FOX-GA: A Genetic Algorithm for Generating and Analyzing Battlefield Courses of Action

J. L. Schlabach
Technology Integration Office
Office of Deputy Chief of Staff for Intelligence
Department of the Army
Pentagon, VA
jlschla@vulcan.belvoir.army.mil

C. C. Hayes
Department of Mechanical and
Industrial Engineering
University of Minnesota
111 Church Street S. E.
Minneapolis, MN 55455-0111
hayes@cs.uiuc.edu

D. E. Goldberg
Department of General Engineering
117 Transportation Bldg.
104 South Mathews Ave.
University of Illinois
Urbana, IL 61801
deg@uiuc.edu

Abstract

This paper describes FOX-GA, a genetic algorithm (GA) that generates and evaluates plans in the complex domain of military maneuver planning. FOX-GA's contributions are to demonstrate an effective application of GA technology to a complex real world planning problem, and to provide an understanding of the properties needed in a GA solution to meet the challenges of decision support in complex domains. Previous obstacles to applying GA technology to maneuver planning include the lack of efficient algorithms for determining the fitness of plans. Detailed simulations would ideally be used to evaluate these plans, but most such simulations typically require several hours to assess a single plan. Since a GA needs to quickly generate and evaluate thousands of plans, these methods are too slow. To solve this problem we developed an efficient evaluator (wargamer) that uses course-grained representations of this problem domain to allow appropriate yet intelligent trade-offs between computational efficiency and accuracy. An additional challenge was that users needed a diverse set of significantly different plan options from which to choose. Typical GA's tend to develop a group of "best" solutions that may be very similar (or identical) to each other. This may not provide users with sufficient choice. We addressed this problem by adding a niching strategy to the selection mechanism to insure diversity in the solution set, providing users with a more satisfactory range of choices. FOX-GA's impact will be in providing decision support to time constrained and cognitively overloaded battlestaff to help them rapidly explore options, create plans, and better cope with the information demands of modern warfare.

Keywords

Knowledge-based niching, military reasoning, course of action planning, computer assisted assessment.

©1999 by the Massachusetts Institute of Technology

1 Introduction

This paper describes FOX-GA, a decision support tool that applies a genetic algorithm (GA) to generate and evaluate plans in the complex domain of military maneuver planning. These plans are called *courses of action* (COAs). The GA uses a niching strategy to ensure that a diverse, yet high quality set of plan options are created. FOX-GA's contributions are to demonstrate an effective application of GA technology to a complex real world planning problem, and to provide an understanding of the properties needed in a GA solution to meet the challenges of decision support in complex domains. The name "Fox" refers to the animal, which is known for being smart and capable of "out-foxing" its enemies through clever planning.

Requirements in the maneuver planning domain include the need to 1) assist users in identifying the "best" of numerous solution options 2) provide diverse yet high quality options from which users can select and 3) rapidly generate and evaluate options under tight time constraints. Many of these these challenges are common to decision support tasks in other domains such as medical decision making, manufacturing planning, mechanical design, and architectural design.

FOX-GA is implemented in C++. The great majority of the implementation effort went into the design and construction of the knowledge representation and the logic in the wargamer. Implementation of the GA itself required only 5 percent of the effort.

We will begin with overviews of the domain of battlefield reasoning, challenges in constructing this application, and related work. We will follow it with a detailed explanation of FOX-GA's genetic algorithm, bit string representation, wargaming algorithm, and fitness function. Lastly we will discuss two experiments evaluating Fox's performance, and possible future work.

2 The Domain of Battlefield Reasoning

The Army organizes its forces into hierarchical echelons as shown in Figure 1. Each unit is composed of a set of smaller *subordinate* units that are under its command. Figure 1 shows a mechanized infantry brigade at the top, with its subordinates underneath. The primary subordinates of the mechanized infantry brigade are the units that maneuver during combat. In this case the primary subordinates are three mechanized infantry battalions (on the far left in Figure 1), and one armored battalion (immediately to their right). The other units provide combat support. When the commander of a unit receives a tactical mission from his higher headquarters he directs his battlestaff to develop a set of possible "Courses of Action" (COAs) that can each accomplish the mission. Typical missions include "attack" and "defend". After analyzing the alternative COAs the commander selects one COA for unit execution.

The terrain on which a unit fights is referred to as the *maneuver box*. Maps provide very detailed representations of the terrain inside a maneuver box in terms of elevations, vegetation and hydrology. However, there are very specific abstractions which battlestaffs draw from the maneuver box terrain in order to plan. Figure 2 shows an example of abstract representations extracted by a battlestaff from the terrain in a maneuver box. This much simplified representation shows many of the essential terrain features used by battlestaffs for maneuver planning, just as the New York City subway map is a much simplified representation of New York showing essential information used by subway customers to do route planning.

The broad arrows represent *avenues of approach* (AAs). AAs 1-3 are wide routes down which the subordinate units will move in order to attack. Typically there may be between two and five AAs in any given maneuver box. The large oval areas to the left are *Tactical Assembly*

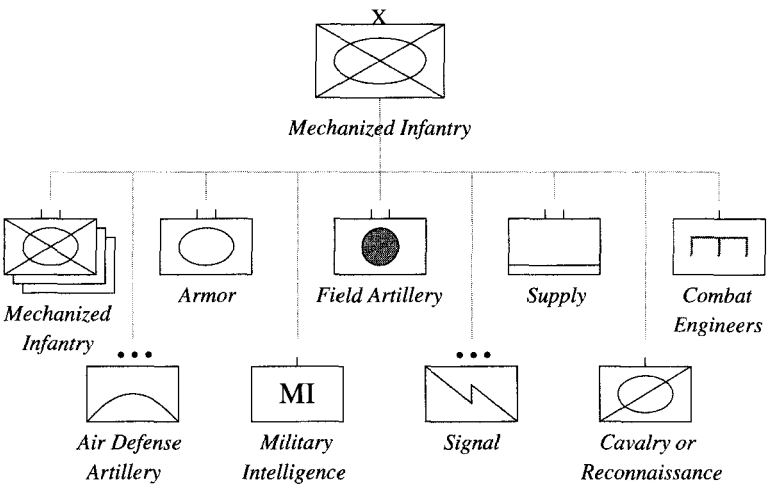

Figure 1: Functional Organization of Army Units at Each Echelon

Areas (TAAs), which provide the starting locations for the friendly units. The ovals to the right provide *objectives* (OBJs) which represent goal locations for the friendly attack: they wish to reach or capture these locations. Additional, non-geographic goals may be to reduce the strength of enemy forces. The *Forward Edge of the Battlefield* (FEBA) is the demarcation line between enemy and friendly forces *prior* to battle, and the *Limit of Advance* (LOA) is the demarcation *after* battle (if the attacker is completely successful).

The gray vertical lines in Figure 2 are lines of defensible terrain (LDTs). LDTs are formed by identifying narrow areas or *choke points* cutting across the AAs. A whole string of roughly adjacent choke points cutting across all AAs forms an LDT. When plotted on a map, the LDTs are not usually straight lines any more than subway routes are when shown on a scale city map. Lines of defensible terrain are usually good points at which to set up a defense. Hence they are also the places at which conflicts between attacker and defender tend to occur.

In its most fundamental form, an offensive COA is the determination of how to assign and sequence subordinate units in the various AAs. Figure 3 shows an example of an offensive COA where a mechanized battalion attacks along AA-2 and the three other battalions attack in column along AA-3. There are many possible ways of organizing a set of subordinate units into COAs in a given maneuver box. COAs are also distinguished by the mission, number, and type of subordinates, and share of general resources assigned to each unit during battle. FOX-GA also considers more detailed distinctions between COAs as will be described later in this paper.

The four LDTs facilitate defensive operations by taking advantage of the chokepoints that produce bottleneck problems for the attacker. Figure 4 shows an example of such a defensive COA where the enemy battalion defends with three companies abreast on LDT-1. The reserve force located at the intersection of LDT-2 and AA-2 is available to conduct a counterattack anywhere within the maneuver box.

