On Combinatoric Approach to Circumvent Internet Censorship using Decoy Routers

Donghyun Kim*, Glenn R. Frye*, Sung-Sik Kwon*, Hyung Jae Chang†, and Alade O. Tokuta*
* Department of Mathematics and Physics, North Carolina Central University, Durham, NC 27707
E-mail: donghyun.kim@nccu.edu, gfrye@eagles.nccu.edu, skwon@nccu.edu, atokuta@nccu.edu.
† Department of Computer Science and Engineering, Johnson C. Smith University, Charlotte, NC 28216.
E-mail: hchang@jcsu.edu

Abstract—The Internet is often referred to as an information highway since it offers a great mean for the general public to access the almost unlimited amount of information. At the same time, this communication technology poses a great threat to the system in those countries that wish to control the information flow. As a result, the Internet is subject to be under censorship, which is, however, a great privacy issue. Deploying decoy routers is one emerging approach to tackle this issue, since it helps each packet whose destination is a prohibited server to hide its ultimate destination from the censorship authority also known as warden. The recent report by Schuchard et al. shows an existence of an efficient approach to detect and isolate active decoy routers from the public. As a countermeasure, they propose to deploy decoy routers surrounding a target country such that isolating the decoy routers will result in isolating the the country from the rest of the Internet. Based on our observation that this problem in fact is with two opposite optimization goals, minimizing the number of decoy routers and maximizing the confinement rate, we introduce two polynomial time algorithms which handle this deployment problem under the presence of uncooperative ASes. We also conduct simulations to compare and analyze the performance of our algorithms.

Index Terms—Keyword: Internet censorship, decoy routing, decoy router deployment, Internet privacy, Internet routing, BGP, Telex, Cirripede.

I. INTRODUCTION

In this Information Age, the Internet is an important infrastructure which allows the members of a society to access almost unlimited amount of knowledge, and is regarded as an essential utility to increase the public good of the society. However, such a network technology poses a serious threat to the systems of the countries under the censorship and control of information flow. In most cases, censorship of the Internet is achieved by controlling internet (eBGP) routers at the autonomous systems (ASes) under their authority. For instance, by checking the destination address of each packet arriving at the routers and dropping the packet if its destination is one of prohibited web servers, those servers can be efficiently isolated from the citizens in the countries.

From network security point of view, however, the Internet censorship is a serious privacy issue. Decoy routing is a new technology to circumvent Internet censorship and has been widely discussed at many recent top security conferences such as FOCI’11 [1], USENIX Security’11 [2], CCS’11 [3], and CCS’12 [4]. The core idea of decoy routing is for each packet to use a fake destination instead of its actual destination which is a prohibited sever so that the packet can pass the internet routers under censorship, which is frequently referred as the warden in the literature. At the same time, the destination address of the packet is designated in a way that the packet will pass by a collaborative internet router, i.e. decoy router, which will forward the packet to its actual destination based on the secure marker encrypted using the public key of the decoy router.

While the idea of decoy routing is quite promising, the recent report by Schuchard et al. [4] identified an efficient strategy for the warden to detect the set of ASes which actively operates decoy routers. As a result, the warden can efficiently create a blacklist including any suspicious ASes and the prohibited servers to isolate them from the general public. As a countermeasure, Schuchard et al. suggested placing decoy routers in a way that the result of isolating the ASes operating decoy routers from the network will make the country under the censorship isolated from the majority of the rest of the Internet. Apparently, this idea makes a good sense since these days, almost all countries heavily utilize the Internet for businesses, such an isolation will lead to an almost immeasurable financial damage.

Motivations. Schuchard et al. suspected the problem of deploying a minimum number of decoy routers could be related to a well-known NP-hard problem, the vertex-separator problem (VSP). Therefore, rather than looking for an exact algorithm for this problem, they introduced a heuristic approach. The approach of Schuchard et al. introduces the concept of a “depth $i$ ring”, $R_i$, of transit ASes for each country under the censorship, where an AS is in $R_i$ if its hop distance from any AS in the country is $i$. They found that if one decoy router is deployed per each AS in $R_i$ for any $i$, we can surround the country with decoy routers. However, they were concerned that if all ASes in $R_1$ are too close to the confined country, their operators may not be cooperative to deploy decoy routers for the political reason. Therefore, they suggest to deploy a decoy router per each AS in $R_2$ for each country since $R_3$ cannot achieve effective confinement of the country. We observe two main drawbacks of this approach: (a) If we deploy a decoy router for each AS in $R_2$, the number of decoy routers needed is extremely huge. Also, an attempt to access any server in an AS in $R_1$ must obtain the (implicit) permission of the warden, whose number is not negligible. (b) The approach
of Schuchard et al. implies that all of the operators of ASes in \( R_2 \) will be collaborative, which is not necessarily the case in practice. In fact, any operator in \( R_i \) for any \( i \) may decide not to deploy decoy router. In this case, their approach is not applicable.

