An Improved Restricted Growth Function Genetic Algorithm for the Consensus Clustering of Retinal Nerve Fibre Data

Stephen Swift, Allan Tucker
School of Information Systems, Computing and Mathematics
Brunel University
Uxbridge, UB8 3PH, UK
+44 (0)1895 265745
[stephen.swift, allan.tucker]@brunel.ac.uk

Jason Crampton
Information Security Group, Department of Mathematics
Royal Holloway, University of London
Egham, TW20 0EX, UK
+44 (0)1784 443101
jason.crampton@rhul.ac.uk

David Garway-Heath
Glaucoma Research Unit, Moorfields Eye Hospital
162 City Road
London, EC1V 2PD, UK
+44 (0)20 7566 2087
david.garway-heath@moorfields.nhs.uk

ABSTRACT
This paper describes an extension to the Restricted Growth Function grouping Genetic Algorithm applied to the Consensus Clustering of a retinal nerve fibre layer data-set. Consensus Clustering is an optimisation based method which combines the results of a number of data clustering methods, and is used when it is unknown which clustering method is expected to perform the best. Consensus Clustering has been shown to produce results which are better than the averaged results of the input methods, but could benefit from a more efficient optimisation method. A Restricted Growth Function grouping Genetic Algorithm is a new method of grouping a number of objects into mutually exclusive subsets based upon a fitness function. This method does not suffer from degeneracy, and thus could be applied to the Consensus Clustering problem more efficiently than Simulated Annealing, the current optimisation method. Within this paper it is shown that this type of Genetic Algorithm can indeed improve the performance of Consensus Clustering, and in fact can be improved further by taking advantage of some application specific properties. These findings are demonstrated on a retinal nerve fibre layer data-set and on a synthetic data-set.

Categories and Subject Descriptors
I.2.8 [**Problem Solving, Control Methods, and Search**] Heuristic methods

General Terms
Algorithms, Measurement, Performance, Reliability

Keywords
Grouping, Genetic Algorithms, Restricted Growth Functions, Consensus Clustering, Retinal Nerve Fibre Layer.

1. INTRODUCTION
There are many data analysis problems which involve the partitioning of a set of objects into a number of mutually exclusive subsets, which is a known NP hard problem [6]. Applications in which partitions are determined using distance or correlation metrics are known as clustering, the more general area is often referred to as grouping problems. Any algorithm that applies a global search for the optimal clustering arrangement in a data-set will run in exponential time to the size of problem space, and so a heuristic or approximate procedure is nearly always required to cope with most real-world problems. Many different heuristic algorithms have been developed to solve clustering type problems, the most common being K-Means [19] and Hierarchical clustering [30]. Most algorithms make use of a starting allocation of variables, for example, based upon random points in the data space or upon the most correlated variables and therefore contain bias in their search. They are also prone to becoming stuck in local maxima during the search. There has also been research into the use of artificial intelligence techniques such as Genetic Algorithms [4], Neural Networks [16] and Simulated Annealing [17] to solve the grouping problem resulting in a more general partitioning method which can be applied to clustering. These methods aim to overcome the biases and local maxima involved with heuristic searches but require fine-tuning of parameters.

Due to the high degree of variation between clustering methods, a method called Consensus Clustering [27] has been previously developed, designed to combine the results of a number of clustering results into a single set which has a high similarity with the input methods. Consensus Clustering (CC) requires an optimisation method in order to combine the input clustering results, and Simulated Annealing [15] has been successfully applied to perform this task, although other more efficient methods could be used. CC is used when it is not known which data clustering method is expected to perform the best, and has been shown to produce results which are better than the averaged results of the input clustering methods. Thus CC provides a "safe bet" rather than having to guess which clustering method may produce the most accurate results.

Within this paper an investigation into the integration of the Restricted Growth Function Genetic Algorithm (RGFGA) and CC is presented. An RGFGA is a grouping Genetic Algorithm based

on Restricted Growth Functions [3, 22] which has shown a high degree of accuracy and efficiency when applied to grouping problems [28]. Additionally it is shown in this paper that by taking advantage of some of the specific properties of the CC problem, a vast improvement in the convergence rate and accuracy of the RGFGA can be obtained. These findings are demonstrated on a retinal nerve fibre layer data-set and on a synthetic multivariate normal data-set.

This paper is organised as follows: Section 2 details the methods utilised in this paper, along with notation and comparison metrics used. Section 3 describes the improvements made to the RGFGA crossover. Section 4 describes the data-sets that the methods presented are applied to, along with the description of the experiments carried out. Section 5 details the results of all of the experiments and discusses their implications. Finally, Section 6 draws some conclusions.

2. METHODS

This section details the methods which are used in this paper.

2.1 Notation

Let $X = [x_1, ..., x_n]$ be a list of variables, such that $x_i = x_j$ only when $i = j$. $G = [g_1, ..., g_m]$, where $g_i \subseteq X$, $i = 1, ..., m$ is a partition of X if the union of all the g_i is X and $g_i \cap g_j = \varphi$ if and only if $i \neq j$. g_{ij} is defined as the jth element of g_i. The cardinality of g_i is denoted as s_i. The term *clustering arrangement* will be used to refer to G and *cluster* to refer to g_i.

