Delay-Robustness in Distributed Control of Timed Discrete-Event Systems Based on Supervisor Localization

(October 26, 2018)

Renyuan Zhang, Kai Cai, Yongmei Gan, W.M. Wonham

Abstract

Recently we studied communication delay in distributed control of untimed discrete-event systems based on supervisor localization. We proposed a property called delay-robustness: the overall system behavior controlled by distributed controllers with communication delay is logically equivalent to its delay-free counterpart. In this paper we extend our previous work to timed discrete-event systems, in which communication delays are counted by a special clock event *tick*. First, we propose a timed channel model and define timed delay-robustness; for the latter, a polynomial verification procedure is presented. Next, if the delay-robust property does not hold, we introduce *bounded* delay-robustness, and present an algorithm to compute the *maximal* delay bound (measured by number of $tick$s) for transmitting a channeled event. Finally, we demonstrate delay-robustness on the example of an under-load tap-changing transformer.

Keywords

Timed Discrete-Event Systems, *Distributed Supervisory Control*, *Supervisor Localization*, *Delay-robustness.*

R. Zhang is with School of Automation, Northwestern Polytechnical University, China; K. Cai is with Urban Research Plaza, Osaka City University, Japan; Y. Gan is with School of Electrical Engineering, Xian Jiaotong University, China; and W.M. Wonham is with the Systems Control Group, Department of Electrical and Computer Engineering, University of Toronto, Canada. (Emails: ryzhang@nwpu.edu.cn; kai.cai@info.eng.osaka-cu.ac.jp; ymgan@mail.xjtu.edu.cn; wonham@control.utoronto.ca).

I. INTRODUCTION

For distributed control of discrete-event systems (DES), supervisor localization was recently proposed [1–4] which decomposes a monolithic supervisor or a heterarchical array of modular supervisors into local controllers for individual agents. Collective local controlled behavior is guaranteed to be globally optimal and nonblocking, assuming that the shared events among local controllers are communicated instantaneously, i.e. with no delay. In practice, however, local controllers are linked by a physical communication network in which delays may be inevitable. Hence, for correct implementation of the local controllers obtained by localization, it is essential to model and appraise communication delays.

In [5] and its conference precursor [6], we studied communication delays among local controllers for untimed DES. In particular, we proposed a new concept called *delay-robustness*, meaning that the systemic behavior of local controllers interconnected by communication channels subject to unbounded delays is logically equivalent to its delay-free counterpart. Moreover, we designed an efficient procedure to verify for which channeled events the system is delay-robust. If for a channeled event r the system fails to be delay-robust, there may still exist a finite bound for which the system can tolerate a delay in r. In unitimed DES, however, there lacks a *temporal* measure for the delay bound (except for counting the number of occurrences of untimed events).

In this paper and its conference antecedent[7], we extend our study on delay-robustness to the timed DES (or TDES) framework proposed by Brandin and Wonham [8, 9]. In this framework, the special clock event *tick* provides a natural way of modeling communication delay as temporal behavior. We first propose a timed channel model for transmitting each channeled event, which effectively measures communication delay by the number of *tick* occurrences, with no *a priori* upper bound, so that the channel models *unbounded* delay. We then define timed delay-robustness with respect to the timed channel, thus extending its untimed counterpart [5, 6] in two respects: (1) the system's temporal behavior is accounted for, and (2) timed controllability is required. A polynomial algorithm is presented to verify timed delay-robustness according to this new definition.

If the delay-robust property fails to hold, we introduce *bounded* delay-robustness and present a corresponding verification algorithm. In particular, the algorithm computes the *maximal* delay bound (in terms of number of $tick$s) for transmitting a channeled event, i.e. the largest delay that can be tolerated without violating the system specifications. These concepts and the corresponding algorithms are illustrated for the case of an under-load tap-changing transformer (ULTC).

Distributed/decentralized supervisory control with communication delay has been widely studied for

untimed DES (e.g. [10–18]). In particular in [11, 15], the existence of distributed controllers in the unbounded delay case is proved to be undecidable; and in [11–14, 16], distributed controllers are synthesized under the condition that communication delay is bounded. We also note that Sadid et al. [18] propose a way to verify robustness of a given synchronous protocol with respect to a fixed or a finitely-bounded delay, as measured by the number of untimed events occurring during the transmitting process. We refer to [5, 6] for a detailed review of these works and their differences from our approach. Communication delay in timed DES, on the other hand, has (to our knowledge) received little attention. The present work is based on our previous research on timed supervisor localization [3, 4].

The paper is organized as follows. Sect. II provides a review of the Brandin-Wonham TDES framework and recalls supervisor localization for TDES. In Sect. III we introduce a timed channel model, and present the concept and verification algorithm for timed delay-robustness.In Sect. IV we define bounded delay-robustness, and present an algorithm to compute the maximal delay bound. These concepts and the corresponding algorithms are demonstrated in Sect. V on the distributed control problem for an under-load tap-changing transformer (ULTC) with communications. Conclusions are presented in Sect. VI.

II. Distributed Control by Supervisor Localization of TDES

A. Preliminaries on TDES

The TDES model proposed by Brandin and Wonham [8] is and extension of the untimed DES generator model of the Ramadge-Wonham framework [9]. A TDES is given by

$$\mathbf{G} := (Q, \Sigma, \delta, q_0, Q_m). \tag{1}$$

Here Q is the finite set of *states*; Σ is the finite set of events including the special event $tick$, which represents "tick of the global clock"; $\delta : Q \times \Sigma \to Q$ is the (partial) *state transition function* (this is derived from the corresponding activity transition function; the reader is referred to the detailed transition rules given in [8, 9]); q_0 is the *initial state*; and $Q_m \subseteq Q$ is the set of *marker states*. The transition function is extended to $\delta : Q \times \Sigma^* \to Q$ in the usual way. The *closed behavior* of $\mathbf{G}$ is the language $L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_0, s)!\}$ and the *marked behavior* is $L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) | \delta(q_0, s) \in Q_m\} \subseteq L(\mathbf{G})$. We say that $\mathbf{G}$ is *nonblocking* if $\bar{L}_m(\mathbf{G}) = L(\mathbf{G})$, where $\bar{\cdot}$ denotes *prefix closure* [9].

Let Σ^* be the set of all finite strings, including the empty string ϵ. For $\Sigma' \subseteq \Sigma$, the *natural projection*

$P : \Sigma^* \to \Sigma'^*$ is defined by

$$P(\epsilon) = \epsilon;$$
$$P(\sigma) = \begin{cases} \epsilon, & \text{if } \sigma \notin \Sigma', \\ \sigma, & \text{if } \sigma \in \Sigma'; \end{cases} \tag{2}$$
$$P(s\sigma) = P(s)P(\sigma), \quad s \in \Sigma^*, \sigma \in \Sigma.$$

As usual, P is extended to $P : Pwr(\Sigma^*) \to Pwr(\Sigma'^*)$, where $Pwr(\cdot)$ denotes powerset. Write $P^{-1} : Pwr(\Sigma'^*) \to Pwr(\Sigma^*)$ for the *inverse-image function* of P.

To adapt the TDES $\mathbf{G}$ in (1) for supervisory control, we first designate a subset of events, denoted by $\Sigma_{hib} \subseteq \Sigma$, to be the *prohibitible* events which can be disabled by an external supervisor. Next, and specific to TDES, we bring in another category of events, called the *forcible* events, which can *preempt* event $tick$; let $\Sigma_{for} \subseteq \Sigma$ denote the set of forcible events. Note that $tick \notin \Sigma_{hib} \cup \Sigma_{for}$. Now it is convenient to define the *controllable* event set $\Sigma_c := \Sigma_{hib} \dot\cup \{tick\}$. The *uncontrollable* event set is $\Sigma_u := \Sigma - \Sigma_c$.

We introduce the notion of (timed) controllability as follows. For a string $s \in L(\mathbf{G})$, define $Elig_{\mathbf{G}}(s) := \{\sigma \in \Sigma | s\sigma \in L(\mathbf{G})\}$ to be the subset of events 'eligible' to occur (i.e. defined) at the state $q = \delta(q_0, s)$. Consider an arbitrary language $F \subseteq L(\mathbf{G})$ and a string $s \in \overline{F}$; similarly define the eligible event subset $Elig_F(s) := \{\sigma \in \Sigma | s\sigma \in \overline{F}\}$. We say F is *controllable* with respect to $\mathbf{G}$ if, for all $s \in \overline{F}$,

$$Elig_F(s) \supseteq \begin{cases} Elig_{\mathbf{G}}(s) \cap (\Sigma_u \dot\cup \{tick\}) \\ \qquad\qquad \text{if} \quad Elig_F(s) \cap \Sigma_{for} = \emptyset, \\ Elig_{\mathbf{G}}(s) \cap \Sigma_u \\ \qquad\qquad \text{if} \quad Elig_F(s) \cap \Sigma_{for} \neq \emptyset. \end{cases} \tag{3}$$

Whether or not F is controllable, we denote by $\mathcal{C}(F)$ the set of all controllable sublanguages of F. Then $\mathcal{C}(F)$ is nonempty, closed under arbitrary set unions, and thus contains a unique supremal (largest) element denoted by $sup\mathcal{C}(F)$ [8, 9]. Now consider a specification language $E \subseteq \Sigma^*$ imposed on the timed behavior of $\mathbf{G}$; E may represent a logical and/or temporal requirement. Let the TDES

$$\mathbf{SUP} = (X, \Sigma, \xi, x_0, X_m) \tag{4}$$

be the corresponding *monolithic supervisor* that is optimal (i.e., maximally permissive) and nonblocking in the following sense: $\mathbf{SUP}$'s marked language $L_m(\mathbf{SUP})$ is

$$L_m(\mathbf{SUP}) = sup\mathcal{C}(E \cap L_m(\mathbf{G})) \subseteq L_m(\mathbf{G})$$

and moreover its closed language $L(\mathbf{SUP})$ is $L(\mathbf{SUP}) = \overline{L}_m(\mathbf{SUP})$.

B. Supervisor Localization of TDES

In this subsection, we introduce the supervisor localization procedure, which was initially proposed in the untimed DES framework [1] and then adapted to the TDES framework [3, 4]. By this procedure, a set of *local controllers* and *local preemptors* is obtained and shown to be 'control equivalent' to the monolithic supervisor $\mathbf{SUP}$ in (4). By allocating these constructed local controllers and preemptors to each component agent, we build a distributed supervisory control architecture.

Let TDES $\mathbf{G}$ in (1) be the plant to be controlled and E be a specification language. As in [9], synthesize the monolithic optimal and nonblocking supervisor $\mathbf{SUP}$. Supervisor $\mathbf{SUP}$'s control action includes (i) disabling prohibitible events in Σ_{hib} and (ii) preempting $tick$ via forcible events in Σ_{for}. By the supervisor localization procedure, a set of local controllers $\{\mathbf{LOC}^C_\alpha \text{defined on } \Sigma_\alpha | \alpha \in \Sigma_{hib}\}$ and a set of local preemptors $\{\mathbf{LOC}^P_\beta \text{defined on } \Sigma_\beta | \beta \in \Sigma_{for}\}$ are constructed. These $\mathbf{LOC}^C_\alpha$ and $\mathbf{LOC}^P_\beta$ are all TDES as in (1), and proved to be control equivalent to $\mathbf{SUP}$ (with respect to $\mathbf{G}$) in the following sense:

$$L(\mathbf{G}) \cap \Big(\bigcap_{\alpha \in \Sigma_{hib}} P_\alpha^{-1} L(\mathbf{LOC}^C_\alpha) \Big) \cap \Big(\bigcap_{\beta \in \Sigma_{for}} P_\beta^{-1} L(\mathbf{LOC}^P_\beta) \Big) = L(\mathbf{SUP}), \tag{5}$$

$$L_m(\mathbf{G}) \cap \Big(\bigcap_{\alpha \in \Sigma_{hib}} P_\alpha^{-1} L_m(\mathbf{LOC}^C_\alpha) \Big) \cap \Big(\bigcap_{\beta \in \Sigma_{for}} P_\beta^{-1} L_m(\mathbf{LOC}^P_\beta) \Big) = L_m(\mathbf{SUP}). \tag{6}$$

Here $P_\alpha : \Sigma^* \to \Sigma^*_\alpha$ and $P_\beta : \Sigma^* \to \Sigma^*_\beta$ are the natural projections as in (2).

