Supervisory Control of Timed Discrete-Event Systems under Partial Observation

F. Lin and W. M. Wonham

Abstract—This paper extends the authors’ previous work on observability of discrete-event systems by taking time into consideration. In a timed discrete-event system, events must occur within their respective lower and upper time bounds. A supervisor can disable, enable, or force some events to achieve a given control objective. We assume that the supervisor does not observe all events, which is often the case in practice. We generalize the concept of observability to timed discrete-event systems and show that it characterizes the existence condition for a supervisor. We also generalize normality, a stronger version of observability, to timed discrete-event systems, which has nice properties that are absent in observability. We then derive conditions under which observability and normality are equivalent. We propose two methods to synthesize a supervisor, a direct approach and an indirect approach. An example is given to illustrate the results.

I. INTRODUCTION

Since the theory of supervisory control was introduced about 10 years ago in [13], many ideas have been developed, notably the concepts of controllability [13], [14], observability [3], [6], [8], and normality [8], [9], [10]. These concepts were, however, defined only for untimed discrete-event systems, in which the occurrence times of events were not modeled. As the theory advances and applications increase, there is a need to generalize these concepts to timed discrete-event systems, in which lower and upper time bounds of events are specified and events must occur within their respective lower and upper time bounds. In [1], controllability is generalized to timed discrete-event systems and supervisory control is discussed under the assumption of full event observation. In practice, however, not all events are observable due to limitations on detection and communication. In such cases, observability becomes relevant, because a supervisor must control the system based on partial observation of events. This paper is devoted to supervisory control under partial observation.

We begin with a brief review of the definitions on timed discrete-event systems and the definitions of controllability of [1] in Section II. In Section III, we formalize supervisory control under partial observation and discuss the constraints it must satisfy. We also generalize the definition of observability to take time into consideration. We then prove that a supervisor exists if and only if the language to be synthesized is controllable, observable, and \( L_m(G) \)-closed. If the language to be synthesized is not controllable and observable, then the problem of synthesizing an optimal supervisor becomes complicated, partly because, as to be shown in Section IV, the supremal controllable and observable sublanguage of a given maximal legal language need not exist. Two methods will be proposed to solve the synthesis problem. In the first method, a stronger version of observability, called normality, is defined. The supremal controllable and normal sublanguage of a given language exists, and preserves the \( L_m(G) \)-closedness of the language, and hence a supervisor can be synthesized.

II. PRELIMINARIES

In this section, we briefly review the relevant results of [1] on untimed discrete-event systems. Following the notation and definitions of [1], a timed discrete-event system is modeled by an automaton

\[ G = (\Sigma, Q, \delta, q_0, Q_u) \]

where the events are partitioned into 1) prospective events \( \Sigma_{\text{pro}} \) that have finite upper time bounds, 2) remote events \( \Sigma_{\text{rem}} \) whose upper time bounds are infinite, and 3) tick of the global clock:

\[ \Sigma = \Sigma_{\text{pro}} \cup \Sigma_{\text{rem}} \cup \{\text{tick}\} \]

Remark 1: To exclude the physically unrealistic possibility that events in \( \Sigma - \{\text{tick}\} \) occur infinitely often during one unit of time, we require that \( G \) be activity loop free [1], that is,

\[ (\forall q \in Q)(\forall s \in (\Sigma - \{\text{tick}\})^*)(\exists(t, q) \neq q. \]

Remark 2: If \( G \) is constructed from the activity transition graph \( G_{\text{act}} \) and time bounds on events as shown in [1], then the advance of time will never stop. Under the assumption of activity loop freedom, this is equivalent to saying that \( G \) is type-I blocking free [11], that is,

\[ (\forall s \in L(G))(\exists L(\Sigma))s \notin \emptyset \]

where \( L(\Sigma) = \{u \in \Sigma: u \in T\} \).

Remak 3: Again, if \( G \) is constructed from \( G_{\text{act}} \) and time bounds on events, then the fact that no tick is possible after a string implies that some prospective events must be possible after that string, that is,

\[ (\forall s \in L(G))(\exists i \in \Sigma_{\text{pro}})(\exists i \in \Sigma_{\text{pro}})(\exists i \in \Sigma_{\text{pro}})(\exists i \in \Sigma_{\text{pro}})(\exists i \in \Sigma_{\text{pro}}) \]

In this paper, we assume that \( G \) is constructed from \( G_{\text{act}} \) and time bounds on events.

Example 1: Consider a single track linking two stations as depicted in Fig. 1. To control the movement of trains, three traffic lights are installed along the track. Also installed are four detectors to observe the movement of trains. The track is divided accordingly into four sections: \( S_1, S_2, S_3, \) and \( S_4 \).

