ASS Symposia GA Talk 2: Multi-Objective Optimisation of the Atrophy Analysis Technique

Paul A. Bromiley

A Multi-Objective Genetic Algorithm

Last week I gave a simple introduction to GA optimisation. That talk covered the basic GA operators, the Fundamental Theorem of Genetic Algorithms, the problems of genetic drift and premature convergence, and a review of some diversity preservation methods. I finished with a brief description of multi-objective optimisation, where the aim is to find a surface of non-dominated solutions in cost space, known as the Pareto Front, and where the problems associated with diversity preservation become even more important.

Overview

This week I am going to describe an optimisation problem that Neil and I are working on as part of the structure and function grand challenge of the IRC: the optimisation of our existing technique for diagnosing dementing diseases by analysing the distribution of cerebral atrophy. In order to solve this problem we have developed a novel multi-objective GA. It includes some of the diversity preservation ideas I spoke about last week, such as preselection and restricted mating, together with some ideas I did not cover, including static extraction. It also incorporates self-adaptive mutation parameters in an effort to reduce the number of free parameters, thus making the algorithm more robust. I will present some results from testing the algorithm on simple multi-objective cost functions gathered from the literature, and finish by describing the current state of the project.

Atrophy Analysis: Introduction

We had an existing technique for diagnosing atrophic diseases of the brain from measurements of CSF volumes. This was based on the hypothesis that various atrophic diseases would result in specific regional patterns of atrophy, which would allow them to be differentiated. We could have tried to get accurate pixel-by-pixel atrophy maps, using non-rigid coregistration. However, we reasoned that it might be possible to avoid the difficulties of non-rigid coregistration by taking a much simpler approach. Atrophy must result in an increase in CSF volume, so by measuring CSF volume over a minimal number of regions in the head, taking account of head size and normal age-related atrophy, we could produce a minimal set of parameters that we could investigate for correlation with disease.

The technique starts by performing a rigid coregistration of a patient brain to a standard coordinate system, so that multiple patient brains can be compared. We then divide the CSF space in the patient brain into twelve equi-sized boxes, using horizontal and vertical dividing planes placed centrally, plus planes to divide the brain into thirds front middle and back.

Atrophy Analysis: CSF Segmentation

We identify CSF using a simple grey-level based segmentation. We use only T2 IR images, since if we plot a histogram of such an image, we see distinct peaks for grey matter, white matter and CSF, with CSF lying lowest. We fit the histogram with the fitting routines in TINA, assuming Gaussian distributions for pure tissues and triangular distributions convolved with Gaussians for partial volume pixels, giving us peak positions for each of the pure tissues. We can then determine a threshold at the 50% probability point for CSF, and threshold the images to produce binary CSF maps.

Atrophy Analysis: Masking

The CSF maps will also contain other fluid spaces, such as the eyes and sinuses. To delete these, we have drawn a set of binary masks in the standard brain space, shown here. The masks also define the lower extent of the CSF space, lined up with certain anatomical points in the brain. All other limits of the maps are simply defined as the outside of the head. Using the coregistration, we rotate these masks to the patient brain, and multiply the masks with the maps to delete the unwanted fluid spaces.

Atrophy Analysis: Age Correction

Next, we count the number of CSF pixels in each of the twelve boxes in the standard brain space. This produces a set of twelve variables. We also measure the extent of the CSF space along each axis of the space, producing a set of three numbers that we can multiply to produce a volume measurement. We divide each of the twelve variables by this volume to normalise for head size. We then need to correct for normal age-related atrophy, which results in about a 1% increase in CSF volume each year after the age of 40. We took a large group of normal patients, fitted curves to plots of each of the twelve volume-normalised variables against age, and then normalised the data to a fixed point on the curve specified by the average age of the patient group. This normalisation can then be applied to all patients in a data set containing diseases in order to remove the effects of age related atrophy. As part of the current project, we have acquired some young normals, so the age range of our normal patients is now 19 to 85. This shows a plot of CSF volume, relative to the total head size, against age for out normal group of 94 patients, for one of the twelve CSF volumes. All of the other volumes show the same trend: the fit here is with an exponential function.

