IMPERIAL COLLEGE LONDON

E4.35
C2.4
ISE4.25

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2003

MSc and EEE/ISE PART IV: M.Eng. and ACGI

**FUZZY SYSTEMS**

Coursework only, t shirt and

Time allowed: 0:15 hours

**There are ZERO questions on this paper.**

**Answer ZERO questions.**

**Any special instructions for invigilators and information for candidates are on page 1.**

Examiners responsible First Marker(s) : De Wilde, P.
Second Marker(s) : Mamdani, E.H.

'CORRECTED COPY. June 03

# Fuzzy Systems Exam Coursework 2002-2003 Deadline April 28th 2003 at 10.00 am *

Download the research paper *Computational Military Tactical Planning System*, R. H. Kewley and M. J. Embrechts, IEEE Transactions on Systems, Man, and Cybernetics, Part C, Vol. 32, No. 2, May 2002 via the library's access to ieeexplore.ieee.org

This paper describes an automatic system to assist with moves on a battlefield. The usefulness of a move is calculated using fuzzy logic. The moves are generated using genetic algorithms, to be explained below. The detection of targets, and the effectiveness of shots is modelled using probability distributions. The aim of this coursework is to refine the fuzzy logic module.

Read sections I and II. Remember that the battlefield application is just an example of a decision making process using geographical data. The expansion of a large company could be modelled along similar lines.

Section III introduces genetic algorithms. This is a technique in which a solution to a problem (here making a battlefield move that brings you closer to your objective) is encoded in a vector. The components of this vector are all the numerical values necessary to specify the move completely. This vector is called a chromosome later in the paper. To start the genetic optimization, a number of such vectors are generated. The set of vectors is called a population. Each element of the population (each vector) has a fitness. The fitness is calculated in section III B. using fuzzy logic. The genetic algorithm optimizes the fitness of the population. It does this by mutation and crossover. Mutation is a random change to the vectors in the population. In crossover, elements of two different vectors are combined into a new vector.You do not have to program genetic algorithms in this coursework, but you will need to change the way fitness is calculated, so it is important you understand the basics of how genetic algorithms work.

Section III also mentions multiple objective criteria. This means that the objective is specified by more than one variable. To reach the objective, several variables need to be close to specified values. Read all of section III,

*Post it in the appropriate locked box in the undergraduate office, level 6.

paying specific attention to III B., on which your coursework will be based. The abbreviation "Obj-Rich" in this section stands for Objective Rich, where Rich is just a label, see Fig. 9. The word "by" on p. 163, line 24 should read "be".

You now also have to read sections IV to X, to get an idea of what the authors have achieved. You can skip sections VII B 3 and 4 on co-evolution, which is a special way of evolving populations. In section VIII, the authors use statistics, a null hypothesis and t-tests, to show the effectiveness of their solution. You do not need to understand this. Now that you have read everything, what are the multiple objective criteria in this problem (2/20)? 2

The main part of your coursework consists in a gradual refinement of the rules for preferences in III B., and the way they are applied. Replace the numerical value for "Best-battle-plan" by a linguistic variable (1/20). 1 2
Re-formulate the preferences as one or more fuzzy graphs (2/20). Whether you use one or more graphs is your decision. How does Fig. 2 now look like (2/20)? Remember that marks are awarded for a description of your results, it is not sufficient to write "I have done this", or show a graph without explanation. 2

The four stages of reaching the objective described in the *first* column of p. 163 imply a time dependence. How would you make the fuzzy graphs you have just derived, dependent on time (2/20)? How would you implement the four stages to work consecutively (1/20)? What difference does it make to implementing them all at the same time (5/20)? To answer the last question, do some simulations. You can also use simulations to answer the other questions about time dependence. Use simple membership functions. 2 1 5

In your simulations, one membership function should be centered around the CID number on your college security card, multiplied by a power of 10 of your own choice, but such that the significant digits used make your simulations verifiably different from those of your friends. I should be able to verify the difference from your report, not by inspecting your code. If your security card bears no number, use your date of birth. integer

**The Challenge**. If you want to get top marks, you have to do this challenge. However, you will get better marks for a good report without the challenge than for a mediocre report with the challenge solved.

The way of calculating preferences in the system in III. B. is called a Sugeno fuzzy inference system. How does this differ from Mamdani fuzzy inference? Does the choice between Sugeno or Mamdani fuzzy inference make any difference for the application in this paper (5/20)? Provide a well-justified answer, based on simulations if necessary. This challenge is the only part of this coursework for which you may have to look up a reference. 5

You could organize your work as follows.

day 1 Read the paper, looking up anything you don't understand in your lecture notes. Plan what you are going to program.

day 2 Answer the questions that can be answered without programming. Do the programming, and debug your program.

day 3 Run the simulations, and collect the results in a form that you can present in your report. Simulations can be in any programming language, on any machine. The use of Matlab or other software packages will simplify your work, but make sure that you have control over the parameters that you want to vary. If you are desperate, you could use pen and paper, but this will make this coursework difficult and abstract.

day 4 Write the report. It should be maximum six pages (single sided) a4, in a font not smaller than 10 point. You will not get marks for anything exceeding six pages, even if it is appendices. Font size in tables and figures should be at least 10 point, or the tables and figures will not be marked. Describe the problem, and how you have solved it. Describe your simulations, but do not give programme listings. Do not give references to the literature. Make sure you do and answer everything that is asked for in the coursework. Do not bind the report, but staple the pages together. Mention your name, and indicate for what degree (e.g. MEng Elec. Eng., MEng ISE, MSc) you are studying.

day 5 Check the consistency and quality of your work. Make last minute changes if necessary. If you feel confident and have the time, tackle the challenge. Resist the temptation to spend more than five 8-hour days of intensive effort on your coursework. You will not be compensated for it in marks. Just as an exam paper requires a concentrated effort over a few hours, this coursework requires a concentrated effort over a few days.

Do not forget to attend on the "exam" day. This day will be advertised in your exam schedule. Bring a copy of your report with you, and your college security card. I will ask you one or two questions based on what you have written in your report, to make sure that you have written it yourself. No preparation is necessary.

Good luck.

Dr. P. De Wilde

# Computational Military Tactical Planning System

Robert H. Kewley and Mark J. Embrechts, *Member, IEEE*

*Abstract*—**A computational system called fuzzy-genetic decision optimization combines two soft computing methods, genetic optimization and fuzzy ordinal preference, and a traditional hard computing method, stochastic system simulation, to tackle the difficult task of generating battle plans for military tactical forces. Planning for a tactical military battle is a complex, high-dimensional task which often bedevils experienced professionals. In fuzzy-genetic decision optimization, the military commander enters his battle outcome preferences into a user interface to generate a fuzzy ordinal preference model that scores his preference for any battle outcome. A genetic algorithm iteratively generates populations of battle plans for evaluation in a stochastic combat simulation. The fuzzy preference model converts the simulation results into a fitness value for each population member, allowing the genetic algorithm to generate the next population. Evolution continues until the system produces a final population of high-performance plans which achieve the commander's intent for the mission. Analysis of experimental results shows that co-evolution of friendly and enemy plans by competing genetic algorithms improves the performance of the planning system. If allowed to evolve long enough, the plans produced by automated algorithms had a significantly higher mean performance than those generated by experienced military experts.**

*Index Terms*—**Fuzzy logic, genetic algorithms, multiobjective optimization, simulation.**

## I. Introduction

MILITARY tactical course of action development is a very complex and essential component of the military tactical decision-making process [1] used to develop tactical operations orders, the plans which direct soldiers into battle. Human planners have difficulties in dealing with the complexities of this task [2]. This paper demonstrates the capabilities of a computationally intelligent system to perform military tactical course of action development. This system models human preference using fuzzy sets from soft computing. It performs tactical planning using genetic algorithms, also from soft computing. It models combat attrition using stochastic system simulation, a more traditional hard computing technique.

Emerging computational intelligence techniques including genetic algorithms and fuzzy inference systems complement combat simulation technology to automatically search for potential courses of action which accomplish specific mission goals. This paper uses fuzzy-genetic decision optimization as a technique to perform this search task. The decision-maker enters his preferences using a graphical user interface to define his preference for any potential outcome of a battle. Planners define a tactical scenario containing the terrain, friendly forces, and enemy forces anticipated for the battle. A genetic algorithm performs an intelligent search which allows a set of courses of action to emerge in the final population. These courses of action yield high achievement of mission goals in the combat simulation.