Contributions. In this paper, we attempt to find a better way to deploy decoy routers despite the fact that there exist some ASes which will not allow the deployment of decoy routers. In fact, this problem has the following two objectives which are opposite to each other. As a result, this complexity of this problem is very high.

(a) Objective 1: the number of decoy routers deployed should be minimized to reduce the deployment cost.
(b) Objective 2: when the decoy routers are blocked, the number of disconnected ASes from the country under the censorship should be maximized to boost up the effectiveness of the deployment.

In the rest of this paper, we refer this problem as the maximum efficiency decoy router placement problem (MEDRPP). We introduce two different algorithms to deal with MEDRPP. The first algorithm, MEDRPA-BT, is a backtracking algorithm to find a feasible solution of MEDRPP with an emphasis on Objective 2. The second algorithm, MEDRPA-VS, is a combinatorial algorithm that seeks to deploy decoy routers with an emphasis on Objective 1. Through simulations, we compare the performance of the two algorithms under the different ratio of uncooperative ASes in terms of Objectives 1 and 2.

The rest of this paper is organized as follows. Section II and Section III introduce the descriptions of the algorithms for MEDRPP, namely MEDRPA-BT and MEDRPA-VS, respectively. In Section IV, we present our simulation result and discuss about it. Finally, we conclude this paper in Section V.

II. A BACKTRACKING BASED ALGORITHM FOR MEDRPP

In this section, we introduce a heuristic algorithm for MEDRPP, the maximum efficiency decoy router placement algorithm with backtracking (MEDRPA-BT). This algorithm uses the backtracking strategy to solve MEDRPP with an emphasis on Objective 2 in the presence of ASes which do not allow the deployment of decoy routers. Consider a topology of the ASes, which can be represented as an undirected graph (directed edges are ignored) \( G = (V, E) \) with \( c \in V \) representing the ASes in the country under the censorship (see Fig. 1(a)). Starting from \( c \), we begin a pre-order traversal of the graph. If we find an AS which allows to deploy a decoy router, then we deploy decoy router at the AS and return to its parent. We also return to its parent if all of its neighbors except the parent already have a decoy router deployed. Clearly, this algorithm will produce a decoy router placement strategy which will tightly confine \( c \) (see Fig. 1(b)) in a sense that all of the ASes between \( c \) and the ring of decoy routers do not want to participate in the decoy router deployment. This algorithm however, does not necessarily minimize the number of decoy routers (see Fig. 1(c)). Clearly, the running time of this algorithm is \( O(n^2) \), where \( n = |V| \).

III. A COMBINATORIAL ALGORITHM FOR MEDRPP

This section introduces the second algorithm for MEDRPP, namely maximum efficiency decoy router placement algorithm with vertex-separator strategy (MEDRPA-VS), which produces a feasible solution of MEDRPP with Objective 1 as its main objective.

A. Preliminaries on Vertex Separator Problem

Suppose \( G = (V, E, w) \) is a connected node-weighted network graph, where \( V = V(G) \) and \( E = E(G) \) represent the set of vertices and edges, respectively, and \( w : V \to \mathbb{R}^+ \) represents the weight function from \( V \) to a positive real number set \( \mathbb{R}^+ \). Given a subset \( U \subseteq V \), \( G[U] \) represents the subgraph of \( G \) induced by the nodes in \( U \), e.g. \( G = G[V] \).