The user's need for a diverse set of choices arises partly because mathematical fitness functions in complex domains are actually only approximations of the "true" function. The "true" (and probably unachievable) fitness function is one that correctly identifies the best option for the situation. An example of a plan, which we call the "sacrificial lamb" strategy

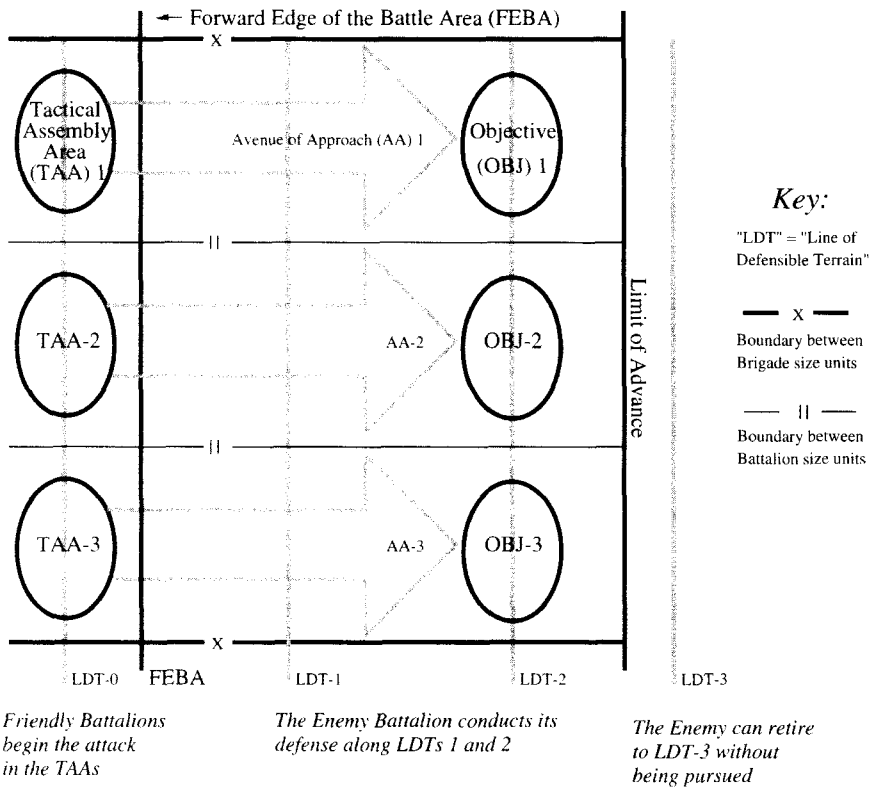

Figure 2: The Terrain Maneuver Box for the current implementation of FOX-GA

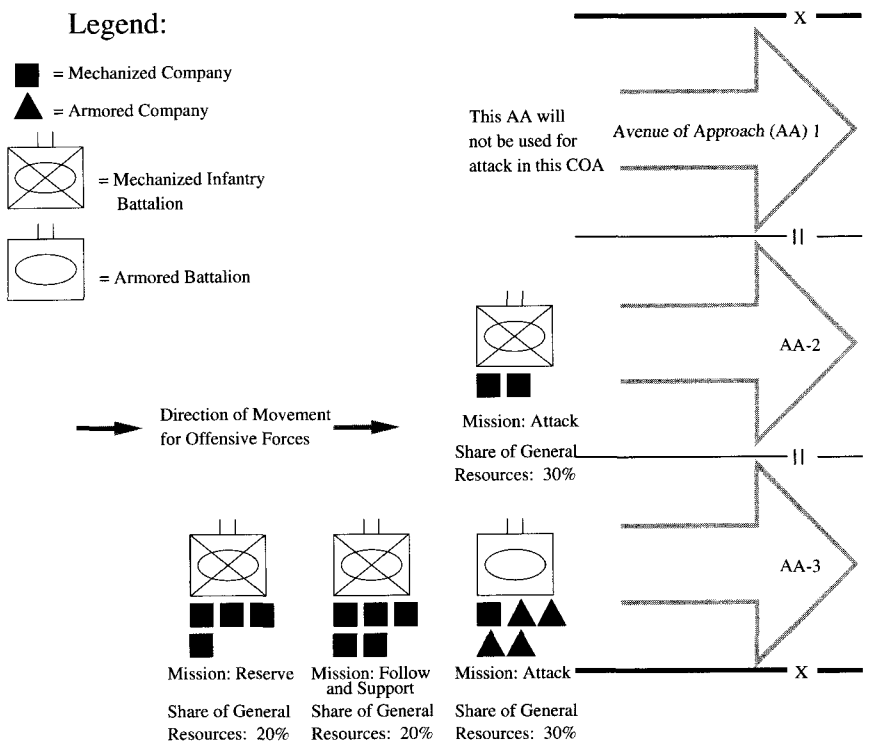

Figure 3: An Example of a Friendly Offensive Course of Action, judged by FOX-GA's fitness function to be of high quality (sacrificial lamb strategy)

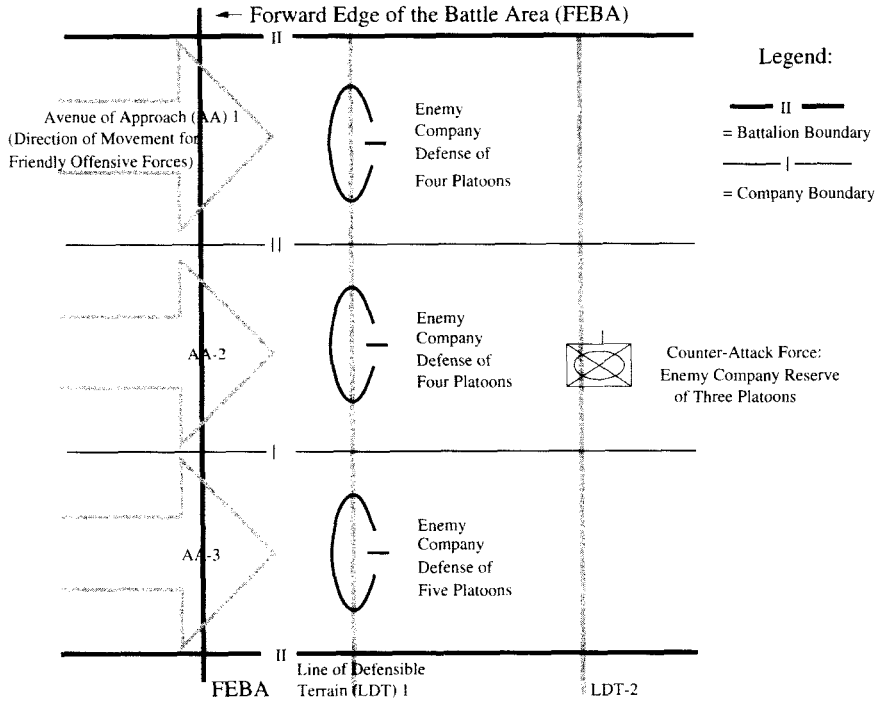

Figure 4: An Example of a representative Enemy Defensive Course of Action

3, will be described in later sections. This plan appears to be of very high quality to the GA's fitness function. However, it is not in fact an option most commanders would wish to chose without exploring alternative COAs that do not require the whole sacrifice of a subordinate.

It is not the intention of this discussion to imply that humans evaluate options using the "true" fitness function. However, humans are very good at taking into account complex, highly situation dependent factors that may not be easy to capture in a mathematical function. The combination of the GA's fitness function with human judgment can create a powerful evaluation method that is more effective than either alone.

Thus, because of the approximate nature of all fitness functions in complex domains, the role of a GA in complex decision support should not just be to select the "best" solution for the user but to identify a diverse set of "good" candidate solutions while allowing the user to make their own assessments for the final decision.

3 Challenges and Design Decisions in Fox-GA

There were a number of challenges we had to address in adapting GA technology to fit users' decision support needs in the context of the domain including the need to:

1. Provide fast fitness evaluations for COAs, so as to provide users with candidate COAs in a timely manner.
2. Avoid overwhelming users with possible options.
3. Provide a diverse set of COAs, that also fit the users' domain-based concepts of diversity.

4. Allow users to exercise their own judgment, which may disagree with the systems' assessments in selecting the "best" options.
5. Provide users with the flexibility to adapt the behavior of the GA according to their varying needs for expediency, breadth of suggested options, etc.

