On the Robust Synthesis of Logical Consensus Algorithms for Distributed Intrusion Detection [⋆]

Adriano Fagiolini [a,b], Antonio Bicchi [b,c]

[a] *DIEETCAM, Faculty of Engineering, Università degli Studi di Palermo, Italy*

[b] *Interdepartmental Research Center "E. Piaggio", Faculty of Engineering, Università di Pisa, Italy*

[c] *Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy*

Abstract

We introduce a novel consensus mechanism by which the agents of a network can reach an agreement on the value of a shared logical vector function depending on binary input events. Based on results on the convergence of finite–state iteration systems, we provide a technique to design logical consensus systems that minimize the number of messages to be exchanged and the number of steps before consensus is reached, and that can tolerate a bounded number of failed or malicious agents. We provide sufficient joint conditions on the input visibility and the communication topology for the method's applicability. We describe the application of our method to two distributed network intrusion detection problems.

Key words: Consensus, distributed algorithms, intrusion detection, security.

1 Introduction

Many control problems with distributed or networked systems require that agents reach an agreement on certain information, by merging their own uncertain and possibly incomplete estimates through neighbor–to–neighbor interaction strategies. In the simplest case, the information to agree about can be represented by real numbers or vectors, and global agreement can be reached by the use of linear iterative strategies [16]. These strategies require that every agent repeatedly update their own real states as weighted combinations of their own local values and those of their neighbors. Examples of problems falling into this linear framework are the rendezvous problem for a set of mobile robots [21] and the clock synchronization problem for a group of distributed processes [23]. A vast literature has recently established under which connectivity properties of the system's communication topology, global agreement can be reached on e.g. the average of the agents' initial values (see e.g. [1, 4, 11, 16, 20]). Other problems, such as general functions consensus [7] and optimal sensing coverage with a team of mobile robots [6], still involve agreement on real information, but need nonlinear, ad–hoc iterative strategies. In [6] a team of robots can adjust their locations moving toward the centroid of the corresponding Voronoi regions. In other application domains, agents have to consent on information represented by (possibly) infinite set–valued data and need to use aggregation strategies operating on such data. A nonlinear iterative strategy is described in [8] which enables a team of robots to simultaneously self–localize and build a map of the surrounding environment, by merging continuous sets that represent locally estimated uncertain positions of detected features. Misbehavior detection in multi–robot systems with event–based coexistence rules is solved via a consensus strategy by which robots can reconstruct the occupancy map of other robots' neighborhoods [3].

Moreover, when dealing with distributed and open systems, one has necessarily to deal with the event that some agents may exhibit unexpected behavior, due to spontaneous failure or even malicious programming. Robust clock synchronization in systems where processes can exchange locally estimated confidence intervals of the clock has been shown to be solvable via a nonlinear iterative strategy that is resilient to possible measurement inconsistency [15]. More in general, failure management has been studied in distributed computing, under

[⋆] Corresponding author A. Fagiolini, Tel. +39 091 23863613, Fax. +39 050 2217051.

Email addresses: `fagiolini@unipa.it` (Adriano Fagiolini), `bicchi@centropiaggio.unipi.it` (Antonio Bicchi).

the framework of so–called Byzantine Generals problem (see e.g. [12, 14]), which addresses how to ensure that all agents of a network reach a consensus also in the event that a bounded number of them experiences failures. In this framework, communication between neighboring agents is allowed via the exchange of "oral" messages, i.e. there exist wired/dedicated connections between any two communicating agents, which enables malicious agents to send different, possibly conflicting values to different neighbors. It has been established that, to tolerate up to γ malicious agents (possibly conspiring together), there are two necessary conditions on the minimum number n of agents, i.e. $n \geq 3\gamma$, and on connectivity c of the communication topology ($c \geq 2\gamma + 1$). More recently, the problem of uncooperative agents has received an increasing attention in the control community [3, 17, 24], most likely due to the strong push given by the foreseen actual implementability of multi–robot systems in a near future. Here, a broadcast communication model can be adopted, since communication is wireless, which implies that agents are only able to transmit identical values to all their neighbors. In this context, the requirement on the number of agents is no longer needed, but the connectivity one is still in place [2].

In the present work, we focus on control problems where a team of agents must cooperatively compute a logical vector function that returns a set of decisions depending on a set of input events. Agents have partial accessibility to the input events, very simple local computation and broadcast communication capabilities, and may be affected by failures. This problem involves reliable network information diffusion that can be achieved by the use of techniques based on Robust Flooding (RF) or Exponential Information Gathering (EIG) [14]. RF is a protocol requiring that every agent forward all incoming messages to its outgoing links, except the ones from which it has received the messages. In its basic formulation, RF introduces a high communication overhead, due to redundant relay of multiple copies of each message [18]. Protocol's variations such as the initialization of every message with a *life–time counter*, which is decreased every time that the message is relayed, and the subsequence discard of the messages that are out of their life span, can reduce the amount of messages circulating in the network only heuristically. In a network with moderate connectivity, if every agent has k neighbors, and the counter strategy with a limit of h hops is used, then every data packet will spawn on the order of k^h copies [18]. Moreover, since communication links have finite capacity, messages are dropped when their buffers become full. To prevent that all messages for the conversation between a given couple of agents are systematically dropped, strategies enforcing fair link utilization by the agents must be implemented, which requires computationally expensive authentication mechanisms (typically based on public key cryptography), assuring that a message generated by an agent occupies its reserved memory buffer slot [18]. Furthermore, the alternative of EIG algorithms requires that every agent send their initial estimates and relay the ones that they receive for $\gamma + 1$ rounds, while recording all such values in a labelled tree that memorizes them by the communication path they have been received. While EIG algorithms are able to solve the agreement problem under the worst failure conditions (occurring when agents may exhibit any arbitrary behaviors and can communicate via "oral" messages), they are known to be costly for the amount of local storage used, and for the dimension of the messages that are exchanged [14]. The labelled tree has indeed $\gamma + 2$ levels, and each node at level k, $0 \leq k \leq \gamma$, has exactly $n - k$ children. EIG algorithms become also unnecessarily redundant with broadcast communication.

To overcome the limitations of existing methods from the distributed algorithm literature, we propose a technique which adopts a dynamic system approach. Our method allows the design of *logical interaction strategies* by using which agents can reach an agreement on the value of the shared logical vector function, via the exchange of binary values of their local estimates of the input events. The technique builds upon known results on the convergence of finite–state iteration systems [22] and previous work by the authors [10], and it is valid for networks with fixed topologies. Our method requires centralized knowledge of the communication topology during an initial design phase, whose aim is to find *secure* minimum–length paths connecting every input event with every agent, so that the input's information can robustly flow over the network with the minimum number of steps and messages. After such a phase, every agent needs only to know what are their neighbors and how to combine their states. This is to be compared with the linear iterative strategies surveyed in in [5, 16], where a centralized design phase is not needed, and it is only required to check that all possible communication graphs are doubly stochastic. Moreover, by formalizing our interaction strategy as a logical dynamic system, the dimension of every agent's state is also minimized, since only the result of the computation performed during the latest one–hop interaction needs to be maintained.

The paper is organized as follows. The logical consensus problem is formalized in Section 2. Results on the convergence of binary dynamic systems are recalled in Section 3. The problem of reaching consensus on a single input with and without faulty agents is settled in Section 4 and 5. The design of logical consensus systems for robust computation of generic logical vector functions with fixed topology networks is described in Section 6. Application to two distributed intrusion detection problems is shown in Section 7.

2 Problem Statement

We consider distributed control problems where p *decisions* depending on the occurrence of m *events* must be *shared* by a group of n agents, $\mathcal{A}_1, \ldots, \mathcal{A}_n$. Events are conditions, such as the discovery of an intruder in a specific region or the failure of a component in a computation system, that can occur at different geographical locations. Decisions are *group–level* actions that have to be performed in response to such events. A set of m binary input variables $u_1, \ldots, u_m$ are associated with the events, and a set of p binary output variables, $y_1, \ldots, y_p$, are associated with the decisions. An input–output relation of the type

$$\begin{cases} y_1 = f_1(u_1, \ldots, u_m)\,, \\ \quad\vdots \\ y_p = f_p(u_1, \ldots, u_m)\,, \end{cases} \tag{1}$$

where each $f_i : \mathbb{B}^m \to \mathbb{B}$ is a logical function, expresses the connection between the two sets of variables. We will refer to Eq. 1 as the logical decision system and rewrite it more concisely as $y = f(u)$, where $u = (u_1, \ldots, u_m)^T \in \mathbb{B}^m$ and $y = (y_1, \ldots, y_p)^T \in \mathbb{B}^p$ are the input and output vectors, respectively, and $f = (f_1, \ldots, f_p)^T$, with $f : \mathbb{B}^m \to \mathbb{B}^p$, is a logical vector function.

Agents are placed at different geographical locations and may have heterogeneous sensors, which implies that each input component u_j may be measurable only by a subset of them. This property can modeled by introducing a *visibility matrix* $V \in \mathbb{B}^{n\times m}$, being a binary matrix s.t. $V(i, j) = 1$ if, and only if, the i–th agent $\mathcal{A}_i$ can read the input component u_j. Moreover, agents are able to exchange messages with each others, but may not be able to communicate with all the members of the group, which can be described by a *communication matrix* $C \in \mathbb{B}^{n\times n}$, where $C(i, k) = 1$ if, and only if, the agent $\mathcal{A}_i$ can receive a message from the agent $\mathcal{A}_k$. The matrix C is instrumental to define the notion of communication neighbors, or in short C–neighbors, of an agent $\mathcal{A}_i$, being identified by the non–null elements of C's i–th row.

The evaluation of f requires in general full knowledge of the input vector u. A possible solution can be obtained where a single *centralized* decision process P_c receives all measures of the input vector components u_j, computes the output vector y, and sends it back to all agents. This naive approach is unsatisfactory at least for three reasons: First of all, it is *non–scalable* since the amount of data to be exchanged through the network and processed by P_c increases with the number of agents, rather than only with the dimensions m and p of the decision task; secondly, the approach requires an explicit *message routing* management to ensure that every agent reaches and is reached by P_c; third, it represents a system with a *single–point of failure* represented by P_c and may be unable to cope with communication failures, unless countermeasures based on e.g. message relay are introduced.

