A DESCRIPTION AND REASONING OF PLANT CONTROLLERS IN TEMPORAL LOGIC

Akira Fusaoka, Hirohisa Seki, Kasuko Takahashi

Central Research Laboratory
Mitsubishi Electric Corporation
Amagasaki, Hyogo, JAPAN

ABSTRACT

This paper describes the methodology to deal with the behavior of a dynamical system such as plant controllers in the framework of Temporal Logic. Many important concepts of the dynamical system like stability or observability are represented in this framework. As a reasoning method, we present an *w* -graph approach which enables us to represent the dynamical behavior of a given system, and an automatic synthesis of control rules can be reduced to a simple decision procedure on the w -graph. Moreover, the typical reasoning about the time-dependent system such as a causal argument or a qualitative simulation can be also treated on the w -graph in the same way.

I. INTRODUCTION

This paper describes a system and methodology to deal with a dynamical system in the framework of Temporal Logic. Especially, the temporal reasoning is examined in the analysis and the synthesis of the behavior of plant controllers.

There have been a lot of methods for plant system description which are widely used, such as a ladder diagram and the special programming language. However, these classical methods often seem to be insufficient to deal with the complicated plant system because they lack the means for logical reasoning. A system designer and a system analyst must fully understand the dynamical behavior of the complicated system and compare the specification with the actual design. So they require a sophisticated formal system in which a logical model of a plant controller can be constructed and the properties of the controller can be extracted from that model by logical deduction.

In this paper, we pay special attention to the logical treatment for behavior of plant controllers and we choose the temporal logic as the logical basis of the methodology. Because many problems of a plant controller are related to its dynamics so that they contain the time factor by nature. For instance, a system operator may require a consultation about how a system should be operated to attain some state. This problem, an automatic synthesis of controls, means the mechanical construction of the desirable sequence of actions from the system dynamics. Another example is the causal argument, that is, a reasoning about how the current state has been reached. A qualitative simulation is a similar problem of the causality which is a question about how the system is going to be. These problems require to repeat the logical deduction of forward and backward chaining of the causa) relations which are extracted from the system dynamics. In order to treat these problems, we use a simple model of the temporal formula called an w -graph. A system dynamics which is represented by the temporal formula is transformed to an w -graph. The above problems are reduced into a simple decision procedure on the w -graph.

II. TEMPORAL FRAMEWORK OF SYSTEM DESCRIPTION

A. System Description

We give a simple description of the dynamical system such as automatic controllers. At first we introduce temporal logic [1] (TL in the following) which is an extension of the ordinary logic to include the notion of time, providing four modal operators : $\Box$(henceforth), $\Diamond$(eventually), $\circ$(next) and $\mathcal{U}$(until). Their informal semantics are :

$\Diamond P$—P is true in some future instant
$\Box P$—P is true in all future instants
$\circ P$—P is true in the next instant
$P\ \mathcal{U}\ Q$—P is true in all instants preceding the (first) instant in which Q is true.

A system allows the descriptions from various points of view. We consider that a system is constructed by a hierarchy of objects, which are connected with each other asynchronously. At the top level, a system is divided into two objects which form a feedback loop : a controller and a controlled system. Each object may have the internal states called "flags". The flags of one object cannot be observed from the other objects. An action of each object is performed by sending a message (usually constant value) to the others. The action is represented by

$$if\ \ S\ \ then\ \ X \leftarrow m$$

where S is a predicate of internal state and X is a name of the object which accepts a message m. This means that "Do the action $X \leftarrow m$ whenever S is satisfied ." Here, the action is interpreted as a momentary one, namely an event. In terms of event, we express the algorithm of control by a set of the following formula :

$$if\ \ p_i\ \ then\ \ A_i \qquad (i = 1, \cdots, n)$$

where p_i is a TL formula and A_i is an event. This is treated as the following equivalent formula :

$$\Box[(p_1 \supset A_1) \wedge \cdots \wedge (p_2 \supset A_2)]$$

As an example of the dynamical system, let's consider the following system model of boiler with a temperature controller (Figure 1).