Definition 1 (Vertex Separator). Given \( G \), a nonempty subset \( W \subseteq V \) is a vertex separator of \( G \) if \( G[V \setminus W] \) consists of two “non-empty” subgraphs \( L \) and \( R \) such that for every pair of nodes \( x \in L \) and \( y \in R \), there exists no path between \( x \) and \( y \) in \( G[V \setminus W] \) (e.g. Fig. 2).
for some positive integer \( b \) and some given condition \([5]\). More formally, VSP can be defined as

\[
\text{minimize } \sum_{v \in V} w(v) \text{ subject to } L(v) \cup R(v) = \{0, 1\},
\]

where the weight of a separator \( S \) is \( \sum_{v \in S} w(v) \). Frequently, \( L \) and \( R \) are called the \textit{shores} of the separator \( W \). In the literature, several variations of VSP have been investigated due to its importance in many practical applications \([6, 7, 8, 9, 10, 11]\). It is well-known that VSP is NP-hard in general. However, several special cases of the problem are known to be polynomial time solvable. Most importantly, Lipton and Tarjan \([8]\) showed that if \( G \) is planar and \( b(n) = \frac{2n}{\sqrt{k}} \), we can compute a separator whose size, \( \max\{|L|, |R|\} \), is bounded by \( 2\sqrt{2n} \) with a linear time.

1) \textbf{Balas and Souza’s idea to solve VSP-S:} In \([5]\), Balas and Souza stated VSP is polynomial time solvable if \( b(n) = n - k \) for some positive integer \( k < n \). To reduce any confusion, let us refer to this special case as VSP-S (VSP-with-special-condition). For this purpose, they first induce a bipartite graph \( G^* = (V^*, (V^*_L, V^*_R), E^*, w) \) from \( G = (V, E, w) \) as follows (e.g. from Fig. 3(a) to Fig. 3(b)).

(a) Prepare two empty node sets \( V^*_L \) and \( V^*_R \) as well as an empty edge set \( E^* \).

(b) For each \( v_i \in V \), add \( v_i \) to \( V^*_L \) and \( v_i \) to \( V^*_R \), and an edge \( e_{(i,j)} \) between \( v_i \) and \( v_j \) to \( E^* \). The weight of \( e_{(i,j)} \) is \( w_{(i,j)} \).

(c) For each edge \( e_{(i,j)} \in E \), if \( v_i \), \( v_j \) in \( V \), add two edges \( e_{(i, \ast)} \) and \( e_{(\ast, j)} \) to \( E^* \).

They claimed the VSP-S instance \( \langle G, b(n) \rangle \) can be solved within \( O(n^3 \cdot n^k) \) time by finding a maximum-weight independent set \( S \) from \( G^* \) such that

\[
\max \left\{ |S \cap V^*_L|, |S \cap V^*_R| \right\} \leq n - k,
\]

but the details were not discussed.

2) \textbf{Computing maximum weight independent set in bipartite graphs:} In this section, we briefly describe the combinatorial algorithm for maximum-weight independent set in bipartite graphs by Fralhing and Faigle \([12]\). Given a node-weighted bipartite graph \( G^* = (V^*, (V^*_L, V^*_R), E^*, w) \), first, induce an edge-weighted graph \( G_F = (V_F = V^*_L \cup V^*_R \cup \{s_t\}, E_F, w_F) \) as follow (e.g. from Fig. 3(b) to Fig. 3(c)).

(a) Copy all nodes in \( V^* \) to \( V_F \), i.e. \( V_F \leftarrow V^* \). Add two new nodes \( s, t \) to \( V_F \).

(b) Copy all edges in \( E^* \) to \( E_F \), i.e. \( E_F \leftarrow E^* \). Add an edge from \( s \) to each node in \( V_F \), such that the node is used to be in \( V^*_L \), to \( E_F \). Add an edge from each node in \( V_F \), such that the node is used to be in \( V^*_R \), to \( E_F \).

(c) For each edge \( e_F \) from \( s \) to \( v \) in \( V_F \), set the edge weight \( w_F(s, v) = w(v) \), i.e. \( w_F(s, v) \leftarrow w(v) \). For each edge from \( u \) in \( V_F \) to \( t \), set the edge weight \( w_F(u, t) = w(u) \), i.e. \( w_F(u, t) \leftarrow w(u) \). For each edge \( (u, v) \in E_F \), set the edge weight \( w_F(u, v) \) to be infinite.

Next, run a maximum flow algorithm such as Flow-Marshall’s algorithm to find a minimum capacity \( (s, t) \)-cut of \( G_F \), which does not include any edge with infinite weight. Then, the nodes in \( G_F \) will be partitioned into two node subsets \( S \) and \( T \), each of which includes \( s \) and \( t \), respectively. Let us denote the minimum capacity \( (s, t) \)-cut of \( G_F \) by \( \langle S, T \rangle \), and \( c(S, T) \) be the capacity of \( \langle S, T \rangle \), which is the sum of the edge weight crossed by the cut. We can obtain an independent set \( M = M(S, T) = (S \cap V^*_L \cup T \cap V^*_R) \) with maximum possible weight. For instance, in Fig. 3(c), \( S = \{v_3\} \) and \( T = \{v_1\} \), and \( M = \{v_1, v_3, v_1, v_3\} \) forms a maximum-weight independent set of the bipartite graph in Fig. 3(b).