In a battle planning experiment, computational techniques developed tactical plans which, when evaluated using a combat simulation, performed as well as or better than plans developed by experienced military experts. This better performance was consistent with the commander's preferences for this battle. This result shows the potential promise of computational techniques to aid human planners in tactical course of action development.

## II. Course of Action Development

The tactical combat decision made on the ground during battle is one of the most difficult decisions faced by the military professional. These decisions are often made under great stress. Battle staff officers go for many days without sleep in their efforts to thoroughly analyze all aspects of the recommendations they make. Commanders often do the same, ensuring that their soldiers are prepared to fight. The physical demands, the time demands, and the emotional stress of the battle weigh heavily on the minds of all involved. These endeavors ultimately have a negative impact on decision making.

Battle planners often find it impossible to analyze the courses of action they develop. The process involves too much data. Enemy options, weapons system performance characteristics, friendly capabilities, and large expanses of terrain all interact in a complex manner to determine a winner or loser. Planners must perform this analysis in poor lighting conditions with smeared maps and mushy pens. They often resort to implementing oversimplified rules of thumb or doing what worked last time, even if the situation has changed.

Course of action development and analysis is a difficult task for even the most experienced planners. They must estimate attrition ratios, line of sight across undulating terrain, movement times, and enemy locations and concentrations in order to successfully place friendly forces. By doctrine, battalion planners place platoons in position. Consider a typical battalion sector which may be 10 km wide by 20 km deep. Discretize this area into possible platoon positions 250 m apart. The problem of placing 12 platoons has $3200^{12}$ or $1.14 \times 10^{42}$ possible solutions. The planner must resort to rules of thumb and experience to identify and select a few alternatives for even the most cursory analysis. The human planner can use some automated help in this process. Tactical decision making can be improved by investigating the potential for computational techniques to assist planning.

Manuscript received March 6, 2001; revised May 28, 2002.

R. H. Kewley is with the Center for Army Analysis, Fort Belvoir, VA 22060-5230 USA (e-mail: Robert.Kewley@us.army.mil).

M. J. Embrechts is with the Department of Decision Sciences and Engineering Systems and the Faculty of the Information Technology Program, Rensselaer Polytechnic Institute, Troy, NY 12181 USA (e-mail: embrem@rpi.edu).

Digital Object Identifier 10.1109/TSMCC.2002.801352.

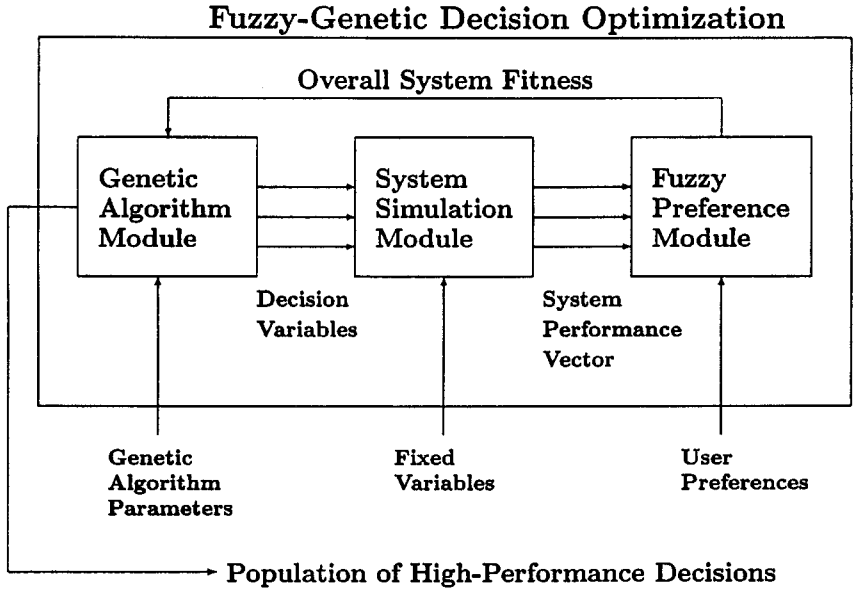


Fig. 1. Block diagram for fuzzy-genetic decision optimization. The genetic algorithm module passes proposed solutions to the simulation for evaluation. The fuzzy preference module aggregates the system performance vector into an overall system fitness, which is passed back to the genetic algorithm module. This process continues to produce a population of high-performance solutions to the problem.

## III. Fuzzy-Genetic Decision Optimization

Fuzzy-genetic decision optimization (FGDO) solves complex problems which require concurrent optimization of multiple objective criteria. It has three modules as shown in Fig. 1. The first module is a genetic algorithm which varies the system decision variables according to the system fitness determined by the output of the fuzzy preference module. FGDO accepts input parameters from the genetic algorithm and iterates the model and fuzzy inference system until it finds a population of good solutions to the problem. The second module is a simulation model which evaluates the proposed solutions to the decision problem. The third module is the fuzzy preference module. A graphical user interface allows the user to select the important variables, their ranges, and their order of satisfaction. These selections define a fuzzy inference system which aggregates the outputs of the simulation model into one overall fitness value for that particular solution. FGDO is best suited to complex problems with two distinct features. They cannot be modeled by simple equations suitable for optimization by other methods (such as linear programming, nonlinear programming, or goal programming), and they have multiple objective criteria.

### *A. Genetic Algorithms for Simulation Optimization*

Current approaches to simulation optimization offer very few techniques which allow optimization of nonlinear and stochastic systems. Azadivar gives a summary of available methods for simulation optimization [3]. Some of these are not appropriate for complex stochastic simulations. Gradient search methods require a differentiable closed form formula for the objective as a function of the controllable inputs. Finite difference techniques for estimating gradient descent are not well suited to stochastic models with high variability. A good estimate of the change in the objective function given a small difference in the input requires too many iterations.

Many complex systems require optimization of their structure, which allows the value, type, and number of inputs variables to vary. Population-based evolutionary algorithms, or genetic algorithms [4], a subclass of evolutionary algorithms, can handle this type of optimization [5]. They address the problems of high computation cost, convergence to local extrema, inability to handle qualitative variables, and ease of implementation. Evolutionary algorithms allow for mutation schemes which govern the selection of a new qualitative variable from the list of possibilities. The probabilistic exploration of the solution space by a population as opposed to a single member makes them less sensitive to local extrema as compared to other methods. An evolutionary algorithm is simpler to implement than solution techniques which require extensive knowledge of statistics and mathematics. Evolutionary algorithms are better able to handle noisy observations [6]. Evolutionary and genetic algorithms have been successfully applied to multiobjective optimization [7]–[9]. The evolutionary algorithm is a very flexible and efficient method for simulation optimization. It has the additional feature of being very easy to parallelize. For these reasons, they are well suited for recommending solutions to complex problems to a human decision-maker.

### *B. Fuzzy Ordinal Preference Model*

Many problems confronted by decision-makers require aggregation of preference over a large number of attributes. The military tactical decision used in this experiment is a representative example. If a military commander accepts a mission to eliminate enemy resistance in an objective area, his mission accomplishment may be measured by the number of enemy vehicles and the number of enemy personnel remaining on the objective. These, however, are not his only considerations. He also wants to preserve his soldiers and his vehicles. Furthermore, follow-on operations may necessitate conservation of fuel and ammunition. Definite conflicts exist in the objectives. A course

of action which completely eliminates all enemy from the objective will likely result in higher friendly casualties. It will also consume more resources. How does the commander define his preference over these conflicting alternatives so that he may devise a plan which seeks to maximize that preference?

Survey literature on multiattribute decision making breaks the main techniques of aggregating utility into two main schools [10], [11]. There exists the "American school" of utility or expected utility which seeks to aggregate all of the decision criteria into one utility function. The "European" or "French speaking" school developed out-ranking methods which develop a matrix of pair-wise comparisons of alternatives. Utility models require a simplifying assumption of preferential independence which often does not exist in real decision situations [12]. Out-ranking methods require the analyst to elicit extensive data from the decision maker [10]. A newer lexicographic approach suggested by Beroggi and Wallace [13], [14] does not allow compensation among criteria.

Fuzzy sets have been applied to additive utility models to represent imprecision in the model's values and weights [15], [16]. Several researchers have extended this idea to represent imprecision in analytical hierarchical process models [17]–[19]. Efstathiou and Rajkovič developed a methodology which interviews decision makers in order to develop a fuzzy inference system which calculates the utility of alternatives [20].

This paper uses an implementation of ordinal preference [14] using a Sugeno fuzzy inference system [21]. In fuzzy ordinal preference, the user, given a certain satisfaction level for the criteria, defines what improvement to seek next in building a preference from the worst conceivable case to the most optimistic. Consider a simple offensive battle in which an attacking force is trying to simultaneously destroy enemy forces, occupy a terrain objective with friendly forces, and keep from being destroyed. The decision maker may go through the following steps of ordinal preference.

