Complex 2004

Proceedings of the 7th Asia-Pacific
Conference on Complex Systems
Cairns Convention Centre, Cairns, Australia
6-10th December 2004

Coevolutionary Approach for Strategic Decision Support

Rodney W. Johnson[1], Michael E. Melich[2], Zbigniew Michalewicz[3], Martin Schmidt[4], and Neal Wagner[5]

[1]*Wayne E. Meyer Institute of Systems Engineering, Naval Postgraduate School, Monterey, CA 93943, USA*
Email: `rwjohnso@nps.navy.mil`

[2]*Wayne E. Meyer Institute of Systems Engineering, Naval Postgraduate School, Monterey, CA 93943, USA*
Email: `melich@alumni.rice.edu`

[3]*Department of Computer Science University of North Carolina, Charlotte, NC 28223, USA, Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland, and Polish-Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008 Warsaw, Poland*
Email: `zbyszek@uncc.edu`

[4]*NuTech Solutions, Inc., 8401 University Executive Park, Suite 102, Charlotte, NC 28262, USA*
Email: `martin.schmidt@nutechsolutions.com`

[5]*Department of Computer Science University of North Carolina, Charlotte, NC 28223, USA*
Email: `nwagner@uncc.edu`

Abstract

We present a description and initial results of a computer code that coevolves Fuzzy Logic rules to play a two-sided zero-sum competitive game. It is based on the TEMPO Military Planning Game that has been used to teach resource allocation to over 20,000 students over the past 40 years. No feasible algorithm for optimal play is known. The coevolved rules, when pitted against human players, usually win the first few competitions. For reasons not yet understood, the evolved rules (found in a symmetrical competition) place little value on information concerning the play of the opponent.

1 Introduction

The notion of "big decisions", those that shape the future evolution of a business or organization, frequently attach to the word "strategic". For example, what a company chooses to do or avoid doing is shaped by its answer to the "strategic" question: *Are we a consulting company or a product company?* Or in the case of the US Navy the question has taken the form: Are we an organization that provides prompt and sustained operations at sea, or are we operators of ocean going naval combatants?

How is a strategic question like this to be answered? First, any analysis will ask if there are customers and competitors, and how are they described. Second, can we profitably obtain and serve the customers in the face of the existing and potential competition? Third, what should be done and in what order to become "successful"? Easily stated questions — but difficult to answer. Even describing what constitutes a "good choice or set of choices" is complicated by the tens to thousands of different actions that can be taken. In larger organizations that have had time to evolve in response to competitive and environmental pressures, the allocation of effort — the decision — is most explicitly presented in the "budget". But budgets tend to describe inputs to the organization's activities and not the outputs. Businesses fall back on measures of profitability over some time period as the measure of their success. Military organizations look to wars and conflicts to characterize their success. Thus, analyzing "strategic" questions can be cast as asking: Will a particular sequence of investments, expressed as budgets, over many years produce a successful result in the face of competition and a changing environment?

In the early 1960s, the Department of Defense created a management system, the Planning, Programming, and Budgeting System (PPBS) of considerable complexity to rationalize its resource allocation problems. A major training program was instituted to teach the PPBS and a "game" was created by General Electric's "TEMPO think tank" to train people in the use of the new system. The Defense Management Resource Institute (DRMI) (see www.nps.navy.mil/drmi/ 98org.htm) has used the TEMPO game in its courses for nearly 40 years. Over 20,000 students from 125 countries have benefited from exposure to this game.

We became interested in resource allocation problems while conducting large scale, multi-nation "futures" studies. Our studies used scenario methods (Schwartz, 1991). An integral part of the multi-year competitive decision environment was allocation of national resources to defense. This forced trade-offs between investment in economic growth, foreign assistance, education, etc. These are a very complex set of decisions and we soon realized that we were trimming a very complex decision tree and had little hope of understanding what other options might offer. The work presented here reports one facet of our research program, initiated in 2000, to deal with aspects of resource allocation problems in a world where your competitors are also able to make choices.

2 Coevolutionary Approaches to Games

Games are characterized by rules that describe the moves each player can make. These moves constitute the behavior of the players, the manner in which each allocates his resources. When a player makes a move, he receives a payoff; usually he tries to maximize the cumulative payoff over a period of time. In some games, such as chess, the payoff comes at the end of the game, but we can imagine a surrogate payoff, or evaluation function, that correlates with a player's chances of winning at each point in the course of the game.

The evaluation function is a key ingredient in a game-playing system. Sometimes, however, we have no idea of how to create a good evaluation function; there may be no clear measure of performance beyond simply whether you win, lose, or draw.

The situation is similar to that of living creatures in nature, who are consummate problem solvers, constantly facing the most critical problem of avoiding being someone else's lunch. Many of their defensive and offensive survival strategies are genetically hard-wired. But how did these strategies begin? We can trace similarities across many species. For example, many animals use cryptic coloration to blend into their background. They may be only distantly related, such as the leafy sea dragon and the chameleon, and yet their strategy is the same: don't be noticed. Other animals have learned that there is "safety in numbers," including schooling fish and herd animals such as antelope. These complex behaviors were learned over many generations of trial and error, and a great deal of life and death.

