https://doi.org/10.1007/11875581_47

Thinking Capability of Saplings Growing Up Algorithm

Ali Karci and Bilal Alatas

Firat University, Department of Computer Engineering, 23119, Elazig / Turkey
{akarci, balatas}@firat.edu.tr

Abstract. Saplings Growing up Algorithm (SGA) is a novel computational intelligence method inspired by sowing and growing up of saplings. This method contains two phases: Sowing Phase and Growing up Phase. Uniformed sowing sampling is aim to scatter evenly in the feasible solution space. Growing up phase contains three operators: *mating, branching, and vaccinating* operator. In this study thinking capability of SGA has been defined and it has been demonstrated that sapling population generated initially has diversity. The similarity of population concludes the interaction of saplings and at consequent, they will be similar. Furthermore, the operators used in the algorithm uses similarity and hence the population has the convergence property.

1 Introduction

The thinking is a social activity, and human culture and cognition are aspects of a single process [1, 2]. People learn from one another not only facts but methods for processing those facts. If knowledge and skills spread from person to person, the population converges on optimal processes. The social activities in a population can be categorized in three levels.

- Individuals learn locally from their neighbors. People are aware of interacting with their neighbors, gleaning insights from them, and sharing their own insights in turn, and local social learning is an easily measured and well-documented phenomenon.
- The spread of knowledge through social learning results in emergent group-level processes. This sociological, economic, or political level of phenomenon is seen as regularities in beliefs, attitudes, behaviors, and other attributes across individuals within a population. A society is self-organized system with global properties that cannot be predicted from the properties of the individuals who make up.
- Culture optimizes cognition. Though all interactions are local, insights and innovations are transported by culture from the originator to distant individuals; further, combination of various innovations results in even more improved methods. This global effect is largely transparent to actors in the system who benefit from it.

The probability of human interaction is a function of the similarity of two individuals: The basic idea is that agents who are similar to each other are likely to interact and then become even more similar.

E. Corchado et al. (Eds.): IDEAL 2006, LNCS 4224, pp. 386 – 393, 2006.

Similarity is a precondition for social interaction and subsequent exchange of cultural features. The probability of interaction depends on similarity and culture is seen to spread and finally stabilize through links between similar individuals. Individuals become more similar as they interact; populations do not converge on unanimity.

The effect of similarity as a casual influence in Axelrod's model is to introduce polarization. Dissimilarity generates boundaries between cultural regions. Inter-individual similarities do not facilitate convergence, but rather, when individuals contain no matching features, the probability of interaction is defined as 0.0, and cultural differences become insurmountable.

In this study, we defined the thinking capability of saplings growing up algorithm (SGA) [3-4]. In order to introduce the thinking skill of algorithm, we used a simple encoding (binary encoding) for sake of understandability and simplicity without lose of generality.

The second part of this paper introduces the Saplings Growing up Algorithm (SGA). Third section describes the thinking skill of algorithm and finally, the last section concludes the paper.

2 Saplings Growing Up Algorithm (SGA)

Solution space can be considered as a garden of saplings, and hence all saplings must be scattered in the garden uniformly (Fig. 1). Each sapling is a potential solution, unless there is multi-criteria problem. In the multi-criteria case, all saplings are solutions. If a farmer wants to sow saplings, he will trivially sow them in equi-length distance for the sake of growing up of saplings more quickly (Fig. 1). In order to solve a problem by simulating the growing up of saplings, arbitrary solutions to be generated initially must be scattered evenly in the feasible searching space. In order to scatter saplings in the garden, the uniform population method in genetic algorithms for generating initial population uniformly can be used [5-10]. Each sapling consists of branches, and initially each sapling contains no branch and it is a body. The algorithm for generation of initial population is seen in Algorithm 1.

