Black Holes and Revelations: Using Evolutionary Algorithms to Uncover Vulnerabilities in Disruption-Tolerant Networks

Doina Bucur[1], Giovanni Iacca[2(✉)], Giovanni Squillero[3], and Alberto Tonda[4]

[1] Johann Bernoulli Institute, University of Groningen,
Nijenborgh 9, 9747 AG Groningen, The Netherlands
d.bucur@rug.nl
[2] INCAS[3],
Dr. Nassaulaan 9, 9401 HJ Assen, The Netherlands
giovanniiacca@incas3.eu
[3] Politecnico di Torino,
Corso Duca degli Abruzzi 24, 10129 Torino, Italy
giovanni.squillero@polito.it
[4] INRA UMR 782 GMPA,
1 Avenue Lucien Brétignières, 78850 Thiverval-Grignon, France
alberto.tonda@grignon.inra.fr

Abstract. A challenging aspect in open ad hoc networks is their resilience against malicious agents. This is especially true in complex, urban-scale scenarios where numerous moving agents carry mobile devices that create a peer-to-peer network without authentication. A requirement for the proper functioning of such networks is that all the peers act legitimately, forwarding the needed messages, and concurring to the maintenance of the network connectivity. However, few malicious agents may easily exploit the movement patterns in the network to dramatically reduce its performance. We propose a methodology where an evolutionary algorithm evolves the parameters of different malicious agents, determining their types and mobility patterns in order to minimize the data delivery rate and maximize the latency of communication in the network. As a case study, we consider a fine-grained simulation of a large-scale disruption-tolerant network in the city of Venice. By evolving malicious agents, we uncover situations where even a single attacker can hamper the network performance, and we correlate the performance decay to the number of malicious agents.

Keywords: Disruption-tolerant network · Routing · Evolutionary algorithm

1 Introduction

In a complex, open environment such as a city, pedestrians and motorized vehicles are heavily mobile agents, and their movement is constrained to well-defined paths and streets. Mobile communication devices carried by these agents are the

A.M. Mora and G. Squillero (Eds.): EvoApplications 2015, LNCS 9028, pp. 1–13, 2015.
DOI: 10.1007/978-3-319-16549-3_3

nodes in such an *urban network*. The nodes' communication range and bandwidth are limited by their hardware platform and energy supply. Consider an *ad hoc message-routing protocol* for these agents which, unlike the cellular network, uses no centralized communication infrastructure. This protocol should achieve both a good *data delivery rate* and a low *latency* of data messages sent by any node to any other node, while only using as communication primitive the peer-to-peer transmission of data between any two nodes (when these agents are within communication range and can connect reliably). This class of data-routing protocols form (Delay- or) Disruption-Tolerant Networks (DTNs) [1,2].

As a complete source-to-destination path may not always exist in a DTN, intermediate nodes are required to *store* the data and *wait* for an opportunity to *forward* the message to another node towards its final destination, thus creating a *space-time* communication path. Originally developed for solving routing problems in space missions [3], DTNs have been applied, over the years, also to terrestrial applications. Nowadays, DTNs are mostly used as a communication service in open urban environments [4] and several different DTN routing protocols have been designed.

While much research has been devoted to optimizing the data delivery rate and latency of DTNs, the *security* of DTN protocols is still an open problem. This is a particularly serious issue, since urban DTNs must often remain *open* to all willing participants (i.e., DTNs do not perform agent authentication). In such scenarios, security attacks typically consist in one or more agents in the network having malicious behavior; examples are *black holes*, i.e., agents which route no messages to any other agents, and *data flooders*, i.e., agents which inject an unusually large number of messages into the network. To assess the vulnerabilities of DTN protocols to such attacks, one must first determine how "badly" an attacker can affect realistic DTNs. However, the problem of determining the optimal attack method for a fixed DTN environment (i.e., the city map and the movement pattern of all honest agents) was proven NP-hard [4].

