Ancestral Networks in Evolutionary Algorithms

Karthik Kuber, Stuart W. Card, Kishan G. Mehrotra, and Chilukuri K. Mohan
Dept. of Electrical Engineering and Computer Science, Syracuse University
Syracuse, NY, USA
{kkuber,swcard,mehrotra,mohan}@syr.edu

ABSTRACT

The behaviors of populations in evolutionary algorithms can be understood in terms of the dynamics of network models whose nodes represent individuals in the population. This paper explores "ancestral networks" in which connections indicate the proximity of the nearest common ancestor of two nodes. Preliminary experimental results show that the formation of large components in such an ancestral network model can be used to identify potential convergence, and to determine when randomly reseeding part of a population can prove beneficial.

Categories and Subject Descriptors: G.2.2 [Graph Theory]: Network problems; I.2.6[Learning]: Knowledge Acquisition; I.2.11 [Distributed Artificial Intelligence]: Intelligent agents

General Terms: Network Science Applications, Evolutionary Algorithms

Keywords: Genetic Algorithms, Network Science, Ancestral Networks, Convergence Detection, Reseeding

1. INTRODUCTION

Ideas borrowed from observing various biological species serve to study the growth of a species from its infancy to maturity as a community or communities. The natural behavior of most species in the biosphere is to form cohesive communities (or networks) for survival and growth, and a deeper understanding of such evolutionary systems can be greatly enhanced by the study of networks.

Genetic Algorithms [1] traditionally use strict convergence criteria based on properties of the best point or the mean of the entire population. We use the formation of a large component in the network space to detect impending convergence of the GA. The corresponding criterion in the networks domain is triggered much more quickly than population-based statistics, i.e., the formation of a large component is much more easily observed and quicker than measuring minute changes in best/mean function values. The largest component is likely to contain the best individual as well as many other points which are also significantly high quality individuals.

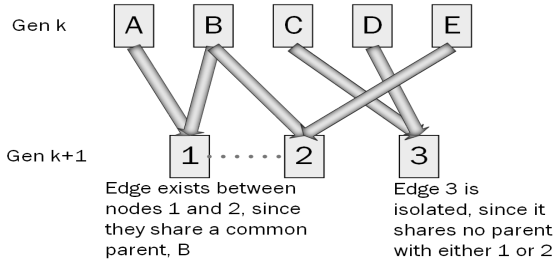

Figure 1: Edges in the next generation, based on parents in the previous generation

2. ANCESTRAL NETWORKS

We define an ancestral network, $G =< V, E >$ as follows:

- V: All n individuals in the population are vertices.
- E: Two individuals have an edge between them if they share at least one parent.

This is similar to the ancestral network defined in [4], which considers the network at multiple parent levels and studies genetic drift. Here, our focus is on convergence detection. One advantage of defining a network of this form is that the edges are "hard", i.e. we do not require any additional information or need to set thresholds (e.g., based on Euclidean networks [2]), since the ancestral connections are very clear and these are the processes which led to the formation of the next generation. An example of the existence of edges based on parents in the previous generation is shown in Figure 1.

The motivation to use a network such as this is that two nodes will share a link because they are likely to contain some common genetic material. Common genetic material would imply a form of "similarity" amongst individuals which we exploit by creating an edge and observing formations of components. Arguably, many "similar" individuals would imply the identification of a region of interest, which in the problem at hand would be the possible location of the global optimum. We show empirical evidence of the above hypotheses in the next sections.

3. RESEEDING ALGORITHM

We describe the algorithm below. This algorithm takes into account component formation, identification of the time instant when convergence is likely to occur, reseeding a fraction to continue exploring, and retaining the rest. We discuss the results of running this algorithm in the next section.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598481.

Algorithm 1 A Reseeding Policy Based on Early Detection of Convergence

```
Initialize the population of size $n$
$\alpha \leftarrow$ Threshold fraction of population in order to identify a large component;
repeat Perform recombinations and mutations to obtain next generation of GA
    if number of individuals in the largest component $\geq \alpha.n$ (i.e., a likely convergence zone) then
        Reseed a fraction, $\beta$, of the population in the search space and include them in the mating pool.
        Retain the remaining representative fraction, $(1 - \beta)$ of the component
    end if
until Convergence criteria are met.
```

4. RESULTS

We pick a well-known difficult benchmark problem, minimizing the Schwefel function [3], which has distant local optima. We tested our approach by experimenting with various population sizes, different dimensions (or number of variables), and different values of representative fractions to retain and reseed.

