THE INTERNATIONAL
C2 JOURNAL

VOLUME 2, NUMBER 2, 2008

SPECIAL ISSUE

Representing Human Decision Making in Constructive Simulations for Analysis

GUEST EDITOR

James Moffat
Dstl

A Multiagent System for Tactical Control of Automated Forces

Robert H. Kewley, Jr.
Mark J. Embrechts

CCRP

THE INTERNATIONAL C2 JOURNAL

About the Journal

The International C2 Journal was created in 2006 at the urging of an international group of command and control professionals including individuals from academia, industry, government, and the military. The Command and Control Research Program (CCRP, of the U.S. Office of the Assistant Secretary of Defense for Networks and Information Integration, or OASD-NII) responded to this need by bringing together interested professionals to shape the purpose and guide the execution of such a journal. Today, the Journal is overseen by an Editorial Board comprising representatives from many nations.

For more information

Visit us online at: www.dodccrp.org
Contact our staff at: publications@dodccrp.org

A Multiagent System for Tactical Control of Automated Forces

Robert H. Kewley, Jr. (USMA Department of Systems Engineering, USA)

Mark J. Embrechts (Rensselaer Polytechnic Institute, USA)

Abstract

This paper proposes a multiagent architecture for the fully autonomous hierarchical and adaptive control of tactical forces on a simulated battlefield and demonstrates the efficacy of this control scheme as compared to human subjects. The multiagent architecture includes a planning agent for course of action development, a route selection agent and position agent for unit reactions to real-time command and control information, and a unit and vehicle agent to adjust formations, speed, and orientations according to local conditions. The agents were terrain aware and maintained a situation awareness picture from reports received across the network during the battle. They key finding in this paper is the validation of this multiagent approach by comparing its mission effectiveness to that of human planners. In a tactical planning experiment, the multiagent system showed dramatic improvement across all mission effectiveness measures when compared to forces that followed plans prepared by human subjects. Empowered by further development of agent architectures that make tactical decisions with this level of performance, constructive simulations will be more effective in several application areas. They will be able to better estimate the mission-level effects of different information pictures, to provide intelligent and reactive automated forces to replace human players, and to provide decision support during the planning and execution phases of combat operations.

Introduction

With the rising complexity of the modern battlefield, credible analysis of the effects of command and control (C2) systems becomes increasingly difficult. The problem with analyzing these systems is that none of them directly defeat the enemy. Instead, they influence the decision-making processes by which commanders at all levels direct actions to defeat the enemy. Traditional analysis tools have difficulty modeling the human decision, so analysts have difficulty analyzing the effects of C2 variables such as alternative network structures or varying information quality. One modeling approach is to use human experts for command and control of forces in a constructive simulation. However, man-in-the-loop experiments require specialized hardware and software along with a cast of trained decision makers. Replication is time consuming. There is a modeling gap within constructive simulations where they cannot model the knowledge, judgment, and decisions that translate battlefield information into combat action (Kewley and Larimer, 2003, 10-14). An alternative approach is to develop intelligent software algorithms, or agents, to replicate human decision-making within constructive simulations. When these agents are presented with information, the resulting decision and combat action will reflect the quality of that information via mission level measures of effectiveness.

The experiment described in this paper models tactical commanders as agents in a hierarchically organized multiagent system. This multiagent system draws its inspiration from existing research into adaptive tactical command and control, control of complex systems, and agent based models of land combat. The multiagent architecture assigns tactical control tasks to five different agents. The planning agent uses a co-evolutionary genetic algorithm to assign missions and general locations to individual units. Each unit has a route selection agent that uses a genetic algorithm to develop routes for tactical movement. A positioning agent uses a random search technique to adjust unit positions in order to best accomplish assigned missions. The unit agent employs a fuzzy rule set and a fuzzy inference system to control unit speeds and formations. The vehicle agent uses a fuzzy inference system to adjust individual vehicle speeds and orientations within the formation. In a tactical command and control experiment, forces controlled by the multiagent system performed better than forces that adhered to manually developed routes and positions entered prior to the battle.

Supporting Research

Researchers studying complexity in organizations agree that business organizations in today's complex world cannot survive by a rigid set of bureaucratic controls. They should instead form smaller flexible, interacting, and adaptive "cells" or "patches." Organizations should manage the goals and objectives of these cells so that they are in line with organizational goals (Lissack 1999, 110-124) (Coleman 1999, 33-48). Margaret Wheatley argues that self organization produces results by allowing adaptability and inspiring creativity in the face of a rapidly changing world (Wheatley, 2006). A number of researchers investigating tactical command and control call for increased flexibility and adaptability on the battlefield. The notion of an "edge organization" offers a way to increase a unit's agility and effectiveness by moving decision-making and execution capabilities away from centralized control centers and to the "edge" of the organization, where it interfaces with its environment (Alberts and Hayes 2003, 4-7). This type of an organization is more likely to thrive in a complex environment characterized by nonlinear and collectivist dynamics, opportunities for self-organization, and unpredictable cascading effects (Atkinson and Moffat 2005, 19-55). Implementation of this vision requires an information-age approach to command and control in which information is distributed broadly, interactions are unconstrained, and decision rights are pushed to the lowest level (Alberts and Hayes 2006, 73-113). Writers from within the military ranks share in this call for agility and adaptability. The military decision making process used in tactical planning doctrine focuses too much effort on the selection of a course of action prior to the battle and fails to prepare decision makers for adaptive and flexible decisions during the fight (Shoffner 2000, 37-39). The US Army Training and Doctrine Command Analysis Center at White Sands Missile Range conducted a study which found that dominant maneuver, "the multidimensional application of information, engagement, and mobility capabilities to position and employ widely dispersed joint air, land, sea and space forces to accomplish the assigned operations tasks," was effective for small tactical forces. This form of maneuver focuses on event-driven actions by which friendly forces react and adapt to the developing situation (Barris, 1999). Understanding these emerging challenges, the US Army has adopted its command and control doctrine to encompass the concept of mission command, where subordinate leaders exercise initiative within the framework of commander's intent, mission orders, and resources (Headquarters Department of the Army, 2003, 1-17 – 1-19). The multiagent system presented in this paper introduces this type of information-enabled adaptability and flexibility into the control of automated forces.