Providing fast fitness evaluations for COAs. A major obstacle we encountered in applying GA technology to generating and evaluating COAs was the lack of efficient algorithms for evaluating COAs. Battlefield simulations can be used for evaluating and assessing the behavior and performance of COAs under battle conditions. However, most such simulators have been designed for detailed training purposes and require several hours to simulate a single COA in a single battle. They cannot be used to provide speedy evaluations having "ball-park" accuracy (which is all the accuracy needed for identifying good candidate solutions for further study). Existing battle simulators were not practical for our purposes since we needed to evaluate thousands of COAs, each simulated in several battle situations, in order to identify the best candidates. All of these COAs need to be evaluated in a timely manner since user's have limited time and patience to wait for results. If the decision support too tool is too slow, they will simply abandon the computer tool and resort to their standard manual methods.

To address this need we developed an efficient wargamer that uses very course-grained representations of the problem domain. Our wargamer represents the battle at a very course grained level of abstraction and provides a less accurate assessment of COA performance than a detailed simulator would provide. However, it simulates a single battle in a fraction of a second, which is what is needed for the purpose of rapidly identifying some "good guesses" of the best COA options. These options will be assessed a second time by the human users, who will then select a few of these initial options for further development and detailed assessment. Detailed assessment at this early phase of problem solving would, in fact, be counter-productive. The need to make trade-offs between efficiency and detail through use of estimates is a common one found not only in GAs but in engineering design in general (Ling et al., 1993), (Miller, 1996).

Avoiding overwhelming users with possible options. Human users typically explore between 1 and 5 COA options. FOX-GA explores thousands. However, it would probably be counterproductive to decision making performance if all of these options were presented at once to the user. He or she would be overwhelmed by the sheer volume of information. In order to avoid information overload, we use the performance assessments produced by the GA to rank order the options, and only present the best few to the user. We give the user the control to decide how many options they want to see.

Providing diverse COA options. Users in this domain prefer to be presented with a diverse set of significantly different plan options from which they can choose, each offering different trade-offs. Diverse options offering various trade-offs are far more useful to them than small variants on essentially similar solutions.

Standard GAs tend to develop a group of "best" solutions that are all very similar to each other. We addressed this challenge by adding a niching strategy to the selection mechanism to insure diversity in the solution set, providing users with a more satisfactory range of choices. However, the specific niching strategy used had to provide diverse options that fit the users' domain-based conceptions of what makes one option distinct from another: there are particular characteristics which their experience has taught them to consider to be most important in making one choice different from another. Thus we faced the additional challenge of incorporating domain based conceptions of "different" into our niching strategy. This strategy will be described in more detail in a later section.

Allowing users to use their own judgment in selecting the "best" options. Potentially, FOX-GA could choose a small set of "best" COA options without input from the user based on the performance assessments it produces. However, users will neither trust nor accept a decision support tool that does not allow them to disagree with the system's assessments, or to have the the final say in what solutions are chosen. Thus, we allow the user to over-ride FOX-GA's performance rankings of the COAs and to select what ever COAs they feel are best.

This design decision is a recognition and accommodation of the fact that a) a GA's fitness function can at best be only an approximation of the "true" fitness assessments, and b) different individual users may favor different assessments of COAs based on differences in goals, preferences and experience.

Providing users with flexibility to adapt the behavior of the GA. The flexible nature of GAs makes it easy for users to tailor the behavior of FOX-GA to suit their specific needs. For example, if they need a faster response, they can decrease the population size or the number of generations used by the GA. If they want more diversity in the options they can increase the size of the initial population.

4 Related Work

The work most closely related to FOX includes the AirLand Battle Management (ALBM) Project (ALBM,) (developed at Lockheed Missiles and Space Company), and the Systems for Operations Crisis Action Planning (SOCAP) (Wilkins and Desimone, 1994) developed at Stanford Research Institute (SRI). Both systems focused on the automated generation of COAs. ALBM specialized in providing effective representations for human-computer interactions, while SOCAP specialized in efficient generation of COAs. Neither of these was intended to provide an assessment of how well the COAs would perform in combat against anticipated enemy COAs.

A barrier to the use of ALBM's methods in a GA, which needs to evaluate many battles per minute, was ALBM's use of detailed terrain representations. This is a problem for rapid COA evaluation because it takes too much time to evaluate at a detailed level. The ALBM team recognized a need to reduce through abstraction the amount of terrain information considered by ALBM. One contribution of the ALBM project was the development of basic terrain representations that would enable a computer to efficiently reason about units maneuvering on a battlefield (Reich, 1995). Unfortunately the representations were not comprehensive enough for our purposes.

The goals of the Battlefield Reasoning System (BRS) Architecture (Schlabach, 1997) were to develop a comprehensive set of abstractions and to establish a reasoning framework to support effective and efficient battlefield reasoning for a critical set of tasks, one of which was COA generation. The architecture provides appropriate abstract representations for concepts such as terrain, order of battle, decision points, and military intelligence propositions. It expands ALBM's terrain representations into a systematic and comprehensive set of data structures suitable for exploitation by battlefield reasoning applications. The BRS Architecture also identifies a set of reasoning processes that comprise a primary "backbone" for battlefield reasoning coupling upstream processes like COA generation with downstream processes like intelligence analysis. FOX-GA is designed to operate in the context of the BRS Architecture, as one of a suite of coordinated intelligent battlefield reasoning agents.

Two GA applications in other military domains investigated decision making in the simulation of small unit actions (Porto and Fogel, 1997) and control of Naval Surface-to-Air

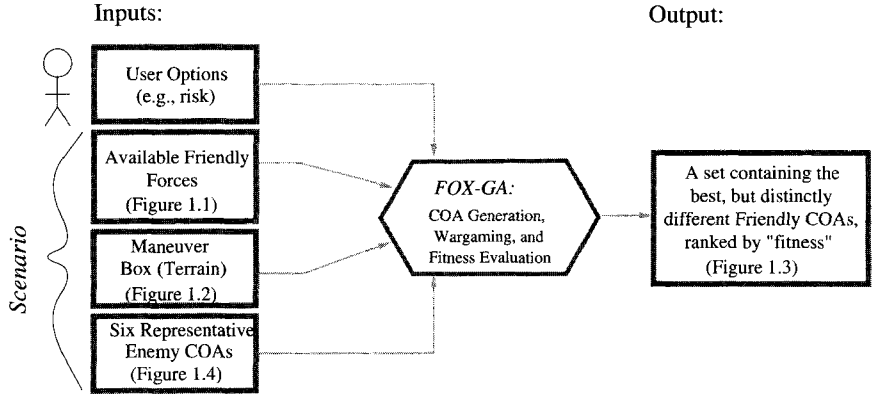

Figure 5: FOX generates and analyzes Offensive COAs

weapons (Kuchinski, 1985). However, the domains of these applications provide only limited insights into the use of GAs for COA generation for battlestaffs.

5 Approach

FOX-GA uses a fixed scenario with the mechanized infantry brigade shown in Figure 1 attacking against an enemy mechanized infantry battalion. By using the knowledge representations (particularly of terrain – Figure 2) defined in the BRS Architecture, FOX-GA generates and evaluates COAs using causal reasoning to efficiently simulate friendly COAs against the enemy. This efficiency is gained by abstracting away detail typically used in training simulations, but retaining most of the information pertinent for COA generation. The accuracy of the wargaming evaluation is slightly degraded because FOX does not use all of the detailed terrain information available in modern digitized maps. Nor does FOX allow human guidance in the wargaming algorithm which could also increase accuracy. However, the efficiency gain is more important than the small accuracy loss. The resulting increase in speed enables FOX-GA to evaluate significantly more COAs per minute than is possible using more detailed representations. [1]

Inputs: As shown in Figure 5 FOX-GA takes two types of inputs: 1) a set of user specified parameters, and 2) a fixed set of parameters which are specified inside the program which define a battlefield scenario. The scenario specification is made up of the terrain maneuver box (Figure 2), six scripted representative enemy COAs (one of which is shown in Figure 4), and a description of available friendly forces (Figure 1). Although the current proof of concept implementation considers only one scenario, the technique can be extended in future work without loss of efficiency or effectiveness, to consider any combination of terrain, enemy COAs, and available friendly forces. The six enemy COAs are generic representations of typical enemy options. The user inputs include: allowable risk, status of friendly forces, status of enemy forces, fitness evaluation mode (terrain or enemy options to be explained later), population size for the GA, and number of generations for the GA.