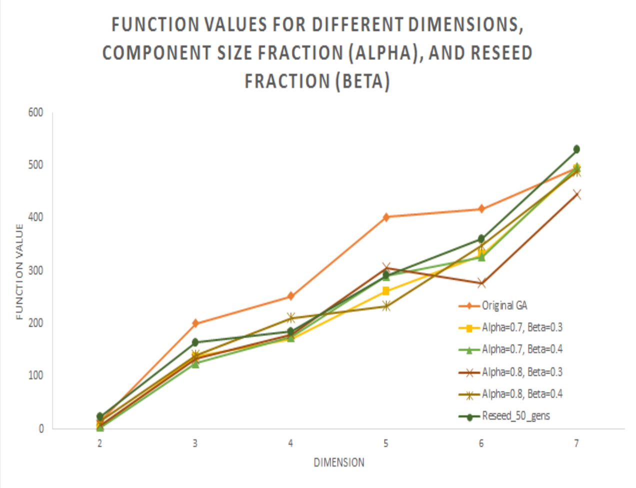

Figure 2: Function values for various dimensions, and parameters α and β, averaged over 30 runs.

We record and examine the results of running GA simulations at various generations and the results are shown in Table 1, depicting function values late in evolution, i.e. Generation 90, at around the point of convergence. A plot showing the above for a population size of 50 is also shown in Figure 2. Comparing the original GA performances against the modified versions as seen in Table 1 as well, the algorithm shows improvements in terms of better (lower) function values (since we are minimizing), with occasional exceptions. High values of standard deviation occur because there are significant numbers of optimally converged values as well as a significant number of premature convergences. In this scenario, a better mean implies having better function values as well as the fact that the algorithm got trapped fewer times (in most cases due to the reseeding).

In order to highlight the effects of reseeding based on emergence of components, we also reseed at predetermined instants (labeled 'Reseed_50_gens') at every 50th generation.

Table 1: Results: Function values for different dimensions and population sizes, at Generation 90. Best values in bold. Average of 30 runs.

Dim	Pop size	Original GA	Reseed_50 _gens	Alpha=0.7, Beta=0.3
2	50	15.43	23.34	**4.45**
3	50	200.06	163.75	**137.46**
4	50	251.94	184.94	**171.86**
5	50	402.59	291.1	**260.98**
6	50	417.84	360.65	**331.23**
7	50	496.65	529.72	**493.43**
2	100	**2.6E-05**	4.19	**2.6E-05**
3	100	113.62	77.18	**71.98**
4	100	117.02	106.19	**76.25**
5	100	142.46	**121.59**	131.03
6	100	225.24	194.39	**171.64**
7	100	291.71	**237.16**	278.23
2	150	**2.6E-05**	**2.6E-05**	**2.6E-05**
3	150	65.97	55.29	**45.14**
4	150	71.36	87.76	**46.32**
5	150	113.93	104.55	**82.25**
] 6	150	**94.24**	128.23	119.18
7	150	166.95	159.59	**133.77**

In a few instances, unmodified GA is better, arguably because the reseeding was too aggressive when a very good solution was already achieved.

From the point of view of time complexity, there is a little overhead due to the reseeds, when compared with the predetermined reseeding option. A single predetermined run for $n = 2$, population size$= 50$ took 1.63 seconds whereas with the component occurrence based reseeding took 1.80 seconds, for an overhead of 10.4% which is the time taken to perform the extra reseeds.

5. CONCLUSIONS AND FUTURE WORK

The proposed approach reduces the number of individuals during the exploitation phase to continue to explore, arguing that we do not really need *all* the individuals to hover close to each other and exploit, and that these resources are perhaps better utilized by continuing to explore potentially unexplored regions. This approach is especially helpful in harder problem surfaces, because more exploration usually implies a less likelihood of premature convergence. We also hypothesize that different kinds of similarities can result in identification of regions of interest in a similar way. We are also in the process of examining other kinds of EAs such as PSOs to study if they demonstrate similar behavior.

6. REFERENCES

[1] Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning. 1st ed. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

[2] Kuber, K., S. W. Card, K. G. Mehrotra, and C. K. Mohan. 2012. A Network Theoretic Analysis of Evolutionary Algorithms. In Swarm, Evolutionary, and Memetic Computing. Lecture Notes in Computer Science 7677, pp. 585-593. Springer Berlin Heidelberg.

[3] Schwefel, H. P. 1981. Numerical Optimization of Computer Models. John Wiley & Sons, Chichester; English translation of Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie (1977)

[4] Whigham, P. A. and G. Dick. 2008. Exploring the use of Ancestry as a Unified Network Model of Finite Population Evolution. Paper presented at Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence).