Taylor & Francis
Taylor & Francis Group

Research Article

Bounded rationality in agent-based models: experiments with evolutionary programs

S. M. MANSON*

Department of Geography, University of Minnesota, 414 SST, 267 19th Ave South, Minneapolis, MN 55455, USA

This paper examines the use of evolutionary programming in agent-based modelling to implement the theory of bounded rationality. Evolutionary programming, which draws on Darwinian analogues of computing to create software programs, is a readily accepted means for solving complex computational problems. Evolutionary programming is also increasingly used to develop problem-solving strategies in accordance with bounded rationality, which addresses features of human decision-making such as cognitive limits, learning, and innovation. There remain many unanswered methodological and conceptual questions about the linkages between bounded rationality and evolutionary programming. This paper reports on how changing parameters in one variant of evolutionary programming, genetic programming, affects the representation of bounded rationality in software agents. Of particular interest are: the ability of agents to solve problems; limits to the complexity of agent strategies; the computational resources with which agents create, maintain, or expand strategies; and the extent to which agents balance exploration of new strategies and exploitation of old strategies.

Keywords: Agent-based model; Bounded rationality; Evolutionary programs; Land change

1. Introduction

Human transformation of the Earth's land surface has environmental and socio-economic impacts that extend from specific locales to the entire globe. We know surprisingly little about this land change because it is a complex phenomenon caused by individual actors, such as households or firms, within larger environmental and social systems (NRC 2001). Models that integrate human decision-making, societal context, and ecological relations are central to understanding land change (Brown *et al.* 2004, Gutman *et al.* 2004). Nonetheless, relatively few land-change models actively consider theories of human behaviour (Irwin and Geoghegan 2001). This work instead has tended to address 'highly empirical questions using statistical methods … and has generally avoided abstract theoretical formulations' (Walker 2004: 248). There is a need for a broader conceptualization of individual decision-making in land-change models as such and in the cognitive and social sciences more generally (Agarwal *et al.* 2002).

*Corresponding author. Email: manson@umn.edu

   Agent-based models (also termed multi-agent systems or individual-based models) are increasingly used to represent decision-making in land-change contexts. An agent-based model is a system of semi-autonomous software programs, or 'agents', that represent the complex behaviour of interacting entities such as individuals, households, or firms. Agent-based models advance our understanding of decision-making by representing actors, such as individuals or households, as autonomous, heterogeneous, and locally interacting entities. To fully leverage these advantages, however, modellers must also invest agents with theoretical and methodological representations of the decision-making that drives land change (Parker *et al.* 2003).

   This research explores the use of evolutionary programming to represent decision-making in agent-based models. It uses the SYPR Integrated Assessment (SYPRIA) model, named for the southern Yucatán peninsular region of Mexico, which is home to deforestation and attendant cultivation that threatens one of the world's largest remaining subhumid tropical forests. The chief proximate cause of this land change is household agricultural activity. To represent these households and their decision-making, SYPRIA agents make land-use decisions in a simulated landscape, decisions that account for individual, social, and environmental context. Other studies describe SYPRIA in general and its capacity for scenario generation (Manson 2000, 2004, 2005, 2006).

   This paper focuses on linking the theoretical and technical imperatives of decision-making research in agent-based models of land change. SYPRIA agents are invested with a form of evolutionary programming, termed genetic programming that serves as a computational analogue to real household decision-making. This paper examines how changes in genetic programming parameters affect the representation of a specific decision-making theory, bounded rationality, in agents. Section 2 examines how agent decision-making is a form of multicriteria evaluation that assesses the suitability of land for agriculture. Agents treat their multicriteria evaluation question as a symbolic regression problem that they solve with evolutionary programming. This approach offers important methodological and conceptual advantages when representing decision-making in agent-based models of land change. Section 3 describes how changes in genetic programming parameters affect the ability of agents to act in accordance with bounded rationality.

2.   Modelling agent decision-making

Agents in many agent-based models of land change choose locations in a simulated landscape for activities such as clearing forest and planting crops (Gimblett 2002, Janssen 2003, Parker *et al.* 2003). Agents can treat this decision-making problem as a form of multicriteria evaluation (Collins *et al.* 2001), where each agent creates strategies to choose locations in the simulated landscape according to their suitability for any given production activity as a function of spatial factors that vary over the landscape (Manson 2005). A key challenge in designing software agents that represent real decision-makers is determining the manner in which each agent solves its multicriteria evaluation problem. When agents use evolutionary programming to solve this problem as a form of symbolic regression, the modeller gains a useful computational tool and a means of representing agent decision-making as a form of bounded rationality.

2.1 *Multicriteria evaluation as a symbolic regression problem*

Agents can treat their multicriteria evaluation problem as a *symbolic regression*. Symbolic regression is empirical modelling that involves inductively building a mathematic or computational statement of relationships among random variables. Symbolic regression approximates an ideal function $f(x)$ that reproduces the value of a dependent variable Y as a function of independent predictor variables $\mathbf{X} = \{X_1, \ldots, X_n\}$. These variables are known solely through empirical observations $\mathbf{O} = \{\mathbf{y}, \mathbf{x}\}$ where $\mathbf{y} := (y_i)_m$, a column vector of m observations on Y, and $\mathbf{x} := (x_{i,j})_{m \times n}$, an $m \times n$ matrix of m observations on the n independent variables $\mathbf{X}$. Any given observation of the m observations in $\mathbf{O}$ is denoted o_i or $\{y_i, x_i\}$. The value of $f(x)$ at x_i, or $f(x_i)$, is denoted $\bar{f}_i$ and is related to the ideal value f_i through error defined as $\varepsilon_i = f_i - \bar{f}_i$. Consider a statistical linear regression, for example, that takes the familiar functional form

$$
\begin{aligned}
Y &= f(x) \\
&= f(x;\ \beta) \\
&= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon
\end{aligned}
\tag{1}
$$

where dependent variable Y is expressed as a linear function of independent variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, coefficients $\boldsymbol{\beta} = \{\beta_1, \ldots, \beta_n\}$, and error term ε.

Generic symbolic regression does not specify a functional form for $f(x)$, but instead combines primitive functions that range from simple arithmetic operators to specialized domain-specific functions. Symbolic regression approximates $f(x)$ as $\hat{f}(x)$

$$
\hat{f}(x) \approx \sum_{i=1}^{n} \alpha_i \phi_i(x)
\tag{2}
$$

comprising n pairings of function $\phi(x)$ with coefficient α estimated to minimize ε over the values of dependent and independent variables given by observations $\mathbf{O} = \{\mathbf{y}, \mathbf{x}\}$ (after Ralston and Rabinowitz 2001). The means used to estimate $f(x)$ vary according to the restrictions imposed by its functional form. In the case of the statistical linear regression, we can employ an approach such as ordinary least squares. Other versions of symbolic regression require specialized methods such as computational intelligence or machine learning to compose independent variables, coefficients, and primitive functions in a way that minimizes the error of $\hat{f}(x)$ with respect to observations $\mathbf{O}$ (Russell and Norvig 1995).

In the case of SYPRIA, the dependent variable Y maps onto actual land use, and the independent variables $\mathbf{X}$ correspond to social and environmental factors. The propensity for real households to clear land for extensive agriculture in the Southern Yucatán is a function of factors related to environmental systems (e.g. water, soil, precipitation) and social drivers (e.g. distance to market, land tenure). As proxies to actual households, each software agent must approximate its own $f(x)$ to establish the suitability of any given location as a function of spatially varying social and environmental factors. Each of the m observations in $\mathbf{O}$ gives the value of the dependent and independent variables for a single location in the area surrounding each household.