The worst conceivable case is one in which enemy losses are *very low*, a *very low* number of friendly vehicles occupy the objective, and friendly losses are *very high*. Note the fuzzy linguistic approximations such as *very high* used to describe numerical results [2222]–[24].

1) In order to improve preference, the attacker may choose to first improve friendly losses from *very high* to *medium*.
2) Once he has retained acceptable friendly forces, he may then choose to occupy the objective with vehicles, improving that variable from *very low* to *very high*.
3) Once he was sure he could occupy the objective, he may seek to eliminate enemy forces, improving enemy attrition from *very low* to *very high*.
4) He could finally return to friendly losses and improve them from *medium* to *very low*, reaching the most optimistic result.

These preferences generate the following rules:

1) IF Percent-Friendly-Vehs-Destroyed is from-to very-high med AND Percent-Enemy-Vehs-Destroyed is greater-than-or-equal-to very-low AND Number-Friendly-Vehicles-In-Obj-Rich is greater-than-or-equal-to very-low THEN Best-battle-plan = $0.5 + -0.0050 *$ Percent-Friendly-Vehs-Destroyed.
2) IF Number-Friendly-Vehicles-In-Obj-Rich is from-to very-low very-high AND Percent-Enemy-Vehs-Destroyed is greater-than-or-equal-to very-low AND Percent-Friendly-Vehs-Destroyed is less-than-or-equal-to med THEN Best-battle-plan = $0.25 + 0.01 *$ Number-Friendly-Vehicles-In-Obj-Rich.
3) IF Percent-Enemy-Vehs-Destroyed is from-to very-low very-high AND Percent-Friendly-Vehs-Destroyed is less-than-or-equal-to med AND Number-Friendly-Vehicles-In-Obj-Rich is greater-than-or-equal-to very-high THEN Best-battle-plan = $0.5 + 0.0025 *$ Percent-Enemy-Vehs-Destroyed.
4) IF Percent-Friendly-Vehs-Destroyed is from-to med very-low AND Percent-Enemy-Vehs-Destroyed is greater-than-or-equal-to very-high AND Number-Friendly-Vehicles-In-Obj-Rich is greater-than-or-equal- to very-high THEN Best-battle-plan = $1.0 + -0.0050 *$ Percent-Friendly-Vehs-Destroyed.

The graph in Fig. 2 shows how the criterion improvement in each rule corresponds to a preference improvement using the linear equation in its consequent. For example, the antecedent of Rule 2 requires that the percentage of friendly vehicles destroyed be less than 50% (*med*), the percentage of enemy vehicles destroyed by less than 100% (*very high*), and the number of friendly vehicles in Objective Rich be less than 25 (*very high*). If all these conditions are true, the preference value for the battle outcome will be 0.25 (the minimum preference corresponding to this rule) plus 0.01 times the number of friendly vehicles in Objective Rich for a maximum preference value of 0.5 when 25 vehicles (the maximum possible—*very high*) are in objective Rich.

Strict ordinal preference requires the decision maker to reach his goal in one criterion before concerning himself with another. In the example in Fig. 2, he would begin to care about occupying the objective only if friendly losses were medium or better. The fuzzy implementation allows the decision maker to gradually consider the next objective as he approaches satisfaction in his current objective. In this example, as friendly losses moved from high to medium, the decision maker would begin to gradually consider occupation of the objective as an important criterion. This preference scheme accurately reflects the user's preference and allows for preferential dependence among the criteria. It is also relatively easy to understand and elicit using a user interface [25].

### C. Stochastic Simulation Models of Complex Systems

A simulation is one of the most powerful tools used in the design and evaluation of complex systems [26]. Many of the systems in the real world are so complex that mathematical models are intractable [27]. One alternative is to do experimentation with the real system itself. One can easily see how experimenting with critical systems such as traffic control, local economies, or military battles is inconvenient and expensive, if not impossible. In other cases, such as system design, the system of interest may not even exist. In these instances, it may be possible to simulate the system of interest [28]. This technique allows modelers to investigate new policies, rules, and designs

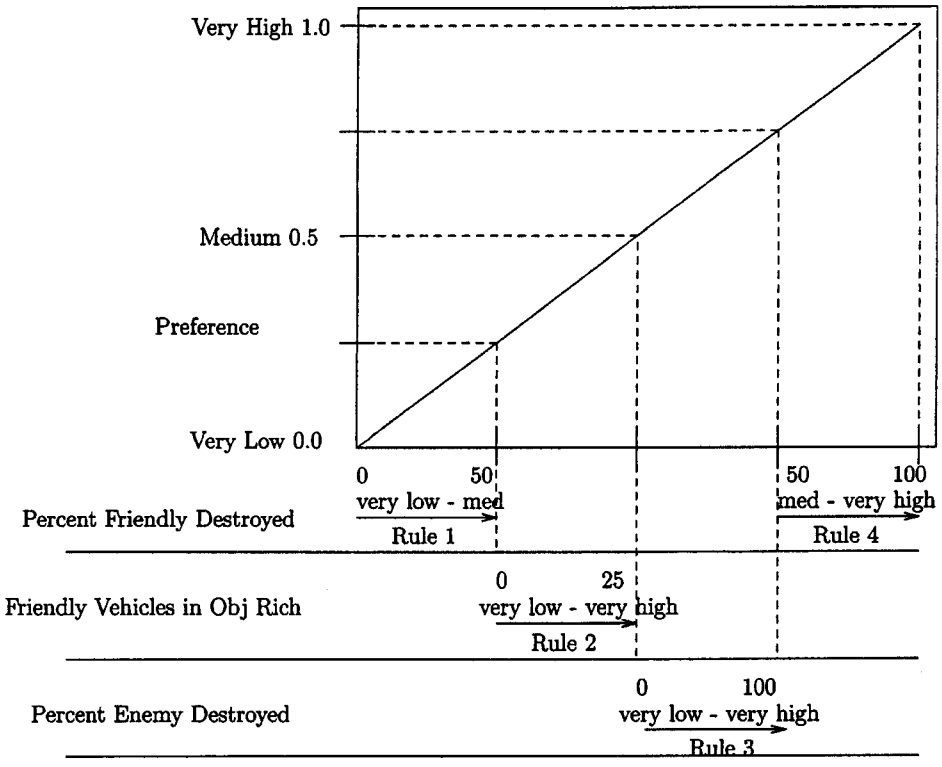


Fig. 2. Each rule in the fuzzy ordinal preference system maps a variable improvement, shown on the horizontal axis, to a preference improvement, shown on the vertical axis. For example, improving the number of friendly vehicles in Objective Rich from zero to 25 improves preference from 0.25 to 0.5 (medium).

without disturbing the real system. They can visualize system performance. Analysts can control and stop time and view data summaries to gain new insights into the critical variables and their effects on the system. A military battle is ideally suited to analysis by simulation.

There are some disadvantages to simulation. Model building requires specialized training, and the model quality depends on the modeler's skill. Results are stochastic and often difficult to interpret. The analysis can often be expensive, time consuming, and not worth the effort. Finally, simulation models are input–output models which yield the output of the system for a given input. These models are "run" rather than "solved." They do not generate an optimal solution. It is up the analyst to develop and evaluate different alternatives in search of better performance [29]. Since most complex systems have many different outputs, a clear notion of a "better" solution escapes the analysis. A combat simulation is no exception. The military planner is trying to optimize performance across a broad range of variables such as friendly losses, enemy losses, and control of certain terrain areas. He also has a wide range of tactical choices (inputs) which he may vary in countless combinations to seek better performance.

Advancing technology alleviates many of the disadvantages of simulation modeling. Many commercial simulation packages allow graphical and intuitive development of simulation models without as much specialized training. Their visual and statistical capabilities allow for easier analysis and interpretation of these models. Furthermore, increasing hardware speed and availability allows for larger run sets. These developments set the conditions for the success of fuzzy-genetic decision optimization. A genetic algorithm may now execute the large number of runs needed to converge to an optimum. The fuzzy preference model allows the user to aggregate simulation outputs to define a clear notion of better system performance. This notion, interpreted as a fitness value, serves as a road map to guide the search for better solutions. A combination of these software, hardware, visual, and algorithmic developments has a synergistic effect which yields a powerful tool for finding better policies, configurations, and designs of complex systems. Military tactical course of action development serves as an example domain where this approach is appropriate.

