A genetic algorithm application in backcross breeding problem

E Carnia*, H Napitupulu, and A K Supriatna

Department of Mathematics, Faculty of Mathematics and Natural Sciences,
Universitas Padjadjaran, Jatinangor 45363, Sumedang, Jawa-Barat, Indonesia

*Email: ema.carnia@unpad.ac.id

Abstract. In this paper we discuss a mathematical model of goat breeding strategy, i.e. the backcrossing breeding. The model is aimed to obtain a strategy in producing better variant of species. In this strategy, a female (doe) of a lesser quality goat, in terms of goat quality is bred with a male (buck) of an exotic goat which has a better goat quality. In this paper we will explore a problem on how to harvest the population optimally. A genetic algorithm (GA) approach will been devised to obtain the solution of the problem. We do several trials of the GA implementation which gives different set of solutions, but relatively close to each other in terms of the resulting total revenue, except a few. Further study need to be done to obtain GA solution that closer to the exact solution.

1. Introduction

In this paper we discuss a mathematical model of goat breeding strategy, i.e. the backcrossing breeding. The model first appeared in [1] and explored further in [2]. The backcrossing breeding strategy is aimed to obtain a better variant of species. In this strategy, a female (doe) of a lesser quality goat, in terms of goat quality such as meat and milk quality, is bred with a male (buck) of an exotic goat which has a better goat quality (a *fullblood*). It is a common practice in Indonesia to breed a doe of Kambing Kacang / *Capra aegagrus hircus*. with an exotic male goat from other variant of the same subspecies (such as Boer goat or Jamnapari goat). The resulting offspring of the first generation is called filial 1 (F1). When the F1 is a female, then she is bred with other male exotic goat of the same variant to produce an F2 female which then subsequently is bred with another male exotic goat of the same variant to produce F3. This process goes on to produce better quality of goat which is often called a *purebred* goat. It is easy to show that lim F*n* → *fullblood*, in terms of quality as *n* approaches infinity. In fact the process is faster to follow the sequence 50% (½), 75% (¾), 88% (7/8), 94% (15/16), 97% (31/32), for *n* = 1,2,3,4,5, respectively, hence the F5 is regarded as a good quality goat (a *purebred*, which the appearence is very closed to the *fullblood*). In this paper we adopt the model in [1], which has a schematic diagram as in Figure 1.

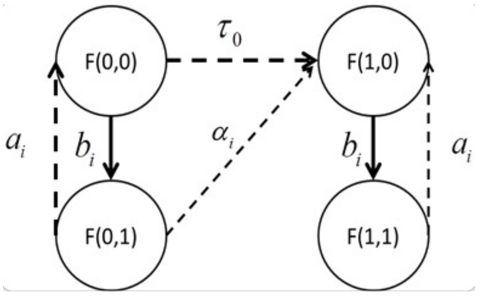

Figure 1. Schematic diagram of backcrossing breeding with two generations and two age-classes (taken from [1])

By refering to Figure 1, suppose that the number of does for each age-class is known at $t = 0$, with:

- $F_t(0,0)$ is the number of lesser quality or local does belong to the first age-class;
- $F_t(0,1)$ is the number of lesser quality or local does belong to the second age-class;
- $F_t(1,0)$ is the number of hybrid does belong to the first age-class;
- $F_t(1,1)$ is the number of hybrid does belong to the second age-class.