In SYPRIA, a different agent represents each of the roughly 5000 real households in the Southern Yucatán. These agents sample nearby locations in the simulated landscape, represented by GIS layers based on real data, to extract values for the

dependent and independent variables found in observations **O**; this formulation is similar to that found in other agent-based models of land change (cf. Polhill *et al.* 2001, Evans and Kelley 2004, Parker and Meretsky 2004, Brown *et al.* 2005). For the purposes of this paper, the dependent variable is land use observed in 1998 as derived from Thematic Mapper remotely sensed imagery. Independent variables include soil type, elevation, slope, aspect, precipitation, surface hydrology, distance to roads and markets, and socio-economic and demographic factors for the same year (Manson 2005). While the general characteristics of these data and the larger model structure are given here as background, these data and most specifics of SYPRIA are exogenous to the analysis of the linkages between evolutionary programming and bounded rationality, and as such are not detailed beyond what is necessary here.

2.2 *Symbolic regression and evolutionary programming*

Agent-based models in which agents conduct symbolic regression use approaches including statistics, linear programming, rules and heuristics, vector weighting schemes, belief–desire–intention models, and classifier systems (Tesfatsion 2001, Parker *et al.* 2003). Agents can also solve their respective multicriteria evaluation problems through *evolutionary programming*, or the creation of software programs through computational means inspired by biological processes, particularly Darwinian evolution (Eiben and Schoenauer 2002).

Genetic programs are trees composed of terminals and functions in a branching structure. In terms of the underlying biological metaphor, a tree that constitutes a genetic program is equivalent to that program's genetic code, and each function and terminal is analogous to a gene. Consider a simple tree that corresponds to the symbolic regression equation $Y = \beta_1 X_1 + \beta_2 X_2$ (figure 1(*a*)). This equation can be represented by a tree that possesses two branches, one composed of the terminals β_1 and X_1, and another by β_2 and X_2. These sub-branches are joined by the multiplication function, $\times$, implicit in the regression equation $Y = \beta_1 \times X_1 + \beta_2 \times X_2$. These sub-branches in turn are joined by the addition function, $+$, at the *root* node of the tree. Tree *depth* is the number of branches spanning the longest descent from the root to terminals, while *length* is the total number of nodes (figure 1(*a*)). Terminals such as β_1 and X_1 are, respectively, numerical coefficients and spatial factors corresponding to the variables in **X**, while functions can range from simple arithmetic operators to more complex operators. Functions tend to be restricted to a small set because the genetic programs will evolve sophisticated functions themselves, just as simple machine languages can undergird complicated programs (Banzhaf *et al.* 1998). The tree structure is useful because it can be interpreted visually while being functionally equivalent to a generic computer program (Koza 1992).

A *genetic programming system* manages individual programs, exerting pressure on the programs to evolve over time to become better solutions to agent symbolic regression problems. Each agent possesses a genetic programming system that controls genetic program population K of size M that evolves over generations G (after Koza 1992). Members of the initial population ($g=0$) comprise randomly chosen terminals and functions. Members of this generation compete for the opportunity to pass on their genetic material to the next generation ($g=1$), which in turn aim to do the same for the next ($g=2$), and so on. Each generation serves as the
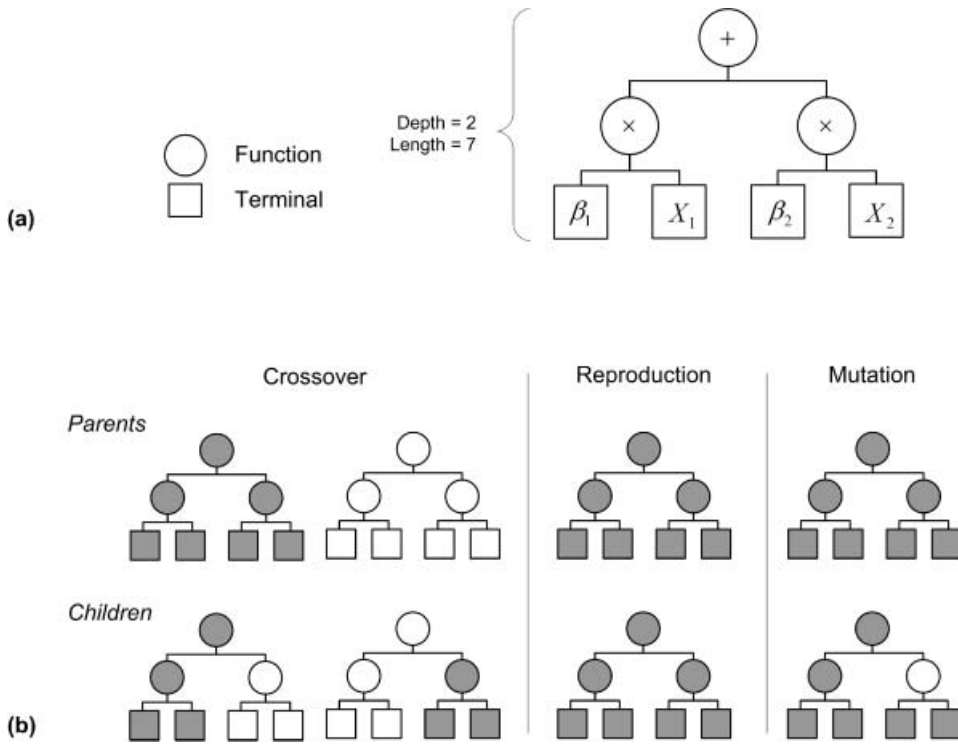
Figure 1.   Genetic program (*a*) and crossover, reproduction, and mutation operators (*b*).

'children' of the preceding generation and the 'parents' for the next. This evolution may involve as few as 10 generations or upwards of several thousand.

The genetic programming system chooses parents to create offspring through *selection*, whereby fit parents are more likely to have children. SYPRIA employs a steady-state approach, where population size is fixed over time by replacing less fit programs in the parent generation with the children of fitter parents. The system selects parents according to their *fitness*, f_k, in approximating function $\hat{f}(x)$ as given by minimizing error over observations **O**. The most common fitness function in genetic programming is summed error, or the sum of the absolute value of the differences between genetic program output and the dependent variable given by **O** (Koza 1999). Variants calculate the mean error in order to include the effect of sample size, sum of the squared errors, or the sum of square root of error (examined below and given in table 1). From this follows the sometimes confusing convention that a lower fitness score indicates a greater ability to solve the symbolic regression problem.

Three genetic operators create child genetic programs from fit parents to replace less fit parents: crossover, reproduction, and mutation (figure 1(*b*)). *Crossover* mingles the genes of two parents by creating a single child that carries a mixture of parental genes. Crossover is the driving force behind evolution because it quickly culls substandard strategies from the population while creating potentially better strategies through the intermingling of well-performing genetic programs. *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

Table 1. Genetic program parameter settings.

Parameter	Value (default)
Tested parameters	
Fitness function: error ε measured by $f(x_i)$, denoted $\bar{f}_i$, and the ideal value f_i given by y_i in $\mathbf{o}=\{\mathbf{y}, \mathbf{x}\}$	$\varepsilon = \sum_{i=1}^{m} \lvert f_i - \bar{f}_i \rvert$ $\varepsilon = \sum_{i=1}^{m} \lvert f_i - \bar{f}_i \rvert / n$ $\varepsilon = \sum_{i=1}^{m} (f_i - \bar{f}_i)^{2a}$ $\varepsilon = \sum_{i=1}^{m} (f_i - \bar{f}_i)^{0.5}$
Creation method: initial profile of programs	Full, Variable, Ramped Variable, Ramped Full, Ramped Variable/Full[a]
Selection mechanism: means by which parents are chosen	Probabilistic, Ranked, Tournament[a], Elitist Tournament, Demetic
Population size (M): programs in the population	10–500 (300[a])
Generations (G): generations over which programs evolve	10–100 (30[a])
Secondary parameters	
Crossover probability (P_c): probability that a program will be the offspring of two other programs	0.9
Mutation probability (P_m): probability that a program mutates	0.001
Reproduction probability (P_r): probability that a program will be the offspring of one parent program	$1-(P_c+P_m)$
Internal crossover points (%): probability that program is crossed or mutated at a function instead of a terminal	0.7
Creation depth: maximum depth of programs when created	6
Crossover depth: maximum depth at which programs cross	17
Function set: functions that can act as tree nodes	$+ - \div \times$
Randomness: probability of choosing an ephemeral constant	0.5
Program length: maximum genetic program length	500
Tournament size: tournament is held between X individuals	5

[a]Default.

generation. Reproduction is useful for maintaining fit strategies across generations. *Mutation* is a special form of reproduction in which a parent creates an identical child but a gene is randomly mutated in the process. Mutation introduces changes in a program by chance and ensures that terminals and functions do not disappear from the population due to crowding by more successful terminals and functions in early generations.

