







A Metaheuristic Approach for the α -separator Problem

Sergio Pérez-Peló , Jesús Sánchez-Oro  , and Abraham Duarte 

Department of Computer Sciences, Universidad Rey Juan Carlos, Madrid, Spain
{sergio.perez.pelo,jesus.sanchezoro,abraham.duarte}@urjc.es

Abstract. Most of the critical infrastructures can be easily modeled as a network of nodes interconnected among them. If one or more nodes of the network fail, the connectivity of the network can be compromised, to the point of completely disconnecting the network. Additionally, disconnecting the network can result in cascade failures, because the remaining nodes may be overloaded because of heavy traffic in the network. One of the main objectives of an attacker is to isolate the nodes whose removal disconnect the network in minimum size subnetworks. On the contrary, a defender must identify those weak points in order to maintain the network integrity. This work is focused on solving the α separator problem, whose main objective is to find a minimum set of nodes that disconnect a network in isolated subnetworks of size smaller than a given value. The problem is tackled from a metaheuristic point of view, analyzing the solutions given by a Greedy Randomized Adaptive Search Procedure over different network topologies. The results obtained are compared with the best algorithm found in the literature.

Keywords: Alpha-separator · GRASP · Networks · Critical nodes

1 Introduction

From large companies and institutions to individual users, cybersecurity is becoming one of the main concerns of the last years. Most of the Internet users nowadays have their personal and professional information stored in the cloud, including sensitive and private information that cannot be revealed. Therefore, analyzing the risks of the underlying network is one of the main tasks to be accomplished by the owner of the network, in order to guarantee the maximum security to their users. However, network attacks are continuously increasing since attackers obtain benefits if they are able to access to that private information. If a company suffers a cyber attack, the loss of information and privacy usually result in the loss of a vast number of clients, thus implying important economic and social damage [1]. Two of the most common cyber attacks are Denial of Service (DoS) and Distributed Denial of Service (DDoS), which are rather dangerous since they can disable the complete service of an Internet provider, or even produce a cascade failure that affects to a large number of clients [3].

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11315, pp. 336–343, 2018.

https://doi.org/10.1007/978-3-030-03496-2_37

The analysis of the security of the network usually starts identifying its most relevant nodes. The responsible of the security of the network is interested of knowing which nodes should be reinforced with modern and robust security measures in order to reduce the impact of a cyber attack in the network. Meanwhile, the attacker needs to know which are the most vulnerable nodes of the network in order to perform an efficient attack to a small number of nodes, maximizing the damage caused. Therefore, both actors are competing for finding the most relevant nodes of a network in the shortest time.

Networks can be easily modeled as graphs. Let $G = (V, E)$ be an undirected graph that represents a network where the set of nodes is modeled as the set of vertices V , with $|V| = n$, and the set of connections among nodes is modeled as the set of edges E , with $|E| = m$. A separator S for a network G is defined as a set of vertices $S \subseteq V$ which is able to split the network into two or more connected components C_1, C_2, \dots, C_p if nodes in S are removed from the network. In mathematical terms,

$$V \setminus S = C_1 \cup C_2 \dots \cup C_p \forall (u, v) \in E^* \exists C_i : u, v \in C_i \tag{1}$$

where E^* represents the set of edges with no endpoint in the separator S , i.e., $E^* = \{(u, v) \in E : u, v \notin S\}$.

The α -separator problem (α -SP) tackled in this work consists in finding a separator S^* that divides a network G into connected components satisfying just one constraint: the size of each connected component must be smaller than a predefined threshold $\alpha \cdot n$, being α a parameter of the problem. More formally,

$$S^* \leftarrow \arg \min_{S \in \mathbb{S}} |S| \quad : \quad \max_{1 \leq i \leq p} |C_i| \leq \alpha \cdot n \tag{2}$$

considering that the network is divided into p connected components after removing nodes in S .

