Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2007)

Algorithmic Tools for Adversarial Games: Intent Analysis Using Evolutionary Computation

Tasha Vanesian and Ken Kreutz-Delgado
ECE Department
Univ. of California, San Diego
La Jolla, CA 92093 USA
{tcvanesi, kreutz}@ucsd.edu

George Burgin and David Fogel
Natural Selection, Inc.
3333 N. Torrey Pines Ct. Suite 200
La Jolla, CA 92137
{gburgin, dfogel}@natural-selection.com

Abstract **– Determining adversarial intent is important on a battlefield. In this paper, we propose a method for intent analysis using evolutionary computation. The proposed approach defines the model as an optimization problem then gives an algorithm for determining the parameters of the model in a Valuated State Space® domain. Example experiments using simulation data are discussed.**

I. INTRODUCTION

One of the keys to successful command and control is to understand the enemy's intent, particularly in light of incomplete and perhaps inaccurate information regarding social and cultural norms. It is inappropriate to project our own goals and aspirations onto the enemy. For example, in asymmetric warfare, the enemy's tactics and objectives may be radically different from our own and those of our allies. Given the same resources to allocate, if placed in obverse roles, our decisions on how best to use those resources might be very different. Furthermore, warfare is becoming increasingly that of semi-autonomous machines versus machines (e.g., swarms of semi-autonomous vehicles reacting to automated defense systems). Understanding the enemy's intent will therefore become less a matter of understanding the thinking of higher command authority and more a matter of inferring the adversary's intent based on a priori beliefs regarding their objectives and observed data reflecting the actual decisions that the enemy takes in real settings.

A novel combination of two technologies, evolutionary computation and the Valuated State Space® Approach used to quantify purpose, holds the promise of a general procedure for inferring the enemy's purpose in combat settings ranging from the campaign-level to the level of the individual. The capability described in this report is the result of the research and development undertaken that examined an automatic method for optimizing models of the adversary's intent, structured in a hierarchic form. The models were evolved (optimized) in light of data acquired on decisions made presuming the adversary is rational (i.e., attempting to maximize success as he defines it) using multi-agent adversarial games. The effort developed and tested software to assess the capability of this procedure in simulated combat settings of sufficient complexity. A collection of alternative models of the adversary's purpose was evolved dynamically over time, with evolutionary algorithms used to adapt those models in light of the most recent data describing the observed adversary's behavior. The feasibility of the approach has been assessed in a series of experiments using a statistical design to determine the computational requirements of the procedure and the identifiability of the adversary's objectives as a function of the complexity of the setting.

A. Modeling the Enemy as a Problem in System Identification

The challenge of inferring the enemy's purpose is similar to the problem of system identification. As indicated in Fig. 1, data are observed regarding the input-output behavior of a system. The goal is to develop a model of the transducer that maps the input stimulus into the output set of observed actions with the least error. The choice of models is often crucial in identifying an appropriate representation of the system. As will be discussed in the next section, the Valuated State Space (VSS) Approach provides the framework for modeling the adversary's mission. Once the class of models is chosen, a search is initiated for the best model of those available. This requires a criterion by which to measure the goodness-of-fit of the model and its associated parameter values to the observed data.

Caines [1] regarded identification as the invention and evaluation of scientific theories, that is, system identification is performed by using the scientific method. This method involves induction and inductive inference, followed by independent verification. It is an iterative process that facilitates gaining new knowledge about the nature of an observable environment.

Fogel et al. [2] remarked that there is a correspondence between the scientific method and natural evolution. In nature,

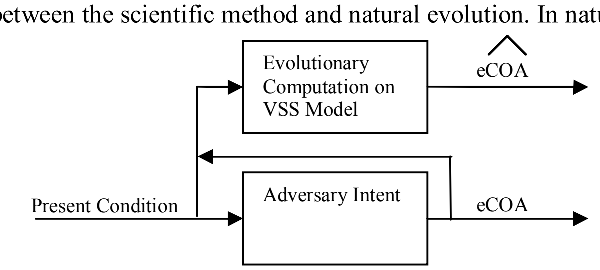


Fig. 1. Evolving models of the adversary using the Valuated State Space (VSS) structure to represent the adversary's intent. The adversary examines the present condition (state) and chooses a course of action (eCOA, for enemy course of action) that is believed to maximize his future success based on his intent. The adversary can be modeled using evolutionary computation by estimating the adversary's intent using the VSS structure. Alternative intents based on relative importance of parameters, degrees of achievement, and normalization are maintained as a population of competing ideas about the adversary's intent. Evolutionary optimization is used to maximize the correspondence between the prior observed conditions and eCOAs based on the hypothesized intent represented in each alternative estimated VSS and rationale for selecting the best eCOA. The best-evolved VSS at each decision point can be used to evaluate alternative eCOAs and arrive at eCOA_hat, the predicted adversary course of action, which can then be used as the basis for COA planning.

organisms serve as hypotheses about their environment. They are in essence predictions about the conditions that the future environment will present over their lifespan (at least to the point of reproduction). Those organisms that are good predictors of their surroundings survive to pass along their behaviors indirectly through their genes to their progeny. The progeny serves as modifications of the hypotheses, altered by mutation, sexual recombination, and/or many other variational processes. Over time, the evolutionary process of variation and selection performs the scientific method and continually optimizes the behaviors of the organisms involved in the process.

