Option Valuation with Generalized Ant Programming

Christian Keber
University of Vienna
Department of Business Administration
Bruenner Strasse 72, A-1210
christian.keber@univie.ac.at

Matthias G. Schuster
University of Vienna
Department of Business Administration
Bruenner Strasse 72, A-1210
matthias.schuster@univie.ac.at

Abstract

For the valuation of American put options exact pricing formulas haven't as yet been derived We therefore determine analytical approximations for pricing such options by introducing the *Generalized Ant Programming* (GAP) approach applicable to all problems in which the search space of feasible solutions consists of computer programs. GAP is a new method inspired by Genetic Programming as well as by Ant Algorithms. Applying our GAP-approximations for the valuation of American put options on non-dividend paying stocks to experimental data as well as huge validation data sets we can show that our formulas deliver accurate results and outperform other formulas presented in the literature.

1 INTRODUCTION

In their seminal papers Black and Scholes (1973) and Merton (1973) derived an analytical solution for the valuation of European call options on stocks paying no dividends during the time to expiration. Merton additionally showed that premature exercising of American call options on this type of stocks is never optimal and that the valuation can also be made using the Black/Scholes–Merton method. If in the case of an American call option, dividends are paid, and these are known with certainty, its value can be obtained using the analytically exact pricing model of Roll (1977).

In comparison to American call options, premature exercise of American put options on non-dividend paying stocks may produce benefits, if the stock price falls below a certain, permanently variable critical value (*killing price*). As a result of this difference between the premature exercising of American call options and of American put options, ascertaining the optimal time for premature exercising, or the killing price, is a part of the problem to be solved, for which there was previously no exact model. Thus, the valuation of American put options is based on numerical procedures or analytical approximations. The best-known numerical procedures are the *lattice approach* of Cox, Ross, and Rubinstein (1979) and the *finite difference method* of Brennan and Schwartz (1977). However, getting accurate results by using numerical procedures normally requires long calculation time. A simple analytical approximation for the valuation of American put options on non-dividend paying stocks was presented by Johnson (1983). This simple analytical approximation, however, is not very accurate, so that many more ambitious analytical approximations have been developed. The best-known of these come from MacMillan (1986) and Geske and Johnson (1984).

MacMillan's analytical approximation for the valuation of American put options consists of raising the value of a suitable European put option by the value of the approximately calculated premature exercising of the option. This method forms the basis for analytical approximations used to evaluate a range of other American options, such as index options, currency options, and options on futures (see, e.g., Barone-Adesi and Whaley (1987)). Geske and Johnson's analytical approximation assumes that premature exercise is possible only at particular, discrete points in time. Accordingly, for each of these moments an option value is calculated, and, based on this, the value of the American put option is ascertained using the polynomial extrapolation method. This method treats American put options both with and without dividends. The MacMillan method, however, deals only with put options on stocks paying no dividend during the maturity of the options, although it was extended by Barone-Adesi and Whaley (1988) as well as Fischer (1993) to

include the dividend paying case. It is characteristic of most approximations for the valuation of American put options found in the literature that they approximate either the stochastic stock price process, or the partial differential equation (with the appropriate boundaries) which implicitly describes the option price. In contrast, Geske and Johnson (1984) as well as Kim (1990) formulate a valuation equation which represents an exact solution of the partial differential equation, and then solve it either by analytical approximation or by numerical techniques.

During the last few decades an increasing number of researchers of various disciplines has been impressed by the problem-solving power of nature. They developed optimization algorithms and heuristics based on the imitation and simulation of admirable natural phenomena. For example, *Artificial Neural Networks* imitate the principle of human brains and *Evolutionary Algorithms* make use of the *Darwinian principle* of the *survival–of–the–fittest*. Both methodologies are used to solve problems which are normally not amenable to traditional solution techniques and have been applied to option pricing problems. For instance, Hutchinson, Lo, and Poggio (1994) employ an *artificial neural network* for the valuation of European options whereas Chen, Lee, and Yeh (1999), Chidambaran, Lee, and Trigueros (2000), and Keber (2000) use Genetic Programming to solve option pricing problems.

A further example and one of the most recent developments of nature-based solution techniques is the *Ant Colony Optimization* meta-heuristic. *Ant Algorithms* originally introduced by Dorigo and colleagues (Dorigo (1992) and Dorigo, Maniezzo, and Colorni (1991)) as a multi-agent approach to difficult combinatorial optimization problems like the travelling salesman problem (TSP) are inspired by the foraging behaviour of real ant colonies, in particular, how ants can find shortest paths between two points. Real ant colonies are capable of solving such problems using *collective behaviour* and indirect communication via a chemical substance called *pheromone* deposited on the ground. It is hardly surprising that for the first time ant algorithms have been applied to the travelling salesman problem because the analogy between the real ants' problem and the TSP is obvious. In the meantime, ant-based algorithms have been applied successfully to a broad field of combinatorial optimization problems (see, e.g., Dorigo, Caro, and Gambardella (1999)).