We pursue a different approach where agents are aware of the logical functions $f_1, \cdots, f_p$, and must cooperatively estimate the output of the logical decision system $y = f(u)$. Each agent $\mathcal{A}_i$ has a *binary vector state* $X_i = (X_{i,1}, \cdots, X_{i,q}) \in \mathbb{B}^{1\times q}$, where q is a proper dimension, and an output decision vector $Y_i = (y_{i,1}, \cdots, y_{i,p}) \in \mathbb{B}^{1\times p}$. Let $X = (X_1^T, \ldots, X_n^T)^T \in \mathbb{B}^{n\times q}$ be the state of the agents' network. The agent's state is updated according to an iterative rule of the form $X_i(t + 1) = F_i(X(t), u(t))$, where t is a discrete time, and the agent's output is computed via a binary output function $Y_i(t) = G_i(X_i(t), u(t))$. The maps $F_i : \mathbb{B}^q \times \mathbb{B}^m \to \mathbb{B}^q$ and $G_i : \mathbb{B}^q \times \mathbb{B}^m \to \mathbb{B}^p$ are required to comply with the agent's local visibility and communication abilities, i.e. they can only depend on $\mathcal{A}_i$'s state, the state of its C–neighbors, and on the input components u_j that it can read. Let $Y = (Y_1^T, \ldots, Y_p^T)^T \in \mathbb{B}^{p\times q}$ be the agents' network output. The evolution of all agents' network is thus described by the logical iterative system

$$\begin{cases} X(t + 1) = F(X(t), u(t))\,, \\ \quad Y(t) = G(X(t), u(t))\,, \end{cases} \tag{2}$$

where $F = (F_1^T, \ldots, F_n^T)^T$ and $G = (G_1^T, \ldots, G_n^T)^T$. Hence, we recast the problem of computing the decision system in Eq. 1 as that of allowing a network of agents to *consent* on the output of a logical vector function, which we will refer to as the problem of reaching *Logical Consensus (LC)*.

In this context, we want to solve the following problem, which is dealt with in Section 4:

Problem 1 (LC in Virtuous Scenarios) *Given the logical decision system of Eq. 1, communication and visibility matrices C and V, design a logical consensus system as in Eq. 2, that is compliant with C and V and ensures logical consensus, from all initial state $X(0)$ and inputs u, on the centralized decision system $y^* = f(u)$, i.e.*

$$Y(t) = \mathbf{1}_n\,(y^*)^T\,,\ \textit{for some}\ t > t'\,,$$

where $\mathbf{1}_n$ is an $n \times 1$ binary vector with all entries to 1.

We also want to solve the same consensus problem within a scenario with possible faults or security attacks, which is dealt with in Section 5:

Problem 2 (LC in Malicious Scenarios) *Assuming that at most γ agents may share incorrect/corrupted data, design a* robust *logical consensus system ensuring all correct agents $\mathcal{A}_i$ to consent on the correct decision, i.e.*

$$Y_i(t) = (y^*)^T\,,\ \textit{for some}\ t > t'\,.$$

Finally, we assume that the input vector u is piece–wise constant, indicating that it may be constant or slowly–changing with respect to the convergence speed of the system in Eq. 2.

3 Convergence of Logical Dynamic Systems

Consider the simplest Boolean algebra described by the sextuple $(\mathbb{B}, +, \cdot, \neg, 0, 1)$, where $\mathbb{B} = \{0,1\}$ is a domain set, $+$ and $\cdot$ are binary operations representing the logical sum and product, respectively, $\neg$ is a unary operation representing the logical complement, 0 (null) and 1 (unity) are the domain's smallest and biggest values, respectively. Consider the partial order relation $\leq$ described by the axioms: $0 \leq 0$, $0 \leq 1$, $1 \leq 1$. An element $\lambda \in \mathbb{B}$ is referred to as a *scalar*. Given an integer number n, a Boolean *vector* v and a Boolean *matrix* A are elements belonging to the sets $\mathbb{B}^n$ and $\mathbb{B}^{n\times n}$, respectively. Given two vectors $v = (v_1, \dots, v_n)$ and $w = (w_1, \dots, w_n)$, and two square matrices $A = \{a_{i,j}\}$ and $B = \{b_{i,j}\}$, we define the scalar product as

$$w^T v \overset{\text{def}}{=} \textstyle\sum_{i=1}^n v_i \cdot w_i \in \mathbb{B}\,,$$

the product Av as the vector whose i–th element is the scalar product between the i–th row of A and the vector v, and the product AB as the matrix whose (i,j)–th element is the scalar product between the i–th row of A and the j–th column of B. In other words products between a matrix and a vector and between two matrices are computed in the usual way, by only replacing the sum and product on reals with those of the Boolean algebra.

Consider an autonomous logical system of the form

$$\begin{cases} x(t+1) = F(x(t))\,, \\ x(0) = x^0\,, \end{cases} \tag{3}$$

where x^0 is an initial state and $F : \mathbb{B}^n \to \mathbb{B}^n$ is an endomorphism on $\mathbb{B}^n$ involving only operations of the binary algebra. It is worth noting that, as $\mathbb{B}^n$ is a finite domain set, the evolution of a generic logical system can either converge in finite time to an equilibrium point or be captured by a cycle. The convergence of these systems is studied in [22], from which we recall the following Def. 1–4 and the results described in the remainder of this section.

First we need to introduce a metric on $\mathbb{B}^n$:

$$\begin{aligned} \Delta : \mathbb{B}^n \times \mathbb{B}^n &\to \mathbb{B}^n \\ (x,y) &\mapsto (x_1 \oplus y_1, \cdots, x_n \oplus y_n) \end{aligned}\,,$$

where $\oplus$ is the exclusive disjunction

$$\begin{aligned} \oplus : \mathbb{B} \times \mathbb{B} &\to \mathbb{B} \\ (x_i, y_i) &\mapsto (\neg x_i\, y_i) + (x_i\, \neg y_i) \end{aligned}\,,$$

This metric, called binary vector distance, is indeed a distance on $\mathbb{B}^n$, since it satisfies the following axioms: $\Delta(x,y) = \Delta(y,x)$, $\Delta(x,y) = 0$ iff $x = y$, $\Delta(x,y) \leq \Delta(x,z) + \Delta(z,y)$, for all $x, y, z \in \mathbb{B}^n$.

Definition 1 (Boolean Eigenvalues/Eigenvectors) *A scalar $\lambda \in \mathbb{B}$ is an* eigenvalue *of a Boolean matrix $A \in \mathbb{B}^{n\times n}$ if there exists a vector $x \in \mathbb{B}^n$, called* eigenvector*, s.t.*

$$A\,x = \lambda\,x\,.$$

Definition 2 (Boolean Spectral Radius) *The spectral radius of a Boolean matrix $A \in \mathbb{B}^{n\times n}$, denoted with $\rho(A)$, is its biggest eigenvalue in the sense of the order relation $\leq$.*

Proposition 1 *Every Boolean matrix $A \in \mathbb{B}^{n\times n}$ has at least one eigenvalue. Hence $\rho(A)$ always exists.*

Proposition 2 *A Boolean matrix $A \in \mathbb{B}^{n\times n}$ has Boolean spectral radius $\rho(A) = 0$ if, and only if, one of the two following equivalent conditions hold:*

- *$P^T A\,P$ is a strictly lower or upper triangular matrix for some permutation matrix P;*
- *$A^n = 0$ (the n–th Boolean matrix power of A).*

Remark 1 (Spectral Radius Computation) *The above propositions provide a procedure for the computation of $\rho(A)$. Indeed, by Prop. 1, we have $\rho(A) \geq 0$. Hence, one has first to check if A admits only the eigenvalue $\lambda = 0$, which can be done, based on Prop. 2, either by showing a permutation matrix P (namely, a reordering of A's rows and columns) that brings A into strictly lower or upper triangular form, or by checking if A^n equals the null matrix. If unsuccessful, one can conclude that $\rho(A) = 1$ implying that the scalar $\lambda = 1$ is an eigenvalue of A, while nothing can be said for the scalar $\lambda = 0$.*

Definition 3 (Contractive Map) *A map $F : \mathbb{B}^n \to \mathbb{B}^n$ is said to be contractive w.r.t. the binary vector distance Δ if there exists a matrix $M \in \mathbb{B}^{n\times n}$ s.t.*

- *$\rho(M) < 1$ (which implies $\rho(M) = 0$), and*
- *$\Delta(F(x), F(y)) \leq M\,\Delta(x,y)$, for all vectors $x, y \in \mathbb{B}^n$.*

Definition 4 (Incidence Matrix) *The incidence matrix of a logical map F is a Boolean matrix $B(F(x)) = \{b_{i,j}\}$, where $b_{i,j} = 1$ if, and only if, the i–th component of $F(x)$ depends on the j–th component of the input vector x, i.e.*

$$\exists\, \bar{x} \in \mathbb{B}^n \text{ s.t. } F(\bar{x}) \neq F(\bar{x} \oplus e_j)\,,$$

where e_j is the j–th basis vector of $\mathbb{B}^n$.