Now, using the constructed local controllers and local preemptors, we build a distributed supervisory control architecture (without communication delay) for a multi-agent TDES plant. Consider that the plant $\mathbf{G}$ consists of N component TDES $\mathbf{G}_i$ ($i \in \mathcal{N} := \{1, 2, ..., N\}$), each with event set $\Sigma_i \ni tick$. For simplicity assume $\Sigma_i \cap \Sigma_j = \{tick\}$, for all $i \neq j \in \mathcal{N}$; namely the agents $\mathbf{G}_i$ are independent except for synchronization on the global event $tick$. As a result, the marked and closed behaviors of the composition of $\mathbf{G}_i$ coincide with those of their synchronous product [9], and thus we use synchronous product instead of composition to combine TDES together, i.e. $\mathbf{G} = \mathop{||}_{i \in \mathcal{N}} \mathbf{G}_i$ where $||$ denotes the synchronous product of TDES.[1]

A convenient *allocation policy* of local controllers/preemptors is the following. For a fixed agent $\mathbf{G}_i$, let $\Sigma_{i,for}, \Sigma_{i,hib} \subseteq \Sigma_i$ be its forcible event set and prohibitible event set, respectively. Then allocate to $\mathbf{G}_i$ the set of local controllers $\mathbf{LOC}^C_i := \{\mathbf{LOC}^C_\alpha | \alpha \in \Sigma_{i,hib}\}$ and the set of local preemptors $\mathbf{LOC}^P_i :=$

[1]The closed and marked behaviors of $\mathbf{TDES} = \mathbf{TDES1} \,||\, \mathbf{TDES2}$ are $L(\mathbf{TDES}) = L(\mathbf{TDES1}) \,||\, L(\mathbf{TDES2})$ and $L_m(\mathbf{TDES}) = L_m(\mathbf{TDES1}) \,||\, L_m(\mathbf{TDES2})$, where $||$ denotes the synchronous product of languages [9].

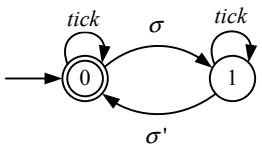

Fig. 1. Timed channel model $\mathbf{CH}(j, \sigma, i)$ for transmitting event σ from $\mathbf{G}_j$ to $\mathbf{G}_i$ with indefinite (i.e. unspecified) time delay.

$\{\mathbf{LOC}_\beta^P | \beta \in \Sigma_{i,for}\}$. This allocation creates a distributed control architecture for the multi-agent plant $\mathbf{G}$, in which each agent $\mathbf{G}_i$ is controlled by its own local controllers/preemptors, while interacting with other agents through communication of shared events. For agent $\mathbf{G}_i$, the set of *communication events* that need to be imported from other agents is

$$\Sigma_{com,i} := \Big(\bigcup_{\alpha \in \Sigma_{i,hib}} \Sigma_\alpha - \Sigma_i \Big) \cup \Big(\bigcup_{\beta \in \Sigma_{i,for}} \Sigma_\beta - \Sigma_i \Big) \tag{7}$$

where Σ_α and Σ_β are the event sets of $\mathbf{LOC}_\alpha^C$ and of $\mathbf{LOC}_\beta^P$ respectively.

However, this distributed control architecture is built under the assumption that the communication delay of communication events is negligible. While simplifying the design of distributed controllers, this assumption may be unrealistic in practice, where controllers are linked by a physical network subject to delay. In the rest of this paper, we investigate how the communication delay affects the synthesized local control strategies and the corresponding overall system behavior.

III. Timed Delay-Robustness

Consider event communication between a pair of agents $\mathbf{G}_i$ and $\mathbf{G}_j$ ($i, j \in \mathcal{N}$): specifically, $\mathbf{G}_j$ sends an event σ to $\mathbf{G}_i$. Let Σ_j be the event set of $\mathbf{G}_j$ and $\Sigma_{com,i}$ as in (7) the set of communication events that $\mathbf{G}_i$ imports from other agents. Then the set of events that $\mathbf{G}_j$ sends to $\mathbf{G}_i$ is

$$\Sigma_{j,com,i} := \Sigma_j \cap \Sigma_{com,i}. \tag{8}$$

We thus have event $\sigma \in \Sigma_{j,com,i}$.

Now consider the timed channel model $\mathbf{CH}(j, \sigma, i)$ for σ transmission displayed in Fig. 1. $\mathbf{CH}(j, \sigma, i)$ is a 2-state TDES with event set $\{\sigma, \sigma', tick\}$. The transition from state 0 to 1 by σ means that $\mathbf{G}_j$ has sent σ to channel, while the transition from state 1 back to 0 by σ' means that $\mathbf{G}_i$ has received σ from channel. We refer to σ' as the *signal event* of σ, and assign its controllability status to be the same as σ (i.e. σ' is controllable iff σ is controllable). The selfloop transition $tick$ at state 1 therefore counts communication delay of σ transmission: the number of $tick$s that elapses between σ and σ'. Measuring

delay by $tick$ events is a major improvement compared to the untimed channel model we used in [6] where no suitable measure exists to count delay. Later in Sect. IV, with the aid of this measure we will compute useful delay bounds for event communication.

It should be stressed that the number of $tick$ occurrences between σ and σ' is unspecified, inasmuch as the selfloop $tick$ at state 1 may occur indefinitely. In this sense, $\mathbf{CH}(j,\sigma,i)$ models possibly *unbounded* communication delay. Note that $tick$ is also selflooped at state 0; this is not used to count delay, but rather for the technical necessity of preventing the event $tick$ from being blocked when synchronizing $\mathbf{CH}(j,\sigma,i)$ with other TDES. The initial state 0 is marked, signaling each completion of event σ transmission; state 1, on the other hand, is unmarked because the transmission is still ongoing.

The capacity of channel $\mathbf{CH}(j,\sigma,i)$ is 1, meaning that only when the latest occurrence of event σ is received by its recipient $\mathbf{G}_i$, will the channel accept a fresh instance of σ from $\mathbf{G}_j$. Hence, $\mathbf{CH}(j,\sigma,i)$ permits reoccurrence of σ (i.e. $\mathbf{G}_j$ sends σ again) only when it is idle, namely at state 0. The capacity constraint of $\mathbf{CH}(j,\sigma,i)$ can be easily relaxed to allow *multi-capacity* channel models, as we shall see in Remark 1 below. We nevertheless adopt $\mathbf{CH}(j,\sigma,i)$ for its structural simplicity and suitability for clarifying the concept of delay-robustness presented next.

With the channel model $\mathbf{CH}(j,\sigma,i)$, we may describe the *channeled behavior* of the system as follows. Suppose given $\mathbf{G}_k$, $k \in \mathcal{N}$; by localization (see Sect. II-B) $\mathbf{G}_k$ acquires a set of local controllers $\mathbf{LOC}_k^C := \{\mathbf{LOC}_\alpha^C | \alpha \in \Sigma_{k,hib}\}$ and a set of local preemptors $\mathbf{LOC}_k^P := \{\mathbf{LOC}_\beta^P | \beta \in \Sigma_{k,for}\}$.[2] So the local controlled behavior of $\mathbf{G}_k$ is

$$\mathbf{SUP}_k := \mathbf{G}_k \;||\; \Big(\mathop{||}_{\alpha\in\Sigma_{k,hib}} \mathbf{LOC}_\alpha^C\Big) \;||\; \Big(\mathop{||}_{\beta\in\Sigma_{k,for}} \mathbf{LOC}_\beta^P\Big). \tag{9}$$

Observe that when $\mathbf{G}_j$ sends σ to $\mathbf{G}_i$ through $\mathbf{CH}(j,\sigma,i)$, only the recipient $\mathbf{G}_i$'s local behavior $\mathbf{SUP}_i$ is affected because $\mathbf{G}_i$ receives σ' instead of σ due to delay. Hence each transition σ of $\mathbf{SUP}_i$ must be replaced by its signal event σ'; we denote by $\mathbf{SUP}_i'$ the resulting new local behavior of $\mathbf{G}_i$. Now let

$$\mathbf{NSUP} := \mathbf{SUP}_i' \;||\; \Big(\mathop{||}_{k\in\mathcal{N},k\neq i} \mathbf{SUP}_k\Big) \tag{10}$$

[2] For each state state x of each controller $\mathbf{LOC}_\alpha^C$ (resp. preemptor $\mathbf{LOC}_\beta^P$), and each communication event $\sigma \in \Sigma_\alpha - \Sigma_k$ (resp. $\sigma \in \Sigma_\beta - \Sigma_k$), if σ is not defined at x, we add a σ-selfloop, i.e. transition (x,σ,x) to $\mathbf{LOC}_\alpha^C$ (resp. $\mathbf{LOC}_\beta^P$). Now, σ is defined at every state of $\mathbf{LOC}_\alpha^C$ (resp. $\mathbf{LOC}_\beta^P$). With this modification, the new local controllers $\mathbf{LOC}_\alpha^C$ (resp. local preemptors $\mathbf{LOC}_\beta^P$) are also control equivalent to SUP (because $\mathbf{LOC}_\alpha^C$ (resp. $\mathbf{LOC}_\beta^P$) does not disable events σ from other components $\mathbf{G}_j$) and the definition of σ at every state of $\mathbf{LOC}_\alpha^C$ (resp. $\mathbf{LOC}_\beta^P$) is consistent with the assumption that $\mathbf{LOC}_\alpha^C$ (resp. $\mathbf{LOC}_\beta^P$) may receive σ after indefinite communication delay.

and then

$$\mathbf{SUP}' := \mathbf{NSUP} \;||\; \mathbf{CH}(j, \sigma, i). \tag{11}$$

So $\mathbf{SUP}'$ is the channeled behavior of the system with respect to $\mathbf{CH}(j, \sigma, i)$. Note that both $\mathbf{SUP}'$ and $\mathbf{NSUP}$ are defined over $\Sigma' := \Sigma \cup \{\sigma'\}$.

Let $P : \Sigma'^* \to \Sigma^*$ and $P_{ch} : \Sigma'^* \to \{\sigma, tick, \sigma'\}^*$ be natural projections (as in (2)). We define delay-robustness as follows.

Definition 1. Consider that $\mathbf{G}_j$ sends event σ to $\mathbf{G}_i$ through channel $\mathbf{CH}(j, \sigma, i)$. The monolithic supervisor $\mathbf{SUP}$ in (4) is *delay-robust* with respect to $\mathbf{CH}(j, \sigma, i)$ if the following conditions hold:
(i) $\mathbf{SUP}'$ in (11) is *correct* and *complete*, i.e.

$$PL(\mathbf{SUP}') = L(\mathbf{SUP}) \tag{12}$$

$$PL_m(\mathbf{SUP}') = L_m(\mathbf{SUP}) \tag{13}$$

$$(\forall s \in \Sigma'^*)(\forall w \in \Sigma^*)\ s \in L(\mathbf{SUP}')\ \&\ (Ps)w \in L_m(\mathbf{SUP}) \\ \Rightarrow (\exists v \in \Sigma'^*)\ Pv = w\ \&\ sv \in L_m(\mathbf{SUP}') \tag{14}$$

(ii) $P_{ch}^{-1}(L(\mathbf{CH}(j, \sigma, i)))$ is controllable with respect to $L(\mathbf{NSUP})$ and $\{\sigma\}$, i.e.

$$P_{ch}^{-1}L(\mathbf{CH}(j, \sigma, i))\{\sigma\} \;\cap\; L(\mathbf{NSUP}) \subseteq P_{ch}^{-1}L(\mathbf{CH}(j, \sigma, i)) \tag{15}$$

In condition (i) above, 'correctness' of $\mathbf{SUP}'$ means that no P-projection of anything $\mathbf{SUP}'$ can do is disallowed by $\mathbf{SUP}$, while 'completeness' means that anything $\mathbf{SUP}$ can do is the P-projection of something $\mathbf{SUP}'$ can do. In this sense, the channeled behavior $\mathbf{SUP}'$ is 'equivalent' to its delay-free counterpart $\mathbf{SUP}$. Specifically, conditions (12) and (13) state the equality of closed and marked behaviors between $\mathbf{SUP}$ and the P-projection of $\mathbf{SUP}'$; condition (14), which is required for 'completeness', states that if $\mathbf{SUP}'$ executes a string s whose projection Ps in $\mathbf{SUP}$ can be extended by a string w to a marked string of $\mathbf{SUP}$, then $\mathbf{SUP}'$ can further execute a string v whose projection Pv is w and such that sv is marked in $\mathbf{SUP}'$. Roughly, an *observationally consistent inference* about coreachability at the "operating" level of $\mathbf{SUP}'$ can be drawn from coreachability at the abstract (projected) level of $\mathbf{SUP}$.