We propose an alternative solution: for a realistic urban DTN, we design a *DTN testing framework* based on evolutionary algorithms (EAs), which can highlight the vulnerabilities of any DTN protocol more effectively than existing methods. As a case study, we focus on the classic First Contact (FC) protocol for routing messages in a DTN [1]. FC adopts a simple logic in which an agent forwards a message to the first agent with which a data connection is formed. In FC, a single copy of each message in the network exists at a time, and it is forwarded until the message reaches its destination. The results of our testing on FC clearly demonstrate the potential of the approach: we found scenarios where even a single attacker, purposely "evolved" by the testing framework to exploit the movement patterns in a large network, can reduce the global data delivery in the network to *half* that of the network with no attackers.

The rest of the paper is organized as follows: the next section summarizes the main results from related literature and positions our work with respect to them. Then, Sect. 3 describes our evolutionary-based methodology; Sect. 4 illustrates the experimental setup and presents the numerical results. Finally, Sect. 5 concludes this work.

2 Related Work

Existing studies [5,6] propose a defense against data-flooding attackers, which consists of embedding into the routing protocol heuristic rules such that agents refuse to forward messages injected by flooders. On the other hand, an effective defense against black holes is more difficult to design, and would require each agent to maintain long-term trust relationships with other agents, thus complicating the protocol logic.

On the other hand, very few studies focused on a thorough empirical analysis of the vulnerabilities in existing DTN protocols. The most relevant work was done by John Burgess et al. [7], who quantified the damages caused by a set of attackers in a given application scenario. In their study, they model the town of Amherst (US) with a sparse bus-and-pedestrian open DTN of 71 nodes. The set of attackers is calculated either randomly or by using a greedy heuristic: attackers are added to the network such that they minimize the number of data connections that can be formed. Then, simulation repetitions are used to obtain an average data delivery rate of the network with attackers present.

The greedy heuristic for node selection was found much more effective than random selection at reducing the data delivery rate. For a protocol similar to FC, the effects of any type of attack were found to be minor: compared to a baseline of 33 % delivery rate without any attackers, only by turning malicious *half* of all the nodes in the network did the network's delivery rate fall to half of the baseline, i.e., a rate of message delivery of 17 %.

From the results, the work understandably draws a very favorable conclusion about the robustness of DTN routing to attacks; however, authors do admit that other DTN scenarios may exist which "cause the DTN to perform extremely poorly even with a small number of attackers. For instance, if node mobility is extremely low, and one node forms a nexus for all routing paths, the DTN will fail to deliver packets after corrupting that node. Similarly, if one attacker can corrupt all nodes by flooding an area with RF noise, the DTN will also fail" [7].

Our results show the exact contrary: even with the ideal conditions of high node mobility and no single-node bottlenecks, a small number of well-located, "strong" attackers executing an unsophisticated attacker logic will lower the delivery rate well under the thresholds found in [7].

3 Proposed Approach

The core idea of this work is to use an evolutionary algorithm to *optimize* the parameters and movement of attackers, with the final goal to disclose vulnerabilities in the DTN. To validate the proposed evolutionary testing framework, we simulate an urban environment composed of a map and a set of moving agents (the network nodes). Depending on their type, agents are constrained to different paths, and have different speeds. The building blocks of their movement patterns are the *points of interest* (POIs) located on the map: agent i randomly chooses a destination p from a set of points of interest P_i; travels to p at a realistic speed on the shortest path; takes a break. Then, it repeats the whole process.