A multiagent system consists of a network of problem solving agents that interact to solve problems that are beyond the capabilities and knowledge of any one individual agent. Research shows that these systems have several general characteristics (Sycara 1998, 79-92). Each agent has a limited viewpoint and incomplete information about the problem. Data are decentralized, and computation is asynchronous. The agents in the system are often heterogeneous and can be planning agents, which satisfy goals and objectives, or reactive agents, which have no internal representation of their environments and simply act upon stimuli in their present state. The designer of a multiagent system must consider, in addition to the design of individual agents, assignment of tasks to those agents, agent organization, and communication between agents. Researchers in several domains have applied multiagent systems to the control of complex systems. Liu and Hsu used a hierarchical organization of fuzzy control agents to manage traffic at stoplights (Liu and Hsu, 2007, 4961-4966). Feng et.al. used a hierarchical organization of cooperative agents to synchronize a manufacturing system with and enterprise resource planning system (Feng et. al, 2007, 1047-1052). The introduction of radio frequency identification (RFID) provides a wealth of information to enable agent-based control of industrial systems (Bratukhin and Treytl 2006, 1199-1205).

Within the military domain, multiagent systems have also been applied to the control of simulated forces. Researchers at Georgia Institute of Technology and the Air Force Institute of Technology applied genetic algorithms to the problem of search and attack for swarms of unmanned aerial vehicles. This work resulted in the evolution of self-organizing behaviors for the swarm (Lamont and Price 2006, 1307-1315). Stensrud et. al. used genetic algorithms to generate novel and effective opposing force tactics for a convoy training simulation (Stensrud et. al., 2007). Ludwig and Farley chose a supervised learning approach called hierarchical dynamic scripting for control of forces in a computer game (Ludwig and Farley, 2008). Ekanayake and Pathirana used a multi-agent control system to control the pattern geometry of cluster bombs as they impact the earth (Ekanayake and Pathirana, 2007, 471-476). Within the maritime domain, Beaumont and Chaib-draa (Beaumont and Chaib-draa, 2007, 373-384) and Randall (Randall, 2008) each used multi-agent systems for planning and tactical tasks in maritime simulations. The work presented in this paper also uses a multiagent system to control simulated forces, and it extends existing research to address the use of agents not only during simulation execution, but also to develop the tactical plans formulated prior to the battle. The use of planning agents was previously published by the authors (Kewley and Embrechts, 2001, 161-171). Further research introduced real-time decision agents and showed their applicability to the study of command and control systems by assessing the value of information (Kewley and Larimer, 2003, 10-14, 25-26) and the effectiveness of centralized or decentralized command and control (Kewley, 2004, 926-930). This paper further extends that work by combining the planning and real-time decision agents in a multi-agent hierarchy. It then compares the performance of these agents to human planners in a simulation experiment.

Command and Control Agent Architecture

The aforementioned supporting research efforts are valuable sources of insight for the command and control agent architecture developed in this paper. This multiagent system is appropriate for control on a complex and unpredictable battlefield. Inspired by the "mission type orders" that power down authority and control on the tactical battlefield, this architecture decomposes forces into a hierarchy and assigns decision agents to different levels in that hierarchy. The high-level planning agent will not attempt to explicitly control all subordinate units. The planning agent's pre-battle information picture is more coarse than that of real-time agents. It is terrain aware, but it cannot consider all possible locations for subordinate elements. Its enemy picture considers the composition of enemy forces and their general location on the map. It will give general locations, goals, and flexibility distances to subordinate units. These subordinate units will consider with greater detail and precision the local terrain, weapons, and enemy forces in order to adapt their behavior (specific locations and routes) to maximize attainment of their assigned goals. The real-time agents search the terrain information in detail for positions of advantage while considering up to date locations of reported enemy units.

The experiment in this paper tests the capability of a networked and cooperative group of command and control agents to improve tactical planning and execution. This network consists of a planning agent to develop the general scheme of maneuver used during the battle. Agent coordination and collaboration are relatively simple. The planning agent serves as the coordinator in a centralized scheme (Lin et. al., 2004, 631). The scheme of maneuver consists of unit positions, missions, and movement techniques along with a scheme for fire support. The planning agent also gives each unit a flexibility distance that tells each subordinate unit agent how much freedom it has to change unit positions. The route selection agent used by each unit develops a route to get to the assigned location. The positioning agent adjusts assigned positions within the constraints of the assigned flexibility distance. These real-time agents do not communicate directly with each other. Instead, each agent reads its situation awareness picture from a centrally managed common operational picture, enabling self-organization and some degree of swarming behavior as they individually pursue the missions assigned by the planning agent within the current situational context. The unit agent controls unit speed and formations, and the vehicle agent adjusts individual vehicle speeds, orientations, and locations within those formations. By this architecture, each vehicle in the force has its position affected by up to five different cooperative agents.

The Planning Agent

This agent uses fuzzy genetic decision optimization and co-evolution (Kewley 2001, 161-171) to evolve a set of robust friendly courses of action which may be expected to perform well against a variety of enemy courses of action. Fuzzy-genetic decision optimization (FGDO) solves complex problems which require concurrent optimization of multiple objective criteria. It has three modules. The first module is a co-evolutionary genetic algorithm (Mitchell 1996, 26-27) which varies tactical planning parameters in order to maximize the overall performance. The second module is a constructive combat simulation model which evaluates the proposed tactical plans against a set of enemy plans. The third module is the fuzzy preference module. A graphical user interface allows the user to select the important combat outcomes and their order of satisfaction. These selections define a fuzzy inference system which aggregates the outputs of the simulated battle into one overall fitness value, which is then returned to the genetic algorithm.

