2015 International Symposium on Nonlinear Theory and its Applications
NOLTA2015, Kowloon, Hong Kong, China, December 1-4, 2015

Identifying Effects of Links in Cascading Failures via an Optimization Method

Huiyun Liu[†‡], Yongxiang Xia[†], and Chi K. Tse[‡]

†Department of Information Science and Electronic Engineering, Zhejiang University
Hangzhou, Zhejiang, 310027, China
‡Department of Electronic and Information Engineering, Hong Kong Polytechnic University
Hunghom, Hong Kong, China
Email: hynal@zju.edu.cn, xiayx@zju.edu.cn, Michael.Tse@polyu.edu.hk

Abstract—Cascading failures have been extensively studied in complex networks. The links in the network can be critical for the propagation of such failures in two aspects. Links which can enhance the propagation can be viewed as negative, while links which can suppress the propagation can be viewed as positive. In this paper, we use an optimization method to identify these two kinds of critical links, which can help people make appropriate measures to defend against cascading failures.

1. Introduction

In modern society, people's life is greatly dependent on the infrastructure networks such as the Internet, power grid and transportation networks. As these networks are becoming more and more complex, it is essential to maintain a high overall efficiency and improve the robustness against destruction. In these critical infrastructures, the breakdown of a single node or link can lead to a load redistribution of other nodes or links, resulting in a failure propagation until there are no more overloaded components. Such behavior is called cascading failures [1, 2]. One typical example is the occurrence of accidents in the power grid of North American in 2003.

With the development of network science in the past decades, much attention has been paid to the study of the topological and dynamical properties of networks. The study of cascade control strategy has been a hot topic in this area [3–5]. There exist mainly two ways to improve the robustness of networks against cascading failures: i) to design efficient load distribution strategies [6], and ii) to modify the underlying network topology [7–9]. In the spreading of cascading failures in the network, links play a vital role in this propagation. According to previous studies, some links are beneficial to defend cascading failures, while some links can enhance the spreading. In many real networks, the time interval between the attack and the propagation of cascades is usually very short. Therefore, it is more reasonable to consider link removal or protection rather than link addition or rewiring [7]. Identification of the critical links having different effects can help to decide which links need to be removed while some others should be protected.

In this paper, we model this problem as a multi-objective combinatorial optimization, aiming at maximizing (or minimizing) the network robustness while minimizing the operating cost. One novel algorithm – non-dominated sorting binary differential evolution algorithm [10] has been applied to solve this problem. This work can provide another perspective to find out the various kinds of critical links in order to help people make appropriate decisions in time when cascading failures occur.

2. Cascading Failure Model

We consider a network with N nodes and L links. Since the traffic or information flow is usually transmitted along the shortest paths in a network, we use the betweenness centrality to measure the load of each node, i.e.,

$$L_i = B_i, i = 1, 2, ..., N. \tag{1}$$

The betweenness centrality B_i of node i is defined as the number of the shortest paths that pass through node i from every source node to every destination node.

The capacity of a node is the maximum load it can handle, and we assume that

$$C_i = (1 + \alpha)L_i(0), i = 1, 2, ..., N \tag{2}$$

where C_i represents the capacity of node i, and $L_i(0)$ is the initial load of node i . The tolerance parameter α is an operation margin allowing the safe operation of the network under potential load perturbations [1, 2].

The occurrence of cascading failures is triggered by an attack. Here we choose the worst case where the node with the highest load is initially removed. When the node is removed, links connected to this node will all be removed and the topology will be changed, leading to a load redistribution of the remaining nodes. If the new load of one node exceeds its capacity, the corresponding node will fail and be removed. As a result of this process, subsequent failures can happen until an equilibrium is finally reached, where there are no more overloaded nodes. To measure the network robustness against cascading failures, the size of the largest connected sub-network (which is called as the giant component) in the stable state is adopted since it can measure the extent of the network disconnection [9]. The

detailed simulation process of the cascading failures can be described as follows:

Step 1: Initially, the node with the highest load is chosen and removed from the network.

Step 2: Apply Floyd's shortest-path algorithm to compute the updated load of each node.

Step 3: Check each node for failure. If the updated load of a node exceeds its capacity, i.e., $L_i > C_i$, the node is regarded as failed and then be removed from the network.

Step 4: Repeat step 2 until the loads of the remaining nodes do not exceed their capacities, which indicates the cessation of the propagation of cascading failures.

Step 5: Compute the size of the giant component of the final network.

