Genetic Programming as an Isomorphic Analog to Bounded Rationality in Agent-Based Models

S. M. Manson

Department of Geography, University of Minnesota
414 SST, 267 19th Ave South, Minneapolis MN 55455
Telephone: +1 612 425 4577
FAX: +1 612 424 1044
Email: manson@umn.edu

Abstract

This paper examines the use of genetic programming and agent-based modeling as an isomorphic analog of bounded rationality in individual decision making. This approach is explored in the context of land change modeling in the southern Yucatán peninsular region of Mexico. Results are reported for experiments linking varied implementations of genetic programming to features of bounded rationality.

1. Introduction

Global environmental change is occurring at a rate and magnitude unparalleled in human experience. Particularly important is human activity that impinges on land-use systems and surficial features, known as land-use and land-cover change (LUCC or land change). Central to understanding this change is use of spatially explicit, dynamic models that examine decision making, societal context, and ecological relations (Brown et al., 2004). While there is no single best approach to modeling LUCC, there is considerable interest in methods based in the complexity sciences, such as genetic algorithms, cellular automata, and artificial neural networks (Agarwal et al., 2002; Verburg et al., 2005). Also attracting interest are agent-based models, collections of semiautonomous software programs, or agents, that represent the complex behavior of interacting entities (Janssen, 2003; Parker et al., 2003).

A key challenge to the use of agent-based models for LUCC research—and one reflective of a perennial issue in land change science—is representing agent decision making. A variety of approaches are employed, including (but not limited to) heuristics, vector weighting schemes, statistical modeling, belief-desire-intention models, and classifier systems (Parker et al., 2003). Regardless of the method used, there are challenges with respect to memory, optimization, complexity of strategies, learning, innovation, and communication with other agents and the larger decision-making environment. This work has ramifications for geocomputation and its cognate fields of human-environment modeling, computational intelligence, decision sciences, and complexity research.

Issues raised by representing agent decision making are explored here through use of the SYPR Integrated Assessment (SYPRIA), named for the southern Yucatán peninsular region of Mexico, which focuses on decision making in the context of institutions and the environment (Manson, 2004b, 2005a) This paper reports on several aspects of how SYPRIA represents rationality in agents with genetic programming. In essence, an agent is invested with a population of genetic programs and can thereby be considered as having multiple strategies with which to deal with structural changes in the environment. There are a number of issues, however, raised by how the corollaries of rationality can be legitimately reflected in genetic programming. Tests are detailed for the genetic programming parameters of creation

mechanisms, number of generations, population size, selection mechanisms, and fitness measures. These parameters are examined with respect to corollaries of rationality related to memory, bounding of computational resources, complexity that strategies can achieve, agent learning, innovation, exploitation of effective strategies, and interagent communication.

2. Land change and models of decision making

Land change in coupled human-environment systems is almost by definition a function of human decision making (Agarwal et al., 2002). Nonetheless, Irwin and Geoghegan (2001) note that land change models tend to ignore individual decision making or rely on ad hoc theoretical frameworks in order to focus on larger-scale social and ecological determinants of land change. They argue for a greater role for explicit incorporation of theories of individual decision making, specifically economic ones, into LUCC models. More broadly, given the variety of settings in which land change is influenced by non-economic factors, standard economic models of decision making can be seen as one of several that relate to individual decision making and land change (Geist and Lambin, 2001).

2.1 Models of decision making: perfect and bounded rationality

The most commonly accepted model of decision making in the social sciences—and by extension land change models that actively engage with decision making—is rational choice theory. Under the rubric of rational choice, perfect rationality has become the standard model of decision making by drawing on several assumptions that grant it power and tractability. Key among these is utility maximization, which holds that a given decision maker chooses from a set of mutually exclusive exhaustive alternatives the one that gives greatest utility (after meeting constraints and meeting stable individual preferences). Also important is how actor determine the optimal alternative through perfect computation and complete information on alternatives jointly distributed with those of other actors (Myers and Papageorgiou, 1991)

Decision making also may be viewed through the lens of bounded rationality. The longstanding normative and mathematical tradition of rational choice has been joined by a more descriptive empirical focus on the individual and societal dimensions of decision making (Cook and Levi, 1990). This exploration has lead to consideration of alternatives to perfect rationality. Key among these is bounded rationality, introduced by Simon as "rational choice that takes into account the cognitive limitations of the decision maker – limitations of both knowledge and computational capacity." (1997: 267) Actors under bounded rationality do not optimize because they possess limited computational resources and information on which to base decisions. Actors instead satisfice—make suboptimal yet acceptable decisions—with a small cadre of partial strategies based on imperfect information. Actors improve strategies through a Darwinian process of learning-by-doing that balances path-dependent exploration of new strategies and exploitation of existing strategies (Gigerenzer et al., 2001). Attempts to reconcile bounded and perfect rationality tend towards rolling boundedness into perfect rationality (e.g., by treating the time necessary for choice generation as another optimization factor or by using probability to approximate subjectivity) but these approaches do not genuinely address underlying issues (Conlisk, 1996; van den Bergh et al., 2000). Bounded rationality and perfect rationality therefore remain different yet complementary theories of decision making that can be applied to land change modeling.

The choice of theoretical framework is important to representing the decision making of individual actors in an agent-based model. Decisions about where agents choose to change land is a multicriteria evaluation problem (for review see Collins et al., 2001), where an actor chooses locations that for land change as a function of suitability S of these locations for given production activities:

$$S = \sum_{i=1}^{m} w_i v_i \prod_{j=1}^{n} b_j \quad (1)$$

as a function of continuously varying spatial factors $V = \{v_1, \ldots, v_m\}$, a set of factor weights $W = \{w_1, \ldots, w_m\}$, and a set of Boolean constraints $B = \{b_1, \ldots, b_n\}$. Equation (1) identifies for each agent the propensity for agriculture, for example, as a function of environmental and social factors over space. Agents can treat Equation (1) as a symbolic regression problem where response-variable observations are used to estimate parameters of independent predictor variables. Function $f(x)$ is known solely through observations $X = \{x_1, \ldots, x_n\}$. The observed value of the function at x_i, or $f(x_i)$, is denoted $\bar{f}_i$ and is related to the true value f_i through error defined as $e_i = f_i - \bar{f}_i$. Symbolic regression approximates $f(x)$ as $\hat{f}(x)$:

$$\hat{f}(x) \approx \sum_{j=1}^{n} a_j f_j(x) \quad (2)$$

comprised of functions $f_j(x)$ with coefficients a_j estimated to minimize e_i over the observations (after Ralston and Rabinowitz, 2001) given by $X = \{x_1, \ldots, x_n\}$ as locations in spatial layers given by V and B.

Suitability may calibrated against a dependent variable as one would in any inductive alternative oriented model (Carroll and Johnson, 1990). The means of approximating $\hat{f}(x)$ is dependent on the theoretical imperatives of the underlying decision-making model. There is a host of analytical methods for solving those problems that almost without exception are based in perfect rationality (Manson, 2004a). There is also a growing number of methods suited to bounded rationality, however, and of these, particularly promising for modeling decision making are evolutionary programming methods that draw on biological metaphors to create computer programming systems (Eiben and Schoenauer, 2002).

2.2 Evolutionary programming for land change modeling

Evolutionary programming techniques are applied to decision making in two general ways: functional and representational (Edmonds, 1999). The ***functional*** use is the most common, where an evolutionary program is used to conduct a strict instrumental search for complex yet well specified problems. Optimization of complex discontinuous functions, for example, is an ideal task to which evolutionary programs are be applied. In some domains, particularly those characterized by noise and complexity, these programs outperform other methods (Koza, 1992; Kaboudan, 2003). The most popular use of evolutionary programming in land change modeling is as a functional optimizing tool for land-use planning (Balling et al., 1999; Matthews et al., 1999; Diplock, 2000; Krzanowski and Raper, 2001; Xiao et al., 2002).

The less common ***representational*** use of evolutionary programming is oriented towards characterizing situations where human do not conform well to the perfectly informed and rational decision makers conceived under rational choice theory (Edmonds, 2001). The link between bounded rationality and evolutionary programming is weakly isomorphic—suggestive similarities are borne out empirically and theoretically but few believe that humans actually use genetic programs for cognition (much in the same way, one could suppose, that humans are not strictly rational) (Dosi and Nelson, 1994; Chen, 2003).

Evolutionary programming can be used explicitly to represent land-use decision making by having agents solve Equation (1) as a symbolic regression problem. Per Equation (2), function $f(x)$ is known solely through observations $X = \{x_1, \ldots, x_n\}$; in the rubric of evolutionary programming, training set $O = \{(o, f(o)) \mid o \in X\}$ is drawn from the problem domain to train a set of programs $K = \{(k, g(k)) \mid k \in X\}$ that are strategies that approximate

function $\hat{f}(x)$. These programs evolve to minimize error e over samples in a manner similar to how other regression methods use error-driven minimization techniques such as ordinary least squares (OLS). Error e for a genetic program is given by an error function; merits of varying fitness measures are explored in detail below.

