Abstract—In this paper we introduce a novel consensus mechanism where agents of a network are able to share logical values, or Booleans, representing their local opinions on e.g. the presence of an intruder or of a fire within an indoor environment. Under suitable joint conditions on agents’ visibility and communication capability, we provide an algorithm generating a logical linear consensus system that is globally stable. The solution is optimal in terms of the number of messages to be exchanged and the time needed to reach a consensus. Moreover, to cope with possible sensor failure, we propose a second design approach that produces robust logical nonlinear consensus systems tolerating a maximum number of faults. Finally, we show applicability of the agreement mechanism to a distributed Intrusion Detection System (IDS).

I. INTRODUCTION

A large number of control problems involving a distributed collection of agents require a form of consensus in order to make the system work properly. Consensus problems that are formulated in the control literature generally concern how to reach an agreement on the value of a real scalar quantity of interest \(x \in \mathbb{R}\), such as a room temperature. This is achieved by imposing that every agent share their direct measures of \(x\) and combine these values with information received by neighbors of a communication graph \(G\) [1]–[3]. A typical form of such distributed consensus systems is the following continuous–time linear system:

\[
\dot{x}(t) = Ax(t) + Bu(t),
\]

where \(A \in \mathbb{R}^{n \times n}\) is a strongly connected doubly–stochastic matrix, \(B \in \mathbb{R}^{n \times m}\) is the input matrix, and \(u \in \mathbb{R}^m\) is a control law. The flourishing literature on this topic have studied continuous– and discrete–time, synchronous and asynchronous, and quantized versions of such systems and has provided useful results on properties such as characterization of equilibria, and convergence rate [4]–[6].

Nevertheless, the rich literature on distributed algorithms shows that many other applications would benefit from availability of more general forms of consensus, where participating agents are de facto able to reach an agreement on non–scalar quantities. Relevant examples of applications are represented by consensus algorithms studied in Lynch’s book [7], or the solution proposed by Cortes et al. for achieving consensus on general functions [8]. Other examples are clock synchronization (see e.g. Marzullo’s work [9]), and cooperative simultaneous localization and map building [10]. More recently, a distributed IDS for multi–agents has been proposed in [11], where agents cooperate to establish whether a common neighbor acts according to shared logical rules. In all such application scenarios a form of consensus on intervals or sets is required and thus solutions based on simple linear consensus can not be applied.

In this perspective, we introduce a novel form of consensus, so–called logical consensus, where a number of agents have to decide on the value of a set of decisions depending on logical inputs. The realization of such logical consensus systems can build upon known results in the literature on cellular automata and convergence of finite–state iteration maps [12]–[14]. Our ambitious objective is to develop a synthesis technique for logical consensus systems, that can reach a certain level of systematicity as it happens in the linear case. Indeed, under suitable joint conditions on the visibility of agents and their communication capability, we provide an algorithm generating logical linear consensus systems that are globally stable. The solution is optimal in terms of the number of messages to be exchanged and the time needed to reach a consensus. Moreover, to cope with possible sensor failure, we propose a second design algorithm that produces robust logical consensus systems. Detecting and tolerating misbehavior of some agents has recently received an increasing attention (see e.g. [11], [15]).

II. PROBLEM STATEMENT

We consider control problems requiring computation of a set of \(p\) decisions, \(y_1, \ldots, y_p\), that depend on \(m\) logical events, \(u_1, \ldots, u_m\). Such events may represent e.g. the presence of an intruder or of a fire within an indoor environment. More precisely, for any given combination of input events, we consider a decision task that requires computation of the following system of logical functions:

\[
\begin{align*}
    y_1 &= f_1(u_1, \ldots, u_m), \\
    \cdots \\
    y_p &= f_p(u_1, \ldots, u_m),
\end{align*}
\]

where each \(f_i : \mathbb{B}^m \to \mathbb{B}\) consists of a logical condition on the inputs. Let us denote with \(u = (u_1, \ldots, u_m)^T \in \mathbb{B}^m\) the input event vector, and with \(y = (y_1, \ldots, y_p)^T \in \mathbb{B}^p\) the output decision vector. Then, we will write \(y = f(u)\) as a compact form of Eq. 2. where \(f = (f_1, \ldots, f_p)^T\), with \(f : \mathbb{B}^m \to \mathbb{B}^p\), is a logical vector function. It is worth noting that computation of \(f\) is centralized in the sense that it may require knowledge of the entire input vector \(u\) to determine the output vector \(y\).