2.2 Consensus Clustering

Consensus Clustering [27] is a method for deriving a single set of clusters from several clustering methods. The aim of the method is to take advantage of where all of the methods agree in order to form the consensus clustering arrangement. The method consists of two stages, a pre-processing stage where a matrix is constructed containing the level of agreement between input methods, and an optimisation stage, where an optimal clustering arrangement is searched given a fitness function applied to the agreement matrix. Similar work can be seen in protein secondary structure prediction; where methods fail to completely agree consensus algorithms are employed [2]. These can either report only full agreements, or the majority of agreements. In [20] a Consensus Clustering type technique was introduced for testing the stability of clustering methods when applied to gene expression data. This method differs from the Consensus Clustering algorithm which is the subject of this paper since the inputs to the consensus method are the results of running a single algorithm on data-sets which are perturbations of the original. An investigation into the use of ensemble methods to combine a set of partitions is carried out in [26].

2.2.1 The Agreement Matrix

The consensus clustering agreement matrix will be denoted as A, and each element as a_{ij}. Given k clustering arrangements, defined as in section 2.1, denoted G_1 ,... ,G_k then an n by n matrix is constructed as follows. The matrix is initially set to zero. For each unique possible pairing of variables (x_i, x_j) for each cluster of each clustering arrangement, element a_{ij} is incremented by one. This means that if for a given pair of variables (x_i, x_j), all of the methods thought they should be clustered together, then the corresponding element of the agreement matrix (a_{ij}) would be k. However, if no method thought the variables should be clustered together, then the value would be zero. Note that $a_{ii} = k$ and that $a_{ij} = a_{ji}$.

2.2.2 Optimising the Clustering Arrangement

Given the agreement matrix, a search for an optimal clustering arrangement can be conducted. The fitness function detailed in equation (1) has been found to be highly effective for rating a candidate arrangement [27]. Given an appropriate value of β, the fitness function rewards clustering arrangements where the input clustering arrangements agree between each other, and penalises any pair of variables which have been placed together where there is low agreement. A good value for β was found to be $k/2$ (number of input methods divided by two) as discussed and justified in [27]. Currently, Simulated Annealing [15] is used to perform the optimisation, since it is a simple method to implement and has a good track record of searching out global maxima.

$$C(G) = \sum_{i=1}^{m} H(g_i) \tag{1}$$

$$H(g_i) = \begin{cases} \sum_{j=1}^{s_i-1} \sum_{k=j+1}^{s_i} (a_{g_{ij}g_{ik}} - \beta) & s_i > 1 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

2.3 Clustering Methods

Consensus Clustering requires a number of clustering results as an input. The methods used within this paper are described within Table 1; these consist of K-means clustering, Partitioning Around Medoids, Hierarchical clustering using average linkage and Model-based Clustering.

Table 1. CC Input Clustering Methods

Method	Abbreviation	Reference
K-Means	KME	[30]
Partitioning Around Medoids	PAM	[13]
Hierarchical (Average)	HAV	[19]
Model-based Clustering	MBC	[5]

These four methods have been chosen as they are amongst the most common methods used for clustering and also represent a number of different approaches to clustering, i.e. agglomerative, partitional and model-based.

To compare how similar two cluster arrangements are, the Weighted Kappa (WK) metric [1, 14] is used. The metric ranges between +1 for identical clustering arrangements to -1 for completely dissimilar arrangements, and the expected WK between two random clustering arrangements is zero. An interpretation of the WK metric can be seen in [1].

2.4 RGFGA

A Restricted Growth Function (RGF) is a list $f = [f(1), f(2), ..., f(n)]$ such that equation (3) holds.

$$\begin{aligned} &f(1) = 1 \text{ and} \\ &f(i+1) \leq \max\{f(1), f(2), ..., f(i)\} + 1 \end{aligned} \tag{3}$$

For example, in the case where $n = 5$, both $f = \{1,1,2,1,3\}$ and $\{1,2,3,4,1\}$ are valid RGFs, but $\{1,2,3,1,6\}$ and $\{1,3,1,2,3\}$ are not. It has been shown that for a given value of n, the set of all

valid RGFs has a one to one mapping to the set of all partitions of the integers 1, ..., n; hence RGFs can be used for a variety of combinatorial optimisation tasks such as data clustering [12], bin packing [6], and Consensus Clustering. A Genetic Algorithm based on RGFs was introduced in [28]. In order to convert an RGF f to a partition of integers (a clustering arrangement), given the sequence $f(i)$, then variable x_i is placed into cluster $g_{f(i)}$.

Falkenauer's Grouping Genetic Algorithm has been designed for dealing with grouping problems [4]. Additionally, the PMX crossover operator, developed for ordering problems [8], can be adjusted to handle grouping problems as shown in [29]. List of lists representations have also been used [18] along with graph partitioning based representations [21].

Many of the representations used in optimisation algorithms (including those in the GAs described above) suffer from *degeneracy*. Degeneracy occurs when multiple chromosomes represent the same solution [25]. Degeneracy can lead to inefficient exploration of the search space as the same clustering arrangements are repeatedly revisited. The minimisation of degeneracy is considered an essential part of the design of any GA representation [23]. This is not the same as *redundancy*, which is defined as the amount of excess information in the chromosome, and in some cases redundancy can be beneficial [9].

Other methods for clustering using Genetic Algorithms represent a clustering arrangement as a series of cluster centres (as points in the clustering space), where the data objects are placed in the cluster whose centre is the closest (e.g. using Euclidean distance), similar to the strategy used in K-Means clustering. Examples of this representation can be found in [11]. However, this representation, although successfully implemented, is not generic enough for the general grouping problem, and could not be used in bin-packing, for example.