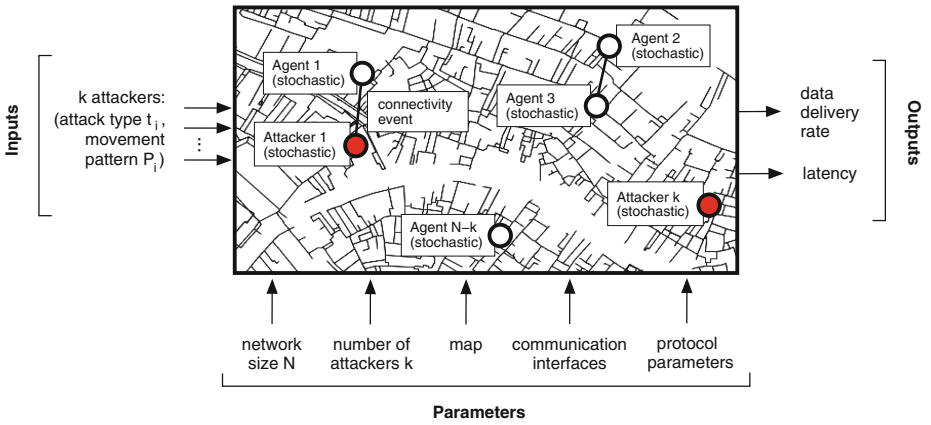

Fig. 1. Representation of the problem: given a set of scenario parameters, calculate the attackers (input) which will produce the most extreme situation in the DTN (output).

We assume two fixed groups of agents: "honest" and "malicious". For any honest agent i, the predetermined set P_i of points of interest includes all the points on the map layer (see also the individual structure description in the next section). For added realism, a small number of these points may be given a higher probability of being selected as next destination. For any malicious agent i, the optimal set P_i of points of interest (i.e. the parameters that would lead to a maximum damage in the network) is instead free to evolve. All honest nodes execute the FC routing protocol, while malicious nodes can act either as data flooders or black holes.

The evolutionary optimization process can then be summarized as follows: given a DTN of N total nodes, a fixed number of malicious nodes $k < N$, and any parameters of the urban environment, find the attacker movement patterns P_i, $i = 1 \ldots k$ which would lower the data delivery rate (DDR) of the DTN the most, while maximizing also its average latency. A graphical representation of this problem is shown in Fig. 1, which visually describes how our testing framework finds those optimal inputs (attackers' parameters) which trigger the most "interesting" (i.e., bad) DTN performance in the outputs, namely lower DDR and higher latency, prioritized in this order.

Evolutionary Framework. We adopt a general-purpose evolutionary framework developed at Politecnico di Torino [8], μGP[1], a toolkit that has been successfully applied to a number of research projects, including the analysis of protocols used in wireless sensor networks [9,10]. Three interesting properties influenced our choice: first, the design of this framework is based on the notion of an *external evaluator*, which simplifies the integration with an external network simulator; secondly, the algorithm available in μGP features a built-in support for multiple fitness functions, that can be evaluated both in a

[1] μGP is available from http://ugp3.sourceforge.net.

lexicographical order and in a multi-objective approach; and finally, the evolutionary engine available in μGP makes use of self-adaptation techniques, greatly limiting the number of parameters that require to be set. The structure of the proposed framework is reported in Fig. 2.

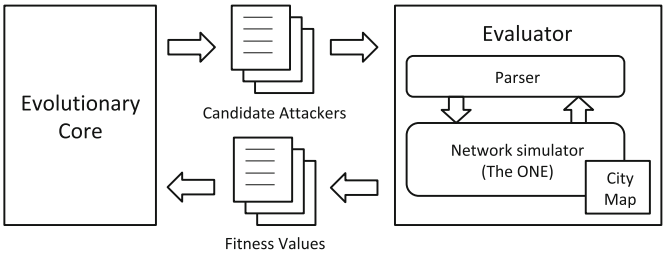

Fig. 2. Our DTN testing framework.

Individual Structure. In the problem under study, an individual represents one or more malicious individuals trying to attack a DTN network. Each attacker (as in Fig. 3) is defined by its type (here, pedestrian or boat) and a set of POIs which effectively define a set of paths on the map.