Outputs: FOX-GA outputs a set of n best COAs (as measured by the fitness function which will be described later). The size of n is set by the user.

[1] FOX-GA can fight 300 battles per second whereas the typical simulator built for training requires at least several hours to fight one battle.

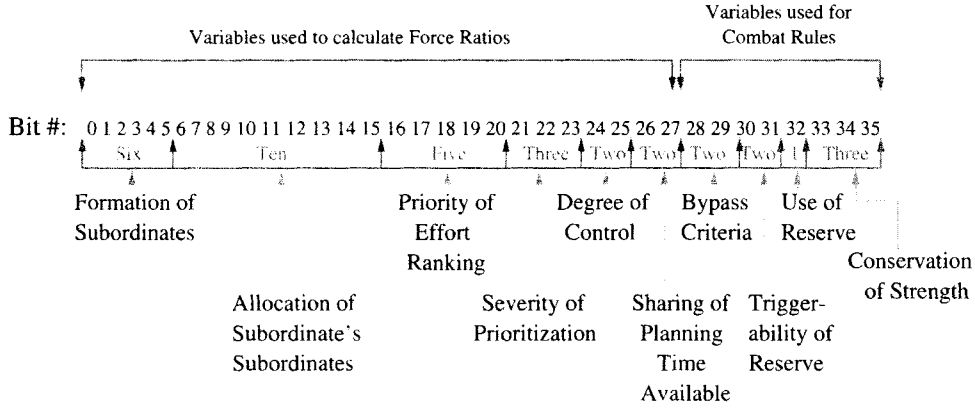

Figure 6: FOX-GA's Bit String Representation for tactical Courses of Action (COAs)

6 System Description

FOX-GA was implemented in C++ on a SUN Ultra 1 workstation. The major components of the FOX-GA system include the genetic algorithm which uses a niching strategy, the bit string used by the GA to represent knowledge about COAs, the combat wargamer used to estimate losses incurred in battle using each COA in a variety of situations, and a fitness function used to assess the quality of the COA's combat performance as assessed by the wargamer.

6.1 Bit String Representation

FOX-GA uses the bit string representation illustrated in Figure 6. The bit string is composed of ten variables representing the COA properties listed below, which are of importance to battlestaff planners in representing COAs for wargaming. These variables are used in FOX-GA by both the GA (to create COAs) and by the wargamer (to evaluate their performance). There are two halves to the wargamer, The first half calculates *force ratios* which are the relative strengths of opposing units, and the second half applies *combat rules* that determine the next move of all units. We mention this because the ten variables can be further subdivided as shown in Figure 6 into those used in computing force ratios (the first six variables) and those used by the combat rules (the last four variables).

- **Formation of Subordinates** refers to the way in which the mechanized infantry brigade allocates and sequences its four battalions (subordinates) used in FOX's problem scenario to the three Avenues of Approach. For the chosen terrain and forces there are 60 possible formations. The Formation of Subordinates variable is the most important one on the list for organizing COAs. Each of the 60 possible formations is generally considered by battlestaff to represent distinctly different COA families (also referred to as COA neighborhoods) while variations in the other variables produce what battlestaff consider to be minor variants within the same COA family. Because of its importance, this variable has been placed at the front of the bit string. This variable is also important in the niching strategy which will be explained later in the section on the genetic algorithm.

  When we designed the implementation we placed the 60 formations contiguously in an indexed look-up table such that the majority of neighbors differ only by a "swap in place" of the armor battalion with one of the three mechanized battalions. This is important

for evolutionary computing in order to insure that a small mutation will result in only a slight difference in fitness.

- **Allocation of Subordinate's Subordinates** refers to the number of mechanized infantry and armor companies (subordinates of the subordinates) that will be attached to each of the four available battalions. There are a total of 12 mechanized and 4 armored companies available for allocation to the subordinate battalions. Each subordinate battalion will receive between two and five (inclusive) companies, for a total of 1013 unique permutations of the 16 companies to the four battalions. The indexing scheme for this set of bits also retains relations between neighbors.

- **Priority of Effort Ranking** Units have limited resources, and they cannot always give their subordinates all the resources they desire. Priority of effort ranking indicates the relative importance of a subordinate role with respect to the overall mission. For example, if a subordinate has first priority it will receive the lion's share of artillery, intelligence, logistics, and other support. There are 24 possible prioritization schemes for the four subordinates. The indexing scheme for this set of bits retains relations between neighbors.

- **Severity of prioritization** refers to the percentage of resources that will be allocated to the number one priority at the expense of lower priority subordinates. This is a hand-scripted set of eight distributions where the first distribution is "flat" (first priority receives only slightly more support than the last priority), and the eighth distribution is "steep" (first priority receives all extra resources).

- **Degree of Control** refers to the amount of centralization (as opposed to autonomy) that the brigade gives to the four subordinate battalions. This is an unsigned binary coded integer of four discrete values.

- **Sharing of Planning time Available** is the percentage of planning time allocated to the brigade staff to produce its Operation Order (OPORD). Since Battalion staffs can't begin their planning until they receive Brigade's Order, this variable determines how much time the Battalion Staffs will have to produce their OPORDs. The more time a battlestaff has to plan, the better they will be able to synchronize their forces during battle, which translates directly into a combat multiplying effect. Hence, there is a natural competition between subordinate and superior staffs for planning time. FOX-GA allows four different planning time allocation schemes for planning. This is an unsigned binary coded integer of four discrete values.

- **Bypass Criteria** is the size of the largest allowable enemy force that an attacking battalion may bypass when moving to the next piece of terrain. Bypassing a large unit is considered risky (with a possible high payoff) whereas bypassing a small unit is conservative (with a guaranteed lower payoff). The risk is derived from the fact that bypassed enemy units have excellent opportunities to counterattack against the brigade's softer rear units. However, bypassed enemy units typically surrender or attempt to withdraw back to enemy lines. This is an unsigned binary coded integer of four discrete values.

- **Triggerability of Reserve** refers to how bad a crisis or how good an opportunity has to be before the brigade commits the reserve. This is an unsigned binary coded integer of four discrete values.

- **Use of Reserve** refers to whether the Brigade uses a "Crisis" or "Opportunity" philosophy. In the crisis mode the reserve helps struggling friendly forces no matter how

vulnerable a local enemy situation might be. In the opportunity mode the reserve attacks when it sees an opportunity to take advantage of enemy vulnerabilities, regardless of the friendly situation. This is a one bit toggle switch where "0" decodes to a crisis philosophy and "1" decodes to an "opportunity" philosophy.

- **Conservation of Strength** specifies the threshold of losses a unit is willing to endure and still continue its mission. This is an unsigned binary coded integer of eight discrete values.

This representation allows for approximately 45 billion valid friendly Courses of Action after subtracting the 20 billion illegal string combinations. There is a test in the genetic algorithm to check for the validity of the bit strings.

6.2 Genetic Algorithm

FOX-GA employs a genetic algorithm for search, and a niching strategy to maintain population diversity. The GA implements single point random crossover with mutation using elitist tournament style selection without replacement (described earlier in the overview of genetic algorithms). The mutation scheme provides for one bit chosen at random immediately after crossover. This is different than the traditional "bitwise" mutation schemes.

The initial population of COAs is generated randomly. To produce a new population, FOX-GA mates each member of the old population with another COA string at random. Since mates are not replaced after they are chosen for mating, each COA mates exactly once per generation (until it dies through non-selection). The crossover point between the two parents is chosen at random to produce two children, which are then subjected to a small mutation rate (2.5 percent) which might flip one of their bits. Finally, the children are checked to ensure that they represent valid friendly COAs. An example of an invalid bitstring is one which has a "64" in the first six bits. The index of "64" is illegal because it is outside the range of the 60 legal values for this variable. If COAs fail this validity check they are mutated until they become legal (which is not difficult since there are twice as many valid as invalid COAs).