The number of resulting connected components is not an indicator of the solution quality for the α -separator problem, since the objective function value is evaluated as the number of vertices included in the separator (i.e., $f(S) = |S|$). The α -SP has been proven \mathcal{NP} -hard for general network topologies for $\alpha \geq \frac{2}{3}$ [7]. Additionally, some special network topologies can be solved in polynomial time, including trees and cycles [11]. The polynomial-time algorithms for special networks are not considered in real-life problems since the topology of the network is not known a priori and the network does not usually follow a special structure.

Figure 1(a) shows a network with 9 vertices and 10 edges, while Figs. 1(b) and (c) we present two different solutions for the α -SP. Notice that we consider $\alpha = \frac{2}{3}$, and therefore, the connected components remaining after selecting a separator must contain, at most, $\lceil \frac{2}{3} \cdot 9 \rceil = 6$ vertices. Figure 1(b) presents a feasible solution S_1 where vertices B, C, and I are included in the separator, dividing the network into the following connected components: $C_1 = \{A, D\}$ (2 vertices), $C_2 = \{F, G, H\}$ (3 vertices, and $C_3 = \{E\}$ (1 vertex). The objective function of S_1 is 3, since there are three vertices in the separator. Solution S_2 ,

depicted in Fig. 1(c), includes vertices A and B in the separator, dividing the network in $C_1 = \{D\}$, $C_2 = \{E\}$, and $C_3 = \{C, F, G, H, I\}$, all of them satisfying the size constraint. The objective function value of S_2 is $f(S_2) = 2$, which is smaller than $f(S_1)$, thus being S_2 a better solution than S_1 , since it requires from a small number of nodes to be removed in order to disconnect the network.

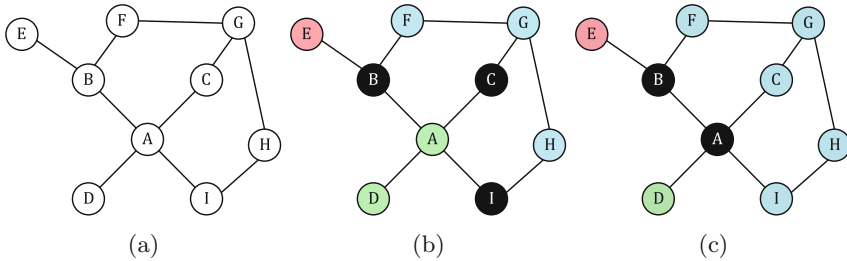


Fig. 1. 1(a) Example of a graph derived from a network, 1(b) a feasible solution with 3 nodes in the separator (B,C, and I), and 1(c) a better solution with 2 nodes in the separator (A and B)

The literature presents several exact and heuristic approaches for solving the α -SP. In particular, trees and cycles can be solved in polynomial time, and also a greedy algorithm with approximation ratio of $\alpha \cdot n + 1$ was presented [11]. The problem of detecting node separators in the Internet Autonomous Systems was tackled by means of a heuristic algorithm [14]. As far as we know, the best heuristic approach is a random walk algorithm based on a Markov Chain Monte Carlo method [10].

The α -SP is equivalent to some well-known problems when considering specific values of α . In particular, if $\alpha = \frac{1}{n}$, then it is equivalent to the minimum vertex cover problem, while if $\alpha = \frac{2}{n}$, the α -SP becomes equivalent to the minimum dissociation set problem. As a consequence, α -SP is a generalization of these problems, which are also \mathcal{NP} -hard [9]. Moreover, the research on α -SP is interesting for different problems related to network theory and resilience assurance [4].

The remaining of the paper is structured as follows: Sect. 2 presents the algorithmic proposal for solving α -SP, Sect. 3 shows the experiments performed to test the proposal and, finally, Sect. 4 draws some conclusions on the research.

2 Algorithmic Proposal

This paper tackles the α -SP from a metaheuristic point of view following the Greedy Randomized Adaptive Search Procedure (GRASP) methodology. GRASP is a metaheuristic formally defined by Feo et al. [8]. We refer the reader to Resende and Ribeiro [13] for a complete survey on this methodology.

