A Binary Particle Swarm Algorithm with A Simple Mutation Operator for Solving Phylogenies Problems

Reem Alsrraj
University of Information Technology and Communications
Baghdad, Iraq
reem.hasan@edu.univ-fcomte.fr

Abstract— Manipulating a huge amount of data sets makes the mechanism of searching very difficult to find the best features combinations in phylogenetic tree construction. This article has proposed a binary particle swarm optimization algorithm that randomly selects chloroplast genes to construct phylogenetic trees. Also, a simple mutation operator the article has suggested to speed up the search and facilitate the search process to find the best chloroplast genes combinations. These combinations can be evaluated according to the bootstrap values of all the nodes of the constructed tree.

Index Terms*—*Phylogenetic tree, Mutation, combinations.

I. INTRODUCTION

Feature selection in DNA sequences had become an important research tool in biological experiments. BPSO selects helpful DNA sequences of genes and exclude genes sequences that have a negative impact by omitting them. In phylogenetic tree construction, the choice of features is important to influence the obtained results. Many algorithms have been proposed for feature selection. Some of these algorithms depend on the mechanism of filtering each element separately depending on specific measures. The most helpful ones try to divide the elements to subsets [1] in order to reduce execution time, find the best subset of elements and obtain a high computational complexity. The Binary swarm optimization technique has been considered as one of the most well-known algorithms for feature selection. Especially when we have a large searching space. More than one swarm can be launched at the same time. Adaptive mutation operators have been proposed in many articles [2] and [3]. These operators make the swarm very capable of finding the optimal solution in local search. The objective of this article is to find the best phylogenetic tree with the largest and the best combinations of genes. Applying BPSO is not enough to find the intended results, so a mutation operator has been proposed in order to control the mechanism of local searching, where each element in the particle can make a difference. This article is established as follows: The MBPSO mechanism is presented in Section(2). The obtained results shown in Section(3), Also, there is a detailed comparison of the obtained results with other algorithms. Finally, this article ends with a conclusion section, in which the article is summarized.

II. METHODOLOGY

Suppose a population of N B, where N B is the number of particles in the swarm. Each particle is restricted by 4 information(position, velocity, score value and the local best position). The position and the velocity are represented by two arrays of length L, where L is the number of all genes. The position is represented by an array of binary numbers where 1 represents the included gene and 0 represents the not included genes. The score is the minimum bootstrap value [4] of the tree and it is the fitness value of the proposed phylogenetic tree. More precisely, each particle represented by a proposed phylogenetic tree of random selected elements (genes) and a fitness value (the measure of the confidence level of a tree). Each element has a position and velocity, if the fitness value is more than 90 and the number of the selected genes is more than or equal to 90% then this particle is considered as the best particle that the swarm ever had at the current time. Otherwise, the swarm will search till it finds the optima. First, we propose a swarm of a specific number of particles. Each particle is a phylogenetic tree constructed from a random number of genes for each gene there is a proposed velocity, the velocity value determines whether the gene will be selected or not in the next iteration. The velocity is controlled by several variables [5] and [6]:

$$\varphi = \varphi 1 + \varphi 2$$

$$\varphi 1 = c1r1$$

and

$$\varphi 2 = c2r2$$

According to these variables, the swarm can either make a rush and this will be considered as global search (we will get the same topologies with missing genes more than expected), or make a slow movement toward the objective which can give us more probabilities where each new probability can be a good solution (we will get the same topologies with more included genes or new topologies which increase the variety). By applying the traditional mechanism of the swarm we got many missing genes and this is inadequate in our case. So, a proposed parameter has taken a place in order to control the velocity of the element which is represented by zero in the array of positions of the particle. In order to increase the included genes in the proposed tree, we count the number of missing genes z for the particle, if they are more than 10% we choose a random number of these elements z with position zero to multiply their velocities by a random parameter r3in the interval of [0.1, 0.5]. We have determined the variable r3 with this interval to increase the probability of the appearance of ones ”included genes”. In order to notify the effect of this step, z should be in the interval [z/2, the total number of missing genes]. For instance, if the position of the proposed particle initialized to p[0110000000101011000000011]. The number of the missing genes is more than 10%. First, we will chose a random number in in the interval of [13, 18] such as 14. This number represent z in our proposed algorithm. Then, 14 cells from position array with zero contain will be chosen randomly. In order to multiply their velocities with r3(v0 *0.312,v3 * 0.211,v4 * 0.101, v5 * 0.122,v6 * 0.223,v7 * 0.211, v8 * 0.331,v9 * 0.412,v10 * 0.451,v11 * 0.102,v13 * 0.222,v18 * 0.335,v20 * 0.267,v23 * 0.477).
In the second step the positions of these elements will be updated according to the previous step and the position will be p[1111111111111111010100011]. While the remaining elements will keep their previous positions. The number of the included genes will be increasing obviously. The fitness function will be computed only one time by the iteration. We find that if we compute the fitness twice: before the mutation step and after it like many articles [2] and [3] the search process will take too long time, at the end we will choose the tree which has the largest subset of included elements.