Of particular interest here is a variant of evolutionary programming termed genetic programming. Genetic programs are ***trees*** are composed of atomistic terminals and functions in a branching structure. In terms of the underlying biological metaphor, the tree that constitutes a genetic program is equivalent to that program's genetic code and each function and terminal is analogous to a gene. Each terminal or function is a ***node*** in a branching network. Consider a simple tree that corresponds to the symbolic regression equation ($Y = \beta_1 X_1 + \beta_2 X_2$). This equation can be represented by a tree that possesses multiple branches, where one branch is composed of the terminals (X_1) and (β_1) and another by (X_2) and (β_2), both joined by the multiplication function ($\times$) that is implicit in the regression equation ($Y = \beta_1 \times X_1 + \beta_2 \times X_2$). These subbranches in turn are joined by the addition function (+) to create the entire tree. This function (+) occupies the ***root*** node of the tree. While limited to a single root, trees are essentially limitless in terms of the number of branches they can have. The ***depth*** of the tree is the number of branches spanning the longest descent of branches from the root to terminals while the ***length*** of the tree is the total number of functions and terminals.

Terminals, such as β_1, X_1, β_2, and X_2, are typically suggested by the problem domain. In terms of Equation (1), the terminals are values drawn from spatial factors used by agents in their decision making. A special class of terminals is ephemeral constants, which take the value of random numbers in order to broaden the applicability of genetic programs to symbolic regression. If a genetic program tree were composed solely of terminals that correspond to spatial variables, it is unlikely that a suitable tree could evolve to solve the symbolic regression equation, since the genetic program could only work with values given by the spatial variables. ***Functions*** compose all nodes between the terminals and the root. Genetic programs can use functions of differing complexity, although in general there is an emphasis on simple functions in order to allow the evolutionary program to evolve higher functions. An arbitrarily complex function foisted upon a genetic program, for instance, may contain inefficiencies for which the genetic program spends resources attempting to compensate (Banzhaf et al., 1998).

There are several means by which genetic programs evolve over time (Figure 1). In the genetic program system there is a population K of size M of genetic program trees. Initially, these trees are completely random in their structure and choice of terminals and functions. This initial population is generation 0. Members of generation 0 vie for the opportunity to pass on their genetic material to the next generation, or generation 1; in essence, the genetic program system manages a series of iterations in which a parent generation seeks to create offspring in a child generation. In the next iteration of the genetic program system, the child generation becomes the parent generation seeking to create a new child generation. The children in generation 2 are in essence the grandchildren of generation 0. A typical genetic program system run may oversee the evolution of as few as three generations and upwards of several thousand generations to a predefined limit G.

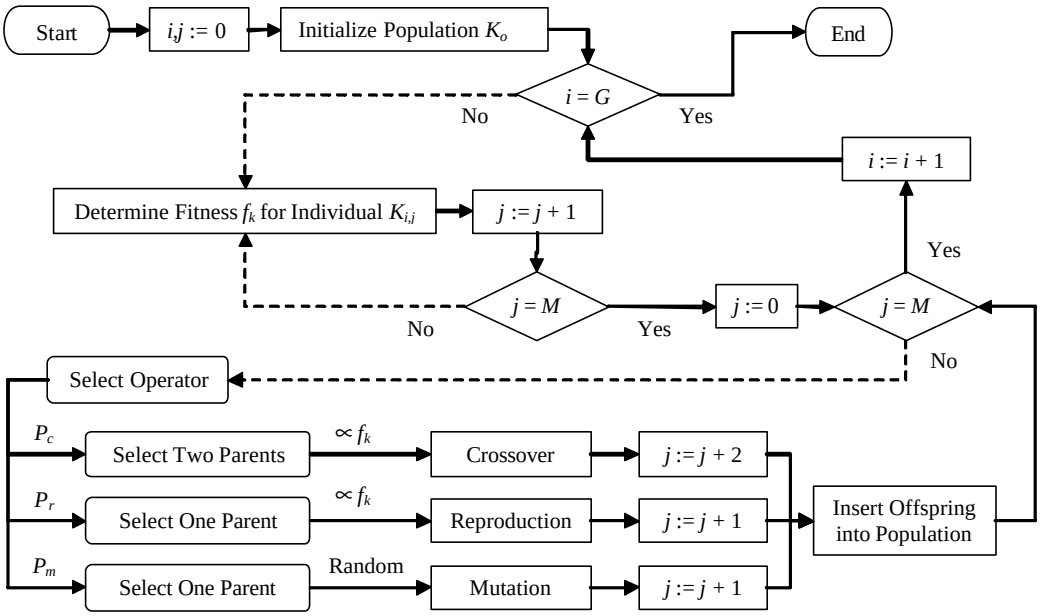

Figure 1. Genetic programming system

Child genetic programs result from one of three different kinds of operation: reproduction, crossover, and mutation. ***Reproduction*** is analogous to asexual reproduction or cloning, whereby a genetic program in the parent generation makes a copy of itself for insertion into the child generation. ***Crossover*** is similar to sexual reproduction, where two parents create a single offspring that carries a mixture of the parents' genes. ***Mutation*** is a special form of reproduction in which a parent creates an identical child but in the process, a gene is randomly mutated. The genetic program system must have some means of ***selection*** by which parents are chosen to create offspring. In Darwinian terms, fit parents have a higher likelihood of finding a partner and are more likely to find a partner of similar fitness (f_k). Over generations, strategies with low fitness are weaned out and replaced with ones that are more successful. Selection procedures and their effects on genetic program fitness are considered in detail below.

Each genetic program operator performs a different role in improving population fitness in addition to having ramifications for representing bounded rationality (below). Reproduction is useful for maintaining coherence of good strategies across generations. Mutation is necessary to ensure that terminals and functions do not disappear from the population, as functions can be lost over generations when early successful strategies do not use them and the offspring of these strategies propagate quickly. This loss of diversity can reduce overall population fitness in the end. Crossover is the driving force behind evolution since it quickly culls the population of substandard strategies while creating potentially better strategies through the intermingling of well performing genetic programs (Korkmaz and Ucoluk, 2004).

2.3 Genetic programming and bounded rationality

Under a representational view of using genetic programs to model decision making, a genetic program tree represents a single strategy designed to solve Equation (2). The use of genetic programs as an isomorphic analog to bounded rationality is supported by the emerging field

of agent-based computational economics, where these approaches have been applied to financial markets, macroeconomics, innovation, environmental management, and labor economics (Tesfatsion, 2002). There is a growing body of research that uses evolutionary programming to examine decision making, particularly in an agent framework (Chattoe, 1998; Beckenbach, 1999; Chen, 2001).

In an agent-based model framework, individual genetic programs can be assigned to single agents or an entire population of strategies may be invested in an individual agent. These are termed the "population interpretation" and the "mental interpretation" respectively (Chattoe, 1998). The ***population interpretation*** is more common because it reduces the computational complexity of the genetic program system. The population interpretation found an early niche with economic modeling designed to prove the compatibility of models based on genetic algorithms (the precursor to genetic programs) with more traditional economic models. This work treats each genetic program as its own agent, or treats each program as representing the single strategy held by each agent (Holland, 1992; Arifovic, 1994). One corollary of the population interpretation is that it introduces a global fitness function and agents are granted perfect information, since the single strategy of each agent is compared against that of other agents. This common comparison can be useful in situations where models of social learning assume complete interchange of information (Vriend, 2000) but may not hold with the informational limits imposed by bounded rationality. Furthermore, it is difficult to justify this model when one is concerned with how individuals face the world, as this use of genetic programs muddies the correspondence between particular individuals and particular trees.

In contrast, the ***mental interpretation*** imbues each agent with a population of strategies that are considered alternative models of the world. The member of this population with the highest fitness is accorded the status of the agent's prime worldview and the agent uses this one to guide its behavior. In a land change context, the fittest genetic program represents a household multicriteria evaluation strategy. An advantage of using the entire genetic program population for a single agent is that precepts of bounded rationality are easier implement. Bounded memory and limited computational ability are easier to implement by simply limiting the number of trees available to a given agent and the depth or length that these trees can achieve. The mental interpretation also allows multiple strategies to coexist even when only one is used as the basis for agent action. This is analogous to an agent having a series of rules of thumb whose use varies according to the agent's current situation, which allows the agent to deal with structural changes in the environment. If each agent possesses its own cadre of genetic programs that represent alternative strategies, every actor can have its own, subjective, fitness criteria defined by its local environment. Evolution can therefore be a local phenomenon even when a single test is employed to measure fitness across all genetic programs. There is also evidence for coevolution, whereby the evolutionary environment is not necessarily independent but determined in part by the actions of evolving objects (Holland, 1995).