Condition (ii) of Definition 1 imposes a basic requirement that channel $\mathbf{CH}(j, \sigma, i)$, when combined with $\mathbf{NSUP}$ in (10) to form $\mathbf{SUP}'$, should not entail uncontrollability with respect to σ. We impose condition (ii) no matter whether σ is controllable or uncontrollable. This is because we view the channel $\mathbf{CH}(j, \sigma, i)$ as a hard-wired passive adjunction to the original system, and therefore $\mathbf{CH}(j, \sigma, i)$ cannot

exercise control on σ. In other words, the channel has to 'accept' any event that the rest of the system might execute, whether that event is controllable or uncontrollable. Thus if there is already an instance of σ in the channel (i.e. $\mathbf{CH}(j,\sigma,i)$ at state 1), then reoccurrence of σ will be (unintentionally) 'blocked', causing condition (ii) to fail. This issue persists, albeit in milder form, even if we use channel models of multiple (finite) capacities (see Remark 1 below).

We note that delay-robustness as defined above is an extension, from untimed DES to timed DES, of the concept proposed under the same name in [6] . In particular, the channel model $\mathbf{CH}(j,\sigma,i)$ used in the definition is capable of measuring transmission delay by counting $tick$ occurrences; and condition (ii) in the definition requires controllability for timed DES.

Finally, we present a polynomial algorithm to verify the delay-robustness property. Notice that when (12) and (13) hold, then (14) is identical with the $L_m(\mathbf{SUP}')$-*observer* property of P [19, 20]. The latter may be verified in polynomial time ($O(n^4)$, n the state size of $\mathbf{SUP}'$) by computing the *supremal quasi-congruence* of a nondeterministic automaton derived from $\mathbf{SUP}'$ and P [19, 21].[3] The following is the delay-robustness verification algorithm.

Algorithm 1

1. Check if P is an $L_m(\mathbf{SUP}')$-observer. If no, return *false*.
2. Check if $PL(\mathbf{SUP}') = L(\mathbf{SUP})$ and $PL_m(\mathbf{SUP}') = L_m(\mathbf{SUP})$. If no, return *false*.
3. Check if $P_{ch}^{-1}(L(\mathbf{CH}(j,\sigma,i)))$ is controllable with respect to $L(\mathbf{NSUP})$ and $\{\sigma\}$. If no, return *false*.
4. Return *true*.

If Step 1 above ($O(n^4)$ complexity) is successful, i.e. P is indeed an $L_m(\mathbf{SUP}')$-observer, then Step 2 of computing $PL(\mathbf{SUP}')$ and $PL_m(\mathbf{SUP}')$ is of polynomial complexity $O(n^4)$ [21]. Then checking the two equalities in Step 2 is of $O(n^2)$ complexity. Finally in Step 3, controllability may be checked using standard algorithm [8] in linear time $O(n)$. Therefore, Algorithm 1 terminates and is of polynomial complexity $O(n^4)$. The following result is straightforward.

Proposition 1. *Consider that* $\mathbf{G}_j$ *sends event* σ *to* $\mathbf{G}_i$ *through channel* $\mathbf{CH}(j,\sigma,i)$. *The monolithic supervisor* $\mathbf{SUP}$ *is delay-robust with respect to* $\mathbf{CH}(j,\sigma,i)$ *if and only if Algorithm 1 returns true.*

Remark 1. (Multi-capacity channel model) So far we have considered the 1-capacity channel model

[3] We note *en passant* that [22] reports an algorithm with quadratic time complexity for verifying the observer property alone; that does not, however, yield structural information which (if the observer property is not satisfied) might be useful for remedial design.

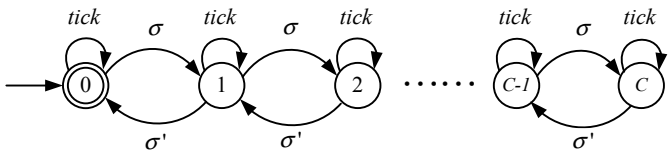

Fig. 2. C-capacity channel model $\mathbf{NCH}(j, \sigma, i)$.

$\mathbf{CH}(j, \sigma, i)$, and defined delay-robustness with respect to it. We now consider the more general C-capacity channel model $\mathbf{NCH}(j, \sigma, i)$, $C \geq 1$ a positive integer, displayed in Fig. 2. The sender $\mathbf{G}_j$ may send at most C instances of event σ to $\mathbf{NCH}(j, \sigma, i)$, each instance subject to indefinite delay. With channel $\mathbf{NCH}(j, \sigma, i)$, one may proceed just as before, by replacing $\mathbf{CH}(j, \sigma, i)$ by $\mathbf{NCH}(j, \sigma, i)$ throughout, to define the corresponding delay-robustness property with respect to $\mathbf{NCH}(j, \sigma, i)$, and then revising Algorithm 1 correspondingly to verify delay-robustness.

It is worth noting that when $\mathbf{NCH}(j, \sigma, i)$ reaches its maximal capacity, and $\mathbf{G}_j$ sends yet another instance of σ, then σ is 'blocked' by $\mathbf{NCH}(j, \sigma, i)$, implying uncontrollability of the channeled behavior. Hence the uncontrollability problem always exists as long as the channel model is of finite capacity and delay is indefinite, although the controllability condition (cf. condition (ii) of Definition 1) is more easily satisfied for larger capacity channels (simply because more instances of σ may be sent to the channel).

IV. Bounded Delay-Robustness and Maximal Delay Bound

Consider again the situation that agent $\mathbf{G}_j$ sends an event σ to $\mathbf{G}_i$. If the monolithic supervisor $\mathbf{SUP}$ is verified (by Algorithm 1) to be delay-robust, then we will use channel $\mathbf{CH}(j, \sigma, i)$ in Fig. 1 to transmit σ subject to unbounded delay, and the system's behavior will not be affected. If, however, $\mathbf{SUP}$ fails to be delay-robust, there are two possible implications: (1) σ must be transmitted without delay (as in the original setup of localization [1, 3, 4]); or (2) there exists a delay bound d (≥ 1) of σ such that if each transmission of σ is completed within d occurrences of $tick$, the system's behavior will remain unaffected. This section aims to identify the latter case, which we call "bounded delay-robust", and moreover to determine the bound d.

To that end, consider the channel model $\mathbf{CH}_d(j, \sigma, i)$ in Fig. 3, with parameter $d \geq 1$. $\mathbf{CH}_d(j, \sigma, i)$ is a $(d+2)$-state TDES with event set $\{\sigma, tick, \sigma'\}$. After an occurrence of σ (state 0 to 1), $\mathbf{CH}_d(j, \sigma, i)$ counts up to d (≥ 0) occurrences of $tick$ (state 1 through $d+1$) by which time the signal event σ' must occur. That is, the occurrence of σ' ($\mathbf{G}_i$ receives σ) is bounded by d $tick$s. Note that the $tick$ selfloop at

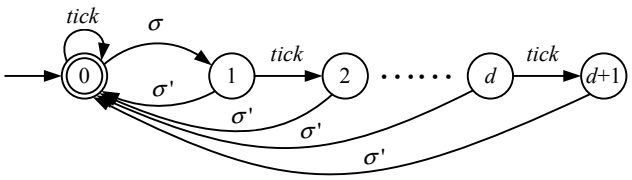

Fig. 3. Timed channel model $\mathbf{CH}_d(j,\sigma,i)$, $d \geq 1$, for transmitting event σ from $\mathbf{G}_j$ to $\mathbf{G}_i$ with delay bound d.

state 0 is again for the technical requirement to prevent the blocking of event $tick$ when synchronizing $\mathbf{CH}_d(j,\sigma,i)$ with other TDES.

Now with $\mathbf{CH}_d(j,\sigma,i)$, the channeled behavior of the system is

$$\mathbf{SUP'}_d := \mathbf{NSUP} \;||\; \mathbf{CH}_d(j,\sigma,i) \tag{16}$$

where $\mathbf{NSUP}$ is given in (10). The event set of $\mathbf{SUP'}_d$ is $\Sigma' = \Sigma \cup \{\sigma'\}$, and we recall the natural projections $P : \Sigma'^* \to \Sigma^*$ and $P_{ch} : \Sigma'^* \to \{\sigma, tick, \sigma'\}^*$.

Definition 2. Consider that $\mathbf{G}_j$ sends event σ to $\mathbf{G}_i$ through channel $\mathbf{CH}_d(j,\sigma,i)$, $d \geq 1$. The monolithic supervisor $\mathbf{SUP}$ in (4) is *bounded delay-robust* with respect to $\mathbf{CH}_d(j,\sigma,i)$ (or *d-bounded delay-robust*) if the following conditions hold:

(i) $\mathbf{SUP'}_d$ in (16) is *correct* and *complete*, i.e.

$$PL(\mathbf{SUP'}_d) = L(\mathbf{SUP}) \tag{17}$$

$$PL_m(\mathbf{SUP'}_d) = L_m(\mathbf{SUP}) \tag{18}$$

$$(\forall s \in \Sigma'^*)(\forall w \in \Sigma^*)\ s \in L(\mathbf{SUP'}_d)\ \&\ (Ps)w \in L_m(\mathbf{SUP}) \\ \Rightarrow (\exists v \in \Sigma'^*)\ Pv = w\ \&\ sv \in L_m(\mathbf{SUP'}_d) \tag{19}$$

(ii) $P_{ch}^{-1}(L(\mathbf{CH}_d(j,\sigma,i)))$ is controllable with respect to $L(\mathbf{NSUP})$ and $\{\sigma\}$, i.e.

$$P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))\{\sigma\} \;\cap\; L(\mathbf{NSUP}) \subseteq P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i)) \tag{20}$$

Bounded delay-robustness is defined in the same way as (unbounded) delay-robustness in Definition 1, but with respect to the new channel model $\mathbf{CH}_d(j,\sigma,i)$ with delay bound d. As a result, d-bounded delay-robustness may be verified by Algorithm 1 with corresponding modifications. For later reference, we state here the modified algorithm.

Algorithm 2

1. Check if P is an $L_m(\mathbf{SUP'}_d)$-observer. If not, return *false*.

2. Check if $PL(\mathbf{SUP'}_d) = L(\mathbf{SUP})$ and $PL_m(\mathbf{SUP'}_d) = L_m(\mathbf{SUP})$. If not, return *false*.

3. Check if $P_{ch}^{-1}(L(\mathbf{CH}_d(j,\sigma,i)))$ is controllable with respect to $L(\mathbf{NSUP})$ and $\{\sigma\}$. If not, return *false*.

4. Return *true*.

Now if the monolithic supervisor $\mathbf{SUP}$ fails to be (unbounded) delay-robust with respect to channel $\mathbf{CH}(j,\sigma,i)$, we would like to verify if $\mathbf{SUP}$ is bounded delay-robust with respect to $\mathbf{CH}_d(j,\sigma,i)$ for some $d \geq 1$. If so, compute the *maximal* delay bound, i.e. the largest delay (number of $ticks$) that can be tolerated without changing the system's logical behavior. We need the following lemma.