2.4.1 RGFGA Representation

Within the RGFGA, a chromosome representing a clustering arrangement G of n objects is represented by an RGF. The initial population is created according to [28], where the value of a gene, $f(i)$, is randomly chosen from the discrete uniform distribution ranging between 1 and $\max\{f(1),f(2),...,f(i\text{-}1)\}$, note that $f(1) = 1$.

2.4.2 RGFGA Crossover

In order to describe the Crossover operator, the notion of Hamming Distance between two RGFs (f and g) must be defined, and is given in equation (4):

$$HD(f,g) = \sum_{i=1}^{n} |f(i) - g(i)| \tag{4}$$

Crossover is different to that of a standard Genetic Algorithm (GA) and consists of mapping a path between the two parents and selecting two children randomly from this path. Similar to path re-linking [24] this path consists of a list (for a given pair of parents) of RGFs where the first item in the path is the first parent, and the last item in the path is the second parent. The Hamming Distance between any two adjacent RGFs in the path is one. A full description of Crossover and the proofs of certain properties of RGFs can be found in [28]; an overview of how the path between two chromosomes/RGFs is constructed follows:

Definition 1. Let f and g be two RGFs. We say $f \le g$ if $f(i) \le g(i)$, $1 \le i \le n$. We write $f < g$ if $f \le g$ and $f \ne g$.

Definition 2. Let f and g be two RGFs, then we define $\overline{fg}$ as:

$$\overline{fg}(i) = \max_{1 \le i \le n}\left(f(i), g(i)\right)$$

The proof that this is an RGF can be found in [28].

Definition 3. Let f and g be RGFs such that $f \ne g$ and let j be the smallest integer such that $f(j) < g(j)$, $2 \le j \le n$ and k be the largest integer such that $f(k) > g(k)$. Two functions can be defined:

$$(f \uparrow g)(i) = \begin{cases} f(i)+1 & \text{if } i = j \text{ and } f(j) < g(j) \\ f(i) & \text{otherwise} \end{cases}$$

$$(f \downarrow g)(i) = \begin{cases} f(i)-1 & \text{if } i = k \text{ and } f(k) > g(k) \\ f(i) & \text{otherwise} \end{cases}$$

Proposition 1. Let f and g be RGFs such that $f \ne g$. Then $(f \uparrow g)$ is an RGF and $HD((f \uparrow g),g) = HD(f,g) - 1$. The proof of this can be found in [28]. A similar result holds for $(f \downarrow g)$.

Let f and g be two distinct RGFs. If $f < g$, then $(f \uparrow g)$ exists. Clearly, $f \le \overline{fg}$. Hence, according to Proposition 1, if $f \ne \overline{fg}$ then $(f \uparrow g)$ is an RGF that is one step closer (in terms of Hamming Distance) to $\overline{fg}$ than f is. Hence, by repeating this construction, there exists a finite sequence of RGFs, $f_1,...,f_k$ such that $f_1 = f$, $f_i = (f_{i-1} \uparrow \overline{fg})$ and $f_k = \overline{fg}$. Similarly, there exists a finite sequence of RGFs, $g_1,...,g_l$ such that $g_1 = \overline{fg}$, $g_i = (g_{i-1} \downarrow g)$, and $g_l = g$. We say the sequence of functions $f_1,...,f_k$ $g_1,...,g_l$, is a path from f to g.

2.4.3 RGFGA Mutation

Three mutation operators are implemented which have one of the following effects:

1) A variable is moved from one cluster to another cluster
2) Two clusters are merged together
3) A cluster is split into two non-empty clusters

An individual is mutated according to the mutation rate, and one of the mutation operators is chosen at random. Elements within a cluster, or clusters are chosen at random depending on what is appropriate to the mutation operator. Note that when a mutation has been applied, there is a chance that the chromosome which has mutated is no longer a valid RGF, i.e. equation (3) no longer holds. If this is the case, a re-labelling will need to be applied as described in [28].

2.4.4 RGFGA Fitness

Within this paper, the fitness function under consideration is the fitness function of the CC problem, defined in equation (1). Given that CC is defined to be evaluated on a list of lists rather than a RGF, equations (1) and (2) have to be rewritten as follows:

$$C_{RGF}(f) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} H_{RGF}(f,i,j) \tag{5}$$

$$H_{RGF}(f,i,j) = \begin{cases} a_{ij} - \beta & f(i) = f(j) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Note that this computational complexity of equation (1) is $O\left(\sum_{i=1}^{m} s_i(s_i - 1)/2\right)$ which is always less than or equal to the computation complexity of equation 5, which is $O(n(n\text{-}1)/2)$.

