Evolutionary Intelligence manuscript No.
(will be inserted by the editor)

Survey on Evolutionary Computation Methods for Cybersecurity of Mobile Ad Hoc Networks

Janusz Kusyk · M. Umit Uyar · Cem Safak Sahin

Received: date / Accepted: date

Abstract In this paper, a comprehensive survey of evolutionary computation (EC) methods for cybersecurity of mobile ad hoc networks (MANETs) is presented. Typically, EC methods are named based on the natural processes inspiring them, such as swarm intelligence (e.g., ant colony optimization, artificial bee colony, and particle swarm optimization), evolutionary algorithms (e.g., genetic algorithms, genetic programming, grammatical evolution, and differential evolution), artificial immune systems, and evolutionary games analyzing strategic interactions among different population types. We introduce these methods with their typical applications, and commonly used algorithms to improve cybersecurity within the scope of MANETs. Ongoing and speedy topology changes, multi-hop communication, non-hierarchical organization, and power and computational limitations are among the intrinsic characteristics of MANETs causing cybersecurity vulnerabilities. We describe basic defense mechanisms in MANETs for vulnerability detection, attack deterrence, prevention and recovery, and risk mitigation. We classify principal applications of EC as intrusion detection, trust management, and cryptography in cybersecurity systems to countermeasure adversarial activities.

This research is supported by D01_W911SR-14-2-0001-0014. The contents of this document represent the views of the authors and are not necessarily the official views of, or endorsed by, the US Government, Department of Defense, Department of the Army or US Army Communications-Electronic RD&E Center. Dr. Cem Safak Sahin is currently an MIT Lincoln Laboratory employee; no Laboratory funding or resources were used to produce the result/findings reported in this publication.

J. Kusyk
NYC College of Technology, City University of New York, NY 11201

M.U. Uyar
City College of New York, City University of New York, NY 10031

C.S. Sahin
MIT Lincoln Laboratory, MA 02420

Keywords MANETs · cybersecurity · artificial intelligence · swarm intelligence · ant colony optimization · artificial bee colony · particle swarm optimization · genetic algorithms · genetic programming · grammatical evolution · differential evolution · artificial immune systems · evolutionary games.

1 Introduction

With ever growing proliferation of interconnected computer systems and, more recently, smart devices, one can observe a significant increase in threats faced by networks of all types and sizes. Adversarial actions in cyber space can jeopardize large corporations, government programs, industrial equipment, power plants, aviation infrastructures, health systems, and military installments ([1,4,126]). *Cybersecurity* encompases a set of measures for protecting computers, networks, programs, and data against security risks and malicious actions in cyber space. Cybersecurity attempts to deter, prevent, and limit impacts of cyberattacks by eliminating vulnerability of computer systems and implementing a set of countermeasures.

The features that make mobile ad-hoc networks (MANETs) attractive to many modern applications, such as lack of a hierarchical node structure, self-governing nodes using multi-hop communication, and continuous and speedy changes in topology due to node mobility, also create security vulnerabilities for MANETs. Biologically inspired computation techniques are excellent candidates to bring effective solutions to cybersecurity issues raised by MANET topology control, routing and node collaboration. These techniques can find desired (optimum or near-optimum) solutions to satisfy cybersecurity objectives in otherwise prohibitively large search spaces. Biologically inspired computation emulates the evolutionary theory in nature, where better adapted individuals have greater chances of survival in a given environmental niche than less fit ones and, therefore, of passing their genetic materials to their offspring. Evolutionary Computation (EC) is a broad area of Artificial Intelligence (AI) in which biologically inspired optimization algorithms mimic evolutionary processes in nature.

We classify EC techniques as: Evolutionary Algorithms (EA), Swarm Intelligence (SI), Artificial Immune Systems (AIS), and Evolutionary Games (EG). In EA, Genetic Algorithms (GAs), Genetic Programming (GP), Grammatical Evolution (GE), and Differential Evolution (DE) are included in our study. Within the SI group, we consider Artificial Bee Colony (ABC), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO), which are among the most frequently used algorithms to deal with MANET-related cybersecurity issues.

Recent surveys on MANET security focus on intrusion detection [28, 90], trust management [26, 88], detection of malicious nodes [123], security of routing protocols [10, 77], and trust in opportunistic networks with intermittent communication channels [135]. Although a few systems using some form of AI for MANET cybersecurity are reported (e.g., artificial immune systems in network security management [98], and EA and GA applications for MANET optimization [35,100]), they are not comprehensive cybersecurity solutions based on EC.

This paper presents a comparative evaluation of the state-of-the-art EC methods reported in literature for MANET cybersecurity. Our contributions include:

- summarize main vulnerabilities stemming from intrinsic characteristics of MANETs,
- identify cyber threats and trust management issues,
- introduce key concepts in EC, and
- classify EC-based solutions against cyberattacks.

The rest of this paper is organized as follows. Section 2 introduces MANETs, their vulnerabilities to cyber threats, and traditional cyber defense methods. Section 3 presents fundamentals of EA including GA, GP, GE, and DE and their applications in MANETs. SI methods and network security countermeasures using ACO, ABC, and PSO are in Section 4. In Section 5, principles of AIS and its application in cybersecurity are outlined. Section 6 familiarizes the reader with EG concepts and their use in mitigating cyber threats for MANETs. Concluding remarks are in Section 7.

2 Mobile Ad hoc Networks

MANETs are infrastructure-less wireless networks formed by autonomous mobile nodes. Decentralized structure of MANETs promotes high scalability and responsiveness that make them particularly suitable for rapid deployments in many civilian and military applications.

A MANET node may temporarily or permanently lose connectivity to its neighbors due to node movements. Constant variation of the number of nodes contribute to unpredictable dynamics in MANETs. Ability to establish structureless networks in unfriendly environments makes MANETs especially vulnerable to security threats to which structured networks are typically immune. We also include decentralized Wireless Sensor Networks (WSNs) [12] and peer-to-peer networks in this study when their security solutions appear to be adaptable to MANETs.

2.1 Vulnerabilities of MANETs

Compared to networks using guided media (e.g., wired or optical networks), MANET communication channels are relatively easy to compromise at least due to propagation characteristics of wireless signal. With respect to other wireless networks, such as cellular or infrastructured networks, MANETs do not rely on access-regulating entities (e.g., a Base Station (BS) or an Access Point (AP)). Main features that make MANETs particularly vulnerable to cyberattacks are: *(a)* wireless multi-hop communication, *(b)* nonhierarchical organization by self-governing nodes, *(c)* continuous and unpredictable changes in topology resulting from node movements to freely join, leave, and relocate, and *(d)* typical limitations in power and computational resources. Often unavoidable in MANETs multi-hop communication makes eavesdropping and obstructive routing attacks fairly easy for adversaries, while lack of centralized authority prevents straightforward applications of protective cryptosystems or implementing enforceable security policies. Due to its ever-changing topology, it is difficult to determine a MANET's security boundaries and appropriate trust levels of neighboring nodes, and, hence, to identify compromised resources in a timely fashion. Limited power resources can be exploited by adversaries to exhaust power of targeted mobile nodes with an intention to cripple an entire MANET by, often permanently, disabling its elements. Although main objective for MANET security include achieving Confidentiality, Integrity, and Availability (CIA), it requires fundamentally new techniques and significant changes to the traditional security countermeasures.

2.2 Cyber Threats for MANETs

Hostile entities can be either external or internal to a MANET. In an external attack, an adversary attempts to compromise a network without joining a MANET, while an internal attack is carried out by a hostile node, which is already a part of it. A node may be unaware of its own network-disturbing activities or be hostile with purely malicious intentions. It is also possible that a cyber threat is not from a hostile entity, but

Table 1: EC-based countermeasures against specific types of cybersecurity attacks

	EC Methods								
	EA				SI			AIS	EG
	GA	GP	GE	DE	ACO	ABC	PSO		
DoS	[13]	[30] [27]	[39] [29]	[37] [124]	[61] [55]	[109] [14]	[6] [66]	[69] [14] [13]	[102] [5] [130] [73]
Attacks on network security		[119] [27]	[85]	[136] [47] [48] [124]	[58]		[6]		[8]
Disruption of data collection	[13]	[30]	[39] [29]		[61] [55]	[14]	[66]	[69] [108]	[17] [130] [99] [110]
Attacks on nodes behavior		[119]			[61] [58]		[6] [128]		[84] [112] [130] [99]
MiM, wiretapping, illigal data access	[113] [106]	[27]					[6]		[8] [57] [99] [110]

from a member node acting selfishly trying to gain an advantage over other nodes. This selfish behavior can be motivated by a node's desire to accomplish its own tasks, while ignoring the requirements for cybersecurity of the network.

Cyber attacks that attempt to alter network operations can be grouped as: *(a)* Denial-of-Service (DoS) attacks (e.g., sleep deprivation, jamming, route modification, black hole, rushing, or flooding attacks), *(b)* attempts to bypass or break into network security measures (e.g., stealth, virus, worm, Trojan horse, or unauthorized network access), *(c)* disruption of data dissemination (e.g., hijacking, dropping or altering data packages, adding artificial latency, injecting false data, gray-hole, wormhole, or neighbor attacks), and *(d)* alternation or impersonation of node behavior or identity (e.g., impersonation, Sybil, blackmailing, on-off, and conflicting behavior). Passive cyber attackers monitor or scan a network, and hence, are often silent. Common examples of passive attacks include: *(e)* Man-in-the-Middle (MiM), *(f)* wiretapping and traffic analysis (eavesdropping), and *(g)* unauthorized location, topology, and data dissemination.

Cyber attacks on MANETs can also exploit weaknesses in routing protocols, focusing on a specific OSI reference model layer, or by targeting security measures. Table 1 summarizes EC implementations recently reported in literature as countermeasures for cyber attacks. Rows of Table 1 represent attack types, whereas columns are the EC-based security methods that address them.

2.3 Traditional Methods for Mitigating MANET Threats

Cryptography has traditionally been at the forefront of network cybersecurity for assuring CIA measures. For example, a Public Key Infrastructure (PKI) can facilitate a network security framework, where a centralized unit called Centralized Authority (CA) issues digital *certificates* (e.g., X.509) for network entities asserting their identities and validity. Since hierarchical structure of PKI is incongruent with distributed networks, shared CAs with a group of nodes needed to collaboratively authenticate services may be more functional for MANETs.

Trust among MANET nodes can be established by various methods such as coalition formation, authentication, access control, intrusion detection, key management, and isolation of misbehaving nodes. Trust relation among nodes [26] is: *(a)* based on temporal and spacial local information, *(b)* subjective, where one trustee can be assigned a distinct levels of trust by different neighboring nodes, *(c)* not necessarily transitive (i.e., if node n_1 trusts n_2, and n_2 trusts n_3, it does not imply that n_1 trusts n_3), *(d)* asymmetric, hence not reciprocal, *(e)* context dependent, such that a node's trust to its neighbor may vary for different applications, and *(f)* self-organized and dynamically reconfigurable.

Monitoring malicious activities has traditionally been performed by Intrusion Detection Systems (IDS). A Network IDS (NIDS) monitors traffic on the entire network, whereas a Host IDS (HIDS) runs as an autonomous agent on selected nodes. NIDS may *sniff* a network communication or obtain network activity information from active Software Agents (SAs) to decide about an intrusion. Typically IDS employs statistical analysis of traffic by comparing packets against known threats. This method, however, suffers from high false-positive and false-negative detections and can miss zero-day attacks.

Table 2 presents cybersecurity defense mechanisms enhanced by EC algorithms such as EA, SI, AIS, and EG. Rows of Table 2 are the defense mechanisms implemented by EC algorithms, including cryptosystems, node trust and reputation mechanisms, IDS, and other EC-based cybersecurity approaches such as evolving policies [85], node identification [47,48], and jamming detection [109].

Table 2: Cybersecurity defense mechanisms based on EC methods

	EC Methods								
	EA				SI			AIS	EG
	GA	GP	GE	DE	ACO	ABC	PSO		
Cryptosystems	[113] [106]			[136]		[61]			[102] [8]
Trust and Reputation		[119]			[58]		[128]	[108]	[84] [112] [130] [110] [73]
IDS and IPS	[13]	[30] [27]	[29] [39]	[37] [124]	[55]	[14]	[6] [66]	[69] [13] [14] [108]	[5] [17] [57] [99]
Other			[85]	[47] [48]		[109]			

3 Evolutionary Algorithms

EAs mimic evolution processes observed in nature, hence, biological terms are embraced to describe them. A *chromosome* represents a possible solution called an *individual*, whereas candidate solutions at a given time form the *population*. A selection mechanism picks parent chromosomes for a *crossover* operation that yields one or more *offspring*. With a small probability, chromosomes may *mutate*. A population evolves from one *generation* to the next by selection, crossover, and mutation operations.

A mathematical function, called *fitness*, provides a numerical representation of a candidate solution's proximity to the overall optimization goal. Determining an appropriate fitness function for the problem at hand is crucial in all biologically inspired algorithms. In a roulette wheel selection, a popular selection mechanism, a chromosome's fitness level is indicative of its chance of being selected as a parent for the next generation. For all populations $\mathscr{S}$, the probability of a solution a_i to be selected as a parent is $p(a_i) = \frac{f(a_i)}{\sum_{a_j \in \mathscr{S}} f(a_j)}$ where $f(.)$ is the fitness function yielding $a_i \in \mathscr{S}$ (for $i = 1, \cdots, |\mathscr{S}|$). In Figure 1(a), selection of a candidate from population $\mathscr{S}$ for reproduction is illustrated. Let function $rand([0, \mathscr{F}))$ generate a random number in the range of $[0, \mathscr{F})$ (i.e., spinning a roulette wheel), where $\mathscr{F}$ is the sum of all fitness values for all candidates, $\mathscr{F} = \sum_{a_i \in \mathscr{S}} f(a_i)$. In the example of Figure 1(a), a_3 was selected as a parent.

Stochastic universal sampling [11] uses a random value to select all parents at once, with each candidate mapped into a segment of a line of length $\mathscr{F}$ proportional to its fitness. Evenly spaced n pointers over a line with segments for the fitness of each candidate with $\mathscr{F}/n$ space between any two pointers, starting at a random number in $[0, \mathscr{F}/n))$. Candidate a_i is a parent if it has pointer to its segment. In Figure 1(b), a_2 was omitted, while a_1 and a_3 are selected as parents.