Lemma 1. *Consider that* $\mathbf{G}_j$ *sends event* σ *to* $\mathbf{G}_i$ *through channel* $\mathbf{CH}_d(j,\sigma,i)$, $d \geq 1$. *If* $\mathbf{SUP}$ *is not* d*-bounded delay-robust, then it is not* $(d+1)$*-bounded delay-robust.*

The result of Lemma 1 is intuitive: if $\mathbf{SUP}$ cannot tolerate a σ transmission delay of d, neither can it tolerate a delay $(d+1)$. By induction, in fact, $\mathbf{SUP}$ cannot tolerate any delay larger than d. The proof of Lemma 1 is in Appendix A. This fact suggests the following algorithm for identifying bounded delay-robustness as well as computing the maximal delay bound.

Algorithm 3

1. Set $d = 1$.

2. Check by Algorithm 2 if $\mathbf{SUP}$ is d-bounded delay-robust relative to channel $\mathbf{CH}_d(j,\sigma,i)$. If not, let $d = d-1$ and go to Step 3. Otherwise advance d to $d+1$ and repeat Step 2.

3. Output $d_{max} := d$.

Lemma 2. *If* $\mathbf{SUP}$ *is not delay-robust with respect to* $\mathbf{CH}(j,\sigma,i)$*, then Algorithm 3 terminates in at most* $2^m * m$ *steps, i.e.* $d_{max} \leq 2^m * m$*, where m is the state size of* $\mathbf{SUP'}$ *in (11).*

The proof of Lemma 2 is given in Appendix B. In Algorithm 3, we work upwards starting from the minimal delay $d = 1$. If $\mathbf{SUP}$ is *not* 1-bounded delay-robust with respect to $\mathbf{CH}_1(j,\sigma,i)$, then by Lemma 1 $\mathbf{SUP}$ is *not* d-bounded delay-robust for any $d > 1$. Therefore $\mathbf{SUP}$ is *not* bounded delay-robust and σ must be transmitted without delay. Note that in this case Algorithm 3 outputs $d_{max} = 0$.

If $\mathbf{SUP}$ is 1-bounded delay-robust, we next check if it is 2-bounded delay-robust with respect to $\mathbf{CH}_2(j,\sigma,i)$. If $\mathbf{SUP}$ fails to be 2-bounded delay-robust, then again by Lemma 1 $\mathbf{SUP}$ fails to be d-bounded delay-robust for any $d > 2$. Hence $\mathbf{SUP}$ is bounded delay-robust, with the maximal delay bound $d_{max} = 1$.

If $\mathbf{SUP}$ is shown to be 2-bounded delay-robust, the iterative process continues until $\mathbf{SUP}$ fails to be $(d+1)$-bounded delay-robust for some $d \geq 2$; this happens in finitely many steps according to Lemma 2.

Then $\mathbf{SUP}$ is bounded delay-robust, with the maximal delay bound $d_{max} = d$. The following result is immediate.

Proposition 2. *Consider that $\mathbf{G}_j$ sends event σ to $\mathbf{G}_i$ through channel $\mathbf{CH}_d(j, \sigma, i)$, $d \geq 1$. The monolithic supervisor $\mathbf{SUP}$ is bounded delay-robust with respect to $\mathbf{CH}_d(j, \sigma, i)$ if and only if the output d_{max} of Algorithm 3 satisfies $d_{max} > 0$. Moreover, if $\mathbf{SUP}$ is bounded delay-robust, then d_{max} is the maximal delay bound for σ transmission.*

To summarize, when an event σ is sent from $\mathbf{G}_j$ to $\mathbf{G}_i$, we determine unbounded or bounded delay-robustness and choose the corresponding channel as follows.

Algorithm 4

1. Check by Algorithm 1 if $\mathbf{SUP}$ is (unbounded) delay-robust. If so, terminate, set the maximal delay bound $d_{max} = \infty$, and use channel $\mathbf{CH}(j, \sigma, i)$ in Fig. 1.

2. Check by Algorithm 3 if $\mathbf{SUP}$ is bounded delay-robust. If so (i.e. $d_{max} \geq 1$), terminate and use channel $\mathbf{CH}_d(j, \sigma, i)$ in Fig. 3 with $d = d_{max}$.

3. In this case $d_{max} = 0$. Terminate and use no channel: σ must be transmitted without delay.

Remark 2. (Multiple channeled events) So far we have considered a single event communication: agent $\mathbf{G}_j$ sends event σ to $\mathbf{G}_i$. Using this as a basis, we present an approach to the general case of multiple channeled events, as is common in distributed control. We will consider that each fixed triple (sender, channeled event, receiver) is assigned with its own communication channel, and the assigned channels operate concurrently. Our goal is to obtain these channels, ensuring unbounded or bounded delay-robustness, one for each triple (sender, channeled event, receiver).

First fix $i, j \in \mathcal{N}$, and recall from (8) that $\Sigma_{j,com,i}$ is the set of events that $\mathbf{G}_j$ sends to $\mathbf{G}_i$. Write $\Sigma_{j,com,i} = \{\sigma_1, ..., \sigma_r\}$, $r \geq 1$, and treat the channeled events σ_1, σ_2, ... *sequentially*, in order of indexing.

Algorithm 5

1. Set $p = 1$.

2. For event $\sigma_p \in \Sigma_{j,com,i}$ apply Algorithm 4 to obtain the maximal delay bound d_{max}.

2.1. If $d_{max} = \infty$, namely unbounded delay-robustness, choose channel $\mathbf{CH}(j, \sigma_p, i)$, and let $\mathbf{NSUP} := \mathbf{NSUP}||\mathbf{CH}(j, \sigma_p, i)$.

2.2. If $d_{max} \geq 1$ is finite, namely bounded delay-robustness, choose channel $\mathbf{CH}_d(j, \sigma_p, i)$, and let $\mathbf{NSUP} := \mathbf{NSUP}||\mathbf{CH}_d(j, \sigma_p, i)$.

2.3 If $d_{max} = 0$, then no channel is chosen and σ_p must be transmitted without delay.

If $p < r$, advance p to $p + 1$ and repeat Step 2.

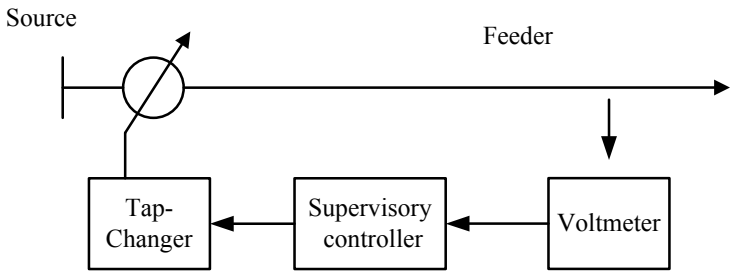

Fig. 4. ULTC: Components and Controller

3. Output a set of channels used for sending events from $\mathbf{G}_j$ to $\mathbf{G}_i$.

Note that at Step 2 of Algorithm 5, if a channel is chosen for event σ_p, then **NSUP** must be reset to be the synchronous product of **NSUP** and the channel, so that in choosing a channel for the next event σ_{p+1} the previously chosen channel is considered together. This ensures that when the derived channels operate concurrently, the system's behavior is not affected. It is worth noting that a different ordering of the set $\Sigma_{j,com,i}$ may result in a different set of channels; if no priority of the transmission delay is imposed on the communication events, we may choose an ordering randomly.

Finally, since the set of all communication events is $\Sigma_{com} := \underset{i,j\in\mathcal{N}}{\cup}\Sigma_{j,com,i}$, we simply apply Algorithm 5 for each (ordered) pair $i, j \in \mathcal{N}$ to derive all communication channels. Again, a different ordering of the set $\mathcal{N}\times\mathcal{N}$ generally results in a different set of channels, because the channels chosen for a pair (i, j) will be used to decide channels for all subsequent (i', j'). For convenience we will simply order the pairs (i, j) sequentially first on j then on i.

V. Case Study: Under-Load Tap-Changing Transformer

In this section we demonstrate timed delay-robustness and associated verification algorithms on an under-load tap-changing transformer system.

A. Model Description and Supervisor Localization

Transformers with tap-changing facilities constitute an important means of controlling voltage at all levels throughout electrical power systems. We consider an under-load tap-changing transformer (ULTC) as displayed in Fig. 4, which consists of two components: Voltmeter and Tap-Changer[23].

This ULTC is operated in two modes: Automatic and Manual. In the automatic mode, the tap-changer works according to the following logic. (1) If the voltage deviation is greater than some threshold value,

TABLE I. PHYSICAL INTERPRETATION OF EVENTS

Event	Physical interpretation	Time bounds (lower, upper)	(hib/for)
11	Initialize voltmeter	(0, ∞)	hib
10	Report $\|\Delta V\| > ID$ and $\Delta V > 0$	(0, ∞)	
12	Report $\|\Delta V\| < ID$, i.e. voltage recovered	(0, ∞)	
14	Report $\|\Delta V\| > ID$ and $\Delta V < 0$	(0, ∞)	
16	Report voltage exceeds V_{max}	(0, ∞)	
30	Tap-up/Down failed	(0, ∞)	
31	Tap-down command with 5 $tick$ delay	(5, ∞)	hib & for
32	Tap-down successful	(0, ∞)	
33	Tap-up command	(0, ∞)	hib & for
34	Tap-up successful	(0, ∞)	
35	Tap-down command without delay	(0, ∞)	hib & for
41	Enter Automatic mode	(0, ∞)	hib
43	Enter Manual mode	(0, ∞)	hib

then a timer will start; when the timer times out, a 'tap increase (or decrease) event' will occur and the timer will reset; a tap increase or decrease should only occur if the voltage change continues to exceed threshold after the time out- this is to avoid tap changes in response to merely occasional random fluctuations of brief duration. (2) If the voltage returns to the dead-band, because of a tap change or some other reason, then no tap change will occur. (3) If the voltage exceeds the maximally allowed value V_{max}, then lowering of the tap command without delay occurs instantaneously. In the manual mode, the system is waiting for 'Tap-up', 'Tap-down', or 'Automatic' commands. An operator can change the operation mode from one to the other, and thus the operator is adjoined into the plant components to be controlled.

Each plant component is modeled as a TDES displayed in Fig. 5, and associated events are listed in Table I. So, the plant to be controlled is the synchronized behavior of Voltmeter (**VOLT**), Tap-changer (**TAP**) and Operator (**OPTR**), i.e.

$$\mathbf{PLANT} = \mathbf{VOLT} \;||\; \mathbf{TAP} \;||\; \mathbf{OPTR}. \tag{21}$$

We consider a voltage control problem of the ULTC: when the voltage is not 'normal', design controllers to recover the voltage through controlling tap ratio after a time delay to recover the voltage. Fig. 6 displays the TDES model **SPEC** for the control specification in Automatic/Manual mode.