Our approach to solve the decision task consists of employing a collection of \(n\) agents, \(A_1, \ldots, A_n\), that are sup-
posed to cooperate and possibly exchange locally available information. We assume that each agent is described by a triple $A_i = (S_i, P_i, C_i)$, where $S_i$ is a collection of sensors, $P_i$ is a processor that is able to perform elementary logical operations such as $\{\text{and}, \text{or}, \text{not}\}$, and $C_i$ is a collection of communication devices allowing transmission of only sequences of binary digits, 0 and 1, namely strings of bits. Although we assume that every agent has the same processing capability, i.e. $P_i = P$ for all $i$, we consider situations where agents may be heterogeneous in terms of sensors and communication devices. Due to this diversity as well as the fact that agents are placed at different locations, a generic agent $i$ may or may not be able to measure a given input event $u_j$, for $j = 1, \ldots, m$. Therefore, we can conveniently introduce a visibility matrix $V \in \mathbb{R}^{n \times m}$ such that we have $V(i, j) = 1$ if, and only if, agent $A_i$ is able to measure input event $u_j$, or, in other words, if the $i$-th agent is directly reachable from the $j$-th input. Moreover, for similar reasons of diversity and for reducing battery consumption, each agent is able to communicate only with a subset of other agents. This fact is captured by introducing a communication matrix $C \in \mathbb{R}^{n \times n}$, where $C(i, k) = 1$ if, and only if, agent $A_i$ is able to receive a data from agent $A_k$. Hence, agents specified by row $C(i, :)$ will be referred to as $C$–neighbors of the $i$–th agent. The introduction of visibility relations between inputs and agents immediately implies that, at any instant $t$, only a subset of agents is able to measure the state of each input $u_j$, for all $j$. Therefore, to effectively accomplish the given decision task, we need that such an information flows from one agent to another, consistently with available communication paths. We require all agents reach an agreement on the centralized decision $y = f(u)$, so that any agent can be polled and provide consistent complete information. In this perspective, we pose the problem of reaching a consensus on logical values.

Furthermore, consider the general fact that all logical expressions in the centralized decision system $y = f(u)$ can formally be written as a combination of a minimal set of $q$ subterms, $l_1, \ldots, l_q \in \mathbb{B}$, i.e. each $f_h$ is in the form $f_h = l_1 \odot_1 l_2 \odot_2 \odot_3 \cdots$, where $\odot_i$ is one of $\{\text{and}, \text{or}, \text{not}\}$, for $i = 1, 2, \odot_3 = \text{not}$, and each subterm may depend on only some of the input components. Due to its minimality, this formal representation is an encoding of the decisions $f$ that is optimal in terms of specification complexity, i.e. the number of bits that are necessary to represent all logical expressions in $f$. To clarify this, consider the following simple example of $p = 2$ decisions depending on $m = 2$ input events:

$$
\begin{cases}
 y_1(t) = u_1(t) \tilde{u}_2(t), \\
 y_2(t) = \tilde{u}_2(t),
\end{cases}
$$

and assume that $n = 4$ agents have the input visibility described by the following $V$:

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

Then, a minimal encoding of the decision system of Eq. 3 is obtained by choosing e.g. $l_1(u) = u_1$, and $l_2(u) = \tilde{u}_2$. Indeed, we can write:

$$
\begin{cases}
 y_1(t) = l_1(t) \odot l_2(t), \\
 y_2(t) = \odot_2(l_2(t)),
\end{cases}
$$

where we have $\odot_1 = \text{and}$, and $\odot_2 = \text{not}$. Note that visibility of each subterm $l_h(u)$ w.r.t. each agent depends on which inputs appear in the subterm itself, and hence another visibility matrix $V$ should be introduced to characterize this fact. Without loss of generality, we will assume that each subterm depends on only one input, so that the same $V$ can be used, i.e. $l_h = \chi_h(u_t)$, for $h = 1, \ldots, q$, where $\chi_h$ are scalar functions, and $l$ is in $\{1, \ldots, m\}$. In the example, we have $l_1(u) = \chi_1(u_1), l_2(u) = \chi_2(u_2)$.