It is important to notice that the same POIs have different meanings depending on the type of attacker: even if a boat and a pedestrian pass close to the same coordinates, they may reach them using different paths, causing different network disruptions along the way. Also, since most of the POIs are accessible by certain types of attackers only (e.g., a point in open water cannot be reached by a pedestrian, nor one on land by a boat), for each type we overlap a *grid* layer onto the city map layer, and define the path of an attacker of that type as

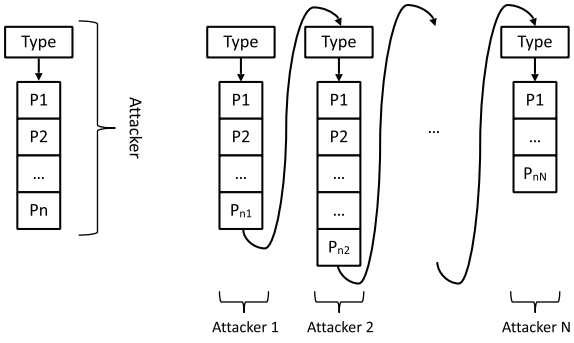

Fig. 3. Individual structure for (left) single and (right) multiple attackers. Multiple attackers are concatenated into a single individual. Each attacker is characterized by a variable number of points of interest and a type (pedestrian, boat) which points to that agent's map layer. The initial numbers of points of interest are drawn from a Gaussian distribution $\mathcal{N}(70, 60^2)$.

a set of grid squares inside that grid. During the simulation, we then map each grid square to the map point *closest* to the square; if a square contains more than one map point, the attacker visits them all.

Fitness Functions. As shown in Fig. 1, the outputs of each simulation (i.e., the fitness functions, in the evolutionary jargon), are:

- (f_1) the data delivery rate, calculated as the percentage of messages originated *only* from honest nodes, and which are delivered successfully;
- (f_2) similarly, the average latency of message deliveries (in seconds).

They are evaluated in lexicographic order: f_1 is minimized, and f_2 is maximized.

Evolutionary Operators. The genetic operators employed by the evolutionary framework include two crossover operators which are able to cross over individuals at the level of their corresponding paths, namely `onePointCrossover` and `twoPointCrossover`, and four mutation operators, namely `insertionMutation`, `removalMutation` and `replacementMutation`, that respectively add, remove or change a single POI in a path, and `singleParameterAlterationMutation`, that changes either a single coordinate in a POI, or the type of attacker.

4 Experimental Evaluation

The approach is validated by coupling μGP with the ad hoc network simulator *The ONE* [11]. As a case study, we consider a DTN with $N = 200$ mobile agents "operating" in the dense, $5\,\mathrm{km}^2$ core of the city of Venice, Italy. For this city, an agent is limited to either of two types: pedestrian, whose movements are confined to the map layer containing only foot paths; boat, whose movements are constrained to the map layer containing only waterways (see Fig. 4, left). Both types of nodes can be either honest or malicious. The number of pedestrians and boats is set, respectively, equal to 150 and 50. In each simulation, a small subset of $k < N$ nodes (of any combination of types, i.e. n pedestrians and $k - n$ boats, $n = 0, \ldots, k$) is turned malicious and evolved by the evolutionary algorithm. The number of malicious nodes is fixed for the duration of the experiment, and all malicious agents perform the same kind of attack. Furthermore, each attacker's type is fixed, as shown by the genetic structure in Fig. 3.

Honest nodes are initially placed randomly on their map layer. The next point of interest visited by each node is then selected randomly from those available on that layer; a small number of map points (3 pedestrian, and 2 waterway locations) situated at main commercial or touristic spots in the city, have a higher probability of selection, at 10 % per map point. Then, the node travels to this next point on the shortest valid path on its map layer. Finally, once the target point of interest is reached, the node randomly selects a new one, and the execution proceeds from that point. This movement model based on shortest-path calculation quickly distributes the nodes non-uniformly on the map. As seen in the snapshot of a sample simulation (Fig. 4, right), preferred paths "emerge" on the map, as in real life. Unlike the honest nodes, attackers select their next

destination *only* from their own list of points of interest, which is generated by the evolutionary framework, as described in the previous section.

Each network is simulated for 12 h (simulated time), starting after a warm-up simulated period of 1000 s. The warm-up helps to remove the transient effects due to the initial random node placement, and allows the movement patterns natural to this large-scale scenario to emerge. Also, to smoothen the fitness landscape and reduce the effect of the random seed on the simulation, we replicate each network simulation 10 times, initialized with different random seeds, and average the measured DDR and latency over the repetitions.