*1) The Stochastic Combat Simulation:* A combat simulation model estimated the performance of a potential friendly plan by running it against a set of possible enemy courses of action to determine the outcome. The simulation used the following models and data in its implementation.

Terrain data was imported from the Battlefield Effects and Weapons System Simulation (BEWSS) [29]. An 80 $km^2$ region of hilly and partially wooded terrain was represented by elevation postings at 50 m intervals. The presence of vegetation was also represented in the database. An explicit terrain line of sight model is needed to ensure that target detections is possible. The basic point-to-point line of sight model is the point-to-point intervisibility algorithm first developed for the SCIMITAR combat model [30]. Explicit line of sight calculations were done using these postings, and firers could not see through the trees. Furthermore, movement speed was slowed to one-third of normal speed in the trees.

The target acquisition model [31] calculates the mean time to detect a target by a regression function fitted to the experimental data. It considers range to target, target height, target velocity across the field of view, terrain complexity, and the probability

that the firer is looking in the target's direction. Detections are exponentially distributed. $P_{detect} = 1 - e^{-D \cdot T}$ where $T$ is the amount of time since the last detection computation for this observer. $P_{detect}$ is assumed to be constant for $T$. $D$ is the detection rate. In the simulation model, units, as opposed to individual vehicles, are acquired. It is assumed that the acquisition of one vehicle in a formation will also give away the position of other vehicles in that formation.

The shot-by-shot attrition model assumes that the target is a rectangular box and that the dispersion of the firer's shots follows a bivariate normal distribution where the elevation and deflection dispersions are independent of each other. If a target is $2L$ m long and $2H$ m high, and the weapons system has a center of aim at $\mu_x$, $\mu_y$ with standard deviations $\sigma_x$, $\sigma_y$, the single shot probability of hit will be

$$P_{hit} = \Phi\left(\frac{L-\mu_x}{\sigma_x}\right) - \Phi\left(\frac{-L-\mu_x}{\sigma_x}\right) \cdot \Phi\left(\frac{H-\mu_y}{\sigma_y}\right) - \Phi\left(\frac{-H-\mu_y}{\sigma_y}\right) \quad (1)$$

where $\Phi(z)$ is the cumulative distribution function of the standard normal distribution [32].

The indirect fire attrition model also uses BEWSS data [29]. Indirect fire systems launch munitions which disperse from their point of aim using a bivariate normal pattern. An individual munition must fall within a lethal radius of the target to have an effect. This radius is approximated by a square for simpler calculation. For submunitions, the probability of impacting within the lethal square is calculated using a triangular distribution based upon the dispersal radius of the submunition. For smart submunitions, each bomblet paints a circular footprint on the ground in which it has a probability of acquiring each type of target. If a target is hit by a submunition, the probabilities of firepower, mobility, mobility and firepower, and catastrophic kills are used to determine the outcome.

The indirect fire target selection model is based upon a simulated reporting system. Once a unit acquires a target, after a random report interval, the target is reported to the unit's fire direction center and considered to be available to all friendly indirect fire systems. All units supported by a particular fire direction center share this common indirect fire target picture.

## IV. Tactical Planning Experiment

A tactical planning experiment was conducted to assess the potential for a computationally intelligent system to autonomously generate high performance courses of action. Human battle planners may also view and modify these courses of action, making them more doctrinal while still retaining the overall scheme of maneuver. This experiment had five goals, as follows.

- To demonstrate intelligent, autonomous generation of high-performance plans by fuzzy-genetic decision optimization.
- To estimate the performance of three different approaches to enemy plan generation within fuzzy-genetic decision optimization. Friendly plans may evolve against a set of fixed enemy plans input by military experts, against autonomously generated enemy plans which co-evolve along with friendly plans, or against a combination of the two.
- To benchmark the performance of autonomously generated plans against those generated by expert military planners.
- To compare tactical planning performance with the aid of autonomously generated recommendations to tactical planning without autonomous aid.
- To collect performance data, parameter data, and insights to aid planning and give direction for follow-on tactical planning experiments.

## V. The Tactical Scenario

The planning task was to generate an offensive courses of action for a future mounted combat scenario in hilly and partially wooded terrain. The friendly forces, just complete with resupply operations in Assembly Area Green in the north (see Fig. 9), were given an immediate mission to continue the attack 15 km to the south to destroy defending enemy forces and seize the key terrain on Objective Rich. The enemy mission required them to destroy friendly forces and prevent them from gaining Objective Rich. Each force had to perform its mission while also preventing its own attrition. The friendly force commander entered his battle outcome preferences using a graphical user interface that implemented fuzzy ordinal preference as discussed in Section III-B. Given values for enemy vehicles destroyed, friendly vehicles destroyed, and friendly vehicles on the objective, the resulting fuzzy inference system determined a preference value from for the battle outcome. This value is used as a fitness value for each friendly plan evaluated in the combat simulation.

## VI. Course of Action Representation

Representation of an attack plan on a chromosome depends on the ability to place different objects, as opposed to bits, real values, or symbols, at different locations on the chromosome. An attack plan consists of one or more phases, each consisting of one or more contingencies. Phases are executed in sequence, one after the other. At each phase, the friendly force executes one of the available contingencies based on selection criteria specified by the commander. A constraints object constrained the allowable unit locations in the friendly area of operations to a set of discrete points each 300 m apart. A location object is simply an $x$, $y$ pair of map coordinates representing one of the allowable locations. Each unit in an attack plan is assigned an allowable destination location for each contingency in each phase of the battle. The chromosome arranges these locations by phase, then contingency, then unit. The algorithm randomly generates possible $x$, $y$ locations for each position on the chromosome using the following distribution. A unit is 50% likely to remain in place during a contingency. Otherwise, it moves to one of the battle area's discrete possible $x$, $y$ locations. Selection of each discrete $x$, $y$ location is equally probable. It initializes the chromosome by randomly generating an $x$, $y$ location for each position on the chromosome. During evolution, it implements mutation at each position on an offspring's chromosome as follows. If a random draw is less than the probability of mutation, the $x$, $y$ location at that position is replaced by another randomly

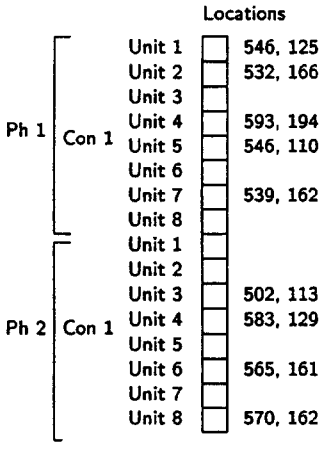


Fig. 3. Genetic algorithm represents a tactical plan as a linear chromosome with each unit's destination for each phase and contingency arranged linearly.

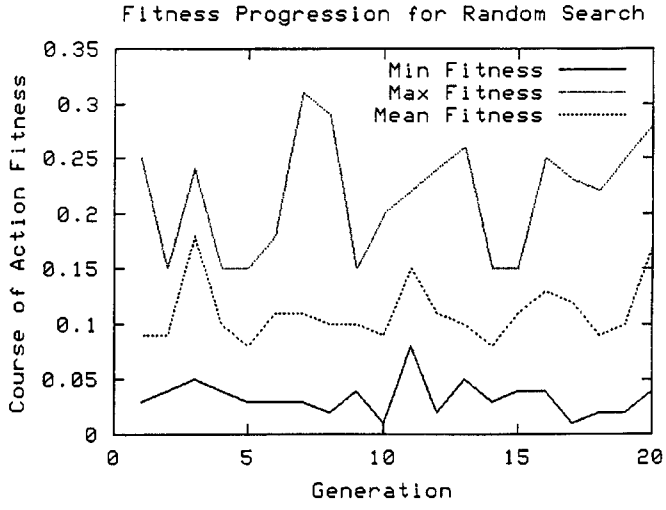


Fig. 4. Fitness evolution during random search. Note the lack of improvement over time, consistent with what one expects from a random search.

selected destination, to include the 50% chance of remaining in place.

For the battle plans used during the tactical planning experiment, each side was limited to two phases, each with one contingency each. In other words, the force would execute all assigned movements (a movement from the current location to its corresponding destination received from the location chromosome) for phase one, contingency one. Once complete with phase one, each unit would get a new destination from the location chromosome and execute those movements for phase two, contingency one. Since the friendly and enemy forces had eight and nine units respectively, the location chromosomes, which represented a battle plan for each force, had 16 and 18 allowable locations respectively. For the friendly plan represented in Fig. 3, Unit 2 would move from its initial location to grid location (532, 166) during phase 1, contingency 1. Then it would remain in place for phase 2, contingency 2.