The fitness of the two children is based upon their performance in wargaming. The combat wargamer runs them through this process and generates parameters to describe their performance. These parameters are used to determine fitness. Of the four members within the nuclear family (two parents and two children), two COAs are selected for survival to the next generation. This elitist tournament selection causes the population to temporarily double in size every generation, but only the more fit half will survive to the next generation.

Each time a new generation of COAs is created, its quality is estimated by 1) running each COA through a combat wargamer that uses *force ratios*[2] to assess the probable lossesincurred on each side in a variety of situations, and 2) running a fitness function (set by the users preferences) to tabulate remaining friendly strength, enemy strength, and gains or losses in terrain holdings. A fitness score is computed for each COA depending on how important each of these factors are to the user.

[2] Force ratios compare the relative strengths of the attacking and defending forces. They are used to predict how much loss will be incurred on each side when a friendly and an enemy unit engage in battle. They are empirically derived from historical data and they are based situational factors such as the size and composition of each side, the terrain conditions, etc.

6.3 Niching Strategy

The selection made by the elitist tournament strategy is constrained by a niching strategy which ensures FOX-GA will return a wide variety of diverse COAs. Diversity is obtained in a niching strategy by favoring distinctly different COAs occurring in different neighborhoods, rather than multiple copies of the best COA found.

Most niching strategies find niches automatically. However, since it was of critical importance in this application that users be able to understand the outputs of the GA, we were concerned that the GA's niches might not make sense to the domain experts. For example, two COAs that are judged by the GA to be in different niches, may appear to the user to be virtually identical variations of the same COA since he or she may judge "difference" by another set of criteria. The GA judges difference between two COAs by the total number of similar and dissimilar parameters. Additionally, human experts appear to apply different weights to different parameters; they are not all considered to be of uniform importance. In this application, the first bit string parameter "formation of subordinates" is considered to be of far more importance than the other parameters and thus has the most weight in determining whether a pair of COAs is considered to be distinctly different from each other. Thus, if two COAs differ in their value only for this single parameter, they are considered to be different COAs even if the rest of the parameters are identical.

In domain terms, this parameter is of such great importance to the user because each arrangement of units represents a different distribution of forces and resources across the terrain on which the battle will take place. One of the major issues in this task is to guess where the enemy will put most of its forces, and to counter appropriately with friendly forces. All other parameters that determine issues such as which unit has first access to the gasoline or ammunition are minor compared to the basic positioning of forces against the enemy.

In order to match the domain experts' concept of difference between solutions it was necessary to implement a variation of existing niching strategies that used this domain-based concept of neighborhoods instead of letting the GA find its own neighborhoods. In particular, we implemented a variation of niching which we call "fixed neighborhood niching" in which we impose a set of domain-relevant neighborhoods on the niching strategy, a priori.

This strategy does not allow the two surviving COA bitstrings from any given nuclear family to be in the same "neighborhood." FOX-GA defines 60 neighborhoods which represent families of closely related COAs. If two COAs allocate their major subordinates in the same manner, then those two COAs are considered to be in the same neighborhood (or family). Since there are 60 ways to allocate major subordinates, there are 60 neighborhoods. Only the better of two in the same neighborhood will survive. This niching strategy ensures that the GA produces a population of high quality COAs that are diverse from a domain experts' perspective.

6.4 Combat Wargamer

The majority of the implementation effort (approximately 95 percent) in FOX-GA went into designing and building the wargamer. The wargamer can be viewed as a course grained, agent-based simulator (somewhat similar in spirit to Laird's TacAir-Soar (Jones et al., 1994)). We hesitate to use the term "simulator" because in this domain it usually implies a much more detailed replication of battlefield events. However, we feel that much of the information available in such detailed simulations (built for training purposes) is not pertinent to COA generation and analysis.

FOX-GA's combat wargamer models the overall battle as a collection of minor engagements between subordinate forces. Each friendly and enemy subordinate unit acts as an agent. The agents follow a set of *combat rules* to determine appropriate maneuvers and changes in mission during battle. The intersections between the *lines of defensible terrain* (LDTs) and *avenues of approach* (AAs) form the vertices of a grid on which the agents can move. These intersections are significant because they represent the choke points in the AAs where enemy forces might set up defenses. These intersections form a superset of eligible locations where fights might occur. At each time step, each agent can either stay in place, or move forward, backward, or laterally by one vertex. Additionally, each agent may change its mission status at each time step. For example, an agent representing a reserve may change its status during the course of battle from "reserve" to "attack" to "disengaged." The agents change position and mission status in accordance with guidance encoded in each COA bitstring (such as conservation of strength). This helps ensure that the agents "cooperate" and act in unison.

Each friendly COA is made to fight 6 battles against a variety of enemy COAs in order to determine how each will perform under a variety of enemy situations. This is important because the commander typically does not know what the enemy will do. It is therefore important to assess each COA's overall robustness under varying circumstances. The enemy COAs are fixed (not computed by the FOX-GA) and are representative of the superset of COAs available to the enemy commander.

Initially, all friendly subordinate units start out in the tactical assembly areas (TAAs) (see Figure 2). Depending upon the enemy COA, the enemy forces start defensive operations on a mixture of positions on LDTs 1 and 2. Given these starting conditions the combat wargamer iterates through the following sequence of actions until there is no more fighting in any AA (**termination conditions**), or 20 time steps have passed (each iteration represents one time step):

- **Examine intersections** of LDTs and AAs. Identify all intersections having both friendly and enemy units on them. These are the locations at which fire fights will occur. Depending upon the interaction of the combat rules each intersection may have zero, one, or more units.
- **Compute force ratios** to assess losses for each side.
- **Apply combat rules** to allow all agents (units) to decide (in accordance with the explicit guidance of the governing COA) both their next move and whether to change their mission status.

Force ratios. Force ratios are a traditional method of computing the losses inflicted on each side during battle. This system of modeling combat has been empirically derived from historical studies of combat. The Army invests considerable resources to maintain accurate values for various types of units in order to insure that force ratio calculations provide reasonable estimations of battle results. FOX-GA uses generic representations of this system, but can be "tuned" using the more comprehensive and accurate Army models.

The first step in computing force ratios is to estimate the "base strength" of each unit. This strength is estimated by examining the size and composition (e.g., armor or mechanized infantry) of each subordinate unit. This base strength is then multiplied with *combat multipliers* that impact the effectiveness of each unit. The factors that contribute combat multipliers to a unit include the availability of general resources (such as field artillery and military intelligence) and how well the available terrain supports offensive and defensive operations.

Once the strengths of the friendly and enemy units have been computed, they are compared by examining their ratio (attacker/defender). This is the force ratio. A ratio of 3:1 (attacker:defender) will result in equal percentages of loss for both sides.[3] A larger ratio favors the attacker and a smaller ratio favors the defender.

Example of a combat rule. The combat rules represent the most important and most knowledge intensive portion of FOX-GA. Combat rules tend to be complex and situation dependent. They use the parameters set in FOX-GA's bit string representation such as "Bypass criteria" and "Conservation of Strength" in order to automatically replicate the commander's decision making. The COA parameters guide the agents' application of the combat rules so that they can cooperate and act in unison. Following is a portion of an example combat rule:

Situation: A friendly and an enemy unit are at the same location (choke point). In engaging the enemy the friendly forces have acquired a "penetration" by destroying a platoon of the enemy's defense.

Conditional Responses:

- If the friendly unit has a sister unit in a "Follow and Support" status, then the original friendly force will exploit the penetration by bypassing the enemy defense, leaving the Follow and Support unit to mop up the remaining enemy.

  action: Move friendly unit forward one vertex to the next LDT.

- If the friendly unit does not have a sister unit in a "Follow and Support" status, the friendly unit checks to see if the remaining enemy forces at the defense are in greater strength than the bypass criteria associated with the overall friendly COA. If the enemy is larger than the bypass criteria the friendly force will stay in place until the enemy is reduced in strength to below the bypass threshold.

  action: No change in position or mission status.

- The friendly unit will also check to see if its *own* strength remains above the conservation of strength threshold associated with the governing COA. If the friendly unit's strength has dipped below that threshold, then

  action: Change mission status to "disengaged.

- If a unit's mission has changed to "disengaged" and it is still vulnerable to fire from the enemy, then it will check the status of reserve forces. If all reserve forces have been committed, the disengaged unit will withdraw to safer terrain.

  action: Move friendly unit back one vertex to previous LDT.