In *tournament* selection [9], a contest between two or more randomly chosen candidates is used to select parents.

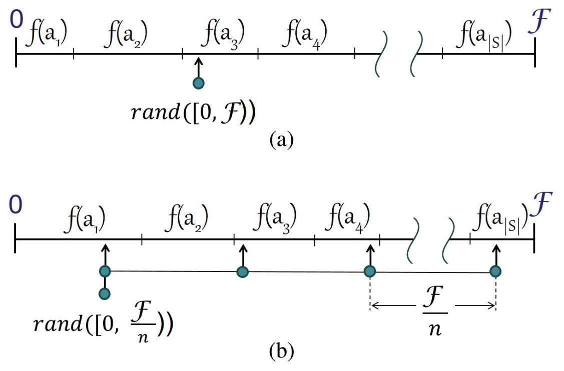


Fig. 1: (a) roulette wheel, and (b) stochastic universal selection mechanisms

The probability of the i^{th} ranked candidate to be selected from the ordered set $\mathscr{S}$ in a tournament among q randomly selected individuals is $p(a_i) = \frac{1}{|\mathscr{S}|^q}\Big((|\mathscr{S}| - i + 1)^q - (|\mathscr{S}| - i)^q\Big)$. In *elitist* selection, the fittest individuals are advanced to the next generation without modifications, where the ratio of elite group to the population size is typically kept small.

In a *single-point crossover* operation, one pivotal point in both parents is selected such that data between parent chromosomes are swapped around it. In Figure 2(a), information after 15th bit in parents $\mathscr{A}$ and $\mathscr{B}$ is swapped to obtain two offspring $\mathscr{A}'$ and $\mathscr{B}'$. Crossover operation can be implemented using multiple points and/or with more than two parents. After crossover, mutation can be performed to maintain genetic diversity in a population. With a small probability, mutation operator changes one or more bits in a chromosome, which may be selected by using probability distribution models.

Table 3: Features of EA-based methods used in network security implementations

		Characteristics of EA-based Techniques for Cybersecurity				
		Description	Selection	Crossover	Mutation	Fitness
GA	[13]	Dynamic IDS for networks using AODV	Tournament	Elitism on two best parents	Gaussian	Maximum coverage of nonself spaces
	[106]	Verification for a key exchange operation	Random	Uniform	Unspecified parameters	String matching to authenticate a node
	[113]	Use GA operations to produce cipher text	Not used	Two point	Flipping randomly selected bits	Not used
GP	[30]	Generate intrusion detection program by means of GP	Tournament among seven random groups of individuals	Mate with probability of 0.9 and elitism	Parameters not specified	Detection minus false positive rates
	[27]	Dynamically evolve SA to customize IDS	Parameters not specified	Prevents crossover of different tree types	Parameters not specified	Differences of agent and actual predictions
	[119]	Artificial trust relation model	Tournament of two random individuals	Two-points	Type not specified	Related to reduction in the number of attacks
GE	[29]	Evolve intrusion detection programs	Type not specified	Elements mate with the probability of 0.9	With probability of 0.01	Difference of detection and false positive rates
	[39]	Detect DoS and data dissemination attacks	Parameters not specified	Parameters not specified	Parameters not specified	Proximity of a given design to a set of goals
	[85]	Evolve fuzzy security policies	Roulette wheel	One-point with probability of 0.9	Flipping bits with probability of 0.01	Based on differences of risk factors
DE	[37]	Evolve rules for detecting anomalies in computer networks	The best individual in a generation	Binomial with rounding satisfying the boundary condition	DE/best/1 [116]	Based on *if-then* rule matching to testing data connections
	[124]	Generate fuzzy rules to construct a classifier	Regardless of the fitness	Uniform with the probability of 0.7	Type not specified	Root-mean square error with replacement
	[136]	Generate attacks on transposition cipher	Better fitted vector	Time-varying in the range of 0.5 and 0.9	Parameters not specified	Compares decrypted message to a language
	[48] [47]	Verification of nodes at physical layer of OSI	Parameters not specified	Vector-based with three random individuals	Parameters not specified	Based on kernel regression criteria

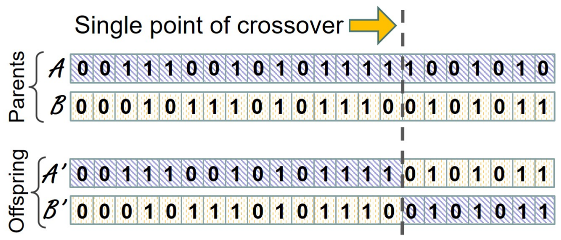

Single point of crossover
Parents
A 0 0 1 1 1 0 0 1 0 1 0 1 1 1 1 1 1 0 0 1 0 1 0
B 0 0 0 1 0 1 1 1 0 1 0 1 1 1 0 0 1 0 1 0 1 1
Offspring
A′ 0 0 1 1 1 0 0 1 0 1 0 1 1 1 1 0 1 0 1 0 1 1
B′ 0 0 0 1 0 1 1 1 0 1 0 1 1 1 0 0 1 0 1 0 1 1

Fig. 2: EA single-point crossover operation

3.1 Examples of Evolutionary Algorithms

In this section, we introduce GA, GP, GE, and DE for improving cybersecurity in MANETs. Table 3 presents examples for each method, and their selection, crossover, mutation operations and fitness functions. EC-based cybersecurity methods for WSNs are also included in Table 3 when they appear to be applicable to MANETs.

3.1.1 Genetic Algorithms

Algorithm 1 presents a pseudocode for a simple GA [52, 70, 83]. First an initial (random) population of candidate solutions is evaluated. Then, parents are selected and offspring is formed by crossover and mutation. Finally offspring is evaluated for inclusion in the next generation. This process is repeated for a predetermined number of iterations (or until a satisfactory solution is found).

Algorithm 1 A simplified implementation of GA

```
Initialize random population;
Evaluate fitness for each population element;
while stopping criteria is not attained do
    Select parents for reproduction;
                //Parents are selected from fittest individuals
    Generate offspring by crossover and mutation;
    Evaluate offspring and form a new generation;
                // Offspring replaces less fit individuals
end while
```

Fixed-length chromosomes make GAs useful to solve problems for which a solution can address an entire search space, for example when determining the best location for a mobile agent to move [78]. GA can be applied to network routing problems [15], while Force GA (FGA) was successful in topology control [104,105,122]. GAs are surveyed in [28] for intrusion detection and data collection in MANETs and multi-objective optimization techniques.

GA-*based Intrusion Detection Systems* GAAIS is an IDS based on GA and artificial immune system for dynamic intrusion detection in AODV-based ad hoc networks [13]. Each node in GAAIS uses a real-value GA with tournament and elitism selections, where tournament winners are advanced to next generation together with the fittest parent chromosomes to obtain best detectors.

GA-*based Secure Key Exchange* In a MANET, freely moving nodes can lose connectivity to their neighbors and, hence, rejoin the network many times, which may generate vulnerabilities exploitable by malicious agents. Security issues due to node mobility are studied in [106], where threshold cryptography ensures trust among MANET nodes. A GA is used to verify secure key exchanges among nodes by providing efficient authentication of the nodes entering the network. In [113], secure symmetric key generation is accomplished by a GA that adopts only crossover and mutation operations.

3.1.2 Genetic Programming

If a solution to a problem can be represented as a computer program, Genetic Programming (GP) [74,75] can evolve candidate programs toward an acceptable solution. In GP, a candidate solution can be generated by combining a set of terminals and functions. Each candidate is then run to determine its fitness. Better programs, obtained by reproduction, crossover, and mutation operations, are advanced to next generations. Figure 3 illustrates two programs generated using $\{AND, OR, NOT\}$ functions and terminals of $\{c_1, c_2, c_3\}$. Figures 3(a) and (b) represent candidate programs of $\mathscr{P}_1 = ((\bar{c_1}\ OR\ c_3)\ AND\ \bar{c_2})$ and $\mathscr{P}_2 = (c_1\ AND\ (\bar{c_3}\ OR\ \bar{c_2}))$, respectively (an overbar means negation). In this context, a crossover operation swaps subtrees to generate new candidates. $\mathscr{P}_3$ in Figure 3(c) is obtained from $\mathscr{P}_1$ by substituting subtree S_1 with the subtree S_2 of $\mathscr{P}_2$. Similarly, $\mathscr{P}_4$ is formed with $\mathscr{P}_2$ with S_2 replaced by S_1 (Figure 3(d)).

In a mutation operation of GP, a subtree is replaced by a random tree. For example, S_1 in Figure 4(a) is replaced by a randomly generated tree shown in Figure 4(b), resulting in program $\mathscr{P}_5$ in Figure 4(c). Mutation in GP can substantially alter programs, and hence, in practice, it is performed with a small probability or not at all. Repetitively performed crossover operations may lead to an increased complexity

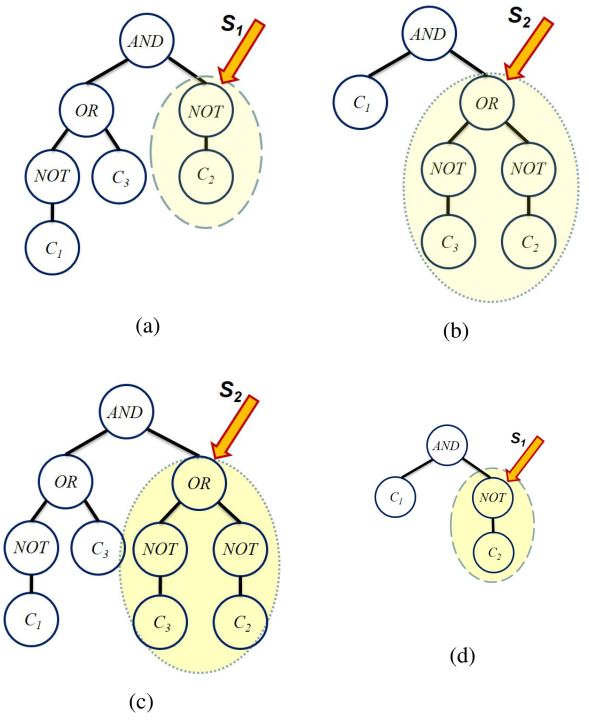


Fig. 3: GP crossover example: initial programs of (a) $\mathscr{P}_1$ and (b) $\mathscr{P}_2$, and final programs of (c) $\mathscr{P}_3$ and (d) $\mathscr{P}_4$

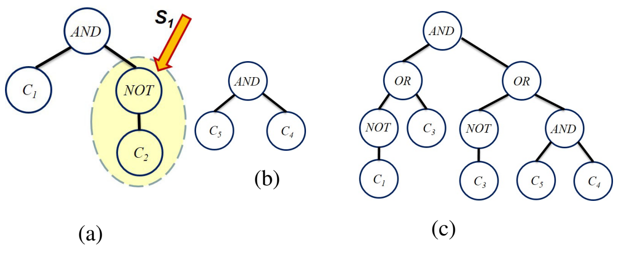


Fig. 4: GP mutation example: (a) program $\mathscr{P}_1$ at point S_3, (b) a random tree, and (c) resulting program $\mathscr{P}_5$

of the tree structure with an uncontrolled growth (*bloating*). Although bloating should be avoided, it can also be helpful to develop attack variants on IDSs to uncover hidden malicious software.

Although program size in GP can grow to consume system resources, arbitrarily limiting the program size can reduce its ability to find optimal solutions. Linear GP (LGP) is a variant of GP, where a program evolves as a sequence of imperative language instructions [18,19]. Basic structure of LPG includes registers and a set of operational and branching instructions. To ensure that only valid programs are generated by LPG, crossover points are selected such that they do not cut through instructions and that the mutation operation cannot change an instruction for a register. Linear solutions of LGP reuses subprograms, and hence, are usually more compact in expressing complex calculations with less instructions than traditional tree-based GP solutions.

GP has been applied to a wide range of problems, including symbolic regression, image classification, pattern recognition, software development, automated learning, information retrieval, finance, and medicine. Program generation for sensor nodes [134] and parallel evolutionary computing models in WSN [60] are examples of GP applications in networking. LGP has been used to provide optimal hierarchical structure of a solution space (e.g., menu layout) while preserving constraints of tree elements [121]. Low complexity intrusion detection programs can be generated by LPG to identify DoS, probe attacks and attempts to circumvent access privileges [89, 115].

GP-based Intrusion Detection Systems IDS agents can be built using GP to effectively monitor network connections under continuously changing conditions. In [27], each node runs its autonomous agent independently to detect suspicious values to be reported to an entity that compiles network wide suspicion reports. GP can train autonomous agents to detect intrusive behavior and engage in a distributed and cooperative NIDS. The GP-based intrusion detection implementation in [27] generates separate programs for monitoring network timing, ports, and connections to privileged ports to avoid obtaining unsafe parse trees. In [30], GP is combined with a multi-objective evolutionary algorithm, where programs to detect known attacks against MANETs (e.g., flooding and route disruption attacks) are evolved using strongly-typed GP and then evaluated on simulated networks to determine their effectiveness.

GP-based Trust and Reputation GP trees can provide mathematical functions to determine if future interactions with peer neighbors should be carried out. Artificial trust relation model [119] uses past interactions to determine trustworthiness against individual, collaborative, and pseudospoofing attacks in sharing peer-to-peer communication systems. Based on a history of interactions and recommendations, a GP tree can be built to include features such as total and successful number of interactions, time between the last two interactions, average trust values, and average satisfaction. The selection, crossover, and mutation operations are similar to the ones used in ECJ toolkit [3], however the fitness function here reduces the number of attacks.