Several characteristics of the planning problem make a genetic algorithm an appropriate choice for a heuristic solution technique. Silver gives an overview of heuristic solution methods (Silver, 2002). The planning solution space is very large and complex, and metaheuristics are concerned with the avoidance of local minima. The challenge of metaheuristics is to appropriately represent the solution structure and constraints. One challenge of metaheuristics such as tabu search and simulated annealing is the appropriate definition of a neighborhood within the tactical planning domain. A genetic algorithm is a population-based heuristic where a group of successive populations is evolved over a number of generations using the genetic operations of selection (the selection of "parents" to evolve new solutions), crossover (the generation of new solutions by combining information from the parents), and mutation (the random adaptation of elements of a new solution) (Goldberg, 1989). A genetic algorithm was chosen because the genome representation was seen as more natural and tractable than the neighborhood representation. In addition, the population-based search naturally enables parallel processing and co-evolution of competing friendly and enemy plans.

The chromosome representation for the planning agent builds a container chromosome from a complex set of sub-chromosomes. The planning agent's chromosome contains five sub-chromosomes, a fires sub-chromosome, a locations sub-chromosome, a flexibility sub-chromosome, a movement technique sub-chromosome, and a mission sub-chromosome. Figure 1 shows a course of action chromosome for a small force. This chromosome will be referred to as Plan A. The first sub-chromosome is the fires sub-chromosome. It contains a randomly generated priority of fires object for each contingency in the battle. Plan A has two phases and three contingencies. Units execute Phase 1, Contingency 1 first. Then, depending upon enemy action, the units will execute either Contingency 1 or Contingency 2 during Phase 2. The remaining four sub-chromosomes each contain, in a linear arrangement, a single object to be used by each unit during each battle contingency. Plan A gives orders to 4 units. Since there are 3 contingencies, these four remaining sub-chromosomes each have 12 objects arranged linearly. The location object is an *x,y* coordinate pair which tells the unit where to go. The flexibility object is a distance on the uniform interval between 0 and *MAXDISTANCE*, a parameter in the algorithm. This distance tells the unit how far it is allowed to stray from the location given by the location chromosome. The movement technique object gives the unit a fuzzy ordinal preference system (Kewley 2000, 72-96) which defines the priority goals a unit should seek during movement. An artillery unit with limited ability to survive in a direct fire fight may execute an infiltrate movement technique, which prioritizes friendly survival and avoidance of enemy contact. On the other hand, a heavily armored tank platoon may execute a movement to contact movement technique, which prioritizes seeking and engaging enemy forces. The missions sub-chromosome is similar to the movement technique sub-chromosome. However, it gives each unit a fuzzy ordinal preference system which defines priority goals a unit should seek when it arrives at its destination, as opposed to during movement.

Figure 1's Plan A shows an example chromosome representation of a tactical plan. During Phase 2, Contingency 2, the force places indirect fires in direct support of Unit 3, and they fire destroy missions against enemy forces. Unit 3 moves to grid coordinate 502, 113. It will use the Movement to Contact movement technique during movement, seeking contact with enemy forces. When it arrives at its location, it will execute an Attack by Fire mission. This mission will cause it to search a circle centered at coordinates 502, 113 with radius 9326 meters (the flexibility distance given by the flexibility chromosome) for better positions that allow it to attack and destroy enemy forces with direct fires.

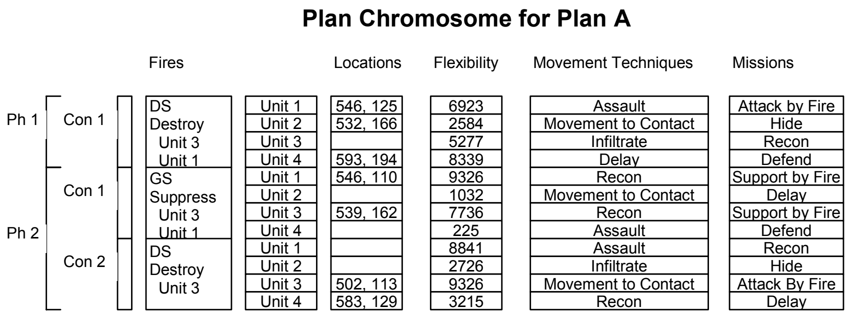

Figure 1. The plan chromosome for a tactical course of action contains a fires sub-chromosome which assigns a fires object to each contingency. It also contains sub-chromosomes which assign each unit, for each battle contingency, a destination, a flexibility distance, a movement technique, and a mission.

When a course of action chromosome performs genetic operators, it simply performs those operations on each of its sub-chromosomes. The probability of crossover applies only to the planning agent's container chromosome, not the sub-chromosomes. If a uniform random draw on the [0,1] interval is less than the probability of crossover, the container chromosome will execute crossover for all of its sub-chromosomes. When it performs crossover, it successively executes crossover for each of the five sub-chromosomes. Each sub-chromosome performs single point crossover. For example, the offsprings' fires sub-chromosomes are formed by crossing the parents' fires sub-chromosomes. The offsprings' location sub-chromosomes are formed by crossing the parents' location sub-chromosomes, and so forth. It executes initialization and mutation in a similar fashion.

The Route Selection Agent

A course of action developed by the planning agent gives each unit a location for each phase of the battle, but instead of giving a route, it gives a movement technique—movement to contact, assault, recon, or infiltrate. These movement techniques define the objectives a unit should seek as it develops its route. For example, the assault movement technique seeks speed and enemy destruction, while the recon technique seeks enemy acquisition as opposed to destruction, with little emphasis on speed.

A genetic algorithm evolves successive populations of routes. The route planning genetic algorithm used by the route selection agent was adapted from the work of Hocaoglu and Sanderson (Hocaoglu and Sanderson 1997, 81-104) and Xiao et. al. (Xiao et al. 1997, 18-28). The genetic algorithm represents a route as a variable length list of points through which a unit must pass while moving from its current destination to its final destination. Each point is an x,y coordinate pair in the continuous domain. In this representation, a unit's route may have from 0 (move straight to destination) to any number of waypoints evolved by the algorithm. The algorithm adds a point to a route by inserting a knot point along one of the route legs. The point insertion algorithm first chooses one of the route legs at random. In figure 2, the algorithm has randomly selected leg 2 from the three possible route legs. It then randomly chooses an interim point along that leg at which it will insert the knot point. It then must determine a knot distance. The knot distance is a random value drawn from a normal distribution with a mean of zero and a standard deviation of 1/2 the length of leg 2. If the knot distance is positive, then a knot point is inserted perpendicularly to one side of the leg. If the knot distance is negative, the know point is inserted perpendicularly to the other side of the leg. In figure 2, the knot point has been inserted perpendicular to leg 2 and knot distance from the interim point. The knot point is inserted in the route between points 1 and 2 to give a new route shown by the dotted line.