Theorem 1 *A map $F : \mathbb{B}^n \to \mathbb{B}^n$ is contractive if, and only if, the following equivalent conditions hold:*

- *$\rho(B(F(x))) = 0$;*

- *$P^T B(F(x))\, P$ is strictly lower or upper triangular, for some permutation matrix P;*
- *$B(F(x))^q = 0$, with $0 \le q \le n$;*
- *$\exists\, q \le n$ s.t. F^q (F's composition with itself q times) is a constant map, i.e. it is independent of $x(0)$.* ⧫

Corollary 1 *A contractive map F globally converges to a unique equilibrium.*

Finally, we can provide the following definition:

Definition 5 *A logical map $F : \mathbb{B}^n \times \mathbb{B}^m \to \mathbb{B}^n$ is* (C, V)–compliant *if, and only if, its incidence matrix $B(F(X, u))$ satisfies the logical vector inequality*

$$B(F(X, u)) \le (C\ V)\ .$$

4 Linear Logical Consensus Systems

First of all, we need to determine if a given combination of input visibility and agent communication topology allows the j–th input u_j to be propagated to the entire network, or in other words which part of the graph is reachable from u_j. The binary vector V_j contains 1 for all entries representing agents that can "see" or measure the input u_j, and thus that are reached from the input in one step. Note also that all binary vectors $C^k V_j$, for $k = 0, 1, \ldots,$ contain 1 for all entries representing agents that can receive the values of input u_j through a sequence of k messages, and thus that are reached from the input after exactly $k+1$ steps. More precisely, if we add a fictitious node representing u_j to the communication graph, the i–th element of $C^k V_j$ is 1 if, and only if, there exists at least one path of length $k+1$ from the fictitious node to the one representing the agent $\mathcal{A}_i$. By definition of graph diameter $\mathrm{diam}(G)$, all agents that are reachable from an initial set of agents are indeed reached in at most $\mathrm{diam}(G)$ steps. It also holds that $\mathrm{diam}(G) \le n-1$. Let us denote with $\kappa(C, V_j)$ the *visibility diameter* of the pair (C, V_j) being the number of steps after which the sequence $\{C^k V_j\}$ does not reach new nodes/agents. Given a pair (C, V_j), we can introduce the *reachability matrix* R_j, assigned with input u_j,

$$R_j = \left(V_j\ CV_j\ C^2 V_j\ \cdots\ C^{n-1} V_j\right)\ ,$$

whose columns "span" a subgraph $G_{\mathcal{R}}(N_{\mathcal{R}}, E_{\mathcal{R}})$ of $G(N, E)$, where $N_{\mathcal{R}}$ is the node set representing agents that are reachable from u_j, $E_{\mathcal{R}}$ is an unspecified edge set that will be considered during the design phase, $N = \{1, \ldots, n\}$, and $E = \{(i, j) \,|\, C(i, j) = 1\}$ is the edge set of available links. Consider the j–th *span vector* obtained as the logical sum of R_j's columns:

$$I_j^{(n)} = \textstyle\sum_{k=0}^{n-1} C^k V_j = \sum_{k=0}^{n-1} R_j(:, k)\,,$$

whose i–th element equals unity if, and only if, there exists a path of any length from the fictitious node representing u_j to the node representing $\mathcal{A}_i$. Given a pair (C, V_j), we define the *span* of the reachability matrix $R_j(C, V_j)$ as the reachable set $N_{\mathcal{R}} = \{i \,|\, I_j^{(n)}(i) = 1\}$. The unreachable set is obtained by complementation: $N_{\bar{\mathcal{R}}} = N \setminus N_{\mathcal{R}}$.

Definition 6 *A pair (C, V_j) is* (completely) reachable *if, and only if, the span of $R_j(C, V_j)$ is the entire graph (i.e. $N_{\mathcal{R}} = N$), or equivalently if*

$$I_j^{(n)} = \mathbf{1}_n\ .$$

Example 4.1 *Consider e.g. a network with $n = 5$ agents characterized by the following communication matrices and j–th visibility vector:*

$$C = \begin{pmatrix} 1\,1\,0\,0\,1 \\ 1\,0\,1\,0\,1 \\ 1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,1 \\ 0\,0\,0\,0\,1 \end{pmatrix}\ ,\quad V_j = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}\ .$$

First note that, in this example, only agent $\mathcal{A}_1$ is able to measure u_j. The reachability matrix associated with the j–th input is

$$R_j = (V_j\ CV_j\ \cdots\ C^4 V_j) = \begin{pmatrix} 1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,1 \\ 0\,1\,1\,1\,1 \\ 0\,0\,1\,1\,1 \\ 0\,0\,0\,0\,0 \end{pmatrix}\ .$$

The span vector is $I_j^{(n)} = (1, 1, 1, 1, 0)^T$, and thus the reachable subgraph is $N_{\mathcal{R}} = \{1, 2, 3, 4\}$, whereas the unreachable one is $N_{\bar{\mathcal{R}}} = \{5\}$. The visibility diameter, being the number of steps within which all the agents in $N_{\mathcal{R}}$ are reached, is $\kappa(C, V_j) = 3$. ⧫

Consider now how to design a consensus map $F : \mathbb{B}^n \times \mathbb{B} \to \mathbb{B}^n$ that is (C, V_j)–compliant and that allows the information on u_j to be propagated throughout the network. It should be evident that the design can only concern the reachable subgraph $G_{\mathcal{R}}(N_{\mathcal{R}}, E_{\mathcal{R}})$, while nothing can be done for the unreachable one. Note that the presence of a non–empty unreachable subgraph $G_{\bar{\mathcal{R}}}$ in our context where node consensus is sought indicates that the design problem is not well–posed. This would require further actions, such as the introduction of new nodes, a better deployment of the existing ones as well as an enhancement of their visibility and communication capabilities, but this goes beyond the scope of the work and will not be considered.

An intuitive yet optimal way to propagate the input u_j is obtained every agent that can directly measure the input send their local estimates to their C–neighbors without overlapping, which in turn will send them to other C–neighbors that have not been received yet, and so on and so forth. If we select from the graph G only the edges representing links through which one message has been sent according to this strategy, we obtain a so–called *Input Propagation Spanning Tree* (IPST), which is rooted at the fictitious node representing u_j and which reaches every agent in $N_{\mathcal{R}}$. It also encodes an optimal message exchange scheme by following which all reachable agents can be informed on the value of u_j with minimum number of messages and steps. To find such an IPST is instrumental to solve our problem and it can be based on the vector sequence $\{C^k V_j\}$ [1]. Indeed, note that $C^k V_j = C(C^{k-1}V_j)$, which tells us that in the optimal propagation scheme the agents that are reachable after k steps must have received a message containing the estimate of u_j from one of the agents that were reachable after $k-1$ steps. Any consecutive sequence of agents that are extracted from non–null elements of the sequence is (C, V_j)–compliant by construction.

Therefore, an IPST of input u_j can be found by considering the sequence of vectors indicating, for all steps k, which agents are reached for the first time

$$L_j^k = \begin{cases} V_j & \text{if } k = 1\,, \\ C^{k-1}V_j \neg \left(\sum_{h=0}^{k-2} C^h V_j\right) & \text{if } k > 1\,. \end{cases}$$

Let $\bar{k}$ be the first step at which no new agents are reached, i.e. $L_j^{(\bar{k})} = 0$. It trivially holds that

$$\kappa(C, V_j) = \min_k \{k \,|\, L_j^k = 0\} = \bar{k} - 1\,.$$

At the generic step k of the design phase, it is possible to chose the update rule $F_i : \mathbb{B}^n \times \mathbb{B} \to \mathbb{B}$ of every agent $\mathcal{A}_i$ s.t. $L_j^k(i) = 1$. In particular we have

$$x_i(t+1) = C^*_{i,:}\, x(t) + V_j(i)\, u_j\,,$$

with

$$C^*_{i,:} = \begin{cases} 0 & \text{if } k = 1\,, \\ e_{h_i}^T & \text{if } k > 1\,, \end{cases}$$

where e_{h_i} being the h_i–th vector of the canonical basis, and

$$h_i = \min_h \{C_{i,h}\, L_j^{k-1}(h) = 1\}\,.$$

Note also that $C^* = S\,C \le C$, where S is a suitable selection matrix. The design phase ends at the step $\bar{k}$.

Moreover, let P be a permutation matrix that reorders the agents by order their update function F_i are decided during the design phase. Let also $\tilde{C}^* = P^T C^* P$ and $\tilde{V}_j = P^T V_j$ be the corresponding communication matrix and visibility vector, respectively. It should be evident that, in the reordered coordinates, we have

$$\tilde{C}^* = \left(\begin{array}{cccc|c} 0 & 0 & \cdots & 0 & 0 \\ \tilde{C}_{0,1} & 0 & \dots & 0 & 0 \\ \vdots & & & \vdots & \vdots \\ 0 & \cdots & \tilde{C}_{\kappa-1,\kappa} & 0 & 0 \\ \hline 0 & \cdots & 0 & 0 & 0 \end{array}\right), \tag{4}$$

which has a strictly lower–block triangular form, and $\tilde{V}_j^* = P^T V_j = (1, \cdots, 1, 0, \cdots, 0)^T$. In the new coordinates, the reachability property of the system should be more apparent. Indeed the reachability matrix $R_j(\tilde{C}^*, \tilde{V}_j)$ is

$$\left(\begin{array}{cccccc|c} \mathbf{1}^T & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & \tilde{C}_{0,1} & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & \tilde{C}_{2,3}\tilde{C}_{1,3} & \dots & 0 & 0 & 0 \\ \vdots & & & & & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \tilde{C}_{\kappa-1,\kappa} \cdots \tilde{C}_{1,\kappa} & 0 \\ \hline 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{array}\right),$$

where the upper–left matrix block (related to the reachable subgraph) contains exactly one element to 1 in all rows, and the two lower matrix blocks (related to the unreachable subgraph) are 0. Finally, observe that all products $\tilde{C}_{i+1,j}\tilde{C}_{i,j}$ are well–defined since the column number of $\tilde{C}_{i+1,j}$ equals the row number of $\tilde{C}_{i,j}$ by construction.

Example 4.2 (Cont'd) *Consider the design phase of the network of Example 4.1. For $k = 1$ we have $L_j^1 = (1, 0, 0, 0, 0)^T$ and thus the update function of agent $\mathcal{A}_1$ is chosen with $C^*_{1,h} = 0$. For $k = 2$ we have $L_j^2 = (0, 1, 1, 0, 0)^T$ and thus the update functions of agents $\mathcal{A}_2$ and $\mathcal{A}_3$ are chosen with $C^*_{2,h} = C^*_{3,h} = (1, 0, 0, 0, 0)$. For $k = 3$ we have $L_j^3 = (0, 0, 0, 1, 0)^T$ and thus the update function of agent $\mathcal{A}_4$ is chosen with $C^*_{4,h} = (0, 1, 0, 0, 0)$. For $k = 4$ we have $L_j^4 = 0$ and thus $\kappa(C, V_j) = 3$. $C^*_{5,h}$ is*

[1] As described below, the computation of an IPST involves only logical operations on binary vectors and, hence, is very efficient from both memory and computation viewpoints.

undetermined since agent $\mathcal{A}_5$ is unreachable and can be set to null. The corresponding communication matrix is

$$C^* = P^T(CS)P = \left(\begin{array}{c|cc|c|c} 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \end{array}\right),$$

where P is the identity matrix and S is a suitable selection matrix. The corresponding consensus system, restricted to the reachable agents, is

$$\begin{cases} x_1(t+1) = u_j(t)\,, \\ x_2(t+1) = x_1(t)\,, \\ x_3(t+1) = x_1(t)\,, \\ x_4(t+1) = x_2(t)\,, \end{cases}$$

which indeed can optimally propagate the input u_j. ♦

We can now discuss the correctness of the above described linear consensus system. The following theorem is a solution to Problem 1.