It is natural, therefore, to use algorithms that model evolution in system identification applications. Evolutionary algorithms can be very effective in system identification, particularly in simultaneously optimizing model parameters and the number of degrees of freedom in light of information-theoretic criteria.

B. Combat as a Game

Combat can be viewed in the framework of a game. Each SAM player faces a current situation and prospective future situations that will be a function of the decisions made on both (or more) sides over time. Each player has objectives that he wants to achieve and must be able to quantify the extent to which his overall mission is being accomplished at any point in the operation. Players can be expected to make decisions that they believe will improve the prospects of attaining their objectives. Note that these objectives may be related solely to each players' individual condition (e.g., survive or increase the probability of survival) and may be related directly to other players' conditions (e.g., eliminate the enemy, decrease the probability that the enemy would survive a particular tactical move). Thus, the objectives of players in combat settings must reflect both their individual concerns as well as the mutual attitudes of the involved players.

With a quantitative statement of each side's purpose, each side can evaluate future possible states in the combat by assessing the numeric effects on its overall degree of achievement. This is the fundamental basis for all computer programs that play strategy games, such as chess. A numeric evaluation function is used to assess features of a current position, or prospective future positions, and the output of this function is used to estimate the value of the input position (current or future state). A rationale such as minimax is used to favor one state over another based on the numeric evaluations across a range of alternative future states. The same procedure can be used in combat simulation, where present and future states are evaluated numerically in terms of how well they fit with a player's objectives (alternatively described as a "belief system") using the VSS Approach.

C. Quantifying Purpose with the Valuated State Space Approach

Optimal decision-making requires a well-defined purpose, for decision-making in the absence of a purpose is meaningless. To be well defined, a purpose must allow for trading off every possible allocation of resources in light of the parameters of concern, their relative importance, degree of criticality, and the degrees of achievement attained with respect to each parameter. These aspects of the purpose can be captured in the form of a VSS and normalizing function [3,4].

A VSS is used to express a purpose in terms of the relative worth of each of the significantly different outcomes, and therefore can be used to measure the overall worth of current and prospective situations. Achieving the most valuable class interval on each of the parameters corresponds with the state of the highest overall worth (a measure of 1.0, or 10 on a 10 scale, 100 on a percent scale). Achieving no success on any parameter corresponds with an overall worth of zero. Any intermediate state has some worth, depending upon the normalizing function.

In many situations, any level of achievement has some overall worth. A multiattribute utility function specifies the overall worth of any particular situation; that is, the VSS and its normalizing function yields a single overall measure for the worth of each significantly different situation.

When applying the VSS approach, decisions are almost always best made in the light of the other players' perceived, known, or assumed intent, capabilities, and motivation. It is, therefore, suitable to construct a similar representation of the purpose of each of the involved players (e.g., the United States, coalition members, and an opposing entity), then examine the joint state space that defines the game. This portrays a finite number of possible situations, those situations that are significantly different from any single or multiple players' points of view. There is a joint payoff in each cell/state for each of the players, this being a function of their marginal worth. Every sequence of moves and countermoves corresponds with a trajectory across states in the joint state space, there being some overall worth for that series of transitions.

D. Modeling the Adversary by Combining Evolutionary Computation and the Valuated State Space Approach

Just as a VSS can provide the justification for optimal decision-making for one's own forces in combat, the adversary can be viewed as using a VSS as the basis for his own decision-making. Therefore, viewing the challenge of modeling the adversary as a problem of system identification, a best estimate of the adversary's VSS (and normalizing function) can be gained by utilizing the empirical data on decisions that the adversary has made based on his presumed knowledge of the current and prospective future conditions. As indicated in Fig. 1, the desired process can utilize evolutionary computation to optimize models of the adversary's VSS over successive generations. At any point in time, the evolutionary algorithm can yield a single best or best collective set of estimates of the adversary's VSS, which can be used to infer how the adversary will respond in future possible settings. This will allow planners to gauge and evaluate the effectiveness of alternative plans under varying actions and reactions, and also facilitate semi-autonomous COA planning should that be desired.

In essence, the evolutionary approach was used to answer the question: Given the observed adversary decisions, what values of his prospective VSS and rationale (e.g., minimax) would yield the same or most similar decisions in the same settings? These values were optimized using evolutionary algorithms that were iterated for simulated generations at each decision point. The best-evolved estimates at each point in time were used as the basis for future estimation based on each next move and countermove as the combat unfolded. As the volume of empirical data on the adversary's decision-making grew so too did the accuracy of the estimates of the relative importance of his parameters of concern.

II. Method And Results

A. Framework for Analysis

The evolutionary approach to identifying the relative preference of the parameters of concern in the VSS can be framed as a mathematical optimization problem. Given n different observed OPFOR plans, assume that the preference (precedence ordering) of the plans is known. For example, for plans A, B, C, and D, it may be that the OPFOR orders the plans where $B > D > C > A$, meaning that B is preferred to D, which is preferred to C, which is preferred to A. The VSS results in a single numeric evaluation of a plan's overall worth, and therefore pathological cases where a strict order of preference is not obtained (e.g., $B > A$, $A > C$, but $C > B$) are excluded by this framework.