2.3 *Evolutionary programming to represent bounded rationality*

The choice of evolutionary programming as a symbolic regression method stems from the broader challenge of establishing how agents may approximate $\hat{f}(x)$ in a manner that satisfies theoretical imperatives. One of the most common theories of decision-making in the social sciences, perfect rationality, is joined by a growing number of alternatives, of which bounded rationality is prominent. Every decision-making theory has specific corollaries that govern its attendant computational

implementation of symbolic regression. Perfect rationality is expressed in a variety of ways, particularly via statistical regression models, while bounded rationality is successfully implemented with evolutionary programming.

Perhaps the most commonly accepted model of decision-making in the social sciences—and, by extension, land-change models—is rational choice theory implemented with statistical regression. One variant of this theory, termed *perfect rationality*, is particularly widespread because it offers power and analytical tractability via assumptions of utility maximization, perfect computation, and complete information on alternatives (Myers and Papageorgiou 1991). Many methods for solving symbolic regression problems are in keeping with the corollaries of perfect rationality. Econometric research in particular adapts a variety of regression methods (e.g. least-squares, hazard, and logistic models) to the underlying principles of perfect rationality for land use (Bockstael 1996, Nelson and Geoghegan 2002) and rational choice in general (McFadden 2001). When applied to empirical data, these symbolic regression methods aim to solve in the aggregate the same multicriteria evaluation problem faced by agents individually.

The normative tradition of rational choice is increasingly joined by descriptive alternatives (Bell *et al.* 1988, van den Bergh *et al.* 2000). Key among these is *bounded rationality*, introduced by Simon (1997: 267) as 'rational choice that takes into account the cognitive limitations of the decision-maker—limitations of both knowledge and computational capacity'. 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. A key effect of these limits is that strategies have limited complexity; decision-makers use simple heuristics or 'rules of thumb' (Baumol and Quandt 1964, Slonim 1999). These key precepts of bounded rationality—limits to information and cognition— have been extended by positing that learning from experience is important to explaining satisficing, creation of heuristics, and limits to information. Actors improve strategies through a Darwinian process of learning-by-doing that balances path-dependent exploration of new strategies by extending current strategies versus simply exploiting existing strategies (Arthur 1993, Gigerenzer *et al.* 2001).

A key advantage claimed for agent-based modelling is the ability to represent key features of bounded rationality in agents, such as learning over time, bounded access to information, and limited computational resources (Epstein 1999). Agent-based models of land use implement bounded rationality (sometimes implicitly) through mechanisms such as limits to knowledge, as when agents use local information or have a limited search capacity, and limits to computation, particularly through symbolic regression approaches such as heuristics or classifier systems (Gimblett 2002, Janssen 2003, Parker *et al.* 2003).

Evolutionary programming is a particularly promising means of representing bounded rationality in agent-based models, although the modeller must choose between two ways in which to do so: functional and representational (Chattoe 1998). Evolutionary programming is functional when it acts in an instrumental manner to find optimal or near-optimal solutions to well-specified problems in high-dimensional domains characterized by noise and complexity (Kaboudan 2003). In land-change research, evolutionary programming is most often used in a functional manner for the optimal allocation of land because it can quickly navigate

complicated search spaces defined by millions of land parcels or raster grid cells (e.g. Balling *et al.* 1999, Matthews *et al.* 1999, Xiao *et al.* 2002).

In contrast, the representational use of evolutionary programming draws several parallels between human decision-making and the genetic programming system (Edmonds 2001, Chen 2003). Representational evolutionary programming grants each agent a population of strategies; in an agent-based model of land change like SYPRIA, each program is an alternative multicriteria evaluation strategy of land use. An agent uses the fittest strategy when it makes a decision at any given time, but it also possesses many other potential strategies to use in the face of changing circumstances (Dosi and Nelson 1994). Importantly, the fittest strategy for an agent is almost always satisficing instead of optimizing because the representational use of evolutionary programming imposes computational limits to cognition and information.

Representational evolutionary programming maps onto many corollaries of bounded rationality. First, limits to computational ability are imposed by reducing the number of trees available to an agent and the depth or length that they can achieve (Edmonds and Moss 2001). Second, limits to information are imposed by the manner in which offspring carry remnants of preceding generations. These portions of earlier programs are equivalent to memory, but this memory does not provide perfect information because it is distorted over time (Chen 2003). Third, evolutionary programming can also represent learning under bounded rationality as the above-noted balance between exploitation and exploration. Agents use crossover to blend successful strategies to create better strategies, in essence learning from experience (Dawid 1999), while agents use reproduction as the equivalent of exploiting existing well-performing strategies (Moss and Edmonds 1998, Beckenbach 1999). Fourth, mutation is akin to the errors, experimentation, and accidents that form the basis for innovation (Chen and Chie 2004). Fifth, crossover implements imperfect information and search costs when less fit programs engage in crossover, leading to suboptimal offspring (Brenner 1999). Finally, agents can engage in imitation, communication, and development of new strategies by borrowing strategies, although this interpretation requires us to adopt the computational conceit that function follows form (Pingle 1995, Dawid 1997).

Of course, the linkages between evolutionary programming and bounded rationality are relatively nascent. There is mounting evidence for the correspondence between evolutionary programming, models of bounded rationality, and bounded rationality in real decision-making (in addition to works noted above, for reviews see Chattoe 1998, Edmonds 1998, Chen *et al.* 2002, Chen and Wang 2004). Evolutionary programming also captures aspects of decision-making that perfect rationality and statistical models do not. Few researchers would claim, however, that either all bounds to rationality or the rationality (as such) of real actors strongly maps onto the artificial cognitive processes of agents equipped with genetic programming systems. Bounded rationality assumes that real decision-makers face limits to their cognitive processing but does not assume that they use genetic programs or that programs fully reconstruct actual strategies that map onto actual cognitive structures (leaving aside the philosophical question of where cognition resides). There is an ongoing need to explore computational analogues, in general, and evolutionary programming, in particular, to bounded rationality.

3. Methods

We face unanswered questions when linking the conceptual underpinnings of bounded rationality to the technical implementation of genetic programming. The modeller must modify the genetic programming system possessed by each agent in order to fulfil the functional task of solving the multicriteria evaluation problem while also satisfying the representational goal of capturing features of bounded rationality. I constructed an experimental frame to test five key genetic programming parameters—fitness function, creation mechanism, selection operator, population size, number of generations—and their effects on both the functional efficiency of programs and their representational fidelity.

3.1 *Experimental frame*

To better understand the role of the five system parameters in modelling bounded rationality with genetic programming, I tested variations of one of the five parameters of interest across 600 model runs in SYPRIA while holding the other parameters constant, for a total of 3000 runs. Table 1 details the range of these parameters and outlines secondary parameters common to genetic programming applications (Banzhaf *et al.* 1998). As noted above in section 2.1, much of the operation of SYPRIA is irrelevant to the experimental frame, which reports on the genetic programming systems of SYPRIA agents solving their symbolic regression problem.

3.1.1 Fitness function. The heart of genetic programming lies in breeding better strategies, which in turn requires the selection of fitter parents for reproduction, crossover, and mutation. Parents are selected according to their fitness f_k as given by minimizing error over observations $\mathbf{O}$ when solving for $\hat{f}(x)$. The four most common functions are summed errors, mean error, sum-of-squared errors, and sum-of-square-root errors (table 1).

