Applied Artificial Intelligence
An International Journal

ISSN: 0883-9514 (Print) 1087-6545 (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

A Greedy Ant Colony System for Defensive Resource Assignment Problems

Mônica De Rezende, Beatriz S. L. P De Lima & Solange Guimarães

To cite this article: Mônica De Rezende, Beatriz S. L. P De Lima & Solange Guimarães (2018) A Greedy Ant Colony System for Defensive Resource Assignment Problems, Applied Artificial Intelligence, 32:2, 138-152, DOI: 10.1080/08839514.2018.1451137

To link to this article: https://doi.org/10.1080/08839514.2018.1451137

APPLIED ARTIFICIAL INTELLIGENCE
2018, VOL. 32, NO. 2, 138–152
https://doi.org/10.1080/08839514.2018.1451137

Taylor & Francis
Taylor & Francis Group

A Greedy Ant Colony System for Defensive Resource Assignment Problems

Mônica De Rezende[a,b], Beatriz S. L. P De Lima[a], and Solange Guimarães[a]

[a]Civil Engineering Program, COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil; [b]Brazilian Navy Research Institute, Rio de Janeiro, Brazil

ABSTRACT

The weapon-target assignment (WTA) problem is crucial for strategic planning in military decision-making operations. It defines the best way to assign defensive resources against threats in combat scenarios. This is a NP-complete problem where no exact solution is available to deal with all possible scenarios. A critical issue in modeling the WTA problem is the time performance of the developed algorithms, subject only recently contemplated in related publications. This paper presents a hybrid approach which combines an ant colony optimization with a greedy algorithm, called the Greedy Ant Colony System (GACS), in which a multi colony parallel strategy was also implemented to improve the results. Aiming at large scale air combat scenarios, simulations controlling the algorithm time performance were executed achieving good quality results.

Introduction

Several barriers that existed in the past were not enough to guarantee the access control over a wide range of technological advances in the military field, as hardware devices, weapons, and military vehicles, making it difficult to prevent its misuse. Reducing risks in this scenario of uncertainties is not easy, what increases the interest in decision support systems capable of improving the results of defensive operations. Nevertheless, defense resources must be applied with rigorous criterions, in order to minimize losses on the defensive side, preferably at low cost. This is the picture depicted by the weapon-target assignment (WTA) problem, a well-known military issue in the operations research (OR) area (Luss e Roseinwein 1997).

Tactical planning is a difficult task even for computer-based systems due to the great amount of possible scenarios. Air defense operations are the most challenging ones since aerial targets have a high degree of maneuverability and can move exceedingly fast, demanding decisions in a very short time. Therefore, special heuristics have been developed to deal with WTA

CONTACT Solange Guimarães sol@coc.ufrj.br Civil Engineering Program, COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil.

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/UAAI.

problems, which, although sometimes not providing the global optimum, generate a high quality distribution of the available resources that minimizes the targets threat level in real time.

In this paper, we propose the hybridization of an improved ant colony optimization with a greedy algorithm for modeling an offensive strategy of the WTA problem, described in detail in the next section. Good quality solutions were obtained in tests within a controlled response time of the algorithm, even for large scenarios.

The assignment problem

The weapon-target assignment (WTA) problem is an example of combinatorial optimization. Given two sets, one being resources and the other being targets, the WTA problem consists of assigning each resource to a target minimizing target survival from the engagements or maximizing the assets survival after the enemy attack, therefore, finding the best way to allocate the existing resources against targets in tactical scenarios of defense operations. Hossein (1990) described two different defense models, target-based model (offensive strategy), and asset-based model (defensive strategy). The asset-based model involves strategic ballistic defense, demanding knowledge about which targets are headed to which assets, while the target-based model applies to offensive strategies types of combat, assuming that targets are known (Murphey 2000). We focused on the latter.

In target-based model, each incoming target receives a value that aggregates its threat level. When a resource engages a target, they form a pair with an associated damage probability, a statistical measure of the likelihood that the resource nullify the target. After an engagement, the target threat level decrease into an expected survival value, reflecting the resource damage probability effect over the target threat level. The WTA objective for this model is to minimize the total expected value of the targets that survive all engagements.

Another aspect of WTA model concerns the evolution of operations with time. The static model, called static weapon-target assignment (SWTA) problem considers that defense operations evolve in one single stage. Dynamic weapon-target assignment (DWTA) problem considers several defense cycles in an ongoing defense operation. Each cycle may be viewed as a SWTA, using a smaller amount of the total available resources, what characterizes the nondeterministic aspect of the model.