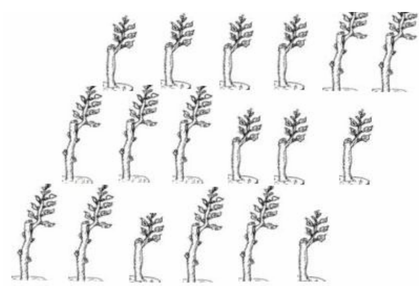

Fig. 1. Scattering saplings in garden uniformly

After being sowed, saplings must grow up (mating, branching and vaccinating). The aim of mating operator (denoted as ⊗) is to generate a new sapling from currently exist saplings by inter-changing genetic information. There will be a mating factor for each pair of saplings, since the distance between a pair is the most important factor which causes the mating of pair or not.

```
   Algorithm 1. SowingSaplings
   // P is population, I is indices set and Ie is the enlarged indices set.
1.  Create two saplings such as one of them P[1] contains all upper bounds for
    variables as branches and the other P[2] contains all lower bounds for
    variables as branches.
2.  Index←3
3.  k←2
4.  While P is not saturated do
    Let ie be an element of Ie and each ie are enlarged with bit value and this
bit value corresponds to part.
   i←1
   While P is not saturated and all saplings are not generated for a specific
value of k (and i≤2^k-2) do
      i is a k-bit number and ie corresponds to the enlarged value of i. Each bit
of i is enlarged upto length of corresponding part of P[0] and P[1].
      For j←1 to n do
      If j^th bit of ie is 1 then j^th branch of P[Index] is equal to P[1]*r
      else  j^th branch of P[Index] is equal to P[2]*r
         r is a random number in interval [0,1] and it is a real number.
      Index←Index+1
     i←i+1
   k←k+1
```

Let $G=g_1g_2\ldots g_i\ldots g_n$ and $H=h_1h_2\ldots h_i\ldots h_n$ be two saplings. The distance between G and H affects the mating process' taking place or not, and it depends on the distance between current pair. Let P(G,H) be probability of not mating of saplings G and H, and $P_m(G,H)$ is mating probability of saplings G and H.

$$P(G,H)=\frac{\left(\sum_{i=1}^{n}(g_i-h_i)^2\right)^{1/2}}{R} \tag{1}$$

$$R=\left(\sum_{i=1}^{n}(u_i-l_i)^2\right)^{1/2} \tag{2}$$

u_i is the upper bound for the corresponding distance between the pair of currently selected saplings, and l_i is the lower bound for the corresponding distance between the pair of currently selected saplings. The probability of mating of two saplings depends on the distance between both saplings. G and H are saplings and the probability of their mating is

$$P_m(G,H)=1-\frac{\left(\sum_{i=1}^{n}(g_i-h_i)^2\right)^{1/2}}{R} \tag{3}$$

Wind and other effects in the nature affect the mating probability. With the mating operator, a sapling gets a branch from the mating partner or sends its branch to mating partner and thus, $G\otimes H$ may yield 2n new saplings.

The mating process takes place for each pair of branches (g_i and h_i), if $P_m(G,H)$ satisfies the mating condition. The mating condition is $P_m(G,H)$ and this is the mating rate for G and H. A random number is generated and if this random number is smaller than or equal to this mating rate, then these saplings are mated.

Algorithm 2. Mating(G,H)

1. j←1, …, n
2. compute $P_m(G,H)=1-\dfrac{\left(\sum_{j=1}^{n}(g_j-h_j)^2\right)^{1/2}}{R}$
3. i←1,…,n
4. if $P_m(G,H)\geq random[0,1)$ then
5. $G\leftarrow G-g_i$, and $H\leftarrow H-h_i$
6. $G\leftarrow G+h_i$, and $H\leftarrow H+g_i$ // $G\leftarrow G+h_i$, and h_i is added to position of g_i, and $H\leftarrow H+g_i$ and g_i is added to position of h_i,

In order to grow up a branch on any point on the body of sapling, there should be no near branch previously occurred there. Assume that a first branch was occurred at point **1** as seen in Fig. 2, the probability of branch occurring on the point **2** is less than the probability of branch occurring on the points **3**. This logic can be used as a method for searching solution locally. This is a local change on the current solution(s).