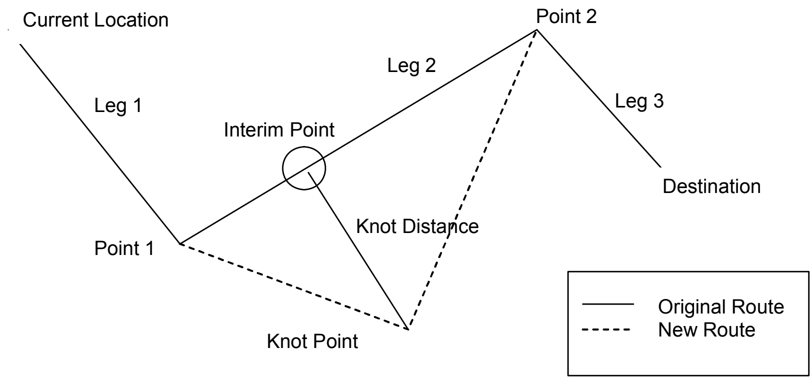

Figure 2. The process of adding a random point to a route. The knot point is inserted between points 1 and 2 to generate the new route shown by the dotted line.

Upon initialization, the route genome draws the initial number of points from a truncated exponential distribution with a mean of 1. It is most likely to have 0 points, next most likely to have 1 point, and so forth. For all initial points, the genome adds a random point to the route using the algorithm shown in figure 2. Upon crossover, two routes will perform single point crossover in order to form two offspring for the next generation. Upon mutation, the route genome will successively test each of its points for mutation. If a mutation trial is successful, one of three things will happen to the point, each with equal probability. The genome may remove the mutated point, the genome may add a random point to the route, or the genome may remove the mutated point *and* add a random point to the route.

A route evaluation model evaluates each route in the population given the current enemy situation, competing weapons capabilities, and surrounding terrain. It discretely steps along the route at given intervals and develops estimates for expected percentage of enemy units seen, expected number of enemy vehicles destroyed, expected friendly vehicles destroyed, and movement time. Based on its move-

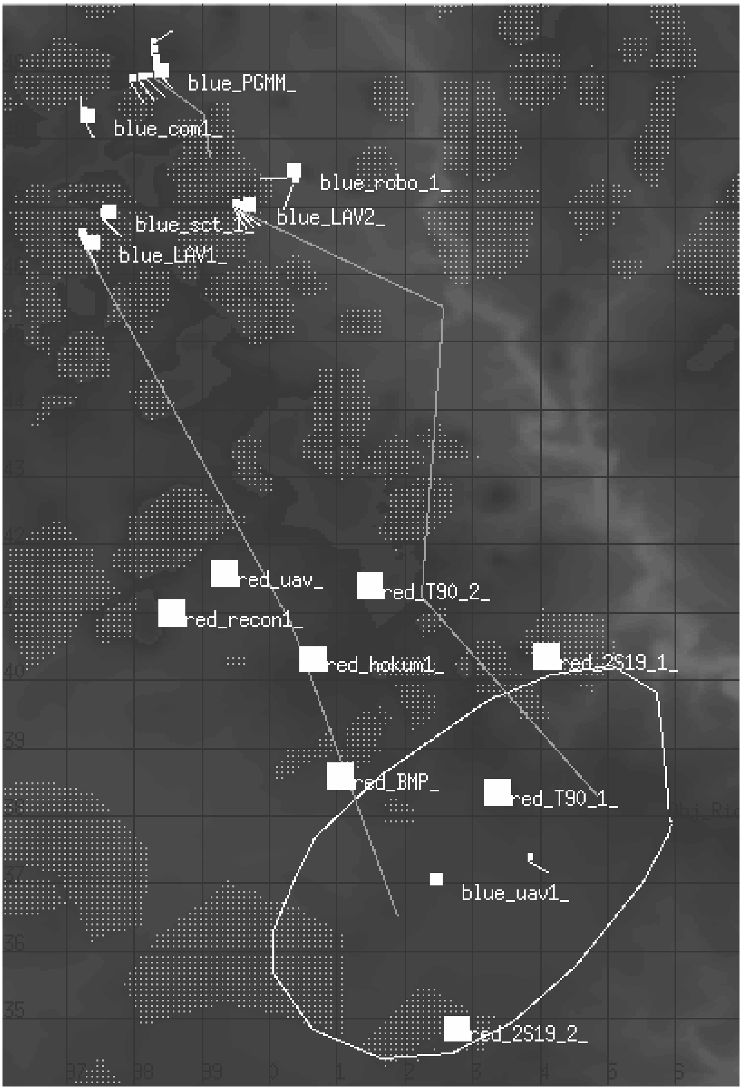

Figure 3. Illustration of the route planning agent in action. Darker areas represent high ground, white dots represent forested areas, and large white squares represent reported enemy positions. The agent applied the *movement to contact* movement technique to move one advancing friendly platoon (blue_LAV1) down the left side of the sector (route shown as left white line) through reported enemy units. It applied the *infiltrate* movement technique to move the rightmost advancing platoon (blue_LAV2) behind a ridgeline and through trees in order to avoid enemy contact.

ment technique, the unit uses a fuzzy ordinal preference system which aggregates these route statistics into an overall route preference, used as a fitness value by the genetic algorithm. The route agent uses different ordinal preference schemes for each of the agent's five different movement techniques. As a unit moves, the route selection agent re-evaluates its current route at fixed time intervals. If the genetic algorithm finds a better route, it will replace the current route. This allows a unit to adjust to the changing enemy situation as it moves. Figure 3 shows the route planning agent in action.