Suppose there are two trains \( T_1 \) and \( T_2 \) scheduled to depart from Station 1 to Station 2. For \( T_i, i = 1, 2 \) the designated events are as follows:

\[ \sigma_{11}: T_1, \quad \sigma_{12}: T_1, \quad \sigma_{13}: T_1, \quad \sigma_{14}: T_1, \quad \sigma_{15}: T_1, \]

\[ \sigma_{21}: T_2, \quad \sigma_{22}: T_2, \quad \sigma_{23}: T_2, \quad \sigma_{24}: T_2, \quad \sigma_{25}: T_2, \]

\[ \sigma_{31}: T_3, \quad \sigma_{32}: T_3, \quad \sigma_{33}: T_3, \quad \sigma_{34}: T_3, \quad \sigma_{35}: T_3, \]

\[ \sigma_{41}: T_4, \quad \sigma_{42}: T_4, \quad \sigma_{43}: T_4, \quad \sigma_{44}: T_4, \quad \sigma_{45}: T_4, \]

The authors would like to acknowledge the support of the National Science Foundation of China under Grant 69232032, the National Science and Engineering Research Council of Canada under Grant A-7399, and the Ford Motor Company.
The activity transition graphs $G^a_{act}$ (for $T^a$) is given in Fig. 2. The lower time bound $l_s$ and upper time bound $u_s$ for events $\sigma$ are

$$\begin{align*}
l_s &= 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 2 \quad 1, \\
u_s &= \infty \quad \infty \quad \infty \quad \infty \quad \infty \quad 1 \quad 2 \quad 1.
\end{align*}$$

Hence the prescriptive events are

$$\Sigma_{pap} = \{ \sigma_{11}, \sigma_{16}, \sigma_{18}, \sigma_{20}, \sigma_{24}, \sigma_{26}, \sigma_{28}, \sigma_{29} \}$$

and the remote events are

$$\Sigma_{rem} = \{ \sigma_{11}, \sigma_{12}, \sigma_{15}, \sigma_{21}, \sigma_{23}, \sigma_{22}, \sigma_{25} \}.$$ 

Simple time bounds are used here for the sole purpose of limiting the number of states. They do, however, illustrate the advantages of the timed model (cf. Section VI). More general time bounds could be used at the expense of more computation.

From $G^a_{act}$ and the time bounds, we can construct the timed discrete-event system $G$ describing the concurrent movement of $T_1$ and $T_2$, which has 256 states.

To introduce control, let $\Sigma_{hmb} \subseteq \Sigma$ be the set of prohibitible (or controllable) events, $\Sigma_{for} \subseteq \Sigma$ be the set of forcible events, and $\Sigma_{unc} = (\Sigma_{rem} \cup \Sigma_{pap}) - \Sigma_{hmb}$ be the set of uncontrollable events. The following definition [11] generalizes the definition in [13].

**Definition 1:** Let $K \subseteq L(G)$. We define $K$ to be controllable (with respect to $L(G)$) if, for all $s \in K$,

1. $\Sigma_{hmb}(s) \cap \Sigma_{for} = \emptyset$.
2. $\Sigma_{hmb}(s) \cap \Sigma_{for} = \emptyset$.

In other words, given history $s \in K$, if there are no forcible events available then $\text{tick}$ is not controllable; if forcible events are available, then $\text{tick}$ can be postponed. In proofs, we will often use the following equivalent definition for controllability.

1. $(\forall s \in K) (\Sigma_{hmb}(s) \cap \Sigma_{for} \subseteq \Sigma_{hmb}(s))$.
2. $(\forall s \in K) (\Sigma_{hmb}(s) \cap \Sigma_{for} = \emptyset \land \text{tick} \in \Sigma_{L(G)}(s))$.

**III. Observability**

In practice, a supervisor may not be able to detect the occurrences of all events. When that happens, the existence of a supervisor is no longer guaranteed by controllability alone: we need to introduce the concept of observability. Observability for untimed discrete-event systems was introduced in [3], [8]. In this paper, we generalize this concept to timed discrete-event systems.