3.1.3 Grammatical Evolution

Fixed-length chromosomes of traditional GA can have limitations when addressing complex solution spaces. Chromosomes with pre-determined lengths can either miss a solution when they are too short, or deplete computations resources when unnecessarily long. Variable-length chromosomes in Grammatical Evolution (GE) alleviate some of these limitations [95, 103]. In GE, a syntactically correct program is generated from binary strings to select production rules in a Backus-Naur Form (BNF) grammar definition [95]. A BNF notation for context-free grammar describes the syntax of a language by a tuple $\{N, T, P, S\}$, where N is the set of non-terminals behaving as variables, T is the set of terminals (i.e., literal symbols) which may appear as the outputs, P stands for a set of production rules, and $S \in N$ is the start symbol. Production rules use meta-symbol ::= to denote "is defined as," and | as a separator of alternative definitions for a non-terminal. The pair of < and > encloses non-terminals. A sequence of production rules maps elements of N to T, where an element of N may appear on a left or right side of a rule, but terminals are never on a left side. For example, consider the production rule

$$< int > ::= < digit > \mid < int >< digit > \tag{1}$$

where *int* and *digit* are non-terminals, and *int* is defined as either a non-terminal *digit* or itself (i.e., *int*) followed by a *digit*. Let us add another rule to the one above as

$$digit ::= 0|1|2|3|4|5|6|7|8|9 \tag{2}$$

where *digit* is defined as one of the terminals 0 to 9. These two rules constitute the BNF grammar for any integer.

Each choice in a production rule can be numbered. Since *digit* has ten possible choices, a binary codon representing *digit* must have 4-bits, whose decimal value ranges from 0 to 15. The modulo operation can be a mapping function as follows: $rule = C_n$ MOD R_n, where C_n is the value of the codon and R_n is the number of choices for this non-terminal. In our example, let $int = S$ be the starting non-terminal with an initial binary string of

1001	1111	0100	1110	1001

whose transcription to a string of codons is

9	15	4	14	9

Since 9 MOD 2 = 1, $int = S$ is expanded by using the rule of $< int > ::= < int >< digit >$, the solution now becomes $< int >< digit >$. Expanding the non-terminals continues with *leftmost derivation* rule as follows. 15 MOD 2 = 1 dictates that leftmost non-terminal $< int >$ is expanded to obtain $< int >< digit >< digit >$. Now, the new leftmost $< int >$ is expanded to get $< digit >< digit >< digit >$. At this point, the leftmost non-terminal is $< digit >$ with 10 possible rules, and the next codon value to consider is 14. Since 14 MOD 10 = 4, the solution becomes 4 $< digit >< digit >$. Application of 9 to $< digit >$ results in 48$< digit >$. When there are no more codons left to traverse with some non-terminals remaining to be processed, the codons must be reused from the beginning, resulting in integer of 488. Wrapping a sequence for codons is consistent with gene-overlapping in biological processes.

Successful GE applications include autonomous robotic systems, (e.g., a GE-based mechanism to automatically design a decision-making function to improves exploration performance [54]), finance sector [31] and neural network optimization [22].

GE-*based Intrusion Detection Systems* With success of GP in generating programs capable of recognizing malicious activities in MANETs, GE, potentially more universal and adjustable than GP, has also found its applicability in IDS implementations. In a GE-based approach [29], a BNF considers mobility and packet-related features to discover dropping, flooding, and route disruption attacks with a fitness function based on the number of correctly detected attacks and false positives. A medium mobility MANET is used as a training environment. which is later deployed on each node as an intrusion detection agent. In [39], a GE technique to detect DoS and data dissemination attacks against AODV protocol is explored. Black hole, neighbor dropping, and route disruption attacks are evolved by a GE algorithm.

GE-*based Policy Evolution* Formulating a right set of security policy rules often requires handling conflicting goals, such as node exposure and security risks. In [85], GE infers fuzzy multi-level security policies that is most consistent with the supplied training decision set. A BNF grammar for evolving policies to determine trustworthy entities with appropriate *need-to-know* access rights is presented, where a steady-state GA with different crossover operations is explored. Although this mechanism is proposed for mobile and sensor networks, it can also be used for securing MANETs.

3.1.4 Differential Evolution

Differential evolution (DE) is a biologically inspired computation technique that was conceived as a means for efficiently solving high order Chebychev polynomial equations [117]. Instead of classical crossover and mutation operations, DE utilizes a new differential operator to create offspring. This operator is self-organizing and requires little input from user. Self-organizing mechanism of DE takes the difference vector of two randomly chosen population vectors to perturb an existing vector. Every population vector is perturbed in this manner. This idea is in contrast to traditional EC strategies where predetermined probability distribution functions determine perturbations.

For a population of N candidate solutions, each individual is represented as a d-dimensional vector. For each individual at generation g, DE creates a *variant* and a *trial solution*. A selection process determines if the trial solution will replace an individual in the next generation. Evolving each individual P_i^g from generation g to $(g+1)$ (i.e., $P_i^g \rightarrow P_i^{g+1}$) begins by creating a variant solution V_i, which is determined by picking three different additional indices k, r, and m. They will be used to select P_k^g, P_r^g, and P_m^g from current population for calculating a variant as

$$V_i = P_m^g + E * (P_k^g - P_r^g) \tag{3}$$

where P_m^g is called the *base vector*, $(P_k^g - P_r^g)$ is called the *difference vector*, and E is the scaling factor controlling amplification of the difference vector (typically $E \in [0,2]$).

Figure 5 illustrates how a variant solution can be calculated using Eq. (3). Let individuals P_k^g, P_r^g, and P_m^g be determined as in Figure 5(a). In Figure 5(b) a variant V_i is computed from P_k^g, P_r^g, and P_m^g when E is selected to be 1. Using V_i, we create a trial solution $U_i = (u_{i,x}, u_{i,y})$ as follows. Random real numbers $rand_x$, $rand_y$, and r_{nbr} in the range of $[0,1]$ are generated. A crossover constant (CR) in a real number in range of $[0,1]$ is then selected. If $rand_x \leq CR$ or $r_{nbr} = 0$, V_i for dimension x is used as the trial solution, otherwise the existing value of P_i^g is used. The same computation is performed for dimension y. For U_i, if fitness $f(U_i)$ is better than $f(P_i^g)$, U_i is used for generation $(g+1)$, otherwise P_i^g remains.

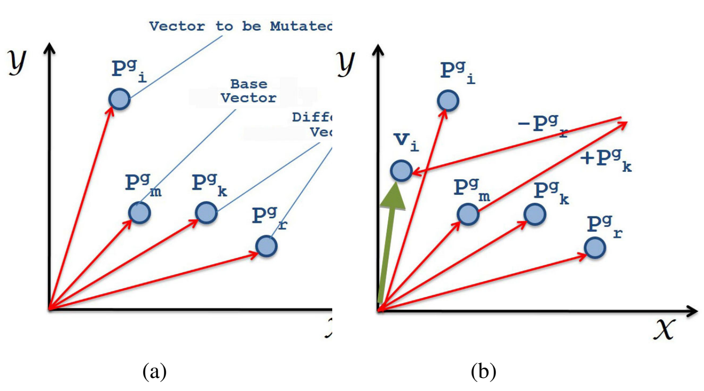


Fig. 5: Example of DE with: (a) base and difference vectors, and (b) variant solution V_i (adopted from [117])

DE has been successfully applied to MANETs and WSN for node clustering [76], routing [62,87], and network topology [45] problems. Cybersecurity applications include integer factorization in RSA algorithm (Rivest, Shamir, and Adleman's encryption/description algorithm) and design of S-boxes for symmetric key cryptosystems [81].

DE-*based Intrusion Detection and Prevention* Anomaly detection in computer networks using a DE algorithm has been investigated in [37], where security policies are in *if-then* format (e.g., *if* some features have particular values *then* anomaly is present). Policies include features such as type of connection protocol, number of failed logins, being in a host list, and ratio of erroneous connections. Candidate policies are evolved by DE with an elitist selection.

In a Redesigned DE (RDE) algorithm to generate policies in fuzzy *if-then* format, pattern recognition and classification techniques detect previously unknown network intrusions [124]. A small set of policies can classify a broad range of intrusions. In experiments on a DARPA database [2], RDE algorithm can be used to evolve fuzzy pattern learning policies to identify classes of attacks.

DE-based Cryptosystems DE can simulate network attacks to expose possible vulnerabilities of a MANET. For example, in [136], DE is adapted to stage cyber attacks on transposition ciphers. In this application, a fitness function compares statistics of key decrypted messages to a known language with *bigrams* and *trigrams* (i.e., the probability of a pair and triple of words appearing in a sequence, respectively). It is shown that the DE-generated attacks can compromise keys of sizes up to nine characters [136].

DE-based Physical Layer Security In wireless communication, eavesdropping (wiretapping) and jamming (DoS) attacks are among the main security risks associated with the physical layer of the OSI model. Although spread spectrum has traditionally been the leading defense mechanism to prevent adversarial actions at this layer, other security countermeasures have also been considered. For example, radio frequency (RF) fingerprinting can discriminate adversarial nodes to prevent unauthorized network access.

An RF attribute detection process incorporating DE-assisted Learning From Signal (LFS) classification engine to determine optimal node detection parameters in multi-node wireless networks is proposed in [47, 48]. LFS approximates unknown nodes based on available data on RF fingerprints with a DE-optimized kernel regression technique on signal propagation parameters.

4 Swarm Intelligence

Based on mechanisms observed in flock of animals in nature, Swarm Intelligence (SI) can be used to control and analyze collective behavior of decentralized autonomous systems [16]. Agents in SI systems interact with their nearest neighbors and environment to obtain intelligence that may be greater than the total intelligence of all individuals.

Self-organization can be defined as a capacity to emerge from interactions among lower-level organisms that have not been explicitly designed to achieve a goal [43]. For example, social wasps are able to build an elaborated nest with each worker depositing building materials based on a local environment. Wasps add deposits to a corner area of existing wall with a greater probability than starting a new wall [21]. With the deposit of new material, nest configuration is altered providing an updated information for the other wasps in the future. This indirect communication through the environment is called *stigmergy*. Consequently, even simple insects, such as wasps, are able to build very complex structures.

Self-organized systems may rely on positive and negative feedback, amplification of random fluctuations, and multiple direct or indirect stigmergic interactions among individuals to obtain a dynamic system exhibiting emergent properties with bifurcation (i.e., capable of obtaining a new stable solution when system properties change).

4.1 Examples of Swarm Intelligence Algorithms

In the following sections, we introduce ACO, ABC, and PSO for improving cybersecurity in MANETs (and some sensor networks). Biological swarm models for securing MANETs are presented in Table 4 using a classification similar to [43]. Rows of Table 4 present applications of SI algorithms, whereas the columns correspond to characteristics of methods.

4.1.1 **Ant Colony Optimization**

Ant Colony Optimization (ACO) mimics ants depositing an ephemeral hormone, the pheromone, on a ground in order to mark the trial path while roaming for food [33]. When the path is traveled more frequently and/or is shorter, the pheromone trace on the ground remains stronger, attracting more ants, and hence serving as stigmergic reinforcement of a good path.

In Figure 6(a), ants collect food from a source. When an obstacle blocks a path (Figure 6(b)), they may at first randomly explore the area around the obstacle to consider other paths to the food source (Figure 6(c)). Since a longer path accumulates less pheromone, a positive feedback from pheromone attracts more ants leading to an optimal solution (Figure 6(d)).

In path selection decision of an ant, if m_1 ants had used the first bridge and m_2 the second one up to the decision point of an experiment, then the probability p_1 of selecting the first bridge by the next ant $(m+1)$ may be calculated as $p_1(m+1) = (m_1+k)^h/((m_1+k)^h + (m_2+k)^h)$, where k and h are parameters tailored to fit a given experiment [34].

Finding an optimal path in a graph can be intuitively mapped into an ant biological system (e.g., finding the shortest route to a food source in Figure 6). Pheromone variables on each node can indicate quality of links. Contrary to real ants that deposit pheromone wherever they move, artificial ants often would first discover a path and then update the respective pheromone variables, and hence mimic deposition of pheromone only on their way back.

In ACO all pheromone values should be decreased over time to imitate pheromone evaporation process. Similarly,

Table 4: Features of SI-based methods used in network security implementations

		Characteristics of SI-based Techniques for Cybersecurity			
		Description	Self-organization mechanism	Collective behavior type	Elements of self-organization
ACO	[61]	PKI over ant routing algorithm	Positive feedback with stigmergic interaction by artificial pheromones	Deliberation with probabilistic selection of a next hop	Dynamic topology adjustment based on outer factors
ACO	[58]	Mechanism for distributing trust certificates	Stigmergic interaction with positive feedback	Indirect cooperation	Inner colony factors such as node mobility
ACO	[55]	Detection and prevention of cross-layer attacks	Feedback via timing of routing packets	Nodes cooperate via shared parameters	Inner colony factors such as forwarding behavior
ABC	[14]	Intrusion detection in AODV-based MANETs	Positive feedback from detectors identifying nonsafe solution spaces	Sharing best immature negative selectors to get more promising search areas	Dynamic change of detection profiles base on changes in network's topology
ABC	[109]	Detecting network areas that are under jamming attacks	Positive and negative feedback via probabilistically ranked solutions	Collaborative information sharing among employees, onlookers, and scouts	Jammers attacking the network
PSO	[6]	IDS based on PSO clustering algorithm	Positive feedback with reinforcing parameters	Particles express collective behavior when reaching an optimal objective	Internal interactions with shared global best is inherent for all PSO algorithms
PSO	[66]	Anomaly IDS with fuzzy rules of confidence ratios	Positive feedback with unspecified local/global bests		
PSO	[128]	Node reputation to modify or exclude collaboration	Positive feedback with varying *w*		

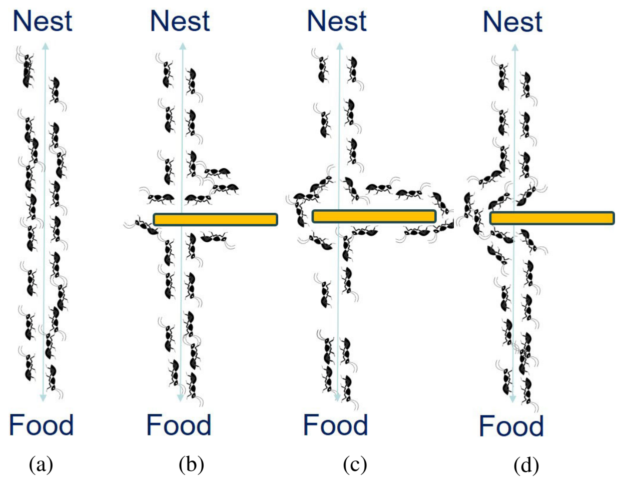


Fig. 6: Ants finding the shortest path between the nest and food source in the dynamical changing environemnt

Algorithm 2 A simple ACO algorithm (adopted from [34])

```
Initialize pheromone trails;
while stopping criteria is not attained do
    Obtain ant solution;
    Apply local search;            // This step is optional.
    Update pheromones;
end while
```

pheromone values associated with good solutions, if discovered, should be increased [34]. A local search for better paths before updating the respective pheromone values may improve performance of ACO. The pseudocode for ACO algorithm presented in Algorithm 2 is based on [34].