This is a process of coevolution. It is not simply one individual or species against its environment, but rather individuals against other individuals, each competing for resources in an environment that itself poses its own threats. Competing individuals use random variation and selection to seek out survival strategies that will give them an edge over their opposition. Each innovation from one side may lead to an innovation from another, an “arms race” wherein individuals evolve to overcome challenges posed by other individuals, which are in turn evolving to overcome new challenges, and so forth.

It is not surprising that coevolutionary processes have been used by many researchers, whether in optimization or in game playing.

Sebald and Schlenzig have studied the design of drug controllers for surgical patients by coevolving a population of so-called “CMAC” controllers, chosen for effectiveness, against a population of (simulated) patients, chosen for presenting difficulties (Sebald and Schlenzig, 1994). Many researchers have studied pursuit-evasion games, for example (Reynolds, 1994; Cliff and Miller, 1996; Floreano and Nolfi, 1997). Various interesting approaches to constraint-satisfaction problems are reported in (Paredis, 1994; Paredis, 1995; Michalewicz and Nazhiyath, 1995; Le Riche et al., 1995). With a bit of thought, what would appear to be a straightforward optimization problem can often be recast with advantage as a problem of coevolution.

In the remainder of this section we mention some developments in game playing.

In 1987 Axelrod studied the Iterated Prisoner’s Dilemma (IPD) by an evolutionary simulation (Axelrod, 1987). Strategies were represented as look-up tables giving a player’s move — cooperate or defect — as a function of the past 3 moves (at most) on each side. Strategies competed in a round-robin format (everyone plays against every possible opponent) for 151 moves in each encounter. The higher-scoring strategies were then favored for survival using proportional selection, and new strategies were created by mutation and by one-point crossover. Axelrod made two observations. First, the mean score of the survivors decreased in the early generations, indicating defection, but then rose to a level indicating that the population had learned to cooperate. Second, many of the strategies that eventually evolved resembled the simple but effective strategy of “tit-for-tat” — cooperate on the first move, and then mirror the opponent’s last move.

These observations are very interesting, yet they perhaps do not fully explain the degradation in payoff that is seen when a continuous range of options is employed. Hundreds of papers about the prisoner’s dilemma are written each year, and very many of the contributions to this literature have involved evolutionary algorithms in different forms. These and many other studies indicate the potential for using coevolutionary simulation to study the emergence of strategies in simple and complex games.

In the late 1990s and into 2000, Chellapilla and Fogel implemented a coevolutionary system that taught itself to play checkers at a level on par with human experts. The system worked like this. Each position was represented as a vector of 32 components, corresponding to the available positions on the board. Components could take on values from -K, -1, 0, +1, K, where K was an evolvable real value assigned to a king, and 1 was the value for a regular checker. A 0 represented an empty square, positive values indicated pieces belonging to the player, and negative values were for the opponent’s pieces. The vector components served as inputs to a neural network with an input layer, multiple hidden layers, and an output node. The output value served as a static evaluation function for positions — the more positive the value, the more the neural network “liked” the position, and the more negative, the more it “disliked” the position. Minimax was used to select the best move at each play based on the evaluations from the neural network.

The coevolutionary system started with a population of 15 neural networks, each having its weighted connections and K value set at random. Each of the 15 parent networks created an offspring through mutation of the weights and K value, and then the 30 neural networks competed in games of checkers Points were awarded for winning (+1), losing (-2), or drawing

(0). The 15 highest-scoring networks were selected as parents for the next generation, with this process of coevolutionary self-play iterating for hundreds of generations. The networks did not receive feedback about specific games or external judgments on the quality of moves. The only feedback was an aggregate score for a series of games.

The best evolved neural network (at generation 840) was tested by hand, using the screen name "Blondie24", against real people playing over the Internet in a free checkers website. After 165 games, Blondie24 was rated in the top 500 of 120,000 registered players on the site. The details of this research are in (Chellapilla and Fogel, 1999a; Chellapilla and Fogel, 1999b; Chellapilla and Fogel, 2001; Fogel, 2002).

Coevolution can be a versatile method for optimizing solutions to complex games, and a reasonable choice for exploring for useful strategies when there's little available information about the domain.

3 The TEMPO game

The TEMPO Military Planning Game is a two-sided zero-sum competitive game. Teams of players compete in building force structures by dividing limited budgets, over a succession of budgeting periods ("years") between categories such as "acquisition" and "operation" of "offensive units" and "defensive units". The rules are no more complex than the rules of, say, Monopoly. However, players learn that the rules' apparent simplicity is deceptive: they pose challenging and difficult decision problems. No feasible algorithm for optimal play is known.

The full set of investment categories for the TEMPO game comprises: (1) operation of existing forces, (2) acquisition of additional or modified forces, (3) research and development ("R&D"), (4) intelligence, (5) counter-intelligence.

There are four types of forces: two offensive ("Offensive A and B") and two defensive ("Defensive A and B"). Each type comprises several weapon systems with varying acquisition and operation costs (measured in "dollars"), measures of effectiveness (in "utils"), and dates of availability (in "years"). A team's objective is to maximize its total "net offensive utils". A team's net offensive utils of type A are the total utils for its operating Offensive A units, minus the opposing team's Defensive A, but not less than zero. Likewise for type B. Thus there is no advantage in investing more in a defensive system than is necessary to counter the opponent's offensive systems of the same type.