Algorithm 1. **Initial Population**

```
$POP \leftarrow N$, $Maxiteration \leftarrow 10$
**for** each particle in POP **do**
    $p[position] \leftarrow [Randomint(0, 1)$ for each gene
    $p[velocity] \leftarrow [Random(0, 1)$ for each gene
    $p[score] \leftarrow 0$
    $p[best] \leftarrow p[position]$
**end for**
```

Our mechanism has been applied on Rosales family. This family has been analyzed and examined in many articles (for more information [7] and [8]). The Rosales family is constructed of 10 genomes according to according to [9] and [10]. These genomes contain ” 9 in-group species and 1 outgroup (Mollissima)”. Each genome of this family consists of common genes. The total number of all the genes in all the genomes = 82 which represents L (the length of the two arrays: position and velocity). Each genome represents a branch in the constructed tree. The essential idea is omitting a random number of genes from each branch with a specific
interval to notify their effect on the bootstrap value of the branch itself and its relation with the other branches (which can give a new topology). The minimum bootstrap value among all these branches is considered the fitness value which should be equal to or larger than 90. That means a well-supported tree where all its branches consist of the best subset of genes. Our proposed MBPSO algorithm illustrates the idea of our methodology: where POP represents the size of population, p[velocity] represents the velocity of the particle p at the current time, p[score] represents the fitness value of the article p. The fitness has been computed by using RAxml application (for more information [10] and [11]). p[best] represents the local best position of the particle. According to many articles [6], [5] and [13] Equation(1) is used to guarantied the convergence of the swarm. Where k is a random number in the interval [0.0,1.0].

Algorithm 2. **Proposed MBPSO Algorithm**

```
Calculate
    $$x = \frac{2k}{|2 - \phi - \sqrt{\phi(\phi - 4)}|} \quad (1)$$
while $Fitness < 90$ OR $it < Maxiteration$ do
  for each particle in pop do
    Calculate Fitness
    if $p[score] < Fitness$ then
      $p[score] \leftarrow Fitness$
      $p[best] \leftarrow p[position]$
    end if
  end for
  $Fitness \leftarrow$ score of the particle which has
   maximum fitness value
  $gbest \leftarrow$ position of the particle which has
   maximum fitness score
  for each particle in POP do
    Calculate particle velocity according to
    $$V_i(t+1) = x.[V_i(t) + \phi_1(P_{ibest} - X_i) + \phi_2(P_{gbest} - X_i)] \quad (2)$$
    Update particle position according to
    $$V'_{ij} = Sig(V_{ij}(t)) = \frac{1}{1 + e^{-V_{ij}(t)}} \quad (3)$$
    $$X_{ij}(t+1) = \begin{cases} 1, & \text{if } r_{ij} \leq Sig(V_{ij}(t+1)) \\ 0, & \text{otherwise} \end{cases} \quad (4)$$
    $missing \leftarrow Countthenumberofmissinggenes$
    if missing ¡ 10% from all genes then
      $z \leftarrow Choosearandomnumberofelementsofpositionzero$
      Multiply the velocity of these $z$ by r3
      Re Updating particle position according to Equa-
      tions(2) and (3)
    end if
  end for
end while
```