The authors in [1] argued that the number of does at time $t = 0$ is $F_{t=0}(G,T) = F_t(0,0) + F_t(0,1) + F_t(1,0) + F_t(1,1)$. In a more compact form, this can be written as a vector $\boldsymbol{F_{t=0}}(G,T) = [F_t(0,0), F_t(0,1), F_t(1,0), F_t(1,1)]^T$. The vector $\boldsymbol{F_t}(G,T)$ is called the initial distribution vector at time t. Further, by the same argument, $F_{t+1}(G,T) = F_{t+1}(0,0) + F_{t+1}(0,1) + F_{t+1}(1,0) + F_{t+1}(1,1)$, or more compactly can be written in the vector form $\boldsymbol{F_{t+1}}(G,T) = [F_{t+1}(0,0), F_{t+1}(0,1), F_{t+1}(1,0), F_{t+1}(1,1)]^T$, with:
$F_{t+1}(0,0) = \tau_0 F_t(0,0) + a_1 F_t(0,1)$; $F_{t+1}(1,0) = \tau_0 F_t(0,0) + \alpha_1 F_t(0,1) + \tau_0 F_t(1,0) + a_2 F_t(1,1)$;
$F_{t+1}(0,1) = b_1 F_t(0,0) + b_r F_t(0,1)$; $F_{t+1}(1,1) = b_2 F_t(1,0) + b_r F_t(1,1)$.
Hence the population growth model can be written in the matrix form as

$$\begin{bmatrix} F_{t+1}(0,0) \\ F_{t+1}(0,1) \\ F_{t+1}(1,0) \\ F_{t+1}(1,1) \end{bmatrix} = \begin{bmatrix} \tau_0 & a_1 & 0 & 0 \\ b_1 & b_r & 0 & 0 \\ \tau_0 & \alpha_1 & \tau_0 & a_2 \\ 0 & 0 & b_2 & b_r \end{bmatrix} \begin{bmatrix} F_t(0,0) \\ F_t(0,1) \\ F_t(1,0) \\ F_t(1,1) \end{bmatrix}, \tag{1}$$

which can also be written in a more compact form $\boldsymbol{F_{t+1}(G,T) = AF_t(G,T)}$. The model is easy to solve iteratively to give the solution for the next *n* year is, i.e. $\boldsymbol{F_{t+n}}(G,T) = \boldsymbol{A^n F_t}(G,T)$.

In this paper we pose a problem on how to harvest the population optimally. If $Q_{1\text{k}}, Q_{2\text{k}}, Q_{3\text{k}},$ and $Q_{4\text{k}}$ kis the numbers of individual form sub population $F_k(0,0)$, $F_k(0,1)$, $F_k(1,0)$, and $F_k(1,1)$, respectively, then the problem is to maximize the profit Π along the time horizon *T*, with

$$\Pi = \sum_{k=0}^{T} (P_1 Q_{1k} + P_2 Q_{2k} + P_3 Q_{3k} + P_4 Q_{4k}) \rho^k,$$

and p_i is the associated unit price of the sub population *i* in the profit function above. The discounting factor is represented by *ρ*. We use genetic algorithm approach [3] to devise the solution of the problem.

2. Genetic Algorithm

Genetic algorithms (GA) was found by John Holland and developed by David Golberg. GA is numerical optimization algorithms inspired by both natural selection and natural genetics [3]. It is regarded as a heuristic approach. Some advantages of genetic algorithm compared to other optimization methods are: the direct, parallel, and powerful technique which can be used to optimize continuous or discrete variables, free derivative information, simultaneously searches from a large domain of objective function, deals with a large number of variables, optimizes variables with extremely complex objective function, provides several optimum solutions instead of one and so on [1,4,5]. This heuristic approach is applicable to many fields of optimization case, which makes GA attractive for many researchers [6]. Following is the basic algorithm of GA as suggested by Golberg in flowchart of Figure 2.