In this view, we can imagine that each agent $A_i$ stores the values of all subterms into a local state vector, $X_i = (x_{i,1}, \ldots, x_{i,q}) \in \mathbb{B}^q$, that is a string of bits. In practice we have $X_{i,h} = l_h$ for all agents $A_i$, and all subterms $l_h$. Then, let us denote with $X(t) = (X_1^T(t), \ldots, X_n^T(t))^T \in \mathbb{B}^{n \times q}$ a matrix representing the network state at a discrete time $t$. Hence, we assume that each agent $A_i$ is a dynamic node that updates its local state $X_i$ through a distributed logical update function $F$ that depends on its state, on the state of its $C$–neighbors, and on the reachable inputs, i.e. $X_i(t+1) = F_i(X_i(t), u(t))$. Moreover, we assume that each agent $A_i$ is able to produce a logical output decision vector $Y_i = (y_{i,1}, \ldots, y_{i,p}) \in \mathbb{B}^p$ through a suitable distributed logical output function $G$ depending on the local state $X_i$ and on the reachable inputs $u$, i.e. $Y_i(t) = G_i(X_i(t), u(t))$. Let us denote with $Y(t) = (Y_1^T(t), \ldots, Y_n^T(t))^T \in \mathbb{B}^{n \times q}$ a matrix representing the network output at a discrete time $t$. Therefore, the dynamic evolution of the network can be modeled by the following distributed finite–state iterative system:

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

where we have $F = (F_1^T, \ldots, F_n^T)^T$, with $F_i : \mathbb{B}^q \times \mathbb{B}^m \to \mathbb{B}^q$, and $G = (G_1^T, \ldots, G_n^T)^T$, with $G_i : \mathbb{B}^q \times \mathbb{B}^m \to \mathbb{B}^p$.

It should be apparent that, given a decision system of the form of Eq. 2, and a minimal encoding, the structure of the output function $G$ is somehow fixed. Indeed, map $G_i$ can readily be obtained by simply replacing those subterms $l_h$ in Eq. 2 that are not visible from agent $A_i$ with the corresponding state component $X_{i,h}$. To show this, consider again the example of Eq. 3 and Eq. 4. For the given $V$, agents’ outputs are $Y_i(t) = (y_{i,1}(t), y_{i,2}(t))$ and can be written as follows:

$$
\begin{align*}
Y_1(t) &= G_1(X_1, u) = (u_1(t) \tilde{u}_2(t), u_2(t)), \\
Y_2(t) &= G_2(X_2, u) = (X_2,1(t) X_2,2(t), X_2,2(t)), \\
Y_3(t) &= G_3(X_3, u) = (X_3,1(t) u_2(t), u_2(t)), \\
Y_4(t) &= G_4(X_4, u) = (u_1(t) X_4,2(t), X_4,2(t)).
\end{align*}
$$

As a matter of fact, the only degree of freedom in the design of a logical consensus is the distributed map $F$ that will be designed based on a given pair $(C, V)$. In this perspective, we are interested in solving the following design problem:
Problem 1 (Globally Stable Synthesis): Given a decision system of the form of Eq. 2, a visibility matrix $V$, and a communication matrix $C$, design a logical consensus system of the form of Eq. 6, that is consistent with $C$ and $V$, and such that, for all initial network state $X(0)$, and all inputs $u$, there exists a finite time $N$ such that the system reaches a consensus on the centralized decision $y^* = f(u)$, i.e. $Y(t) = 1_n (y^*)^T$, for all $t \geq N$.

Another important property is the ability of a distributed logical consensus system to tolerate a given number of possible faulty or misbehaving agents. In this perspective we are interested in solving also the following:

Problem 2 (Robust Design): Under the same hypotheses of Problem 1, and assuming that at most $\gamma$ agents in a set $\Gamma$ can send corrupted data, design a robust logical consensus system guaranteeing that all other agents reach an agreement on the correct consensus value, i.e. $Y(t) = (y^*)^T$, for all $i \not\in \Gamma$, and $t \geq N$.

Finally, we need an operative test to determine whether a distributed map is compliant with a given pair $(C,V)$. To this aim, let us denote with $B(F)$ the incidence matrix of $F$ being a matrix having a generic element in position $(i,j)$ equal to 1 if, and only if, the $i$-th function of $F$ depends on $x_j$. Then, we can provide the following:

\textbf{Definition 1:} A logical map $F : \mathbb{B}^n \times \mathbb{B}^m \rightarrow \mathbb{B}^n$ is $(C,V)$-compliant if, and only if, its incidence matrix $B(F(X,u))$ w.r.t. state and input satisfies the following logical inequalities: $B(F(X,u)) \leq (C|V)$.

