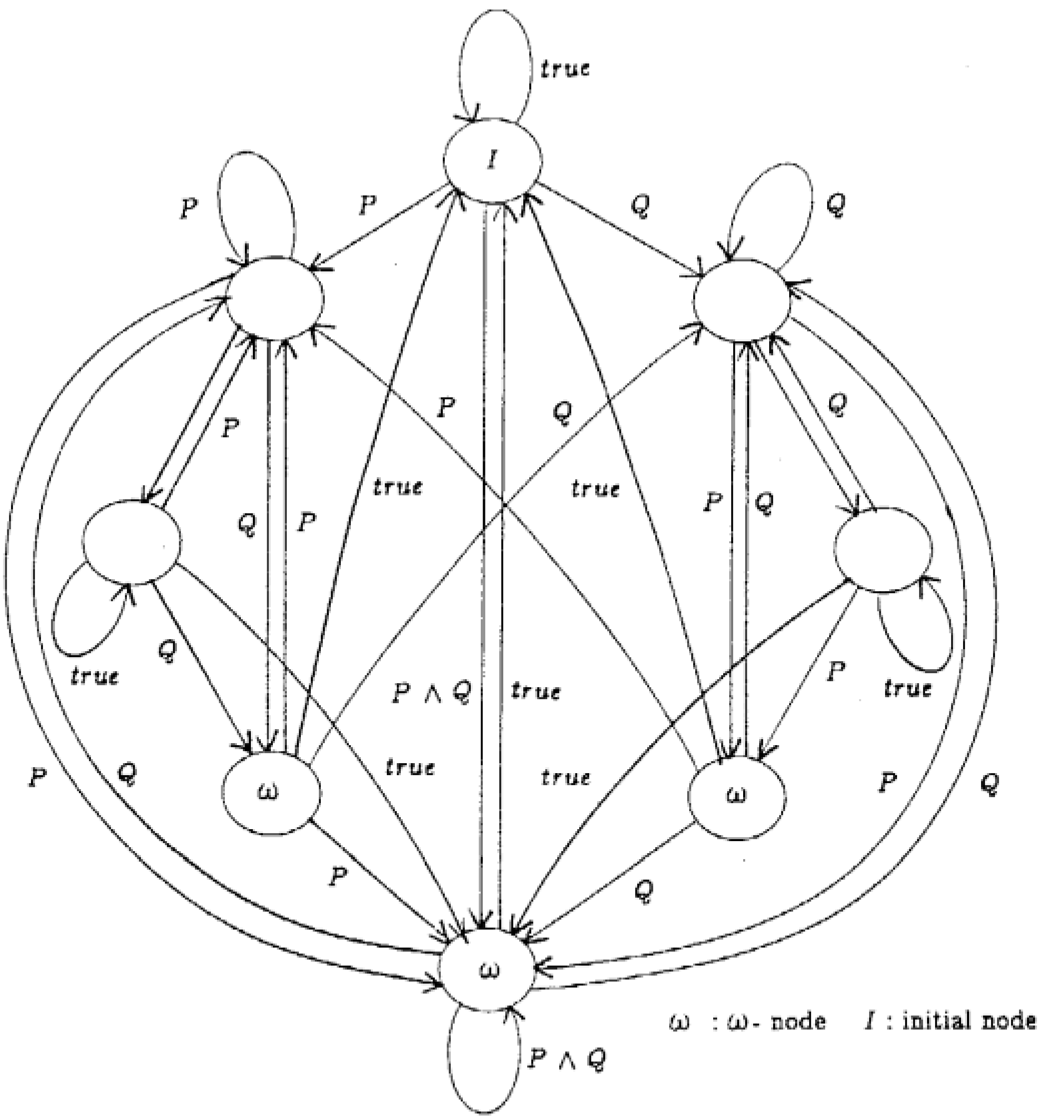

Fig 1 ω- graph of $\Box\Diamond P \wedge \Box\Diamond Q$

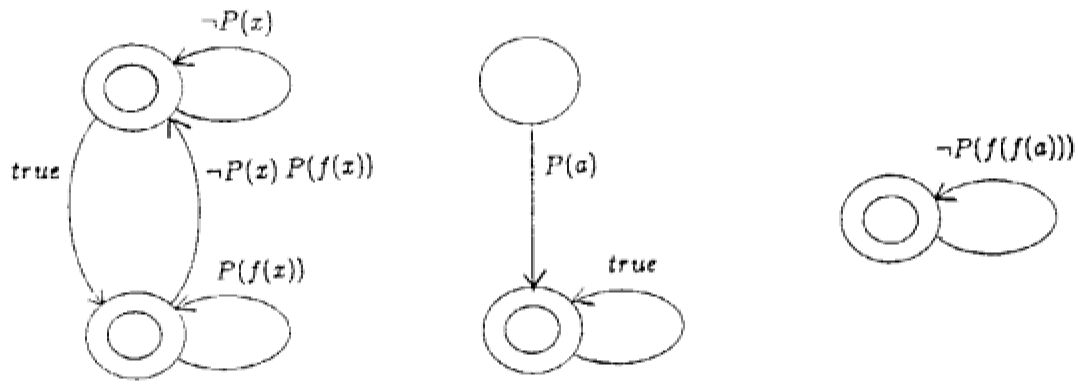

Fig 2.1 ω-graphs $\Omega_1, \Omega_2, \Omega_3$

