Identifying Influential Vertices in Boolean Networks through Dynamical Voter Rank

Jiannan Wang, Xiangnan Feng, Ziqiao Yin, Binghui Guo, Zhiming Zheng
School of Mathematics and Systems Science, Beihang University, Beijing, China
Key Laboratory of Mathematics Informatics Behavioral Semantics, Ministry of Education, China
Email: wangjiannan@buaa.edu.cn

***Abstract*—Boolean network model has been applied to describe a series of biological systems. The stability of attractors of certain type of Boolean Networks is considered as one of the key directions to investigate the properties of Boolean network model. Due to the vast heterogeneity in topological and dynamical properties among different vertices, a small fraction of vertices could make a great influence on the dynamics. In this paper, we propose a dynamical voter rank algorithm to identify the influential vertices regarding the stability. In this algorithm, the voting score takes into account not only the topological properties, but also the dynamical properties of the vertices. The dynamical voter rank algorithm is observed to be more efficient than high degree adaptive, eigenvector centrality and Google PageRank algorithms in cases of both real and classical Boolean network model simulation. Our work provides an efficient method to identify the important vertices in Boolean networks, which may help to locate certain kinds of virulence genes.**

***Index Terms*—Boolean network, stability, influential vertices, dynamical voter rank**

I. Introduction

The Boolean network model has been attracting much interest since it was first proposed by Kauffman in 1969 [?]. Due to its capability of illustrating the dynamics of complex systems in a rather concise manner, the Boolean network model has been widely applied to a variety of research fields, such as intracellular genetic regulatory dynamics [?], [?] and spreading process in human social networks [?], [?]. A typical Boolean network usually consists of N vertices which take binary values. Among these vertices there exist M directed edges representing their regulatory relationships. The specific interactions among the vertices are given by a set of Boolean functions, which are often referred to as the update functions. The system evolves at discrete time steps. At each step, the value of each vertex updates according to the update function. During the past decades, researchers have been studying the dynamics of Boolean networks with various topology and various update functions [?], [?], [?], [?]. These contributions have greatly extend the range of application of Boolean network models.

As the system evolves, the Boolean network will eventually goes back to one of its previous states, since the size of the state space is finite. Given fixed update functions, the system will eventually evolves along a cycled orbit in the state space, which is usually called an attractor. The attractors are so important that they are used to explain various phenomena in real world complex system dynamics. For example, in single-celled organisms, the attractors correspond to the different stages of life circle, such as growing, cell division and dying [?]; As for multicellular organisms, such as humans, the attractors explain the variety of types of cells generated from cell differentiation [?]. One big problem on the study of attractors is the stability, which is, the ability to eliminate small perturbations as time evolves. Without the stability of gene regulatory network, the human race could suffer so serious genetic mutations that we are even unlikely to survive so many generations to create the civilization that we enjoy today.

Over the past decades, continuous contributions have been made by various researchers to studying the stability of Boolean networks. In 1985, the annealed approximation was proposed by Derrida *et al.* [?] that gave the stability criterion of their random Boolean network model. This method, which is based on the mean field theory, has been quickly applied to networks with various degree distributions [?], [?] and update functions [?], [?]. However, due to the complexity of real world networks, there are lots of topological properties that is beyond the description of degree distributions, like community structure and so on. It was still rather difficult to deal with the stability problem of a single target network until Pomerance *et al.* proposed the semi-annealed approximation [?] in 2009. The semi-annealed approximation takes into account the adjacent matrix of the network, so it enables us to deal with the stability of Boolean networks with arbitrary topology. With this method, researchers are finally able to study a series of Boolean network models constructed from real biological systems [?], [?]. Till today, the stability of Boolean networks still attracts the attention of many researchers, as evidence illustrates that this problem has a close relationship with certain kinds of cancer [?].