### III. DISTRIBUTED MAP SYNTHESIS FOR LOGICAL CONSENSUS

In this section a solution for Problem 1 is presented consisting of an algorithm that generates an optimal distributed logical linear consensus system. More precisely, the algorithm produces a $(C,V)$-compliant linear iteration map $F$ minimizing the number of messages to be exchanged, and the time needed to reach a consensus (a.k.a. rounds).

To achieve this we first need to understand how the agent network can reach a consensus on the value of the $j$-th subterm $l_j$ in the decision system of Eq. 2. Without loss of generality, let us pose $l_j = u_j$ and consider the $j$-th column $V_j$ of the visibility matrix $V$ that also describes the visibility of $l_j$. Then, we need a procedure for finding to which agents the value of input $u_j$ can be propagated. First note that vector $V_j$ contains 1 in all entries corresponding to agents that are able to “see” $u_j$, or, in other words, it specifies which agents are directly reachable from $u_j$. Then, it is useful to consider vectors $C^kV_j$, for $k = 0, 1, \ldots$, each containing 1 in all entries corresponding to agents that are reachable from input $u_j$ after exactly $k$ steps. The $i$-th element of $C^kV_j$ is 1 if, and only if, there exists a path of length $k$ from any agent directly reached by $u_j$ to agent $A_i$.

Recall that, by definition of graph diameter, all agents that are reachable from an initial set of agents are indeed reached in at most $diam(G)$ steps, with $diam(G) \leq n - 1$. Let us denote with $\kappa$ the visibility diameter of the pair $(C, V_j)$ being the number of steps after which the sequence $\{C^kV_j\}$ does not reach new agents. Thus, given a pair $(C, V_j)$, we can conveniently introduce the following reachability matrix $R_j$, assigned with input $u_j$:

\[ R_j = \left( V_j \ CV_j \ C^2V_j \cdots C^{\kappa-1}V_j \right) \]  

whose columns span a subgraph $G_{R_j}(N_{R_j}, E_{R_j})$ of $G$, where $N_{R_j}$ is a node set of all agents that are eventually reachable from input $u_j$, and $E_{R_j}$ is an unspecified edge set, that will be considered during the design phase. Computing the span of $R_j$ is very simple and efficient, and indeed all reachable agents, that are nodes of $N_{R_j}$, are specified by non-null elements of the Boolean vector $I_j = \sum_{k=0}^{\kappa-1} C^kV_j = \sum_{i=0}^{\kappa-1} R_j(:,i)$, that is the logical sum of all columns in $R_j$ and that contains 1 for all agents for which there exists at least one path originating from an agent that is able to measure $u_j$. Then, we can partition the agent network into $N_{R_j} = \{ i \mid I_j(i) = 1 \}$, and $N_{\bar{R}_j} = N \setminus N_{R_j}$, where $N = \{ 1, \ldots, n \}$. In this perspective we can give the following:

\textbf{Definition 2:} A pair $(C, V_j)$ is (completely) reachable if, and only if, the corresponding reachability matrix $R_j(C, V_j)$ spans the entire graph, i.e. $N_{R_j} = N$.

Consider e.g. a network with $n = 5$ agents, and the following pair of communication and visibility matrices:

\[ C = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad V_j = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \]  

Observe that only agent $A_1$ is able to measure $u_j$. The $j$-th reachability matrix $R_j = (V_j \ CV_j \ C^2V_j \ C^3V_j \ C^4V_j)$ is:

\[ R_j = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \]  

Simple computation gives $I_j = (1, 1, 1, 1, 0)^T$ which readily reveals that $N_{R_j} = \{ 1, 2, 3, 4 \}$ is the node set of the reachable subgraph, and $N_{\bar{R}_j} = \{ 5 \}$ is the node set of the unreachable subgraph. Furthermore, all agents in $N_{R_j}$ are reached by input $u_j$ within $\kappa = 3$ steps.

The design phase can obviously concern only the reachable subgraph $G_{R_j}(N_{R_j}, E_{R_j})$ of $G$, and in particular will determine the edge set $E_{R_j}$. Moreover, observe that a non-empty unreachable subgraph $G_{\bar{R}_j}$ in a consensus context is a symptom of the fact that the design problem is not well-posed, and it would require changing sensors’ visibility and locations in order to have a reachable $(C, V_j)$ pair.