B. \textbf{Implementation of Balas and Souza’s idea to solve VSP-S} In this section, we elaborate the details of the strategy to implement Balas and Souza’s idea for VSP-S and show its running time is \( O(n^3 \cdot n^k) \). In their method, given a VSP-S instance \( \langle G, b(n) \rangle \) with \( b(n) \leq n - k \) for some positive integer \( k \leq n \), \( G^* = (V^* = (V^*_L, V^*_R), E^*) \) is induced from \( G = (V, E, w) \) as discussed at the beginning of Section III-A1.

One important thing is that the maximum weight independent set computation algorithm for bipartite graphs introduced in Section III-A2 is not applicable to \( G^* \) induced in Section III-A1 directly. This is because for any node \( v_i \in G \), the weights of \( (s_i, v_i) \) and \( (v_i, t) \) for \( v_i \in V^*_L \) and \( v_i \in V^*_R \) are equal in \( G_F \) as in Fig. 3(c). As a result, when applying the maximum flow algorithm on \( G_F \) in Section III-A2, the minimum cut will always span over all of the edges connecting \( s \) to all nodes in \( V^*_L \) or all of the edges connecting \( t \) to all nodes in \( V^*_R \). This means that one of \( S \) or \( T \) is going to be empty set.
Theorem 1. The running time of our implementation of Balas and Souza’s idea for VSP-S with \( b(n) = n-k \) is \( O(n^3 \cdot n^2 \cdot n^k) \).

Proof: First, as discussed above, there are at most \( O(n^k) \) cases that we need to consider. For each case, we need to
(a) induce \( G^* = (V^*_L, V^*_R, E^*, w) \),
(b) compute a maximum weight independent set in \( G^* \),
(c) extract a feasible solution of VSP-S and check if this satisfies the inequality in Eq. (1).

Clearly, the running time of later process will be dominated by the maximum flow algorithm while setting one of the edges connecting \( s \) to a node in \( V^*_L \) as well as a node in \( V^*_R \) to \( t \) to be infinity in Step (b), which takes \( O(n^3 \cdot n^2) \). As a result, the total running time of our implementation takes \( O(n^3 \cdot n^2 \cdot n^k) \) time.

To solve our implementation of Balas and Souza’s idea for VSP-S, we need to set \( k = 1 \). Furthermore, we will remove the factor \( O(n^2) \) from the running time. As a result, the running time of our algorithm will be \( O(n^4) \) which will be significantly faster than \( O(n^{5+k}) \).

C. VSP-SL: What if not all nodes are eligible for the vertex-separator?

In this section, we introduce a variation of VSP-S, namely VSP-SL (VSP-S-with-limited-choices), in which not all nodes in the graph can be included in the vertex separator. The formal definition of VSP-SL is as follow.

Definition 3 (VSP-SL). Given a connected node-weighted graph \( G = (V, E, w) \), where \( w : V \rightarrow \mathbb{R}^+ \) represents the cost of the node, and a subset \( X \subseteq V \) the goal of VSP-SL is to find non-empty partition \( A, B \) of \( V \) such that
(a) \( W \subseteq X \) is a separator of \( A \) and \( B \) in \( G \), and
(b) \( w(W) = \sum_{v \in W} w(v) \) is minimum.

In the following, we provide a polynomial time algorithm for VSP-SL. The main idea of this algorithm is given a VSP-S instance \((G = (V, E, w), X \subseteq V)\), we induce a new VSP-S instance \((G' = (V', E', w'))\) such that an optimal solution of VSP-S is an optimal solution of VSP-SL. In detail,
(a) for each \( u \in X \), add \( u' \) to \( V' \) such that \( w'(u') = w(u) \),
(b) for each pair \( u' \) and \( v' \), add an edge between \( u' \) and \( v' \) only if there exists a path between \( u \) to \( v \) in \( G \).

Fig. 4 provides an example of this graph induction. Now, we provide an important theorem.

Theorem 2. An optimal solution \( W \) of VSP-S in \( G' \) is equivalent to an optimal solution of VSP-SL in \( G \).