Among the research articles studying the stability of Boolean networks, most of them focus on the macroscopical criterion that determines the stability with the topological and dynamical properties. However, due to the vast heterogeneity of these properties among different vertices in real complex systems, it is more than likely that the influence of each vertex is quite different and a small fraction of them is enough to make a disproportionate effect on the stability of the whole system. In this paper, we propose the dynamic voter rank algorithm through which we can identify the highly influential vertices regarding the stability of Boolean networks. Unlike former metrics such as high degree [**?**], eigenvector centrality [**?**] and Google PageRank [**?**], the proposed algorithm takes into account not only the topological properties, but also the dynamical properties of vertices with respect to stability. To verify the effectiveness of our algorithm, we perform simulations on several different Boolean networks, including two classical toy models and a real network constructed according to data obtained from a survey. Results show that our algorithm outperforms the traditional benchmark metrics. What's more, our algorithm also has superior computational efficiency.

II. Stability of Boolean Networks

In this section, we propose a mathematical framework of Boolean network model and its stability. A Boolean network consists of N vertices connected with M directed edges. For each vertex i, its state, often denoted as x_i, can only take one of the following two states: *on* (1) or *off* (0). The interactions among the vertices are given by N Boolean functions $\{f_1, f_2, ...f_N\}$, which are often referred to as the update functions. At a time step t, the state of the network is then represented as an N-dimensional vector $X(t) = (x_1(t), x_2(t), ...x_N(t))$. At the next step, all vertices update their states synchronously according to $X(t)$ and the update functions. For vertex i, given that it has k_i inputs $X_i = (x_{i_1}, x_{i_2}, ...x_{i_{k_i}})$, the update process can be represented as:

$$x_i(t+1) = f_i(x_{i_1}(t), x_{i_2}(t), ...x_{i_{k_i}}(t)) = f_i(X_i(t)). \quad (1)$$

The adjacent matrix A is used to represent the structure of Boolean network, whose elements $A_{ij} = 1$ if there exists an edge from vertex j to i, otherwise $A_{ij} = 0$.

As for the stability of Boolean network, we start by considering two initial states $X(t_0)$ and $\tilde{X}(t_0)$. The Hamming distance between the states is defined as:

$$H(t_0) = \frac{1}{N}\sum_{i=1}^{N} |x_i(t_0) - \tilde{x}_i(t_0)|. \quad (2)$$

The Hamming distance is the fraction of nodes that take different values in $X(t_0)$ and $\tilde{X}(t_0)$, which corresponds to the fraction of genes that are not working normally at the beginning. To define stability, we assume that N is large enough and the initial values are close, i.e., $H(X(t_0), \tilde{X}(t_0)) \ll 1$. Our main concern is the long-time behavior of $H(t)$ as $t \to \infty$. If the Hamming distance $H(t)$ decreases to zero, the network is considered to be stable, which indicates the system eventually goes back to normal after given small perturbation. If it could not converge to 0 and grows to $\mathcal{O}(1)$, the network is considered unstable. The dynamics of an unstable Boolean network can be easily influenced by small perturbation, and it will not go back to normal state spontaneously.

III. Dynamical Voter Rank Algorithm

In this section, we discuss the dynamical voter rank algorithm to identify the influential vertices regarding the stability of Boolean networks. In previous research articles, researchers mainly focus on the criterion that decides whether a Boolean network is stable or not. In a stable Boolean network, small perturbation tends to vanish spontaneously with time, while in an unstable network, it could make remarkable influence on the dynamics of the whole system. Concerning the vast heterogeneity in topological and dynamical properties among different vertices in Boolean networks, we can not help to think of the problem of identifying the most influential vertices regarding stability. By controlling these influential vertices, we are able to make a great influence on the dynamics of the network at a low expense. Here by controlling we mean to keep the vertex unaffected from all possible perturbation. Let us suppose that we are able to control all vertices in the network, in this case it is easy to achieve full influence that is enough to stabilize any Boolean network. But what if we could only control a small fraction of them? It is quite obvious that we certainly could not expect the maximum influence by selecting the targets randomly. In the following passages, we propose our dynamical voter rank algorithm to identify the most influential set of vertices regarding the stability of Boolean networks. With the help of the greedy algorithm, we pick the influential vertices one after another until the desired influence is finally achieved.

