Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Formal Modeling and Synthesis of State-Transferring Communication among Decentralized Supervisors for Discrete-Event Systems

A. Mannani and P. Gohari

***Abstract*— This work proposes to model and synthesize communicating decentralized supervisors for a Discrete-Event System (DES) within the framework of Distributed Supervised Discrete-Event Systems (DSDESs), which was introduced by the authors. To this end, first a Polynomial Dynamical System (PDS) representation for DSDESs is derived, which reveals the informational dependencies of distributed supervisors. To serve these dependencies, communication between every two supervisors is modeled by a communication event, whose semantics is defined by a map from observable events of the issuer of the communication to its variables, employed by the PDS representation. Thereby, the synthesis of communicating decentralized supervisors is reduced to the design of these maps using standard algebraic tools. The approach is illustrated through formal design of an information policy.**

I. INTRODUCTION

Supervisory Control Theory (SCT) seeks to minimally restrict the behavior of a DES, called plant, within the language of a given specification by designing a (centralized) supervisor, which disables some of the plant's transitions. In the absence of global observation of the plant's behavior, a finite set of decentralized supervisors, each observing the behavior locally, has to be designed such that their synchronous supervision leads to the same closed-loop behavior which is enforced by the centralized supervisor. If the specification is not coobservable [1], i.e. some illegal plant's moves cannot be distinguished from legal moves by any of the supervisors which can disable them, the supervisors need to communicate to each other to meet the specification.

The study of communication should specify the sender, receiver, content, time, and the order of communication [2]. Most pioneering research, listed in [3], rely on behavioral formulations and are limited to high-level algebraic structures of DESs. In the authors' viewpoint, this study would be more fruitful if the algebraic structures of the supervisors are utilized in more details. The idea is to encode the states of the corresponding centralized supervisor in a distributed way and represent its observation and control tasks as dynamic and algebraic equations. This makes the dynamics-related information amenable to algebraic characterizations and the study systematic, simplified and practically appealing [4].

In [4], the authors introduced distributed EFSM framework which, using an Agent-wise Labeling Map (ALM), represents the state structure of a centralized supervisor in a distributed way, and captures its observation and control information as *guards* and *actions* over Boolean variables, respectively. DSDES framework, then proposed in [3], employs *guard* and *updating functions*, defined on integer labels. Utilizing the meaningful and compact representation of a state, DSDES framework improves mathematical proofs and computations and, on top of its qualitative-like vantage point to system representation, it can be readily put into concrete implementation as distributed EFSMs.

Continuing the authors' work on modeling and synthesis of communicating supervisors [3], [4], this paper has two contributions. First, it introduces a PDS representation of a DSDES by formulating guard and updating functions as polynomial equations over a finite field. Then this representation is used to study state-transferring communication between each two supervisors. Communication between two supervisors helps the receiver reevaluate its polynomial equations, and is modeled as an event, whose semantics is defined by a map from the sender's observable events to its state-encoding Boolean variables. Given a PDS representation, the synthesis of decentralized supervisors is reduced to the design of these maps over algebraic structures such as fields. This is justified by designing an *information policy*.

After reviewing the DSDES framework of [3] in Section II, Section III presents a PDS formulation of DSDESs. Section IV formalizes the communication among supervisors and derives a solution.

II. DSDES FRAMEWORK

Let Σ be a finite alphabet and $L \subseteq \Sigma^*$ be plant's behavior. Consider a network consisting of distributed sensors and actuators as means to observe and control, respectively, the plant's behavior by n supervisors. Denote by $\mathbf{S}_i$ the i'th supervisor in the network, where $i \in I = \{1, 2, \ldots, n\}$. Associate with $\mathbf{S}_i$ observable and controllable event subsets $\Sigma_{o,i}$ and $\Sigma_{c,i}$, respectively, where $\Sigma_{o,i}, \Sigma_{c,i} \subseteq \Sigma$. Thus, from the viewpoint of the i'th supervisor we have $\Sigma_{uo,i} = \Sigma \setminus \Sigma_{o,i}$ and $\Sigma_{uc,i} = \Sigma \setminus \Sigma_{c,i}$. Define $\Sigma_i = \Sigma_{c,i} \cup \Sigma_{o,i}$. Associated with each event σ denote by $I_o(\sigma)$ the set of all sensors which can observe σ, i.e. $I_o(\sigma) = \{i \in I \mid \sigma \in \Sigma_{o,i}\}$. We define the centralized supervisor, denoted by $\mathbf{S}$, to be one which has access to all sensors' observations and can exercise control over all controllable events. For this supervisor we define $\Sigma_c = \bigcup_{i \in I} \Sigma_{c,i}$, $\Sigma_o = \bigcup_{i \in I} \Sigma_{o,i}$, $\Sigma_{uo} = \Sigma \setminus \Sigma_o$, and $\Sigma_{uc} = \Sigma \setminus \Sigma_c$. $\mathbf{S}$ is modeled by an automaton $\mathbf{S} = (R, \Sigma, \xi, r_0, R_m)$, where R is the finite set of states, r_0 is the initial state, R_m is the set of marked states, and $\xi : R \times \Sigma \to R$ is the partial transition function[1]. Let

This work is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

The first author is with the ECE Dept., Concordia University, Montreal, QC, Canada, H3G 1M8, amin_man@ece.concordia.ca. The second author was with the same institute, but unfortunately passed away in November 2008.

[1]Same hold for any recognizers such as $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$, too

$\mathbb{N} = \{0, 1, 2\cdots\}$. Denote by $\underline{v} = (v_1, \ldots, v_n) \in \mathbb{N}^n$ a vector of n natural numbers and let $\underline{0}$ denote a vector of n zeros. Consider a map $\pi_i : \mathbb{N}^n \to \mathbb{N}$ such that $\pi_i(\underline{v}) = v_i$ which picks the ith component of v, and extend π_i to a map $pwr(\mathbb{N}^n) \to pwr(\mathbb{N})$. The prefix closure of a language $L \in \Sigma^*$ is shown by $\overline{L}$.