- Otherwise the disengaged force will stay in place (in case the reserve force, upon commitment, needs to use the terrain that would otherwise be ceded to the enemy).

  action: No change in position or mission status.

Termination conditions: fighting stops in an AA when either: the enemy disengages, the friendly units disengage, or one side is completely eliminated. However, the war gaming

[3]The attacking force needs to be about 3 times as large as the defending force in order to offset the advantages that the lines of defensible terrain confer upon the defender.

does not stop until the fighting has stopped in *every* AA. This is because active (engaged) forces can move laterally between the AAs. Premature termination is prevented because every unit starts battle in a mission status other than "disengaged", and can only consider disengagement after incurring combat losses.

6.5 The Fitness Function:

Once the wargamer has finished all six battles for a given COA, its fitness can be computed. Fitness is a function of the remaining strengths of the friendly units, the remaining strength of the enemy units, and the amount of terrain gained or lost during the battle. However, depending upon the situation commanders place differing importance on each of these factors. FOX-GA allows the commander to use two different fitness functions, *enemy oriented* or *terrain oriented.* This preference mode is specified as a user input, and remains a constant throughout the duration of FOX-GA's evaluations.

Enemy oriented. The enemy oriented fitness function measures success of the battle in terms of the amount by which the enemy forces have been reduced relative to the friendly forces. The fitness score is acquired by summing all of the (multiplied) combat strengths at the end of combat, and then subtracting the sum of all enemy strengths.

Terrain oriented. The terrain oriented fitness function places importance on both the amount by which the enemy has been weakened and the amount of terrain captured or lost. The fitness function for this mode of evaluation modifies the above equation by multiplying the combat strength of all forces (both enemy and friendly) by the "depth" of their position. Depth of position refers to the distance (number of LDTs) that a unit has moved forward from its original starting position during the battle.

The combat wargamer and fitness function used jointly can realistically assess domain fitness to various COAs. For example, the COA illustrated in Figure 3 is properly identified by FOX as being of high quality. The COA illustrated in Figure 7 is properly identified by FOX as being of very low quality. In domain terms, because the middle AA is not used in the attack, friendly forces have no "mutual support" between the battalion attacking along AA-1 and the remainder of the brigade in AA-3. Additionally, the prioritization scheme has unintelligently assigned all available general resources to support the "Follow and Support" (middle) battalion in AA-3, which severely cripples the lead battalions. Finally, this COA has allocated two armored companies and *no* infantry companies to an infantry battalion, while simultaneously assigning five infantry companies and *no* armored companies to an armored battalion. This does not take advantage of th! e habitual relationships and ingrained expertise that would otherwise be available. The armored battalion is the right most battalion in AA-3, whereas the other three battalions are all mechanized infantry.

7 Performance of FOX-GA

The performance of FOX-GA was assessed using two experiments. In the first, FOX-GA's ability to produce high-quality (i.e.fit) solutions was compared to two other search techniques, monte carlo (MC), and monte carlo combined with hill climbing (MCHC), in order to compare the average solution fitness achieved by each. The results of the first experiment are reported in Figure 8.

In the second experiment, the two best performing algorithms identified by the first experiment were examined in greater detail in order to compare how method choice and population affected the 1) solution diversity, 2) solution quality within each niche. The results

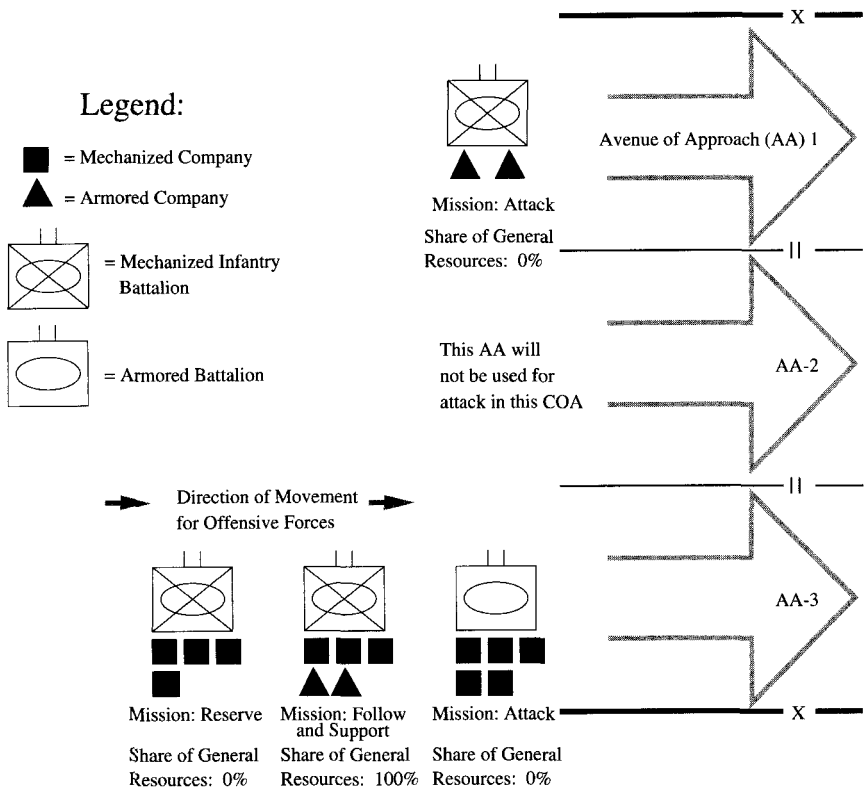

Figure 7: An Example of a Friendly COA judged by FOX-GA's fitness function to be of poor quality.

for this experiment are summarized in Figures 9, 10 and 11.

7.1 Experiment 1

FOX-GA was tested on a Sun Ultra 1 workstation with 192 Megabytes of RAM. 50 COAs (300 battles) per second, which is equivalent to 3000 COAs per minute. The speed of execution for this prototype produced its best set of solutions in less than 30 minutes. To provide a comparison, two additional search engines, FOX-MC and FOX-MCHC, were built that use FOX-GA's combat wargamer and fitness function, but use non-GA techniques and search strategies. The results and comparisons between the three search engines (GA, MC, and MCHC) are shown in Figure 8.

FOX-MC searches through possible COAs using a "Monte Carlo" scheme. In this scheme, one random solution is generated each iteration. At each interaction it saves a solution if 1) it is one of the 10 best solutions found thus far, and 2) it is the best in its neighborhood. Thus, there is niching, but no hill climbing involved in the search for a solution; each new solution is produced independently of the previous ones. The goal of this test is to see how much the various aggregate properties of the GA outside of niching, such as hill climbing, help or hinder the search as compared to a search by random statistical sampling. The number of random COAs generated and evaluated is equal to the total number of COAs evaluated by the other two methods (when multiplying population size times the number of generations).

FOX-MCHC uses a "Monte Carlo with Hill Climbing" scheme. The MCHC engine generates an initial population of 1000 solutions at random, and then conducts simple hill climbing on each member of the population. The simple hill climbing for each member of a population occurs by mutating one bit at random. This is different and possibly weaker than the traditional "bitwise" mutation operator. The mutated COA replaces the old COA if and only if it demonstrates an improvement in wargaming fitness.

Comparisons. Figure 8 shows a comparison of all three search strategies: FOX-GA, FOX-MC, and FOX-MCHC. The comparative performance between FOX-GA and FOX-MCHC is *not* meant to suggest the relative merit of GA and non-GA evolutionary computing techniques. There are advantages and disadvantages to both approaches. The vertical axis shows the average fitness of the ten best strings for each generation, and the horizontal axis shows the number of generations elapsed (or equivalent number of random tests for the Monte Carlo engine). For this comparison, all three engines used the following user specified wargaming input parameters of:

- Risk = 0,
- Friendly Status = 100 percent,
- Enemy Status = 100 percent,
- Fitness Evaluation Mode = terrain-oriented,
- Population Size = 1000 (for both FOX-GA and FOX-MCHC, population size for FOX-MC was 1,000,000),
- Number of generations = 1000, and
- Number of COAs returned = 10.