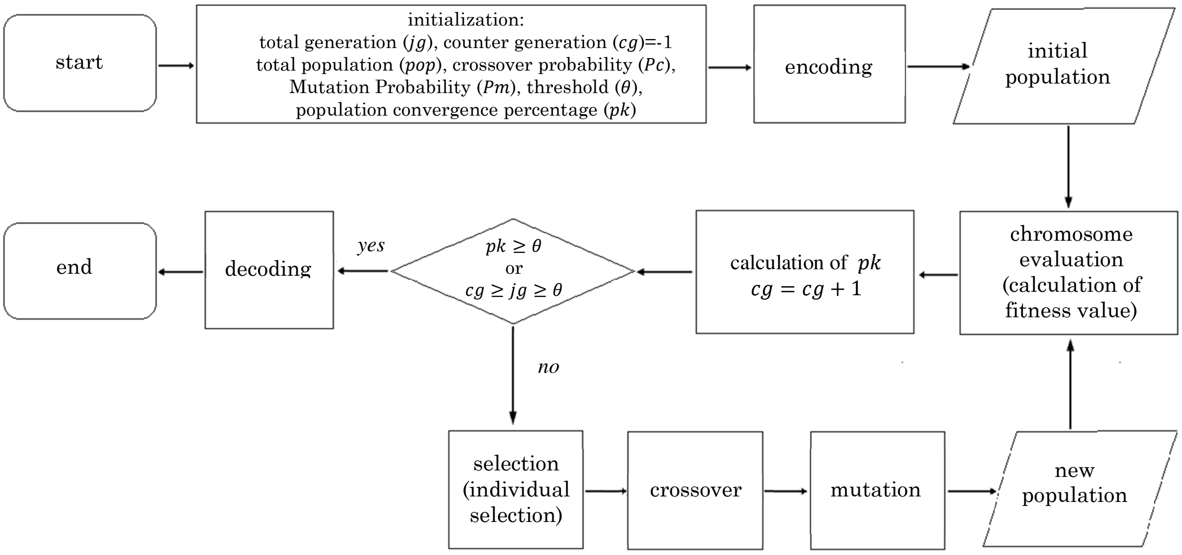
start
initialization:
total generation (jg), counter generation (cg)=-1
total population (pop), crossover probability (Pc),
Mutation Probability (Pm), threshold (θ),
population convergence percentage (pk)
encoding
initial population
end
decoding
yes
$pk \geq \theta$
or
$cg \geq jg \geq \theta$
calculation of pk
$cg = cg + 1$
chromosome evaluation (calculation of fitness value)
no
selection (individual selection)
crossover
mutation
new population

Figure 2. Genetics Algorithm flowchart developed by David Golberg

3. Genetic Algorithm implemented in Backcross Breeding Matrix

As explained earlier, in this research, our attention is directed to the population growth model given in [1]. In [1] it is explained that

a_i= Upgrading birth rate with $i = 0,1,2,\dots,n$
α_i= Non-Upgrading birth rate $i = 0,1,2,\dots,n$
b_i= Survival rate with $i = 0,1,2,\dots,n-1$
b_r= Survival rate of the last class
τ_0= Birth rate of the 1st age class

Let $\rho = \frac{1}{(1+r)}$, where r is the rate of return. Suppose that

$$\begin{gathered} Q_{ik} = 0,1,2,3,\dots \\ Q_{1k} \leq F_k(0,0) \\ Q_{2k} \leq F_k(0,1) \\ Q_{3k} \leq F_k(1,0) \\ Q_{4k} \leq F_k(1,1) \end{gathered} \tag{2}$$

then, our consideration is to solving the following problem,

$$\max\left(\Pi = \textstyle\sum_{k=0}^{T}(P_1Q_{1k} + P_2Q_{2k} + P_3Q_{3k} + P_4Q_{4k})\rho^k\right) \tag{3}$$

subject to:

$$\begin{bmatrix} F_{t+1}(0,0) \\ F_{t+1}(0,1) \\ F_{t+1}(1,0) \\ F_{t+1}(1,1) \end{bmatrix} = \begin{bmatrix} \tau_0 & a_1 & 0 & 0 \\ b_1 & b_r & 0 & 0 \\ \tau_0 & \alpha_1 & \tau_0 & a_2 \\ 0 & 0 & b_2 & b_r \end{bmatrix} \begin{bmatrix} F_t(0,0) - Q_{1t} \\ F_t(0,1) - Q_{2t} \\ F_t(1,0) - Q_{3t} \\ F_t(1,1) - Q_{4t} \end{bmatrix}. \tag{4}$$

Our purpose is to solve the maximization problem (3) with constrained $\boldsymbol{AF_t(G,T)}$ as in (4) using Genetic Algorithm method. We built the GA routine using C++ Program. Following are some remark for the GA program construction.