The VSS is constructed as a weighted sum of the degree of achievement on each parameter and the relative importance of the parameter. For example, suppose there are three parameters of concern, say, damage inflicted, survivability, and sensing performed. Suppose further that the relative importance for these parameters are w_1, w_2, and w_3, respectively, and the degree of achievement made with respect to each parameter is d_1, d_2, d_3, respectively. Then the overall worth under a linear normalization function is the dot product, or $w_1d_1 + w_2d_2 + w_3d_3$. The framework for evolving the relative importance weights, w_i, $i = 1$ to p, where p is the number of parameters, assumes knowledge of the degrees of achievement, d_i, and the precedence orderings of the plans, A, B, C, D, etc.

For the cases studied in this research, there were at most eight parameters ($p = 8$), corresponding to ammunition used, fuel remaining, damage inflicted, sensing performance, bomb damage assessment (BDA) performed, platform survival, returning to base, and reconnaissance, but many cases studied used $p < 8$. Given $\mathbf{d} = \{d_1, \ldots, d_p\}^{\mathrm{T}}$ and constraints:

$$0 \le d_i \le 1$$

$$0 \le w_i$$

$$\sum_i w_i = 1$$

the objective is to find the vector $\mathbf{w} = \{w_1, \ldots, w_p\}$ that satisfies the precedence ordering of the available plans. The constraint that all weights must sum to 1.0 means that all solutions $\mathbf{w}$ lie in a simplex. If the vector $\mathbf{w}$ can be identified uniquely, then as a consequence so can the purpose of the force using that VSS. Even if the vector $\mathbf{w}$ cannot be identified uniquely, relationships between the elements $w_1, \ldots, w_p$ may be obtainable.

With n available plans, there are at most $n!$ different precedence orderings. If a granularity for W, the space of all $\mathbf{w}$, is defined, there is a finite set of possible candidate vectors $\mathbf{w}$; however, an enumerative approach can only be effective for a small number of parameters p and available plans n. This approach has been implemented and the results are most easily visualized for $p = 3$. Typically, not all $n!$ precedence orderings can be satisfied because of the constraints on the weight space based on the available degrees of achievement.

For example, suppose there are four plans (A, B, C, and D) with three weight parameters ($p = 3$), with the following observed (or inferred) degrees of achievement (d_1, d_2, d_3):

A = [0.8 0.8 0.2];
B = [0.8 0.0 1.0];
C = [0.2 0.0 1.0];
D = [0.7 0.3 0.9];

Fig. 2 shows the precedence orderings as a function of the weights taken pairwise; due to the weight restrictions, w_3 is completely determined by w_1 and w_2. With four plans there are 4! = 24 different precedence orderings, but the constraint that the weights must be nonnegative and sum to one limit the available permutations such that only nine of these orderings are viable in light of the degrees of achievement above.

Fig. 3 shows an alternative view of the results with plots of the planes formed by calculating the score for each potential weight vector in the space W. Projecting the intersections of these score planes onto the w_1w_2 plane results in Fig. 2.

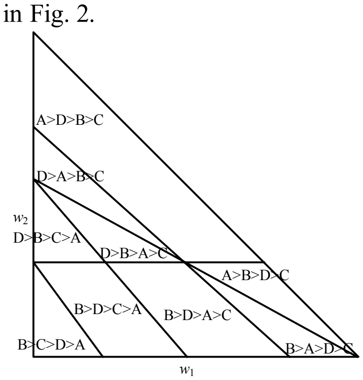


Fig. 2. Mapping of precedence relationships satisfied in each region for the given degrees of achievement by weight parameter. Each region is a different precedence relationship. Note that although there are 4! = 24 possible precedence orderings, only 9 of these orderings can be satisfied with the weight restrictions.

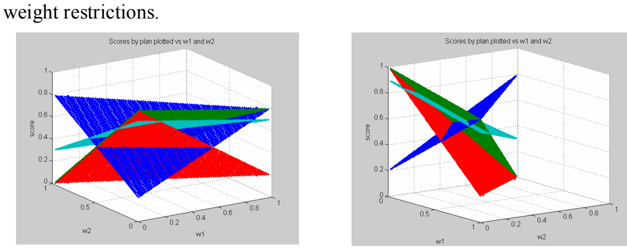


Fig. 3. Two views of the planes formed by the scores produced for each plan. Each color here corresponds to a different plan (A = Blue, B = Green, C = Red, D = Cyan). Scores are plotted as a function of w_1 and w_2. The weight w_3 is completely determined by w_1 and w_2 because of the weight space restrictions.

An enumerative search for satisfactory weights is very slow and memory intensive for larger values of p and n. An alternative approach is to use evolutionary computation to optimize the set of weights that satisfies the constraints induced by both the weight constraints and the precedence relationships.

An examination of the precedence relationship map (Fig. 2) and the score planes (Fig. 3) reveals some geometric insight into the search space. The desired region (the region in weight space that we want to find) satisfies all ${}_nC_2$ precedence relationships. Regions that border on this optimal region satisfy ${}_nC_2 - 1$ precedence relationships. Adjacent regions have only one difference in precedence relationships; this difference occurs where score planes intersect and the score plane ordering changes. This suggests that the optimal region need not be found by blind resampling but instead that an evolutionary approach can gradually approach and discover the optimal region.

