Evolution of Bidding Strategies Using Self-Adaptation

Gan Kim Soon
School of Engineering and Information Technology
Beg Berkunci 2073, 88999 Kota Kinabalu, Sabah
g_k_s967@yahoo.com

Patricia Anthony
School of Engineering and Information Technology
Beg Berkunci 2073, 88999 Kota Kinabalu, Sabah
panthy@ums.edu.my

Jason Teo
School of Engineering and Information Technology
Beg Berkunci 2073, 88999 Kota Kinabalu, Sabah
jtwteo@ums.edu.my

ABSTRACT
Online auctions play an important role in today's e-commerce for procuring goods. The proliferation of online auctions has caused the increasing need of monitoring and tracking multiple bids in multiple auctions. This paper investigates the application of self adaptive genetic algorithms on a flexible and configurable heuristic decision making framework that can tackle the problem of bidding across multiple auctions that apply different protocols (English, Vickrey and Dutch) by using an autonomous agent to search for the most effective strategies (offline). Genetic algorithm has shown promising results in searching in large and little priori information space but our study shows that self adaptive genetic algorithm is able to perform better than genetic algorithm in many cases. An empirical evaluation on the effectiveness of genetic algorithm and self adaptive genetic algorithm for searching the most effective strategies in the heuristic decision making framework are discussed in this paper.

Keywords
Agent, genetic algorithm, heuristic making decision, self adaptive.

1. INTRODUCTION
Online auction has been a main tool for procuring goods and items in today's e-commerce market either for commercialize use or for personal use. EBay.com, lelong.com, uBid.com and Yahoo! Auction are some of the well known auction for procuring goods in today's online market. As the number of auctions increase, the process of monitoring, tracking bid and bidding in multiple auctions become a problem. The user needs to monitor so many auctions sites, picks the right auction to participate, and makes the right bid in making sure that the desired item satisfied the user’s preference. All these tasks are somewhat complex and time consuming. The task even gets more complicated when they are different start and end times and when the auctions employ different protocols. For this reasons, many auctions sites have provide bots to free the user to a certain extent. This doesn't really help much because the bots is only for a particular auction with a particular protocol. Moreover, the bots remains in the same auction sites and does not move to other auction sites. These bots still need the intervention of the user in that the user still needs to make decision on the starting bid initially and the bid increment. One potential problem that will be faced by the user is the winner’s curse where the user will end up paying more than the actual value for the desired item.

To address these shortcomings, we believe it is necessary to develop an autonomous agent that can participate in multiple heterogeneous auctions, that is empowered with trading capabilities and that can make purchases autonomously. In more detail, the agent should monitor and collect information from the ongoing auctions, make decisions on behalf of the consumer and endeavor to guarantee the delivery of the item. The agent must ensure that it never bids above the private valuation (the maximum amount that the consumer is willing to pay) and it tries to get the item in a manner that is consistent with the consumer’s preferences (e.g., at the earliest time, at the lowest price, or with maximum chance of succeeding) [2].

Genetic algorithm (or GA) was first introduced by John Holland in 1975. GA is a search technique used in computing to find true or approximate solutions to optimization and search problems. Genetic algorithms are a particular class of evolutionary computation that use techniques inspired by evolution biology such as inheritance, mutation, selection, and crossover (also called recombination) [1]. To date, many principles has been use to improve the performance of genetic algorithms. One of the principles is self-adaptation. Self adaptation has been applied in evolutionary algorithms as a method to improve the limitation in the performance of evolutionary algorithms. Many researches have been conducted by the evolutionary computation community such as adaptive mutation operator [13] and adaptive reproductive operators [15]. Self adaptation is a method to adjust the setting of the control parameters using the algorithm where the control parameters are embedded in the chromosome of the individual [11]. The principle of self-adaptation has commonly been use in evolutionary programming and evolutionary strategies but it is rarely use in genetic algorithms. In standard genetic algorithms the chromosome will only have the representation of genotype which means the representation will not have any control parameters or phenotypes. Many researches have shown that self adaptation aims at biasing the distribution towards appropriate

1 http://www.ebay.com
2 http://www.lelong.com
3 http://www.ubid.com
4 http://auction.yahoo.com
regions of the search space and maintaining sufficient diversity among individuals in order to enable further evolvability. For this paper, the individual will embed two important control parameters into the chromosome which are the crossover rate and the mutation rate. These two parameters will evolve along with the object parameters by the algorithms. Detail of the experiment will be discussed in the later part of this paper.

