1-1-2002

Genetic algorithm-based simulation of electric power markets

David Samuel Doty
Iowa State University

Genetic algorithm-based simulation of electric power markets

by

David Samuel Doty

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Iowa State University

Ames, Iowa

2002

Graduate College
Iowa State University

This is to certify that the master's thesis of

David Samuel Doty

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The electric power industry has attracted much attention in the past decade following the movement toward deregulation. This movement has the potential to lead to greater profits for electricity producers and consumers. It requires a shift in thinking, however, as companies used to a regulated industry learn to treat electricity as an economic commodity.

Economic markets are a complex area of study. Due to incomplete information and occasional irrationality on the part of market participants, they have the potential to careen wildly away form theoretical predictions. Electric markets in particular, having been regulated for so long, have had a bumpy re-entry into the atmosphere of de-regulated capitalism.

For all entities vested in the electric power industry, with this re-entry comes the need to protect themselves from risk as well as new opportunities for profit. This thesis explores the use of genetic algorithms to learn profit-maximizing strategies in a variety of simulated electric markets.

CHAPTER 1. OVERVIEW

1.1 Problem Statement

Electric power deregulation has been one of the major economic shifts to occur in recent years. This shift has focused mainly on replacing the regulated electric power industry with a competitive environment. This is based on the assumption that competition between profit-seeking companies will lead to lower prices for consumers and a more efficient economy in general. The Energy Policy Act of 1992 spells out the federal rules regarding this transition.

Companies which in the past had a vertical monopoly now must deal with the risks of losing customers to competitors, lower demand than they were previously guaranteed, and the requirement of granting other companies access to one's own power lines. Of course, this transition has been less than smooth in some areas, such as California, in which the industry was partially deregulated and ended up costing companies much more than the regulated environment did.

Part of the transition from regulation to competition involves setting up systems to treat electricity as a commodity. This includes determination of distribution of electricity from producers to consumers and determination of price paid for the electricity. Currently these factors are handled through the use of regional energy exchanges. In these exchanges, auctions are held in order to determine distribution and price. The auctions apply to multiple timescales: a monthly auction may determine distribution for each day in the month. Then a day-ahead daily auction may be held in order to finalize distribution for the next next day. Hourly auction would then be held on that day.

Such exchanges behave similarly to exchanges that trade other commodities such as wheat or gold. However, the physical constraints placed on power plants by the laws of physics give rise to unique considerations. For example, minimum and maximum quantities of electricity are available from each power-producing unit. Therefore, one must not be too quick to consider electricity as if it were like any other commodity. Production, transportation, and consumption of electricity all occur in non-trivially different ways than other commodities.

1.2 Scope of This Work

This works concerns itself primarily with the development of electric power market bidding strategies through the use of genetic algorithms. The bidding strategies were represented by two different modifications of a classical data processing structure known as a finite state automata. The genetic algorithms used were varied as well, some incorporating a semi-fixed fitness function, and others using a co-evolutionary (population-specific) fitness function.

Both types of fitness functions were based on maximizing profit in a competitive bidding situation. The only thing that changed was the choice of bidders against which to play. The auctions that were held to determine profits and possibly losses were played in an iterative fashion. In other words, the same bidder would play the same opponents multiple times, in order to allow the bidding agents to learn their opponents' behavior and adjust their own behavior accordingly in order to maximize profit.

Both of the representations were tested on each variation of experimental parameters. In addition, in the co-evolutionary setup, a third, very simple, representation was also run in for the sake of a comparison baseline for the other two representations, in order to show how their behavior compared with a "standard" solution.

There are many possible auction scenarios imaginable that could be used to determine distribution of electricity. This work uses a single-sided auction, but the work could easily be modified in order to accommodate a double-sided auction or more complex auctions.

1.3 Organization of This Work

Chapter 2 presents a review of the existing research literature relevant to this work. It concerns itself with literature primarily discussing two areas: bidding in electric power markets and genetic algorithms as applied to economic games. Included in the discussion of electric power markets are the areas of market design, market simulation, optimization, genetic algorithms, and reinforcement learning. This discussion of genetic algorithm literature reviews three economic games: the Prisoner's Dilemma, Divide the Dollar, and the Public Investment Game.

Chapter 3 discusses the theory behind the economic markets and genetic algorithms. It gives justifications for the choices of market parameters used, such as price and cost determination. It also gives justification for the choices of parameters used in the genetic algorithms, such as choices of representations used. Finally, it explains the parameters that were varied in order to create the different experiments producing the data in chapter 4.

Chapter 4 gives the experiments that were run and the results of each experiment. It also interprets briefly the results of each set of experiments.

Chapter 5 gives conclusions based on the results in chapter 4. It also discusses future research that could be done in this area.

CHAPTER 2. REVIEW OF LITERATURE

This chapter reviews work in the field of economic auctions, particularly as applied to electric power markets. It also presents research in the field of genetic algorithms applied to economic topics such as auctions and other economic games.

Much of the auction research varies in the specific auction design employed. This is due to the disparity in auction rules used in different regions throughout the world, as well as different simplifying assumptions made on the part of the researchers.

2.1 Electric Power Market Bidding

2.1.1 Market Design

Contreras, et al., in (Contreras01), compare different implementations of electric market designs. Specifically, they explain the differences between single-round and multi-round auctions, in terms of maximization of social welfare, computational cost, resultant market prices, etc. They reach the conclusion that iterative bidding (multi-round auctions) are not advisable for use in day-ahead markets.

Sheblé, in (Sheblé96), proposes details for the rules governing deregulated electric markets. He defines and lays out rules for the interaction of the electric commodity market and its various derivatives markets, such as the futures market, options market, and swap market. He also proposes breaking the trading into periods (hours, weeks, months), and bidding on production of electricity during these periods. Allocation already determined for a larger period (e.g. monthly) would constrain the possible allocation for smaller periods (e.g. weekly).

2.1.2 Market Simulation

Otero-Novas, et al., in (Otero-Novas00), discuss the simulation of a wholesale electricity market, called COMSEE, based on Wilson's rules proposed for the power exchange in the Californian market (Wilson97). These rules are:

1. The price cannot be increased.
2. The price can be decreased only if the new price is less than the clearing price in the previous iteration by at least a specified price decrement (e.g. \$1.00 or \$0.10/MWh). This new price is said to "improve" the previous price.
3. The price cannot improve any previous clearing price not improved at the first opportunity.

They use these rules to simulate a perfectly competitive market, by considering each generator to be independent entities seeking to maximize personal profit. They also simulate an oligopolistic market in which each firm coordinates the bids of its own units to maximize total profit.

2.1.3 Optimization

Weber and Overbye, in (Weber99), modelled the problem of bidding in an electric power market as a two-level optimization problem. The two levels of optimization consist of the problem of determining an optimal bid (first level) under the constraint that the price and dispatch quantity are determined by an optimal power flow (OPF) optimization (second level). This thesis assumes that the OPF problem is solved outside of the market simulation and embeds the solutions to the OPF into the generators' cost curves.

Song, et al., in (Song99), optimize bidding strategies by use of a Markov decision process. The Markov decision process used is similar to the finite state machine representation in use in this thesis, but with transitions between states determined stochastically instead of deterministically.

2.1.4 Genetic Algorithms

Richter and Sheblé, in (Richter98), used a genetic algorithm to develop bidding strategies for an electric power double auction. This used a representation based on evolving bid multipliers, or a number multiplied by the producer's marginal cost (or the buyer's marginal utility) to get the bid to make.

Richter, et al., in (Richter99), used genetic algorithms to evolve bidding strategies in a double auction for electric power. They conducted two experiments. One used genetic programming, or the evolution of parse trees, to evolve the bidding strategies. The other approach was to use GP-Automata to represent the bidding strategies.

2.1.5 Reinforcement Learning

Wu, et al., in (Wu02), introduced a machine learning algorithm to learn to bid in electric power markets. It is general enough to apply to any market, but they tested it specifically on a single-sided auction with discriminatory pricing.

Petrov and Sheblé, in (Petrov01), used the generic Roth-Erev learning algorithm ((RothErev95) and (ErevRoth95)) to learn effective bidding strategies for a double auction for electric power. They pointed out that the algorithm in its original form was unsuited to learning when profits were at or near 0. They modified the algorithm to account for this deficiency and thereby allowed it to learn much more quickly and efficiently.

2.2 Evolving Strategies to Play Simple Economic Games

Much of economic game theory is concerned with finding *Nash Equilibria* (Nash50). A Nash Equilibrium is a set of strategies, one strategy played by each player, with the property that no player can increase its payoff by unilaterally changing its strategy. Given certain assumptions, then all players should converge to that equilibrium. Some of these assumptions, however, are not met in practice: the game must have a unique Nash Equilibrium, all players must have perfect information and behave rationally (i.e. choose the best strategy once they know what it is).

2.2.1 Prisoner's Dilemma

The standard *Prisoner's Dilemma* game is a two-player non-zero-sum game. Each player makes a choice to "cooperate" or "defect". If both cooperate, both receive a payoff of 3. If both defect, both receive a payoff of 1. If one cooperates and the other defects, the cooperator receives a payoff of 0, and the defector receives a payoff of 5.

Standard economic theory shows that the standard prisoner's dilemma game has a Nash equilibrium in which both players defect. In other words, if both players have chosen to defect, neither can increase its payoff by changing its strategy. However, in any other state, in which one or both of the players cooperates, a higher payoff can be achieved by changing one's strategy to defect instead. In this respect, the choice "defect" is said to *dominate* choice "cooperate". Therefore, one would expect any agent learning to play the prisoner's dilemma to learn quickly that defecting is the optimal action, despite the higher payoff possible if both cooperate.

A variant of the prisoner's dilemma is called the iterated prisoner's dilemma. In this game, two players play multiple rounds of the prisoner's dilemma against each other. Axelrod evolved finite state machines to play the iterated prisoner's dilemma (Axelrod87). He found that when given the opportunity to play multiple rounds, agents will often learn to cooperate with one another in order to get the higher payoff that results when both players cooperate. However, they require a more complex strategy than simply "always cooperate," lest they be taken advantage of by more malicious players. Hence the choice to encode strategies as finite state automata rather than simply as a single action to take.

This experiment has relevance to the current work in two respects. First, the market being simulated is a complex, multi-round economic game. Second, finite state automata are being used to encode the strategies for playing this game. It is important that subsequent rounds of play be against the same opponents in order for the strategies to have any meaning. Otherwise, there is no such concept as reaction to an opponent's actions, because the opponent will not be the same the next time one meets him.

It is of interest to determine which Prisoner's Dilemma strategies are *evolutionarily stable.*

An evolutionarily stable strategy is one that, given that it is in use across the entire population, cannot be "invaded" by another strategy. To be invaded is to be beaten consistently so as to be driven out of the population by virtue having a lower fitness than the invading strategy. It has been shown that no pure (deterministic) strategy is completely evolutionarily stable (Boyd87). However, many strategies, such as Tit-For-Tat (TFT), are stable against almost all invading strategies, and the strategies that successfully invade TFT are themselves very unstable.

The iterated prisoner's dilemma has also been studied by Wagner, et al., in (Wagner00). They analyzed the effect of strategy representation on the evolution of cooperation in playing iterated prisoner's dilemma, investigating finite state machines, plain logical formulas, logical formulas with a time delay operation, If-Skip-Action(ISAc) lists, Markov chains, and neural networks.

Mayfield and Ashlock, in (Mayfield98), discovered a non-trivial effect of evolution. FSM's were evolved for 1000 generations, more than long enough for fitness to converge, and the population was saved. The FSM's were then evolved for an additional 9000 generations, with no apparent change in fitness. However, when the FSM's from generation 10000 were played against the FSM's from generation 1000, the former achieved much higher fitness than the latter. This effect, in which a population evolves for a very long time with no apparent change in fitness, but is able to build up general skills in achieving high fitness, has casually been called the *Mayfield Effect.*

2.2.2 Divide the Dollar

Ashlock, in (Ashlock97), evolved GP-Automata to play a simple game known as *Divide the Dollar.* In this game, two players bid a monetary amount. If the sum is less than or equal to a dollar, they each receive their bid amount, otherwise they each receive zero. This is an interesting game to study theoretically because it has a continuum of Nash Equilibria corresponding to real number solutions to the equation $a + b = 1$, where a is the first player's bid and b is the second player's bid. This means there are an infinite number of Nash equilibria, and so predicting how the game will end up being played is difficult theoretically. It turns out that GP-Automata ended up converging to a population in which every player bid just under \$0.50.

A study similar to (Wagner00) was conducted by Leahy and Ashlock (Leahy00). In this work they investigated the effects of representation choice on evolving agents to play the Divide the Dollar game. They used artificial neural nets, lookup tables, real valued mathematical formulae, integer valued formulae modulo 101, and GP-Automata to represent strategies.

2.2.3 Public Investment Game

Ashlock, in (Ashlock01), studied a simple evolutionary algorithm that played a population against itself in the *Public Investment Game.* In this game, multiple bidders submit a sealed bid between \$0 and \$100. The sum total that they invested is then doubled and distributed evenly among the bidders. This game attempts to model publicly funded utilities, such as roads requiring maintenance, in which everyone gets nothing if no one contributes, but the total effect on the group is negligible if only a single bidder lowers its contribution. This game's Nash equilibrium consists of every player making a bid of 0, and that is exactly the result that Ashlock observed after a few generations of evolution. He was able to achieve higher payoffs by introducing "laws" (minimum required investment levels) and "fines" (penalties subtracted from payoffs) for evading the laws.

CHAPTER 3. METHODS AND PROCEDURES

This chapter describes the setup of the experiments performed and gives background information necessary to understand the nature of these experiments.

3.1 Introduction

Background information is given here on both electric power markets and genetic algorithms.

3.1.1 Electric Power Markets

Electric power prices in the marketplace are determined for the most part by single-sided auctions. Generation companies (GENCOs) produce power and consumers buy power for consumption. Typically, energy service companies (ESCOs) consolidate demand from a group of consumers, so that GENCOs do not deal with individual electricity consumers. Transmission companies (TRANSCOs) own the power lines and are paid to transport electricity from one physical location to another. Ancillary services are provided by ancillary companies (ANCILCOs). An Energy Mercantile Association (EMA) serves as a buffer between producers and consumers, or a market maker. Figure 3.1 shows the interconnection between these entities. (Kumar96) Note that the value chain is different from the physical chain. The physical chain refers the the actual transportation of power from the producer to the final consumer. The value chain refers to the flow of money that is paid for this power, which involves extra entities that do not themselves necessarily consume, transport, or produce power.

The EMA holds a single-sided auction on either side. It will estimate demand from consumers and hold an auction taking bids from generators to sell their electricity. It will then

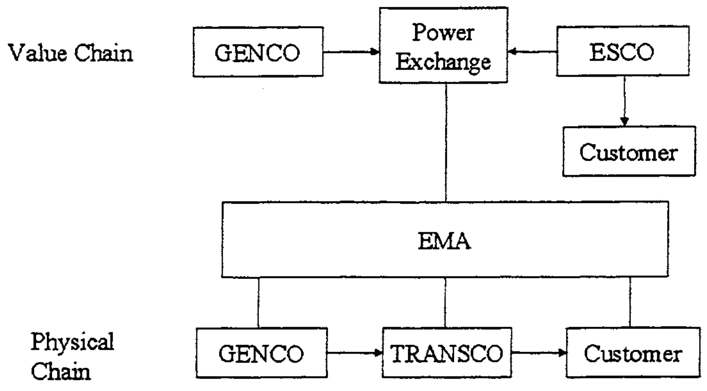

Figure 3.1 Electricity Marketplace

take the electricity it bought and turn around and hold an auction taking bids from consumers, who buy the electricity. This paper models the former auction, in which sellers, assumed to consist of competing GENCOs, make bids to sell power and must then produce the amount they agreed to sell. The simulations could just as easily be applied to a buyer's auction.

The purpose of an auction is to expose information about buyers' and sellers' respective willingness to pay or sell. Commodities like electricity often have no explicit fixed worth; their worth is a function of the current market conditions, and an auction attempts to find this worth (McAfee87). Essentially, an auction allows discovery of the *equilibrium price*, defined as the intersection of the demand and supply curves of the buyers and sellers, respectively. In this case the demand curve will be a straight line, as the EMA will estimate demand as a single fixed quantity.

According to economic theory, in the long run all profits should go to zero as sellers underbid each other in the marketplace. This prediction is disrupted, however, by any number of factors that appear in a real marketplace: cost curves that do not pass through the origin, capacity limits on generation quantity due to physical limitations of generators, and irrational behavior on the part of market participants. The most disruptive factor that leads to violation of theoretical predictions is information uncertainty on the part of market participants.

3.1.2 Genetic Algorithms

A genetic algorithm is a type of evolutionary algorithm. An evolutionary algorithm is an algorithm that uses the biological paradigm of evolution to solve mathematical problems. Many researchers in the field will disagree on definitions and terms. Part of this is due to confusion in the use of borrowed biological terms whose original definition is somewhat obscured by their use in evolutionary algorithm descriptions. What follows is as close to a common description of the field as possible.

3.1.2.1 Evolutionary and Genetic Algorithms

An evolutionary algorithm is any algorithm that follows the loop given in figure 3.2.

```
1. Create an initial population of potential solutions
2. Evaluate the fitness of the population
3. Repeat
4.          Select pairs from the population to be parents, with a fitness bias
5.          Copy the parents to make children
6.          Perform crossover on the children (optional)
7.          Mutate the resulting children
8.          Place the new structures in the population
9.          Evaluate the fitness of the new structures
10. Until Done
```

Figure 3.2 Basic Evolutionary Algorithm Loop

Each iteration of this loop is known as a generation. A *solution* (also called a *creature* or *agent*) in Figure 3.2 refers to some reasonable ("reasonable" being defined by the problem at hand) encoding of potential solutions to a problem. A subset of evolutionary algorithms, known as genetic algorithms, is the most prevalent and is the type used in the simulations described in this paper. These algorithms always execute line 6 of the basic evolutionary algorithm loop, which performs *crossover*. Crossover is the process of exchanging subsets of representations between two solutions. This mimics the process of sexual reproduction, as two parent solutions are copied and their children crossed over. Often genetic algorithms are defined to operate on fixed data structures. This is primarily to distinguish them from another type of independently developed evolutionary algorithm known as genetic programming (Koza92).

Genetic programming uses the loop shown in figure 3.2 to evolve parse trees (a variable-sized representation) to solve problems. However, it is possible to evolve data structures which are neither parse trees nor have a fixed size. Such an algorithm will also be termed a genetic algorithm in this paper.

In addition to crossover, the solution space is explored through the use of *mutation*, the process of randomly perturbing solutions. This mimics the biological process of random genetic mutation in creatures of a living population, which occasionally give the creature an advantage over others.

A *fitness function*, which numerically evaluates the optimality of a solution, must also be defined. Whatever scheme is used to select the parents, it always biases selection toward those creatures with higher fitness. It may also bias replacement of creatures toward those that are less fit. This mimics the biological process of natural selection, in which more fit creatures are more likely to survive and pass on their genes to offspring, while less fit creatures die off.

3.1.2.2 Representation

To develop solutions to a problem using a genetic algorithm, the potential solutions must be encodable in such a way that allows mutation, crossover, and fitness evaluation. Two different, but related, encodings were used to represent bidders in an auction. Each is a generalization of an information processing structure known as a *finite state machine* (FSM) or *finite state automaton* (FSA). The first extension is known as a *GP-Automaton*, and the second is known as a *Neural-Automaton*. GP-Automata were first introduced by Ashlock in (Ashlock97). Neural-Automata are introduced here by the author for reasons explained in the description of Neural-Automata.

A finite state machine is a theoretical data processing structure. It takes input from some external source, changes its internal state in response to the input, and may or may not produce an output, known as a *response*. Mathematically, it is a directed graph, with each node called a *state*. Each state has associated with it a *transition table*. The transition table enumerates all possible inputs and states, for each input, what response to give and to what state to transition. Each directed edge in the graph, therefore, represents a transition from one state to another, and each edge's satellite data consists of an input that triggered the transition and a response to output. The FSM receives inputs sequentially, each time transitioning to a new state and outputting a response. Each transition has only one response in the model used in this paper, though in theory there could be more than one transition between two states, each with a different input and response. An FSM also has an initial state and an initial response, since it must start somewhere before it receives its first input. All computer programs may be thought of as finite state machines with memory.

Finite state machines are good structures to evolve, as the division of functionality into discrete states and transitions allow for natural choices of crossover operators. Finite state machines have been evolved to play simple economic games such as the Iterated Prisoner's Dilemma (Axelrod87) and as control structures for virtual robots (Ashlock00).

Because all possible inputs must be enumerated, FSM's are impractical to use on problems that have a large number of inputs. Even a single unbounded input renders the number of possible inputs infinite. Since the actual amount of information carried in the input tends actually to be much smaller, *bandwidth compression* is performed on the data. Bandwidth compression is a term borrowed from communications, and it refers to reducing the resources needed to represent some data. In this case, the bandwidth of the input data needs to be compressed to the set of integers in the range [0,number of states - 1], since this is the maximum number of possible transitions possible from any state.

Some method must be used to map all possible inputs into an integer in the range [0,number of states - 1]. GP-Automata and Neural-Automata provide two different methods of accomplishing this bandwidth compression.

3.1.2.3 GP-Automata

GP-Automata (GPA) have been described as a combination of genetic algorithms and genetic programming (hence the "GP-", for genetic programming). Each state in a GPA replaces the transition table with a parse tree whose input nodes are the external inputs to the

FSM. When a GPA iterates (takes an input and decides what to do) it runs the input through the parse tree at the current state. The parse tree outputs an integer, and the parity of this integer is used to determine the next state transition. Therefore, only two distinct transitions may be made from any state.

During evolution, the parse trees are modified according to the standard mutation and crossover operators used on parse trees in genetic programming. The exact details are given later.

Start: 130.651→2			
	If Even	If Odd	Deciders
0	254.072→2	160.181→2	(~ (ITE -2 2 (Odd -1)))
1	116.981→2	138.538→5	(ITE x_1 (Odd 0) (Odd 2))
2	208.991→0	124.01→4	(Odd (~ (ITE 0 2 x_1)))
3	67.2954→6	165.088→1	(~ (min (> 2 1) x_1))
4	165.586→7	200.285→4	(ITE (Odd 0) 0 (~ -1))
5	237.8→7	51.2287→3	(Com (ITE x_1 0 (~ x_1)))
6	142.16→7	39.2616→1	(ITE x_1 (Com x_1) (Odd 0))
7	141.045→7	151.488→2	(Com (Com (Odd (= x_1 0))))

Figure 3.3 Example 8-state GP-Automata

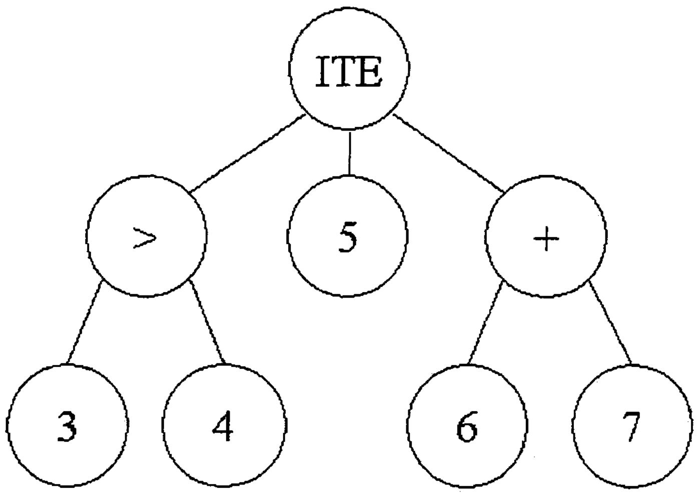

Figure 3.4 Example Parse Tree

Figure 3.3 shows an example GP-Automata with 8 states. The top line represents the initial response on the left side of the arrow, and the initial state on the right side of the arrow.

Likewise, the response and next state are shown for each state under the labels "If Even" and "If Odd". The "Deciders" are the parse trees used to decide next-state transitions and responses. The parse trees are shown in a LISP-like notation. Figure 3.4 shows a parse tree in a graphical notation. The tree in figure 3.4 would output the value 13. ITE is if-then-else, and would evaluate the subtree on the left to false, since $3 > 4$ is false. If it were true, it would output the middle child node, 5, but since it is false, ITE outputs the left child node, $6 + 7 = 13$.

3.1.2.4 Neural-Automata

While the GP-Automata scheme has the advantage of being adaptable to an arbitrary number of states, it has the disadvantage that each state may contain at most two next-state transitions. Such an FSM is strictly less expressive than an FSM that allows arbitrary transitions from any state to any other state. A proof of this follows.