Proof: To complete the proof, we first show that a VSP-S \( W \) of \( G' \) is also a feasible solution of VSP-SL over \( G \). Then, we show the cost of optimal VSP-S \( W \) of \( G' \) is equivalent to the cost of optimal solution of VSP-SL over \( G \). The first claim can be proven by contradiction. Suppose a VSP-S \( W \) of \( G' \) is not a feasible solution of VSP-SL over \( G \). Then, either \( W \not\subseteq X \) or \( X \setminus W \) does not partition \( G \) into two nonempty subset \( A \) and \( B \). Then, VSP-SL becomes an instance of VSP-S with \( b(n) = n - 1 \) except the existence of \( a' \) and \( b' \). This special case can be further eliminated by adding an edge between \( a' \) and \( b' \) in the graph which will be used as an input of the solution of VSP-SL instance. Note that the problem of finding the maximum independent set is equivalent to finding the minimum vertex cover (by taking complement of the result), and Konig’s theorem states that minimum vertex cover in bipartite graphs is equivalent to maximal matching, and that that can be in polynomial time.

Clearly, the graph induction can be done in polynomial time. By Theorem 1, a VSP-S \( S \) can be computed within polynomial time. As a result, VSP-SL can be solved within polynomial time.

D. MEDRPA-VS: A combinatorial algorithm for MEDRPP based on vertex-separator strategy

Now, we introduce the detail of the second algorithm for MEDRPP, namely maximum efficiency decoy router placement algorithm with vertex-separator strategy (MEDRPA-VS),
which will produce a feasible solution of MEDRPP with an emphasis on Objective 1. Largely speaking, MEDRPA-VS is an iterative algorithm to find a valid vertex separator to isolate \( x \), a country under the censorship, within a given graph \( G = (V, E, w) \) by repeatedly increasing \( l \geq i \) and find a valid vertex separator from \( O = \cup_{i \leq l} R_i \). Note that we set \( w : V \rightarrow 1 \) since we are concerned about the number of decoy routers.

First, MEDRPA-VS first computes the hop distance from the center node \( x \) to each node. We also introduce a new node \( y \) and to \( G \), and for each node \( v \) in \( R_i \) connected to some node in \( R_{i+1} \), we connect \( v \) to \( y \). Note that \( y \) means “the rest of the world outside the outermost ring \( R_i \).” To eliminate all nodes representing the ASes which do not want to deploy decoy routers, we apply the graph conversion strategy introduced in Section III-C and induce a new graph. Next, we apply the algorithm for VSP-SL over this induced graph and find a separator which can circumvent \( x \) such that there is no path from \( x \) to \( y \) after the nodes in the separator is eliminated from the induced graph.

We note that in order to make sure that once a separator \( W \) of \( G \) is computed and \( V \) is partitioned into three groups, \( L, W, \) and \( R \), we ensure \( x \in L \) and \( y \in R \) or vice versa as well as \( x, y \notin W \). To guarantee this, we slightly modify the structure of the bipartite graph \( G_F \) which will be induced from \( G \) and the max-flow min-cut algorithm is applied in a way that there is no edge from \( s \) to \( y_1 \) and \( t \) to \( x_1 \) (see Fig. 5). Also, set the weight of edges on \( (s, x_1) \) and \( (y_2, t) \) to be infinity. The min-cut on this modified graph will never include those edges. \( x \) and \( y \) will be always within a different partition of \( G \) divided by the minimum weight separator \( W \). Note that since \( L \) and \( W \) will always include \( x \) and \( y \), respectively, we can eliminate the factor \( O(n^2) \) which comes from the procedure to guarantee that both of \( L \) and \( R \) are non-empty.

This process will continue from \( i = 1 \) by increasing \( i \) by one, each time we fail to find a valid separator \( W \). Clearly, this takes \( O(n^3) = O(n^3) \cdot O(n^{k-1}) \) unit time since it is okay to set \( k = 1 \), i.e. we actually do not consider the size difference between \( L \) and \( R \) as long as they are non-empty.

![Fig. 5. To make sure that the output of MEDRPA-VS can confine the country under censorship, Fig. (b) should be used instead of Fig. 3(d).](image)

Now, we evaluate the performance of our algorithms, BT (MEDRPA-BT) and VS (MEDRPA-VS) as well as compare them with Schuchard et al.’s strategy in terms of Objective 1 and Objective 2 through simulation. We first obtain the Internet topology from [13] in which we consider all ASes rather than just transit ASes like [4]. We consider all of the ever-censored-countries considered by [4]. Given the Internet topology graph and a target country, we first contract all ASes in the target country into a single node and calculate the hop distance from this node to each node in the graph so that we can identify the nodes in each of \( R_1, R_2, \ldots, R_5 \). We assume that all ASes further than 6 hops are always willing to deploy decoy routers.