In this contribution we develop the *Generalized Ant Programming* (GAP) approach as a new variant of ant algorithms and apply the proposed method to the option valuation problem. GAP enables computers to solve problems without being explicitly programmed. It is applicable to all problems in which the search space of feasible solutions consists of computer programs. GAP works by using artificial ants to automatically generate computer programs. As an acid test for GAP we derive analytical approximations for the valuation of American put options on non-dividend paying stocks. We focus on this problem for several reasons. Firstly, analytical exact solutions for pricing American puts have not as yet been derived. Secondly, testing new techniques should not be based on simple problems because any assessment of the proposed method would be open to criticism. Furthermore, our results have to be compared with other approximations presented in the literature. This can be easily done by using the most frequently quoted approximations mentioned above. Using experimental data as well as huge validation data sets we can show that the GAP based formulas for the valuation of American put options on non-dividend paying stocks deliver accurate approximation results and outperform other approximations presented in the literature.

In the second section we focus on the concept of the Generalized Ant Programming approach. The third section presents the GAP-approximations for the valuation of American put options on non-dividend paying stocks. In the fourth section we show the experimental results. The contribution concludes with a summary.

2 GENERALIZED ANT PROGRAMMING

2.1 INTRODUCTION

The Generalized Ant Programming (GAP) approach is a new method inspired by the Genetic Programming approach introduced by Koza (1992) as well as by Ant Algorithms originally presented by Dorigo (1992) as a multi-agent approach to difficult combinatorial optimization problems like TSP. GAP is an approach designed to generate computer programs by simulating the behaviour of real ant colonies. When travelling real ants deposit pheromone on the ground which influence the choices they make. Ants tend to choose steps marked by strong pheromone concentrations. Pheromone trails can be seen as "public information" which is modified by ants to reflect their experience while solving a problem, e.g., to find shortest paths between the nest and food sources. The quantity of pheromone left by an ant depends on the amount of food found. Within a given interval of time, shorter paths can be travelled more often, which causes a stronger pheromone concentration. In return, this

increases the probability of the path to be chosen.

2.2 METHODOLOGY

Generalized Ant Programming is an algorithmic framework which enables computers to solve problems without being explicitly programmed. It is applicable to all problems in which the search space of feasible solutions consists of computer programs. GAP works by using artificial ants to automatically generate computer programs. Similar to real ants, the artificial ants explore a search space now representing the set of all feasible computer programs which we describe as paths through a graph. The pheromone amount deposited by an artificial ant depends on the quality of the solution found. In other words, it depends on which path (computer program) was chosen. The quality of a path is measured using the corresponding computer program as an "input parameter" to an "algorithmic regression problem". These transpositions lead to our proposed GAP approach which we describe in a more detailed way in the next few paragraphs.

Computer programs are usually based on a well defined programming language. In our GAP–application we therefore use a programming language $\mathcal{L}$ specified by the context-free grammar $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, \mathcal{S})$, see, e.g., Aho and Ullmann (1972). $\mathcal{L}(\mathcal{G})$ is to be seen simply as the set of all analytical expressions which can be produced from a start symbol $\mathcal{S}$ under application of *substitution rules* $\mathcal{R}$, a finite set of *non-terminal* symbols $\mathcal{N}$, and a finite set or vocabulary of *terminal symbols* $\mathcal{T}$. Thus,

$$\mathcal{L} = \{p \mid \mathcal{S} \Longrightarrow p \ \wedge \ p \in \mathcal{T}^*\} \tag{1}$$

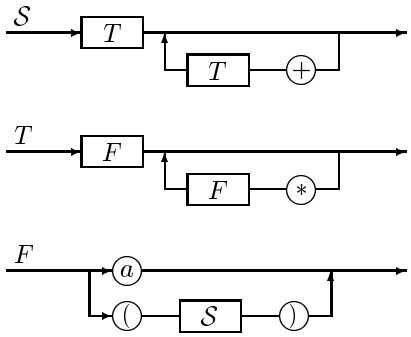
where $\mathcal{T}^*$ represents the set of all analytical expressions which can be produced from the symbols of the vocabulary $\mathcal{T}$. Using the grammar $\mathcal{G}$ a derivation of an analytical expression $p \in \mathcal{L}$ consists of a sequence $\mathfrak{t}_1, \mathfrak{t}_2, \ldots, \mathfrak{t}_\mathfrak{p}$ of terminal symbols and the corresponding derivation steps (productions) $\mathfrak{t}_i \rightarrow \mathfrak{t}_{i+1}$ (for $i = 1, \ldots, \mathfrak{p} - 1$). This derivation is denoted by

$$\mathcal{S} \underset{\mathcal{G}}{\overset{*}{\Longrightarrow}} p. \tag{2}$$

To take a simple example, assume

$$\begin{aligned}\mathcal{G} = (&\mathcal{N} = \{\mathcal{S}, T, F\},\\ &\mathcal{T} = \{a, +, *, (,)\},\\ &\mathcal{R} = \{\mathcal{S} \rightarrow \mathcal{S} + T | T, T \rightarrow T * F | F, F \rightarrow (\mathcal{S}) | a\},\\ &\mathcal{S})\end{aligned}$$

and let us express this grammar in an equivalent graphical representation (*syntax diagram*).



