Attacks and Defenses on Consensus Algorithm for Distributed Networks

Kassem Kallas

October 30, 2015

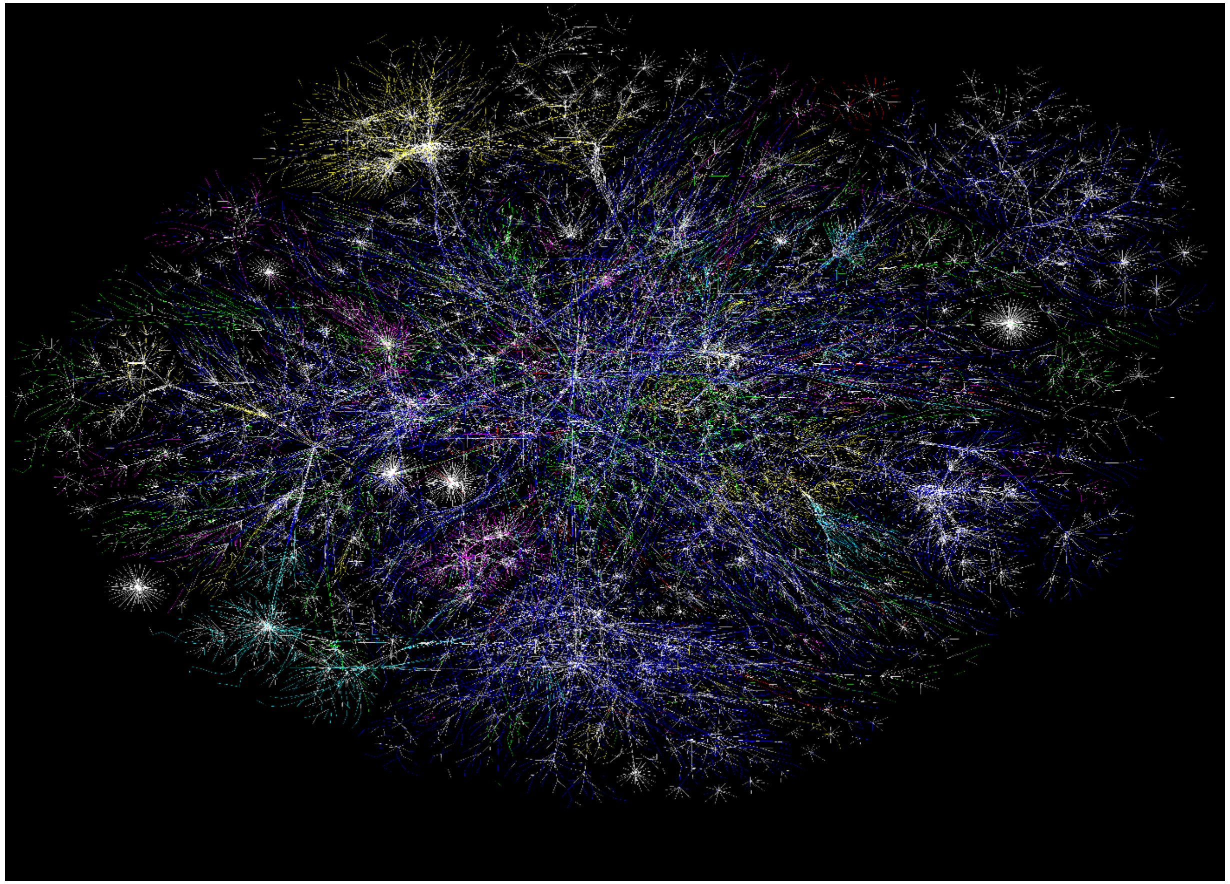
UNIVERSITÀ DI SIENA
1240

Outline

- Introduction
- Signal Processing over Graphs
- Consensus Algorithms: An Introduction
- Consensus Algorithm for Distributed Networks
- Attacks and Defenses on Consensus Algorithm

Introduction

- Network science has emerged in science and engineering to help to understand the interactions between agents.
- Networks are everywhere
  - Transportation networks.
  - Energy Networks.
  - Biological Networks (GRN…).
  - Social Networks (Facebook, Twitter …).
  - Communication Networks (WSN, CRN …).
  - Many others …
- Networks can be: static, dynamic and random.



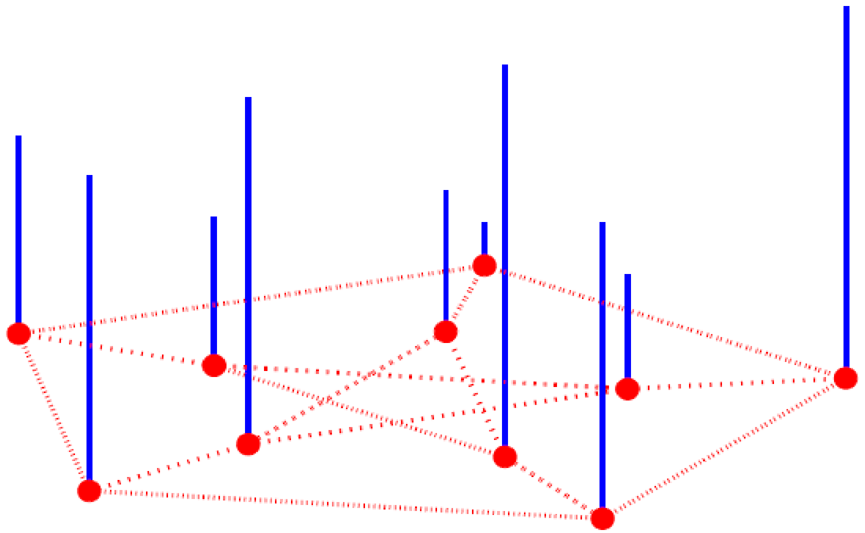
Internet Map- Jan. 15 2005: http://www.opte.org/

Introduction

- Some of them are still growing …

Signal Processing over Graphs

- All of these networks can be represented/modeled by Graphs.
- Graphs are generic data representation.
- Consists of a set of entities.
- Relationships between entities are edges.
- Each entity has one data sample.
- The collection of samples is called Graph Signal.
- A graph signal is a function that assigns a real value to each entity $f : \mathcal{V} \rightarrow \mathbb{R}$.



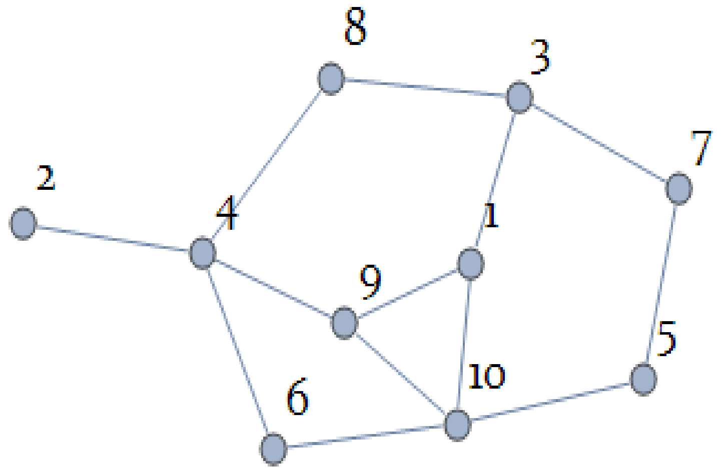
Signal Processing over Graphs

- Classical signal processing techniques ignores the graph structures.
- SPG uses spectral and algebraic graph theory to process signals over graphs.
- A theoretical framework trying to answer questions like the following:
  - What it means to translate a graph signal?
  - What is graph down-sampling?
  - What is a graph Fourier transform?
  - The notion of frequency.
  - Frequency filtering on graphs.
  - And many others …
- We are interested in the applications of SPG.
- An introduction to the field if SGP can be found in [1].

[1] Shuman, David, et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." *IEEE Signal Processing Magazine,* March 2013, pp. 83-98.

Signal Processing over Graphs – What is a Graph?

- A graph [2],[3] $G = \{V, E\}$ consists of finite set of vertices $|V| = N$ and a set of edges E.
- Nodes v_i and v_j are connected if $e_{ij} \in E$.
- $\mathcal{N}_i = \{j : e_{ij} \in E\}$ is the set of neighbors of node i.
- A walk: a set of vertices where each consecutive pair belongs to edge set.
- A path (with $l \geq 2$)is a sequence of links connecting two nodes.
  - It's a walk with no repeating edges.
- Length of a path: cardinality or sum of edges weights(if weighted G) along path.
- A graph is **connected** if any two distinct nodes are connected by a path.
- A graph can be directed and undirected.

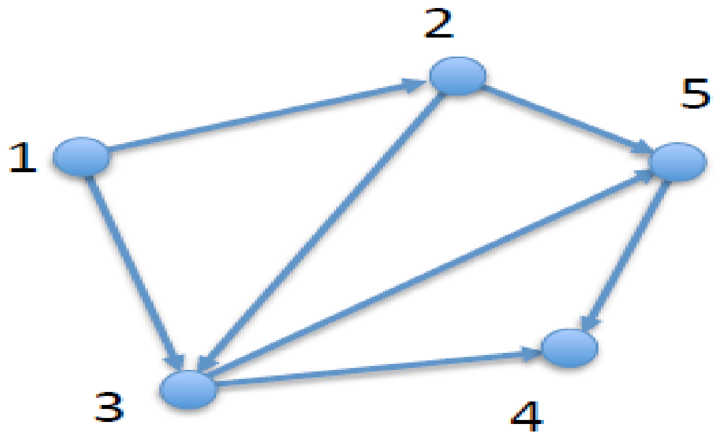


[2] Newman, Mark. *Networks: an introduction*. Oxford University Press, 2010.
[3] Mesbahi, Mehran, and Magnus Egerstedt. *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

Signal Processing over Graphs – What is a Graph?

- The graph can be characterized by a set of Matrices:

1. Adjacency Matrix $A(N \times N)$: $a_{ij} = 1$ if there's an edge between i and j and 0 otherwise.
2. Degree Matrix $D(N \times N)$: it's a diagonal matrix with $d_{ii} = \sum_{i=1}^{N} a_{ij}$.
3. Laplacian Matrix $L(N \times N)$: $L = D - A$.
4. Incidence Matrix $B(N \times E)$: $b_{ij} = 1$ if vertex i is the tail of edge j, $b_{ij} = -1$ if vertex i is in the head of edge j and 0 otherwise.



Signal Processing over Graphs – What is a Graph?

- Properties of the Laplacian Matrix $L = D - A$:

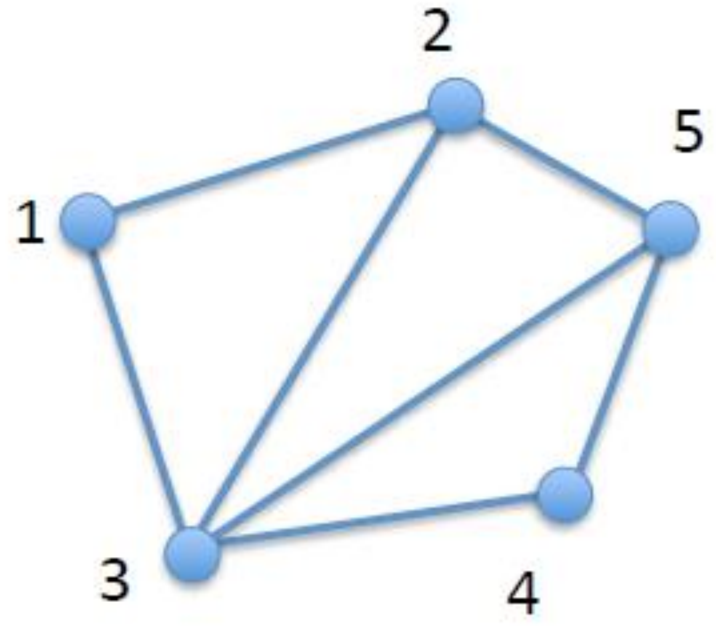
1. It is a real symmetric matrix.
2. The spectrum of L is: $\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq ... \leq \lambda_N$.
3. By construction $\lambda_1 = 0$.
4. The eigenvector of $\lambda_1 = 0$ composed of all ones.
5. The graph is connected for $\lambda_2 > 0$ and λ_2 is called the *"algebraic connectivity"*.

Signal Processing over Graphs – What is a Graph?

- Example [4]:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{bmatrix}$$



[4] Barbarossa, S. (2015, June 26). Signal Processing over graphs: Distributed optimization and bio-inspired mechanisms [Online]. Available: http://clem.dii.unisi.it/~vipp/index.php/teaching/35-signal-processing-over-graphs .

Consensus Algorithm: An Introduction

- Definition:
  - "a general agreement about something : an idea or opinion that is shared by all the people in a group"- *Merriam-Webster dictionary*.

- It's a bio-inspired mechanism that tries to mimic the natural phenomena of interaction between animals (i.e. schooling of fish) [5],[6] that interact to reach a common objective. Other bio-inspired mechanisms and their applications [7], [8]:

  - Formations: agents move to a desired geometric shape.
  - Assignment: fair assignment of tasks among agents.
  - Flocking/swarming: exhibit behavior observed in nature.
  - Many others….

[5] P.K. Visscher, "Animal behaviour: How self-organization evolves." *Nature, vol.* 421, Feb. 2003, pp. 799-800 .
[6] I.D. Couzin, "Collective cognition in animal groups." *Trends in cognitive sciences, vol.* 13, Dec. 2009, pp. 36-43.
[7] Mesbahi, Mehran, Magnus Egerstedt. " *Graph theoretic methods in multiagent networks*". Princeton University Press, 2010
[8] Sayed, H. Ali (2013, May 30). Adaptation and Learning over Networks [Online]. Available: http://asl.ee.ucla.edu/ .

Consensus Algorithm: An Introduction

- First formal study is by Morris H. DeGroot [9] in 1974 : model how a group of individuals act together to reach an agreement about a value – the average of their initial values.