## VII. Different Approaches to Battle Planning

This experiment tested six different methods of generating tactical courses of action—generation by military experts, generation by four different automated algorithms, and modification of automated plans by military experts.

### A. Course of Action Generation by Military Experts

A group of experienced military experts participated in this experiment. They were trained, experienced, and considered to be very capable of developing tactical courses of action for the size forces used in this experiment. This group had up to 30 min to develop an offensive plan for the experimental scenario. They used a military map to reference the terrain in the area of operations. They understood the composition, capabilities, and intentions of both the friendly and enemy forces. They used a computer graphical user interface showing the terrain in the battle area to input each course of action. When complete, they had generated a set of friendly courses of action for evaluation.

### B. Automated Algorithms

Four different automated course of action generation algorithms were tested. All of these methods generated courses of action with no human interaction. Each of these experiments used a desktop personal computer with a 450 MHz processor running the Linux operating system (ver. 2.2).

*1) Random Search:* In the random search method, the algorithm randomly generated 400 courses of action and tested each of them against two enemy plans developed by military experts. The algorithm submitted for evaluation the course of action which had the best average performance against two opponent plans. This experiment required 35 min of computation time. Fig. 4 shows the fitness progression for one of the random search trials. The minimum, mean, and maximum fitness values represent the statistics for each of 20 samples of 20 random plans each, analogous to a genetic algorithm population. Note the high variability within the population and the lack of an improvement in mean fitness value over time. This is what one would expect from a random search. It contrasts with the graphs for more intelligent algorithms. This experiment used the results of the random search to confirm the intelligence of the other automated algorithms. A truly intelligent algorithm should outperform a random search, given the same number of trial courses of action.

*2) Genetic Algorithm Evolution Against Static Enemy Plans:* The first intelligent search method employed a genetic algorithm. The genetic algorithm used a single population of 20 members for 20 generations, generating and evaluating 400 battle plans. The fitness of each plan was its average fitness against two enemy courses of action developed by military experts. The genetic algorithm replaced 60% of the population with each generation. Since the fitness evaluation function, the combat simulation, was stochastic, this replacement rate is intended to ensure that a high performance plan did not drop out of the population due to an unlucky fitness evaluation. A plan only had to score in the top 40% to remain in the population. Over time, consistently high performing members remained in the populations as others dropped out. After fitness evaluation for all members, the eight best members were copied into the next generation. A tournament selection algorithm selected two parents for mating as follows. It randomly selected three members of the current population, compared fitness values,

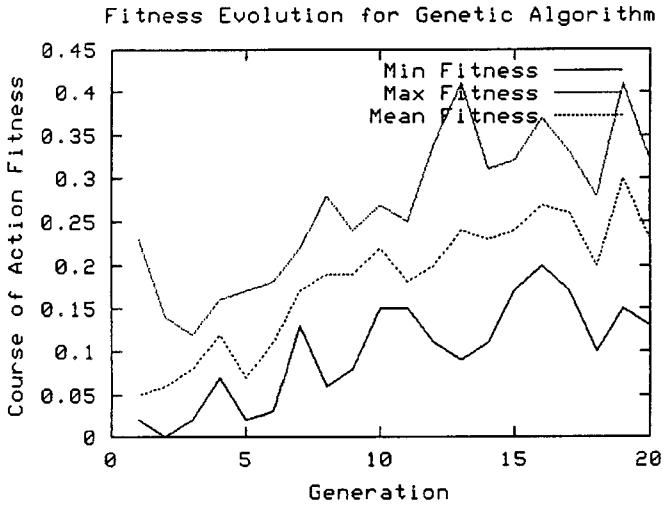


Fig. 5. Fitness evolution during the genetic algorithm. Note the upward trend in both maximum and mean fitness, demonstrating at least some level of intelligence.

and selected the best as a parent. All members were replaced, and the tournament algorithm selected a second parent. If a random draw with a probability of success of 0.9 yielded a crossover, the two parents performed single point crossover to form two offspring. Without crossover, the two offspring were simply clones of their parents. Before the offspring were inserted into the new population, each location on the chromosome was subject to a 0.1 probability of mutation. Upon mutation, the $x$, $y$ coordinate pair on the chromosome was replaced with a uniform random draw from the set of possible locations in the area of operations. The procedure continued until 12 new offspring joined the eight members already copied into the new population. The genetic algorithm submitted the best member of the final population as a recommended high-performance course of action. Four independent genetic algorithms generated four different courses of action to evaluate against other methods. This experiment required 36 minutes of computation time, about the same amount of time as the random search. The graph in Fig. 5 shows the fitness evolution for one of the genetic algorithms. Note the improving trend in both the maximum and mean fitness values for each population and the lower diversity of the genetic algorithm as compared to the random search. This trend graphically demonstrates improving fitness as the population of plans evolves.

*3) Co-Evolution:* For the co-evolution method, a competing genetic algorithm, as opposed to a human planner, developed the enemy courses of action used to evaluate the fitness of the friendly courses of action. This technique employed four friendly and four enemy genetic algorithms each evolving an independent population of tactical plans. The genetic algorithms employed during co-evolution used the same parameters and genetic operators as the genetic algorithm method described in Section VII-B2. However, the fitness evaluation was different. The four enemy genetic algorithms employed a fuzzy preference model for enemy mission accomplishment in order to compete with the friendly courses of action. The enemy preference model sought destruction of the friendly force and preservation of enemy forces. The co-evolution algorithm randomly initialized four independent friendly populations and four independent enemy populations. For each generation, the algorithm shuffled each of the eight populations, giving each member a random ordering from one to 20. It then evaluated the first member of each friendly population against the first member of each enemy population. The fitness value for each first friendly member was its average performance against the four enemy courses of action ordered first in each enemy population. The fitness value for each first enemy member was its average performance against the four friendly courses of action ordered first in each friendly population. This scheme repeated itself from the second members all the way through the twentieth members. Once the combat simulation runs determined fitness for all members of all populations, each of the eight genetic algorithms independently generated new populations using the parameters and genetic operators described in Section VII-B2.

Upon completion, this algorithm submitted the best member of each population as a recommended course of action for the battle, giving four different friendly courses of action and four different enemy courses of action. The competitive design of this algorithm sought to cause the enemy side to continuously find and exploit potential weaknesses in the friendly plan, causing the friendly side to adapt and patch the weaknesses where possible. Although this technique evaluated the same number of plans (400) as the static genetic algorithm, doubling the number of enemy plans to evaluate against from two to four doubled the computation time required for this experiment to 72 minutes. The graphs in Fig. 6 show this process for two of the competing populations. As the populations co-evolve, the average fitness of the friendly plans goes up slightly, but then levels off. The fitness of the enemy plans does not change much during the course of the entire algorithm. Even though each side continues to evolve better courses of action, the improvements counteract each other, and the average fitness remains fairly constant for each side.

*4) Co-Evolution and Genetic Algorithm Hybrid:* This method was similar to the co-evolution method. However, the fitness value for the friendly forces was a linear combination of the course of action's average performance against four automated enemy courses of action in the competing enemy populations (50% weight) and the average performance against two static courses of action developed by military experts (50% weight). This technique sought to evolve friendly courses of action which performed well against a military expert's best guess of enemy intentions and against machine generated courses of action which seek to exploit friendly weaknesses. Because this experiment evaluated each of the 400 friendly plans against six different enemy plans, it required 110 min of computation time.

### C. Human Modification of Automated Results

For this technique, the co-evolutionary/genetic hybrid approach generated two independent friendly courses of action. The human experts then modified these courses of action as they saw fit. They were instructed to develop their plans by first visualizing the automated plans in the simulation. They could then select one plan to modify, build a combination of the two plans, or build their own plan from scratch.