GRASP is a trajectory-based metaheuristic that follows a multi-start scheme conformed with two different phases: construction and improvement. The former consist of a greedy, randomized, and adaptive construction of a solution, while the latter is designed for improving the quality of the generated solution through a local search method, or a more complex improvement strategy like Tabu Search, for example. These two stages are applied iteratively until reaching a certain stopping criterion, which is usually a number of construction or a predefined computing time.

2.1 Constructive Method

The α -SP tries to find the minimum number of nodes that must be removed from the network in order to disconnect it, satisfying a constraint with respect to the size of the remaining subgraphs. With this in mind, we can use structural features of a node in a network, such as its degree, or its position in the network, among others, as a selection criterion for the separator.

Following GRASP methodology, the initial solution is constructed by considering a greedy constructive procedure which requires a greedy function value to evaluate the relevance of adding a node to the separator. In this work, we propose a novel approach that leverages the centrality measures traditionally used in the field of social network in order to evaluate the relevance of a node. In particular, we propose closeness centrality [2] as the greedy function for evaluating the nodes of a network. Given a vertex $v \in V$, its closeness is evaluated as:

$$C(v) = \frac{1}{\sum_{u \in V \setminus \{v\}} d(v, u)} \quad (3)$$

where $d(v, u)$ is evaluated as the distance between nodes v and u . In the context of α -SP, we evaluate $d(v, u)$ as the length of the shortest path that connects v and u in the network.

Analyzing the definition of the closeness metric, the larger the closeness, the shorter the distances to other nodes and, therefore, the closer it is to the remaining nodes of the network. Thus, the node with the largest value of closeness in the network is the most reachable one, which usually coincides with the most relevant one. Therefore, removing those nodes with large closeness values will eventually disconnect the network faster.

A completely greedy algorithm always produces the same solution, but it has been shown that considering some randomness in the construction usually leads to better initial solutions [5, 12], which corresponds to the randomized part of the GRASP methodology.

We propose a greedy randomized adaptive constructive procedure which starts by selecting the first vertex v_f at random from the graph. Next up, a Candidate List CL is created with all the vertices in V except v_f , i.e., $v \in V \setminus \{v_f\}$. Then, the method iterates until reaching a feasible solution which satisfies the constraint of the α -SP. Specifically, in each iteration a Restricted Candidate List, RCL , is created with all the vertices whose closeness value is larger than a

predefined threshold μ . The threshold is evaluated as:

$$\mu = g_{\max} - \beta \cdot (g_{\max} - g_{\min}) \quad (4)$$

where g_{\min} and g_{\max} are the smallest and largest closeness values, respectively, among all the vertices in the *CL*. The next vertex to be included in the solution is then randomly selected from the *RCL*. These steps are repeated until reaching a feasible solution. It is worth mentioning that $\beta \in [0, 1]$ is an input parameter of the constructive procedure which controls the randomness of the method. On the one hand, if $\beta = 0$, the *RCL* is constructed with those vertices with maximum closeness value, resulting in a totally greedy algorithm. On the other hand, when $\beta = 1$, all vertices are included in the *RCL*, which corresponds to a completely random procedure. Experiments performed in Sect. 3 will test different values for this parameter, analyzing how it affects to the quality of the generated solutions.

2.2 Local Improvement

The second stage of a GRASP algorithm consists of improving the constructed solution in order to find a local optimum. To achieve this, it is possible to consider a simple local search method or more complex search procedures, even hybridizing GRASP with other complete metaheuristics as Variable Neighborhood Search, or Tabu Search, among others. Notice that the more complex the improvement, the more computationally demanding.

Bearing in mind that the α -SP requires from obtaining solutions in the shortest possible computing time, we have considered a local search method in order to find a local optimum for the solution constructed with the aforementioned constructive procedure.

The local search proposed for the α -SP considers removing two vertices from the solution and include a single one to replace them. Notice that any feasible move performed with this local search will lead us to a better solution, since the objective function will be reduced in one unit. However, the main drawback of this local search is that it is not easy to reach a feasible solution, since the interchange of two vertices by a single one usually violates the constraint regarding the size of the resulting connected components.

