On the Practical Use of Computers for the Exploration of Military Alternatives

Stephen C. Upton
Jan 1, 2001

Introduction

Alternatives exist in many forms. Example alternatives in an operational setting might be the many tactical plans possible for achieving some mission. Typically, three plans are developed (who's had time for more?). What if a thousand variations on those plans or potentially new plans could be effectively developed and analyzed prior to whittling the choices down to a small number of very good options? In a combat development setting, force alternatives are partially defined by the characteristics and number of various weapon systems that must be selected for use by a future Marine Corps. What if the technology existed to look at thousands upon thousands of different future Marine Corps' that also laid out the strengths and weaknesses for each of these future forces?

As part of the Marine Corps Combat Development Command's Project Albert, we have been developing the seeds of this technology that would enable *any* Marine to explore the consequences of a large number of differing alternatives, whether those alternatives be, for example, tactical plans, weapon systems, or force structures. This integrated technology we call generative analysis, which depends on a set of enabling technologies: distributed computing; agent-based simulations; knowledge discovery methods; and heuristic search techniques. Project Albert has been, and is currently, pursuing a program of developing, integrating, and applying these technologies to problems in the military domain.

The central idea behind generative analysis is the notion that we allow a computer to automatically generate and evaluate a large number of alternatives, represented in a computational form, and do so without human intervention. As we mentioned above, potential alternatives could number in the thousands, and coupling those with an equal number of scenario variations results in potentially millions of combinations. These combinations are represented in the form of agent-based simulations, which are then run on a large number of distributed computers. Agent-based simulations are characterized by their simplicity, in ease of use and level of detail, and fast running times. Distributed computing technology, a field of high performance computing, uses a large number of connected computers to work together on a common problem; in our case, the generation and evaluation of many agent-based simulations.

Though we can generate millions of runs using the technologies described above, the potential number of alternatives and variations can be vastly greater than that. For example, the number of possible rulebases (a rulebase is used to determine the actions of an agent given a particular state) in a very simple agent-based simulation can easily exceed 10^{35}. Throughout this vastness of possible alternatives, heuristic search techniques, such as genetic algorithms, coupled with knowledge discovery methods, enable the computer to intelligently select only those alternatives that are interesting to the user.

Finally, at the completion of these runs, and if we're lucky, there could still be a large number of alternatives that the computer has identified as interesting. These alternatives then become the "seeds" for the next iteration of the Data Farming process. Data Farming is an interactive and iterative process between a human user, a computer, an agent-based simulation, and a set of tools, such as generative analysis. The human provides the intuition and creativity necessary for asking the "right" questions, and the computer, the simulation, and the tools provide the means for addressing those questions, through a series of computational experiments. (See "Data Farming: A Meta-technique for Research in the 21st Century by A. G. Brandstein and G. E. Horne in Maneuver Warfare Science 1998, edited by F. G. Hoffman and G. E. Horne (1998))

Following this introduction is an example application of generative analysis. We next suggest other potential applications, including examining tactical and organizational structures. After the conclusion, we include a "further readings" section for the interested reader, which lists some books and articles on the enabling technologies listed above.

Example Application

Now that we have briefly described the components of generative analysis, we now illustrate how it might be used in a potential example application. Our current efforts to date have not achieved this level of detail, but our hope, and expectation, is that this capability is achievable in the near future. We will apply generative analysis to a typical problem that occurs during the combat development process, namely developing a new weapon system concept. Hopefully, the application of generative analysis to other problems of similar nature will be evident.

The ultimate goal of developing a new weapon system concept, as with any combat development activity, is to improve the effectiveness of MAGTF's. We might do this by substituting a current system with the new system, e.g., replacing the M-16 with the Super-Duper Urban Combat Destroyer, or adding this new system to a unit's Table of Equipment (T/E), e.g., MicroUAVs to perform reconnaissance. (MicroUAVs are small, insect-like Unmanned Air Vehicles that can sense their environment, but have very limited range and performance.) Determining the characteristics of this new weapon system, the number needed, and what tactics and techniques to use are some of the tasks associated with developing any new systems concept.

For our example, we want to explore the impact on mission effectiveness of adding MicroUAVs to an infantry squad's T/E. Our hypothesis is that these MicroUAVs will substantially improve the effectiveness of squads operating in, say an urban environment, however, we are unsure which characteristics of the MicroUAVs have the most impact. We are also unsure as to the number that are needed and the particular procedures to use. Answers to these questions typically decide whether to pursue this technology, and if so, also affect design and acquisition decisions later in the concept development process. For simplicity, we will assume we have available specific descriptions of some MicroUAV concepts provided by technology companies that we wish to explore, as well as some guesses as to the appropriate procedures that we arrived at by wargaming.