A Distributed SDES (DSDES) is denoted by $\mathcal{D} = \{\mathcal{D}_i\}_{i\in I}$, where each quadruple $\mathcal{D}_i = (\Sigma, L, \mathcal{A}_i, \mathcal{G}_i)$ is defined as follows. Σ is a finite set of events (alphabet), L is a (regular) language defined over Σ, i.e. $L \subseteq \Sigma^*$, $\mathcal{A}_i : \Sigma_i \times \mathbb{N}^n \to \mathbb{N}$ is an *updating* function, and $\mathcal{G}_i : \Sigma_i \to pwr(\mathbb{N}^n)$ is a *guard* function. We extend the domain of $\mathcal{A}_i$ and $\mathcal{G}_i$ to the alphabet of all events. Define $\hat{\mathcal{A}}_i : \Sigma \times \mathbb{N}^n \to \mathbb{N}$ and $\hat{\mathcal{G}}_i : \Sigma \to pwr(\mathbb{N}^n)$ according to: for $\sigma \in \Sigma$ and $\underline{v} \in \mathbb{N}^n$,

$$\hat{\mathcal{A}}_i(\sigma, \underline{v}) = \begin{cases} \mathcal{A}_i(\sigma, \underline{v}) & ; \sigma \in \Sigma_i \\ \pi_i(\underline{v}) & ; \sigma \notin \Sigma_i \end{cases}, \hat{\mathcal{G}}_i(\sigma) = \begin{cases} \mathcal{G}_i(\sigma) & ; \sigma \in \Sigma_i \\ \mathbb{N}^n & ; \sigma \notin \Sigma_i \end{cases} \quad (1)$$

In the natural recursive way, $\mathcal{A}_i$ is extended to $\hat{\mathcal{A}}_i : \Sigma^* \times \mathbb{N}^n \to \mathbb{N}$. We shall use $\mathcal{A}_i$ and $\mathcal{G}_i$ to denote $\hat{\mathcal{A}}_i$ and $\hat{\mathcal{G}}_i$, respectively. Define a map $\mathcal{A} : \Sigma^* \times \mathbb{N}^n \to \mathbb{N}^n$ recursively as follows: for all $\underline{v} \in \mathbb{N}^n$, $s \in \Sigma^*$, and $\sigma \in \Sigma$

$$\mathcal{A}(\epsilon, \underline{v}) = \underline{v};\ \mathcal{A}(s\sigma, \underline{v}) = \Big(\mathcal{A}_i(\sigma, \mathcal{A}(s, \underline{v}))\Big)_{i\in I}. \quad (2)$$

Associated with each $i \in I$, a DSDES owns guard and updating functions to capture control and observation, respectively. Control for each $\mathcal{D}_i$ is based upon n-vectors of natural numbers; component i of a vector is updated with $\mathcal{A}_i$.

The semantics of $\mathcal{D}$ is as follows: to each string $s \in \Sigma^*$ a label $\mathcal{A}(s, 0)$ is attached. Thus, starting recursively from ϵ, if s is in the behavior of $\mathcal{D}$ and $\sigma \in \Sigma$ is eligible in $\overline{L}$ after s (i.e. $s\sigma \in \overline{L}$), then σ is "enabled" if the label of s is in the image of σ under the guard function, i.e. $\mathcal{A}(s, 0) \in \mathcal{G}(\sigma)$. When σ is taken, the label of $s\sigma$ is computed according to $\mathcal{A}(s\sigma, 0) = \mathcal{A}(\sigma, \mathcal{A}(s, 0))$. The behavior of $\mathcal{D}$ is a subset of L. A DSDES is obtained by *guarding* events, i.e. limiting their occurrence, based on the *observation* of event sequences of the guarded language. Thereby, a DSDES is equipped with means to *control* and *observe* a given behavior L; in other words, a DSDES may be used to *implement* the control decisions of an already designed supervisor for L, and thus it is suitable to model a closed-loop DES.

Problem 1 *DSDES Control problem*: Let a proper, feasible and admissible centralized supervisor $\mathbf{S} = (R, \Sigma, \xi, r_0, R_m)$ enforce a specification E for a plant $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$. Design guard and updating functions for each $\mathcal{D}_i = (\Sigma, L_m(G), \mathcal{A}_i, \mathcal{G}_i)$ s.t. $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. □

The solution to this problem relies on the assignment of integer vector labels to the states of $\mathbf{S}$ using ALMs. An ALM for $\mathbf{S}$ is a map $\ell : R \to pwr(\mathbb{N}^n)$ such that $\underline{0} \in \ell(r_0)$,

- $\forall r, r' \in R.\ r \neq r' \Rightarrow \ell(r) \cap \ell(r') = \emptyset$, and
- $\forall r, r' \in R, r \neq r', \forall\sigma \in \Sigma_o, \forall\underline{v} \in \mathbb{N}^n.\ \ \underline{v} \in \ell(r) \wedge r' = \xi(r, \sigma) \implies \exists! \underline{v}' \in \mathbb{N}^n.\ \underline{v}' \in \ell(r') \wedge$ $[\forall i \in I_o(\sigma).\ v_i \neq v'_i] \ \wedge\ [\forall j \in I \setminus I_o(\sigma).\ v_j = v'_j]$.

The fact that a finite ALM, i.e. one with finite image, exists for every $\mathbf{S}$ [4], paves the way for defining the updating functions associated with $\mathcal{D}$. To this end, a map $\mu : \Sigma \times \mathbb{N}^n \to \mathbb{N}^n$ can be defined such that

$$\forall r, r' \in R,\ \forall\sigma \in \Sigma,\ \forall\underline{v} \in \mathbb{N}^n. \quad \underline{v} \in \ell(r) \wedge r' = \xi(r, \sigma) \implies [\mu(\sigma, \underline{v}) \in \ell(r') \ \wedge\ \big(\forall i \in I_o(\sigma).\ \pi_i(\mu(\sigma, \underline{v})) \neq \pi_i(\underline{v})\big) \wedge \big(\forall j \in I \setminus I_o(\sigma).\ \pi_j(\mu(\sigma, \underline{v})) = \pi_j(\underline{v})\big)].$$

The updating functions can be defined using map μ:

$$\forall r, r' \in R, \forall\sigma \in \Sigma, \forall\underline{v} \in \mathbb{N}^n.\ r' = \xi(r, \sigma) \ \wedge\ \underline{v} \in \ell(r) \implies \mathcal{A}_i(\sigma, \underline{v}) = \pi_i(\mu(\sigma, \underline{v})). \quad (3)$$

Proposition 1 *A solution to Problem 1:* For all $i \in I$, let $\mathcal{A}_i$ be as in (3), and $\mathcal{G}_i$ be as follows: For every $\sigma \in \Sigma$

$$\mathcal{G}_i(\sigma) = \begin{cases} \{\ell(r) \mid r \in R \wedge \xi(r, \sigma)!\}; & \text{if } \sigma \in \Sigma_{c,i}, \\ \cup_{r\in R}\ell(r); & \text{if } \sigma \in \Sigma_{uc,i}. \end{cases} \quad (4)$$

Then $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. ■

Using the definitions of μ and $\mathcal{A}_i$ we can prove the following.

$$\forall i \in I, \forall\underline{v} \in \mathbb{N}^n, \forall\sigma \in \Sigma_{uo,i}.\ \ \mathcal{A}_i(\sigma, \underline{v}) = v_i \quad (5)$$

III. PDS REPRESENTATION OF A DSDES

Proposition 1 insures that the choice of updating and guard functions in (3) and (4) would indeed implement the centralized supervisory control in a decentralized way. Although these two kinds of functions provide a compact way to present observation- and control-related information using a finite number of integer (vector) labels, it should be clear that the ALM does not rely on the absolute positions of these labels. In fact, it is the relative positions of them which reflect the dynamic information structure, which, as in the case of (state-space) dynamical systems in control theory, could serve more control purposes if put in a *symbolic* form. To this end, once the labels of the states of $\mathbf{S}$ are represented by vectors of integer variables, the observation- and control-related information may be captured by (polynomial) functions over the underlying finite field from which the label values are selected. Accordingly, one may compute for the guard and updating functions associated to an event σ their characteristic equation and transition function, respectively. Thereby, system information can be accessed and studied in a unified manner using the algebraic structure of finite fields and their well developed software tools.