- The objective function is the function $\boldsymbol{\Pi}$ with maximization problem
- The calculation of fitness for maximization is using $eval(v_k) = \boldsymbol{\Pi}, k = 1,2, \dots N$; *N*= Number of population size. We take *N*=10 for this case
- The total fitness calculated by: $TF = \sum_{k=1}^{N} eval(v_k), k = 1,2,..,N$
- The cumulative probability q_k for each chromosome v_k calculated by $q_k = \sum_{k=1}^{N} p_k, k = 1,2,..,N$.

Generally, the algorithm genetics processes include the following important steps: encoding, selection, crossover, and mutation ([8]). The details of the steps are explained in detail in this section.

Pre-step 1: Initialization
We begin the process with initialization as follows:

Number of generation $jg = 1000$
Counter generation $cg = -1$
Number of Population $pop = 10$
Crossover Probability $pc = 0.25$
Number of mutation per generation $mut = 4$
Threshold $\theta = 90\%$

Pre-step 2: Encoding
The objective is to find Q_{11}, Q_{12} which maximize of $\boldsymbol{\Pi}$ in the interval solution

$$0 = a_1, a_2 \le Q_{21}, Q_{42} \le b_1, b_2 = 0.$$

In this research we use binary encoding method to represent chromosome and the gen length (denoted by m_j). We calculate m_j ($j = 1,2$) as

$$2^{m_j - 1} < (b_j - a_j) * 2^k < 2^{m_j} - 1.$$

To convert the binary to decimal for variable Q_{11} and Q_{12}, we use formula

$$Q_{1j} = a_j + decimal(substring_j) * \frac{(b_j - a_j)}{2^{m_j - 1}}. \quad (5)$$

For this case we have $m_1 = m_2 = 4$, so that we will have length of chromosome or number of gen $m = m_1 + m_2 = 8$.

Step 1: Initial Population
The random process is done to obtain 10 initial populations. This obtained 10 random initial population are in the form of binary number with length 8. The decimal value of each initial population is calculated by substituting the decimal of each substring to formula (5)

Step 2: Chromosome Evaluation
The fitness value of chromosome, in our case is maximization problem, is the value of objective function Π. The fitness value of each chromosome is evaluated in this step, in order to see the best value of Π (maximum value).

Step 3: Calculation of Population Convergence Percentage
The Population Convergence Percentage (pk) defined as percentage of number of individual with most number of same fitness value. The formula given as follows

$$pk = \frac{n}{pop} * 100\%$$

where n is the most number of same fitness value and pop is the number of individual. The counter generation is calculated by formula,

$$cg = cg + 1.$$

Step 4: Termination Criteria
The genetic algorithm process will be terminated when the counter generation cg reach the defined number of generation ($jg = 1000$) or the convergence of population pk reach the defined threshold level ($\theta = 90\%$). After initialization of population, the value of $cg = 0$, and if there is no individual with same fitness number, $pk = 10\%$ so that the genetics algorithm process is continue.

Step 5: Chromosome Selection
Since the maximum point which are try to be found then the fitness value used are defined by

$$eval(v_k) = \Pi(Q_{11}, Q_{12})$$

then calculate the total fitness of the population

$$TF = \sum_{k=1}^{10} eval(v_k)$$

and calculate the probability selection p_i for each chromosome v_k by formula

$$p_k = \frac{eval(v_k)}{TF}.$$

Then, calculate the cumulative probability

$$q_k = \sum_{i=1}^{k} p_i.$$

Select a set of 10 chromosome using roulette wheel for 10 times, and get a random numbers $r \in [0,1]$. We apply the following criteria to addresses the position to replace the selected $k-th$ current individual,

$$if \begin{cases} q_{k-1} < r \leq q_k, & select \;\; v_k \\ r < q_1, & select \;\; v_1 \end{cases}.$$

Step 6: Crossover
The crossover will be done to the new population, in this paper, the Single Point Crossover (SPX) method are used. SPX is the most simple crossover method where a random number are generated in the interval [1,m] as an intersection point. According to [7] the SPX algorithm is as follows:

- Generate a number randomly in interval [1,m], and use this number as the starting position of a gen in a chromosome which will encounter crossover. Mark alleles which will get crossover (from the starting position until the last gen).
- Exchange the marking allele between parent 1 and 2.