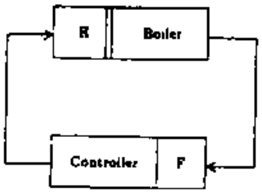

Fig.1 A model of boiler

The boiler has five states *E1, E2, E3, E4* and E5 which indicate the situation of the temperature *$* of water as follows.

$$(E_1 \equiv \theta < \alpha); \quad (E_2 \equiv \theta = \alpha);$$
$$(E_3 \equiv \alpha < \theta < \beta); \quad (E_4 \equiv \theta = \beta);$$
$$(E_5 \equiv \beta < \theta)$$

where a and B are constants of the temperature (a < *P).* The boiler has an additional state *H* which corresponds to the state of fire. If *H* is ON, then the state will transfer to the higher temperature state sometime. The boiler sets a fire if and only if it accepts a message "ON" and also puts off a fire if and only if it accepts a message "OFF" from the controller. When the boiler changes its temperature state from *E* to *E',* it sends E^1 as a message to the flag of the controller. The controller changes the state of *F* if and only if it accepts a message from the boiler, and sends a message to the boiler.

The formulation of these dynamics in TL is as follows.

FRAME AXIOMS

$$\Box[E_1 \vee E_2 \vee E_3 \vee E_4 \vee E_5]$$
$$\Box[\neg(E_i \wedge E_j)] \quad \textit{for each } i,j\,(i \neq j)$$
$$\Box[F_0 \vee F_1 \vee F_2 \vee F_3 \vee F_4 \vee F_5]$$
$$\text{where } F_i = F \leftarrow E_i \quad (i = 1, \cdots 5)$$
$$\text{and } F_0 = \neg[(F \leftarrow E_1) \vee (F \leftarrow E_2) \vee \cdots \vee (F \leftarrow E_5)]$$
$$\Box[\neg(F_i \wedge F_j)] \quad \textit{for each } i,j\,(i \neq j)$$
$$\Box[E_i \wedge \bigcirc E_j \equiv F_j] \quad \textit{for each } i,j\,(j = i \pm 1)$$
$$\Box[H \vee \neg H]$$

MESSAGE PASSING

$$\Box[X \leftarrow m \supset X = m]$$
$$\Box[X = m \supset X = m\, \mathcal{U}\, X \leftarrow m']$$
where X is a flag and m and m' are messages $(m \neq m')$

DYNAMICS

$$\Box[E_1 \wedge \neg H \supset \bigcirc E_1 \wedge F_0]$$
$$\Box[E_1 \wedge H\, \mathcal{U}\, E_2 \supset \Diamond(E_1 \wedge \bigcirc E_2 \wedge F_2) \wedge E_1\, \mathcal{U}\, E_2]$$
$$\Box[E_2 \wedge \neg H \supset \bigcirc E_1 \wedge F_1]$$
$$\Box[E_2 \wedge H \supset \bigcirc E_3 \wedge F_3]$$
$$\Box[E_3 \wedge \neg H\, \mathcal{U}\, E_2 \supset \Diamond(E_3 \wedge \bigcirc E_2 \wedge F_2) \wedge E_2\, \mathcal{U}\, E_3]$$
$$\Box[E_3 \wedge H\, \mathcal{U}\, E_4 \supset \Diamond(E_3 \wedge \bigcirc E_4 \wedge F_4) \wedge E_3\, \mathcal{U}\, E_4]$$
$$\Box[E_4 \wedge \neg H \supset \bigcirc E_3 \wedge F_3]$$
$$\Box[E_4 \wedge H \supset \bigcirc E_5 \wedge F_5]$$
$$\Box[E_5 \wedge \neg H\, \mathcal{U}\, E_4 \supset \Diamond(E_5 \wedge \bigcirc E_4 \wedge F_4) \wedge E_5\, \mathcal{U}\, E_4]$$
$$\Box[E_5 \wedge H \supset \bigcirc E_5 \wedge F_0]$$

B. Goal Specification

The major goal specifictions which appear frequently in system theory are eventuality, stability and observability.

1. Eventuality

Eventuality is a requirement that the control should realize a desirable state sometime in future, so that the eventuality corresponds to the transient behavior in system thoery. This is represented by

$$\textit{Initial condition} \supset \Diamond(\textit{the final state})$$

2. Stability

Stability requires the system to keep some state forever after the state is attained. Namely, stability is formulated as

$$\varphi \supset \Box\varphi$$

Usually eventuality and stability are used together in the following way :

$$\textit{Initial condition} \supset \Diamond\Box(\textit{the final state}).$$

3. Observability

Observability means a requirement that the value of a state of the controlled system can be known to the controller in the allowable delay. Namely, for every state E of the controlled system, there exists a state F of the controller such that

$$\Box[(E = a) \supset (\neg R\, \mathcal{U}\, (F = a) \wedge \Diamond(F = a)]$$

where a is any state value and R is a condition of delay.

