Visualization of evolutionary process in genetic programming

Chatchawan Wongsiriprasert , Prabhas Chongstitvatana, Somchai Prasitjutrakul
Department of Computer Engineering
Chulalongkorn University
prabhas@chula.ac.th

Abstract

Genetic programming (GP) is a process of finding solutions of a problem through the evolution of population of computer programs to solve that problem. As the pattern of searching for solutions in a problem landscape can be quite complicate, it is difficult to explain, in many cases, some phenomena that happened during the evolutionary process. This work describes an attempt to visualize that evolutionary process based on a case study of using GP to solve a problem in robot learning. There are many parameters in the system that affected its performance. We are especially interested in how the genetic operations affected individuals in the population. The result from "understanding" of the evolutionary process via visualization has been used to explain some phenomena in our GP experiments.

1. Introduction

Visualization of algorithm and data is a means to study how algorithms work by using graphical views and animations of the algorithms and data in action [3, 4]. This work describes a visualization technique used to understand many aspects of evolution process in solving the problem by genetic programming. Genetic programming (GP) is a process of finding solutions of a problem through the evolution of population of computer programs to solve that problem. For example our previous work [1] used GP to solve a robot learning task. A robotic hand-eye system learned to reach a target in an unknown cluttered environment by observing its own motions and self improved using GP method.

As the pattern of searching for solutions in a problem landscape can be quite complicate, it is difficult to explain, in many cases, some phenomena that happened during the evolutionary process. There are many parameters in the system that affected its performance. We are especially interested in how the genetic operations affected individuals in the population. An individual is represented by a tree structured of functions and terminals that composed an abstract computer program to solve the problem. The characteristics of an individual that we observed are : size, shape such as height and skewness, and type of operators such as sensing and actuation. These characteristics are important factors in determining the success of an individual measured by an objective function. The use of computer graphics to represent data and concepts from computer simulation is called data visualization. We used the data visualization techniques to view the evolution process. We observed the evolutionary process through various views by employing different presentations of different types of data, by the use of colours, and by animation.

We are interested in how various parameters in the GP process affects the *performance* of the system, for example, as measured by the *effort* which measured how *difficult* a problem is (defined in Koza [2]), or the *quality* of the solution as in the optimization problems. It is difficult to gauge the *linkage* effect of parameters in the system to the metrics of the result which makes it very difficult to tune the system to achieve desirable behaviours. There are a number of *standard* parameters in GP process such as : the maximum number of generation to run the evolution process, the number of individual in the population, the probability of applying crossover genetic operation, the probability of applying mutation. These parameters define the *run* of evolution process and give the overall view of the process but we are most interested in the *details* of each individual. These details are the shape and size of the successful individual and how each successful individual evolves from its parents. Understanding these details will give us a better idea how to improve the performance of the system and the quality of the solution. It should also enable us to explain some phenomena that happened during the evolutionary process.

2. The main idea

We propose to visualize how each individual evolves through generations in terms of measurements of some parameters. In general cases this will result in a 4-D presentation, a movie of 3-D graph where each time step is a generation and the 3-D graph shows the measure of the contribution of some parameters to each individual. In special case the 4-D presentation can be restricted to 3-D, a movie of 2-D graph through the generations where a 2-D graph shows the measure of some parameters of each individual. We will demonstrate this idea using a simple problem in GP in the next section.

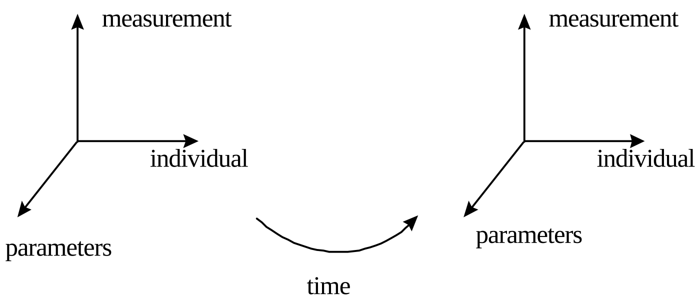


Figure 1 3-D graphs through generations

3. A case study of visualizing a GP process

A simple GP problem will be used to illustrate the proposed visualization method. The GP problem is to generate a program (sometime called *controller*) for a mobile robot to go to a specified destination in a cluttered environment. First, we discuss the simulated robot and its environment. The environment is a grid world of size 80 x 24 units. Each tile can either be occupied by an obstacle or is free space (see Fig. 2). The mobile robot can move forward by one unit or turns left or right which changes its heading by 90 degrees. The *program* of a robot is a language generated by this grammar :

```
tree := IF  cond true false | term
cond := AND tree tree | OR tree tree | NOT tree | term
true := tree
false := tree
term := sense | act
sense := N | NE | E | SE | S | SW | W | NW
act := FORWARD | LEFT | RIGHT
```

where `IF` is a 3-arity function, if the `cond` is true it will execute the `true` else the `false`. `AND, OR, NOT` are the usual logical connectives. The terminal set composes of sensing and action. Sensing operations will return TRUE when the destination is in the specified direction `{N, NE, E, SE, S, SW, W, NW}` else return FALSE. The action operations always return TRUE except when `FORWARD` caused the robot to hit an obstacle in which case the robot will not move and will return FALSE.

```
M



***************   *


                                       ***


                             **********   ***
                          *                 *
                             *           @  *
                             *              *
                             *              *
                             *   ************
                               *
                               *
                     ***********************
```

Figure 2 The simple GP problem, a mobile robot in a grid world. `M` is the robot, `*` is the obstacles, `@` is the destination.