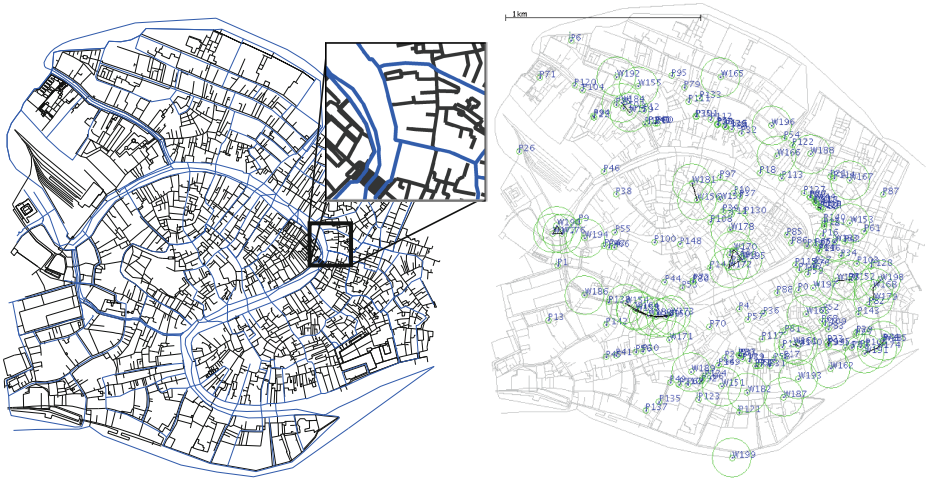

Fig. 4. (**left**) Map of Venice, Italy: a pedestrian layer L_P (drawn in black), and a waterways layer L_W (drawn in blue); the layers only overlap at bridges and boat stops. (**right**) A snapshot of a simulation of shortest-path movement of 200 nodes at a simulation time of 6 h: node identifiers (prefixed by P for pedestrians and W for boats) are drawn in blue, and their communication range in green (Color figure online).

During each simulation, a new data message of size 10 kB is created every 30 s by a random honest node in the network, with another random honest node as destination. All honest nodes have a message buffer of 5 MB, and execute the FC protocol [1], while attackers behave in one of two ways:

- A *black hole* attacker executes another, "passive" protocol logic in which the node offers to route data messages for other nodes, but in effect does nothing, effectively ending the communication path of all messages that reach it.
- A *data flooding* attacker executes the correct FC protocol, but creates new data messages of much larger size, aiming to overload the message buffers of honest nodes. These malicious messages are generated every 3 s, their size is set to 100 kB (a ten-fold increase with respect to a honest message) and the size of the message buffer for honest nodes is decreased five-fold to 1 MB.

These choices ensure that the flooding attack is sufficiently "heavy" to create an interesting fitness landscape for the problem.

Table 1. Simulation settings for all nodes in the DTNs under test

Map settings	Map:	Figure 4 (left), size 2210 m × 2340 m
	Map layers:	L_P (pedestrian paths), L_W (waterways)
	No. of line segments:	4993 in L_P, 362 in L_W
	No. of map points:	6910 in L_P, 1354 in L_W
Simulation settings	Simulation time:	12 h
	Types of nodes:	150 pedestrians (constrained to L_P), 50 boats (constrained to L_W)
Movement model	Next point:	Chosen randomly from a map layer
	Path choice:	Shortest path to the next map point
	Pedestrian speed:	[0.5 . . . 1.5] mps
	Boat speed:	[1.0 . . . 5.0] mps
	Wait time:	[0 . . . 120] s at each destination point
Communication interfaces	Bluetooth:	Range 15 m, speed 250 kBps
	High-speed:	Range 100 m, speed 10 MBps
	Pedestrians use:	Bluetooth
	Boats use:	Bluetooth and High-speed
Data settings	Message created:	Every 30 s (honest node), 3 s (data flooder)
	Message size:	10 kB (honest node), 100 kB (data flooder)
	Message buffer:	5 MB (default), 1 MB (under flooding)
	Message TTL:	5 h

A summary of the simulation configuration parameters is reported in Table 1, while Table 2 shows the values for the parameters of the evolutionary algorithm. The latter were chosen according to simple recommendations from [8].