Suppose we wish to design an FSM that takes nickels, dimes, or quarters as input and outputs a "1" if at least 15 cents have been entered cumulatively and outputs a "0" otherwise. The initial state would correspond to 0 cents being entered so far. After the first coin, either 5, 10, or 25 cents will be the total. Since $25 > 15$, but $5 < 15$ and $10 < 15$, the next state transition corresponding to a quarter being entered must be different from the transition resulting from the other two coins, since a quarter should cause a response of "1", and the other two should cause a response of "0". Since we allow only 2 transitions per state, then if either a nickel or a dime is entered, the FSM must end up in the same state regardless of which was entered.

If a dime was entered, then no matter what coin is input the next time, the cumulative total will be equal to or exceed 15 cents, and the FSM must therefore output a "1". However, if a nickel was entered the first time, then the total may or may not exceed 15 cents, depending on the second coin entered. However, since the first coin sends the FSM to the same state whether the input was a nickel or a dime, there is a contradiction. To handle the second coin, there must exist 3 possible states to go to from either the initial state or the second state.

Limiting the number of next state transitions to 2 renders the FSM incapable of executing this function. However, an FSM with no limit on transitions could easily execute this function. Therefore an FSM with only 2 next state transitions per state is strictly less expressive than a general FSM.

Neural-automata filter the input data while allowing an arbitrary number of next-state transitions. Instead of a parse tree, a neural-automata uses a feed-forward neural net at each state to decide the next state transition. The output node of the neural net uses a sigmoid transfer function

$$\frac{1}{1+e^{-x}}$$

where x is the weighted sum of the inputs. The output of this function is bounded in the range [0,1]. Therefore any number in the range [0 , number of states - 1] can be acquired by multiplying the output of the neural net by the number of states and truncating the result. Furthermore, since a feed-forward neural net with two hidden layers is capable of approximating any mathematical function (see (Cybenko88)), it is capable of approximating the expressive power of a parse tree.

3.2 Market Setup

A variety of experiments were performed. In each case, where applicable, three different representations were evolved. GP-Automata and Neural-Automata were each evolved with a genetic algorithm. The third representation was simply an ordered pair of real numbers (p, q), which represented a constant bid to make. The reason for introducing this representation is discussed shortly.

The market consisted of an auctioneer, representing the EMA, and any number of bidders, representing the competing generation companies. The auction proceeds as follows:

3.2.1 Auction Process

The auction process is shown in figure 3.5. The auctioneer announces a demand to be met. Bidders each submit a bid, which is an ordered pair (p, q) representing a price, p, and a

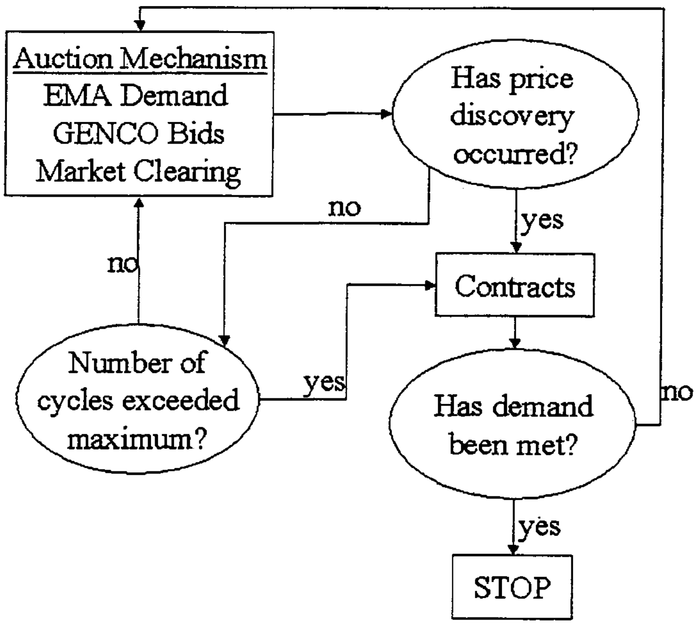

Figure 3.5 Auction Process

quantity, q. *Market clearing* then follows. These bids are sorted in ascending order by price. They are then accepted sequentially, adding the quantity of each bid to a running total until this total meets the demand. At this point no more bids are accepted, and the last one is accepted for only the amount of demand that remained, not for the total quantity given in the bid.

After market clearing, *price discovery* is tested. The condition used for price discovery in these experiments was whether at least 50% of the bids were accepted. If price discovery did not occur, new bids are taken and the market is cleared again. The results of the previous bids and market clearing are forgotten and do not affect profits. One round of bid submission and market clearing is termed a *cycle.* If after ten cycles, price discovery still has not occurred, typically the results of the last market clearing would be accepted. In these experiments, however, the the profits of all bidders was simply zeroed, in order to introduce selective pressure to make bids conducive to price discovery.

Once price discovery occurs, the bids are committed, and contracts are written. Now any bidder whose bid was accepted is obligated to provide the quantity of electricity accepted. If the total quantity accepted did not meet demand, then another auction is held with demand revised to be the previous demand minus the quantity accepted in the last auction. If after ten auctions, demand has not been met, then no more auctions are held.

3.2.2 Price and Cost Determination

There are two methods implemented in determining the price each bidder is paid. The actual money gained is the price the bidder is paid times the quantity accepted. The first is simply to pay the bidder the price it gave on its bid, known as *discriminatory pricing*. The second is to give it the *market clearing price*, defined as the price submitted by the last bidder whose bid was accepted. This is known as *uniform pricing*. The reasoning behind this is that the uniform market clearing price is the equilibrium price, or the price at which the supply curve crosses the demand curve.

Cost determination is modelled by a quadratic function of quantity, a common scheme for approximating the cost of producing electricity (Wood96). Each generation company may own more than one generator. However, once a generation company has a quantity it has committed to deliver, there are constrained optimization techniques, such as the use of Lagrangian multipliers, that can be used to find the optimal power production from each generator. That optimization is not explicitly performed in this simulation, and the total cost to a generation company to produce a given quantity is modelled as a single quadratic function of quantity. This is a simplifying assumption. In a real situation, any number of optimization techniques such as LaGrangian Relaxation, could be used to find the actual optimal power production for each unit. These optimizations would take into account the fact that a power plant may operate more than one unit, and that these units not only produce electricity at different costs, but are connected to transmission lines that charge different amounts to transport electricity.

Also, Real Options could be used to take the uncertainty due to different factors such as uncertainty about an opponent's bid, uncertainty about future demand, and uncertainty about future failures and inefficiencies in power plant operation, and uncertainty about congestion on transmission lines, and reduce all these uncertainties to a single number ??. This approach uses the mathematical tools traditionally used to value financial options, such as call and put orders, to analyze the monetary worth of options in real life.

In this work, each power producer was treated as if it had a single unit that had a single quadratic cost curve. This differs from previous work done in this area, in which the optimization was performed explicitly.

The total profit made by a seller after an auction is (price paid) $*$ (quantity delivered) $-$ (cost to deliver quantity). If more than one auction is held, total profit is the sum of the profits from each auction. All the auctions needed to meet demand are termed one *round* of auctions.

In the genetic algorithms, 24 rounds of auctions would be held against the same opponents for fitness evaluation, summing the total profit from each auction round. The number 24 was chosen to correspond with the 24 hourly auctions held in a single day. In the next generation, different opponents would be played, but the same opponents would always be played over the course of the 24 rounds of auctions.

The auctions being simulated in this work are representing day-ahead (forward) markets for electricity. This is why the auctioneer must estimate demand. The auction is held to determine distribution for each hour of the following day. In reality, an hourly auction would be held the next day in order to cover the difference between the estimated demand (which this forward market is designed to cover) and the actual demand (which may end up being different from the predicted value).

3.3 Experiments

3.3.1 Variable Parameters

There were a number of choices to be made concerning the implementation of a genetic algorithm simulating this market. Varying these choices led to the different experiments reported here.

3.3.1.1 Representations

Two different representations were evolved using a genetic algorithm, GP-Automata and Neural-Automata. In the discussion of FSMs previously, it was stated that they produce a response, but exactly what this response consists of was not covered. In these experiments, the response was an ordered pair (p, q), representing a bid price and bid quantity to submit. At the beginning of the 24 rounds of bidding, the FSMs would be reset (internal state set to initial state and initial response taken as first bid). After that, the each bidder would update its internal state and output the appropriate response in each subsequent auction. Note that because more than one cycle per auction may occur in order to achieve price discovery, and more than one auction may occur in order to meet demand for one round, there may be more than 24 bids taken from each bidder in one fitness evaluation.

The inputs fed to each FSM, after the first bid, were:

- Previous high bid
- Previous low bid
- Previous average bid
- Bidder's own previous bid
- Previous number of bids accepted
- Demand remaining to be met
- Quantity that the bidder has agreed to deliver so far

For each argument that is a bid, the FSM was actually fed two real numbers: the price and the quantity of the bid. "High" bid and "low" bid refer to the bid with the highest and lowest price, respectively, not the highest and lowest quantity. The "demand remaining to be met" refers to residual demand left over if an auction is held, all bids were accepted, and demand was not met. Since another auction will be held, agents need to know what the new demand is.

Additionally, a third representation was "evolved." This was simply an ordered pair (p, q), which represented a constant bid to make. However, crossover was not performed, and so this was not really a genetic algorithm. The bid was simply mutated every generation by adding Gaussian noise to each of the numbers in the bid.

The algorithm to develop these constant bids was therefore more of a *population-based stochastic search algorithm.* This is an algorithm which starts with a population of initial random solutions and perturbs each slightly with random noise, keeping and copying more optimal solutions over those that were less successful. The population was initialized with random bids, and at each "generation" the most fit half of the population replaced the least fit half of the population. Each newly copied bid was then perturbed by Gaussian noise.

The reason for using this representation was two-fold: to test the dynamics of the market, and to reject the null hypothesis that evolution of more complex FSM-based strategies is no better than random guessing. Initially, when debugging the simulated marketplace, the author needed to see if it would match the predictions of economic theory under perfect conditions, and how exactly the market would be affected by imperfect conditions. This representation should conform to theoretical predictions that say which single bid is optimal given bidders who attempt to maximize profit. Deviance from theoretical predictions would then point to a flaw in the implementation of the simulated marketplace or a theoretical assumption not being upheld. This aids in understanding the behavior of the more complex strategies in the same simulated environment.

3.3.1.2 Co-evolution vs. Fixed Fitness vs. Immortality

In general, fitness functions in evolutionary algorithms can be divided into two classes: co-evolutionary and fixed. A fixed fitness function evaluating one member of a population is independent of the other members of the population. A co-evolutionary fitness function evaluating the same member will give different results depending on the other members of the population. For example, a genetic algorithm evolving strategies to play chess might evaluate fitness by having the creatures play an expert alpha-beta strategy such as the one programmed into Deep Blue, IBM's grand champion chess-playing computer. This would be a fixed fitness function. The algorithm might alternately evaluate fitness by simply playing the strategies against each other in a round-robin tournament and averaging the scores. This would be a co-evolutionary fitness function.

The marketplace simulated has agents which are not identical to one another, the difference being that they have different production cost curves. Therefore, one is not as concerned with how an agent might perform against another agent like it, but how it would perform in a marketplace with agents different from it. The co-evolutionary strategy used therefore requires a bit of a re-definition (which actually brings it closer to the original definition in biology). Instead of evolving one population of agents, one population for each of the number of bidders in the market is evolved. When breeding is done each generation, members from a population are bred only with members from the same population. When fitness evaluation happens, however, the fitness of a member of a population depends only on members from the other populations, because the agent bids only against one member from each of the other populations. This is actually closer to the biological notion of co-evolution, in which two species are said to co-evolve with one another if "a change in one species acts as a new selective force on another species, and counteradaptation by the second species, in turn, affects selection on individuals in the first." (Campbell87)

A method of introducing a fixed fitness function when no external one exists is possible in cases where the agents being evolved play against each other. First, evolve using co-evolution for a certain number of generations. Then pick the best members of the population and "immortalize" (save) them. The fitness function now becomes the profits attained by playing against the immortal strategies, instead of playing against each other. A ratcheted immortalizing fitness function can be used as well. This involves immortalizing the best members of the population every n generations. Fitness is determined by playing all the sets of agents that have been immortalized so far. To account for space considerations, a shortcut may be used, such as "play the last 5 immortalized sets of agents." For example, if $n = 1000$, and we are on generation 8500, the fitness would be determined by playing the best agents from generations

4000, 5000, 6000, 7000, and 8000.

This sounds similar to *elitism*, but this is a distinct concept. An elite genetic algorithm is one in which at least one member of the current population is guaranteed to be a part of the population in the next generation. In other words, there is no way for the children of breeding to replace all the members of the population. However, this elite group differs from the immortal group introduced here in a number of ways. First, the immortal group is not considered when breeding. Only non-immortal members of the population can breed to create children. Secondly, the immortal group is the fitness function. An elite group in a normal genetic algorithm has no special role in the fitness function. Finally, the immortal group is the same from one generation to the next, unless explicitly replaced. The elite "group" in the normal genetic algorithm, however, is defined by the current population. Usually an elite member is one with the highest fitness. However, it may not have the highest fitness during subsequent generations.

3.3.1.3 Uniform vs Discriminatory Pricing

Under uniform pricing, all "winners" (bidders whose bids were accepted) pay the same amount. In the simplest case, this amount is the market clearing price, the highest price that was accepted. The theoretical justification for this is that the highest accepted price in an auction is the price at which the supply curve would intersect the demand curve. This is the equilibrium price, which is the price that all trades should be made at in a perfectly competitive market.

Under discriminatory pricing, winners pay different amounts. In the simplest case, this amount is the price they bid. In any auction where there is more than one winner, this would give more money to the auctioneer at the expense of the sellers. However, it would also tend to inflate the bids above the bidders' marginal costs, which may actually lead to less profit for the seller in the long run.

3.3.1.4 Number of Bidders

In a market with few bidders, each bidder has more market power. Bidders with market power have enough influence to affect the outcome of the auction. In a uniform pricing scheme, one would say they have the power to change the price. In the context of an auction, the most immediate effect of having few bidders is that it becomes easy for bidders to cooperate with each other and raise prices by raising their bids. Alternately, a bidder could withhold capacity in order to drive the price up and undercut it later.

3.3.1.5 Cost Curves and Capacity Limits

Different producers will have different generation cost curves. This models the real world fact that some electricity producers are coal-driven, some are oil-driven, some are nuclear, some have many generators, and some may only have one. They also have different capacity limits, or minimum and maximum quantities that they are able to produce. This models the physical limitations of power plants. The fact that different bidders have different cost curves and capacity limits is the motivation behind separating the genetic algorithm into separate populations. Each population represents a different type of producer and is evolving to get better at bidding under the constraint of its own cost curve and capacity limits. Note that previous studies in this area assumed that all power producers had similar cost curves and capacity limits.

3.3.2 Genetic Algorithm Parameters

There are many generic parameters of genetic algorithms. Some, such as mutation and crossover operators, seriously affect the outcome of the algorithm. Some, such as population size and number of generations, are merely tweaked according to the problem at hand, usually by making them as big as possible while allowing for the program to finish running in a reasonable amount of time.

Unless noted otherwise, these experiments used a population size of 32 and ran for 1000 generations. The model of evolution (method of breeding creatures with high fitness) used for the GP-Automata and Neural-Automata is known as *single tournament selection.* See figure 3.6 for a diagram showing how single tournament selection works.

Creatures before Breeding

Creature	Fitness
Creature 1	6
Creature 2	5
Creature 3	7
Creature 4	2
Creature 5	7
Creature 6	1
Creature 7	2
Creature 8	3

Creatures after Breeding

Creature	Fitness
Creature 1	6
Child of 1 and 3	?
Creature 3	7
Child of 1 and 3	?
Creature 5	7
Child of 5 and 8	?
Child of 5 and 8	?
Creature 8	3

Figure 3.6 Single Tournament Selection of Size 4

In this model of evolution, the population is divided into sets of four creatures. Within each set of four, the two most fit are picked as parents. They are then bred (copied, crossed over, and then mutated) and their children replace the two least fit. Since the fixed-bid representation did not use a genetic algorithm, in each "generation" the most fit half of the population would replace the least fit half, and then the newly copied agents would be "mutated".

The mutation and crossover operators depended on the representation. For both of the FSM-based representations, there were a number of possible mutation and crossover operators that could be chosen. Each was assigned a probability and picked to be the mutation or crossover operator with that probability. The *mutation rate* was a number between 0 and 1 that correlated with the severity of mutation. If the number of possible mutation operators was given by m, and the mutation rate given by r, then $m*r$ mutations would be performed on each child in each generation. Each mutation, the probability of picking any of the individual mutation operators remained the same.

3.3.2.1 Fixed Bid Mutation

Since the fixed bid representation did not use a genetic algorithm, there is no crossover to speak of. However, it randomly perturbed the bid between each fitness evaluation, like a mutation. Both the price and quantity were perturbed by Gaussian noise with standard deviation equal to $\frac{\text{maximum}-\text{minimum}}{10}$. Recall that the minimum and maximum price were 0 and 120, respectively, and the minimum and maximum quantity depended on the capacity limits of the producer.

3.3.2.2 GP-Automata Mutation

The GP-Automata mutation operator selected one of the mutation operators listed in table 3.1.

Table 3.1 GP-Automata Mutation Operators

Probability	**Mutation**
0.1	Change initial state
0.1	Perturb initial response
0.2	Change a state transition
0.2	Change a response
0.1	New decider parse tree
0.1	Crossover two decider parse trees
0.1	Exchange two decider parse trees
0.1	Copy one decider parse tree over another

A state (initial state or state transition) was changed by randomly selecting a new state uniformly from the set of states. A response was perturbed by adding gaussian noise with standard deviation $\frac{\text{maximum}-\text{minimum}}{10}$. Since a response consisted of both a price and a quantity, both of these were perturbed. In the case that a state transition or response was changed, the particular transition edge to mutate was selected uniformly from all the state transitions. A new parse tree was created by randomly generating nodes until the tree was of size 6 (had 6 nodes). Crossover between parse trees involved randomly selecting a node in each tree and exchanging the subtrees rooted at those nodes. The trees on which to perform crossover were randomly selected uniformly from all the decider trees.

3.3.2.3 Neural-Automata Mutation

The Neural-Automata mutation operator selected one of the mutation operators listed in table 3.2.

Table 3.2 Neural-Automata Mutation Operators

Probability	Mutation
0.1	Change initial state
0.1	Perturb initial response
0.2	Change a state transition
0.2	Change a response
0.1	Mutate decider neural net
0.1	Crossover decider neural nets
0.1	Exchange decider neural nets
0.1	Copy one decider neural net over another

These mutation operators follow the same rules as those of GP-Automata. Crossover between neural nets involved randomly selecting the indices of two edges within the neural nets and exchanging the values of the edges in between these.

3.3.2.4 GP-Automata Crossover

The GP-Automata mutation operator selected one of the crossover operators listed in table 3.3.

Table 3.3 GP-Automata Crossover Operators

Probability	Mutation
0.2	Exchange initial states
0.2	Exchange initial responses
0.6	Exchange states

If the "exchange states" operator was selected, two random indices were selected, and all states in between these indices were exchanged.

3.3.2.5 Neural-Automata Crossover

The Neural-Automata mutation operator selected one of the crossover operators listed in table 3.4.

Table 3.4 Neural-Automata Crossover Operators

Probability	**Mutation**
0.2	Exchange initial states
0.2	Exchange initial responses
0.6	Exchange states

If the "exchange states" operator was selected, two random indices were selected, and all states in between these indices were exchanged.

3.3.2.6 Fixed-Bid Initialization

The two numbers constituting the fixed bids were initialized to a random number distributed uniformly over the possible range of each number. Price varied from 0 to 120, and quantity depended on the capacity limits of the generator.

3.3.2.7 GP-Automata Initialization

The FSM was initialized with 6 states. The initial response was initialized in the same manner as the fixed-bid representation. Each transition next state and response was initialized in the same manner as the initial state and transition. The decider parse tree was initialized to a random parse tree with 6 nodes. A sample parse tree is shown in figure 3.4. The nodes could be any of those shown in table 3.5. These nodes were inserted at random until a tree of size 6 was obtained.

3.3.2.8 Neural-Automata Initialization

The FSM was initialized in the same manner as the GP-Automata. The decider neural net was initialized to a random feed-forward neural net with two hidden layers. Each hidden layer had 3 nodes. The weights were initialized to random values distributed uniformly in the range $[-1, 1]$.

Table 3.5 Parse Tree Nodes

Node	**Name**	**Return Type**	**Args**	**Returns**
ITE	if-then-else	args 2 and 3	3	arg 2 if arg 1 is true; arg3 3 otherwise
Odd	odd	Boolean	1	true if arg1 is odd, false otherwise
Max	maximum	Real	2	maximum of args 1 and 2
Min	minimum	Real	2	minimum of args 1 and 2
~	negation	Real	2	negation of arg 1
Com	complement	Real	2	1 - arg 1
>	greater than	Boolean	2	true if arg 1 $>$ arg2; false otherwise
>=	greater than or equal to	Boolean	2	true if arg 1 $\geq$ arg2; false otherwise
<	less than	Boolean	2	true if arg 1 $<$ arg2; false otherwise
<=	less than or equal to	Boolean	2	true if arg 1 $\leq$ arg2; false otherwise
+	add	Real	2	arg 1 + arg 2
-	subtraction	Real	2	arg 1 - arg 2

CHAPTER 4. RESULTS

This chapter describes the experiments performed and the results obtained.

4.1 Introduction

The experiments performed were based on changing the variable parameters discussed in chapter 3. In the cases of co-evolution, all three of the representations were run and compared. In the cases of evolution by periodic immortalization, only the finite state automata were compared.

Each experiment also shows the average bid and average fitness of the whole population versus generation of evolution. In most cases, a single run the algorithm is shown in addition to an average of many runs of the algorithm to demonstrate general behavior.

The experiments were divided into two general cases: evolution by co-evolution and evolution by periodic immortalization. Co-evolution has the potential to develop good strategies, but since fitness will not necessarily prove to be an effective measure of improvement of the strategies. Evolution by periodic immortalization, however, evolves the bidders against a set of fixed strategies after the first 1000 generations. Therefore, if the bidders are learning better bidding strategies, we should see the average fitness of a population increase with this evolutionary scheme.

Unless stated otherwise, all experiments used a mutation rate of 0.5. Each figure shows the average fitness (labelled as "Fitness"), average bid price (labelled as "Bid"), average committed bid price (labelled as "Committed Bid"; does not count bids that were taken in cycles that failed to meet the condition of price discovery), and equilibrium price (labelled as "Equilibrium Price"; taken to be the price or the last bid that was accepted), each graphed against generation of evolution.

4.2 Co-evolution

The experiments described in this section developed the bidders through co-evolution. These experiments tested all 3 representations, GP-Automata, Neural-Automata, and Fixed-Bid. The Fixed-Bid equilibrium behavior in each case should give a decent approximation to expected theoretical behavior. In many cases it gives an equilibrium price higher than expected until ones takes into account the effect of market power on price.

4.2.1 Experiment 1: Co-Evolutionary Fitness Function, Discriminatory Price

Table 4.1 Variations on Experiment 1

Figures	Bidders	Demand	Cost Curve / Min / Max 1	Cost Curve / Min / Max 2
4.1, 4.2, 4.3	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.4, 4.5, 4.6	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.7, 4.8, 4.9	10	2000	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.004q^2 + 5.3q + 500)$ / 200 / 500
4.10, 4.11, 4.12	10	2000	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.012q^2 + 15q + 1500)$ / 100 / 300
4.13, 4.14, 4.15	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.16, 4.17, 4.18	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.19, 4.20, 4.21	4	800	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.004q^2 + 5.3q + 500)$ / 200 / 500
4.22, 4.23, 4.24	4	800	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.012q^2 + 15q + 1500)$ / 100 / 300

This experiment used discriminatory pricing. Recall that with discriminatory pricing, each bidder, if it has a bid accepted, gets paid the price it listed in the bid. Variations on the cost curves and capacity limits and number of bidders were explored. The different trials are shown in table 4.1. Each of the three figures listed refers to the graphed results of a particular representation (GP-Automata, Neural-Automata, Fixed-Bid). "Cost Curve / Min / Max" refer to the cost curves and upper and lower capacity limits, respectively. In all cases, either a single cost curve and capacity limits were used for all bidders, or the population was split into two groups, each of which had its own set of cost curves and capacity limits. In these cases, if there were 10 bidders, 7 of them would have Cost Curve 1 and Min/Max 1 capacity limits, and the other 3 would have Cost Curve 2 and Min/Max 2 capacity limits. If there were 4 bidders, 2 of them would have Cost Curve 1 and Min/Max 1 capacity limits, and the other 2 would have Cost Curve 2 and Min/Max 2 capacity limits. In the cases that every bidder had the same cost curve, these quantities are equal in table 4.1.

Demand was set to be proportional to the number of bidders and to the average capacity limits of the bidders. The demand listed is the average demand, d, generated for each auction. The actual demand for each auction was drawn from a uniform probability distribution in the range $\left[d - \frac{d}{10}, d + \frac{d}{10}\right]$.