3. IMPROVEMENTS TO CROSSOVER

Given two candidate clustering arrangements within the CC problem, f and g such that HD(f,g) = 1, and p is the position that they differ ($2 \le p \le n$), then if $C_{RGF}(f)$ is known, then $C_{RGF}(g)$ can be calculated according to an update formula, derived as follows:

$$C_{RGF}(f) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} H_{RGF}(f,i,j) \text{ and } C_{RGF}(g) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} H_{RGF}(g,i,j)$$

$$\text{Let } D_{RGF}(f,g,i,j) = H_{RGF}(g,i,j) - H_{RGF}(f,i,j)$$

Note that $D_{RGF}(f,g,i,j)$ is zero when both $i \neq p$ and $j \neq p$.

$$C_{RGF}(g) - C_{RGF}(f) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} D_{RGF}(f,g,i,j)$$

$$C_{RGF}(g) - C_{RGF}(f) = \underbrace{\sum_{\substack{i=1 \\ i\neq p}}^{n-1}\sum_{\substack{j=i+1 \\ j\neq p}}^{n} D_{RGF}(f,g,i,j)}_{=0} + \sum_{j=p+1}^{n} D_{RGF}(f,g,p,j) + \sum_{i=1}^{p-1} D_{RGF}(f,g,i,p)$$

$$C_{RGF}(g) - C_{RGF}(f) = \sum_{j=p+1}^{n} D_{RGF}(f,g,p,j) + \sum_{i=1}^{p-1} D_{RGF}(f,g,i,p)$$

$$C_{RGF}(g) = C_{RGF}(f) + \sum_{j=p+1}^{n} D_{RGF}(f,g,p,j) + \sum_{i=1}^{p-1} D_{RGF}(f,g,i,p)$$

Hence the calculation of the fitness $C_{RGF}(g)$ becomes of complexity O(n-1) (from the two final summations immediately above). Therefore, with the RGFGA Crossover operator, it should theoretically be possible to explore a large proportion of the children on the path between two parents. The intention is that adjacent RGFs on the two sides of the paths are explored and evaluated using the update formulae. The starting points will be the two parents, and a number of children are explored moving from one parent to the other from both edges of the path; the best child from each edge will be the resultant children. The computational effort to explore a number of children from each edge will be set to equal the same complexity as evaluating a child, thus being equal to the complexity of the current RGFGA Crossover. Within equation (7) the value of x is the number of children to be explored using the update formulae from each edge.

$$\underbrace{x(n-1)}_{\text{O(Update)}} = \underbrace{\frac{n(n-1)}{2}}_{\text{O(Fitness)}} \therefore x = \frac{n}{2} \qquad (7)$$

The proportion of the total possible children explored will depend on the Hamming Distance between the parents. If this proportion is too small, then the variation in the children produced by the update formulae version of Crossover may be too small and may adversely affect the results. It is reasonable to assume that the average HD between two random RGFs may be related to the size of the RGF, n; from equation (4) it can be seen that as n increases $HD(f,g)$ would be expected to increase). To see if this was indeed true, a number of simulations (1,000,000 for each value of n) were conducted for varying values of n from 1 to 100 (in steps of 1), this range was chosen as it encompasses the dimensionality of the two test data-sets, see section 4.1. For each RGF size (n), 1,000,000 random pairs of RGFs were created using the method described in [28], and the HD measured between them and averaged over the number of simulations.

Figure 1 shows a plot of $\sqrt{n}$, n, $n\sqrt{n}$ and n^2 against the average Hamming Distance from the simulations; these four functions of n have been chosen since it is known that the Hamming Distance between two RGFs will always be less than or equal to $n(n\text{-}1)/2$ [28]. Note that some of the data for the plot against n^2 has been omitted since the values on the y-axis become large and obscure the plots for the other functions. All four appear to have a clear linear relationship; however Table 2 shows the results of applying linear regression to the four sets of data. From this set of results it can be seen that $n\sqrt{n}$ has the best fit against Hamming Distance, since the value for R^2 is the greatest, and it has the most plausible value for the y-axis intercept (we know that when n=1, HD=0).

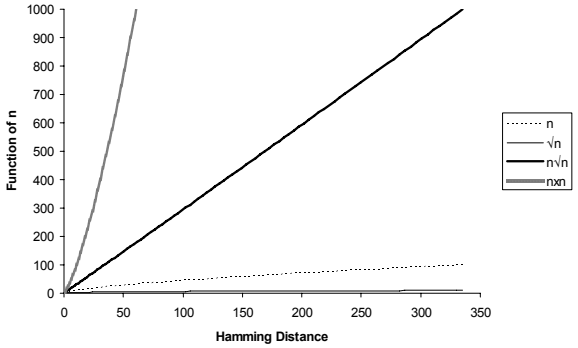

Figure 1. The Four Functions of *n* Against Hamming Distance

Table 2. Linear Regression Results for the Four Functions of *n*

Function of *n*	R^2	Gradient	Y-Intercept
$\sqrt{n}$	0.898	41.247	-140.511
n	0.982	3.476	-39.094
$n\sqrt{n}$	1.000	0.336	0.461
n^2	0.987	0.033	23.331

From Table 2, is would be reasonable to assume that the HD between two random RGFs of size n is approximately $\frac{1}{3}n\sqrt{n}$ (using the linear fit for $n\sqrt{n}$ from Table 2). From equation (7) it is clear that n children out of the total possible number of children on the path between two parents will be explored. This means that if given two random RGFs as parents then the proportion of children considered is equal to $3/\sqrt{n}$ (n children explored divided between $\frac{1}{3}n\sqrt{n}$ expected children).