A. Finite field formulation for a DSDES

In the following we derive a polynomial representation of a DSDES over a finite field using a simple, yet general method called *interpolation polynomials in the Lagrange form* [5].

Notation 1 Let $p \geq 2$ be a natural number, $\mathbb{F}_p$ be a finite field of p integers with addition defined modulo p, x_i ($i \in I$) be a variable taking values from $\mathbb{F}_p$, $\mathbf{x} = (x_1, \cdots, x_n)'$, and $\mathbb{F}_p[\mathbf{x}]$ be the ring of polynomials in the variables $x_1, \cdots, x_n$ and coefficients taken from $\mathbb{F}_p$ [6]. For a function or formula f, by writing $f(\mathbf{x})$ we mean that f can in general depend on some or all of the elements of $\mathbf{x}$. The set of variables on which f precisely depends is denoted by either $\arg(f)$ or explicit listing of such variables. Let x_i be $\mathbf{S}_i$'s private variable, with respect to which all x_j's, $j \in I \setminus \{i\}$, are referred to as *external* variables. For an event $\sigma \in \Sigma$, the polynomials corresponding with $\mathcal{A}_i(\sigma, .)$ and $\mathcal{G}_i(\sigma)$ are denoted by $\mathfrak{a}_i^\sigma(\mathbf{x})$ and $\mathfrak{g}_i^\sigma(\mathbf{x})$, respectively. □

Given the graph of a function $f : \mathbb{F}_p^n \to \mathbb{F}_p$ as a set $\mathcal{U} = \{(\mathbf{u}, y) \in \mathbb{F}_p^n \times \mathbb{F}_p \mid y = f(\mathbf{u})\}$, the method of "interpolation polynomials in the Lagrange form" computes a polynomial

$q \in \mathbb{F}_p[\mathbf{x}]$ such that $q(\mathbf{u}) = f(\mathbf{u})$. The computation is based on the following fraction [7]

$$L_i(x) = \frac{x(x-1)\cdots(x-i+1)(x-i-1)\cdots(x-p+1)}{i(i-1)\cdots(i-(i-1))(i-(i+1))\cdots(i-p+1)}, \quad (6)$$

which is 1 at $x = i$ and 0 otherwise. Let $\mathbf{u} \in \mathbb{F}_p^n$ and define

$$L_{\mathbf{u}}(\mathbf{x}) = L_{u_1 u_2 \cdots u_n}(\mathbf{x}) = L_{u_1}(x_1) \cdots L_{u_n}(x_n), \quad (7)$$

where we assume that the subscripts of L correspond to its arguments in the order they appear. Observe that $L_{\mathbf{u}}(\mathbf{x})$ is 1 at $\mathbf{x}=\mathbf{u}$ and 0 otherwise. The required polynomial would be

$$q(\mathbf{x}) = \sum_{\mathbf{u} \in \mathbb{F}_p^n} L_{\mathbf{u}}(\mathbf{x}) f(\mathbf{u}). \quad (8)$$

Algorithm 1 *Computation of polynomial functions associated with guard and updating functions:* Given a DSDES with $\mathbf{S}$ and ℓ as in Proposition 1 and updating and guard functions as in (3) and (4), and using the interpolation polynomials in the Lagrange form, do the following.

1) For each $i \in I$ compte $V_i = \pi_i(\bigcup_{r \in R} \ell(r))$ and $p_i = \max_{v \in V_i} v$. Determine $p = \inf\{p' \in \mathbb{N} \mid [\forall i \in I.\, p' \geq p_i] \wedge \mathbb{F}_{p'}$ is a field$\}$. Choose $\mathbb{F}_p^n$ as the smallest common underlying finite field which accommodates the whole range of the label values assigned by ℓ.
2) For each $\mathbf{S}_i$ and associated with each event σ, define $\mathfrak{g}_i^\sigma$ as the characteristic polynomial associated with $\mathcal{G}_i(\sigma)$ in (4), and assign the polynomial arbitrarily elsewhere, i.e. for every $i \in I$ and $\sigma \in \Sigma$ the following holds.
$$\begin{cases} \mathfrak{g}_i^\sigma(\underline{v}) = 1; & \forall \underline{v} \in \mathcal{G}_i(\sigma) \\ \mathfrak{g}_i^\sigma(\underline{v}) = 0; & \forall \underline{v} \in \cup_{r \in R} \ell(r) \setminus \mathcal{G}_i(\sigma) \\ \mathfrak{g}_i^\sigma(\underline{v}) = \text{arbitrary}; & \forall \underline{v} \in \mathbb{F}_p^n \setminus \cup_{r \in R} \ell(r) \end{cases} \quad (9)$$
3) For each $\mathbf{S}_i$ and associated with each event σ, define $\mathfrak{a}_i^\sigma$ as the restriction of the $\mathcal{A}_i(\sigma, \underline{v})$ in (3) to $\mathbb{N}^n \to \mathbb{N}$, where σ-labeled transitions is eligible and arbitrary elsewhere, i.e. for every $i \in I$ and $\sigma \in \Sigma$ we have
$$\begin{cases} \mathfrak{a}_i^\sigma(\underline{v}) = \mathcal{A}_i(\sigma, \underline{v}); & \forall \underline{v} \in \mathcal{G}_i(\sigma) \\ \mathfrak{a}_i^\sigma(\underline{v}) = \text{arbitrary}; & \forall \underline{v} \in \mathbb{F}_p^n \setminus \mathcal{G}_i(\sigma) \end{cases} . \quad (10)$$
□

By the ordering of $\mathbb{N}$, p in Step 1 always exists. Notice that $V_i \subseteq \{0, 1, \cdots, p-1\}$ holds for every $i \in I$. Next we specify which polynomials serve to represent a DSDES.