R&D is current investment that buys the possibility in a future year of acquiring new weapon systems, possibly with better price/performance ratios than those now available. Investment in intelligence buys information about the opponent's operating forces and investment activities. Investment in counter-intelligence degrades the information the opponent obtains through intelligence.

Every year the probability of war (PWar) is announced. When this is low, players may well decide to invest heavily in R&D and acquisition of new units; when it is high, they may prefer to concentrate on operating existing units.

4 Initial experiments

In 2000 we did some experiments applying EC to a very rudimentary version of the TEMPO game. There were only one offensive and one defensive weapon system ("OA1" and "DA1"). R&D was eliminated; however the operating and acquisition costs of the systems could vary from year to year. Intelligence and counter-intelligence were also eliminated, but each player was given the opponent's current inventory of the two weapon systems. A game ended with the outbreak of war, or after a specified number of years (around 10). Finally, utils were equated with units, i.e., the util values were set at 1 per unit. This meant that operating an acquisition costs were effectively given in dollars per util, and operating and acquisition decisions could be made with a granularity of 1 util.

The experiments were done with the help of John Koza's Simple Lisp Code for Genetic Programming (Koza, 1992). This is a generational, tree-based Genetic-Programming kernel written in Common Lisp. Individuals (candidate algorithms) are represented as computer programs in a simple Lisp-like language, written in terms of user-specified terminals (constants and variables) and functions. In addition to (1) the set of terminals and (2) the set of functions, the user must specify (3) the fitness measure, (4) a set of fitness cases, (5) a termination condition, and (6) a set of GP parameters such as population size and probability of mutation.

For the rudimentary TEMPO game, the terminals were random floating-point constants and variables describing the current state of the game. State variables included the current total available budget, PWar, current acquisition limits, prices, and operating costs for the offensive and defensive units, and both the player's and the opponent's current inventories of these units.

The function set included operations that attempt to allocate funds for the coming budgeting period to acquisition and operation of the offensive and defensive units. For example (AcOA1 u) allocates funds to acquiring at most u units of "Offensive A1" subject to constraints: the number of units is a nonnegative integer, total expenditures do not exceed the available budget, and total units acquired do not exceed the acquisition limit for OA1. Arguments u that attempt to violate these constraints incur a penalty; for example, if the requested number of units would exceed the acquisition limit, the player receives only the allowed number of units, but the budget is still decremented by the total cost of the number requested. Besides these TEMPO-specific operations, the function set included the elementary arithmetic operations $(+, -, \times, \div)$ and two general programming constructs: (if3 n x y z) evaluates n and then, depending on whether the result is negative, zero, or positive, evaluates and returns the value of x, y, or z; and (progn3 x y z) evaluates its three arguments in order and returns the value of the last.

Investment algorithms were evaluated for fitness by pitting each in games against a selection of others from the same population and adding up penalties according to the result: 0 for a win, 1/2 for a draw, and 1 for a loss. The fitness was $1/(1 + P)$, where P is the sum of the penalties.

A fitness case consisted of initial inventories of the two weapon systems, the maximum number of game years, and initial values, rates of increase or decrease, and volatilities for the budget, PWar (actually the corresponding odds) and the acquisition costs, acquisition limits, and operating costs of the weapon systems. (These latter parameters were updated from year to year within each game by random factors drawn from log-normal distributions determined by the corresponding rates of change and volatilities.) Six fitness cases were defined, and each player was evaluated by one game (each with a different opponent) for each fitness case.

The termination criterion for evolution was simply reaching the specified number of generations.

The main GP parameters for the run reported here were: population 12,000, number of generations 100, and the following proportions for reproduction methods: crossover 89%, copying 10%, and mutation 1%. Other parameters included method of selection (fitness-proportionate) and method of generation (ramped half-and-half, see Koza (Koza, 1992) for the definition).

The question was whether anything reasonable would emerge in such a simple framework. And indeed, starting from an initial generation of completely random programs, an algorithm was evolved that allocated funds according to rudimentary sensible rules, which can be characterized as "dumb, but not crazy". If the budget is adequate, it is equivalent to:

```
(OPOA1 OA1INV)
(ACDA1 DA1ACLIM)
(OPDA1 DA1INV)
(ACOA1 OA1ACLIM)
```

Here OP means “operate”, AC means “acquire”, INV is “inventory,” and ACLIM is “acquisition limit”. Thus the algorithm would not attempt to acquire units beyond the appropriate acquisition limits or to operate units beyond the number in inventory. The actual code was nearly 100 lines of Lisp, mostly introns, which some hand editing reduced to:

```
(OPOA1 OA1INV)
(IF3 (+ PWAR BUDGET) 0 0
    (PROGN
       (ACDA1 DA1ACLIM)
       (IF3
          (OPDA1
             (IF3 DA1INV
                DA1ACCOST
                (OPDA1 DA1ACLIM)
                DA1INV))
          0
          (OPOA1 OA1OPCOST)
          (ACOA1 OA1ACLIM))))
```

This incorporates a check to assure that an initial allocation to operation of offensive units has not exhausted available funds (by more than a fractional dollar) before further allocations are attempted.