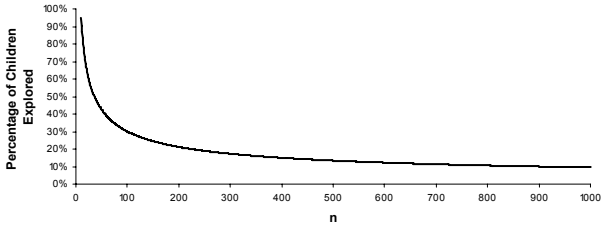

Figure 2. *n* Against Percentage of Children Explored for the Improved Crossover Operator

Given that the dimensionality of the problems being considered in this paper is less than 100, over 30% of the total possible children will be considered, a significant enough percentage to justify the new Crossover operator. Since the population starts off random and then starts towards becoming homogeneous (as seen in [28]), this proportion will be expected to increase as the number of generations increases. Figure 2 shows this percentage for values of n ranging from 10 to 1000 (for two random parents/RGFs), note that when n=1000, the percentage is approximately 10% which is still a reasonable proportion of the children.

4. DATA-SETS AND EXPERIMENTS

This section describes the two data-sets which are used within this paper along with the experiments which have been carried out.

4.1 Data-sets

A real-world data-set and a synthetic data-set are used within this paper and are described as follows.

4.1.1 The GDx Data

The GDx data-set is a set of ophthalmic data that was the result of a study carried out in Australia called the Blue Mountain Eye Study [11]. This study concerned the vision and prevalence of common eye diseases of an urban population, and was carried out between 1992 and 1994 on several thousand people. One of the sets of measurements taken during this study was the thickness of the retinal nerve tissue, using GDx test equipment. GDx stands for Glaucoma Diagnosis. It is thought that the thickness of the retina nerve tissue is strongly related to numerous eye conditions, especially Glaucoma [7]. The GDx test measures the thickness of the retinal nerve tissue (Nerve fibre layer) on 64 evenly spaced points on an annulus centred on the optic nerve head in the retina and uses polarimetric imaging techniques to estimate the nerve fibre layer thickness at each point. Research has been carried out which maps the distribution of nerve fibre bundles around the optic nerve head [7], and current theory states that any measurements in the same nerve fibre bundle sector should be highly related. Within this paper it is aimed to test this theory by applying a number of clustering methods to the GDx data-set and then obtaining a consensus set of clusters through the use of Consensus Clustering. The accuracy of the clustering methods and Consensus Clustering can be directly measured using the WK metric and the allocation of GDx points to nerve fibre bundle.

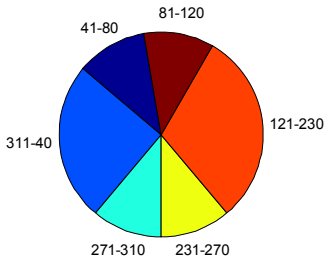

Figure 3. The Optic Nerve Head Divided into Sectors

The mapping of the distribution of nerve fibre bundles can be found in [7] and is shown in Figure 3. Within this figure the text for each section refers to the angle between the retina and the optic nerve head; the first GDx point starts at zero degree, and the rest are evenly spaced clockwise every 5.625 degrees. The data used in this paper is a subset of the entire Blue Mountain GDx data-set; it has been filtered on a study specific quality metric and only data for the right eye has been selected.

4.1.2 The Multivariate Normal Data

This data-set was previously developed and used in [27]. A vector of random variables $\underline{x}$ has a Multivariate Normal distribution if every linear combination of the vector is also normal; if this is the case then the notation $\underline{x} \sim N(\underline{\mu},\Sigma)$ is used. Here $\underline{\mu}$ is a vector of means and Σ is a matrix of covariances. The Multivariate Normal data-set used in this paper is a concatenation of ten length samples from Multivariate Normal distributions varying in dimensionality from 1 to 11. This results in a 66 variable data-set. The expected clustering results are therefore known, i.e. variable 1 in cluster 1, variable 2 and 3 in cluster 2 etc... However, due to the low sample size (10), methods may result in non-perfect clustering results, i.e. a WK value of less than 1 when compared with the expected arrangement. This data-set will be referred to as the MVN data-set.

Table 3. Description of Experiments

Abbreviation	Description
GDx_IMP	The improved RGFGA using the update formulae based Crossover run on the GDx data.
GDx_NOX	The RGFGA with no Crossover, (see below for GA parameter settings), run on the GDx data.
GDx_S	The standard RGFGA run on the GDx data.
GDx_SA	Simulated Annealing run on the GDx data.
MVN_IMP	The improved RGFGA using the update formulae based Crossover run on the MVN data.
MVN_NOX	The RGFGA with no Crossover, (see below for GA parameter settings), run on the MVN data.
MVN_S	The standard RGFGA run on the MVN data.
MVN_SA	Simulated Annealing run on the MVN data.

4.2 Experiments

Both data-sets are clustered once using the methods described in section 2.3. A Consensus Clustering agreement matrix is then constructed for each of the sets of results. Consensus Clustering experiments were then conducted as described in Table 3. Since each method is stochastic, the experiments were repeated 25 times and the results averaged to reduce the chance of "fluke" results. The _NOX experiments were introduced to demonstrate the improvements gained by the Crossover operators. All experiments were restricted to a run of 250,000 fitness function evaluations (or equivalent), which was found to be a high enough value to allow the methods to converge. For the Genetic Algorithms, a population size of 500 has previously been found to be appropriate, the Crossover rate was set to 100% and the Mutation rate was set to 50%. With the _NOX experiments, the Crossover rate was set to 0% and the mutation rate was set to 100%.

5. RESULTS

The results section is split up into three parts. The first part considers the clustering results for both data-sets, the second looks at the quality of the Consensus Clustering results, and the third part looks at the performance of the optimisation methods applied to the Consensus Clustering of the input clustering results.