Note that since the tap increase (decrease) and lowering tap commands would preempt the occurrence of $tick$, the corresponding events 31, 33 and 35 are designated as forcible events. In the following, we

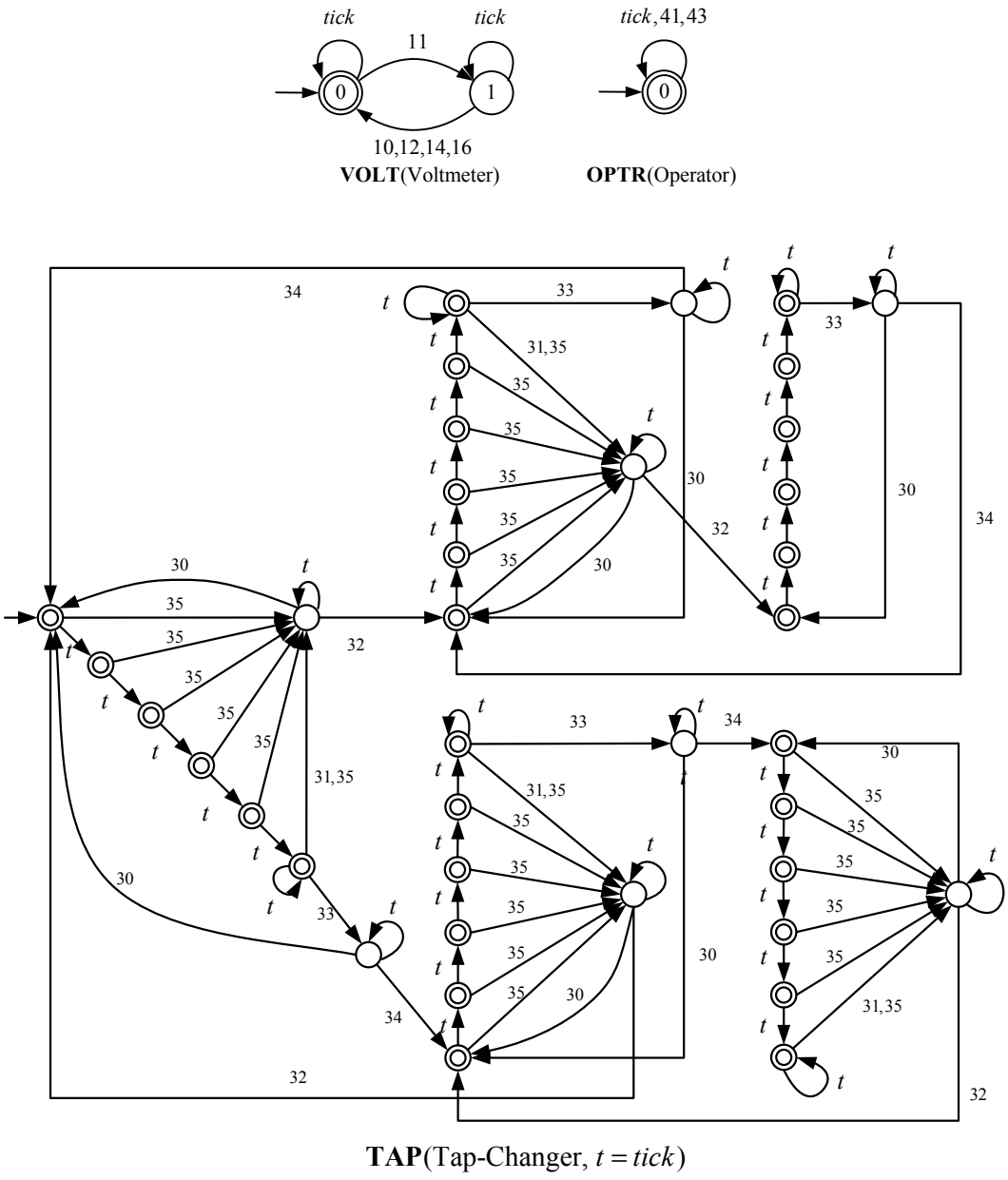

Fig. 5. Timed Transition Graph of ULTC Components

synthesize the monolithic supervisor **SUP** by the standard TDES supervisory control theory [8, 9] and the local controllers by TDES supervisor localization [3, 4].

First, synthesize the monolithic supervisor TDES **SUP** in the usual sense that its marked behavior

$$L_m(\mathbf{SUP}) = Sup\mathcal{C}(L_m(\mathbf{SPEC}) \cap L_m(\mathbf{VOLT})) \tag{22}$$

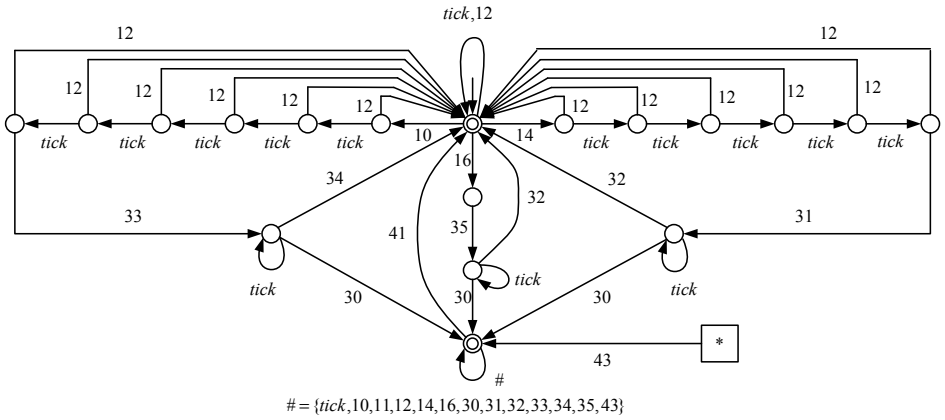

Fig. 6. Control Specification **SPEC** in Automatic/Manual Mode. The transition 43 from the square with '*' represents similar transitions from all states to the 'manual operation mode'.

and its closed behavior $L(\mathbf{SUP}) = \overline{L_m(\mathbf{SUP})}$. **SUP** has 231 states and 543 transitions, and embodies disabling actions for all the prohibitible events and preempting actions relative to $tick$ for all the forcible events.

Next, by supervisor localization, we obtain a set of local controllers $\mathbf{LOC}^C_{11}$, $\mathbf{LOC}^C_{31}$, $\mathbf{LOC}^C_{33}$, $\mathbf{LOC}^C_{35}$ $\mathbf{LOC}^C_{41}$ and $\mathbf{LOC}^C_{43}$ for controllable events 11, 31, 33, 35, 41 and 43 respectively, and a set of local preemptors $\mathbf{LOC}^P_{31}$, $\mathbf{LOC}^P_{33}$ and $\mathbf{LOC}^P_{35}$ for forcible events 31, 33 and 35 respectively; their transition diagrams are shown in Fig. 7.

Finally, using these constructed local controllers/preemptors, we build a distributed control architecture without communication delays for ULTC as displayed in Fig. 8. The local controlled behaviors of the plant components are

$$\begin{aligned}
\mathbf{SUP}_V &= \mathbf{VOLT} \;||\; \mathbf{LOC}^C_{11}, \\
\mathbf{SUP}_T &= \mathbf{TAP} \;||\; (\mathbf{LOC}^C_{31} \;||\; \mathbf{LOC}^C_{33} \;||\; \mathbf{LOC}^C_{35}) \\
&\qquad\qquad ||\; (\mathbf{LOC}^P_{31} \;||\; \mathbf{LOC}^P_{33} \;||\; \mathbf{LOC}^P_{35}), \\
\mathbf{SUP}_O &= \mathbf{OPTR} \;||\; (\mathbf{LOC}^C_{41} \;||\; \mathbf{LOC}^C_{43}).
\end{aligned}$$

Let $\Sigma_{A,com,B}$ represent the set of events that component A sends to component B; the sets of commu-

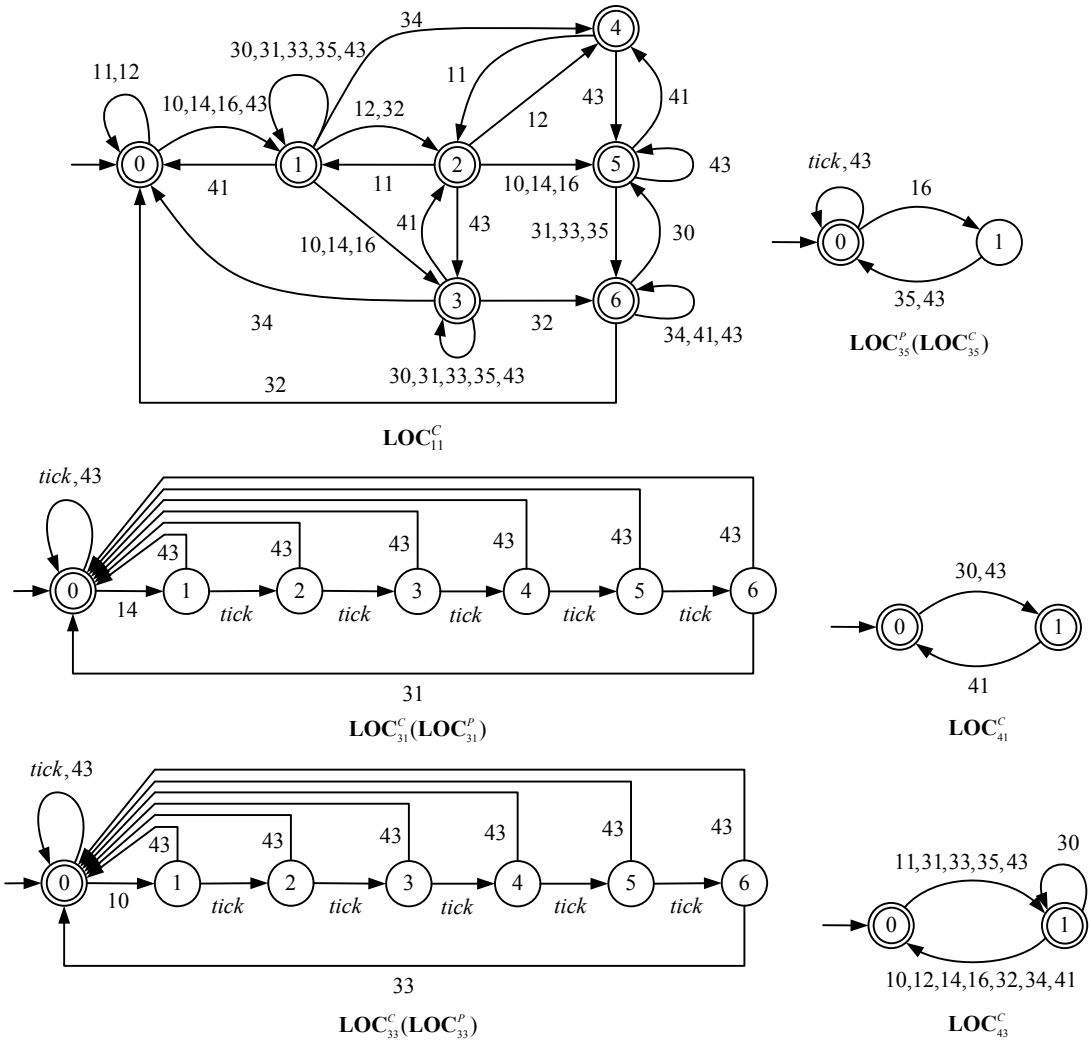

Fig. 7. Local controllers and local preemptors for ULTC. According to Footnote 2, for each state state x of each local controller/preemptor, and each communication event σ, if σ is not defined at x, we add a σ-selfloop. Let *(x) be the set of events whose selfloops need to be adjoined at state x. In $\mathbf{LOC}_{11}^{C}$, $*(0) = *(2) = *(4) = \{30, 31, 32, 33, 34, 35, 41\}$, $*(1) = \{43\}$, $*(5) = \{30, 32, 34\}$, and $*(6) = \{31, 33, 35\}$; in $\mathbf{LOC}_{31}^{C}$ ($\mathbf{LOC}_{31}^{P}$), $*(1) = *(2) = *(3) = *(4) = *(5) = *(6) = \{14\}$; in $\mathbf{LOC}_{33}^{C}$ ($\mathbf{LOC}_{33}^{P}$), $*(1) = *(2) = *(3) = *(4) = *(5) = *(6) = \{10\}$; in $\mathbf{LOC}_{35}^{C}$ ($\mathbf{LOC}_{35}^{P}$), $*(1) = \{16\}$; in $\mathbf{LOC}_{41}^{C}$, $*(1) = \{30\}$; in $\mathbf{LOC}_{43}^{C}$, $*(0) = \{10, 12, 14, 16, 30, 32, 34\}$, and $*(1) = \{11, 31, 33, 35\}$.