3. Formulation of the Multi-objective Combinatorial Optimization Problem

After the network is attacked, links play an important role in the propagation of cascading failures. Some links can help spread the failure [10]. If we can recognize the links with this negative effect and shut them down in time, we can somehow reduce the damage. On the other hand, some existing links can suppress the propagation of cascading failures. More protection should be given to the links with this positive effect to avoid being attacked again. When actions are taken, the cost should be considered. Gaining maximum benefit with the lowest cost is what we are trying to achieve.

In this work, we formulate two multi-objective combinatorial optimization problems to identify the critical links with the two kinds of effects. Two objectives are taken into consideration: one is the network robustness, the other is the operation cost. The first problem is to identify the links that can enhance the cascading failure, which is formulated as follows,

$$\max_x R(x)\ ,\ \min_x C(x) \tag{3}$$

where $x = [x_1, x_2, ..., x_V]$ is the V dimensional binary variable, i.e., $x_k \in \{0, 1\}, k = 1, 2, ..., V$, and $x_k = 1$ represents removing the link k and remain it in the network otherwise. The network robustness function $R(x)$ may be an implicit function, and here is computed as the size of the giant component. The cost function $C(x)$ represents the cost dealing with link removal. In this work, we take an equal cost for removing each link, which means $C(x) = \sum_k x_k$. When maximizing the network robustness by removing links, we can identify the critical links that can enhance cascading failures. These links can be efficiently removed to help prevent the damage.

The second problem is to identify the links that can help to suppress cascading failures, which can be formulated as

$$\min_x R(x)\ ,\ \min_x C(x) \tag{4}$$

After removing the critical links that are most effective to defend against cascading failures, the left network is most vulnerable. Therefore, from the view of protection, these links should be protected further to avoid attacks.

Since the variables in our problems are discrete and there are two objectives to be optimized, problems (3) and (4) are multi-objective combinatorial optimizations. In contrast to single-objective optimization, optimal solution of a multi-objective optimization is a set of points which are called Pareto optimality instead of a single global optimal point. A point is said to be Pareto optimal if none of the objective functions can be improved in value without degrading some of other objective values. All Pareto optimal points lie on the boundary of the feasible criterion space, and they are equally good without additional subjective preference information. The corresponding values of the objectives form the Pareto optimal front in the objective functions space [11].

4. Non-dominated Sorting Binary Differential Evolution Algorithm

Because of the complexity of any large-scale multi-objective combinatorial optimization problem, an overwhelming number of multi-objective metaheuristics designed for solving multi-objective combinatorial optimization problems have been proposed. The goal of these optimization algorithms is to guide the search for the solutions in the Pareto optimal set, while at the same time to maintain the diversity to cover the Pareto optimal front well [12].

One latest novel algorithm – non-dominated sorting binary differential evolution (NSBDE) algorithm [10] has been proved to solve the multi-objective combinatorial optimization problems efficiently. This algorithm takes the advantage of two classical algorithms, namely, modified binary differential evolution (MBDE) [13] and non-dominated sorting genetic algorithm-II (NSGA-II) [14]. MBDE is developed to tackle single-objective binary-coded optimization problems, while NSGA-II has the fast non-dominated sorting, ranking and elitism techniques dealing with multi-objective optimization problems. The detailed procedure of NSBDE algorithm is as follows:

Step 1: Initialization of parameters. Define the values of the population size M, the maximum number of generations N_{max}, the crossover rate CR, the scaling factor F and the bandwidth factor b.

Step 2: Generation of initial population and evaluation. Set the generation number $t = 1$ and initialize the population $X_t = \{x_1^t, x_2^t, ..., x_M^t\}$ which contains M binary-valued parameter vectors of length V. Each vector represents a chromosome and forms a candidate solution to the optimization problem. Each element of each vector $x_{ij}^t (i = 1, 2, ..., M, j = 1, 2, ..., V)$ takes a value from the set $\{0, 1\}$ with the equal probability 0.5. The element takes value 1 if the corresponding link is to be removed and 0 otherwise. Then evaluate each of the M chromosomes in the initial population X_t by performing the cascading failure simulation process presented in Section 2. The values

of two objectives can be obtained.

Step 3: Generation of intermediate population. Apply the binary tournament selection operator to the population X_t to generate the trial population $X_t^{'} = \{x_1^{'t}, x_2^{'t}, ..., x_{M/2}^{'t}\}$, which undergoes the evolution operations of mutation and crossover to become an intermediate population $Y_t = \{y_1^t, y_2^t, ..., y_{M/2}^t\}$.