Definition 1 Associated with Proposition 1, for each $\mathbf{S}_i$ and each $\sigma \in \Sigma$ let polynomial equations $x_i := \mathfrak{a}_i^\sigma(\mathbf{x})$ and $\mathfrak{g}_i^\sigma(\mathbf{x}) = 1$ over $\mathbb{F}_p^n$ replace the updating function $\mathcal{A}_i(\sigma, .)$ in (3) and guard function $\mathcal{G}_i(\sigma)$ in (4), respectively. The polynomials are said to *represent* the DSDES if they result in $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. □

Proposition 2 Let $\mathbf{G}$, E, $\mathbf{S}$, ℓ, be as in Proposition 1. The polynomial equations which are obtained by computation of $\mathfrak{a}_i^\sigma$ and $\mathfrak{g}_i^\sigma$ using Algorithm 1 represent the DESDES. ■

Remark 1 By (5), Algorithm 1 computes identity function for every updating function which is associated with a non-observable event. Similarly, by (4), the algorithm computes unity function as guard function of each uncontrollable event. These polynomials are not mentioned explicitly. □

Remark 2 (Computation of simplified polynomials) As we will see in Section IV, to reduce the communication among supervisors, it is often required that updating or guard polynomials depend on fewer number of external variables. In DES models transition relations are usually partial functions and this results in the existence of a number of unassigned points in the domain of the updating functions, generally called *arbitrary* or *don't care* cases [8]. Polynomial expressions for updating functions can be simplified by assigning to arbitrary cases, the points in codomain which make the transition structure symmetric, thus eliminating the dependency on some (external) variables. To simplify, using a symmetric structure, the expression for the guard function associated with an event σ, one may assume that σ is enabled at states which are not reachable (i.e. there is no $\sigma \in \Sigma$, $s \in L(\mathbf{S})$, $i \in I$, such that $\mathcal{A}(\sigma, \mathcal{A}(s, \underline{0})) = \underline{v} \wedge \mathcal{A}(s, \underline{0}) \in \mathcal{G}_i(\sigma)$) or at states at which σ is disabled by the plant. When symmetry is available, the identities

$$\sum_{i=0}^{p-1} L_i(x) = 1, \ \sum_{j=0}^{p-1} L_k(x) L_j(x') = L_k(x), \quad (k \in \mathbb{F}_p) \quad (11)$$

can be used to simplify polynomials. □

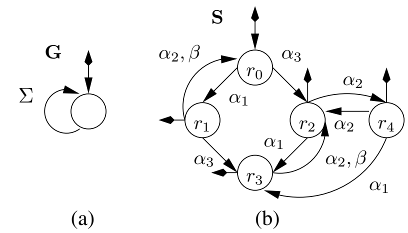

Fig. 1. (a) The plant's model. (b) The specification which is also a centralized supervisor.

Example 1 Figure 1-a shows the model of a distributed network consisting of three plant components and four events $\alpha_1, \alpha_2, \alpha_3$ and β, where $\Sigma_{c,1} = \Sigma_{o,1} = \{\alpha_1, \beta\}$, $\Sigma_{c,2} = \Sigma_{o,2} = \{\alpha_2, \beta\}$, and $\Sigma_{c,3} = \Sigma_{o,3} = \{\alpha_3\}$. The specification $\mathbf{S}$ is shown in part (b) and all its states are marked. Observe that $\mathbf{S}$ is a proper centralized supervisor enforcing itself. An ALM can be defined for $\mathbf{S}$ as follows (A point $(a, b, c) \in \mathbb{N}^3$ is denoted by 'abc') $\ell(r_0) = \{000, 210, 120\}$, $\ell(r_1) = \{100, 010, 220\}$, $\ell(r_2) = \{001, 211, 121\}$, $\ell(r_3) = \{101, 011, 221\}$, and $\ell(r_4) = \{201, 021, 111\}$. Correspondingly, guard functions are computed as follows: $\mathcal{G}_1(\alpha_1) = \ell(r_0) \cup \ell(r_2) \cup \ell(r_4)$, $\mathcal{G}_2(\alpha_2) = \ell(r_1) \cup \ell(r_2) \cup \ell(r_3) \cup \ell(r_4)$, $\mathcal{G}_1(\beta) = \mathcal{G}_2(\beta) = \ell(r_1) \cup \ell(r_3)$, $\mathcal{G}_3(\alpha_3) = \ell(r_0) \cup \ell(r_1)$. The updating functions are listed in Table I. Arbitrary cases are denoted by "$-$" and are used to simplify the guard and updating polynomials later. Table II shows the polynomials which represent the updating and guard functions. Here $\mathbf{x} = [x_1, x_2, x_3]'$, $V_1 = V_2 = \{1, 2, 3\}$, $V_3 = \{1, 2\}$, $p_1 = p_2 = 3$, and $p_3 = 2$. Step 1 of Algorithm 1 yields $p = 3$, i.e. $\mathbb{F}_3$ is the underlying field. ◊

The above "symbolic" formulation may be viewed as a PDS [9] in which equations associated with updating and guard functions represent the dynamics of the DES and its algebraic constraints, respectively.

Definition 2 [9] Let $\mathbf{x} = (x_1, \cdots, x_n)'$ and $\mathbf{y} = (y_1, \cdots, y_m)'$ be the vectors of *states* and *events*, respectively. Define $\mathcal{P}(.) = \big(\mathsf{p}_1(.), \cdots, \mathsf{p}_\nu(.)\big)'$ and $\mathcal{Q}(.) = \big(\mathsf{q}_1(.), \cdots, \mathsf{q}_\omega(.)\big)'$ as finite dimensional vectors of polynomials over $\mathbb{F}_p[\mathbf{x}, \mathbf{y}]$. The set of polynomial equations

$$\begin{cases} \mathbf{x} := \mathcal{P}(\mathbf{x}, \mathbf{y}) \\ \mathcal{Q}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \end{cases} \quad (12)$$

TABLE I

UPDATING FUNCTIONS ($a, b \in \{0,1,2\}$, $c \in \{0,1\}$)

$\underline{v}$	$00c$	$21c$	$12c$	201	111	021	else
$\mathcal{A}(\alpha_1, \underline{v})$	$10c$	$01c$	$22c$	101	011	221	–
$\underline{v}$	$01c$	$10c$	$22c$	021	111	201	001
$\mathcal{A}(\alpha_2, \underline{v})$	$00c$	$12c$	$21c$	001	121	211	021
$\underline{v}$	121	211	else				
$\mathcal{A}(\alpha_2, \underline{v})$	111	201	–				
$\underline{v}$	$10c$	$01c$	$22c$	else			
$\mathcal{A}(\beta, \underline{v})$	$21c$	$12c$	$00c$	–			
$\underline{v}$	$ab0$	else					
$\mathcal{A}(\alpha_3, \underline{v})$	$ab1$	–					