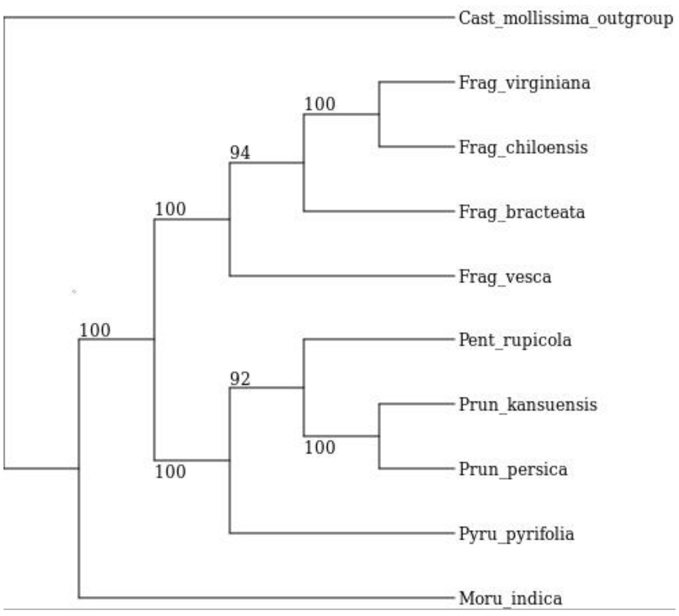
III. 11Results and Discussions

Our method namely MBPSO: mutated binary particle swarm optimization with φ1 = 2.5 and φ2 = 2.5 in the interval 0,1) has been executed 10 times for swarms of 3, 5 and 10 particles. The Results have been compared with the BPSO: traditional binary particle swarm optimization algorithm with the φ1 (c1 = 1 and r1 in the interval 0,1) and φ2 (c2 = 1 and r2 in the interval 0,1) as shown in Table(1) and another method DPSO: discrete particle swarm optimization method is described in [8] with the φ1 (c1 = 1 and r1 in the interval 0,1) and φ2 (c2 = 1 and r2 in the interval 0,1) and the proposed idea worked on minimizing the interval of r (random number in the interval of 0.0, 0.5) in Equation4 and a condition named b+p (where p represents the number of the included genes and b represents the minimum bootstrap value of the obtained tree). We noticed that there are many lost genes in the BPSO compared with MBPSO and DPSO. The best topologies obtained from MBPSO have less missing genes than that in DPSO. Moreover, we have got 13 topologies. Whereas, only 7 topologies have been obtained with DPSO. Tabel1 represents the results of the three methods. For each method 3 swarms have been lunched (3 particles, 5 particles and 10 particles) the column Particle represents the number of particles in the swarm, Min Bootstrap represents the minimum bootstrap value of the best obtained tree that the current swarm has reached and Missing Genes represents the omitted genes from the best obtained tree. Figure1 shows the best topology obtained by the MBPSO method with 92 minimum bootstrap value.

Table1. the results of the three Experiments

	Particle	MinBootstrap	MissingGenes
MBPSO	3	98	44
	5	86	3
	10	92	5
BPSO	3	82	37
	5	76	33
	10	82	33
DBPSO	3	73	4
	5	88	20
	10	92	19

Fig. 1.the best tree obtained



IV. Conclusion

A simple mutation operator for binary particle swarm optimization algorithm has been proposed in this article, which is redacted to solve the problem of finding the best largest subset of genes sequences. This approach has been applied to Rosales family. The proposed method has been compared with the traditional particle swarm optimization technique (BPSO) and a discrete particle swarm optimization technique (DPSO) for the well supported phylogenies in order to approve our MBPSO. 13 topologies have been obtained. The best topology was obtained with swarm of 10 particles where only 5 genes have been missed and the minimum bootstrap value was 92 (compared with other methods DPSO: 19 missing genes and BPSO:33 missing genes).

Acknowledgements

First of all, I would like to express my deep appreciation to, Professor Christophe Guyeux at the university of Franche-Comté, who inspired me to continue my research in this field. He provided the researcher with the data set Rosales.

REFERENCES

[1] MateuszAdamczyk.Parallel feature selection algorithm based on rough sets and particle swarm optimization. Computer Science and Information Systems, 2(10.15439/2014F389), 2014.

[2] Shengxiang Yang Changhe Li and Imtiaz Korejo. An adaptive mutation operator for particle swarm optimization. In-
formation Engineering, 2009. ICIE ’09. WASE, 2(10.1109/I-
CIE.2009.59).

[3] Tapas Sia Nanda Dulal Janaa and Jaya Silb. Particle swarm optimization with adaptive mutation in local best of particles. International Congress on Informatics, Environment, Energy and Applications-IEEA 2012, 38, 2012.

[4] SUSANHOLMESBRADLEY EFRON, ELIZABETH HALORAN. Bootstrap confidence levels for phylogenetic trees. Cross Mark, 93(14).

[5] James Kennedy Riccardo Poli and Tim Blackwell: P. Particle swarm optimization. Swarm Intelligence, 1(10.1007/s11721-007-0002-0).

[6] Siti Sophiyati Yuhaniz Dian Palupi Rini, Siti Mariyam Shamsuddin. Particle swarm optimization: Technique, system and challenges. International Journal of Computer Applications, 14(1).

[7] Christophe Guyeux Laurent Philippe Reem Alsrraj, Bassam AlKindy and Jean-FranÃğois Couchot. Well-supported phylogenies using largest subsets of core-genes by discrete particle swarm optimization. Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics, 2015.

[8] Bassam AlKindy, Christophe Guyeux, Jean-FranÃğois Couchot, Michel Salomon, Christian Parisod, and Jacques M. Bahi. Hybrid genetic algorithm and lasso test approach for inferring well supported phylogenetic trees based on subsets of chloroplastic core genes. International Conference on Algorithms for Computational Biology, AlCoB 2015.

[9] Bassam Alkindy, Jean-Fran¸ois Couchot, Christophe Guyeux,
Arnaud Mouly, Michel Salomon, and Jacques M. Bahi. Finding
the core-genes of chloroplasts. Journal of Bioscience, Bio-chemistery, and Bioinformatics, 4(5):357–364, 2014.

[10] Bassam Alkindy, Christophe Guyeux, Jean-Fran¸ois Couchot,
Michel Salomon, and Jacques Bahi. Gene similarity-based approaches for determining core-genes of chloroplasts. November
2014. Short paper.

[11]”Andries P.Engelbrecht”. ”Computational Intelligence”. ”WI-LEY”, ”www.wiley.com”, ”2007”

[12] Alexandros Stamatakis. The raxml v8.1.x manual. http://sco.hits.org/exelixis/resource/download/NewManual.pdf, 2014.

[13] Alexandros Stamatakis. The raxml 7.0. 4 manual. Department of Computer Science. Ludwig-Maximilians-Universit˙ta M¨nchen, 2008.