An Optimal Graph-Search Method for Secure State Estimation ⋆

Xusheng Luo [a], Miroslav Pajic [b], Michael M. Zavlanos [a]

[a] *Department of Mechanical Engineering and Materials Science, Duke University, Durham, NC 27708, U.S.A*

[b] *Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, U.S.A*

Abstract

The growing complexity of modern Cyber-Physical Systems (CPS) and the frequent communication between their components make them vulnerable to malicious attacks. As a result, secure state estimation is a critical requirement for the control of these systems. Many existing secure state estimation methods suffer from combinatorial complexity which grows with the number of states and sensors in the system. This complexity can be mitigated using optimization-based methods that relax the original state estimation problem, although at the cost of optimality as these methods often identify attack-free sensors as attacked. In this paper, we propose a new optimal graph-search algorithm to correctly identify malicious attacks and to securely estimate the states even in large-scale CPS modeled as linear time-invariant systems. The graph consists of layers, each one containing two nodes capturing a truth assignment of any given sensor, and directed edges connecting adjacent layers only. Then, our algorithm searches the layers of this graph incrementally, favoring directions at higher layers with more attack-free assignments, while actively managing a repository of nodes to be expanded at later iterations. The proposed search bias and the ability to revisit nodes in the repository and self-correct, allow our graph-search algorithm to reach the optimal assignment faster and tackle larger problems. We show that our algorithm is complete and optimal provided that process and measurement noises do not dominate the attack signal. Moreover, we provide numerical simulations that demonstrate the ability of our algorithm to correctly identify attacked sensors and securely reconstruct the state. Our simulations show that our method outperforms existing algorithms both in terms of optimality and execution time.

Key words: cyber-physical systems; system security; state estimation; graph-search algorithm

1 Introduction

Cyber-Physical Systems (CPS) are networked systems consisting of embedded physical components, such as sensors and actuators, and computational components, such as controllers. Recently, CPS have been successfully utilized in large-scale applications, including power network control, industrial manufacturing processes, and traffic control. However, the growing complexity of CPS and the frequent communication between components make them vulnerable to malicious attacks. Such attacks often manipulate the state of the system by injecting faulty data through compromised sensors, leading to undesirable feedback control signals. Recently, cyber-attacks have been responsible for some incidents of safety-critical automobiles (Koscher et al. 2010, Greenberg 2015, Shoukry et al. 2013) and UAVs (Javaid et al. 2017), and even worse, catastrophic losses of large-scale systems (Slay & Miller 2007, Chen & Abu-Nimeh 2011). Therefore, developing attack-resilient methods for secure state estimation in CPS has recently gained significant attention (Lee 2008, Cardenas et al. 2008).

In this paper, we consider CPS modeled as linear time-invariant systems, where a subset of sensors is subject to malicious attacks represented as attack vectors added to the measurements. Our goal is to detect the attacked sensors fast and use the attack-free sensors to accurately estimate the state. In this context, secure state estimation is closely related to robust control (Pasqualetti et al. 2011, Manandhar et al. 2014), where the control design is subject to process and measurement noise, modeled as

⋆ This work is supported in part by the ONR under agreements #N00014-18-1-2374 and #N00014-17-1-2504, the AFOSR under award #FA9550-19-1-0169, as well as the NSF under grant CNS-1652544. This paper was not presented at any IFAC meeting. Corresponding author Xusheng Luo, xusheng.luo@duke.edu

Email addresses: xusheng.luo@duke.edu (Xusheng Luo), miroslav.pajic@duke.edu (Miroslav Pajic), michael.zavlanos@duke.edu (Michael M. Zavlanos).

an unknown disturbance that is bounded or follows some probability distribution. Nevertheless, such assumptions on the noise restrict the general application of robust control methods for secure state estimation, since it is difficult to predict the attacker's attack strategy. Similarly, fault tolerant control methods (Blanke et al. 2006, Teixeira et al. 2015) focus on internal faults with known failure modes and statistical properties, rather than arbitrary adversarial attacks. Secure state estimation under specific attack signals has been investigated in (Teixeira et al. 2010, Sundaram et al. 2010, Teixeira et al. 2012, Miao et al. 2013, Mo, Hespanha & Sinopoli 2014, Hendrickx et al. 2014, Mo, Chabukswar & Sinopoli 2014).

Compared to the literature discussed above, we do not impose any assumptions on the type of the attack signal. We assume that the number of attacked sensors is smaller than an upper bound, which is necessary to ensure observability of the attacked system that is needed to reconstruct the state. Under this assumption, we propose a new optimal graph search-based algorithm to correctly identify malicious attacks even in large-scale CPS and securely estimate their state, when the power of attack signals exceeds a certatin threshold. The graph consists of layers, each one containing two nodes capturing a truth assignment of any given sensor, and directed edges connecting adjacent layers only. Then, our algorithm searches the layers of this graph incrementally, favoring directions with more attack-free assignments and higher layer, while actively managing a repository of nodes whose expansion are intentionally delayed. The combination of search bias, intentionally delayed expansion and the ability to self-correct allow our graph-search algorithm to reach the optimal assignment fast and tackle larger problems. Assuming that process and measurement noise does not dominate the attack signal, we show that our algorithm is complete and optimal meaning that it will find a feasible attack assignment, if one exists, which does not incorrectly identify any attack-free sensor as attacked. Finally, numerical simulations show that our method outperforms existing algorithms both in terms of optimality and execution time.

Most closely related to the work proposed here are the methods in (Fawzi et al. 2014, Pajic et al. 2014, 2015, 2017, Chong et al. 2015, Shoukry et al. 2017, Mishra et al. 2017, Shoukry et al. 2018), which exploit the measurement within a finite-length time window to conduct state estimation. Specifically, (Fawzi et al. 2014) considers discrete-time LTI systems without noise and provides necessary and sufficient conditions under which the state of the system can be reconstructed when a subset of the sensors are under attack. The idea is to formulate the secure state estimation problem as an ℓ_0 minimization problem that is computationally expensive, and then relax it into an ℓ_1/ℓ_r problem that can be efficiently solved using convex optimization, which mitigates the combinatorial complexity of the methods in (Pasqualetti et al. 2013, Yong et al. 2015, Lee et al. 2015). However, ℓ_0 and ℓ_1/ℓ_r optimization are not always equivalent, thus their relaxation can result in incorrect estimates. The work in (Pajic et al. 2014) extends this method to LTI systems where process and measurement noises are considered in the presence of malicious attacks, and formulates the ℓ_0 optimization problem as a mixed integer linear program (MILP). However, solving MILPs is NP-hard, so this method can not be used for very large problems. Analytic bounds on the state-estimation error for the proposed ℓ_0 state estimator and its convex ℓ_1 relaxation in the presence of noise are derived in (Pajic et al. 2015, 2017), where it is shown that using relaxation results in inaccurate estimation. The work in (Chong et al. 2015) provides similar necessary and sufficient conditions for continuous-time LTI systems, and proposes two methods to estimate the state involving the observability Gramian and the Luenberger observer. However, both methods are computationally expensive. An alternative approach based on Satisfiability Modulo Theory (SMT) is proposed in (Shoukry et al. 2017, Mishra et al. 2017, Shoukry et al. 2018), that formulates the secure state estimation problem as a satisfiability problem subject to Boolean constraints and convex constraints over real state variables. The proposed iterative algorithm combines SMT solvers to obtain a possible attack assignment for the sensors with convex optimization methods to check whether this assignment is valid given the dynamical system equations. Due to the formulation as a feasibility problem, the solution is not guaranteed to be optimal even in the absence of process and measurement noise. Compared to the literature discussed above, our graph-search method is provably optimal, meaning it identifies the true attack assignment and does not incorrectly identify attack-free sensors as attacked. Moreover, numerical experiments show that our method compares favorably to existing methods in terms of execution time. This is due to the proposed search bias that favors directions at higher layers with more attack-free assignments and the ability of our algorithm to self-correct by managing a repository of nodes that can be expanded at later iterations if needed.

The rest of the paper is organized as follows. Section 2 provides the problem formulation. In Section 3, we present the proposed graph-search algorithm for secure state estimation, and examine its completeness, optimality, and complexity in Section 4. Finally, comparative numerical simulations are shown in Section 5, while Section 6 concludes the paper.

2 Problem Formulation

2.1 Linear Dynamical Systems under Attack

Consider the linear time-invariant dynamical system:

$$\begin{aligned} \mathbf{x}(t+1) &= A\mathbf{x}(t) + B\mathbf{u}(t) + \mathbf{v}(t), \\ \mathbf{y}(t) &= C\mathbf{x}(t) + \mathbf{e}(t) + \mathbf{w}(t). \end{aligned} \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n, \mathbf{u}(t) \in \mathbb{R}^m$, and $\mathbf{y}(t) \in \mathbb{R}^p$ denote the state vector, control vector, and measurement vector for p sensors at time instant t, respectively; A, B, C are system matrices with appropriate dimensions; $\mathbf{v}(t)$ and $\mathbf{w}(t)$ represent the process noise and measurement noise at time t; and $\mathbf{e}(t) \in \mathbb{R}^p$ is an attack vector so that if the i-th element $\mathbf{e}_i(t)$ of $\mathbf{e}(t)$ is non-zero then sensor i is under attack, and is attack-free otherwise. We assume that the set of sensors that the attacker has access to does not change over time. Moreover, let $|\text{supp}(\mathbf{e}(t))|$ denote the number of attacked sensors, where $\text{supp}(\mathbf{e}(t)) \subseteq \{1, ..., p\}$ denotes the support of the vector $\mathbf{e}(t) \in \mathbb{R}^p$, that is the set of indices that correspond to non-zero elements in $\mathbf{e}(t)$, $|\cdot|$ denotes the cardinality of a set.

In this paper, we do not consider the case where the actuators are under attack, thus, all the control inputs are known and we can subtract their effect from the dynamical equations due to linearity. Therefore, for simplicity, we set the matrix B to be zero. Given T measurements $\mathbf{y}(t-T+1), \ldots, \mathbf{y}(t)$ that are subject to attack vectors $\mathbf{e}(t-T+1), \ldots, \mathbf{e}(t)$, we can express them as a function of the states $\mathbf{x}(t-T+1)$ as:

$$\mathbf{Y}_{t,T}(t) = \mathcal{O}_T\, \mathbf{x}(t-T+1) + \mathbf{e}_{t,T}(t) + \mathbf{w}_{t,T}(t), \quad (2)$$

where

$$\begin{aligned}
\mathbf{Y}_{t,T}(t) &= [\mathbf{y}^\intercal(t-T+1),\, \mathbf{y}^\intercal(t-T+2),\, \cdots,\, \mathbf{y}^\intercal(t)]^\intercal, \\
\mathcal{O}_T &= \left[C^\intercal,\, A^\intercal C^\intercal,\, \cdots,\, (A^\intercal)^{T-1} C^\intercal\right]^\intercal, \\
\mathbf{e}_{t,T}(t) &= [\mathbf{e}^\intercal(t-T+1),\, \mathbf{e}^\intercal(t-T+2),\, \cdots,\, \mathbf{e}^\intercal(t)]^\intercal, \\
\mathbf{w}_{t,T}(t) &= \begin{bmatrix}
\mathbf{w}(t-T+1) \\
C\mathbf{v}(t-T+1) + \mathbf{w}(t-T+2) \\
C\sum_{i=1}^{2} A^{2-i}\mathbf{v}(t-T+i) + \mathbf{w}(t-T+3) \\
\cdots \\
C\sum_{i=1}^{T-1} A^{T-1-i}\mathbf{v}(t-T+i) + \mathbf{w}(t)
\end{bmatrix}.
\end{aligned}$$

The term $\mathbf{e}_{t,T}(t)$ denotes the attack vector, and with a slight abuse of notation, $\mathbf{w}_{t,T}(t)$ represents noise vectors, including both the process and measurement noise. If the system is noiseless and attack-free, equation (2) becomes

$$\mathbf{Y}_{t,T}(t) = \mathcal{O}_T\, \mathbf{x}(t-T+1). \quad (3)$$

As is shown in (Fawzi et al. 2014), for noiseless systems that are under attack, we can reconstruct the state from T measurements when s sensors are under attack if and only if, $\forall\, \mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, $|\text{supp}(C\mathbf{x}) \cup \text{supp}(CA\mathbf{x}) \cup \ldots \cup \text{supp}(CA^{T-1}\mathbf{x})| > 2s$. Therefore, there is an upper limit on the number of attacked sensors, denoted by $\bar{s}$, beyond which states can not be correctly estimated. This limit depends on the system matrices A and C. In this paper, we assume the number of attacked sensors is less than or equal to $\bar{s}$, i.e., $|\text{supp}(\mathbf{e}(t))| \leq \bar{s}$, and $\bar{s}$ is assumed to be known a priori, a common assumption used in relevant work. Furthermore, we assume the system in (1) is $2\bar{s}$-sparse observable, which means the system is still observable after any $2\bar{s}$ sensors are removed. As shown in (Fawzi et al. 2014, Shoukry et al. 2017), it is impossible to correctly estimate the state if $\lceil p/2 \rceil$ or more sensors are attacked. Thus, $\bar{s} \leq \lceil p/2 \rceil - 1$. Besides these assumptions, we do not impose additional constraints on the attack vector, which can be arbitrary and unbounded.