5 Design of a New System

In an attempt to improve the evolution of human readable rules we designed a coevolutionary system that evolves Fuzzy Logic rule-bases to play the TEMPO game. The new system has the following features:

- Each individual can encode a maximum of $w = 30$ rules for acquiring and operating weapons (“weapon rules”) and $i = 30$ rules for buying intelligence or counter-intelligence (“intel rules”).

  The chromosome is based on the structure given in Figure 1.

  There are m rules altogether; each Rule$_i$ is built from several fields (Figure 1 expands Rule$_3$):

  - U_3 is a boolean defining whether the rule Rule$_3$ is used,
  - B_{i3} are booleans defining whether input i is used,
  - C_{i3} are centers of the Gaussian in range $[0, 1]$ for input i,

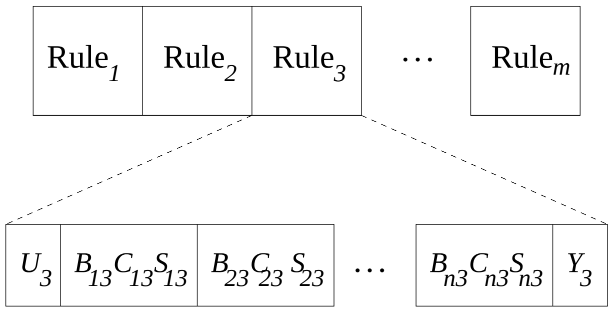

Figure 1. *A structure of a chromosome*

- S_{i3} are sigmas of the Gaussian in range $[0, \inf)$ for input i, and
- Y_3 is the output in range $[0, 1]$ of rule Rule$_3$.

There is one rule set for weapon allocations and a second alike rule set for buying intelligence. Each rule set has a total number of floating point genes equal to $m(1 + 3n + 1)$ (see Figure 1). The parameters of the weapon rule set used in experiments reported further in the paper were $m = 30$ and $n_w = 15$.[1] The intelligence rule set used $m = 30$ and $n_i = 7$.[2] The complete chromosome was encoded as one string of floating point numbers and booleans.

- We use the Mamdani Fuzzy Logic System with Gaussian membership functions,[3] Singleton Fuzzyfier, product operation rule for fuzzy AND, and Centre of Average Defuzzyfication.

- The weapon rules assign a value (a "desirability") to each weapon system; the intel rules assign a value to each intelligence/counter-intelligence category. The budget is allocated by linear scaling of these values, followed by normalization in order not to exceed the available budget.

- We use two populations A and B, each consisting of $pop\_size = 200$ individuals with a fixed genotype length $l = 30(1 + 3 \cdot 15 + 1) + 30(1 + 3 \cdot 7 + 1) = 2,100$. The decoded phenotype has a varying number of rules and membership functions[4] with a maximum number (given by the maximum length l of the chromosome).

- Each individual from population A (B) is evaluated by letting it compete against $o = 40$ randomly chosen opponents from the population B (A). The fitness of each individual is its average "net offensive utils" minus a penalty term that is linear in the number of parameters used by the Fuzzy Logic System (for "pruning"). More precisely, the fitness f of an individual is calculated as follows:

[1]Note again, that each rule can use one or more of the available environmental parameters for weapon allocation and intelligence gathering. The environmental weapon related parameters are: probability of war, budget, weapon category (offensive or defensive), weapon type (A or B), weapon subtype (0, 1, 2, etc), initial units available, maximum number of units available for acquisition, acquisition cost per unit, operations cost per unit, utils (i.e., effectiveness of weapon), utils per operation cost, utils per acquisition cost, year available, enemy offensive force change, and enemy defensive force change.

[2]The environmental intelligence related parameters are: probability of war, budget, intelligence category, offensive and defensive force change for weapon types A and B.

[3]We use one Gaussian spreading parameter for each input.

[4]Rules and inputs can be "pruned", i.e., be "NULL" and hence not used. Pruning reduced the effect of overlearning. The fitness reflects that a smaller rule-base is preferable using a static penalty approach.

$$f = k \cdot u - p \cdot (w + w_i/n_w) - p \cdot (g + g_i/n_i),$$

u is "the average net offensive utils", w is "the total number of weapon rules used", w_i is "the total number of weapon inputs used", g is "the total number of intelligence rules used", g_i is "the total number of intelligence inputs used", $n_w = 15$ is the maximum number of weapon inputs, $n_i = 7$ is the maximum number of intelligence inputs, $k = 10^8$, and $p = 10^4$. Hence, more compact well-performing rule-bases are preferred during the evolution.

- The mutation operator could perform either small or large mutations of each parameter with a probability $p_{mut} = 0.7$. If mutation is selected then each gene has a probably of 0.5 to be mutated. If a gene is selected for mutation then there is a fixed probability of 0.1 for a "big mutation" and 0.9 for a "small mutation". A "big mutation" adds a random number in range $[-d, +d]$ to the gene, while the "small mutation" adds a random number in range $[-d/10, +d/10]$ ($d = 0.1$ is used). Notice that it might seem like overly strong mutation, but due to unexpressed parts of the chromosome which are not translated to the phenotype there are many unused intron-like segments in the chromosome.

- The crossover operator is a standard two-point crossover with $p_{cross} = 0.3$.

- For each population the environment changes from game to game, i.e., available weapons and effectiveness and prices change. This results in a dynamically changing environment in which the rule-bases have to make budget allocations.