Results. The proposed DTN testing framework (from Fig. 2) chaining the evolutionary framework with the DTN simulator is run in separate evolutionary experiments for a fixed number of attackers k ranging between 1 and 4, for

Table 2. μGP settings used during the experiments.

Parameter	Description	Value
μ	Population size	30
λ	Number of genetic operators applied at every step	5
τ	Size of the tournament selection	2
§	Stagnation condition (generation)	50

Table 3. The best experiments obtained in this study. The number of evaluations and wall-clock time vary, respectively, because of the stagnation stop condition and due to different load on our computing facility ([16 ÷ 24] available cores).

Fitness functions max. (↑) or min. (↓)	Attacker type	No. of attackers	No. of evaluations	Wall-clock time	Best fitnesses
`DDR` ↓, `Latency` ↑	black holes	1	4645	∼29 h	26.17 %, 5142
		2	9074	∼63 h	16.66 %, 3499
		3	9418	∼130 h	12.73 %, 2794
		4	10285	∼94 h	10.72 %, 2494
`DDR` ↓, `Latency` ↑	data flooder	1	5739	∼ 98h	21.89 %, 2866
		2	5813	∼95 h	18.43 %, 3752
		3	3743	∼47 h	13.66 %, 3058
		4	7859	∼93 h	12.11 %, 2804

each attack methodology (black hole or data flooding). Each of the 8 resulting experiments is run twice, with different random seeds. We summarize the configurations and best fitness of each experiment in Table 3 and Fig. 5. In the figure, we also show the comparison of the best f_1 obtained for each of the 8 experimental settings ($k \times$ attack method) with two baseline scenarios:

- A scenario in which all 200 nodes are honest (i.e., $k = 0$). In this case, given a sample of 30 simulations, the average f_1 (the delivery rate for the First Contact protocol) is 47.57 % with a standard deviation of 1.15 %.
- A scenario in which k *random* attackers of the same attack method exist in the network. Their lists of points of interest are generated randomly. This baseline is calculated as the average and standard deviation of f_1 over 30 simulations (with different random seeds).

With reference to Fig. 5, it can be seen that, in each case, the evolutionary experiment found that single attackers (of either type) exist which lower the delivery rate to half that of the $k = 0$ baseline; also, there exist pairs of two attackers (also of either type) which lower the delivery rate to a third of the $k = 0$ baseline. The evolution of groups of attackers performs efficiently, lowering by approximately 10 % the delivery rate with respect to the baseline of k random attackers with the same characteristics.

The progress of the evolutionary process for two of our experiments (with $k = 1$ and different types of attackers) is shown in Fig. 6. Each experiment is shown from the initial, random population until the stagnation condition is met. For each generation in the sequence, both fitnesses are plotted: both the best and worst data delivery rate (f_1), which is minimized first, and the best message latency (f_2), which is then maximized. It is the data delivery rate which has the most interesting evolution here: its smooth progress is likely sign of a generally smooth landscape for f_1. Also, we can observe that in both cases the latency positively correlates to the minimization of DDR, despite evolutionary attempts

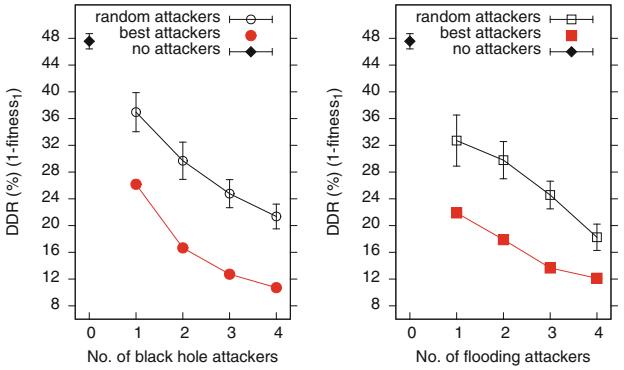

Fig. 5. Baseline values for f_1 (avg. $\pm$ std.dev.) and best values for f_1 obtained using our DTN testing framework for each number of attackers k and method of attack: black hole (**left**), and data flooding (**right**). $k = 0$ corresponds to the baseline scenario with no attackers.