Moreover, let $\mathcal{I} \subseteq \{1, \ldots, p\}$ be a subset of sensors and define by $\mathbf{y}(t)|_{\mathcal{I}}$ the vector composed of elements of $\mathbf{y}(t)$ indexed by the set $\mathcal{I}$. Then, we can define the set of measurements corresponding to sensors in the set $\mathcal{I}$ as

$$\mathbf{Y}_{t,T}(t)|_{\mathcal{I}} = [\mathbf{y}^\intercal(t-T+1)|_{\mathcal{I}}, \mathbf{y}^\intercal(t-T+2)|_{\mathcal{I}}, \cdots, \mathbf{y}^\intercal(t)|_{\mathcal{I}}]^\intercal.$$

Considering only measurements from sensors indexed by the set $\mathcal{I}$, (2) can be rewritten as

$$\mathbf{Y}_{t,T}(t)|_{\mathcal{I}} = \mathcal{O}_T|_{\mathcal{I}}\, \mathbf{x}(t-T+1) + \mathbf{e}_{t,T}(t)|_{\mathcal{I}} + \mathbf{w}_{t,T}(t)|_{\mathcal{I}}.$$

For notational simplicity, we use $\mathbf{Y}_{\mathcal{I}}, \mathcal{O}_{\mathcal{I}}, \mathbf{e}_{\mathcal{I}}, \mathbf{w}_{\mathcal{I}}$ to denote $\mathbf{Y}_{t,T}(t)|_{\mathcal{I}}$, $\mathcal{O}_T|_{\mathcal{I}}, \mathbf{e}_{t,T}(t)|_{\mathcal{I}}, \mathbf{w}_{t,T}(t)|_{\mathcal{I}}$ and $\mathbf{x}$ to denote $\mathbf{x}(t-T+1)$ when they are clear from the context. Moreover, when $\mathcal{I} = \{i\}$, for $i \in \{1, \ldots, p\}$, is a singleton, we use the notations $\mathbf{Y}_i, \mathcal{O}_i, \mathbf{e}_i, \mathbf{w}_i$. Finally, we assume that the noise term $\mathbf{w}_{t,T}(t)$ is upper bounded. This is a reasonable assumption since, otherwise, it is impossible to estimate the state. Specifically, we assume that $\|\mathbf{w}_i\| \leq \overline{w}_i$, for $\forall t \in \mathbb{N}, T \in \{1, \ldots, n\}, i \in \{1, \ldots, p\}$, where $\|\cdot\|$ is ℓ_2-norm of a vector. Similarly, $\|\mathbf{w}_{\mathcal{I}}\|^2 \leq \overline{w}_{\mathcal{I}}^2 = \sum_{i \in \mathcal{I}} \overline{w}_i^2$ and $\overline{w}^2 = \sum_{i=1}^{p} \overline{w}_i^2$.

2.2 Secure State Estimation

Let $\mathbf{b} = (b_1, \ldots, b_p)$ be a vector of binary variables such that $b_i = 1$ if sensor i is under attack and $b_i = 0$, otherwise. Let $\mathbf{x}_0$ and $\mathbf{b}_0$ denote the true state and the true attack assignment at time $t - T + 1$, respectively. Assuming the $|\text{supp}(\mathbf{b}_0)| \leq \bar{s}$, our goal is to find the attack assignment $\mathbf{b}^*$ that satisfies $\mathbf{b}^* = \mathbf{b}_0$, and use the attack-free sensors in $\mathbf{b}^*$ to reconstruct the state. Specifically, we formulate the following problem.

Problem 1 *Consider the linear dynamical system in* (1) *that is under attack. Determine the optimal state vector and attack assignment* $(\mathbf{x}^*, \mathbf{b}^*) \in \mathbb{R}^n \times \mathbb{B}^p$ *that solve the optimization problem*

$$\min_{(\mathbf{x},\mathbf{b}) \in \mathbb{R}^n \times \mathbb{B}^p} \quad |supp(\mathbf{b})| \quad (4)$$

$$s.t. \quad \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|_2 \leq \overline{w}_{\mathcal{I}} + \sqrt{\epsilon}, \quad (4a)$$

$$|supp(\mathbf{b})| \leq \bar{s}, \quad (4b)$$

where $\bar{s}$ *is the maximum allowable number of attacked sensors in order to reconstruct states,* $\mathcal{I} = \{1, \ldots, p\} \setminus supp(\mathbf{b})$*, and* ϵ *is the acceptable numerical accuracy in the solution specified by the user, which serves as the stopping criterion of numerical iterations.*

As shown in (Shoukry et al. 2017), if the noise and solution accuracy are zero, i.e., if $\overline{w}_i = 0$ and $\epsilon = 0$, and if the system is $2\bar{s}$-sparse observable, then any assignment $\mathbf{b}$ with $\overline{\text{supp}(\mathbf{b})} \subseteq \overline{\text{supp}(\mathbf{b}_0)}$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ is a feasible assignment, where $\overline{\text{supp}(\mathbf{b})}$ is the complement of the set $\text{supp}(\mathbf{b})$, i.e., $\overline{\text{supp}(\mathbf{b})} = \{1, \ldots, n\} \setminus \text{supp}(\mathbf{b})$. In words, a feasible solution to Problem 1 correctly identifies all attacked sensors, but it can also incorrectly treat attack-free sensors as attacked. As shown in (Shoukry & Tabuada 2016), if the noise and solution accuracy are zero, then the solution to Problem 1 correctly identifies the true attack assignment, i.e., it satisfies $\mathbf{b}^* = \mathbf{b}_0$. However, in the presence of noise and non-zero solution accuracy, a feasible solution to Problem 1 can incorrectly identify attacked sensors as attack-free if the attack signal is undetectable meaning that its effect is hidden by the noise and solution accuracy. [1] Similarly, if the noise is relatively large or the solution accuracy is low, then, some attack-free sensors may be incorrectly treated as being under attack. Nevertheless, if the attack vector is strong enough so that it can not be hidden by noise and solution accuracy, then it is reasonable to expect that the solution to Problem 1 coincides with the true attack. This discussion on attack detection in the presence of noise is also supported by the theoretical analysis in (Pajic et al. 2017). The following proposition quantifies this discussion in the current problem formulation.

However, before we show this result, we provide some definitions. Let $\mathbf{x}'$ be any reachable state of system (1). Then, using the noiseless and attack-free model (3), we can get T measurements $\mathbf{Y}' = \mathcal{O}\mathbf{x}'$. Let $\tilde{\mathbf{x}}$ denote the solution of the problem $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}' - \mathcal{O}\mathbf{x}\|$ for the given $\mathbf{x}'$. Then, we can define the solution accuracy ϵ^* as

$$\epsilon^* = \inf\{\epsilon \,|\, \|\mathbf{Y}' - \mathcal{O}\tilde{\mathbf{x}}\| \leq \sqrt{\epsilon}, \forall \mathbf{x}'\}. \tag{5}$$

In other words, ϵ^* is a uniform lower bound on ϵ so that for any $\mathbf{x}'$ the solution $\tilde{\mathbf{x}}$ of the problem $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}' - \mathcal{O}\mathbf{x}\|$ satisfies $\|\mathbf{Y}' - \mathcal{O}\tilde{\mathbf{x}}\| \leq \sqrt{\epsilon^*}$. If $\epsilon < \epsilon^*$, then this constraint can not be satisfied. Now, consider a set $\mathcal{I}$ containing only attack-free sensors. Then, the solution $\bar{\mathbf{x}}$ to the problem $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|$ also satisfies $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\bar{\mathbf{x}}\| \leq \sqrt{\epsilon^*}$. The reason is that for $\tilde{\mathbf{x}}$, we have $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\tilde{\mathbf{x}}\| \leq \|\mathbf{Y}' - \mathcal{O}\tilde{\mathbf{x}}\|$, and $\|\mathbf{Y}' - \mathcal{O}\tilde{\mathbf{x}}\| \leq \sqrt{\epsilon^*}$ by definition of ϵ^*. Therefore, $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\tilde{\mathbf{x}}\| \leq \sqrt{\epsilon^*}$. Moreover, since $\bar{\mathbf{x}}$ is the minimizer of $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|$, we have $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\bar{\mathbf{x}}\| \leq \|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\tilde{\mathbf{x}}\|$. We conclude that $\|\mathbf{Y}'_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\bar{\mathbf{x}}\| \leq \sqrt{\epsilon^*}$.