Let us suppose, as in the example, that only agent $A_1$ is able to measure $u_j$. Then, a straightforward and yet optimal strategy to allow the information on $u_j$ flowing through the network is obtained if agent $A_1$ communicates its measurement to all its $C$-neighbors, which in turn will communicate it to all their $C$-neighbors without overlapping, and so on. In this way, we have that every agent $A_i$ receives $u_j$ from exactly one minimum-length path originating from
agent \( A_1 \). The vector sequence \( \{C^k V_j\} \) can be exploited to this aim. Indeed, it trivially holds that
\[
C^k V_j = C(C^{k-1} V_j),
\]
meaning that agents reached after \( k \) steps have received the input value from agents that were reached after exactly \( k-1 \) steps. Then, any consecutive sequence of agents that is extracted from non–null elements of vectors in \( \{C^k V_j\} \) are \( (C, V_j) \)–compliant by construction. A consensus strategy would minimize the number of rounds if, and only if, at the \( k \)–th step, all agents specified by non–null elements of vector \( C^k V_j \) receives the value of \( u_j \) from the agents specified by non–null elements of vector \( C^{k-1} V_j \). Nevertheless, to minimize also the number of messages, only agents specified by non–null elements of vector \( C^k V_j \) and that have not been reached yet must receive \( u_j \). If vector \( I_j = \sum_{i=1}^k C_i V_j \) is iteratively updated during the design phase, then the set of all agents that must receive a message on \( u_j \) are specified by non–null elements of vector \( C^k V_j \land -I_j \). By doing this, an optimal pair \( (C^*, V^*_j) \) allowing a consensus to be established over the reachable subgraph is obtained. In the considered example, we have:
\[
C^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad V^* = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.
\]
Observe that is \( C^* = SC \leq C \), where \( S \) is a suitable selection matrix.

This procedure actually gives us only a suggestion on how to construct consensus system that solves Problem 1. Indeed, we can prove in following Theorem 1 that a simple logical linear consensus algorithm of the form
\[
x(t+1) = F_j x(t) + B_j u_j(t),
\]
where \( F_j = C^*, B_j = V^*_j \), and \( x \in \mathbb{R}^n \), allows a consensus to be reached through the entire reachable subgraph. The state \( x \) must be interpreted as the network distributed estimation of the value of the subterm \( l_j \) or \( u_j \). It is indeed a vector and not a matrix, since we are concerned here only with the \( j \)–th input. In our example, we have:
\[
\begin{cases}
  x_1(t+1) = u(t), \\
  x_2(t+1) = x_1(t), \\
  x_3(t+1) = x_1(t), \\
  x_4(t+1) = x_2(t).
\end{cases}
\]
In all cases where a unique generic agent \( A_i \) is directly reachable from input \( u_j \), an optimal communication matrix \( C^* \) for a linear consensus of the form of Eq. 12 can be iteratively found as the incidence matrix of a input–propagating spanning tree having \( A_i \) as the root. Then, an optimal pair \( (C^*, V^*_j) \) can be written as \( C^* = P^T (SC) P \), and \( V^*_j = P^T V_j \), where \( S \) is a selection matrix, and \( P \) is a permutation matrix. Furthermore, \( C^* \) has the following lower–block triangular form:
\[
C^* = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \hat{C}_{1,1} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \hat{C}_{i,\kappa_i} & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 \end{pmatrix},
\]
and \( V^*_j = P^T V_j = (1, 0, \ldots, 0)^T \). It is worth noting that the optimal pair \( (C^*, V^*_j) \) preserves the reachability property of the original pair \( (C, V_j) \). This can be shown by direct computation of the reachability matrix \( R^*_j \), but it is omitted for the sake of space.

