INF3490 exercise answers - week 11 2013

Important note: It is not expected answers on the exam to to be as verbose as the answers below. These answers are long to give a little extra explanation. On the exam you should usually answer with just one or two paragraphs, sometimes a single sentence may be enough. Also note that some exams questions may have multiple different ways of answering them that all show sufficient understanding of the topic to warrant a full score.

Problem 1a (2010 exam)

Typically, an evolutionary algorithm will have a progress generation-by-generation that is close to a geometric sequence, for example, the best solution might get approximately 5% closer to the optimum each generation. This means that the biggest improvements happens in the first generations, and that the improvement will converge to zero as the generations increase, making long runs of little benefit.

Likewise, intelligent initialization will often be a waste of time, since the evolutionary algorithm is often able to "invent" the same solutions in just a few generations. The exception here is really treacherous problems where intelligent initialization is needed to get the population distributed across the basins of attraction in a good way.

Problem A4 (2012 exam)

Generally the selection operators regulate the selection pressure, in effect controlling steering the evolution by controlling the dynamic between exploration (don't care much about fitness) and exploitation (focus heavily on picking and keeping fit solutions). In fact, since parent selection follows immediately after survival selection, you could in principle merge them, and use only one selection operator.

However, the task is usually (and perhaps mostly for traditional reasons) split into two phases with survival selection culling the population down to a manageable size, and parent selection chooses which solutions to base the next iteration of our search on, which can be practical because it splits up some complexity.

Problem 1e (2011 exam)

An overview of the four main classes of evolutionary algorithm:

Type	GA	ES	EP	GP
Representation	Any	$\mathbb{R}^n$	$\mathbb{R}^n$	Trees
Strategy parameters	No	Yes	Yes	No
Typical λ	$\lambda \leq \mu$	$\lambda \geq \mu$	$\lambda = \mu$	Any
Parent selection	Any	Random	One each	Any
Recombination	Any	Arithmetic	None	Swap
Mutation	Any	Gauss	Gauss	Replace
Survivor selection	Any	Best μ	Tournament	Any

$(\mu + \lambda)$ means both parents and offspring are included in the survival selection, while with (μ, λ) only offspring can survive.

Problem B2 (2012 exam)

The complexity class of the decision boundary is decided by the number of layers:

- One layer (output only) can form any hyperplane (linear discrimination)
- With two layers (one hidden layer) hyperplanes are formed by the nodes in the hidden layer, and these planes can be combined into any convex surface by the output
- With three layers (two hidden layers) convex surfaces are formed by the second hidden layer, and the output can create every possible surface using these (in the same sense that you can create every shape using infinitely many triangles)

The complexity of the surface will also rise with the number of neurons in each layer in the sense that the surfaces gain more detail. Each layer in a multilayer perceptron is fully connected to all nodes in the previous layer plus one bias value, so the number of weights can be calculated from the number of layers M and the number of neurons in each layer N_i:

$$|W| = \sum_{i=1}^{M} N_i \left(N_{i-1} + 1\right)$$

Problem B11 (2012 exam)

In order to focus more on exploration (by choosing actions at random often) at first, and then gradually switch to more exploitation (by increasingly choosing actions based on the expected reward).

Problem B12 (2012 exam)

Self-organizing maps reduce dimensionality by mapping the topology of the data to a lower-dimensional topology, usually a grid in two dimensions. This is done by taking the training data points one by one and updating the weights of the most similar node and those close to it be slightly close to that data point. Dimensionality reduction is useful because of several factors:

- Reduces time complexity: Less computation
- Reduces space complexity: Less parameters
- Saves the cost of acquiring irrelevant features
- Simpler models are more robust
- Easier to interpret; simpler explanation
- Data visualization (structure, groups, outliers, etc.) if plotted in 2 or 3 dimensions