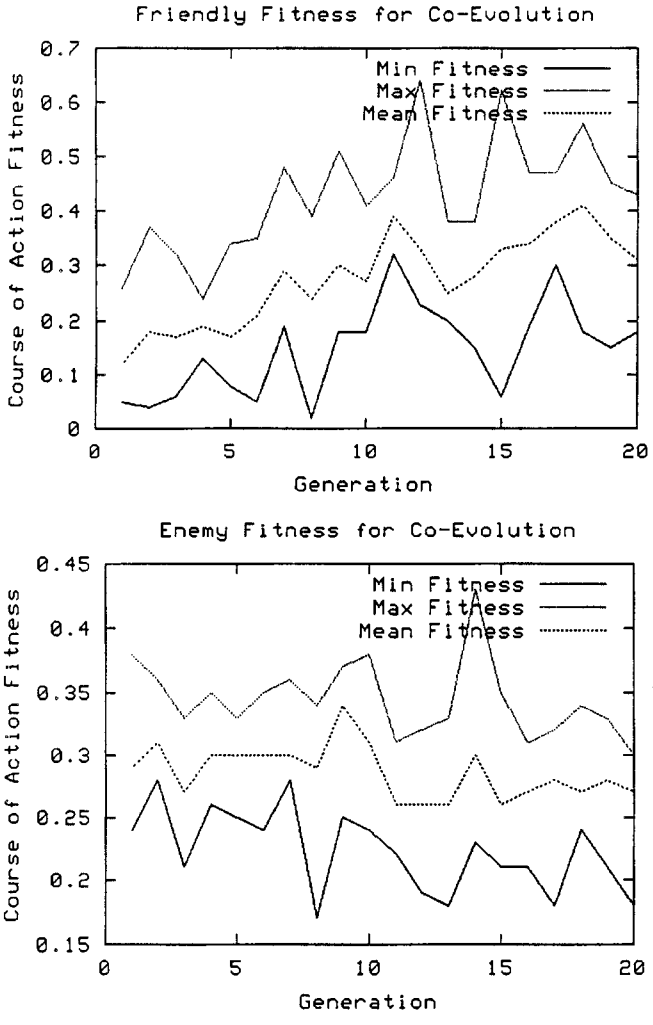


Fig. 6. Fitness for friendly and enemy forces during co-evolution. The average and maximum fitness levels for each side level off and do not improve with additional time. However, this does not mean that continued evolution is fruitless. Each side improves, and their improvements counteract each other.

## VIII. Experimental Results

The initial phase of the experiment sought to determine which of the automated algorithms showed the best performance upon subsequent evaluation of the plans they developed. Each algorithm, in separate and independent trials, developed four friendly attack plans for the battle scenario. The combat simulation model evaluated each attack plan five times against six different enemy courses of action developed by six different military experts, resulting in 30 different results. The fuzzy preference system evaluated the fitness of each result, and the average fitness produced by the 30 results was the estimated performance of a friendly plan. This yielded an estimated performance for each of the four plans developed by each algorithm. The diamond plots in Fig. 7 show the mean performance and standard deviation by each automated algorithm. The plot clearly shows the better mean performance by the co-evolution method and the hybrid method. It appears clear that each of the co-evolutionary methods outperforms standard genetic evolution against static enemy courses of action. Finally, the fact that each of these automated methods outperforms a random search by a factor of two confirms the intelligence

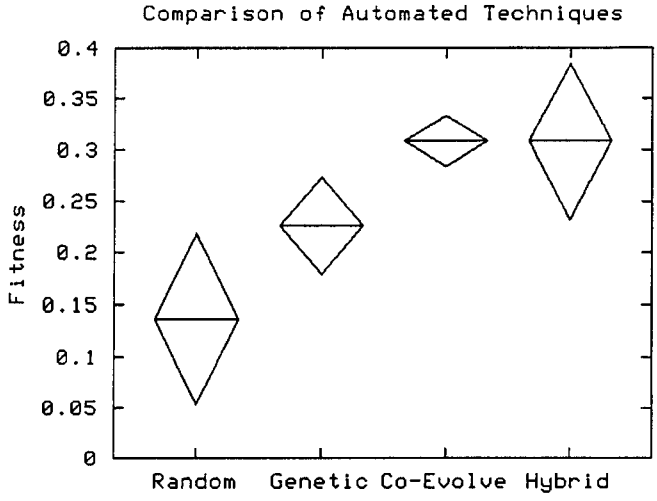


Fig. 7. These diamond plots show mean performance (center line) and standard deviation (tips) for each of the automated planning methods. Of the four methods tested, co-evolution and the co-evolution/genetic hybrid outperformed both a random search and a genetic algorithm evolving against a set of fixed enemy plans.

of these algorithms. A series of $t$-tests tested the hypothesis that the mean performance of the co-evolution method was equal to the mean performance of the other algorithms, with the alternate hypothesis that the mean performance of the co-evolution method was greater than the mean performance of other methods. The statistics in Table I confirm the graphical observations. The increased performance of co-evolution over a standard genetic algorithm and over a random search is statistically significant at the $\alpha = 0.10$ level, but the difference between co-evolution and the hybrid method is not.

This analysis has shown that both co-evolution methods do indeed produce intelligent results. It follows from the theory behind these methods that their performance should increase as they are allowed more computation time. This will give the friendly genetic algorithm more opportunity to search for better solutions and the enemy genetic algorithm more opportunity to provide counter-plans to which the friendly algorithm must adjust. To test this theory, the experimenters allowed each algorithm to continue to evolve past 20 generations to 40 generations, and the algorithm submitted the best performing members of each population for evaluation. One would expect these plans to perform better than those plans submitted after 20 generations. As shown in Table II, the analysis rejected the null hypothesis with $\alpha = 0.10$ for both planning methods, concluding that performance improves with computation time. The experiment proceeds to the next phase with confidence that both methods of co-evolution intelligently search for and find high-performance plans.

The final phase of this experiment sought to see if the automated methods, or the human planner with automated assistance, could outperform unassisted human planners. The co-evolution and the co-evolution/genetic hybrid techniques each evolved for 40 generations to produce four friendly attack plans. This evolution took 145 min for the co-evolution technique and 219 min for the hybrid technique. These plans, along with four automated plans modified by human experts, were compared to four plans developed by unassisted experts.

TABLE I

RESULTS OF ONE-TAILED $t$-TEST USED TO TEST THE HYPOTHESIS THAT THE MEAN PERFORMANCE OF PLANS PRODUCED BY THE CO-EVOLUTION METHOD IS EQUAL TO THE MEAN PERFORMANCE OF PLANS PRODUCED BY OTHER METHODS WITH $\alpha = 0.10$. THE ALTERNATIVE HYPOTHESIS WAS THAT THE MEAN PERFORMANCE OF PLANS PRODUCED BY CO-EVOLUTION WAS GREATER THAN THE MEAN PERFORMANCE OF PLANS PRODUCED BY OTHER METHODS

| Planning Method | Time req'd | n | Mean Perf | Std Dev | $t$-value | $p$-value | Sig? |
|---|---|---|---|---|---|---|---|
| Co-Evolution | 35 min | 4 | 0.308 | 0.025 | - | - | - |
| Random Search | 36 min | 4 | 0.136 | 0.083 | 3.98 | 0.008 | Y |
| Genetic Alg | 72 min | 4 | 0.226 | 0.047 | 3.09 | 0.013 | Y |
| CE/GA Hybrid | 110 min | 4 | 0.308 | 0.076 | 0.01 | 0.495 | N |

TABLE II

PAIRED $t$-TEST TESTED THE HYPOTHESIS THAT THE MEAN PLAN PERFORMANCE FOR BOTH CO-EVOLUTIONARY ALGORITHMS AFTER 40 GENERATIONS WAS EQUAL TO THE MEAN PLAN PERFORMANCE AFTER 20 GENERATIONS WITH THE ALTERNATIVE HYPOTHESIS THAT THE MEAN PLAN PERFORMANCE AFTER 40 GENERATIONS WAS GREATER THAN THE MEAN PLAN PERFORMANCE AFTER 20 GENERATIONS. FOR BOTH CO-EVOLUTION AND THE HYBRID TECHNIQUE, THE ANALYSIS REJECTED THE NULL HYPOTHESIS AT $\alpha = 0.10$ AND CONCLUDED THAT PLAN PERFORMANCE IMPROVES AS COMPUTATION TIME INCREASES

| Method | $n$ | Mean Perf 20 Gen | Mean Perf 40 Gen | t-value | p-value | Sig? |
|---|---|---|---|---|---|---|
| Co-Evolution | 4 | 0.308 | 0.376 | 1.85 | 0.081 | Y |
| CE/GA Hybrid | 4 | 0.308 | 0.389 | 3.61 | 0.018 | Y |

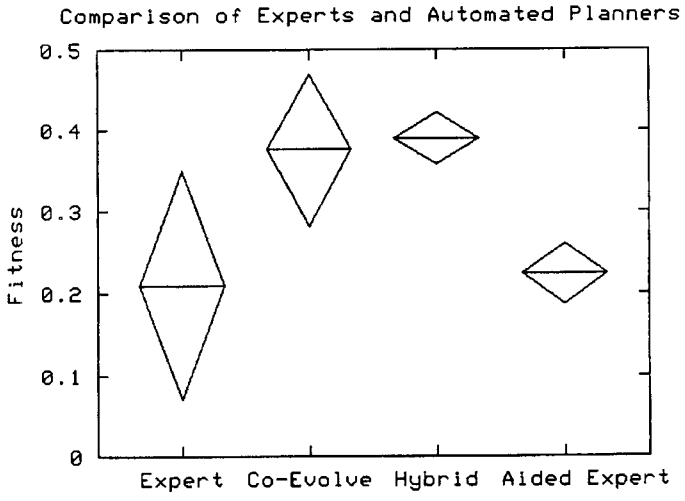


Fig. 8. These diamond plots show mean performance (center line) and standard deviation (tips) for plans developed by two automated schemes, by military experts modifying automated plans, and by unassisted military experts. Co-evolution and the co-evolution/genetic hybrid outperformed the human experts while the human experts modifying automated plans did not.