Each derivation in this grammar represents a simple arithmetic expression including the symbols $a, +, *, (,$ and $)$ and can be interpreted as a path through the syntax diagram. An example of a derivation in our simple grammar would be

$$\begin{aligned}\mathcal{S} &\Rightarrow \mathcal{S} + T \Rightarrow T + T \Rightarrow F + T \Rightarrow a + T\\ &\Rightarrow a + T * F \Rightarrow a + F * F \Rightarrow a + a * F \Rightarrow a + a * a\end{aligned}$$

In the sense of GAP, $\mathcal{L}$ is the *search space* of all potential analytical expressions to be generated, $p \in \mathcal{L}$ is a *path* which can be visited by ants, and $\mathcal{J}(t) \subset \mathcal{L}$ is a set of paths already visited at time t. Furthermore, each path $p \in \mathcal{L}$ consists of a sequence of terminal symbols $\mathfrak{t}_1, \mathfrak{t}_2, \ldots, \mathfrak{t}_\mathfrak{p}$ and the corresponding derivation steps $\mathfrak{t}_i \rightarrow \mathfrak{t}_{i+1}$ (for $i = 1, \ldots, \mathfrak{p} - 1$).

In GAP each path $p_i \in \mathcal{J}(t)$ can be seen as a derivation

$$\mathcal{S} \underset{\mathcal{P}_i}{\overset{*}{\Longrightarrow}} p_i, \tag{3}$$

where $\mathcal{P}_i \subset \mathcal{G}$. While walking each ant forms a new path p' which is a derivation based on $\mathcal{P}' = \cup_{i=1}^{n} \mathcal{P}_i \subset \mathcal{G}$,

$$\mathcal{S} \underset{\mathcal{P}'}{\overset{*}{\Longrightarrow}} p', \tag{4}$$

and where all derivation steps contained in p' are selected according to the pheromone amounts of the corresponding paths p_i. In the proposed GAP-application the pheromone trail is put on whole paths implying that the derivation steps describing a path are equally weighted. This is just for simplification and can be extended to different pheromone amounts along a path. We are going to present this extension in one of our next papers.

The amount of pheromone trail on path p at time t is given by

$$\tau_p(t) = (1 - \rho) \cdot \tau_p(t) + \Delta\tau_p(t), \tag{5}$$

where $0 < \rho \leq 1$ is the coefficient representing pheromone evaporation, and

$$\Delta\tau_p(t) = \sum_{k=1}^{\mathcal{K}} \Delta\tau_p^k(t)$$

is the pheromone increase obtained by cumulating the contributions $\Delta\tau_p^k(t)$ of each ant $k = 1, \ldots, \mathcal{K}$. In other words, this is the amount of pheromone deposited on path p by the k^{th} ant at time t. This quantity of pheromone trail is given by

$$\Delta\tau_p^k(t) = \begin{cases} Q \cdot L_k(t,p) & \text{if } k^{\text{th}} \text{ ant takes path } p \\ 0 & \text{otherwise} \end{cases} \qquad (6)$$

where Q is a constant and $L_k(t,p)$ is the value of the *objective function* obtained by ant k at time t. As GAP is similar to the Genetic Programming approach each path $p \in \mathcal{L}$ represents a computer programm (or an analytical expression) and can therefore be seen as a function $p\colon \mathcal{E} \to \mathcal{A}$ transforming input data $\mathcal{E}$ into a solution or output data $\mathcal{A}$. Accordingly, the function $L_k : \mathcal{A} \to I\!R$ has to be defined in a way that it awards higher values to those paths (computer programs) which represent a good solution to the task in hand, and lower values to less suitable paths (computer programs). The value of the objective function is measured using a representative set of test records $\mathcal{E}_i$, for $i = 1, \ldots, \mathcal{D}$. If these input data are processed using the computer program $p \in \mathcal{J}(t)$, the result will be the output data $\mathcal{A}_i$, which can then be compared to the target output data $\mathcal{A}_i^S$. Using a deviation function $\delta(\mathcal{A}_i, \mathcal{A}_i^S)$ in a way that it delivers higher values the more the output data differ from the target output data, the aggregated deviation is given by

$$F(p) = \sum_{i=1}^{\mathcal{D}} \delta(\mathcal{A}_i, \mathcal{A}_i^S) \quad \text{with} \quad \mathcal{A}_i = p(\mathcal{E}_i). \qquad (7)$$

The objective function $L_k(t,p)$ can be formulated as

$$L_k(t,p) = \frac{1}{1 + F(p)} \qquad (8)$$

so that the values of the objective function lie between zero and one, and larger values represent better paths (computer programs).

For the (new) path $p' \in \mathcal{L}(\mathcal{P}')$ being built by the k^{th} ant, see (4), the probability of selecting derivation steps describing old paths $p_i \in \mathcal{J}(t)$ is given as

$$P_{p_i}^k(t) = \begin{cases} \dfrac{[\tau_{p_i}(t)]^\alpha \cdot [\eta_{p_i}]^\beta}{\displaystyle\sum_{\pi \in \mathcal{P}'_k(t)} [\tau_\pi(t)]^\alpha \cdot [\eta_\pi]^\beta} & \text{if } p_i \in \mathcal{P}'_k(t) \\ 0 & \text{otherwise} \end{cases} \qquad (9)$$

where $\mathcal{P}'_k(t) \subseteq \mathcal{P}'(t)$ is the set of derivation steps of path p_i that the k^{th} ant has not visited yet. η_{p_i} is a heuristic value of including derivation steps of p_i. The parameters α and β control the relative importance of pheromone trail versus visibility.