Step 7: Mutation
Number of gen in one generation is $m * N = 8 * 10 = 80$. If we want to arrange the mutation probability value equal to pm, then we should have calculate number of mutation (nm) as

$$nm = m * N * pm$$

In this simulation, we use $pm = 0.05$, then, in the mutation process we generate randomly $nm = 4$ number of integer in interval $[1, pop * m]$. This 4 number is used as locating in which gen will be mutated. The replacement process of allele is 1 and 0, as follows

$$allele \begin{cases} 0 \to 1 \\ 1 \to 0 \end{cases}.$$

After the mutation process then the calculation in one generation is finished. The next process is to repeat the process in step 2 until the termination criteria is satisfied.

Step 8: Decoding
Decoding is the process to recode the gens in a chromosome so that the value is return into the value (before encoded).

4. Numerical Results

In this section we give an example of the GA implementation discussed above. We assume that the parameters used in the example are a_1=0.56, a_2 =0.244, α_1= 0.44, b_1=0.945, b_2=0.975, b_r= 0.875, τ_0= 0, $P_1 = 650{,}000$ currency unit, $P_2 = 2{,}600{,}000$ currency unit, $P_3 = 1{,}800{,}000$ currency unit, $P_4 = 6{,}000{,}000$ currency unit, $r = 0.1$, $\rho = \frac{1}{1+r} = 0.9090$, and there is no cost associated in the model.

We run the implemented GA algorithm in C++ for 10 trials with different set of initial population of solutions. We only consider a short time horizon to go, i.e. $T = 2$. We do 10 trials just to illustrate the method. The results of one of the trial is presented in Table 1. The colums of Table 1 represent 10 different individual of initial solution. The rows represent the updated solution due to the GA algorithm. For example, initially we have a goat population of $\begin{bmatrix} F_0(0,0) \\ F_0(0,1) \\ F_0(1,0) \\ F_0(1,1) \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \\ 10 \\ 0 \end{bmatrix}$ at time t=0. The population grows and at time t=1 (by ceiling each vector component) we have $\begin{bmatrix} F_1(0,0) \\ F_1(0,1) \\ F_1(1,0) \\ F_1(1,1) \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 0 \\ 10 \end{bmatrix}$. According to Table 1, row 8 (the most updated solution resulting from GA algorithm) shows that at time t=1 we should take $\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 7 \end{bmatrix}$ out from $\begin{bmatrix} F_1(0,0) \\ F_1(0,1) \\ F_1(1,0) \\ F_1(1,1) \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 0 \\ 10 \end{bmatrix}$ and sell them. The remaining unsold goat population then grows and at the final time t=2 we have $\begin{bmatrix} F_2(0,0) \\ F_2(0,1) \\ F_2(1,0) \\ F_2(1,1) \end{bmatrix} = \begin{bmatrix} 6 \\ 9 \\ 6 \\ 3 \end{bmatrix}$ which is automatically taken as the solution, i.e. $\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} = \begin{bmatrix} 6 \\ 9 \\ 6 \\ 3 \end{bmatrix}$. The total revenue from selling these population is 84,545,454 currency unit. Figure 3 shows the plot for the total revenue from selling the goat population according to the GA results for consecutive generations, associated with the consecutive rows in Table 1, for all 10 different initial solutions. The final generation shows that at least 90% of the evolving solutions has already convergent to the same solution. Table 2 shows the results for the whole 10 trials done in the experiment. The table reveals that the GA gives different set of solutions but relatively close to each other in terms of the resulting total revenue, except a few.