To start with, we introduce the concepts of activity and sensitivity proposed by Shmulevich and Kauffman [**?**]. For an update function f_i, we use X_i to represent the set of vertices that input into i. Let us consider one single vertex j in X_i, the activity of variable x_j regarding f_i is defined as:

$$a_j^{f_i} = \frac{1}{2^K}\sum_{X\in\{0,1\}^K} \frac{\partial f_i(X)}{\partial x_j}. \quad (3)$$

The partial derivative of f_i with respect to x_j is $\frac{\partial f_i(X)}{\partial x_j} = f(X_{(j,0)}) \bigoplus f(X_{(j,1)})$, where $\bigoplus$ is addition modulo 2 and $X_{(i,n)} = (x_1, ..., x_{i-1}, n, x_{i+1}, ..., x_K), n = 0, 1$. The sensitivity of f_i is defined as the probability that f_i outputs different values when given different inputs. Specifically, for vertex i, its sensitivity can be represented as:

$$s^{f_i} = \sum_{i=1}^{K} E[\chi[f_i(X \bigoplus e_i) \neq f_i(X)]], \quad (4)$$

where e_i is the unit vector with 1 only in the ith position, and $\chi[A]$ is the indicator function which equals to one if and only if A is true. Assuming that all of the inputs are uniformly distributed, the sensitivity of vertex i is equal to the sum of the activities of its input vertices:

$$s^{f_i} = \sum_{j=1}^{K} a_j^{f_i}. \quad (5)$$

In our dynamical voter rank algorithm, every vertex i is attached with a tuple $(\overrightarrow{\mu_i}, \nu_i)$. $\overrightarrow{\mu_i}$ is an $N-$ dimensional voting vector and its elements μ_i^j encodes the activity of vertex i regarding the update function of vertex j, which is

$$\mu_i^j = a_i^{f_j}. \quad (6)$$

ν_i is the voting score of vertex i, which is defined as the sum of the activities of vertex i over all the update functions:

$$\nu_i = \sum_{j=1}^{N} \mu_i^j. \tag{7}$$

The voting score ν_i stands for the maximum possible impact of vertex i regarding the dynamics of the Boolean network, which we use to identify its influence. To achieve the maximum influence with a minimum set of controlled vertices, we use the greedy algorithm and pick the influential vertices one after another. Specifically, we follow the following procedures:

step 1: Calculate the initial voting vector μ_i and the voting score ν_i for each vertex.

step 2: Pick the vertex with the highest voting score and remove it from the network. This vertex will not participate in subsequent steps.

step 3: Update the information of the voting vectors and voting scores for all vertices.

step 4: Repeat step 2 and step 3 till the desired influence is achieved.

The dynamical voter rank algorithm is scalable for large networks with a computational complexity $O(N \log N)$. Computing the voting vector and the voting score is equivalent to visiting the neighbors of each vertex, therefore it takes $O(1)$ time to compute the voting score of each vertex. Initially, we have to calculate the voting score of all vertices in the network. However, during later steps, we only have to recalculate for vertices within a $l = 1$ radius from the selected vertex, which scales as $O(N)$. When it comes to selecting the vertex with the highest voting score, we can make use of the data structure of heap that takes $O(\log N)$ time. Therefore, the overall complexity of the dynamical voter rank algorithm is $O(N \log N)$.

IV. Results

In this section, we test the performance of the dynamical voter rank algorithm on three different network, including two classical toy models and a real network. The first one that we consider is Kauffman's original $N-K$ network model, in which all vertices have exactly K inputs randomly selected from the other $N-1$ vertices. The degree distribution of $N-K$ network model does not exhibit much heterogeneity, which is quite different from the reality for most genetic regulatory networks. Considering that real genetic regulatory networks usually have short tailed in-degree distributions and long-tailed out-degree distributions [?], [?], the second network is constructed with Poisson distributed in-degrees and scale-free out-degrees. These two networks are both typical toy models of Boolean network. However, it's not enough if we only test the performance of dynamical voter rank algorithm only on toy models. We have to prove its validity on real networks as well. Therefore, we construct the third network using data from a survey on the social relationship among adolescents [?]. On each of the three networks above, the update functions are given in the form of truth tables. For a given input of f_i, its corresponding output is randomly chosen from $\{0, 1\}$ with probability 0.5.