- A theoretical framework for analysis of consensus algorithms for Multi-Agent Networks is provided by Olfati-Saber in[10].
  - It shows many types of consensus algorithms like f-unconstrained and f-constrained consensus…
  - Many applications:
    - Synchronization of coupled oscillators.
    - Flocking theory: for mobile sensor networks to achieve velocity matching w.r.t. neighbors.
    - Fast-consensus in Small-Worlds network design (most nodes are not neighbors of each other, but they can be reached from every other by a small number of hops).
    - Rendezvous in space: consensus in positions.
    - Distributed sensor fusion in sensor networks.
    - Distributed formation control: Multi-vehicle systems.

[9] DeGroot, Morris H. "Reaching a consensus." *Journal of the American Statistical Association, vol.* 69, 1974, pp. 118-121
[10] Olfati-Saber, R.; Fax, J.A.; Murray, R.M., "Consensus and Cooperation in Networked Multi-Agent Systems," in *Proceedings of the IEEE* , vol.95, no.1, pp.215-233, Jan. 2007

Consensus Algorithm: An Introduction

- Agents or entities living on graphs can reach agreement by employing consensus algorithms.

- The iterative form of consensus algorithm works as follows [10]:

  1. The initial step: each agent/vertex/node makes its initial opinion/state/measurement $x_i(0), \forall i \in V$.
  2. The iteration step:

     $$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$$

     Where,
     - ➢ The opinion/state of agent/vertex/node i at consensus iteration k is $x_i(k)$.
     - ➢ The step-size: ϵ.
     - ➢ The neighbor set of node i is $\mathcal{N}_i$.
     - ➢ The state received at node i from neighbor j at consensus iteration k is $x_j(k)$.
  3. Each node will take the final decision $\mathbf{H} = \begin{cases} 1, & \text{if } x^* > \lambda. \\ 0, & \text{if } x^* \leq \lambda. \end{cases}$

- The Matrix form: $\mathbf{x}(k+1) = \mathbf{P}\mathbf{x}(k)$ where, $\mathbf{P} = \mathbf{I} - \epsilon\mathbf{L}$.
- The final agreement when the network asymptotically reach a consensus is the average of initial values as:

$$x^* = (1/N) \sum_{i=1}^{N} x_i(0)$$

Consensus Algorithm: An Introduction

- *Theorem 1:* if a network with connected topology G with $0 < \epsilon < 1/\Delta$ where Δ is the maximum degree of the network then:
  1. A consensus is asymptotically reached for all initial states.
  2. $\mathbf{P}$ is doubly stochastic and a consensus is asymptotically reached as $x^* = (1/N)\sum_{i=1}^{N} x_i(0)$ for all individual states.

- So, we should be careful about the choice of ϵ → The nodes must know Δ apriori to avoid information exchange regarding the network structure.

- The role of the Laplacian Matrix $\mathbf{L}$:
  - Its spectral properties used to analyze the convergence performance of a graph G applying the consensus algorithm.
  - The second smallest eigenvalue λ_2 of $\mathbf{L}$ is called the *"algebraic connectivity"* and measures the speed of convergence.

Consensus Algorithm: An Introduction

- The previous form uses the equal weight combination, the weighted average form of consensus is as follow:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} a_{ij}(x_j(k) - x_i(k))$$

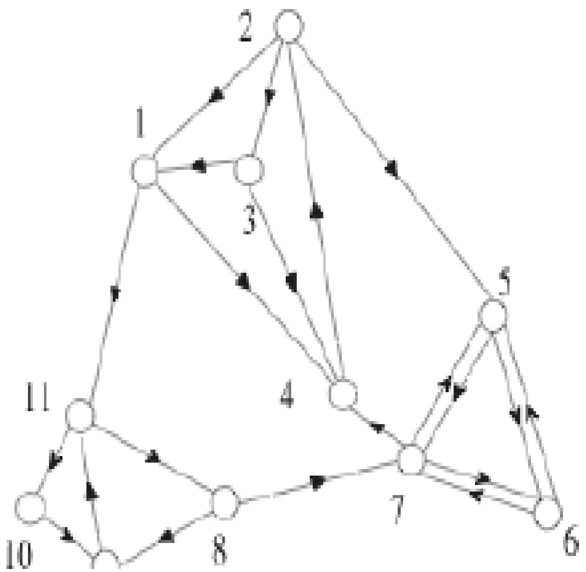
Where, a_{ij} is the weight associated with the edge e_{ij} and $\Delta = \sum_{j \neq i} a_{ij}$.

- With the same conditions in <u>*Theorem 1*</u>, the weighted consensus reach asymptotically the weighted average of the initial values.

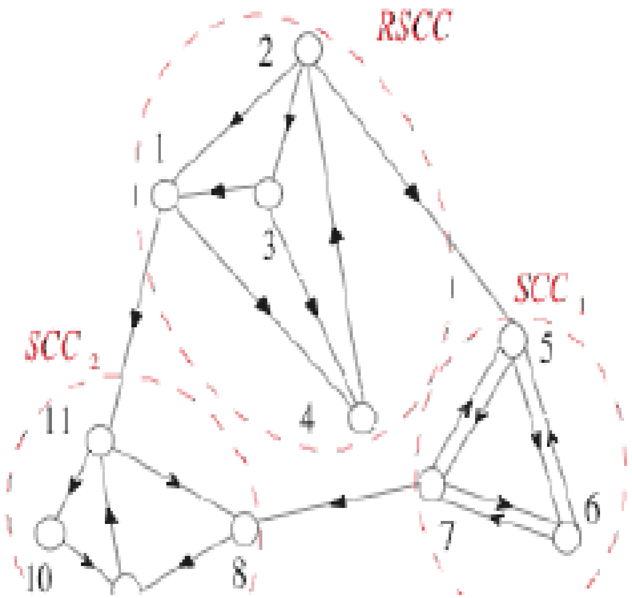
- The convergence in <u>directed</u> graphs (digraphs) applying consensus algorithm is interesting [4].

  - Strongly Connected (SC) digraph: if there exist a "strong path" between each pair of nodes.
  - Quasi Strongly Connected (QSC) digraph: for each pair of nodes there exist a third node that can reach both by a strong path.
  - Weakly Connected (WC) digraph: if each pair is connected by a "weak path".
  - Otherwise, disconnected digraph.

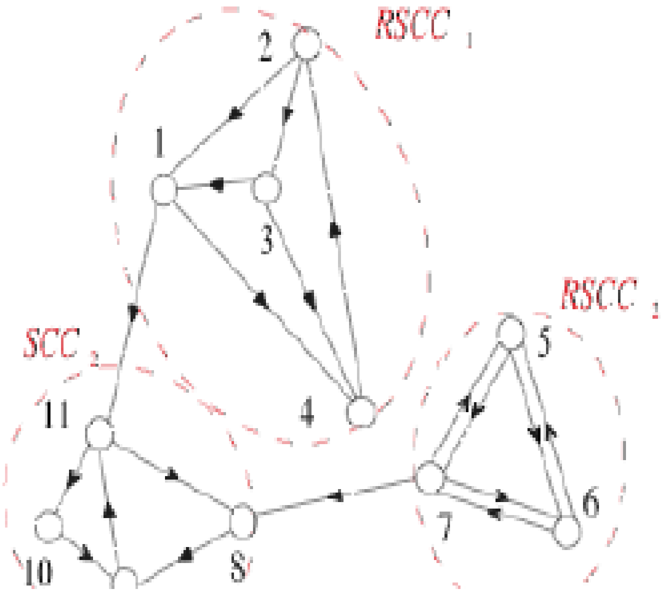
Consensus Algorithm: An Introduction



Strongly Connected Digraph
SC.



Quasi-Strongly Connected Digraph
QSC.



Weakly Connected Digraph
WC (2-tree forest/clusters).

Consensus Algorithm: An Introduction

- The right eigenvector associated to the zero eigenvalue is zero by construction. If γ is the left eigenvector associated to the zero eigenvalue then,

Strongly Connected Digraph	Digraph with one connected spanning Tree	The digraph is a K-root SCC→K clusters
$$x^* = \frac{\sum_{i=1}^{N} \gamma_i x_i(0)}{\sum_{i=1}^{N} \gamma_i}$$	$$x^* = x_{root}(0)$$	$$x^* = \frac{\sum_{j \in C_k} \gamma_i x_i(0)}{\sum_{j \in C_k} \gamma_i}$$ For each cluster.

- Some Applications of Consensus Algorithms in communication technologies:
  - Wireless sensor Networks (WSN).
  - Mobile Ad Hoc Networks (MANET).
  - Vehicular Ad Hoc Networks (VANET).
  - Cognitive Radio Networks (CRN).

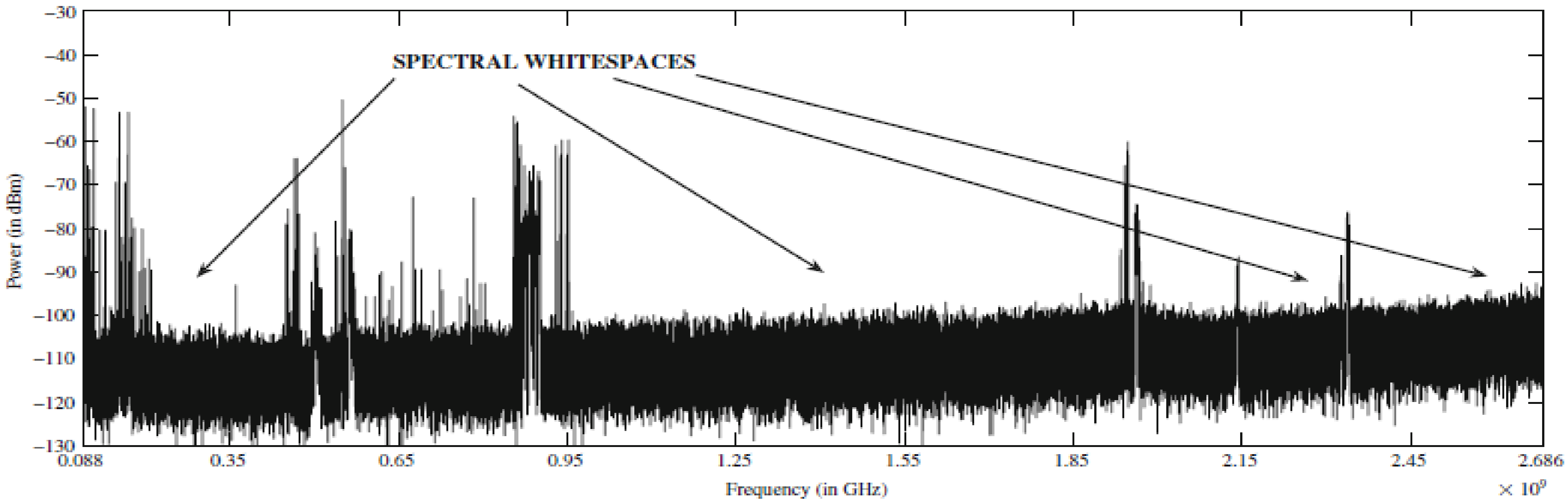
- What's Cognitive Radio [11] ,[12]?
  - Many definitions…
  - It's a brain-empowered wireless communication.
  - It has intelligent adaptation of environmental changes by self-reconfiguration of its parameters.
  - Enhance the real-time experience of the end user by increasing the spectrum utilization.
  - Interoperable with the existing and heterogeneous technologies.
  - Has several functionalities: Spectrum Sensing, Dynamic Spectrum Access, Interference Management …

- Why Cognitive Radio?

[11] Haykin, Simon, "Cognitive radio: brain-empowered wireless communications," in *IEEE Journal on Selected Areas in Communications,* vol.23, no.2, pp.201-220, Feb. 2005
[12] Mitola III, Joseph. "Cognitive radio." PhD dissertation, Royal Institute of Technology, 2000.

Consensus Algorithm for Distributed Networks –Introduction to Cognitive Radio

- The spectrum reaches a crisis and the wireless technology demand is ever increasing (fixed assignment regulation is a problem).
- Measurements of Federal Communications Commission (FCC) shows that the spectrum is heavily underutilized.
- Spectrum Holes (Spectral white spaces) exploitation increases the quality of user experience.
- CR is a promising technology to solve the spectrum scarcity problem.



Power Spectral Density (PSD) measured on July 11,2008 in Worcester, Massachusetts[13].

[13] Alexander M. Wyglinski, Maziar Nekovee, and Y. Thomas Hou. “Cognitive Radio Communications and Networks”. Elsevier Inc, 2010, ch. 6, sec. 6.1, pp. 150

Consensus Algorithm for Distributed Networks –Introduction to Cognitive Radio

- The license holder is called Primary User (PU) and the user who access dynamically the vacant space is called Secondary User (SU).
- SUs can use the available spectrum providing no interference to PU → Spectrum Sensing is **key enabling** for CR.
- To detect the spectrum holes SUs perform continuous spectrum sensing using one of the following mechanisms: Energy Detection, Cyclo-Stationary feature detection, matched-filter detection …
- Cooperative Spectrum Sensing increases the detection performance of the spectrum by exploiting the spatial diversity of the SUs [14].
- Most of the solutions for cooperative spectrum sensing are centralized where a common Fusion Center (FC) gather the measurements or decisions from SUs to make a final decision about the spectrum occupancy.
- Recently, decentralized and peer-to-peer solutions start to appear … Why?

[14] Ghasemi, A.; Sousa, E.S., "Collaborative spectrum sensing for opportunistic access in fading environments," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, DySPAN 2005.* , Nov. 2005, pp.131-136.

Consensus Algorithm for Distributed Networks –Introduction to Cognitive Radio

- Why decentralized and peer-to-peer solutions?
  - A common receiver may not be available to perform data fusion (i.e. Ad Hoc Networks).
  - Gathering the entire data in one place may be difficult under communication constraints (deep fading).
  - FC can become a single point of failure.
  - In large networks the FC can become an information bottleneck.
  - Users do not want to share the information with a remote device.
  - Distributed nature of future networks.