A state of the controlled system may be known to the controller before it is reached. This condition is represented as

$$\Box[(F = a) \supset (\neg R\, \mathcal{U}\, (E = a) \wedge \Diamond(E = a))]$$

where E is a state of the controlled system. The observability of the state E is the essential requirement if E is a state to be controlled.

III. REASONING METHOD

A. An ω - graph

In order to reason about the system represented in TL, we construct its model by making a graph called "ω - graph " which is essentially based upon the tableau method [2]. First, for a temporal formula F, we decompose F into the part of present instant and the rest. The decomposition rules are as follows :

$$\Box F \longrightarrow F \quad \wedge \quad \bigcirc\Box F$$
$$\Diamond F \longrightarrow F \quad \vee \quad \{\neg F \wedge \bigcirc(\Diamond F, F)\}$$
$$F_1\, \mathcal{U}\, F_2 \longrightarrow F_2 \quad \vee \quad \{\neg F_2 \wedge F_1 \wedge \bigcirc(F_1\, \mathcal{U}\, F_2)\}$$

Note that the above rule for *oF* contains a term *{oF,F).* We call such terms "marked formulas" and we put this mark in order to express the eventuality property that F will be fulfilled in future.

An w -graph is a graphical representation of the model of a temporal formula whose nodes are corresponding to formulas and whose edges are labeled literals (we call "handles of those edges"). In an w -graph, nodes which correspond to marked formulae are called "transitive nodes" and the other nodes "w -nodes". We call a path of the *w*-graph of *F* "a behavior of F^9 and especially the behavior passing cycles at most one time is called "a skeleton behavior of F". Note that if an a;-node has a cycle, then the infinite sequence of handles on its edges can be a model of the temporal formula. An w -graph can be regarded as a finite state automaton for *w* -language[3]. Therefore, let *F* and G_1 be a temporal formula and its w -graph and *L(F)* the language accepted by the automaton corresponding to G/, then the following relations clearly hold :

(i) $F = F_1 \wedge F_2 \iff L(F) = L(F_1) \bigcap L(F_2)$
(ii) $F = F_1 \vee F_2 \iff L(F) = L(F_1) \bigcup L(F_2)$
(iii) $F = true \iff L(F) = \Sigma^\omega$
(iv) $F = false \iff L(F) = \phi$
(v) $F_1 \supset F_2 \iff L(F_1) \subseteq L(F_2)$,

where Σ is a set of the alphabet. From the property of ω-language, it follows that the containment relation of (v) is decidable.

Using these correspondence, we have developed the following "containment algorithm" to decide the above relation (v). Let F_1 and F_2 be temporal formulae and G_i and $L(F_i)(i = 1, 2)$ corresponding ω-graphs and ω-languages, respectively. Then in order to decide whether $L(F_1) \supseteq L(F_2)$ or not (for convenience, we say "G_1 contains G_2" if $L(F_1) \supseteq L(F_2)$ holds), we perform the following two checking rules :

(1) [Checking Rule 1]
Consider the handles H_{ij} on edges E_{ij} which are outgoing from each initial node N_i of $G_i(i = 1, 2$ and $j = 1, ..., m_i)$. Check whether

$(*) \quad H_{2j} \supset \bigvee H_{1j}$ for each j.

If this relation doesn't hold, then the above $L(F_1) \supseteq L(F_2)$ doesn't hold. Otherwise, for each pair of the handles (H_{1k}, H_{2l}), if $H_{1k} \bigwedge H_{2l}$ is not $false$, then check whether $L(G_{1k}) \supseteq L(G_{2l})$, where G_{1k} and G_{2l} are formulae corresponding to the nodes to which the edges E_{1k} and E_{2l} lead, respectively. This checking rule is applied iteratively until no new pair of nodes appears.