Let L represent the set of all direct links in a network, $l_{ij} \in L$ denote the link between nodes i and j, and φ_{ij} be the pheromone of l_{ij}. The pheromone update for the link joining nodes i and j can be computed as $\varphi_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$, where m is the total number of ants and $\Delta\varphi_{ij}^{k}$ is the quantity of pheromone on link l_{ij} that was deposited there by ant k. Let c_{ij}^{k} denote the connection cost by ant k using link l_{ij} and C^k be the total cost for k to travel current path. The amount of pheromone left by k on l_{ij} is inversely proportional to C^k. $\Delta\varphi_{ij}^{k}$ can be computed as $\Delta\varphi_{ij}^{k} = \frac{c_{ij}^{k}}{C^k}$.

Let ρ be the rate of pheromone evaporation per unit time t, then at time $(t+1)$ the amount of pheromone on $l_{ij} \in L$ is computed as $\varphi_{ij}^{(t+1)} = (1-\rho)\,\varphi_{ij}^{t}$. Finally, the probability that an ant will select node j next when it is in i is

$$p(l_{ij}) = \frac{\varphi_{ij}^{\alpha}\,\eta(l_{in})^{\beta}}{\sum_{l_{in}\in C}\varphi_{in}^{\alpha}\,\eta(l_{in})} \tag{4}$$

where function $\eta(.)$ assigns a heuristic value to each link in L, α weights an importance of a pheromone, and β determines the influence of the heuristic information.

The basic computational steps of ACO can be applied to secure packet routing in MANETs. For example, L could be audited frequently in a dynamic topology, larger ρ may reduce the reinforcement of inactive links, communication cost may reflect the power needed for signal propagation, and $\eta(.)$ represents reliability and trust for a next hop node or a link.

An ant system where only the best ants can update the pheromone is presented in [118]. AntNet [23] was introduced as a proactive routing algorithm that uses a forward ant for path discovery and a backward ant for link status changes. AntHocNet [24] offers a hybrid ant based routing protocol for finding a route on demand and proactively maintaining and improving the existing routes, whereas HOPNET [129] introduces an ACO based routing for MANETs with local proactive route discovery and reactive communication between neighborhoods.

ACO-*based Trust and Reputation* Trust and reputation among mobile nodes of a MANET are defined as the parameters reflecting the suitability for cooperation with a given node. Computation and distribution of trust values for mobile nodes are two main components of a comprehensive trust model [58]. Trust computation determines the trust level of a node through its direct or indirect interactions with others. Analyzing historical behavioral patterns can help to obtain an assessment of node trustworthiness. Similarly, relying on information from other nodes may be helpful in establishing secure interactions with new neighbors or in refining already held believes. Trust levels for mobile nodes, which are obtained directly or indirectly, are distributed among MANET nodes.

Ant-Based Evidence Distribution (ABED) [58] is an example for distribution of trust certificates, where a similarity between cooperative interactions of MANET nodes and real ants are exploited. Mobile artificial ants search for an optimal path toward a trust evidence, similar to real ants searching for food sources. Cybersecurity metrics can be incorporated into decision criteria for finding an optimal path. Each node maintains a local Certificate Table (CT) with entries for each known trust certificate and a probability associated with it. When searching for a trust certificate of a node, *unicast forward ants* are routed to a neighbor with the highest probability in CT, whereas *broadcast forward ants* are dispatched if there is no entry in CT. A *backward ant* carries the trust certificate to the requester, while updating all CTs along the return path. As such, ABED mimics known ACO routing protocols with CTs resembling routing tables consulted when dispatching packets.

ACO-*based Public Key Infrastructure* Network maintenance overhead due to message exchanges in a partially distributed CA and a cluster-based PKI can be avoided by deploying ant routing algorithms. In AntPKI [61], a lightweight PKI implementation, forward ants for route discovery carry self-issued certificates and backward ants, which clinch final paths, carry session keys. AntPKI uses Ant colony based Routing Algorithm (ARA) [46] for piggybacking cryptographic information. ARA creates new routes with forward and backward ants that lay down pheromone tracks to source and destination nodes, respectively. AntPKI algorithms include positive feedback via stigmergy, deliberate choice of best paths using a probabilistic selection.

ACO-*based Intrusion Detection and Prevention* Analogous to other uses of ACO for network security, forward and backward ants can be explored to perform cybersecurity related services. For example, forward and backward nodes can be employed to detect and prevent cross layered attacks, such as attacks on routing and medium access protocols [55]. Forward ants can be dispatched towards random destinations to estimate delays between path request (PREQ) and path respond (PREP) packets. Backward ants collect the delay information to update a (pheromone) table at the source node to reflect the recent mean times for intermediary nodes to be used to identify abnormal delays, which may be due to malicious behavior. Another application of ACO is the detection of suspicious neighbors by comparing the number of neighbors found by the second and third OSI layers and the number of receptions found separately by these layers [55]. A node is excluded from further forwarding if the parameter values from these two OSI layers do not match.

4.1.2 Artificial Bee Colony Algorithms

Three main components of honey bee forage selection process are *(a)* food sources and their values, *(b)* employed forgers associated with the food sources, and *(c)* unemployed forgers continually looking for new food sources. Unemployed forgers are scouts searching (randomly) for new food, while onlookers wait in the nest to establish new food sources. Employed foragers convey information about the quality, distance, and direction of their food sources through duration and type of their dance performed at a *dance floor*, such that the onlookers can decide to employ themselves at the most profitable food source.

Figures 7 and 8 demonstrate behavior of honey bees foraging for nectar (adopted from [63]), where the hive contains a dance floor for sharing information about possible food sources and an unloading area to store the collected nectar. The employed bees, represented by EB_1, repeat a set of actions: deposit nectar in the unloading area, share food source information on the dance floor to attract new bees to collect them, and go back to the nectar area to continue food collection. Another set of employed bees (denoted by EB_2) are involved only in iterative gathering of food and depositing it in the unloading area. Occasionally, bees may become uncommitted (UB) to known food sources and, as unemployed bees, are free to perform other tasks.

The unemployed bees may either try to discover new nectar or monitor the dance floor to obtain information about the most profitable food sources that are known at the time.

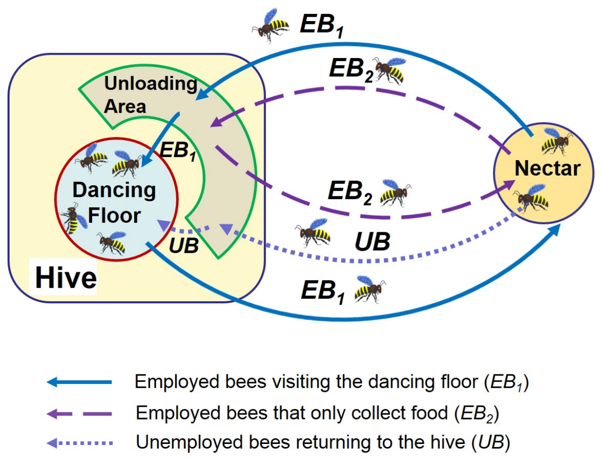

EB1
EB2
Unloading Area
Dancing Floor
Hive
Nectar
UB
Employed bees visiting the dancing floor (EB1)
Employed bees that only collect food (EB2)
Unemployed bees returning to the hive (UB)

Fig. 7: Honey bees foraging for food (adopted from [63])

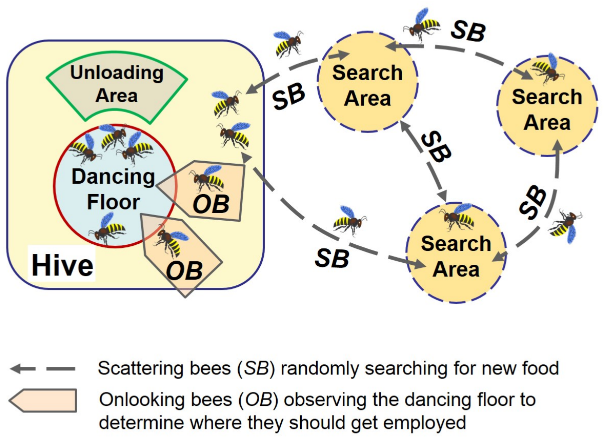

Unloading Area
Dancing Floor
Hive
OB
SB
Search Area
Scattering bees (SB) randomly searching for new food
Onlooking bees (OB) observing the dancing floor to determine where they should get employed

Fig. 8: Unemployed honey bees scouting for new food sources or observing dance floor (adopted from [63])

Figure 8 illustrates onlooking bees OB observing the dance floor to gather information about profitability of the existing food sources demonstrated to them through the dance of employed bees EB_1. Onlooking bees determine where to get employed (i.e., which food source to exploit) by responding to the quality and intensity of the dances. Contrary to EB_1 and EB_2 bees employed for collecting the food, scattering bees SB in Figure 8 randomly search the area around the hive.

Artificial Bee Colony (ABC) algorithm was introduced for solving unimodal and multimodal numerical optimization problems [63] and was quickly adopted to tackle constrained optimization problems [64]. The canonical ABC simulates foraging behavior of honey bees, employing artificial bees of *employee forgers*, *onlookers*, and *scouts*. An ABC is divided into two parts: one with employed bees, where the number of employee bees is equal to the number of food sources, and the other with onlookers.

Initially, one of the employed bees is designated as the scout, which randomly searches the solution space. Employee bees associate probability values with their food sources reflecting the quality of these food sources, which are used by the onlooking bees in selecting the food sources. Employed bees whose food sources have been exhausted become scouts and start looking for new food sources.

Algorithm 3 outlines a simple ABC procedure, which meticulously exploits a search space in promising regions, while searching for reaming space via scout bees.

Algorithm 3 An ABC algorithm (adopted from [64])

```
Randomly select and evaluate initial solutions;
while stopping criteria is not attained do
    Employed bees evaluate solutions located nearby
            current results and keep the fittest ones;
    Calculate probabilities for the best solutions so far;
    Onlookers choose solutions based on their probabilities;
    Onlookers explore vicinity of the fittest solutions so far
            to determine and store the best ones;
    if solutions exist that can be abandoned then
        Randomly scout for new solutions;
    end if
    Store the best solutions found so far;
end while
```

A fitness function used in Algorithm 3 indicates the quality of a solution such that a better solution yields a larger value than a less fitted one (e.g., to continue our analogy with honey bees, the fitness could reflect the proportional amount of food at a source). Onlookers choose to explore locations based on a probability distribution over the best solutions so far.

After choosing a solution, an onlooker produces a candidate in the neighborhood of the selected solution. Let a_i^k refer to the k^{th} dimension of solution a_i, where $a_i \in \mathscr{S}$ set of all possible solutions. We compute a candidate solution v_i^k as $v_i^k = a_i^k + rand([-1,1])(a_i^k - a_j^k)$, where $k \in \{1,2,\dots,d\}$ is a random dimension selector, $rand([-1,1])$ is a random number function, and a_j is a random candidate ($a_i \neq a_j$). A food source is abandoned if it cannot be improved.

Examples of ABC-aided routing include a bee-inspired design of mobile agents in MANETs [132], and enhancing energy efficiency of routing protocols in wireless sensor networks [65]. A secure trust-based energy efficient clustering for WSN [107], energy management [125], and service discovery [7] systems can be listed as other ABC applications.

ABC-*based Intrusion Detection Systems* Maintaining a static library (i.e., a profile) of malicious traffic is often inadequate for a MANET with continuously changing topology since an activity classified as perfectly acceptable at one point can be recognized as malicious later.

An ABC and negative selection algorithm based biological immune system, called BeeID, for intrusion detection in

AODV-based MANETs is presented in [14]. It can be used to detect flooding, blackhole, neighbor, rushing, and wormhole attacks. Employed bees modify their positions to find a minimum overlap with positive antigens (known results). The best of old and new positions is selected by each employe bee and shared with onlookers in nearby locations.

ABC*-based Physical Layer Security* A typical physical layer DoS attack jams communication frequencies used by MANET nodes by saturating the spectrum with unwanted radio signal. Therefore, it is not surprising that jamming attack detection is an important cybersecurity goal for wireless networks, MANETs, and WSNs.

DoS attacks by jammers within a WSN can be detected by using packet delivery ratio, energy, distance, packet loss ratio, and Signal to Noise Ratio (SNR) [109]. Based on these parameters in employed and onlooker bees, an ABC implementation computes fitness for each node and selection probabilities. An increase in the number of hops between nodes may indicate that the network is under a jamming attack.

4.1.3 Particle Swarm Optimization Algorithms

Particle Swarm Optimization (PSO) is an SI optimization technique inspired by social behavior observed in bird flocking and fish schooling. It was developed as a particularly suitable tool for solving problems, in which the best solution can be represented as a point in the space [67]. As a robust stochastic optimization technique based on the movement and intelligence of swarms, PSO applies the concept of social interaction to multi-dimensional optimization problems. In PSO, if one of the individuals finds a good solution, all the other individuals make use of that knowledge.

PSO is a multi-agent parallel search method where each particle *flies* over a search space. A particle j adjusts its flying based on own experience as well as the experiences of others. At any given time t, node j with a position $p(t)$ and velocity $v(t)$, keeps track of the best fitness it has achieved so far (P^{best}), and the best value global value obtained so far by the swarm (G^{best}).