nication events are

$$\begin{aligned}
\Sigma_{T,com,V} &= \{30, 31, 32, 33, 34, 35\}, \\
\Sigma_{O,com,V} &= \{41, 43\}, \\
\Sigma_{V,com,T} &= \{10, 14, 16\}, \\
\Sigma_{O,com,T} &= \{43\}, \\
\Sigma_{V,com,O} &= \{10, 11, 12, 14, 16\}, \\
\Sigma_{T,com,O} &= \{30, 31, 32, 33, 34, 35\}.
\end{aligned} \tag{23}$$

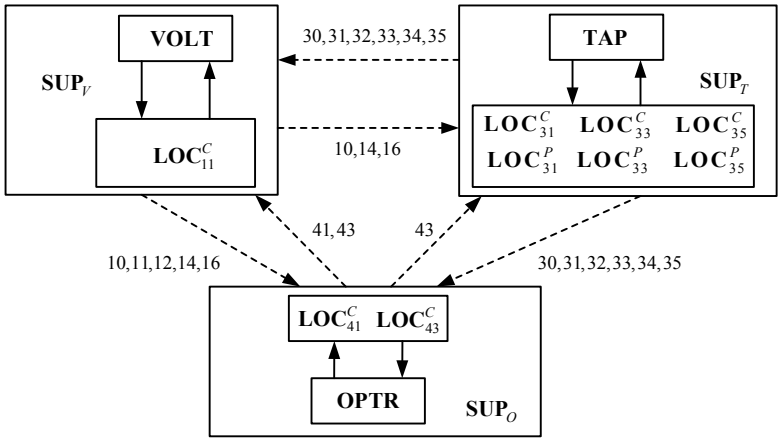

Fig. 8. Distributed Control Architecture of ULTC

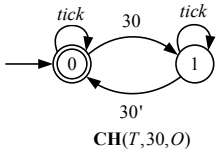

Fig. 9. Communication Channel $\mathbf{CH}(T, 30, O)$

It is guaranteed by supervisor localization of TDES [3, 4] that the ULTC under the control of these local controllers and preemptors without communication delay, has closed and marked behavior identical to $\mathbf{SUP}$ in (22).

B. Delay-Robustness Verification

Now we investigate the timed delay-robustness property for ULTC. For illustration, we consider the following three cases.

(1) Event 30 in $\Sigma_{T,com,O}$

Applying Algorithm 4, at Step 1 we verify by Algorithm 1 that $\mathbf{SUP}$ is delay-robust with respect to the communication channel $\mathbf{CH}(T, 30, O)$ transmitting event 30, as displayed in Fig. 9.

To illustrate that the overall system behavior will not be affected by indefinite communication delay of event 30, consider the case that the voltmeter reported an increase in voltage (in $\mathbf{VOLT}$ as displayed in Fig. 5, events 11 and 10 have occurred), and the tap has received a tap-up command, but the tap-up operation failed (in $\mathbf{TAP}$ as displayed in Fig. 5, events $tick$, $tick$, $tick$, $tick$, $tick$, 33 and 30 have

occurred in sequence). By inspection of the transition diagrams, the plant components **VOLT**, **TAP** and **OPTR** in Fig. 5 are at states 0, 0, and 0 respectively, and thus the events that are eligible to occur are 11, 35, 41, 43, and $tick$. However, according to the transition diagrams of the local controllers and preemptors displayed in Fig. 7: (1) $\mathbf{LOC}_{11}^{C}$ is at state 1 and disables event 11; (2) $\mathbf{LOC}_{35}^{C}$ is at state 0 and disables event 35; (3) $\mathbf{LOC}_{41}^{C}$ will disable or enable event 41 depending on the communication delay of event 30; (4) $\mathbf{LOC}_{43}^{C}$ is at state 1 and disables event 43; (5) $tick$ will not be preempted, since no forcible event is enabled. If 30 is transmitted instantly, event 41 is enabled by $\mathbf{LOC}_{41}^{C}$ and the system will enter the automatic mode. If the transmission of 30 is delayed, only event $tick$ is enabled, and other events will not be enabled until the system enters the automatic mode. However, according to the transition diagram of $\mathbf{LOC}_{41}^{C}$ displayed in Fig. 7, only after $\mathbf{LOC}_{41}^{C}$ has received the occurrence of event 30, will it enable event 41, and bring the system into the automatic mode. Hence, the overall system behavior will not be affected even if the communication of event 30 is delayed.

(2) Event 10 in $\Sigma_{V,com,O}$

Applying Algorithm 4, at Step 1 we verify by Algorithm 1 that **SUP** fails to be delay-robust with respect to the channel $\mathbf{CH}(V, 10, O)$, as displayed in Fig. 10; then at Step 2, we check by Algorithm 3 that the maximal delay bound for event 10 is 4, i.e. **SUP** is bounded delay-robust with respect to the channel $\mathbf{CH}_4(V, 10, O)$, as displayed in Fig. 10.

To illustrate that **SUP** is not delay-robust with respect to $\mathbf{CH}(V, 10, O)$, but is bounded delay-robust with respect to $\mathbf{CH}_4(V, 10, O)$, we consider the case that an increase in the voltage is reported (i.e. events 11 and 10 in **VOLT** have occurred sequentially). By inspection of the transition diagrams of the plant components shown in Fig. 5, the events that are eligible to occur are 11, 35, 41, 43, and $tick$. According to the transition diagrams of the local controllers and preemptors displayed in Fig. 7, if **OPTR** knows the voltage increase before the fifth *tick* occurs, the tap-changer will generate a tap-up command and the operator can switch the system into manual mode; otherwise, the tap-changer will also generate a tap-up command, but the system cannot enter the manual mode. In terms of language, event 43 will be enabled after the event sequence $s := 11.10.tick.tick.tick.tick.tick.310.33$ (where event 310 is the signal event of 10), but is disabled after $s' := 11.10.tick.tick.tick.tick.tick.33$. When observing s and s' from the projection P that erases the signal event 310, they cannot be distinguished. However, the system can enter the manual mode after the sequence s, but not after s'. In other words, the system can not 'complete' the behavior of entering manual mode after s', but this behavior can be finished in its delay-free counterpart **SUP**. So, the observer property (19) required by bounded delay-robustness is violated when the delay bound d exceeds 4 $tick$s, and we conclude that the maximal delay bound for event 10 is 4.

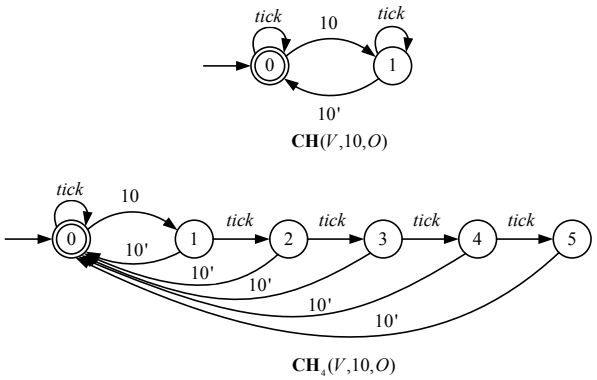

Fig. 10. Communication Channels $\mathbf{CH}(V, 10, O)$ and $\mathbf{CH}_4(V, 10, O)$

Similarly, one can verify by Algorithm 4 that $\mathbf{SUP}$ is bounded delay-robust with respect to $\mathbf{CH}_4(V, 14, O)$, as displayed in Fig. 11, and any other events except 10, 14 and 30 must be transmitted without delay.

(3) All communication events in (23)

Applying Algorithm 5 to each of the sets of communication events in (23) in sequence, we obtain that $d'_{max}(T, 30, O) = \infty$, $d'_{max}(V, 10, O) = d'_{max}(V, 14, O) = 4$, and for the remaining events, $d'_{max} = 0$. In the following, we verify that if all the communication events are communicated within their corresponding delay bounds, the overall system behavior will still not be affected.

First, use $\mathbf{CH}(T, 30, O)$, $\mathbf{CH}_4(V, 10, O)$ and $\mathbf{CH}_4(V, 14, O)$ to transmit events 30, 10 and 14 respectively. Second, connected by these channels, the overall system behavior is

$$\begin{aligned}\mathbf{SUP}'_{com} =& \mathbf{SUP}_V||\mathbf{SUP}_T||\mathbf{CH}_4(V, 10, O)||\\ & \mathbf{CH}_4(V, 14, O)||\mathbf{CH}(T, 30, O)||\mathbf{SUP}'''_O)\end{aligned}$$

over the augmented alphabet $\{10, 11, ..., 43, 10', 14', 30'\}$, where $\mathbf{SUP}'''_O$ is obtained by replacing 10, 14, and 30 by $10'$, $14'$ and $30'$ respectively. Third, one can verify that: (1) $\mathbf{SUP}'_{com}$ is correct and complete, and (2) $\mathbf{CH}(T, 30, O)$, $\mathbf{CH}_4(V, 10, O)$ and $\mathbf{CH}_4(V, 14, O)$ will not cause uncontrollability with respect to the uncontrollable communication events. Finally, we conclude that the overall system behavior is still optimal and nonblocking.

VI. Conclusions

In this paper we have studied communication delays among local controllers obtained by supervisor localization in TDES. First, we have identified properties of 'timed delay-robustness' which guarantee

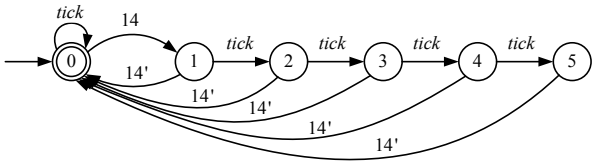

Fig. 11. Communication Channel $\mathbf{CH}_4(V, 14, O)$

that the specification of our delay-free distributed control continues to be enforced in the presence of (possibly unbounded) delay, and presented a polynomial verification algorithm to determine delay-robustness. Second, for those events that fail to be delay-robust, we have proposed an algorithm to determine their maximal delay bound d_{max} such that the system is d_{max}-bounded delay-robust. Finally, a ULTC example has exemplified these results, showing how to verify the delay-robustness, determine the maximal delay bound for bounded delay-robustness, and in addition, obtain a set of maximal delay bounds, one for each communication event, under the condition that the overall system behavior is still optimal and nonblocking.

With the definitions and tests reported here as basic tools, our future work will include the investigation of alternative more complex channel models and, of especial interest, global interconnection properties of a distributed system of TDES which may render delay-robustness more or less likely to be achieved.

APPENDIX A
PROOF OF LEMMA 1

To prove Lemma 1, we need the following Lemmas 3 and 4.

Lemma 3. *For any delay bound $d \geq 1$, there hold*

$$L(\mathbf{SUP}) \subseteq PL(\mathbf{SUP}'_d) \tag{24}$$

$$L_m(\mathbf{SUP}) \subseteq PL_m(\mathbf{SUP}'_d) \tag{25}$$

Proof: Note that for different delay bounds d, the alphabets of $\mathbf{SUP}'_d$ and $\mathbf{CH}_d(j, \sigma, i)$ are $\Sigma' = \Sigma \cup \{\sigma'\}$ and $\{\sigma, tick, \sigma'\}$, respectively. Here we only prove that $L(\mathbf{SUP}) \subseteq PL(\mathbf{SUP}'_d)$; (25) can be proved in the same way by replacing L by L_m throughout.