TABLE II

COMPUTED POLYNOMIALS FOR UPDATING AND GUARD FUNCTIONS

$$x_1 := \mathfrak{a}_1^{\alpha_1}(\mathbf{x}) = 2(x_2+2), \quad x_1 := \mathfrak{a}_1^{\beta}(\mathbf{x}) = x_1 + 1$$
$$x_2 := \mathfrak{a}_2^{\alpha_2}(\mathbf{x}) = x_1(x_1+1) + (x_2+2)(2x_1+x_2+1)$$
$$x_2 := \mathfrak{a}_2^{\beta}(\mathbf{x}) = x_2 + 1, \qquad x_3 := \mathfrak{a}_3^{\alpha_3}(\mathbf{x}) = 2(x_3^2+2)$$

$$\mathfrak{g}_1^{\alpha_1}(\mathbf{x}) = 2(x_1+2)^2(x_2^2+2) + 2x_1^2x_2(x_2+1) + 2(x_1+1)^2x_2(x_2+2)$$
$$\mathfrak{g}_2^{\alpha_2}(\mathbf{x}) = 2x_1^2(x_2^2+2) + 2(x_1+1)^2x_2(x_2+1) + 2(x_1+2)^2x_2(x_2+2) + 2x_3(x_3+1)[x_1x_2 + 2x_1^2 + 2x_2^2 + 1]$$
$$\mathfrak{g}_{1,2}^{\beta}(\mathbf{x}) = x_1(x_1+1)(x_2^2+2) + (x_1^2+2)x_2(x_2+1) + x_1(x_1+2)x_2(x_2+2)$$
$$\mathfrak{g}_3^{\alpha_3}(\mathbf{x}) = 2(x_3^2+2)$$

are said to be a PDS in *state explicit* form, where $:=$ denotes the assignment of new values to its left vector, $\mathbf{x}$, and the first and second equations are referred to as *state transition* equation and *constraints* equation, respectively. □

The construction using interpolation polynomials in the Lagrange form in Algorithm 1 yields the following.

$$\forall i \in I. \begin{cases} x_i := \mathfrak{a}_i^{\sigma}(\mathbf{x}) & ; \forall \sigma \in \Sigma_{o,i} \\ \mathfrak{g}_i^{\sigma}(\mathbf{x}) - 1 = 0 & ; \forall \sigma \in \Sigma_{c,i} \end{cases} \tag{13}$$

This, together with Remark 1, lead to the following result.

Theorem 3 Every DSDES can be represented as a PDS in state explicit form. ■

As Table II reads, the PDS representation of the DSDES in Example 1 consists of 5 dynamic and 4 algebraic equations. In the dynamics part, each event can only affect the variable(s) owned by the supervisor(s) which can observe its occurrence. Also the algebraic constraint associated with an event designates where in (here 3-dimensional) integer space that event is enabled.

B. EFSM implementation of a DSDES

EFSM framework [4], offers a bit-wise representation of supervisors' information which suits practical purposes (see [4]). The system information, which is captured by x_is, $i \in I$, in DSDES framework, can be implemented in EFSM framework using Boolean variables. This can be done by either direct encoding of the centralized supervisor or using the polynomials of the PDS representation of the associated DSDES. In the first approach [4], first the number of Boolean variables necessary for encoding the labels associated with each $\mathbf{S}_i$ is determined and then observation and control are captured by *actions* and *guards* on Boolean variables, respectively. Here, we use the second approach, which is introduced next.

Definition 3 ([4]-Definition 1 and Subsection III.A) *The EFSM formalism:* Let $\mathbb{B} = \{0, 1\}$, $h = \lceil log_2(p) \rceil$, $J = \{1, 2, \cdots, h\}$, and $i, j \in I$, $j \neq i$. Denote by $X_{ii} = \{x_{ii}^k \mid k \in J\}$ the set of $\mathbf{S}_i$'s *private* Boolean variables which encodes x_i such that $x_i = (x_{ii}^h \cdots x_{ii}^1)$. Denote by x_{ij}^k a copy of $x_{jj}^k \in X_{jj}$ stored by $\mathbf{S}_i$ and let $X_{ij} = \{x_{ij}^k \mid x_{jj}^k \in X_{jj}\}$, $X_{ci} = \bigcup_{j \in I \setminus \{i\}} X_{ij}$, $X_i = X_{ii} \dot{\cup} X_{ci}$, and $X = \bigcup_{i \in I} X_i$. Let G_i denote the set of all Boolean formulas over X_i and A_i denote the set of all Boolean functions $\mathsf{b} : \mathbb{B}^{nh} \to \mathbb{B}$. Associated with each $i \in I$, $\mathbf{M}_i = (\mathbf{S}, X_i, g_i, a_i)$ is an EFSM. Here $g_i : \Sigma \to \mathsf{G}$ assigns to each $\alpha \in \Sigma$ the *guard* (formula), $g_i(\alpha)|_{X_i}$, which is evaluated using the binary values of the variables in X_i and guards all the transitions labeled with α. Also $a_i : X_{ii} \times \Sigma \to \mathsf{A}$ assigns to each pair of $\alpha \in \Sigma$ and $x \in X_{ii}$ an *action* $a_i(x, \alpha) : \mathbb{B}^{nh} \to \mathbb{B}$, which is a Boolean function and, upon the occurrence of α, results in the assignment $x := a_i(x, \alpha)(X_i)$. It is assumed that for the network of EFSMs, $(\mathbf{M}_i)_{i \in I}$, X is initialized to zero. □

Remark 3 Following [4], for every $\sigma \in \Sigma_{uc,i}$, $\sigma' \in \Sigma_{uo,i}$, and $x \in X_i$, $g_i(\sigma)$ is an always-true formula and $a_i(x, \sigma')$ is an identity function, which are not mentioned explicitly. □

Definition 4 Let PDS (13) be defined on $\mathbb{F}_p^n$ and h, J, and for all $i \in I$, X_i, $\mathbf{M}_i$, and $x_i = (x_{ii}^h \cdots x_{ii}^1)$ be as in Definition 3. The network of EFSMs, $(\mathbf{M}_i)_{i \in I}$, implements the PDS if for every $i, j \in I, i \neq j$, $k \in J$, $\mathbf{x} \in \mathbb{F}_p^n$ and $\sigma \in \Sigma$ we have $x_{ij}^k = x_{jj}^k$, $[\mathfrak{g}_i^{\sigma}(\mathbf{x}) = g_i(\sigma)|_{X_i}]$ and $[x_i := \mathfrak{a}_i^{\sigma}(\mathbf{x}) \iff \forall k \in J.\ x_{ii}^k := a_i(x_{ii}^k, \sigma)(X_i)]$. □

Proposition 4 Associated with Proposition 1, let PDS (13) be computed using Algorithm 1 and replace (3) and (4). If the network of EFSMs, $(\mathbf{M}_i)_{i \in I}$, implements PDS (13), it holds that $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. ■

The following Algorithm computes an implementing EFSM.