For this reason, the proposed local search traverses all available pairs of vertices in the solution, interchanging them for every vertex not considered in the incumbent solution. This exhaustive exploration of the search space lead us to maximize the probability of finding a feasible solution thus improving the results.

Traditionally, local search methods follows two different schemes: best or first improvement. The former explores the complete neighborhood, performing the best move found, while the latter performs the first move that leads to a better solution. Notice that, in the context of the proposed local search, all the feasible moves leads to the same improvement of the objective function value. Because of this, there is no reason for considering a best improvement approach and, therefore, the proposed local search method follows a first improvement scheme.

3 Computational Results

This Section analyzes and discusses the results obtained by the proposed algorithms, comparing them with the best method found in the state of the art. All the algorithms have been implemented in Java 9, and the experiments have been conducted on an Intel Core 2 Duo 2.66 GHz with 4 GB RAM.

In order to have a fair comparison, we have considered the same type of instances that the ones proposed in the best previous method found in the literature [10]. However, due to the impossibility to contact with the previous authors, the instances are not exactly the same. We have generated the set of instances following the Erdős-Rényi model [6], as stated in the previous work. In particular, we have considered a set of 50 instances with nodes ranging from 100 to 200 and edges ranging from 200 to 2000.

The experiments have been divided into two different sections. The preliminary experiments are devoted to find the best value for the β parameter of the GRASP algorithm, while the final experimentation compares the results obtained by the GRASP algorithm with the ones obtained by the best previous method.

In all the experiments we report the same metrics: Avg., the average objective function value; Time (s), the average computing time in seconds; Dev (%), the average deviation with respect to the best solution found in the experiment; and # Best, the number of times that an algorithm reaches the best solution of the experiment.

The preliminary experiment, designed for tuning the β parameter, considers a subset of 20 out of 50 representative instances to avoid overfitting. We have tested the following values for $\beta = \{0.25, 0.50, 0.75, RND\}$, where *RND* value indicates that a random value in the range 0–1 is selected for each constructed solution. Table 1 shows the results obtained when constructing and improving 100 independent solutions, returning the best solution found.

Table 1. Performance of GRASP constructive procedure with different values for the β parameter.

β	Avg.	Time (s)	Dev(%)	#Best
0.25	54.20	298.34	1.58	12
0.5	54.55	289.95	1.93	10
0.75	55.95	308.25	4.31	7
-1.00	54.45	294.75	1.88	10

Analyzing Table 1 we can clearly see that the best results are obtained when considering $\beta = 0.25$, closely followed by $\beta = RND$. Specifically, $\beta = 0.25$ is able to find 12 out of 20 best solutions, and the average deviation of 1.58% indicates that in those instances in which it is not able to reach the best value, the constructive method obtains a high quality solution really close to the best one. The

value of $\beta = 0.25$ indicates that the best results are obtained when introducing a small random part in the constructive procedure, and increasing the randomness of the method results in worse solutions. The worst results are obtained with the largest β value, 0.75, obtaining just 7 out of 20 best solutions with an average deviation of 4.31%. This behavior also confirms that the closeness metric is a good selection as a greedy function value for the constructive procedure.

The final experiment is intended to compare the best variant of GRASP with the best previous method found in the state of the art [10]. Specifically, it consists of a random walk (RW) algorithm with Markov Chain Monte Carlo method. It is worth mentioning that we have not been able to contact to the authors of the previous work neither to obtain the set of instances nor an executable file of the algorithm. Therefore, we have reimplemented the previous algorithm following, in detail, all the steps described in the manuscript. Table 2 shows the results obtained by the proposed algorithm (GRASP with $\beta = 0.25$) and the best previous method found (RW).

Table 2. Comparison of the best variant of RVNS with the best previous method found in the state of the art.