4.2.1.1 Auction Size 10

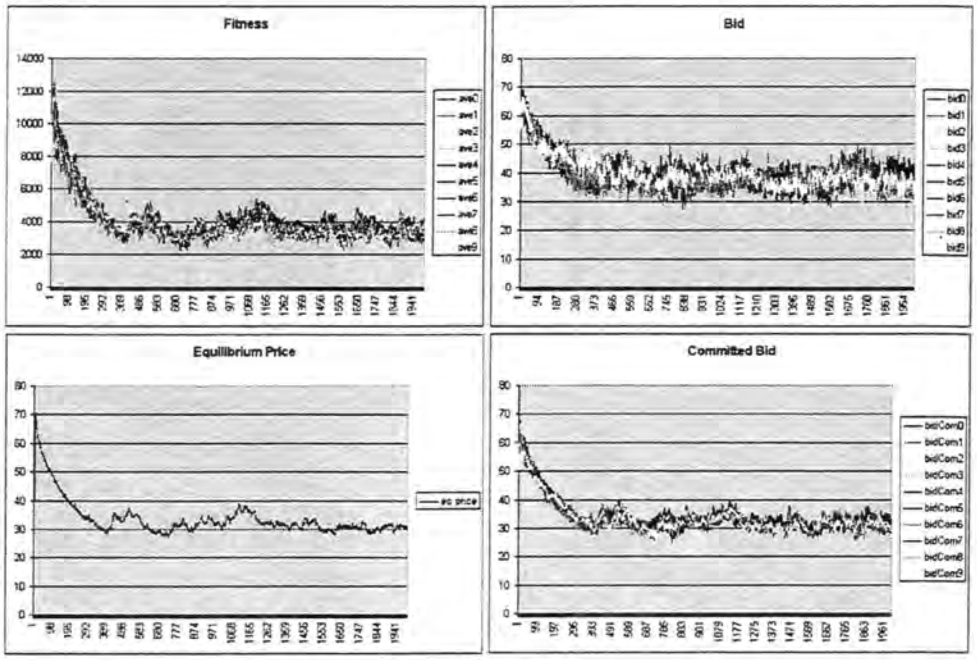

Figure 4.1 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.1 through 4.12 show four different cost curve and capacity limit choices for all three representations using the co-evolutionary fitness function and discriminatory pricing, with an auction size of 10 bidders. Examining the fixed-bid representation as a guide to the expected behavior of a market, a few general trends are evident. These trends are in place in the other two representations as well, although they appear amid the more complex behavior that the FSM's display.

Not surprisingly, when the bidders have different cost curves, those with a higher cost curve get a lower fitness. Higher capacity limits lead to a lower equilibrium price. This is due to the demand (which is independent of the capacity limits of the bidders) being more easily met when the capacity limits are higher. Therefore more competition is present, driving the price down.

Note that in the first figures 4.1 through 4.6, the GP-Automata and Neural-Automata consistently achieved a higher equilibrium price than the fixed-bid representation.

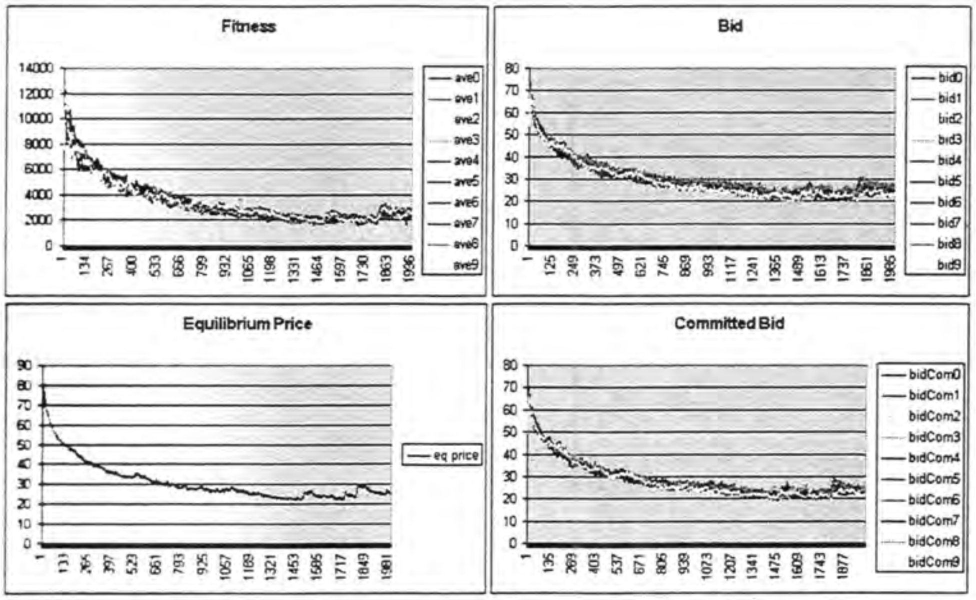

Figure 4.2 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

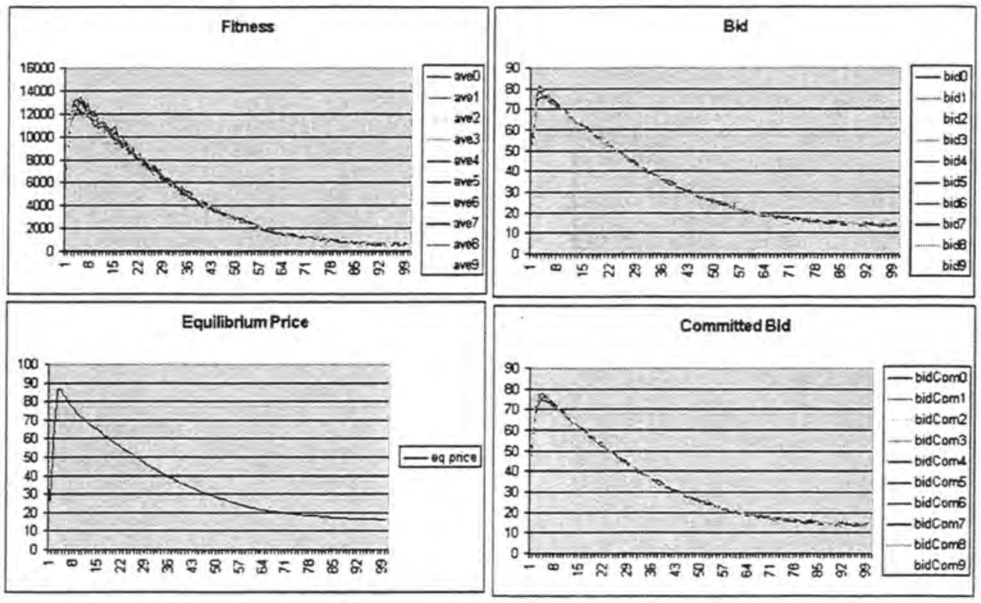

Figure 4.3 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

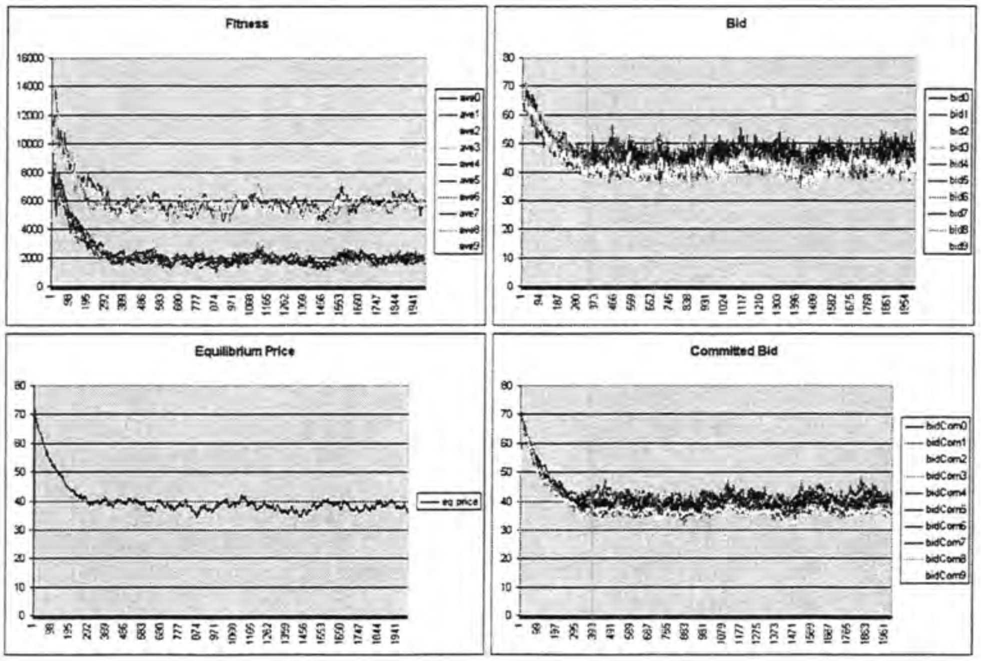

Figure 4.4 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

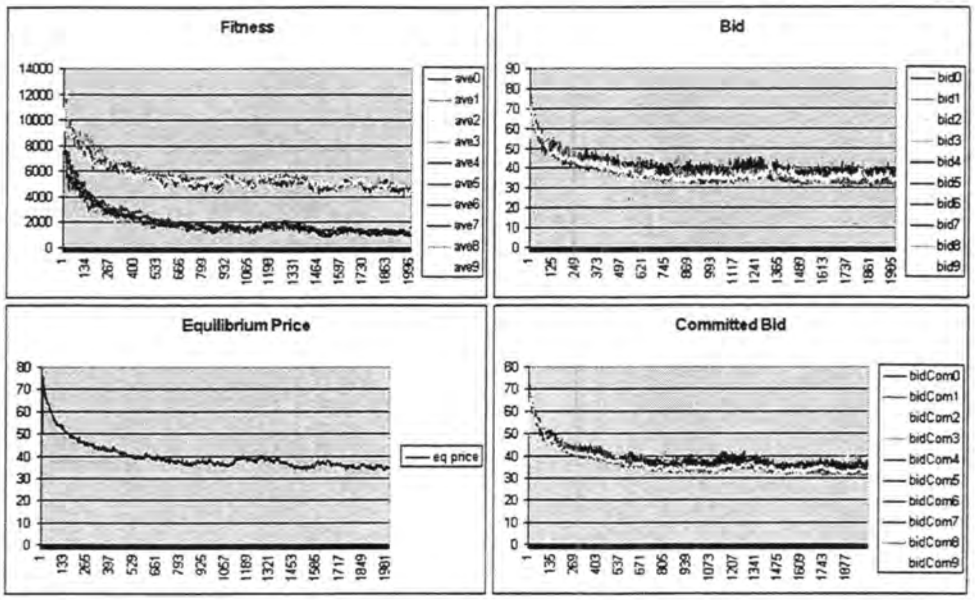

Figure 4.5 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

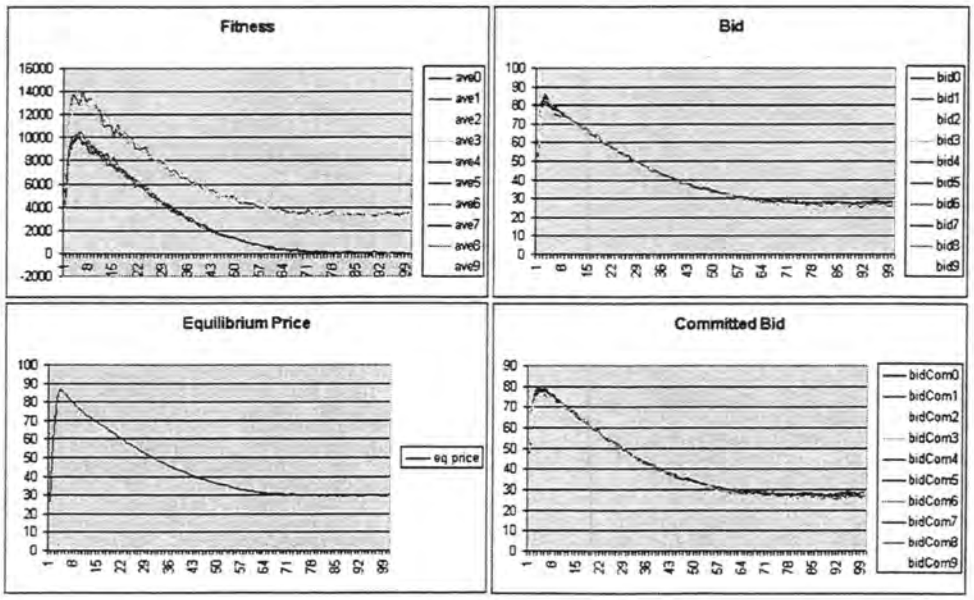

Figure 4.6 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

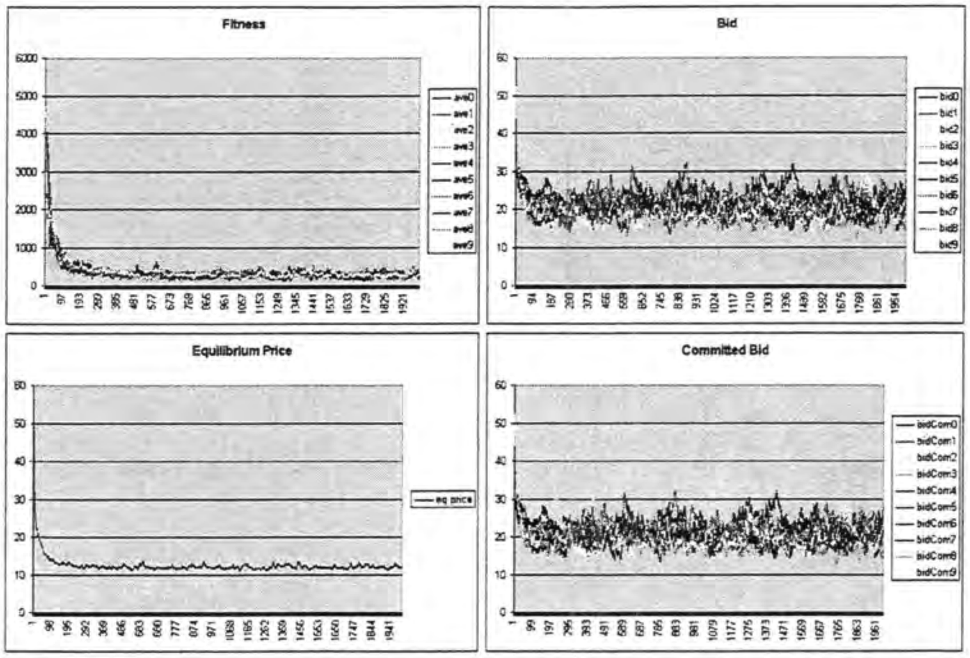

Figure 4.7 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

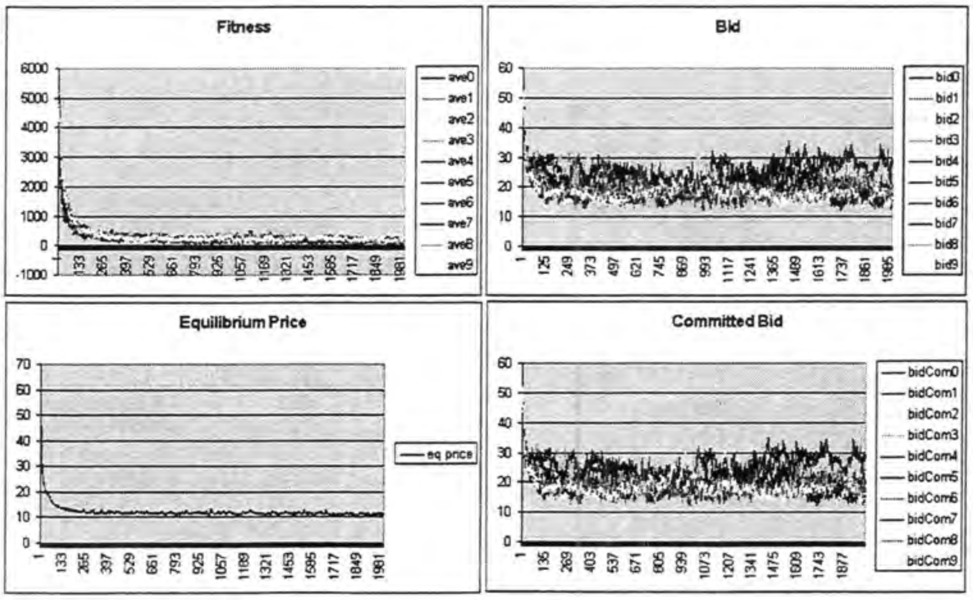

Figure 4.8 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

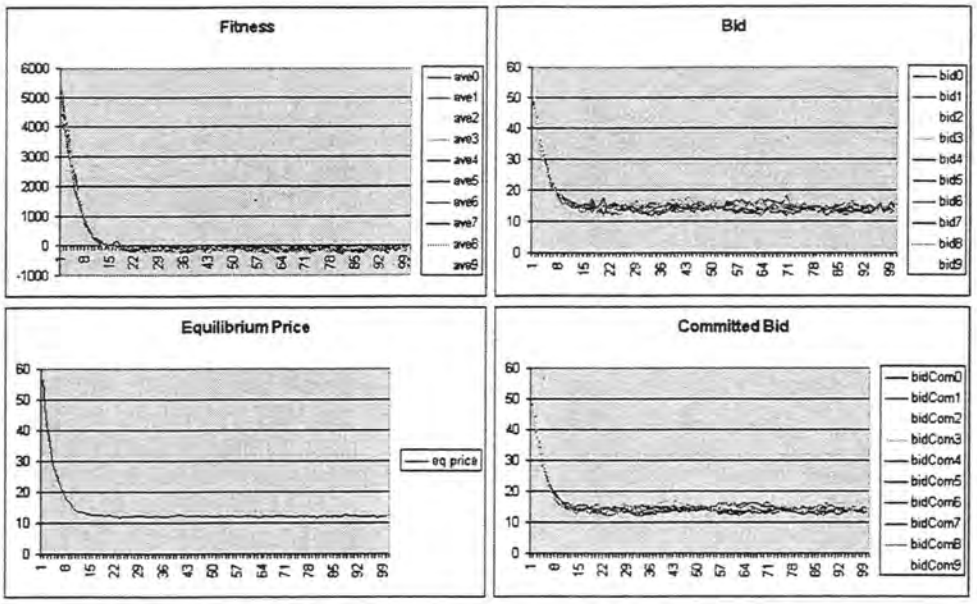

Figure 4.9 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

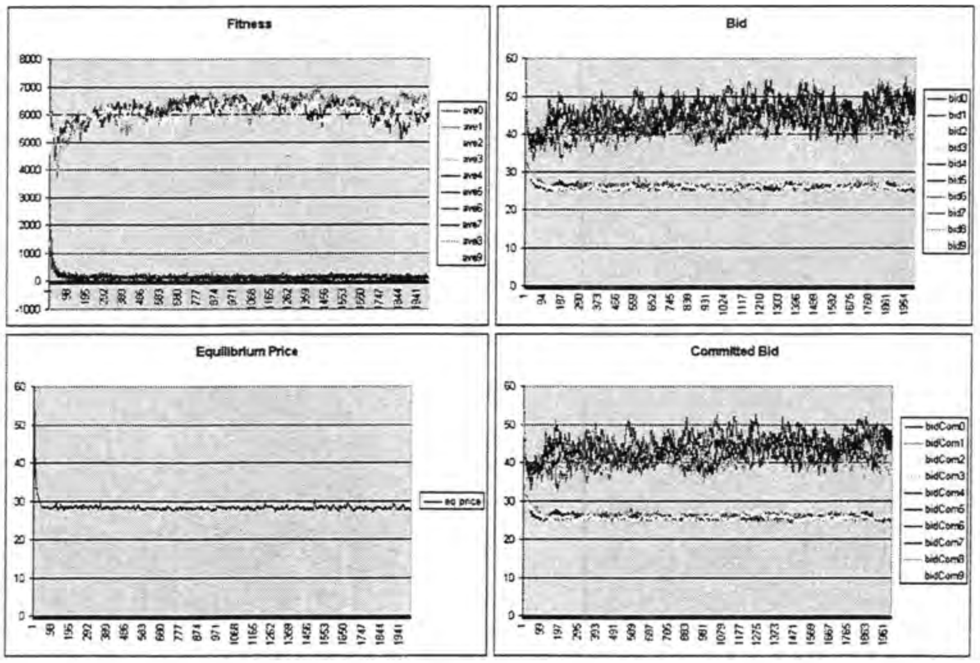

Figure 4.10 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

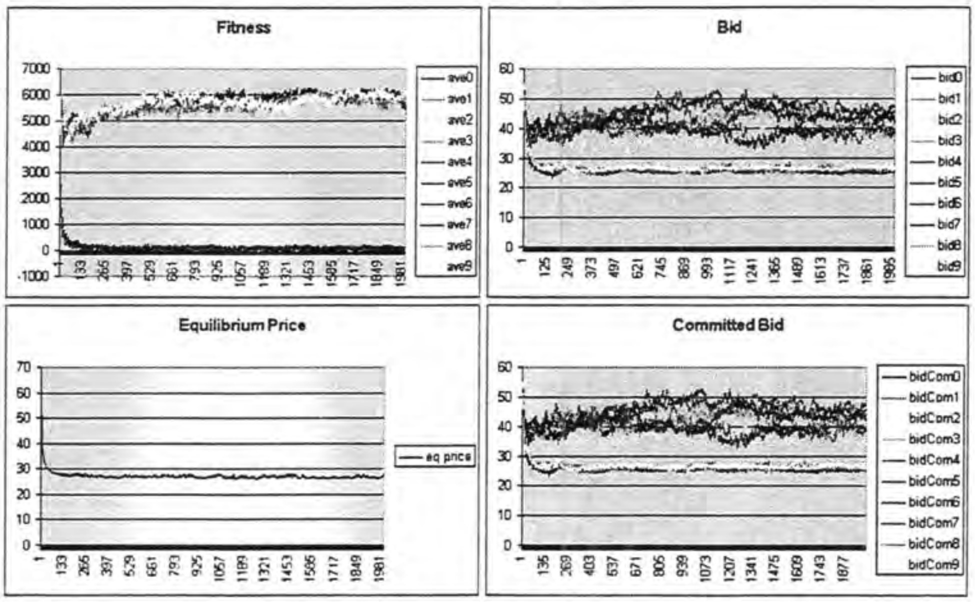

Figure 4.11 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

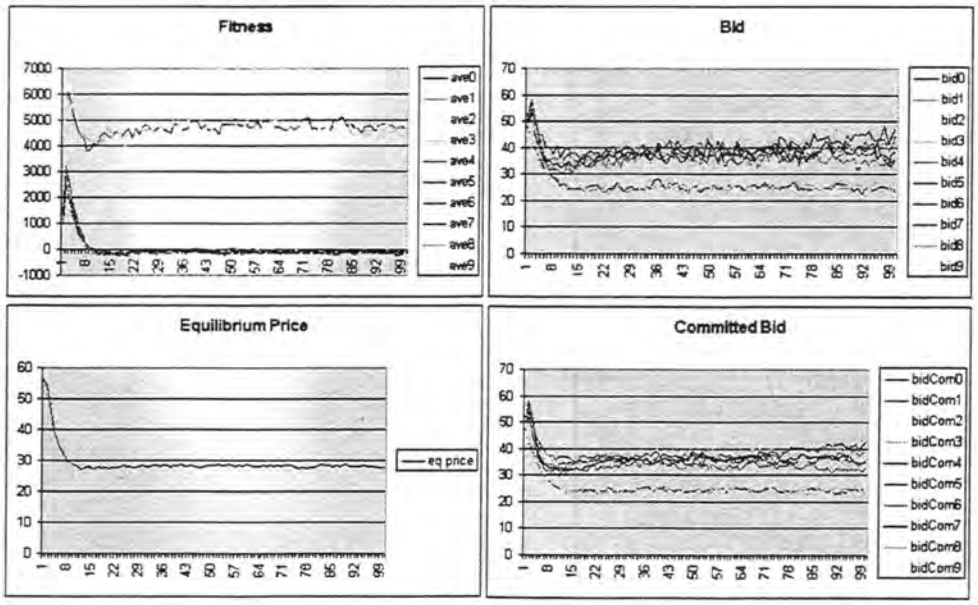

Figure 4.12 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

4.2.1.2 Auction Size 4

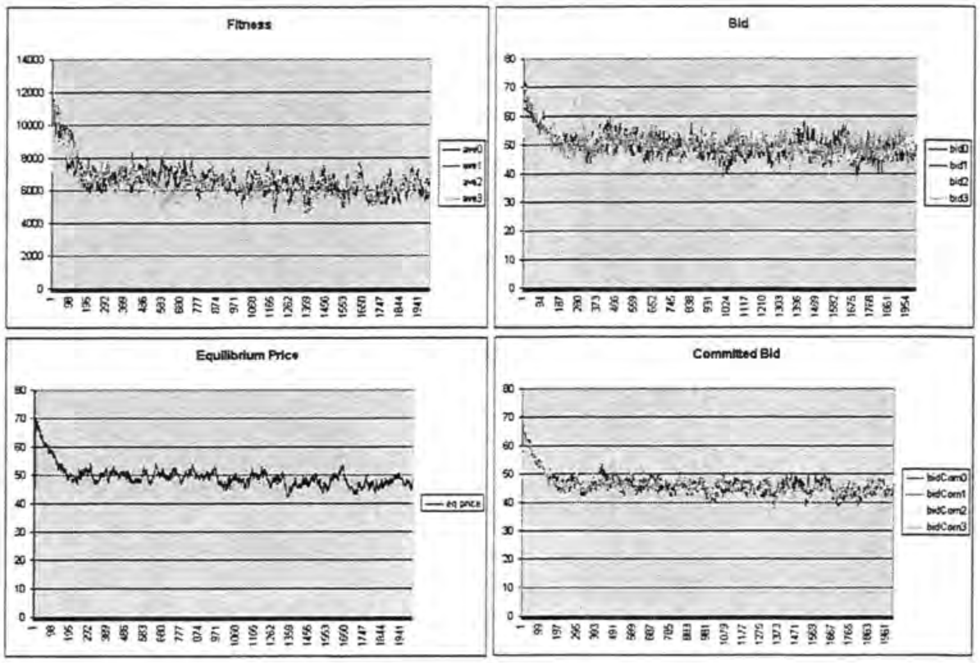

Figure 4.13 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.13 through 4.24 show four different cost curve and capacity limit choices for all three representations using the co-evolutionary fitness function and discriminatory pricing, with an auction size of 4 bidders.