The Positioning Agent

A unit may adjust not only its route, but also its final destination, bounded by the flexibility distance given by the planning agent. In a manner similar to the one used by the route selection agent to evaluate routes for a given movement technique, the positioning agent evaluates candidate positions for a mission given by the planning agent—attack by fire, support by fire, defend, recon, delay, or hide. A mission is a set of objectives sought by the unit when it reaches its destination. For example, both the attack by fire and support by fire missions seek enemy destruction. However, the support by fire mission also places greater emphasis on staying close to the location given by the planning agent, so that it does not get too far away from the unit whose movement it supports. Since the search for a single position is a much simpler search task than the search for a route, the positioning agent uses a simple uniform random search as opposed to a genetic algorithm. At constant intervals, the agent uniformly selects a set of random points from within a circle. The center and radius of the circle are determined by the location and flexibility distance given to the unit by the planning agent. A position evaluation model takes into account the surrounding terrain and enemy forces to give estimates for percentage of enemy units acquired, expected number of enemy vehicles destroyed, and expected number of friendly vehicles destroyed from each selected location. It also considers the distance from its assigned location. Based on its mission, the agent aggregates these criteria using a fuzzy ordinal preference system to get an overall score for each location. The agent will reposition the unit to the best location found.

The Unit Agent

This agent determines the speed, formation, and spacing of a unit based upon whether the unit is suppressed, the level of enemy danger, terrain restriction, and the direction to the most dangerous enemy. The rules are implemented using linguistic variables (Zadeh, 1975, 199-249) and a Mamdani fuzzy inference system (Mamdani, 1975, 1-15). This technique allows mathematical interpretation and execution of rules which also make sense to a human decision maker. The rules strongly suggest consequents (the THEN clause) for strong matches to their antecedent (the IF clause). They weakly suggest the actions of their consequents if there is a weak match to the antecedent. This structure allows a fairly concise and readable rule set to determine a potentially large set of potential outcomes. The rules below are two of those used by the unit agent:

> *[IF Terrain_Restriction is not very_high THEN Spacing is med AND Column is very_low AND Wedge is high AND Line is med AND Speed is med]*
>
> *[IF Suppression is greater_than_or_equal_to high OR Enemy_Danger is greater_than_or_equal_to high THEN Spacing is very_high AND Column is very_low AND Wedge is high AND Line is med AND Speed is very_high]*

The Vehicle Agent

This agent is similar to the unit agent, but it controls individual vehicles as opposed to units. This agent considers the level of enemy danger, whether or not it is firing, the distance the vehicle is from its position in the formation, and the direction to the most dangerous enemy to adjust an individual vehicle's orientation and speed. For example, a vehicle will tend to slow down to fire and orient toward the most dangerous enemy. However, as it gets further from its position in the formation, it will increase speed and maintain orientation in order to get its position back. The rules below are two of those used by the vehicle agent:

> *[IF Enemy_Danger is greater_than_or_equal_to med_high AND Firing is very_low AND Dist_To_Goal is greater_than_or_equal_to high THEN Direction is straight AND Speed is very_high]*
>
> *[IF Enemy_Danger is greater_than_or_equal_to med_high AND Firing is very_high AND Dist_To_Goal is less_than high AND Enemy_Flank_Dir is greater_than_or_equal_to right THEN Direction is right AND Speed is very_low]*

The first rule causes a vehicle in imminent danger but not firing at the enemy to move quickly along its line of march to get out of the danger area. The second rule causes a vehicle in danger from the right and returning fire to slow down and orient toward the enemy, exposing frontal armor as opposed to flank armor.

Collaborative Command and Control Agent Experiment

One possible application of agent technology is for the control of automated forces during training or analysis simulations. Current simulations used in the Army training and analysis communities require the scenario developer to generate and input the tactical plan used by the enemy. The automated forces react to local contact with battle drills, but they do not use their collective situation awareness to adjust to enemy action. The agent-based architecture used in this experiment provides this capability. This experiment hypothesizes that within a constructive simulation, units controlled by conventional, manually developed, routes and positions will have the same level of success in tactical operations as units controlled by a multiagent system. The alternative hypothesis is that the performance of units controlled by the multiagent system and the performance of manually controlled units are not equal. This experiment defines performance (fitness) as the ability of the course of action to satisfy the prioritized goals given by the higher commander in the form of a fuzzy ordinal preference system.

Scenario

The planning task was to generate an offensive course 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 Gold in the north (see figure 4), were given an immediate mission to continue the attack 15 kilometers 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.

Experimental Design

The subjects for this experiment developed the manual tactical plans used for comparison against the multiagent system. They were junior and senior cadets in a Decision Support Systems class. They are certainly not experts when it comes to tactical planning. The seniors had a course in combined arms operations, so they have at least been exposed to the doctrinal principles behind tactical planning for combined arms forces, but the juniors have only experienced planning at the infantry platoon level. If the system can be shown to improve upon their performance, one can conclude that this multiagent system outperforms inexperienced planners. This is a strong conclusion, given the infancy of this technology.

The experiment used a 2 x 2 factorial design. The first factor was the source of the tactical plan for the overall force, manual or automated (using the planning agent). The second factor was the ability of forces to react to the developing situation on the battlefield. The baseline forces strictly adhered to the routes and positions given in the tactical plan, and the intelligent forces used four different intelligent agents (route selection agent, positioning agent, unit agent, and vehicle agent) to adapt unit and vehicle locations on the battlefield.

Table 1. Experimental design for collaborative command and control agent experiment.

Group	Size	Plan Development	Routes and Positions	Comment
1	12	Automated	Static	Automated plans with static execution.
2	12	Automated	Adaptive	Automated plans with execution by collaborative route, position, unit, and vehicle agents.
3	12	Manual	Static	Automated plans with static execution
4	12	Manual	Adaptive	Manual plans with execution by collaborative route, position, unit, and vehicle agents.

Genetic Evolution of Automated Plans

The experiment's first step was to allow the planning agent to evolve a set of distinctly different, high performing, and robust friendly plans using co-evolution against a combination of two fixed enemy plans, developed by human planners, and four parallel populations of adapting enemy plans, which co-evolved in an effort to defeat the friendly plans. During co-evolution, the automated forces in the combat simulation used the entire network of intelligent agents to control movements on the battlefield. This use of intelligent forces greatly increased the computation time required to evaluate a course of action.