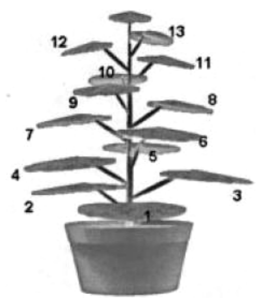

Fig. 2. Effects of the point where a branch trying to grow up

There is a branch growing up at point **1**. The probability of a branch growing up at point different from **1** is proportional to $1-\dfrac{1}{d^2}$ where d is the distance between that point and point **1**. The probability of a branch growing up at point **2** is $1-\dfrac{1}{d_1^2}$, where d_1 is the distance between and point **1** and **2**. The probability of a branch growing up at point **3** is $1-\dfrac{1}{(d_1+d_2)^2}$ if d_2 is the distance between point **2** and **3.**

Let $G=g_1g_2\ldots g_i\ldots g_n$ be a sapling. If a branch occurs in point g_i (the value of g_i is changed), then the probability of a branch occurring in point g_j could be calculated in two ways: linear and non-linear.

The distance between g_i and g_j can be considered as $|j\text{-}i|$ or $|i\text{-}j|$. If g_i is a branch, then the probability of g_j being a branch is $P(g_j \mid g_i) = 1 - \frac{1}{(|j-i|)^2}, \quad i \neq j$ in linear case, and $P(g_j|g_i)$ is similar to conditional probability, however, it is not pure conditional probability. In the non-linear case, the probability can be considered as $P(g_j \mid g_i) = 1 - \frac{1}{e^{(|j-i|)^2}}$. If $i=j$, then $P(g_j|g_i)=0$.

Algorithm 3. Branching(G)

$i \leftarrow 1, \ldots, n$
    $j \leftarrow i+1, \ldots, n$
        if there is no branch then
            $P(g_j|g_i)=1$ and branching process is applied
        else
            $P(g_j \mid g_i) = 1 - \frac{1}{(|j-i|)^2}, \quad i \neq j$ or $P(g_j \mid g_i) = 1 - \frac{1}{e^{(|j-i|)^2}}$
            if $P(g_j|g_i) \geq \text{random}[0,1)$ then
                g_j will be a branch

The vaccinating process takes place between two different saplings in case of similarity of saplings. Since the similarity of saplings affects the success of vaccinating process, and also vaccinating success is proportional to the similarity of both saplings. In this study, the similarity of saplings is computed in two ways. $G=g_1g_2\ldots g_i\ldots g_n$ and $H=h_1h_2\ldots h_i\ldots h_n$ for $1 \leq i \leq n$, $g_i, h_i \in \{0,1\}$.

$$\text{Sim(G,H)}=\sum_{i=1}^{n} g_i \oplus h_i$$

The vaccinating process takes place as follow, if Sim(G,H)≥ threshold.

$$G' = \begin{cases} g_i & \text{if } g_i = h_i \\ \text{random}(1) & \text{if } g_i \neq h_i \end{cases} \quad \text{and} \quad H' = \begin{cases} h_i & \text{if } h_i = g_i \\ \text{random}(1) & \text{if } h_i \neq g_i \end{cases}$$

where G' and H' are obtained as consequence of applying vaccinating process to G and H. Saplings are not vaccinated arbitrarily. The saplings to be vaccinated must satisfy the inequality defined by the similarity (Sim(G,H)≥ threshold). The initial value of threshold depends on the problem solvers. The smaller value of threshold results in more accurate solution, and the bigger value of threshold results in more non-accurate solution.

Algorithm 4. Vaccinating(G,H)

$i \leftarrow 1, \ldots, n$
    $\text{Sim(G,H)}=\sum_{i=1}^{n} g_i \oplus h_i$
    if $\text{Sim(G,H)} \geq r$ then
        $G' = \begin{cases} g_i & \text{if } g_i = h_i \\ \text{random}(1) & \text{if } g_i \neq h_i \end{cases}$ and $H' = \begin{cases} h_i & \text{if } h_i = g_i \\ \text{random}(1) & \text{if } h_i \neq g_i \end{cases}$

where r is a threshold value defined by problem solver. random(1) generates a random number which is 0 or 1.

In order to determine the quality of saplings, in contrast to genetic algorithm objective function is used. The objective function measures the goodness of the saplings in the population space and there is no necessity to take objective function scores and processes them to produce a number for each sapling.

3 Thinking Capability

Assume that the length of a sapling is n (a sapling has n branches), and the initial population contains m saplings. In order to demonstrate the thinking capability of SGA, we must firstly see the structure of initial population. The amount of knowledge in the initial population and its type must be known.

Algorithm 1 generates the initial population and this population has the following knowledge. Let S_1 and S_2 be two saplings.