The effects noted in section 4.2.1.1 are present here as well. Note, however, that the average equilibrium prices achieved are higher than those achieved with 10 bidders, if one compares figures 4.1 through 4.12 to figures 4.13 through 4.24, respectively (the cost curve and capacity limits are varied in the same order in this section as in section 4.2.1.1, so one may compare any figure n in this section to figure $n - 12$ in section 4.2.1.1). This illustrates the effect of the number of market participants on the market power of each individual market participant; i.e. less participants gives each participant more market power.

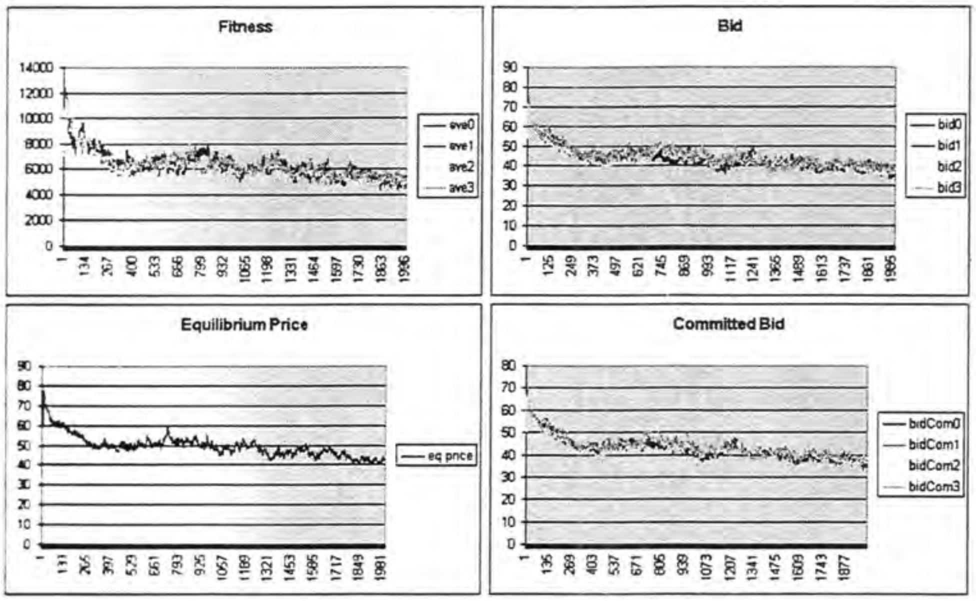

Figure 4.14 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

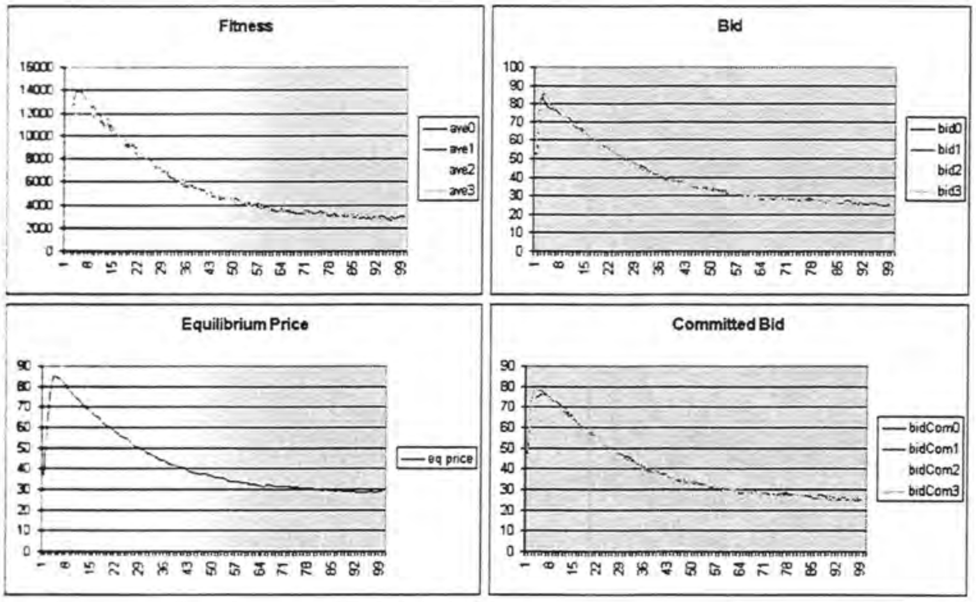

Figure 4.15 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

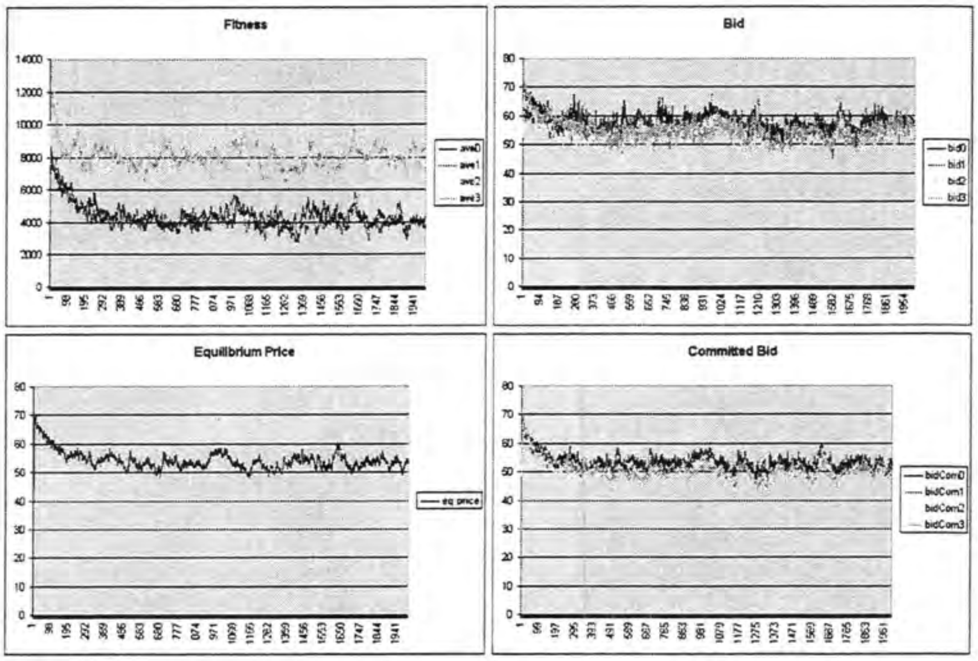

Figure 4.16 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

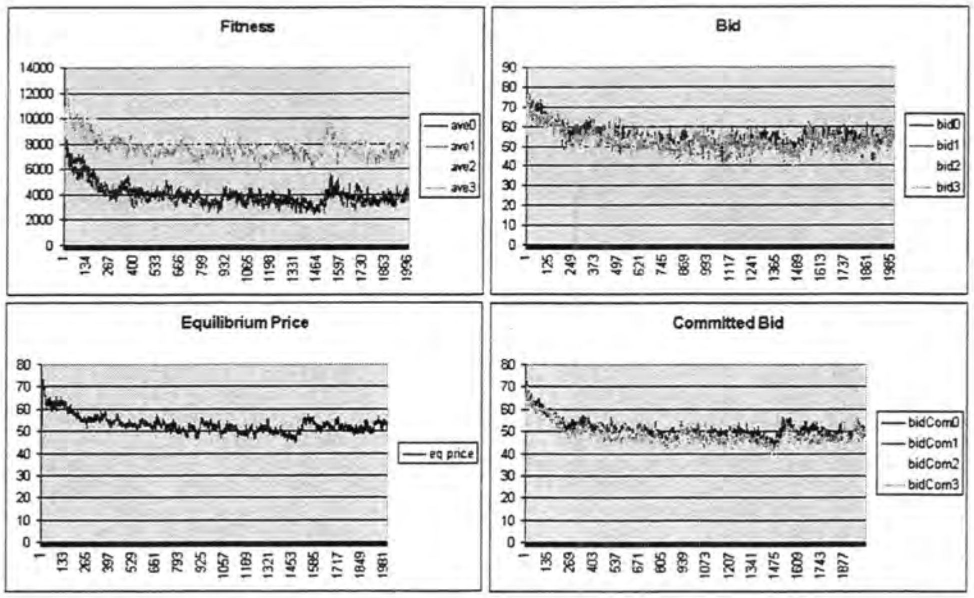

Figure 4.17 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

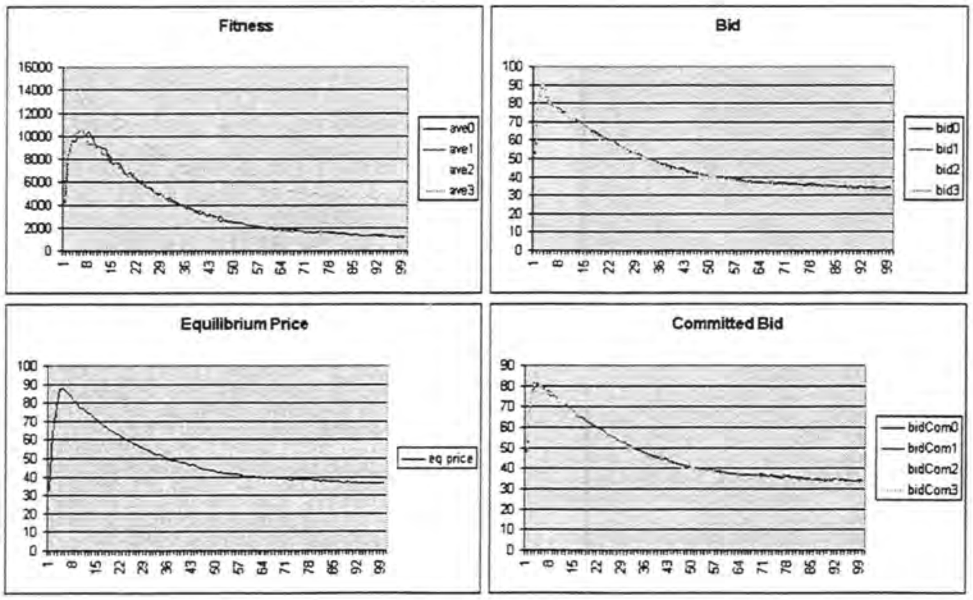

Figure 4.18 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

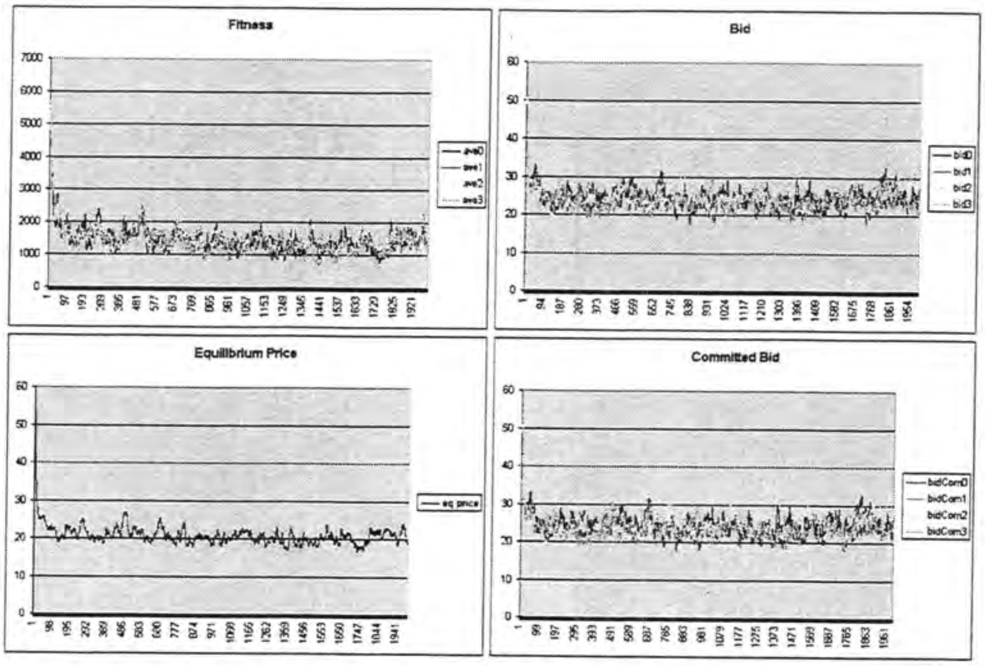

Figure 4.19 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

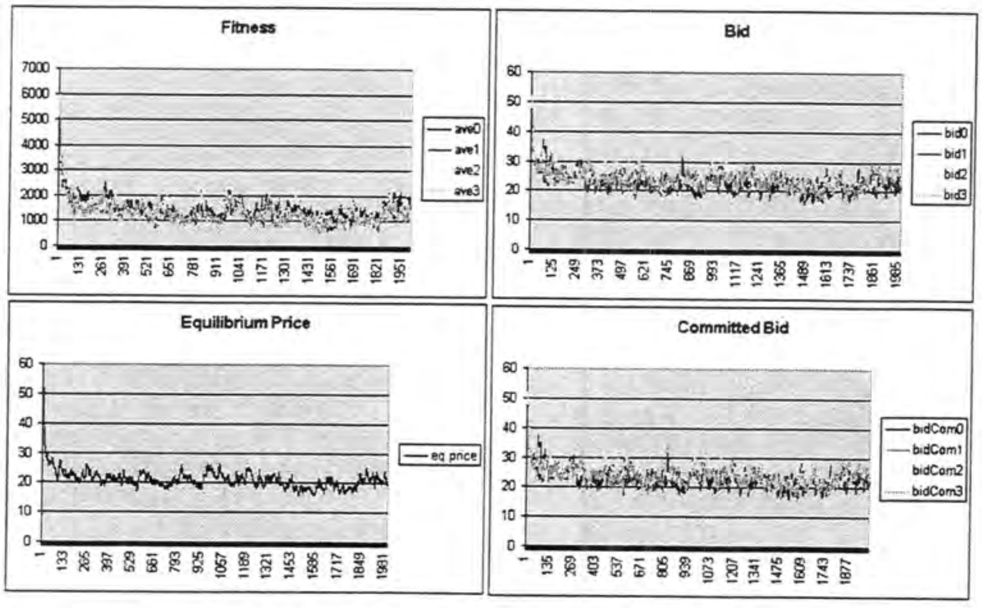

Figure 4.20 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

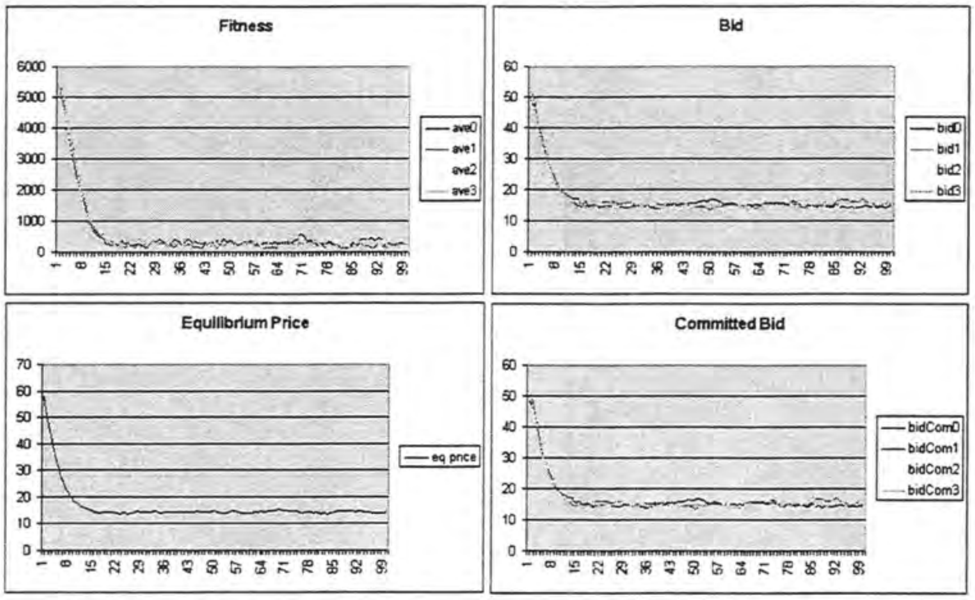

Figure 4.21 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

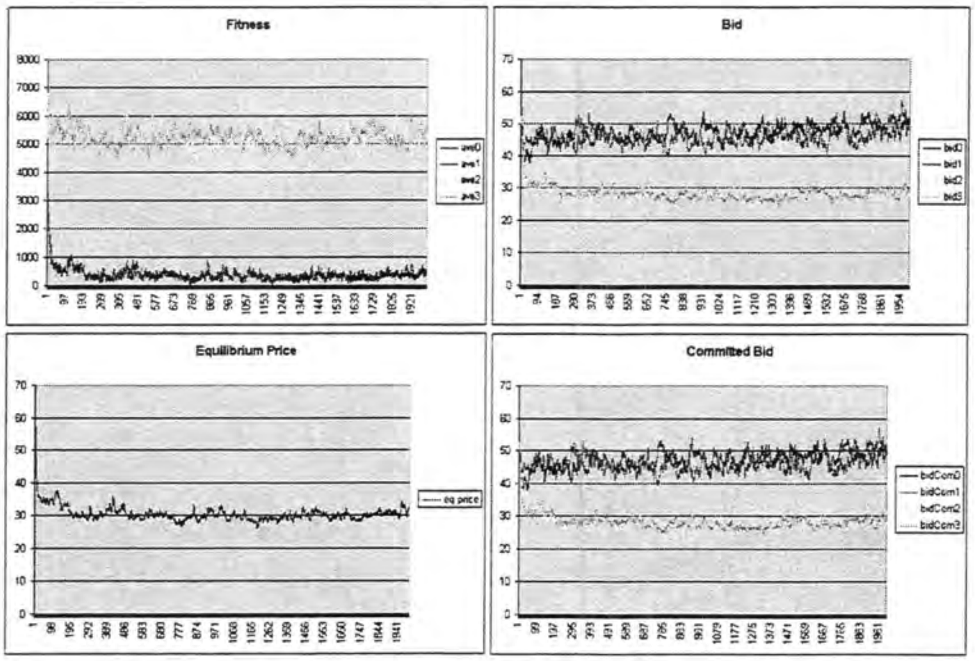

Figure 4.22 Co-Evolutionary Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

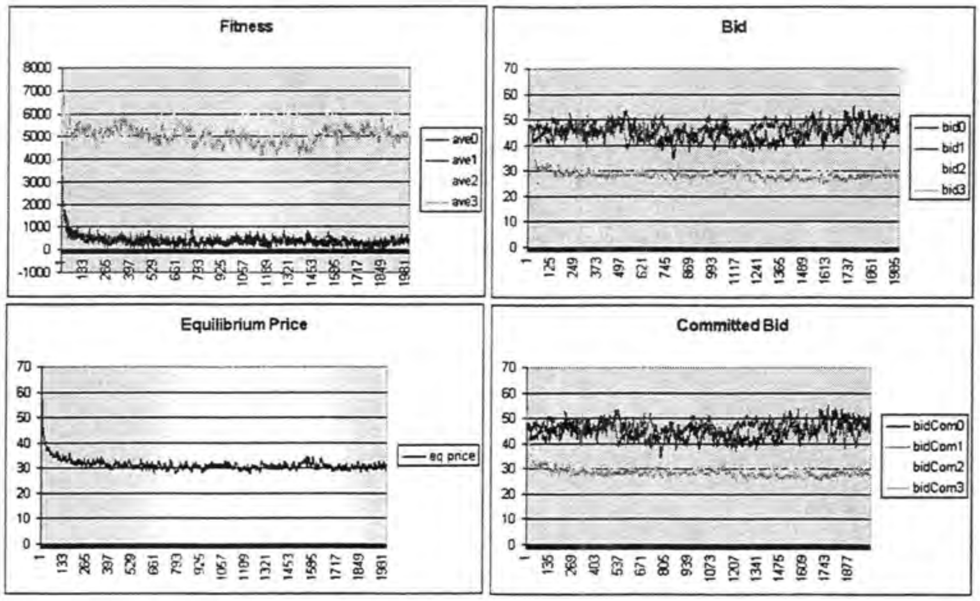

Figure 4.23 Co-Evolutionary Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

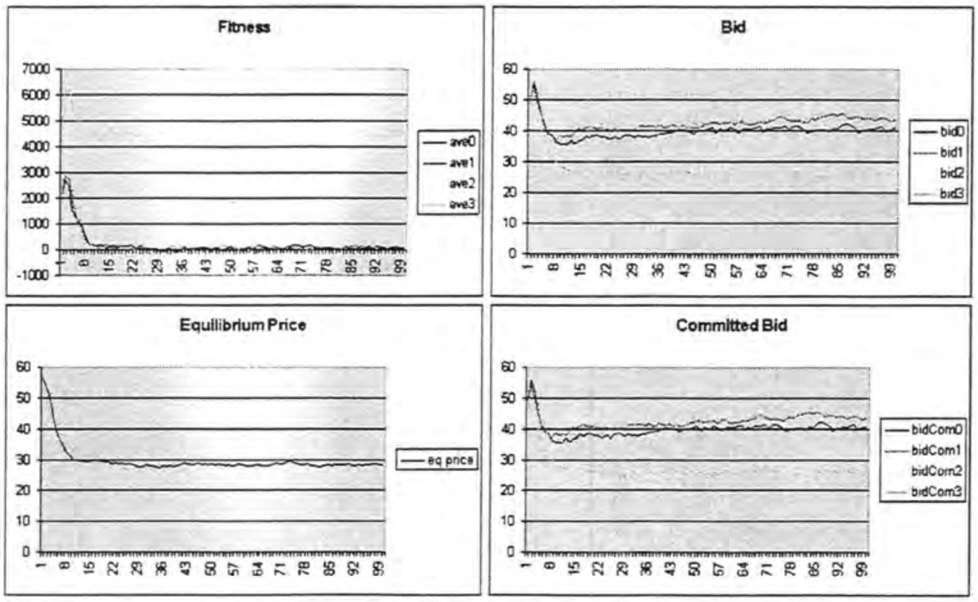

Figure 4.24 Co-Evolutionary Fitness Function, Discriminatory Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

4.2.2 Experiment 2: Co-Evolutionary Fitness Function, Uniform Price

Table 4.2 Variations on Experiment 2

Figures	Bidders	Demand	Cost Curve / Min / Max 1	Cost Curve / Min / Max 2
4.1, 4.2, 4.3	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.4, 4.5, 4.6	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.7, 4.8, 4.9	10	2000	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.004q^2 + 5.3q + 500)$ / 200 / 500
4.10, 4.11, 4.12	10	2000	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.012q^2 + 15q + 1500)$ / 100 / 300
4.13, 4.14, 4.15	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.16, 4.17, 4.18	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.19, 4.20, 4.21	4	800	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.004q^2 + 5.3q + 500)$ / 200 / 500
4.22, 4.23, 4.24	4	800	$(0.004q^2 + 5.3q + 500)$ / 200 / 500	$(0.012q^2 + 15q + 1500)$ / 100 / 300

This experiment used uniform pricing. Recall that with uniform pricing, each bidder, if it has a bid accepted, gets paid the equilibrium price, or the price of the highest accepted bid. The rest of the details are identical to those described in section 4.2.1.