The diamond plot in Fig. 8 shows the relative performance for each method. Graphical analysis shows that both automated algorithms produced plans which, on average, clearly outperformed the military experts when evaluated in the combat simulation model. However, the mean performance of the automated plans modified by human planners was nearly the same as the mean performance of plans developed by unaided experts. Statistical analysis in Table III confirms the graphical analysis. From these tests, one concludes that co-evolution and the hybrid technique each outperformed the military experts while the human modifications to automated plans did not.

The graphical and statistical analysis of this experiment's results have shown that fuzzy-genetic decision optimization does indeed produce intelligent and effective solutions to complex military tactical problems. It has also shown that co-evolution of friendly and enemy plans, as opposed to evolving against a static set of enemy plans, is a powerful way to evolve robust friendly plans which perform well against a variety of enemy schemes. These automated planning techniques have a time cost, but advancing technology will make it less of an issue. If allowed to evolve long enough, they produce courses of action which perform better than plans developed by military experts. The best-performing automated techniques required 145 and 219 min at 40 generations, compared to the 30 min given to military experts. However, desktop computers with four times the processor speed of the one used in this experiment are currently available, and these machines will only get faster. Computational techniques have the potential to produce better performing tactical plans than human experts within the time constraints of the tactical situation.

## IX. EVOLUTION OF TACTICAL DOCTRINE

In addition to the objective analysis of this experiment's results, a subjective tactical analysis of the courses of action produced by fuzzy-genetic decision optimization illustrates that this system evolves sensible doctrinal behavior for the modeled forces. The evolved tactics make sense when viewed with a trained military eye. Fig. 9 shows the tactical plan discovered by using the co-evolution method of Section VII-B3. This plan uses the ground maneuver forces, two mechanized infantry platoons, to attack along Axis Attack to seize Objective Rich. The attack helicopters conduct a supporting attack along Air Axis Comanche. The unmanned aerial vehicle conducts aerial reconnaissance and observation from Observation Post 1 to spot enemy forces so that artillery units in Assembly Area Green can destroy them. Friendly scouts move beyond the objective to occupy a screen line. From there they protect the attackers moving across the objective. The roles assumed by forces in this mission match the doctrinal roles for which the combat vehicles in these units were designed. First, reconnaissance moves in to spot enemy and attrit him with artillery fires. Then, attack helicopters support by fire while maneuver forces move onto and seize the objective area.

In this mission, the friendly force adapted to a point where individual units performed doctrinal missions in advantageous terrain. These doctrinal roles were not entered as a constraint

TABLE III
RESULTS OF ONE-TAILED $t$-TEST USED TO TEST THE HYPOTHESIS THAT THE MEAN PERFORMANCE OF PLANS PRODUCED BY THREE ALTERNATIVE PLANNING METHODS IS EQUAL TO THE MEAN PERFORMANCE OF PLANS PRODUCED BY MILITARY EXPERTS WITH $\alpha = 0.10$. THE ALTERNATIVE HYPOTHESIS WAS THAT THE MEAN PERFORMANCE OF PLANS PRODUCED BY THE ALTERNATIVE METHODS WAS GREATER THAN THE MEAN PERFORMANCE OF PLANS PRODUCED BY MILITARY EXPERTS

| Planning Method | Time Req'd | n | Mean Perf | Std Dev | $t$-value | $p$-value | Sig? |
|---|---|---|---|---|---|---|---|
| Military Experts | 30 min | 4 | 0.209 | 0.139 | - | - | - |
| Co-Evolution | 145 min | 4 | 0.376 | 0.094 | 1.97 | 0.053 | Y |
| CE/GA Hybrid | 219 min | 4 | 0.389 | 0.032 | 2.50 | 0.044 | Y |
| Experts w/ Auto Aid | 249 min | 4 | 0.224 | 0.037 | 0.19 | 0.429 | N |

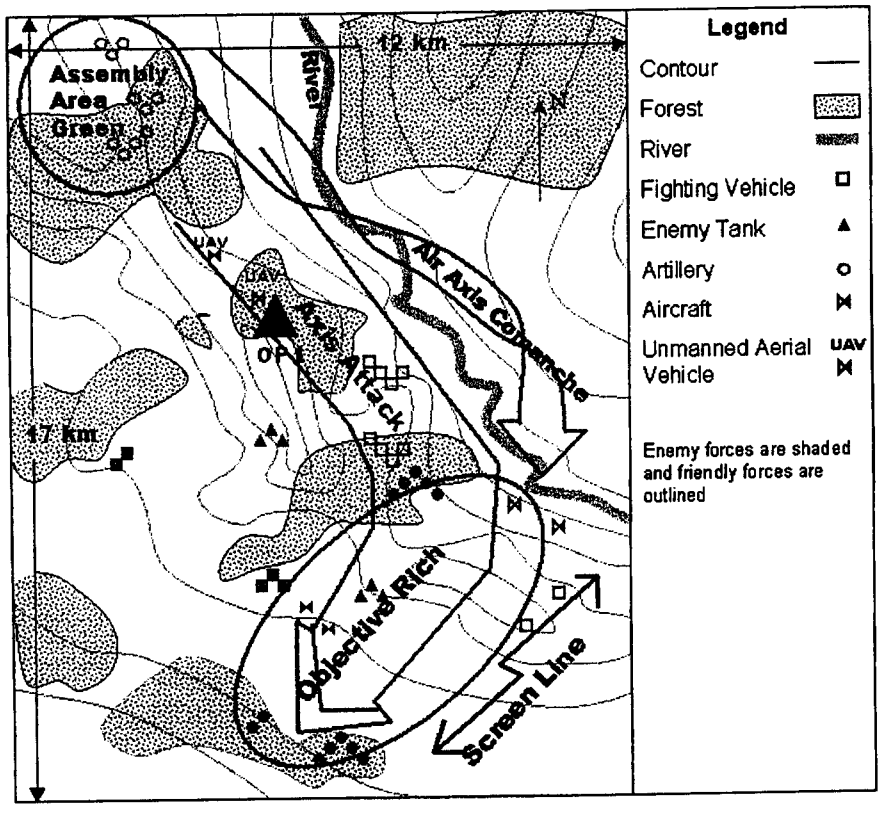


Fig. 9. Tactical plan developed by co-evolution. Two mechanized infantry platoons conduct the main attack along Axis Attack to seized Objective Rich while attack helicopters support from the eastern flank along Air Axis Comanche. An unmanned aerial vehicle occupies an observation post (OP1) to spot enemy forces for destruction by artillery, which remains behind in Assembly Area Green. Friendly scouts move beyond the objective to occupy the screen line and protect attacking forces moving across Objective Rich.

or restriction for the algorithm. The doctrine was an emergent property that evolved from successive trial and error in order to maximize accomplishment of mission goals. The doctrinal unit roles are due to a genetic search which discovered excellent performance using these roles. This feature of fuzzy-genetic decision optimization suggests that it could be used as a method by which military forces experimentally use computational methods to evolve possible doctrine for future forces, especially in situations where new technology, terrain, and enemy composition render old doctrine obsolete. This methodology, as opposed to expert opinion, is not subject to the bias and narrowed focus of current doctrine.

## X. CONCLUSIONS AND SUBSEQUENT ANALYSIS

It was demonstrated that military tactical course of action development is a complex and difficult task with which even the most experienced military professionals have difficulties. Fuzzy-genetic decision optimization is a technique which may be applied to this complex task. The military commander enters his preferences into a graphical user interface in order to develop a fuzzy inference system to estimate his preference for all possible battle outcomes. A genetic algorithm iterates a combat simulation model in order to search for a final population of high performance tactical plans which meet the commander's goals for the scenario. A planning experiment demonstrated that co-evolution of competing friendly and enemy plans increased the performance of this algorithm. In fact, these plans performed well when compared to plans developed by qualified military tactical planners.