Proposition 2 *Let the linear system in* (1) *be* $2\bar{s}$*-sparse observable and let the number of attacked sensors be less than or equal to* $\bar{s}$*. Moreover, let* $\epsilon = \epsilon^*$*, i.e., the lower bound defined in* (5)*. If the attack signal satisfies*

$$\|\mathbf{e}_i\| > \left(\frac{2}{\sqrt{1-\Delta_s}}\right)\overline{w} + \frac{\sqrt{\epsilon}}{\sqrt{1-\Delta_s}}, \tag{6}$$

where

$$\Delta_s = \max_{\substack{\Gamma\subset\mathcal{I}\subset\{1,\ldots,p\} \\ |\Gamma|\leq\bar{s}, |\mathcal{I}|\geq p-\bar{s}}} \lambda_{\max}\left\{\left(\sum_{i\in\Gamma}\mathcal{O}_i^T\mathcal{O}_i\right)\left(\sum_{i\in\mathcal{I}}\mathcal{O}_i^T\mathcal{O}_i\right)^{-1}\right\},$$

and $\lambda_{\max}(\cdot)$ *is the maximal eigenvalue of a matrix, then* $\mathbf{b}_0$ *is the unique optimal solution of Problem 1.*

PROOF. First, we show that if there is an attacked sensor in the index set $\mathcal{I}$ with $|\mathcal{I}| \geq p - \bar{s}$, when (6) is satisfied, $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\| \leq \overline{w}_{\mathcal{I}} + \sqrt{\epsilon}$ does not hold anymore. It is shown in Theorem IV.3 in (Shoukry et al. 2017) that if there is an attacked sensor in the index set $\mathcal{I}$ with $|\mathcal{I}| \geq p - \bar{s}$, for which the attack signal satisfies (6), then the following two inequalities hold,

$$\|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|^2 \geq \left(\|(I - \mathcal{O}_{\mathcal{I}}\mathcal{O}_{\mathcal{I}}^+)\mathbf{e}_{\mathcal{I}}\| - \|(I - \mathcal{O}_{\mathcal{I}}\mathcal{O}_{\mathcal{I}}^+)\mathbf{w}_{\mathcal{I}}\|\right)^2, \tag{7}$$

$$\|(I - \mathcal{O}_{\mathcal{I}}\mathcal{O}_{\mathcal{I}}^+)\mathbf{e}_{\mathcal{I}}\| - \|(I - \mathcal{O}_{\mathcal{I}}\mathcal{O}_{\mathcal{I}}^+)\mathbf{w}_{\mathcal{I}}\| > \overline{w}_{\mathcal{I}} + \sqrt{\epsilon}, \tag{8}$$

where $\mathcal{O}_{\mathcal{I}}^+ = (\mathcal{O}_{\mathcal{I}}^T\mathcal{O}_{\mathcal{I}})^{-1}\mathcal{O}_{\mathcal{I}}^T$, I is the identity matrix, and (7) corresponds to (13) and (8) is the second to last inequality in (14) in (Shoukry et al. 2017). It's worth noting that the right-hand side of constraint (4a) takes the form $\overline{w}_{\mathcal{I}} + \epsilon$ in equation (8) in (Shoukry et al. 2017). However, this reformulation does not invalidate the theoretical results in (Shoukry et al. 2017), since the derivation of inequalities (7) and (8) is irrelevant to any form of constraint (4a). Thus, combining (7) and (8), we have that $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\| \leq \overline{w}_{\mathcal{I}} + \sqrt{\epsilon}$ does not hold anymore. Therefore, the set $\mathcal{I}$ should only include attack-free sensors. This implies that when (6) is satisfied, any feasible solution $\mathbf{b}$ of Problem 1 needs to correctly identify all attacked sensors, i.e., $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$, and therefore $|\text{supp}(\mathbf{b}_0)| \leq |\text{supp}(\mathbf{b})|$.

In what follows, we show that the converse is also true, i.e., that any assignment $\mathbf{b}$ that satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ is a feasible solution to Problem 1, provided that (6) is satisfied. Then, we can conclude that since $\mathbf{b} = \mathbf{b}_0$ satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ with equality, $\mathbf{b}_0$ is a feasible and in fact the optimal solution to Problem 1. Moreover, since $\mathbf{b}_0$ is a binary vector, there is no other assignment $\mathbf{b}$ that satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ with equality. Therefore, $\mathbf{b}_0$ is the unique optimal solution to Problem 1. Note that if $\epsilon < \epsilon^*$, then it is possible that an assignment $\mathbf{b}$ satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ but does not satisfy the constraint (4a). Therefore, constraint (6) is a necessary condition.

[1] Typically, noise is the main factor that can hide the effect of an attack signal.

To show that any assignment $\mathbf{b}$ that satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$ and $|\text{supp}(\mathbf{b})| \leq \bar{s}$ is a feasible solution to Problem 1, note first that if all sensors in the set $\mathcal{I} = \overline{\text{supp}(\mathbf{b})}$ are attack-free, we have that $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}_0 + \mathbf{w}_\mathcal{I}$ for the true $\mathbf{x}_0$. Moreover, using the true state $\mathbf{x}_0$, we can generate measurements $\mathbf{Y}'_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}_0$ according to the noiseless and attack-free model in (3). Combining those two equations we get $\mathbf{Y}_\mathcal{I} = \mathbf{Y}'_\mathcal{I} + \mathbf{w}_\mathcal{I}$. By definition of ϵ^*,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x} - \mathbf{w}_\mathcal{I}\| = \min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}'_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \sqrt{\epsilon^*}, \quad (9)$$

where $\mathbf{Y}_\mathcal{I} = \mathbf{Y}'_\mathcal{I} + \mathbf{w}_\mathcal{I}$. Since by the triangle inequality $\|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x} - \mathbf{w}_\mathcal{I}\| + \|\mathbf{w}_\mathcal{I}\|$, and $\mathbf{w}_\mathcal{I}$ is irrelevant to the minimization over $\mathbf{x}$, we have

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x} - \mathbf{w}_\mathcal{I}\| + \|\mathbf{w}_\mathcal{I}\|. \quad (10)$$

Finally, combining (9) and (10), we get $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \|\mathbf{w}_\mathcal{I}\| + \sqrt{\epsilon^*}$. Thus, providing (6) holds, if the assignment $\mathbf{b}$ satisfies $\text{supp}(\mathbf{b}_0) \subseteq \text{supp}(\mathbf{b})$, then $\mathcal{I} = \overline{\text{supp}(\mathbf{b})}$ satisfies constraint (4a), completing the proof.

3 Graph Search-based Secure State Estimation

In this section, we propose a graph search algorithm that incrementally assigns a truth value to each binary variable in $\mathbf{b}$. Specifically, our algorithm searches for the true attack assignment on a directed graph with $p+1$ levels and 2 nodes on each level; see Fig. 1. The graph is initialized with an artificial root to provide a unique starting point. This root corresponds to level 0. Each level except level 0 captures the truth assignment of one sensor, so that the nodes with values 1 and 0 at this level indicate whether this sensor is under attack or not, respectively. The edges in this graph connect nodes in adjacent levels only, and all edges point towards a higher level. A path starting from level 1 and ending at level p corresponds to a possible attack assignment $\mathbf{b}$. Throughout the rest of this paper, we refer to the partial and full assignment when part of or all of p sensors are assigned, respectively.

The key idea of the proposed algorithm is to prioritize search along paths in the graph that contain more attack-free sensors. Search along these paths returns sensor assignments with more 0 entries that minimize the objective in (4). To decide whether a partial/full assignment is valid, our algorithm also checks the feasibility of the system of linear equations $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$, as per the constraints in (4a). [2] During the early stages of the search, this system of equations may be underdetermined or square (when $T \cdot |\mathcal{I}| \leq n$), according to the observability matrix $\mathcal{O}_\mathcal{I} \in \mathbb{R}^{T|\mathcal{I}|\times n}$. Therefore, it is possible that

[2] Here, $\mathcal{I}$ only includes sensors that have been assigned and assigned as attack-free.

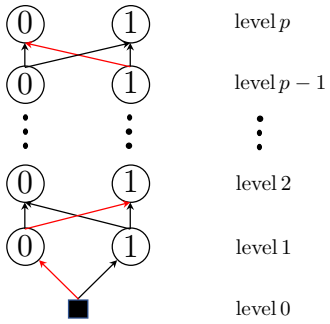


Fig. 1. The red path stands for one possible full assignment $\mathbf{b} = 0, 1, \ldots, 1, 0$.

attacked sensors are incorrectly treated as attack-free. However, as the search progresses, more sensors are assigned 0 values, which reduces the dimension of the kernel space of the system of linear equations $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$, making the search less tolerant to such mistakes. To see this, assume that the observability matrix $\mathcal{O}_\mathcal{I}$ has full rank n and the corresponding partial assignment is correct. Then, all subsequent sensors will also be correctly assigned. This is because, if attacked sensors are incorrectly identified as attack-free, then the system of linear equations $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$ will become inconsistent. On the other hand, if the observability matrix has full rank n and the corresponding partial assignment is wrong, then the search along this path will terminate immediately as soon as the next sensor assigned 0 is added to the system of equations $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$, making it inconsistent. Together with prioritizing search along paths that contain more attack-free sensors to minimize the objective in (4), the proposed algorithm also actively manages a repository of nodes whose exploration is deliberately postponed until there is need to correct a search that early on has made an incorrect sensor assignment. The combination of search bias and the ability to self-correct allow our graph-search algorithm to identify the true attack assignment fast.

Before presenting our algorithm, we introduce the information that a node needs to store and the resulting order between any two nodes. We first associate with every node a 5-tuple $(\mathit{level}, \mathit{value}, \mathit{parent}, \mathcal{I}, \mathit{residual})$, where: (i) $\mathit{level} = l$ indicates that this node corresponds to the l-th sensor except when $l = 0$ which corresponds to the artificial root; (ii) $\mathit{value} = 1$ means that the l-th sensor is under attack and $\mathit{value} = 0$ means that the l-th sensor is attack-free; (iii) parent denotes the parent node of the l-th sensor obtained from level $l-1$; (iv) $\mathcal{I}$ is the set of levels with value 0 (attack-free sensors) from level 1 to level l. The set $\mathcal{I}$ and the value level together provide complete knowledge of the truth assignments of the first l sensors, and they also contain information about the objective (4) and the satisfaction of constraint (4b); (v) $\mathit{residual} = 0$ if the inequality $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \overline{w}_\mathcal{I} + \sqrt{\epsilon}$ holds, otherwise $\mathit{residual} = 1$. We can tell from $\mathit{residual}$ whether the current attack assignment of the

Algorithm 1 Graph-Search for Secure State Estimation

Input: measurements $\mathbf{Y}$, observability matrix $\mathcal{O}$, number of sensors p, noise bound $\overline{w}_i$, $i \in \{1, \ldots, p\}$, solution accuracy ϵ
Output: estimated state $\mathbf{x}$, indices of attacked sensor $\mathcal{I}$

1: $node.level \leftarrow 0$; $node.value \leftarrow 1$; $node.parent \leftarrow None$; $node.\mathcal{I} \leftarrow \emptyset$; $node.residual = 0$
2: $frontier \leftarrow \{node\}$; $explored \leftarrow \emptyset$; $repo \leftarrow \emptyset$
3: **while true do**
4: | **if** `Empty`($frontier$) and `Empty`($repo$) **then**
5: | | **return** *failure*
6: | **if** `Empty`($frontier$) **then**
7: | | $frontier$.put($repo$.get())
8: | | $explored \leftarrow \emptyset$
9: | $node \leftarrow frontier$.get()
10: | **if** $node.level = p$ **then**
11: | | **return** $\text{argmin}_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|_2, node.\mathcal{I}$
12: | $explored$.add($node$)
13: | **foreach** $id \in [0, 1]$ **do**
14: | | $child \leftarrow$ `GetChild`$(node, id, \mathbf{Y}, \mathcal{O}, \overline{w}_i, \epsilon)$
15: | | **if** $child.residual = 0$ **then**
16: | | | **if** $child.level - |child.\mathcal{I}| \geq \lceil p/2 \rceil$ **then**
17: | | | | **continue**
18: | | | **if** $child \in frontier$ **or** $explored$ **then**
19: | | | | $repo$.put($child$)
20: | | | | **continue**
21: | | | **else**
22: | | | | $frontier$.put($child$)

first l sensors satisfies constraint (4a). Next, with the help of the 5-tuple, we define an ordering between two nodes v and v' in the graph as follows. We say $v = v'$ if $v.value = v'.value$ and $v.level = v'.level$. That is, two nodes are treated as equivalent if they correspond to the same sensor and have identical Boolean values, as shown in Fig. 1. Moreover, we use the lexicographical order to compare any two nodes. Specifically, we define $v > v'$ if $v.level - |v.\mathcal{I}| < v'.level - |v'.\mathcal{I}|$ or $(v.level - |v.\mathcal{I}| = v'.level - |v'.\mathcal{I}|) \wedge (v.level > v'.level)$. In words, ordering of nodes is first determined by the number of attacked sensors in the paths leading to those nodes, so that fewer attacked sensors in these paths correspond to nodes with higher priority. This is because the goal of the algorithm is to find the true attack assignment and avoid incorrect assignments. On the other hand, if the paths leading to two nodes contain the same number of attacked sensors, ordering is determined by the levels of these nodes; higher level corresponds to higher priority in the ordering. This is because expanding a node at a higher level can advance the algorithm faster to terminate.