An additional complication is the increased dimensionality introduced by the complex course of action representation used in this problem. The plan chromosome used in the experiment scenario had two phases, one phase per contingency, and 7 units, giving a total of 14 different orders and two priority of fires objects. The complexity of the fires object is calculated first. There are two types of support and 3 mission types. The priority of fires list was limited to a maximum of four units. The complexity c_f of the fires object is given by:

$$c_f = 2 \bullet 3 \bullet \left[\binom{7}{4} + \binom{7}{3} + \binom{7}{2} + \binom{7}{1} \right] = 588$$

The orders object is even more complex. The friendly area of operations, discretized into a grid of points separated by 300 meters, has 1253 possible locations to which a unit could move in one phase. Although the flexibility distance is continuous, one may define "different" flexibility distances as those that are different by 300 meters. Since flexibility ranges from 0 to 10000, there are 33 different flexibility distances a unit could have. There are 5 different movement techniques and 6 different missions. The complexity c_o of an order is given by:

$$c_o = 1253 \bullet 33 \bullet 5 \bullet 6 = 1240470$$

Finally, the complexity c_p of a tactical plan is all possible combinations of 2 fires objects and 14 orders:

$$c_p = c_f^2 \bullet c_o^{14} = 7.06256 \bullet 10^{90}$$

This is clearly a difficult search problem, even with this small tactical plan. An added problem is that plan evaluation takes 25-40 seconds to give a stochastic result. Despite these formidable difficulties, it is possible for a genetic algorithm to search this complex space to give acceptable answers. The genetic algorithm used the parameters shown in table 2. These parameters were chosen, based on experience in previous experiments, considering trade-offs between speed and performance, and between global exploration and local exploitation.

Table 2. Co-evolutionary genetic algorithm parameters.

Parameter	Value	Comment
Number of Populations	4	4 populations of friendly courses of action evolved independently
Migration Rate	0	No information shared between the 4 populations
Population Size	20	20 members of each population
Replacement Rate	60%	Replace the worst 60% of each population for each generation
Mutation Rate	0.10	Perform mutation 10% of the time
Crossover Rate	0.90	Perform crossover 90% of the time
Number of generations	60	Evolve for 60 generations

The algorithm used a tournament selector with 3 members in the tournament. In order to select parents to perform crossover and mutation, the tournament selector randomly selected 3 members of the previous population and compared fitness values. The one with the highest value was retained as a parent. It replaced all members and repeated to select the second parent.

Using these parameters, the algorithm generated 4800 different friendly and enemy courses of action for evaluation. Each friendly course of action fought against two fixed enemy courses of action and 4 evolving enemy courses of action. This scheme required resolution of 28800 battles at 25-40 seconds each. A parallel implementation of the genetic algorithm greatly reduced the time needed for evolution. Seventeen different networked workstations established themselves as battle servers to evaluate individual courses of action, and one client workstation managed the overall evolution. For each generation, the client workstation sent scenario information to each server. It then used genetic operators to evolve the friendly and enemy courses of action for the current generation. It sent friendly/enemy course of action pairs over network sockets to the battle servers for evaluation and waited for the servers to return fitness values for both the friendly and enemy courses of action. When all evaluations were done, the client workstation used the fitness information to evolve the next generation and repeated the process. This evolution took 16.5 hours to complete. It is interesting to note that the tactics evolved by this scheme make military

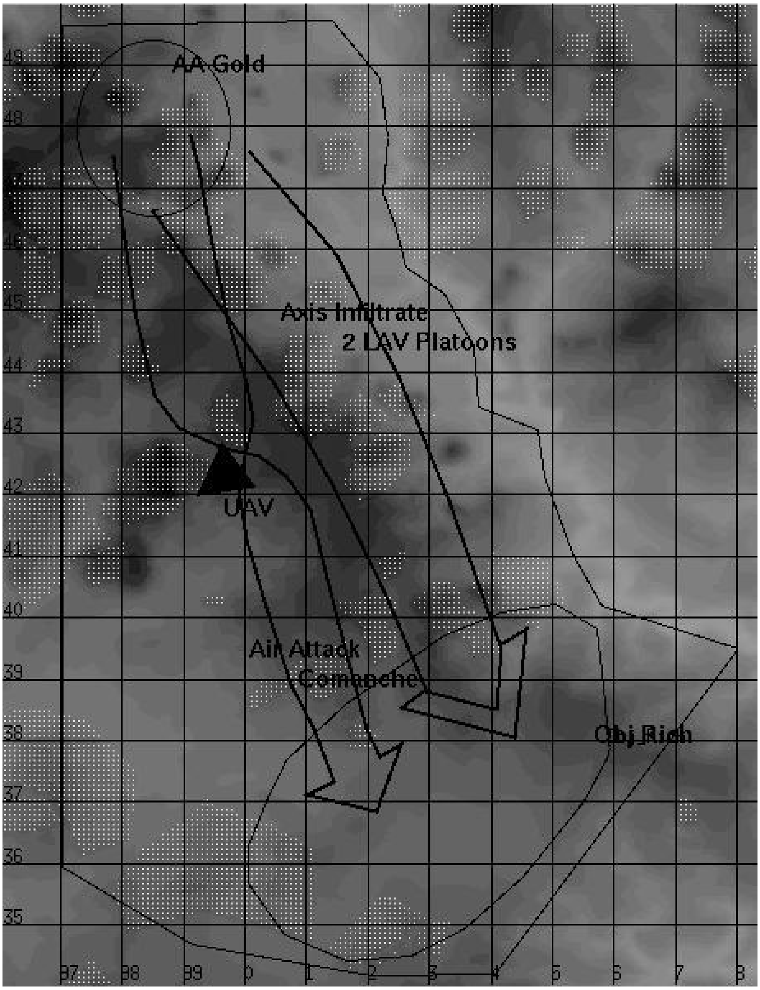

Figure 4. This sample automated course of action sent the unmanned aerial vehicle (UAV) to recon at the observation post in mid-sector and sent the attack helicopters on a movement to contact along axis Air Attack to destroy enemy forces in sector and on the objective. Indirect fires also aid in destruction of forces found by the UAV and attack helicopters. Two LAV platoons infiltrate along Axis Infiltrate avoiding enemy contact to get to Objective Rich, where they each execute defend missions. In phase 2, the scout section and robot move up to the objective area. This plan makes good use of the killing abilities of the attack helicopters and indirect fires while placing the maneuver forces at minimum risk until they get to the objective.

sense. The example course of action developed by the planning agent (Figure 4) employs friendly strengths to find and destroy enemy forces while minimizing exposure to enemy direct and indirect fire systems.