$S_1=\boxed{s_1 s_2 \ldots\ldots s_n}$ $\qquad$ $S_2=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_n}$

These two saplings are deterministically generated initially. Then the remaining saplings are generated with respect to the rules in Algorithm 1 up to population is completed.

k=1

$S_3=\boxed{s_1 s_2 \ldots\ldots s_{\lfloor \frac{n}{2} \rfloor} \bar{s}_{\lfloor \frac{n}{2} \rfloor+1} \ldots\ldots \bar{s}_n}$ $\qquad$ $S_4=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_{\lfloor \frac{n}{2} \rfloor} s_{\lfloor \frac{n}{2} \rfloor+1} \ldots\ldots s_n}$

k=2

$S_5=\boxed{s_1 s_2 \ldots\ldots s_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots s_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots \bar{s}_n}$ $\qquad$ $S_6=\boxed{s_1 s_2 \ldots\ldots s_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots s_n}$

$S_7=\boxed{s_1 s_2 \ldots\ldots s_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots \bar{s}_n}$ $\qquad$ $S_8=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots s_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots s_n}$

$S_9=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots s_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots \bar{s}_n}$ $\qquad$ $S_{10}=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \ldots\ldots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \ldots\ldots s_n}$

For any branch s_i, $1 \le i \le n$, the number of 1s and number of 0s are equal to each other. This case is valid for all branches.

Theorem: The probability of similarity of population is greater than or equal to 0.5.

Proof: The similarity of population means that $\forall S_i, S_j$, $1 \le i,j \le m$, $i \ne j$, $Sim(S_i,S_j)>0$. In order to prove this theorem, the knowledge contained by the initial population must be determined. $Sim(S_i,S_j)=n$ means that S_i and S_j saplings are not similar and their similarity is zero, since they do not have branches that have same values.

Initially generated saplings

$S_1=\boxed{s_1 s_2 \ldots\ldots s_n}$

$S_2=\boxed{\bar{s}_1 \bar{s}_2 \ldots\ldots \bar{s}_n}$

$Sim(S_1,S_2)=n$

Generated saplings for k=1

$$S_3 = s_1 s_2 \cdots\cdots s_{\lfloor \frac{n}{2} \rfloor} \bar{s}_{\lfloor \frac{n}{2} \rfloor+1} \cdots\cdots \bar{s}_n$$

$$S_4 = \bar{s}_1 \bar{s}_2 \cdots\cdots \bar{s}_{\lfloor \frac{n}{2} \rfloor} s_{\lfloor \frac{n}{2} \rfloor+1} \cdots\cdots s_n$$

$$\mathrm{Sim}(S_3,S_4)=n$$

Generated saplings for k=2

$$S_5 = s_1 s_2 \cdots\cdots s_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots s_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots \bar{s}_n$$

$$S_6 = s_1 s_2 \cdots\cdots s_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots s_n$$

$$S_7 = s_1 s_2 \cdots\cdots s_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots \bar{s}_n$$

$$S_8 = \bar{s}_1 \bar{s}_2 \cdots\cdots \bar{s}_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots s_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots s_n$$

$$S_9 = \bar{s}_1 \bar{s}_2 \cdots\cdots \bar{s}_{\lfloor \frac{n}{3} \rfloor} s_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots s_{\lfloor \frac{2n}{3} \rfloor} \bar{s}_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots \bar{s}_n$$

$$S_{10} = \bar{s}_1 \bar{s}_2 \cdots\cdots \bar{s}_{\lfloor \frac{n}{3} \rfloor} \bar{s}_{\lfloor \frac{n}{3} \rfloor+1} \cdots\cdots \bar{s}_{\lfloor \frac{2n}{3} \rfloor} s_{\lfloor \frac{2n}{3} \rfloor+1} \cdots\cdots s_n$$

$$\mathrm{Sim}(S_5,S_{10})=n$$

$$\mathrm{Sim}(S_6,S_9)=n$$

$$\mathrm{Sim}(S_7,S_8)=n$$

This situation is conserved for all values of k. The number of disjoint saplings in the initial population is $m/2$ and there are $\binom{m}{2}$ pairs. So, there are $\binom{m}{2}$-$m/2$ saplings have branches whose values are equal to each others. Hence

$$\frac{\binom{m}{2}-\frac{m}{2}}{\binom{m}{2}}=\frac{m-2}{m-1}. \text{ For } m\geq 3,\ \frac{m-2}{m-1}\geq 0.5 \qquad \blacklozenge$$