The proposed search algorithm, illustrated in Alg. 1, generates the true attack assignment and estimates the state. It starts by initializing the artificial root and the three node sets, *frontier*, *explored*, and *repo* [lines 1-2]. The first set is the priority queue *frontier*, which contains nodes that their parents have been expanded and they themselves are eligible for expansion but have not yet been selected for expansion. The priority is based on the lexicographical order between nodes introduced before. The second set is called *explored*, which stores nodes that have been expanded. Storing those nodes avoids repeated search. Finally, the third set is a priority queue *repo* which is a repository for nodes for which (i) an equivalent node is in queue *frontier*, or (ii) an equivalent node has been expanded and is in the set *explored*. [3]

After initialization, Alg. 1 repeatedly performs the following steps. First, it checks whether the queues *frontier* and *repo* that contain nodes to be expanded are empty [lines 4-8]. If both are empty, then the algorithm has searched the whole graph and it was not able to find a solution. If only *frontier* is empty, it is possible that the solution is in the set *repo*, since *repo* also contains nodes to be explored next. In this case, the algorithm picks the node from *repo* with the highest priority and puts it in the set *frontier*, meanwhile clearing the set *explored*. One can think of selecting a node from *repo* as reinitializing the search. Once a node is included in the set *frontier*, Alg. 1 proceeds to check and expand it. If this node is at the highest level [lines 10-11], the algorithm terminates with a full assignment that satisfies the constraints (4a) and (4b); otherwise, this node is expanded [lines 13-22]. Note that each node has two possible children nodes. We discuss how to generate a child node later in Alg. 2.

Given a generated child node, Alg. 1 will discard this node if it violates the constraint (4a). Otherwise, it checks whether the number of attacked sensors in the path leading to this child is larger than or equal to $\lceil p/2 \rceil$. If this is true, then this node is also discarded due to the violation of (4b). If not, then the algorithm searches the set *frontier* and *explored* for equivalent nodes, i.e., nodes with same values for *value* and *level*. If such nodes exist in *frontier* or *explored*, then the child node is added to *repo* and its expansion is delayed [lines 18-20]. This node is not discarded because, compared to its equivalent nodes that share the same *level* and *value*, it is associated with a different partial assignment. Later we show that a node with a specific partial assignment exists in *frontier* at most once throughout the whole search. Furthermore, we place the child node into *frontier* if no equivalent node exists in these two sets [line 22]. A simple case study is shown in Example 3.

[3] The sets *frontier* and *explored* exist in a typical search algorithm, where a graph is divided into three disjoint parts: the nodes already explored, the nodes to be explored next (stored in *frontier*) and the remaining unexplored nodes. The set *frontier* separates the explored nodes from the remaining nodes (Russell & Norvig 2016). But in this paper, we add another set *repo*. *frontier* and *repo* together separate the explored nodes from unexplored nodes since part of nodes are added to *repo* due to the definition of "equivalence" of nodes, which will be discussed later.

Algorithm 2 `GetChild` $(parentnode, id, \mathbf{Y}, \mathcal{O}, \overline{w}_i, \epsilon)$

1: $child.level = parentnode.level + 1$
2: $child.value = id$
3: $child.parent = parentnode$
4: **if** $id = 0$ **then**
5: $\quad child.\mathcal{I} = parentnode.\mathcal{I} \cup \{child.level\}$
6: $\quad$ **if** $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|_2 < \overline{w}_{\mathcal{I}} + \sqrt{\epsilon}$ **then**
7: $\quad\quad child.residual = 0$
8: $\quad$ **else**
9: $\quad\quad child.residual = 1$
10: **else**
11: $\quad child.\mathcal{I} = parentnode.\mathcal{I}$
12: $\quad child.residual = parentnode.residual$

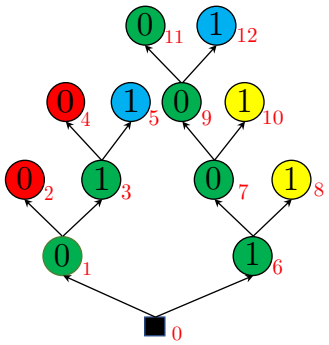


Fig. 2. Graphical illustration of Example 3

Alg. 2 describes how to generate a child node. The key requirement is that the child node is always one level higher than its parent [lines 1]. If $child.value = 0$, that is, the corresponding sensor is attack-free, then the algorithm checks whether sensors in the updated set $\mathcal{I}$ of attack-free sensors are consistent meaning that the inequality (4a) holds [lines 4-9]. If not, it means the current partial assignment is wrong in that it incorrectly treats attacked sensors as attack-free. If $child.value = 1$, i.e., the corresponding senor is under attack, then the set $\mathcal{I}$ remains unchanged and the child node inherits the value *residual* from its parent [lines 10-12]. This is because the algorithm considers this sensor attacked, so it is not used to reconstruct the state.

Example 3 *Assume that a system is 3-sparse observable, which means the observability matrix of each sensor has full rank and we can use only one sensor to reconstruct states. There are 4 sensors in this system and only the first sensor is under attack. The tree structure in Fig. 2 shows how Alg. 1 searches the graph in Fig. 1. We differentiate equivalent nodes with different assignment explicitly, thus each path corresponds to one partial/full assignment. Nodes are numbered with indices to their right. Blue, green and yellow nodes are those that are in sets* frontier, explored *and* repo *upon termination, respectively; and red nodes are discarded due to the violation of constraints* (4a) *or* (4b)*. The node numbered 11, with tuple* $(4, 0, 9, \{2, 3, 4\}, 0)$*, is the first one to reach the highest level, and the path leading to it provides the optimal solution, which is* $1, 0, 0, 0$*. Thus, only the first sensor is under attack. Table 1 shows the node indices in three sets at the beginning of each iteration.*

Table 1
The evolution of three different sets in Example 3 as the Alg. 1 proceeds, where "–" means no nodes exist, and nodes in *frontier* and *repo* are sorted by priority.

iteration	*frontier*	*explored*	*repo*
1	0	–	–
2	1,6	0	–
3	3,6	0,1	–
4	6,5	0,1,3	–
5	7,5	0,1,3,6	8
6	9,5	0,1,3,6,7	10,8
7	11,12,5	0,1,3,6,7,9	10,8
8	12,5	0,1,3,6,7,9,11	10,8

4 Completeness, Optimality, and Complexity

In the following results, it will help to view two nodes with the same *value* and *level* but distinct $\mathcal{I}$ as different. The reason is that, while such nodes are really equivalent, they are treated differently by the algorithm in the sense that a node is added to *repo* instead of being discarded when another node with same *value* and *level* is in queue *frontier*. In what follows, we show that Alg. 1 is complete and optimal and analyze its complexity.

4.1 Completeness and Optimality

Theorem 4 *Let the attacked linear dynamical system in* (1) *be* $2\bar{s}$*-sparse observable and* $\epsilon = \epsilon^*$*. Assume also that the number of attacked sensors is less than* $\bar{s}$ *and that each attack signal satisfies* $\|\mathbf{e}_i\| > \left(\frac{2}{\sqrt{1-\Delta_s}}\right)\overline{w} + \frac{\sqrt{\epsilon}}{\sqrt{1-\Delta_s}}$*. Then Alg. 1 is complete that is, if there exists a solution* $(\mathbf{x}, \mathbf{b})$ *that satisfies the constraints* (4a) *and* (4b) *in Problem 1, then Alg. 1 will find it.*

PROOF. We show that Alg. 1 terminates in a finite number of iterations at which point the queue *frontier* contains a feasible node that satisfies constraints (4a) and (4b). A detailed proof of this result is provided in Appendix A.

Next, we show that Alg. 1 is optimal, meaning that Alg. 1 can identify true attacks and makes no mistakes in treating attack-free sensors as attacked.

Theorem 5 *Let the attacked linear dynamical system in* (1) *be* $2\bar{s}$*-sparse observable and* $\epsilon = \epsilon^*$*. Assume also that the number of attacked sensors is less than* $\bar{s}$ *and that each attack signal satisfies* $\|\mathbf{e}_i\| > \left(\frac{2}{\sqrt{1-\Delta_s}}\right)\overline{w} + \frac{\sqrt{\epsilon}}{\sqrt{1-\Delta_s}}$*. Then, the solution of Problem 1 constructed by Alg. 1 is*

optimal meaning that $\mathbf{b}^* = \mathbf{b}_0$, *where* $\mathbf{b}_0$ *represents the true attack assignment.*

PROOF. We show that it is not possible that a suboptimal path can be formed before the optimal one. This is because Alg. 1 always prioritizes search in directions of possible attack-free assignments and in the case of mistakes, it uses the nodes in the set *repo* to take corrective actions. A detailed proof of this result is provided in Appendix B.

4.2 Complexity Analysis

Given a LTI system in (1) and a true attack assignment, in this section we discuss the complexity of Alg. 1 in terms of the number of iterations until termination. Note that at each iteration, a node is selected from the queues *frontier* or *repo*, thus, we can focus on the number of nodes that have been expanded before termination.

To simplify the analysis of complexity, consider an "ideal" model without noise and solution accuracy. Then the constraint (4a) becomes

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\| = 0. \tag{11}$$

The following theorem provides an upper bound on the number of iterations taken to find the true assignment.

Theorem 6 *Let the attacked linear dynamical system in* (1) *be* $2\bar{s}$*-sparse observable. Assume also that the true number of sensors that are under attack is* s*. Let* $S = p - 2\bar{s}$*, where* p *is the number of sensors and* $\bar{s}$ *is the maximum allowable number of sensors under attack. Then, without considering noise and solution accuracy, Alg. 1 takes at most* $N_{upper} = \sum_{i=1}^{S} \binom{s}{i}\binom{\bar{s}+S-s}{S-i}(\bar{s}+S) + p$ *iterations to find the true assignment, where* $\binom{s}{i} = 0$ *if* $s < i$.

PROOF. Alg. 1 achieves its worst perfomance if the first s sensors are the ones that are under attack. This is because, in this case, the attacked sensors can be treated as attack-free when the system of equations $\mathbf{Y}_{\mathcal{I}} = \mathcal{O}_{\mathcal{I}}\mathbf{x}$ (where $\mathcal{O}_{\mathcal{I}} \in \mathbb{R}^{T\cdot|\mathcal{I}|\times n}$) is underdetermined or square, i.e., when $T \cdot |\mathcal{I}| \leq n$, and Alg. 1 is biased towards expanding attack-free sensors first. Recalling the graph in Fig. 1, this worst scenario corresponds to the case where the first s levels are associated with sensors that are under attack. In what follows, we focus on this worst-case attack scenario where the first s sensors are under attack.