3.1.2 Creation operator. There are three ways to modify genetic programs at their creation, and these can be used to create five kinds of population (figure 2). The first is to fill a population with *full* programs, each of which has branches that extend outward from the root to a uniform maximum depth. The second is to create a population of *variable* trees, the branches of which can vary in depth up to a maximum value. The third is to create a *ramped* population of either full (*ramped full*) or variable (*ramped variable*) trees by dividing it into subpopulations that vary in their maximum depth. A ramped full population of 10 programs can consist of five subpopulations, for example, in which the members of the first have a depth of two, the second a depth of four, and so on to the fifth group having a depth of 10. Finally, a population may also have both full and variable members to create a *ramped variable/full* population. The purpose of creating a ramped or variable population is to grant it greater diversity of program size and complexity, instead of forcing all programs to share a common structure at the outset.

3.1.3 Selection operator. The overall fitness of the genetic program population increases over time because successful parents are more likely to have offspring that replace poorly performing programs. Five functions are commonly used to select parents: probabilistic, ranked, tournament, elitist tournament, and demetic. The widely used *probabilistic* approach was developed by Holland (1975) for genetic algorithms, the precursor to genetic programming. The probability of a program
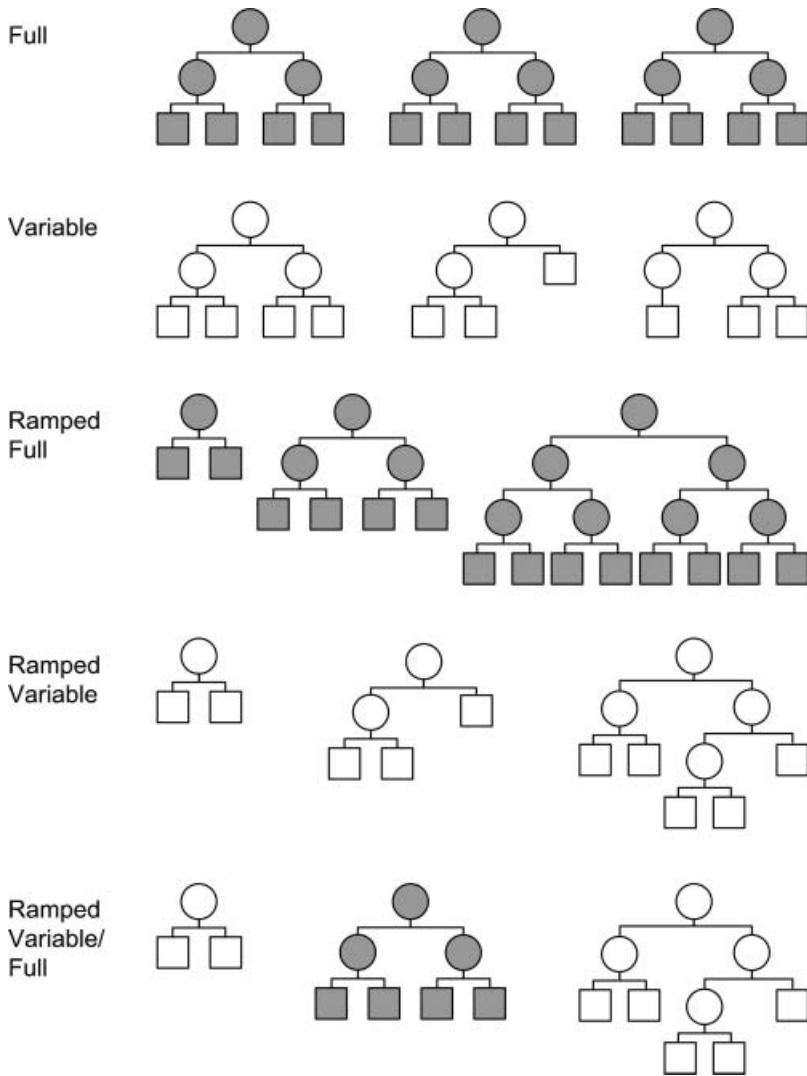
Figure 2. Creation operators.

being chosen is proportional to its individual fitness compared with the summed fitness of all other programs in the population. Parent programs are therefore selected according to their fitness as a proportion of the total fitness, while programs to be replaced in the steady-state population are selected with the inverse operation. Under *ranked* selection, parents are chosen according to their fitness rank in the population overall, and their offspring replace the worst-ranked individuals. The *tournament* method chooses a small group of programs at random from the entire population. Each member in this group is compared with the others, and the program with the best fitness is selected for crossover, reproduction, or mutation, and its child replaces that with the worst fitness. The *elitist tournament* method is identical to the tournament technique except that the fittest individual in the tournament automatically reproduces. Finally, the *demetic grouping* method divides the population into subpopulations, or demes. Within each deme, parents are

chosen at random, but migratory programs are 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 (Langdon 1998).

3.1.4 Population size. When each agent has a larger initial population, it has a greater diversity of candidate solutions and an increased likelihood of finding good solutions. Population size in applications across the social, natural, and information sciences range across several orders of magnitude, with an average size of 500 for smaller problems, 10 000 for difficult ones, and upwards of 1 000 000 for very large and near-intractable problems (Banzhaf *et al.* 1998, Koza 1999).

3.1.5 Generations. The number of generations works much in the same way as population size, in that a genetic programming system with a greater number of generations is more likely to produce better solutions (Kushchu 2002). The number of generations varies widely across applications, with 30–500 being typical (Banzhaf *et al.* 1998).

3.2 Evaluating outcomes

I used linear regression and ANOVA to examine changes in two measures— program complexity and fitness—to elucidate the effects of varying the five key parameters in agent genetic programming systems. Program complexity, expressed primarily as length, gives a simple measure of how many terminals and functions a tree possesses (Kaboudan 2003). In this application, genetic programs are limited to a length of 500 because we are generally interested in agents creating short programs that serve as rules of thumb with limited complexity. Program fitness measures how well programs solve the multicriteria evaluation problem for each agent, keeping in mind that a lower fitness score indicates greater program fitness because the score is an error measure. Fitness is expressed as either the average population fitness in a model run, $f_{\bar{p}}$, or the fitness, f_k, of the best program in a run.

Statistical analysis of program complexity and fitness establishes how well genetic programs fulfil their functional and representational roles. The *functional* efficiency of a genetic programming system is determined by its ability to create candidate solutions to the multicriteria evaluation problem faced by agents, each of which performs a symbolic regression to approximate $f(x)$ given observations **O**. The *representational* fidelity of a genetic programming system is determined by the extent to which programs, as measured by changes in their complexity and fitness, provide reasonable representations of bounded rationality.

Of the many potential aspects of such representations, four are of particular interest here. First, each agent must solve its multicriteria evaluation problem by maximizing the fitness of its strategies. These strategies in turn are limited to satisficing solutions because the genetic programming system imposes a number of constraints, and each agent has only a limited number of observations **O** with which to plumb its surroundings. Second, strategies should have limited complexity, as expressed by program length and depth, in keeping with the assumption of bounded rationality that decision-makers use simple heuristics. In this respect, shorter programs are preferred over longer programs, as program length and depth are proxies for agent memory and strategy complexity. Third, agents should possess limited computational resources with which to create, maintain, or expand strategies. These limits to resources are controlled directly by population size and

number of generations, for example, and indirectly by strategy complexity. Finally, the genetic programming system must balance exploitation of existing strategies and exploration of new ones.

In many respects, genetic programming parameters must balance the functional and representational roles of evolutionary programming. An agent could limit its genetic programming system to a few generations of programs, for example, but this configuration would not adequately represent either the learning of new strategies through crossover or exploitation of existing strategies through reproduction. Nor would the majority of resultant strategies solve the multicriteria evaluation problem because they would not have had the opportunity to evolve much beyond their initial random structures. Conversely, an agent could command an extraordinarily large population of programs or enough computing power to find a near-optimal solution to its multicriteria evaluation problem. This resource endowment is not in keeping with corollaries of bounded rationality, however, such as simple strategies or limits to computational resources. It is therefore important for the modeller to determine parameter settings that balance functional and representational needs.

