Coevolutionary "TEMPO" Game

Rodney W. Johnson* Michael E. Melich† Zbigniew Michalewicz‡
Martin Schmidt§

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 no value on information concerning the play of the opponent.

1 Introduction

Resource allocation in mission or market oriented large enterprises, either government departments or large businesses, is made difficult by the large number of possible investment plans that could be considered. This complexity is in addition to the normal uncertainties associated with a changing environment — changing competition, technical innovation, etc. For example, within the US Department of Defense it is not uncommon for tens of thousands of different categories to be routinely examined annually. Decisions are then made to allocate funds and personnel for the forthcoming budget year as well projections for six years in the future. Similar activities and associated complexity are found in non-governmental organizations [18].

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

*Wayne E. Meyer Institute of Systems Engineering, Naval Postgraduate School, Monterey, CA 93943
†Wayne E. Meyer Institute of Systems Engineering, Naval Postgraduate School, Monterey, CA 93943
‡Department of Computer Science, University of North Carolina, Charlotte, NC 28223, USA, *and* Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland, *and* Polish-Japanese Institute of Information Technology, Warsaw, Poland
§NuTech Solutions, Inc., 8401 University Executive Park, Suite 102, Charlotte, NC 28262

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 [23]. 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 appreciated 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.

Some games are competitive, others are cooperative, and still others are mixed, depending on the form of the evaluation function. If, for example, when one player is gaining in payoff, the other player is losing payoff, it's a competitive 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 hardwired. 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. Furthermore, herding animals of many species have learned to seek out high elevations and form a ring looking outwards, so as to sight predators as early as possible. 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. Antelope learned to form a ring to spot predators more quickly; predators learned to hunt in teams, and use the tall grasses of the savanna to mask their approach. 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.

Note that the individual antelopes did not gather in a convention to discuss new ideas on survival and come up with the strategy of defensive rings on high ground. Nevertheless, the strategies are unmistakably a process of learning. When instinctual, they have been accumulated through random variation and selection, with no evaluation function other than life and death. The entire genome of the species is then the learning unit, with individuals as potential variations on a general theme.

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

An example in optimization is Hillis's now famous example of minimizing a sorting network: a fixed sequence of operations for sorting a fixed-length string of numbers [13]. By an evolutionary search he had found a network that sorted 16 numbers with just 65 comparisons. Networks were scored on the fraction of all test cases (unsorted strings) that they sorted correctly. Hillis then noted that many of the sorting tests were too easy and only wasted time. He therefore devised a method in which two populatons coevolved: sorting networks and sets of test cases. The networks were scored according to the limited number of test cases presented (10 to 20), and the test sets were scored on how well they found problems in the networks. Variation and selection were applied to both populatons; the test cases became more challenging as the networks improved. Hillis reported that the coevolutionary approach avoided stalling at local optima, and that it eventually found a network comprising only 61 comparisons. (This is just 1 short of the best known network to date, discovered by Green and using 60 comparisons [16].)

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 [24]. Many researchers have studied pursuit-evasion games, for example [22, 5, 7]. Various interesting approaches to constraint-satisfaction problems are reported in [20, 21, 19, 17]. 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 [1]. 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.

In 1993 Fogel studied the effect of changing the representation of strategies in the IPD, replacing Axelrod's look-up tables with finite state machines [8, 9]. The results were essentially the same as what Axelrod had observed. Harrald and Fogel, on the other hand, observed entirely different behavior in a version of IPD where the player could make moves on a continuous numeric scale from ?1 (complete defection) to 1 (complete cooperation) [12]. Strategies were represented by artificial neural networks. In the vast majority of trials payoffs tended to decrease, not increase. Darwen and Yao observed similar results in a variation of IPD with eight options, rather than two or a continuous range [6]. They observed, first, that as the number of options to play increases, the fraction of the total game matrix that is explored decreases. Second, when the IPD had more choices, strategies evolved into two types, where the two types depended on each other for high payoffs and did not necessarily receive high payoffs when playing against members of their own type.

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 [2, 3, 4, 10].

There are 1020 possible positions in checkers, far too many to enumerate, and checkers remains an unsolved game: nobody knows for sure whether the game is a win for red, a win for white, or a draw. Chess, at 1064 positions, is still further from being solved. But Fogel and Hays have combined neural networks and coevolution to create a master-level chess-playing program, again without giving the simulated players any feedback about specific games [11].