Before we describe how generative analysis might be applied to this concept exploration, we might explain how a typical concept exploration might be done using a more conventional approach. After development of Measures of Effectiveness (MOE - these are developed to characterize which alternatives are "better"), a traditional concept development process might then use a detailed simulation to evaluate the different alternatives. If the number of alternatives were large, say greater than 5, then some process would be applied to narrow the field to a manageable number. This approach would probably also entail selecting a small (2 or 3) set of "representative" scenarios and then manually develop (e.g., through a series of wargames) a set of tactics for the squads with MicroUAVs. The "representative" scenarios might contain "mixtures" of urban environments, e.g., industrial, suburban, city center, ports, etc., to determine if these factors impact squad effectiveness with and without MicroUAVs. Each of the scenarios would also probably include a fixed structure and tactics for the Red force, e.g., specific numbers, positions, and types of weapons. A considerable amount of time would also be spent on identifying and collecting data for the simulation. Detailed models for the MicroUAVs would probably need to be developed, in addition to any additional required changes to the simulation, if for example, we couldn't modify an existing object, e.g., using an aircraft to "model" the MicroUAV. Analyses would then be performed on this relatively small number of simulation runs, looking at a small number of alternatives over a limited set of scenarios. Detailed data requirements, the time required to manually develop tactics, and the typically long running time of the usual set of simulations all contribute to constraining the number of alternatives that are effectively explored. Other conventional approaches such as wargaming (in the broadest sense) are also limited in the number of alternatives that could be explored, mainly due to the extensive time and manpower resources required for preparation and conduct of the exploration for each alternative.

How might a generative analysis look? As in the analysis above, we would probably start with development of MOEs; however, we would not feel constrained to develop them fully prior to an exploration of the concept. One of the premises underlying the use of generative analysis is that we don't know all of the boundaries of the analysis, i.e., we don't know what we don't know, and so we would typically begin our exploration with incomplete, imprecise, and uncertain knowledge.

Next, using our simplified agent-based simulations, we would construct an agent model of the MicroUAV. This would include determining, in a coarse fashion, its behavior, such as movement, and its attributes, e.g., how long can it fly, its rate of movement, what kinds of sensors it has, to whom and where does it sends its information, etc. We fill out any parameters with data we know or postulate, in addition to specifying a range of values that it *could* have. Any uncertainty in the parameters will eventually be sampled over, and thus indicate whether that parameter is important to overall effectiveness. Since our simulation agents will have similar structure, the work of building a MicroUAV agent boils down to adding a propulsion, sensor, and communication component, and then filling out the parameters for each of those components. Additionally, we might need to change the cognitive system of the agent to properly reflect its behavior. Once that work is completed, which, we postulate, could take as little as a few hours, a MicroUAV agent is added to an agent repository for potential reuse by other analysts.

We then need to decide on the bounds of the scenario space in which to conduct our exploration of this systems concept. This decision is based on the objectives of the analysis, and also on the amount of resources (e.g., computers, analysts) and time allotted for conducting the analysis. Is it an unconstrained or constrained exploration? If constrained, in what way? Do we examine only inner city environments? suburban? industrial zones? Are the Blue force structures and tactics fixed? Is Red similarly constrained? Which questions we ask and how we make our decisions here to bound our exploration are part of the Data Farming process we mentioned above.

Once we have determined the bounds on the scenario space, we then turn the sampling of the scenario space over to the computer, in effect, generating large number of possible alternatives. From this broad specification given to it by the user, the computer generates points in this scenario space. Each point in the scenario space is a particular scenario with a particular alternative, e.g., a specific number of Blue and Red forces, the tactics and operating procedures used by the Blue and Red forces, a specific number of MicroUAVs, the MicroUAVs specific characteristics, etc. A large number of alternatives generated will most likely be undesirable from a practical or professional standpoint, but by the very nature of this generation, the computational system is in some sense "learning" what works and doesn't work. At the completion of this automated exploration the computer not only outputs a set of alternatives that were considered "best" but also information concerning why those alternatives were "better". (We put the terms "better" and "best" in quotes to emphasize these notions are dependent on the objectives of a particular exploration. For example, a user may be interested in alternatives that incur long mission completion times. For the computer, then, longer mission completion times are "better" and are preferred. The user can then study this set to determine those factors that contribute to longer mission completion times.) These outputs then feed the Data Farming process, where the user, possibly using other techniques, can then study this set, expand his exploration, or go in a different direction.

Depending on the Data Farming objectives, there may be several runs of the simulation with this particular scenario, e.g., if learning is required on the part of the infantry agents using the MicroUAVs, or finding the "best" behavior of the MicroUAVs. The sequence or procedure in which the computer generates the points in scenario space is also a function of the Data Farming requirements. If the goal is an unconstrained exploration, we might use a space filling experimental design. If the goal is to optimize some metric, we might use a natural nonlinear optimization technique, such as simulated annealing or a genetic algorithm. As is probably obvious, the number of possible differing alternatives, in addition to the various parametric changes, result in a very large space to search. However, given the maturation of the enabling components described above - the current and projected availability of very fast computers, fast running agent-based simulations, the new and emerging techniques in the statistical and optimization sciences - we are confident this generative analysis approach will yield additional insight that can not be delivered using conventional techniques. In the same vein, we do not expect to supplant conventional techniques, but to complement them.