In the first simulation setting, we vary the percentage of ASes which do not want to deploy decoy routers in \( R_1 \) to \( p = 60\%, 70\%, 80\%, 90\%, \) and 100\% (we refer each of them by Degree A, B, C, D, and E, respectively). For each rate \( p \), we also set the rate of decoy routers which do not want deploy routers in \( R_i \) to \( p(0.8)^i \), so that with higher \( i \), more number of ASes allow to deploy decoy routers. This is reasonable since as an AS is further from the country, it has less chance to be under influence of the warden of the country. Fig. 6 illustrates our simulation result. From the figures, we can observe VS

![Fig. 6. Performance Comparison Between BT (MEDRPA-BT) and VS (MEDRPA-VS).](image)

### IV. Simulation Results and Analysis

- **Table 1:** Performance Comparison Between BT (MEDRPA-BT) and VS (MEDRPA-VS).

<table>
<thead>
<tr>
<th>Country</th>
<th>China</th>
<th>Russia</th>
<th>Australia</th>
<th>Egypt</th>
<th>Iran</th>
<th>Vietnam</th>
<th>France</th>
<th>Turkey</th>
</tr>
</thead>
<tbody>
<tr>
<td>BT</td>
<td>0.963</td>
<td>0.969</td>
<td>0.922</td>
<td>0.995</td>
<td>0.963</td>
<td>0.946</td>
<td>0.960</td>
<td>0.964</td>
</tr>
<tr>
<td>VS</td>
<td>0.945</td>
<td>0.953</td>
<td>0.885</td>
<td>0.986</td>
<td>0.950</td>
<td>0.931</td>
<td>0.951</td>
<td>0.958</td>
</tr>
<tr>
<td>Ratio (BT/VS)</td>
<td>1.009</td>
<td>1.016</td>
<td>1.019</td>
<td>1.007</td>
<td>1.012</td>
<td>1.008</td>
<td>1.016</td>
<td>1.017</td>
</tr>
<tr>
<td>Degree A</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
</tr>
<tr>
<td>Degree B</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
</tr>
<tr>
<td>Degree C</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
</tr>
<tr>
<td>Degree D</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
</tr>
<tr>
<td>Degree E</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
<td>2,2</td>
</tr>
</tbody>
</table>

(a) Performance comparison in terms of Objective 1 (minimizing decoy routers)

(b) Performance comparison in terms of Objective 2 (maximizing confinement)
outperforms BT in terms of Objective 1 and BT outperforms VS in terms of Objective 2, which coincide with their design goals.

In the second simulation setting, we vary the percentage of ASes which do not want to deploy decoy routers in \( R_1 \) to 60%, 70%, 80%, 90%, and 100% (we refer each of them by Degree A, B, C, D, and E, respectively). However, we set the rate of decoy routers which do not want deploy routers in \( R_i \) to 0 for \( i > 1 \). In this way, we can fairly compare the performance of Schuchard et al.’s strategy against our two algorithms. Fig. 7 illustrate our simulation result. From the figures, we can observe that in terms of Objective 1, VS is the best. On the other hand, in terms of Objective 2, BT is the best. From the result, we can learn that if there is any ASes which allow to deploy a decoy router, we are better to deploy.

V. CONCLUSIONS

This paper investigates the decoy router deployment problem introduced by Schuchard et al.’s. We notice that (a) the solution provided by Schuchard et al. has a limitation coming from a strong assumption, and (b) their problem is with two opposite objective. Based on the observation, we introduce two heuristic algorithms for the problem, each of which consider one of two objective as its main optimization goal. Our simulation results show each algorithm outperforms the other in terms of its main optimization goal. Also, both them outperforms Schuchard et al.’s strategy in both objectives. While our algorithms are nicely dealing with this optimization problem, it is theoretically challenging to design a single algorithm to produce optimized solution in terms of both objectives at the same time. As a future work, we plan to study the problem to introduce such an algorithm.

ACKNOWLEDGEMENT

This work was supported in part by US National Science Foundation (NSF) CREST No. HRD-0833184 and by US Army Research Office (ARO) No. W911NF-0810510.

REFERENCES