5.1 Clustering Results

Table 4 shows the WK results for the four input methods on the two data-sets; the average results are also shown. The value in parentheses is the ranking of the method with regards to the other methods (across the same data-set).

The HAV method produces the best set of results across both data-sets, achieving a high WK result and a ranking of 1. The MBC method produces a below average result, with a WK of less than 0.4, and a rank of 4th place. For the two remaining methods, they attain approximately the same WK result, being within a WK difference of 0.050 from each other for both data-sets. The average results are in the same general quality category according to [1], being a "Moderate" result. As can be seen from the table, the standard deviation (St.Dev.) of the WK results is quite high. For the GDx data-set, the WK results range from approximately 0.30 to 0.55, and for the MVN the range is from approximately 0.35 to 0.65, demonstrating how variable clustering results can range from method to method.

Table 4. WK Results for the CC Input Methods

Method	GDx	MVN
KME	0.506 (3)	0.490 (2)
PAM	0.556 (2)	0.450 (3)
HAV	0.559 (1)	0.666 (1)
MBC	0.296 (4)	0.354 (4)
Average:	0.479	0.490
St.Dev.:	0.125	0.130

5.2 Consensus Clustering Results

Table 5 summaries the CC results when applied to the input clustering methods. Both the final fitness and WK are given, averaged over the 25 repeated runs.

Table 5. CC Results for Both Data-sets

Method	Fitness		WK	
	Mean	St.Dev.	Mean	St.Dev.
GDx_IMP	341.000	0.000	0.515	0.006
GDx_NOX	341.000	0.000	0.497	0.006
GDx_S	341.000	0.000	0.515	0.006
GDx_SA	339.120	5.848	0.512	0.014
MVN_IMP	217.000	0.000	0.619	0.015
MVN_NOX	217.000	0.000	0.591	0.020
MVN_S	213.960	1.719	0.618	0.020
MVN_SA	214.440	5.987	0.592	0.033

With the GDx data, looking at the fitness results, it is clear to see that the RGFGA based versions of CC outperform the Simulated Annealing version (GDx_SA) with regards to average fitness attained, and they also achieve a very consistent set of results since the standard deviation is zero. When compared to the WK results in Table 4, the performance of CC (WK = 0.515) would place it in third overall ranking, and only a WK difference of 0.044 away from the maximum.

With the MVN data-set, looking at the fitness results, two methods tie together; these are the improved RGFGA (MVN_IMP) and the Crossover-less RGFGA (MVN_NOX). These two methods both obtain the result for each of the 25 repeated runs, since the standard deviation of the fitness is zero. The worst performing method is the standard version of the RGFGA (MVN_S), which gets a lower average than the other three methods. When compared to the WK results in Table 3, the performance of CC (WK = 0.619) would place it in second overall ranking (with a WK difference of 0.047 from the top ranking method), and a standard deviation (approximately) greater than the mean result of the input methods.

Within both sets of results, there is what appears to be a slightly strange result. For example, with the GDx_IMP and GDx_NOX results, both attain the same average fitness but the average WK results are different. It is worth noting that this could indeed occur. The fitness function described in equation (5) is intended as an approximation to WK, since in most applications an analyst would not know the expected clustering results and therefore could not calculate the WK for any clustering results.

The correlation between all of the evaluated pairs of data for the CC fitness function and WK which form the results presented in this paper can be calculated; this data is collated from all of the experiments carried out, and not just from the final results but from various stages during the experiments run. Only unique pairs are considered, since if a method has converged then the data being correlated may become biased (i.e. a large portion of the data being correlated containing the same numeric pair). For the GDx data the correlation is 0.889 and for the MVN data it is 0.776. The overall relation between the CC fitness function and the WK metric is very high, demonstrating that the fitness function is indeed suitable for its purpose. However, there is a noted difference between the correlations for the two data-sets, this suggests that anomalous results may occur more often with the MVN data-set results than with the GDx data-set results, and indeed we have anecdotal evidence of this as noted above.

To conclude this section of the results, it can be clearly seen that for both data-sets, the CC results produce a set of clustering results which is better than the average of the input methods. With regards to clustering the GDx data and investigating if the results match the allocation of GDx points to nerve fibre bundle, the results are very promising, there is clearly enough evidence to suggest that this is indeed the case. The results from Tables 4 and 5 show that if an analyst did not know what the expected clustering results should be, then the use of Consensus Clustering with a diverse number of input clustering methods would be a reliable way to obtain a clustering arrangement with a high degree of confidence. This claim is also backed up by the results on the MVN data-set. However, the results do not irrevocably indicate which method of optimisation performs the best. The results of looking at the rate of convergence shed light on this and are now discussed in the next section.

5.3 Optimisation Method Results

This section looks at the convergence of the methods across the two data-sets. Figure 4 shows the convergence graph for the GDx experiments and Figure 5 shows the convergence graph for the MVN experiments. The figures are a plot of number of fitness calls against fitness, averaged over the 25 runs.

From Figure 4 (the GDx data-set results), it can be seen that the best rate of convergence is achieved by the improved version of the RGFGA (GDx_IMP), followed by the standard version of the RGFGA (GDx_S), then the Crossover-less version (GDx_NOX), and then Simulated Annealing (GDx_SA). GDx_IMP and GDx_S seem to have a much more rapid rate of convergence than the other two methods. From Figure 5 (the MVN data-set results), it can be seen that the rate of convergence rankings is the same as in Figure 4 (MVN_IMP, MVN_S, MVN_NOX and MVN_SA), but the difference between the methods is not as high. Both graphs indicate that both forms of RGFGA Crossover have a significant positive effect in the early generations of the algorithms execution, when compared with a Crossover-less version.