This article aims the target-based SWTA model, assigning each resource to a target so as to minimize the target survival values $f(\mathbf{S})$ presented in Equations (1) and (2):

$$\text{Minimize } f(\mathbf{S}) = \sum_{i=1}^{T} V_i \left(\prod_{j=1}^{W} \left(1 - P_{ij}\right)^{x_{ij}(\mathbf{S})} \right) \tag{1}$$

$$x_{ij}(\mathbf{S}) = \begin{cases} 1 \ if \ S_j = i \\ 0 \ otherwise \end{cases} \tag{2}$$

In Equations (1) and (2), the vector **S** is a feasible solution representation for the WTA problem, where each element S_j, $\forall\ j = 1, \ldots, W$, represents the assignment of the jth resource, the element position j, to the ith target, i.e. $S_j = i$, $\forall\ i = 1, \ldots, T$. T denotes the total number of targets and W the total number of available resources. Note that the representation **S** has the advantage of always producing a feasible solution, automatically satisfying the constraint $\sum_{i=1}^{T} x_{ij} = 1$, $\forall\ j = 1, \ldots, W$. Therefore, any resource can engage any target and more than one resource can engage the same target, but each resource engages only one target. Vector **V** represents the initial target values, a known threat level for each target of the current scenario. **P** is a given matrix of kill probabilities, where each element P_{ij} is an independent event probability of resource j to nullify target i in an engagement, also a known matrix. Therefore, the term $\left(1 - P_{ij}\right)$ is the probability of target i to survive if engaged by resource j. The number of feasible solutions for a problem involving T targets and W resources is T^W .

It is important to notice that when a resource j is assigned to the target i, its target value is updated to $V_i\left(1 - P_{ij}\right)$, diminishing the current threat level V_i of the target. Considering the target-resources ensemble, if k resources have already been assigned to the same target, the target value decreases of $\sum_{j=1}^{W} V_i P_{ij} \delta_{is_j}$, where δ_{is_j} is the Kronecker delta and $\sum_{j=1}^{W} \delta_{is_j} = k$. If no resource has been assigned to the target yet, the target value is equal to its initial threat value V_i.

Algorithms for WTA optimization

The explicit enumeration method would ensure a global solution for this problem if we could verify all possible weapon-target pair combinations for a specific scenario in a reasonable time. However, the SWTA problem is NP-complete, meaning that exhaustive search is not a viable method for this problem. Heuristic approaches are necessary as they can provide near optimal solutions in a shorter time.

First studies about the WTA problem appeared within the operations research field in the middle of the past century (Manne 1958). Analytical

approaches at that time include classic optimization methods, like implicit enumeration, branch and bound algorithm, integer linear programming and dynamic programming (Karasakal 2008), (Eckler e Burr 1972). Den Broeder et al. (1959) proposed the Maximal Marginal Return (MMR) approach to solve WTA. This greedy algorithm reaches the optimum only for a special type of WTA problem, since it considers the damage probability independent of resources: $P_{ij} = P_i$. Kolitz (1988) enhanced the MMR with results that outperformed the first one, fact vastly shown in other works (Johansson and Falkman 2011).

Heuristic methods inspired by nature have led to a new wave of proposals improving the modeling of the WTA problems. Based on genetic algorithms (GA), the work of Lee *et al.* (2002a) included a domain specific knowledge in the crossover operator and a local search mechanism. They proposed a greedy eugenics contribution to achieve better results. Other works focused on hybridization of different bio-inspired methods, combining them to improve the results by unifying the advantages offered by each one. Lee *et al.* (2002) hybridized ant colony optimization (ACO) and artificial immune system. In Wang and Chen (2012), a particle swarm optimization (PSO) algorithm and a cultural algorithm were combined, improving the local and global search scheme of PSO. Zeng et al. (2006), associated PSO and GA methods to solve WTA problems. In Lee and Lee (2005) and in Zhang, Xiaojing, and Chuanqing (2012), the combination of ACO and GA methods was considered.

Although all these algorithms results considerably minimize the initial threat level in combat scenarios, the computational time can be prohibitive high when the problem size increase. Decision making to an air threat defense scenario requires a reaction time in the range of 1 to 2 seconds (Johansson e Falkman 2010). Many works show results obtained after running algorithms for hours, compromising its practical applications (Lee, Lee and Su 2002a), (Lee, Su and Lee 2002b), (Lee, Su, and Lee 2003), (Lee and Lee 2005). Johansson and Falkman (2011) considered explicitly time limitation adopting it as a termination criterion in their work.

Julstrom (2009) showed the results obtained with two different types of GA, the string-coded and permutation-coded GA. In the string-coded GA the chromosome representation is the same as **S**, defined in Eq. (1). In the permutation-coded GA, the chromosome represents the order of resources to be followed by a MMR algorithm procedure. Julstrom (2009) reported improvements achieved by both GA algorithms when the solution obtained by the MMR algorithm of Kolitz (1988) is used to introduce a seed in the initial population. Seeded and unseeded versions of each GA algorithm were run 30 independent times using 15 randomly generated scenarios. Seeded versions of both GA algorithms performed better than unseeded versions

but, in large scenarios, the string coded GA performed better being suitable for air scenarios involving up to 40 resources and 80 targets. Nevertheless, the tests were limited to only one instance of each scenario size.