Next, we describe the parameters in genetic programming runs. The fitness measure is based on the distance how close the robot to the destination in the end of run. Other parameters show in the table 1.

Table 1 The parameters in genetic programming run

population size	50
max. generation	100
fitness measure	distance from target
genetic operation	
reproduce	10%
crossover	85%
mutate	5%

3.1 Visualization of GP process

We visualize how genetic operators affect an individual. Each node in the tree representing an individual is tagged by an ID. We can track how an individual is changed by genetic operations such as a node may be deleted, duplicated, or newly generated. To visualize how individuals evolve, the final population is related to the initial population by the ID tags. Each ID tag will be considered as a *specie*. All the ID tags are created in the initial population. Recombination operations retain the ID tags from the parents. Mutation operations create new ID tags (a new *specie*). We measure the number of node from each specie in an individual as percentage of total number of node in that individual (Fig. 3). This will tell us what successful individuals inherited from their parents.

We also illustrate the measure of node type as a special case. The node is classified into three types : connective `{IF,AND,OR,NOT}`, sensing and action. At each generation this can be represented as 2-D graph of the number of node of each type in an individual (Fig. 4).

3.2 Observed results from visualization

We can observe the domination of some species in Fig 3. Initially each specie has equal contribution to the solution. Some species die out during the evolution process and some species start to dominate the population. In the final generation, the highly fit individuals are dominated by very few species. Figure 4 shows the effect of node types. Initially there are large variations in the population and it becomes *stabilized* through the later generation. In the final generation, it is as if the highly fit individuals are composed of the same number of node types. This effect is attributed to the dominant of a few species in the final population.

4. Conclusion

This work demonstrates how to use data visualization technique to help understanding evolutionary process in genetic programming. We propose to visualize how each individual evolves through generations in terms of measurement of some parameters. The result is a 4-D presentation, a movie of 3-D graph where each time step is a generation and the 3-D graph shows the measure of the contribution of some parameters to each individual. We illustrate this technique using a simple GP problem and observe some interesting phenomena in GP process in our experiments.

References

1. Chongstitvatana, P., Polvichai, J., “Learning a visual task by genetic programming”, *Proc. of IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems,* Osaka, Japan, 1996.
2. Koza, J., *Genetic Programming : On the programming of computers by means of natural selection,* MIT Press, 1992.
3. Brown, M., *Algorithm visualization*, MIT Press, 1988.
4. Rosenblum and others (eds), *Scientific Visualization*, IEEE and Academic Press, 1994.