Table 1. Numerical results by Genetics Algorithm of Trial 7

cg	Individual										
		1	2	3	4	5	6	7	8	9	10
1.	Q_{i1} Q_{i2} i=1,..,4	(0,0,0,0), (6,9,7,9)	(0,9,0,4), (1,1,2,6)	(0,0,0,1), (6,9,7,8)	(0,8,0,9), (2,2,2,1)	(0,10,0,0), (0,0,3,9)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,0), (6,9,7,9)	(0,1,0,8), (6,8,5,2)	(0,6,0,6), (3,4,3,4)
2.		(0,1,0,8), (6,8,5,2)	(0,0,0,7), (6,9,6,3)	(0,1,0,0), (6,8,7,9)	(0,0,0,8), (6,9,5,2)	(0,0,0,1), (6,9,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)
3.		(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,8), (6,9,5,2)	(0,0,0,8), (6,9,5,2)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,8), (6,9,5,2)	(0,0,0,7), (6,9,6,3)

4.		(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,10), (6,9,5,0)	(0,0,0,7), (6,9,6,3)	(0,0,0,6), (6,9,6,4)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,1), (6,9,7,8)	(0,0,0,1), (6,9,7,8)
5.		(0,1,0,1), (6,8,7,8)	(0,0,0,0), (6,9,7,9)	(0,0,0,1), (6,9,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,1), (6,9,7,8)	(0,0,0,1), (6,9,7,8)	(0,0,0,1), (6,9,7,8)
6.		(0,1,0,1), (6,8,7,8)	(0,0,0,1), (6,9,7,8)	(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,0,0,7), (6,9,6,3)	(0,1,0,1), (6,8,7,8)	(0,0,0,1), (6,9,7,8)	(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)
7.		(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,1,0,0), (6,8,7,9)	(0,1,0,1), (6,8,7,8)	(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,1), (6,9,7,8)	(0,1,0,1), (6,8,7,8)	(0,0,0,7), (6,9,6,3)
8.		(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,7), (6,9,6,3)	(0,0,0,6), (6,9,6,4)	(0,0,0,7), (6,9,6,3)