Experimental Results

Analysis of the experimental results shows a dramatic improvement in force performance using the command and control agents. In fact, group two (see table 3), which made use of all agents, showed, on average, a 40% reduction in enemy forces on the objective accompanied by a 120% increase in friendly forces on the objective when compared to group 3, which used no agents. Group 2 had similar improvements in total losses for friendly and enemy forces. This resulted in a 154% improvement in overall fitness.

Table 3. Average performance of forces in each experimental group.

Plan	Data	Real Time Agents			
		no		yes	
auto	% Red Destroyed	Group 1	68.40	Group 2	74.31
	% Blue Destroyed		24.83		18.92
	Red in Obj_Rich		7.83		6.63
	Blue in Obj_Rich		8.38		8.71
	Fitness		0.42		0.49
manual	% Red Destroyed	Group 3	55.44	Group 4	64.93
	% Blue Destroyed		38.54		41.49
	Red in Obj_Rich		10.92		7.92
	Blue in Obj_Rich		3.92		3.58
	Fitness		0.19		0.24

Group 2, which made use of all intelligent agents, showed a marked performance increase over group 3, which used no agents.

Visualization (Figure 5) and statistical analysis of variance (Table 4) of the overall performance (fitness) of the units in each experimental group show that performance using the planning agent is greatly improved and statistically significant at a high level of confidence. The effect of real time agents yields an average improvement in fitness, but the improvement is not statistically significant. However, real-time agents are best suited for looking for positions of advantage in killing the enemy. Analysis of variance for the effect of real time agents on enemy losses (Table 5) shows a 12% increase in enemy losses when real time agents are used, and this effect is statistically significant with a *p* value of 0.022. In summary, the architecture of intelligent command and control agents produced a significant improvement in combat performance when compared to forces that followed a manually developed plan, failing to adapt to the changing battlefield situation.

Table 4. Analysis of variance for overall performance (fitness) of a combat action.

Analysis Of Variance: Effects Tests (Fitness)					
Source	**Sum-of-Squares**	**df**	**Mean-Square**	**F-Ratio**	**P-Value**
Plan	0.70	1	0.70	22.70	<.0001
Real Time Agents	0.04	1	0.04	1.23	0.27262
Plan*Real Time Agent	0.00	1	0.00	0.04	0.84434
Error	1.35	44	0.03		
Total	2.09	47			

The use of the planning agent (Plan) is statistically significant while the use of the route, position, unit, and vehicle agents (Real Time Agents) is not.

Table 5. Analysis of variance for the percentage of enemy forces destroyed.

Analysis of Variance: Effects Tests (% Red Destroyed)					
Source	**Sum-of-Squares**	**df**	**Mean-Square**	**F-Ratio**	**P-Value**
Plan	1496.67	1	1496.67	11.81	0.0013
Real Time Agents	710.94	1	710.94	5.61	0.0223
Plan*Real Time Agent	38.63	1	38.63	0.30	0.58364
Error	5574.94	44	126.70		
Total	7821.18	47			

The use of the planning agent (Plan) and the use of the route, position, unit, and vehicle agents (Real Time Agents) are both statistically significant effects.

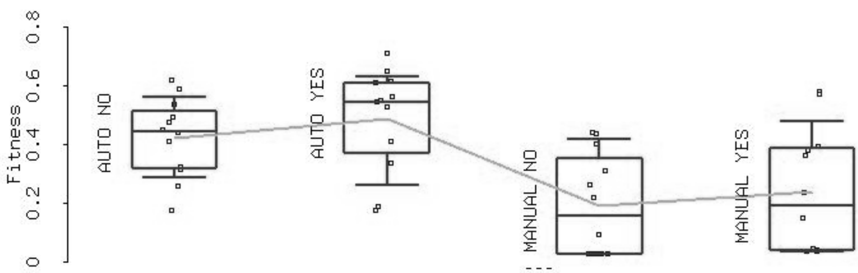

Figure 5. A box and scatter plot of fitness scores for forces in each experimental group. The terms MANUAL and AUTO refer to manual planning or automated planning using the planning agent. The terms YES and NO refer to whether or not the unit was adaptively controlled by the route, position, unit, and vehicle agents during the fight. Means for each group are connected by a solid line. The intelligent agents improve the mean performance of automated forces.

Conclusions and Future Work

This experiment is evidence that a multiagent system composed of computationally intelligent planning and real-time command and control agents can significantly improve mission performance for automated forces. Although, the planning agent requires extensive computation time, processors with ten times the speed of those used in this experiment are currently available. This time can be further reduced by adding up to 120 parallel processors for fitness evaluation, potentially accessed from widely distributed locations in a network centric environment. With the addition of these resources, overall computation time can be reduced to under one hour, but there is still a need for extensive research into understanding, improving, and refining the algorithm to reduce computation time while maintaining performance. Overall, these results provide a convincing argument for continued research and development to improve this multiagent command and control system to a level where it may be implemented within constructive simulations or decision support systems.