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, Monopolyœ,A.œ(B. 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 counterintelligence 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 highly simplified version of the TEMPO game. There were only one offensive and one defensive weapon system. R&D, intelligence, and counterintelligence were eliminated. The experiments were done with the help of John Koza's simple Lisp code for Genetic Programming [23]. Individuals (candidate algorithms) were represented as computer programs in a simple Lisp-like language, written in terms of variables that describe the current state, and operations that attempt to allocate funds to various categories for the coming budgeting period. State variables included the current total available budget, PWar, current inventories, acquisition limits, prices, and operating costs for the offensive and defensive units. The budget categories were acquisition and operation of offensive and defensive units. Investment algorithms were evaluated for fitness by pitting each in games against a selection of others from the same population and recording wins and losses.

The question was if 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": it would not attempt to acquire units beyond the appropriate acquisition limits or to operate units beyond the number in inventory, and it incorporated a check to assure that an initial allocation to operation of offensive units had not exhausted available funds before further allocations were 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 rulebases to play the TEMPO game. The new system has the following features: Each individual can encode a maximum of w rules for acquiring and operating weapons ("weapon rules") and i rules for buying intelligence or counterintelligence ("intel rules"). Each rule can use one or more of the available environmental parameters 1. We use the Mamdani Fuzzy Logic System with Gaussian membership functions 2, 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/counterintelligence

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 m individuals with a fixed genotype length n. The decoded phenotype has a varying number of rules and membership functions 3 with a maximum number (given by the maximum length n of the chromosome). Each individual from population A (B) is evaluated by letting it compete against o 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"). Hence, more compact well-performing rulebases are preferred during the evolution. The mutation operator could perform either small or large mutations of each parameter with a small probability PMut. A mutation would be a variation of a parameter by adding a randomly distributed value in the range [?d, +d] or [-d/10, +d/10] for large or small mutation respectably.

The crossover operator is a standard two-point crossover. 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 rulebases have to make budget allocations. Starting from random populations the coevolutionary system develops powerful Fuzzy Logic rulebases. 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 performance against the "expert" is not included in any fitness calculations but is used to understand the quality of the evolved rulebases 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:

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 ?. The program uses the actual numeric values of c and ? 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 numbers like 48.7682, 17.1056. The range of meaningful acquisition costs is divided into five subranges running from "Very Low" to "Very High". Here the interval cœ,A1œ(B? lies within the "Very Low" subrange for acquisition costs. 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 "goodness" values. Thus, to the extent that the rule applies to a weapon, it is a reason not to buy it.

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 a 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 [24, 25]). 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 onefrom a small, heterogeneous network of workstations, say, to a large, homogeneous, high-performance shared-memory multiprocessor system.

We also modified the information provided to be more useful. A rule input was provided indicating whether the opponent had bought counterintellignence. An input giving the initially available number of units of a weapon system was replaced with one giving the player's current inverntory. 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 spreadsheed 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.

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]

if [Budget IS Low]
[CATEGORY IS OFFENSIVE]
[TYPE IS B]
[Utils IS Very High]
[YearAvailable IS Medium - High]
[EnemyOffensiveUtils IS Unknown OR Very Low]
[EnemyDefensiveUtils IS Unknown OR Very Low]
then [Evaluation IS Very High]

Such a low number of rules is not atypical. Figure 1 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.

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.

7 Conclusions and Future Work

Four years ago as 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.

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 utils 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. This would bring us closer to a tool for improving our understanding of complex planning problems that initiated this research in 2000.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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

[6] 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.

[7] 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.

[8] D.B. Fogel, "Evolving Behaviors in the Iterated Prisoner's Dilemma", in Evolutionary Computation, Vol. 1, No. 1, pp. 77-97, 1993.

[9] D.B. Fogel, "Evolutionary Computation: Toward a New Philosophy of Machine Intelligence", in IEEE Press, Piscataway, NJ, 1995.

[10] D.B. Fogel, "Blondie 24: Playing At The Edge of AI", Morgan Kaufmann, San Francisco, CA, 2002.

[11] 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.

[12] P.G. Harrald and D.B. Fogel, "Evolving Continuous Behaviors in the Iterated Prisoner's Dilemma", in BioSystems, Vol. 37, pp. 135-145, 1996.

[13] 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.

[14] D. Kaplan, "Introduction to the Processing Graph Method", U.S. Naval Research Laboratory, Mar. 1997.

[15] J.R. Koza, "Genetic Programming", Cambridge, MA: MIT Press, 1992.

[16] D.E. Knuth, "Sorting and Searching", Vol.3, in "The Art of Computer Programming", Addison-Wesley, New York, NY, 1973.

[17] 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.

[18] D. Lovallo and D. Kahneman, "Delusions of Success," Harvard Business Review, vol. 81, no. 7, pp. 57-63, July 2003.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] Peter Schwartz, The Art of the Long View: Planning for the Future in an Uncertain World. New York: Currency/Doubleday, 1991.

[24] 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.