B. Evolutionary Algorithm

The evolutionary algorithm employed in the experiments proceeded as follows:

Given the desired precedence ordering:

1. Set the number of initialization vectors (*popsize*) and number of generations (G). The number of initialization vectors is the number of survivors allowed at each generation.
2. Randomly initialize the population by taking a uniform sample from the ($p - 1$)-simplex (i.e., choose only from those weights that satisfy the weight constraints).
3. Repeat the evolutionary process of mutation and selection G times

A. *Mutate*:

Create *popsize* offspring from the parents:

Randomly choose a point on the simplex and interpolate between the parent and the random point based on the inverse of the parent fitness score (provides for taking small evolutionary steps for a parent with high fitness and large evolutionary steps for a parent with low fitness).

Create *popsize* offspring randomly:

Randomly choose *popsize* points on the simplex.

B. *Selection*

For each member of the population, compute the score for each plan, along with several fitness values based on the scores.

i.) Overall fitness, $F(i)$, which is the number of precedence relationships satisfied by the weights, the maximum is ${}_nC_2$.

ii.) Shared fitness, $F'(i)$. This fitness score is designed to keep the population from converging on a single point and encourages the population to spread across a region, thereby providing not just a single example of a point in a region that satisfies the constraints, but instead *the entire region*. The shared fitness for an individual solution is calculated following standard evolutionary computing literature with the following function:

$$F'(i) = \frac{F(i)}{\sum_{j=1}^{\mu} sh(d(i,j))}$$

where

$$sh(d) = \begin{cases} 1-(d/\sigma_{share})^{\alpha}, & if \quad d < \sigma_{share} \\ 0 & otherwise \end{cases}$$

and $d = d(i,j) = \| \mathbf{w}_i - \mathbf{w}_j \|^2$ and σ_{share} is chosen by the user as the "niching radius" for the neighborhood in which fitness must be shared. Fitness sharing reduces the fitness of each individual in a neighborhood based on the number of other solutions that are already in the neighborhood. This encourages more diversity.

iii.) Closeness-to-boundary fitness, $F_B(i)$. This fitness score is designed to encourage the population to move near the boundaries of a region. The boundaries of a region are where one of the precedence relationships is satisfied by an equality. For example, if the desired precedence relationship is $B > A > D > C$, then one boundary of the region will be created where $B = A > D > C$. Increased fitness is also awarded to solutions that lie on the weight restriction boundaries, so the following method is used:

$F_B(i) = 1 - \min(\min(\mathbf{w}), \min(\textit{score differences}))$,

where $\mathbf{w}$ is the vector of weights and *score differences* are the magnitudes of all pairwise differences in scores computed for each plan. Thus a point that lies exactly on a boundary achieves the maximum score of 1.

iv.) Shared closeness-to-boundary fitness, $F_B'(i)$. This uses the same sharing function described above, but considers only members of the population with the same overall fitness $F(i)$. This fitness measure encourages members of the population to spread out along boundaries.

Selection can be performed in one of two ways. The first method fills the desired region; the second fills the boundaries of the desired region. Both selection types use the same two-part selection concept, but they use different fitness measures to determine survivors.

1. Group the individuals in the population based on overall fitness $F(i)$ Select the group with the highest $F(i)$. If that group is not large enough to provide the sufficient number of survivors in the population (at least *popsize*), then the next highest scoring group is taken, and so forth, until enough survivors have been selected. At some point, the number of desired survivors will be exceeded by adding some group to the surviving population. To determine which members of this last group survive, go to step 2.

2. Choose the appropriate number of survivors from the last selected group based on their shared fitness score, $F'(i)$ for region filling, or based on their shared closeness-to-boundary score, $F_B'(i)$, for boundary filling.

Return the survivors as input to step 3A for the next generation.

For example, consider the same set of degrees of achievement used in the enumerative search example above, where:

A = [0.8 0.8 0.2];
B = [0.8 0.0 1.0];
C = [0.2 0.0 1.0];
D = [0.7 0.3 0.9];

Suppose that the observed precedence relationship is B > C > D > A. The task is to find the region of weights that satisfy this relationship in light of the degrees of achievement, thereby yielding the relative importance of the parameters. In Figs. 4 and 5, examples of several generations of evolution are shown. The color coding assigned reflects the fitness of each solution:

$F(i)$ = 1 – 3 Not seen in images
$F(i)$ = 4 => Yellow
$F(i)$ = 5 => Red
$F(i)$ = 6 => Blue

Note that $F(i) = 6$ is the maximum fitness that can be achieved for this example, and indicates the weight constraints and all precedence relationship constraints have been satisfied (as there are ${}_4C_2 = 6$ such precedence relationships).

Figs. 4 and 5 show boundary finding and region filling for the region $B > C > D > A$, iterated over the first 15 generations of evolutionary optimization. The first generation shown is at the end of the first mutation and selection cycle. By this generation, the population members with low fitness have been eliminated. The method reliably finds the boundaries or fills the appropriate precedence region given the available degrees of achievement on each parameter.