	Avg.	Time (s)	Dev (%)	#Best
RW	71.78	1070.35	27.26	4
GRASP	55.1	299.19	0.00	50

The results obtained shows that our proposal reaches better results than previous work. In particular, GRASP is able to obtain 50 out of 50 best solutions, while RW only reaches the best solution in 4 out of 50 instances. Furthermore, the computing time for GRASP is less than half of the time required by RW. Finally, the average deviation of GRASP is zero, which indicates that in all instances GRASP is able to match the best solution. However, the deviation of RW is higher, indicating that its results are not close to the best solution obtained by the GRASP.

4 Conclusions

This work has proposed a Greedy Randomized Adaptive Search Procedure algorithm for detecting critical nodes in networks. The greedy criterion for this GRASP algorithm is a criterion adapted from the social network field of research, which is named closeness. The GRASP proposal is able to obtain better results than the best previous method found in the literature, which consists of a random walk algorithm in both quality and computing time. This results, supported by non-parametric statistical tests, confirms the superiority of the proposal. The adaptation of a social network metric to the problem under consideration in this work has led us to obtain high quality solutions, which reveals the relevance of the synergy among different fields of research.

Acknowledgements. This work has been partially founded by Ministerio de Economía y Competitividad with grant ref. TIN2015-65460-C2-2-P.

References

1. Andersson, G., et al.: Causes of the 2003 major grid blackouts in North America and Europe, and recommended means to improve system dynamic performance. *IEEE Trans. Pow. Syst.* **20**(4), 1922–1928 (2005)
2. Bavelas, A.: Communication patterns in task-oriented groups. *Acoust. Soc. Am. J.* **22**, 725 (1950)
3. Crucitti, P., Latora, V., Marchiori, M.: Model for cascading failures in complex networks. *Phys. Rev. E* **69**, 045104 (2004)
4. Cuadra, L., Salcedo-Sanz, S., Del Ser, J., Jiménez-Fernández, S., Geem, Z.W.: A critical review of robustness in power grids using complex networks concepts. *Energies* **8**(9), 9211–9265 (2015)
5. Duarte, A., Sánchez-Oro, J., Resende, M.G., Glover, F., Martí, R.: Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization. *Inf. Sci.* **296**, 46–60 (2015)
6. Erdős, P., Rényi, A.: On random graphs. *Publications Mathematicae* **6**, 290 (1959)
7. Feige, U., Mahdian, M.: Finding small balanced separators. In: Kleinberg, J.M. (ed.) *STOC*, pp. 375–384. ACM (2006)
8. Feo, T.A., Resende, M.G., Smith, S.H.: Greedy randomized adaptive search procedure for maximum independent set. *Oper. Res.* **42**(5), 860–878 (1994)
9. Garey, M., Johnson, D.: *Computers and Intractability - A Guide to the Theory of NP-Completeness*. Freeman, San Fransisco (1979)
10. Lee, J., Kwak, J., Lee, H.W., Shroff, N.B.: Finding minimum node separators: a Markov chain Monte Carlo method. In: *13th International Conference on DRCN 2017 - Design of Reliable Communication Networks*, pp. 1–8, March 2017
11. Mohamed-Sidi, M.: *K-Separator Problem (Problème de k-Séparateur)*. Ph.D. thesis, Telecom & Management SudParis, Évry, Essonne, France (2014)
12. Quintana, J.D., Sánchez-Oro, J., Duarte, A.: Efficient greedy randomized adaptive search procedure for the generalized regenerator location problem. *Int. J. Comput. Intell. Syst.* **9**(6), 1016–1027 (2016)
13. Resende, M.G.C., Ribeiro, C.C.: GRASP: greedy randomized adaptive search procedures. In: Burke, E., Kendall, G. (eds.) *Search Methodologies*, pp. 287–312. Springer, Boston (2014). https://doi.org/10.1007/978-1-4614-6940-7_11
14. Wachs, M., Grothoff, C., Thurimella, R.: Partitioning the internet. In: Martinelli, F., Lanet, J.L., Fitzgerald, W.M., Foley, S.N. (eds.) *CRiSIS*, pp. 1–8. IEEE Computer Society (2012)