Learning as represented by evolutionary search techniques can be seen as a contest between deduction and induction, or alternatively, between exploitation and exploration. A search algorithm must balance the two in that exploration is necessary to find new areas in the search space and exploitation allows the technique to employ the strategies already found. A random search, for instance, is highly exploratory at the cost of not exploiting past searches. A genetic program represents the use of intelligent shortcuts, in the form of previous generations of candidate solutions, in finding better candidate solutions. Genetic programming can be considered deductive or exploitive because current knowledge and understanding about the problem domain leads to new knowledge. At the same time, it is deductive in that strategies are tested against reality; strategies become hypotheses that are

played out against the environment. Usefully, evolutionary programs tend towards a metastable state that hovers between asymptotic convergence and explosion (Dawid, 1999). In other words, a genetic program never stops experimenting with, and trying, new strategies (Riechmann, 1999). Reproduction equates with exploitation or ***persistence*** of good strategies over time, as it allows the agent to exploit effective programs that it has already discovered. This is the essence of rules of thumb because it offers a good solution that is very efficient in terms of not having to devote effort to search for new candidate solutions at the cost of not finding new and potentially better solutions. Mutation is accorded the status of ***innovation*** as a form of exploration (Chen and Chie, 2004). Random errors and accidents are no strangers to decision-making theory. Many human innovations have come about by accident but the decision maker can deliberately incorporate random changes in strategy into later generations.

Interaction, ***communication***, and ***learning*** are broad interpretations given to crossover, in the form of agents engaging in imitation, communication, and development of new strategies through combination of older strategies (Pingle, 1995; Dawid, 1997). Interaction occurs between agents when agents trade strategies and within an agent when crossover takes place among strategies in the population contained by that agent. Either way, crossover is a form of imitation and communication where agents can borrow strategies from other agents or create new strategies. Treating crossover as interaction poses several challenges to the task of linking evolutionary program to decision making (Bosch-Domenech and Vriend, 1998; Chattoe, 1999). One is related to genotype visibility, or the extent to which individuals glean knowledge of other individual's genotypes. Linking phenotype and genotype supports the assumption that it is easier to perceive other's actions that it is to understand their thoughts. Households may share similar organization and underlying precepts, for instance, but behave differently within their environment. In the meantime, the computational conceit of genetic programs is that function follows form and that the structure is directly tied to action. In addition, it can be argued that an agent should not be able to reconstruct the underlying strategy of another agent merely by examining the outcome, not only because of genotype-phenotype correspondence, but also because it implies direct communication between agents (Beckenbach, 1999). Alternatively, crossover may be a way to implement search costs since genetic programs are chosen probabilistically and therefore less than perfect programs can engage in crossover (Brenner, 1999). Moreover, crossover represents imperfect information because it can be a destructive force as well as a constructive one in terms of overall population fitness. While offspring generally possess greater fitness than parents do, a minority of offspring fare less well when the two inherited subtrees fail to work together well.

Limits to computational resources can be implemented in genetic programs. ***Memory*** is implicit in offspring programs because subtrees are passed on from parents and the mental interpretation gives actors, not individual programs, access to longer-term variables or environmental variables. ***Learning time*** is also a potential disconnect between genetic programming and decision making. Humans typically have fewer than ten or so opportunities to learn strategies when many evolutionary techniques require hundreds of repetitions (Arthur, 1993; Andreoni and Miller, 1995). One solution is to ignore this dilemma by holding that genetic programs are not a direct analog of decision making but instead are just proxies to decision making. A better solution equates the power of human imagination with multiple generations of genetic programs. This view credits the decision maker with the ability to mentally pursue and weight alternative strategies and equates this process with many genetic program generations.

In summary, multicriteria evaluation can be recast as a symbolic regression problem that can be solved with genetic programs. Genetic program trees are composed of terminals

chosen from the problem domain to reflect pertinent factors while functions are chosen to allow candidate solutions to solve the symbolic regression problem. The mental interpretation, whereby each agent is given a population of genetic program trees, allows agents to have multiple simultaneous strategies and grants a number of advantages over the more common population interpretation. Genetic programs evolve through reproduction, mutation, and crossover and these operations are accorded, respectively, the status of persistence, innovation, and interaction, communication, and learning. Genetic programs therefore provide a convenient means of representing many facets of decision making.

3. Methodology

SYPRIA is named for the southern Yucatán peninsular region, which lies on Mexico's border with Guatemala (Figure 2). The 22 000 km^2 expanse is the site of potentially massive deforestation that threatens one of the largest remaining subhumid tropical forests and its associated socioeconomic and biogeochemical systems. The chief proximate cause of this land change is smallholder agriculturalist households whose production activities include swidden cultivation, agroforestry, modest logging, and market cultivation. These households are represented within SYPRIA by agents in an agent-based model and each agent is equipped with genetic programs in accordance with the mental interpretation noted above. Fuller explorations of SYPRIA are considered elsewhere (Manson, 2000, 2004b, 2005a; 2005b).

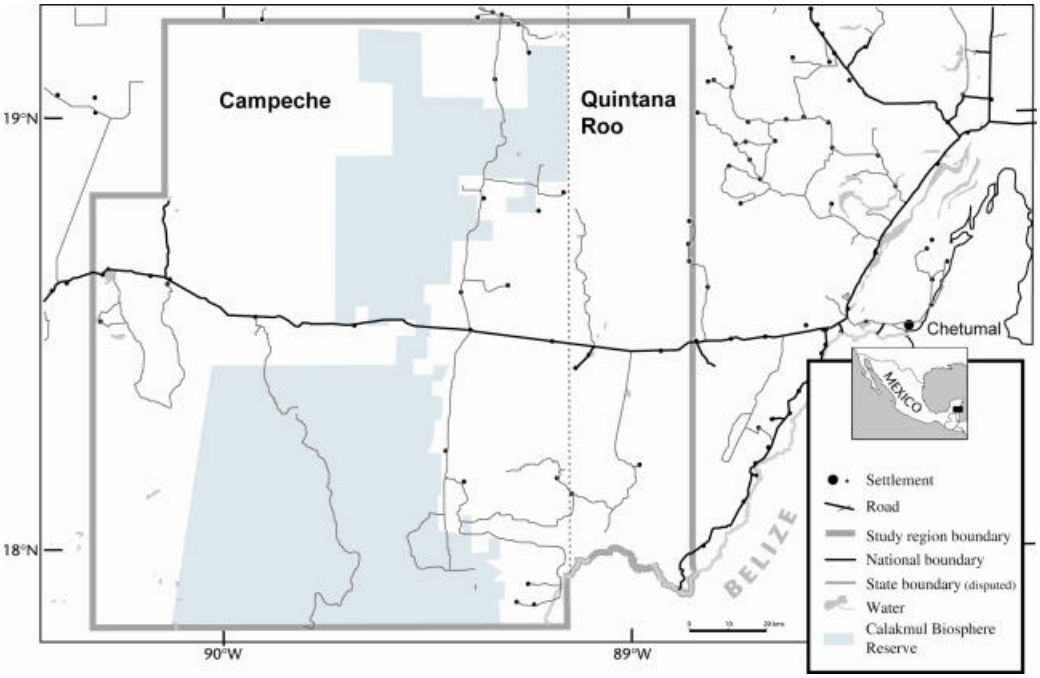

Figure 2. Southern Yucatán peninsular region

There are a variety of parameters governing genetic program behavior, such as population size or mutation rate, which are important for modeling decision making. The values for these parameters, outlined in Table 1, were initially selected from the literature.

Parameter	Value
Population size (M). Number of genetic programs in the population.	100 – 1000 Default: 300
Generations (G). Number of generations over which genetic programs evolve.	10 – 100 Default: 50
Crossover Probability (P_c). Probability that a given genetic program will be the offspring of two other programs as opposed to propagation or mutation.	0.7 – 0.9 Default: 0.9
Mutation Probability (P_m). Probability that a given genetic program will mutate in a generation.	0.001 – 0.9 Default: 0.001
Reproduction Probability (P_r). Probability that a given genetic program will be the offspring of two other programs.	$1 - (P_c - P_m)$
Internal crossover points (%). Probability that a given genetic program will be crossed or mutated at a function instead of a terminal. This also determines the odds that a newly introduced mutation is a function.	0.5 – 0.9 Default: 0.7
Creation depth. Maximum depth of programs when created.	6
Crossover depth. Maximum depth at which a program may be crossed with another.	17
Function set. Functions that can act as tree nodes.	$+ - \div \times$
Random number. Probability of choosing a random number as a terminal. These numbers, also termed ephemeral constants, are necessary for a genetic program to solve a symbolic regression problem.	0.3 – 0.7 Default: 0.5
Program length. Limit placed on maximum genetic program length.	100 – 1000 Default: 500
Tournament size. Tournament is held between X individual genetic programs, where X is equal to the value of tournament.	2 – 7 Default: 5

Table 1. Genetic program parameter settings

In order to test the effects of key parameters, SYPRIA was run 600 times in a test of the five parameters of interest. The result of interest is the average fitness scores of the genetic programs created by the model, or $f_{\bar{P}}$, the population fitness. This score is a measure of how well the genetic program performs as a candidate solution to the multicriteria evaluation problem defined in Equations (1) and (2). Linear regression and ANOVA methods are used to elucidate the effects of varying configurations of model parameters.