Step 3.1: Mutation. Apply the mutation operator into each binary chromosome of $X_t^{'}$ according to a probability estimation operator which is defined as the following equation:

$$P(x_{ij}^{'t}) = 1/\left(1 + exp\left(\frac{-2b[x_{r1j}^{'t} + F(x_{r2j}^{'t} - x_{r3j}^{'t}) - 0.5]}{1 + 2F}\right)\right) \quad (5)$$

where $j = \{1, 2, ..., V\}$, and $x_{r1j}^{'t}$, $x_{r2j}^{'t}$ and $x_{r3j}^{'t}$ are the elements at the $j-th$ position of three randomly chosen chromosomes $x_{r1}^{'t}$, $x_{r2}^{'t}$ and $x_{r3}^{'t}$ with the indexes $r1 \neq r2 \neq r3 \neq i$. The developed probability estimation operator uses the standard mutation operator to derive the differential information of three parent individuals to construct the probability distribution model of the mutant individual to be bit 1. The bandwidth factor b can tune the range and shape of the probability distribution model, and an appropriate b value can improve the search efficiency and maintain population diversity simultaneously.

Then the corresponding mutant individual is generated as

$$u_{ij}^t = \begin{cases} 1, & \text{if } rand() \leqslant P(x_{ij}^{'t}) \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

where $rand()$ is a uniformly distributed random number within the interval $[0, 1]$.

Step 3.2: Crossover. The crossover operator is used to produce the intermediate population by mixing the target individual and its mutant individual. The commonly used binomial crossover is defined as

$$y_{ij}^t = \begin{cases} u_{ij}^t, & \text{if } rand() \leqslant CR \text{ or } j = randi(V) \\ x_{ij}^{'t}, & \text{otherwise} \end{cases} \quad (7)$$

where $randi(V)$ is a uniform discrete random number in the set $\{1, 2, ..., V\}$.

Step 4: Evaluation. Evaluate each of the M chromosomes in the population Y_t by performing the cascading failure simulation process in Section 2. Obtain the values of the two objectives.

Step 5: Union and sorting. Combine the parent population and the intermediate population to form a union population $R_t = X_t \bigcup Y_t$. Rank the chromosomes in the population R_t based on the fast non-dominated sorting algorithm with respect to the objective values. By this process, we can get the ranked non-dominated fronts $F_1, F_2, ..., F_k$ where F_1 is the best front (with the lowest rank), F_2 is the second best front (with the second lowest rank) and F_k is the least good front (with the highest rank).

Table 1: The parameters of the NSBDE algorithm

	Problem (3)	Problem (4)
M	200	200
V	270	270
CR	0.8	0.8
F	0.2	0.5
b	6	10
N_{max}	2000	2000

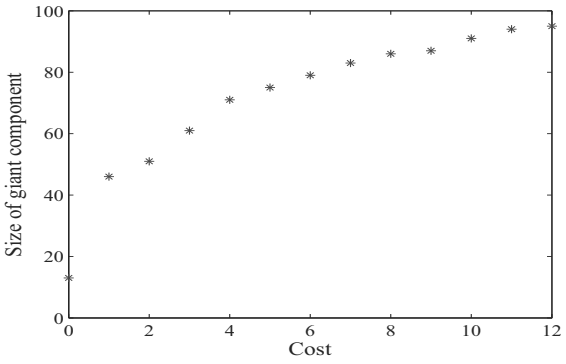

Figure 1: The Pareto front of problem (3)

Step 6: Selection. Select the first NP chromosomes from R_t to produce the new parent population X_{t+1}. Between two solutions with different non-dominated front, we prefer the solution with lower rank. Otherwise, if both solutions belong to the same front (i.e., with the same rank), we prefer the solution that is located in a less crowded region, which can be measured by the crowding distance. A more crowded chromosome has lower priority than a less crowded one. Finally, increase the generation number t by 1. Repeat from step 3 until the generation number reach the maximum.

5. Simulations

Previous results in network science have shown that many real large-scale traffic networks display the scale-free property [15]. Therefore, in order to get insights into on real-world situations, we use the BA scale-free network model for simulation. The network is generated with $N = 150$ nodes and $L = 300$ links. The tolerance parameter in the cascading failure model is set to 0.2. The configurations of the NSBDE parameters used to solve two optimization problems (3) and (4) are shown in Table 1.

Figs. 1 and 2 show the simulation results of problem (3) and (4) respectively. From the results, we notice that if no operation is taken, the size of the giant component when the network is stable is only 13. If one link can be removed, the best improvement of the network robustness can be three times. The worst case to remove links can lead to the decrease of the robustness, making the network more

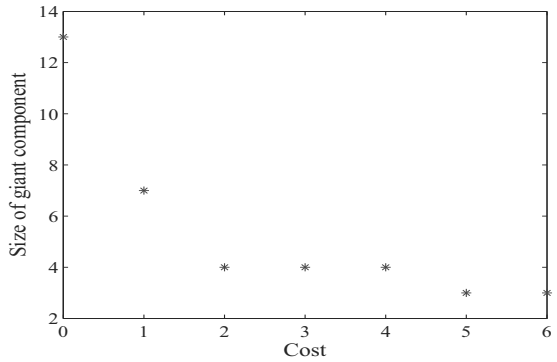

Figure 2: The Pareto front of problem (4)

Table 2: The identified links.