(2) [Checking Rule 2]
When N_2 is an ω-node and it contains a behavior of which node-sequence is in a form $(N_2, T_1, ..., T_i, ..., T_k, N_2)$, then check whether in G_1, we can find a ω-node W_i and a behavior B_1 in a form : $(N_1, ..., S_m, W_i, ..., S_n, W_i)$ which satisfy the condition such that for the infinte behaviors whose node-sequences are $(N_1, ..., S_m) \cdot (W_i, ..., S_n, W_i)^\omega$ and $(N_2, T_1, ..., T_i, ..., T_k, N_2)^\omega$, the corresponding pairs of nodes satisfy the condition (*) of the above [checking rule 1]. Note that this procedure always terminates because such pairs of the nodes are finite.

B. Reasoning on an ω-graph

A lot of reasonings about the behaviors of plant controllers can be reduced to a simple procedure of the ω-graphs for the dynamics of the plant controllers. We examine the typical reasonings which often appear in practical applications.

1. Automatic Synthesis of Controls

An automatic synthesis of controls means a mechanical derivation of the control rules which guide the system to the desirable state. More specifically, it means to determine the unknown control rules which satisfy the following condition :

$Dynamics \wedge Unknown\ Control\ Rules \supset Goal.$

The unknown control rules can be determined by the process of checking the requirement that the ω-graph of $Goal$ must contain the ω-graph corresponding to $Dynamics \wedge Unknown\ Control\ Rules$. This process will be explained in section IV. Furthermore, we can check the observability and the stability of the system in the same way.

2. Qualitative Simulation and Causal Arguments

A qualitative simulation([4]) is a reasoning about how the system is going to be in future. Namely, it means to find out the qualitatively distinct states which can be reached from the current state. This reasoning is corresponding to determining the sequence of unknown states $X_1, X_2, \cdots$ such that

$Dynamics \wedge P \supset \diamond(\neg P \wedge X_1)$
$Dynamics \wedge X_1 \supset \diamond(\neg X_1 \wedge X_2)$
...

where the predicate P specifies the current state. The simple way to solve this problem is to examine a set of skeleton behaviors with respect to the initial condition P, because a skeleton behavior contains all the distinct states which can be reached in that behavior.

A causal argument([4]) means to find out the past state which has caused the present state. Namely, it is a repetition of the process to determine the past states $X_1, X_2, ...$ such that

$Dynamics \supset \diamond(X_1 \supset \diamond P)$,
$Dynamics \supset \diamond(X_2 \supset \diamond X_1)$,
...

so that it corresponds to performing the qualitative simulation backwardly.

IV. An Example

In order to illustrate the procedure of the control synthesis on an ω-graph, we construct the control rules for the example of the boiler described in section II.

Figure 2 shows an ω-graph G_D which corresponds to the dynamics D and the frame axioms for the boiler. By using G_D, we try to find the control rules to guide the system to the stable situation of some states, for instance $E_2 \vee E_3 \vee E_4$. Controls C should satisfy the formula :

$Dynamics\ D \wedge Controls\ C \supset \diamond\Box(E_2 \vee E_3 \vee E_4)$

The control rules for the boiler are represented in the form:

$if \quad E_i \quad then \quad H \leftarrow ON/OFF \quad (i = 1, ..., 5)$

where $H \leftarrow ON/OFF$ means that H is set to be ON or OFF
This is transformed to the temporal formula C:

$C \equiv \Box[(E_1 \supset H_1) \wedge (E_2 \supset H_2) \wedge \cdots \wedge (E_5 \supset H_5)]$
$\equiv \Box[E_1H_1 \wedge E_2H_2 \wedge E_3H_3 \wedge E_4H_4 \wedge E_5H_5]$

where $H_i = H / \neg H$.
The ω-graph G_C corresponding to $C \wedge D$ can be built by substituting E_iH_i for E_i appearing in G_D.

We try to extract the control rules to achieve the eventuality $\diamond E_3$. The ω-graph G_E for this goal shown in Figure 3 should contain the ω-graph G_C, in order to establish

$Dynamics\ D \wedge Control\ C \supset \diamond E_3$

By using the containment algorithm for ω-graphs, we can decide the desired control rules in the following way. Assume that the edge labeled $E_1H_1\overline{H}F_1 + E_2H_2\overline{H}F_1$ exists. Then we have to check the pair of nodes (T_0, ω_1). And if the edge labeled $E_1H_1\overline{H}F_0$ exists, again we go to check (T_0, ω_1). Since ω_1 is an ω-node, it contains the infinite behavior of $(E_1H_1\overline{H}F_0)^\omega$. On the other hand, T_0 contains no infinite behaviors of the form $(E_3)^\omega$. $(D \wedge C \supset \diamond E_1)$ is not satisfied if

$(E_1H_1\overline{H}F_1 + E_2H_2\overline{H}F_1) \cdot E_1H_1\overline{H}F_0 = true$

Namely, $H_1 = H$. By repeating the similar procedure for the other edges, we get the following value assignments of H_i.