Let $P_j^{(g+1)}$ be the new position for j, and P_j^g its current position. We evolve the speed of j at generation $(g+1)$ as $\mathrm{V}(g+1) = w\mathrm{V}(g) + \mathrm{C}_1\mathrm{R}_1(\mathrm{P}_j^{best} - \mathrm{P}_j^g) + \mathrm{C}_2\mathrm{R}_2(\mathrm{G}^{best} - \mathrm{P}_j^g)$, where $\mathrm{V}(g+1)$ and $\mathrm{V}(g)$ are the new and current speeds for j, respectively, the weight factor for inertia w is a real number within [0,1], C_1 and C_2 are real numbers within [0,4], and R_1 and R_2 are random numbers within [0,1].

The momentum of j is controlled by w as a trade-off between global and local exploration. For $w = 0$, no knowledge from the past is kept, whereas $w \approx 1$ implies that particles do not easily change their directions. R_1 influences P_j^{best}, while R_2 impacts G^{best} in velocity updates. C_1 represents particle self-confidence for contribution of own experience to the search. Similarly, C_2 indicates the confidence on swarm motion. If the new position of j, given as $P_j^{(g+1)} = P_j^g + V(g+1)$, has a better fitness than the personal or global best fitness, they are updated accordingly. Regardless of its fitness, j moves to its new location. A typical PSO implementation is in Algorithm 4.

Algorithm 4 A typical implementation of PSO

```
for each particle do
    Initialize each particle with feasible random number;
end for
while stopping criteria is not attained do
    for each particle do
        if fitness better than fitness value of $P^{best}$ then
            Set current value as the new $P^{best}$;
        end if
    end for
    Choose the particle with the best fitness as $G^{best}$
    for each particle do
        Calculate particle velocity and update position;
    end for
end while
```

PSO algorithms have been popular for optimal path discovery and maintenance in networks with power constrained mobile nodes. For example, [101] uses PSO to obtain optimal paths while considering transmission cost, energy consumption, and traffic intensity. Simulated annealing and PSO can schedule optimal time slot and channel for establishing communication in WSNs [71]. Another example is maintaining topology of underwater wireless networks for unmanned underwater vehicles with limited local information [138].

PSO*-based Intrusion Detection Systems* Clustering is a method for finding distinct patterns in a group of data without requiring extensive knowledge about the data itself. Network clustering identifies groups of similar nodes which are disjointed from others. In cybersecurity, clustering analysis can be employed to build an IDS model by formulating the intrusion detection as an optimization problem.

For example, MapReduce [6] is an IDS tool utilizing a PSO-based clustering algorithm, where each cluster is represented by a centroid. Initial centroids (particles) are selected randomly, then they are continuously updated with new velocities until a global best vector of particles is obtained. The fitness of a cluster centroid is evaluated based on its average minimum distance to data instances. At the completion of PSO, traffic data is clustered as normal or one of the four groups of attacks (i.e., probing, DoS, unauthorized remote and root directory access).

In another application, an anomaly based IDS implementation incorporates PSO to protect computer networks (e.g., MANETs) against unknown cyber attacks [66]. The mechanism for intrusion detection relies on fuzzy rules with associated confidence ratio for comparing current network traffic with a behavioral model of normal network activities. Optimum membership functions for the fuzzy anomaly detection are evolved using a PSO algorithm.

PSO*-based Trust and Reputation* Trustworthiness of a node can be used to identify cyber attacks in a network. Based on its reputation, the usefulness of data from a WSN node can be classified using a near-neighbor voting mechanism, and data from untrustworthy nodes can be excluded [128]. Trust-weighted medians are computed for each node based on information from its local neighbors, which are later used to select nodes for further participation in a mutual task. The most trustworthy node aggregates data with a PSO-driven centralized data fusion.

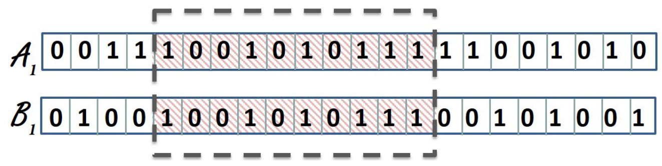
5 Artificial Immune Systems

Artificial Immune System (AIS) derives its principles from infection defense mechanisms observed in vertebrates [38]. An Immune System (IS) in nature comprises *lymphocytes* capable of detecting and responding to foreign (*nonself*) agents, referred to as *pathogens*, by recognizing *antigens* (identifiable proteins) of pathogens. A host organism trains its immune system to coexist with substances it identifies as *self*, to discriminate against them if nonself pathogens invade. A lymphocyte can recognize a pathogen by binding to an antigen, where the degree of binding is reflected by their *affinity* [97].

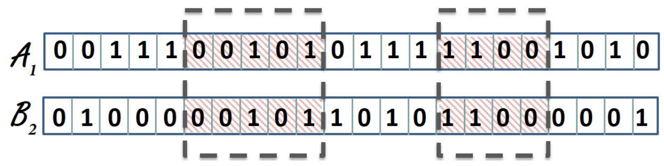
Lymphocytes include T-cells *maturing* (i.e., learning how to detect nonself elements) in a thymus and B-cells maturing in bone marrow. During maturation, T-cells are exposed to examples of self molecules. Any cell that reacts strongly to self is not allowed to mature, hence the organism uses *negative selection* to prevent an over-reactive immune system. An organism maintains a few *memory cells* to trigger a rapid and targeted response in case a pathogen reappears. B-cells may respond to pathogens by *clonal selection*, producing large quantities of *antibodies* to neutralize foreign objects. They may undergo *positive selection*, where efficient antibodies are reproduced with greater probability. To prevent lymphocytes from destroying unknown but not harmful organisms, T-cells require two signals to react: one from binding with harmful antigens and a second (*costimulatory*) signal triggered by recognizing that ”something” is wrong (*danger theory* [20]).

After maturing, detectors of an AIS can identify unknown and potentially harmful (nonself) activities. Actions of friendly nodes can be represented by a set of binary strings. A node can train on this set by determining if two strings, one from the set and one representing an immature detector, match.

Matching two strings, simulating biological binding, can be determined by, for example, the number of the same contiguous bits in them [51]. In Figure 9, binary strings $\mathscr{A}_1$ from a friendly action set and $\mathscr{B}_1$ and $\mathscr{B}_2$ from immature detectors. The affinity between $\mathscr{A}_1$ and $\mathscr{B}_1$ in Figure 9(a) is 10



(a) Match



(b) No match

Fig. 9: Binding strings: (a) 10-bit and (b) 5-bit match rule

bits. If, for example, only 8 bits are needed to determine a match, we can eliminate $\mathscr{B}_1$ from further consideration as a mature detector. If, on the other hand, match is less than 8 bits (e.g., in Figure 9(b)), we keep training (i.e., *tolerizing*) $\mathscr{B}_2$ with other actions from the friendly action set. Detector $\mathscr{B}_2$ matures if it does not match to any friendly actions. If $\mathscr{B}_2$ survives this negative selection, it is added to matured selectors.

Affinity of two numerical strings can also be determined by the Hamming or Euclidean distance between them. Let $\mathscr{A}^i$ and $\mathscr{B}^i$ represent ith elements of strings $\mathscr{A}$ and $\mathscr{B}$, respectively, then Hamming distance [25] between $\mathscr{A}$ and $\mathscr{B}$ is calculated as

$$H_{\mathscr{A},\mathscr{B}} = \sum_{i=1}^{\ell} \delta \ \text{,where } \delta = \begin{cases} 1 & \text{if } \mathscr{A}^i \neq \mathscr{B}^i \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

For distance $d = \sqrt{\sum_{i=1}^{\ell}(\mathscr{A}^i - \mathscr{A}^i)^2}$ between $\mathscr{A}$ and $\mathscr{B}$, Euclidean affinity $E_{\mathscr{A},\mathscr{B}}$ of two strings is $E_{\mathscr{A},\mathscr{B}} = d/\sqrt{\sum_{i=1}^{\ell} r_i^2}$, where r is the data range for i [131].

Since negative selection use only the dataset of known and acceptable actions, a mature detector may match a valid but unknown action, leading to erroneously recognizing it as a cyber attack. This problem may be alleviated by implementing the danger theorem in AIS, where an additional signal reflecting well-understood symptoms of a compromised MANET is needed to identify an attack. Clonal selection of B-cells can be modeled in AIS by evolving fitter mature detectors. The fitter detectors would bind with higher affinity to malicious activities. AIS maintains a set of countermeasures to furnish rapid responses, simulating memory cells of a biological IS.

Table 5: Features of AIS-based methods used in network security implementations

	Characteristics of AIS-based Techniques for Cybersecurity				
	Description	Negative selection	Affinity measure	Danger theory	Memory cells upkeep
[69]	HIDS for MANETs using AODV	Neighboring nodes recognized as self	Parameters not specified	Using DC algorithm	Clonal evolution with termination of bad cells
[14] [13]	Dynamic IDS for networks using AODV	ABC [14], GA [13] with acceptable activities	Euclidean distance	Not used	Updating otherwise expiring good detectors
[108]	Detect misbehaving nodes	Virtual tolerization on normal network actions	Clustering mechanism	Signal from packet losses	Clonal evolution of successful detectors

An AIS-based anti-virus technique developing antibodies to previously unknown computer viruses and worms is proposed in [68]. LISYS [51], an IDS implementation based on AIS, is shown to be adaptable, decentralized, error tolerant, dynamic, and self-monitoring by harnessing a biological IS to provide anomaly and signature-based detection. For anomaly detection in HIDS, identification of unknown attacks by an AIS can be combined with either EA [32], which evolves memory cells for secondary responses, or with fuzzy rules [111]. A review of AIS applications in MANET security can be found in [59].

5.0.4 AIS-based Cybersecurity

In this section, we present a sample group of cybersecurity techniques that implement AIS, which are organized by their cyber defense mechanisms, including IDS, trust management techniques and the specific attack that they prevent. Table 5 summarizes different AIS approaches for network security improvement in terms of negative and positive selections, affinity measures, danger theory, and memory cells.

AIS-based Intrusion Detection Dendritic Cell (DC) algorithm offers an extension to the danger theorem [44] by using immature, semi-mature, and mature cells (detectors). The semi-mature cells continue tolerization process of immature cells whenever they bind to unself antigens when safe costimulatory signals are present. However, they cannot initiate a response, which can only come from mature cells in the presence of unsafe costimulatory signals.

In [69], negative selection, clonal selection, and DC algorithms are incorporated into a Combined Immune Theories Algorithm (CITA) to implement adaptive and distributed IDS for MANETs. A MANET node runs CITA as a part of the AODV protocol whenever it receives an alarm signal (e.g., a high number of requests from one source, a request from unself node, or a request to forward sensitive information). After detecting a malicious node, CITA excludes it from all valid routes and drops all packets arriving from it.

AIS can be used together with ABC [14] to provide IDS-based protection in MANETs. Each node uses normal traffic features in a niching ABC (NicheABC) algorithm to train mature detectors for tolerance. A mature negative detector can evolve through an updating process searching for nearby solutions to discriminate normal and malicious activities. AIS can also be combined with GA [13] to obtain comparable countermeasures in MANETs.

AIS-based Trust and Reputation Misbehaving nodes in a MANET can be automatically detected by implementing a virtual thymus, danger signals, and memory cells [108]. In a virtual thymus, detectors are tolerized on a set of Dynamic Source Routing (DSR) events from friendly nodes. A faulty, selfish, or malicious node can be recognized by matching its actions with mature detectors in a clustering mechanism and costimulatory signal from packet losses. A clonal selection evolves persistent selection rules from the most successful mature detectors that would otherwise expire shortly. AIS-based detection of misbehaving nodes can be augmented to obtain a *reputation system* to forbid unfriendly nodes from relaying DSR packets.

6 Evolutionary Games For MANET Cybersecurity

Traditional Game Theory (GT) provides a framework for analyzing behavior of rational players in strategic situations, where an outcome depends on actions of all participants. Since its initial developments, GT has been broadly applied to problems in economics, political science, and computer science. Evolutionary Games (EG) are the games that use evolutionary computation components in their decision making mechanisms. For many MANET operations (e.g., topology control of mobile nodes), a game-theoretical approach can be adapted where incentives and deterrents can be built into node actions in a game structure to provide an optimal or near-optimal solution, eliminating a need for broad coordination and cooperation among nodes.

6.1 Traditional Game Theory

Recent developments in GT include *normal form* simultaneous move games represented as matrices, *extensive form* sequential games represented as trees and *cooperative* games in which any cooperation is enforceable by an outside party [94]. Nash Equilibrium (NE) is a stable state in a game from which players have no advantage to deviate [91, 92, 93] . In incomplete information games players are not aware of others' strategies or payoffs (numerical representations of incentives) [49].

A normal form game is defined by a nonempty and finite set I of m players, a strategy profile space S, and a set U of *payoff* (utility) functions. Each individual player $u_i \in I$ has an associated set S_i of possible strategies from which, in a *pure strategy* normal-form game, she chooses a single strategy $s_i \in S_i$ to realize. A game strategy profile is defined as a vector $s = \langle s_1, s_2, \ldots, s_m \rangle$. If a strategy profile s is played in a game, $u_i(s)$ denotes a payoff function reflecting the profit for player u_i. It is usually convenient to single out a strategy of a player u_i and refer to strategies of all other players as a deleted strategy profile s_{-i}.

If a player randomly selects among her pure strategies, we say that she is playing a *mixed strategy game*. A mixed strategy σ_i is a probability distribution over S_i, where $\sigma_i(s_i)$ is the probability of s_i being selected. We denote a mixed strategy profile as a vector $\sigma = \langle \sigma_1, \sigma_2, \ldots, \sigma_m \rangle = \langle \sigma_i, \sigma_{-i} \rangle$, where in $\langle \sigma_i, \sigma_{-i} \rangle$ we separate a mixed strategy of player u_i from a deleted mixed strategy of the remaining players. Contrary to a deterministic payoff function $u_i(s)$ in pure strategy games, the payoff function $u_i(\sigma)$ for a player u_i in a mixed strategy game expresses her *expected payoff*.

Many operations performed by MANET nodes can be modeled as non-cooperative competitions in either a normal or extensive form. In *repeated games*, interactions of a simple game are repeated more than once. Unlike a game played once, in a repeated game a strategy can be contingent on the past moves, and hence, a player can build a reputation and introduce retribution actions [41].