Johansson and Falkman ((2009), (2010), (2011)) reviewed deterministic and nondeterministic heuristic approaches for solving WTA problems: exhaustive search algorithm, two different MMR algorithms, naïve random search algorithm, greedy local search algorithm, ACO algorithm, GA and PSO algorithms. They reported better performance in their seeded versions of the GA and PSO algorithms, seeded by the MMR result of Kolitz (1988). The PSO implemented was the classical (Kennedy e Eberhart 1995). All algorithms computed the best fitness values obtained during the loop search of 1 second, but the tests were limited to small scenarios, involving up to 30 resources and 30 targets.

Ahuja et al. (2007) proposed an algorithm based on a partition WTA problem formulation where a very-large-scale neighborhood search algorithm (VLSN) were used to deal with integer non-linear optimization problems as the WTA. According to their results, the algorithm showed to be very efficient within a few seconds run, even though the comparison gaps were calculated between the VLSN solution and an estimated lower bound objective function value, considered as the optimal solution.

Our work hybridizes two approaches: the deterministic MMR method of Kolitz (1988) and the stochastic ACO. The study adapted to the WTA problem a new variant of an Ant Colony System (ACS) (Dorigo and Gambardella 1997) and adopted a multi colony parallel strategy to improve the performance. Our algorithm, GACS, achieved good results in a reasonable run time, including large scenarios of up to 80 resources and 80 targets.

Heuristic approaches adopted

Deterministic approach

Greedy algorithms solve optimization problems by selecting at each iteration the choice that best fits the objective function at that instant, choosing a local optimal solution. We used the greedy approach of MMR algorithm (Kolitz 1988), denoted here as MMR2, for comparison purposes since it is well known that this MMR algorithm provide better results (Johansson and Falkman 2011) than the MMR of Den Broeder et al. (1959), named here MMR1. Both algorithms have as input parameters vector **V** of target values and matrix **P** of kill probabilities, and, as output, vector **S**. Differently from MMR1, the MMR2 solution is independent to list sequences, since it scans all target values and all resources, searching for the pair that currently maximizes reduction of

threat values. Also, as in all such an algorithm, after the assignment of a resource to a target, the target value diminishes for the next iteration, reducing the threat level associated with this target. Both MMR algorithms provide solutions very quickly, even for large-scale scenarios. For non-complex input scenarios, MMR2 reaches the global optimum more frequently, being very effective. However, for complex situations, MMR1 and MMR2 solutions are frequently reduced to poor quality local optima.

Nondeterministic approach

Ant colony optimization (ACO) is a swarm intelligence population-based metaheuristic inspired by the cooperative behavior performed by real ants when they leave their nest in search for food (Dorigo and Stützle 2004). One of the most famous ACO applications is the traveling salesman problem (TSP). Initially, the ants start their search for food randomly. Since ants have low visual capacity they leave pheromones on the ground as a way to mark visited paths, helping them coming back to their nest, and also, sharing information with other ants since they all detect pheromone trails. Directions with intense levels of pheromone are highly attractive to them, probabilistic increasing the amount of ants passing by. Ants spend less time to go back and forth on shorter paths accumulating more pheromone on these trails, privileging shorter paths. Pheromone trail also evaporates over time, but at a lower rate than the pheromone deposit if ants continually use the trail, keeping its power of attraction.

The ACO metaheuristic uses directed graphs to model optimization problems. A population of m artificial ants is put to transverse the graph in order to find the shortest path. The full path crossed by an artificial ant represents a feasible solution for a problem to be evaluated and compared to other paths. In optimization problems, the evaporation process is a way to avoid premature convergence to local optima forgetting poor results.

Therefore, ACO is a probabilistic metaheuristic that generates nondeterministic solutions. Its great advantage lies on how well its internal mechanisms perform in the exploration and exploitation of the solution space, expanding the search scope if compared to problem-specific greedy methods. ACO is a constructive metaheuristic, that maintains an indirect memory of crossed paths by the use of pheromone deposits. Ant System (AS) was the first ACO algorithm proposed in literature (Dorigo and Gambardella 1997), where the pheromone was updated only after all ants had finished their routes being the amount of pheromone deposited by each ant a function of the route quality.

The management of an ant activity in Dorigo's work is centered in the probability, $P_{ij}^{k}(\mathrm{t})$, of an ant k, positioned at node i of the directed graph, to go

to the node j at a time t. The probability $P_{ij}^{k}(\mathrm{t})$ is given by Equation (3) (Dorigo and Stützle 2004).

$$P_{ij}^{k}(\mathrm{t}) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(\mathbf{t})\ \eta_{ij}^{\beta}(t)}{\sum_{l}\ \tau_{il}^{\alpha}(\mathbf{t})\ \eta_{il}^{\beta}(t)}, & j \wedge l\ \epsilon\ allowed\ l(t) \\ 0, & otherwise \end{cases} \tag{3}$$