Since the system is $2\bar{s}$-sparse observable, any observability matrix $\mathcal{O}_{\mathcal{I}}$ corresponding to $\mathcal{I}$ with $|\mathcal{I}| \geq p - 2\bar{s}$ has full rank n. Assume first that $s \geq S = p - 2\bar{s}$. When $|\mathcal{I}| < S$, the rank of the observability matrix $\mathcal{O}_{\mathcal{I}}$ can be smaller than n. We further assume that the system of linear equations $\mathbf{Y}_{\mathcal{I}} = \mathcal{O}_{\mathcal{I}}\mathbf{x}$ is feasible, which could occur when it is underdetermined or square. Then, the algorithm can assign 0's to S sensors out of the first s sensors. But after the S-th sensor that is assigned 0, if the next sensor is also assigned 0, then the system of linear equations corresponding to those $S+1$ sensors, which are treated as attack-free, becomes inconsistent since it incorrectly treats attacked sensors as attack-free. Therefore, the algorithm can only assign 1 to all sensors past the S sensors that have been assigned 0 until there are $\bar{s}$ 1's in this path. Hence, in this case, the whole searched path has $\bar{s}+S$ nodes. Similarly, if $S-1$ nodes are assigned 0 and $s-(S-1)$ nodes are assigned 1 for the first s sensors, Alg. 1 can assign 0 to one more sensor and 1 to another $\bar{s}-[s-(S-1)]$ sensors. Following this logic, we get that if i nodes are assigned 0 among the first s sensors, where $1 \leq i \leq S$, Alg. 1 can assign 0 to another $S-i$ sensors and 1 to $\bar{s}-(s-i)$ sensors. We conclude that our algorithm can only explore $i+(S-i)+(s-i)+[\bar{s}-(s-i)] = \bar{s}+S$ nodes on one path if this path is not feasible. Each assignment to the first $S+\bar{s}$ sensors corresponds to one path. In the worst-case scenario considered here, the path leading to the optimal solution has the lowest priority in the sense that a node at a specific level $l \leq S$ on an infeasible path has higher priority than the node with same *value* and *level* on the optimal path. Hence, all infeasible paths will be searched before the algorithm terminates, and we denote by Π the set of these paths.[4] Therefore, the search algorithm searches at most $\sum_{i=1}^{S}\binom{s}{i}\binom{\bar{s}+S-s}{S-i}(\bar{s}+S)$ nodes corresponding to infeasible paths. When no nodes are assigned 0 at the first s levels, Alg. 1 will assign 0's to the remaining $p-s$ sensors, which is exactly the number of nodes needed to be required corresponding to the worst-case attack scenario. Because Alg. 1 explores one node per iteration, we get the upper bound on the number of iterations taken to find the true assignment.

When $s < S$, the situation is less complex since the path corresponding to the optimal solution is inside Π, that is, the path where the first s nodes are assigned 1 is inside Π, so there is no need to explore the graph until the first S nodes are assigned 1, completing the proof.

Remark 7 *(Ideal worst case): The worst-case complexity in Theorem 6 assumes that if* $\mathbf{Y}_{\mathcal{I}} = \mathcal{O}_{\mathcal{I}}\mathbf{x}$ *is underdetermined or square, i.e., if* $T \cdot |\mathcal{I}| \leq n$*, then feasible solutions exist. However, in practice, it is possible that no solution exists. Furthermore, as the index set* $\mathcal{I}$ *grows, the system of linear equations becomes overdetermined* ($T\cdot|\mathcal{I}| > n$)*, which is almost always inconsistent and thus has no solutions. In this case, the wrong assignment will*

[4] Note that infeasible paths can only be partially searched. As shown in Theorem 5, it is not possible that infeasible paths can be fully searched before the optimal path.

terminate much sooner, as demonstrated in Section 5.2. Therefore, we refer to the worst case in the proof as the "ideal" worst case. Whether the ideal worst case might occur is application-dependent, and relies on the dynamical system, the number of measurements T and injected attack signals.

5 Experimental Results

In this section, we present several test cases for values of p and n, implemented using Python 3.6.3 on a computer with 2.3 GHz Intel Core i5 and 8G RAM, that illustrate the correctness and efficiency of the proposed algorithm for large-scale estimation problems. To validate our method, we compare with the Mixed Integer Quadratically Constrained Programming (MIQCP) method in (Winston & Goldberg 2004) and with the solver IMHOTEP-SMT. [5] The formulation of the MIQCP problem takes the form

$$\min_{(\mathbf{x},\mathbf{b})\in\mathbb{R}^n\times\mathbb{B}^p} \sum_{i=1}^{p} b_i \tag{12}$$
$$s.t. \quad \|\mathbf{Y}_i - \mathcal{O}_i\mathbf{x}\| \leq M b_i + \overline{w}_i + \sqrt{\epsilon},\ \forall i \in \{1,\dots,p\}\,.$$

Since $M \in \mathbb{R}$ is a very big number, the constraints in (12) are called Big-M constraints, which are used to model the binary activation/deactivation of constraints defined over real decision variables. The performance of MIQCP is sensitive to the value of M. Given different M, the resulting attack assignment may vary significantly. Note that we can get $\mathbf{x}$ and $\mathbf{b}$ simultaneously by solving the MIQCP, but in our experiments, using the commercial solver Gurobi(Gurobi Optimization 2018), we observed that selecting a very large M can provide a feasible assignment $\mathbf{b}$ but a bad state $\mathbf{x}$. This is because using large M places more emphasis on the optimization over the binary variables $\mathbf{b}$ rather than the real variables $\mathbf{x}$. To overcome this issue, we implement MIQCP in two steps. First, we solve the MIQCP problem (12) for large M to obtain a feasible assignment $\mathbf{b}$ and then use this assignment to solve the unconstrained least square problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}}\mathbf{x}\|,$$

for the state $\mathbf{x}$, where the set $\mathcal{I}$ is the set of attack-free sensors predicted by the solution of the MIQCP. On the other hand, IMHOTEP-SMT is mainly designed to find a feasible solution, and it can be used to find the optimal solution in the noiseless case if executed repeatedly for different values of $\bar{s}$ by performing a binary search over $\bar{s}$, checking feasibility of the system of equations in (4a), decreasing $\bar{s}$ if these equations are feasible, and repeating this process until the constraint (4a) is violated or until $\bar{s} = 0$. However, IMHOTEP-SMT needs to search all possible combinations of truth assignments to all sensors to ensure that a problem is infeasible, thus selecting a feasible but less conservative $\bar{s}$ for IMHOTEP-SMT is not easy. Hence, we run IMHOTEP-SMT once until the first feasible solution is found.

[5] IMHOTEP-SMT source code in Matlab can be found at `http://nesl.github.io/Imhotep-smt/index.html`.

Table 2
Properties of LTI systems

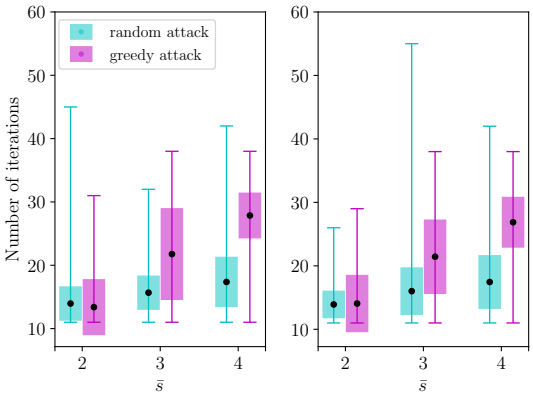
$\bar{s}$	Δ_s				$\frac{2}{\sqrt{1-\Delta_s}}$	N_{upper}
	mean	std	min	max		
2	0.985	0.024	0.894	0.999994	16.341	226
3	0.986	0.021	0.910	0.999890	17.202	248
4	0.999	0.002	0.994	0.999994	55.541	94

We randomly construct sparse matrices A and C for various parameters p and n with entries in the interval $[0, 1]$. Furthermore, the initial state and the set of the attacked sensors and corresponding attack signals are also randomly generated, as in (Fawzi et al. 2014, Shoukry et al. 2017, Pajic et al. 2015, 2017). Specifically, given an LTI system, we adopt two schemes to select the set of attacked sensors (Park et al. 2017). The first scheme attacks the first s sensors in the graph in Fig. 1, which we refer to as the greedy attack, and the second scheme randomly selects s sensors to attack. For the random attack scheme, we randomly generate m different true attack assignments, and attack signals that satisfy (6) over $T = n$ time instants for each attack assignment. Specifically, the attack signals are generated by first sampling a vector which follows a standard normal distribution, then normalizing it, and finally multiplying by a number which captures the magnitude, as in (Shoukry et al. 2017). For the greedy scheme, we randomly generate m such attack signals. The solution accuracy is set to $\epsilon = 10^{-5}$. The quantity M in MIQCP is set to be 10^8. We monitor the execution time and relative estimation error of each method, defined as $\frac{\|\mathbf{x}_0 - \mathbf{x}^*\|}{\|\mathbf{x}_0\|}$. Note that $\mathbf{x}_0$ is the true state, while with a slight abuse of notation, $\mathbf{x}^*$ is the output of each method.

5.1 Optimality and Complexity

In this section, we validate the optimality and complexity of the proposed algorithm. For this, we need to get the maximum allowable number of attacked sensors $\bar{s}$ by increasing $\bar{s}$ from 1 to $\lceil p/2 \rceil - 1$ and checking the observability matrix after excluding any $2\bar{s}$ sensors. Since this problem is combinatorial, we consider small systems with $n = p = 10$ for which $\bar{s}$ and the resulting Δ_s in (6) are computationally tractable. Then, we validate optimality by generating attack signals that satisfy (6), and complexity by comparing the number of iterations taken with the upper bound.

Specifically, we consider both the noiseless and noisy cases. For each case, we randomly generate three sets



(a) Noiseless case (b) Noisy case

Fig. 3. Secure state estimations for small systems

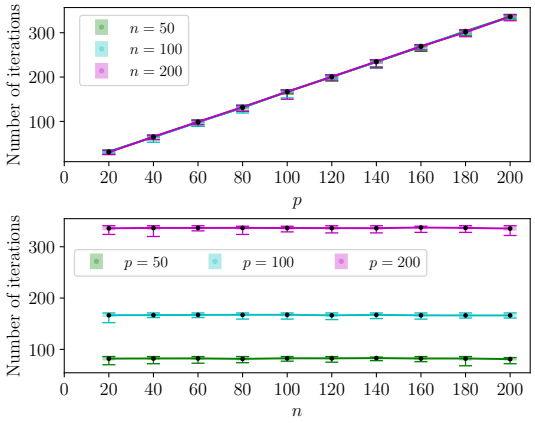


Fig. 4. Number of iterations versus scales of systems

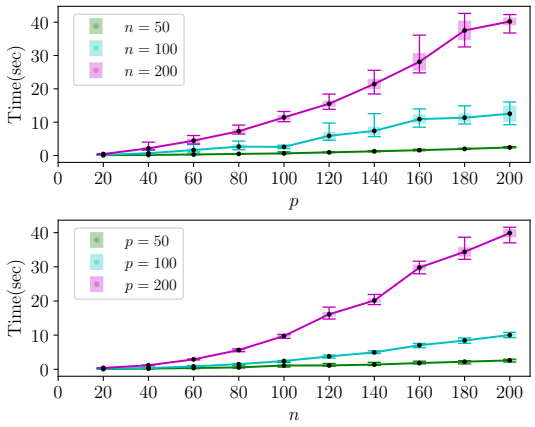


Fig. 5. Runtimes versus scales of systems

of $2\bar{s}$-sparse-observable and small-scale systems, 50 systems per set, with $p = n = 10$, such that each set corresponds to systems with $\bar{s} = 2, 3, 4$, respectively. Then, for each system, we set the number of attacked sensors s to be $\bar{s}$. Table 2 records the statistics, including the mean, standard deviation, and minimum and maximum value of Δ_s with respect to the maximum allowable number of attacked sensors $\bar{s}$, computed over the 50 systems per set. The column $\frac{2}{\sqrt{1-\Delta_s}}$ in Table 2 shows the mean value and represents the average power of the attack signal compared to that of the noise. The last column shows the theoretical upper bound on the maximum number of iterations that Alg. 1 will take given p, $\bar{s}$ and $s = \bar{s}$. For the noiseless and noisy cases, we compare the performance of Alg. 1 for different values of $\bar{s}$ and for the two different attack schemes. From our simulations, we observe that the true attack assignment can be identified every time for both the noiseless and noisy cases. Furthermore, in Fig. 3 we report the statistics on the number of iterations taken over 1000 trials per case. It can be seen that, on average, identifying the true attack assignment for the greedy attack requires more iterations compared to the random attack. For small-scale systems, we found that when the index set $\mathcal{I}$ contains only one sensor that is also under attack, it is often the case that the square matrix $\mathcal{O}_{\mathcal{I}}$, with $T = n$, is non-singular and thus the system of linear equations $\mathbf{Y}_{\mathcal{I}} = \mathcal{O}_{\mathcal{I}}\mathbf{x}$ is consistent. In this case, the sensor will be mistakenly identified as attack-free. Thus, more iterations are required, which verifies the complexity analysis for the ideal worst case presented in Section 4.2. Notably, the number of iterations of Alg. 1 is lower than the upper bound regardless of the presence or absence of the noise and the attack scheme. This is because when the index set $\mathcal{I}$ includes more than one sensors and at least one of those sensors is under attack, the system of linear equations is overdetermined and inconsistent, so that the wrong assignment will terminate immediately.