As mentioned above, genetic algorithm has been applied in the framework to explore the optimal strategy in the auction environment. This paper will endeavor to investigate the relative performance of self adaptive genetic algorithm by making use of the online auction strategy developed in this work. The main contribution of this paper is to make a comparison between the performances of the strategies evolved using genetic algorithms as opposed to performance of the strategies evolved using self-adaptive genetic algorithms in an auction environment.

The remainder of the paper is organized as follow. Section 2 describes the bidding strategy framework, and the evolved strategy is discussed in Section 3. The experimental evaluation is discussed in Section 4. Section 5 discusses related work and finally Section 6 presents the conclusion and future work.

2. Bidding Strategy Framework

Before describing the decision-making framework, it is necessary to detail our assumptions about the environment. Firstly, we consider three auction protocols: English, Dutch and Vickrey (three of the most common types). Secondly, all auctions have a known start time and English and Vickrey auctions have a known end time. Thirdly, our bidding agent is given a deadline (tmax) by when it must obtain the desired item and it is told the consumer’s private valuation (p) for this item. Fourthly, the agent must not buy more than one instance of the desired item.

The agent’s decision-making model works in the following manner (see [2] for a complete description). The bidder agent builds an active auction list (auctions that have started but not reached their end times, denoted as L(t)) and gathers relevant information (e.g. start and end times, current bid values) about them. It then calculates the current maximum bid it is willing to make at the current time. This current maximum bid, by definition, will always be less than or equal to the private valuation. To determine the current maximum bid, the agent considers several bidding constraints including the remaining time left, the remaining auctions left, the user’s desire for a bargain and the user’s level of desperation. For each such bidding constraint, there is a corresponding function that suggests the value to bid based on that constraint at that time. These (polynomial) functions (based on [6]) are parameterized by two key values: k (range [0,1]) is a constant that determines the value of the starting bid and \( \beta \) (range [0.005 - 1000]) defines the shape of the curve (and so the rate of concession to \( p_r \)). As an example, when \( k \geq 0.5 \) and \( \beta \geq 1 \), the agent demonstrates a reasonable degree of desperation and starts bidding close to \( p_r \) and quickly reaches \( p_r \). At the other extreme, the agent can demonstrate hard bargaining behaviour \((k < 0.5 \) and \( \beta < 1 \)), where it makes a low initial bid and only concedes up to \( p_r \) in a very slow fashion. All behaviours in between are also possible by setting the parameters appropriately. At any given time, the agent may consider any of the bidding constraints individually or it may combine them depending on the situation (what the agent sees as being important at that point in time). If the agent combines multiple bidding constraints, it allocates a weight to each of them to denote their relative importance. The set of functions is referred to as the tactics and the combination of these tactics are referred to as the strategy. Based on the value of the current maximum bid, the agent selects the potential auctions in which it can bid and calculates what it should bid at this time in each such auction. The auction and corresponding bid with the highest expected utility is then selected from the potential auctions as the target auction. Finally, the agent bids in the target auction.

A series of experiments were conducted in a controlled environment to test the efficiency (in terms of success rate and average payoff) of the agent’s strategy (details can be found in [2]). The results of these experiments led to several conclusions. Firstly, \( p_r \) is one of the most important factors that needs to be considered when determining the strategy that should be employed by the agent. This is important, for example, since an agent with a very low \( p_r \) cannot practically look for a bargain and the agent should therefore consider this when accepting the user’s preferences. The second observation is that the remaining time and auction tactics are the key determinants of successful behaviour. Thirdly, the strategies to be used by the agent need to be dynamic, since not all strategies work well in all situations. Thus, a successful strategy in one situation may perform badly in another. Nevertheless, it is possible to determine that certain classes of strategy are effective in environments that have particular characteristics. In this case, the key defining characteristics of an environment were found to be the number of auctions that are active before \( t_{max} \) and the time the agent has to purchase the item. Given this, it was decided to evolve strategies that are effective in these classes of environment.

3. Evolving Strategies

The number of strategies that can be employed is infinite because the bidding agent is heavily influenced by the strategy employ which in turn relates to the values of k and \( \beta \) in the given tactics and the weights for each tactic when these are to be combined. The search space for the solutions in very large, thus a means of automating the process of finding the successful strategies is necessary. Genetic algorithm is decided to be used to explore the search space because genetic algorithm has been shown to perform well in areas where the space to be searched is large and is not well understood. Since genetic algorithm is being applied to explore the solution, the strategies parameters need to be encoded into a chromosome or an individual to represent them as the genotype in the genetic algorithm. Figure 1 shows the encoding of the bidding strategy.