The pheromone deposit at a time t in each trail is $\tau_{ij}(t)$ and the corresponding heuristic information is $\eta_{ij}(t)$, indicating the likelihood of an ant k displacement from the vertex i of the graph to another, j. The heuristic information variation throughout the evolution is specific to each application, modeling the key characteristics of the problem. Power parameters α and β are weights for setting the influence of pheromone deposit and heuristic information, respectively. Element, *allowed l(t)*, is the set of neighbor nodes of i allowed to be selected as j to form a path ij at time t. The pheromone is updated from one interaction to another by the expression shown in Equation (4).

$$\tau_{\mathrm{ij}}(\mathrm{t}+1) = (1-\rho)\ \tau_{\mathrm{ij}}(\mathrm{t}) + \Delta\tau_{\mathrm{ij}}(\mathrm{t}) \tag{4}$$

$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$ indicates total amount of pheromone deposited in ij path by all m ants that crossed ij path, being $\Delta\tau_{ij}^{k}(t)$ the amount deposited by the ant k; $(1-\rho)\ \tau_{\mathrm{ij}}(\mathrm{t})$ indicates the evaporation process, where $0 < \rho \leq 1$, is a constant rate which stand for the amount of pheromone evaporated between time t and $t + 1$, mimicking the natural process.

Ant Colony System (ACS) (Dorigo and Gambardella 1997) is a known improvement of Ant System (AS). ACS improves the original AS algorithm by the addition of three new features. The modified heuristic considers deposits and evaporation of pheromone occurring just on arcs of the "best-so-far" path and increases exploration by reducing pheromone deposits each time an ant crosses an arc. Furthermore, the elitism is indirectly contemplated by including an additional clause to the random probabilistic choice, with path probabilities distribution given by Equation (3). The ACS pseudo-random proportional rule that govern the movement of ant k, positioned on node i, towards node j, is shown in Equation (5):

$$j = \begin{cases} argmax_{l}\left\{\tau_{il}\left[\eta_{il}\right]^{\beta}\right\}, & if\ q \leq q0, \forall l\ \epsilon\ allowed\ l(t) \\ Eq.(3)\ distribution, & otherwise \end{cases} \tag{5}$$

q is a random real number uniformly distributed, ranging from 0 to 1, and q_0 is a pre-established constant value, $0 \leq q_0 \leq 1$. This extra rule in

Equation (5) specifies that ants move to the best possible path with probability q_0.

In ACS, pheromone is updated twice, locally and globally. Ants perform the local update immediately after crossing an arc. The idea is to reduce attractiveness exerted by the most visited arcs on subsequent ants, increasing the exploration; the rule is in Equation (6),

$$\tau_{ij}(t) = (1 - \xi)\tau_{ij}(t) + \xi\tau_0 \tag{6}$$

where $0 < \xi \leq 1$, is a constant, and τ_0 is the pheromone initial value. According to Dorigo and Stützle (2004), good values for these parameters are $\xi = 0.1$ and $\tau_0 = 1/(m f_{nn})$, where m is the total number of ants in the population and f_{nn} is a value produced by a heuristic known as nearest-neighbor, which is specific for the problem.

Only one ant, "the best so far", add pheromone at the end of each iteration. The ACS rule for the global pheromone update is shown in Equation (7):

$$\tau_{ij}(t+1) = \begin{cases} (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}^{best}(t) & if\ (i,j) \in R^{best} \\ \tau_{ij}(t) & otherwise \end{cases} \tag{7}$$

where, $\Delta\tau_{ij}^{best} = 1/L^{best}$, and L^{best} is the evaluation of the *best-so-far* route, in our application, $f(\mathbf{S})_{Best}$. We implemented two different versions of the ACS algorithm as detailed below.

ACS strategy for WTA

The ACS directed graph representation for the SWTA problem is simple: the nodes represent the resources, totalizing $W + 1$ nodes, posed in any desired consecutive order, W_1 to W_{w+1}, the last being the *End* node. The edges, linking consecutive nodes, represent the targets. Two consecutive nodes have always T linking edges, since one may assign two different resources to the same target. From the first resource W_1, the starting node, one of the m ants, an ant k, chooses probabilistically one of the T edges departing from W_1 node to W_2 to compose the assigned pair W_1T_i . The ant k moves from one node to another choosing probabilistically edges i in accordance to the rule defined in Equation (5), until it reaches the *End* node. Each pair W_jT_i of the graph is associated to an updated target value V_i and to a heuristic information value η_{ij}, defined as V_iP_{ij} . The target value V_i is updated after each edge choice, T_i . Each edge has also an actual pheromone value τ_{ij}, actualized at each choice, being its initial value $\tau_{ij} = 1/\left(mf(\mathbf{S})_{MMR1}\right)$, $\forall\ i = 1, ..T$, where $f(\mathbf{S})_{MMR1}$ is the target survival value for the solution found by the MMR1 algorithm (Equation (1)).

