Modelling Niches of Arbitrary Shape in Genetic Algorithms using Niche Linkage in the Dynamic Niche Clustering Framework

Justin Gan & Kevin Warwick
Cybernetic Intelligence Research Group
Department of Cybernetics
University of Reading, Whiteknights, Reading
Berkshire, RG6 6AY, UK
Email: jgrg@cyber.reading.ac.uk

Abstract – We present here some additions to a fuzzy variable radius niche technique called Dynamic Niche Clustering (DNC) [3, 4 & 5] that enable the identification and creation of niches of arbitrary shape through a mechanism called *Niche Linkage*. We show that by using this mechanism it is possible to attain better feature extraction from the underlying population.

I. Introduction

Most radius based niching algorithms in Genetic Algorithms (GAs) tend to make a number of assumptions about the fitness landscape and the types of peak that will be encountered. These assumptions are that the peaks are hyperspherical in shape, the size of the peaks are approximately the same, and each peak is distinctly separate. Of course, these assumptions very rarely coincide with the demands of real-world applications where fitness landscapes are noisy, multi-modal and highly irregular. Many of the nichers in the literature make these assumptions by enforcing a hard clustering philosophy, and perhaps more significantly, employing only a single value of niche radius to describe *all* of the niches[1] in the search space. Choosing an appropriate value for this single niche radius has become known as the *niche radius problem*. A review of the many niche techniques in publication is beyond the scope of this paper and the reader is referred to the relevant literature for further details.

In [3,4 and 5] we presented a potential solution to the niche radius problem called *Dynamic Niche Clustering* (DNC). DNC is a fitness sharing [2, 7] based niching framework that allows the creation of overlapping niches, each with its own independent variable niche radius. However, the only assumption that DNC makes is that the peaks are hyperspherical in shape. In this paper, we present a further extension to this framework that allows the creation of niches of arbitrary shape through a mechanism we have termed *Niche Linkage*. We show how this mechanism frees DNC of the assumption that the peaks must be hyperspherical, and that as a result, a drastic improvement in performance on fitness landscapes containing ridges and plateaus can be achieved. We also introduce the concept of niche dynamics and its use both as an additional decision tool within the framework, and as a possible measure of overall population convergence. We start first by briefly reviewing the DNC framework, and describing the *Niche Linkage* mechanism and how it fits into DNC.

II. Overview of DNC

DNC is a fuzzy variable radius fitness sharing based niching framework that operates over a population in a standard GA. It is superficially similar to classical agglomerative clustering techniques in that it starts, in generation 0, by looking at niches that initially contain one individual. However, the fundamental difference is that DNC is designed to operate over an evolving, dynamic population. Only one stage of clustering is executed per generation, as there is no point in performing a complete cluster analysis in every generation. This is especially the case in the initial generations where the population is randomly distributed throughout the search space. It is only once the population has completely converged that a full cluster analysis (for example, k-means) would be useful and computationally efficient. In DNC, the clustering is achieved through a generational process of the movement, merging and splitting of niches, running concurrently with the evolving GA population.

A unique element of DNC is the fact that the niches are fuzzy – that is they are allowed to overlap. Overlapping the niches allows for a much greater acuity[2] than that attainable using a hard clustering technique whilst still maintaining good niche coverage. Its use means that there is no need for peaks to be distinctly separate. Hard clustering cannot achieve this without using disproportionately small clusters. In DNC, there is still an absolute minimum peak distance that can be realised due to the nature of the merge action (see [5]), but this distance can be decreased even further through the use of the Hill-Valley function (see Section II-D).

The fundamental concept behind DNC is that the niches are persistent – that is the same niches are maintained from one generation to the next. In this way the niches are able to track the fit individuals in a converging population. All niches start with the same initial niche radius which is calculated based on the relationship between population size, the size of the search space, the amount of initial overlap required between adjacent niches, and the assumption that the initial population is uniformly distributed about the search space. The premise is to start with small niches that gravitate towards large clusters of fit individuals within their basin of attraction. Then, as the small niches converge they are merged into single, larger, more representative niches.

[1] A niche is defined as a species or subset of individuals that are quantifiably similar.

[2] We define acuity as the algorithms ability to distinguish between two very close peaks.