![Figure 1. Encoding of a bidding strategy.](image)

The genes of the individual contain the parameters for the four bidding tactics and the associated weight of the bidding tactics. The values for all the parameters are represented as floating point. The performance of the individuals in the population is evaluated using the fitness function. There are three fitness function use to
evaluate the performance of the individual in the population. These are the individual success rate in obtaining the item (Fitness Equation 1) and two variations based on the average utility. In the first case (Fitness Equation 2), the agent gets a utility of 0 if it fails to obtain the item. If it is successful, the utility of winning in an auction \( i \) is computed as

\[
U_i(v) = \left( \frac{p_r - v}{p_r} \right) + c
\]

where \( v \) is winning bid and \( c \) is arbitrary constant ranging from 0.001 to 0.005 to ensure the agent is awarded with some value where \( v \) is winning bid and \( c \) is arbitrary constant ranging from 0.01 to 0.05. This value incase the winning bid is equivalent to its private valuation. The final utility function (Fitness Equation 3) is similar to Fitness Equation 2 but the individual is penalized if it fails to get the item. In this case the penalty incurred ranges from 0.01 to 0.05. These values were chosen to analyze how the population evolves with varying degrees of penalty. The fitness score is then computed by taking the average utility from a total of 2000 runs.

There are many variations of genetic algorithms today. However, the one applied in this problem domain is the simple genetic algorithm. The initial bidder population is represented by 50 individuals that are generated randomly from the range of specified values. Each individuals will be evaluated based its fitness function. The selection process ensures the fitter individuals survive to the next generation. “Elitism” is applied to retain the best 10% individual of the current generation to the next generation. Tournament selection is use to select the 90% of the population of the next generation. After the selection process, each individual will be subjected to crossover and mutation. In the crossover, two individuals are randomly selected from the mating pool with crossover probability of \( p_c = 0.6 \), and using 2 crossover points that are randomly picked to exchange their genetic material. The exchanging of genetic material process is performed using an extension combination operator [Beasley et al. 1993b], which works by taking the difference between two values of the crossover point, adding this difference to the higher (giving the maximum range) and subtracting it from the lower (giving a minimum range). The new values are then generated between the minimum and maximum range. Mutation follows the crossover process where this allows an increase in the variation of the population to be explored. The mutation in this case is set to \( P_m = 0.02 \). The gene from the chosen individual is picked randomly and a small value (0.05) is added or subtracted, depending on the range limitation for that particular gene. The mutation process is only applied to the values of \( k \) and \( B \) for each tactic. The weights are not considered here because adding a small value to the weight requires a renormalization and will have very little effect on the agent’s online behavior. The stopping criterion for the process normally stop after the population has converges. In this case, the population always converges before 50 iterations (typically, the value lies between 24 and 40). Thus, for this case the stop criterion for the searching will only stop if the iteration reaches 50 times.

At the same time, we also implemented the self-adaptive genetic algorithm as a variation of the simple genetic algorithm. The main difference between the simple genetic algorithm and the self adaptive genetic algorithm is in the way the crossover and the mutation rate is represented. Instead of having a fixed values for both the crossover and mutation rate, these two values will be encoded as part of the chromosome and these values will also be evolved. The crossover and mutation process in self adaptive genetic algorithm is similar with genetic algorithm where the crossover process occurred after the selection process and the mutation process follow after the crossover process. The difference is that crossover and mutation process in self adaptive genetic algorithm is not based on a global crossover and mutation rate but is based on the crossover and mutation rate of the parent that selected to produce offspring.