Starting from random populations the coevolutionary system develops powerful Fuzzy Logic rule-bases. In order to get an understanding of some kind-of "absolute" performance the best performing individual we let it play against a static "expert" based on simple heuristics (expressed as Fuzzy Logic rules). The "expert" uses the following rules:

if [UtilsPerOperationCost IS Very Low – Low] **then** [Evaluation IS Very Low]

if [UtilsPerOperationCost IS Low – High] **then** [Evaluation IS Medium]

if [UtilsPerOperationCost IS High – Very High] **then** [Evaluation IS Very High]

In other words, the "expert" looks at the effectiveness of each weapon only and disregards any other available information. The performance against the "expert" is not included in any fitness calculations but is used to understand the quality of the evolved rule-bases during the evolution.

The rules can be presented in a form that can be understood easily by humans; one reason for choosing Fuzzy Logic. Here is an example:

RULE 1:
  if [PWar IS Very Low – Low]
    [CATEGORY IS DEFENSIVE]
    [SUBTYPE IS 1 OR 2]
    [Inventory IS Low]
    [MaxAcquisitonUnits IS Low – Medium]
    [AcquisitionCost IS Very Low]
    [UtilsPerAcquisitionCost IS Very Low – Low]
then [Evaluation IS Low]

The terms of the "if" part refer to seven of the environment variables that are available for constructing a weapon rule. A term like "AcquisitionCost IS Very Low" refers to the degree of membership of the acquisition cost in a certain fuzzy set represented internally by a Gaussian membership function with a given center c and standard deviation σ. The program uses the actual numeric values of c and σ internally, and these are the quantities that mutation and crossover operate on. But for the human reader, expressions like "Very Low" are presented, as presumably more palatable than a pair of floating point numbers.

The ranges of meaningful acquisition costs are divided into five subranges running from "Very Low" to "Very High." These ranges depend on the center c of the Gaussian in the following way: "Very Low" if c is in range $[0, 1/8]$, "Low" if c is in range $(1/8, 3/8]$, "Medium" if c is in range $(3/8, 5/8]$, "High" if c is in range $(5/8, 7/8]$ and "Very High" if c is in range $(7/8, 1]$. The reason for this split-up is that the "imaginary center" for each category is placed as follows: "Very Low" is at 0, "Low" is at $1/4$, "Medium" is at $1/2$, "High" is at $3/4$, and "Very High" is at 1. Hence, the imaginary category centers have maximal distance to each other. The human-readable output is generated using the category with the minimal distance to the center c.

The "Evaluation IS Low" in the "then" part of the rule refers to a "desirability" value. Again the program uses a specific number. The human reader is told that the number is in the low subrange of possible desirability values (the same category limits are used as for the "if" part).

6 New Experiments

Initial experience with the coevolution code immediately demonstrated the power of the approach — it proceeded to win first games with most of those who played against the derived rules. It was also clear early on, Winter 2001, that the coevolved rules didn't value information about the opponent's choices. That is, no rules for buying intelligence or counter-intelligence were of sufficient value to be included in the evolved set. Similar behavior had been seen in play of the TEMPO paper game when we were using it to teach our students. We attributed this either to avoidance of excessive inputs — a common human strategy for coping with information overload — or to the "gaming of the game" that occurs when you know approximately when the game will be over. Another possibility was that since the initial version of the TEMPO code provided information that was not quantitative on what the opponent was getting with investments made, there was truly little value. We did some preliminary investigations to determine if we could configure the game so that there might be value to buying intelligence. We gave player X a larger budget and immediate access to all weapons as they became available, while player Y had a smaller budget and was delayed one year in having investment opportunity on the various weapons. This coupled with a reduction in the prolixity penalty did produce a few "weak" rules for the purchase of intelligence by the disadvantaged player.

These efforts immediately highlighted another problem. It had taken approximately 2 weeks of computation on a single 3 GHz processor to coevolve the initial rules. To properly investigate issues of the sort just described would require faster computational turnaround. We embarked on porting of the coevolutionary code to the Processing Graph Method Tool (PGMT), a parallel computing program support system developed at the Naval Research Laboratory (see (Kaplan, 1997; Anderson, 2002)). An application under PGMT is represented as a data-flow graph (similar to a Petri net) whose processing nodes can run in parallel on separate processors. The mapping of nodes to physical processors takes place at runtime. This flexibility facilitates moving an application, without rewriting, from one parallel-processing system to a very different one — from a small, heterogeneous network of workstations, say, to a large, homogeneous, high-performance shared-memory multi-processor system (TEMPO examples were run on two machines of the latter type at NRL's Distributed Center for High Performance Computing: Silicon Graphics Origin 3800 and Altix systems).

We also modified the information provided to be more useful. A rule input was provided indicating whether the opponent had bought counter-intelligence. An input giving the initially available number of units of a weapon system was replaced with one giving the player's current inventory. The opponent's operating forces were given in total utils rather than number of weapon units and these values were given as absolute current values, rather than as changes relative to the previous year. This last change was motivated by the fact that the rules incorporate no "memory" of previous years' decisions.