On each network, we compare the performance of dynamical voter rank algorithm with the following methods: high degree (HD), eigenvector centrality (EC), and Google PageRank (PR), which have been widely applied to detect influential vertices in complex networks. High degree metric, as its name implies, defines the influence of a vertex i with its degree. To get better performances, here we adopt the adaptive version of high degree metric (HDA). After each selection, the degree of each vertex is recalculated. However, as it has been mentioned in various studies, vertices with high degree don't necessarily possess high influence. The eigenvector centrality fixes this problem by considering not only the number of a vertex's neighbours, but also the influence of them, known as the mutual enhancement effect. For each vertex i, its eigenvector centrality EC_i is given by

$$EC_i = \sum_j A_{ij} EC_j. \tag{8}$$

PageRank is another famous algorithm that is used to rank websites in Google search engines and other commercial scenarios. According to the RageRank algorithm, the influence of a webpage is determined by random walking on the network constructed from the relationships of web pages. Mathematically, the PageRank score of vertex i is calculated as

$$PR_i = \sum_j A_{ij} \frac{PR_j}{k_j^{out}}. \tag{9}$$

To derive the influence of the selected controlled vertices, we observe the behavior of the average Hamming distance $\langle H \rangle$. To start with, we create a Boolean network with fixed topology and fixed update functions. Then we calculate the initial value of $\langle H \rangle$. After that, we select and control one top influential vertex and recalculate $\langle H \rangle$. As we repeat the selection and recalculation steps we obtain a series of values of the average Hamming distance. Let q represent the fraction of controlled vertices. When q increases gradually from 0 to 1, the average Hamming distance will decreases till $\langle H \rangle = 0$, which represents that the network achieves stabilization. Our main concern is the critical point q_c at which the network becomes stable. If an algorithm enables us to achieve stabilization with a smaller set of vertices controlled, we can conclude that this algorithm outperforms the others by identifying a set of more influential vertices.

As for the calculation of the Hamming distance $\langle H \rangle$, we take it as the average of 100 initial values. For each initial value, H is calculated through the following procedure. First, we randomly generate an initial values $X(t)$ and evolve it according to update functions till $t_0 = 100$, when we expect it to have completed any transient behaviors. Next, we chose a small fraction ($\varepsilon = 0.01$) of its components and flip their states to create a perturbed value $\tilde{X}(t_0)$. In other words, $\tilde{x_i}(t_0) = 1 - x_i(t_0)$ if vertex i is perturbed, otherwise $\tilde{x_i}(t_0) = x_i(t_0)$. The initial Hamming distance is then $H(X(t_0), \tilde{X}(t_0)) = 0.01$. Finally, we take $X(t_0)$ and $\tilde{X}(t_0)$ as the initial values and evolve both of them in parallel. Here we stress that for vertices that has been controlled, their states in both orbits are always the same, since they are immune to any possible perturbations. Our main interest lies on the long-time behavior of H, which is calculated by averaging $H(X(t), \tilde{X(t)})$ from *t=400* to *t=500*. This whole procedure is repeated for 100 times and $\langle H \rangle$ is the average of H over all initial values.

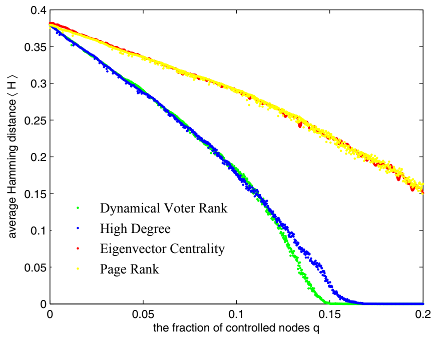

Fig. 1. Normalized average Hamming distance $\langle H \rangle$ plotted against the fraction of controlled vertices q in $N-K$ network. The network consists of 5000 vertices and its average degree equals to 3. The performances of dynamical voter rank, high degree, eigenvector centrality and PageRank are respectively represented in green, blue, red and yellow.

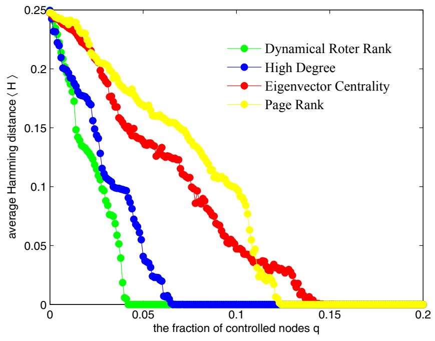

Fig. 2. Normalized average Hamming distance $\langle H \rangle$ plotted against the fraction of controlled vertices q in a heterogeneous network. The networks is constructed according to the configuration model. The in-degrees follow Poisson distribution and the out-degrees are scale-free. The network consists of 1000 vertices and its average degree equals to 3. The performances of dynamical voter rank, high degree,eigenvector centrality and PageRank are respectively represented in green, blue, red and yellow.