4.2.2.1 Auction Size 10

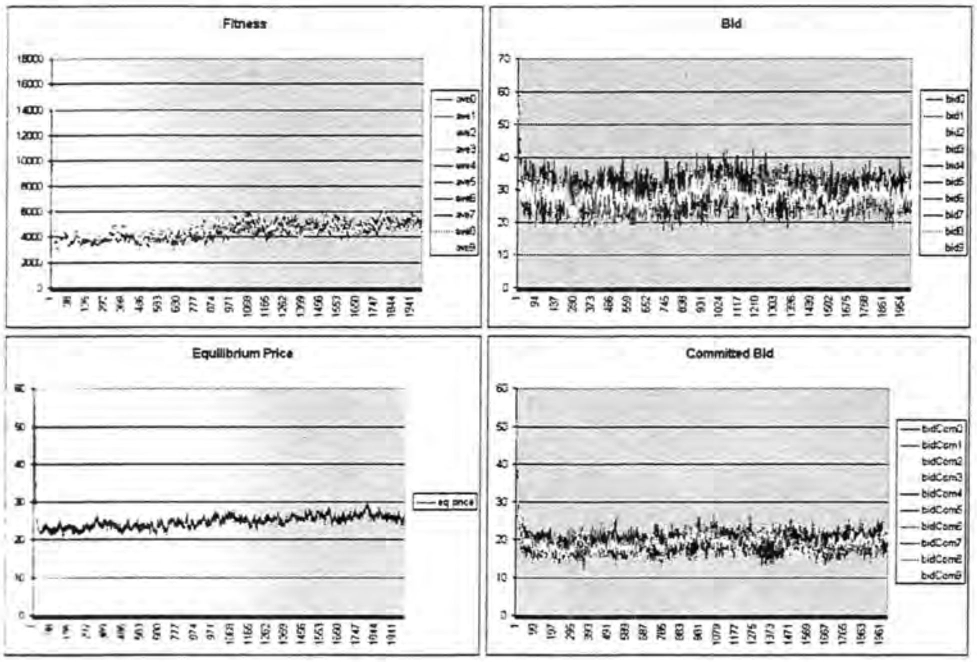

Figure 4.25 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.25 through 4.36 show four different cost curve and capacity limit choices for all three representations using the co-evolutionary fitness function and uniform pricing, with an auction size of 10 bidders.

Comparing these results to those obtained with the same conditions except for using discriminatory pricing, we see that on average, the equilibrium price and bids are lower under a uniform pricing scheme. This makes intuitive sense economically. When one's own bid price determines whether the bidder wins or not, and the bidder's payoff varies with the size of the bid, then there is a tradeoff. A lower bid, for instance, has a better chance of winning, but results in a lower payoff. However, under uniform pricing, one's own bid determines whether one wins or not, but has no effect on the bidder's payoff (unless it was the last accepted bid). Therefore the pressure to bid higher for a higher payoff is removed, and only the pressure to bid lower for a better chance of winning remains.

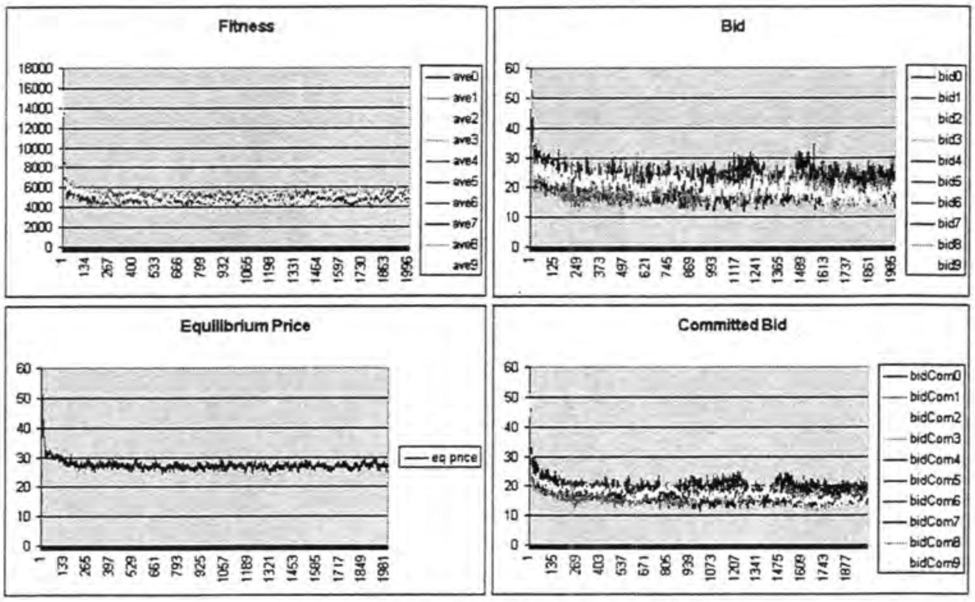

Figure 4.26 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

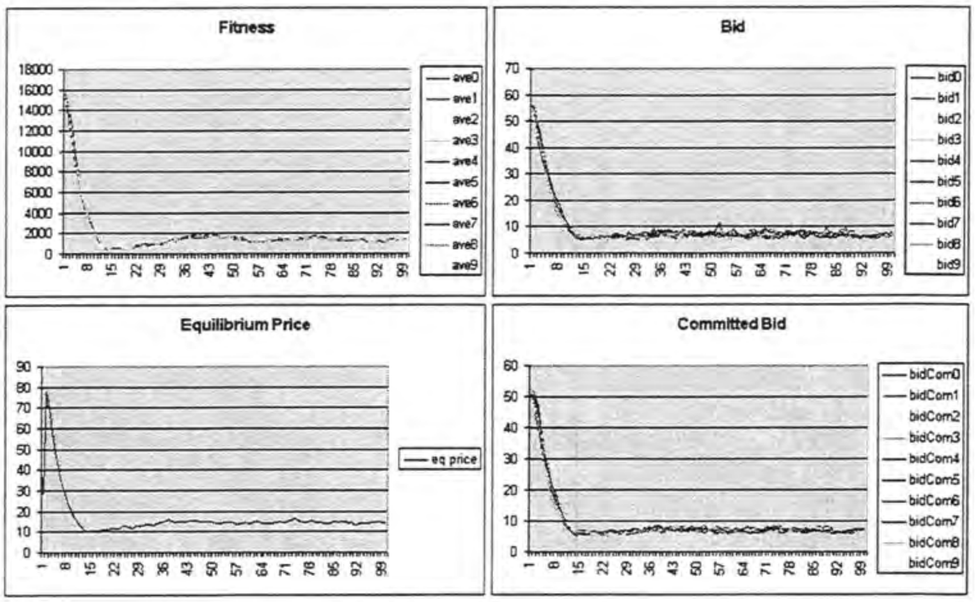

Figure 4.27 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

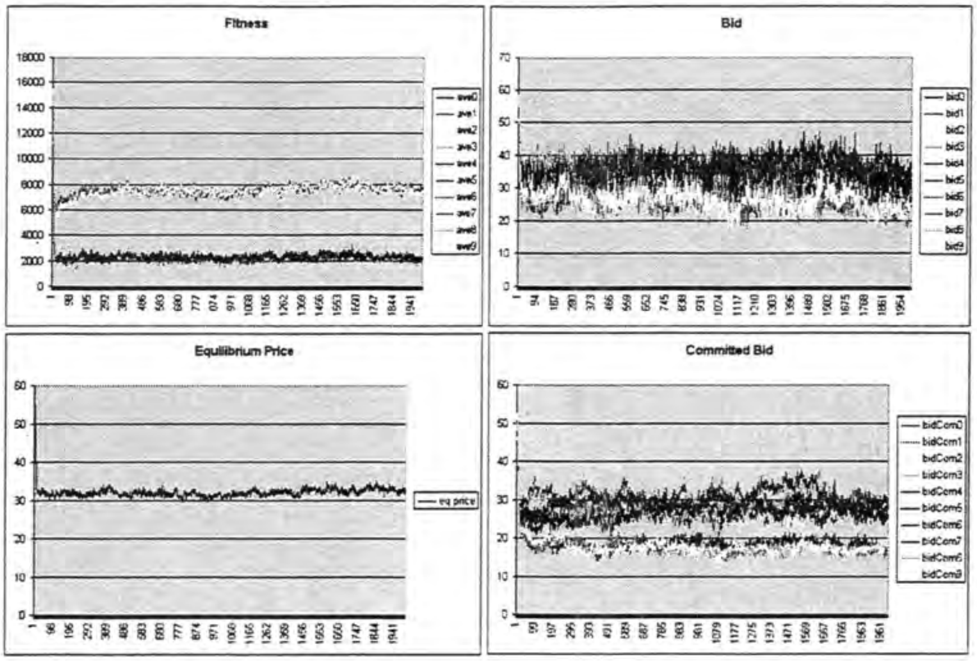

Figure 4.28 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

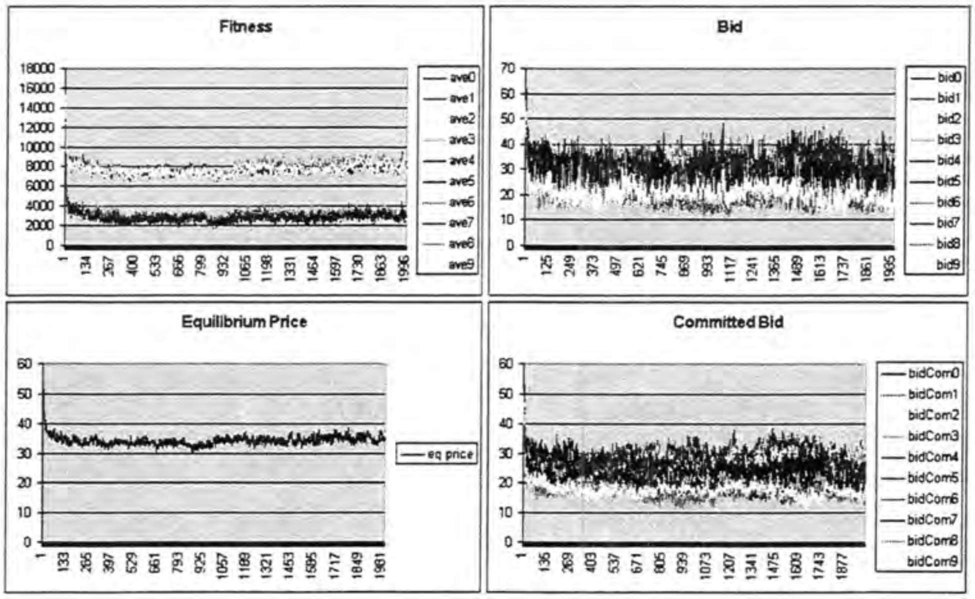

Figure 4.29 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

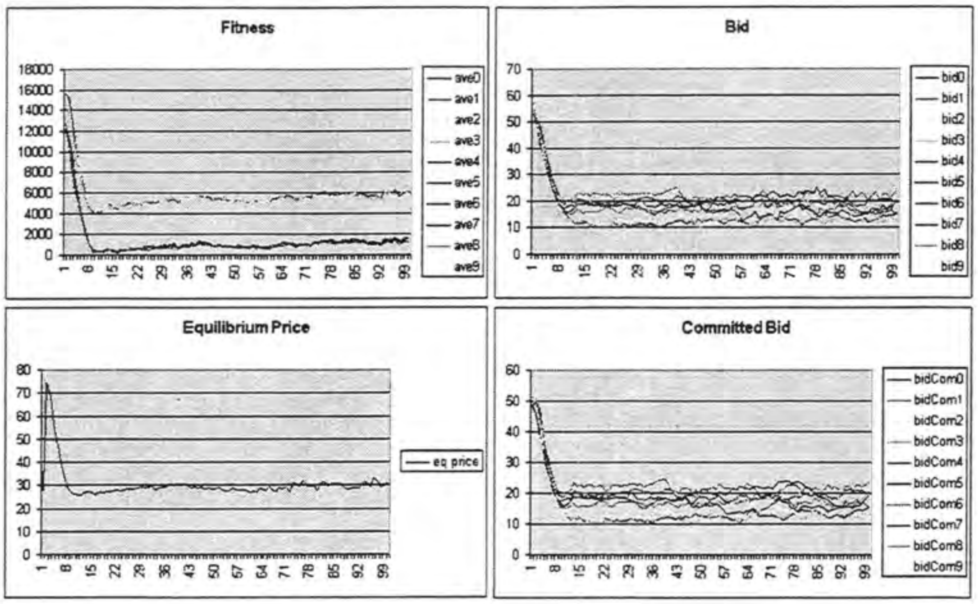

Figure 4.30 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

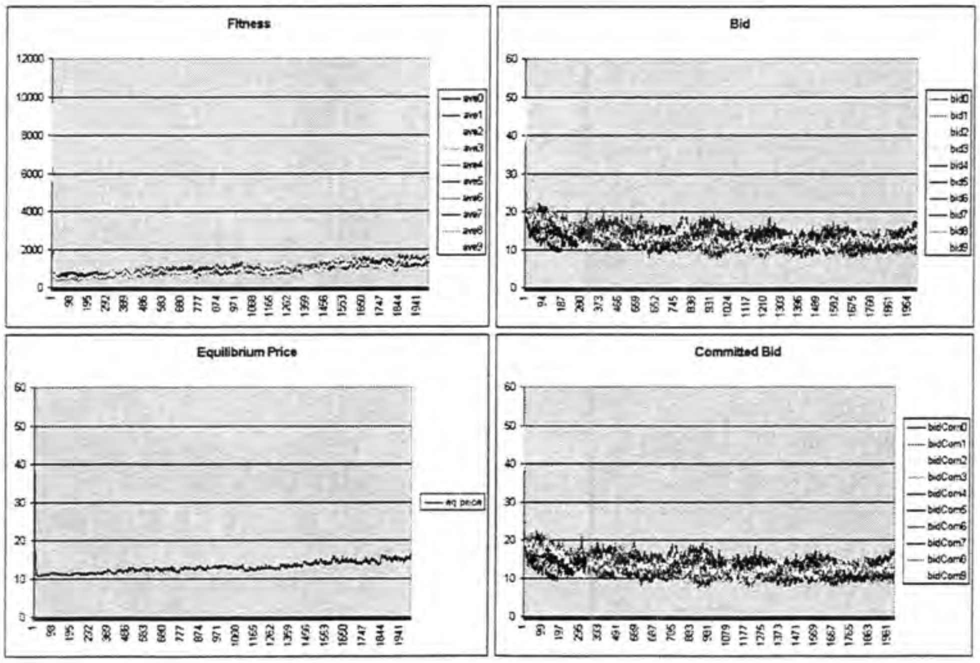

Figure 4.31 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

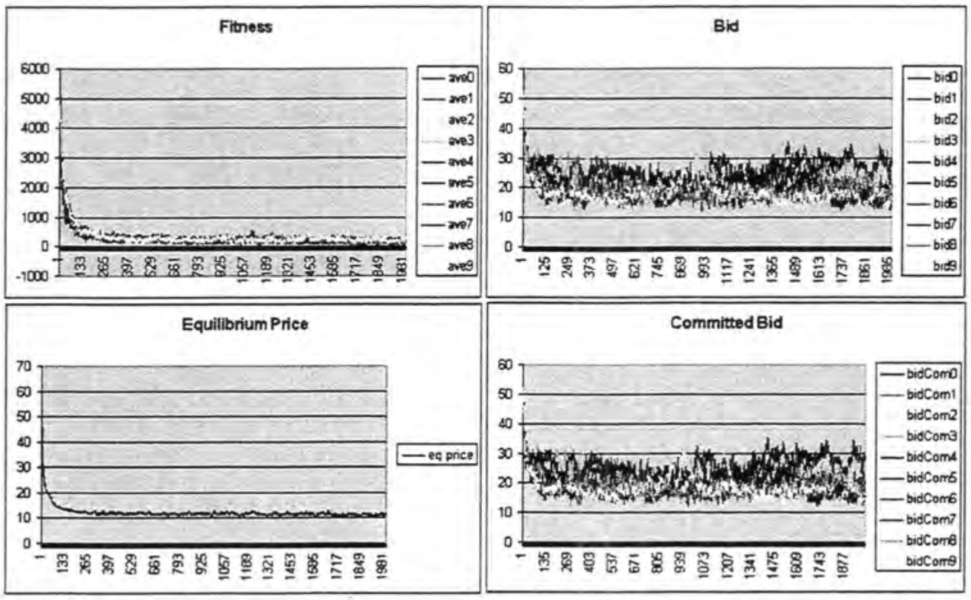

Figure 4.32 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

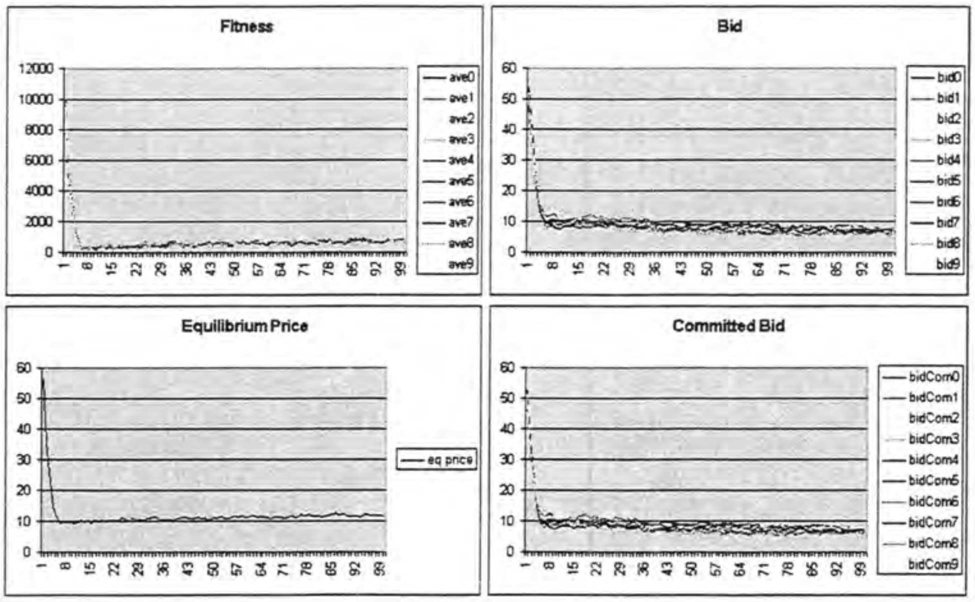

Figure 4.33 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

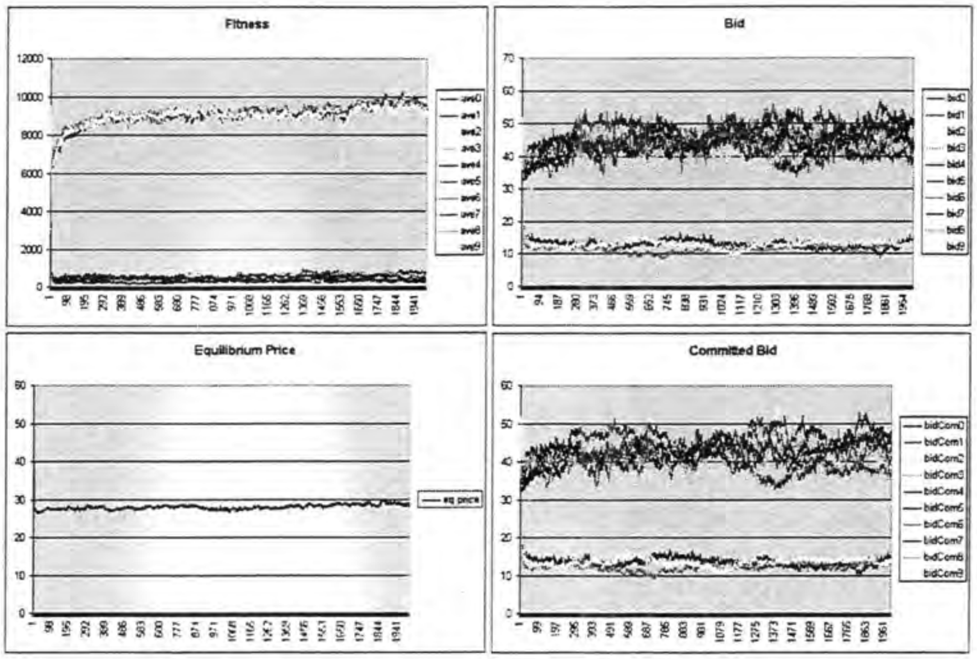

Figure 4.34 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 10 Bidders. Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500. Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

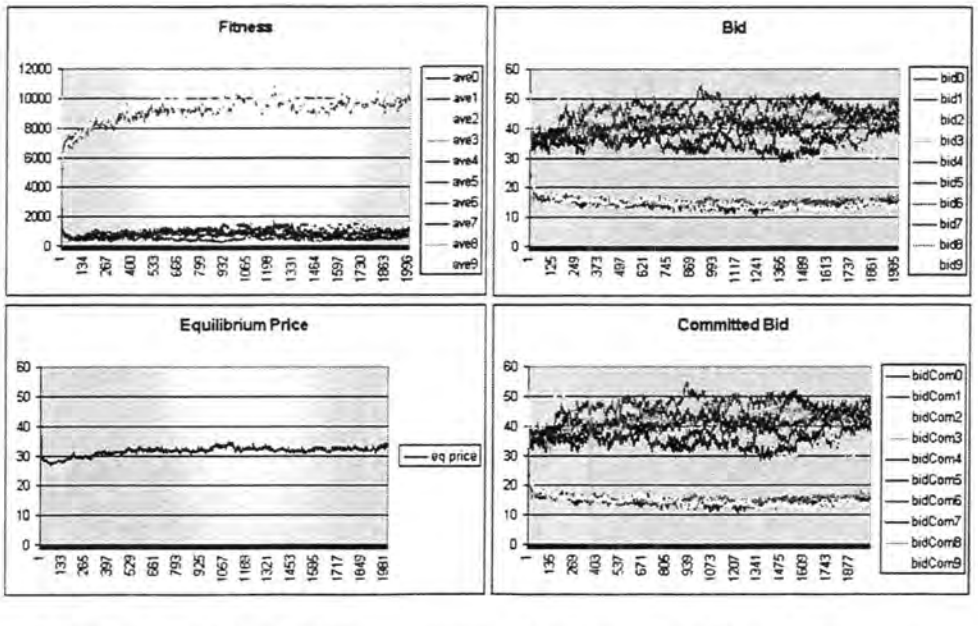

Figure 4.35 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

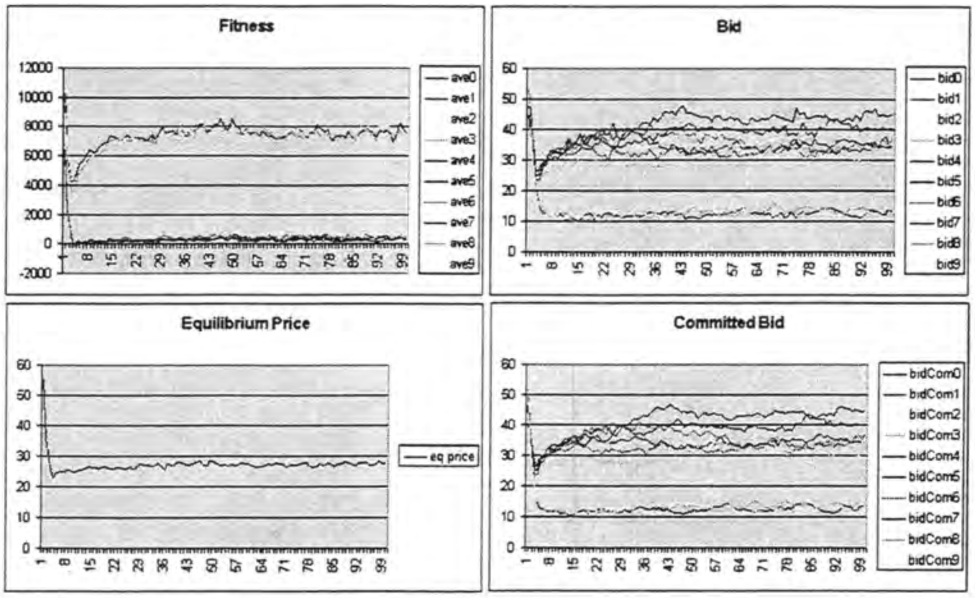

Figure 4.36 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

4.2.2.2 Auction Size 4

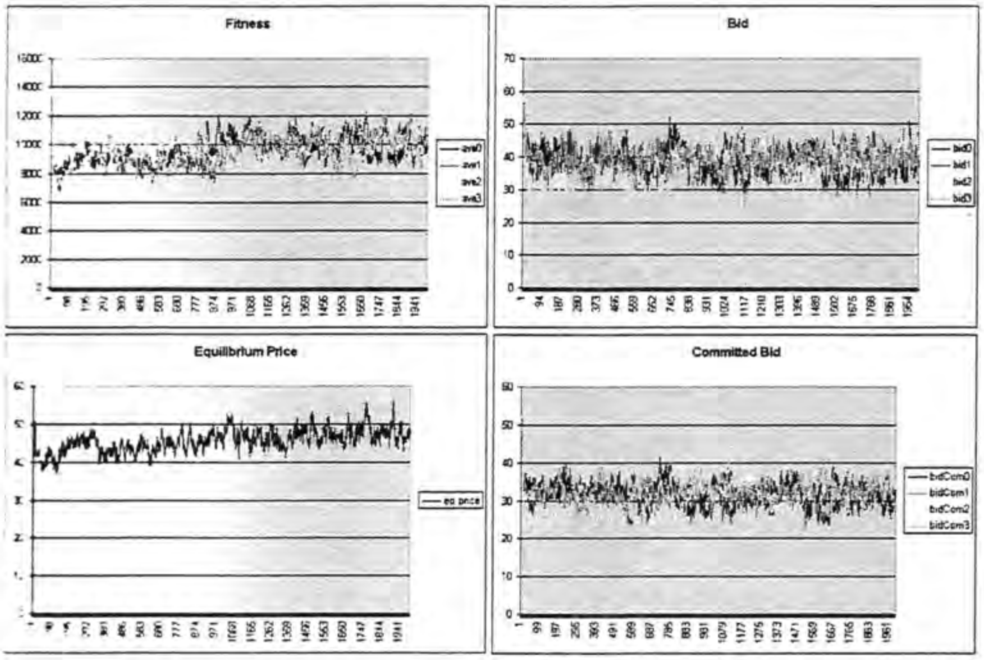

Figure 4.37 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.37 through 4.48 show four different cost curve and capacity limit choices for all three representations using the co-evolutionary fitness function and uniform pricing, with an auction size of 4 bidders.

