Thesis Draft

Mads Anthony

April 15, 2016

Contents

1   Intro

2   Background

[11pt, a4paper]article amsmath [utf8]inputenc amsfonts amssymb tikz array cancel ulem
graphicx [a4paper]geometry gauss amsmath lastpage fancyhdr multirow listings   enu-
merate   Mads Anthony

3   Methods

In this section I will describe the methods that I've been using to evolve multitasking
agents. Since most of these methods is build atop other methods, I've decided to start
off with a description of the basics. I then use this as a foundation to describe the next
methods that is building upon that. This bottom-up approach is then repeated until
I've described all the methods used in this project.

3.1   Evolutionary Algorithms

Evolutionary Algorithms (EA) is a group of algorithms that are based on implementing
the principles of evolution. The general idea is to mimic real life, and let the algo-
rithm "discover" the bests solutions. This is done by having a population of solutions
be evaluated on a defined fitness parameter. We then let the fittest solutions survive,
and combine them with another fit solution to produce a new offspring solution. The
offspring then has a possibility to mutate and is placed in a new generation. We then
repeat the procedure on the new generation until we have a good enough solution.

An overview of the algorithm can be seen in figure (x) and the following will give a
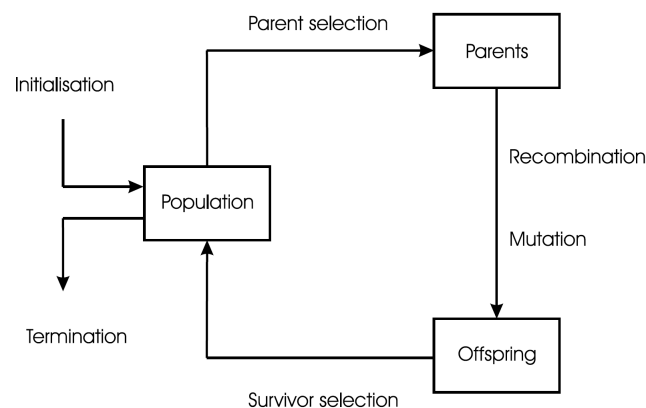more in-depth explanation of the components used to implement an EA.

Figure 1: [ref Eiben-Smith]

3.1.1   Initialization

To start using EA, we first need to have an initial population of individuals and decide how we want to represent these individuals. The representation of the individuals that the EA works with is called the genotype, and an individual that can be used on the actual problem domain is called the phenotype.

The phenotype space can be different from the genotype space, and choosing a genotype space is important, since it's here the evolutionary search takes place. The mapping from phenotype to genotype is called encoding and it is possible to have multiple genotypes that encode the same phenotype. The reverse is called decoding, however one genotype can only produce one phenotype.

With a representation chosen we can now initialise the first population. The initialisation of an EA is often kept simple, and the first population is therefore mostly generated randomly [p.23 Eiben Smith].

3.1.2   Evaluation and Parent Selection

With an initial population in place we now want to evaluate the whole population and choose which individuals will be used for parents. To do this, we introduce a fitness function. The fitness function is usually applied in the phenotype space (as it's here we want to find the best solution). The fitness of the genotype is therefore assigned by first decoding the genotype and then applying the fitness function.

Dependent on the problem being a minimisation or maximization problem we value the individuals with lowest or highest fitness values respectively. The idea for parent selection is to let the higher quality individuals be the base for the new generation. However we don't want to make the selection solely on the fitness, but instead use a probabilistic approach where individuals with higher fitness still gets favoured. This means that other "weaker" individuals still get a chance to get picked. This is done to avoid the search being greedy and getting stuck on local optimum [p.21 Eiben-Smith].

3.1.3   Recombination and Mutation

Recombination merges two genotypes into one or more genotypes, and the way parts from the parents gets chosen, is decided in a randomly fashion. It is possible to use more than two parents but is not common as doesn't have a biological equivalent [p.22 Eiben-Smith]. Using only one parent is called a-sexual reproduction and can produce a different offspring dependent on the variation operator [ref?].

With a new offspring created there is a chance that a mutation might happen. A mutation will change the offspring's genotype in random unbiased way. This is done to fill the gene pool with "fresh blood" [p. 21 Eiben-Smith].

3.1.4   Survivor Selection

Survivor selection is similiar to parent selection, but is not stochastic. Instead it selects in a deterministic way by choosing from the unified multi set of parents and offspring the one with highest fitness.

3.1.5   Termination

When to terminate the EA can be chosen in many different ways. Some examples include: set a limit of number of generation or check if the diversity in the population drops under a certain a threshold.

3.2   Neuroevolution

Neuroevolution is evolutionary algorithms applied on artificial neural networks (ANN). In the following section I will give an overview of how ANN works.

ANN is a model that is inspired by biological neural networks as observed in brain.

3.3   NEAT

3.4   HyperNEAT

3.5   Compositional Pattern Producing Network (CPPN)

3.6   Single Unit Pattern Generator (SUPG)