Algorithm 2 *Computation of guards and actions in the EFSM framework from guard and updating functions of the DSDES framework:* Given PDS (13) and p from Algorithm 1,

1) Determine h using Definition 3.
2) For each $i \in I$, define X_i and encode each x_i using $(x_{ii}^h \cdots x_{ii}^1)$ as in Definition 3.
3) For each guard polynomial $\mathfrak{g}_i^{\sigma}$ ($i \in I$, $\sigma \in \Sigma_{c,i}$), compute $g_i(\sigma)$ as a function of the Boolean variables in $Y = \{x_{ii}^k, x_{ij}^k \mid k \in J, j \in I, x_j \in \arg(\mathfrak{a}_i^{\sigma})\}$ such that for every $x_i \in \mathbb{F}_p$ it holds that $\mathfrak{g}_i^{\sigma}(\mathbf{x}) = g_i(\sigma)|_Y$. For $i, j \in I$, take all encodings $(x_{ij}^1, \cdots, x_{ij}^h)$ of numbers $p, \cdots, 2^h$ as "don't care" cases, if there exists any.
4) For each updating polynomial $\mathfrak{a}_i^{\sigma}$ ($i \in I$, $\sigma \in \Sigma_{o,i}$), compute $a_i(x_{ii}^k, \sigma)$ as a function of the Boolean variables in $Z = \{x_{ii}^k, x_{ij}^k \mid k \in J, j \in I, x_j \in \arg(\mathfrak{a}_i^{\sigma})\}$ such that for every $x_i \in \mathbb{F}_p$, it holds that $x_i := \mathfrak{a}_i^{\sigma}(\mathbf{x})$ if and only if $\forall k \in J.\ x_{ii}^k := a_i(x_{ii}^k, \sigma)(Z)$. For $i, j \in I$, take all encodings $(x_{ij}^1, \cdots, x_{ij}^h)$ of numbers $p, \cdots, 2^h$ as "don't care" cases, if there exists any.
5) For every $i \in I$ and $\sigma \in \Sigma_{uo,i}$, set $a_i(x, \sigma)$ equal to identity function ($x \in X_{ii}$), and for every $i \in I$ and $\sigma \in \Sigma_{uc,i}$, set $g_i(\sigma) = 1$ (see Remark 3). □

Proposition 4 insures that under a correct update of copy variables, the actions and guards computed by Algorithm 2 implement PDS (13), as summarized in the following result.

TABLE III

ACTIONS AND GUARDS IN THE EFSM FRAMEWORK

$a_1(x_{11}^2, \alpha_1) = x_{12}^2, \quad a_1(x_{11}^1, \alpha_1) = \overline{x}_{12}^2 \overline{x}_{12}^1$
$a_1(x_{11}^2, \beta) = x_{11}^1, \quad a_1(x_{11}^1, \beta) = \overline{x}_{11}^2 \overline{x}_{11}^1$
$a_2(x_{22}^2, \alpha_2) = \overline{x}_{22}^2[\overline{x}_{21}^2 \overline{x}_{22}^1 + x_{21}^1]$
$a_2(x_{22}^1, \alpha_2) = x_{22}^2 x_{21}^1 + \overline{x}_{22}^1 x_{21}^2$
$a_2(x_{22}^2, \beta) = x_{22}^1, \quad a_2(x_{22}^1, \beta) = \overline{x}_{22}^2 \overline{x}_{22}^1$
$a_3(x_{33}^2, \alpha_3) = 0, \quad a_3(x_{33}^1, \alpha_3) = \overline{x}_{33}^2 \overline{x}_{33}^1 = \overline{x}_{33}^1$
$g_1(\alpha_1) = x_{11}^2 \overline{x}_{12}^2 + \overline{x}_{11}^2 x_{12}^2 + x_{11}^1 x_{12}^1 + \overline{x}_{11}^2 \overline{x}_{11}^1 \overline{x}_{12}^1$
$g_1(\beta) = x_{11}^1 \overline{x}_{12}^2 \overline{x}_{12}^1 + \overline{x}_{11}^2 \overline{x}_{11}^1 x_{12}^1 + x_{11}^2 x_{12}^2$
$g_2(\alpha_2) = x_{21}^1 \overline{x}_{22}^2 + \overline{x}_{21}^2 x_{22}^1 + \overline{x}_{21}^1 x_{22}^2 + x_{21}^2 \overline{x}_{22}^1 + x_{33}^1$
$g_2(\beta) = x_{21}^1 \overline{x}_{22}^2 \overline{x}_{22}^1 + \overline{x}_{21}^2 \overline{x}_{21}^1 x_{22}^1 + x_{21}^2 x_{22}^2$
$g_3(\alpha_3) = \overline{x}_{33}^2 \overline{x}_{33}^1 = \overline{x}_{33}^1$

Corollary 1 If for every $i, j \in I, i \neq j$, it holds that $x_{ij}^k = x_{jj}^k$, then the actions and the guards computed by Algorithm 2 implement PDS (13). ■

Example 2 Corollary 1 lets us employ Algorithm 2 to compute the EFSM implementation of the PDS in Example 1. Observe that $p = 3$, thus, $\mathbf{S}_1$, $\mathbf{S}_2$, and $\mathbf{S}_3$ each would require $h = \lceil log_2(3) \rceil = 2$ Boolean variables to implement their labels. By Definition 3, for $i \in I$ we have $X_{ii} = \{x_{ii}^2, x_{ii}^1\}$. Two variables in general can represent four integers. However there are only three labels in $\mathbb{F}_3$, i.e. 0, 1, and 2. The unused label, 3, may be used arbitrarily to simplify the expressions. Actions and guards are listed in Table III, where the complement of a variable x is denoted by $\overline{x}$. ◊

IV. MODELING AND SYNTHESIS OF STATE-TRANSFERRING COMMUNICATION

In DSDES framework, communication among decentralized supervisors is needed for reevaluation of their guard and updating functions. For example assume that the vector of values after a string s is observed is $\underline{v} := \mathcal{A}(s, \underline{0})$. Then $\sigma \in \Sigma_i$ is enabled at s if and only if $\underline{v} \in \mathcal{G}_i(\sigma)$. To determine if this is the case, $\mathbf{S}_i$ may need to receive the value v_j, for some $j \neq i$, from $\mathbf{S}_j$. When σ is taken, $\mathbf{S}_i$ updates v_i with the value $\mathcal{A}_i(\sigma, \underline{v})$. Again, to correctly evaluate $\mathcal{A}_i(\sigma, \underline{v})$, $\mathbf{S}_i$ may need to receive the value v_j, for some $j \neq i$, from $\mathbf{S}_j$.

In PDS (13), $\mathbf{S}_i$'s *informational dependency* is reflected in the functional forms of its updating and guard polynomials and on their dependencies on external variables. Such observations form the basis to model the communication, define the communication problem, and synthesize solutions for it.

Let the network of n supervisors be strongly connected, data transfer be instant with no loss, and disabling controllable events affects none of communication-related events (to keep the network connected). Assume a DSDES with PDS representation (13) over $\mathbb{F}_p^n$ and EFSM implementation (**??**).

A. Communication-related events

Denote a communication-related event which is sent from $\mathbf{S}_j$ to $\mathbf{S}_i$ by the subscript indices ji and assume that it is observable by both supervisors and controllable by $\mathbf{S}_j$. Although $\Sigma_{o,j}$, $\Sigma_{c,j}$, and $\Sigma_{o,i}$ should be enlarged by these new events, to keep the notations simpler we avoid introducing new sets and assume that this fact is clear from the context.