To this end, we denote the observable events from $\Sigma_{o} \subseteq \Sigma$ and the unobservable events from $\Sigma_{uo} = \Sigma - \Sigma_{o}$. A projection $P: \Sigma^* \to \Sigma^*_o$ is defined inductively as follows.

$$P \epsilon = \epsilon, \quad \epsilon \text{ is the empty string}$$

$$P(\sigma \sigma') = \begin{cases} 
Ps, & \text{if } \sigma \in \Sigma_{uo} \\
(|Ps|)_{\sigma}, & \text{if } \sigma \in \Sigma_{o}.
\end{cases}$$

In other words, if a sequence of events $s$ occurred in $G$, a supervisor can only observe $Ps$. Therefore, the supervisor is now described by a map $P: L(G) \to 2^{\Sigma_{o}}$. The language generated by $\gamma$, $L(G, \gamma)$ (called closed behavior), is defined inductively as follows.

1. $\epsilon \in L(G, \gamma)$;
2. If $s \in L(G, \gamma)$, $\sigma \in \gamma(Ps)$, and $\sigma \in L(G)$, then $\sigma \in L(G, \gamma)$; and
3. No other strings belong to $L(G, \gamma)$.

The language marked by $\gamma$ (called marked behavior) is defined as

$$L_m(G, \gamma) = L(G, \gamma) \cap L_m(G).$$

The supervisor $\gamma$ is said to be nonblocking if

$$L_m(G, \gamma) = L(G, \gamma).$$

We say a supervisor is admissible if it satisfies the following two requirements. First, no uncontrollable events can be disabled; and second, if $\text{tick}$ is physically possible and no forcible events can preempt it, then it cannot be disabled. Formally, for all $s \in L(G, \gamma)$, we require

1. $\Sigma_{hmb}(s) \subseteq \gamma(Ps)$; and
2. $\Sigma_{L(G)}(s) \cap \gamma(Ps) \cap \Sigma_{for} = \emptyset \land \text{tick} \in \Sigma_{L(G)}(s) \Rightarrow \text{tick} \in \gamma(Ps)$.

The system supervised under an admissible supervisor cannot have type 1 blocking as noted in the following proposition.

**Proposition 1 ([12]):** For all $s \in L(G, \gamma)$

$$\Sigma_{L(G)}(s) \cap \gamma(Ps) \neq \emptyset.$$ 

We show that for a supervisor to make a correct decision (i.e., to synthesize a language $K$ based on partial observation, the following observability condition must be satisfied.

**Definition 2:** Let $K \subseteq L(G)$. We define $K$ to be observable (with respect to $L(G)$) if, for all $s, s' \in \Sigma^*$

$$Ps = Ps' \Rightarrow \text{cons}(s, s')$$

where $\text{cons}(s, s')$ is true if and only if

1. $(\forall s \in K) (\Sigma_{hmb}(s) \cap \Sigma_{for} \subseteq \Sigma_{hmb}(s))$, and
2. $(\forall s \in K) (\Sigma_{hmb}(s) \cap \Sigma_{for} = \emptyset \land \text{tick} \in \Sigma_{L(G)}(s))$.

In words, two strings $s$ and $s'$ satisfy $\text{cons}(s, s')$ if and only if $s$ and $s'$ are consistent with respect to one-step continuations in $K$. Observability requires that if two strings look the same to a supervisor, then they must be consistent. This definition is a modification of observability as defined in [8].

This observability condition together with the controllability condition guarantees the existence of a supervisor as shown in the following theorem.

**Theorem 1([12]):** Let $K \subseteq L_m(G)$ be a nonempty language. There exists a nonblocking supervisor $\gamma$ such that $L_m(G, \gamma) = K$ if only if the following three conditions are all satisfied.

1. $K$ is controllable with respect to $L(G)$;
2. $K$ is observable with respect to $L(G)$; and
3. $K$ is $L_m(G)$-closed.

If we are only interested in the closed behavior, then we have the following corollary.

**Corollary 1:** Let $K \subseteq L(G)$ be a nonempty language. There exists a supervisor $\gamma$ such that $L(G, \gamma) = K$ if and only if the following three conditions are all satisfied.

1. $K$ is controllable with respect to $L(G)$;
2. $K$ is observable with respect to $L(G)$; and
3. $K$ is closed.

**IV. Normality**

The language describing a control objective is not very likely to be controllable and observable when a control objective is first specified, because the objective is usually laid down independently of the consideration of controllability and observability of events. When this happens, the original control objective is not achievable by a supervisor. So we will try to achieve the largest possible part of the original control objective. This means finding, if possible, the largest sublanguage of the original language that is controllable and
observable. Such a supervisor is "optimal" in the sense that a system generating a larger language can "run" faster when driven by the same stochastic process generating event lifetimes, if a certain "fairness" condition is satisfied. The reader is referred to [7] for this result.

Let $\emptyset \neq E \subseteq L_m(G)$ be the language describing the original control objective, called the maximal legal language. We assume that $E$ is $L_m(G)$-closed.