Next, we demonstrate that the upper bound on the number of iterations given in Thm. 6 is independent of the number of states. For this, we assume no process or measurement noise. Specifically, we first fix n at values 50, 100, 200, respectively, and then gradually increase the number of sensors up to 200. Then, we fix p at values 50, 100, 200, respectively, and gradually increase the number of states up to 200. For each combination of p and n, we randomly generate 5 LTI systems. For each LTI system, the number of attacked sensors is chosen to equal 30% of the total number of sensors, and the corresponding attack assignment is generated according to the random attack scheme with $m = 5$. The number of iterations of Alg. 1 averaged over 25 trials are shown in Fig. 4. It can be seen that the number of iterations increases as p grows but it is not affected by n. We show the execution time in Fig. 5. Although the number of iterations of Alg. 1 for the same p does not change significantly with n, the runtime of the algorithm for the same p increases as n increases. This is due to the time required to solve the minimization problem in line 6, Agl. 2.

5.2 Secure State Estimation without Noise

In this section, we consider the noiseless scenario where the number of states n and the number of states p are

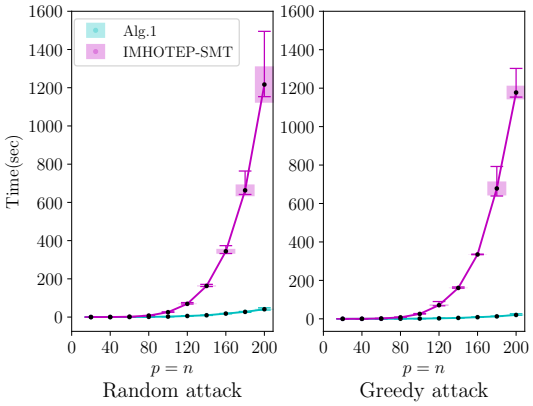


Fig. 6. Runtimes versus scales of systems for the noiseless case

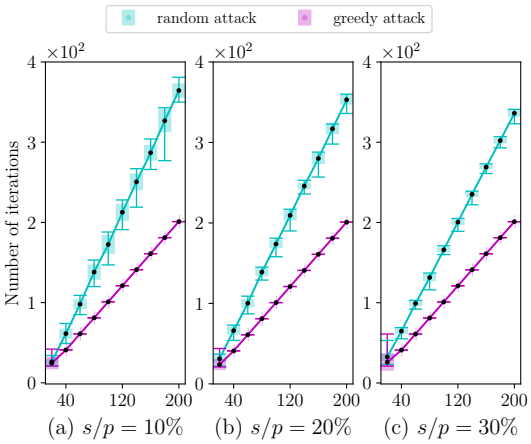


Fig. 7. Number of iterations taken versus system scales

the same, both varying from 20 to 200. For each combination of p and n, we randomly generate 5 LTI systems. For each LTI system, the number of attacked sensors is chosen to equal $10\%, 20\%, 30\%$ of the total number of sensors, respectively, and the corresponding attack support is generated according to the greedy and random attack schemes, respectively, with $m = 5$. The solutions returned by Alg. 1 and IMHOTEP-SMT are identical to the true attack assignment in each trial, and their estimation error is close to 0. However, this is not the case for MIQCP. The statistics of the runtimes averaged over 25 trials for $s/p = 30\%$ are shown in Fig. 6. The results for $s/p = 10\%, 20\%$ are similar to those shown in Fig. 6 and thus, we omit them due to space limitations. When the number of states and sensors are small, execution times are comparable for Alg. 1 and IMHOTEP-SMT. However, as the system scale becomes large, Alg. 1 significantly outperforms IMHOTEP-SMT in terms of runtime. Specifically, for the random attack scheme and $n = p = 200$, Alg. 1 requires $57.7s$, $49.7s$ and $41.0s$ on average to find the truth assignment when $s/p = 10\%, 20\%, 30\%$, respectively, while IMHOTEP-SMT requires $691.1s$, $992.2s$ and $1216.7s$ and these times grow much faster than Alg. 1. Fig. 7 shows the number of iterations required by Alg. 1, which grow almost linearly with p and n. Note that fewer iterations are required in the greedy attack case, contrary to the result in Thm. 6 where the ideal worst case is analyzed. For large-scale systems, we found that when the index set $\mathcal{I}$ contains only one sensor and this sensor is also under attack, it is more likely that the square matrix $\mathcal{O}_{\mathcal{I}}$ is singular so that the system of linear equations $\mathbf{Y}_{\mathcal{I}} = \mathcal{O}_{\mathcal{I}}\mathbf{x}$ is inconsistent. Therefore, the ideal worst case does not take place and the inconsistency causes the wrong partial assignment to terminate early. Although it is computationally almost impossible to calculate the maximum allowable number of attacked sensors $\bar{s}$ and the upper bound N_{upper} for large systems, we here provide the upper bound N_{upper} when $\bar{s} = 1, 2, 3, 97, 98, 99$, respectively, for $p = 200, s/p = 10\%$. Approximately, these bounds are $39801, 3.86 \times 10^6, 2.47 \times 10^8, 1.08 \times 10^{11}, 2.55 \times 10^8, 1.83 \times 10^5$, respectively. Calculating N_{upper} for other $\bar{s}$ is not achievable within reasonable amount of time. Observe that in Fig. 7(a) the maximum number of iterations required by Alg. 1 is lower than 400, which makes our algorithm at least two orders of magnitude faster than the above upper bound.

Table 3
Misidentification ratio of MIQCP for the noiseless case

s/p	20	40	60	80	100	120	140	160	180	200
0.1	0.24	0.0	0.2	0.64	0.24	0.52	0.88	0.52	0.6	0.8
0.2	0.32	0.04	0.24	0.56	0.2	0.6	0.8	0.48	0.48	0.8
0.3	0.44	0.08	0.24	0.56	0.28	0.52	0.8	0.52	0.56	0.72
0.1	0.2	0.0	0.2	0.56	0.24	0.56	0.8	0.44	0.56	0.8
0.2	0.24	0.08	0.24	0.6	0.24	0.52	0.8	0.48	0.52	0.92
0.3	0.6	0.12	0.32	0.44	0.2	0.56	0.8	0.48	0.64	0.84

As for MIQCP, Table 3 shows the misidentification ratio between the number of trials where the truth assignment cannot be identified divided by the total number of 25 trials, where the first 3 rows show the results for the random attack scheme and the last 3 rows for the greedy scheme. It can be seen that the performance of MIQCP is heavily affected by the selection of big M and, in simulation, it behaves poorly in terms of estimation accuracy. Note that, in practice, tuning M does not help recover the true attack assignment, as suggested by extensive numerical simulations we have conducted. Moreover, there is no way to justify the selection of M if the truth assignment is not known beforehand.

Since IMHOTEP-SMT has been already compared to the event-triggered projected gradient descent method (ETPG)(Shoukry & Tabuada 2016) in terms of execution time and relative estimation error and has been shown to exhibit better performance, we did not compare with this method here. Moreover, we did not compare with methods that relax ℓ_0-based formulations to convex problems since they lack correctness guarantees.

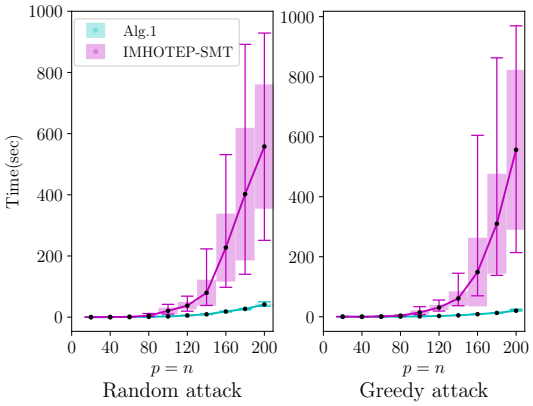


Fig. 8. Runtimes versus scales of systems for the noisy case

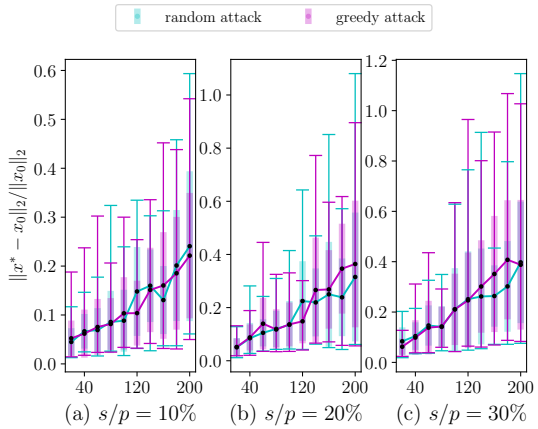


Fig. 9. Relative estimation errors versus scales of systems

5.3 Secure State Estimation with Noise

In this part, we consider the noisy case by following the same simulation procedure as in the noiseless case. Note that it is computationally expensive to calculate the term Δ_s when p and n are large. Instead, we estimate Δ_s by simulating the process and measurement noises subject to truncated normal distributions. Then, we select attack signals with relatively large strengths. Since the selected attack vectors can be much stronger than the noise, which is a strategy that a true attacker would not typically follow, we numerically tune the strength of the attack vector by running a large number of trials and checking if IMHOTEP-SMT is able to correctly identify the truly attacked sensors in all trials. If yes, we reduce the strength of the attack vector and repeat this process until our algorithm fails to identify the true attack assignment in all trials. When this happens, we select values for the attack vector from the last successful trial. Then, Alg. 1 is applied to the same attacked dynamical system and corresponding measurements. It turns out that Alg. 1 and IMHOTEP-SMT can identify the truth assignment, which validates the workaround above when n and p are large.

As in Section 5.2, we monitor the execution time averaged over 25 trials. The results for $s/p = 30\%$ are shown in Fig. 8. The results for $s/p = 10\%, 20\%$ are similar to those shown in Fig. 8 and thus, we omit them due to space limitations. We observe that, Alg. 1 outperforms IMHOTEP-SMT in terms of runtime. Specifically, for $p = n = 200$, Alg. 1 requires $55.1s$, $47.5s$ and $41.3s$ on average to return the solution for $s/p = 10\%, 20\%$ and 30%, respectively, while IMHOTEP-SMT requires $275.4s$, $526.1s$ and $557.7s$ and this runtime grows much faster than Alg. 1, as in the noiseless case. The relative estimation errors of Alg. 1 are shown in Fig. 9. Furthermore, the misidentification ratio of MIQCP is similar to Table 3 and we omit showing it due to space limitations.