In Fig.**??** we plot the average Hamming distance as a function of the fraction of controlled vertices q 1 in Kauffman's $N-K$ network model. The network consists of $N=5000$ vertices and its average degree $K=3$. For each algorithm, the average Hamming distance $\langle H \rangle$ decreases with the increase of the fraction of controllers q. During the process, the dynamical voter rank algorithm outperforms the others and stabilizes the network with the minimal fraction of controllers $q_c^{DVR} \approx 0.15$, followed by high degree metric that stabilizes the network at $q_c^{HD} \approx 0.17$. As for the other two algorithms, their performances are rather close to each other. The average Hamming distance remains $\langle H \rangle \approx 0.15$ even when $q=0.2$ of vertices in the network are under control. It is beyond our expectation that the performance of high degree metric beats those more complex algorithms like eigenvector centrality and PageRank, which usually perform quite well. One possible reason could lie in the difference between the stability of Boolean networks and those problems that these algorithms are designed for. In these algorithms, a vertex usually exhibits higher importance if it is pointed by more vertices with higher importance. However, it's pretty much the opposite when it comes to the stability of Boolean network. In this problem, one vertex enjoys higher influence by pointing to more vertices with higher influence.

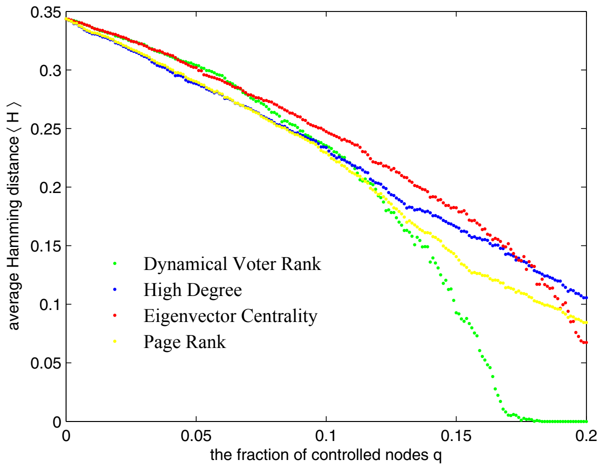

Fig. 3. Normalized average Hamming distance $\langle H \rangle$ plotted against the fraction of controllers q in student social network. The network consists of 1000 vertices and 4175 edges, which is part of the adolescent social network constructed according to the survey. The performances of dynamical voter rank, high degree,eigenvector centrality and PageRank are respectively represented in green, blue, red and yellow.

Fig.**??** shows the performances of the four algorithms on a heterogeneous network. The networks is constructed using the configuration model, which consists of 1000 vertices and its average degree is 3. From Fig.**??** we can see that with the increase of the fraction of controlled vertices q, the network reaches stable region first at $q_c^{DVR} \approx 0.04$ when the dynamical voter rank algorithm is applied. Again, the high degree strategy outperforms the other two algorithms and stabilizes the network at $q_c^{HD} \approx 0.07$. The performances of eigenvector centrality and PageRank are pretty similar, which could not stabilize the network until $q=0.14$. When we compare Fig.**??** and Fig.**??**, we find it interesting that although these two networks have the same average degree, the critical point at which the network becomes stable is quite different even when the same algorithm is applied. In general, an algorithm can achieve stabilization with a much smaller fraction of controlled vertices in the heterogeneous network than in the $N-K$ network model, since it is easier to control the dynamics of the whole system if the network exhibits more topological heterogeneity.