- A proposal to design the SUs interactions to reach a global agreement about the presence/absence of PU in a spectrum band:

$$\boxed{\text{CONSENSUS ALGORITHM}}$$

Consensus Algorithm for Distributed Networks

- First proposal to apply the consensus algorithm to cognitive radios was in 2010 by F. Richard Yu in [15].
  - Bio-inspired mechanism for spectrum sensing in MANET with cognitive radios.
  - Energy detection is used at the sensing step to obtain initial data: $x_i(0) = Y_i = \sum_{k=1}^{M} |{x_i}^k|^2$
  - They apply the equal weight combining version of consensus algorithm at the iteration stage:
    $$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$$
  - Decentralized mechanism outperforms the OR-rule centralized mechanism in terms of P_{fa} and P_{MD}.
  - The decentralized solution needs less threshold for signal detection than the centralized solution, to achieve P_{fa} and P_{MD} below 10^{-2} : $\lambda = 11.4dB$ for consensus scheme and $\lambda = 14.8dB$ for centralized OR-rule scheme.

[15] Yu, F.R.; Minyi Huang; Tang, H., "Biologically inspired consensus-based spectrum sensing in mobile Ad Hoc networks with cognitive radios," in *IEEE Network* , vol.24, no.3, pp.26-30, May-June 2010.

Consensus Algorithm for Distributed Networks

- The same author F. Richard Yu extends the previous work in 2010 [16].
  - They use consensus-based cooperative spectrum sensing in CR to cope with *fixed* and *random undirected* graphs.
  - Equal weight combining in consensus iteration step.
  - The *random* graph $G(k) = (V, E(k))$ is used to model random (independent with the same probability) link failures.

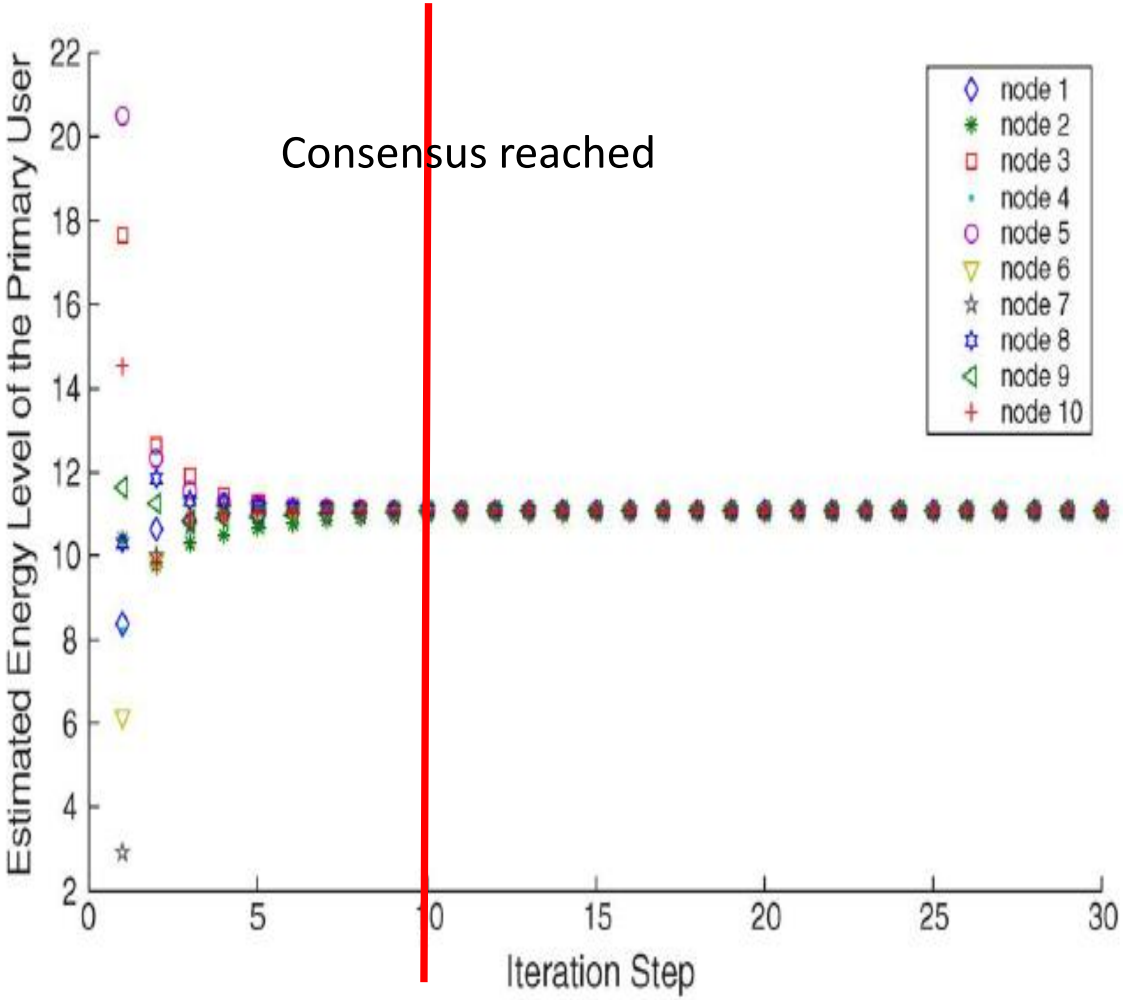
$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i(k)} (x_j(k) - x_i(k))$$

- <u>*Theorem* 2</u>:
  - In the independent link failures assumption, asymptotically the algorithm reach an average consensus of the initial values.

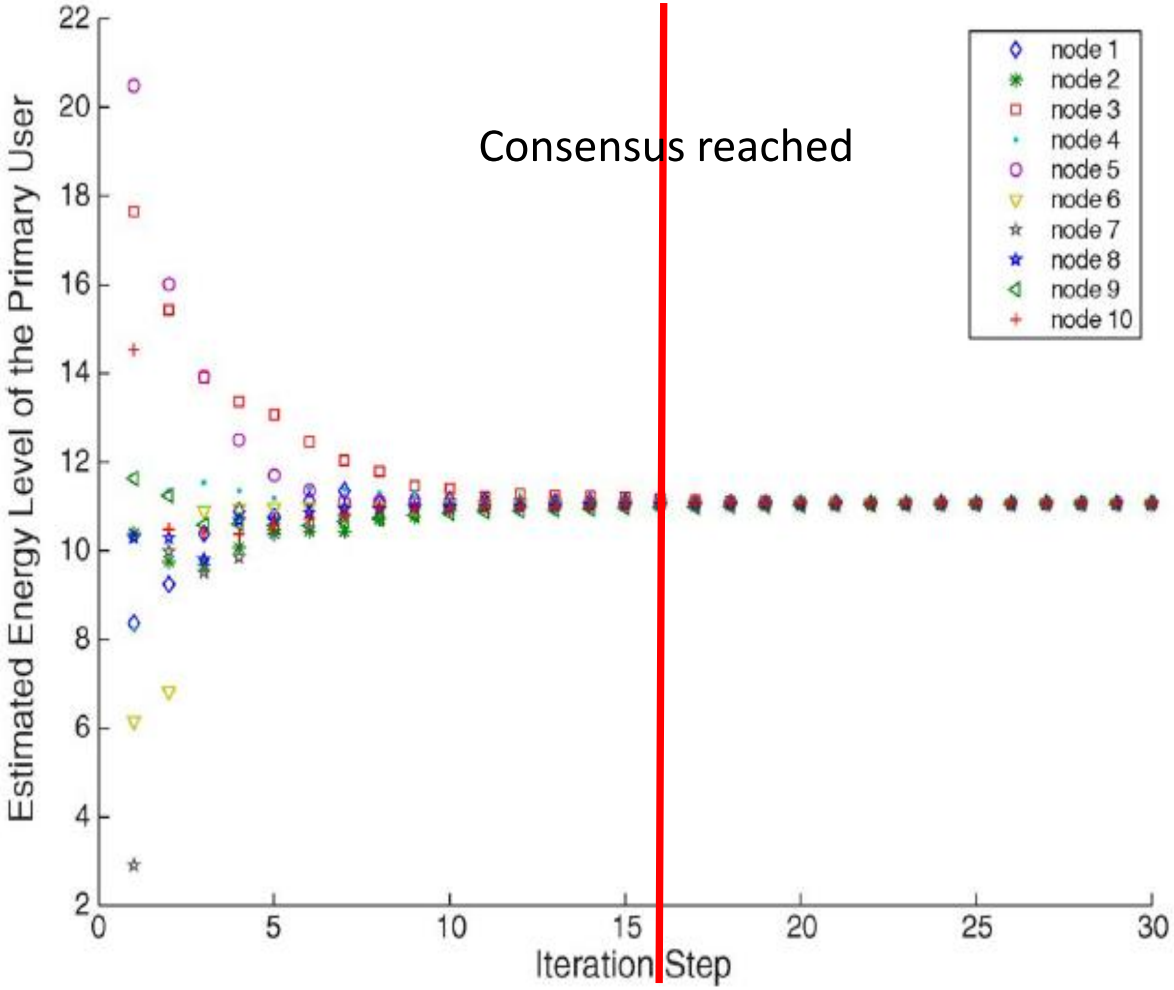
[16] Zhiqiang Li; Yu, F.R.; Minyi Huang, "A Distributed Consensus-Based Cooperative Spectrum-Sensing Scheme in Cognitive Radios," in *IEEE Transactions on Vehicular Technology*, vol.59, no.1, pp.383-393, Jan. 2010.

Consensus Algorithm for Distributed Networks

- Simulation results show the convergence of consensus algorithm for both fixed and random(more slow) graphs.



Convergence with 10-nodes fixed Graph, epsilon=0.19.



Convergence with 10-nodes random Graph, epsilon=0.19, p=0.4.

Consensus Algorithm for Distributed Networks

- Joseph Mitola III et al. in 2011 proposed weighted average consensus for distributed cooperative spectrum sensing in [17].
  - At the sensing stage, the CRs employ Energy Detection.
  - At the iteration stage:

$$x_i(k+1) = x_i(k) + \frac{\epsilon}{\sigma_i} \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$$

Where, $\sigma_i \geq 1$ is weighting factor of i^{th} SU and changes according to its confidence about its own measurement, how? Ans: Based on the estimated average SNR $\bar{\gamma}_i$.

$$\begin{cases} \sigma_i = \bar{\gamma}_i \text{ if } \bar{\gamma}_i > 1 \\ \sigma_i = 1, \text{otherwise} \end{cases}$$

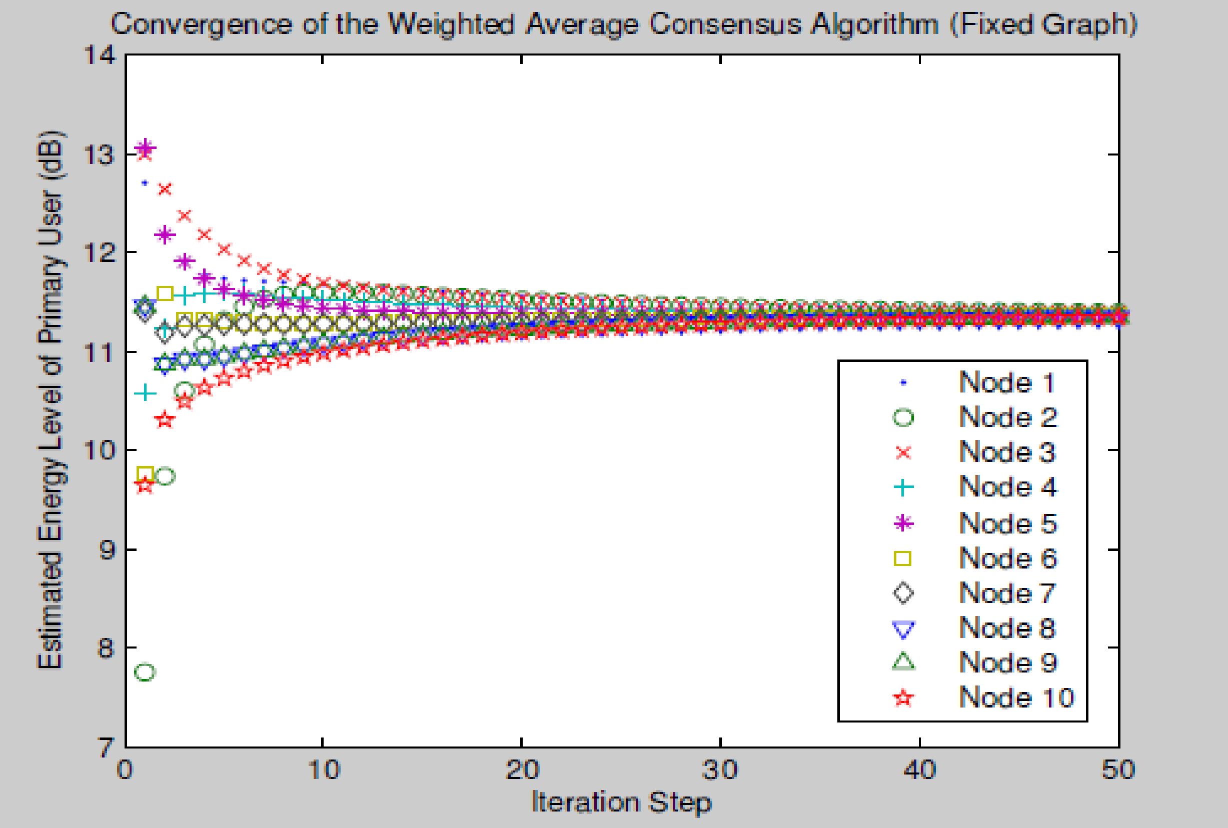
Level of trust is based on the link quality

  - The Perron Matrix becomes: $\mathbf{P} = \mathbf{I} - \epsilon diag\{\sigma_1, ...\sigma_n\}\mathbf{L}$
  - The final convergence value is: $x^* = \frac{\sum_{i=1}^{n} \sigma_i x_i(0)}{\sum_{i=1}^{n} \sigma_i}$