4. Results

4.1 Fitness and error function

The heart of genetic programming lies in breeding better strategies. Parents are selected with a variety of selection schemes that probabilistically choose them based on their fitness. The sum fitness of a population is denoted f_P, average fitness $f_{\bar{P}}$, and the fitness for a given individual is f_k. As discussed above, the fitness function minimizes error e over samples, much like other regression techniques. Candidate solutions, $K = \{(k, g(k)) \mid k \in X\}$, are compared to the training set, $O = \{(o, f(o)) \mid o \in X\}$. The fitness function gives feedback to enable learning in the genetic program system by comparing how well genetic programs do against the learning set. The fitness function is applied to a standardized measure of fitness, whereby the performance of genetic programs is rescaled every generation to give the best-performing individual a fitness of zero.

The most popular form of error function in summed error, or the sum of the absolute value of the differences between program output and the corresponding learning set:

$$e = \sum_{i=1} |k_i - o_i| \qquad (3)$$

Another form of the error function calculates mean error in order to include the effect of sample size, but is otherwise identical to total-sum-of-errors formulation:

$$e = \sum_{i=1} \left| k_i - o_i \right| \Big/ n \tag{4}$$

Total and mean summed errors are used because they do not rescale the results very much, and are seen to represent the most basic measure of error (Michalewicz, 1993; Banzhaf et al., 1998).

Two other versions of the error function rescale values in a manner often used with other forms of symbolic regression, such as OLS. The first calculates the sum of the squared differences between *O* and *K*, given by:

$$e = \sum_{i=1} \left(k_i - o_i \right)^2 \tag{5}$$

Squared error is useful in situations where the programmer wants to dampen small errors and highlight the effects of large errors. Conversely, the programmer may want to dampen the effects of large deviations by calculating the sum of square root of errors:

$$e = \sum_{i=1} \left(k_i - o_i \right)^{0.5} \tag{6}$$

Despite some debate (Banzhaf et al., 1998; Langdon, 1998), changing the error function has little discernable effect when applied to the decision-making problem explored here. An ANOVA analysis on ln ($f_{\bar{p}}$) for differences between error metrics demonstrated that they are not statistically significant (p = 0.3374, F = 1.13, df = 596). The SYPRIA framework uses sum-of-squared-errors formulation given its similarly to error functions used in other methods.

4.2 Creation

There are two basic ways of creating genetic programs. The first it to grow a full genetic program, which entails extending all of its branches outward from the root to achieve a uniform, maximum depth marked by terminals and everything above this depth as a function. Creating a variable tree allows its branches to vary in depth up to a preset maximum value. Full genetic programs are therefore symmetrical and uniform in appearance while variable trees are more ragged, since some branches have terminals much closer to the root. Full programs are useful where uniformity of trees is desired in order to reduce variation in eventual program size, while variable trees are designed to introduce diversity in the initial genetic population.

In addition to these two basic techniques, genetic programs are subject to the further twist of ***ramping***. Each population is divided into subpopulations, each of which is given a maximum depth. A population of ten full programs, for instance, could be divided into a ***ramped full population*** consisting of five subpopulations of two members each. The members of the first group would grow to a depth of two, the second a depth of four, and so on to the fifth group's members having a depth of ten. A ***ramped variable population*** has a maximum depth that ranges from two for the first pair, four for the second pair, and so on to the fifth group's members having a maximum depth of ten, as with a ramped full population, but branch length can vary. Finally, a population can be created with both methods, to create a ***ramped variable/full*** population. Ramping is meant to introduce genetic diversity into the population.

Between-group variation in genetic program fitness for creation methods is significant at the 99.99% level but not all pair-wise comparisons are significant, according to a one-way analysis of variance of ln ($f_{\bar{p}}$) (p = 0.0000, F = 10.61, df = 596). Several pairings are also significantly different from one another in their effects on genetic program fitness (Table 2).

In general, the largest differences occur between full or variable when either one is ramped. The only two combinations that are uniformly significantly different are ramped variable combined with either full or ramped full.

Creation type	**Likelihood of between-class variance by creation type (Difference [±] in ln ($f_{\bar{P}}$) and p-values [*p*])**							
	Variable		***Full***		***Ramped Full/Variable***		***Ramped Variable***	
	±	*p*	±	*p*	±	*p*	±	*p*
Full	-0.152	0.000						
Ramped Variable/Full	-0.058	0.486	0.094	0.023				
Ramped Variable	-0.024	1.000	0.129	0.000	0.035	1.000		
Ramped Full	-0.141	0.000	0.011	1.000	-0.083	0.079	-0.118	0.001

Table 2. Creation type: between-class variance

In terms of average population genetic program fitness, $f_{\bar{P}}$, there is a small, significant difference of means between the methods. Full tends to outperform variable somewhat but only ramped variable does noticeably less well than the other methods (Table 3). Note that for the standardized measure of fitness used in genetic programming, a lower fitness scores represents a fit individual. Koza (1992) finds that the ramped variable/full method works well with many instrumental problems, which he attributes to the early diversity created in the genetic program population by this method.

Creation Type	**Fitness (mean ln ($f_{\bar{P}}$))**
Variable	13.34
Full	13.19
Ramped Variable/Full	13.28
Ramped Variable	13.32
Ramped Full	13.20

Table 3. Creation type: mean fitness scores

The difference in effectiveness between varying combinations of full, variable, and ramped initial programs can also be seen in the histograms in Figure 3. The major difference between methods is that the apparent center of mass is lower with full programs, in accordance with Table 3, which denotes a greater number of more fit programs. At the same time, the full creation technique results in a non-trivial number of poorly performing programs.

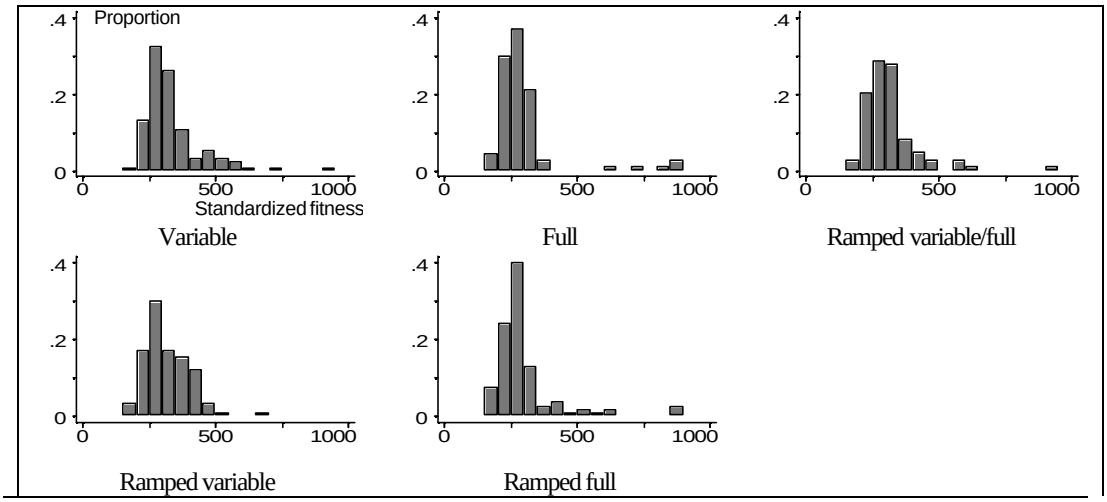

Figure 3. Creation type: effect on standardized fitness score

In addition to creating larger numbers of very poor programs, full tends to create noticeably longer genetic programs (Figure 4). The figure also illustrates how variable programs create a large proportion of relatively short programs, a few medium sized programs, and slightly more large programs. In contrast, full creates few short programs and instead results in a gradually increasingly greater proportion of long programs. The ramped variable/full combination attenuates the extreme tendencies of either the full or the variable method by ensuring that about a third of the programs are short.