Having found the region in the simplex that is associated with the precedence ordering and the degrees of achievement on each parameter, Fig. 6 shows the six different possible regions of weight relationships (e.g., $w_1 < w_2 < w_3$). The intersection of these regions with the precedence regions of the simplex indicates the possible relative importance of the parameters, given the available information about precedence ordering and degrees of achievement on each parameter. As seen in Fig. 6, the triangular region found in Figs. 4 and 5 (lower-left corner) has two possible regions: $w_1 < w_2 < w_3$ or $w_2 < w_1 < w_3$. Therefore, from the information presented in the degrees of achievement and the precedence ordering of the plans (A, B, C, D), parameter three definitively has the greatest relative importance (because w_3 is always greatest), and the relative values of w_1 and w_2 are undecidable. This means that more observations would be required to facilitate differentiating the relative importance of the parameters in this case.

III. Results On Simulation Data

A. Use Best Evolved Plans

Several experiments were performed to test the algorithm on simulation data. For the first experiment, the following setup of terrain, platform, and sequence of revealed combat objects was used (see Fig. 7a-e). Plans were evolved with our Evolutionary Server/Client for each of the setups with the following weights on the parameters in the VSS: Fuel = 5, Damage = 7, Platform Survival = 10, AtBase (Return to Base) = 10, and all other possible parameters = 0.

The evolutionary algorithm used a population of plans at each generation. Attention was focused on the best plan at each of the five stages indicated in Fig. 7(a-e). Each of these plans was scored against the state of combat objects when it was created and the state of combat objects at each future time in light of the newly revealed targets/threats. Table 1 summarizes the degree of achievement that each plan had on each parameter and the total score achieved by that plan.

Table 1 shows the state of combat objects when the plan was created in the first column. The second column indicates

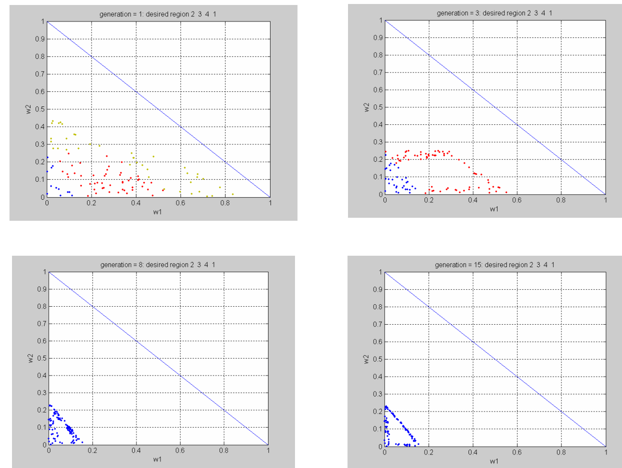


Fig. 4. Sequence of generations demonstrating boundary finding of the region $B > C > D > A$.

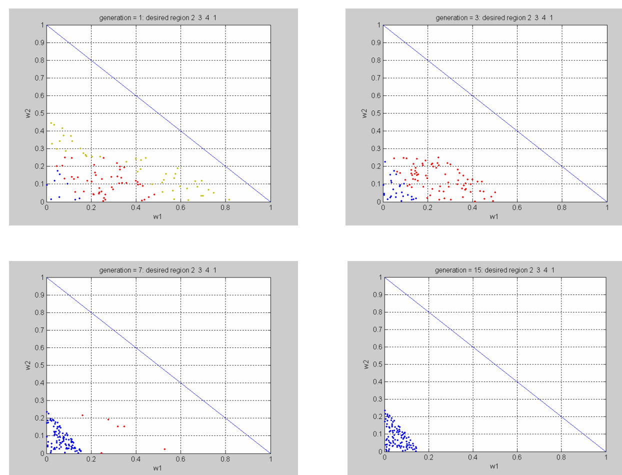


Fig. 5. Sequence of generations demonstrating region filling of the region $B > C > D > A$.

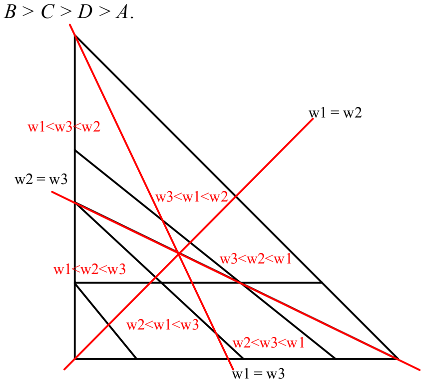


Fig. 6. The relative importance of the weights for each of the six different possible relationships.

the state of combat objects against which the plan was scored. The next four columns give degrees of achievement for fuel, damage, platform (survival), and atBase (returning safely to base). The final column gives the overall weighted score. The top row gives the names of the non-zero parameters and the value assigned in the VSS. The second row gives the normalized values of the VSS. For simplified reference in text, we will refer to scoring a particular plan against a particular set of combat objects as (state of combat objects when the plan was created, state of combat objects used for scoring). For example, "2 Threat, 3 CombObj" achieved an overall score of 0.791476.

The process of introducing combat objects at points in time is intended to generate a persistence of excitation condition sufficient for identifying the relative importance weights. A starting state of combat objects was introduced and the enemy degrees of achievement were observed. More combat objects were then revealed and changes to the degrees of achievement of the altered enemy plans were observed. With constant intent, the enemy would not change plans unless the new plans were better, i.e., the new degrees of achievement resulted in a higher score than the previous degrees of achievement for the new combat object setup. This condition is verified in Table 1. When scored against the new combat object setup, the score for the plan is higher than the score for any previous plan. Had a previous plan resulted in a higher score for the new combat object setup, it would have been readopted instead of a new plan. This means that $score(n, n) > score(n - m, n)$ for all n, and $0 < m < n - 1$. This provides many precedence relationships between plans when determining enemy intent.