We are now ready to consider the more general case with \( \nu, 1 \leq \nu \leq n \) agents that are reachable from input \( u_j \), and let us denote with \( A = \{i_1, \ldots, i_\nu\} \) the index set of such agents. Then, the optimal strategy for propagating input \( u_j \) consists of having each of the other agents receive the input measurement through a path originating from the nearest reachable agent in \( A \). This naturally induces a network partition into \( \nu \) disjoint subgraphs or spanning trees, each directly reached by the input through a different agent. Let us extract \( \nu \) independent vectors \( V_j(i_1), \ldots, V_j(i_\nu) \) from vector \( V_j \) having a 1 in position \( i_b \). Then, the sequences \( \{C^k V_j(i_b)\} \) are to be considered to compute the optimal partition. Let us denote with \( \kappa_i \), for all \( i \in A \) the number \( k \) of steps for the sequence \( \{C^k V_j(i)\} \) to become stationary. Therefore, we have that the visibility diameter of the pair \( (C, V_j) \) is \( \text{vis-diam}(C, V_j) = \max_i \{\kappa_i\} \). Without loss of generality, we can imagine that \( \kappa_1 \geq \kappa_2 \geq \cdots \geq \kappa_\nu \). Therefore, for the generic case, there exist a permutation matrix \( P \) and a selection matrix \( S \) such that an optimal pair \( (C^*, V^*_j) \) can be obtained as \( C^* = P^T (SC) P \), \( V^*_j = P^T V_j \), where
\[
C^* = \text{diag}(C_1, \ldots, C_\nu), V^*_j = (V_{j,1}^T, \ldots, V_{j,\nu}^T)^T,
\]
and where each \( C_i \) and \( V_{j,i} \) have the form of the Eq. 14.
Finally, the actual optimal linear consensus algorithm is obtained choosing \( F_j = P C^* \), and \( B_j = PV_j^* \). Consider e.g. a network of \( n = 5 \) agents and the following pair \( (C, V_j) \) with \( \nu = 2 \):
\[
C = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad V_j = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.
\]
An optimal pair \( (C^*, V^*_j) \) allowing a consensus to be established over the complete graph \( G \) is given by
\[
C^* = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}, \quad V^*_j = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.
\]
5253
Algorithm 1 Optimal Linear Synthesis by Input–Propagation

Inputs: $C, V_j$

Outputs: Minimal pair $(C^*, V^*_j)$, permutation $P$.

1. Set $A \leftarrow \{i \in V_j(i) = 1\}$ \hspace{1em} $\triangleright$ nodes directly reachable from $u_j$
2. Set $I(i) \leftarrow 1$ for all $i \in A$ \hspace{1em} $\triangleright$ nodes reached from $i \in A$
3. Set $N \leftarrow \{1, \ldots, n\} \setminus I$ \hspace{1em} $\triangleright$ nodes not yet reached
4. repeat
5. for all nodes $i \in A$ do
6. Set $Adj(i) \leftarrow C^kV_j(i) \wedge \neg I(i) \wedge N$ \hspace{1em} $\triangleright$ new nodes
7. Set $I(i) \leftarrow I(i) \vee Adj(i)$
8. Set $N \leftarrow N \wedge \neg Adj(i)$
9. Compute $\mathcal{I} \leftarrow \{h : Adj(i)(h) = 1\}$ \hspace{1em} $\triangleright$ index list
10. for all new nodes $h \in \mathcal{I}$ do
11. Set $C(h,:) \leftarrow C(h,:) \wedge Adj(i)^T$ \hspace{1em} $\triangleright$ every new node must communicate with one reach at $k - 1$
12. end for
13. end for
14. until $\exists i \in A | Adj(i) \neq 0$
15. Compute $\kappa_i \leftarrow \text{card}(I(i))$ for all $i \in A$
16. Find $P \mid C^* \leftarrow P^T C$ has $\kappa_1 \geq \cdots \geq \kappa_{\nu}$ \hspace{1em} $\triangleright$ re-order
17. Set $V^*_j \leftarrow P^T V_j$

The corresponding optimal linear consensus algorithm is:

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

Algorithm 1 allows computation of an optimal pair $(C^*, V^*_j)$ as in Eq. 15. Its asymptotic computational complexity is in the worst case $O(n^2)$, where $n$ is the number of agents, and its space complexity in terms of memory required for its execution is $\Omega(n)$. However, its implementation can be very efficient since it is based on Boolean operations on bit strings. Finally, communication complexity of a run of the consensus protocol in terms of the number of rounds is $\Theta(\text{vis-diam}(C, V_j))$.

To conclude, we need to prove that a so–built logical consensus system does indeed solve Problem 1. Hence, for the general case with $\nu \geq 1$ agents that are reachable from input $u_j$, we can state the following result (the proof is omitted for space limitation):

**Theorem 1 (Global Stability of Linear Consensus):**

A logical linear consensus system of the form $x(t+1) = C^* x(t) + V^*_j u_j(t)$, where $C^*$ and $V^*_j$ are obtained as in Eq. 15 from a reachable pair $(C, V_j)$, converges to a unique network agreement given by $\mathbf{1}_n u_j$ in at most $\text{vis-diam}(C, V_j)$ rounds.