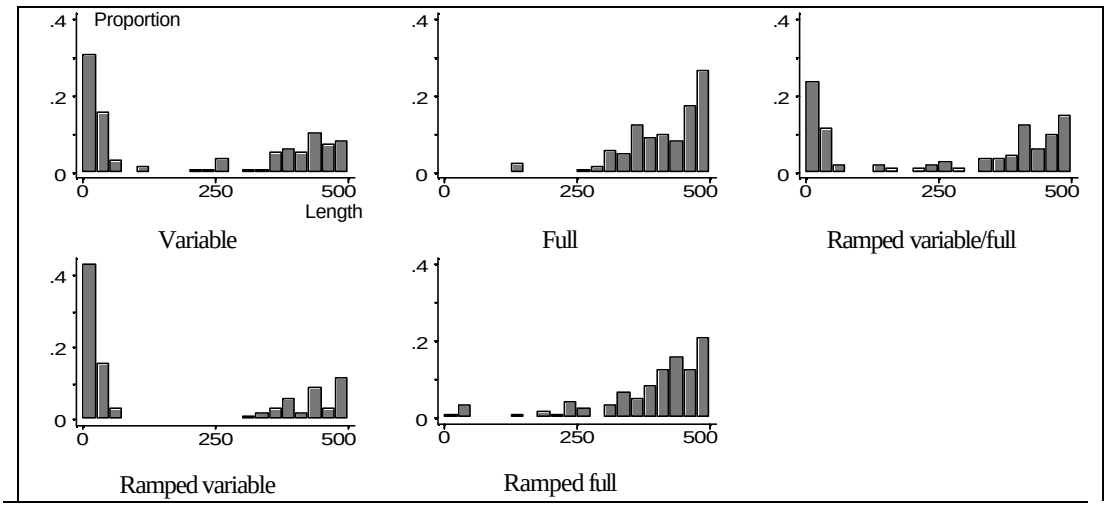

Figure 4. Creation type: effect on program length

The ramped variable/full creation method is used to create genetic programs for agent decision making in SYPRIA. This creation technique balances performance and genetic program length. While there is not a single length that can be said to be realistic in terms of decision making, shorter is probably better if genetic programs are meant to act as rules of thumb or have reduced length in order to be in accordance with the precepts of bounded rationality, since actors should be limited in their computational capacity. The bimodality in lengths that result from the various creation types also suggests a natural break point where the genetic program system should be brought to a halt after individual programs reach a

given length. Experiments with creation type suggest this limit is less than 100. This length represents reduced computational resources for decision making because the strategy is limited in its complexity, but it also allows an agent some room for deliberation and experimentation.

4.3 Selection

The success of genetic programming lies in having fit parents combine to create fit offspring. Over successive generations, the overall fitness of the population increases because parents should be more likely to have good fitness while those to be replaced should be more likely to have poor fitness. SYPRIA uses five functions: probabilistic, selection: ranked, tournament, elitist tournament, and demetic.

The first means of selection, ***probabilistic***, is the oldest in terms of the research literature and the most widely used. It was employed by Holland (1975) when developing the first genetic algorithm, where probability of being chosen, P_k, is:

$$P_k = f(k_j) / \sum_{j=1}^{n} f(k_j) \quad (7)$$

Each genetic program is considered in terms of its fitness as a proportion of the total fitness summed over the entire population. Programs are selected according to this proportion. A program accounting for a third of the population fitness, for example, has a 1 in 3 chance of being selected as a parent. The inverse operation is applied to selecting the individual to be replaced by the child of another.

The second form of selection is ***ranking*** (Goldberg, 1989). As the term implies, each member of the population is ranked according to its fitness against all other programs and then P_k is assigned as a function of this rank. The genetic programming environment then moves down the list, pairing off parents and using their children to replace the poor performers at the bottom of the list. In algorithmic terms, ranked probability P_k is as follows:

$$P_k = 1/n\left(p^- + \left(p^+ - p^-\right) i - 1/n - 1\right) \quad (8)$$

where p^- is the probability assigned to the worst individual and p^+ to the best. While poor-performing strategies are not entirely forgotten, choosing them is improbable. The key drawback of ranking selection is that successful genotypes can quickly take over a population and impart their distinctiveness on the entire population, leaving less room for innovation.

The third form of selection is the ***tournament*** method, where a small group of programs is chosen at random from the entire population. Each member in this group is compared to the others, and the program with the best fitness replaces that with the worst fitness. The number of programs composing a tournament group typically varies from two to twenty. Tournament selection offers a good way of controlling how much evolutionary pressure is placed on genetic programs. Low group size reduces the odds that poorly performing programs will by replaced while large groups increase the likelihood of this occurring. The fourth form of selection, ***elitist tournament***, is identical to the ordinary tournament technique but the most fit individual in the population is automatically placed in the next generation.

The final form of selection is through ***demetic*** grouping, an operation that is analogous to the evolutionary pressures found in island geography (Langdon, 1998). The population is divided into a series of subpopulations, or demes, and within each group, parents and children to be replaced are chosen at random. Evolutionary pressure is exerted in the form of migratory programs being probabilistically chosen from each deme and sent to another deme. Demetic grouping is useful for halting premature convergence in the

population, since each deme can evolve without pressure save for the occasional incursion of migratory programs. As with tournaments, the size of a deme can vary, with smaller demes leading to slower convergence.

Selection Type	Fitness by year (mean ln ($f_{\bar{p}}$))
Probability	13.68
Ranked	13.64
Tournament	13.39
Elite Tournament	13.41
Demetic	13.55

Table 4. Selection type: mean fitness scores

There is little agreement in the research literature on which selection technique is the best (Mitchell, 1996; Banzhaf et al., 1998). The control offered by tournament and demetic techniques is tempting but they are more complicated than ranking or probabilistic selection. Table 4 demonstrates that tournament methods, both regular and elite tournament, perform better than the other three kinds by margins wider than those seen in tests of creation method or fitness function. The demetic method lies in the middle while probability and ranked method do the least well. A one-way analysis of variance on ln ($f_{\bar{p}}$) is significant (p = 0.0000, F = 34.36, df = 596).

Table 5 illustrates differences between individual techniques and supports the conclusion that tournament selection is the most effective. There are highly significant differences between three groups: probability and ranked methods; both tournament methods; and demetic grouping. There is no appreciable difference between probabilistic and ranked methods, nor is there a real difference between elite tournament and regular tournament methods.

Selection type	Likelihood of between-class variance by selection type (Difference [±] in ln ($f_{\bar{p}}$) and p-values [*p*])							
	Probabilistic		***Ranked***		***Tournament***		***Elite Tournament***	
	±	*p*	±	*p*	±	*p*	±	*p*
Ranked	-0.041	1.000						
Tournament	-0.295	0.000	-0.253	0.000				
Elite Tournament	-0.273	0.000	-0.232	0.000	0.021	1.000		
Demetic	-0.137	0.000	-0.096	0.045	0.158	0.000	0.136	0.001

Table 5. Selection type: between-class variance

The differences between probability and ranking on the one hand and tournament and demetic methods on the other, are further illustrated by Figure 5. Ranked and probability methods produce highly leptokurtic fitness curves. The best-performing programs gain an early foothold within the population and quickly dominate other individuals. As a result, the average population fitness over repeated runs is poor because little evolutionary pressure is brought to bear on the population as a whole. In comparison, tournament and demetic methods show broader distributions that are, on average, lower scoring and therefore create programs of higher fitness.

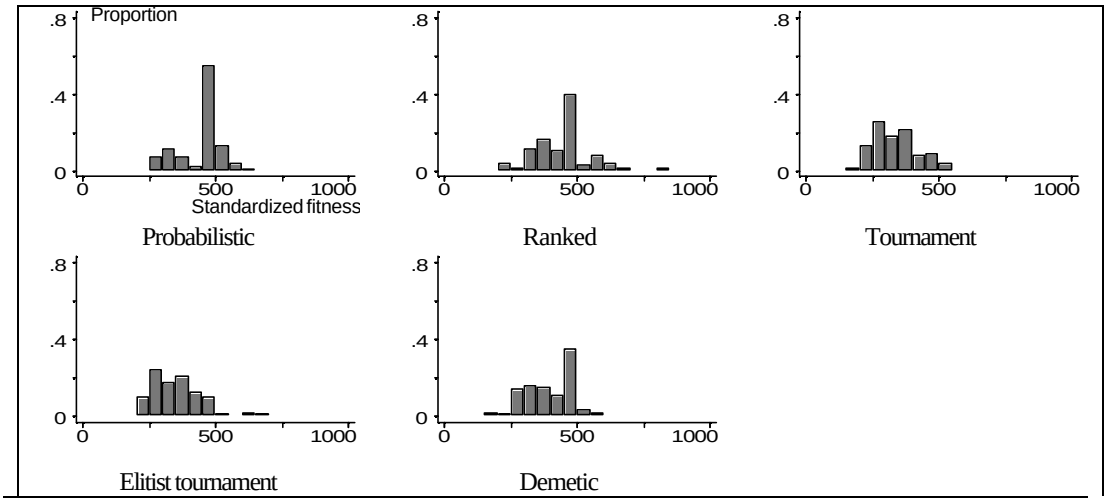

Figure 5. Selection type: effect on standardized fitness score

Figure 6 highlights another aspect of selection technique. Given the propensity for ranked and probability methods to create genetic programs that quickly dominate population, these early leaders tend to be far shorter since they have evolved less over time. As a result, the bulk of the population is dominated by short programs relatively early in terms of generations, which leads to relatively short offspring. In contrast, tournament and demetic selection produce less extreme distributions, as individuals are more diverse.