In a *cooperative* game, players are allowed to form coalitions, whereas players of a *non-cooperative* game act rationally only with respect to their own objectives and regardless of others. A mixed strategy profile $(\sigma_i^*, \sigma_{-i}^*)$ is NE if

$$\forall_{u_i \in I},\ \forall_{s_i \in S_i} \quad u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(s_i, \sigma_{-i}^*) \tag{6}$$

NE is an important condition for predicting outcomes in games with rational players. Any normal-form mixed strategy game has at least one NE, whereas some pure strategy games may have either one, multiple, or no NE.

In Bayesian games [49], also called games of incomplete information, players are uncertain about strategy profiles or payoffs of their opponents. In general, payoffs are numerical representations of profits that can rank a player's preferences. In classical Bayesian games a special game participant, called *nature*, randomly selects a type for each player based on a probability distribution across a player's possible types [49]. Although a player may be uncertain of another's type at the beginning of an interaction, it may update her beliefs over time. An expected payoff for a Bayesian game is calculated as a function of probabilities for all possible types of all players.

In a *perfect information* game, history of previous actions taken by all players is known to everyone. In an *asymmetric information* game, one player's knowledge is broader than the others. Players of a *symmetric* game earn the same payoff for the same actions against the same set of opponents, whereas players of an asymmetric game may either have distinct strategy sets or they share the same strategy set but with different payoffs for the same interactions.

A wide range of challenges seen in MANET operations can be alleviated by GT-based solutions, including control dynamic spectrum sharing [53, 56, 96], routing [42, 50], and dynamic network topology control [36, 72, 80]. Additional applications of GT in wireless networks can be found in [86].

6.2 Evolutionary Game Theory

EG improves traditional GT by alleviating rationality assumption of players. It introduces a fully dynamic game model and hence emerges as an effective method to predict equilibrium for realistic models of player interactions in MANETs. We first introduce basic concepts in EG and then we discuss its applications to MANET cybersecurity. Description of EG includes measuring a fitness of a member by its probability of survival [82], defining Evolutionary Stable Strategy (ESS) [114] and formalizing Replicator Dynamics (RD) [120].

In EG, players form a population. All possible strategies that players can adapt are represented as a set of player types. Payoff in EG is the fitness, promoting more successful strategies to reproduce faster than less prosperous ones. A large population size and repeated interactions among randomly drawn players are typical assumptions in an EG. Therefore, the probability of a player encountering the same opponent twice is negligible and hence each interaction can be treated as an independent game.

ESS represents a strategy that cannot be invaded by any other strategy in a population. Let $u(s^*, s')$ be the payoff for a player with strategy s^* against an opponent strategy s'. Then, s^* is an ESS if one of the following conditions holds

$$u(s^*, s^*) > u(s', s^*)\,; \tag{7}$$

$$\big(u(s^*, s^*) = u(s', s^*)\big) \wedge \big(u(s^*, s') > u(s', s')\big) \tag{8}$$

where $\wedge$ represents the logical AND operation. Since ESS does not require players to be rational or reason perfectly, it represents a desirable NE refinement.

RD models the changes of each strategy in a population over time. It highlights the role of selection mechanism without considering mutation processes. In a typical RD, the players with the same pure strategy form a group whose representation in a population is reciprocal to the number of individuals in it [133]. For example, all individuals in group G_s play strategy s, while individuals in $G_{s'}$ play strategy s' for $s \neq s'$ and $s, s' \in S$, where S is the set of all strategies played in the game. Let I be a set of m players (population), and $|G_s|$ represent the number of individuals in group G_s, then $\sum_{\forall s \in S} |G_s| = m$ and $0 \leq |G_s| \leq m$. We define x_s (*population share*) as the frequency with which a strategy s is played (i.e., $x_s = \frac{|G_s|}{m}$ and $\sum_{s \in S} x_s = 1$) and $u(s, s')$ as the payoff for s when interacting with strategy s'. Expected payoff E_s for strategy s at a random match is

$$E_s = \sum_{\forall s' \in S} x_{s'}\, u(s, s') \tag{9}$$

Hence, the expected payoff for an individual interacting with a randomly drawn opponent is identical to her expectations when she plays a pure strategy against a player mixing strategies proportionally to population shares of all x_s in S. The average payoff in a population is

$$\phi = \sum_{\forall s \in S} x_s\, E_s \tag{10}$$

and RD of s is defined by [120] as

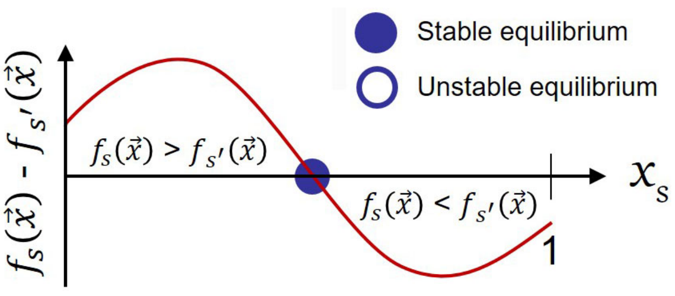
$$\dot{x}_s = x_s\, (E_s - \phi) \tag{11}$$

where $\dot{x}_s$ is the time derivative of frequency of s. Therefore, $\forall s \in S$ the *growth rate* ($\dot{x}_s / x_s$) of G_s is equal to the difference between the current payoff (fitness) of s and the average payoff of the population [133]. As the strategies in a population change, the expected payoff for a player alters and, eventually strategies with better payoffs grow. Equilibrium with two interacting strategies s and s' can be achieved when

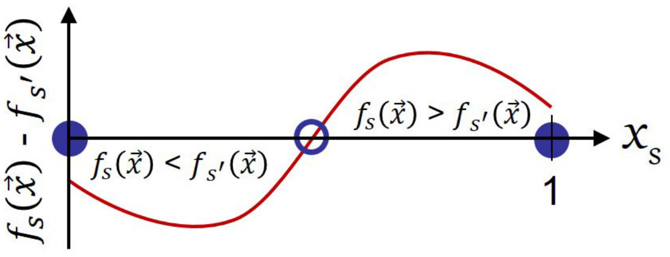
$$f_s(\mathbf{x}) = f_{s'}(\mathbf{x}) \tag{12}$$

where vector $\mathbf{x}$ defines a proportional composition of the population (i.e., $\mathbf{x} = (x_s, x_{s'}) = 1$) and $f_s(\mathbf{x})$ and $f_{s'}(\mathbf{x})$ are the fitness of strategies s and s', respectively.

Relative fitness of two interacting strategies is depicted in Figure 10, where the vertical axis marks the fitness difference of $f_s(\mathbf{s}) - f_{s'}(\mathbf{s})$ and the horizontal one is for the proportional representation of x_s in the population. After achieving a stable equilibrium, the balance in the population between two strategies does not change. If, however, x_s accidentally increases, $f_s(\mathbf{s})$ decreases in the next generation (with Eq. (11)) and lowers x_s back to the stable equilibrium. Similarly, if x_s decreases from a stable equilibrium, a higher fitness of s than s' will ensure that x_s grows. In Figure 10(a), the stable equilibrium (shown as a solid circle) is at the point where the vertical axis is zero (i.e., $f_s(\mathbf{s}) - f_{s'}(\mathbf{s}) = 0$). Some systems may not have a stable equilibrium. In an unstable equilibrium, an increase in x_s increases $f_s(\mathbf{s})$ and makes s dominate the entire population (i.e., $x_s = 1$) (with Eq. (11)). Similarly, a small decrease in x_s results in $x_s = 0$. Once an unstable equilibrium moves to $x_s = 0$ or 1, it becomes stable and will not recover. Figure 10(b) shows the unstable equilibrium (hollow circle) at the point where $f_s(\mathbf{s}) - f_{s'}(\mathbf{s}) = 0$, which becomes stable at $x_s = 0$ or 1 (solid circles).



(a)



(b)

Fig. 10: Behavior of (a) stable and (b) unstable equilibrium in an evolutionary game with two interacting strategies

In an EG application to a network routing problem, traditional GT assumptions are replaced with a learning process based on previous experiences of players [40]. The interactions among users for cooperative spectrum shearing as an EG is investigated in [127], where a population of secondary users implementing distributed learning algorithm converges to a stable state. A node spreading EG that runs at each MANET node to autonomously make movement decisions based on local data while the movement probabilities of possible next locations are assigned by a GA was presented in [79].

6.3 Evolutionary Games in Cybersecurity

In this section, we analyze implementations of cybersecurity techniques based on EG. Research reported in the literature can be grouped based on the employed cyber defense techniques (e.g., IDS, cryptosystems, and trust management techniques). Table 6 summarizes features of different EG techniques for improving network security in terms of their

Table 6: Features of EG-based methods used in network security implementations

	Characteristics of EG-based Techniques for Cybersecurity				
	Description	Players	Strategies	Utility Factors	Stable Point
[5]	Analysis of DDoS attacks	Users accessing BS and jammers	Elect to transmit or not (users) and (jammers)	Difference of SINR and transmission costs	ESS where either users or jammers transmitting
[99]	Active defense system for WSNs	Defending and attacking nodes	Deployment of security measures (defender) and attack or not (attacker)	Costs and rewards for attack, defense, and loss of a node	ESS where attacker do not attack
[8]	Select appropriate security measures in WSNs	Nodes representing defenders and adversaries	Strong or weak cryptography for defenders		Any mix of strategies can be supported as ESS
[102]	Broadcast authentication with preventing DoS	Uncertified and normal sensor nodes	Apply or not buffer-selection to prevent DoS	Value of broadcast packets and costs for corresponding security levels	Unique ESS for different scenarios
[110]	Enforce cooperation in MANETs	Senders, forwarders, and receivers	Forward or discard a packet (forwarder)	Following prisoner dilemma game's format	None of the strategy has been proven to be ESS
[57]	Physical layer security to keep away evasdroppers	All sensor nodes capable of adjusting own strategies	Select a level of transmission power	The secrecy level minus the power consumed	Multiple ESSs with all nodes obtaining adequate secrecy
[130]	Level classifications for transmissions with risks	Transmitter, receiver, and neighboring nodes	Attack, cooperate, or not cooperate	Cost, benefit, and risk levels	ESS under sufficient conditions
[17]	Malicious and HIDS node interactions	Malicious and regular (defending) nodes	Levels of monitoring and attacking	Attack cost, detection rate, and asset value	ESS for limited initial setup parameters
[112] [84]	Trust management system in WSN	All sensor nodes	Trust and distrust	Gain and cost for a selected strategy	ESS with nodes either trusting or distrusting their neighbors
[73]	Trust based packet forwarding	Forwarding and next-hop nodes	Between neighboring nodes incentivizing packet forwarding	History of neighbors actions	Fully cooperative network at low load

players, strategies, utility functions, and abilities to generate stable solutions.

EG-based Intrusion Detection An EG can be implemented at network nodes dynamically adjusting defense strategies to protect against selective forwarding, sinkhole, Sybil, false disruptive data injection, and data aggregation attacks [99]. An RD system representing attacker and defender actions can demonstrate an ESS equilibrium, where attackers do not attack and defenders do not have to defend [99].

In another application, EG is deployed to improve a physical layer security in WSNs [57]. A wiretap model over a noisy channel among a transmitter, a receiver, and an eavesdropper [137] is considered to obtain secrecy without a shared key in [57]. The rate of reliable communication at which an eavesdropper is unable to decode the information is maximized by using a non-cooperative secrecy-rate game. A transmitter communicates with a cluster head by adaptively adjusting transmission power level to maximize the secrecy level, hence it balances between conflicting objectives of intrusion prevention and energy preservation. The secrecy rate game evolves to multiple ESSs, each indicating appropriate strategies (transmission powers) of sensor nodes.

Distributed Denial of Service (DDoS) attacks occur when multiple adversaries coordinate their actions to diminish resources of a node. DDoS attacks carried by a group of jammers are investigated by an EG-based approach in [5], where jammers deny or degrade performance of users connected to a BS or an AP by injecting interference into communication spectrum. Users trying to enhance their Signal-to-Interference-plus-Noise Ratio (SINR) and jammers intending to lower users' SINR form two separate populations in an EG. In an asymmetric game, where a strategy may be more beneficial to a user than a jammer (or vice versa), an asymptotically stable ESS can be identified using RD.

An EG system to predict cyber attacks and to determine the best countermeasures to improve IDS performance in MANETs is proposed in [17], where a malicious node can choose levels of attack based on the value of the attacked asset and detection probability of its attack. Similarly, a defending node, that implements HIDS, calculates its own cost and level of monitoring. Payoff for a malicious node is high for undetected attacks while the defender benefits more from higher ratio of detection rate.

EG-*based Cryptosystems* An EG-based adaptive approach for dynamic selection of cryptographic protocols to obtain an appropriate defense mechanism is presented in [8]. Defenders can select a cryptographic protocol as deterrence against attackers by taking into account the cost of defense and reward for protection by a strong cryptography. Player payoffs can influence an asymptotical ESS [8].

DoS attacks on multi-level lightweight broadcast authentication protocols (i.e., μTESLA) can be modeled with EG. To resist DoS attacks, each packet with attached Message Authentication Code (MAC) should be buffered as long as its cryptographic key is secret. In DoS-resistant Authentication Protocol (DAP) [102], a node's strategy depends on actions of other nodes. To minimize packet size, nodes in DAP store μMACs as hashes.

EG-*based Trust and Reputation* In a game-based packet forwarding system [110], a node computes trustworthiness of its neighbors to decide its forwarding strategy. With an EG model, a cooperation is imposed among selfishly motivated nodes to make decisions based on outcomes of a repeated 2-player game and a long-term strategy. A cooperative or selfish trust value is calculated based on the route-request acceptance ratio and cooperation threshold to capture uncertainty of a trust. With a *direct reciprocity* model (e.g., help those who helped us) and history of node transactions, a cooperation can be attained despite selfish nodes. None of the strategies in [110] are evolutionary stable but they are resilient against being invaded by selfish strategies.