The research in this paper leaves many opportunities for future work. Each of the command and control agents developed still has room for improvement. Different search strategies may be applied to the planning agent, route selection agent, and positioning agent. The unit agent and vehicle agent may experiment with different rules sets, or high performance rules could be learned during the simulation. Also, different agent architectures and coordination mechanisms may decompose the tactical command and control tasks in different ways. Finally, research should be done to integrate these agents into existing combat simulations and to address the verification and validation issues that arise with their use. Once this capability is realized, military analysts will be better equipped to analyze the effects of C2 systems on military battles. Decision agents, acting within the simulation as military commanders, will use the information provided by candidate C2 systems to make decisions that direct forces to locations where they can achieve military objectives. Not only will analysts be able to measure the effects of C2 systems on combat outcomes, they may also be able to observe the tactics evolved by the multiagent system in order to gain insights about how to fight on the information enabled battlefield.

References

Alberts, David S., and Richard E. Hayes. 2003. *Power to the Edge.* Washington, D.C: Department of Defense Command and Control Research Program.

Alberts, David S., and Richard E. Hayes. 2006. *Understanding Command and Control.* Washington, D.C.: Department of Defense Command and Control Research Program.

Atkinson, Simon Reay, and James Moffat. 2005. *The Agile Organization.* Washington, D.C: Department of Defense Command and Control Research Program.

Barris, Benjamin J., 1999. *FBCB2 Dominant Maneuver Analysis*. In presentation at 38th Annual US Army Operations Research Symposium, October 1999, at Fort Lee, Virginia.

Beaumont, Patrick and Brahim Chaib-draa, 2007. Multiagent Coordination Techniques for Complex Environments: The Case of a Fleet of Combat Ships. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, pp. 373-385, vol. 37, no. 3, May 2007.

Bratukhin, Aleksey, and Albert Treytl. 2006. Applicability of RFID and Agent-Based Control for Production Identification in Distributed Production. In *Proceedings of IEEE Conference on Emerging Technologies in Factory Automation*: 1199-1205.

Coleman, Henry J. Jr., 1999. What Enables Self-Organizing Behavior in Business. *Emergence,* 1(1): 33-48.

Ekanayake, Samitha W. and Pubudu N Pathirana, 2007. Smart Cluster Bombs – Control of Multi-agent Systems for Military Applications. *Proceedings of the 2007 IEEE International Conference on Networking, Sensing and Control*, pp. 471-476, London, UK, 15-17 April 2007.

Goldberg, David E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading, MA: Addison Wesley.

Feng, Quibin, A. Bratukhin, A. Treytl, and T. Sauter, 2007. A Flexible Multi-Agent System Architecture for Plant Automation, *Industrial Informatics, 2007 5th IEEE International Conference on*, vol.2, no., pp.1047-1052, 23-27 June 2007

Headquarters Department of the Army. 2003. *Mission Command: Command and Control of Army Forces*. Washington, D.C.: Headquarters Department of the Army.

Hocaoglu, Cem and Arthur C. Sanderson, 1997. Multimodal Function Optimization Using Minimal Representation Size Clustering and its Application to Planning Multipaths. *Evolutionary Computation*, 5(1): 81-104.

Liu, Hsing-Han and Pau-Lo Hsu, 2006. Design and Simulation of Adaptive Fuzzy Control on the Traffic Network, SICE-*ICASE, 2006. International Joint Conference*, vol., no., pp.4961-4966, Oct. 2006

Kewley, Robert H., 2000. *Computational Intelligence for Support of Military Tactical Decision Making.* PhD Thesis, Rensselaer Polytechnic Institute.

Kewley, R. H. 2004. Agent-based model of Auftragstaktik: self organization in command and control of future combat forces. *In Proceedings of the 36th Conference on Winter Simulation,* pp. 926-930, Washington, D.C., December 05-08, 2004.

Kewley, Robert H. and Mark J. Embrechts, 2001. A Computational Military Tactical Planning System. *IEEE Transactions on Systems, Man, and Cybernetics, Special Issue: Fusion of Soft Computing and Hard Computing in Industrial Applications,* 32(2): 161-171.

Kewley, Robert H. and Larry. Larimer. 2003. An Agent-¨Based-Modeling Approach to Quantifying the Value of Battlefield Information. *Phalanx* V 36(2): 10-14, 25-26.

Lamont, Gary B., and Ian C. Price. 2006. GA Directed Self-Organized Search and Attack UAV Swarms. *Proceedings of the 2006 Winter Simulation Conference,* 1307-1315.

Lin, Donghui, Cheng Zhu, Peng Ding, and Huanye Sheng, 2004. Supporting Intercultural Collaboration: An Agent Coordination Workflow Model. In *IEEE International Conference on Services Computing*, pp. 629-632.

Lissack, Michael R., 1999. Complexity: The Science, its Vocabulary, and its Relation to Organizations. *Emergence*, 1(1): 110-124.

Ludwig, Jeremy and Arthur Farley, 2008. Using Hierarchical Dynamic Scripting to Create Adaptive Adversaries. *Proceedings of the 2008 Behavior Representation in Modeling and Simulation (BRIMS) Conference.* Availavle at www.sisostds.org.

Mamdani, E. H. and S. Assilian, 1975. An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. *International Journal of Man-Machine Studies*, 7:1-13.

Mitchell, Melanie, 1996. *An Introduction to Genetic Algorithms.* MIT Press.

Silver, Edward A., 2002. *An Overview of Heuristic Solution Methods.* Haskayne School of Business, University of Calgary Working Paper 2002-15.

Stensrud, Brian S., Douglas A. Reece, Nicholas Piegdon, and Annie S. Wu, 2007. Asymmetric Adversary Tactics for Synthetic Training Environments. *Proceedings of the 2007 Behavior Representation in Modeling and Simulation (BRIMS) Conference.* Availavle at www.sisostds.org.

Sycara, Katia P., 1998. Multiagent Systems. *AI Magazine,* 19(2): 79-92.

Xiao, Jing, Zbigniew Michalwicz, Lixin Zhang, and Krzysztof Trojanowski. Adaptive Evolutionary Planner/Navigator for Mobile Robots. *IEEE Transactions on Evolutionary Computation*, 1(1):18-28.

Zadeh, Lofti A., 1975. The Concept of a Linguistic Variable and its Application to Approximate Reasoning, Parts I-III. *Information Sciences*, 8/9: 199-249 v8 (part I), 301-357 v8 (part II), 43-80 v9 (part III).