Using the above definitions, GAP can be outlined by the following (pseudo) computer program.

```
[ 0] program Generalized AntProgramming;
[ 1]   t = 0;
[ 2]   Init α, β, ρ, Q, J(t), τ_p(t);
[ 3]   repeat
[ 4]     t = t + 1;
[ 5]     for each ant k do
[ 6]       Build a path p' according to (4) and (9);
[ 7]       Calculate L_k(t, p') using (8);
[ 8]       J(t) = J(t − 1) ∪ p';
[ 9]       end;
[10]     Save the best solution found so far;
[11]     Update trail levels τ_p(t) according to (5);
[12]     Shrink J(t);
[13]     Perform global shaking on τ_p(t);
[14]   until termination;
[15] end.
```

While most of the programming steps are already discussed, programming step [12], Shrink $\mathcal{J}(t)$, is to be seen in conjunction with step [8], $\mathcal{J}(t) = \mathcal{J}(t-1) \cup p'$. Executing [8] repeatedly implies that the set of paths already visited, $\mathcal{J}(t)$, becomes bigger and bigger or could go to infinity in worst case which is highly undesirable. On the other hand, $\mathcal{J}(t)$ also contains paths where pheromone trails are completely evaporated so that they can be excluded in further exploration, i.e., when ants choose a new path. Hence, in step [12] we shrink $\mathcal{J}(t)$ accordingly. A second step not yet mentioned is the global shaking procedure in [13]. In (1) we choose a specification (of our language) which is of finite size, although the language being specified is not finite. Hence, GAP is comparable to dynamic problems such as the TSP with the insertion or deletion of cities, see, e.g., Bonabeau, Dorigo, and Theraulaz (1999) and Guntsch, Middendorf, and Schmeck (2001). If, in GAP, the pheromone amount on a derivation step (or path) becomes much higher than all others, this step (or path) will almost certainly always be chosen. This would be fine in a static model but is a problem in GAP because it prevents ants from taking new derivation steps. Similar to dynamic approaches we therefore apply the "shaking technique" to normalise the pheromone levels. The formula used in our application is a logarithmic one and is given by

$$\tau_p(t) = \tau_p^0 \cdot \left[1 + \ln\left(\frac{\tau_p(t)}{\tau_p^0}\right)\right], \qquad (10)$$

where τ_p^0 is the minimum value for $\tau_p(t)$, forced by the algorithm, so that $\tau_p(t) \geq \tau_p^0$ (Eyckelhof (2001) and Stützle and Hoos (2000)). In our GAP-application we use the initial value as the lower boundary.

2.3 RELATED WORK

To our knowledge, the first attempt using ants for automatic programming comes from Roux and Fonlupt (2000). At first glance one tends to believe that their approach can solve symbolic regression problems. But on a closer view it is doubtful whether the approach provides accurate approximation results. Hence, we have retraced and implemented their method. Testing the approach on several problems we have got poor approximation results. This is in accordance with the results presented in their own paper which, from our point of view, are not very promising. Furthermore, in the present version of the proposed method the authors focus only on symbolic regression problems. In comparison, our approach can be used for general "*algorithmic* regression problems". Thus, we refer to our approach as *Generalized* Ant Programming.

3 APPLICATION

3.1 EXPERIMENTAL DESIGN

By applying the GAP approach to option pricing we have to specify some parameters. For the aggregated deviations used in the objective function (8) we used a randomly generated training sample of 1,000 American put options on non-dividend paying stocks described by the tuple $\langle P_0, S_0, X, r, T, \sigma\rangle$ where $S_0 > S^*$. S^* represents the killing price calculated using MacMillan's (1986) procedure, P_0 refers to the "exact" value of an American put option on a non-dividend paying stock at $t = 0$, S_0 denotes the stock price at $t = 0$, X is the exercise price, r represents the annual continuous risk-free interest rate (in %), T is the time to expiration (in years), and σ is the annual volatility of the stock price (in %). Each option price P_0 was calculated by using the finite difference method[1] and served as the exact value of the American puts. Applying the property that option prices are linear homogenous in S_0 and X, i.e., $\gamma \cdot f_0(S_0, X, r, T, \sigma) = f_0(\gamma \cdot S_0, \gamma \cdot X, r, T, \sigma)$ where γ is a constant and f_0 denotes a function for calculating the option price, we are able to use a stock price $S_0 = 1$ for all the options of the training sample. The remaining parameters were drawn randomly based on uniform distributions. In accordance with the literature as well as realistic circumstances we defined the following domains: $2 \le r \le 10$, $1/360 \le T \le 1/4$, $10 \le \sigma \le 50$, and $0.8 \le \alpha \le 1.2$. The domain of the moneyness ratio $\alpha = S_0/X$ is based on the consideration that option trading always starts near-the-money. Additionally, Stephan and Whaley (1990) look at a sample of 950,346 stock option transactions and report that the moneyness is between 0.9 and 1.1 in about 78 % of cases.