The same effects that occurred moving from 10 to 4 bidders under the discriminatory pricing model (higher equilibrium price and bids) occur in the uniform pricing model. In some cases these effects are more pronounced.

Curiously, comparing these results with those obtained under the same conditions under the discriminatory pricing scheme, we see that the uniform pricing scheme led to the same or higher equilibrium prices and bids than the discriminatory pricing. So it seems that uniform pricing with 10 bidders lowered the price, but with only 4 bidders, the price-raising effect of market power is much more pronounced in uniform pricing than in discriminatory pricing. The price-lowering effect of uniform pricing is overcome by this compound effect.

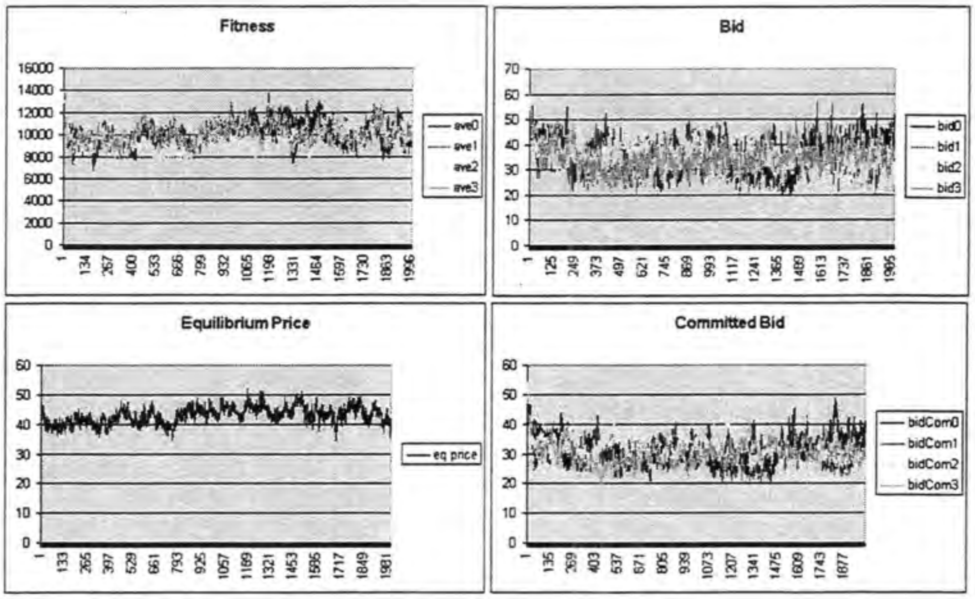

Figure 4.38 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

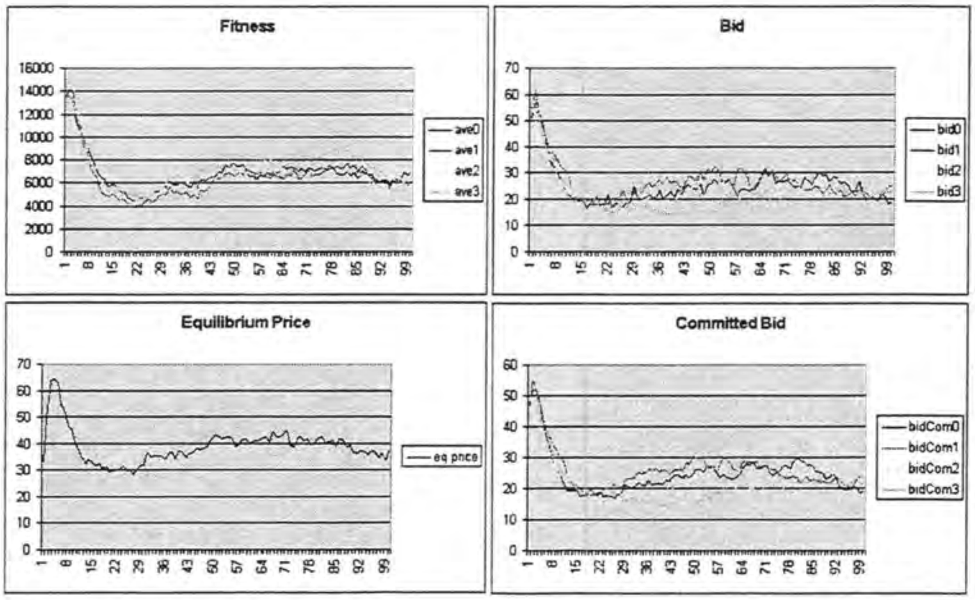

Figure 4.39 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

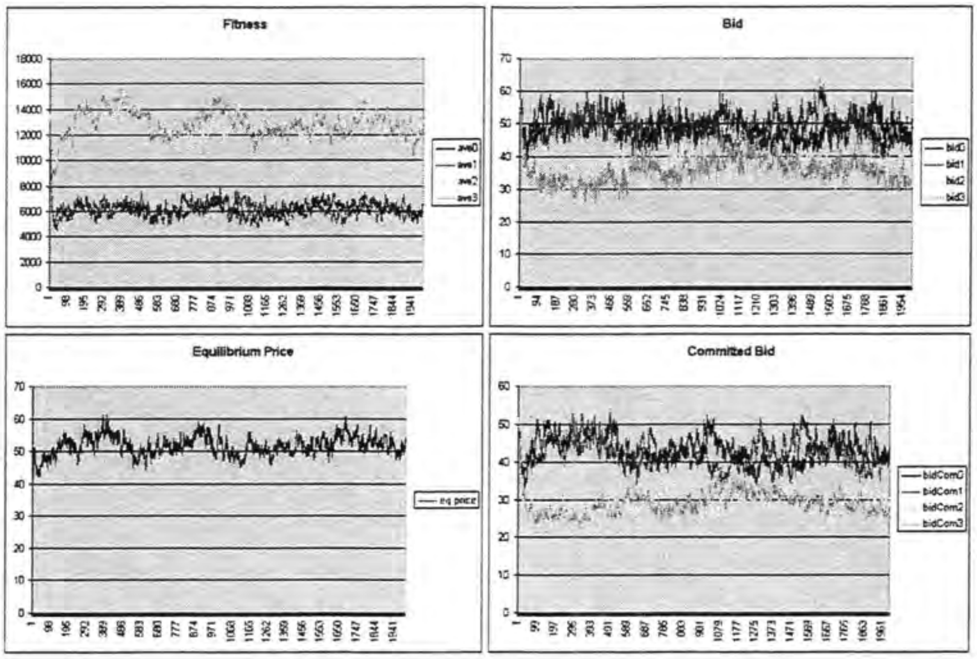

Figure 4.40 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

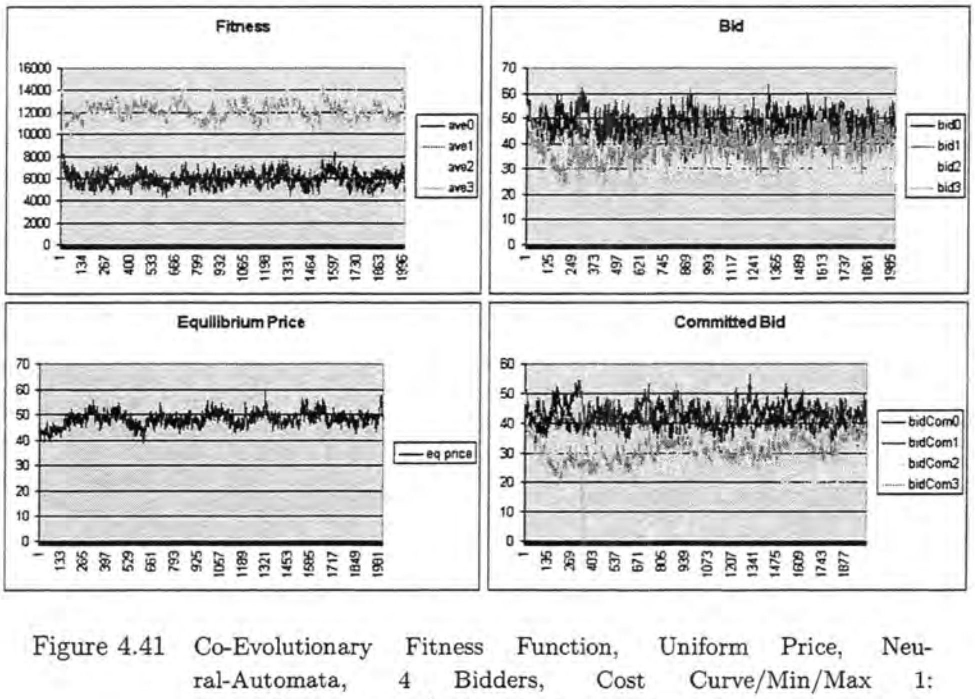

Figure 4.41 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

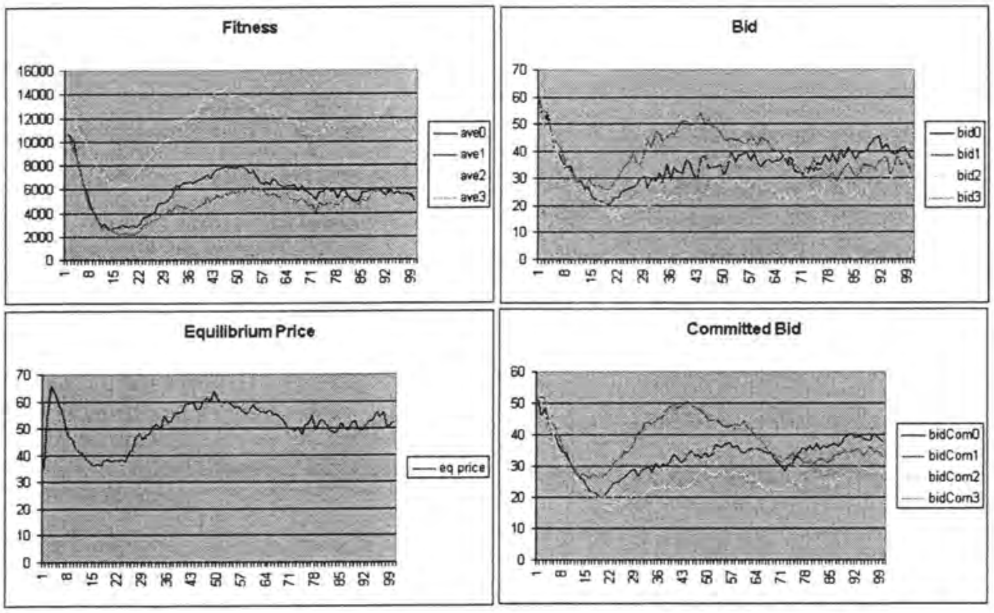

Figure 4.42 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

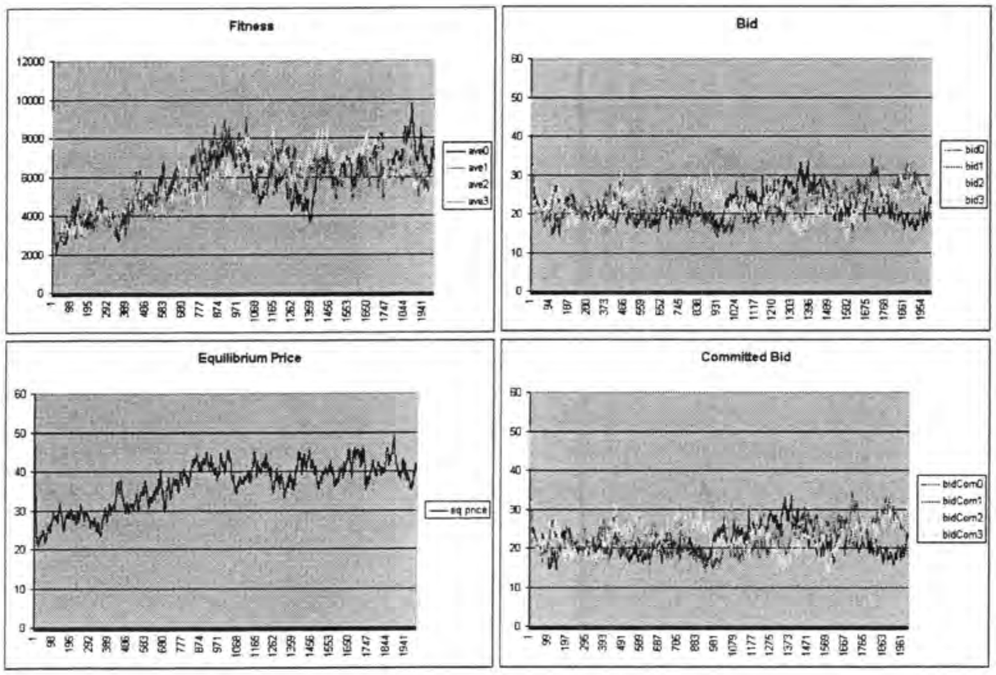

Figure 4.43 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

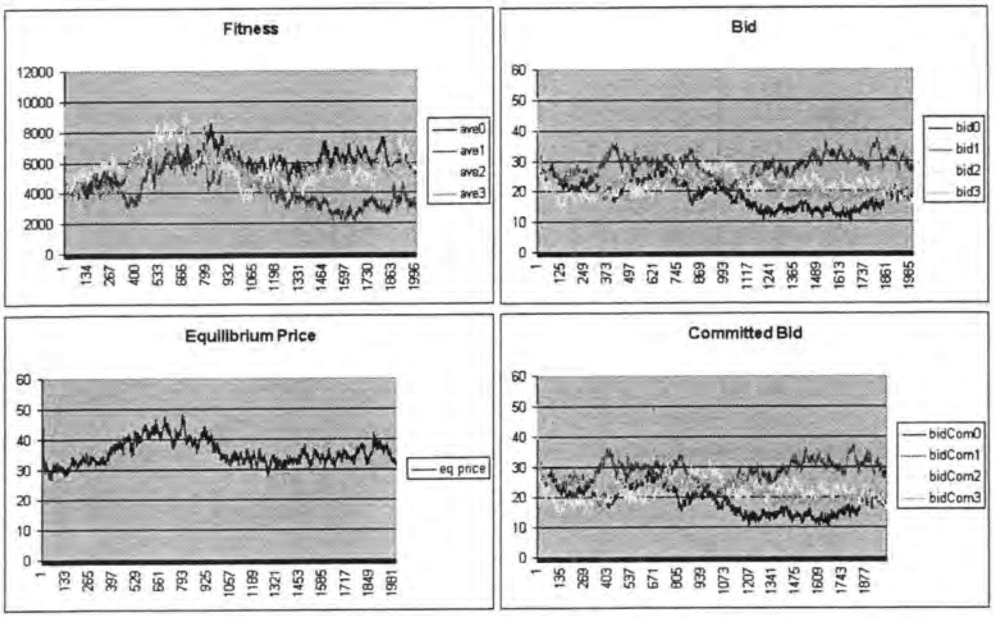

Figure 4.44 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

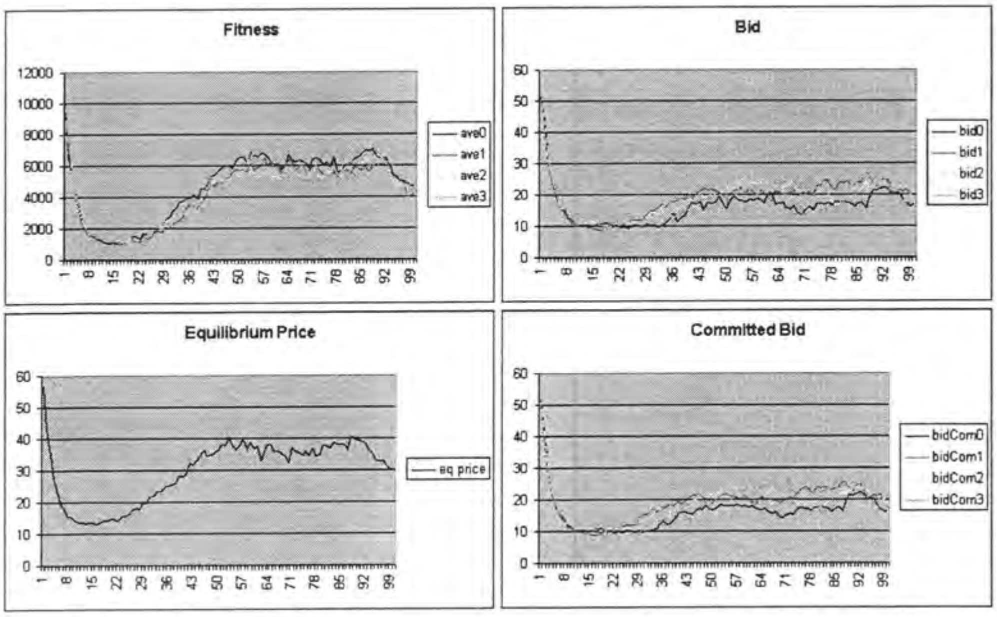

Figure 4.45 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 200 / 500

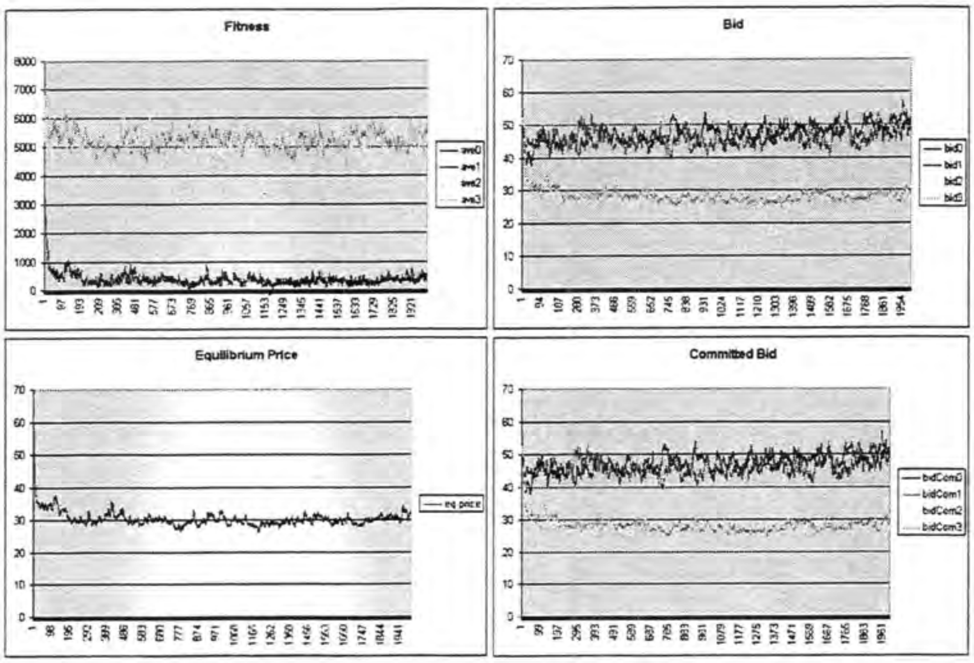

Figure 4.46 Co-Evolutionary Fitness Function, Uniform Price, GP-Automata, 4 Bidders. Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500. Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

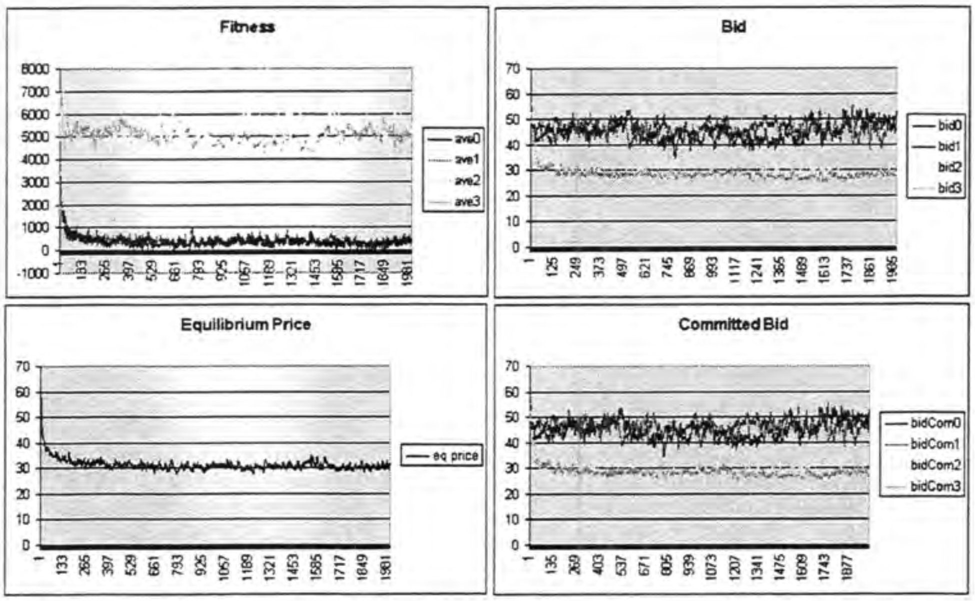

Figure 4.47 Co-Evolutionary Fitness Function, Uniform Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

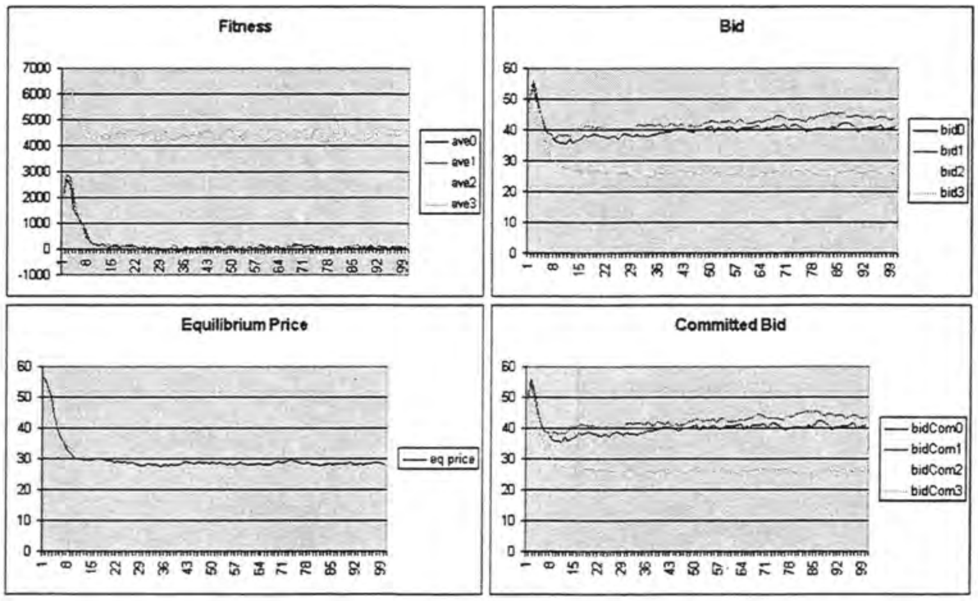

Figure 4.48 Co-Evolutionary Fitness Function, Uniform Price, Fixed-Bid, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 200 / 500, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 100 / 300

4.3 Evolution by Periodic Immortalization

The experiments described in this section developed the bidders through co-evolution for 1000 generations. At that point, the best half of the population was immortalized, and the fitness function was thereafter measured by performance against the immortals. A set of the last 5 immortalized populations was maintained, and the fitness was determined by playing against all of these populations. Before 5 populations had been immortalized, the bidders simply played all the populations that had thus far been immortalized. Each of the following experiments continued evolution for 10000 generations, immortalizing the best half of the population every 1000 generations.