Atrophy Analysis: Variable Reduction

Finally, we produce a set of five reduced variables describing relative degrees of atrophy between different regions of the brain. This reduces the dimensionality of the data to make further analysis simpler, and also removes to first order systematic errors at key points in the analysis, for example in the age correction.

Atrophy Analysis: Results

This plot shows two of these reduced variables, the relative amount of atrophy between the middle four and back four volumes against the relative amount of overall atrophy. Here we have a set of 55 patients, including normals shown in blue, Vascular Dementia in red, Fronto-Temporal Dementias in green and Alzheimer's disease in orange. You can see that even on this projection of the data we get good separation between the patient groups. These are the results from the original paper (Thacker et. al., 2002, Radiology224 p278-285). We can then determine how useful this technique is for disease diagnosis by applying a leave-one-out Parzen classifier, with a kernel size set to the measurement accuracy of the data, determined by repeat analysis of the data by two independent observers. This table gives the classifier result against the true diagnosis, again taken from the original publication on the technique. Overall, we can conclude that this relatively simple technique has significant diagnostic utility. One aspect of particular interest is the ability to discriminate between Alzheimers Disease and Fronto-Temporal Dementia. The diagnostic criteria for FTD are well established, but require input from a nueropsychological specialist, whereas this technique is completely automatic.

Motivation for GA Optimisation

The existing technique has been shown to have high diagnostic utility, but we could potentially do better. We are dividing the brain into twelve boxes, and the size of these boxes is of the same order of magnitude as the coarsest anatomical divisions in the brain, the lobes. However, we have made no attempt to align the boxes with any underlying anatomical features. We might expect that this would result in improved performance, since some diseases will preferentially affect one or more lobes. For example, Fronto-Temporal Dementia often results in degeneration of the frontal lobes. In short, there may be better positions for the dividing planes, and the problem becomes one of how to find them.

As a first step, we can avoid the use of hard coded boundaries and instead specify the volumes using a set of centre points. When counting, each CSF pixel can be assigned to its nearest centre point. This gives a more natural expression for the problem, as it is easier to define a set of points than a set of planes.

This is clearly an optimisation problem, with a few characteristics that we can easily see. Each dividing plane will be specified by at least two of the centre points: the points tessellate the space with Voronoi cells, and the dividing planes are the cell boundaries. The point pairs are not entirely independent: movement of one point will shift at least two planes, but there is not complete correlation, as movement of one point will not shift all planes. Therefore, we might expect that we can to some degree independently optimise different planes, and then combine these sub-solutions to generate better solutions. This is all starting to sound familiar: pairs of points form fairly good building blocks for a genetic algorithm. The GA has other features that are ideal for this problem. Since

we are using real data and not some idealised function, we can expect the cost function to be noisy, and GAs are robust to noise due to implicit parallelism: the fitness of each schema is estimated from multiple samples. We known nothing about the cost function: it almost certainly does not have well specified derivatives, due to noise; it may be discontinuous, and may feature a maze of local optima. So, we need a robust optimisation technique, insensitive to noise, which will find global minima in a problem that naturally divides into building blocks, and GAs are known to have all of these properties.

GA Goals

Given a set of points which specify regions for counting the CSF, we want to optimise the positions of those points to maximise the elements on the diagonal of the confusion matrix shown earlier. Now, we could simply sum the diagonal to produce a single cost function. However, the most natural statement of the problem is clearly multi-objective. Different sets of points may differentiate better between different diseases. For example, there may be one set of points that differentiates best between Alzheimers and FTD, and a different set that differentiate best between normals and Vascular Dementia. Therefore, we can consider each of the diagonal elements of the confusion matrix as a separate cost function. As I said last week, GAs are ideal for multi-objective optimisation, since the presence of a population of solutions allows the entire Pareto front to be captured in a single run, but this assumes that we solve the diversity preservation problem, so we need a GA specifically written to preserve diversity.