In accordance with the Generalized Ant Programming approach we transformed each tuple $\langle P_0, S_0, X, r, T, \sigma\rangle$ into an input data record $\mathcal{E}_i$ ($\widehat{=}\ \langle S_0, X, r, T, \sigma\rangle$) and a corresponding target output data record $\mathcal{A}_i^S$ ($\widehat{=}\ \langle P_0\rangle$), for $i = 1, \ldots, 1000$. For the aggregated deviations used in (7) we used the sum of the squared errors $\sum_{i=1}^{1000}(\mathcal{A}_i - \mathcal{A}_i^S)^2$. In the terminal symbol set we included the variables S_0, X, r, T, σ, α, ephemeral constants, the commonly used mathematical operators $+$, $-$, $*$, $\div$, $\sqrt{x}$, $\ln(x)$, x^2, x^y, and the cumulative distribution function of the univariate standard normal distribution $\Phi(x)$. For the substitution rules, $\mathcal{R}$, and the non-terminal symbol set, $\mathcal{N}$, we used subsets of the grammar defining Jensen and Wirth's "Standard Pascal" (see Jensen and Wirth (1975), pp. 110–118). This subset was chosen so that simple analytical expressions can be derived. For the other GAP related parameters we used the following settings resulting in a balanced relationship between convergence speed, calculation effort and effectiveness of the GAP algorithm. The ant colony includes $\mathcal{K} = 50$ members and for the relative importance of pheromone trails we used $\alpha = 1$. The remaining parameters are defined as follows: $\beta = 1, \rho = 0.5$ and $Q = 1$. Among other things, η is used to restrict the formula complexity. Furthermore, we stopped the GAP algorithm after 100,000 cycles.

3.2 APPROXIMATION

Applying the GAP approach as described we get various approximations for the valuation of American put options on non-dividend paying stocks. One of our best approximations (for options with $S_0 = 1$ and $S_0 > S^*$ as mentioned above) is given by

$$P_0 \approx P_0^{GAP} = X \cdot e^{-\bar{r}\cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1) \qquad (11)$$

where

$$d_1 = \frac{\ln(S_0/X)}{\sigma\sqrt{T}}, \qquad d_2 = d_1 - \sigma\sqrt{T}.$$

The above formula is as derived by the GAP algorithm without possible simplification. Looking at (11) it is amazing that our formula can be characterised as a simple approximation when compared to other analytical approximations presented in the literature. Furthermore, our formula shows a strong structural similarity to the Black/Scholes–Merton equation for valuing European put options. Both formulas are identical except for the parameters d_1 and d_2 as well as $\bar{r}$ given in the appendix.

[1] With $\Delta t = (1/365)/5$, $\Delta S = 0.01$, and $S_{\max} = 2 \cdot S_0$.

4 EXPERIMENTAL RESULTS

To give a first impression of the accuracy of the GAP based analytical approximations for the valuation of American put options on non-dividend paying stocks we use the data sets of Geske and Johnson (1984) and Barone-Adesi and Whaley (1988) because these data sets are often used as comparisons. Furthermore, we use a huge sample of 50,000 randomly generated American put options (validation data set). This ensures that the assessment of the approximations will be highly accurate. The corresponding parameters are independent from those of the training sample and their domains are as follows: $10 \leq S_0 \leq 100$, $2 \leq r \leq 8$, $1/360 \leq T \leq 1/4$, $5 \leq \sigma \leq 50$, and $0.8 \leq \alpha \leq 1.2$. The numerically exact put option and killing prices P_0 and S^* were calculated using the finite difference method (with $\Delta t = (1/365)/5$, $\Delta S = 0.01$, and $S_{\max} = 2 \cdot S_0$) and MacMillan's (1986) procedure, respectively. For the *method-related* comparison we use the most frequently quoted analytical approximations P_0^J of Johnson (1983), P_0^{GJ} of Geske and Johnson (1984), and P_0^{MM} of MacMillan (1986).

In Table 1 and 2 the GAP based approximation P_0^{GAP} as well as the most frequently quoted approximations P_0^J, P_0^{GJ}, and P_0^{MM} are applied to the data sets of Geske and Johnson (1984) and Barone-Adesi and Whaley (1988), respectively. P_0 denotes the numerically exact put option price. At the end of each table the three error measures mean absolute error (MAE), mean squared error (MSE) and mean absolute percentage error (MAPE) are given for each approximation. The approximation results can be summarised as follows:

- From Tables 1 and 2 it can be seen that Johnson's (1983) put pricing formula delivers less accurate approximations. The mean absolute errors are about 7 and 71 pence, respectively. The next best approximation comes from the GAP approach having mean absolute errors of about two and three pence, respectively. MacMillan's (1986) and Geske and Johnson's (1984) approximations are better than the others because their mean absolute errors are between about four tenths of a penny and two pence. With respect to the accuracy of the GAP based approximation we have to keep in mind that only 40 % of the options are consistent with the parameter domains used in the GAP application. However, if we look at these options exclusively the GAP based formula delivers the best approximations.