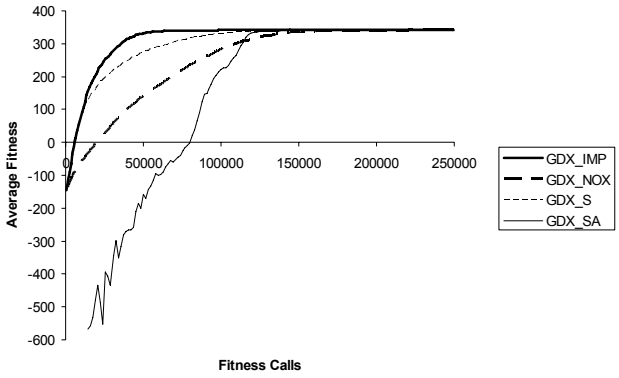

Figure 4. Convergence Graph for the GDx Data-set

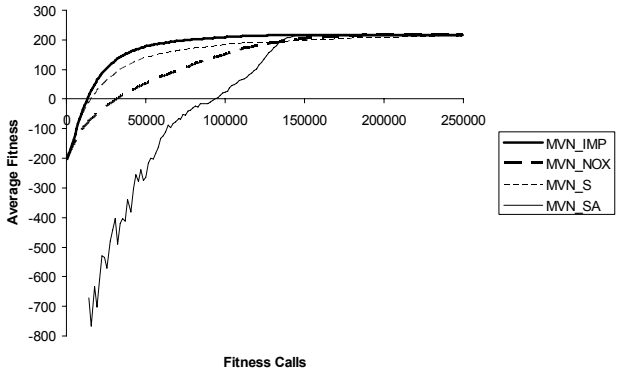

Figure 5. Convergence Graph for the MVN Data-set

To ascertain at what point a method has finally converged, the maximum fitness a method reaches is recorded, and then the first point this happens (in terms of fitness calls) is then ascertained. This value is averaged over the 25 runs to give an indication of at what point the methods converge. This is detailed in Table 6. For both the GDx and MVN data-sets, Table 6 supports the results presented in Figure 4 and Figure 5 respectively. In both cases the improved version of the RGFGA converges first, followed by Simulated Annealing. With the GDx data-set, the improved version converges significantly sooner than all of the other methods, whilst with the MVN data-set it converges just before Simulated Annealing. Note that these results should be considered in conjunction with those in Table 5, since convergence does not necessarily occur at the highest fitness value. This is the case with the improved version of the RGFGA but not with Simulated Annealing.

Table 6. Average Convergence Point

Method	Convergence Point (Rank)
GDx_IMP	72,456 (1)
GDx_NOX	166,260 (4)
GDx_S	141,289 (3)
GDx_SA	128,859 (2)
MVN_IMP	142,347 (1)
MVN_NOX	194,260 (3)
MVN_S	242,299 (4)
MVN_SA	151,364 (2)

Combining the results from these two figures and from Table 6, it would seem that the new version of Crossover for the RGFGA achieves a very rapid convergence in a low number of calls. However, convergence may not always be at the best solution, but instead be at a high scoring local maximum. The RGFGA will then rely on mutation to find a better solution. From [28] it was found that the standard version of the RGFGA had a tendency towards the population becoming homogeneous, since the Crossover operator is designed to produce children which are similar to the parents. The improved version of the Crossover operator, should if anything, be worse than the standard version for producing such homogeneity. This is due to the children being drawn from the two edges of the path, whilst the standard version chooses two children randomly from anywhere on the path. The Hamming Distance between a parent and a child under the new Crossover will never be more than $n/2$ whilst with the current Crossover the distance depends on how far apart the two parents are from each other. The new Crossover operator may well sacrifice diversity of the population for a much more rapid convergence; new methods of creating a more diverse initial population may well help this behavior.

6. CONCLUSIONS

Within this paper, an adapted version of Crossover for a Restricted Growth Function Genetic Algorithm has been introduced. This new crossover operator takes advantage of the way that the existing Crossover generates potential children, and the structure of the fitness function for the problem being addressed, Consensus Clustering. When applied to a real-world and synthetic data-set, the results show that the improved version of Crossover achieves a faster rate of convergence than the original RGFGA version and Simulated Annealing. However, there is evidence to suggest that although the new Crossover vastly improves convergence, it also increases the rate of population homogeneity; this will be rigorously investigated as future work (see below).

The results on the ophthalmologic GDx data-set (and the MVN data-set) show that using Consensus Clustering can improve the accuracy of clustering, if it is unknown what the most appropriate method would be. The GDx results also show that there is good agreement between the clustering of the GDx annulus thickness measurements and the nerve fibre bundle sectors.

Finally, the new Crossover operator is not restricted to the Consensus Clustering problem, but can be applied to any problem where an update formula can be derived as in section 3, for

example one should exist for bin packing problems. There is plenty of scope for extending this research, for example, an analysis of population homogeneity could be carried out, along with more experiments to investigate the scalability of the method as the problem dimensionality increases. Finally, the methods could be applied to other clustering and grouping type problems such as time series clustering and bin packing.