IV. SENSOR FAILURE AND ROBUST MAP DESIGN

In Problem 2 we consider the fact that sensor failure may lead a logical linear consensus system to incorrect global decisions, such as raising false alarms in an intrusion detection application. Consider e.g. the consensus algorithm of Eq. 17 and assume that agent $A_2$ outputs 1 instead of 0 when the input is $u_j = 0$. This will break the network into two disagreeing parts. Within such a scenario, we can require that every agent except $A_2$ will continue to work properly and eventually reach the correct consensus.

To find a solution for the robust design problem is more complex and it imposes the use of more conservative rules for propagating the information through the network. For space limitation, we report here only the synthesis procedure in Algorithm 2 and give the following intuition of the solution. Suppose that a maximum number of $\gamma \in N$ faulty agents have to be tolerated. The key to solve such a problem is in redundancy of input measurement and communication. Intuitively, a minimum number of $r$ of such sensors must be able to measure the $j$–th input $u_j$ and/or confirm any transmitted data $x$ on $u_j$. Then, redundant minimum–length paths are to be found such that information on $u_j$ can robustly flow through the network. Algorithm 2 can indeed generate robust logical nonlinear consensus systems, based on the communication graph $C$ and on the visibility matrix $V$.

V. APPLICATION TO INTRUSION DETECTION

Consider an indoor environment $W$, with a number $n$ of agents or observers $A_i$ whose task is to detect and locate possible intruders in $W$. We assume that agents have sensors with star-shaped visibility regions that define a partition of the environment $W_i$, $i = 1, \ldots, m$ (hence, $\cup_{i=1}^m W_i = W$ and $\bigcap_{i=1}^m W_i = \emptyset$). The presence or the absence of an intruder in region $W_i$ can be seen as an input $u_j$ to the following system of $p = m$ logical decisions: $y_i(t) = u_i(t)$, $i = 1, \ldots, m$, that each agent is required to estimate. However, agents are able to detect the presence of intruders only within their visibility areas, which is described by a visibility matrix $V \in \mathbb{B}^{n \times m}$, with $V_{ij} = 1$ if, and only if, an intruder in region $W_i$ can be seen by agent $A_j$. Moreover, let $X \in \mathbb{B}^{n \times m}$ denote the alarm state of the
system: $X_{i,j} = 1$ if agent $A_i$ reports an alarm about the presence of an intruder in region $W_j$. The alarm can be set because an intruder is actually detected by the agent itself, or because of communications with neighboring observers. Indeed, agents have communication devices that allows them to share alarm states with all other agents that are within line-of-sight or nearby. In this context, we aim at designing a distributed update rule of the form $X(t+1) = F(X(t), u(t))$, s.t. agents can achieve the same state value ($X_{i,j} = X_{k,j}$ for all $i, k$ and $j$). In other terms, at consensus, each column of $X$ should have either all zeros or all ones, depending on the corresponding column of $1_n f(u) = 1_n u$.

Consider first applying Algorithm 1 that 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 $C$–neighbors was set at time $t$. The visibility diameter of this pair $(C, V)$ is $2$, which will correspond to the maximum number of steps before consensus is reached. Fig. 1 shows snapshots from a typical run of this linear consensus algorithm 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 (see Fig. 2). A more conservative mechanism can be obtained by applying Algorithm 2, with $\gamma = 1$, that generates a nonlinear rule requiring that agent $A_i$ sets an alarm regarding $W_j$ at time $t+1$ if at least two neighboring sensors having visibility on $W_j$ are in alarm at time $t$, or if it sees an intruder (through $u$). By means of this second system, false alarms raised by at most $\gamma = 1$ misbehaving agents are correctly handled (see again Fig. 2).

VI. CONCLUSION

In this work we introduced a novel consensus mechanism where agents of a network are able to share logical values. We proposed two algorithms producing optimal logical consensus systems and applied them to a distributed IDS.

VII. ACKNOWLEDGMENT

This work has been partially supported by the EC Project CHAT (FP7-IST-2008-224428), by EC NoE HYCON (FP6-IST-2004-511368), and by Research Project 2007 funded by Cassa di Risparmio di Livorno, Lucca e Pisa.

REFERENCES

[14] ——, “Itérations sur des ensembles finis convergence d’automates cellulaires contractants.”