The application results just shown cannot be used for a general assessment of the accuracy of the approximation formulas because the underlying data sets are too small. If we use the 1,000 data records used in the GAP approach as a basis for a general assessment the problem arises that this data sample represents training data, and thus the assessment would be open to criticism of being a "self-fulfilling prophecy". Therefore, the definitive judgement of the approximation formulas is to be made from the above mentioned validation data set which is independent of the training data set. Based on the validation data set Table 3 gives the error measures and the graph in Figure 1 shows the accuracy of the GAP based approximations as well as Johnson's (1983), Geske and Johnson's (1984), and MacMillan's (1986) put pricing formulas. The accuracy is shown in terms of cumulated frequencies of the absolute deviations between the numerically calculated exact put option price and the approximations just mentioned.

- Johnson's (1983) put pricing formula delivers the weakest approximation results. This can be seen from Table 3 as well as the graph in Figure 1. While for the other approximations it can be said with almost 100 % probability that the approximated option prices differ from the numerically

Table 1: Approximations for the value of American put options on non-dividend paying stocks ($S_0 = 40, r = 4.88$, data from (Geske and Johnson 1984, page 1519)).

T	σ	X	P_0	P_0^J	P_0^{MM}	P_0^{GJ}	P_0^{GAP}
0.0833	0.20	35.00	0.0063	0.0062	0.0065	0.0062	0.0063
0.3333	0.20	35.00	0.2001	0.1969	0.2044	0.2000	0.2042
0.5833	0.20	35.00	0.4323	0.4205	0.4415	0.4318	0.4582
0.0833	0.20	40.00	0.8509	0.8406	0.8503	0.8521	0.8533
0.3333	0.20	40.00	1.5787	1.5262	1.5768	1.5759	1.5937
0.5833	0.20	40.00	1.9894	1.8916	1.9888	1.9827	2.0551
0.0833	0.20	45.00	5.0000	4.8403	5.0000	4.9969	5.0000
0.3333	0.20	45.00	5.0875	4.7882	5.0661	5.1053	5.1069
0.5833	0.20	45.00	5.2661	4.8584	5.2364	5.2893	5.3536
0.0833	0.30	35.00	0.0777	0.0777	0.0780	0.0772	0.0773
0.3333	0.30	35.00	0.6967	0.7056	0.7014	0.6972	0.7012
0.5833	0.30	35.00	1.2188	1.2390	1.2281	1.2198	1.2506
0.0833	0.30	40.00	1.3081	1.3047	1.3078	1.3103	1.3104
0.3333	0.30	40.00	2.4810	2.4757	2.4783	2.4801	2.4952
0.5833	0.30	40.00	3.1681	3.1634	3.1667	3.1628	3.2331
0.0833	0.30	45.00	5.0590	4.9910	5.0470	5.0631	5.0578
0.3333	0.30	45.00	5.7042	5.6090	5.6794	5.7017	5.7232
0.5833	0.30	45.00	6.2421	6.1265	6.2150	6.2367	6.3292
0.0833	0.40	35.00	0.2466	0.2499	0.2472	0.2461	0.2461
0.3333	0.40	35.00	1.3447	1.3886	1.3491	1.3461	1.3421
0.5833	0.40	35.00	2.1533	2.2475	2.1619	2.1553	2.1698
0.0833	0.40	40.00	1.7659	1.7763	1.7659	1.7688	1.7670
0.3333	0.40	40.00	3.3854	3.4494	3.3825	3.3863	3.3902
0.5833	0.40	40.00	4.3506	4.4728	4.3494	4.3475	4.3943
0.0833	0.40	45.00	5.2856	5.2706	5.2735	5.2848	5.2847
0.3333	0.40	45.00	6.5078	6.5535	6.4875	6.5015	6.5211
0.5833	0.40	45.00	7.3808	7.4874	7.3597	7.3695	7.4498
MAE				0.0692	0.0082	0.0040	0.0210
MSE ($\times 10^{-5}$)				1351.8420	15.4237	4.4433	117.0473
MAPE				0.0217	0.0045	0.0025	0.0096

Table 2: Approximations for the value of American put options on non-dividend paying stocks ($X = 100$, data from (Barone-Adesi and Whaley 1988, page 315)).

T	σ	S_0	P_0	P_0^J	P_0^{MM}	P_0^{GJ}	P_0^{GAP}
$r = 8.00$							
0.25	0.20	80.00	20.0000	18.0909	20.0000	20.0012	20.0000
0.25	0.20	90.00	10.0353	9.0470	10.0130	10.0730	10.0456
0.25	0.20	100.00	3.2217	3.0378	3.2201	3.2115	3.2262
0.25	0.20	110.00	0.6642	0.6406	0.6810	0.6647	0.6725
0.25	0.20	120.00	0.0888	0.0865	0.0967	0.0879	0.0863
$r = 12.00$							
0.25	0.20	80.00	20.0000	17.1344	20.0000	20.0112	20.0000
0.25	0.20	90.00	10.0000	8.2620	10.0000	9.9811	10.0000
0.25	0.20	100.00	2.9225	2.6265	2.9251	2.9110	2.9050
0.25	0.20	110.00	0.5541	0.5189	0.5781	0.5541	0.5549
0.25	0.20	120.00	0.0685	0.0654	0.0789	0.0676	0.0609
$r = 8.00$							
0.25	0.40	80.00	20.3196	19.7588	20.2478	20.3699	20.3078
0.25	0.40	90.00	12.5635	12.4222	12.5142	12.5511	12.5769
0.25	0.40	100.00	7.1049	7.1204	7.0999	7.1018	7.1162
0.25	0.40	110.00	3.6968	3.7476	3.7120	3.7017	3.7002
0.25	0.40	120.00	1.7885	1.8310	1.8068	1.7892	1.7824
$r = 8.00$							
0.50	0.20	80.00	20.0000	16.6555	20.0000	19.9402	20.0000
0.50	0.20	90.00	10.2890	8.8392	10.2348	10.3712	10.4406
0.50	0.20	100.00	4.1885	3.7889	4.1933	4.1519	4.3474
0.50	0.20	110.00	1.4095	1.3140	1.4459	1.4121	1.5142
0.50	0.20	120.00	0.3969	0.3768	0.4244	0.3961	0.4239
MAE				0.7083	0.0184	0.0173	0.0256
MSE ($\times 10^{-4}$)				14886.3928	7.3752	8.2725	29.2733
MAPE				0.0721	0.0218	0.0034	0.0177