Theorem 2 (Linear Logical Consensus) *Given a reachable pair (C, V_j), where C is a communication matrix and V_j is a visibility vector, the linear logical consensus system*

$$\begin{cases} x(t+1) = C^*\,x(t) + V_j\,u_j(t)\,, \\ \quad x(0) = x^0\,, \end{cases}$$

where x^0 is an initial state and the pair (C^*, V_j) is an IPST of input u_j, is (C, V_j)–compliant and, for all piecewise constant inputs $u_j(t) = \bar{u}_j$, globally converges in at most $\kappa(C, V_j)$ steps to the equilibrium

$$\bar{x} = \mathbf{1}_n \bar{u}_j\,.$$

Proof 1 *The consensus state $\bar{x} = \mathbf{1}_n\bar{u}_j$ is an equilibrium and is globally stable. Indeed, the update rule gives*

$$x(t+1) = C^*\mathbf{1}_n\,\bar{u}_j + V_j\,\bar{u}_j = \tilde{V}_j\,\bar{u}_j = \mathbf{1}_n\bar{u}_j$$

since $\tilde{V}_j = C^*\,\mathbf{1}_n + V_j = \mathbf{1}_n$. Moreover, the incidence matrix of the system is

$$B(F(x)) = C^*\,,$$

which is similar to the strictly lower triangular matrix in Eq. 4. Then, the global stability of the equilibrium follows from Theorem 1.

To prove the convergence time, consider the system in coordinates sorted by the order their update function are chosen during the design phase. Since $\tilde{C}^*$ is a block lower triangular matrix, the system can be solved block–wise via Gauss' method. Indeed we have

$$\begin{cases} z_0(t+1) \,=\, u(t)\,, \\ z_1(t+1) \,=\, \tilde{C}_{0,1}\,z_0(t)\,, \\ \qquad\vdots \\ z_l(t+1) \,=\, \tilde{C}_{l-1,l}\,z_{l-1}(t)\,, \\ \qquad\vdots \\ z_\kappa(t+1) \,=\, \tilde{C}_{\kappa-1,\kappa}\,z_{\kappa-1}(t)\,, \end{cases}$$

where z_i is the i–th block of reordered components. The system's evolution with constant input $\bar{u}_j$ is thus: $z_0(1) = \bar{u}_j$, $z_1(2) = \tilde{C}_{0,1}\,z_0(1) = \bar{u}_j$, $\cdots$, $z_\kappa(\kappa) = \tilde{C}_{\kappa-1,\kappa}\,z_{\kappa-1}(\kappa-1) = \bar{u}_j$, which proves that the consensus is reached after κ steps.

Example 4.3 *Consider a network of $n = 5$ agents with the following pair of communication and visibility matrices (note that two agents are able to measure the input):*

$$C = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix},\quad V_j = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (5)$$

Consider applying the design procedure above and reordering the agents into 2 disjoint groups based on the agent that can measure the input. An IPST in the reordered coordinates is characterized by the matrices

$$\tilde{C}^* = \left(\begin{array}{c|cc|c|c} 0 & 0 & 0 & 0 & 0 \\ \cline{1-3} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \cline{4-5} 0 & 0 & 0 & 1 & 0 \end{array}\right),\quad \tilde{V}_j^* = \left(\begin{array}{c} 1 \\ \hline 0 \\ 0 \\ \hline 1 \\ \hline 0 \end{array}\right).$$

The corresponding linear logical consensus system is

$$\begin{cases} x_1(t+1) = u_j(t)\,, \\ x_2(t+1) = u_j(t)\,, \\ x_3(t+1) = x_2(t)\,, \\ x_4(t+1) = x_2(t)\,, \\ x_5(t+1) = x_1(t)\,, \end{cases}$$

operating condition	d_i
correct agent	0
inverted agent	1
stuck on 0	$F_i(x, u_j)$
stuck on 1	$\neg F_i(x, u_j)$

Table 1
Possible operating modes of the generic i–th agent

and the visibility diameter $\kappa(C, V_j) = 2$.

Consider the evolution of the system from the initial state $x^0 = (x_1^0, x_2^0, x_3^0, x_4^0, x_5^0)^T$ and input $u_j(t) = \bar{u}_j$ for $t \geq 0$. Direct computation gives $x(1) = (\bar{u}_j, \bar{u}_j, x_2^0, x_2^0, x_1^0)^T$ and $x(t) = x(2) = (\bar{u}_j, \bar{u}_j, \bar{u}_j, \bar{u}_j, \bar{u}_j)^T$ for $t \geq 2$. ♦

5 Dealing with Agent Failure

Consider the case where some agents may incorrectly update their binary states, because of internal failures due to spontaneous malfunctioning or malicious intervention. In this work we assume that a faulty agent may either update its state x_i with the complement of the correct value $F_i(x, u_j)$, or be stuck at the constant values 0 or 1. As a consequence, the behavior of a generic agent $\mathcal{A}_i$ can be described by an equation of the type

$$x_i(t+1) = F_i(x(t), u_j(t)) \oplus d_i \,, \qquad (6)$$

where $F_i : \mathbb{B}^n \times \mathbb{B} \to \mathbb{B}$ is the *nominal update function* and $d_i \in \mathbb{B}$ is a *binary disturbance* that can take on the forms listed in Table 1. We say that an agent $\mathcal{A}_i$ is *correct* if it applies the nominal update function F_i to determine its state x_i (i.e. $d_i = 0$), and *fault* otherwise.

The presence of a faulty agent may in general prevent the establishment of the correct consensus within a network of agents exploiting the linear logical update rule, $F_i = C^*(i, :)\, x + V_j(i)\, u_j$, as shown in the following example.

Example 5.1 (Linear Consensus Failure) *Consider a network composed of $n = 5$ agents that need to consent on the input u_j through direct visibility and message exchange. Suppose that the network is characterized by the communication and visibility matrices*

$$C = \begin{pmatrix} 1&0&1&0&0 \\ 0&1&1&1&0 \\ 1&1&1&1&0 \\ 0&0&0&0&0 \\ 1&1&1&0&1 \end{pmatrix}, \; V_j = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

The corresponding linear logical consensus system is

$$\begin{cases} x_1(t+1) = u_j \,, \\ x_2(t+1) = u_j \,, \\ x_3(t+1) = x_1(t) \,, \\ x_4(t+1) = u_j \,, \\ x_5(t+1) = x_1(t) \,. \end{cases}$$

If, for example, agent $\mathcal{A}_1$ is faulty ($d_1 \neq 0$), the equilibrium point that is reached by the system is

$$\bar{x} = (u_j \oplus d_1, u_j, u_j \oplus d_1, u_j, u_j \oplus d_1)^T \neq \mathbf{1}_n u_j \,,$$

which shows that the network is inconsistent and has not reached the desired consensus. ♦

Our problem is equivalent to make a binary vote based on a number of independent evaluations, some of which may be compromised or faulty. It is well known that, to tolerate up to γ faults, it is sufficient that the voter always have at least $r = 2\gamma + 1$ independent evaluations, so that at least $\gamma + 1$ of them, the majority, are guaranteed to be correct and consistent [12].

In our context, we assume that every sensor reading is correct as well as the computation of the nominal update function F_i, but a faulty agent $\mathcal{A}_j$ may be subject to the disturbance term $d_j \neq 0$, which alters its final decision and the data that it shares via message exchange. Under this hypothesis, the strategy to update every agent state can be the following. If $\mathcal{A}_i$ is able to see the input u_j, its update rule can simply be $x_i(t+1) = u_j$. If $\mathcal{A}_i$ is unable to see u_j, its state can be updated by applying the majority rule on r estimates x_h out of the ones received from its C–neighbors:

$$x_i(t+1) = \begin{cases} 0 \text{ if } \mathrm{card}(S_0(t)) > \mathrm{card}(S_1(t)) \,, \\ 1 \text{ if } \mathrm{card}(S_0(t)) < \mathrm{card}(S_1(t)) \,, \end{cases} \qquad (7)$$

where $S_z(t) = \{h \,|\, C_{i,h} = 1, x_h(t) = z\}$ (note that the case $\mathrm{card}(S_0(t)) = \mathrm{card}(S_1(t))$ cannot occur since r is odd). The above majority rule can be written by using only binary operations. Indeed, Eq. 7 requires that $\mathcal{A}_i$ sets to 1 its state if, and only if, at least $\gamma + 1$ messages received from its C–neighbors contain values x_h set to 1, i.e., if there exists a choice of indices $i_1, \cdots, i_{\gamma+1} \in K_i$, with $K_i = \{h \,|\, C_{i,h} = 1\}$, s.t. $x_{i_1} = 1, x_{i_2} = 1, \cdots, x_{i_{\gamma+1}} = 1$, or equivalently s.t. $x_{i_1} x_{i_2} \cdots x_{i_{\gamma+1}} = 1$. Having denoted with $S_i = S(K_i, \gamma + 1)$ the set composed of $(\gamma + 1)$–tuples from elements extracted from K_i, the rule can be written as $x_i(t+1) = \sum_{H \in S_i} \Pi_{h \in H} x_h$.

Note that, if $\gamma^* \leq \gamma$ is the actual number of faults, the number λ_{γ^*} of tuples in S_i that are guaranteed to give the correct estimate of u_j equals the number of combinations of $\gamma+1$ elements extracted from the index set of the remaining correct agents, which is composed of $2\gamma+1-\gamma^*$ elements, i.e.

$$\lambda_{\gamma^*} = \binom{2\gamma+1-\gamma^*}{\gamma+1} = \frac{(2\gamma+1-\gamma^*)!}{(\gamma+1)!(\gamma-\gamma^*)!} \,.$$

In the worst case we have $\gamma^* = \gamma$ and thus $\lambda_\gamma = 1$, which guarantees that there exists at least one such secure product.