Step 1: Given W, T, V and P, initialize the number of ants as $m = \max(T, W)$, maximum number between resources and targets. Initialize pheromone matrix, $\tau_{ij} = \tau_0 = 1/(m\, f(\mathbf{S})_{MMR1})$. Initialize heuristic information matrix with elements $\eta_{ij} = V_i P_{ij}$, the maximum hostility reduction possibility associated with the *ij* path. Sensibility analysis will define β, ρ, ξ, q_0, and last time stamp (LTS), here predefined as LTS=1s, the termination condition, in seconds, that controls indirectly the number of iterations

Step 2: Control CPU time. If the current time is smaller than LTS continue to the next step, otherwise jump to Step 11.

Step 3: Control ant loop. Put an ant *k* to start out search. Start with *k*=1 and go until *k*=*m*.

Step 4: Control resource loop. Ant *k* start from resource *j*=1 (nest node), and go in sequence until end in end node, after passing throw resource *j*=*W*.

Step 5: Ant *k* at node *j* selects probabilistically one arc, target *i*, according to Eq. (5). *T* possible targets (arcs) are available from each resource node *j*.

Step 6: Apply a pheromone local update τ_{ij} according to Eq. (6), where τ_0 is given in Step 1.

Step 7: Update $V_i = V_i(1 - P_{ij})$. Update heuristic information from *l*=*j*+*1* to *W*, $\eta_{il} = V_i P_{il}$, according to the updated target value. Go to Step 4.

Step 8: Reset the target values to the original values. If the value obtained by ant *k* is equal or greater than the current best value $f(\mathbf{S})_{tBest}$ found, go to Step 3. Otherwise, update $f(\mathbf{S})_{tBest}$ and continue to Step 9.

Step 9: If $f(\mathbf{S})_{tBest}$ is smaller than the current *best-so-far* target survival value $f(\mathbf{S})_{Best}$, update $f(\mathbf{S})_{Best}$. Go to Step 3.

Step 10: Update global pheromone, according to Eq. (7), where $L^{best} = f(\mathbf{S})_{Best}$; reset the target values to the original values and reset heuristic information to the initial values $\eta_{ij} = V_i P_{ij}$, $\forall\, i = 1, T \text{ and } j = 1, W$, for the next cycle. Go to Step 2.

Step 11: Output $f(\mathbf{S})_{Best}$.

Figure 1. ACS algorithm for SWTA problem.

The traditional ACS algorithm implementation was adapted to the WTA problem to be compared with our proposed GACS algorithm. The implementation, based on the procedure of Johansson and Falkman (2010), follows the pseudo-code of Figure 1, a list of 11 self-explanatory steps. Step 1 shows all initialization details used in this adaptation to WTA problems.

The heuristic information is also updated during the current iteration, as seen in Step 7 of Figure 1, and reset to the original values each time ant *k* crosses the *ij* path. The update follows the rule, $\eta_{il}(t) = V_i\left(1 - P_{ij}\right) P_{il}, \forall l = j + 1, W$, where $V_i\left(1 - P_{ij}\right)$ is the updated target value of target *i* once the resource *j* is assigned to it. This value will influence the next ant *k* path choice, the next resource $(j + 1)$ assignment to a target.

Greedy ACS (GACS) strategy for WTA

The general heuristics of GACS is the same as the one given in Figure 1 with few modifications. Although ACO metaheuristic is well suited for parallel implementations, since each ant of the colony could construct a solution concurrently in an independent and asynchronous way, it is difficult to find papers discussing ACO parallel strategies for WTA problems (Lee, Lee e Su 2002b), (Gao et al. 2010). Our algorithm denoted GACS, employs a parallel technique improving

the search and optimizing CPU time. In GACS, 10 threads of our sequential ACS algorithm run concurrently, using the same initialization parameters. This model considers a distributed and non-cooperative population organization, where 10 colonies search in parallel for the best solution. A master procedure controls threads initialization parameters and evaluates solutions. Each colony has its local pheromone matrix, and they do not exchange data. Each colony keeps the master procedure informed about its "*best so far*" solution and they follow an asynchronous communication model.

MMR2 procedure searches for the best pair *ij* at each interaction obtaining a solution in which the pairs are not ordered according to the initialized sequence of resources, from $j = 1$ to W. Sorting the solution for $j = 1$ to W is needed to obtain **S**. Therefore, we tested if changing the order of the search in step 4 of ACS algorithm (Figure 1) would fasten the search. Instead of following the given order $j = 1$ to W, we used in the GACS directed graph the same construction order of the sequence of resources obtained by the MMR2 algorithm. The results showed the overall reduction of target survival values with this arrangement, even though modest. Initializations in GACS are the same as in the ACS implementation, explained in Step 1 of Figure 1, except for the number of ants in each thread, $m = 15$, and for the initial pheromone matrix, set to $\tau_{ij} = 1/\left(mf(\mathbf{S})_{MMR2}\right)$. Therefore, we used a fixed amount of 150 ants no matter the scenario.