I listed some diversity preservation strategies last week. The majority of these methods required the introduction of additional arbitrary parameters. Crowding requires a crowding factor which sets the size of the subpopulation considered in the replacement step. Fitness scaling and fitness sharing require scaling or sharing functions. Optimal values for these parameters need to be determined experimentally, and, through an analogy with annealing schedules in simulated annealing, we might expect the optimal values to be problem specific.

Also, none of these methods really work. It has been conjectured that global convergence of the population to a single point will occur in any GA that allows interaction between any pair of solutions in the population, even when this interaction occurs rarely. The same phenomenon has been recorded in natural systems. Global convergence in nature is prevented by very strict mating restrictions: species never interbreed to produce viable offspring. In addition, complete convergence within a species is prevented by the dynamic nature of the cost function. When situations occur where a large number of individuals become genetically identical, the cost function changes. The effect can be seen in agriculture: modern farming techniques have led to huge monoculture fields, but this makes the population vulnerable to disease or parasites, and so requires the use of large amounts of pesticides.

So, what are the goals for our GA? We want convergence to occur locally, but not globally. We also want to minimise the number of arbitrary control parameters, in which we include population size, and mutation probability.

Multi-Objective Fitness Evaluation

In order to adapt a GA to multi-objective optimisation, we need to consider fitness evaluation in more detail. Early work in this area focused on non-Pareto approaches, such as Schaffer's vector evaluated genetic algorithm. This reduced to a linear weighting of the objectives to produce a single fitness value. Last week I stated that this would allow us to find a single point on the Pareto Front and, by extension, you might think that we could map the whole front by altering the weighting over multiple GA runs. However, a simple example will show that this will not allow us to map concave regions of the front. Taking a simple function to define a concave Pareto Front, a unit circle, and producing a cost f through a linear weighting of the two objective values x and y, we can look at the variation in cost across the front.

Multi-Objective Fitness Evaluation 2

Plotting the value of f against x for points across the Pareto Front, it is clear that the points on the ends of the front have the lowest costs for any linear weighting. Therefore, optimisation will cause solutions to converge to these points, and the concave region of the front will not be found.

Multi-Objective Fitness Evaluation 3

The failings of non-Pareto based approaches to multi-objective fitness evaluation led to research into Pareto based

approaches, initiated by Goldberg's Pareto ranking scheme. All individuals in the population were ranked according to dominance: non dominated individuals were assigned rank 1, then non-dominated solutions from the remainder of the population were assigned rank 2, and so on until the entire population had been ranked. Fitness proportional selection was then performed on the basis of rank. The important point to note is that the procedure is one step removed from the actual objective values of the solutions: fitness is evaluated purely on the basis of dominance.

The Larcombe GA: Reproduction

The GA that we have developed has significant departures from the simple three-operator GA formalism. As a starting point, we adopted the GA developed by Steve Larcombe in his PhD (supervised by Neil). This used an idea similar to Cavicchio's preselection, discussed last week. It is a steady state GA that avoids fitness proportional selection entirely. At each iteration, a single pair of solutions are selected at random, and subjected to crossover and mutation to produce two offspring. The offspring are then compared to their parents and replace them if their fitness is higher on a first-come, first-served basis.