\[
\begin{align*}
\beta_1 & = \beta_2 & \beta_3 & = \beta_4 & \beta_5 & = \beta_6 & \beta_7 & = \beta_8 & \beta_9 & = \beta_{10} & \beta_{11} & = \beta_{12} & \beta_{13} & = \beta_{14} & \beta_{15} & = \beta_{16} & \beta_{17} & = \beta_{18} & \beta_{19} & = \beta_{20} & \beta_{21} & = \beta_{22} & \beta_{23} & = \beta_{24} & \beta_{25} & = \beta_{26} & \beta_{27} & = \beta_{28} & \beta_{29} & = \beta_{30} & \beta_{31} & = \beta_{32} & \beta_{33} & = \beta_{34} & \beta_{35} & = \beta_{36} & \beta_{37} & = \beta_{38} & \beta_{39} & = \beta_{40} & \beta_{41} & = \beta_{42} & \beta_{43} & = \beta_{44} & \beta_{45} & = \beta_{46} & \beta_{47} & = \beta_{48} & \beta_{49} & = \beta_{50} & \beta_{51} & = \beta_{52} & \beta_{53} & = \beta_{54} & \beta_{55} & = \beta_{56} & \beta_{57} & = \beta_{58} & \beta_{59} & = \beta_{60} & \beta_{61} & = \beta_{62} & \beta_{63} & = \beta_{64} & \beta_{65} & = \beta_{66} & \beta_{67} & = \beta_{68} & \beta_{69} & = \beta_{70} & \beta_{71} & = \beta_{72} & \beta_{73} & = \beta_{74} & \beta_{75} & = \beta_{76} & \beta_{77} & = \beta_{78} & \beta_{79} & = \beta_{80} & \beta_{81} & = \beta_{82} & \beta_{83} & = \beta_{84} & \beta_{85} & = \beta_{86} & \beta_{87} & = \beta_{88} & \beta_{89} & = \beta_{90} & \beta_{91} & = \beta_{92} & \beta_{93} & = \beta_{94} & \beta_{95} & = \beta_{96} & \beta_{97} & = \beta_{98} & \beta_{99} & = \beta_{100}.
\end{align*}
\]

Figure 2. Encoding of a bidding strategies for self-adaptive GA

4. Experimental Evaluation

The purpose of the experiment is to compare the effectiveness of self-adaptive genetic algorithm with genetic algorithm in searching for the nearly optimal bidding strategies. The parameters for the empirical evaluations are shown in Table 1. The parameters include the agent reservation price, the agent bidding time and the number of active auctions. The agent reservation price is the maximum amount that the agent is willing to pay for the item. The bidding time is time allocated for the agent to obtain the user’s required item. The active auctions are the list of auctions that is ongoing before \( t_{\text{max}} \).

| Table 1. Configurable parameters for the testing environment |
|------------------|------------------|
| Agent reservation price | 73 \leq P_r \leq 79 |
| Bidding time for each auction | 21 \leq t_{\text{max}} \leq 50 |
| Number of active auction | 20 \leq L(t) \leq 45 |

The performance of the evolved strategies is evaluated based on three measurements:

1. The Average Fitness

This is the average fitness obtained by the population at each generation over 50 generations. The average fitness show how well the strategy converges over time to find the best solution.

2. The Success Rate

This is the percentage of time that an agent succeeds in acquiring the item by the given time at any price less than or equal to its private valuation. This measure is important in determining how successful the agent is, since one of the agent’s key tasks is to deliver the item to the user (when possible). Thus, this measure will determine the efficiency of the agent in terms of guaranteeing the delivery of the requested item.

3. The Average Payoff

Let \( p_i \) be the agent’s private valuation of the target item, and let \( n \) be the number of times the marketplace was run. Here, the average payoff (\( \rho \)), is defined as

\[
\rho = \sum_{i=1}^{n} \frac{p_i - v_i}{p_i}
\]

where \( v_i \) is the winning bid for auction \( i \). The average payoff is calculated by deducting the agent’s bid value (the value at which it acquires the item) from the private valuation. This value is then
divided by the agent’s private valuation, summed and average over the number of runs. The agent’s payoff is 0 if it is not successful in obtaining the item.

A set of experiments had been run to test the performance of both the self-adaptive genetic algorithms and the genetic algorithm (can also be referred to as non-adaptive genetic algorithm). Figure 3 shows the average fitness for the both the non-adaptive genetic algorithm and the self-adaptive genetic algorithm. It can be seen that the self-adaptive mode converges faster than the non-adaptive GA. However, the self-adaptive genetic algorithm achieved a higher average fitness when compared to the non-adaptive model. This is an indication that the population evolving from the self-adaptive genetic algorithm model will have higher fitness and would most likely perform better than the individuals from non-adaptive genetic algorithm model. The next two experiments will be able to confirm this hypothesis.