The computer continues to sample over the scenario space through a compositional variation, i.e., changing the numbers and types of components in a simulation, and a parametric variation, i.e., changing the values of the attributes of those components. This

exploration is continued until satisfying a user specified termination condition. As part of the Data Farming process, the analyst could then modify his assumptions, change the bounds on the scenario space to search, or ask other questions.

Other Applications

Other concept explorations are similar in methodology, structure and content as the system concept exploration detailed above. The major difference in the system concept exploration and other concept explorations, such as tactical and organizational concepts, is the degree to which we hold elements of the agents and their component's behavior and attributes fixed. For example, if we assume tactics are modeled by the embedded rules in the cognitive system of an agent and their subsequent emergence within a collective of agents (possibly an oversimplified assumption), then to examine a tactical concept, we hold fixed the system components, and vary only these rules.

Tactical concepts are interrelated to specific systems, the environment, and threat capabilities. For example, infantry tactics are very different when the environment is in open terrain (e.g., desert) or restrictive terrain (e.g., urban or jungle). If the threat has significant anti-infantry capability, then the infantry tactics possibly change. For tactical concepts, it is these relationships we wish to explore, and concurrently, have the agents in their synthetic environment explore. The goal is to see if these changes do, in fact, occur, and identify the causal factors. A typical tactical concept exploration might address questions similar to "Given a fixed force structure, how do differing tactics, such as Urban Swarm, Urban Penetration, and Urban Thrust affect operations?" In this case, we have humans who have proposed some initial tactics to investigate.

A typical organizational concept exploration might address questions such as "How do different C2 structures affect urban operations?" In this case, again, we might have human experts propose several different structures in which to evaluate. The ideal case, again, is to have the computer generate and explore different options. An example of this type of exploration might be a change in squad organization, either by modifying the number of Marines in a squad, the composition of the fire teams or how that squad requests resources, such as fire support. Other, more complicated organizational structures can be attempted as we learn how to move up the scale in the number, types and organizational complexity of the agents.

Conclusion

Generative analysis is potentially a powerful new technique within the more encompassing Data Farming process for exploring the consequences of various alternatives. Once mature, generative analysis could provide added understanding and decision support to the whole combat development process. In addition, areas such as mission planning, education, and training could benefit from the increased exploration, and subsequent understanding, of the mission space.

Further Readings

We have included this section for the interested reader who wishes to learn more about the enabling technologies of distributed computing, agent-based simulations, knowledge discovery methods, and heuristic search techniques.

1. Distributed computing technology, a field of high performance computing, uses a large number of connected computers to work together on a common problem. How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters by Thomas L. Sterling, John Salmon, Donald J. Becker, Savarese, Daniel F. Savarese (1999) has a decent introduction to distributed computing. Also is a good how-to on using the Linux operating system to cobble together PC's in a cluster to solve problems. Also, check out these articles on the web: http://www.redherring.com/mag/issue86/mag-computing-86.html (From the Dec 04, 2000 issue of Red Herring magazine, declaring one of the top ten trends for 2001 is distributed computing) and http://www.sciam.com/2000/1100issue/1100cyber.html (From the Nov 2000 issue of Scientific American – an article describing a host of companies that are using the Internet and participant's unused computing power to solve various problems, ranging from ways to look at flu vaccines to search for radio signals from an extraterrestrial intelligence).

2. Agent-based simulations, as we define them, are simple, fast running simulations of a military scenario in what we call "dot warfare" (from the "dots" used to depict the agents on a computer screen). The first, general-purpose agent-based simulation, developed for the Marine Corps is called ISAAC. A description of ISAAC can be found in Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-Life Approach to Land Warfare (CNA Research Memorandum 97-61.10) by Andrew Illichinski (1996). (Available at http://www.cna.org/isaac/downref_isaac.htm). Another good reference that ties together distributed computing, artificial intelligence, and agents is Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence by Jacques Ferber (1999).

3. Knowledge discovery methods are techniques and algorithms designed to extract useful information and even "knowledge" from a set of data. Data mining techniques are a subset of these, and a good introduction to data mining that also includes a description of a free software package is Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations by Ian H. Witten, Eibe Frank (1999). A concise (and inexpensive) introduction is Introduction to Data Mining and Knowledge Discovery, Third Edition by Herbert A. Edelstein (1999). A more technical introduction is the collection of papers in Advances in Knowledge Discovery and Data Mining by Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy (Editors) (1996).

4. Heuristic techniques are a set of procedures, or "rules of thumb", one might use to solve problems. A number of these techniques are designed to search through, or find the optimum of, a large collection of choices. In our case, the choices are the various alternatives that the agent-based simulation could generate. Genetic algorithms are perhaps the most well known heuristic, based on a model of biological evolution. A good introduction to genetic algorithms is An Introduction to Genetic Algorithms (Complex Adaptive Systems Series) by Melanie Mitchell (1998). Genetic algorithms are but one heuristic – for a discussion of other heuristics and their place in problem solving, see How to Solve It: Modern Heuristics, by Zbigniew Michalewicz, David B. Fogel (1999).