5.4 Discussion

Our numerical results for the noiseless and noisy case studies in Sections 5.2 and 5.3, show that our method outperforms IMHOTEP-SMT in terms of runtime. The reason is that, in IMHOTEP-SMT, the optimization solver only checks the correctness of a full truth assignment for all sensors provided by the SMT solver. However, our method also checks partial assignments, which allows a search to terminate early if an assignment is infeasible. Second, in IMHOTEP-SMT, certificates that exploit the geometry of the problem are explicitly generated to serve as heuristics targeting a smaller set of sensors where at least one sensor is attacked, when a full set of truth assignments is not successful. The idea is that the affine half-spaces $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$ corresponding to the attack-free sensors should intersect, and a small set of affine subspaces failing to intersect means there exists at least one attacked sensor that is incorrectly identified as attack-free. In our algorithm, prioritizing search along paths with more attack-free sensors reduces the dimension of the kernel space of the equations $\mathbf{Y}_\mathcal{I} = \mathcal{O}_\mathcal{I}\mathbf{x}$, which makes the search less tolerant to mistakes. Therefore, assigning higher priority to nodes in *frontier* that contain larger numbers of attack-free sensors implicitly acts as a heuristic that shrinks the space to be explored. This allows our method to significantly outperform competing methods in practice, even for larger problems. What's more, although IMHOTEP-SMT can detect the true assignments in the simulation, (Shoukry et al. 2017) doesn't provide the optimality guarantee.

6 Conclusion

In this paper, we proposed a new optimal graph-search algorithm to correctly identify malicious attacks and to securely estimate the states in large-scale CPS modeled as linear time-invariant systems. The graph consists of levels, each one containing two nodes capturing a truth assignment of any given sensor, and directed edges connecting adjacent layers only. Then, our algorithm searches the levels of this graph incrementally, favoring

directions with attack-free assignments, while actively managing a repository of partially explored paths with early truth assignments, that can be further explored in the future. The combination of search bias and the ability to self-correct allow our graph-search algorithm to reach the optimal assignment fast. We showed that our algorithm is complete and optimal provided that the attack signal is not dominated by process and measurement noise. Moreover, numerical simulations show that our method outperforms existing algorithms both in terms of optimality and execution time.

References

Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M. & Schröder, J. (2006), *Diagnosis and fault-tolerant control*, Vol. 2, Springer.

Cardenas, A. A., Amin, S. & Sastry, S. (2008), Secure control: Towards survivable cyber-physical systems, *in* 'Distributed Computing Systems Workshops, 2008. ICDCS'08. 28th International Conference on', IEEE, pp. 495–500.

Chen, T. & Abu-Nimeh, S. (2011), 'Lessons from stuxnet', *Computer* **44**(4), 91–93.

Chong, M. S., Wakaiki, M. & Hespanha, J. P. (2015), Observability of linear systems under adversarial attacks, *in* 'American Control Conference (ACC), 2015', IEEE, pp. 2439–2444.

Fawzi, H., Tabuada, P. & Diggavi, S. (2014), 'Secure estimation and control for cyber-physical systems under adversarial attacks', *IEEE Transactions on Automatic Control* **59**(6), 1454–1467.

Greenberg, A. (2015), 'Hackers remotely kill a jeep on the highway'.
URL: *http://www.wired.com/2015/07/hackers-remotely- kill-jeep-highway/*

Gurobi Optimization, L. (2018), 'Gurobi optimizer reference manual'.
URL: *http://www.gurobi.com*

Hendrickx, J. M., Johansson, K. H., Jungers, R. M., Sandberg, H. & Sou, K. C. (2014), 'Efficient computations of a security index for false data attacks in power networks', *IEEE Transactions on Automatic Control* **59**(12), 3194–3208.

Javaid, A. Y., Jahan, F. & Sun, W. (2017), 'Analysis of global positioning system-based attacks and a novel global positioning system spoofing detection/mitigation algorithm for unmanned aerial vehicle simulation', *Simulation* **93**(5), 427–441.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H. et al. (2010), Experimental security analysis of a modern automobile, *in* '2010 IEEE Symposium on Security and Privacy', IEEE, pp. 447–462.

Lee, C., Shim, H. & Eun, Y. (2015), Secure and robust state estimation under sensor attacks, measurement noises, and process disturbances: Observer-based combinatorial approach, *in* 'Control Conference (ECC), 2015 European', IEEE, pp. 1872–1877.

Lee, E. A. (2008), Cyber physical systems: Design challenges, *in* '11th IEEE Symposium on Object Oriented Real-Time Distributed Computing (ISORC)', IEEE, pp. 363–369.

Manandhar, K., Cao, X., Hu, F. & Liu, Y. (2014), Combating false data injection attacks in smart grid using kalman filter, *in* 'Computing, Networking and Communications (ICNC), 2014 International Conference on', IEEE, pp. 16–20.

Miao, F., Pajic, M. & Pappas, G. J. (2013), Stochastic game approach for replay attack detection, *in* '52nd IEEE conference on decision and control', IEEE, pp. 1854–1859.

Mishra, S., Shoukry, Y., Karamchandani, N., Diggavi, S. N. & Tabuada, P. (2017), 'Secure state estimation against sensor attacks in the presence of noise', *IEEE Transactions on Control of Network Systems* **4**(1), 49–59.

Mo, Y., Chabukswar, R. & Sinopoli, B. (2014), 'Detecting integrity attacks on scada systems', *IEEE Transactions on Control Systems Technology* **22**(4), 1396–1407.

Mo, Y., Hespanha, J. P. & Sinopoli, B. (2014), 'Resilient detection in the presence of integrity attacks', *IEEE Transactions on Signal Processing* **62**(1), 31–43.

Pajic, M., Lee, I. & Pappas, G. J. (2017), 'Attack-resilient state estimation for noisy dynamical systems', *IEEE Transactions on Control of Network Systems* **4**(1), 82–92.

Pajic, M., Tabuada, P., Lee, I. & Pappas, G. J. (2015), Attack-resilient state estimation in the presence of noise, *in* '2015 54th IEEE Conference on Decision and Control (CDC)', IEEE, pp. 5827–5832.

Pajic, M., Weimer, J., Bezzo, N., Tabuada, P., Sokolsky, O., Lee, I. & Pappas, G. J. (2014), Robustness of attack-resilient state estimators, *in* 'ICCPS'14: ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014)', IEEE Computer Society, pp. 163–174.

Park, J., Ivanov, R., Weimer, J., Pajic, M., Son, S. H. & Lee, I. (2017), 'Security of cyber-physical systems in the presence of transient sensor faults', *ACM Transactions on Cyber-Physical Systems* **1**(3), 15.

Pasqualetti, F., Dörfler, F. & Bullo, F. (2011), Cyber-physical attacks in power networks: Models, fundamental limitations and monitor design, *in* 'Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on', IEEE, pp. 2195–2201.

Pasqualetti, F., Dörfler, F. & Bullo, F. (2013), 'Attack detection and identification in cyber-physical systems', *IEEE Transactions on Automatic Control* **58**(11), 2715–2729.

Russell, S. J. & Norvig, P. (2016), *Artificial intelligence: a modern approach*, Malaysia; Pearson Education Limited,.

Shoukry, Y., Chong, M., Wakaiki, M., Nuzzo, P.,

Sangiovanni-Vincentelli, A., Seshia, S. A., Hespanha, J. P. & Tabuada, P. (2018), 'Smt-based observer design for cyber-physical systems under sensor attacks', *ACM Transactions on Cyber-Physical Systems* **2**(1), 5.

Shoukry, Y., Martin, P., Tabuada, P. & Srivastava, M. (2013), Non-invasive spoofing attacks for anti-lock braking systems, *in* 'International Workshop on Cryptographic Hardware and Embedded Systems', Springer, pp. 55–72.

Shoukry, Y., Nuzzo, P., Puggelli, A., Sangiovanni-Vincentelli, A. L., Seshia, S. A. & Tabuada, P. (2017), 'Secure state estimation for cyber-physical systems under sensor attacks: A satisfiability modulo theory approach', *IEEE Transactions on Automatic Control* **62**(10), 4917–4932.

Shoukry, Y. & Tabuada, P. (2016), 'Event-triggered state observers for sparse sensor noise/attacks', *IEEE Transactions on Automatic Control* **61**(8), 2079–2091.

Slay, J. & Miller, M. (2007), Lessons learned from the maroochy water breach, *in* 'International Conference on Critical Infrastructure Protection', Springer, pp. 73–82.

Sundaram, S., Pajic, M., Hadjicostis, C. N., Mangharam, R. & Pappas, G. J. (2010), The wireless control network: Monitoring for malicious behavior, *in* '49th IEEE Conference on Decision and Control (CDC)', IEEE, pp. 5979–5984.

Teixeira, A., Amin, S., Sandberg, H., Johansson, K. H. & Sastry, S. S. (2010), Cyber security analysis of state estimators in electric power systems, *in* '49th IEEE Conference on Decision and Control (CDC). Atlanta, GA. DEC 15-17, 2010', pp. 5991–5998.

Teixeira, A., Pérez, D., Sandberg, H. & Johansson, K. H. (2012), Attack models and scenarios for networked control systems, *in* 'Proceedings of the 1st international conference on High Confidence Networked Systems', ACM, pp. 55–64.

Teixeira, A., Shames, I., Sandberg, H. & Johansson, K. H. (2015), 'A secure control framework for resource-limited adversaries', *Automatica* **51**, 135–148.

Winston, W. L. & Goldberg, J. B. (2004), *Operations research: applications and algorithms*, Vol. 3, Thomson Brooks/Cole Belmont.

Yong, S. Z., Zhu, M. & Frazzoli, E. (2015), Resilient state estimation against switching attacks on stochastic cyber-physical systems, *in* 'Decision and Control (CDC), 2015 IEEE 54th Annual Conference on', IEEE, pp. 5162–5169.

A PROOF OF THEOREM 4

The proof of completeness of Alg. 1 can be divided into two steps. First, we show by contradiction, that each node that is associated with a specific partial truth assignment in the set $node.\mathcal{I}$, can be visited at most once. Then, to show completeness we show that the algorithm can terminate in a finite number of iterations and the queue *frontier* is not empty.

First, assume that a node associated with a given partial assignment, is visited more than once. Since each partial assignment corresponds to a distinct path from the root to that node, we get that the parent of the node that is associated with this partial assignment except for the last value, is also visited more than once. Repeating this argument, we have that the root is visited more than once, which is a contradiction. Thus, every node associated with a given partial assignment is visited at most once, which means that every node is visited no more times than the maximum number of truth assignments starting from the root and ending at that node. For a node that is at level l, the maximum number for such truth assignments is 2^l, i.e., the number of truth assignments of the first l sensors.

Next, since the nodes and edges in the graph are finite, the previous result implies that the algorithm will terminate in a finite number of iterations. Furthermore, the queue *frontier* can not be empty when the algorithm terminates. This is because we do not discard any node unless its partial assignment $\mathcal{I}$ is invalid, that is, the corresponding residual violates $\min_{\mathbf{x}\in\mathbb{R}^n}\|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \overline{w}_\mathcal{I} + \sqrt{\epsilon}$. The reason is that we have shown that if all sensors in $\mathcal{I}$ are attack-free, then we have that $\min_{\mathbf{x}\in\mathbb{R}^n}\|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \overline{w}_\mathcal{I} + \sqrt{\epsilon}$ when $\epsilon = \epsilon^*$ and (6) is satisfied. Therefore, violation of this inequality means that at least one attacked sensor is incorrectly treated as attack-free. Therefore, only the node with an invalid assignment is discarded. If there exists a solution, the algorithm will search the right node eventually. Once it terminates, the output is a feasible solution. Recall that a child node can be added to the queue *frontier* or *repo* if $\min_{\mathbf{x}\in\mathbb{R}^n}\|\mathbf{Y}_\mathcal{I} - \mathcal{O}_\mathcal{I}\mathbf{x}\| \leq \overline{w}_\mathcal{I} + \sqrt{\epsilon}$. Hence, the constraint (4a) is met. Moreover, the constraint (4b) is checked each time before a child node can be added to *frontier* or *repo* [Alg. 1, line 16]. Thus, each node existing in *frontier* must satisfy (4b), completing the proof.