Moreover, starting from an initial condition in which the maximum fault number constraint is satisfied, it is necessary that u_j is propagated while guaranteeing that the constraint remains satisfied. To this aim, consider the sequence of binary vectors $I_j^{(k)}$, each containing a non–null element for the agents that are reached from u_j with *multiplicity* r in at most k steps:

$$I_j^{(k)}(i) = \begin{cases} V_j(i) & k=1\,, \\ I_j^{(k-1)}(i) & k>1,\ \mathrm{card}(K_i^k) < r\,, \\ 1 & k>1,\ \mathrm{card}(K_i^k) \geq r\,, \end{cases}$$

with $K_i^k = \{h \,|\, C(i,h) I_j^{(k-1)}(h) = 1\}$. Let

$$\kappa_i^r = \min_k \{k \,|\, I_j^{(k)}(i) = 1\}$$

be the step at which $\mathcal{A}_i$ is first reached with multiplicity r from u_j and $\kappa^r(C,V_j) = \max\{\kappa_1^r, \cdots, \kappa_n^r\}$ be the visibility diameter with multiplicity r. We can introduce the reachability matrix

$$R_j^r(C,V_j) = \left(I_j^{(1)}\, I_j^{(2)} \,\cdots\, I_j^{(n)} \right) \,,$$

whose columns tell us which agents can be securely reached from input u_j.

Definition 7 *A pair (C,V_j) is said to be (completely) reachable with multiplicity r, or r–*reachable *for short, if the span of the reachability matrix $R_j^r(C,V_j)$ is the entire graph or equivalently*

$$I_j^{(n)} = \mathbf{1}_n \,.$$

Let $K_i^* \subseteq K_i^{\kappa_i}$ a minimum index set s.t. $\mathrm{card}(K_i^*) = r$. We can now discuss the robustness of the above described nonlinear consensus system. The following theorem is a solution to Problem 2.

Theorem 3 (Robust Nonlinear Consensus) *Given a maximum number γ of possible faults and a $(2\gamma+1)$–reachable pair (C,V_j), where C is a communication matrix and V_j is a visibility vector, the nonlinear logical system*

$$\begin{cases} x(t+1) = F^*(x(t), u_j(t))\,, \\ x(0) = x^0\,, \end{cases} \tag{8}$$

where x^0 is an initial state and $F^* = (F_1^*, \cdots, F_n^*)^T$, with

$$F_i^* : \mathbb{B}^n \times \mathbb{B} \to \mathbb{B}$$
$$(x,u_j) \mapsto \begin{cases} u_j & \text{if } V_j(i) = 1\,, \\ \sum_{H \in S_i^*} \Pi_{h \in H} x_h & \text{if } V_j(i) = 0\,, \end{cases}$$

with $S_i^* = S(K_i^*, \gamma+1)$, is (C,V_j)–compliant and, for all piecewise constant inputs $u_j(t) = \bar{u}_j$, globally converges in at most $\kappa^{2\gamma+1}(C,V_j)$ steps to an equilibrium $\bar{x} = (\bar{x}_1, \cdots, \bar{x}_n)^T$ s.t.

$$\bar{x}_i = \bar{u}_j$$

for all i corresponding to correct agents, i.e. with $d_i = 0$.

Proof 2 *Let us first show that the state*

$$\bar{x} = \mathbf{1}_n \bar{u}_j \oplus (d_1, \cdots, d_n)^T$$

is an equilibrium of the system in Eq. 8, perturbed by a disturbance vector $d = (d_1, \cdots, d_n)^T$ satisfying the maximum fault number constraint, i.e. with $card(\{i \,|\, d_i = 1\}) \leq \gamma$. We need to show that

$$F(\bar{x}, \bar{u}_j) \oplus d = \bar{x}\,.$$

Let us proceed by considering the agents as they are reached by the sequence of vectors $I_j^{(k)}$. If $\mathcal{A}_i$ is reached after one step (i.e. it is able to directly measure the input being $V_j(i) = 1$), its perturbed update function gives

$$x_i(t+1) = \bar{u}_j \oplus d_i = \bar{x}_i\,,$$

which trivially satisfies the condition. If $\mathcal{A}_i$ is reached after two steps, its update function depends on messages x_h received from a subset composed of $2\gamma+1$ agents that were reached after one step. Since we have $x_h = \bar{u}_j \oplus d_h = \bar{u}_j \neg d_h + \neg \bar{u}_j d_h$, $\mathcal{A}_i$'s perturbed update function gives

$$\begin{aligned} x_i(t+1) &= \left(\textstyle\sum_{H \in S_i^*} \Pi_{h \in H} x_h \right) \oplus d_i = \\ &= (\bar{u}_j\, A + \neg \bar{u}_j\, B) \oplus d_i\,, \end{aligned}$$

with $A = \sum_{H \in S_i^*} \Pi_{h \in H} \neg d_h$ and $B = \sum_{H \in S_i^*} \Pi_{h \in H} d_h$. Recall that $\mathcal{A}_i$ is guaranteed by hypothesis to receive at least $\gamma+1$ correct estimates of $\bar{u}_j$ from its neighbors and that the cardinality of every tuple in S_i^* is exactly $\gamma+1$. Therefore, there always exists a tuple $H \in S_i^*$ s.t. $d_h = 0$ for all $h \in H$, and thus s.t. $\Pi_{h \in H} \neg d_h = 1$, which *implies $A = 1$. Moreover, provided that the maximum fault number constraint is satisfied, there does not exist a tuple $H \in S_i^*$ s.t. $d_h = 1$ for all $h \in H$, which implies $B = 0$. Therefore, we finally have*

$$x_i(t+1) = (\bar{u}_j\, A + \neg\bar{u}_j\, B) \oplus d_i = \bar{u}_j \oplus d_i = \bar{x}_i\,.$$

The fact that $\bar{x}$ is an equilibrium is proved by carrying on the same reasoning for all agents as they are encountered by $I_j^{(k)}$. For every correct agent $\mathcal{A}_i$, the disturbance is null, i.e. $d_i = 0$, which guarantees that they all consent on the value

$$\bar{x}_i = \bar{u}_j \oplus d_i = \bar{u}_j \oplus 0 = \bar{u}_j\,.$$

Let us also prove that $\bar{x}$ is the unique equilibrium and that is global stable. Without loss of generality, suppose that the agents are sorted by the order they are reached by the vectors $I_j^{(k)}$. Consider first the agents that are reached after one step. Their update function F_i^* is independent of the state and thus their corresponding rows in the incidence matrix are null. If $\mathcal{A}_i$ is an agent reached after k steps, its updated function F_i^* depends on a subset composed of $2\gamma + 1$ agents that were reached after $k' < k$ steps, and that necessarily need to have an index $h < i$. Therefore, the incidence matrix of F^* is strictly lower triangular. In the general case, agents are reached by the vectors $I_j^{(k)}$ in generic order, but they can be reordered by a permutation matrix P according to the sequence of vectors $I_j^{(k)}$ itself. This also implies that $B(F^*)$ is similar to a strictly lower triangular matrix and thus its spectral radius needs to be null, i.e. $\rho(B(F^*)) = 0$ (see Theorem 1). In conclusion, the update function F^* is contractive, i.e. it possesses a unique, globally stable equilibrium point $\bar{x}$. The proof of the convergence time follows on the same line of the linear consensus approach above, and thus it is omitted. Finally note that F^* is (C, V_j)–compliant by construction.

Example 5.2 (Cont'd) *Consider again the network of Example 5.1. Suppose that at most $\gamma = 1$ agents may be compromised and incorrectly setting their state.*

The required multiplicity is $r = 2\gamma + 1 = 3$, which is satisfied by the pair (C, V_j) as shown by the vector sequence

$$I_j^{(1)} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix},\ I_j^{(2)} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix},\ I_j^{(3)} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix},$$
$$I_j^{(4)} = I_j^{(3)}.$$

The synthesis of the robust nonlinear consensus is feasible and gives the update rule

$$\begin{cases} x_1(t+1) = u(t)\,, \\ x_2(t+1) = u(t)\,, \\ x_3(t+1) = x_1(t)\,x_2(t) + x_1(t)\,x_4(t) + \\ \qquad\qquad + x_2(t)\,x_4(t)\,, \\ x_4(t+1) = u(t)\,, \\ x_5(t+1) = x_1(t)\,x_2(t) + x_1(t)\,x_3(t) + \\ \qquad\qquad + x_2(t)\,x_3(t)\,. \end{cases}$$

As an example, suppose that agent $\mathcal{A}_1$ uncooperatively sets its state with the perturbed update function $x_1(t+1) = \bar{u}_j \oplus d_1$, $d_1 \neq 0$. After three steps the network converges to the equilibrium point $\bar{x} = (\bar{u}_j \oplus d_1, \bar{u}_j, \bar{u}_j, \bar{u}_j, \bar{u}_j)^T$, where all correct agents consent on the values $\bar{u}_j$. As another example, suppose that agent $\mathcal{A}_3$ applies the uncooperative update function $x_3(t+1) = (x_1(t)\,x_2(t) + x_1(t)\,x_4(t) + x_2(t)\,x_4(t)) \oplus d_3$, with $d_3 \neq 0$. Again the network reaches the equilibrium point $\bar{x} = (\bar{u}_j, \bar{u}_j, \bar{u}_j \oplus d_3, \bar{u}_j, \bar{u}_j)^T$, and all correct agents are able to consent on the value of $\bar{u}_j$. ♦

Remark 2 *It is straightforward to show that, for $\gamma = 0$, Theorem 3 produces a linear logical consensus system that is equivalent to the one obtained by Theorem 2. In principle, one could use the nonlinear approach even for the case $\gamma = 0$; however, the reachability test and design approach presented for the hypothesis of virtuous scenario are simpler and thus should be used when possible.*