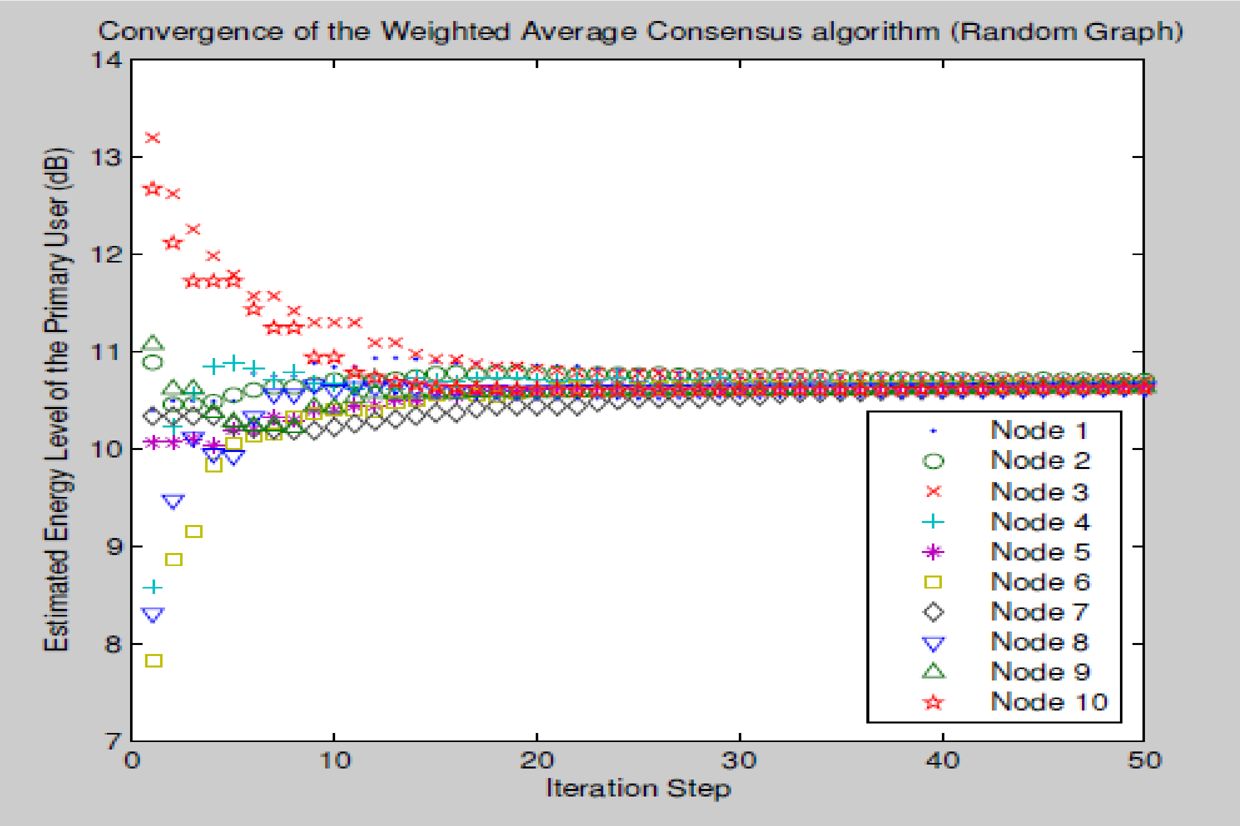
[17] Wenlin Zhang; Zheng Wang; Yi Guo; Hongbo Liu; Yingying Chen; Mitola, J., "Distributed Cooperative Spectrum Sensing Based on Weighted Average Consensus," in *IEEE GLOBECOM,* pp.1-6, Dec. 2011.

Consensus Algorithm for Distributed Networks

- *Theorem 3:* a graph with temporary independent random link failures can asymptotically converges to the consensus value (the weighted average) if the union graph of the unidirected graphs $\{G_1, G_2, \ldots, G_r\}$ formed at failures is connected.



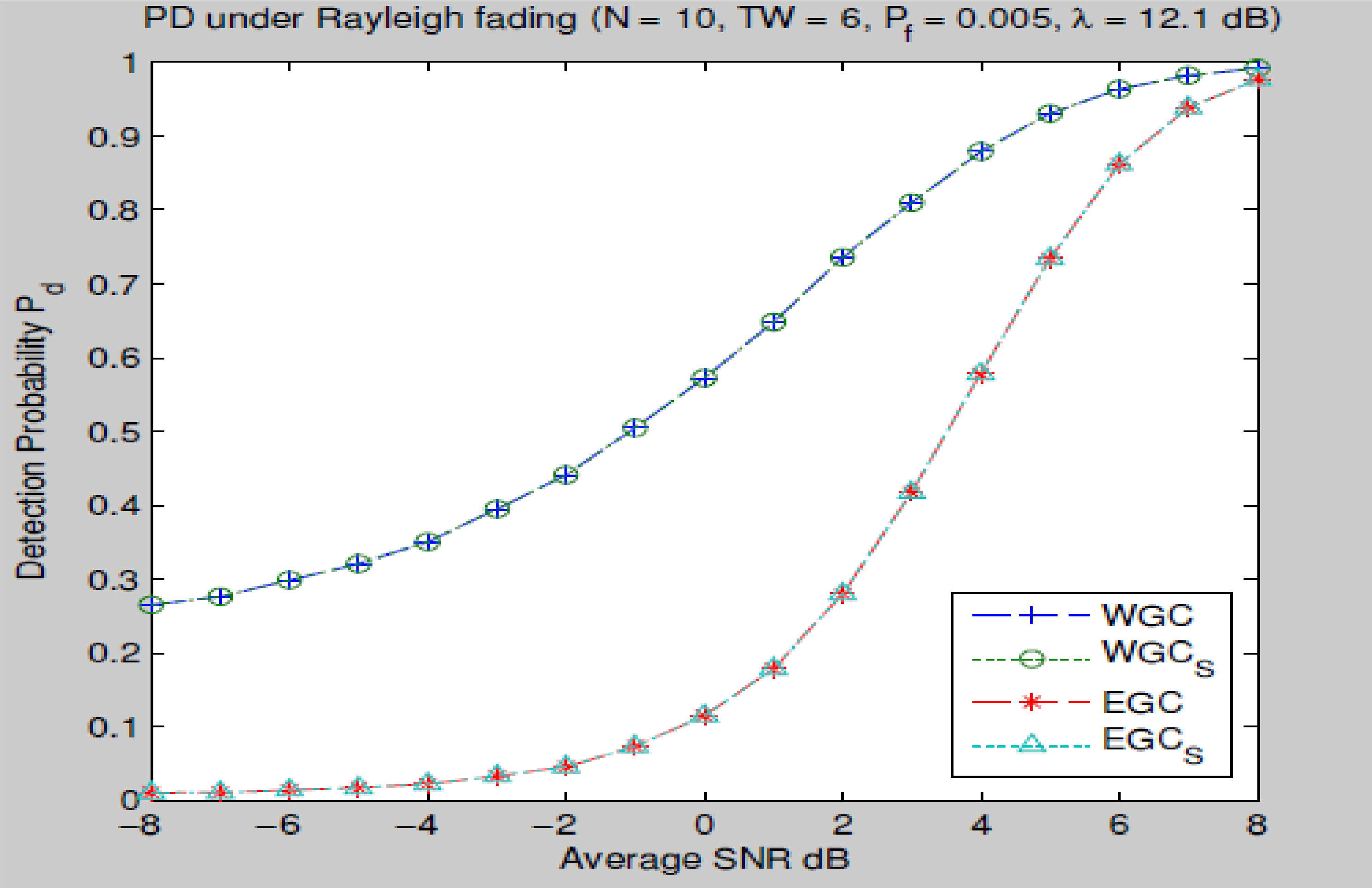
Convergence with 10-nodes fixed Graph, epsilon=0.19.



Convergence with 10-nodes random Graph, epsilon=0.19, p=0.4.

Consensus Algorithm for Distributed Networks

- [17] also shows that weighted gain combining outperforms the equal gain combining for both fixed and random Graphs.



Attacks and Defenses on Consensus Algorithm - Attacks

- Everything works fine and employing consensus algorithm for distributed networks works well.

- But, THE ATTACKER has another opinion.

- Many solutions for centralized networks to defend against attackers
exist: [18],[19],[20] and many others … BUT very less for distributed networks.

[18] Rawat, A.S.; Anand, P.; Hao Chen; Varshney, P.K., "Collaborative Spectrum Sensing in the Presence of Byzantine Attacks in Cognitive Radio Networks," in *IEEE Transactions on Signal Processing* , vol.59, no.2, pp.774-786, Feb. 2011
[19] Abrardo, A.; Barni, M.; Kallas, K.; Tondi, B., "Decision fusion with corrupted reports in multi-sensor networks: A game-theoretic approach," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, pp.505-510, Dec. 2014.
[20] Abrardo, A.; Barni, M.; Kallas, K.; Tondi, B. "Optimum Fusion of Possibly Corrupted Reports for Distributed Detection in Multi-Sensor Networks." *arXiv preprint arXiv:1503.05829*, 2015.

Attacks and Defenses on Consensus Algorithm - Attacks

- The attackers effect on consensus algorithms:
  - Can make the network diverges.
  - Can reverse the correct state about the system/observed phenomena.
  - Can makes the network converges to its injected value.
  - Can prevent the network from reaching a consensus.

- False data injection attacks is called in CRN context as Spectrum Sensing Data Falsification(SSDF) attacks and are a type of Byzantine attacks.

- Other types for attacks for CRN exists in addition to traditional network attacks i.e. Primary User Emulation Attack [21]. A survey on attacks and defense in CRN is found in [22].

- The type of SSDF attack depends on the attacker's objective:
  - Selfish SSDF: exploitation objective $H_0 \rightarrow H_1$.
  - Interference SSDF: vandalism objective $H_1 \rightarrow H_0$.
  - Confusing SSDF: confusion objective.

[21] Ruiliang Chen; Jung-Min Park; Reed, J.H., "Defense against Primary User Emulation Attacks in Cognitive Radio Networks," in *IEEE Journal on Selected Areas in Communications* , vol.26, no.1, pp.25-37, Jan. 2008.
[22] Fragkiadakis, A.G.; Tragos, E.Z.; Askoxylakis, I.G., "A Survey on Security Threats and Detection Techniques in Cognitive Radio Networks," in *IEEE Communications Surveys & Tutorials,* vol.15, no.1, pp.428-445, First Quarter 2013

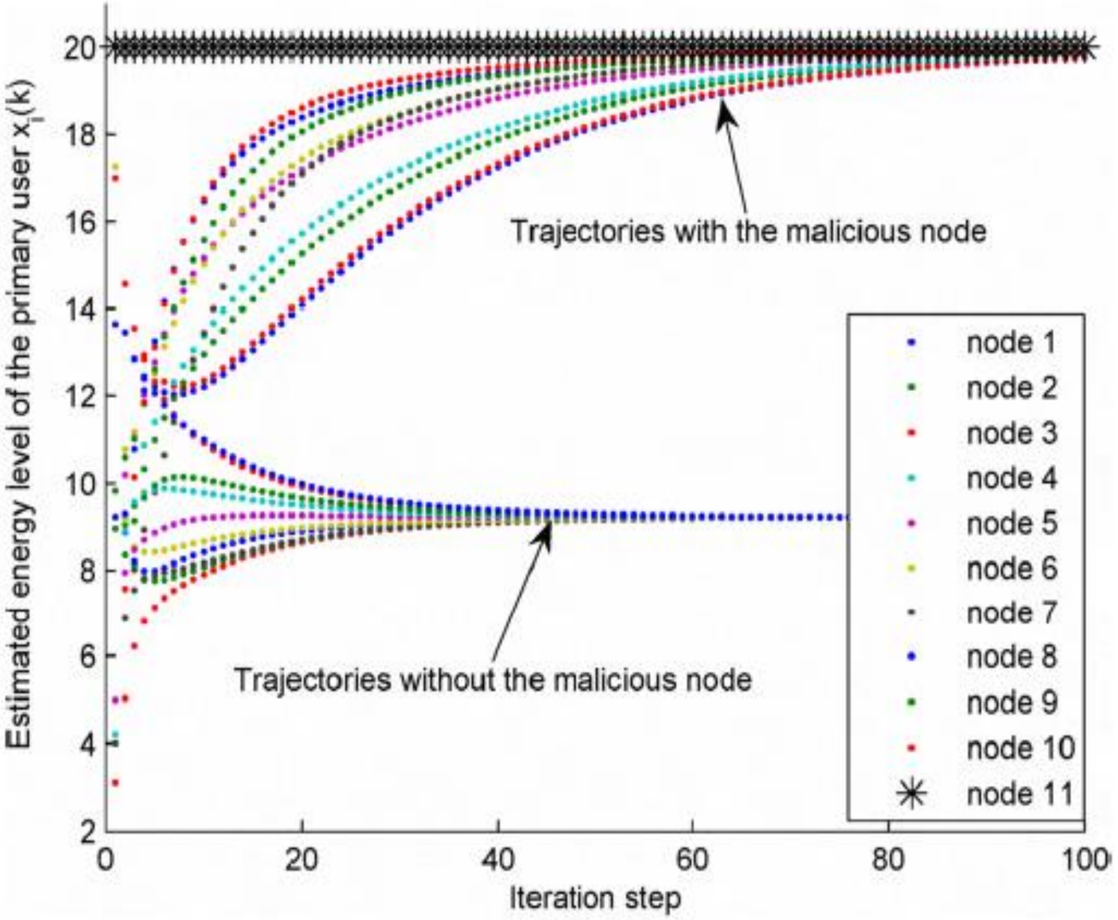
Attacks and Defenses on Consensus Algorithm - Attacks

- Also can be classified according to the stage of consensus it chooses to attack:

  - Sensing Data Falsification (SDF): occurs in the sensing stage.