Let $s \in L(\mathbf{SUP})$; we must show that there exists a string $t \in L(\mathbf{SUP}'_d)$ such that $P(t) = s$. We first consider that only one instance of σ appeared in s, and write $s = x_1 \sigma x_2$ where x_1, x_2 are free

of σ. By (16) and observing that $\mathbf{SUP}_i'$ is obtained by replacing each instance of σ by σ', we obtain that $t := x_1\sigma\sigma' x_2 \in L(\mathbf{SUP}_d')$. Furthermore, $P(t) = s$. So, $L(\mathbf{SUP}) \subseteq PL(\mathbf{SUP}_d')$. This result can be easily extended to the general case that s has multiple instances of σ, because σ is transmitted by the channel model and the reoccurrence of σ is permitted only when transmission of the previous σ is completed. Namely, if $s = x_1\sigma x_2\sigma ..., x_{k-1}\sigma x_k$, there exists a string $t = x_1\sigma\sigma' x_2\sigma\sigma' ..., x_{k-1}\sigma\sigma' x_k$ such that $t \in L(\mathbf{SUP}_d')$ and $Pt = s$. Hence, we declare that $L(\mathbf{SUP}) \subseteq PL(\mathbf{SUP}_d')$. ■

Lemma 4. *Let $t = x_1\sigma x_2 x_3\sigma' x_4 \in L_m(\mathbf{SUP}_d')$ where x_1, x_2, x_3 and x_4 are strings free of σ and σ', i.e. $x_1, x_2, x_3, x_4 \in (\Sigma - \{\sigma\})^*$. Then $t' := x_1\sigma x_2\sigma' x_3 x_4 \in L_m(\mathbf{SUP}_d')$.*

Proof of Lemma 4: Recall that $\mathbf{SUP}_i'$ is $\mathbf{SUP}_i$ with transitions labeled σ relabeled σ'. By definition of synchronous product, x_2, x_3 and σ' can be re-ordered without affecting the membership of t in $L_m(\mathbf{SUP}_d')$, namely the strings t' formed from t by the successive replacement

$$x_1\sigma x_2 x_3\sigma' x_4 \to x_1\sigma\sigma' x_2 x_3 x_4$$

$$\to x_1\sigma x_2\sigma' x_3 x_4$$

will belong to $L_m(\mathbf{SUP}_d')$ as well. In other words, if the transmission of σ is completed in a shorter time (the number of *tick*s in x_2 will be smaller than that in x_2x_3), the behavior is still legal. ■

Proof of Lemma 1: We prove Lemma 1 by contraposition, i.e. if $\mathbf{SUP}$ is $(d+1)$-bounded delay-robust, then it is also d-bounded delay-robust. To that end, we must verify (17)-(20).

(1) For (17), we prove that $PL(\mathbf{SUP}_d{}') \supseteq L(\mathbf{SUP})$ and $PL(\mathbf{SUP}_d{}') \subseteq L(\mathbf{SUP})$ in sequence. $PL(\mathbf{SUP}_d{}') \supseteq L(\mathbf{SUP})$ is obtained from Lemma 3 immediately. By inspection of the transition diagram of $\mathbf{CH}_d(j, \sigma, i)$ in Fig. 3, we get that $L(\mathbf{CH}_d(j, \sigma, i)) \subseteq L(\mathbf{CH}_{d+1}(j, \sigma, i))$. So according to (16),

$$L(\mathbf{SUP}_d') \subseteq L(\mathbf{SUP}_{d+1}'). \tag{26}$$

Since $\mathbf{SUP}$ is $(d+1)-$bounded delay-robust, $PL(\mathbf{SUP}_{d+1}') \subseteq L(\mathbf{SUP})$. Hence, $PL(\mathbf{SUP}_d{}') \subseteq L(\mathbf{SUP})$.

(2) Condition (18) can be confirmed from the proof of (17) by replacing L by L_m throughout.

(3) For (19), assume that $s \in L(\mathbf{SUP}_d')$ and $(Ps)w \in L_m(\mathbf{SUP})$; we must show that there exists a string $v \in \Sigma'^*$ such that $Pv = w$ and $sv \in L_m(\mathbf{SUP}_d')$.

By (26), we have $s \in L(\mathbf{SUP}_{d+1}')$. Since $\mathbf{SUP}$ is $(d+1)-$bounded delay-robust, there exists a string $u \in \Sigma'^*$ such that $Pu = w$ and $su \in L_m(\mathbf{SUP}_d')$. Here we consider the case that only one instance of σ

exists in su; the general cases can be confirmed similarly (since the transmission of multiple instances of σ does not result in mutual interference). In the following, we prove (19) from these three cases: (i) $su = s_1\sigma s_2\sigma' s_3 u_1 u_2$, (2) $su = s_1\sigma s_2 u_1\sigma' u_2$, and (iii) $s_1 s_2 u_1\sigma u_2\sigma' u_3$, where $s_1, s_2, s_3, u_1, u_2, u_3$ are free of σ and σ'.

(i) $su = s_1\sigma s_2\sigma' s_3 u_1 u_2$. By (16), we have $su \in L_m(\mathbf{NSUP})$. Similarly, since $s \in L(\mathbf{SUP}'_d)$, $s \in P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))$. Further, $s = s_1\sigma s_2\sigma' s_3$, which means that after string s, σ' has reset the channel $\mathbf{CH}_d(j,\sigma,i)$. Thus $s \in P_{ch}^{-1}L_m(\mathbf{CH}_{d-1}(j,\sigma,i))$. On the other hand, because u is free of σ, $su \in P_{ch}^{-1}L_m(\mathbf{CH}_d(j,\sigma,i))$. Hence, $su \in L_m(\mathbf{SUP}'_d)$. Define $v = u$; then $Pv = Pu = w$ and $sv \in L_m(\mathbf{SUP}'_d)$, as required by (19).

(ii) $su = s_1\sigma s_2 u_1\sigma' u_2$. By Lemma 4,, it results from $su \in L_m(\mathbf{SUP}'_d)$ that $s_1\sigma s_2\sigma' u_1 u_2 \in L_m(\mathbf{SUP}'_d)$. The rest is similar to case (1); in this case, $v = \sigma' u_1 u_2$.

(iii) $su = s_1 s_2 u_1\sigma u_2\sigma' u_3$. By Lemma 4, we have $s_1 s_2 u_1\sigma\sigma' u_2 u_3 \in L_m(\mathbf{SUP}'_d)$. Also, the rest is similar to case (1); in this case, $v = u_1\sigma\sigma' u_2 u_3$.

(4) Let $s \in P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))$ and $s\sigma \in L(\mathbf{NSUP})$; we show that $s\sigma \in P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))$ by contraposition. Assume that $s\sigma \notin P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))$. Write $\mathbf{CH}_d(j,\sigma,i) = (C_d, \Sigma_{ch}, \tau_d, c_{d,0}, \{c_{d,0}\})$ where $\Sigma_{ch} = \{\sigma, tick, \sigma'\}$. We claim that $\tau_d(c_{d,0}, P_{ch}s) \neq c_{d,0}$; otherwise, σ is defined at state $\tau_d(c_{d,0}, P_{ch}s)$ and $s\sigma \in P_{ch}^{-1}L(\mathbf{CH}_d(j,\sigma,i))$. By inspection of the transition diagrams of $\mathbf{CH}_d(j,\sigma,i)$ and $\mathbf{CH}_{d+1}(j,\sigma,i)$, it results from $\tau_d(c_{d,0}, P_{ch}s) \neq c_{d,0}$, that $\tau_{d+1}(c_{d+1,0}, P_{ch}s) \neq c_{d+1,0}$. Hence, $s\sigma \notin P_{ch}^{-1}L(\mathbf{CH}_{d+1}(j,\sigma,i))$, in contradiction to the fact that $\mathbf{SUP}$ is $(d+1)-$bounded delay-robust. ■

APPENDIX B
PROOF OF LEMMA 2

Since $\mathbf{SUP}$ is not delay-robust wrt. $\mathbf{CH}(j,\sigma,i)$, by Definition 1, one of the conditions (12)-(15) is violated. In the following, we prove that in each case, $d_{max} \leq 2^m * m$, where m is the states number of $\mathbf{SUP}'$ in (11).

(1) Condition (12) is violated. Since that $L(\mathbf{SUP}) \subseteq PL(\mathbf{SUP}')$ always holds (similar to Lemma 3), we have $PL(\mathbf{SUP}') \nsubseteq L(\mathbf{SUP})$. So, there exists at least one string $s \in \Sigma'^*$ such that $s \in L(\mathbf{SUP}')$, but $Ps \notin L(\mathbf{SUP})$. We claim that s can be written as $s_1\sigma w$ where $s_1, w \in \Sigma'^*$; otherwise, s does not contain any σ, and it follows from the construction of $\mathbf{SUP}'$ that $Ps \in L(\mathbf{SUP})$, a contradiction. As illustrated in Fig. 12, we prove in the following that there exist strings $s'_1 \in L(\mathbf{SUP}')$ and $w' \in \Sigma'^*$ such that $\#tick(w') \leq 2^m * m$ (where $\#tick(w')$ represents the number of events $tick$ appearing in string w'), $s'\sigma w' \in L(\mathbf{SUP}')$, but $P(s'\sigma w') \notin L(\mathbf{SUP})$, from which we can conclude: to prevent the occurrence

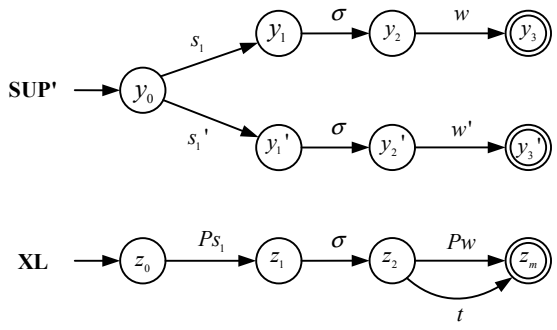

Fig. 12. $Ps_1 = Ps_1'$, $Pw' = t$ and t is a simple string.

of string $s_1'\sigma w'$, the maximal communication delay of σ must be less than $\#tick(w') \leq 2^m * m$, i.e. $d_{max} \leq 2^m * (m' + 1)$.

By $s_1\sigma w \in L(\mathbf{SUP}')$ and $P(s_1\sigma w) \notin L(\mathbf{SUP})$, we have $P(s_1\sigma w) \in PL(\mathbf{SUP}') \cap (\Sigma^* - L(\mathbf{SUP}))$. To identify such strings, we build an TDES $\mathbf{XL} = (Z, \Sigma, \zeta, z_0, Z_m)$ such that

$$L_m(\mathbf{XL}) = PL(\mathbf{SUP}') \cap (\Sigma^* - L(\mathbf{SUP}))$$

and

$$L(\mathbf{XL}) = PL(\mathbf{SUP}'),$$

i.e., $P(s_1\sigma) \in L(\mathbf{XL})$, and $P(s_1\sigma w) \in L_m(\mathbf{XL})$.

First, we build $\mathbf{XA}$ such that $L_m(\mathbf{XA}) = PL(\mathbf{SUP}')$ and $L(\mathbf{XA}) = L_m(\mathbf{XA})$ by the following two steps: (i) construct $\mathbf{PSUP}'$ by applying the subset construction algorithm on $\mathbf{SUP}'$ with natural projection P, and (ii) obtain $\mathbf{XA}$ by marking all states of $\mathbf{PSUP}'$. Second, we build $\mathbf{XB}$ such that $L_m(\mathbf{XB}) = \Sigma^* - L(\mathbf{SUP})$ and $L(\mathbf{XB}) = \Sigma^*$ by first adjoining a (non-marker) dump state $\hat{q}$ to the state set of $\mathbf{SUP}$ and transitions $(q, \sigma, \hat{q})$ for each state q of $\mathbf{SUP}$ if $\sigma \in \Sigma$ is not defined at q (i.e. $L(\mathbf{XB}) = \Sigma^*$), and secondly setting $\hat{q}$ to be the only marker state. Third, let $\mathbf{XL} = \mathbf{XA}||\mathbf{XB}$; then $L_m(\mathbf{XL}) = PL(\mathbf{SUP}') \cap (\Sigma^* - L(\mathbf{SUP}))$, $L(\mathbf{XL}) = PL(\mathbf{SUP}')$. The state size $|Z| \leq 2^m * (m' + 1)$, since $\mathbf{XA}$ has at most 2^m states (due to the subset construction algorithm), and $\mathbf{XB}$ has $m' + 1$ states .