The result of a coevolution using these modifications has been used in an economics course at NPS. In addition to the coevolutionary system, there is a game system that lets a human player play against a saved individual. The computer distributes its budget according to its rule base, while the human player interacts with the game system through a spreadsheet interface. Many of the students needed three or four tries before achieving an outcome that they were willing to submit for grading. Thus we continue to see human-competitive play in the coevolved rules. One of our colleagues, an economist with previous experience with the DRMI paper form of the game, was able through prolonged and concerted effort to beat the machine by a small margin on a first try. During play, he was ascribing all manner of sophisticated motivations to the machine for its moves. He was dismayed to learn afterward that he had been competing against a set of precisely three rules: the one shown above in section V and the following two others.

```
RULE 2:
   if [Budget IS Low – Medium]
      [EnemyCounterintel IS NOT BOUGHT]
      [SUBTYPE IS 1]
      [Utils IS Low]
      [UtilsPerAcquisitionCost IS Very High]
      [YearAvailable IS Medium]
then [Evaluation IS Low]
```

```
RULE 3:
   if [Budget IS Low]
      [CATEGORY IS OFFENSIVE]
      [TYPE IS B]
      [Utils IS Very High]
      [YearAvailable IS Medium - High]
      [EnemyOffensiveUtils IS Unkn. OR Very Low]
      [EnemyDefensiveUtils IS Unkn. OR Very Low]
then [Evaluation IS Very High]
```

Such a low number of rules is not atypical. The above three rules RULE1, RULE2, and RULE3 constitute a complete rule-base after a completed coevolutionary run. Some analysis revealed that RULE1 acts as a baseline rule stating that unless a weapon has special characteristics it is not worth investing in. RULE2 states that certain defense weapons are worth investing in. It might seem counter-intuitive that this is the case since both RULE1 and RULE2 state "Evaluation IS Low", but analysis revealed that the Evaluation from RULE1 is significantly lower than from RULE2. Both have the same human-readable output "Evaluation IS Low" but the actual value in the phenotype is very different. Hence, this is an example where the human-readable output actually can be misleading the interpretation of the rule-base. RULE3 on the other hand declares that certain offensive weapons are desirable.

Figure 2 shows how the number of rules used by the best player during the coevolution varied over the first 600 generation. The actual run went to generation 1927, but instances of 3-rule best players were already appearing before generation 500.

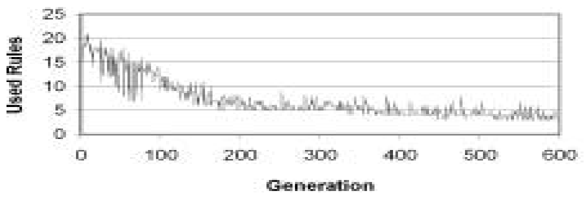

Figure 2. *Number of rules used by best player as a function of generation number*

With the availability of the PGM2 port, we are beginning to be able to use larger populations experiment with somewhat larger problems than previously, in particular to increase the number of weapon systems from 2 to a dozen or so. Doing so, with further relaxation of parsimony pressure, seems to encourage appearance of rule sets (now larger than 3 rules) containing intel rules with High or Very High in their "then" parts.

Figure 3 shows the average number of used rules over all generations versus the prolixity setting p. The figure reports on results for three proxility settings: $p = 1,000$, $p = 10,000$, and $p = 100,000$. The experiment confirmed the intuition that the average number of used rules tends to be higher when the prolixity setting is lower.

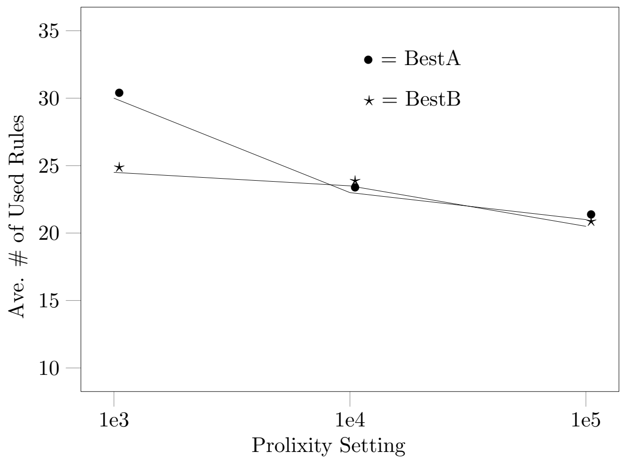

Figure 3. *Average number of rules used by best player as a function of prolixity setting*

Figures 4 and 5 display the diversity of two competing populations, respectively, versus the generation number. Each figure contains 3 plots, one of each setting of the prolixity penalty p.

As shown in these figures, the population diversity decreases with the number of generations till some equilibrium point is reached. Furthermore, after the diversity stabilizes, the general tendency is that increases in population A's diversity correspond with decreases in population B's diversity and vice versa. This can be seen by comparing matching plots from the two figures (i.e., comparing the solid line plot from figure 4 with the solid line plot from figure 5, and so on).