  - Iterative Stage Falsification(ISF): forge sensing data and inject fake states at each iteration step.

  - Random Data Falsification(RDF): randomly choosing between to send correct or manipulated state at each step.



- Other types:
  - Insider attacker: if it has all the key material of the network→authentic.
  - Outsider attacker: doesn't have the key material but can perform replay, camouflage other nodes by their captured identities … But, can be easy removed by authentication.

Attacks and Defenses on Consensus Algorithm – Defense Solution 1

- The first proposal to defend against SSDF attacks was for MANET with CRs by F. Richard Yu in [23].
  - The attackers considered are **static** and send **constant data** value at each iteration step.
  - Equal weight combining version of consensus algorithm is considered.
  - The outlier detection is based on the deviation for the mean of the reports and works at $k \geq 1$ iteration step and when the number of neighbors is $|\mathcal{N}_i| > 2$.
  - How it works?
    1. User i gets the local mean value at $k-1$: $\mu_i(k-1) = \frac{x_i(k-1)+\sum_{j \in \mathcal{N}_i} x_j(k-1)}{1+|\mathcal{N}_i|}$
    2. User i identifies the neighbors with maximum deviation from the mean value:
    $$\hat{j} = \arg\max_{j \in \mathcal{N}_i} |x_j(k) - \mu_i(k-1)|$$
    3. User i forms an authentic neighbor set as: $\hat{\mathcal{N}}_i(k) = \mathcal{N}_i \setminus \{\hat{j}\}$

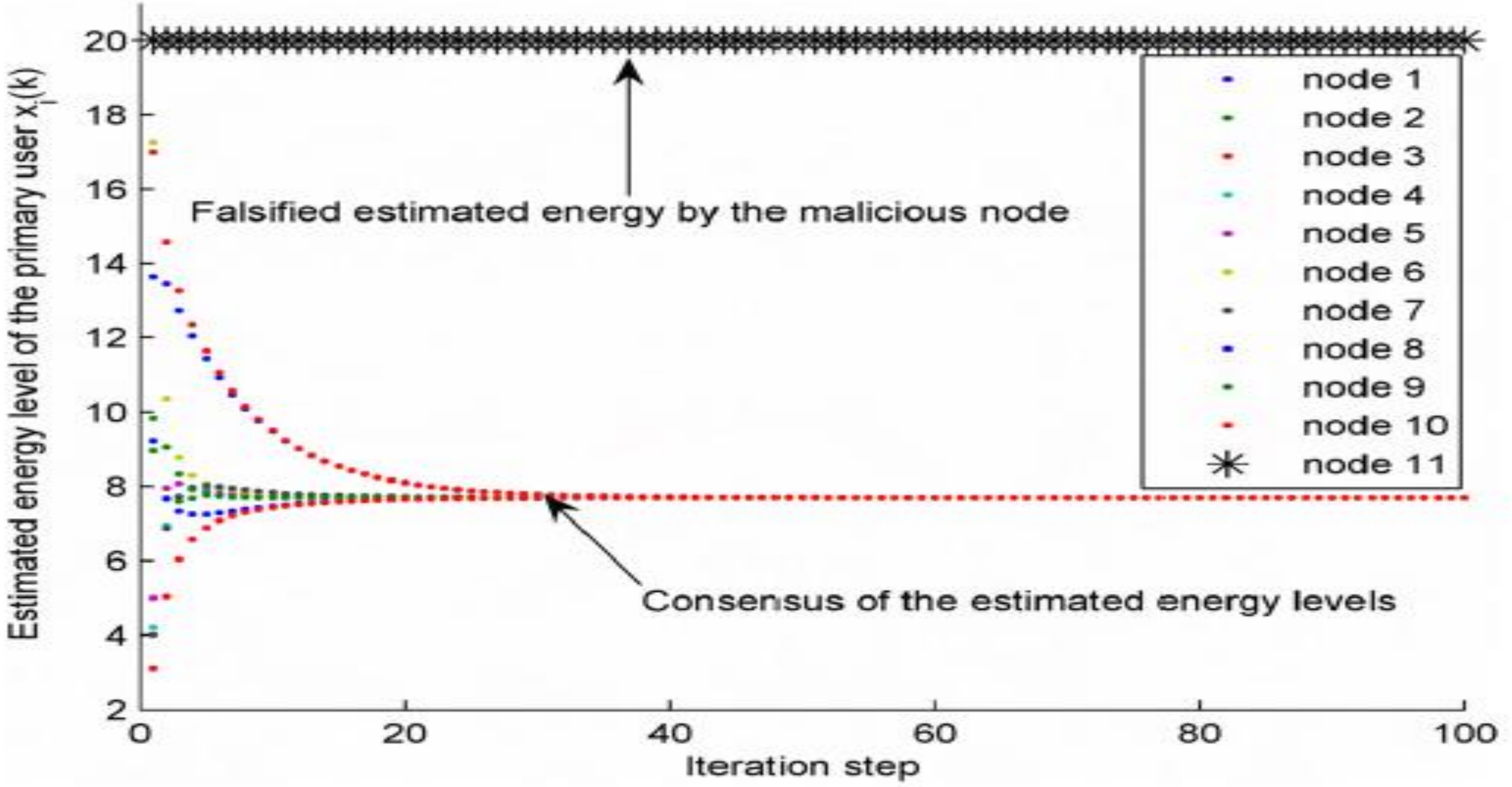
[23] Yu, F.R.; Tang, H.; Minyi Huang; Zhiqiang Li; Mason, P.C., "Defense against spectrum sensing data falsification attacks in mobile ad hoc networks with cognitive radios," in *IEEE MILCOM 2009.* pp.1-7, Oct. 2009

Attacks and Defenses on Consensus Algorithm – Defense Solution 1

- Then, the update will be as: $x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \hat{\mathcal{N}}_i(k)} (x_j(k) - x_i(k))$

Authentic neighbor set

- Drawbacks:
  - It will remove a user even if there are no attackers.
  - It's possible to result in unidirectional information exchange in graph G.



Attacks and Defenses on Consensus Algorithm – Defense Solution 2

- The previous work was extended by the same authors [24] in 2012. Prior to the neighbor with maximum deviation filtration they add Authentication technique using ID-based cryptography with threshold secret sharing.

  - How to works? Suppose an SU **S** wants to send the estimated energy level to its neighbor SU **D**:
    1. **S** sign the message with its private key and encrypt it using **D**'s ID and then he send.
    2. **D** decrypt using its private key and then using **S**'s public key.
    3. If the verification succeed, the message can enter to the consensus update procedure.
  - Note that, there's no Central Authority to distribute the certifications but instead are distributed in (k out of N) decentralized manner.

- This works show a slight improvement over the previous scheme without the ID-based cryptography.

[24] Tang, H.; Yu, F.R.; Huang, M.; Li, Z., "Distributed consensus-based security mechanisms in cognitive radio mobile ad hoc networks," in *IET Communications* , vol.6, no.8, pp.974-983, May 2012.

Attacks and Defenses on Consensus Algorithm – Defense Solution 3

- Another defense method called “Adaptive Deviation Tolerant(ADS)” is proposed in [25].
  - Idea is to tolerate large deviation for honest users and defend against false data injection from attackers.
  - How ADS works?
    1. At each iteration step, each node counts the number of neighbors with deviation larger and not larger than a threshold λ_2 denoted as $m_i(k)$ and $n_i(k)$, respectively.

    2. **If** $n_i(k)+1 > m_i(k)$ → the node believe that its measurement is relatively correct and the update is:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i^T(k)} (x_j(k) - x_i(k)) + \frac{\epsilon}{a} \sum_{j \in \mathcal{N}_i^F(k)} (x_j(k) - x_i(k))$$

$$|x_j(k) - x_i(k)| < \lambda_2^i$$

Reducing factor

complement of $\mathcal{N}_i^T(k)$

Else, the update procedure is back to normal:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$$

3. i^{th} node computes the threshold $\lambda_2^i(k)$ as $\lambda_2^i(k) = \frac{1}{|\mathcal{N}_i|} \sum_{j^* \in \mathcal{N}_i} \left| x_{j^*}(k) - \frac{x_i(k) + \sum_{j \in \mathcal{N}_i} x_j(k)}{|\mathcal{N}_i|+1} \right|$

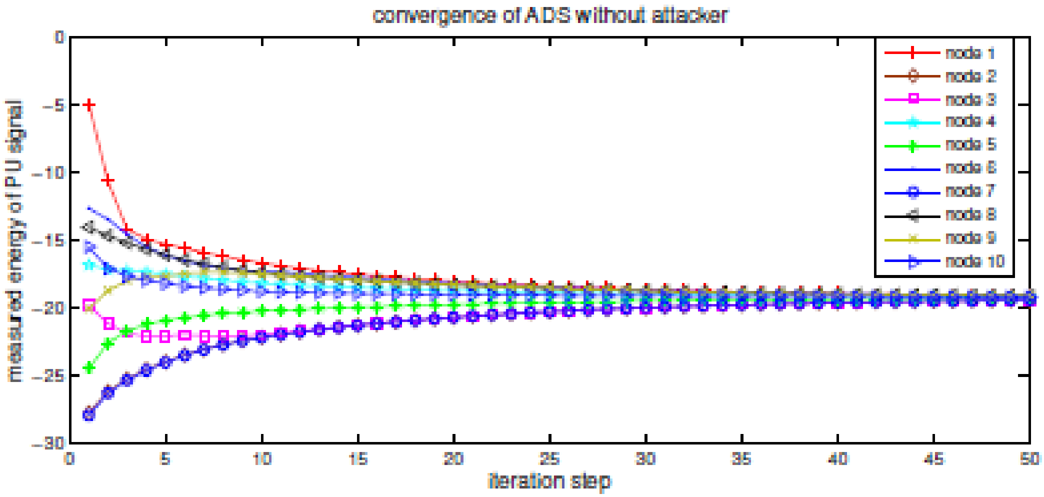
[25] Sheng Liu; Haojin Zhu; Shuai Li; Xu Li; Cailian Chen; Xinping Guan, "An Adaptive Deviation-tolerant Secure Scheme for distributed cooperative spectrum sensing," in *IEEE GLOBECOM, 2012* , pp.603-608, Dec. 2012

Attacks and Defenses on Consensus Algorithm – Defense Solution 3

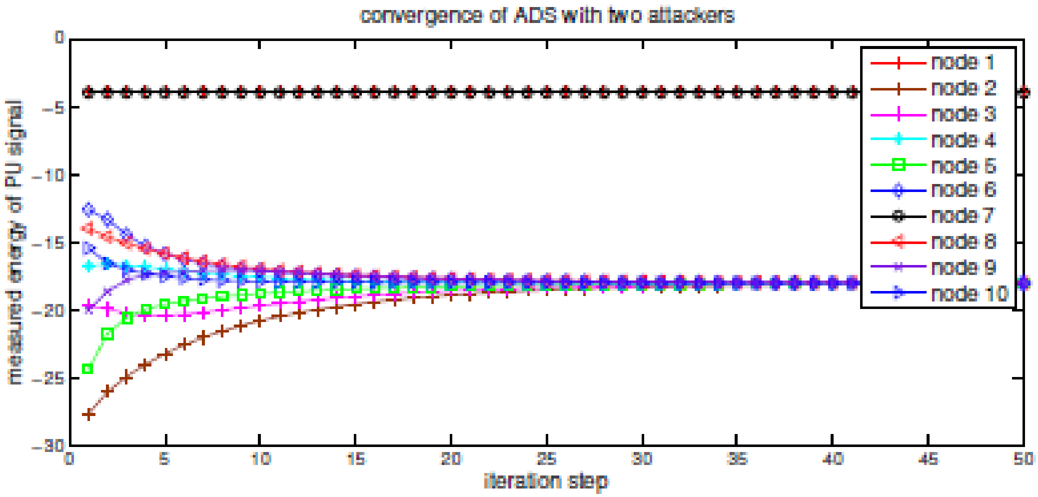
- Due to convergence property of the network, the threshold converges to zero to give zero tolerance for the attacker.

- Note: they do not explain the choice of the reducing factor a. Also, all nodes that exceed the threshold will receive the same a reduction *despite of how much* are far from the mean drift.

- The results show that ADS can filter out the attacker and tolerate the honest nodes with temporary large deviation.



(a) Convergence without attacker



(b) Convergence with two attackers

Attacks and Defenses on Consensus Algorithm – Defense Solution 4

- [26] proposes a novel type of attacks called Covert Adaptive Data Injection Attack.

- ED scheme is used at SUs to output the received power of the signal propagation model:

$$P_i = P_0 - (10\alpha log_{10}(\frac{d_i}{d_0}) + S_i + M_i)(dB)$$

Where, P_0 is the transmit power, α is the path loss exponent, $d_0 = 1m$ is the reference distance, S_i is the power loss due to shadowing and M_i multi-path fading effect.

  - M_i is considered negligible and $P_i \sim N(\mu_i, \sigma^2)$ with $\mu_i = P_0 - 10\alpha log_{10}(d_i)$.

- Equal Gain combining state update is used in consensus: $x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$.

- Covert Adaptive Data Injection Attack:
  - **Covert** means the attacker wants to inject false data without being detected.
  - **Adaptive** means using the knowledge of the detection algorithm, the attacker adapts its strategy based on neighbors' state update information.

[26] Qiben Yan; Ming Li; Jiang, T.; Wenjing Lou; Hou, Y.T., "Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks," in *IEEE Proceedings of INFOCOM, 2012*, pp.900-908, March 2012

Attacks and Defenses on Consensus Algorithm – Defense Solution 4

- How Covert Adaptive Data Injection Attack works?
    1. Collect the neighbors reports.
    2. Using the reports + outlier detection threshold λ → compute maximum acceptable deviated state.
    3. Attack strength $a(\hat{k})$ = maximum acceptable deviated state – genuine state = $\max_{i=1 \to \mathcal{N}_a} |st_i - \lambda|$. → then, inject the data into neighbors.