To measure the success rate and the average payoff of the individuals evolved from the non-adaptive and the self-adaptive genetic algorithm, we selected individuals from both the algorithms and run them into an offline market environment. For each strategy, we ran it 200 times in the marketplace. Figure 4 shows the comparison of the success rate between the individuals of the non-adaptive and the self-adaptive model. The result shows that the self-adaptive model performed better than the non-adaptive model. All the five individuals evolved from the self-adaptive model outperformed the top five individuals evolved from the non-adaptive genetic algorithm. Here, we can conclude that the self-adaptive model can evolve better strategies that will deliver higher success rate when bidding in online auctions.

Based on the results of the experiments that had been carried out, it can be concluded that the strategies evolved from the self-adaptive genetic algorithm performed better than the strategies evolved from the non-adaptive genetic algorithm in terms of success rate and average payoff in an online auction environment setting. It also achieved a higher average fitness function during the evolution process. Self-adaptive genetic algorithm performed better because the aims of the self-adaptation is to focus on the appropriate region of the search space thus allowing better bidding strategies to be found. Besides that, the self-adaptive genetic algorithm is able to maintain sufficient diversity among individuals in order to enable further evolvability so that the search would trap in local optima.

![Figure 3. Compare average fitness between non adaptive GA and self adaptive GA.](image)

Figure 3. Compare average fitness between non adaptive GA and self adaptive GA.

![Figure 4. Compare success rate between non adaptive GA and self adaptive GA.](image)

Figure 4. Compare success rate between non adaptive GA and self adaptive GA.

![Figure 5. Compare average payoff between non adaptive GA and self adaptive GA.](image)

Figure 5. Compare average payoff between non adaptive GA and self adaptive GA.

5. Related work

Genetic algorithm had been proved work well in many of the real-world applications. One of the examples is the optimization of routes in a telecommunication network [15]. The objective is to find the best route to connect a call between the origin and destination switches. Genetic algorithm proves work well in achieving this objective. At the same time, prove that genetic algorithm perform well in search for a nearly optimum solution in large space solutions. Many research had conducted to enhance genetic algorithm in order improve its performance. Esponiza, et al. have proposed adaptive genetic algorithm where individuals can explore solutions in local search independently of each other [8]. The algorithm emphasize on the exploration efficiency per evaluation improved by automatically adjusting ratio between computation costs of crossover/mutation and local search. Back proposed an adaptive genetic algorithm where each individual has...
its own mutation rate encoded in its gene so that individuals with good mutation rates are expected to survive [6]. Yoshio Murata et al. proposed an agent oriented self adaptive genetic algorithm [12]. This algorithm uses a hybrid algorithm consisting of genetic algorithm with distributed environment scheme and the meta genetic algorithm. Multiple agents is use to run a genetic algorithm to reduced computation costs in this algorithm. This algorithm could simultaneously adapt for four parameters during exploration process in reasonable computation cost. Genetic algorithm has been use for evolution of utility bidding strategies for the competitive marketplace [14]. Genetic algorithm is being use to evolves bidding strategies as gencos and discos trade power.

6. Conclusion and future work

This paper investigates on how well the application of self adaptive genetic algorithm can perform on a flexible and configurable heuristic decision making framework that can tackle the problem of bidding across multiple auctions that apply different protocols (English, Vickrey and Dutch) by using an autonomous agent to search for the most effective strategies (offline) compared to non adaptive genetic algorithm. The polynomial nature of the strategies formulation creates a large search space and making it difficult to search for the optimal solution. In this paper, both the non adaptive and the self adaptive genetic algorithm were applied to enhance the search for the effective strategy. The experimental evaluation showed that the strategies evolved from the self adaptive genetic algorithm performed better than the strategies evolved from the non adaptive genetic algorithm in terms of success rate and average payoff when bidding in the online auction marketplace, Self adaptive genetic algorithm is still in a premature stage because self adaptation is rarely used in genetic algorithm compared to evolutionary programming and evolutionary strategies. This is just a preliminary result of the research. The self adaptive genetic algorithm may be able to perform even better by tuning the control parameters. There are several areas that require further investigation. First, further explore how self adaptive genetic algorithm can help in improving the performance of the bidding strategies include manipulating the control parameters of the individuals Second, explore how well can other variations of genetic algorithm perform in this framework. A comparison of the different genetic algorithm can be done to show how well each algorithms perform in this particular framework.

7. REFERENCES

[18] Vickrey, W. Counterspeculation, auctions and competitive sealed tenders. J. Fin. 16, 8–37, 1961