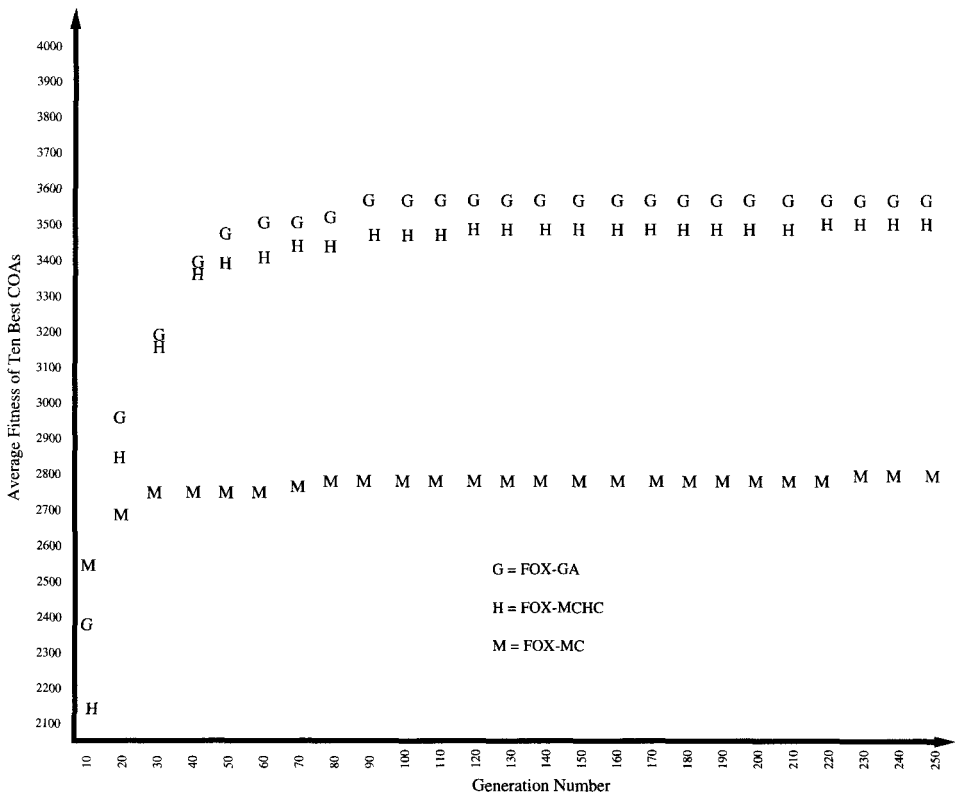

Figure 8: Comparison of FOX-GA, FOX-MC, and FOX-MCHC

"Risk" was set to zero so that the wargamer would assess each friendly COA in terms of its performance against *all* six enemy COAs. If the risk option were set higher, the wargamer would not have included the performance of the more damaging enemy COAs in the fitness score.

Results. We found that both FOX-GA and FOX-MCHC substantially outperformed the FOX-MC; the ten best COAs returned by FOX-MC were far lower in quality than those returned by the other two. FOX-GA found slightly better quality COAs than FOX-MCHC, and in faster time. FOX-GA tends to produce good COAs after as few as 50 generations, with best performance occurring after 200-300 generations.

7.2 Experiment 2

In order to get a closer, more qualitative look at the differences in performance between FOX-GA and FOX-MCHC we created 2 histograms showing the fitness of COAs found in each of the 60 COA neighborhoods. The first histogram (Figure 9) shows the fitness of COAs found in each neighborhood for a population of 200 after 200 generations. The horizontal axis represents the index for the 60 formations (neighborhood niches), while the vertical axis represents the average fitness for all the strings within each niche at the end of the last generation. Figure 10 shows what happens when the population is increased to *600*.

What we found was that FOX-GA's niching strategy allowed it to concentrate on the

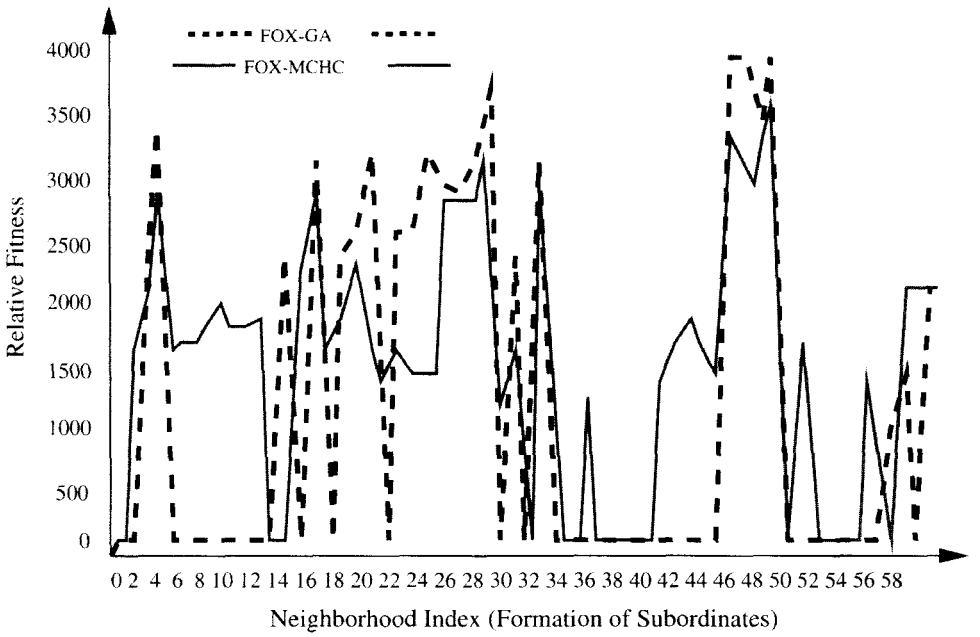

Figure 9: Niching Comparison of FOX-GA and FOX-MCHC; population of 200

most promising niches at the expense of the less promising niches. For example, in Figure 9, FOX-GA produced COAs in fewer niches (23 niches) than FOX-MCHC, which produced COAs in 45 niches. However, FOX-GA found better COAs in any given niche than did FOX-MCHC. Furthermore, the niches that FOX-GA elected not to preserve produced only mediocre COAs. This leads to the notion of "useful" versus "useless" diversity.

In general, battlestaff typically prefer to be presented with large sets of diverse COAs so that they can consider different trade-offs. However, having a large variety of COAs is not the only criteria that is important to battlestaffs. It is better, in general, to select from a smaller set of higher quality choices than it is to select from a wider range of mediocre choices. Thus we conclude that FOX-GA probably provides a more appropriate set of choices to battlestaffs than does FOX-MCHC. The search must provide the user not only with *diverse* solutions, but also with *high quality diversity*.

Figure 10 illustrates that if the battlestaff wish to increase the diversity number of COAs while still maintaining high quality, they can achieve it by adjusting the size of the base population. When the population size is increased to 600 (as shown in Figure 10), FOX-GA produces 36 excellent solutions, which is thirteen more than it produced with a population size of 200.

Figure 11 summarizes the differences between FOX-GA and FOX-MCHC for populations of 200, 600, and 1200. FOX-MCHC consistently produces more solutions, but FOX-GA consistently produces better solutions at all population sizes.