    - $\hat{k} \in [0, k_{stop}]$. The knowledge of k_{stop} for the attacker is crucial. If $k_{stop} \to \infty$ the consensus algorithm will not converge.
    - How much the amount of change the attacker has to inject to achieve an objective?
        - $\bar{x} + \frac{\sum_{i=0}^{k_{stop}} a(i)}{m} > \gamma, a(i) \geq 0$, to change $H_0 \to H_1$ for exploitation objective.
        - $\bar{x} + \frac{\sum_{i=0}^{k_{stop}} a(i)}{m} < \gamma, a(i) \leq 0$, to change $H_1 \to H_0$ for vandalism objective.
        - $\bar{x}$ is the average of all the original measurements. BUT, the Attacker DOES NOT know it.
        - Instead, if the min. value from neighbors $\tilde{x}_{min}(k) < \gamma$ or the max. value $\tilde{x}_{max}(k) > \gamma$, the attacker inject false data $a(\hat{k})$ for exploitation or vandalism objectives, respectively.

Attacks and Defenses on Consensus Algorithm – Defense Solution 4

- This attack becomes more powerful when malicious nodes start to collude together.

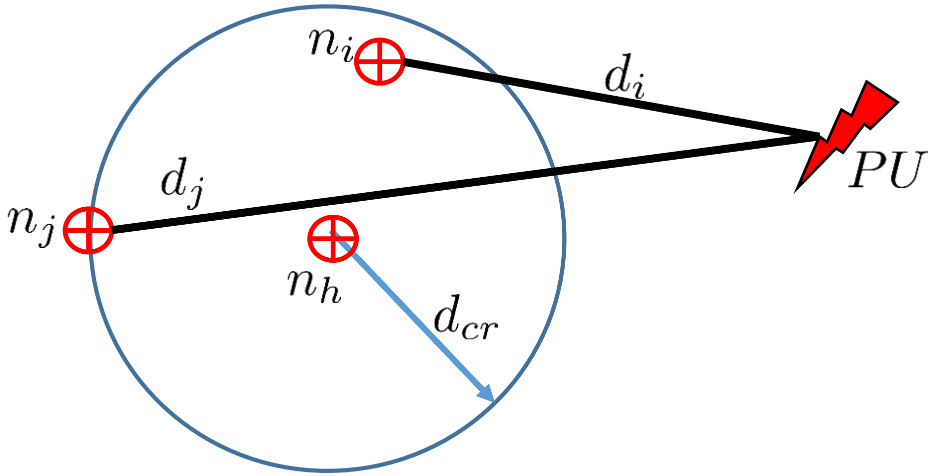
- The existing outlier detection mechanisms rely on fixed and known detection threshold.
- How to defend?
  - Observe that the max./min. states are monotonically decrease/increase until reaching consensus→the updates at all nodes are bounded by these values.
  - Difference between updates of SUs are bounded by the difference of max. and min. states.
  - Max. – Min states tends to zero with iterations.
  - Idea: use a threshold that adapt with the diminishing behavior of state differences→ tends to zero.

➢ $d_j = d_i + \Delta d_{ij}, 0 < \Delta d_{ij} \leq 2d_{cr}.$

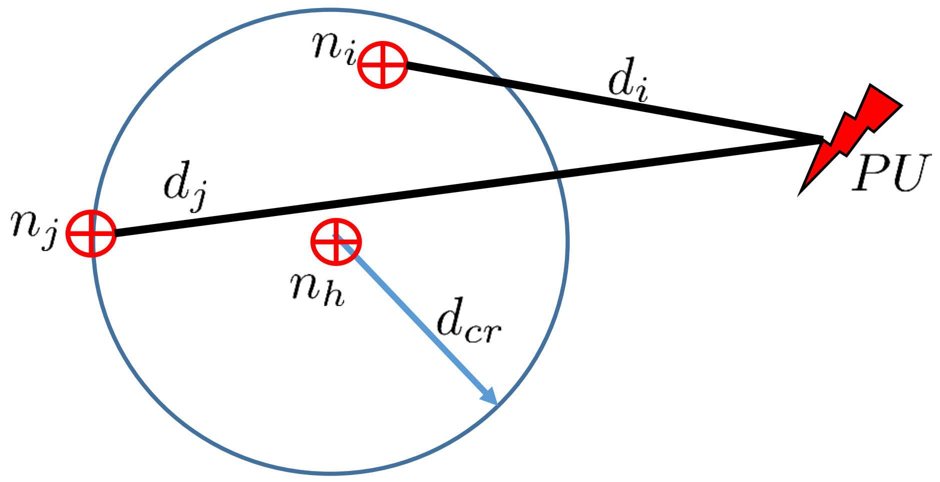
➢ $x_i(0) - x_j(0) = N(10\alpha log_{10}\frac{d_i + \Delta d_{ij}}{d_i}, 2\sigma^2).$

➢ Median and Bi-weight estimate used to estimate d_i .

➢ $x_i(0) : x_{est} = median(x_k(0))$ or $biweight(x_k(0)).$



Attacks and Defenses on Consensus Algorithm – Defense Solution 4



- So, $d_i \approx d_{est} = 10^{\frac{P_0 - x_{est}}{10\alpha}}$.
- Now, $x_i(0) - x_j(0) = N(10\alpha log_{10}\frac{d_{est}+2d_{cr}}{d_{est}}, 2\sigma^2)$.
- $\mathbf{Pr}(x_i(0) - x_j(0) < \lambda_{h0}) > 1 - \mu$, μ is detection parameter and typically, it's 0.01.
- Finally, $\lambda_{h0} = \sqrt{2}Q^{-1}(\mu) + 10\alpha log_{10}(\frac{d_{est}+2d_{cr}}{d_{est}})$ is the $k = 0$ threshold for n_h .
- At $(k+1)^{th}$ iteration $\lambda_{h(k+1)} = \frac{estdif_{h(k+1)}}{estdif_{hk}}\lambda_{h(k)}$ where, $estdif_{hk} = median(|x_{j1}(k) - x_{j2}(k)|)$

or $biweight(|x_{j1}(k) - x_{j2}(k)|)$ and $n_{j1}, n_{j2} \in \mathcal{N}_h$.

- It's apparent how the threshold and the state differences tends to zero.

Attacks and Defenses on Consensus Algorithm – Defense Solution 4

- When the neighbor's data exceed the threshold → Broadcast a primitive alarm to neighbors.

- If B is the total number of nodes in common between a node and an attacker→ the node should receive at least $\lceil B/2 \rceil$ primitive alarms to Broadcast a confirmed alarm to the rest of the network → the attacker will be removed.

- This works **ONLY** when the majority of the neighbors are honest.

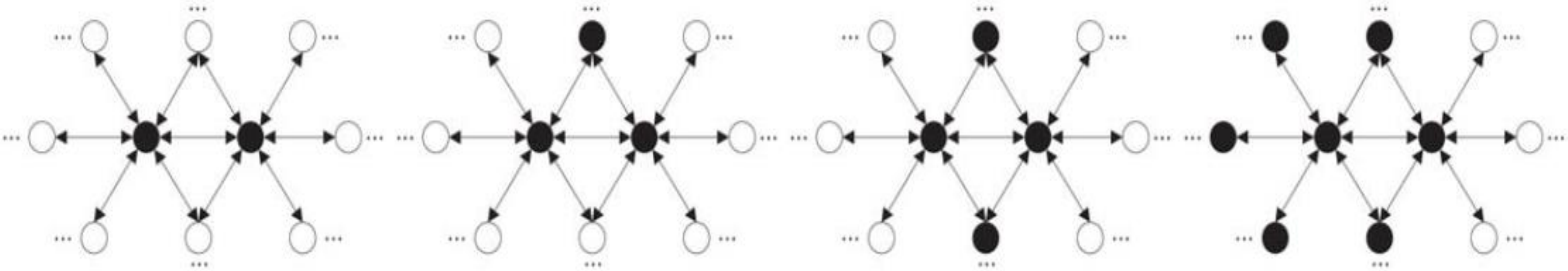
- When attackers collude, outlier detection methods become less effective.

- Solution when attackers collude?
  - Hash-based computation verification of neighbor state update.

    - Have to ensure the authenticity and integrity of the updates sent by neighbors.
    - Have to ensure that the update algorithm is followed correctly at the neighbor node.

  - This algorithm for verification works only when at least there is a common honest neighbor between attackers.

Attacks and Defenses on Consensus Algorithm – Defense Solution 4



(a) Pairwise collusion attack (b) Collusion attack with honest common neighbors (c) Collusion attack without honest common neighbor (d) Neighborhood collusion attack

- How it works?
  - Two-phases: Update Commit and Distributed Verification.
  - Update Commit:
    1. Each node receives the initial submissions from the neighbors: $\langle k, ID_h, value \rangle$
    2. After, each node compute and resend an updated submission:

$$\langle k, ID_h, state^{(k)}, H(k \| ID_h \| state^{(k)} \| ID_1 \| d_1^{(k-1)} \| ID_2 \| d_2^{(k-1)} .... \| ID_q \| d_q^{(k-1)}) \rangle, \ \ k > 0$$

Attacks and Defenses on Consensus Algorithm – Defense Solution 4

- Distributed Verification:
  1. Each node disseminates its neighbor's data using authenticated broadcast:

  $$\langle ID_1, d_1^{(k-1)}, ID_2, d_2^{(k-1)}, ..., ID_q, d_q^{(k-1)} \rangle$$

  2. Now each node have the data of TWO-HOP neighbors.
  3. Each node compared the received IDs with the stored IDs from Update Commit Step.
  4. Re-compute each updated state and each hash to verify if its data and common neighbor's data are used in the update step.
  5. If any of point 3. or 4. fails node v will broadcast a $MAC_{K_{v_i}}(ERR, ID_p)$ that will spread to the whole network.

- Simulations show that the network can converges to the true state and remove the effect of covert data inject attack.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- The last and most recent work about defending against SSFD attacks for consensus algorithm is in [27].
  - The network is undirected graph.
  - Detection at nodes is using ED technique.
  - Weighted Average Consensus Algorithm is considered:

Each node assigns the weight to its own data

$$x_i(k+1) = x_i(k) + \frac{\epsilon}{w_i} \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k))$$

In matrix form, $x(k+1) = Wx(k)$ where, $W = I - \epsilon diag(1/w_1, ..., 1/w_N)L$

  - Attack model is considered in the sensing stage and the attacker assumed to know the true system state:
    1. Under H_0: $$\tilde{Y}_i = \begin{cases} Y_i + \Delta_i, & \text{with probability } P_i. \\ Y_i, & \text{with probability } (1 - P_i). \end{cases}$$
    2. Under H_1: $$\tilde{Y}_i = \begin{cases} Y_i - \Delta_i, & \text{with probability } P_i. \\ Y_i, & \text{with probability } (1 - P_i). \end{cases}$$

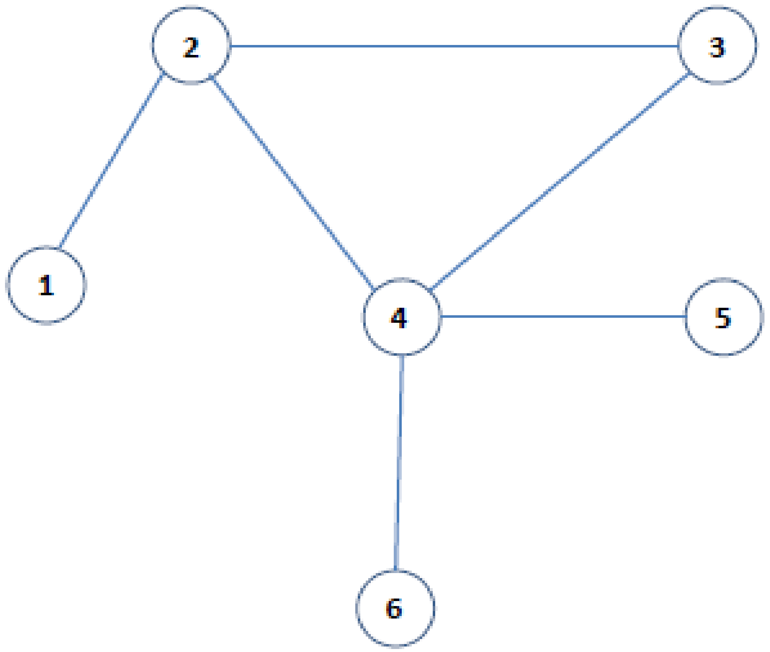
[27] Kailkhura, Bhavya, Swastik Brahma, and Pramod K. Varshney. "Consensus based Detection in the Presence of Data Falsification Attacks." *arXiv preprint arXiv:1504.03413* (2015).

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- The deflection coefficient is used to characterize the security performance: $\mathcal{D}(\Lambda) = \frac{(\mu_1 - \mu_0)}{\sigma^2_{(0)}}$.
- The consensus value is $x^* = \sum_{i=1}^{N} w_i Y_i / \sum_{i=1}^{N} w_i$.
- Steady State performance of consensus under attack:
  - If $i = 1, ..., N_1$ are Byzantines and $i = N_1 + 1, ..., N$ are Honest and $w = [\, \tilde{w}_1, ..., \tilde{w}_{N_1}, w_{N_1+1}, ..., w_N]^T$ then, the condition to make $\mathcal{D}(\Lambda) = 0$ is:

$$\sum_{i=1}^{N_1} \tilde{w}_i (2P_i \Delta_i - \eta_i \sigma_i^2) = \sum_{i=N_1+1}^{N} w_i \eta_i \sigma_i^2$$

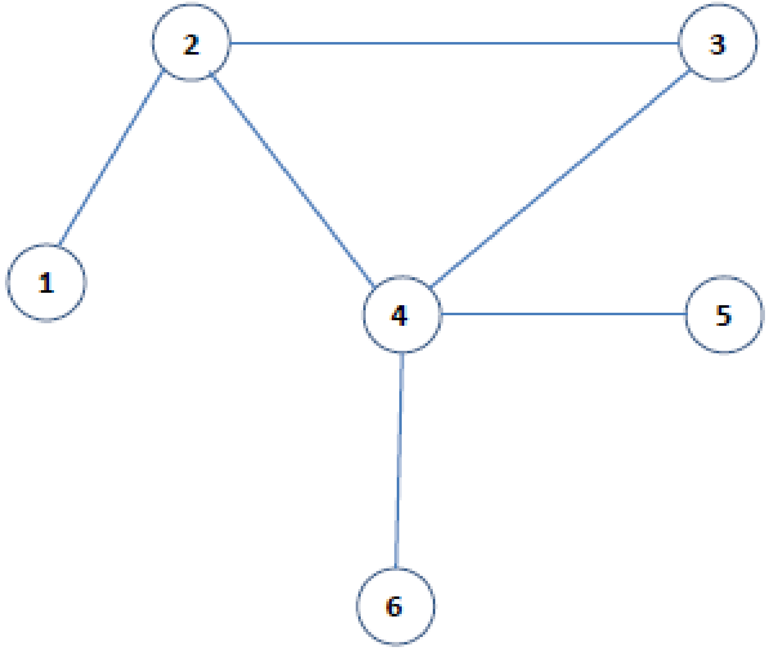
Where, P_i is the attack probability, Δ_i is the attack power, $\tilde{w}_i$ are the tampered weights, η_i is the SNR and w_i are the honests weights.