The event-driven nature of DESs limits the release of the system-related information to the occurrence of observable events such that only the supervisors, who observe them, gain information about the system's evolution. As an information-providing mean, communication should rely on the observation of the occurrence of events, i.e. issuing a communication event would follow the occurrence of an observable event, only. The semantics of each communication event is defined by a map from a subset of observable events to an information-containing set. For a DSDES, the system-related information is captured by $\mathbf{x}$. Therefore, the information transferred by a communication event, i.e. the elements of its image set, can be taken from (bit-wise encodings of) either $\mathbf{x}$, as provided in EFSM framework. We refer to this type of communication as *state-transferring* and formalize it here through designing the above-mentioned maps.

We distinguish two types of communication events which are referred to as $\mathscr{I}$ and $\mathscr{R}$ events. This classification divides the communication design into two levels of *information exchange* and *routing*, which, respectively address what information needs to be exchanged mutually among supervisors and how these exchanges can be performed using the available communication channels. We limit the focus of this paper to $\mathscr{I}$ events, introduced next. Let $i, j \in I, i \neq j$.

$\mathscr{I}$ **events**: An event $\mathscr{I}_{ji}$ transfers the private information of $\mathbf{S}_j$ or constant messages to $\mathbf{S}_i$. For $j \in I$, let $\Sigma_{\mathscr{I},j} \subseteq (\Sigma_{o,j} \cup \bigcup_{k \in I, k \neq j} \{\mathscr{I}_{kj} \mid \mathscr{I}_{kj} \text{ is defined}\})$ denote the set of observable events by $\mathbf{S}_j$, after which $\mathbf{S}_j$ issues an $\mathscr{I}$ event. Correspondingly, for $i, j \in I, i \neq j$, define $\mathscr{I}_{ji} : \Sigma_{\mathscr{I},j} \rightarrow pwr(X_{jj})$ as a function which associates to an event in $\Sigma_{\mathscr{I},j}$, a piece of information stored by Boolean variables in X_{jj} according to a rule which will become specified upon the design of the communication. Whereas the definition of $\Sigma_{\mathscr{I},j}$ allows the firing of an $\mathscr{I}$ event after another $\mathscr{I}$ event, circular definitions should be avoided. Since I is finite, there is a finite number of distinct $\mathscr{I}$ events and this, together with the finiteness of X, guarantee that the information is exchanged in a finite number of communication steps. Once received by $\mathbf{S}_i$, $\mathscr{I}_{ji}(.)$ provides it with the updated copies of the variables in its image set, i.e.

$$\forall \mathbf{x} \in \mathbb{F}_p^n, \ \forall i, j \in I, i \neq j, \ \forall k \in J, \ \forall x_{jj}^k \in X_{jj},$$
$$\forall \sigma \in \Sigma_{\mathscr{I},j}. \quad x_{jj}^k \in \mathscr{I}_{ji}(\sigma) \Longrightarrow x_{ij}^k := x_{jj}^k. \quad (14)$$

In a strongly connected network, $\mathscr{I}$ events can singly implement the exchange of information. However, if some direct communication links are missing or due to channel constraints "indirect" data transfers are preferred, $\mathscr{R}$ events should be designed. This issue is left for future work. Notice that even in the second case, $\mathscr{I}$ events still specify what information needs to exchange between supervisors.

Inherently, the definition of $\mathscr{I}_{ji}(\sigma)$ determine *who* (i.e. $\mathbf{S}_j$) sends *what* (a subset of X_{jj}) to *whom* (i.e. $\mathbf{S}_i$). Implicitly, the dependency on an event as its argument, bears a notion of "logical" time which roughly specifies the soonest moment at which the communication can start. Also notice that whereas the dependency on observed events makes the communication *event-triggered*, its content, which is a (Boolean)-variable representation of the history of the system's behavior, is *state-based*. We can now formalize the "communication problem" within DSDES framework.

Definition 5 Let PDS (13) and its implementation (**??**) be given. An *information policy* for (13) and (**??**) is equivalent to designing $\big(\Sigma_{\mathscr{I},j}, \mathscr{I}_{ji}(.)\big)$ for every $i, j \in I$. $\square$

Problem 2 *State-Transferring Communication Problem in DSDES framework:* Associated with Problem 1 and Proposition 1, let PDS (13) represent the DSDES and the network of EFSMs (**??**) implement the PDS. Find an information policy such that $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. $\square$

Communication is the third mean, on top of observation and control, with which decentralized supervisors confine a plant‘s behavior within given specifications. By presenting the system information of the centralized supervisor in a distributed way and putting it in PDS form, DSDES framework provides a general, flexible, and systematic approach for analysis and synthesis of decentralized supervisors. Supervisors' private variables form the largest set of system information, owned by a given state representation of the centralized supervisor. Communication helps each supervisor reevaluate its guard and updating functions by providing the values of the external variables on which they depend. Computation of communication, as a solution to Problem 2, deserves a separate work of its own, which is based on the study of algebraic structures. Here, we illustrate the applicability of the proposed approach by designing an information policy and verifying its correctness.

B. A simple information policy

In a PDS representation (13), a supervisor $\mathbf{S}_i$ depends on $\mathbf{S}_j$'s private information if and only if x_j appears as the argument of one of $\mathbf{S}_i$'s guard or updating functions. Intuitively, if $\mathbf{S}_i$ receives the last updated value of such x_j's, it can reevaluate its guard and updating functions correctly. Accordingly, during the system‘s evolution and upon the occurrence of each event, first all $\mathbf{S}_j$s, which observe it, update their x_j's. Next, every such $\mathbf{S}_j$, whose x_j is one of the arguments of an $\mathbf{S}_i$'s polynomials ($i \neq j$), sends the value of x_j to such $\mathbf{S}_i$. Back to the EFSM implementation, the value of every variable $x \in X_{jj}$ should be sent upon getting updated. However, it suffices that $\mathbf{S}_j$ send $\mathbf{S}_i$ the updated values for the subset of these variables which have *changed* upon their update. Supervisor $\mathbf{S}_i$ then utilizes the received bits to reevaluate either its Boolean variables $y \in X_{ii}$, which implement x_i, or its guards. This policy can be formally defined as follows.