Some GA researchers would argue that fitness proportional selection is an essential characteristic of a GA, and conversely our algorithm is not a GA since it lacks this feature. Also, we have to consider the effect that our selection routine would have on the applicability of the fundamental theorem to our algorithm. Last week I put up the equations

$$m(H, t+1) = m(H, t)\frac{f(H)}{\bar{f}}$$

and

$$m(H, t) = m(H, 0)(1 + c)$$

to describe schema growth in a three-operator GA under the influence of reproduction alone. Thus, we see that schema frequency grows exponentially in proportion to the ration of the schema fitness to the population average. However, this leads to genetic drift: errors in the operators, such as the variance of roulette wheel selection, can be amplified by this behaviour, and so most GAs don't in fact work like this. It is probably easiest to see with fitness scaling. We scale the fitness of strings such that

$$f'_{avg} = f_{avg}$$

and

$$f'_{max} = C_m f_{avg}$$

with typical values of $C_m = 2$ for population sizes of around 100. So, the maximum value for c is 2, and the growth of schemata is quadratic at best. It is simple to see that our technique has the same property: a given schema can at most double in frequency in any generation, since it can move from one parent to both children. Therefore, we get the same rate of schemata growth as a GA with fitness scaling, but without any of the overheads of roulette wheel selection.

Fitness and Direction Vectors

Some additional steps have to be implemented in order to compare multiple objectives simultaneously. Larcombe found that, if the requirement is that a child beats its parents on any single objective, oscillatory behaviour results. If a child is fitter than its parents on objective A but not B, then it is more likely that the next generation will be fitter on objective B but not A. Pairs of solutions can then continually replace each other without converging to the Pareto Front. To avoid this Larcombe added direction vectors to the solutions. These consisted of a bit string, initialised to zero, where each bit represented one objective. When replacement occurred, the bit representing the objective that had improved was set to 1. In order for this solution to be replaced, the same objective had to improve again. Other objectives might also improve, in which case their bits would also be set to 1. This prevented oscillatory behaviour.

Static Extraction

Introduction of the direction vectors produced another behaviour. Solutions would begin to converge to the Pareto Front, but many would become stuck at local minima and cease to converge. Larcombe used these "statics" to derive a new mechanism for diversity preservation. Each solution in the population is assigned an age, which is

increased each time the individual is selected to mate. If the children replace the parents, the child starts off with age 0. However, for statics the age can increase indefinitely. These static represent super-fit individuals, which will cause premature convergence if left in the population.

We can define an extraction age, at which the individual is removed from the population and stored in a secondary population, which will over time fill up with local and global optima. The gap left in the population can be filled by a new random solution: these new entries are known as random immigrants. The result is that, given time, the GA will exhaustively search the space. The use of static extraction means that the GA has access to an effectively infinite population, so the dependence on population size is removed. In addition, it splits the GA into two populations: the general population, which participates in reproduction, and the extracted set, which contains a record of all the best solutions found.

GA Coding and Mutation

Next, we had to consider the coding we would use and the mutation operator. Most researchers have used binary codings on the principle that you want to maximise the number of schemata available for the search. However, binary codings are prone to arbitrary orderings, and other researchers have found real-valued codings to be superior for some problems, on the principle that it is better to have building blocks that are relevant to the problem. The general conclusion is that the optimal coding is probably problem dependent.

In our atrophy analysis problem, we have natural building blocks in the form of pairs of centre points for the boxes. Therefore, we want to preserve this by using a real-valued coding. The atrophy analysis problem will involve the manipulation of six real-valued points. We have tested the GA on simple single and multi-objective cost functions, where the solution is represented by a single point. In each case, we have the problem of how to perform crossover and mutation. First mutation: we decided to use two parameters: a probability of mutation, and a maximal mutation distance between 0 and 1. When mutation is implemented, a single coordinate of a real valued point is selected for mutation, and multiplied by a random number between 0 and 1, and by the allowed range of the space in that direction. In order to minimise the number of free parameters in the search, we code up these parameters in a header region in each genome. We then allow the GA to modify these parameters, by adding an additional mutation probability and mutation distance for header values. This gives us a header of four numbers between 0 and 1. The headers are passed intact between genomes during crossover, and any of the four numbers can be mutated according to the header mutation probability in the normal way. On initialisation of the population, the numbers are set to some initialisation values typical of numbers used in the literature.