Constraints were taken from relationships with $score(n, n) > score(n - m, n)$ to determine the set of weight vectors that would satisfy these constraints. Fig. 8 depicts the evolved population of candidate weight sets plotted on all possible $w_i w_j$ axes, at generation 10. By this point, the population had converged to the appropriate region in the 4-D simplex.

Fig. 8 does not present any clear ordering for the weights in this experiment. However, if each population member is considered a vote for the actual relationship between the weights, then at generation 10, the population voted for $w_2 > w_3 > w_4 > w_1$. The actual weight ordering was $w_3 = w_4 > w_2 > w_1$. The population correctly determined $w_3 > w_1$, $w_4 > w_1$, and $w_2 > w_1$ (thus, that w_1 was the least important parameter of the four). The remaining relationships were incorrectly determined by voting. The following matrix shows the number of population members voting in each category (e.g., 30 means 30 of 100 population elements determined $w_1 > w_2$).

	w_1	w_2	w_3	w_4
w_1	100	30	42	44
w_2	70	100	67	68
w_3	58	33	100	58
w_4	56	32	42	100

It is evident that obtaining enemy plans by observing changes in response to new combat objects does not produce the desired persistence of excitation; more observations are needed.

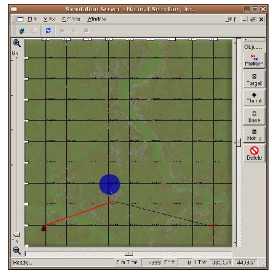


Fig. 7a. Evolved plan for one combat object (target and co-located threat).

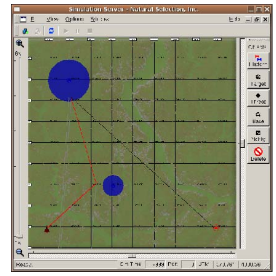


Fig. 7b. Evolved plan when second combat object revealed (target and co-located threat).

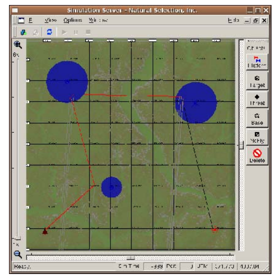


Fig. 7c. Evolved plan when third combat object revealed (target and co-located threat).

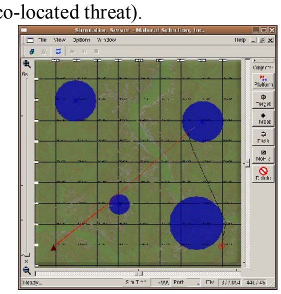


Fig. 7d. Evolved plan when fourth combat object revealed (threat only).

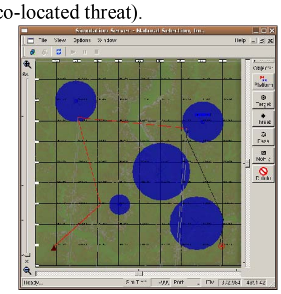


Fig. 7e. Evolved plan when fifth combat object revealed (threat only).

TABLE 1.
DEGREES OF ACHIEVEMENT OF EACH PLAN WITH RESPECT TO THE WEIGHTED PARAMETERS OF CONCERN AT EACH STAGE IN THE SCENARIO OF EXPERIMENT 1.

	Weights	Fuel (5)	Damage (7)	Platform (10)	AtBase (10)	Total Score
		0.156250	0.218750	0.312500	0.312500	
Plan	CombObj					
1 Threat	1 CombObj	0.888296	1.000000	1.000000	1.000000	0.982546
1 Threat	2 CombObj	0.888296	0.333333	1.000000	1.000000	0.836713
2 Threat	2 CombObj	0.776111	0.994713	0.796520	1.000000	0.900273
1 Threat	3 CombObj	0.888296	0.166667	1.000000	1.000000	0.800255
2 Threat	3 CombObj	0.776111	0.497356	0.796520	1.000000	0.791476
3 Threat	3 CombObj	0.740963	0.998977	0.624702	1.000000	0.842021
1 Threat	4 CombObj	0.888296	0.166667	0.096369	1.000000	0.517870
2 Threat	4 CombObj	0.776111	0.497356	0.049736	1.000000	0.558106
3 Threat	4 CombObj	0.740963	0.998977	0.084823	1.000000	0.673309
4 Threat	4 CombObj	0.806926	0.662701	0.796313	1.000000	0.832396
1 Threat	5 CombObj	0.888296	0.166667	0.096369	1.000000	0.517870
2 Threat	5 CombObj	0.776111	0.497356	0.003471	1.000000	0.543649
3 Threat	5 CombObj	0.740963	0.998977	0.084823	1.000000	0.673309
4 Threat	5 CombObj	0.806926	0.232729	0.032373	1.000000	0.499608
5 Threat	5 CombObj	0.748185	0.988184	0.623654	1.000000	0.840461

B. Use All Evolved Plans

Rather than focusing only on the best-evolved plans at each step in the scenarios, attention was given to all plans in the population, which could be rank ordered. When evolving a population of, say, 20 plans at each step, as new combat objects were added, the number of available observed precedence relationships increased considerably as compared to the previously described experiments. Only plans within the same generation were compared.