Finally, by $P(s_1\sigma) \in PL(\mathbf{SUP}') = L(\mathbf{XL})$, there exists a state $z_2 \in Z$ such that $z_2 = \zeta(z_0, P(s\sigma))$; by $P(s_1\sigma w) \in L_m(\mathbf{XL})$, there exists a marker state $z_m \in Z_m$ such that $z_m = \zeta(z_0, P(s_1\sigma w)) = \zeta(z_2, P(w))$. So, there exists at least a *simple string*[4] $t \in \Sigma^*$ joining z_2 and z_m such that $z_m = \zeta(z_2, t)$,

[4]The concept 'simple string' is derived from the 'simple path' in graphic theory, where a path is called *simple* if no vertex is traversed more than once[24]. Here string t is called *simple* if no state is traversed more than once.

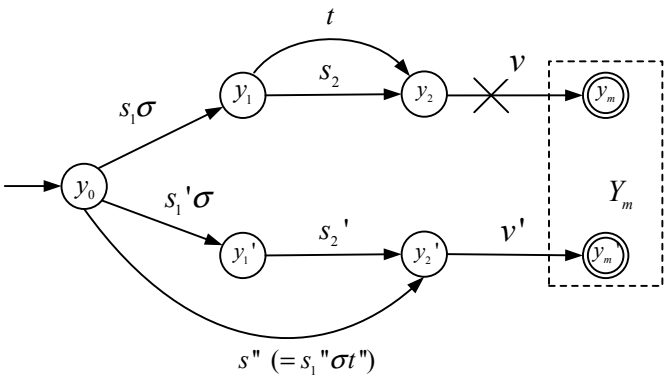

Fig. 13. P-Normality of $\mathbf{SUP}'$

and thus $P(s_1\sigma)t \in L_m(\mathbf{XL})$. It follows that $(Ps_1)\sigma t \in PL(\mathbf{SUP}') \cap (\Sigma^* - L(\mathbf{SUP}))$. So, there exist strings $s_1', w' \in \Sigma'^*$ such that $Ps_1' = Ps_1$, $Pw' = t$, $s_1'\sigma w' \in L(\mathbf{SUP}')$, and $P(s_1'\sigma w') \notin L(\mathbf{SUP})$, namely the occurrence of w' after $s_1'\sigma$ violates condition (12). Since t is simple, we have $\#tick(t) \leq |Z| \leq 2^m * (m'+1)$. By $Pw' = t$, we have $\#tick(w') = \#tick(t) \leq 2^m * (m'+1)$. Furthermore, since $\mathbf{SUP}'$ represents the system behavior with communication delay, we always have $m'+1 \leq m$. So $\#tick(w') \leq 2^m * m$, as required.

(2) Condition (13) is violated. $d_{max} \leq m * 2^m$ can be confirmed similar to case (1).

(3) Condition (14) is violated. Since delay-robustness of $\mathbf{SUP}$ is violated by the communication delay of σ, there must exist strings s_1, s_2, and w, such that $s_1\sigma s_2 \in L(\mathbf{SUP}')$ and $P(s_1\sigma s_2)w \in L_m(\mathbf{SUP})$, but no string v satisfies that $Pv = w$ and $s_1\sigma s_2 v \in L_m(\mathbf{SUP}')$. As illustrated in Fig.13, we prove in the following that the condition (14) is also violated by the string pair $s_1\sigma t$ and $s_1''\sigma t''$ where $\#tick(t) \leq 2^m * m$ and $\#tick(t'') \leq 2^m * m$, from which we conclude: to prevent the occurrences of the strings $s_1\sigma t$ and $s_1''\sigma t''$, the communication delay of σ must be less than $min(\#tick(t), \#tick(t'')) \leq 2^m * m$, i.e. $d_{max} \leq m * 2^m$.

To that end, we need the concept 'normal automaton'[25]. For $\mathbf{SUP}' = (Y, \Sigma', \eta, y_0, Y_m)$, we say that $\mathbf{SUP}'$ is P-normal if

$$(\forall s, t \in L(\mathbf{SUP}'))R(s) \neq R(t) \Rightarrow R(s) \cap R(t) = \emptyset \tag{27}$$

where $R(s) := \{y \in Y | y = \eta(y_0, s'), Ps = Ps'\}$. In case $\mathbf{SUP}'$ is not P-normal, replace $\mathbf{SUP}'$ by $\mathbf{SUP}'||\mathbf{PSUP}'$ where $\mathbf{PSUP}'$ is a deterministic generator over Σ obtained by the subset construction. $\mathbf{SUP}'||\mathbf{PSUP}'$ is always P-normal, and $L(\mathbf{SUP}') = L(\mathbf{SUP}'||\mathbf{PSUP}')$ and $L_m(\mathbf{SUP}') = L_m(\mathbf{SUP}'||\mathbf{PSUP}')$. The state size of the new $\mathbf{SUP}'$ is at most $m * 2^m$.

By $P(s_1\sigma s_2)w \in L_m(\mathbf{SUP}) \subseteq PL_m(\mathbf{SUP}')$, there must exist strings s_1', s_2', and v' such that $Ps_1' = Ps_1$, $Ps_2' = Ps_2$, $Pv' = w$, and $s_1'\sigma s_2'v' \in L_m(\mathbf{SUP}')$, as displayed in Fig. 13. Let $y_1 = \eta(y_0, s_1\sigma)$, $y_2 = \eta(y_1, s_2)$, $y_1' = \eta(y_0, s_1'\sigma)$, and $y_2' = \eta(y_1', s_2')$. Joining y_1 and y_2, there must exist a simple string t such that $y_2 = \eta(y_1, t)$. So, $R(s_1\sigma s_2) \cap R(s_1\sigma t) = y_2$. By P-normality of $\mathbf{SUP}'$, there must exist a string $s'' \in L(\mathbf{SUP}')$ such that $y_2' = \eta(y_0, s'')$, $P(s_1\sigma t) = P(s'')$, and $y_2' \in R(s_1\sigma t)$. So string s'' can be written as $s_1''\sigma t''$ where $Ps_1'' = Ps_1$ and $Pt'' = Pt$, and the condition (14) is also violated by the string pair $s_1\sigma t$ and $s_1''\sigma t''$. Because t is simple, $\#tick(t) \leq m$, where m is the state size of P-normal form of $\mathbf{SUP}'$. So, when $\mathbf{SUP}'$ is not P-normal, $\#tick(t) \leq m * 2^m$. In addition, since $Pt'' = Pt$, $\#tick(t'') = \#tick(t) \leq m * 2^m$, as required.

(4) Condition (15) is violated. In this case, assume that σ is blocked at state y of $\mathbf{SUP}'$, and the last occurrence of σ occurs at state y' of $\mathbf{SUP}'$. From y' to y, there must exist a simple string t. We claim that the maximal communication delay of σ must be less that $\#tick(t)$; otherwise, the system will arrive at state y by string t. Hence $d_{max} \leq \#t(tick) \leq m$.

Finally, by comparing d_{max} in the above four cases, we conclude that if $\mathbf{SUP}$ is not delay-robust with respect to $\mathbf{CH}(j, \sigma, i)$, $d_{max} \leq m * 2^m$.

REFERENCES

[1] K. Cai and W. M. Wonham, "Supervisor localization: a top-down approach to distributed control of discrete-event systems," *IEEE Trans. on Automatic Control*, vol. 55, no. 3, pp. 605–618, March 2010.

[2] K. Cai and W. Wonham, "Supervisor localization for large discrete-event systems: case study production cell," *International J. of Advanced Manufacturing Technology*, vol. 50, no. 9-12, pp. 1189–1202, October 2010.

[3] R. Zhang, K. Cai, Y. Gan, Z. Wang, and W. Wonham, "Supervision localization of timed discrete-event systems," *Automatica*, vol. 49, no. 9, pp. 2786–2794, 2013.

[4] K. Cai, R. Zhang, and W. Wonham, "Supervision localization of timed discrete-event systems," in *Proc. 2013 American Control Conference*, Washington DC, 2013, pp. 5666–5671.

[5] R. Zhang, K. Cai, Y. Gan, Z. Wang, and W. Wonham, "Distributed supervisory control of discrete-event systems with communication delay," 2014, technical report, online at http://arxiv.org/abs/1207.5072.

[6] ——, "Checking delay-robustness of distributed supervisors of discrete-event systems," in *Proc. Int. Conf. on Information Science and Control Engineering*, Shenzhen, China, 2012, pp. 350–355.

[7] R. Zhang, K. Cai, and W. Wonham, "Delay-robustness in distributed control of timed discrete-event systems based on supervisor localization," accepted by the 53rd IEEE Conference on Decision and Control, 2014.

[8] B. Brandin and W. Wonham, "Supervisory control of timed discrete-event systems," *IEEE Trans. on Automatic Control*, vol. 39, no. 2, pp. 329–342, February 1994.

[9] W. Wonham, *Supervisory Control of Discrete-Event Systems*. Systems Control Group, ECE Dept, Univ. Toronto, Toronto, ON, Canada, July 2014, available at http://www.control.utoronto.ca/DES.

[10] G. Barrett and S. Lafortune, "Decentralized supervisory control with communicating controllers," *IEEE Trans. on Automatic Control*, vol. 45, no. 9, pp. 1620–1638, September 2000.

[11] S. Tripakis, "Decentralized control of discrete-event systems with bounded or unbounded delay communication," *IEEE Trans. on Automatic Control*, vol. 49, no. 9, pp. 1489–1501, September 2004.

[12] K. Schmidt, E. Schmidt, and J. Zaddach, "A shared-medium communication architecture for distributed discrete event systems," in *Proc. Mediterranean Conf. on Control and Automation*, Athens, Greece, 2007, pp. 1–6.

[13] S. Xu and R. Kumar, "Asynchronous implementation of synchronous discrete event control," in *Proc. 9th Int. Workshop on Discrete Event Systems (WODES'08)*, 2008, pp. 181–186.

[14] K. Hiraishi, "On solvability of a decentralized supervisory control problem with communication," *IEEE Trans. on Automatic Control*, vol. 54, no. 3, pp. 468–480, March 2009.

[15] G. Kalyon, T. L. Gall, H. Marchand, and T. Massart, "Synthesis of communicating controllers for distributed systems," in *Proc. 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, Orlando, FL, USA, December 2011.

[16] F. Lin, "Control of networked discrete event systems: dealing with communication delays and losses," *SIAM J. Control and Optimization*, vol. 52, no. 2, pp. 1276–1298, 2014.

[17] P. Darondeau and L. Ricker, "Distributed control of discrete-event systems: A first step," *Transactions on Petri Nets and Other Models of Concurrency*, vol. 6, pp. 24–45, 2012.

[18] W. Sadid, L. Ricker, and S. Hashtrudi-Zad, "Robustness of synchronous communication protocols with delay for decentralized discrete-event control," *Discrete Event Dynamic Systems*, 2014, published online, available at http://link.springer.com.

[19] K. Wong and W. Wonham, "On the computation of observers in discrete-event systems," *Discrete Event Dynamic Systems*, vol. 14, no. 1, pp. 55–107, January 2004.

[20] L. Feng and W. M. Wonham, "Supervisory control architecture for discrete-event systems," *IEEE*

Trans. Autom. Control, vol. 53, no. 6, pp. 1449–1461, 2008.

[21] L. Feng and W. Wonham, "On the computation of natural observers in discrete-event systems," *Discrete Event Dynamic Systems*, vol. 20, no. 1, pp. 63–102, March 2010.

[22] H. Bravo, A. da Cunha, P. Pena, R. Malik, and J. Cury, "Generalised verification of observer property in discrete event systems," in *Proc. 11th International Workshop on Discrete Event Systems(WODES2012)*, Guadalajara, Mexico, October 2012, pp. 337–342.

[23] A. Afzalian, A. Saadatpoor, and W. Wonham, "Systematic supervisory control solutions for under-load tap-changing transformers," *Control Engineering Practice*, vol. 16, pp. 1035–1054, 2008.

[24] G. Danielson, "On finding simple paths and circuits in a graph," *IEEE Trans. on Circuit Theory*, vol. 15, no. 3, pp. 294–295, 1968.

[25] S. Takai and T. Ushio, "Effective computation of Lm(g)-closed, controllable, and observable sublanguage arising in supervisory control," *Systems & Control Letters*, vol. 49, no. 3, pp. 191–200, 2003.