These experiments used only the complex representations, GP-Automata and Neural-Automata. The fixed-bid representation was not evolved.

4.3.1 Experiment 3: Immortalized Population Fitness Function, Discriminatory Price

Table 4.3 Variations on Experiment 3

Figures	Bidders	Demand	Cost Curve / Min / Max 1	Cost Curve / Min / Max 2
4.57, 4.58	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.59, 4.60	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.61, 4.62	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.63, 4.62	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500

This experiment used discriminatory pricing. Recall that with discriminatory pricing, each bidder, if it has a bid accepted, gets paid the price it listed in the bid. The rest of the details are identical to those described in section 4.2.1.

4.3.1.1 Auction Size 10

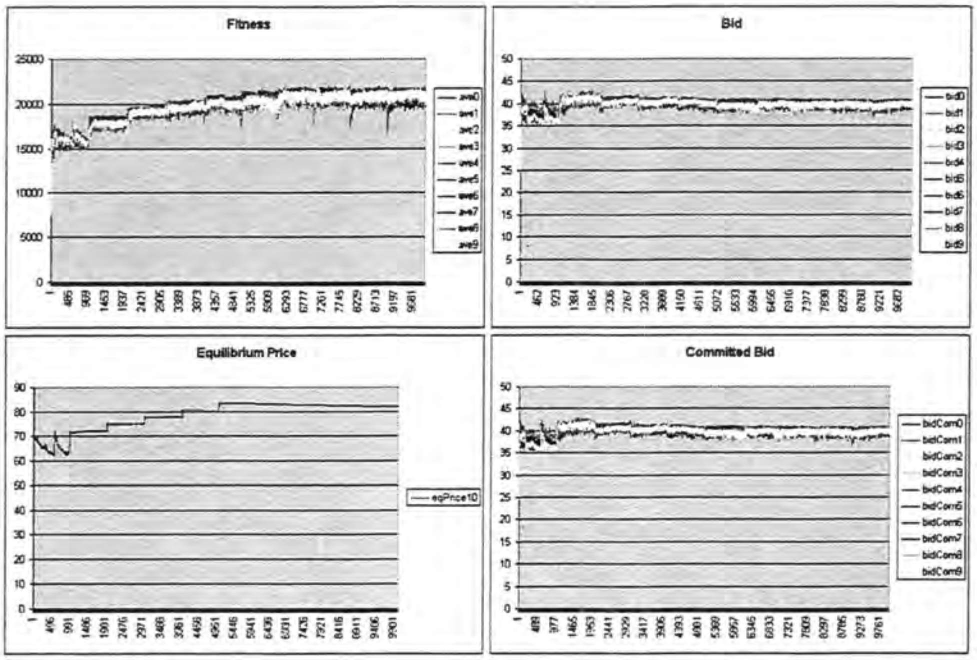

Figure 4.49 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.49 through 4.52 show two different cost curve and capacity limit choices for the two finite-state automata representations using the immortalizing fitness function and discriminatory pricing, with an auction size of 10 bidders.

The difference between these graphs, as in the co-evolutionary case, stems from the difference between the cost curves and capacity limits of the bidders. In each case, the population gradually improved in fitness (and the equilibrium price increased along with it) until about the 5000th generation, after which it levelled off. At this point, the equilibrium price actually began decreasing slightly.

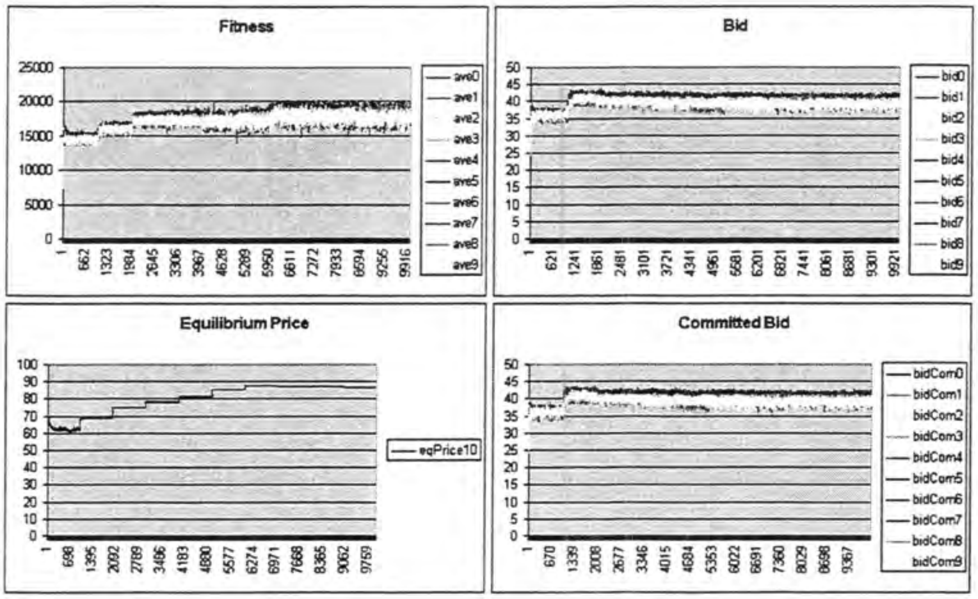

Figure 4.50 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

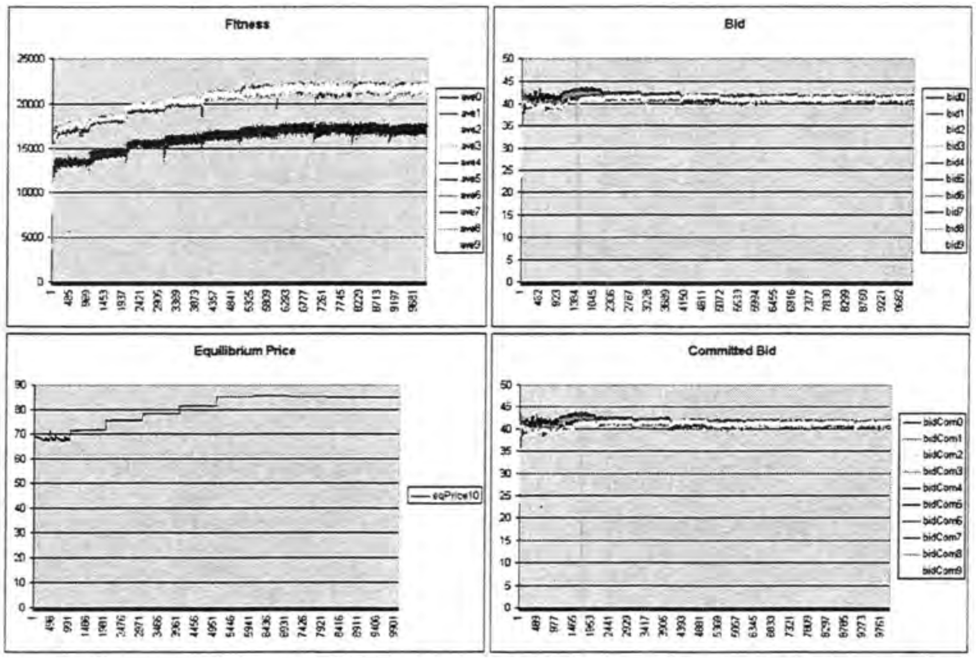

Figure 4.51 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

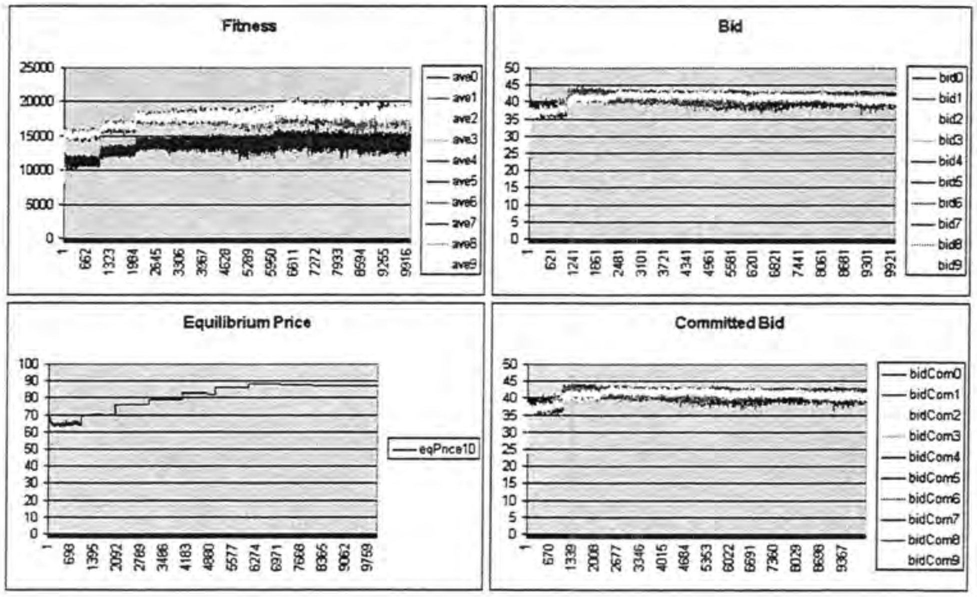

Figure 4.52 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

4.3.1.2 Auction Size 4

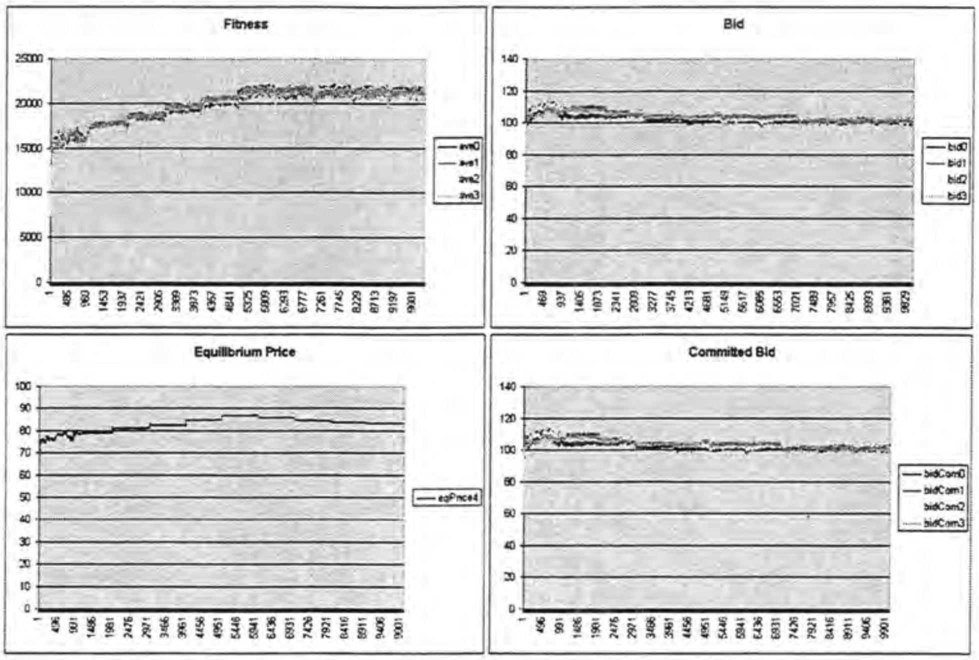

Figure 4.53 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.53 through 4.56 show two different cost curve and capacity limit choices for the two finite-state automata representations using the immortalizing fitness function and discriminatory pricing, with an auction size of 4 bidders.

It is interesting to note that the effect of number of bidders on average equilibrium price (less bidders $\rightarrow$ higher price) is not nearly as pronounced in the immortalizing fitness function case as in the co-evolutionary case.

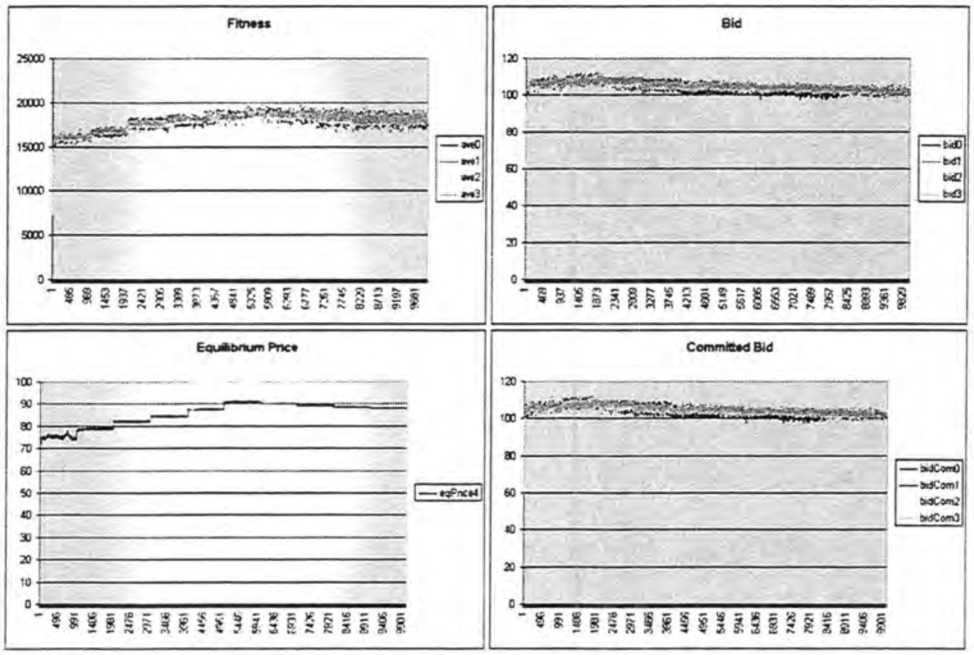

Figure 4.54 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

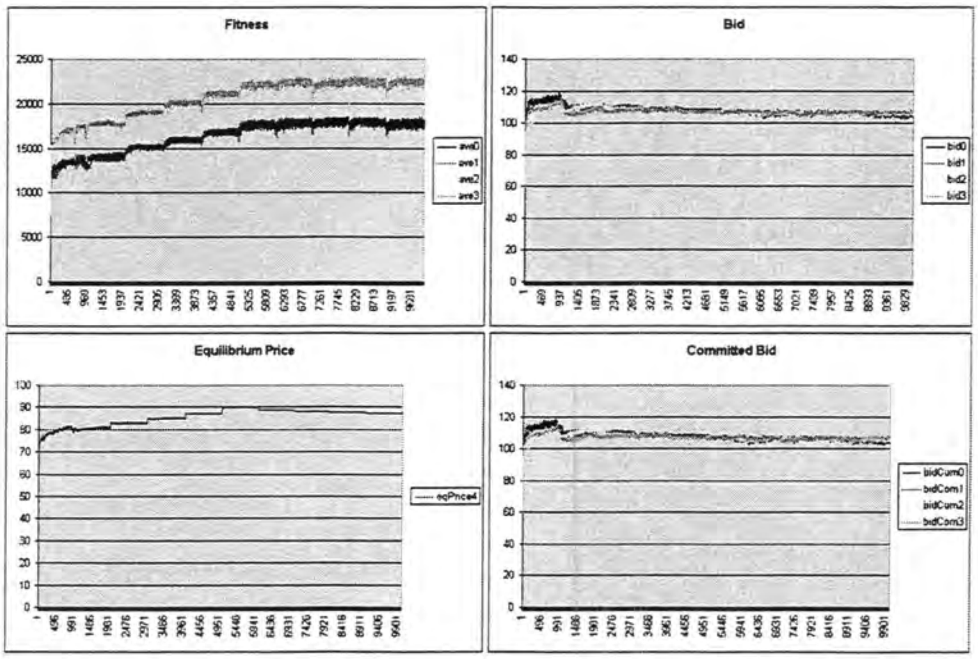

Figure 4.55 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

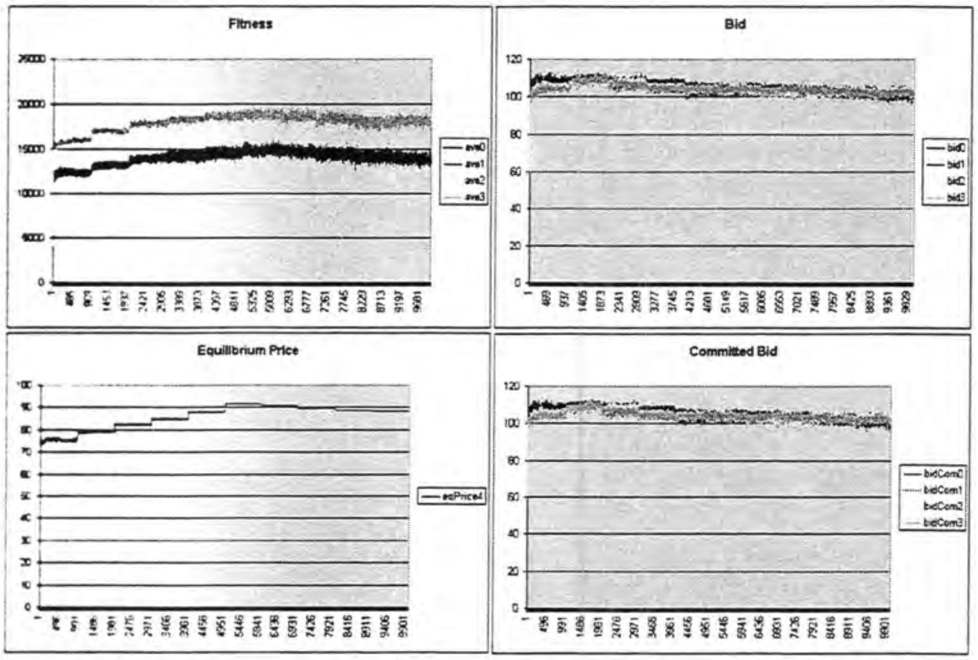

Figure 4.56 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

4.3.2 Experiment 4: Immortalized Population Fitness Function, Uniform Price

Table 4.4 Variations on Experiment 4

Figures	Bidders	Demand	Cost Curve / Min / Max 1	Cost Curve / Min / Max 2
4.57, 4.58	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.59, 4.60	10	2000	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500
4.61, 4.62	4	800	$(0.004q^2 +.5.3q + 500)$ / 100 / 300	$(0.004q^2 + 5.3q + 500)$ / 100 / 300
4.63, 4.62	4	800	$(0.004q^2 + 5.3q + 500)$ / 100 / 300	$(0.012q^2 + 15q + 1500)$ / 200 / 500

This experiment used uniform pricing. Recall that with uniform pricing, each bidder, if it has a bid accepted, gets paid the equilibrium price, or the price of the highest accepted bid. The rest of the details are identical to those described in section 4.2.1.

4.3.2.1 Auction Size 10

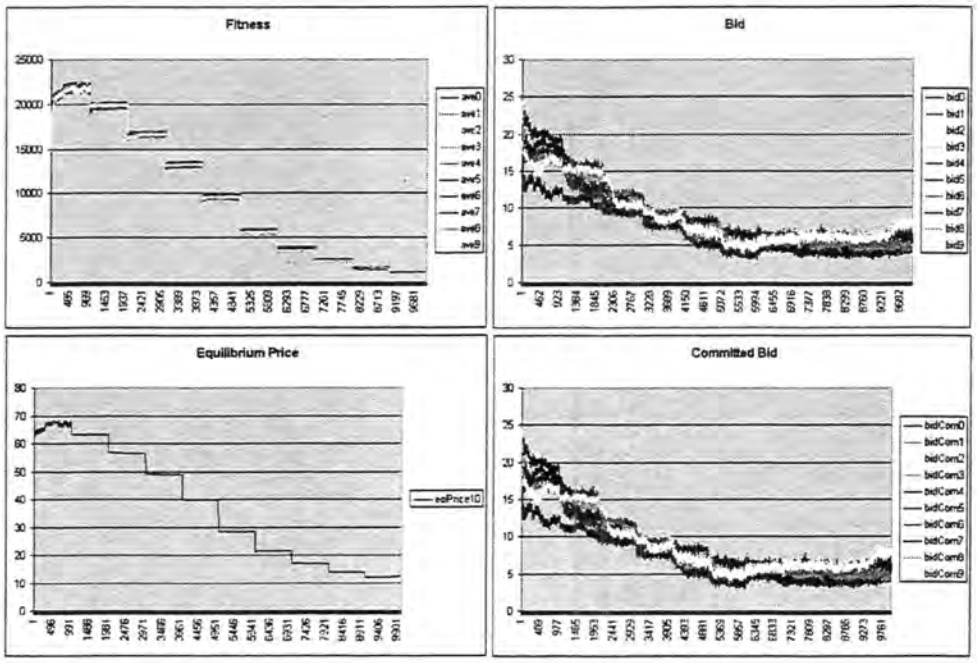

Figure 4.57 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.57 through 4.60 show two different cost curve and capacity limit choices for the two finite-state automata representations using the immortalizing fitness function and uniform pricing, with an auction size of 10 bidders.

These results differ remarkably from those of the discriminatory case. Whereas the discriminatory pricing model led to gradual fitness and price increase, uniform pricing led to decrease in both fitness and price. Note that the fitness increases for the first 1000 generations, when the fitness function is co-evolutionary, and afterwards it decreases sharply at each replacement of the immortal bidders.

Of course, the effect of differences in cost curves and capacity limits is still in place: those with higher costs bid higher and got lower fitness.