Remark 3 (Link failures) *The introduction of the binary disturbance vector d in Eq. 6 has enabled the analysis of a logical system's behavior in the event of process failure. It also allows discussing how consensus systems designed according to Theorem 3 behave in the presence of communication failures involving* message loss, *i.e. when some messages from a sender agent $\mathcal{A}_h$ may not reach their recipient agent $\mathcal{A}_i$. Message loss can be caused by a number of factors, including signal degradation over the network medium, channel congestion, corrupted packets rejected in–transit, faulty networking hardware [19].*

As these systems are synchronous, every agent $\mathcal{A}_i$ must receive, at every step t, the status of all their C–neighbors, in order to update their current state x_i through the function $F_i(x, u_j)$. If the current status of an agent $\mathcal{A}_h$ is not received from time $\bar{t} + 1$, a reasonable choice for $\mathcal{A}_i$ is that of holding the most recently received value $x_h(\bar{t})$. In dealing with this type of malfunctioning, the worst case occurs when the failure of a link is permanent, i.e. when no message from a sender $\mathcal{A}_h$ can ever reach its recipient $\mathcal{A}_i$. Let M be the set of agents whose messages are lost by at least one of their recipients. From the time $\bar{t}+1$, *the dynamic behavior of all agents in M, as seen from all other the agents, is described by Eq. 6, where $d_h(t) \equiv 0$ if $\mathcal{A}_i$ can receive $\mathcal{A}_h$'s messages, or*

$$d_h(t) = \begin{cases} F_h(x(t), \bar{u}_j) & \text{if } x_h(\bar{t}) = 0 \\ \neg F_h(x(t), \bar{u}_j) & \text{if } x_h(\bar{t}) = 1 \end{cases}, \quad \text{for } t > \bar{t}\,.$$

otherwise. Under the hypotheses of Theorem 3, where γ is the maximum number of failures in the incoming links of every agent $\mathcal{A}_i$, all agents $i \notin M$ correctly converge to a state $\bar{x}_i = \bar{u}_j$. Moreover, observing that also the agents $i \in M$ are correct processes (only their outgoing links are failed), they correctly compute the update function F_i. Hence, the entire system converges to the consensus state $\bar{x} = \mathbf{1}_n \bar{u}_j$.

6 Distributed Synthesis of Logical Maps

In the previous Sections 4 and 5, we have presented two strategies allowing all (correct) agents of a network to consent on the value of the j–th input u_j. It is worth noting that, given a logical map f, only a subset of its inputs are actually needed for its computation, which is captured in the following:

Definition 8 *A binary input u_j is* essential *for a decision system $y = f(u) \in \mathbb{B}^p$ if, and only if, the incidence matrix of f satisfies the following relation*

$$\textstyle\sum_{i=1}^{p} B(f(u))(i,j) = 1\,.$$

Given a map f with $\mu \leq m$ essential inputs, consider a simplified logical map $f^* : \mathbb{B}^\mu \to \mathbb{B}^p$ obtained by fixing the values of all its non–essential inputs. Note that f^* is equivalent to f, i.e. it possesses the same truth table.

We can now show how the agents can agree on a generic decision system , which is the main result of the paper:

Theorem 4 (Distributed Synthesis) *Given an input visibility matrix V, a communication matrix C, and a maximum number γ of faults, a generic decision system $y = f(u)$, with $f : \mathbb{B}^m \to \mathbb{B}^p$ and u a piecewise constant input, can be computed in a distributed way if the following* feasibility inequality *holds:*

$$(\neg I_1^{(n)} \cdots \neg I_m^{(n)})^T \mathbf{1}_n \leq \neg \left(B(f(u))^T \, \mathbf{1}_m \right)\,. \tag{9}$$

Moreover, network consensus on the decision y can be obtained if every agent $\mathcal{A}_i$ run the logical consensus rule

$$\begin{aligned} X_i(t+1) &= F_i(X(t), u(t))\,, \\ Y_i(t) &= G_i(Z_i(t)) = (f_1^*(Z_i(t)), \cdots, f_p^*(Z_i(t)))\,, \end{aligned}$$

where $X \in \mathbb{B}^{n \times \mu}$ and $Y \in \mathbb{B}^{n \times p}$ are the network's state and output, respectively, μ is the number of f's essential inputs, $f^* = (f_1^*, \cdots, f_p^*)$ is a simplified map equivalent to f involving only such inputs, $F_i = (F_{i,1}, \cdots, F_{i,\mu})^T$, where $F_{i,j}$ is designed by applying Theorem 2, if $\gamma = 0$, or Theorem 3, if $\gamma \geq 0$, to the j–th essential input, $Z_i = (Z_{i,1}, \cdots, Z_{i,\mu})$, and

$$Z_{i,j} = \begin{cases} u_j & \text{if } V_j(i) = 1 \\ X_{i,j} & \text{otherwise} \end{cases}.$$

Proof 3 *Let us first prove the feasibility inequality. Its left term is the binary vector*

$$(\neg I_1^{(n)} \cdots \neg I_m^{(n)})^T \mathbf{1}_n = \begin{pmatrix} \sum_{i=1}^{n} \neg I_1^{(n)}(i) \\ \vdots \\ \sum_{i=1}^{n} \neg I_m^{(n)}(i) \end{pmatrix},$$

whose j–the component, $\sum_{i=1}^{n} \neg I_j^{(n)}(i)$, equals the unity if, and only if, $I_j^{(n)}(i) = 0$ for some i, i.e., at least an agent is unreachable from input u_j. In this case, the unique possibility to expect the problem feasibility is that input u_j is unnecessary for the computation of f.

By applying De Morgan's law, the inequality's right term can be written as

$$\neg(B(f(u))^T \, \mathbf{1}_m) = \neg \begin{pmatrix} \sum_{i=1}^{p} b_{i,1} \\ \vdots \\ \sum_{i=1}^{p} b_{i,p} \end{pmatrix} = \begin{pmatrix} \Pi_{i=1}^{p} \neg b_{i,1} \\ \vdots \\ \Pi_{i=1}^{p} \neg b_{i,p} \end{pmatrix},$$

which is a vector whose j–th element, $\Pi_{i=1}^{p} \neg b_{i,j}$, equals the unity if, and only if, $b_{i,j} = 0$ for all i, i.e. none of the decision functions f_i depends on u_j. The inequality itself simply expresses the requirement that network must be reachable from every input that is essential for the computation of f.

Furthermore, whenever the inequality is satisfied, every input can be (robustly) propagated through the network according to the state update rule F_i of Theorem 2 if $\gamma = 0$ or Theorem 3 if $\gamma \geq 0$. Recall from Remark 2, that the two theorems produce two equivalent linear logical consensus systems for $\gamma = 0$, but the first design procedure is simpler and thus preferable when possible.

As for the output decision function G_i, observe that, if a generic agent $\mathcal{A}_i$ is able to measure the j–th input, it can directly use it to evaluate the decision function f. If not, it can use the corresponding local state component $X_{i,j}$ that, after consensus is reached, will contain the agreed *value u_j. This strategy can be formally described by introducing a fictitious variable $Z_{i,j}$ that equals u_j in the first case, $X_{i,j}$ otherwise, which concludes the proof.*

Example 6.1 *Consider the following task involving computation of three decisions $y = (y_1, y_2, y_3)$ depending on four inputs $u = (u_1, u_2, u_3, u_4)$:*

$$\begin{cases} y_1(t) = u_1(t)\,\neg u_3(t)\,, \\ y_2(t) = u_3(t)\,, \\ y_3(t) = \neg u_2(t)\,u_4(t) + u_1(t) + \neg u_2(t)\,\neg u_4(t)\,. \end{cases} \tag{10}$$

Assume that a network of $n = 4$ agents is available, which is characterized by the visibility and communication matrices

$$C = \begin{pmatrix} 1&1&0&0 \\ 0&1&0&1 \\ 0&1&1&0 \\ 1&0&0&1 \end{pmatrix}, \quad V = \begin{pmatrix} 1&0&0&0 \\ 0&1&1&0 \\ 0&1&0&1 \\ 1&0&1&0 \end{pmatrix}.$$

Let us suppose for simplicity that no fault can occur ($\gamma = 0$), so that a linear logical consensus rule can be adopted to propagate each input. The span vectors of the inputs are $I_1 = I_2 = I_3 = (1,1,1,1)^T$ and $I_4 = (0,0,1,0)^T$, which tell us the network is completely reachable from u_1, u_2 and u_3, while only agent $\mathcal{A}_3$ is reachable from u_4.

Moreover, based on the incidence matrix of the decision function f,

$$B(f(u)) = \begin{pmatrix} 1&0&1&0 \\ 0&0&1&0 \\ 1&1&0&0 \end{pmatrix},$$

input u_4 is not essential for its computation, and thus no problem is posed by the fact that the network is unreachable from it. The distributed synthesis problem is solvable since the feasibility inequality is indeed satisfied. Furthermore, the computation of function f_3 formally depends on u_4, but a distributed evaluation of the function cannot involve it. To remove this input, we can replace it with any value chosen at convenience, e.g. $u_4(t) = 0$, which gives the simplified decision function

$$\begin{cases} y_1(t) = u_1(t)\,\neg u_3(t)\,, \\ y_2(t) = u_3(t)\,, \\ y_3(t) = u_1(t) + \neg u_2(t)\,. \end{cases} \tag{11}$$

By computing an IPST for each input, we obtain the following linear logical consensus system:

$$\begin{pmatrix} X_1(t+1) \\ X_2(t+1) \\ X_3(t+1) \\ X_4(t+1) \end{pmatrix} = \begin{pmatrix} u_1(t) & X_{2,2}(t) & X_{2,3}(t) \\ X_{4,1}(t) & u_2(t) & u_3(t) \\ X_{2,1}(t) & u_2(t) & X_{2,3}(t) \\ u_1(t) & X_{4,1}(t) & u_3(t) \end{pmatrix},$$

where the generic agent's state is $X_i \in \mathbb{B}^{1\times q}$ with $q = 3$. Note that the i–th row of the right term of last equation represents the i–th local update function F_i.