8 Discussion

With smaller populations, FOX-GA is capable of producing reasonable COAs in less than five minutes, but takes up to 20-30 minutes to deliver the best COAs. Both of these times are

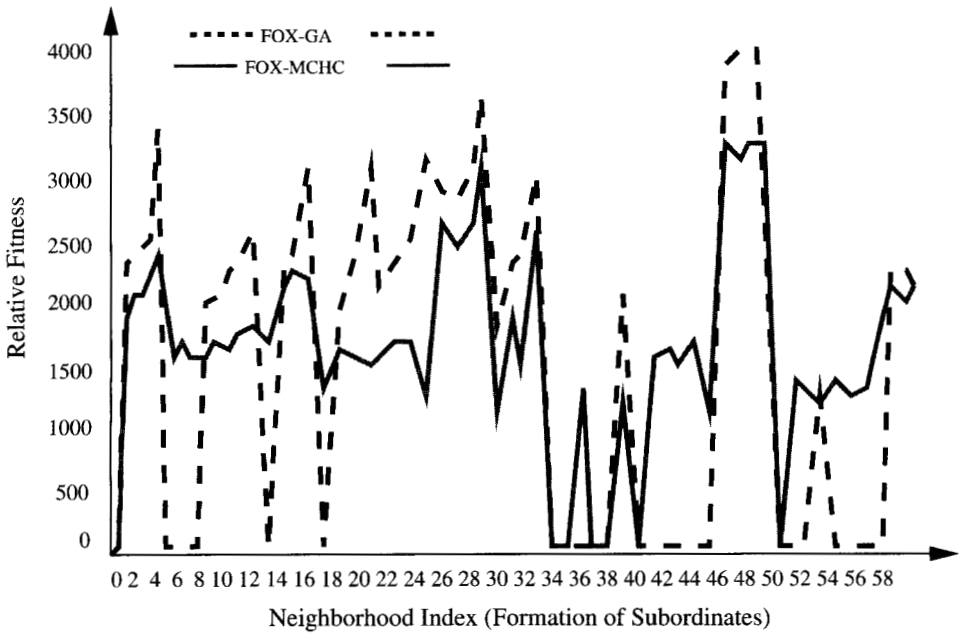

Figure 10: Increasing the population to 600 increases diversity

Population Size	# of Neighborhoods Competed in by: FOX-GA	FOX-MCHC	# of GA wins over MCHC	# of MCHC wins over GA
200	23	43	21	0
600	36	53	36	0
1200	52	56	49	2

Figure 11: Users can adjust the diversity of COAs produced by FOX-GA by changing population size

similar to typical development times for current battlestaffs. This is important because most battlestaffs would be willing to wait 5 minutes, but rarely have the time available to wait 30 minutes.

FOX-GA offers four main advantages over the typical manual methods currently used by battlestaffs. First, FOX-GA produces large numbers of distinctly different and valid COAs. Typical battlestaffs sometimes have trouble identifying more than a handful of COAs. Second, COAs generated by FOX-GA are already stored and represented in the computer, which eliminates the need to manually enter developed COAs into the computer. Data entry is a time consuming process when performed by humans. Third, FOX-GA provides more comprehensive wargaming data than humans can usually produce manually. A battlestaff typically requires 10-15 minutes to wargame one friendly COA against one enemy COA manually, while FOX evaluates 3000 friendly COAs per minute, with each friendly COA wargamed against 6 enemy COAs. FOX-GA allows a much more thorough analysis to be performed much more rapidly. Fourth, FOX-GA provides users with a very flexible way to explore COA options because of the flexibility inherent in GAs. This enables them to tailor FOX-GA to meet needs unanticipated in the original objectives. Thus, by changing the parameters of the GA (e.g., population size and the number of generations) they can change the behavior of FOX-GA to return solutions more quickly or generate a broader, more useful diversity of solutions.

We feel the research on FOX-GA demonstrates the benefits that can be attained when domain experts (MAJ Schlabach) closely collaborate with methods experts (Hayes and Goldberg) for extended periods of time. While such a claim may seem obvious, the close cooperation between domain and methods experts is too often missing in development of intended solutions. Our hope is that the contributions of FOX-GA will reinforce the notion that domain experts need to be intimately involved in teams for developing real systems for complex domains.

9 Future Work

Future work includes projects to generalize FOX to allow for any combination of terrain, enemy forces, and friendly forces; and to construct an intelligent human-computer interface to allow users to view trade-offs offered by selected COAs. Such a display would be presented on a three dimensional graph which would simultaneously show the COA's ability to 1) capture terrain, 2) inflict enemy losses, and 3) to preserve the lives of enemy troops. This would allow the battlestaff to quickly visualize trade-offs. We also plan to enhance FOX-GA's performance by investigating the use of other GA methods. We will conduct a comparative analysis of FOX-GA and various non-GA search methods such as non-GA evolutionary computing, simulated annealing, and linear programming. We have already done a number of usability studies in which feedback from users was used to identify missing functions and awkward interfaces in FOX-GA. Finally, a user evaluation of the effectiveness of FOX-GA for users of a variety of experience levels is currently underway. Early results indicate that users explore a wider variety of COAs, and in more detail, than they do by hand.

10 Summary and Conclusions

FOX-GA demonstrates the effective application of a genetic-algorithm for rapidly generating large numbers of offensive COAs (plans), evaluating their fitness, and identifying a diverse set of distinctly different high quality options which can be presented to users. In designing this system, it was important adapt the GA methods and simulators used by the system so as to produce solutions that made sense in the context of the domain, and to meet users'

domain-based needs for speed, solution diversity and quality, control, and flexibility. Many of these challenges are common to decision support tasks in other domains such as medical decision making, manufacturing planning, mechanical design, and architectural design.

A major difficulty in addressing the challenges of this problem was in satisfying all of the needs simultaneously: diversity, quality and efficiency. The niching strategy used by the GA proved important in producing solutions that were both diverse and high quality within each niche. An important issue in enabling efficient evaluation of solutions was the development of an efficient wargamer to evaluate COAs rapidly. This gain in efficiency was achieved by first developing the BRS Architecture, a set of course-grained representations of domain concepts that allow appropriate yet intelligent trade-offs to be made between computational efficiency and accuracy of evaluation of fitness.

A comparative evaluation of three search techniques (FOX-GA, FOX-MC, and FOX-MCHC) showed that FOX-GA's genetic algorithm and niching strategy substantially outperformed a more traditional monte carlo search (FOX-MC). The reason for FOX-GA's superior performance was that its genetic algorithm allows the search to take advantage of the structure inherent in the solution space, while the monte carlo search can not take advantage of this structure. A second finding from the comparative evaluation was that although FOX-GA's average "fitness" performance was similar to that of the third search strategy which used a mixed monte carlo and hill climbing (FOX-MCHC), FOX-GA's individual solutions tended to be superior and its solution set more appropriate for the application. Although the hill climbing strategy produced a broader variety of solutions, FOX-GA produced a reasonably diverse set of COAs that were higher in quality, and therefore more useful overall for our application which requires both solution diversity and quality. Finally, we found that the diversity of the solutions produced by FOX-GA could be increased by simply increasing the population size. This flexibility inherent in genetic algorithms provides users with flexibility in the way they can choose to explore the problem space. They can adjust the amount of solution diversity produced to suit their preferences.

FOX-GA's impact will be to assist battlestaffs, who are both time constrained and cognitively overloaded, to rapidly explore options, plan, and better cope with the information demands of modern warfare. Additionally, since the challenges in this problem are common to a wide range of complex decision support problems, we believe FOX-GA's approach will be useful in a wide range of domains.

References

ALBM Final Report, LMSC f376129. Lockheed Missiles and Space Company's Final Report on the AirLand Battle Management Project.

Jones, R. M., Wray, R., van Lent, M., and Laird, J. E. (1994). Planning in the tactical air domain. In *Planning and Learning: On to real applications.* Technical Report No. FS-94-01, pages 83–87.

Kuchinski, M. J. (1985). *Battle Management Systems Control Rule Optimization Using Artificial Intelligence.* Technical Report MP 84-329, Naval Surface Weapons Center.

Ling, R., Steinberg, L., and Jaluria, Y. (1993). Msg: A computer system for automated modeling of heat transfer. *Journal of Artificial Intelligence in Engineering Design, Analysis and Manufacturing,* 7(4):287–300.

Miller, J. (1996). *A Load Flow Simulator to Aid in Modification of Existing Steam Networks.* Master's thesis, University of Illinois at Urbana-Champaign.

Porto, W. V. and Fogel, L. J. (1997). Evolution of intelligently interactive behaviors for simulated forces. In *Evolutionary Programming IV, Sixth International Conference, EP97,* Indianapolis, Indiana.

Reich, A. J. (1995). An efficient representation of spatial data for terrain reasoning by computer generated forces. ELECSIM 95 - Electronic Conference on Scalability in Training Simulation. Held on the Internet April 10 - June 16 1995.

Schlabach, J. L. (1997). *The Illinois Conceptual Architecture: Enhanced Support to Battlefield Reasoning Applications*. Master's thesis, University of Illinois at Urbana-Champaign.

Wilkins, D. E. and Desimone, R. V. (1994). Applying an AI planner to military operations planning. In Zweben, M. and Fox, M., editors, *Intelligent Scheduling*. Morgan-Kaufmann.