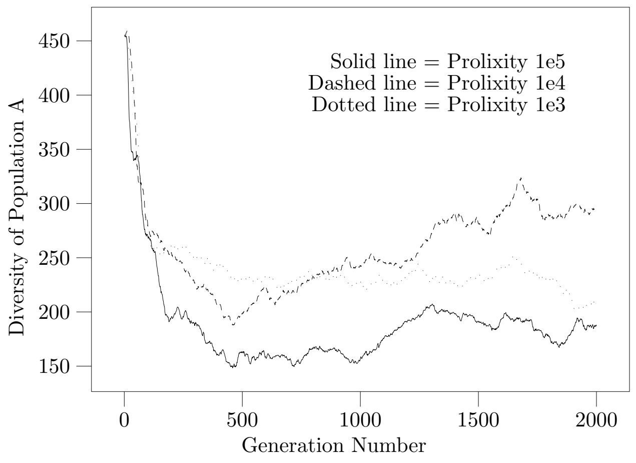

Figure 4. *Diversity of the first population as a function of generation number*

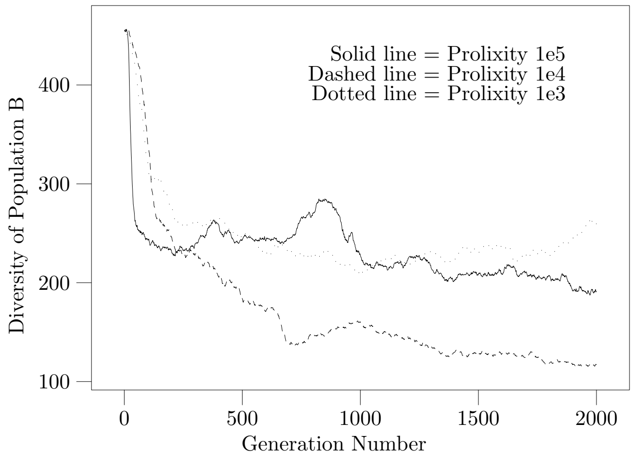

Figure 5. *Diversity of the second population as a function of generation number*

Further experiments may give us some hints on convergence properties of our coevolutionary system.

7 Conclusions and Future Work

Four years ago when we started this work we didn't know if resource allocation problems of the type described in the TEMPO-paper game would be approachable using coevolutionary computation methods. And, even though the initial LISP experimental system suggested an affirmative answer, the nature of what we could learn from a coevolutionary system was not obvious to us. We have learned a number of surprising things:

1. The environment, e.g., PWar, budget size, sequence of available weapon types, cost per utile, etc, is important but knowing what your opponent is doing — intelligence information — is a far slippier component in a sequential game of non-abelian decisions.

2. Though the derived rules can beat most human players immediately, the human players are able to learn the "manner of play" of the machine codes. This suggests that "intel" may after all be important for the machine players to consistently win.

3. Exploring questions that arise, as in 1. and 2., will require a large and growing computational environment. Fortunately, the choice of PGMT has facilitated our ability to move between different multi-processor environments with a minimal amount of recoding.

4. The prolixity penalty appears likely to be useful as a proxy for the information handling capacity of the "decision maker". The more rules an organization uses to make decisions the greater the demand for information processing capability. Since most large organization use fairly simple metaphors for making decisions, and these metaphors can be captured as "if-then" rules, it is possible to imagine exploring alternative rules using different fitness functions to determine why organizations have come to the rules they use. This is very much like the "inverse problem" where given a result we have to find out what was a potential cause of that result. This is why "1R" and "Naive Bayes" work so well so often.

5. Non-transitive (paper, rock, scissors) ordering of strategies seems to be possible.

We are just at the beginning of this research and its application. We need to understand why the code produces rules that favor allocation of most effort to evaluating the cost per utiles of weapon capability and pay little attention to the value of the opponent's behavior. We also need to include investment opportunities in R&D, including both modernization and totally new systems. The current system has PWar and budgets as externally supplied values. A hierarchical competition where the higher-level system that determines these values interacts with the current game is of considerable interest.

Our research has now moved from our preliminary trials on desktop computing machines to networks of processors. We thus far have done our computation in 2-3 Ghz desktop computers, on a loosely coupled cluster of 2 Sun and 3 Silicon Graphics workstations, on 28 processors in a tightly coupled network of 128 processor in the SGI 3800, and are prepared to do experiments in other networked environments. The use of PGMT has permitted us to move efficiently from one computing environment to another. Our expectation is that this combination of a very flexible representation of the resource allocation problem and the computational environment that PGMT provides will permit research on a growing family of poorly understood problems encountered in large living systems.

Acknowledgements

This work was initially supported by Dr. William Mularie of DARPA/ISO and subsequently by the Office of the Secretary of Defense. We greatly appreciate their interest and encouragement for our research on the complexities of using scenarios for planning. The parallel TEMPO game coevolutionary examples were run on resources provided under the auspices of the High Performance Computing Modernization Program Office. We are grateful for their support.

The authors extend thanks to Yunjun Mu for writing code for the TEMPO Fuzzy-Logic coevolutionary system; to Wendell Anderson and Roger Hillson for an education in PGMT and for support in parallelizing the system; and to John Thornton, also for support in the parallelization and for thought-provoking discussions as well.

References

W. Anderson, "Processing Graph Method (PGMT) User's Manual," US Naval Research Laboratory, October 2002.

R. Axelrod, “Evolution of Strategies in the Iterated Prisoner’s Dilemma,” in “Genetic Algorithms and Simulated Annealing,” L. Davis, ed., Pitman, London, pp. 32-41, 1987.