The proposed solution adds an extra strategy for controlling repetitions of the best values inside the sequential algorithm of each population, avoiding premature convergence. Paths most visited receive a penalty strategy, reducing its attractiveness by interrupting accumulation of pheromone after a certain amount of CPU time. Therefore, the pheromone global update, step 10 of Figure 1, was modified by the introduction of a multiplier φ in the term responsible for the deposit process control of Equation (7), at first accelerating the convergence to exploit a local optimum, as shown in Equation (8).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \varphi\rho\Delta\tau_{ij}^{best}(t), \quad \forall(i,j) \in T^{best} \tag{8}$$

φ is an integer value ranging from 1 to 3. We have chosen to control the increase of φ value at each 200 ms of CPU time. When the maximum CPU time is over 600ms, the pheromone levels are restarted to its initial value to stimulate exploration, $\tau_{ij} = 1/(m\ f(\mathbf{S})_{\mathrm{MMR2}})$, as well as the factor, φ =1. The best solutions are kept in a memory list to avoid losing it during evolution.

Simulation and analysis of results

The comparison with other works has inherent difficulties beyond the formulation of the problem itself, since the confidential nature of the matter inhibits the

rise of public databases. Therefore, in our tests, we follow the same standard guidelines present in other publications (Ahuja et al. 2007), (Johansson 2010). We implemented and ran the algorithms in C++ language in a PC computer with processor Intel® Core™ i7, 2.2GHz and 6GB RAM.

Evaluating the performance of a SWTA algorithm demands the scenario representation in a specific instant of time, i.e., the number T of targets and W of resources, and the respective instance values of **V** and **P**. W and T define the scenario size, WxT, the search/space for current scenario problem. The values of **V** and **P**, generated randomly, define the instance of each scenario size. Following the proposal of Ahuja et al. (2007), the initial target values (**V**) were obtained randomly from a uniform distribution of integers, ranging from 25 to 100, and damage probabilities (**P**) are obtained from a uniform distribution, ranging from 0.6 to 0.9. We simulated 10 different problem instances for each scenario size to test the algorithm performance in different levels of complexity. CPU termination criterion was set to 1s. We executed two groups of tests: the first group contains less complex small scenarios with sizes up to 9×9; the second group includes examples with sizes varying from 5×10 to 100×100, comprising more complex scenarios.

Table 1 shows the minimized target survival values, averaged over 10 instances (Johansson and Falkman 2011) achieved by an exhaustive search algorithm (ES), MMR2, ACS, and GACS algorithms. In small-scale scenarios, the exhaustive search algorithm easily finds the global optimal among all possible solutions, an important certainty for the comparison. Table 1 omits CPU times, because, while MMR2 spent less than one millisecond in its search procedure, the time spent by exhaustive search algorithm were considerably higher (Johansson and Falkman 2009), but the results represent the global optimum. We performed 10 runs of 1s each, for both evolutionary algorithms, ACS and GACS, to compare the quality and robustness of the solutions. The results compare the ES search (Johansson and Falkman 2009) and MMR2 with the average of the best results in 10 runs for each instance (average over 10 best results of each scenario) and the average of the mean results in 10 runs for each instance, meaning an average over 100 results in each scenario.

GACS delivered the closer to the exact solution results, given by ES, in both mean and best results computations. ACS presented also a good minimization of the initial target survival values but the results are not as steady as in GACS with high standard deviation. The results in Table 1 shows the quality and robustness of the GACS algorithm, delivering the Best and Mean results very close to each other, with low deviation values, especially noticed if compared to ACS results, considering that ACS had a chance of 10 runs to search for a better solution. The good result achieved from GACS shows also that the steadiness of the algorithm is due to the heuristic implemented rather than only to the parallel evolution of populations since the ACS didn't have the same

Table 1. *f*(**S**) results comparing ES and MMR2 with ACS and GACS in 10 runs.

Scenario				ACS		GACS		ACS		GACS	
WxT	<V_i>	ES	MMR2	Best	σ	Best	σ	Mean	σ	Mean	σ
5 × 5	342.5	**52.64**	54.38	57.04	7.42	**53.32**	1.19	62.08	15.14	53.71	1.97
6 × 6	351.7	**55.20**	61.40	64.02	13.19	**59.42**	6.44	72.97	23.86	60.51	8.21
7 × 7	445.6	**67.99**	70.87	80.30	14.49	**70.74**	3.60	85.78	20.43	70.89	3.77
8 × 8	503.1	**73.07**	80.18	86.50	15.61	**76.35**	3.87	94.83	22.89	78.96	7.05
9 × 9	537.7	**80.88**	86.78	94.62	14.20	**84.58**	4.41	101.92	26.01	86.30	6.32

<V_i>, average of initial target values sum in each instance; **Best**, best *f*(**S**) in 10 runs averaged over 10 instances of each size (mean of 10 solutions); **Mean**, *f*(**S**) mean in 10 runs averaged over 10 instances of each size (mean of 100 solutions); **σ**, standard deviation.

performance in 10 separate runs. Obviously, results obtained in small-scale scenarios are easier and faster to obtain than in larger scenarios, but it is useful to show clearly the relative performance of the heuristic approaches.