Future experimentation and analysis will refine and improve these techniques. The performance of the military planners modifying automated plans has potential for improvement. Some empirical research and analysis is necessary to develop a system by which human planners can observe and evaluate a set of recommended courses of action produced by automated

algorithms, select a good candidate, and modify it to be implemented by military forces. An extension of this analysis would be to investigate the applicability of interactive evolutionary computation to plan development [33]. At certain intervals in the evolutionary process, human experts could subjectively augment the objective fitness evaluation by the combat simulation. Potentially, the evolved plans would perform well in the combat simulation and conform to the subjective standards of expert planners.

Computationally intelligent control of military forces fits naturally into the rapidly changing combat analysis landscape. The concurrent evolution of computer power, battlefield connectivity, and network-centric warfare [34] will yield an environment in which the most current friendly, enemy, and terrain information will be available for analysis. Computers distributed at all echelons of the battlefield will be able to use this information to continually search for improvements to the current tactical situation and share the results of that search. At appropriate intervals, military commanders will be able to query this collective network for recommended changes to force deployment plans, based upon the most current intelligence picture. Another more near-term use for computationally intelligent planning is for computer-generated forces embedded in current virtual and constructive combat simulation models. Computer-generated forces could develop and execute tactical plans entirely by automated algorithms. This could streamline scenario development and execution for these simulations, relieving military experts of the need to plan for and control the automated forces.

## REFERENCES


[1] "The command estimate process," U.S. Army Command and General Staff College, Fort Leavenworth, KS, Student Text 100-9, Feb. 1996.

[2] "National training center trends compendium," Center for Army Lessons Learned, Fort Leavenworth, KS, 99–1 ed., 1999.

[3] F. Azadivar, "A tutorial on simulation optimization," in *Proc. 1992 Winter Simulation Conf.*, 1992, pp. 198–204.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.

[5] H. P. Schwefel, *Evolution and Optimum Seeking*. New York: Wiley, 1995.

[6] H. Pierrevel and L. Tautou, "Using evolutionary algorithms and simulation for the optimization of manufacturing systems," *IIE Trans.*, vol. 29, no. 3, pp. 181–189, 1997.

[7] L. J. Fogel, "Scheduling projects to maximize net present value—The case of time-dependent, contingent cash flows," *Eur. J. Oper. Res.*, vol. 96, no. 1, pp. 90–96, 1997.

[8] T. Murata and H. Ishibuchi, "MAGA: Multi-objective genetic algorithms," in *Proce. 2nd IEEE Int. Conf. Evolutionary Computing*, Perth, Australia, 1995, pp. 289–294.

[9] S. Obayashi, D. Sasaki, Y. Takeguchi, and N. Hirose, "Multiobjective evolutionary computation for supersonic wing-shape optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 182–187, 2000.

[10] G. Colson and C. De Bruyn, "Models and methods in multiple objectives decision making," *Math. Comput. Mod.*, vol. 12, no. 10/11, pp. 1201–1211, 1989.

[11] J. M. Martel, "Aggregating preferences: Utility function and outranking approaches," in *Multicriteria Analysis: Proceedings of the XIth International Conference on MCDM*. Coimbra, Portugal: Springer, Aug. 1995, pp. 76–84.

[12] R. T. Clemen, *Making Hard Decisions: An Introduction to Decision Analysis*, 2nd ed. Belmont, CA: Duxbury, 1996.

[13] G. E. G. Beroggi and W. A. Wallace, "The effect of reasoning logics on real-time decision making," *IEEE Trans. Syst., Man, Cybern. A*, vol. 27, pp. 743–749, Nov. 1997.

[14] E. G. Beroggi and W. A. Wallace, "Operational risk management: A new paradigm for decision making," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, pp. 1450–1457, Oct. 1994.

[15] S. M. Baas and H. Kwakernaak, "Rating and ranking of multiple-aspect alternatives using fuzzy sets," *Automatica*, vol. 13, pp. 47–58, 1977.

[16] S. R. Watson, J. J. Weiss, and M. L. Donnell, "Fuzzy decision analysis," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 1–9, 1979.

[17] T. L. Saaty, "Exploring the interface between hierarchies, multiple objectives, and fuzzy sets," *Fuzzy Sets Syst.*, vol. 1, no. 1, pp. 57–68, 1978.

[18] C. H. Cheng and D. L. Mon, "Evaluating weapon system by analytical hierarchy process based on fuzzy scales," *Fuzzy Sets Syst.*, vol. 63, no. 1, pp. 1–10, 1994.

[19] B. K. Mohanty and N. Singh, "Fuzzy relational equations in the analytical hierarchical process," *Fuzzy Sets Syst.*, vol. 63, no. 1, pp. 11–19, 1994.

[20] J. Efstathiou and V. Rajkovič, "Multiattribute decisionmaking using a fuzzy heuristic approach," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 6, pp. 326–333, 1979.

[21] M. Sugeno and T. Takagi, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 1, pp. 116–132, 1985.

[22] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning, part I," *Inform. Sci.*, vol. 8, pp. 199–249, 1975.

[23] ——, "The concept of a linguistic variable and its application to approximate reasoning, part II," *Inform. Sci.*, vol. 8, pp. 301–357 43–80, 1975.

[24] ——, "The concept of a linguistic variable and its application to approximate reasoning, part III," *Inform. Sci.*, vol. 9, pp. 43–80, 1975.

[25] R. H. Kewley, "Automated tactical course of action development: a computational intelligence approach," United States Military Academy Operations Research Center, West Point, NY, Tech. Rep., Mar. 2000.

[26] C. D. Pedgen, R. E. Shannon, and R. P. Sadowski, *Introduction to Simulation Using SIMAN*. New York: McGraw-Hill, 1995.

[27] J. G. Ecker and M. Kupferschmid, *Introduction to Operations Research*. New York: Wiley, 1982.

[28] F. R. Giordano and M. D. Weir, *A First Course in Mathematical Modeling*. Monterey, CA: Brooks/Cole, 1985.

[29] "BEWSS analyst manual—Volume II attrition methodology," U.S. Army Missile Command, Huntsville, AL, Tech. Manual, July 1994.

[30] *Military Operations Research Analyst's Handbook—Volume I: Terrain, Unit Movement, and Environment*, W. K. Olson, Ed., Military Operation Research Society, Alexandria, VA, 1994.

[31] S. H. Parry and M. A. Youngren, "High resolution combat models,", Monterey, CA, Course notes for high resolution combat models course at Naval Postgraduate School, May 1997.

[32] *Military Operations Research Analyst's Handbook—Volume II, Area 1: Conventional Weapons Effects (Ground)*, S. H. Parry and M. A. Youngren, Eds., Military Operation Research Society, Alexandria, VA, 1995.

[33] H. Takagi, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation," *Proc. IEEE*, vol. 89, pp. 1275–1296, 2001.

[34] D. S. Alberts, J. J. Garstka, and F. P. Stein, *Network Centric Warfare: Developing and Leveraging Information Superiority*, 2nd ed. Vienna, VA: CCRP Publ. Ser., 1999.



**Robert H. Kewley** received the B.S. degree in mathematics and commission in the United States Army from the United States Military Academy, West Point, NY, in 1988. He received the M.S. degree in industrial and managerial engineering and the Ph.D. degree in decision science and engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1998 and 2001, respectively.

After serving two tours of duty with active Army units, Major Kewley performed the research presented in this paper while on the faculty of the United States Military Academy Department of Systems Engineering, West Point. He currently serves as a theater campaign analyst at the Center for Army Analysis, Fort Belvoir, VA. His research interests include agent-based modeling, computational intelligence, combat modeling, and decision support systems.

**Mark J. Embrechts** (S'74–M'81) received the M.S. degree in electrical engineering from the University of Leuven, Belgium, and the M.S. and Ph.D. degrees in nuclear engineering from Virginia Polytechnic Institute, Blacksburg, in 1976, 1977, and 1981, respectively.

After working as a postdoctoral fellow at Los Alamos National Laboratory as a student and Postdoctoral Staff Member (1980–1983), he joined the department of Nuclear Engineering at Rensselaer Polytechnic Institute (RPI), Troy, NY, in 1983. He is currently an Associate Professor, Department of Decision Sciences and Engineering Systems, and on the Faculty of the Information Technology Program at RPI. He has published more than 150 conference and journal papers and coauthored *Exchange Rate Theory* (Oxford, U.K.: Basil Blackwell, 1993). His current areas of interest relate to neural networks, evolutionary computing, fuzzy logic, data mining and applications of soft computing to data mining, biotechnology, and molecular chemistry.