The result of this is that the GA can optimise its own mutation parameters. Schemes like this have been used in the past, for example by Caviccio (1970), although many previous schemes have implemented centralised control of the parameters. It is important to note that, in the absence of centralised control, the optimisation of mutation parameters is one step removed from the optimisation of the genome itself. A change to the genome directly imposes a change in cost. A change to the mutation parameters does not, but instead changes the probability that the genome will generate fit children. Therefore we might expect the mutation parameters to optimise more slowly than the position. We also hypothesise that the optimal mutation parameters will depend on the position in the search space. For example, close to a local optimum, a smaller mutation distance will be better, allowing search on a finer scale. Therefore, on addition of a random immigrant to the population, it picks up the header information from the nearest genome in the space. This allows localised optimisation of the header information.

Crossover

Next we have to consider the crossover operator. For our atrophy analysis problem, given that we have six real-valued numbers, we can just treat each one as a separate gene and implement the normal crossover operator. However, we also wanted to test the GA on more simple cost functions, where the position was represented by a single point. We could treat each coordinate as a single gene and implement regular crossover, but there is no reason to suppose that coordinates make good building blocks. Previous work on real-valued crossover has produced a range of operators. Simple methods, such as placing the children randomly on a line between the two parents, fail because they bias the search towards the middle of the search space. Ono et. al. used the two parent genomes, plus one other randomly-selected genome, to define a multi-dimensional Gaussian distribution in the search space, and then placed the two child genomes randomly according to that probability distribution. Their paper (GECCO '99) contains references to much of the work in this area. We adopted a relatively simple real-valued crossover operator found in the literature. Two parent genomes encode two points: imagine these as opposite vertices of a cube. We then place the child points randomly inside that cube, correcting where necessary

to hold the points inside the search space.

Initial Testing

To summarise, we have adapted Larcombe's GA, which implemented static extraction and random immigration, to use a real-valued representation, and added self-adaptive mutation parameters. This has reduced the number of free parameters to one: the extraction age. Larcombe's thesis contains some conjectures on how this parameter could be made self-adaptive.

The GA was written in such a way that it could operate either on single-objective or multi-objective cost functions, and during testing a popular single objective cost function from the literature, known as the six-hump camelback function (SHCF), was used. It has six local optima of which two are global, and the other four occur in two pairs: one with relatively low costs, similar to the global optima, and one with relatively high costs.

Initial Testing 2

The upper plot shows the positions of solutions in the extracted set of our GA after 25 generations on the SHCF. We have used an extraction age of 5 and a population size of 100. In order to facilitate comparison with generational GAs, I am using a generation to mean the number of iterations equal to half the population size, so that one population size number of solutions participate in mating during one generation. Both the the global optima, and both of the low cost local optima, have been found. The high cost local optima are not found, since so much of the space has lower costs that solutions in those positions do not become static.

The lower plot shows the 2D positions of solutions in the main population plotted against generation number. You can see that the two global optima are found relatively quickly, and remain in the population for some time. Both of the low cost local optima are also in there, but are difficult to see in this plot. However, one of the global optima is eventually eliminated. If the extracted population is managed carefully, we maintain both global solutions there, and so we could say that the algorithm has been successful. However, the elimination of one of the global minima shows that we are still getting genetic drift in the general population, caused by the influence of super fit individuals near the global minima. Even with static extraction and random immigration, the global minima are so fit that random immigrants are quickly eliminated, so, at any time after the two global minima are found, if one is lost we cannot recover it.

Diversity Preservation

In order to see what is happening, recall from last week's talk the idea that fitness scaling was developed to prevent two problems. At the start of the GA run, a few super-fit individuals will dominate the population, focusing the search towards themselves. At the end of the run, all individuals will have roughly the same fitness, so there is a lack of differentiation that prevents effective searching of the space. On multi-modal functions, this can result in a loss of all but one of the global optima through a gambler's ruin effect. Larcomb'e static extraction removes the first problem by removing the super-fit individuals, but does not cope with the second problem, and so global optima are still lost at the end of the run. Once lost, they cannot be re-found through searching by random immigrants, since solutions around the remaining global optimum act as super-fit individuals and quickly destroy the random immigrants.