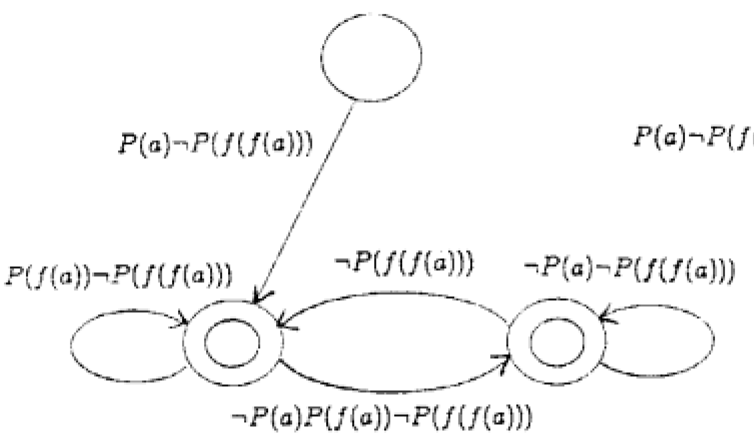

Fig 2.2 ω-graph $\Omega_{11} \bigcap \Omega_2 \bigcap \Omega_3$

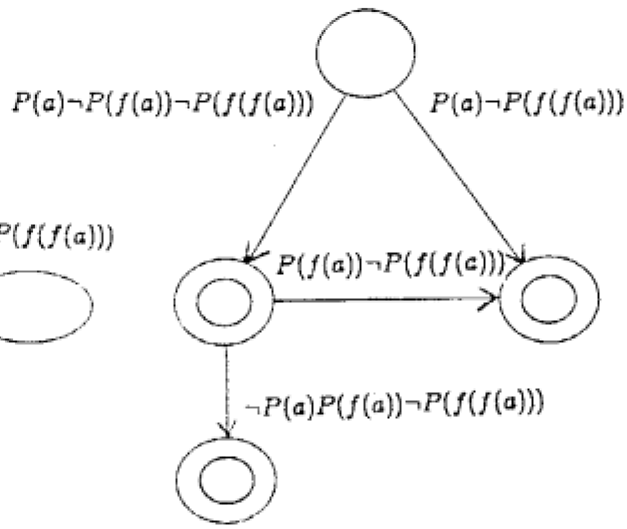

Fig 2.3 ω-graph $\Omega_{11} \bigcap \Omega_{12} \bigcap \Omega_2 \bigcap \Omega_3$