$H_1 = H, \quad H_2 = H \quad H_4 = \neg H \quad H_5 = \neg H$

As for the stability and observability, we can build the w -graph, and controls synthesis can be done similarly. In this way, we get the following desired control rules by combining with the above control rules altogether.

$$\textbf{\textit{if}}\ F_1\ \textbf{\textit{then}}\ H\leftarrow ON\ ;\ \textbf{\textit{if}}\ F_2\ \textbf{\textit{then}}\ H\leftarrow ON\ ;$$
$$\textbf{\textit{if}}\ F_4\ \textbf{\textit{then}}\ H\leftarrow OFF\ ;\ \textbf{\textit{if}}\ F_5\ \textbf{\textit{then}}\ H\leftarrow OFF$$

V. Concluding Remarks

A method for description and reasoning of plant controllers is discussed in the framework of temporal logic. As a method of reasoning about a dynamical system, we present an w -graph approach. Since the w -graph representation gives the inner model of the dynamical system and we can perform various kinds of reasoning on it, it appears to be more effective than usual theorem proving methods such as natural deduction or tableau method. A system which manipulates an w -graph is currently under development for practical applications.

Although the major concern of this paper is in the complicated plant system, the temporal framework which is developed here can be extended to the other species of dynamical systems.

References

[1] Manna,Z. and Pnueli,A. "Verification of Concurrent Programs, Part 1: The Temporal Framework," Report No. STAN-CS-81-836, Department of Computer Science, Stanford University, CA, June 1081.

[2] Wolper,P.L., "Synthesis of Communicating Processes from Temporal Logic Specifications," Report No. STAN-CS-82-925, Department of Computer Science, Stanford University, CA, August 1982.

[3] McNaughton,R., "Testing and Generating Infinite Sequences by a Finite Automaton," Information and Control 9 (1966) 521-530.

[4] de Kleer,J. and Brown,J.S., "Foundations of Envisioning," In Proc. AAAI-82. Pittsburgh, Pennsylvania, August, 1982, pp. 434-437.

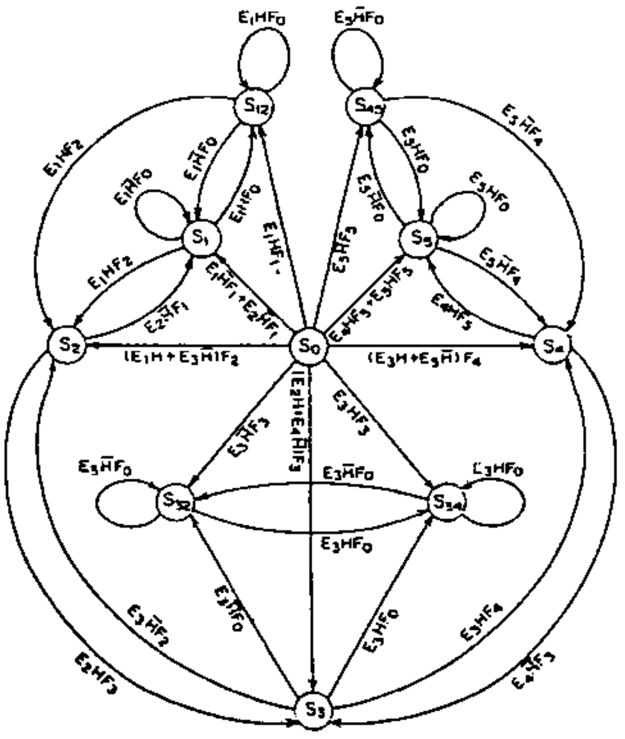

Fig.2 ω-graph G_D of the system dynamics

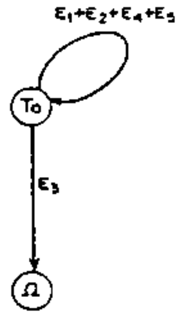

Fig.3 ω-graph G_E of the eventuality