4. Results and discussion

4.1 *Fitness function*

Switching among the fitness functions (summed errors, mean error, sum-of-squared errors, and sum-of-square-root errors) had little discernible effect, which is somewhat surprising given that sum-of-squared errors highlights large errors, while the sum-of-square-root errors dampens the effects of large deviations. An ANOVA of ln $(f_{\bar{p}})$ to test for differences in program fitness among fitness functions demonstrated that they were not statistically significant ($p=0.3374$, $F=1.13$, df$=596$).

This lack of significant differences is likely due to the programs evolving with the sole goal of approximating $\hat{f}(x)$. In contrast, a multiobjective problem such as maximizing fitness while minimizing program length would likely see differences in the effectiveness of varying fitness functions (Ekart and Nemeth 2005). Otherwise, sum-of-squared errors is useful because it is analogous to the error function used by other symbolic regression methods, such as ordinary least squares.

More broadly, there is no immediately compelling reason stemming from the theory of bounded rationality to suggest one form of error function over another, although risk perception research suggests that humans tend to overemphasize large errors (Tversky and Kahneman 1974). Despite the lack of a significant relationship in this application, linkages between fitness function and bounded rationality may prove a fertile topic for exploration given potential ties to research on risk perception.

4.2 *Creation operator*

Although an ANOVA of ln $(f_{\bar{p}})$ indicates that between-group variation in genetic program fitness for creation methods was significant overall ($p=0.0000$, $F=10.61$, df$=596$), only several pairings were significantly different among the operators full, variable, ramped variable, ramped full, and ramped variable/full (table 2). The largest differences occurred between full or variable populations when either one was ramped but only two combinations were uniformly significantly different, namely ramped variable versus either full or ramped full.

Table 2. Creation method between-class variance.

	[Difference ($\pm$) in ln ($f_{\bar{p}}$) and p-values (p)]							
	Variable		Full		Ramped full/ variable		Ramped variable	
Creation type	$\pm$	p	$\pm$	p	$\pm$	p	$\pm$	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

In terms of average population genetic program fitness, there was a small significant difference of means between the methods. Full populations (ln ($f_{\bar{p}}$)=13.19) outperformed variable (13.34), ramped variable (13.32), ramped full (13.20), and ramped variable/full (13.28) populations (again noting that *lower* scores indicate *better* programs). Figure 3 better illustrates the range of effects of creation operators on program complexity. Figure 3(*a*) indicates the distribution of standardized fitness for each creation method, highlighting that while full populations generally were more fit (having lower f_k scores), they also possessed a higher proportion of poorly performing programs (higher f_k scores). Figure 3(*b*) demonstrates that full populations possessed longer genetic programs, while variable populations possessed larger proportions of relatively short programs balanced against a number of larger programs (bearing in mind that programs have a maximum length of 500). The ramped variable/full approach attenuated the excesses of either the full or the variable methods by ensuring that about a third of the programs were short.

The ramped variable/full creation method appears to balance genetic program performance and length. While no one program length can be said to be realistic in terms of decision-making, shorter programs are better if they are meant to act as rules of thumb and if actors have limited computational capacity. In computational terms, the bimodality in lengths that results from the various creation types in figure 3(*b*) suggests that there is a natural break point in the distribution at programs with a length of roughly 50–75. The agent can choose among the roughly one-third of the individual programs that have reached this length and ignore the group of larger programs. In representational terms, short programs are useful because they are simple strategies. At the same time, these strategies survived over the full amount of time required to create programs; these programs are not short because they were plucked from the population early. They are both simple and competitive against longer programs; otherwise, they would be culled. In another sense, agents retain their capacity to experiment and evolve programs over the full operation of the genetic programming system while maintaining short but fit programs.

4.3 *Selection operator*

An ANOVA of ln ($f_{\bar{p}}$) across the five selection operators indicates a significant difference among them (p=0.0000, F=34.36, df=596, where the operators are probabilistic, ranked, tournament, elitist tournament, and demetic). Tournament methods, both regular (mean ln ($f_{\bar{p}}$)=13.39) and elite (13.41), outperformed
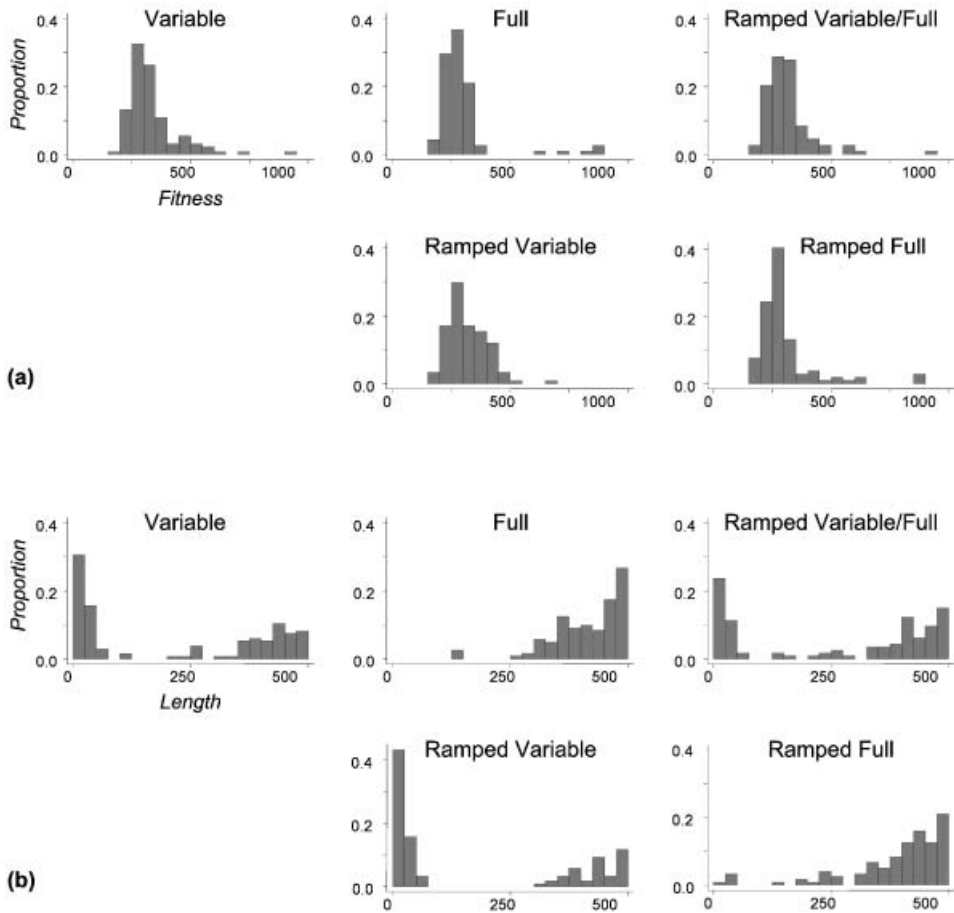
Figure 3. Creation operator effect on standardized fitness score (*a*) and length (*b*).

probabilistic (13.68), ranked (13.64), and demetic (13.55) methods by margins wider than those seen in tests of creation methods. Table 3 illustrates significant differences between three groups of selection operators: probability and ranked methods vs. tournament and elite tournament methods vs. demetic grouping.

Figure 4 further illustrates the differences between selection operators. Ranked and probability methods produced leptokurtic fitness curves because the earliest best-performing programs quickly dominated other individuals (figure 4(*a*)). In essence, these programs gained a large evolutionary advantage because their offspring were far more likely than those of other programs to propagate through the population. As a result, the average population fitness over repeated runs was poor because less evolutionary pressure was brought to bear on the population as a whole. In contrast, tournament and demetic methods showed more uniform distributions of programs that are generally low scoring and therefore fitter because a broader array of programs were allowed to develop and propagate. Similar evolutionary pressures led to differences in program length (figure 4(*b*)). The ranked and probability selection methods created a higher proportion of short genetic programs because smaller successful programs quickly dominated the population

Table 3. Selection operator between-class variance.

| | [Difference ($\pm$) in ln ($f_{\bar{p}}$) and p-values (p)] | | | | | | | |
| | Probabilistic | | Ranked | | Tournament | | Elitist tournament | |
Selection type	$\pm$	p	$\pm$	p	$\pm$	p	$\pm$	p
Ranked	−0.041	1.000						
Tournament	−0.295	0.000	−0.253	0.000				
Elitist 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

and stifled further evolution in the population as a whole. The tournament and demetic methods were less prone to creating such lopsided distributions.