Moreover, each output decision map G_i can readily be obtained by replacing in Eq. 11 every input u_j that is not visible from $\mathcal{A}_i$ with the corresponding state component $X_{i,j}$. By doing this, we obtain

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{pmatrix} = \begin{pmatrix} u_1\,\neg X_{1,3} & X_{1,3} & u_1 + \neg X_{1,2} \\ X_{2,1}\,\neg u_3 & u_3 & X_{2,1} + \neg u_2 \\ X_{3,1}\,\neg X_{3,3} & X_{3,3} & X_{2,1} + \neg u_2 \\ u_1\,\neg u_3 & u_3 & u_1 + \neg X_{4,2} \end{pmatrix},$$

where the dependence from t is omitted for brevity. ♦

Remark 4 *The method proposed in Theorem 4 can be straightforwardly extended to some switching–topology networks, in which the topology switching signal is known. However, such an extension would require an IPST to be computed for each possible topology configuration, and thus it would remain applicable only to small size networks.*

7 Application to Intrusion Detection

7.1 Distributed Detection of Physical Intruders

Consider the problem of detecting possible physical intruders within an indoor environment $\mathcal{W}$. Suppose that the environment is divided into m rooms, $\mathcal{W}_i$, $i = 1, \ldots, m$, separated by walls, and that $n \geq m$ guards are responsible for patrolling the rooms. Each guard has sensors with star–shaped visibility ($V_{i,j} = 1$ if, and only if, an intruder in region $\mathcal{W}_j$ can be seen by agent $\mathcal{A}_i$) and can communicate only with neighboring guards that are within line–of–sight ($C_{i,j} = 1$ if, and only if, $\mathcal{A}_i$ can see $\mathcal{A}_j$). Presence or absence of an intruder in the j–th region $\mathcal{W}_j$ can be described by a binary input u_j and the guards' network is required to compute the logical decisions $y_i(t) = u_i(t)$, $i = 1, \cdots, m$. Denote with $X \in \mathbb{B}^{n\times m}$ the alarm state of the system: $X_{i,j} = 1$ if agent $\mathcal{A}_i$ reports an alarm about the presence of an intruder in region $\mathcal{W}_j$. The alarm can be set because an intruder is actually detected by the agent itself, or based on the information exchanged with neighboring guards.

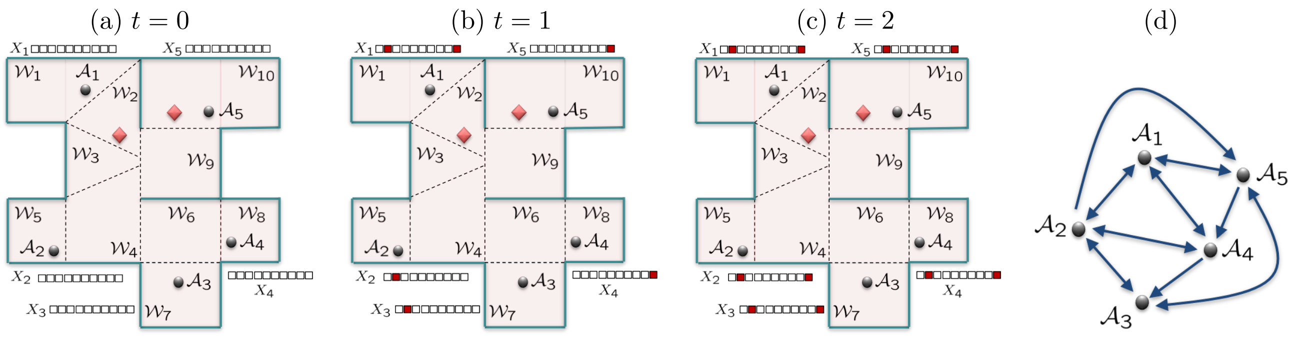

Figure 1. (a)–(c) Run of the linear consensus system with 2 intruders (rhombus) in regions $\mathcal{W}_2$ and $\mathcal{W}_{10}$, respectively. The sequence in the figure shows that a correct agreement is reached (X_i's components are represented by empty (filled) boxes when no (at least one) intruder is detected in the corresponding region). (d) Available communication graph C.

In this context, our objective is to design a logical consensus system of the form $X(t+1) = F(X(t), u(t))$, with $u = (u_1, \ldots, u_m)^T$, so that every agent can provide, *at consensus*, consistent complete information on the environment if polled, i.e. $X_{i,j} = X_{k,j}\ \forall i, k$ and $\forall j$. This requires that F has a unique equilibrium depending on the corresponding column of $\mathbf{1}_n f(u)^T = \mathbf{1}_n u^T$.

In case of a virtuous scenario, we can apply Theorem 2, which produces a linear logical consensus of the form $X(t+1) = F\,X(t) + B\,u(t)$, where each row basically expresses the rule that an observer alarm is set at time $t+1$ if it sees an intruder (through u), or if one of its communication neighbors was set at time t. The visibility diameter is $\kappa(C, V) = 2$, which will correspond to the maximum number of steps before consensus is reached. Fig. 1 shows snapshots from a typical execution of this linear consensus system where every agents converge to consensus after 2 steps. If all agents correctly set their alarm states, the system correctly converges to a state where all columns of X are either zero or one. However, this system is not robust to permanent faults (Fig. 2). A more conservative mechanism can be obtained by applying Theorem 3, with $\gamma = 1$, that generates a nonlinear rule requiring that agent $\mathcal{A}_i$ sets an alarm regarding $\mathcal{W}_j$ at time $t+1$ if at least two neighboring sensors having visibility on $\mathcal{W}_j$ are in alarm at time t, or if it sees an intruder (through u). The false alarm raised by the misbehaving agent $\mathcal{A}_1$ is thus correctly handled by the second system.

7.2 *Detection of Malicious Users in Networked Distributed Systems*

Consider a network of n hosts comprising a set Γ of fully operational workstations and a set W of simpler computers with limited functionalities. Neighboring hosts can communicate via wired and/or wireless links. Authorized users can log in at any host and are allowed to create files, launch applications, open P2P connections, etc. Protection against malicious users trying to spread viruses or spyware through the network, to e.g. cause damage or gather classified information, is necessary. We assume that traces of possible threats are known so that a model–based approach to such intrusion detection problem can be adopted. We assume also that the generic i–th host is able to measure input events generated by actions of a user on the same host and on neighboring ones. The list of events corresponding to critical user's actions are reported in Table 2.

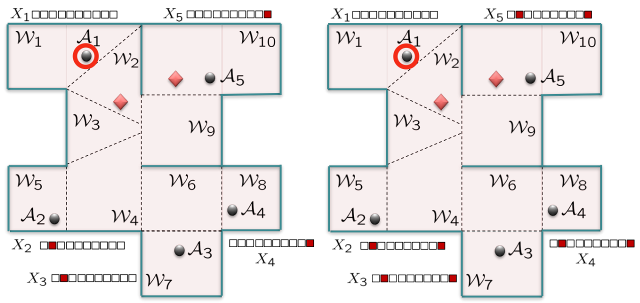

Figure 2. Final network decisions with permanent fault of $\mathcal{A}_1$ incorrectly setting its state to 0. The correct agreement is not reached by the linear consensus system (left), whereas $\mathcal{A}_i$'s misbehavior is tolerated by the nonlinear one (right).

Two types of attacks are assumed to be possible depending on the host type the menace is started from. On a fully operational workstation, a manifest evidence of an attack is represented by having the user creating a file in the host's system folder, launching an application on it, establishing a large number of connections with another host j, launching a remote application on j, and then establishing a large number of connections with a third host k. From a simpler computer, an intruder may try to get some hosts stuck by establishing a round connection among them and creating no file.

The input event vector is $u \in \mathbb{B}^m$, with

$$u = (a_1, \cdots, a_n, b_1, \cdots, b_n, c_1, \cdots, c_n,\\ d_{1,1}, \cdots, d_{1,n}, d_{2,1}, \cdots, d_{2,n}, \cdots, d_{n,1}, \cdots, d_{n,n})^T,$$

Event	Description
a_i	The user creates a file in its home folder on the i–th workstation
b_i	The user creates a file in the i–th workstation's system folder
c_i	The user launches an application on the i–th workstation
$d_{i,j}$	The user opens more than p TCP-IP connections between the i–th and j–th workstations

Table 2
Possible events generated by a malicious user trying to attack a distributed computer network.

and $m = n^2 + 3n$. The two considered attacks can be detected by the centralized decision task

$$y = (y_1, y_2)^T = f(u),$$

where the logical vector function is

$$f : \mathbb{B}^m \to \mathbb{B}^2$$
$$u \mapsto \begin{pmatrix} \sum_{i\in\Gamma} b_i\, c_i \sum_{j\neq i} \left(d_{i,j}\, c_j \sum_{k\neq i,j} d_{j,k}\right) \\ \sum_{i\in W} \neg a_i \sum_{j\neq i} \left(d_{i,j}\, \neg a_j \left(\sum_{k\neq i,j} \neg a_k\, d_{k,i}\right)\right) \end{pmatrix}$$

The communication matrix C is s.t. $C_{i,j} = 1$ if, and only if, host j is a neighbor of host i. Moreover, input events a_i, b_i, c_i, for $i = 1, \cdots N$, can be directly seen from the i–th host and its one–hop neighbors, while the input events $d_{i,j}$, for $i, j = 1, \cdots, N$, $i < j$, can be seen both form the i–th and j–th hosts, and from their neighbors, i.e.

$$V = (V_a, V_b, V_c, V_d)\ ,$$

with $V_a = V_b = V_c = \bar{V}$, $\bar{V}(i, j) = 1$, if, and only if, host j is a neighbor of host i, and

$$V_d(i, j) = \bar{V}\,(i, \alpha(j)) + \bar{V}\,(j, \alpha(i))\ ,$$

with $\alpha(k) = \left\lfloor \frac{k}{n} \right\rfloor + 1$. We assume that at most γ hosts can return incorrect estimates of a user action. We want to realize a distributed network intrusion detector that is able to discover a malicious user only via communication and consensus.