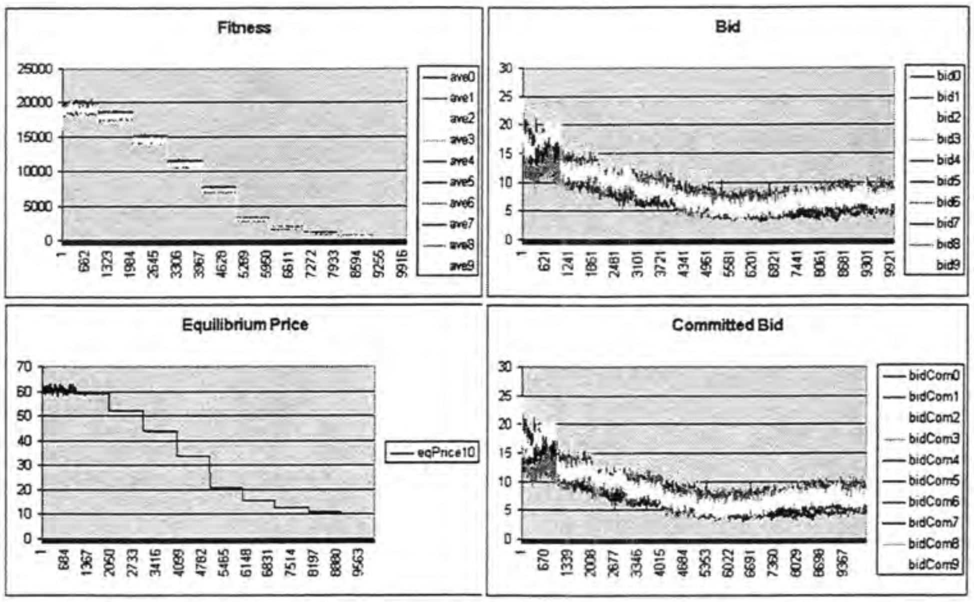

Figure 4.58 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

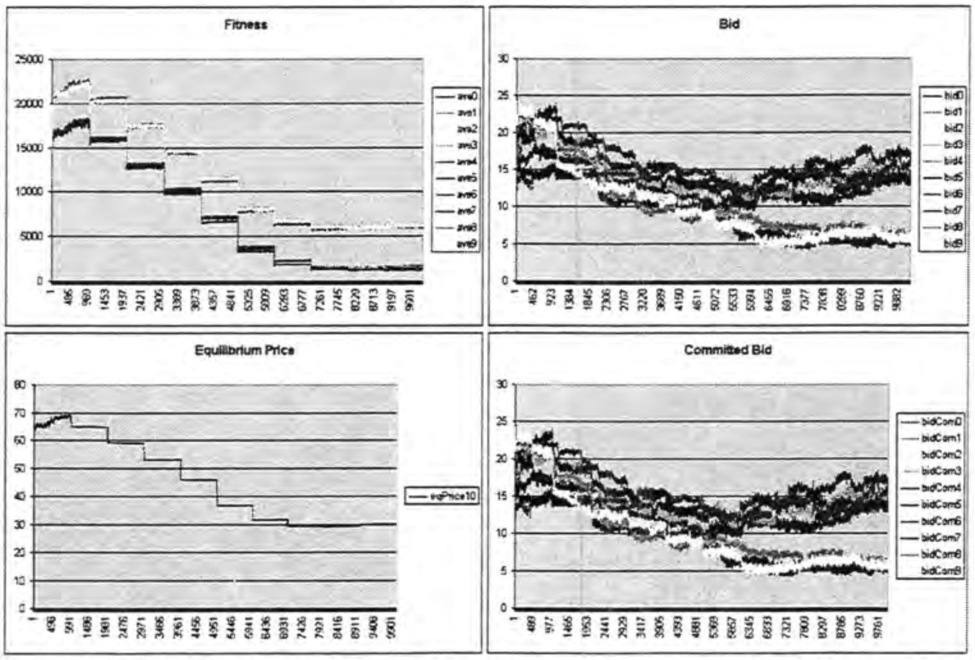

Figure 4.59 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

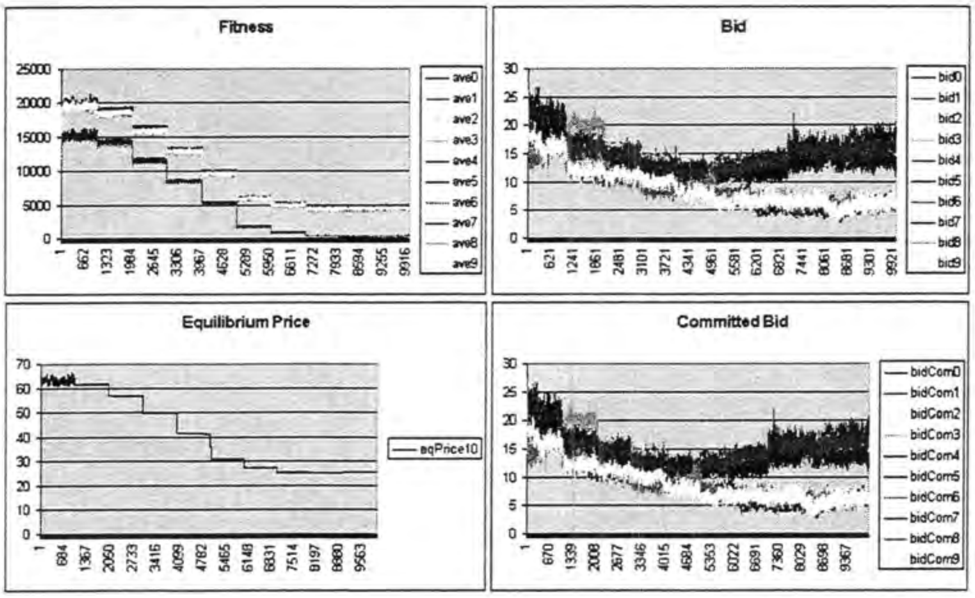

Figure 4.60 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 10 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

4.3.2.2 Auction Size 4

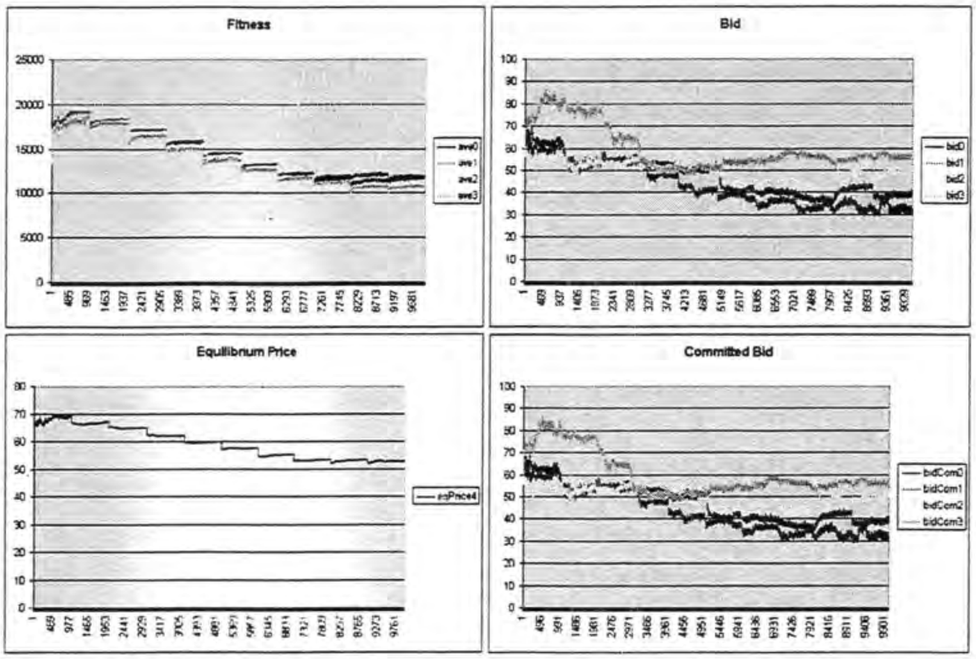

Figure 4.61 Immortalized Population Fitness Function, Discriminatory Price. GP-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

Figures 4.61 through 4.64 show two different cost curve and capacity limit choices for the two finite-state automata representations using the immortalizing fitness function and uniform pricing, with an auction size of 4 bidders.

Comparing these results to those in the case with 10 bidders, we see the usual effect: less bidders means higher average equilibrium price and higher fitness.

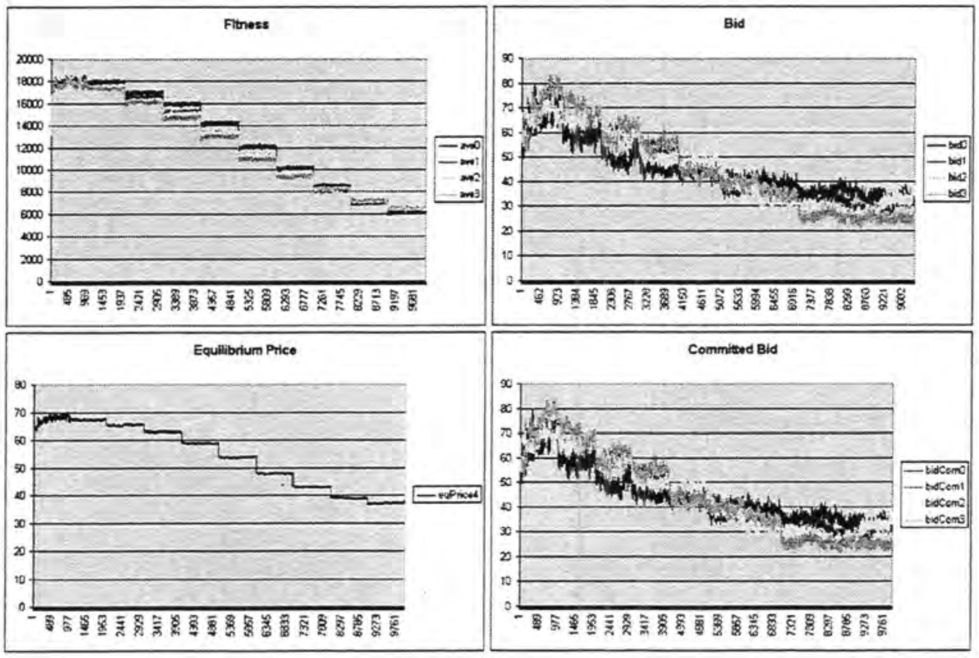

Figure 4.62 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max: $(0.004q^2 + 5.3q + 500)$ / 100 / 300

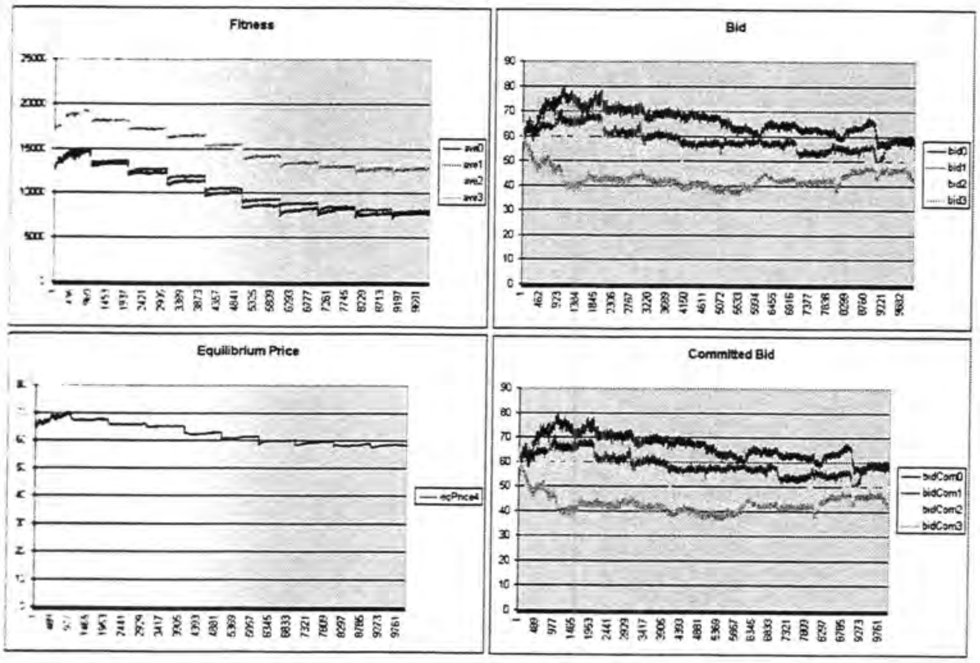

Figure 4.63 Immortalized Population Fitness Function, Discriminatory Price, GP-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

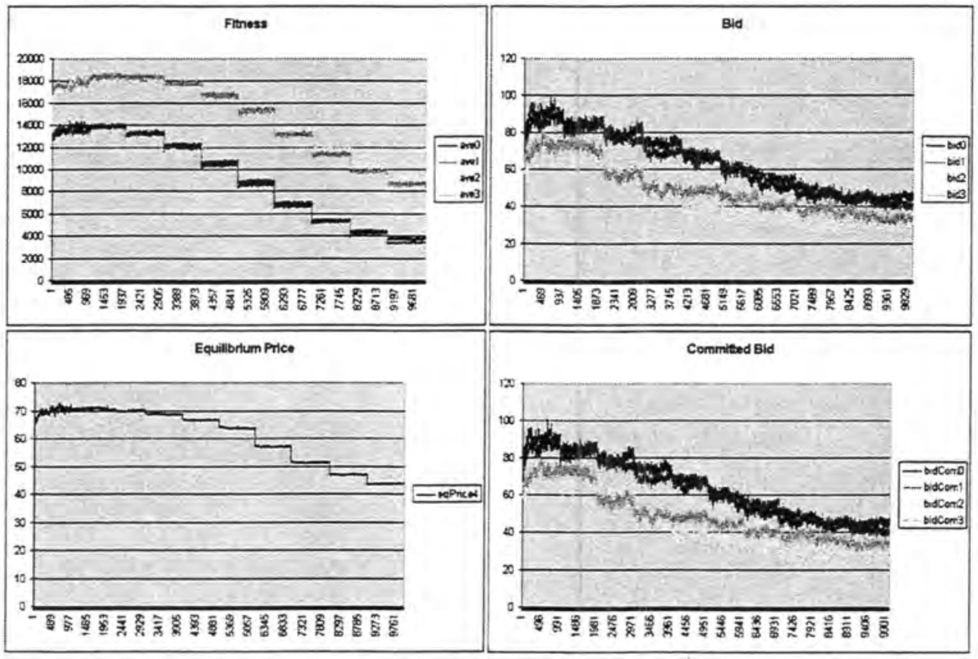

Figure 4.64 Immortalized Population Fitness Function, Discriminatory Price, Neural-Automata, 4 Bidders, Cost Curve/Min/Max 1: $(0.004q^2 + 5.3q + 500)$ / 100 / 300, Cost Curve/Min/Max 2: $(0.012q^2 + 15q + 1500)$ / 200 / 500

CHAPTER 5. SUMMARY AND DISCUSSION

This research explored the adaptability of finite state automata to a simulated electric power market. The finite state automata were developed using a genetic algorithm. Two different types of finite state automata, GP-Automata and Neural-Automata, were tried, and their performance was compared to each other and to a third simplified representation that served as a baseline of comparison.

5.1 Conclusions

Recall that the difference between these two representations lies in the data processing structure used to compress the bandwidth of the input to the finite state machine. GP-Automata used parse trees, and Neural-Automata used neural nets. The first conclusion to draw from the data in the previous chapter is that GP-Automata and Neural-Automata performed nearly the same under all conditions. From this one can conclude that both a parse tree and a neural net are equally capable of processing the data received during rounds of bidding in an auction.

The second conclusion to draw is that the finite state automata behaved, within reasonable limits, in ways similar to the baseline representation. In those cases in which the market outcome was dissimilar, the finite state automata achieved higher fitness than the baseline. Since the average fitness was higher for all populations, each of which represented one bidder in the auction, this indicates cooperation taking place between bidders in the auctions.

Finally, though the fitness often changed during the course of evolution, this does not indicate exactly what is happening. With co-evolution especially, the average fitness going down does not necessarily mean the population is getting worse, and the average fitness going up does not necessarily mean the population is getting better. It only indicates performance of the agents relative to each other for the co-evolutionary case and relative to the immortal agents in the immortality fitness function case.

5.2 Future Work

The markets rules were not chosen arbitrarily, but neither were they unique. A single-sided auction was studied; future work could study a double-sided auction.

This research assumed that the cost of producing electricity by a multi-unit GENCO could be simplified as a single quadratic cost curve. This could be modified in the future to perform an explicit economic dispatch optimization once the quantity to be delivered is settled in the auction.

This research considered only the day-ahead market used to determine allocation for the next day. It ignored the existence of forward markets on larger timescales (weekly, monthly, etc.) and the hourly spot market held the hour before delivery. Future research could explore these types of markets or develop strategies that make decisions for all of these timescales.

The effectiveness of evolving finite state automata as learning agents was measured only relative to other evolved finite state automata. Future research could compare strategies developed through other learning mechanisms to these agents. One could also evolve the finite state automata in parallel with another learning algorithm, in order to determine which is more adaptable.

In early runs of the experiments, the author conducted informal tests for the *Mayfield Effect*. Recall that this is an effect observed by saving population members after the average fitness has converged, evolving the population for a long time, and playing the new population against the saved population. The Mayfield Effect occurs if the new population is able to achieve a higher fitness against the old population, despite the fact that they receive the same fitness playing within their own population. The effect is that of building general adaptive skills during evolution. The author's informal tests for this effect on the finite state machines being evolved to play electric power markets revealed that it was not present. Future work in this area would involve discovery of a representation or a method of evolving the representations used that is able to build up general adaptive features through evolution.

The Neural-Automata representation has not been tried for any other genetic algorithm problems. Future research into this representation could test this representation on other problems that accommodate the use of finite state machines. Obvious test problems are those that have already been tried with GP-automata, such as simple economic games like Divide the Dollar (Ashlock97) or as control structures in simulated robots (Ashlock01).

The parse trees used in the GP-Automata took real numbers as input and hence, used real numbers as intermediate values during evaluation of the tree. Since limits were not placed on the possible connections between parse tree nodes, this means that a real number could potentially be fed into a boolean input. Since booleans are defined in terms of integers by the convention (0 = false, everything else = true), this could lead to every real being fed as a boolean argument being evaluated as true, since the odds of getting a real exactly equal to 0 are infinitesimal. However, this does not preclude other data types, such as booleans and integers, still being fed as boolean inputs, and so it will not prevent the boolean functions from operating normally. Randomly generated trees will simply have a bias toward true values being fed to boolean inputs. One potential way to correct this would be to implement a grammar to build the parse trees in which feeding real numbers as boolean inputs would be disallowed. This work assumed that evolution would simply weed out those parse trees that use reals as booleans if they cause problems.

A potential problem with using the output of a neural net as an array index is that there may be a bias toward the high and low numbered states. Since the sigmoid function used stays close to its upper-bounded or lower-bounded values for extreme values of the input, there is the possibility of the states 0 and (n-1) appearing more often. Evolution must simply find a way around this bias or use it. One potential way to eliminate this bias is to use a linear output node instead of a sigmoid. This, however, requires a way of limiting the output, since a linear transfer function is not bounded. In this work the sigmoid transfer function was kept on the output node on the assumption that any biases introduced would be either used or weeded out by evolution. Future work wold find a way to eliminate this bias altogether.

APPENDIX Glossary

auction A process, constrained by a set of rules, by which economic goods may be allocated and price paid for these goods determined.

bandwidth compression Reduction of the resources needed to represent some data.

co-evolutionary fitness function A fitness function that is dependent on members of the population besides the member being evaluated.

crossover The exchange of subsets of representations between two population members of a genetic algorithm.

discriminatory pricing Giving different winners in an auction different prices paid for goods. Contrast to *uniform pricing.*

equilibrium price The intersection of the demand and supply curves on an economic price vs quantity graph. This is the theoretically predicted market price of a good.

evolutionary algorithm An optimizing algorithm the mimics the biological process of Darwinian evolution find an optimal solution.

evolutionarily stable A strategy that, given that it is in use across an entire co-evolving population, cannot be invaded by another strategy.

finite state machine A theoretical data processing structure. It takes input from some external source, changes its internal state in response to the input, and may or may not produce an output. The state change and output response are determined by a lookup table that enumerates all possible inputs. Also called a *finite state automaton.*

fitness A problem-specific numerical measure of the optimality of a solution being developed by a genetic algorithm.

fixed fitness function A fitness function that does not depend on other members of a population besides the member being evaluated.

genetic algorithm An evolutionary algorithm that uses crossover.

genetic programming A genetic algorithm that evolves parse trees.

GP-automaton A modification of a classical finite state machine in which state change and output made in response to an input are determined by a parse tree having the input as leaf nodes.

immortalizing fitness function A fitness function that relies on the genetic algorithm "immortalizing" (saving) members of a population after some number of generations of evolution, and changing the fitness function to be dependent on competing with these fixed members.

invade To overtake and replace another strategy in a co-evolving system.

market clearing The process of translating the bids collected in a market auction into an allocation of the goods from sellers to buyers.

Mayfield effect The effect observed when a population evolves for a very long time with no apparent change in fitness, but is able to build up general skills in achieving high fitness. The effect may be demonstrated by picking two populations from generations after the fitness converged and playing them against each other. If the effect is present, the more evolved population should win.

mutation The random perturbation of a population member in an evolutionary algorithm.

Nash equilibrium A set of strategies assigned to all players in an economic game having the property that if any player unilaterally changes its strategy, it will lower its payoff.

neural-automata A modification of a classical finite state machine in which state change and output made in response to an input are determined by a neural net having the input as nodes in the input layer.

neural net A set of connected neurons, each of takes multiple inputs and produces one output. The neural net as a whole takes multiple inputs and produces a single output. Also called *artificial neural net.*

optimal power flow Given a set of electricity-generating units, the quantity of electricity that each must produce to minimize the total cost of production across all generators.

parse tree A tree data structure used to store mathematical formulas.

population-based stochastic search algorithm A stochastic search algorithm in which a set (population) of potential solutions is developed, with some rule determine how many solutions to perturb at each iteration, and some rule determining which solutions to keep and which to discard. An evolutionary algorithm with no crossover is a type of population-based stochastic search algorithm.

single tournament selection A model of evolution used in genetic algorithms in which the population is split into groups of four, and the best two members of the group mate, and their children replace the other two members.

stochastic search algorithm An optimizing algorithm that begins with a random potential solution to a problem, randomly perturbs it slightly, and replaces the original with the perturbed version if the perturbed version is more optimal.

uniform pricing Giving different winners in an auction a single price paid for goods. Contrast to *discriminatory pricing.*

BIBLIOGRAPHY

[Ashlock97] D Ashlock (1997). "GP-Automata for Dividing the Dollar", Genetic Programming 1997: Proceedings of the Second Annual Conference, July 13-16, 1997, pg. 18-26

[Ashlock00] D Ashlock, J Freeman. (2000). "A Pure Finite State Baseline for Tartarus", Proceedings of the 2000 Congress on Evolutionary Computation, 2000.

[Ashlock01] D Ashlock (2001). "Math 378: Artificial Life, Class Notes", Department of Mathematics, Iowa State University, 2001.

[Axelrod87] R Axelrod. (1987). "The Evolution of Strategies in the Iterated Prisoner's Dilemma", Genetic Algorithms and Simulated Annealing, chapter 3, pages 32 41. Morgan Kaufmann, Los Altos, Calif., 1987.

[Boyd87] R Boyd, J Lorberbaum, "No Pure Strategy is Evolutionarily Stable in the repeated Prisoner's Dilemma Game" Nature, 327 (7 May 1987): 58-59

[Campbell87] N A Campbell. (1987). Biology, Third Edition. pg. 1108. The Benjamin/Cummings Publishing Company, Inc., 1987.

[Contreras01] J Contreras, O Candiles, J de la Fuente, T Gomez (2001). "Auction Design in Day-Ahead Electricity Markets", IEEE Transactions on Power Systems, Vol 16, No 1, February 2001.

[Copeland01] T Copeland, V Antikarov (2001). Real Options: A Practitioner's Guide, New York: Texere, LLC.

[Cybenko88] G Cybenko. (1988). "Continuous valued neural networks with two hidden layers are sufficient (Technical Report)". Department of Computer Science, Tufts University, Medford, MA.

[ErevRoth95] A E Roth, I Erev. (1995). "Learning in Extensive-Form Games: Experimental Data and Simple Dynamic Models in the Intermediate Term", Games and Economic Behavior, SPecial Issue: Nobel Symposium, vol. 8, January 1995, 164-212

[Koza92] J Koza. (1992). Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: The MIT Press.

[Kumar96] J Kumar, G Sheblé (1996). "Auction Market Simulation for Price-Based Operation", Department of Electrical and Computer Engineering, Iowa State Univeristy, 1996.

[Leahy00] N Leahy, D Ashlock (2000). "Representational Sensitivity in A Simple Agent-Based Computational Economics Experiment", IEEE Transactions on Evolutionary Computation.

[Mayfield98] J Mayfield, D Ashlock. (1998). "Acquisition of General Adaptive Features by Evolution", Lecture Notes in Computer Science, v1447, 1998, p 75.

[McAfee87] R McAfee, J McMillan. (1987). Auctions and bidding. Journal of Economic Literature, 25(2):699738.

[Nash50] J Nash (1950). "Equilibrium points in N-Person Games", 1950, Proceedings of the National Academy of Science.

[Otero-Novas00] I Otero-Novas, C Meseguer, C batlle, J Alba (2000). "A Simulation Model for a Competitive Generation Market", IEEE Transactions on Power Systems, Vol 15, No 1, February 2000.

[Petrov01] V Petrov, G Sheblé (2001). "Building Electric Power Auctions with Improved Roth-Erev Reinforced Learning", Department of Electrical and Computer Engineering, Iowa State University, 2001.

[Richter98] C Richter, G Sheblé. (1998). "Genetic Algorithm Evolution of Utility Bidding Strategies for the Competitive Marketplace". IEEE Transactions on Power Systems. Vol. 13, No. 1, February 1998.

[Richter99] C Richter, G Sheblé, Ashlock, D. (1999). "Comprehensive Bidding Strategies with Genetic Programming/Finite State Automata". IEEE Transactions on Power Systems. Vol. 14, No. 4, November 1999.

[RothErev95] I Erev and A E Roth, (1995). "On the Need for Low Rationality, Cognitive Game Theory: Reinforcement Learning in Experimental Games with Unique, Mixed-Strategy Equilibria", mimeo, July 1995, University of Pittsburgh.

[Sheblé96] G Sheblé (1996). "Price Based Operation in an Auction Market Structure", IEEE Transactions on Power Systems, Vol 11, No 4, November 1996.

[Song99] H Song, C Liu, J Lawarrée, R Dahlgren (2000). "Optimal Eelctricity Supply Bidding by Markov Decision Process", IEEE Transactions on Power Systems, Vol 15, No 2, May 2000.

[Wagner00] B Wagner, N Leahy, D Ashlock (2000). "A Representational Sensitivity Study of Game Theoretic Simulations", submitted to 2000 Congress on Evolutionary Computation, 2000.

[Weber99] J D Weber, T J Overbye (1999). "A Two-Level Optimization Problem for Analysis of Market Bidding Strategies", Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign

[Wilson97] R Wilson. (1997). "Activity Rules for the Power Exchange", Report to the California Trust for Power Industry Restructuring, March 14, 1997.

[Wood96] A Wood, B Wollenberg (1996). Power Generation Operation and Control, Wiley Interscience, 1996.

[Wu02] Q H Wu, J Guo, D R Turner, Z X Wu, X X Zhou. (2002) "Optimal Bidding Strategies in Electrcity Markets Using Reinforcement Learning", Department of Electrical Engineering and Electronics, The University of Liverpool, Liverpool, L69 3Gj, U.K. and Electric Power Research Institute Qinghe, Beijing 100085, P.R. China

ACKNOWLEDGEMENTS

I would like to thank those who made it possible to complete this thesis. I wish to thank Dr. Gerry Sheble for his help in conducting research and writing my thesis. I also wish to thank Dr. Eric Bartlett for his support in my first days of graduate school and his continued support until I finished. Finally, I wish to thank Dr. Dan Ashlock for his support, his teaching, and for helping to remind me why I started graduate school in the first place.