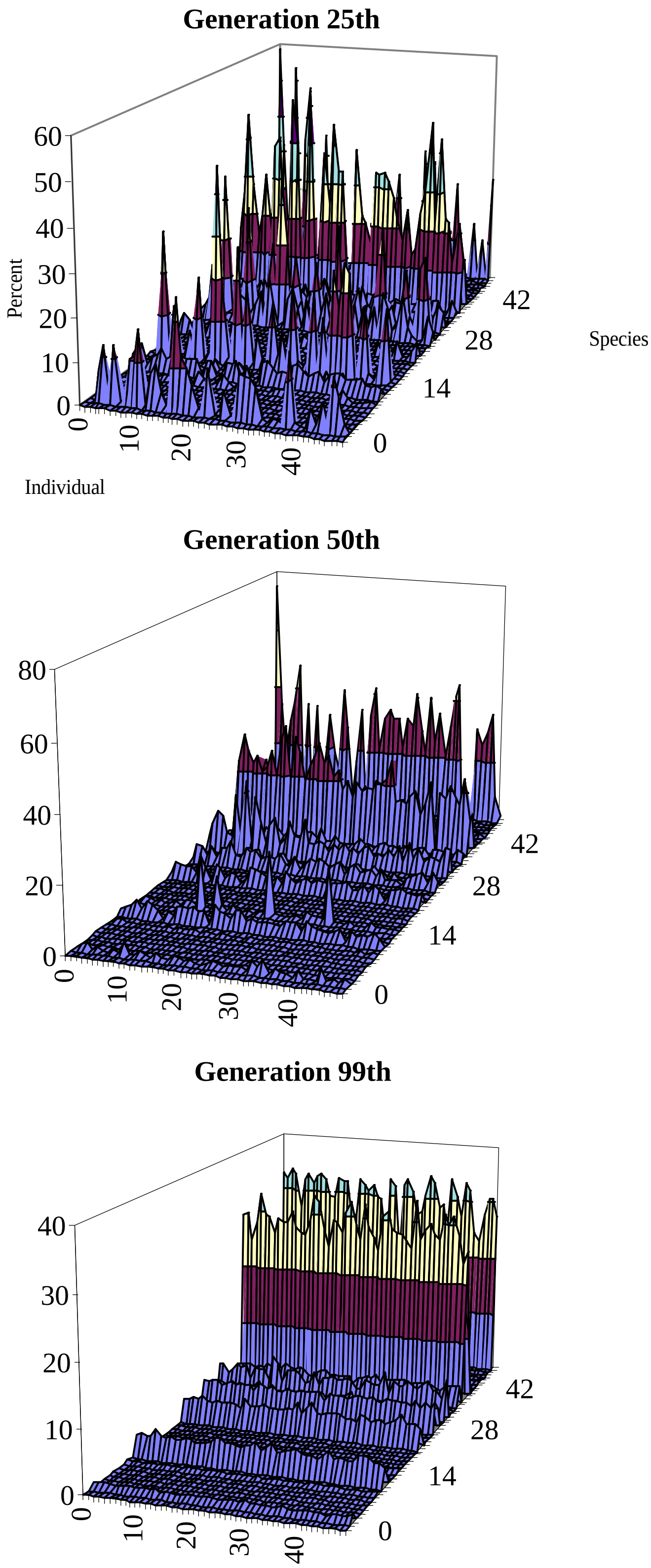
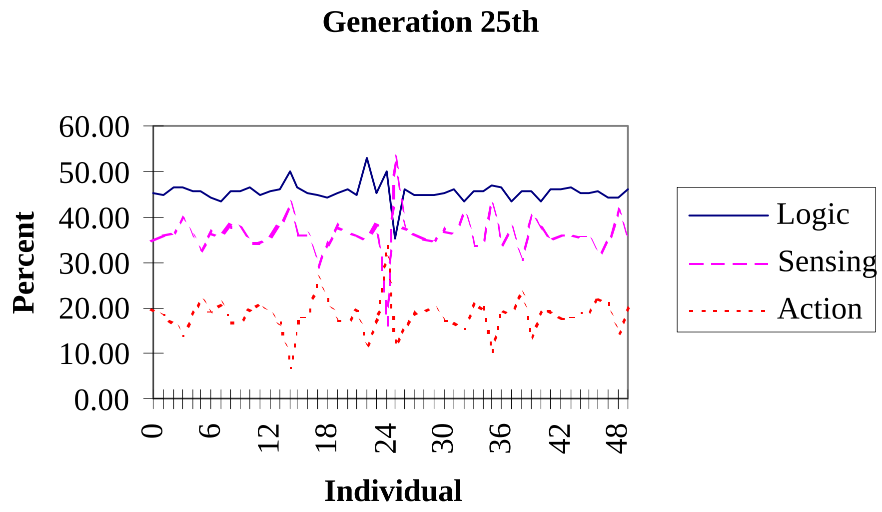
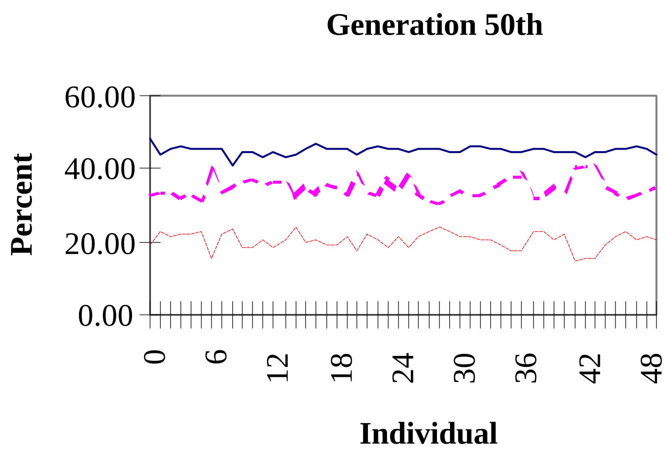


Figure 3. A movie of 3-D graph of the evolution of an individual parameterized by species





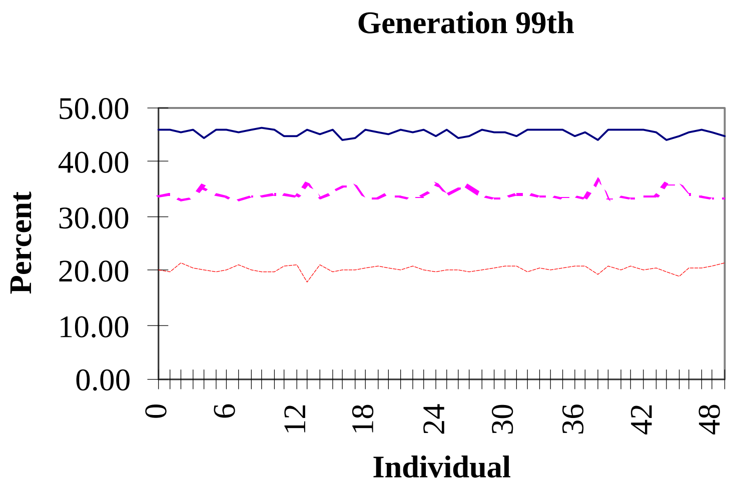


Figure 4. A movie of 2-D graph of the evolution of node type in population