By using the approach proposed in Section 6, we have realized a distributed network intrusion detection system for a system with $n = 50$ hosts, and thus $m = 2650$ input events. The size of each agent state is aligned with 332 bytes, which can be afforded by commercially available wireless connection types. Indeed, both the 802.11x and the 802.15.4 protocols allow for exchanging messages of at most 2200 bytes and 169 bytes, respectively. Thus, each agent can share its full state with one of its neighbors by sending at most 2 messages of such commercial protocols. We have assumed $\gamma = 2$ and chosen a communication matrix C ensuring that each pair (C, V_j) is 5–reachable. The obtained visibility diagram is $\kappa(C, V) = 23$. In the simulation, a malicious user is

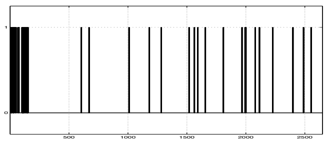

Figure 3. Value of the input event vector $u = (a_1, \cdots, a_{50}, b_1, \cdots, b_{50}, c_1, \cdots, c_{50}, d_{1,1}, \cdots, d_{50,50})$, representing the activity of a malicious user, including creation of a file in the system's folder of host 33, launch of an application on the same host, opening a large number of connections from host 33 to host 50, remotely start of an application on host 50, and opening of a large number of connections from host 50 to host 1.

performing an attack of the first type that is "hidden" in other normal operations: it creates a file in the system's folder of host 33, launches an application on the same host, opens a large number of connections from host 33 to host 50, remotely starts an application on host 50, and finally opens a large number of connections from host 50 to host 1. Moreover, in the simulation, a first compromised host, host 23, always returns the opposite of the value that it detects, while a second compromised host, host 38, is stuck to the value 0. Fig. 3 reports the network's input representing the above described behavior of the user. The centralized decision system applied to the above described input gives $y = f(u) = (1, 0)^T$, indicating that an attack of the first type is recognized. Each host is able to process local information as well as information received from neighboring monitors, via a distributed nonlinear consensus rule that is obtained by applying Theorem 4. Fig. 4 reports the evolution of the network consensus, while Fig. 5 shows that the total disagreement $e = (e_1, e_2)^T$ of local hosts w.r.t. the centralized detection task $f(u)$, i.e.

$$e_i = \textstyle\sum_{j=1}^{n} f_i(X(j, :)) \oplus f_i(u)\,,$$

where $X_{i,j}$ is the i–th local monitor estimate of the j–th input event, and $\sum$ represents here the sum of two real numbers, converges to zero. The two figures show that the entire network is able to detect the user maliciousness, as expected from theory, even in the presence of the two compromised local monitors.

8 Conclusion

A novel mechanism enabling network consensus on the value of decision functions depending on binary values

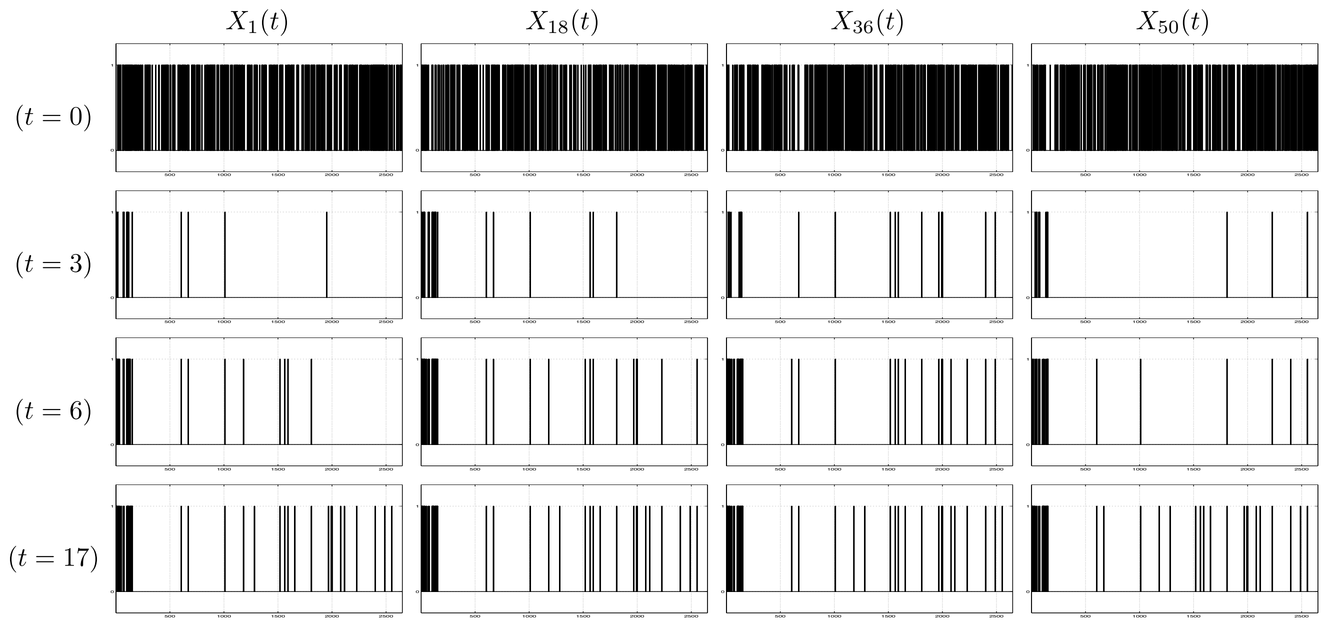

Figure 4. Simulation run of a distributed computer network with 50 hosts. Local hosts are able to detect the user maliciousness by running a distributed nonlinear logical consensus rule. Only 4 of the 50 hosts are reported, at 4 distinct iterations during convergence to consensus, which is reached after 17 steps.

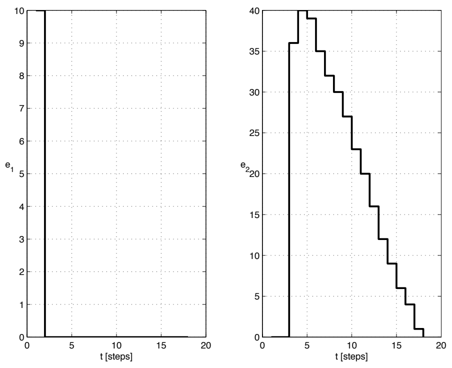

Figure 5. Total disagreement $e = (e_1, e_2)^T$ of local hosts w.r.t. the centralized decision $y = (1, 0)^T$. The convergence of e to zero indicates that the first–type attack executed from the malicious user is detected.

was introduced. Two design procedures for the synthesis of optimal logical consensus systems with and without agent failures was proposed. While the approach allows the design of distributed update rules, the design phase itself requires complete knowledge of the communication and visibility matrices. It is worth saying that, in applications where sub–optimal communication paths are acceptable, one would also consider heuristic flooding approaches to propagate all binary inputs [13]. Future work will study how to obtain a fully distributed design synthesis. Some preliminary results toward this direction can be found in [9].

References

[1] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Englewood Cliffs, 2003.

[2] Vartika Bhandari and Nitin H. Vaidya. On reliable broadcast in a radio network. In *ACM Symp. on Principles of Distr. Computing*, pages 138–147, 2005.

[3] A. Bicchi, A. Fagiolini, and L. Pallottino. Toward a society of robots: Behavior, misbehavior, and security. *IEEE Robotics & Automation Magazine*, 17(4):26–36, December 2010.

[4] VD Blondel, JM Hendrickx, A. Olshevsky, and JN Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. *IEEE Conf. on Decision and Control*, pages 2996–3000, 2005.

[5] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks.* Applied Mathematics Series. Princeton University Press, 2009.

[6] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Trans. on Robotics and Automation*, 20(2):243–255, 2004.

[7] Jorge Cortés. Distributed algorithms for reaching consensus on general functions. *Automatica*, 44(3):726–737, March 2008.

[8] M. Di Marco, A. Garulli, A. Giannitrapani, and A. Vicino. SLAM for a team of cooperating robots: a set membership approach. *IEEE Trans. on Robotics and Automation*, 19(2):238–249, 2003.

[9] A. Fagiolini, S. Martini, D. Di Baccio, and A. Bicchi. A self–routing protocol for distributed consensus on logical information. In *IEEE Intl. Conf. on Intelligent Robots and Systems*, pages 5151–5156, October 2010.

[10] A. Fagiolini, E.M. Visibelli, and A. Bicchi. Logical consensus for distributed network agreement. In *IEEE Conf. on Decision and Control*, pages 5250–5255, 2008.

[11] L. Fang, PJ Antsaklis, and A. Tzimas. Asynchronous Consensus Protocols: Preliminary Results, Simulations and Open Questions. *IEEE Int. Conf. on Decision and Control and Eur. Control Conference*, pages 2194–2199, 2005.

[12] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[13] H Lim and C Kim. Flooding in wireless ad hoc networks. *Computer Communications*, 24:353–363, 2001.

[14] Nancy A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[15] Keith Marzullo. Maintaining the time in a distributed system: A loosely-coupled distributed service. *Dissertation Abstracts Intl. Part B: Science and Engineering*, 46(1), 1985.

[16] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi–agent systems. *Proceedings of the IEEE*, 95(1):215–233, January 2007.

[17] F. Pasqualetti, A. Bicchi, and F. Bullo. Consensus computation in unreliable networks: A system theoretic approach. *IEEE Trans. on Automatic Control*, 57(1):90–104, January 2012.

[18] R. Perlman. *Network layer protocols with byzantine robustness.* PhD thesis, MIT, 1989.

[19] Charles P. Pfleeger and Shari Lawrence Pfleeger. *Security in Computing.* Prentice Hall, 2004.

[20] W. Ren, R.W. Beard, and E.M. Atkins. Information consensus in multi-vehicle cooperative control. *IEEE Control System Magazine*, 27(2):71–82, 2007.

[21] Wei Ren and Randal W. Beard. *Distributed Consensus in Multi–vehicle Cooperative Control: Theory and Applications.* Springer, 2008.

[22] F. Robert. Itérations sur des ensembles finis — convergence d'automates cellulaires contractants. *Linear Algebra and its applications*, 29:393–412, 1980.

[23] L. Schenato and G. Gamba. A distributed consensus protocol for clock synchronization in wireless sensor network. In *IEEE Conf. on Decision and Control*, pages 2289 –2294, December 2007.

[24] S. Sundaram and C.N. Hadjicostis. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE Trans. on Automatic Control*, 56(7):1495 –1508, July 2011.