The ACS and GACS results in the second group of tests for each given scenario size were also averaged over 10 different instances. Approaching the experiment to a real environment combat, only one run were performed to obtain the ACS and the GACS results in each instance, observing that GACS strategy comprises 10 parallel runs and only the best solution were taken into consideration. Notice that the termination criterion for both algorithms was the CPU time of 1 second, although the best result was achieved before 1s in most of the cases. Table 2 comprises the mean results and the corresponding mean CPU time in seconds for which the best result, $f(\mathbf{S})_{Best}$, was registered during the algorithm search of 1s. The majority of the scenarios sizes for this experiment are higher than the ones previously shown in Table 1. We also introduced scenarios with resource-targets relations not favorable for defense purposes ($T > W$). The algorithms parameters settings, chosen from preliminary tests, were, for basic ACS algorithm, $\beta = 1$, $\rho = 0.1$, £ = 0.1, and $q_0 = 0.5$. The parameters adopted for GACS were $\beta = 12$, $\rho = 0.1$, £ = 0.1, $q_0 = 0.75$, $\varphi = 1$ and $m = 15$, except for the last three scenarios sizes where $\beta = 17$.

The best overall performance (Table 2) is of the GACS algorithm for 11 over 12 scenarios, losing only to MMR2 in the last scenario, the biggest scenario size. The ACS algorithm showed the second best results, but its running times are the worst of all four tested algorithms. The algorithms achieved high reduction levels of initial target values (varying from 77% to 98%) for the cases where $W > T$, what is in line with the strongest defense conditions represented by this type of scenario. In the cases where $W < T$, typifying weak defense conditions, all algorithms achieved a considerably lower reduction of initial target values (varying from 51% to 57%) if compared to the strong defense condition, but still a very good result.

Table 2. Results for MMR2, ACS, and GACS.

Scenario		MMR2		ACS		GACS	
$W \times T$	$<V_i>$	Mean	T(s)	Mean	T(s)	Mean	T(s)
5 × 10	536	260.2	<0.001	237.0	<.001	**235.8**	<0.001
10 × 10	660	147.4	<0.001	114.4	<.001	**92.7**	<0.001
20 × 10	636	12.2	<0.001	17.3	0.151	**11.9**	<0.001
10 × 20	1213	569.7	<0.001	526.8	0.204	**526.6**	<0.001
20 × 20	1225	158.9	<0.001	202.6	0.271	**157.6**	<0.001
40 × 20	1238	19.0	<0.001	31.2	0.293	**18.7**	<0.001
20 × 40	2244	1057.8	<0.001	1100.7	0.209	**1057.3**	<0.001
40 × 40	2497	290.7	<0.001	395.2	0.507	**290.3**	0.020
80 × 40	2420	31.1	<0.001	63.4	0.555	**30.9**	0.154
40 × 80	5018	2140.6	<0.001	2247.1	0.518	**2140.4**	0.174
80 × 80	4983	548.6	<0.001	688.6	0.621	**547.8**	0.265
100 × 100	6246	**667.5**	<0.001	849.5	0.540	676.6	0.181

$<V_i>$ average of initial target values sum in each instance. **Mean**: f(**S**) averaged over 10 instances of each size; and run time, **T(s).**

Conclusions

In this paper, we proposed a new variant for the traditional ACS algorithm, named GACS, to solve a SWTA problem. GACS hybridizes ACS with MMR, a classical deterministic heuristic, in an implementation that adopts problem specific initialization of pheromone and heuristic information as well as specific selection strategy.

Nowadays, military defense operations simulations focus on the assignment of resources in complex scenarios. We tested GACS in large-scale problems and compare its performance with two greedy heuristics and an ACS algorithm. In critical scenarios simulations, even small improvements can make the difference justifying the usage of time-consuming algorithms with limited time criterion condition.

The usage of a multi colony parallel technique in the GACS implementation improved the quality of the solution if compared to the sequential implementation versions seen in other evolutionary algorithms, including the ACS here implemented, increasing the number of searches executed in a limited time period. Furthermore, allowing a master procedure to choose the best result among all the colonies fasten the process in an essential step to validate the algorithm application to real defense air operation in large scenarios. Also, the usage of MMR2 to initialize the pheromone paths as well as the order of search given in MMR2, together with the strategy for controlling repetitions of the best values to avoid premature convergence, seemed to provide a robust repetition of local minimum results throughout the runs, finding a consistently stable result with lower local optimum. The GACS algorithm achieved high reduction levels of initial target values for the cases where $W > T$, the strongest defense conditions type of scenario, but more importantly, achieved a competitive reduction of initial target values in weak defense conditions, where $W < T$.