A complex, campaign-level scenario and set of combat objects were created and tested with several sets of importance weights. Fig. 9a shows the scenario with evolved plans; Table II shows the weights and results.

Degrees of achievement and corresponding scores were extracted from the data file for non-zero parameter weights and a selected generation. Then, the data were sorted by score. "Duplicate" members of the population were eliminated, where "duplicate" means members of the population with identical degrees of achievement (and hence identical scores), but not necessarily the same plan. Note that it is possible to have the same score with different degrees of achievement, so the precedence ordering is assumed to be "better than or the same as" instead of strictly "better than."

For a test case, there were 320 potential distinct population members (plans to compare). Typically after elimination of duplicate population members, this number was reduced to less than 200. Accuracy could be sacrificed to further reduce the number of comparisons and the computational complexity by sampling randomly from those remaining. In ad hoc experimentation, it was determined that anywhere from 25-50 population members were needed to obtain reasonable results. The variation stems from the particular choice of population members and in the choice of random potential weight vectors used in the inverse problem.

Each set of weights considered had between 4 and 7 parameters of importance (non-zero weights), as shown in Table II. The forward simulation was run with each of these sets of weights to obtain the best plan. Degrees of achievement for intermediate plans were collected as well, to allow for adequate information in estimating the parameter weights. Table II shows the values input to the VSS, the normalized values for each parameter, and the resulting estimated values (center of mass of the evolved population of potential weights) for the parameters based on the degrees of achievement of evolved plans. In the table, only the non-zero parameter weights were evolved. The estimated values listed in the tables are those found after 500 generations of evolving the estimated weights for various groups of degrees of achievements considered. The error, measured as Euclidean distance, is also shown, and is small in each case.

Additionally, using the weights labeled "A", the algorithm was tested while including an unimportant parameter, one whose true weighted value was zero. The algorithm performed well, with an error of 0.0018616 after 500 generations. The actual normalized weights were:

ammo: 0.0000 fuel: 0.1563 damage: 0.3125 plat. surv.: 0.2188 base: 0.3125

The estimated evolved weights were:

ammo: 0.016557 fuel: 0.1530 damage: 0.30878 plat. surv.: 0.21496 base: 0.3067

indicating close agreement.

Better performance was observed when estimating the actual weights when using more plans. This suggests that (1) better performance in identifying the purpose is observed with increasing samples and (2) better performance is observed when degrees of achievement for comparatively bad plans are taken into account.

Fig. 9a-d shows the original evolved plans for the set of weights "A" and, for comparison, plans evolved using the estimated weights. Additionally, the error in estimated weight values is plotted as a function of the generation in the inverse problem. The weights were evolved based on plans considered in a specific generation during the forward evolution. Each graph shows the estimated weight evolved with two different seeds to the random function. Typically, the error in estimated weight stabilizes by generation 100.

In the graphs, the error in the estimated weights actually increases before decreasing again. A plausible explanation for this behavior is based on the geometry of the divided simplex. Some region that is "closer" in terms of the number of precedence relationships may actually be an oddly shaped region that contains some points that are further in a Euclidean sense than all points contained in a region that is "further" in terms of number of precedence relationships. For example, in Fig. 10, the desired region to find is labeled 1. All regions that are one precedence relationship away are labeled 2; regions that are two precedence relationships away are labeled 3, etc. The region labeled 4 that is diagonally adjacent to the desired region is actually closer in a Euclidean sense than the region in the lower-left corner, which is labeled 3. However, the region labeled 3 is closer in a precedence relationship sense than the region labeled 4. A progression of population members from the described region 4 to the described region 3 would actually result in an increase in error of the estimated weights; further progression to the regions labeled 2 and then 1 would result in the desired reduction of error again.

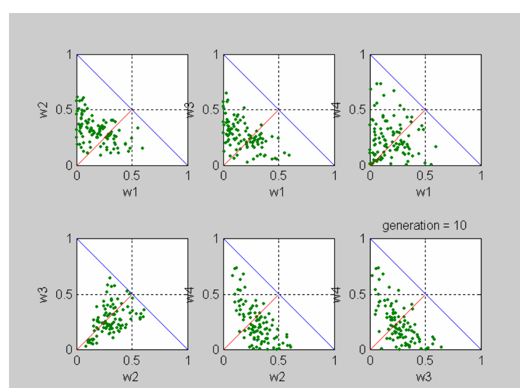


Fig. 8. The evolved weight relationships at generation 10 based on the precedence ordering of the best-evolved plans when encountering five combat objects in experiment 1.

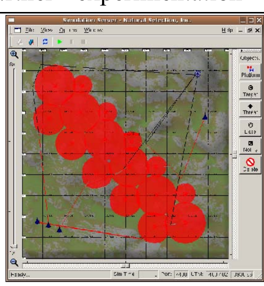
Table II

Importance weights used in the scenario in Fig. 9. Each set of weights lists the actual numbers input to the VSS, the normalized values for each parameter, and the estimated values for the parameters based on the degrees of achievement of evolved plans. Additionally, the error between the estimated and actual weights is listed. The evolved estimate of the weights only took into consideration the non-zero-weighted parameters.