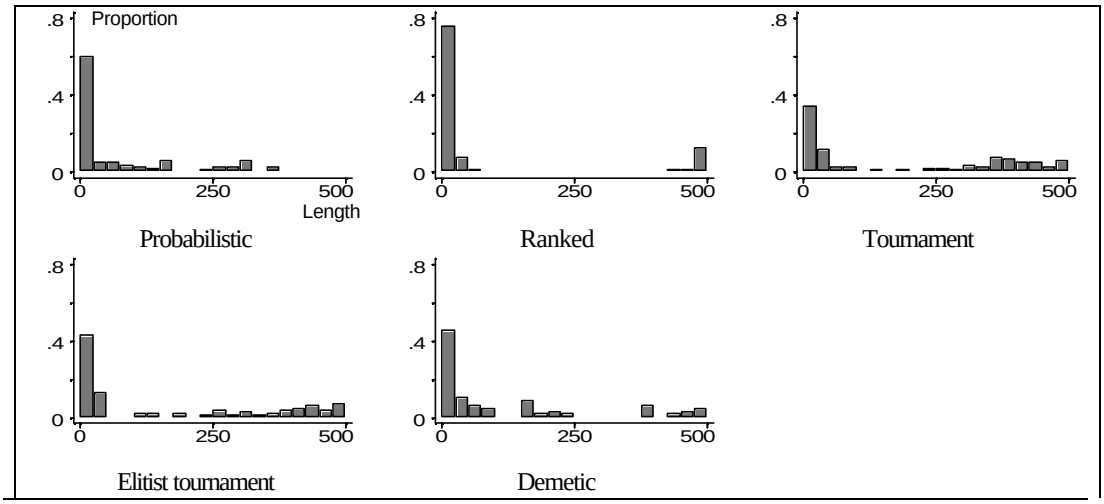

Figure 6. Selection type: effect on program length

The choice of selection method is not necessarily clear, but there are some general considerations to bear in mind. Ranked and probabilistic methods are less appropriate candidates for decision-making analogs given that they produce poor distributions in terms of length and fitness. Early solutions are adopted at a higher rate than with other selection methods, precluding a good deal of potential strategies. Not only does this seem to overly constrain the nature of possible strategies, these methods limit the amount of exploration accorded to the genetic program system in order to pursue exploitation, and therefore do not accord well with bounded rationality.

This leaves demetic and tournament techniques. According to the mental interpretation, each agent possesses a cadre of competing strategies that represent a set of ideas floating around in the mind of the actor. The actor experiments with these strategies, either directly or by imagining how strategies would fare according to what the actor knows of its environment. Tournament selection and elitist tournaments map nicely onto this construct since they involve comparison of a small group of programs. This is in keeping with bounded rationality, where actors can only directly consider a few strategies at one time. Demetic selection is intriguing but complicates the potentially tenuous relationship between actual decision making and genetic programming as a model of decision making. Additionally, research on demetic selection is still undergoing and its behavior is not as well understood as that of other methods. For these reasons, SYPRIA typically uses tournament selection methods for agent decision making.

4.4 Population

Under the mental interpretation of using genetic programs to represent decision making, a population of programs represents a collection of strategies possessed by a single agent. Population has an intuitive effect on fitness scores given that a higher initial population grants the population a higher probability of possessing good solutions and greater diversity of candidate solutions. Population sizes of 10 up to 1,000,000 are used in a variety of applications, although the recommended size for many problems ranges from 300 to 500 for smaller problems (Koza, 1992) and up to 1000 to 10,000 for very complex problems (Banzhaf et al., 1998). As noted by both of these sources, however, there is little general research on choosing optimal population size.

For the problem explored here, there is a linear, positive relationship between genetic program population and fitness, keeping in mind that fitness is better with lower fitness scores. A linear regression for fitness, ln ($f_{\bar{p}}$), as a function of population size yields an R^2 of 0.488 (coefficient = -.000924, p = 0.0000, t = -23.871, df = 599). The influence of population size on fitness may be even stronger in tests with larger sample sizes since, as shown by Figure 7, there are outliers. The relationship may also be stronger with larger population sizes, as 500 is still a modest value for the population parameter. Nonetheless, the figure demonstrates the generally positive relationship between population and fitness found in the regression analysis.

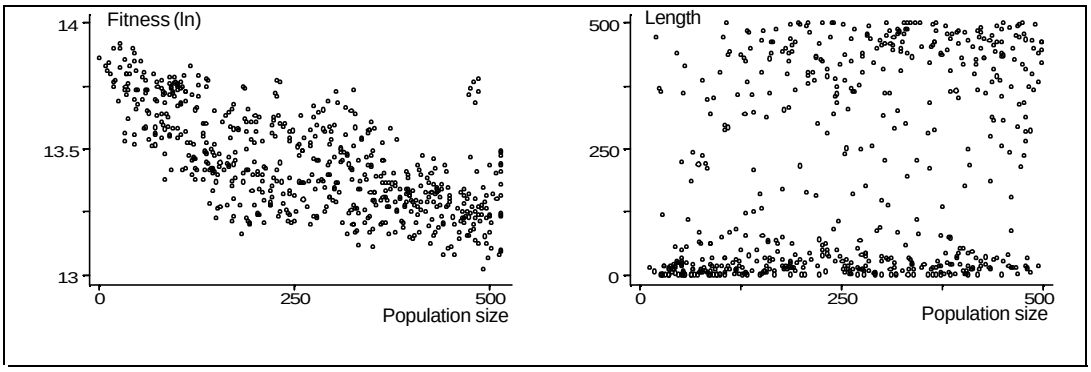

Figure 7. Population: effect on fitness score and program length

Figure 7 also illustrates the relationship between population size and length. There is no clear linear relationship between the two, which reflects the bimodality of the ramped variable/full technique used for this batch of tests. There is, however, an apparent

relationship between the two in that clustering indicates that smaller populations tend to be dominated by shorter programs. This is in accordance with the general principle that there is a greater likelihood of longer programs as population size grows. Relatively fit programs that appear in early generations quickly dominate the population because they face less competition from other programs, which in turn are subject to greater overall selection pressure. With larger populations, there is less competitive pressure overall and therefore more latitude for larger programs to develop and eventually have higher fitness scores than smaller programs.

Given the weak isomorphism between decision-making theory and genetic programming, there is no obvious way to assert that a given population size best represents actual human cognition. Does the average person have 100 strategies or a million? Does each genetic program represent part of a single actual strategy or a strategy in and of itself? The SYPRIA system falls back on the general bounded-rationality precept of limited computational resources. A value of 300 is the lowest realistic value for most applications, and this value is therefore used by the SYPRIA model. Higher values would move the agents away from representing boundedly rational actors and towards actors with infinite search capacity, as under perfect rationality.

4.5 Generations

The number of generations, in theory, works much in the same way as population size, in that a greater number makes it more likely that a better solution is produced by the population of genetic programs at the risk of overtraining (Kushchu, 2002). There is a weak, yet significant positive relationship between genetic program generation size and fitness. Linear regression results for fitness on ln ($f_{\bar{p}}$) against population size yields an R^2 of 0.110 (coefficient = -.00173, p = 0.0000, t = -8.578, df = 599). As indicated by Figure 8, the effect of generation size is marked for roughly the first 30 generations and then becomes less noticeable afterwards, which partially accounts for the weak relationship between generation size and fitness.

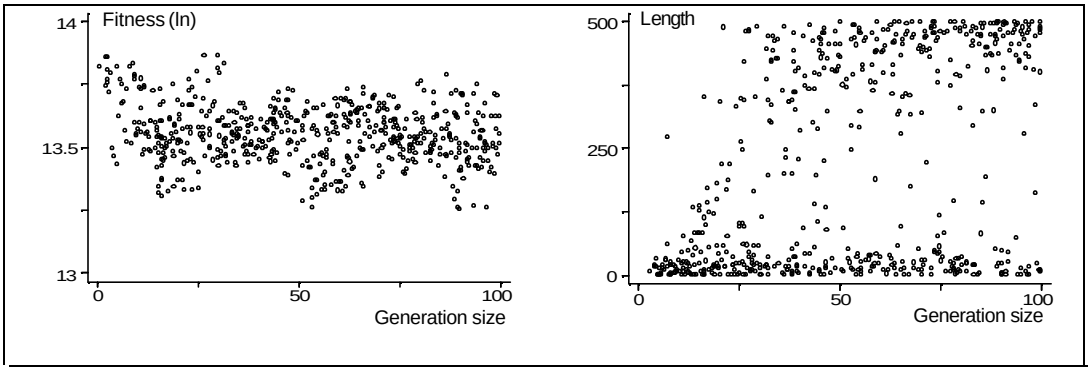

Figure 8. Generation size: effect on fitness score and program length

Program length generally increases with the number of generations, as illustrated by Figure 8. This relationship is more apparent in the case of the genetic program system depicted in Figure 9, which depicts length and fitness of the fittest member of the population in every generation. The fitness of this member increases with the number of generations but its length also increases rapidly before leveling off.