The network structure in Fig.**??** is created from a survey including 2539 students. In the survey, each student was asked to list his five best female and five male friends. Each vertex represents a student and an edge from vertex i to j means that student i chose student j as a friend. In the simulation, we chose part of the network that contains 1000 students among them. In Fig.**??** we find again that dynamical voter rank algorithm outperforms the others. As a matter of fact, the dynamical voter rank is the only algorithm that is able to achieve stabilization at $q_c^{DVR} \approx 0.18$, while for the other algorithms, the network will not be stabilized even when 20 percent of the vertices are under control. This result fully support our theory and our numerical simulations that dynamical voter rank algorithm is able to identify the top influential vertices regarding the stability of Boolean networks with higher accuracy.

V. Conclusion

In this paper, we propose the dynamical voter rank algorithm to identify the influential vertices regarding the stability of Boolean networks. Different from the other benchmark algorithms such as high degree adaptive, eigenvector centrality and Google PageRank, the dynamical voter rank consider not only the topological properties, but also the dynamical properties of vertices, which play an important role in the stability of Boolean networks. To test the performance of dynamical voter rank, we construct two classical Boolean networks and a real-world network on which we compare the performance of our algorithm with other three algorithms: high degree adaptive, eigenvector centrality and PageRank. In all of the three networks, the performance of dynamical voter rank outperforms the others. We also find that for networks with the same average degree, the top influential vertices possess even superior influence in a Boolean network with more heterogeneity. The computational complexity of the dynamical voter rank algorithm is $O(N \log N)$, which allows its application to networks of enormous size. This could contribute to identifying the virulence genes that cause serious inherited disease and to stopping the wide spread of rumor in online social networks.

Acknowledgment

This work was supported by National Natural Science Foundation of China (11401017, 11571028 11671025 and 11290141), Fundamental Research Funds for the Central Universities, Fundamental Research of Civil Aircraft (MJ-F-2012-04).

References

[1] S. A. Kauffman, J. Theor. Biol. 22, 437 (1969).
[2] R. Serra, M. Villani, A. Barbieri, S. A. Kauffman, and A. Colacci, J. Theor. Biol. 265.2, 185 (2010).
[3] M. Villani, A. Barbieri, and R. Serra, PloS ONE 6(3), e17703 (2011).
[4] M.E.J. Newman, Phys. Rev. E 66.1, 016128 (2002).
[5] P. Rämö, S. A. Kauffman, J. Kesseli, and O. Yli-Harja, Physica (Amsterdam) 227D, 100 (2007).
[6] I. Shmulevich, E. R. Dougherty, S. Kim, and W. Zhang, Bioinformatics 18, 261 (2002).
[7] I. Shmulevich, E. R. Dougherty, and W Zhang, Proceedings of the IEEE 90.11, 1778-1792 (2002).
[8] M. Aldana, Physica D 185, 45 (2003).
[9] A. A. Moreira and L. A. Nunes Amaral, Phys. Rev. Lett. 94, 218702 (2005)
[10] B. Derrida, and Y. Pomeau, Europhys. Lett. 1(2), 45 (1986).
[11] D.-S. Lee and H. Rieger, J. Phys. A 41, 415001 (2008).
[12] S. A. Kauffman, C. Peterson, B. Samuelsson, and C. Troein, Proc. Natl. Acad. Sci. 101(49), 17102 (2004).
[13] A. Pomerance, E. Ott, M. Girvan, and W. Losert, Proc. Natl. Acad. Sci. 106(20), 8209 (2009).
[14] E. Cozzo, A. Arenas, and Y. Moreno, Phys. Rev. E 86(3), 036115 (2012).
[15] S. Squires, A. Pomerance, M. Girvan, and E. Ott, Phys. Rev. E 90(2), 022814 (2014).
[16] R. Albert, H. Jeong, and A.-L. Barabsi, Nature (London) 406, 378 (2000).
[17] P. F. Bonacich, J. Math. Sociol. 2, 113 (1972).
[18] L. Page, S. Brin, R. Motwani, and T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web (Stanford InfoLab, 1999).
[19] I. Shmulevich and S. A. Kauffman, Phys. Rev. Lett. 93, 048701 (2004).
[20] N. Guelzim, S. Bottani, P. Bourgine, and F. Kepes, Nat. Genet. 31, 60 (2002).
[21] R. Dobrin, Q. K. Beg, A. L. Barabasi, and Z. N. Oltvai, BMC Bioinformatics 5, 10 (2004).
[22] J. Moody, Soc. Networks 23, 261 (2001).