0-7803-7282-4/02/$10.00 ©2002 IEEE

A - DNC in Brief

We present here a brief outline of the DNC algorithm. For full details the reader is referred to [3, 4 & 5]. The *nicheset* is the set of niches in the current generation. Each niche is persistent, so the same set of niches from generation *t* will be carried over to generation *t+1*. The minimum and maximum values of niche radius are also stored in the nicheset. These values are dependent on the initial niche radius and are defined as ½ the initial niche radius and 2 times the initial niche radius, respectively.

Each niche is described by a midpoint, mid_i, in decoded parameter space, and a current niche radius, σ_{shi}. The initial value of niche radius is calculated using the following equation. See [5] for a detailed explanation.

$$\sigma_{sh_{initial}} = \frac{\lambda\sqrt{d}}{\sqrt[d]{p}} \quad (1)$$

Here, *d* is the number of dimensions in the search space and *p* is the population size. For this work $\lambda=1$. Each niche has a number of additional parameters; the generation and position at which the niche spawned and a list of references to the individuals that are currently members of this niche. Two more radii are defined for each niche; an inner niche radius and an outer niche radius. The purpose of these will become clear later (see [5] for more details), but they are defined as ½ the current niche radius, and 2 times the current niche radius, respectively. In order to determine the niche dynamics, the location of the niche in the previous generation is also stored. An individual is considered to be a member of a niche if it lies within the hypersphere described by the niches midpoint and current niche radius. Individuals *can* be a member of more than one niche.

B - The Initial Generation

The nicheset is initially empty. The first generation of individuals is randomly generated and a new niche is created for each individual with its midpoint centred on that individual and with that individual as its only member. The fitness of the niche is equal to the fitness of the member individual. We then remove redundant niches by performing the following process.

1) Calculate and sort the nearest niche pairset – Each niche is compared to each other niche in the nicheset and the Euclidean distance between the midpoints is stored in a list of nearest niche pairs, along with references to the two niches under comparison.

2) Cycle through the sorted list of niche pairs starting with the closest pair – If the midpoints of a pair of niches lie within the inner niche radius of one another, *and* the Hill-Valley function (see Section II-D) does not indicate that a valley lies between the two points, then delete the niche with lower fitness.

C - The DNC Generational Process

This process is executed before selection is performed in a standard GA.

1) Recalculate the niche members – If any individual is not currently a member of a niche, then a new niche is created with its midpoint centred on that individual, and niche radius calculated using equation 1.

2) Move the midpoints of each niche – Each niche's midpoint is modified by the following equation:

$$\overline{mid_j} = \overline{mid_j} + \frac{\sum_{i=1}^{n_j}(\overline{x_i} - \overline{mid_j})\bullet f_i}{\sum_{i=1}^{n_j} f_i} \quad ind\ x_i \in niche\ j \quad (2)$$

Here, mid_j is the midpoint of niche *j*, n_j is the niche count of niche *j*, x_i is the location of individual *i*, and f_i is the fitness of individual *i*.

3) Calculate and sort the nearest niche pairset

4) Cycle through the sorted list of nearest niche pairs starting with the closest pair – For a given pair of niches, if the midpoint of either niche lies inside the inner niche radius of the other niche *and* the Hill-Valley function (see Section II-D) does not indicate that a valley lies between the two midpoints, then the two niches are merged together into a single niche. See [5] for full details of the *merge* process.

5) Cycle through the niches in the nicheset – If any niche has a population size greater than 10% of the total population size, check a number of random pairs of individuals from within the niche using the Hill-Valley function (see Section II-D). If a valley is detected, then the niche is split into 2 new niches. See [5] for full details of the *split* process.

6) Apply the sharing function – The fitness of each individual is divided by the extended triangular sharing function of the niche to which it is a member. In the case where an individual is a member of more than one niche, its fitness is only divided by the niche with greater value sharing function, m_i:

$$m_i = n_j - n_j \bullet \left(\frac{d_{ij}}{2\sigma_{sh_j}}\right) \quad if\ \ ind\ i \in niche\ j \quad (3)$$

Here, d_{ij} is the Euclidean distance between individual *i* and the midpoint of niche *j*, n_j is the niche count of niche *j*, σ_{shj} is the niche radius of niche *j*, and m_i is the resultant sharing function for individual *i*.