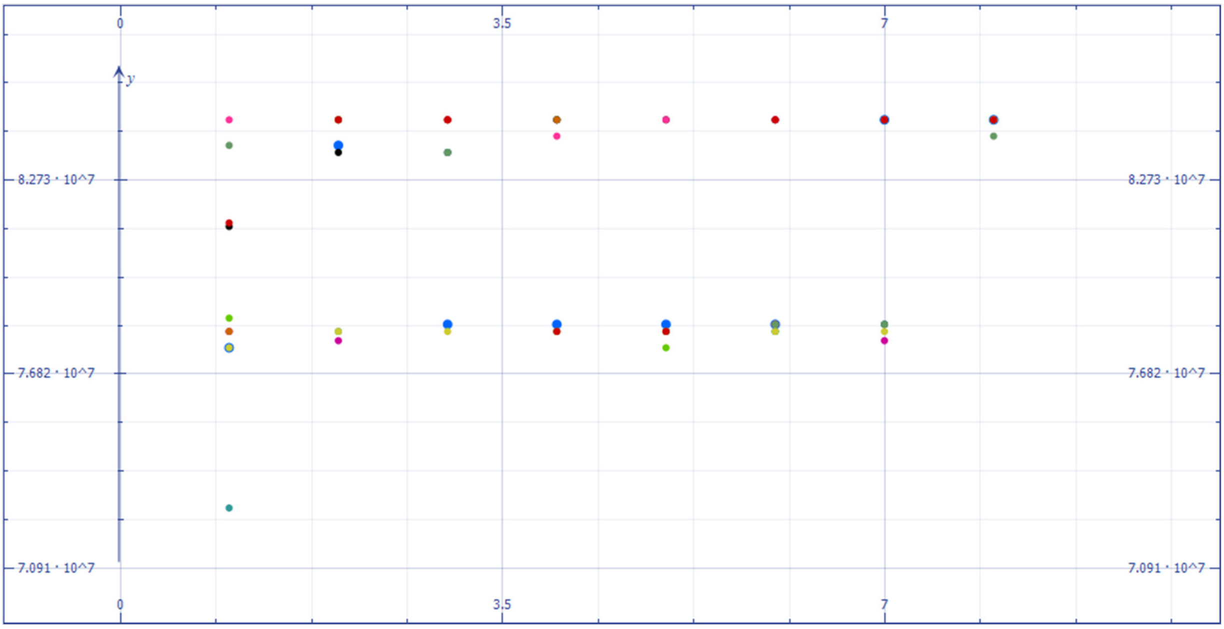

Figure 3. Plot of Numerical values by Genetics Algorithm of Trial 7

Table 2. Numerical results by Genetics Algorithm of 10 Trials

Trial	cg	Q_{i1}	Q_{i2}	$\Pi(Q_{ij})$
1.	9	(0,0,0,4)	(6,9,6,6)	83,057,851
2.	3	(0,1,0,1)	(6,8,7,8)	78,314,049
3.	21	(0,5,0,8)	(3,5,3,2)	82,190,082
4.	10	(0,0,0,4)	(6,9,6,6)	83,057,851
5.	21	(0,0,0,1)	(6,9,7,8)	78,099,173
6.	21	(0,5,0,5)	(3,5,4,5)	82,190,082
7.	7	(0,0,0,7)	(6,9,6,3)	84,545,454
8.	13	(0,6,0,8)	(3,4,3,2)	82,404,958
9.	11	(0,0,0,5)	(6,9,6,5)	83,553,719
10.	12	(0,2,0,2)	(5,7,6,7)	77,000,000

5. Conclusion

In this paper we have discussed a mathematical model of goat breeding strategy, i.e. the backcrossing breeding. The model is aimed to obtain a strategy in producing better variant of species. In this strategy, a female (doe) of a lesser quality goat, in terms of goat quality is bred with a male (buck) of an exotic goat which has a better goat quality. In this paper we posed a problem on how to harvest the population optimally. If $Q_{1\mathrm{k}}, Q_{2\mathrm{k}}, Q_{3\mathrm{k}},$ and $Q_{4\mathrm{k}}$ is the numbers of individual form sub population

$F_k(0,0)$, $F_k(0,1)$, $F_k(1,0)$, and $F_k(1,1)$, respectively, then the problem is to maximize the profit Π along the time horizon *T*. A genetic algorithm approach has been devised to obtain the solution of the problem. The trials we did in the experiment reveals that the GA gives different set of solutions but relatively close to each other in terms of the resulting total revenue, except a few. Further study need to be done to obtain GA solution that closer to the exact solution.

Acknowledgment
The authors thank to the Universitas Padjadjaran, due to the funding of this work through the scheme of Academic Leadership Grant (ALG) with contract number 855/UN6.3.1/PL/2017.

References

[1] Raihan A U, Supriatna A K and Anggriani N 2016 *Pros. Sem. Nasional Matematika Unpar 2016* MS pp 86-92

[2] Supriatna A K, Anggriani N, Carnia E and Raihan A 2017 *AIP Conference Proceedings* 1868, 040012 (doi: 10.1063/1.4995127)

[3] Haupt R L and Haupt S E 2004 *Practical Genetic Algorithms (2nd ed.)* (New Jersey: A John Wiley & Sons, Inc., Publication)

[4] Bimlendu S, Sujata D, Abhinav M, Vinay K, Prasanna K C 2016 A Review over Genetic Algorithm and Application of Wireless Network System *Procedia Computer Science* **78** pp 431-438

[5] Wagner F S, Claudio M N A, Pereira, Pius P M S, Roberto S 2002 *Genetic algorithms applied to turbine extraction optimization of a pressurized-water reactor.* Applied Energy **773** pp 217-222

[6] Kramer O 2017 *Studies in Computational Intelligence Volume 679: Genetic Algorithm Essensials* (Oldenburg: Springer)

[7] Weise T 2009 *Global Optimization Algorithms Theory and Application* (http://www.it-weise.de/, downloaded 10/10/2017)

[8] Adriana F and Fadly R L 2015 *Algoritma Genetika dan Penerapannya* (Yogyakarta: Teknosain)