to maximize it. This can be explained as follows. In the black hole case, the latency of the few messages that are not intercepted by the attacker (and can reach their destination) is not affected. On the other hand, in the flooding case, the large number of messages injected by the flooder (sent to random honest nodes) creates a network congestion which lets very few messages reach their destination, likely only those destined to close neighbors; this explains why the average latency of delivered messages is significantly lower with a single flooder than with a single black hole attacker.

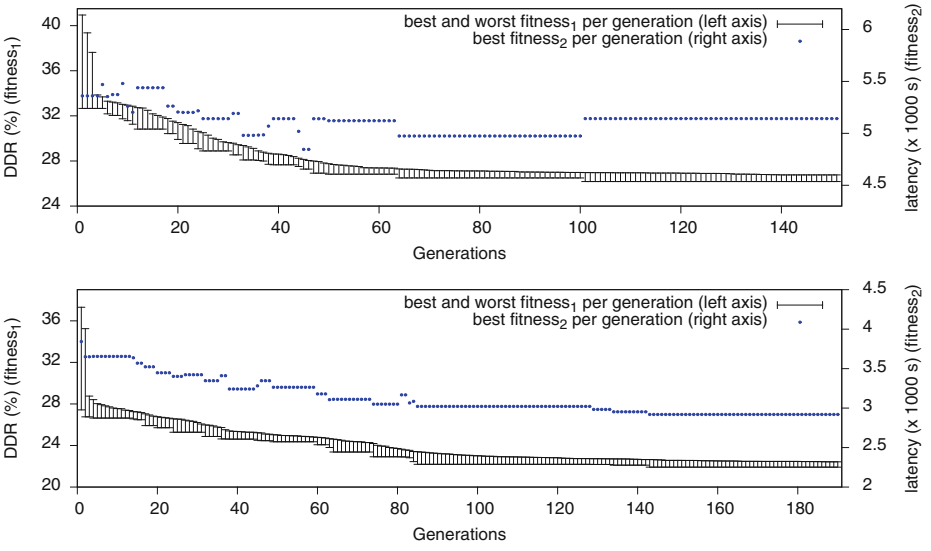

Fig. 6. Evolving a single black hole attacker (**top**) and data flooder (**bottom**): the progress of the two fitness functions through generations.

Discussion of Results: Significance and Limitations. The numerical test results (summarized in Fig. 5) clearly show that evolved attackers are *more efficient* than random attackers. Also, the drop in delivery rate experienced in that network which includes the optimized attackers is unexpectedly large, considering that only few nodes are malicious out of a large network of 200 nodes.

Fig. 7. (**left**) The best single black hole attacker. **(right)** The best single data flooding attacker. Both attackers are *boats* and move among the respective points of interest shown here on the waterways map layer.

We briefly present insights obtained from analyzing the movement patterns of the best attackers obtained by our tests; further analysis is left for future work. Figure 7 (left) depicts the best black hole attacker found. This attacker is a boat which only travels on the shortest path between any two of the points shown, and thus "covers" only the southern half of the main canal, and a small number of the lesser canals. For the result to be explained, these points should be seen in the context of the general mobility patterns of honest nodes (Fig. 4, right): many lead to map areas where either honest pedestrians or honest boats have their preferred routes. In contrast, the best data flooder (Fig. 7, right), which is also a boat, also travels among crowded areas, yet maintains a much wider territory. Interestingly, while the best attackers found are generally boats, the best group of four attackers was found to be mixed, composed of one pedestrian and three boats.