The choice of selection operator in representing bounded rationality requires a greater trade-off than for the choice of creation methods. Ranked and probabilistic methods produce short programs, for example, which is useful for representing
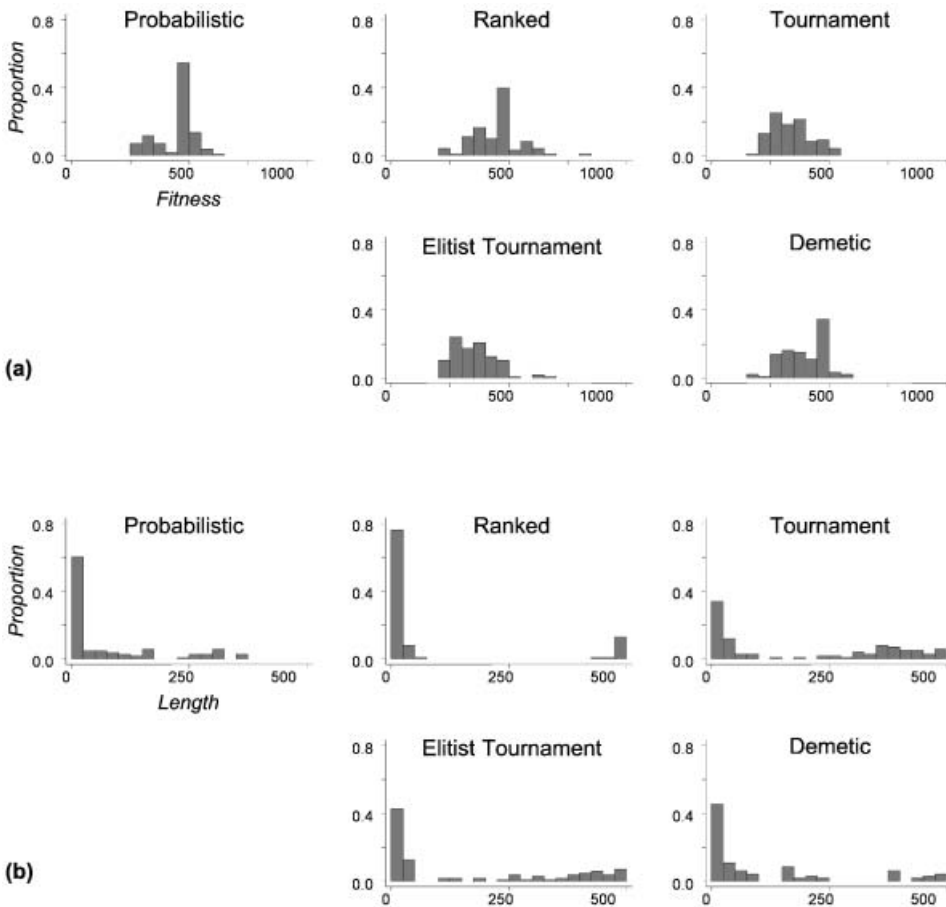

Figure 4. Selection type effect on standardized fitness score (*a*) and length (*b*).

simple strategies, but at the cost of generating poor length and fitness distributions. Perhaps more important is the manner in which the genetic programming system adopts early solutions more readily than with other selection methods. This early adoption precludes many potential strategies by limiting the amount of exploration in the long run in trade for overwhelming exploitation of early successful strategies. Ranked and probabilistic methods highlight the fundamental tension between the need to create short programs while allowing the population in aggregate (and thereby the agent) to explore other strategies.

Tournament and demetic grouping offer a functional advantage in that they produce programs of higher fitness. Demetic selection is intriguing but complicates the already tenuous association between actual decision-making and genetic programming. Does each agent have subpopulations of related strategies that occasionally spill over into one another? Barring a compelling answer, tournament selection is a compromise solution for agent decision-making. In addition to not suffering the functional and representational drawbacks of the other methods, each agent that uses tournament methods is in essence experimenting with a small cadre of competing strategies at any one time. This batchwise comparison accords well with findings that actual decision-makers can simultaneously consider and compare relatively few strategies (Chase *et al.* 1998)

4.4 *Population size*

Population size had an effect on both program fitness and length (figure 5). Not surprisingly, there was a positive relationship between population size and fitness (figure 5(a)). A linear regression for 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$=599$). Smaller populations yielded shorter programs because, as seen with probabilistic and ranking methods of selection, relatively fit programs that appeared early in small populations quickly dominated others because they faced less competition. These early programs tended to be shorter because they had less time to grow. In larger populations, there was less competitive pressure among programs in early generations, which gave the population greater latitude for growing larger programs that could develop over time. These larger programs in turn were more likely to have higher fitness than smaller programs.

Conceptually, just as there is no single program length that represents a human decision strategy, there is no obvious population size that best represents actual human cognition. Does the average person have one strategy, a hundred, or a
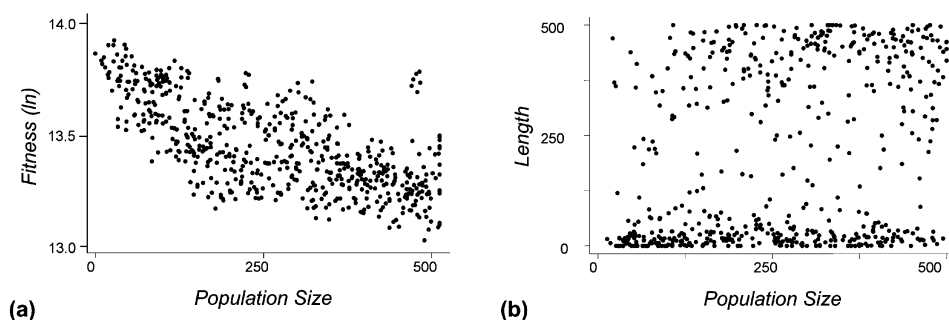

Figure 5. Population effect on standardized fitness score (a) and program length (b).

million? Does each genetic program represent part of a larger compound strategy or a strategy in and of itself? We must again balance two representational precepts: an agent should command limited computational resources, but it should also produce useful solutions to its multicriteria evaluation problem. In this case, a population in the range of 200–300 programs is the lowest realistic number, as smaller populations do not converge well on useful strategies. Conversely, populations larger than approximately 300 programs would move the agents away from representing boundedly rational actors and towards those with infinite computational capacity under perfect rationality.

4.5 *Generations*

There was a weak but significant relationship between genetic program generation size and fitness, where 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$=599$). Figure 6 illustrates the effect of generations on program fitness and length. Figure 6(*a*) indicates to a limited extent how the genetic programming system required 25–30 generations to settle into a weak relationship with fitness. Figure 6(*b*) shows how the number of programs with greater lengths generally increased with the number of generations before being overtaken by bimodality imparted by the ramped variable/full selection method.

The relationship between generations and program fitness and length becomes more apparent when examining the fitness and length of the fittest member of the population in every generation (figure 7). Given that the initial random programs were poor solutions, the fitness of the fittest member increases dramatically over the first eight to 10 generations before settling down to a slower rate of increase over the remaining generations. The length of the fittest member also demonstrated strong and steady growth to generation 20 before levelling off, which is due in part to genetic programs converging on successful solutions and, importantly, by the length of programs in later generations being limited to a maximum of 500.