The similarity in the population is used in mating and vaccinating steps of the SGA. Mating operator is a global search operator and uses similarity between saplings. Let S_0, S_1, S_2, and S_3 be the saplings and let them be encoded with binary strings as shown in Table 1. In growing up step, mating points are determined. If the mating points are 2, 3, 2, 1, 4, and 3; then new saplings 12 new sapling are created as shown in Table 2.

Table 1. Generated saplings

	Saplings
S_0	00000
S_1	11111
S_2	00011
S_3	11100

Table 2. Mating

	New candidate saplings	New candidate saplings
$S_0 \otimes S_1$	11000	00111
$S_0 \otimes S_2$	00000	00011
$S_0 \otimes S_3$	11000	00100
$S_1 \otimes S_2$	01111	10011
$S_1 \otimes S_3$	11101	11110
$S_2 \otimes S_3$	11111	00000

Vaccinating operator is a search operator and aims to generate new saplings from currently existing saplings which are similar. In vaccinating step, $\mathrm{Sim}(S_0,S_1)=0$;

Sim(S_0,S_2)=3; Sim(S_0,S_3)=2; Sim(S_1,S_2)=2; Sim(S_1,S_3)=3; Sim(S_2,S_3)=0. If threshold value is greater than 3, vaccinating process will not be performed. If it is less than 3, then S0 and S3; S1 and S2 are vaccinated. $S_0 \oplus S_3$={01100, 01100}, $S_1 \oplus S_2$={00011, 00111}.

Briefly, similarity is used in the SGA operators. When the algorithm continues, mating operator uses similarity measures and makes a global search. Vaccinating is also uses similarity and similar saplings are vaccinated. Competition and cooperation are observed among saplings.

4 Conclusions

The generated population with respect to SGA has disjoint saplings. However, the probability of similarity of population is greater than or equal to 0.5. This means that a similar population has diversity, and this is a desired case for obtaining better result. The similarity of population concludes the interaction of saplings and at consequent, they will be similar. Furthermore, the operators used in the algorithm uses similarity and hence, the population has the convergence property.

Acknowledgement

This study is supported by Turkish Scientific and Technical Research Council grant 105E144.

References

1. Axelrod, R.: The Evolution of Cooperation. New York: Basic Book (1984).
2. Axelrod, R.: The Dissemination of Culture: A Model with Local Convergence and Global Polarization. Journal of Conflict Resolution, 41 (1997) 203-226.
3. Karci, A., Alatas, B., Akin, E.: Sapling Growing up Algorithm (in Turkish). INISTA, Turkey (2006).
4. Karci, A., Alatas, B., Akin, E.: Evolutionary Algorithms and Imitating the Saplings (in Turkish). IKECCO 2006, Kyrgyzstan (2006).
5. Gundogan K. K., Alatas, B. Karci, A., Tatar, Y.: Comprehensible Classification Rule Mining with Two-Level Genetic Algorithm. 2nd FAE International Symposium, TRNC (2002) 373-377.
6. Alatas, B., Karci, A.: Genetik Sürecin Düzenlilik Operatoruyle Global Cozume Dogru Harekete Zorlanmasi. ELECO'2002, 364-368, Turkey (2002) 18-22.
7. Karci, A., Arslan, A.: Uniform population in genetic algorithms. I.U. Journal of Electrical & Electronics, 2 (2) (2002) 495-504.
8. Karci, A., Cinar, A.: Comparison of Uniform Distributed Initial Population Method and Random Initial Population Method in Genetic Search. The 15th International Symposium on Computer and Information Sciences, Istanbul / Turkey (2000) 159-166.
9. Karci, A.: Novelty in the Generation of Initial Population for Genetic Algorithms. Lecture Notes in Artificial Intelligence, 3214 (2004) 268-275.
10. Karci, A., Arslan, A.: Bidirectional evolutionary heuristic for the minimum vertex-cover problem. Journal of Computers and Electrical Engineering, vol. 29 (2003) 111-120.