An EG based management for a WSN node can either play a *trust* strategy and provide forwarding services to its neighbors or *distrust* strategy without cooperation [112]. Using RD, evolution of trust in a population of sensor nodes is possible. Cooperation gain due to a particular neighbor's strategy may lead a population to ESS with all WSN nodes either participating in network activities or not cooperating. In another application, trust is built into an evolution process as an incentive for cooperation among nodes [84].

Repeated interactions among nodes in wireless ad hoc networks can be managed by EG with nodes observing their neighbors, predicting their actions based on a game model, and selecting the best neighbor to forward a packet [73]. The best next node is determined among all one-hop neighbors using trustworthiness and expected payoff of a game. A neighbor with a high ratio of dropped packets may be isolated to prevent potential DoS attacks.

7 Concluding Remarks

Evolutionary algorithms are effective tools in implementing cyber defense and counter measure mechanisms against a broad array of adversarial activities in MANETs or akin networks. This paper presents a tutorial containing evolutionary algorithms, their applications to cybersecurity, and a classification based on the attacks that they counteract and defense methods that they employ.

For cybersecurity of MANETs, we found successful applications of evolutionary algorithms in the forms of genetic algorithms, grammatical programming, grammatical evolution, and differential evolution. They implement trust management, policy evaluation, and cryptographic functions to protect MANETs. Swarm intelligence methods of ant colony optimization, artificial bee colony, and particle swarm optimization are promising tools for tackling many network-related problems. Although they may improve intrusion detection, trust, cryptography, and jamming defense mechanisms, their applications in MANETs are not widespread. Popularity of artificial immune systems in image processing and pattern recognition was not matched in providing cybersecurity for MANETs, with only limited applications to intrusion detection with dynamic rule updates. Evolutionary games are mainly used in MANETs to identify strategies for attackers and defenders and to determine stable solutions in IDS, trust, and cryptographic designs.

The algorithms used in implementation of evolutionary computation methods outlined in this paper are typically light-weight with low computational complexities. There is a linear relation between search space size and computational cost of these algorithms to obtain optimal or near-optimal solutions. They can easily handle large search spaces involving, for example, thousands of variables over thousands of generations.

References

1. 2016 U.S. government cybersecurity report. URL `https://cdn2.hubspot.net/hubfs/533449/SecurityScorecard_2016_Govt_Cybersecurity_Report.pdf`. (Accessed: 2018-03-14)
2. KDD Cup 1999 Data. URL `kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`. (Accessed: 2018-03-14)
3. A Java-based evolutionary computation and genetic programming research system (2013). URL `https://cs.gmu.edu/~eclab/projects/ecj/`. (Accessed: 2018-03-14)
4. The department of defense cyber strategy (2015). URL `https://www.defense.gov/Portals/1/features/2015/0415_cyber-strategy/Final_2015_DoD_CYBER_STRATEGY_for_web.pdf`. (Accessed: 2018-03-14)
5. Abass, A., Hajimirsadeghi, M., Mandayam, N., Gajic, Z.: Evolutionary game theoretic analysis of distributed denial of service attacks in a wireless network. In: Annual Conf. on Information Science and Systems (CISS) (2016)
6. Aljarah, I., Ludwig, A.: Mapreduce intrusion detection system based on a particle swarm optimization clustering algorithm. In:

IEEE Congr. on Evolutionary Computation (CEC), pp. 955–962 (2013)
7. Arenella, G., de Santis, F., Malandrino, D.: BeeAdHocServiceDiscovery: A MANET service discovery algorithm based on bee colonies. In: 11th Int'l. Conf. on Informatics in Control, Automation and Robotics (ICINCO), pp. 1–6 (2014)
8. Arora, S., Singhb, P., Gupta, A.: Adaptive selection of cryptographic protocols in wireless sensor networks using evolutionary game theory. In: Int'l. Conf. on Information Security & Privacy (ICISP'15), 78, pp. 358–366. Procedia Computer Science (2015)
9. Bäck, T.: Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In: IEEE World Congr. on Computational Intelligence, Proc. of the 1st IEEE Conf. on Evolutionary Computation, pp. 9–16 (1994)
10. Bagga, E., Adhikary, E.: A review on various protocols and security issues in MANET. In: Int'l. J. of Advanced Research in Computer and Communication Engineering, vol. 3, pp. 7478–7482 (2014)
11. Baker, J.: Reducing bias and inefficiency in the selection algorithm. In: J.J. Grefenstette (ed.) Proc. of the 2nd Int'l. Conf. on Genetic Algorithms on Genetic Algorithms and Their Application. Lawrence Erlbaum Associates, Publishers (1987)
12. Ball, M., Qela, B., Wesolkowski, S.: A review of the use of computational intelligence in the design of military surveillance networks. In: Recent Advances in Computational Intelligence in Defense and Security, vol. 621, pp. 663–693. Springer (2015)
13. Barani, K.: A hybrid approach for dynamic intrusion detection in ad hoc networks using genetic algorithm and artificial immune system. In: Iranian Conf. on Intelligent Systems (ICIS), pp. 1–6. IEEE (2014)
14. Barani, K., Abadi, M.: BeeID: Intrusion detection in AODV-based MANETs using artificial bee colony and negative selection algorithms. Int'l. J. of Information Security (ISC) **4**(1), 25–39 (2012)
15. Barolli, L., Koyama, A., Shiratori, N.: A QoS routing method for ad-hoc networks based on genetic algorithm. Proc. of the 14th Int'l. Work. on Database and Expert Systems Applications (DEXA) pp. 175–179
16. Beni, G., Wang, J.: Swarm intelligence. In: Proc. of Seventh Annul Meeting of the Robotics Society of Japan, pp. 425–428. RSJ Press (1989)
17. Bouhaddi, M., Adi, K., Radjef, M.: Evolutionary game-based defense mechanism in the MANETs. In: Proc. of the 9th Int'l. Conf. on Security of Information and Networks (SIN'16), pp. 88–95 (2016)
18. Brameier, M., Banzhaf, W.: A comparison of linear genetic programming and neural networks in medical data mining. IEEE Trans. on Evolutionary Computation **5**(1), 17–26 (2001)
19. Brameier, M., Banzhaf, W.: Linear Genetic Programming. Springer (2007)
20. Bretscher, P., Cohn, M.: A theory of self-nonself discrimination. Science **169**, 1042–1049 (1970)
21. Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraulaz, G., Bonabeau, E.: Selforganization in biological systems. Princeton University Press (2001)
22. de Campos, L., de Oliveiraa, R., Roisenbergb, M.: Optimization of neural networks through grammatical evolution and a genetic algorithm. Expert Systems with Applications **46**, 368–384 (2016)
23. Caro, G.D., Dorigo, M.: AntNet: distributed stigmergetic control for communication networks. J. of Artificial Intelligence Research **9**, 317–365 (1998)
24. Caro, G.D., Ducatelle, F., Gambardella, L.: AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. Emerging Telecomunications Technologies **16**, 443–455 (2005)
25. Castro, L.D., Timmis, J.: Artificial immune systems: A novel paradigm to pattern recognition. Artificial Neural networks in pattern Recognition **1**, 67–84 (2002)
26. Cho, J., Swami, A., Shen, I.: A survey on trust management for mobile ad-hoc networks. In: IEEE Communications Surveys & Tutorials, vol. 14, pp. 562–583 (2011)
27. Crosbie, M., Spafford, E.: Applying genetic programming to intrusion detection. AAAI Technical Report FS-95-01, Purdue University (1995)
28. Şen, S.: A survey of intrusion detection systems using evolutionary computation. In: Bio-Inspired Comput. in Telecomm, pp. 73–94. Elseiver (2015)
29. Şen, S., Clark, J.: A grammatical evolution approach to intrusion detection on mobile ad hoc networks. In: Proc. of the 2nd ACM Conf. on Wireless Network Security (WiSec '09), pp. 95–102. AMC, New York (2009)
30. Şen, S., Clark, J., Tapiador, J.: Power-aware intrusion detection in mobile ad hoc networks. In: Ad Hoc Networks, vol. 28, pp. 224–239 (2010)
31. Cui, W., Brabazon, A., O'Neill, M.: Evolving dynamic trade execution strategies using grammatical evolution. Int'l. J. of Financial Markets and Derivatives **2**(1-2), 4–31 (2011)
32. Dal, D., Abraham, S., Abraham, A., Sanyal, S., Sanglikar, M.: Evolution induced secondary immunity: An artificial immune system based intrusion detection system. In: 7th Int'l. Conf. on Computer Information Systems and Industrial Management Applications (CISIM) (2008)
33. Dorigo, M.: Optimization, learning and natural algorithms. Ph.D. thesis, Politecnico di Milano, Italy (1992)
34. Dorigo, M., Socha, K.: An introduction to ant colony optimization. In: T. Gonzalez (ed.) Approximation Algorithms and Metaheuristics. CRC Press (2007)
35. Dorronsoro, B., Ruiz, P., Danoy, G., Pigne, Y., Bouvry, P.: Evolutionary algorithms for mobile ad hoc networks. Wiley (2014)
36. Eidenbenz, S., Kumar, V., Zust, S.: Equilibria in topology control games for ad hoc networks. Mobile Networks and Applications **11**(2), 143–159 (2006)
37. Elsayed, S., Sarker, R., Slay, J.: Evaluating the performance of a differential evolution algorithm in anomaly detection. In: 2015 IEEE Congr. on Evolutionary Computation (CEC), pp. 2490–29497 (2015)
38. Farmer, J., Packard, N., Perelson, A.: The immune system, adaptation, and machine learning. Physica **22**, 187–204 (1986)
39. Fidalcastro, A., Baburaj, E.: An advanced grammatical evolution approach for intrusion detection on multicast routing in MANET. In: 2014 Int'l. Conf. on Information Communication and Embedded Systems (ICICES) (2014)
40. Fischer, S., Vöcking, B.: Evolutionary game theory with applications to adaptive routing. Euro. Conf. on Complex Systems (ECCS) pp. 1–6 (2005)
41. Fudenberg, D., Tirole, J.: Game theory. The MIT Press (1991)
42. Gairing, M., Monien, B., Tiemann, T.: Selfish routing with incomplete information. In: ACM Symp. on Parallel Algorithms and Architectures, pp. 203–212 (2005)
43. Garnier, S., Gautrais, J., Theraulaz, G.: The biological principles of swarm intelligence. Swarm Intelligence **1**(1), 3–31 (2007)
44. Greensmith, J., Aickelin, U., Cayzer, S.: Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In: Int'l. Conf. on Artificial Immune Systems (ICARIS), pp. 153–167 (2005)
45. Gundry, S., Zou, J., Kusyk, J., Sahin, C., Uyar, M.: Differential evolution based fault tolerant topology control in MANETs. In: IEEE Military Communications Conf. (MILCOM) (2013)
46. Güneş, M., Sorges, U., Bouazizi, I.: ARA the ant-colony based routing algorithm for MANETs. In: Int'l. Work. on Ad Hoc Networks (IWAHN), pp. 79–85 (2002)