The supremal controllable and observable sublanguage of $E$ does not exist in general (see [12, example 2]) because the set of observable languages is not closed under union. To overcome this difficulty, we first modify the definition of normality introduced in (8).

**Definition 3:** Let $K \subseteq L(G)$. We define $K$ to be normal (with respect to $L(G)$) if, for all $s \in \Sigma^*$,

$$s \in L(G) \land Ps \in PK \Rightarrow s \in K.$$ 

Since $s \in K \Rightarrow s \in L(G) \land Ps \in PK$ is always true, $K$ is normal if and only if $L(G) \land P^{-1}P_K$ is. Therefore, if $K$ is normal, we can check whether a string $s \in L(G)$ is in $K$ by checking whether its projection $Ps$ is in $P_K$. In other words, information on occurrences of unobservable events is not needed in deciding whether $s \in K$.

Hence, we expect that normality will be stronger than observability.

**Proposition 2 ([12]):** If $K$ is normal, the $K$ is observable.

The set of normal languages is algebraically better behaved than that of observable languages, in the sense that it is closed under arbitrary unions.

**Theorem 2 ([12]):** The set $C(N) = \{K \subseteq E : K$ is controllable and normal}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $C(N)$, $\sup C(N)$, exists and belongs to $C(N)$.

Assume that all events are observable. Then normality is automatically satisfied. So we have the following corollary.

**Corollary 2:** The set $C(E) = \{K \subseteq E : K$ is controllable}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $C(E)$, $\sup C(E)$, exists and belongs to $C(E)$.

Similarly, by assuming that all events are controllable, we have the following corollary.

**Corollary 3:** The set $N(E) = \{K \subseteq E : K$ is normal}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $N(E)$, $\sup N(E)$, exists and belongs to $N(E)$.

From the above results, we propose a direct approach to synthesize a supervisor when $E$ is not controllable and observable. The approach is to synthesize a supervisor for $\sup C(N(E))$. For this to work, however, we need to know that $\sup C(N(E))$ is also $L_m(G)$-closed.

**Proposition 4 ([12]):** If $E$ is $L_m(G)$-closed, then $\sup C(N(E))$ is also $L_m(G)$-closed.

Proposition 2 shows that normality is stronger than observability. But, how strong is it? The following proposition partially answers this question. It states that if tick and all controllable events are observable, then normality is equivalent to observability in the presence of controllability.

**Proposition 4 ([12]):** Assume $\Sigma_{ob} \cup \{tick\} \subseteq \Sigma_e$. If $K$ is controllable and observable, then $K$ is controllable and normal.

V. MODIFICATION FOR PARTIAL OBSERVABILITY

In the previous section, a direct approach was proposed to synthesize a supervisor if $E$ is not controllable and observable, that is, a supervisor was synthesized for $\sup C(N(E))$. In this section, we propose a different approach: we will first synthesize a supervisor for $\sup C(N)$ under the assumption of full observation and then modify the supervisor for partial observation. We show that, under certain conditions, this indirect approach leads to a supervisor with a larger closed behavior. Furthermore, as shown in [4], when the resulting supervisor is implemented on-line, the computational complexity for updating control after observing a new event is linear with respect to the number of states in the supervisor for full observation. In other words, in the worst case, the computational complexity of the indirect approach at each step is of the order $O(|X|)$, in comparison with the total complexity of $O(2^{|X|})$ for the direct approach, where $|X|$ is the number of states in the refined generator of $K$.

We need the following preliminary results.

**Proposition 5 ([12]):** The set $\bar{C}(E) = \{K \subseteq E : K$ is closed, controllable, and normal}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $\bar{C}(E)$, $\sup C(E)$, exists, and belongs to $\bar{C}(E)$.

By this proposition, we do not need to distinguish $\sup C(E)$ and $\sup C(N)$.

**Corollary 4:** The set $\bar{C}(E) = \{K \subseteq E : K$ is closed and controllable}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $\bar{C}(E)$, $\sup C(E)$, exists, and is given by $\sup C(E) = \sup C(\bar{E})$.

**Corollary 5:** The set $\bar{N}(E) = \{K \subseteq \bar{E} : K$ is closed and normal}$

is nonempty and closed under arbitrary unions. Therefore, the supremal element of $\bar{N}(E)$, $\sup \bar{N}(E)$, exists, and is given by $\sup \bar{N}(E) = \sup \bar{N}(\bar{E})$.

Since $\sup C(\bar{E}) = \sup C(\bar{E})$ is closed and controllable, we can synthesize a supervisor with full observation $\gamma' : L(G) \rightarrow 2^\Sigma$ such that $L(G, \gamma') = \sup C(\bar{E})$.

where $L(G, \gamma')$ is defined in the same way for $L(G, \gamma)$ except that $Ps = s$.