B PROOF OF THEOREM 5

We prove the optimality of Alg. 1 using contradiction. Suppose that the algorithm returns a feasible but suboptimal assignment, which is equivalent to say that the suboptimal path π, is formed before the optimal π^*. This is because the algorithm terminates as soon as the first feasible path is found. Since the paths π and π^* are different, they share identical attack assignments up to a level and then differ in their assignments beyond that level (this level could also be the root). Consider the time when the last shared node v_{l-1}, at level $l-1$, is selected to be expanded. Its child node, denoted by v_l^0 with superscript 0 for $value = 0$, is on the optimal path π^* and v_l^1 is on the suboptimal path π. This is because when (6) is satisfied, a feasible path can only treat attack-free sensors as attacked by mistake, but the optimal path does not contain such mistakes.

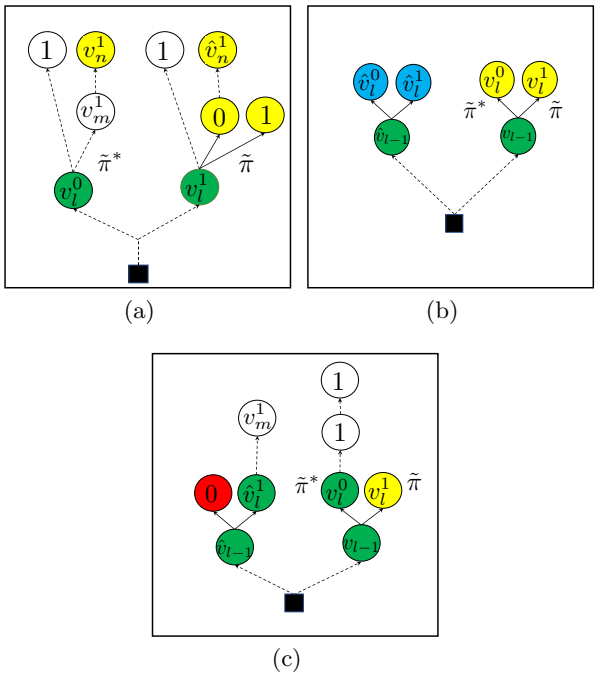


Fig. B.1. (a) Case (i), the dashed arrow represents that the tail node is on the path starting from the head node, with intermediate nodes omitted. v_n^1 enters *repo* because another node with $value = 1$ is in *frontier* or *explored*, shown as a node with number 1 inside on its left. (b) Case (iii), nodes $\hat{v}_l^0$ and $\hat{v}_l^1$ are in *frontier* or *explored*; nodes v_l^0 and v_l^1 are in *repo*; (c) Case (iv), all nodes between $\hat{v}_l^1$ and v_m^1 are assigned 1. The level of the first node with $value = 1$ on subpath $\tilde{\pi}^*$ can be higher, equal or lower than the level of v_m^1.

Let $\tilde{\pi}^*$ denote the subpath of π^* starting with v_l^0 and $\tilde{\pi}$ denote the subpath of π starting with v_l^1 . There are four cases as to whether v_l^0 and v_l^1 can be added to the priority queue *frontier* or *repo*: (i) both v_l^0 and v_l^1 are added to the priority queue *frontier*; (ii) both v_l^0 and v_l^1 are added to the reserved queue *repo*; (iii) v_l^1 is added to the *frontier* queue but v_l^0 is added to the queue *repo*; and (iv) v_l^0 is added to the *frontier* queue but v_l^1 is added to the queue *repo*. We discuss these four cases separately. The idea is that by assumption, the suboptimal path is formed before the optimal path, thus the algorithm has to switch to expand v_l^1 eventually. But the following discussion states that even though the algorithm searches the path starting from v_l^1, it will switch back to searching the optimal path.

(i) Both v_l^0 and v_l^1 are added to the priority queue *frontier*. Since both v_l^0 and v_l^1 are added to the set *frontier*, there does not exist a third node in *explored* or *frontier* with the same *value* and *level* as v_l^0 or v_l^1, see Fig. B.1(a). Therefore, no node with $level = l + 1$ is in *frontier* or *explored*. Moreover, v_l^0 is selected before v_l^1 since it has higher priority. Expanding v_l^0, our algorithm generates two children v_{l+1}^0 and v_{l+1}^1. If the $(l + 1)$-th sensor is attack-free, then both these children nodes will enter *frontier*. The algorithm will continue expanding nodes on the path starting from v_l^0. When the search algorithm reaches the first node with $value = 1$ on $\tilde{\pi}^*$, denoted by v_m^1, then either v_m^1 will enter *frontier* and Alg. 1 will continue expanding it, or v_m^1 will enter *repo* and the algorithm will expand v_l^1. Since, v_l^0 has been expanded, the children of v_l^1 will enter *repo*. Moreover, v_m^1 will be selected from *repo* before the children of v_l^1 since it has higher priority. Therefore, the algorithm continues searching on the path starting from v_l^0. Since we assume that the assignment corresponding to π is the output, the algorithm should switch to searching the children of v_l^1 eventually. In order to expand the children of v_l^1, except for v_m^1, there must exist another node on $\tilde{\pi}^*$ with *value* 1, denoted by v_n^1, and it should be put in *repo*. This is because if v_n^1 enters *frontier*, the search will still advance on $\tilde{\pi}^*$. But if v_n^1 enters *repo*, the children of v_l^1 will exit *repo* before v_n^1, since they contain fewer number of attacked sensors. Meanwhile, the set *explored* is emptied. However, the counterpart $\hat{v}_n^1$ of v_n^1 on $\tilde{\pi}$ will also enter *repo*, since the algorithm will take identical steps searching $\tilde{\pi}$ as when searching $\tilde{\pi}^*$. But v_n^1 has higher priority than $\hat{v}_n^1$, therefore, the algorithm switches to searching $\tilde{\pi}^*$ again. Following this logic, we conclude that the search on $\tilde{\pi}$ never surpasses that on $\tilde{\pi}^*$. Intuitively, for those nodes that are assigned 1 on $\tilde{\pi}^*$, the nodes on $\tilde{\pi}$ will be assigned 1 as well, since all feasible paths should detect all attacked sensors correctly. Furthermore, the zero assignments on $\tilde{\pi}$ are a subset of those on $\tilde{\pi}^*$. The nodes on $\tilde{\pi}^*$ always have the advantage over their counterparts on $\tilde{\pi}$ with the same *value* and *level* in terms of priority since the optimal path has the least number of attacked sensors. In addition, v_l^0 is expanded before v_l^1. Therefore, the last node on the optimal path will exit *frontier* before the last node on the suboptimal path. Thus, it is impossible that the suboptimal path is fully formed before the optimal one.

(ii) Both v_l^0 and v_l^1 are added to the reserved queue *repo*. If v_l^0 is selected from *repo* before v_l^1, *frontier* only contains v_l^0 and *explored* is empty. Thus, the algorithm starts searching nodes on the path starting from v_l^0, as if v_l^0 is the new root. As we have discussed in (i), the algorithm will continue searching on the path starting from v_l^0. In order to expand v_l^1, since v_l^1 is on the suboptimal path that corresponds to the output of the algorithm, there must exist a second node v_n^1 with $value = 1$ which enters *repo*. Then v_l^1 can be selected from from *repo*. But even in this case, when v_l^1 is selected for expansion, the counterpart $\hat{v}_n^1$ of v_n^1 on the subpath starting from v_l^1 still enters *repo*. Same as in (i), the search on $\tilde{\pi}$ never surpasses that on $\tilde{\pi}^*$

(iii) v_l^1 is added to the *frontier* queue but v_l^0 is added to the queue *repo*. The graphical illustration of this case is shown in Fig. B.1(b). If v_l^0 is added to the queue *repo*, then a node $\hat{v}_l^0$ with the same *value* and *level* as v_l^0 is already in the queue *frontier* or the set *explored*. Therefore, the parent $\hat{v}_{l-1}$ of node $\hat{v}_l^0$, with same level as v_{l-1},

is expanded before v_{l-1}. Note that only when $\hat{v}_{l-1}$ has higher or equal priority than v_{l-1}, can $\hat{v}_{l-1}$ be expanded before v_{l-1}. The reason is that if v_{l-1} has higher priority than $\hat{v}_{l-1}$, then the only way that $\hat{v}_{l-1}$ can be expanded before v_{l-1} is if v_{l-1} or one of its predecessor on π^* are in the queue *repo* and remain there until they are selected. When this happens, the set *explored* is emptied and *frontier* contains only v_{l-1} itself or its predecessor. Therefore, there is no need to check whether an equivalent node is in *frontier* or *explored*, and v_l^0 will not enter *repo*. Thus, v_{l-1} can not have higher priority than $\hat{v}_{l-1}$. Since $\hat{v}_{l-1}$ is expanded before v_{l-1}, $\hat{v}_l^1$ enters *frontier* or *repo* (because an equivalent node is in *frontier* or *explored*) before v_{l-1} is expanded. This means that v_l^1 will be put in *repo*, a contradiction. Hence, case (iii) is impossible.

(iv) v_l^0 is added to the *frontier* queue but v_l^1 is added to the queue *repo*. Like case (iii), since v_l^1 is put in *repo* when expanding v_{l-1}, there exists a node $\hat{v}_{l-1}$ that is expanded before v_{l-1} and $\hat{v}_l^1$ is in *frontier* or *explored*. Since v_l^0 can enter frontier, we have that $\hat{v}_l^0$ is invalid, see Fig. B.1(c). $\hat{v}_l^0$ being invalid means that the observability matrix corresponding to the path leading to $\hat{v}_{l-1}$ has full rank or sensor l is under attack. However, the true attack assignment to sensor l is 0, thus the first case holds. Then, $\hat{v}_l^0$ makes the system of linear equations overdetermined and inconsistent, meaning an attacked sensor is mistreated as attack-free on the path leading to $\hat{v}_l^0$. Thus, for the following nodes on the path starting from $\hat{v}_l^1$ that shares the same parent $\hat{v}_{l-1}$ with $\hat{v}_l^0$, *value* can only be set to 1 when those nodes are generated. Assume that when Alg. 1 expands v_{l-1}, the highest node that is in *frontier* or *repo* on the path starting from $\hat{v}_l^1$ is v_m^1. Since the algorithm switches to search v_{l-1}, we have that v_{l-1} has fewer attacked sensors than v_m^1. After the algorithm switches back to search v_{l-1}, we analyze what will happen next by the relationship between the level of the first node with $value = 1$ on $\tilde{\pi}^*$ and the level of v_m^1. (a) If the level of the first node with *value* 1 on $\tilde{\pi}^*$ is less than or equal to the level of v_m^1, this node will enter *repo* and exit *repo* before v_l^1. (b) If the first node on $\tilde{\pi}^*$ with *value* 1 is at a higher level than v_m^1, then the search will advance on $\tilde{\pi}^*$ until reaching the second node with *value* 1. But this node will enter *frontier* instead of *repo*, hence, the search will continue on $\tilde{\pi}^*$. The search on $\tilde{\pi}$ never surpasses that on $\tilde{\pi}^*$.

We conclude that, except case (iii) which does not occur, in the other three cases the algorithm always switches back to expanding the optimal path. This means that the optimal path is formed first, which contradicts our assumption that the suboptimal path is formed before the optimal one, completing the proof.