47. Harmer, P., Temple, M.: An improved LFS engine for physical layer security augmentation in cognitive networks. In: Int'l. Conf. on Computing, Networking and Communicatin, Cognitive Computing and Networking, pp. 719–723 (2013)
48. Harmer, P., Williams, M., Temple, M.: Using DE-optimized LFS processing to enhance 4G communication security. In: Proc. of Int'l. Conf. on 20th Computer Communicqtion and Networks (ICCCN) (2011)
49. Harsanyi, J.: Games with incomplete information played by 'Bayesian' players, parts i, ii and iii. Management Science **14**(3) (1967)
50. van Hoesel, S.: An overview of Stackelberg pricing in networks. Euro. J. of Operational Research **189**(3), 1393–1402 (2008)
51. Hofmeyr, S., Forrest, S.: Architecture for an artificial immune system. Evolutionary Computation **8**(4), 443–473 (2000)
52. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975)
53. Huang, J., Berry, R., Honig, M.: Auction-based spectrum sharing. Mobile Networks and Applications **11**(3), 405–418 (2006)
54. Ibrahim, M., Alexander, B.: Evolving decision-making functions in an autonomous robotic exploration strategy using grammatical evolution. In: Int'l. Conf. on Intelligent Robots and Systems (IROS), pp. 4340–4346. IEEE (2013)
55. Indirani, G., K.Selvakumar: Handling cross-layer attacks using neighbors monitoring scheme and swarm intelligence in MANET
56. Ji, Z., Liu, K.: Multi-stage pricing game for collusion-resistant dynamic spectrum allocation. IEEE Journal on Selected Areas in Communication **26**(1) (2008)
57. Jiang, G., Shen, S., Hu, K., Huang, L., Li, H., Han, R.: Evolutionary game-based secrecy rate adaptation in wireless sensor networks. Int'l. J. of Distributed Sensor Networks **2015** (2015)
58. Jiang, T., Baras, J.: Ant-based adaptive trust evidence distribution in MANET. In: 24th Int'l. Conf. on Disttributed Computig Sysystems Workshops (2004)
59. Jim, L., Gregory, M.: A review of artificial immune system based security frameworks for MANET. Int'l. J. of Communications, Network and System Sciences **9**(1), 1–18 (2016)
60. Johnson, D., Teredesai, A., Saltarelli, R.: Genetic programming in wireless sensor networks. In: Genetic Programming, vol. 3447, pp. 96–107. Springer (2005)
61. Kadri, B., Moussaoui, D., Feham, M.: Agreensmith. Int'l. Network Security **15**(1), 2231–5268 (2013)
62. Kamal, A., Warip, M., Elshaikh, M., R.Badlishah: Differential evolution (DE) algorithm to optimize Berkeley-MAC protocol for wireless sensor network (WSN). J. of Theoretical and Applied Information Technology **89**(2), 314–319 (2016)
63. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report TR06, Erciyes University, Computer Engineering Department (2005)
64. Karaboga, D., Basturk, B.: Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In: Foundations of Fuzzy Logic and Soft Computing, vol. 4529, pp. 789–798. Springer (2007)
65. Karaboga, D., Okdem, S., Ozturk, C.: Cluster based wireless sensor network routing using artificial bee colony algorithm. Wireless Networks **18**(7), 847–860 (2015)
66. Kavitha, K., Kumari, S., Phil, M.: Particle swarm optimization for adaptive anomaly-based intrusion detection system using fuzzy controller. Int'l. J. of Computer Trends and Technology (IJCTT) **4**(10), 3536–3541 (2013)
67. Kennedy, J., Eberhart, R.: A new optimizer using particle swarm theory. In: Proc. of the 6th Int'l. Symp. on Micro Machine and Human Science, pp. 39–43 (1995)
68. Kephart, J.: A biologically inspired immune system for computers. In: R. Brooks, P. Maes (eds.) Proc. of 4th Int'l. Work. on Synthesis and Simulatoin of Living Systems, pp. 130–139. MIT Press (1994)
69. Khannous, A., Rghioui, A., Elouaai, F., Bouhorma, M.: Securing manet using the integration of concepts from diverse immune theories. J. of Theoretical and Applied Information Technology **88**(1), 35–50 (2016)
70. Kim, I., de Weck., O.: Variable chromosome length genetic algorithm for progressive refinement in topology optimization. Structural and Multidisciplinary Optimization **29**, 445–456 (2005)
71. Kim, Y., Lee, M.: Scheduling multi-channel and multi-timeslot in time constrained wireless sensor networks via simulated annealing and particle swarm optimization. IEEE Communications Magazine **52**, 122–129 (2014)
72. Komali, R., MacKenzie, A., Gilles, R.: Effect of selfish node behavior on efficient topology design. IEEE Transactions on Mobile Computing **7**(9) (2008)
73. Komathy, K., Narayanasamy, P.: Secure data forwarding against denial of service attack using trust based evolutionary game. In: IEEE Vehicular Technology Conference (VTC), pp. 31–35 (2008)
74. Koza, J.: Genetic programming: On the programming of computers by means of natural selection. The MIT Press (1992)
75. Koza, J.: Genetic programming as a means for programming computers by natural selection. In: Statistics and Computing, vol. 4, pp. 87–112 (1994)
76. Kuila, P., Jana, P.: A novel differential evolution based clustering algorithm for wireless sensor networks. In: Applied Soft Computing, vol. 25, pp. 414–425 (2014)
77. Kumar, B., Sekhar, P., Papanna, N., Bhushan, B.: A survey on MANET security challenges and routing. Int'l. J. of Computer Technology & Applications **4**(2), 248–256 (2013)
78. Kusyk, J.: Game-theoretic and bio-inspired techniques for self-positioning autonomous mobile nodes. Ph.D. thesis, CUNY, New York, NY (2011)
79. Kusyk, J., Sahin, C., Uyar, M., Urrea, E., Gundry, S.: Self organization of nodes in mobile ad hoc networks using evolutionary games and genetic algorithms. J. of Advanced Research **2**, 253–264 (2011)
80. Kusyk, J., Urrea, E., Sahin, C., Uyar, M.: Game theory and genetic algorithm based approach for self positioning of autonomous nodes. Int'l. J. of Ad Hoc & Sensor Wireless Networks **16**, 93–118 (2012)
81. Laskari, E., Meletiou, G., Vrahatis, M.: Problems of cryptography as discrete optimization tasks. Nonlinear Analysis: Theory, Methods & Applications **63**, 831–837 (2004)
82. Lewontin, R.C.: Evolution and the theory of games. J. of Theoretical Biology **1**, 382–403 (1961)
83. Li, C., Antonsson, A.: Variable length genomes for evolutionary algorithms. In: Genetic Programming 1996: Proc. of the First Annual Conf., pp. 512–520. MIT Press (2000)
84. Li, Y., Xu, H., Cao, Q., Li, Z., Shen, S.: Evolutionary game-based trust strategy adjustment among nodes in wireless sensor networks. Int'l. J. of Distributed Sensor Networks **11**(2), 1–12 (2015)
85. Lim, Y., Cheng, P., Clark, J., Rohatgi, P.: Policy evolution with grammatical evolution. In: Lecture Notes in Computer Science, Simulated Evolution and Learning, vol. 5361, pp. 71–80 (2008)
86. MacKenzie, A., DeSilva, L.: Game theory for wireless engineers, 1 edn. Morgan and Claypool Publishers (2006)
87. Montana, D.: Automatic tuning of communication protocols for vehicular ad hoc networks using metaheuristics. Engineering Applications of Artificial Intelligence **23**, 795–805 (2010)
88. Movahedi, Z., Hosseini, Z., Bayan, F., Pujolle, G.: Trust-distortion resistant trust management frameworks on mobile ad hoc networks: A survey. In: IEEE Communication Surveys & Tutorials, vol. 18, pp. 1287–1309 (2016)
89. Mukkamala, S., Sung, A., Abraham, A.: Modeling intrusion detection systems using linear genetic programming approach. In: Int'l. Conf. on Industrial, Engineering and other Applications of applied Intelligent Systems, pp. 633–642. Springer (2004)

90. Nadeem, A., Howarth, M.: A survey of MANET intrusion detection & prevention approaches for network layer attacks. In: IEEE Communications Surveys & Tutorials, vol. 15, pp. 2027–2045 (2013)
91. Nash, J.: The bargaining problem. In: Econometrica, vol. 18, pp. 155–162 (1950)
92. Nash, J.: Equilibrium points in n-person games. In: Proc. of the National Academy of Sciences of the United States of America, vol. 36, pp. 48–49 (1950)
93. Nash, J.: Non-cooperative games. In: Annals of Mathematics, vol. 54, pp. 286–295 (1951)
94. von Neumann, J., Morgenstern, O.: Theory of games and economic behavior. Princeton University Press (1944)
95. O'Neill, M., Ryan, C.: Grammatical evolution. In: IEEE Trans. on Evolutionary Computation, vol. 5, pp. 349–358 (2001)
96. Pan, M., Liang, S., Xiong, H., Chen, J., Li, G.: A novel bargaining based dynamic spectrum management scheme in reconfigurable systems. In: Int'l. Conf. on Systems and Networks Communications, pp. 54–54 (2006)
97. Parham, P.: The Immune System, 3 edn. Garland Science (2009)
98. Pathan, A.: Security of self-organizing networks: MANET, WSN, WMN, VANET. Auerbach Publications (2010)
99. Qiu, Y., Chen, Z., Xu, L.: Active defense model of wireless sensor networks based on evolutionary game theory. In: 6th Int'l. Conf. on Wireless Communications Networking and Mobile Computing (WiCOM), pp. 1–4 (2010)
100. Reina, D., Ruiz, P., Ciobanu, R., Toral, S., Dorronsoro, B., Dobre, C.: A survey on the application of evolutionary algorithms for mobile multihop ad hoc network optimization problems. Int'l. J. of Distributed Sensor Networks (2016)
101. Robinson, Y., Rajaram, M.: Energy-aware multipath routing scheme based on particle swarm optimization in mobile ad hoc networks. The Scientific World J. **2015** (2015)
102. Ruan, N., Gao, L., Zhu, H., Jia, W., Li, X., Hu, Q.: Toward optimal dos-resistant authentication in crowdsensing networks via evolutionary game. In: IEEE 36th Int'l. Conf. on Distributed Computing Systems (ICDCS), pp. 364–373 (2016)
103. Ryan, C., O'Neill, M.: Grammatical evolution: A steady state approach. In: Late Breaking Papers, Genetic Programming, pp. 180–185 (1998)
104. Sahin, C., Urrea, E., Uyar, M., Conner, M., Bertoli, G., Pizzo, C.: Design of genetic algorithms for topology control of unmanned vehicles. Special Issue of the Int'l. J. of Applied Decision Sciences (IJADS) on Decision Support Systems for Unmanned Vehicles **3**(3), 221–238 (2010)
105. Sahin, C., Urrea, E., Uyar, M., Conner, M., Hokelek, I., Bertoli, G., Pizzo, C.: Genetic algorithms for self-spreading nodes in MANETs. In: Proc. of the 10th annual conf. on genetic and evolutionary computation (GECCO), pp. 1141–1142 (2008)
106. Sahoo, D., Rai, S.C., Pradhan, S.: Threshold cryptography & genetic algorithm based secure key exchange for mobile hosts. In: IEEE Int'l. Advance Computing Conf. (IACC), pp. 1297–1302. IEEE (2009)
107. Sahoo, R., Singh, M., Sahoo, B., Majumder, K., Ray, S., Sarkar, S.: A light weight trust based secure and energy efficient clustering in wireless sensor network: Honey bee mating intelligence approach. In: First Int'l. Conf. on Computational Intelligence: Modeling Techniques and Applications (CIMTA), vol. 10, pp. 515–523 (2013)
108. Sarafijanovic, S., Boudec, J.L.: An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal, and memory detectors. In: Int'l. Conf. on Artificial Immune Systems (ICARIS), pp. 342–356 (2004)
109. Sasikala, E., Nandhakumar, N.: An intelligent technique to detect jamming attack in wireless sensor networks (WSNs). Int'l. J. of Fuzzy Systems **17**, 76–83 (2015)
110. Seredynski, M., Bouvry, P.: Evolutionary game theoretical analysis of reputation-based packet bforwarding in civilian mobile ad hoc networks. pp. 1–8 (2009)
111. Shamshirband, S., Anuar, N., Kiah, L., Rohani, V., Patković, D., Misra, S., Kahan, A.: Co-FAIS: Cooperative fuzzy artificial immune system for detecting intrusion in wireless sensor networks. J. of Network and Computer Applications **42**, 102–117 (2014)
112. Shen, S., Jiang, C., Jiang, H., Guo, L., Cao, Q.: Evolutionary game based dynamics of trust decision in WSNs. In: Int'l. Conf. on Sensor Network Security Technology and Privacy Communication System, pp. 1–4 (2013)
113. Sindhuja, K., Devi, S.P.: A symmetric key encryption technique using genetic algorithm. Int'l. J. of Computer Science and Information Technologies (IJCSIT) **5**, 414–416 (2014)
114. Smith, J., Price, G.: The logic of animal conflict. Nature **246**, 15–18 (1973)
115. Song, D., Heywood, M., Zincir-Heywood, A.: A linear genetic programming approach to intrusion detection. In: Genetic and Evolutionary Computation Conference (GECCO'03), vol. 2724
116. Storn, R.: On the usage of differential evolution for function optimization. In: Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conf. of the North American, pp. 519–523 (1996)
117. Storn, R., Price, K.: Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. J. of Global Optimization **23**(1) (1995)
118. Stützle, T., Hoos, H.: MAX-MIN ant system. Future Generation Computer Systems **16**, 889–914 (2000)
119. Tahta, U., Şen, Can, A.: Gentrust: A genetic trust management model for peer-to-peer systems. Applied Soft Computing **34**, 693–704 (2015)
120. Taylor, P., Jonker, L.: Evolutionary stable strategies and game dynamics. Mathematical Biosciences **16**, 76–83 (1978)
121. Troiano, L., Birtolo, C., Armenise, R.: Searching optimal menu layouts by linear genetic programming. J. of Ambient Intelligence and Humanized Computing **7**(2), 239–256 (2016)
122. Urrea, E., Sahin, C., Hokelek, I., Uyar, M., Conner, M., Bertoli, G., Pizzo, C.: Bio-inspired topology control for knowledge sharing mobile agents. Ad Hoc Networks **7**(4), 677–689 (2009)
123. Varshney, P., Bibhu, V., Sahoo, B., Gupta, A.: Quantitative review of malicious node detection of mobile ad-hoc network. Int'l J. of Computer & Mathematical Sci. (IJCMS) **3**, 102–110 (2014)
124. Victoire, T., Sakthivel, M.: A refined differential evolution algorithm based fuzzy classifier for intrusion detection. Euro. J. of Scientific Research **65**(2), 246–259 (2011)
125. Visu, P., Janet, J., Kannan, E., Koteeswaran, S.: Optimal energy management in wireless adhoc network using artificial bee colony based routing protocol. Euro. J. of Scientific Research **74**(2), 301–307 (2012)
126. Walters, R.: Cyber attacks on U.S. companies in 2016 (2016). URL `http://www.heritage.org/defense/report/cyber-attacks-us-companies-2016`
127. Wang, B., Liu, K., Clancy, T.: Evolutionary game framework for behavior dynamics in cooperative spectrum sensing. IEEE Global Telecommunications Conference (GLOBECOM) pp. 1–5 (2008)
128. Wang, X., Ding, L., Bi, D.: Reputation-enabled self-modification for target sensing in wireless sensor networks. IEEE Trans. on Instrumentation and Measuremens **59**, 171–179 (2010)
129. Wang, X., Osagie, E., Thulasiraman, P., Thulasiram, R.: HOP-NET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network. Ad Hoc Networks **7**, 690705 (2009)
130. Wang, X., Wu, Y., Ren, Y., Feng, R., Yu, N., Wan, J.: An evolutionary game-based trust cooperative stimulation model for large scale MANETs. Int'l. J. of Distributed Sensor Networks **13** (2013)

131. Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (AIRS): An immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines **5**(3), 291–317 (2004)
132. Wedde, H., Farooq, M., Pannenbaecker, T., Vogel, B., Mueller, C., Meth, J., Jeruschkat, R.: Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In: 7th Annual Conf. on Genetic and Evolutionary Computation (GECCO'05), pp. 153–160 (1995)
133. Weibull, J.: Evolutionary game theory. MIT Press (1997)
134. Weise, T.: Genetic programming for sensor networks. Technical report, University of Kassel, Germany (2006)
135. Wu, Y., Zhao, Y., Riguidel, M., Wang, G., Yi, P.: Security and trust management in opportunistic networks: A survey. Security and Communication Networks **8**(9), 1812–1827 (2015)
136. Wulandari, G., Rismawan, W., Saadah, S.: Differential evolution for the cryptanalysis of transposition cipher. In: 3rd Int'l. Conf. on Information and Comm. Tech. (ICoICT), pp. 45–48 (2015)
137. Wyner, A.: The wire-tap channel. The Bell System Technical J **54**, 1355–1387 (1975)
138. Zou, J., Gundry, S., Uyar, M., Kusyk, J., Sahin, C.: Bio-inspired topology control mechanism for unmanned underwater vehicles. In: Recent Advances in Computational Intelligence in Defense and Security, pp. 727–752. Springer (2016)