Under partial observation, we modify $\gamma'$ to a supervisor $\gamma : PL(G) \rightarrow 2^\Sigma$ as follows. Let $\{t\} = P^{-1}(t) \land L(G, \gamma')$ be the set of strings in $L(G, \gamma')$ having the projection $t$ and \begin{align*}
\gamma(t) &= U_{e \in \{t\}} (\gamma'(s) \cup (\Sigma \setminus \Sigma_{L(G)}(s)))
\end{align*}

A similar modification was introduced in [4] for untimed discrete-event systems. However, for a timed discrete-event system, the issues of forficible events and nonblocking need to be dealt with differently.

To this end, we define coherence as follows.

**Definition 4:** Let $K \subseteq L(G)$. We define $K$ to be coherent (with respect to $L(G)$) if, for all $s \in K$,

$$tick \in \Sigma_{L(G)}(s) \Rightarrow (\exists s', s'' \in P^{-1}Ps \land K) : (\Sigma_K(s') \land \Sigma_{for}) \land (\Sigma_{L(G)}(s') \land \Sigma_{for}) = \Sigma_{L(G)}(s'') \land \Sigma_{for}.$$ 

In words, if $K$ is coherent, then for every string $s$ that can be followed by tick in $L(G)$, the set of forficible events that are feasible (or legal) under any string having the same projection as $s$ is the same.
Proposition 6 ([12]): If $\gamma'$ is an admissible supervisor and $L(G, \gamma')$ is coherent, then $\gamma$ is also an admissible supervisor.

This modified supervisor $\gamma$ may block. To guarantee nonblocking, it is convenient to introduce the additional condition of livelock-freedom [5]. First, let us recall the definition of livelock-freedom given in [5].

Definition 5: Let $K \subseteq \Sigma^*$. We define $K$ to be livelock free if, for all $s \in K$

$$(\exists n \in N)(\forall t \in \Sigma^*)[t \geq n \land st \in K \Rightarrow (\exists u \in \Sigma^*) \cdot (s \leq u \leq st \land u \in K)]$$

where $N$ denotes the set of natural numbers and $s \leq u$ denotes that $s$ is a prefix of $u$.

In words, if $K$ is livelock-free, then every infinite chain of strings in $K$ visits the set of marker states infinitely often.

Proposition 7 ([12]): If $L_m(G)$ is livelock-free and $\gamma$ is admissible, then $\gamma$ is nonblocking.

The implication of the above proposition is that, under the assumption of livelock-freedom, any admissible supervisor is nonblocking. Finally, we can prove that the modified supervisor generates only legal behavior that contains at least $sup C N(E)$.

Theorem 3 ([12]): The language marked by the modified supervisor $\gamma$ is bounded by

$${sup C N(E) \leq L_m(G, \gamma) \leq E}$$

To conclude this section, we note that $L(G, \gamma)$ obtained this way may not be maximal (see Example 3 in [12]).

VI. EXAMPLE

We continue our discussion of the example in Section II. Our control objective is to prevent two trains from colliding. So the corresponding maximal legal language $E$ can be obtained by deleting states in $G$ corresponding to two trains being in the same section at the same time.

Because of the locations of the traffic lights and detectors, the controllable and observable events are

$${\Sigma}_{\text{obs}} = \{\sigma_{11}, \sigma_{13}, \sigma_{15}, \sigma_{21}, \sigma_{23}, \sigma_{25}\}$$

$${\Sigma}_{\text{o}} = \{\sigma_{11}, \sigma_{13}, \sigma_{15}, \sigma_{16}, \sigma_{21}, \sigma_{22}, \sigma_{25}, \sigma_{26}\}.$$

Apology and Correction to “Process Control and Machine Learning: Rule-Based Incremental Control”

Dominique Luzeaux and Bertrand Zavidovique

APOLOGY AND CORRECTION

It has come to our attention that the name of the second author was omitted from the above paper appearing in the June, 1994, issue of Transactions on Automatic Control, pp. 1166-1171. Due to an apparent error in transcription, a former editorial assistant omitted the name of Bertrand Zavidovique, who was the second author on this paper. Through a regrettable and remarkably coincidental sequence of oversights, the original error was not corrected either in the typesetting of the article or in the proofreading of the galleys by the first author. The Transactions offers a sincere apology to Prof. Zavidovique for this mistake and hereby provides publication of the title and authors as it should have appeared in June. The paper in question appears in vol. 39, no. 6.