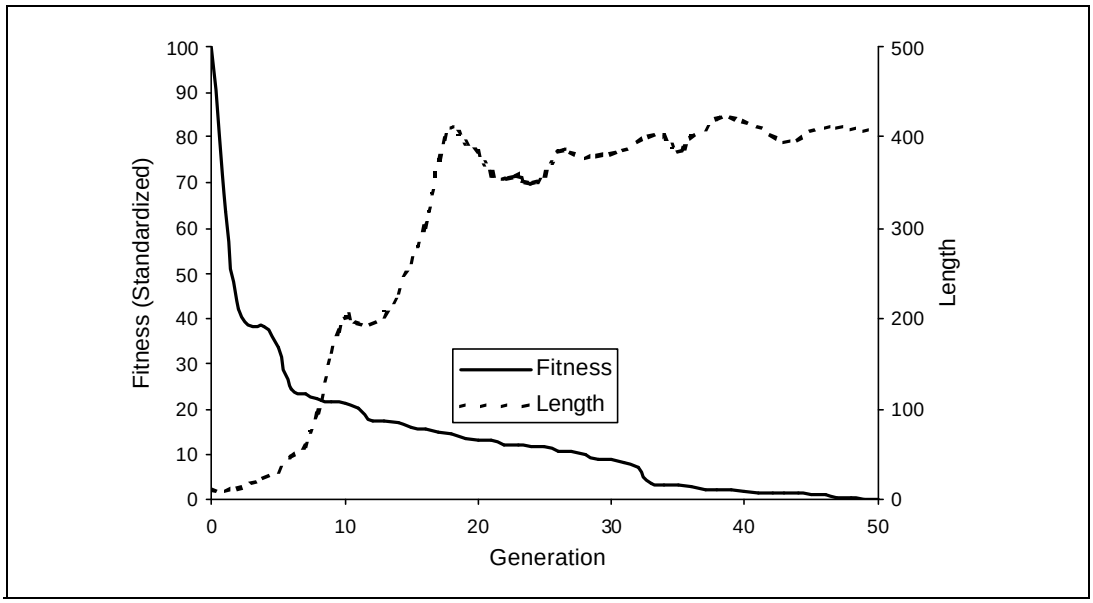

Figure 9. Generation size: effect on fitness and program length

There are exceptions, however, where evolution will suddenly resume after a long period of quiescence in to the thousands of generations. The amount of computational effort required to mount so many generations may make it not worth the time and is not considered here since bounded rationality suggests that actors have limited computational resources. Figure 9 does not tell the whole story of the relationship between generations and length because the length of programs in later generations is pruned according to settings mandating the maximum genetic program length and maximum depth at which crossover may occur. A ***length error*** occurs when mutation or crossover between two parents creates a genetic program that is too long, typically 500 genes. A ***crossover error*** occurs when the depth at which the randomly selected crossover point in either of the parents is too deep, typically 17 branches. Parents continue to create and discard offspring until one is created that does not exceed the maximum program length or maximum crossover depth. This error-free program becomes the single, official offspring of the parents. The removal of genetic programs committing length and crossover errors is the chief reason why the length of the best genetic program depicted in Figure 9 levels off and is partially responsible for the diminishing returns of generations in terms of fitness. As Figure 10 illustrates, the number of length errors committed by a single genetic program run is linearly related to generation size.

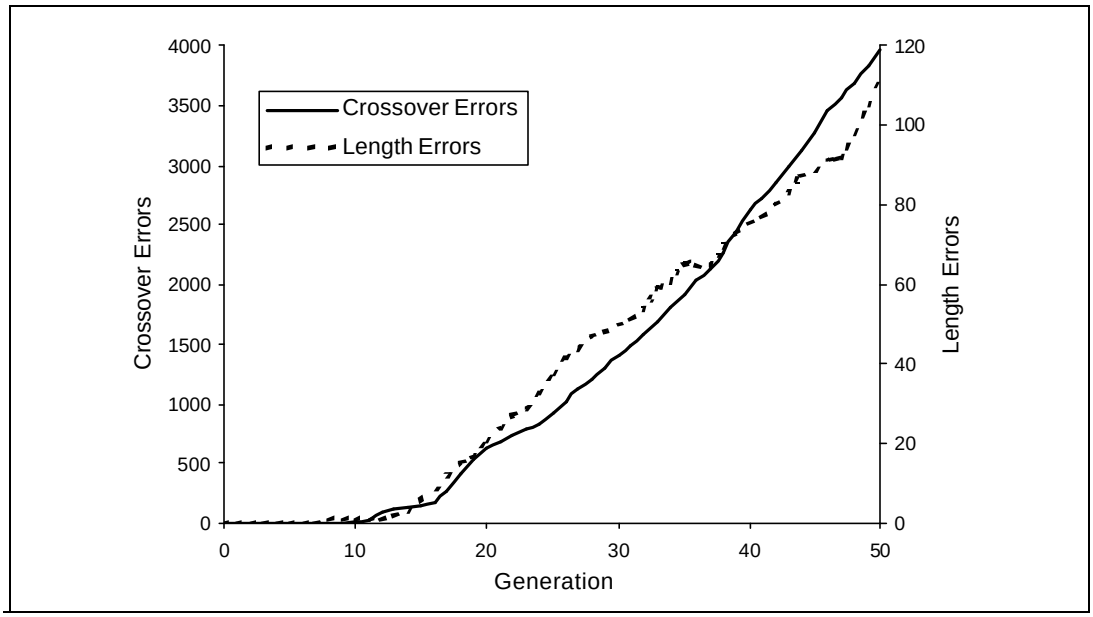

Figure 10. Generation size: effect on length errors and crossover errors

In terms of generation size and SYPRIA, many of the arguments made for a low population value hold for generation size, especially the bounded-rationality dictate of limited computational resources. Values of 30 to 50 are the lowest recommended values in general and they are adopted by the SYPRIA model. As above, higher values would make for more of an instrumental search on the part of agents. Another rationale for low generation values is that they are less likely to create long genetic programs. In choosing the tournament selection technique over ranking or probabilistic methods, for instance, the one advantage of the latter techniques is their propensity to create shorter programs. Limiting generations, and population size, is one way to limit the length of resultant programs and stay in keeping with bounded rationality. This is seen in Figure 8, where there are very few long programs before generation 30 and a limited number before generation 50. Figure 9 also demonstrates this somewhat, but is less representative since it only applies to one model run while Figure 8 is for hundreds of runs.

5. Conclusion

Land change research increasingly employs computational land change models as a means of combining theory, method, and data. Modeling methods are expanding to encompass those stemming from computational intelligence, such as genetic programming, cellular modeling, and agent-based modeling. The SYPRIA model demonstrates the use of a computational intelligence method, genetic programming, to model boundedly rational decision making in the context of human-environment relationships. In doing so, it addresses theoretical and methodological needs in human-environment modeling, particularly with respect to decision making. Importantly, it offers a way to model individual actors and their decision-making in a manner that addresses issues of memory, optimization, complexity of strategies, learning, innovation, and communication with other agents. This work has ramifications for geocomputation and its cognate fields of human-environment modeling, computational intelligence, decision sciences, and complexity research.

6. Acknowledgements

This work is supported by the National Aeronautics and Space Administration (NASA) Earth System Science Fellowship program (ESS 99-0000-0008) and a National Science Foundation Doctoral Dissertation Improvement grant (NSF 99-07952). It is also supported by NASA's Land-Cover and Land-Use Change program through the Southern Yucatán Peninsular Region project (NAG 56406 and NAG 511134) and the Center for Integrated Studies of Global Environmental Change, Carnegie Mellon University (NSF-SBR 95-21914). The author gratefully acknowledges the assistance of SYPR project personnel.

7. References

Agarwal, C., G. L. Green, J. M. Grove, T. Evans and C. Schweik (2002). *A Review and Assessment of Land-Use Change Models: Dynamics of Space, Time, and Human Choice*, Center for the Study of Institutions Population and Environmental Change (Indiana University Bloomington Indiana) and the USDA Forest Service Northeastern Forest Research Station (Burlington Vermont).

Andreoni, J. and J. H. Miller (1995). Auctions with artificial adaptive agents. Games and Economic Behavior 10: 39-64.

Arifovic, J. (1994). Genetic algorithm learning and the cobweb model. Journal of Economic Dynamics and Control 18: 3-28.

Arthur, W. B. (1993). On designing economic agents that behave like human agents. Journal of Evolutionary Economics 3(1): 1-22.

Balling, R. J., J. T. Taber, M. Brown and K. Day (1999). Multiobjective urban planning using a genetic algorithm. ASCE Journal of Urban Planning and Development 125(2): 86-99.

Banzhaf, W., P. Nordin, R. E. Keller and F. D. Francone (1998). *Genetic Programming: An Introduction*. San Francisco, California, Morgan Kauffman Publishers.