Definition 6 For PDS (13) and its EFSM implementation (**??**), *information policy 1* is defined as follows: For every $i, j \in I, i \neq j$, and each $\sigma {\in} \Sigma_{o,j}$ we have $\Sigma_{\mathscr{I},j} {=} \Sigma_{o,j}$, and

$$\mathscr{I}_{ji}(\sigma){=}\{\hat{x}^k_{jj}{\in}X_{jj} \mid \hat{x}^k_{jj}{:=}a_j(x^k_{jj},\sigma)(X_j) \wedge \ \hat{x}^k_{jj}{\neq}x^k_{jj}\wedge$$
$$[\exists\sigma'{\in}\ \Sigma_i, \exists x' \in X_{ii}.\ x^k_{jj}{\in}\ \arg(a_i(x',\sigma'))\vee$$
$$x^k_{jj}{\in}\ \arg(g_i(\sigma'))]\}. \square$$

Proposition 5 *A solution to Problem 2:* Associated with Problem 2, if the network is strongly connected with lossless channels and if communication is instantaneous, information policy 1 insures that $L(\mathcal{D}) = \overline{E}$ and $L_m(\mathcal{D}) = E$. ■

Remark 4 Definition 6 implies that the communication content may be reduced by eliminating from PDS polynomials as many external variables as possible (see Remark 2). $\square$

TABLE IV

COMMUNICATION AMONGST SUPERVISORS BASED ON POLICY 2

$\mathscr{I}_{12}(\alpha_1){=}\begin{cases}\{x^1_{11},x^2_{11}\} & ;\text{if } (x^1_{11}\oplus[\overline{x}^2_{12}\overline{x}^1_{12}])\wedge(x^2_{11}\oplus x^2_{12})\\ \{x^1_{11}\} & ;\text{if } (x^1_{11}\oplus[\overline{x}^2_{12}\overline{x}^1_{12}])\wedge \neg(x^2_{11}\oplus x^2_{12})\\ \{x^2_{11}\} & ;\text{if } \neg(x^1_{11}\oplus[\overline{x}^2_{12}\overline{x}^1_{12}])\wedge (x^2_{11}\oplus x^2_{12})\\ \emptyset & ;\text{if } \neg(x^1_{11}\oplus[\overline{x}^2_{12}\overline{x}^1_{12}])\wedge \neg(x^2_{11}\oplus x^2_{12})\end{cases}$
$\mathscr{I}_{12}(\beta){=}\begin{cases}\{x^1_{11},x^2_{11}\} & ;\text{if } (x^1_{11}\oplus[\overline{x}^2_{11}\overline{x}^1_{11}])\wedge(x^2_{11}\oplus x^1_{11})\\ \{x^1_{11}\} & ;\text{if } (x^1_{11}\oplus[\overline{x}^2_{11}\overline{x}^1_{11}])\wedge \neg(x^2_{11}\oplus x^1_{11})\\ \{x^2_{11}\} & ;\text{if } \neg(x^1_{11}\oplus[\overline{x}^2_{11}\overline{x}^1_{11}])\wedge (x^2_{11}\oplus x^1_{11})\\ \emptyset & ;\text{if } \neg(x^1_{11}\oplus[\overline{x}^2_{11}\overline{x}^1_{11}])\wedge \neg(x^2_{11}\oplus x^1_{11})\end{cases}$
$\mathscr{I}_{21}(\alpha_2){=}\begin{cases}\{x^1_{22},x^2_{22}\} & ;\text{if } (x^1_{22}\oplus[x^2_{22}x^1_{21}+\overline{x}^1_{22}x^2_{21}])\wedge (x^2_{22}\oplus[\overline{x}^2_{22}(\overline{x}^2_{21}\overline{x}^1_{22}+x^1_{21})])\\ \{x^1_{22}\} & ;\text{if } (x^1_{22}\oplus[x^2_{22}x^1_{21}+\overline{x}^1_{22}x^2_{21}])\wedge \neg(x^2_{22}\oplus[\overline{x}^2_{22}(\overline{x}^2_{21}\overline{x}^1_{22}+x^1_{21})])\\ \{x^2_{22}\} & ;\text{if } \neg(x^1_{22}\oplus[x^2_{22}x^1_{21}+\overline{x}^1_{22}x^2_{21}])\wedge (x^2_{22}\oplus[\overline{x}^2_{22}(\overline{x}^2_{21}\overline{x}^1_{22}+x^1_{21})])\\ \emptyset & ;\text{if } \neg(x^1_{22}\oplus[x^2_{22}x^1_{21}+\overline{x}^1_{22}x^2_{21}])\wedge \neg(x^2_{22}\oplus[\overline{x}^2_{22}(\overline{x}^2_{21}\overline{x}^1_{22}+x^1_{21})])\end{cases}$
$\mathscr{I}_{21}(\beta){=}\begin{cases}\{x^1_{22},x^2_{22}\} & ;\text{if } (x^1_{22}\oplus[\overline{x}^2_{22}\overline{x}^1_{22}])\wedge(x^2_{22}\oplus x^1_{22})\\ \{x^1_{22}\} & ;\text{if } (x^1_{22}\oplus[\overline{x}^2_{22}\overline{x}^1_{22}])\wedge \neg(x^2_{22}\oplus x^1_{22})\\ \{x^2_{22}\} & ;\text{if } \neg(x^1_{22}\oplus[\overline{x}^2_{22}\overline{x}^1_{22}])\wedge (x^2_{22}\oplus x^1_{22})\\ \emptyset & ;\text{if } \neg(x^1_{22}\oplus[\overline{x}^2_{22}\overline{x}^1_{22}])\wedge \neg(x^2_{22}\oplus x^1_{22})\end{cases}$
$\mathscr{I}_{32}(\alpha_3) = \{x^1_{33}\}$ ($x^1_{33}\oplus\overline{x}^1_{33}$ is always true)
$\mathscr{I}_{13}(\alpha_1){=}\ \mathscr{I}_{13}(\beta){=}\ \mathscr{I}_{23}(\alpha_1) = \mathscr{I}_{23}(\beta) = \mathscr{I}_{31}(\alpha_3) = \emptyset$

Example 3 For the system in Example 1 with guards and actions in Table III, $\mathscr{I}$ events are computed using information policy 1 and listed in Table IV. $\lozenge$

REFERENCES

[1] K. Rudie and W. M. Wonham, "Think globally, act locally: Decentralized supervisory control," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1692–1708, Nov. 1992.

[2] D. Teneketzis, "On information structures and nonsequential stochastic control," *CWI Quarterly*, vol. 10, no. 2, pp. 179–199, 1997.

[3] A. Mannani and P. Gohari, "A framework for modeling communication among decentralized supervisors for discrete-event systems," in *Proc. of IEEE Conference on Sys., Man, and Cyb. (SMC'07)*, Montreal, Canada, Oct. 2007, pp. 1339–1344.

[4] ——, "Decentralized supervisory control of discrete-event systems over communication networks," *IEEE Trans. on Automat. Contr.*, vol. 53, no. 2, pp. 547–559, Mar. 2008.

[5] R. Germundsson, "Symbolic systems: Theory, computation, and applications," Ph.D. dissertation, Univ. of Linköping, S-581 83 Linköping, Sweden, Sept. 1995. [Online]. Available: http://www.control.ee.liu.se

[6] R. Lidl and H. Niederreiter, *Finite Fields*, 1st ed. Cambridge University Press, 1997.

[7] R. Germundsson, "Basic results on ideals and varieties in finite fields," 1991. [Online]. Available: citeseer.ist.psu.edu/germundsson91basic.html

[8] M. M. Mano, *Digital design*, 3rd ed. Prentice-Hall, 2002.

[9] M. L. Borgne, A. Benveniste, and P. L. Guernic, "Polynomial dynamical systems over finite fields," in *Algebraic Computing in Control*. Heidelberg: Springer Berlin, 1991, vol. 165/1991, pp. 212–222.