In order to guarantee that we eventually find the global optimum, we need to prevent premature convergence indefinitely in the general population. To put this in a more concrete way, we want the algorithm to converge locally but search globally. The behaviour we are trying to achieve is for solutions in the general population to locally converge to optima, without effecting the general population globally. This local convergence will produce statics that can be removed to the extracted set. Random immigrants replace these statics and go on searching the space globally. The extracted set can be filtered so that it contains only the current set of non-dominated solutions. Therefore, the diametrically opposite behaviours of searching and convergence are effectively split over the two populations, and we do not have to tweak the parameters of the algorithm to achieve a problem-specific balance.

Voronoi Mating Restriction

We took the most direct route: if the problem is in mating between super-fit individuals and random immigrants, we have to stop these mating events, so we looked at mating restriction techniques. Now, we also want to avoid adding free parameters to define what we mean by local. Therefore, we use the natural definition of local imposed by the algorithm. You can consider the solutions tesellating the search space with Voronoi cells. Then, for any given cell, its Voronoi neighbours can be defined as the local solutions. We derived a relatively simple mating restriction. If the pairing AB is being considered, then it was compared to all other pairings AC. The bisecting planes for vectors between the pairings are necessarily Voronoi boundaries if the vectors do not pass through intermediate cells. Therefore, if the bisecting plane for AC intersects the vector AB at less than half its length, AB passes through an intermediate cell and must be rejected.

Using such a strict mating restriction means that the influence of super-fit individuals is limited to their immediate neighbourhood. They will still cause the local solutions to converge, so the search will be focused onto local optima. However, the combination with static extraction means that they will eventually be removed from the population, and random immigrants introduced, so a proportion of the population, set by the extraction age, will still search the space.

Influence of Mating Restriction

We can see this in action on the six-hump camelback function. Again we have the plot of the general population for 25 generations, with an extraction age of 5 and a population size of 100, with random pairing for mating, where one of the global optima disappears. Below we have the same problem with the mating restriction implemented. This time, both global optima are maintained in the population indefinitely. The search is focused around these points, but there are still points searching the rest of the space. We have achieved our goal of preserving diversity indefinitely in the general population.

Multi-Objective Testing

So we have produced an algorithm that indefinitely preserves diversity, and used the minimal number of free parameters. Time to test it on multi-objective test functions. Testing a GA on a single cost function is never enough. Given the infinite number of possible cost functions, it is theoretically possible to define one on which any particular GA beats all other algorithms. This is one statement of what is known in the field as the "No Free Lunch" theorem. Therefore, we want to define a set of test functions that exhibit a broad range of behaviours. Also, I didn't want to spend time re-implementing other peoples GAs for comparison, so I used a set of test functions and performance metrics from the literature. Fortunately, Van Veldhuizen collected together all the most popular multi-objective test functions and performance metrics, and tested them on the most common multi-objective GAs, in his thesis and subsequent papers, so I copied his evaluation methods and compared my results to his. His test set consisted of seven multi-objective test functions, exhibiting Pareto Fronts that were both connected and unconnected in both search and cost space, had both complex and concave regions, and had both single and multiple Pareto Fronts.

To define any meaningful test metrics, you need to know if the algorithm has found the right answer. Therefore, Van Veldhuizen performed exhaustive searches of the test problems on massively parallel computers. He used binary representations, and defined the coding such that the search space had 2^{24} points. Then he performed GA searches on each test function. He did not include any testing of the speed of convergence. Instead, he reasoned that what limits any optimisation project is the ratio of computer power to project deadline. Therefore, he terminated the GA when 2^{16} evaluations of the cost function had been performed, such that a maximum of 0.39% of the search space had been explored.