In terms of representing bounded rationality, many of the same arguments for low population size hold true for a small number of generations, as do some of the conceptual conundrums. An agent must create a useful solution to the multicriteria evaluation problem with a minimum of computational resources. A range of 10–30 generations maintains both the functional and representational utility of evolutionary programming. Interestingly, this range is low relative to other genetic programming efforts (Aler *et al.* 2001), but higher values would make for more of an
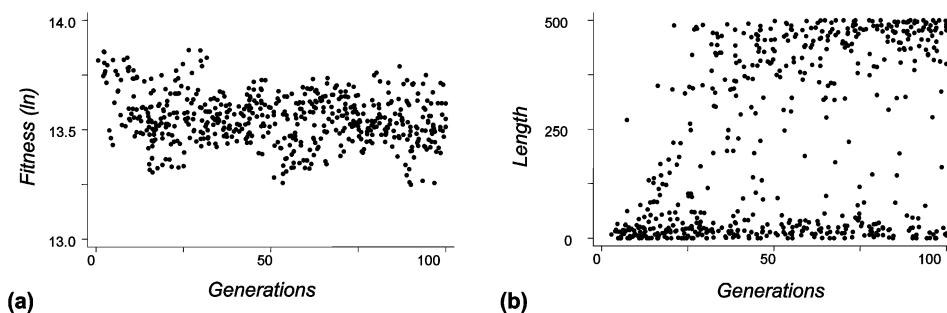
Figure 6. Generation size effect on standardized fitness score (*a*) and program length (*b*).
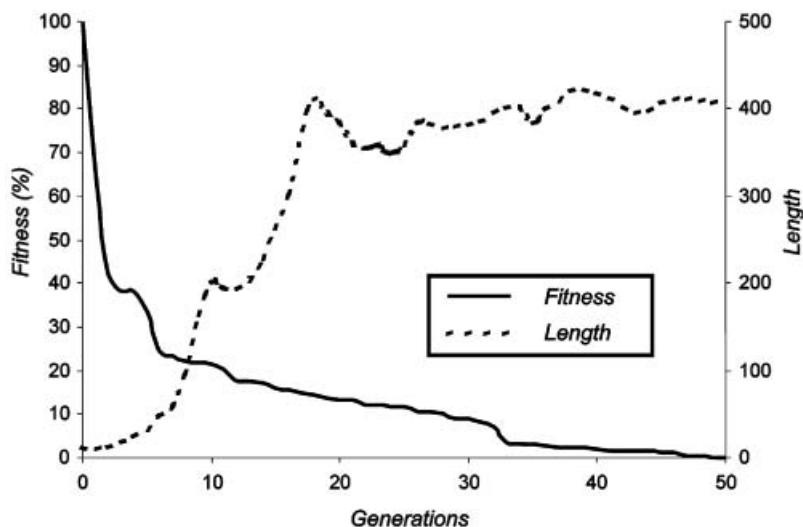
Figure 7. Generation size effect on standardized fitness score and program length for fittest programs.

instrumental or functional search for solutions by agents that is not in keeping with bounded rationality. Low generation values lead to shorter programs, which offsets the tendency of other preferred parameters (such as tournament selection or ramped variable/full creation) to create longer genetic programs. As with population size, limiting the number of generations limits both the length of programs and the computational resources granted to agents.

More broadly, humans typically have fewer than 10 or so opportunities to learn strategies in many decision-making situations (Arthur 1993), while genetic programming can require hundreds or thousands of generations to reach workable solutions for large and complex problems. We can address this issue in part by again acknowledging that genetic programs are merely proxies to actual strategies. We can also posit that human imagination allows the decision-maker to mentally assess the fitness of competing alternative strategies in addition to testing them outright.

5. Conclusion

Understanding human decision-making is a central challenge in the social and cognitive sciences in general and land-change research in particular. We increasingly rely on computational models that in turn require a combination of theory, method, and data on decision-making. While modellers will continue to employ perfect rationality and its associated methods, it is necessary to explore alternatives such as bounded rationality. Despite persistent interest from the cognitive and social sciences, however, research on bounded rationality is less theoretically developed and methodologically integrated than that on perfect rationality.

Land change provides an excellent domain for examining decision-making because this change is caused by individual actors guided by, and contributing to, social and environmental drivers of change. Moreover, land-change research offers an interdisciplinary blend of data, method, and theory with which to test the computational representation of key features of cognition. By using agent-based modelling and genetic programming to represent agriculturalist households and

their decision-making, SYPRIA contributes to a broader movement towards use of evolutionary programming to represent decision-making.

Genetic programming offers a way to model the cognition of individual actors in a manner that addresses the need for agents to solve their multicriteria evaluation problems; limits to strategy complexity; limits to the computational resources with which to create, maintain, or expand strategies; and the balance between exploration of new strategies and exploitation of existing ones. In a broader sense, this work is part of a larger movement that draws ever firmer ties between evolutionary programming and bounded rationality in particular and between computational modelling and decision-making in general.

Further research is required to better understand the linkages between evolutionary programming and bounded rationality in agent-based models. Agent-based modelling is a relatively young endeavour and faces a number of challenges, perhaps the most important of which is moving from agents representing entities with little or cognition (cells, molecules) to agents that have many forms of rationality (humans, institutions). Agent-based models can recreate real-world patterns of land change, but they must arguably draw on realistic generating processes in order to explain this change, while at the same time not becoming so complex so as to lose their ability for generalization. In this respect, the elegance and universality of perfect rationality must be balanced against the need for more realistic decision-making models such as bounded rationality.

At the same time, alternatives to perfect rationality require a good deal more research before they will be fully accepted into the mainstream. As argued throughout this paper, there is a growing body of research centred on empirical investigations designed to advance our theoretical understanding of decision-making. This research highlights the need for closer integration between qualitative field research and quantitative modelling. Agent-based modelling is proving a useful vehicle for combining geospatial technologies such as remotely sensed imagery, global positioning system data, and GIS with qualitative in-depth interviews or field surveys necessary for deeper insight into human cognition (e.g. D'Aquino *et al.* 2003, Brown *et al.* 2005, Evans and Kelley 2004).

References

AGARWAL, C., GREEN, G.L., GROVE, J.M., EVANS, T. and SCHWEIK, C., 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, VT).

ALER, R., BORRAJO, D. and ISASI, P., 2001, Learning to solve planning problems efficiently by means of genetic programming. *Evolutionary computation*, **9**, pp. 387–420.

ARTHUR, W.B., 1993, On designing economic agents that behave like human agents. *Journal of Evolutionary Economics*, **3**, pp. 1–22.

BALLING, R.J., TABER, J.T., BROWN, M. and DAY, K., 1999, Multiobjective urban planning using a genetic algorithm. *ASCE Journal of Urban Planning and Development*, **125**, pp. 86–99.

BANZHAF, W., NORDIN, P., KELLER, R.E. and FRANCONE, F.D., 1998, *Genetic Programming: An Introduction* (San Francisco, CA: Morgan Kauffman).

BAUMOL, W.J. and QUANDT, R.E., 1964, Rules of thumb and optimally imperfect decisions. *American Economic Review*, **54**, pp. 23–46.

BECKENBACH, F., 1999, Learning by genetic algorithms in economics. In *Computational Techniques for Modelling Learning in Economics*, T. Brenner (Ed.), pp. 73–100 (Boston, MA: Kluwer Academic).

BELL, D.E., RAIFFA, H. and TVERSKY, A., 1988, *Decision Making: Descriptive, Normative, and Prescriptive Interactions* (Cambridge: Cambridge University Press).

BOCKSTAEL, N.E., 1996, Modeling economics and space: the importance of a spatial perspective. *American Journal of Agricultural Economics*, **78**, pp. 1168–1180.

BRENNER, T., 1999, *Computational Techniques for Modelling Learning in Economics* (Boston, MA: Kluwer Academic).

BROWN, D.G., RIOLO, R., ROBINSON, D.T., NORTH, M. and RAND, W., 2005, Spatial process and data models: toward integration of agent-based models and GIS. *Journal of Geographical Systems*, **7**, pp. 1–23.

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

CHASE, V.M., HERTWIG, R. and GIGERENZER, G., 1998, Visions of rationality. *Trends in Cognitive Sciences*, **2**, pp. 206–214.