Table 3: Error measures for the approximations for the value of American put options on non-dividend paying stocks based on a sample of 50,000 put options.

	P_0^J	P_0^{MM}	P_0^{GJ}	P_0^{GAP}
MAE	0.0674	0.0059	0.0031	0.0025
MSE ($\times 10^{-3}$)	27.0546	0.1197	0.1008	0.0201
MAX	1.8806	0.0827	1.2102	0.0584

exact option prices by no more than 5 pence, deviation to this level is only found in Johnson's (1983) solution in 72 % of cases. Put another way, in 28 % of cases there is a deviation of more than 5 pence. Due to these poor results, Johnson's (1983) approximation is not considered in further discussions.

- The next best approximations come from MacMillan (1986) and Geske and Johnson (1984) having MAEs of about 6 and 3 tenths of a penny and maximum absolute deviations of about 8 and 120 pence, respectively. In comparison, the GAP based formula delivers the best approximations having a MAE of two and a half tenths of a penny and a maximum absolute deviation of about 6 pence.

- In option pricing we usally look at the penny accuracy of an approximation. From the graphs in Figure 1 it can be seen that the penny accuracy

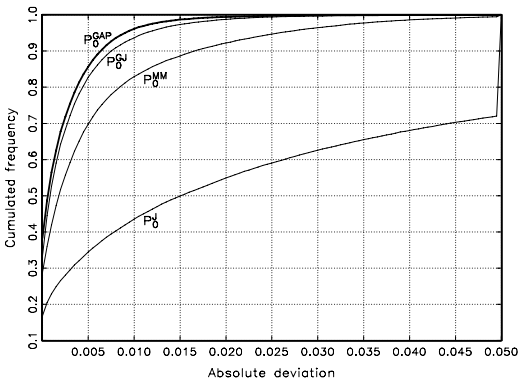


Figure 1: Cumulated frequencies of the absolute deviations between P_0 and the approximations based on a sample of 50,000 put options.

of MacMillan's and Geske and Johnson's, approximations is achieved in about 83 and 94 % of cases, respectively. In comparison, the penny accuracy of the GAP approximation is already achieved in about 96 % of cases.

Summing up, for realistic and frequently observed option parameters we can conclude that the GAP based approximation clearly outperforms the approximations of Johnson, MacMillan as well as Geske and Johnson. Additionally, our approximation consists only of fundamental mathematical operations and is therefore easy to use whereas, e.g., Geske and Johnson's formula requires at least the distribution function of the trivariate, possibly also the multivariate, standard normal distribution which is normally calculated using numerical integration. Moreover, the results presented so far represent work in progress and seem to be very promising.

5 CONCLUSION

In this paper we have introduced the Generalized Ant Programming approach as a new method for solving problems in which the search space of feasible solutions consists of computer programs. We have shown that Generalized Ant Programming can be used to derive accurate analytical approximations for the valuation of American put options on non-dividend paying stocks. Based on experimental data as well as huge validation data sets we have shown that our formula delivers accurate approximation results and outperforms other formulas presented in the literature.

Appendix

$$\bar{r} = -r \cdot \Phi\left(\frac{c_6 \ln(r)}{T}\right)\left[\frac{\ln(X)^2}{\sigma^2} C \sqrt{\Phi\left(\frac{r}{\ln(X)\sigma^2}\right)\frac{r}{T}} - \Phi\left(\Phi\left(\ln(r) D\right)\right)\right]$$

where

$$\begin{aligned}
A &= (r - c_1) X / \Phi\left(T\left[\ln(r) + \frac{r\sqrt{\Phi\left(r/\ln(X)\right) r/T}}{T\sqrt{\alpha X}} - \frac{\ln(X)}{\sigma}\right]\right) \\
B &= \sqrt{X\Phi\left(A\sigma X \Phi\left(-c_2 + T/B^*\right)\right)/B^*}, \quad B^* = \sigma(\ln(X) S_0/\sigma)^2 \\
C &= \sqrt{B + \Phi(-\sigma) - \sqrt{T} + \alpha/\sigma + \Phi(S_0 + C^*) - T} \\
C^* &= \left(\sqrt{\ln(\sigma)\ln(r)} - c_3\right)(T + \ln(X)/\sigma) \\
D &= c_4 \sigma / T\left[\sigma\Phi\left(\ln(X)/\sigma^2 + T\right) + \left(\sqrt{T}\sigma^2 \ln(r)\right)^2 - \Phi(-T)\right] - c_5
\end{aligned}$$

and

$$\begin{array}{lll}
c_1 = 2.190564158 & c_2 = 1.757516557 & c_3 = 7.556576426 \\
c_4 = 0.219427468 & c_5 = 0.119453070 & c_6 = -0.158050093
\end{array}$$

References

Aho, A. V., and J. D. Ullmann, 1972, *The Theory of Parsing, Translation, and Compiling* vol. I Parsing. (Prentice Hall Englewood Cliffs).