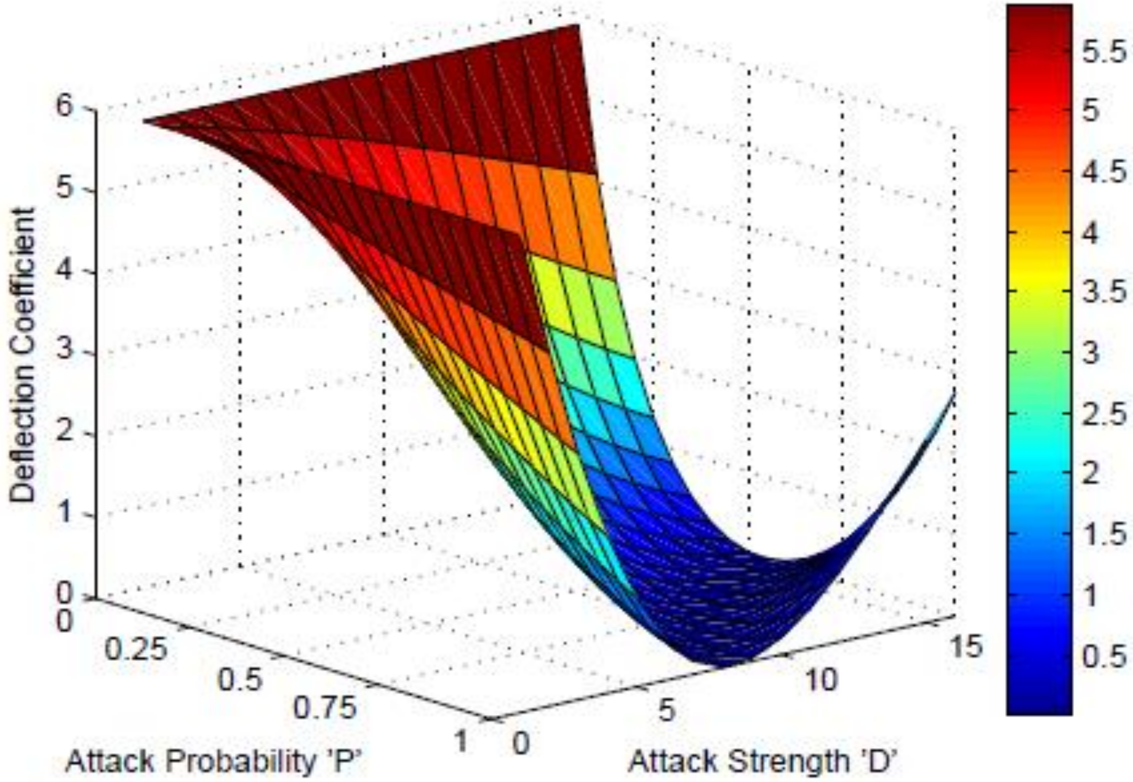


The Network Considered.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- When the attack power, SNR, attack probability and weights are the same for all nodes, the condition reduces to: $\frac{N_1}{N} = \frac{\eta\sigma^2}{2PD}$.



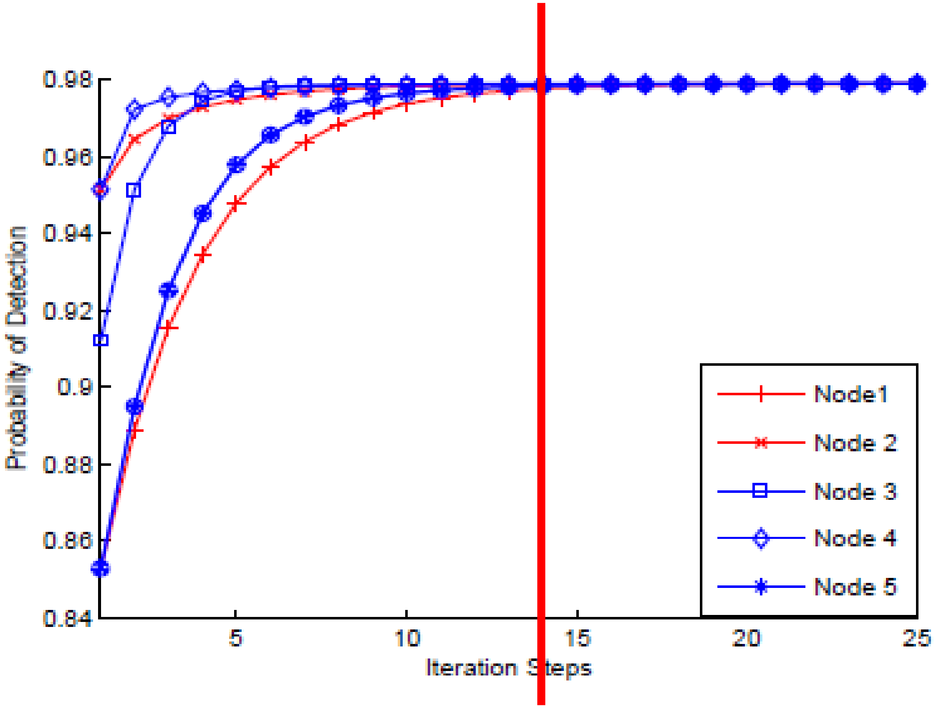


Nodes 1,2 are Byzantines

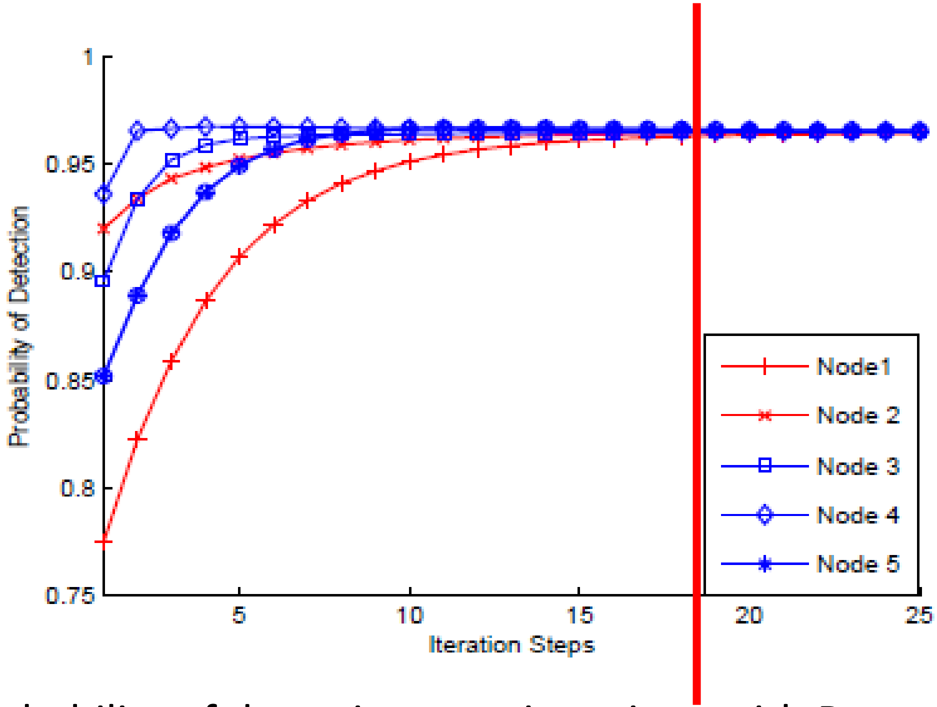
The deflection coefficient can be equal to zero when 2/6 are Byzantines.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- Transient performance under attack:
  - It's characterized by P_d^t and P_{fa}^t at iteration t. A closed form expression is derived.
  - The assumption used is that distribution of Byzantine's data is a Gaussian Mixture under H_k from $\mathcal{N}((\mu_{1k})_i, (\sigma_{1k}^2)_i)$ with probability $(1-P)$ and $\mathcal{N}((\mu_{2k})_i, (\sigma_{2k}^2)_i)$ with probability P .

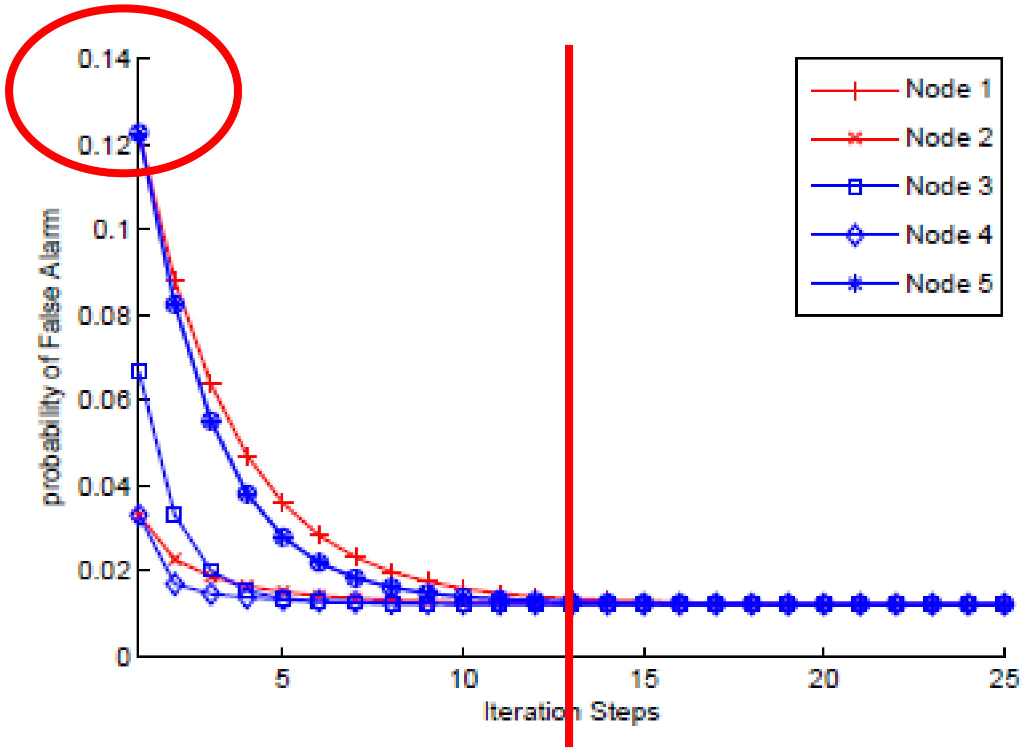


Probability of detection over iterations without Byzantines.

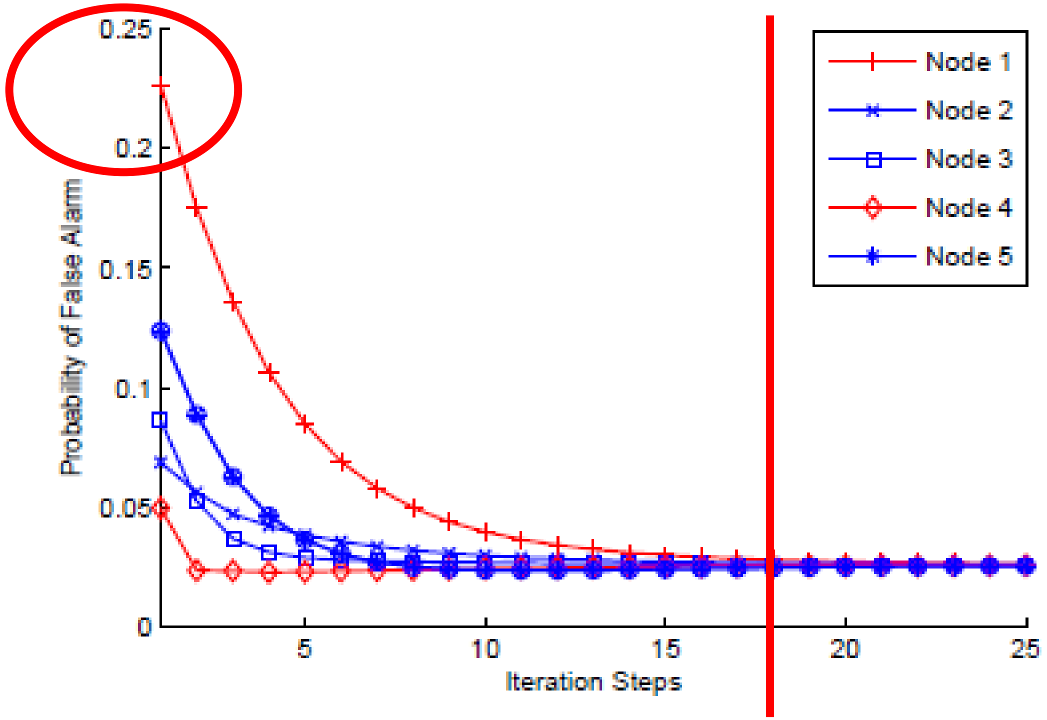


Probability of detection over iterations with Byzantines.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5



Probability of False Alarm over iterations without Byzantines.