Acknowledgments

We thank the Brazilian Navy Research Institute (IPqM) and the Brazilian funding agencies CNPq and CAPES for supporting this work.

Funding

We thank the Brazilian funding agencies CNPq and CAPES for supporting this work.

References

Ahuja, R. K., A. Kumar, K. C. Jha, and J. B. Orlin. 2007. Exact and heuristic algorithms for the weapon-target assignment problem. *Operations Research* 55:1136–46. doi:10.1287/opre.1070.0440.

Den Broeder, G. G., and R. E. Ellison. 1959. On optimum target assignment. *Operations Research* 7:322–26.

Dorigo, M., and T. Stützle. 2004. *Ant Colony Optimization. 1*. Cambridge, MA: MIT Press.

Eckler, A. R., and E. S. A. Burr. 1972. *Mathematical models of target coverage and missile allocation*. Technical Report DTIC:AD-A953517, Alexandria,VA: Military Operations Research Society.

Gao, D., G. Gong, L. Han, and N. Li. 2010. Application of multi-core parallel ant colony optimization in target assignment. Taiyuan: In: Proceedings of the International Conference on Computer Application and System Modeling (ICCASM), IEEE 514–18.

Goldberg, D. E., and J. R. Lingle. 1985. Alleles, loci, and the traveling salesman problem. *Proceedings of the First International Conference on Genetic Algorithms and their Applications*. NJ: Lawrence Erlbaum Associates, 154–59.

Hossein, P. A. 1990. *A class of dynamic nonlinear resource allocation problems*. PhD. Thesis, Massachussets Institute of Technology, Massachussets, USA.

Johansson, F. 2010. *Evaluating the performance of TEWA systems*. Phd.Thesis, University of Skövde, Sweden.

Johansson, F., and G. Falkman. 2009. An empirical investigation of the static weapon-target allocation problem. In *Procedings of the 3rd skövde worhshop on information fusion topics (SWIFT 2009)*, Edited by J. Laere, and J. M. R. Johansson, 63–67. Skövde: CSREA Press.

Johansson, F., and G. Falkman. 2010. A suite of metaheuristic algorithms for static weapon-target allocation. *Proceedings of the 2010 International Conference on Genetic and Evolutionary Methods*. CSREA Press, 132–38.

Johansson, F., and G. Falkman. 2011. Real-time allocation of firing units to hostile target. *Journal of Advances in Information Fusion* 6:187–99.

Julstrom, B. A. 2009. String-and permutation-coded genetic algorithm for the static weapon-target assignment problem. *In: Procs of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference (GECCO2009)*. New York: ACM, 2553-2558.

Karasakal, O. 2008. Air defense missile-target allocation models for a naval task group. *Computers & Operations Research* 35:1759–70.

Kennedy, J., and R. Eberhart. 1995. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 1942–48.

Kolitz, S. E. 1988. Analysis of a maximum marginal return assignment algorithm. *Procs. of the 27th conference on Decision and Control*, 2431–36.

Lee, Z. J., C. Y. Lee, and S. F. Su. 2002b. Parallel ant colonies with heuristics applied to weapon-target assignment problems. In: *Proceedings of the 7th Conference on Artificial Intelligence and Aplications*, 201–06. Taichung, Taiwan.

Lee, Z.-J., C.-Y. Lee, and S.-F. Su. 2002a. An immunity-based ant colony optimization for solving weapon target assgnment problem. *Applied Soft Computing* 2:39–47.

Lee, Z.-J., and W. L. Lee. 2005. A hybrid search algorithm with heuristics for resource allocation problem. *Information Sciences* 173:155–67.

Lee, Z. J., S. F. Su, and C. Y. Lee. 2003. Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics. *IEEE Transactions on Systems, Man and Cybernetics, Part B* 33:113–21.

Luss, H., and E. M. B. Rosenwein. 1997. Operations research applications: Opportunities and accomplishments. *European Journal of Operational Research* 97:220–44.

Manne, A. S. 1958. A target-assignment problem. *Operations Research* 6:346–51.

Murphey, R. A. 2000. *Target-based weapon target assignment problems*. In *Nonlinear assignment problems. vol. 7, in combinatorial optimization*, edited by P. M. Pardalos, and L. S. Pitsoulis, 39–53. New York: Springer US.

Wang, S., and W. Chen. 2012. Solving weapon-target assignment problems by cultural particle swarm optimization. *In*: *4th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*. Nanchang, Jiangxi: IEEE, 141–44.

Zeng, X., Y. Zhu, L. Nan, K. Hu, B. Niu, and X. He. 2006. Solving weapon-target assignment problem using discrete particle swarm optimization. *Proceedings of the 6th World Congress on Intelligent Control and Automation*. Dalian: IEEE, 3562-3565.

Zhang, J., W. Xiaojing, and X. Chuanqing. 2012. ACGA algorithm of solving weapon - target assignment problem. *Open Journal of Applied Sciences* 2:74–77.