Barone-Adesi, G., and R. E. Whaley, 1987, Efficient Analytic Approximation of American Option Values, *Journal of Finance* 42, 301–320.

Barone-Adesi, G., and R. E. Whaley, 1988, On the Valuation of American Put Options on Dividend–Paying Stocks, *Advances in Futures and Options Research* 3, 1–13.

Black, F., and M. Scholes, 1973, The Pricing of Options and Corporate Liabilities, *Journal of Political Economy* 81, 637–659.

Bonabeau, E., M. Dorigo, and G. Theraulaz, 1999, *Swarm Intelligence: From Natural to Artificial Systems.* (Oxford University Press New York).

Brennan, M., and E. Schwartz, 1977, The Valuation of American Put Options, *Journal of Finance* pp. 449–462.

Chen, S.-H., W.-C. Lee, and C.-H. Yeh, 1999, Hedging Derivative Securities with Genetic Programming, *International Journal of Intelligent Systems in Accounting, Finance and Management* 8, 237–251.

Chidambaran, N. K., C. W. J. Lee, and J. R. Trigueros, 2000, Option Pricing via Genetic Programming, in *Computational Finance 1999*, ed. by Y. S. Abu-Mostafa et al. pp. 583–598 Cambridge, Massachusetts. The MIT Press.

Cox, J., S. Ross, and M. Rubinstein, 1979, Option Pricing: A Simplified Approach, *Journal of Financial Economics* pp. 229–263.

Dorigo, M., 1992, Optimization, Learning, and Natural Algorithms, Ph.D. thesis Politecnico di Milano Milano.

Dorigo, M., G. Caro, and L. M. Gambardella, 1999, Ant Algorithms for Discrete Optimization, *Artificial Life* pp. 137–172.

Dorigo, M., V. Maniezzo, and A. Colorni, 1991, Positive feedback as a Search Strategy, Working paper, 91–016 Politecnico di Milano.

Eyckelhof, C. J., 2001, Ants Systems for Dynamic Problems: The TSP Case – Ants caught in a traffic jam, Working paper, University of Twente.

Fischer, E. O., 1993, Analytic Approximation for the Valuation of American Put Options on Stocks with Known Dividends, *International Review of Economics and Finance* pp. 115–127.

Geske, R., and H. E. Johnson, 1984, The American Put Option Valued Analytically, *Journal of Finance* pp. 1511–1524.

Guntsch, M., M. Middendorf, and H. Schmeck, 2001, An Ant Colony Optimization Approach to Dynamic TSP, in *Proceedings GECCO–2001*, ed. by L. Spector et al. pp. 860–867 San Fransico. Morgan Kaufmann.

Hutchinson, J. M., A. W. Lo, and T. Poggio, 1994, A Nonparametric Approach to Pricing and Hedging Derivative Securities Via Learning Networks, *Journal of Finance* pp. 851–889.

Jensen, K., and N. Wirth, 1975, *Pascal, User Manual and Report.* (Springer Verlag New York) 2nd edn.

Johnson, H. E., 1983, An Analytic Approximation for the American Put Price, *Journal of Financial and Quantitative Analysis* pp. 141–148.

Keber, C., 2000, Option Valuation with the Genetic Programming Approach, in *Computational Finance 1999*, ed. by Y.S. Abu-Mostafa et al. pp. 689–703 Cambridge, Massachusetts. The MIT Press.

Kim, I. J., 1990, The Analytic Valuation of American Options, *Review of Financial Studies* pp. 547–572.

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

MacMillan, L. W., 1986, Analytic Approximation for the American Put Option, *Advances in Futures and Options Research* 1, 119–139.

Merton, R. C., 1973, Theory of Rational Option Pricing, *Bell Journal of Economics and Management Science* 4, 141–183.

Roll, R., 1977, An Analytic Valuation Formula for Unprotected American Call Options with Known Dividends, *Journal of Financial Economics* pp. 251–258.

Roux, O., and C. Fonlupt, 2000, Ant Programming: Or How to Use Ants for Automatic Programming, in *Proceedings of ANTS'2000*, ed. by M. Dorigo et al. pp. 121–129.

Stephan, J. A., and R. E. Whaley, 1990, Intraday Price Changes and Trading Volume Relations in the Stock and Stock Option Markets, *Journal of Finance* 45, 191–220.

Stützle, T., and H. H. Hoos, 2000, MAX-MIN Ant System, *Future Generation Computer Systems Journal* 16, 889–914.