Two test metrics were defined: generational distance, or the average distance between the non-dominated front produced by the GA and the true Pareto Front; and overall non-dominated vector generation, or the number of non-dominated solutions found by the GA. Each GA was run ten times on each test problem, and the results averaged.

Multi-Objective Results 1-3

Here are plots of the Pareto Fronts in cost space found by the algorithm. In each case, we have successfully found all or most of the Pareto front for each problem. Visual comparison of the results with those in Van Veldhuizens papers shows they are similar in all cases, but it is more instructive to look at plots of the performance metrics.

Multi-Objective Results 4

Van Veldhuizen tested four MOGAs from the literature: MOGA (multi-objective genetic algorithm) (Fonseca and Fleming, 1998); MOMGA (multi-objective messy GA) (Van Veldhuizen and Lamont, 2000); NPGA (niched Pareto GA) (Horn et al, 1994); and NSGA (non-dominated sorting GA) (Srinivas and Deb, 1994). I won't discuss the details of each, except to note that they all incorporate fitness sharing in an attempt to avoid premature convergence, they were all developed to investigate key theoretical issues, and they are amongst the most cited MOGA methods, so probably represent a good cross section of the competition.

This shows Van Veldhuizen's results for generational distance, together with our results on the same scale. This metric measures the accuracy to which the Pareto Front has been found. Smaller is better here. Van Veldhuizen's MOMGA more or less outperforms the other algorithms he tested, and our algorithm is comparable with MOMGA over the test set as a whole.

ONVG shows much the same trend. These results show the total number of non-dominated vectors found, and again our results compare favourably with MOMGA. Higher is better here. Note also that, although we have no direct measure of the rate of convergence, ONVG does give an indirect measure, as for most of the test problems the Pareto Front does not become saturated with solutions. Van Veldhuizen reported some measurements of CPU time, but concluded that MOMGA had an unfair advantage because it was implemented as an executable, whereas some of the other GAs were implemented as Matlab code. He typically found CPU times of five minutes for MOMGA runs on these problems, which is roughly what our algorithm requires.

Conclusions

To conclude, we have developed a multi-objective GA that uses a real-valued representation and incorporates a novel combination of diversity preservation strategies. These include something similar to Cavicchio's preselection, Larcombe's static extraction, and a novel mating restriction based on the concept of Voronoi neighbours. We have compared this to other MOGAs popular in the literature, over a test set and using performance metrics defined in the literature, and found performance comparable to the best in the field.

Further Testing

It should be noticed that possible test problems could be divided into two classes. Those we have looked at are in some sense simple, as they involve solutions that represent a single point in a function space. Such problems do not naturally lend themselves to generation of building blocks. Better test problems would include the Travelling Salesman Problem, where given a set of cities, you have to find the shortest path that visits every city and returns to the starting point. In that case, building blocks are obvious, as it may be possible to optimise some portion of the path independent of the rest.

So, where is the atrophy analysis project at the moment? The GA is finished, and tested on simple functions. There is clearly scope for further testing on complex cost functions, and both theoretical analysis, to relate it back to the schema theorem, and practical analysis, to study aspects like the extraction age, and the adaptation of the mutation parameters, to explore in detail what the algorithm is doing.

Foe the main project, we have assembled a data set of 200 MRI volumes, spread across normals, Alzheimers, FTD, VAD, Schizophrenia and Lewy Body disease. These have been subjected to extensive quality control. The original software has been rewritten for fully automatic analysis, and adapted to form a GA cost function. The age range of the normal group has been extended downwards to 19, allowing more accurate age correction, and the age correction has been recalculated with some interesting results. Further work needs to be done on this: if age related atrophy affects different regions of the head differently, for example if the brain moves systematically in the skull, we will need a way to interpolate the age correction as the box centre points are moved around. Finally, I also intend to look at parallelising the GA, in order to reduce the time taken to do the optimisation. We anticipate a run time of about three months