7. Acknowledgements

We would like to thank Paul Healey and Paul Mitchell for making the Blue Mountain data-set available. We would also like to thank David Crabb and Haogang Zhu for preparing the GDx data.

8. References

[1] Altman, D.G., *Practical Statistics for Medical Research*. Chapman and Hall, London, 1997.

[2] Cuff, J.A., Clamp, M.E., Siddiqui, S.A., Finlay, M., and Barton, G.J., JPred: A consensus secondary structure prediction server. *Bioinformatics*, 14 (1998), 892-893.

[3] Er, M., A fast algorithm for generating set partitions. *The Computer Journal*, 31, 3 (1988), 283–284.

[4] Falkenauer, E., *Genetic Algorithms and Grouping Problems*. Wiley, 1998.

[5] Fraley, C., Raftery, A.E., Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association* 97 (2002), 611-631

[6] Garey, M. and Johnson, D., *Computers and Intractability*. W. H. Freeman and Company, New York, NY, 1979.

[7] Garway-Heath, D.F., Poinoosawmy, D., Fitzke, F., Hitchings, R.A., Mapping the Visual Field to the Optic Disc. *Opthalmology* 107 (2000), 1809-1815.

[8] Goldberg, D. and Lingle, R., Alleles, loci, and the travelling salesman problem. In *Proceedings of the First International Conference on Genetic Algorithms and their Applications* (1985), 154–159.

[9] Hackworth, T., Genetic algorithms; Some effects of redundancy in chromosomes, In *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-1999) (Orlando, Florida, USA, 1999), 99-106.

[10] Hall, L.O., Ozyurt, I.B. and Bezdek, J.C., "Clustering with a genetically optimized approach", *IEEE Transactions on Evolutionary Computation* 3, 2 (1999), 103-112.

[11] Healey P.R. and Mitchell P., Visibility of lamina cribrosa pores and open-angle glaucoma, *American Journal of Ophthalmology* 138, 5 (2004), 871-872.

[12] Jain, A., Murty, M., and Flynn, P., Data clustering: A review. *ACM Computing Surveys* 31, 3 (1999), 264–323.

[13] Kaufman, L., Rousseeuw P.J., Clustering by means of medoids, In *Statistical Analysis Based Upon the L1 Norm*. Edited by: Dodge Y., Amsterdam, Holland, 1987, 405-416.

[14] Kellam, P., Liu, X., Martin, N., Orengo, C., Swift, S., and Tucker, A., Comparing, Contrasting and Combining Clusters in Viral Gene Expression Data. In *Proceedings of the Intelligent Data Analysis in Medicine and Pharmacology Workshop* (IDAMAP-2001) (London, UK, 2001), 56-62.

[15] Kirkpatrick, S., Gelatt Jr, C.D., and Vecchi M.P., Optimization by simulated annealing. *Science*, 220 (1983), 671-680.

[16] Kohonen, T., *Self Organization and Associative Memory.* 3rd edition, Springer-Verlag, New York, 1989.

[17] Lukashin, A.V., and Fuchs, R., Analysis of temporal gene expression profiles: clustering by simulated annealing and determining the optimal number of clusters. *Bioinformatics*, 17 (2001), 405-414.

[18] Ma P.C.H., Chan K.C.C., Xao, X. and Chiu K.Y., An Evolutionary Clustering Algorithm for Gene Expression Microarray Data Analysis, *IEEE Transactions on Evolutionary Computation* 10,3 (2006), 296-314.

[19] McQueen, J., Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability* (Berkeley, 1967), 281-297, 1967.

[20] Monti, S., Tamayo, P., Mesirov, J., and Golub, T., Consensus clustering: a resampling-based method for class discovery and visualization of gene expression, microarray data. *Machine Learning*, 52 (2003), 91-118.

[21] Park, Y. and Song, M., A genetic algorithm for clustering problems. In *Proceedings of the 3rd Annual Conference on Genetic Programming*, 1998, Morgan Kaufmann, 568-575.

[22] Proskurowski, A., Ruskey, F., and Smith, M., Analysis of algorithms for listing equivalence classes of k-ary strings. *SIAM Journal on Discrete Mathematics*, 11, 1 (1998), 94-109.

[23] Radcliffe, N. and Surry, P., Fitness variance of formae and performance prediction. In *Whitley, D. and Vose, M., editors, Foundations of Genetic Algorithms* 3, (San Mateo, 1995), Morgan Kaufmann, 51–72.

[24] Radcliffe, N., Equivalence class analysis of genetic algorithms. *Complex Systems* 5 (1991), 183–205.

[25] Reeves, C. and Yamada, T., Genetic algorithms, path relinking, and the flowshop sequencing problem. *Evolutionary Computation* 6,1 (1998), 45–60.

[26] Strehl, A., and Ghosh, J., Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, 3 (2002), 583-617.

[27] Swift, S., Tucker, A., Vinciotti, V., Martin, N., Orengo, C., Liu, X., and P. Kellam, Consensus Clustering and Functional Interpretation of Gene Expression Data. *Genome Biology* 5, 11 (2004), R94.1-R94.16.

[28] Tucker, A., Crampton, J., and Swift, S., RGFGA: An Efficient Representation and Crossover for Grouping Genetic Algorithms. *Evolutionary Computation*, 13, 4 (2005), 477-499.

[29] Tucker, A., Swift, S., and Liu, X., Grouping multivariate time series via correlation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 31 (2001), 235-245.

[30] Ward, J.H., Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association* 58 (1963), 236-244.