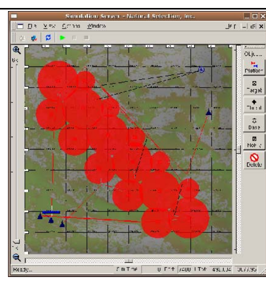
Weight/ Parameter	Ammo	Fuel	Damage Done	Sensing	BDA	Platform Survival	Return to Base	Reconn	Error
A	0	5	10	0	0	7	10	0	
A (normalized)	0	0.15625	0.31250	0	0	0.21875	0.31250	0	
A (estimated)	0	0.15633	0.31257	0	0	0.21874	0.31236	0	0.00018023
B	2	5	20	3	0	7	10	0	
B (normalized)	0.04255	0.10638	0.42553	0.06383	0	0.14894	0.21277	0	
B (estimated)	0.041818	0.1058	0.42189	0.065999	0	0.14709	0.2174		0.0066079
C	5	3	15	2	1	10	7	0	
C (normalized)	0.11628	0.06977	0.34884	0.04651	0.02326	0.23256	0.16279	0	
C (estimated)	0.11923	0.07085	0.34542	0.04718	0.023324	0.23216	0.16184	0	0.00480
D	5	1	0	10	0	15	7	20	
D (normalized)	0.08621	0.01724	0	0.17241	0	0.25862	0.12069	0.34483	
D (estimated)	0.08645	0.01726	0	0.17247	0	0.25876	0.12127	0.34379	0.0012274

IV. Conclusion

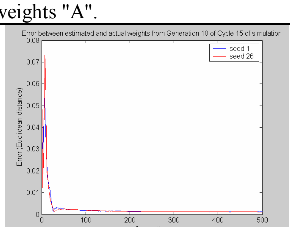
This paper explored the use of evolutionary algorithms to accomplish a form of system identification on the weighted parameters of importance guiding an enemy force. The experiments showed that with sufficient knowledge of the preference of enemy plans, particularly when including the rank ordering of all models available in an evolving population of plans, an evolutionary algorithm can determine the enemy's intent as a normalized VSS with high accuracy and yield prospective eCOAs. This may provide an opportunity to gain valuable insight into anticipating eCOAs in more realistic settings. The experiments also showed that when only a small sample size of observed plans is available, the evolutionary method was not capable of identifying the relative importance of the weighted parameters reliably. Further experimentation is warranted to determine the computational requirements as a trade off to the degree of accuracy of weight parameter estimation.



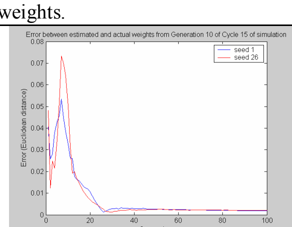
a. Plan generated from original weights "A".



b. Plan generated from estimated weights.



c. Error between estimated and actual weights from generation 10 of cycle 15 of the simulation, shown for 500 generations.



d. Error zoomed to 100 generations.

Fig. 9a-d. Scenario and combat objects with actual and estimated evolved plans; error of estimated weights. A "wall" of co-located targets and threats separates three platforms from returning to base via a direct path; an additional platform starting near the base is also available for planning purposes. The evolutionary algorithm generates good agreement between the estimated and true weights. For more results, see www.natural-selection.com/CISDA07.html.

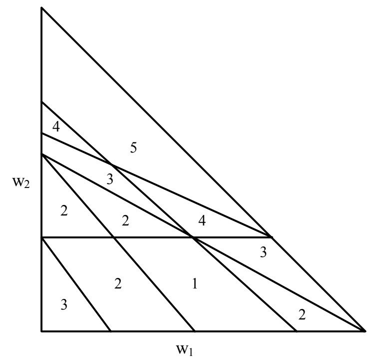


Fig. 10. Region 1 is the desired target region. All regions that are one precedence relationship away are labeled 2; regions that are two precedence relationships away are labeled 3, etc. The region labeled 4 that is diagonally adjacent to the desired region is actually closer in a Euclidean sense than the region in the lower-left corner, which is labeled 3. However, the region labeled 3 is closer in a precedence relationship sense than the region labeled 4. A progression of population members from the described region 4 to the described region 3 would actually result in an increase in error of the estimated weights; further progression to the regions labeled 2 and then 1 would result in the desired reduction of error again.

The research and development focused on scenarios that were in some cases simple and in other cases more complex. The SAM player did not have the freedom to move assets in response to the evolving aircraft plans. Future research could offer both players freedom to adjust and adapt plans on the fly as conditions change. It would be of interest to determine the effectiveness of identifying the weighted parameters of interest of both players, iteratively.

Acknowledgment

Some of this work was funded under Air Force STTR FA9550-05-C-0070.

References

[1] P.E. Caines, *Linear Stochastic Systems*, John Wiley & Sons, NY, 1988.

[2] L.J. Fogel, A.J. Owens, and M.J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, NY, 1966.

[3] Z. Michalewicz and D. Fogel. *How to Solve It: Modern Heuristics*, 2nd ed. Germany: Springer-Verlag Berlin Heidelberg, 2004, pp. 443-449.

[4] L.J. Fogel. "The Valuated State Space Approach and Evolutionary Computation for Problem Solving," in *Computational Intelligence: A Dynamic System Perspective*, M. Palaniswami, Y Attikiouzel, R. Marks II, D. Fogel, T. Fukuda, Eds. Autralia, 1995, pp. 129-136.