5 Conclusions

We introduced an evolutionary-driven testing framework for Disruption Tolerant Networks. The proposed framework uses an evolutionary algorithm to evolve the most disruptive paths that attackers should take to reduce the network

performance (expressed in terms of data delivery rate and latency) the most. We showed the applicability of the approach with realistic *in silico* experiments which simulated the movement of a large number of agents of different types and networking behavior ("malicious" vs "honest", black holes vs flooders) in a complex urban environment. The evolution led us to discover the most malicious movement patterns that attackers should follow. We quantified the damage inflicted by groups of up to four attackers, and compared the resulting network performance with baseline values obtained in absence of attackers and with randomly generated attackers.

The study shows that single attackers can disrupt the network equally as much as (in the related work, Sect. 2) turning half of the entire network into attackers. This is partly caused by the fact that the scenarios (city map, movement model, network size) considered there are different than ours. Even so, our evolutionary testing method is more effective, and more generally applicable to testing complex networked systems than the previous methods based on heuristics: it requires no *prior* knowledge as to what an attacker type and location have as effect upon the overall network. Thus, it is likely to uncover *new* knowledge as to the cause-and-effect of security attacks.

Importantly, our methodology can be easily applied to assess the vulnerabilities of virtually any ad hoc network, allowing network experts to identify the weaknesses of protocols in large networks and, possibly, pro-actively find countermeasures. In future works, we plan to extend this approach, scaling up to larger networks and larger groups of attackers, studying alternative routing protocols, and more advanced evolutionary schemes such as cooperative evolution.

References

1. Jain, S., Fall, K., Patra, R.: Routing in a delay tolerant network. In: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM 2004, pp. 145–158. ACM, New York (2004)
2. Borrel, V., Ammar, M.H., Zegura, E.W.: Understanding the wireless and mobile network space: a routing-centered classification. In: Proceedings of the Second ACM Workshop on Challenged Networks, CHANTS 2007, pp. 11–18. ACM, New York (2007)
3. Jenkins, A., Kuzminsky, S., Gifford, K., Pitts, R., Nichols, K.: Delay/disruption-tolerant networking: flight test results from the international space station. In: Aerospace Conference, 2010 IEEE, pp. 1–8, March 2010
4. Burgess, J., Gallagher, B., Jensen, D., Levine, B.: Maxprop: routing for vehicle-based disruption-tolerant networks. In: INFOCOM 2006, 25th IEEE International Conference on Computer Communications, pp. 1–11, April 2006
5. Li, F., Wu, J., Srinivasan, A.: Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets. In: INFOCOM 2009, IEEE, pp. 2428–2436, April 2009
6. Li, Q., Gao, W., Zhu, S., Cao, G.: To lie or to comply: defending against flood attacks in disruption tolerant networks. IEEE Trans. Dependable Secure Comput. **10**(3), 168–182 (2013)

7. Burgess, J., Bissias, G.D., Corner, M.D., Levine, B.N.: Surviving attacks on disruption-tolerant networks without authentication. In: Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2007, pp. 61–70. ACM, New York (2007)
8. Sanchez, E., Schillaci, M., Squillero, G.: Evolutionary Optimization: The μGP Toolkit, 1st edn. Springer, New York (2011)
9. Bucur, D., Iacca, G., Squillero, G., Tonda, A.: The impact of topology on energy consumption for collection tree protocols: an experimental assessment through evolutionary computation. Appl. Soft Comput. **16**, 210–222 (2014)
10. Bucur, D., Iacca, G., Squillero, G., Tonda, A.: The tradeoffs between data delivery ratio and energy costs in wireless sensor networks: a multi-objective evolutionary framework for protocol analysis. In: Proceedings of the Sixtienth Annual Conference on Genetic and Evolutionary Computation Conference, GECCO 2014. ACM, New York (2014)
11. Keränen, A., Ott, J., Kärkkäinen, T.: The ONE simulator for DTN protocol evaluation. In: SIMUTools 2009, Proceedings of the 2nd International Conference on Simulation Tools and Techniques, New York, NY, USA, ICST (2009)