K. Chellapilla and D.B. Fogel, “Evolution, neural networks, games, and intelligence,” in Proceedings of the IEEE, Vol. 87, No. 9, pp. 1471-1496, 1999.

K. Chellapilla and D.B. Fogel, “Evolving neural networks to play checkers without expert knowledge,” in IEEE Transactions on Neural Networks, Vol. 10, No. 6, pp. 1382-1391, 1999.

K. Chellapilla and D.B. Fogel, “Evolving an expert checkers playing program without using human expertise,” in IEEE Transactions on Evolutionary Computation, Vol. 5, No. 4, pp. 422-428, 2001.

D. Cliff and G. F. Miller (1996), “CoEvolution of Neural Networks for Control of Pursuit and Evasion,” University of Sussex, U.K. [Online]. Available: http://www.cogs.susx.ac.uk/users/davec/pe.html

P. J. Darwen and X. Yao, “Why More Choices Cause Less Cooperation in Iterated Prisoner’s Dilemma,” in Proceedings of the 2001 Congress on Evolutionary Computation, IEEE, Piscataway, NJ, pp. 987-994.

D. Floreano. and S. Nolfi, “God Save the Red Queen! Competition in Co-evolutionary Robotics,” in Genetic Programming 1997, J.R. Koza, K. Deb, M. Dorigo, D.B. Fogel, M. Garzon, H. Iba, and R.L. Riolo, eds., Morgan Kaufmann, San Mateo, CA, pp.398-406, 1997.

D.B. Fogel, “Evolving Behaviors in the Iterated Prisoner’s Dilemma,” in Evolutionary Computation, Vol. 1, No. 1, pp. 77-97, 1993.

D.B. Fogel, “Evolutionary Computation: Toward a New Philosophy of Machine Intelligence,” in IEEE Press, Piscataway, NJ, 1995.

D.B. Fogel, “Blondie 24: Playing At The Edge of AI,” Morgan Kaufmann, San Francisco, CA, 2002.

D.B. Fogel and T.J. Hays, “New results in evolving strategies in chess,” in Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation VI, Vol. 5200, B. Bosacchi, D.B. Fogel, and J.C. Bezdek (chairs), SPIE, Bellingham, WA, pp. 56-63, 2003.

P.G. Harrald and D.B. Fogel, “Evolving Continuous Behaviors in the Iterated Prisoner’s Dilemma,” in BioSystems, Vol. 37, pp. 135-145, 1996.

W.D. Hillis, “Co-evolving Parasites Improve Simulated Evolution as an Optimization Procedure,” in “Artificial Life II”, C.G. Langton, C. Taylor, J.D. Farmer, and S. Rasmussen, eds., Addison-Wesley, Reading, MA, pp. 313-324, 1992.

E.T. Jaynes, “Probability Theory: The Logic of Science,” Cambridge University Press, 2003.

D. Kaplan, “Introduction to the Processing Graph Method,” U.S. Naval Research Laboratory, March 1997.

J.R. Koza, “Genetic Programming,” Cambridge, MA: MIT Press, 1992.

D.E. Knuth, “Sorting and Searching,” Vol.3, in “The Art of Computer Programming,” Addison-Wesley, New York, NY, 1973.

J.G. Miller, “Theory of Living Systems,” University of Colorado Press Niwot, Colorado, 1995.

G. Lakoff and M. Johnson, “Philosophy in the Flesh: The Emobided Mind and Its Challenge to Western Thought,” Basic Books, New York, NY, 1999.

R. Le Riche, C. Knopf-Lenoir, and R.T. Haftka, “A Segregated Genetic Algorithm for Constrained Structural Optimization,” in Proceedings of the Sixth International Conference on Genetic Algorithms, L.J. Eshelman, ed., Morgan Kaufmann, San Mateo, CA, pp.558-565, 1995.

D. Lovallo and D. Kahneman, “Delusions of Success,” Harvard Business Review, vol. 81, no. 7, pp. 57-63, July 2003.

Z. Michalewicz and G. Nazhiyath, “GENOCOP III: A Coevolutionary Algorithm for Numerical Optimization Problems with Nonlinear Constraints,” in Proceedings of the 1995 IEEE Conference on Evolutionary Computation. IEEE Press, Piscataway, NJ, pp.647-651, 1995.

J. Paredis, “Co-Evolutionary Constraint Satisfaction,” in Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, Y. Davidor, H.-P. Schwefel, and R. Manner, eds., Lecture Notes in Computer Science, vol.866, Springer, Berlin, pp.46-55, 1994.

J. Paredis,. “The Symbiotic Evolution of Solutions and Their Representations,” in Proceedings of the Sixth International Conference on Genetic Algorithms, L.J. Eshelman, ed., Morgan Kaufmann, San Mateo, CA, pp.359-365, 1995.

C. Reynolds, “Competition, Coevolution and the Game of Tag,” in Proceedings of Artificial Life IV, R. Brooks and P. Maes, eds., MIT Press, Cambridge, MA, pp. 56-69, 1994.

P. Schwartz, “The Art of the Long View: Planning for the Future in an Uncertain World,” Currency/Doubleday, New York, NY, 1991.

A.V. Sebald and J. Schlenzig, “Minimax Design of Neural-net Controllers for Uncertain Plants,” IEEE Transactions on Neural Networks, Vol. 5, No. 1, pp. 73-82, 1994.