Probability of False Alarm over iterations with Byzantines.

So We Need a Solution!

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- Solution: Robust Consensus Algorithm.
  - In the weighted average consensus formula we can see that each node set its own weight w_i .
  - The Byzantine can always set a higher weight to its falsified data.
  - IDEA: assign the weights to node i by the neighbors $\mathcal{N}_i$ rather than i itself.
    - This way is more robust to weight manipulation.
    - The weight matrix $\hat{W}$ that corresponds to this idea is: $\hat{W} = I - \epsilon(T \otimes L)$ where,

$$[T]_{ij} = \begin{cases} \frac{\sum_{j \in \mathcal{N}_i} w_j}{l_{ii}}, & \text{if } i = j. \\ w_j, & \text{otherwise.} \end{cases}$$

    - The proposed consensus algorithm using the idea becomes:

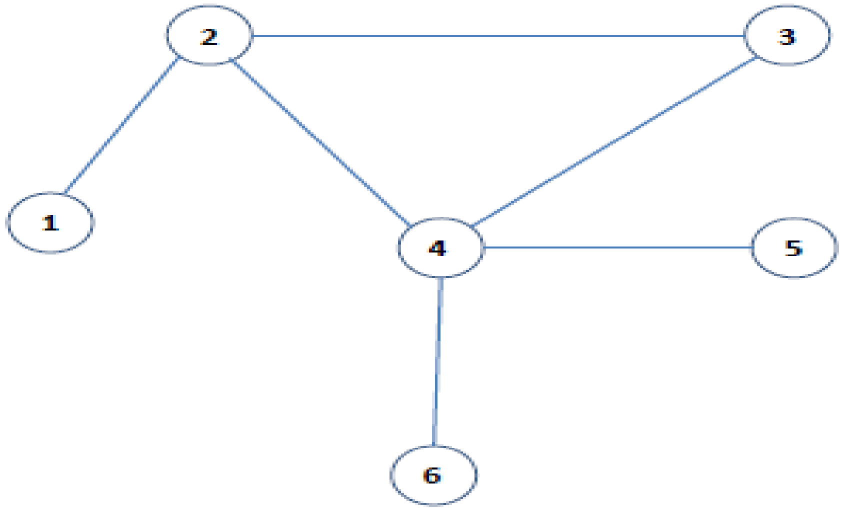
$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} w_j (x_j(k) - x_i(k))$$

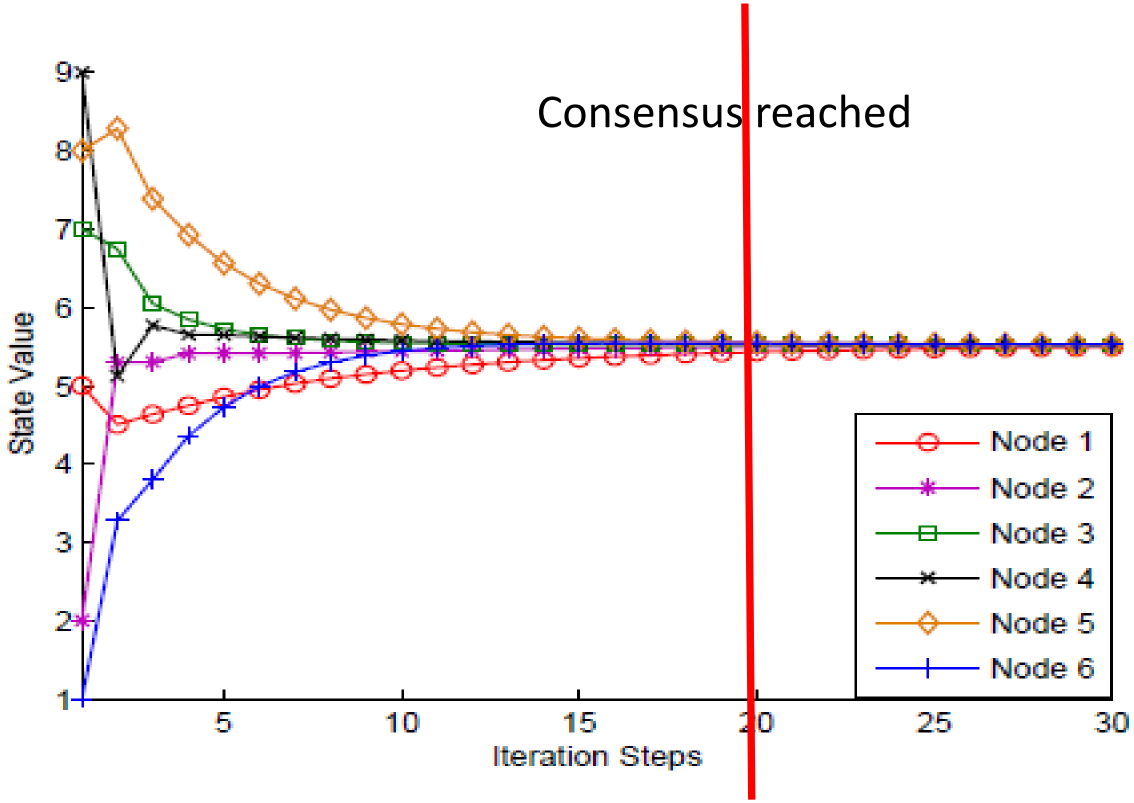
    - If a node does not follow the update using the proper weight → it will be considered as consensus disruption attack and claimed to be solved by [28],[29] but, can be others ...

[28] Sundaram, S.; Hadjicostis, C.N., "Distributed Function Calculation via Linear Iterative Strategies in the Presence of Malicious Agents," in *IEEE Transactions on Automatic Control*, vol.56, no.7, pp.1495-1508, July 2011.
[29] Pasqualetti, Fabio; Bicchi, A.; Bullo, F., "Consensus Computation in Unreliable Networks: A System Theoretic Approach," in *IEEE Transactions on Automatic Control*, vol.57, no.1, pp.90-104, Jan. 2012.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- They show that the modified matrix $\hat{W}$ satisfies the Perron-Forbenius theorem: it's primitive non-negative with left and right eigenvectors u and v, respectively. Then $\lim_{k\to\infty} \hat{W}^k = \frac{vu^T}{v^T u}$ → when this condition hold then, the consensus can be reached asymptotically.





- Example: Proof the convergence of the proposed algorithm.
  1. The step: $\epsilon = 0.3$.
  2. Initial data: $x(0) = [5, 2, 7, 9, 8, 1]^T$
  3. Weight vector: $w = [0.65, 0.55, 0.48, 0.95, 0.93, 0.90]^T$

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

Identification of Byzantines. → Estimation of parameters of Identified Byzantines. → Adaptation of consensus Algorithm(update weights) Using estimated parameters.

- Design of Optimal Weights when the *nodes identities (H or B) are known*:
  - The optimal weight design problem:

$$\max_{\{\delta_i^B\}_i^{N_1}, \{\delta_i^H\}_{i=N_1+1}^{N}} \frac{(\mu_1-\mu_0)^2}{\sigma_{(0)}^2}$$

$$\text{s.t.} \quad \sum_{i=1}^{N_1} \delta_i^B + \sum_{i=N_1+1}^{N} \delta_i^H = 1$$

Centralized optimal weights

  - The solution is: $\delta_i^B = \frac{w_i^B}{\sum_{i=1}^{N_1} w_i^B + \sum_{i=N_1+1}^{N} w_i^H}$ and $\delta_i^H = \frac{w_i^H}{\sum_{i=1}^{N_1} w_i^B + \sum_{i=N_1+1}^{N} w_i^H}$

Where, $w_i^B = \frac{(\eta_i \sigma_i^2 - 2P_i\Delta_i)}{\Delta_i^2 P_i(1-P_i) + 2M\sigma_i^4}$ and $w_i^H = \frac{\eta_i}{2M\sigma_i^2}$.

Decentralized optimal weights: Parameters are unknown to neighbors!!!!!

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- Learn/estimate these parameters for honest and byzantine neighbors → weight assignment will be adaptive.

  - First we need to determine the identities of the nodes.
  - Remember:
    - for honest node under H_k the data is normally distributed with $\mathcal{N}((\mu_{1k})_i, (\sigma^2_{1k})_i)$.
    - For byzantine node under H_k the data is Gaussian mixture of $\mathcal{N}((\mu_{1k})_i, (\sigma^2_{1k})_i)$ with $\alpha^i_1 = 1 - P_i$ and $\mathcal{N}((\mu_{2k})_i, (\sigma^2_{2k})_i)$ with $\alpha^i_2 = P_i$.

  - The problem of identifying the identity of the node becomes a hypothesis testing problem:
    - $I_0(I^i = H)$: the node's data generated from a Gaussian distribution $\mathcal{N}((\mu_{1k})_i, (\sigma^2_{1k})_i)$.
    - $I_1(I^i = B)$: the node's data is generated from Gaussian Mixture.
    - The classification will follow the maximum likelihood: $f(Y_i/I_0) \lessgtr^H_B f(Y_i/I_1)$

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- For honest nodes we need to estimate $((\mu_{1k})_i, (\sigma_{1k})_i^2)$ while for Byzantines $\theta = \{\alpha_j^i, (\mu_{jk})_i, (\sigma_{jk})_i^2\}$.

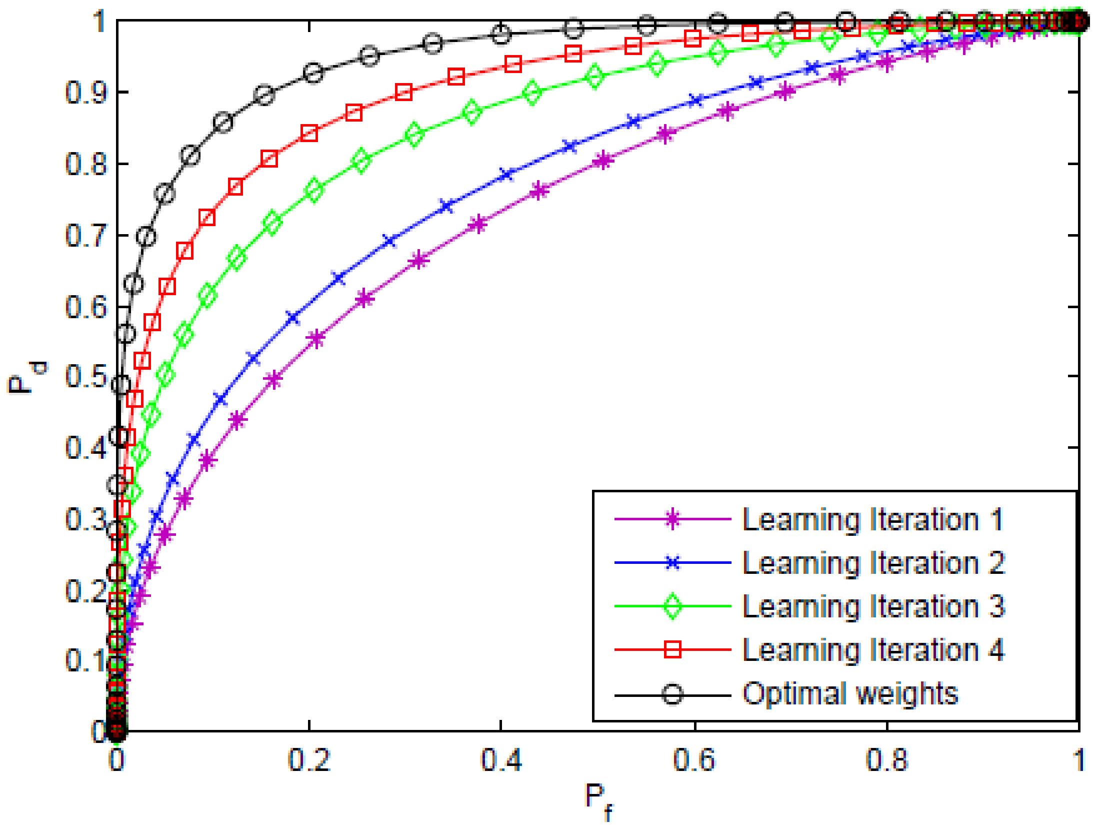
  - How for Honests?
    - ➢ By observing the data over t learning iterations where, each iteration contains D detection slots.
    - ➢ The estimators(nodes) are assumed to know the true hypothesis of the system during these slots.
    - ➢ Counts the number of times each hypothesis occur and apply Maximum Likelihood Estimator to the mean at the previous learning iteration and the data submitted by the neighbor.
    - ➢ The estimation of mean and variance under each hypothesis changes when new samples in the learning iterations come →adaptive.

  - How for Byzantines?
    - ➢ Applying the Expectation-Maximization Algorithm where the hidden variable is α_j^i.

  - The optimal weight after learning iteration t is:
    - ➢ For Honests: $w_i^H(t) = \frac{(\hat{\mu}_{11})_i(t) - (\hat{\mu}_{10})_i(t)}{(\hat{\sigma}_i^2)(t)}$.
    - ➢ For Byzantines: $w_i^B(t) = \frac{\sum_{j=1}^{2} \hat{\alpha}_j^i(t)[(\mu_{j1})_i(t) - (\mu_{j0})_i(t)]}{\hat{\alpha}_1^i(t)\hat{\alpha}_2^i(t)((\mu_{10}(t))_i - (\mu_{20}(t))_i)^2 + (\hat{\alpha}_1^i(t)(\hat{\sigma}_{10})_i^2(t) + \hat{\alpha}_2^i(t)(\hat{\sigma}_{10})_i^2(t))}$.

Attacks and Defenses on Consensus Algorithm – Defense Solution 5

- This shows that with 4 learning iterations the learned weights about the neighbors get close to the optimal weights.



THANK YOU