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

CHEN, S.-H., 2003, Fundamental issues in the use of genetic programming in agent-based computational economics. In *Agent-Based Approaches in Economic and Social Complex Systems*, A. Namatame, T. Terano and K. Kurumatani (Eds), pp. 208–221 (Amsterdam, IOS Press).

CHEN, S.H. and CHIE, B.T., 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**, pp. 2376–2386.

CHEN, S.-H., DUFFY, J. and YEH, C.-H., 2002, Equilibrium selection via adaptation: using genetic programming to model learning in a coordination game. *The Electronic Journal of Evolutionary Modeling and Economic Dynamics*, **10**, pp. 1002.

CHEN, S.-H. and WANG, P.P., 2004, *Computational Intelligence in Economics and Finance* (Berlin: Springer).

COLLINS, M.G., STEINER, F.R. and RUSHMAN, M.J., 2001, Land-use suitability analysis in the United States: historical development and promising technological achievements. *Environmental Management*, **28**, pp. 611–621.

D'AQUINO, P., LE PAGE, C., BOUSQUET, F. and BAH, A., 2003, Using self-designed role-playing games and a multi-agent system to empower a local decision-making process for land use management: the SelfCormas experiment in Senegal. *Journal of Artificial Societies and Social Simulation*, **6**, Article 5.

DAWID, H., 1997, Learning of equilibria by a population with minimal information. *Journal of Economic Behavior and Organization*, **6**, pp. 361–373.

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

DOSI, G. and NELSON, R.R., 1994, An introduction to evolutionary theories in economics. *Journal of Evolutionary Economics*, **4**, pp. 153–173.

EDMONDS, B., 1998, Modelling socially intelligent agents. *Applied Artificial Intelligence*, **12**, pp. 677–699.

EDMONDS, B., 2001, Towards a descriptive model of agent strategy search. *Computational Economics*, **18**, pp. 111–133.

EDMONDS, B. and MOSS, S., 2001, The importance of representing cognitive processes in multi-agent models. *Lecture Notes in Computer Science*, **2130**, pp. 759–766.

EIBEN, A.E. and SCHOENAUER, M., 2002, Evolutionary computing. *Information Processing Letters*, **82**, pp. 1–6.

EKART, A. and NEMETH, S.Z., 2005, Stability analysis of tree structured decision functions. *European Journal of Operational Research*, **160**, pp. 676–695.

EPSTEIN, J.M., 1999, Agent-based computational models and generative social science. *Complexity*, **4**, pp. 41–60.

EVANS, T.P. and KELLEY, H., 2004, Multi-scale analysis of a household level agent-based model of landcover change. *Journal of Environmental Management*, **72**, pp. 57–72.

GIGERENZER, G., SELTON, R. and SELTON, R., 2001, *Bounded Rationality: The Adaptive Toolbox* (Cambridge, MA: MIT Press).

GIMBLETT, H.R., 2002, *Integrating Geographic Information Systems and Agent-Based Modeling Techniques for Simulation of Social and Ecological Processes* (New York: Oxford University Press).

GUTMAN, G., JANETOS, A., JUSTICE, C., MORAN, E., MUSTARD, J., RINDFUSS, R., SKOLE, D. and TURNER, B.L. II., 2004, *Land Change Science: Observing, Monitoring, and Understanding Trajectories of Change on the Earth's Surface* (Dordrecht, The Netherlands: Kluwer Academic).

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

IRWIN, E.G. and GEOGHEGAN, J., 2001, Theory, data, methods: developing spatially explicit economic models of land use change. *Agriculture, Ecosystems, and Environment*, **85**, pp. 7–23.

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

KABOUDAN, M.A., 2003, Forecasting with computer-evolved model specifications: a genetic programming application. *Computers and Operations Research*, **30**, pp. 1661–1681.

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

KOZA, J., 1999, *Genetic Programming III: Darwinian Invention and Problem Solving* (San Francisco, CA: Morgan Kauffman).

KUSHCHU, I., 2002, Genetic programming and evolutionary generalization. *IEEE Transactions on Evolutionary Computation*, **6**, pp. 431–442.

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

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

MANSON, S.M., 2004, The SYPR integrative assessment model: complexity in development. In *Integrated Land-Change Science and Tropical Deforestation in the Southern Yucatan: Final Frontiers*, B.L. Turner, II, D. Foster and J. Geoghegan (Eds), pp. 271–291 (Oxford: Clarendon Press).

MANSON, S.M., 2005, Agent-based modeling and genetic programming for modeling land change in the Southern Yucatan Peninsular Region of Mexico. *Agriculture, Ecosystems and Environment*, **111**, pp. 47–62.

MANSON, S.M., 2006, Land use in the Southern Yucatan Peninsular Region of Mexico: scenarios of population and institutional change. *Computers, Environment, and Urban Systems*, **30**, pp. 230–253.

MATTHEWS, K.B., SIBBALD, A.R. and CRAW, S., 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**, pp. 9–26.

MCFADDEN, D., 2001, Economic choices. *American Economic Review*, **91**, pp. 351–378.

MOSS, S. and EDMONDS, B., 1998, Modelling economic learning as modelling. *Systems and Cybernetics*, **29**, pp. 5–37.

MYERS, G.M. and PAPAGEORGIOU, Y.Y., 1991, Homo economicus in perspective. *The Canadian Geographer*, **35**, pp. 380–399.

NELSON, G.C. and GEOGHEGAN, J., 2002, Deforestation and land use change: sparse data environments. *Agricultural Economics*, **27**, pp. 201–216.

NRC 2001, *Grand Challenges in Environmental Sciences* (Washington, DC: National Academy Press).

PARKER, D.C., MANSON, S.M., JANSSEN, M., HOFFMANN, M.J. and DEADMAN, P.J., 2003, Multi-agent systems for the simulation of land use and land cover change: a review. *Annals of the Association of American Geographers*, **93**, pp. 316–340.

PARKER, D.C. and MERETSKY, V., 2004, Measuring pattern outcomes in an agent-based model of edge-effect externalities using spatial metrics. *Agriculture, Ecosystems and Environment*, **101**, pp. 233–250.

PINGLE, M., 1995, Imitation versus rationality: an experimental perspective in decision making. *Journal of Socio-economics*, **24**, pp. 281–315.

POLHILL, J.G., GOTTS, N.M. and LAW, A.N.R., 2001, Imitative versus nonimitative strategies in a land-use simulation. *Cybernetics and Systems*, **32**, pp. 285–307.

RALSTON, A. and RABINOWITZ, P., 2001, *A First Course in Numerical Analysis* (New York: Dover Publications).

RUSSELL, S. and NORVIG, P., 1995, *Artificial Intelligence: A Modern Approach* (Upper Saddle River, NJ: Prentice-Hall).

SIMON, H.A., 1997, Behavioral economics and bounded rationality. In *Models of Bounded Rationality*, H.A. Simon (Ed.), pp. 267–298 (Cambridge, MA: MIT Press).

SLONIM, R.L., 1999, Learning rules of thumb or learning more rational rules. *Journal of Economic Behavior and Organization*, **38**, pp. 217–236.

TESFATSION, L., 2001, Introduction to the special issue on agent-based computational economics. *Journal of Economic Dynamics and Control*, **25**, pp. 281–293.

TVERSKY, A. and KAHNEMAN, D., 1974, Judgment under uncertainty: heuristics and biases. *Science*, **185**, pp. 1124–1131.

VAN DEN BERGH, J.C.J.M., FERRER-I-CARBONELL, A. and MUNDA, G., 2000, Alternative models of individual behaviour and implications for environmental policy. *Ecological Economics*, **32**, pp. 43–61.

WALKER, R., 2004, Theorizing land-cover and land-use change: the case of tropical deforestation. *International Regional Science Review*, **27**, pp. 247–270.

XIAO, N.C., BENNETT, D.A. and ARMSTRONG, M.P., 2002, Using evolutionary algorithms to generate alternatives for multiobjective site-search problems. *Environment and Planning A*, **34**, pp. 639–656.