Cost	Indexes of negative links in the solution to (3)	Indexes of positive links in the solution to (4)
1	211	125
2	211,231	125,126
3	155,211,231	19,126,154
4	155,195,211,231	20,78,154,178
5	79,155,195,211,231	78,106,154,215,219
6	139,155,173,195,211,231	20,78,106,154,219,249

vulnerable as shown in Fig. 2. To avoid further attacks in cascading failures, more protection should be given to these links. At the expense of increased cost, the size of the giant component increases by removing the negative links. Decision makers can take the optimal measurement according to the budget. In this simulation, removing one negative link could be a good choice since the improvement is high enough with the least cost. As we can see, the gradient of this point is the highest compared with other points in the figure.

Table 2 illustrates the corresponding solutions to both problems. The indexes given in the table show which links should be removed to get the best (worst) performance. More specifically, the solutions to problem (3) can give suggestions to defend against cascading failures by removing the indexed links. And the solutions to problem (4) give the critical links removing which makes the cascading failures the most severe. Thus, these links should be protected to suppress the propagation of failures.

6. Conclusions

This paper investigates the search for critical links having different effects by formulating two optimization problems. An algorithm for dealing with multi-objective combinatorial optimization problems has been applied to solve the search problems. The results show that removing links with negative effects can help to defend against cascading failures efficiently, while protecting links with positive effects can be vital to avoid large-scale failures during the cascade. This work may be useful for decision makers to take appropriate measures when cascading failures happen.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61174153.

References

[1] A. E. Motter and Y.-C. Lai, "Cascade-based attacks on complex networks," *Phys. Rev. E*, vol. 66, no. 6, p. 065102, 2002.

[2] P. Crucitti, V. Latora, and M. Marchiori, "Model for cascading failures in complex networks," *Phys. Rev. E*, vol. 69, no. 4, p. 045104, 2004.

[3] L. Zhao, K. Park, and Y.-C. Lai, "Attack vulnerability of scale-free networks due to cascading breakdown," *Phys. Rev. E*, vol. 70, no. 3, p. 035101, 2004.

[4] L. Huang, Y.-C. Lai, and G. Chen, "Understanding and preventing cascading breakdown in complex clustered networks," *Phys. Rev. E*, vol. 78, no. 3, p. 036116, 2008.

[5] B. Mirzasoleiman, M. Babaei, M. Jalili, and M. Safari, "Cascaded failures in weighted networks," *Phys. Rev. E*, vol. 84, no. 4, p. 046114, 2011.

[6] W.-X. Wang and G. Chen, "Universal robustness characteristic of weighted networks against cascading failure," *Phys. Rev. E*, vol. 77, no. 2, p. 026101, 2008.

[7] A. E. Motter, "Cascade control and defense in complex networks," *Phys. Rev. Lett.*, vol. 93, no. 9, p. 098701, 2004.

[8] V. H. Louzada, F. Daolio, H. J. Herrmann, and M. Tomassini, "Smart rewiring for network robustness," *Journal of Complex Networks*, vol. 1, no. 2, pp. 150–159, 2013.

[9] X.-B. Cao, C. Hong, W.-B. Du, and J. Zhang, "Improving the network robustness against cascading failures by adding links," *Chaos Solitons Fractals*, vol. 57, pp. 35–40, 2013.

[10] Y.-F. Li, G. Sansavini, and E. Zio, "Non-dominated sorting binary differential evolution for the multi-objective optimization of cascading failures protection in complex networks," *Reliab. Eng. Syst. Saf.*, vol. 111, pp. 195–205, 2013.

[11] K. Miettinen, *Nonlinear multiobjective optimization*. Springer Science & Business Media, 1999, vol. 12.

[12] C. A. C. Coello, C. Dhaenens, and L. Jourdan, "Multi-objective combinatorial optimization: Problematic and context," in *Advances in multi-objective nature inspired computing*. Springer, 2010, pp. 1–21.

[13] L. Wang, X. Fu, Y. Mao, M. I. Menhas, and M. Fei, "A novel modified binary differential evolution algorithm and its applications," *Neurocomputing*, vol. 98, pp. 55–75, 2012.

[14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.

[15] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, no. 1, p. 47, 2002.