Beckenbach, F. (1999). Learning by genetic algorithms in economics. In (Ed. T. Brenner. *Computational Techniques for Modelling Learning in Economics*. Boston, Massachusetts, Kluwer Academic Publishers, pp. 73-100.

Bosch-Domenech, A. and N. J. Vriend (1998). *Do Boundedly Rational People Imitate?* London, Working Paper No. 379 Queen Mary and Westfield College Department of Economics.

Brenner, T. (1999). *Computational Techniques for Modelling Learning in Economics*. Boston, Massachusetts, Kluwer Academic Publishers.

Brown, D. G., R. Walker, S. Manson and K. Seto (2004). Modeling land use and land cover change. In Eds.), G. Gutman, A. Janetos, C. Justice, E. Moran, J. Mustard, R. Rindfuss, D. Skole and B. L. Turner, II. *Land Change Science: Observing, Monitoring, and Understanding Trajectories of Change on the Earth's Surface*. Dordrecht, Netherlands, Kluwer Academic Publishers, pp. 395-409.

Carroll, J. S. and E. J. Johnson (1990). *Decision Research : A Field Guide*. Newbury Park, New Jersey, Sage Publications.

Chattoe, E. (1998). Just how (un)realistic are evolutionary algorithms as representations of social processes? Journal of Artificial Societies and Social Simulation 1(3): Section 2.

--- (1999). Review of computational techniques for modelling learning in economics. *Advances in Computational Economics*, pp. 585-91.

Chen, S.-H. (2001). On the relevance of genetic programming to evolutionary economics. In (Ed. K. Aruka. *Evolutionary Controversy in Economics towards a New Method in Preference of Trans Discipline*. Tokyo, Springer-Verlag.

--- (2003). Fundamental issues in the use of genetic programming in agent-based computational economics. In Eds.), A. Namatame, T. Terano and K. Kurumatani. *Agent-Based Approaches in Economic and Social Complex Systems*. Amsterdam, IOS Press.

Chen, S. H. and B. T. Chie (2004). Functional modularity in the fundamentals of economic theory: toward an agent-based economic modeling of the evolution of technology. International Journal of Modern Physics B 18(17-19): 2376-86.

Collins, M. G., F. R. Steiner and M. J. Rushman (2001). Land-use suitability analysis in the United States: historical development and promising technological achievements. Environmental Management 28(5): 611-21.

Conlisk, J. (1996). Bounded rationality and market fluctuations. Journal of Economic Behavior & Organization 29(2): 233-50.

Cook, K. S. and M. Levi (1990). *The Limits of Rationality.* Chicago, Illinois, University of Chicago Press.

Dawid, H. (1997). Learning of equilibria by a population with minimal information. Journal of Economic Behavior and Organization 6: 361-73.

--- (1999). *Adaptive Learning by Genetic Algorithms: Analytical Results and Applications to Economic Models*. Berlin, Springer.

Diplock, G. (2000). Genetic programming: a new approach to spatial model building. In Eds.), S. Openshaw and R. J. Abrahart. *GeoComputation*. London, Taylor and Francis, pp. 220.

Dosi, G. and R. R. Nelson (1994). An introduction to evolutionary theories in economics. Journal of Evolutionary Economics 4(3): 153-73.

Edmonds, B. (1999). Modelling bounded rationality in agent-based simulations using the evolution of mental models. In (Ed. T. Brenner. *Computational Techniques for Modelling Learning in Economics*. Boston, Massachusetts, Kluwer Academic Publishers, pp. 305-32.

--- (2001). Towards a descriptive model of agent strategy search. Computational Economics 18(1): 111-33.

Eiben, A. E. and M. Schoenauer (2002). Evolutionary computing. Information Processing Letters 82(1): 1-6.

Geist, H. and E. Lambin (2001). *What Drives Tropical Deforestation? A Meta-Analysis of Proximate and Underlying Causes of Deforestation Based on Subnational Case Study Evidence*. Louvain-la-Neuve, Belgium, LUCC International Project Office.

Gigerenzer, G., R. Selton and R. Selton (2001). *Dahlem Workshop Reports; Bounded Rationality: The Adaptive Toolbox.* Cambridge, Massachusetts, MIT Press.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading, Massachusetts, Addison-Wesley.

Holland, J. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Cambridge, Massachusetts, MIT Press.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor, Michigan, University of Michigan Press.

--- (1995). *Hidden Order: How Adaptation Builds Complexity*. Reading, Massachusetts, Addison-Wesley.

Irwin, E. and J. Geoghegan (2001). Theory, data, methods: developing spatially explicit economic models of land use change. Agriculture, Ecosystems, and Environment 85: 7-23.

Janssen, M. (2003). *Complexity and Ecosystem Management: The Theory and Practice of Multi-Agent Approaches*. Northampton, Massachusetts, Edward Elgar Publishers.

Kaboudan, M. A. (2003). Forecasting with computer-evolved model specifications: a genetic programming application. Computers & Operations Research 30(11): 1661-81.

Korkmaz, E. E. and G. Ucoluk (2004). A controlled genetic programming approach for the deceptive domain. IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics 34(4): 1730-42.

Koza, J. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, Massachusetts, MIT Press.

Krzanowski, R. and J. Raper (2001). *Spatial Evolutionary Modeling.* Oxford, United Kingdom, Oxford University Press.

Kushchu, I. (2002). Genetic programming and evolutionary generalization. IEEE Transactions on Evolutionary Computation 6(5): 431-42.

Langdon, W. B. (1998). *Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming*. Dordrecht, Netherlands, Kluwer Academic Publishers.

Manson, S. M. (2000). *Agent-based dynamic spatial simulation of land-use/cover change in the Yucatan peninsula, Mexico*. Fourth International Conference on Integrating GIS and Environmental Modeling (GIS/EM4), Banff, Alberta.

--- (2004a). *Boundedly rational land-use strategies and the local dimensions of human vulnerability and resilience in the Yucatan peninsula*. Meeting of the Latin American Studies Association, Las Vegas, NV.

--- (2004b). The SYPR integrative assessment model: complexity in development. In Eds.), B. L. Turner, II, D. Foster and J. Geoghegan. *Integrated Land-Change Science and Tropical Deforestation in the Southern Yucatan: Final Frontiers*. Oxford, United Kingdom, Clarendon Press, pp. 271-91.

--- (2005a). Agent-based modeling and genetic programming for modeling land change in the Southern Yucatan Peninsular Region of Mexico. Agriculture, Ecosystems & Environment (In press).

--- (2005b). Land use in the Southern Yucatan Peninsular Region of Mexico: scenarios of population and institutional change. Computers, Environment, and Urban Systems (In press).

Matthews, K. B., A. R. Sibbald and S. Craw (1999). Implementation of a spatial decision support system for rural land use planning: integrating geographic information system and environmental models with search and optimisation algorithms. Computers and Electronics in Agriculture 23(1): 9-26.

Michalewicz, Z. (1993). *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin, Springer Verlag.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. Cambridge, Massachusetts, MIT Press.

Myers, G. M. and Y. Y. Papageorgiou (1991). Homo economicus in perspective. The Canadian Geographer 35(4): 380-99.

Parker, D. C., S. M. Manson, M. Janssen, M. J. Hoffmann and P. J. Deadman (2003). Multi-agent systems for the simulation of land use and land cover change: a review. Annals of the Association of American Geographers 93(2): 316-40.

Pingle, M. (1995). Imitation versus rationality: an experimental perspective in decision making. Journal of Socio-economics 24: 281-315.

Ralston, A. and P. Rabinowitz (2001). *A First Course in Numerical Analysis*. New York, Dover Publications.

Riechmann, T. (1999). Learning and behavioural stability: an economic interpretation of genetic algorithms. Journal of Evolutionary Economics 9(2): 225-42.

Simon, H. A. (1997). Behavioral economics and bounded rationality. In (Ed. H. A. Simon. *Models of Bounded Rationality*. Cambridge, Massachusetts, MIT Press. Vol. 1; Vol. 1, pp. 267-433.

Tesfatsion, L. (2002). Agent-based computational economics: growing economies from the bottom up. Artificial Life 8(1): 55-82.

van den Bergh, J. C. J. M., A. Ferrer-i-Carbonell and G. Munda (2000). Alternative models of individual behaviour and implications for environmental policy. Ecological Economics 32(1): 43-61.

Verburg, P. H., P. Schota, M. Dijsta and A. Veldkamp (2005). Land use change modelling: current practice and research priorities. GeoJournal (In press).

Vriend, N. J. (2000). An illustration of the essential difference between individual and social learning, and its consequences for computational analyses. Journal of Economic Dynamics and Control 24: 1-19.

Xiao, N. C., D. A. Bennett and M. P. Armstrong (2002). Using evolutionary algorithms to generate alternatives for multiobjective site-search problems. Environment and Planning A 34(4): 639-56.