D - The Hill-Valley Function

The Hill-Valley function is a fitness topology function that was adapted from [8] and initially used in DNC in [5]. It is an analytical decision tool that is used to indicate if a significant valley lies between two points in the fitness landscape. We have modified it in the interim to also indicate if a significant peak lies between the two points. The operation is simple; given two endpoints in Euclidean space, generate a line segment that intersects them both. Then, choose a number of points along the line segment and calculate the fitness at those points. For a peak to be significant, the fitness of an interpolated point must be greater than the endpoint with higher fitness. The converse is true for a valley to be significant. If a valley is detected, return the difference in fitness between the lowest interpolated point and the endpoint

0-7803-7282-4/02/$10.00 ©2002 IEEE

with least fitness (this will be negative). If a peak is detected, return the difference in fitness between the highest interpolated point and the endpoint with highest fitness (this will be positive). In the case where both a peak and a valley are detected between the two endpoints, always return the depth of the valley.

III. Some Issues with DNC

When two niches are merged, a new single midpoint is calculated based on the individuals within the original niches. It is possible for the new midpoint to be moved a significantly large enough distance to bring the new midpoint within merging range of another niche. These two niches will then be merged and this can move the new midpoint to within merge range of another niche. These will then be merged, which can move the new niche to within merge range of another niche, and so on. Thus, it is possible for a single niche to absorb many smaller niches and become excessively large. In order to prevent this, we restrict merging from taking place if a niche that has already been merged in this generation has moved further than its inner niche radius at the start of this generation. This allows several niches that are directly on top of one another to be merged together, but stops the uncontrolled growth of *'run-away'* niches. Table 1 shows the effects of restricted merges on the number of niches per generation on De Jong F5 (Shekel's Foxholes, see [6] for a definition) with a population size of 300. It is clear that the restricted merge slows down the convergence of niches. The *'pre'* column shows the effects of removing redundant niches in the initial generation (see Section II-B).

Table 1. Effects of merge restriction on #niches per generation

Gen (t)	pre	0	1	2	3	4	5	6	7
DNC	180	106	56	33	30	28	30	26	30
restricted merge DNC	180	153	91	54	34	31	29	30	26

One unfortunate side-effect of employing distance based fitness sharing within DNC is its inability to correctly deal with ridges and plateaus within the fitness landscape (this is due to the assumption that the peaks are hyperspherical in shape). In Figure 1, a simple 1-dimensional ridge is shown along with two covering niches. The solid line represents the actual raw fitness. The dotted line represents the shared fitness on the peak (using the extended triangular sharing function, see Section II-C for details). The shared fitness is essentially what the GA sees once sharing has been applied. As is clearly visible, there are so-called phantom peaks within a ridge region which should be flat. This leads to the population migrating from the comparatively lower regions of fitness at the centre of the niches, to the phantom peaks where the fitness is higher. The niches, in turn, track the population to these phantom peaks, which creates new phantom peaks that the population in the next generation will go on to occupy, and so on. So with ridges and plateaus we get the constant migration of individuals and niches, and hence relatively poor performance on fitness landscapes containing these features.

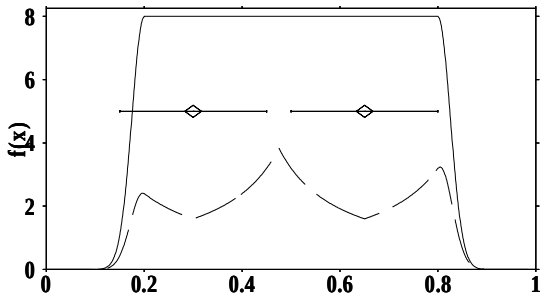

Figure 1. Phantom Peak Formation on a Simple Ridge Function

IV. Niche Linkage

We present here our proposed solution to the ridge and plateau problem by a mechanism we have termed *Niche Linkage*. Fundamentally, *Niche Linkage* is the action of linking two or more overlapping niches with a line segment between the midpoints, and then treating the linked niches as a single niche. In this way it is possible to create a niche of any shape that can be approximated in a piecewise linear fashion. This concept is very similar in principle to the Agglomerative-Partitional Clustering (APC) mechanism employed in [9]. Here, the authors take the clusters discovered by a classical clustering method over a static data set, and then connect those centroids that coexist within regions of relatively high density. The *Niche Linkage* mechanism is subtly different in operation and approach, in that DNC is a continuous process that must operate over a dynamic population of individuals in a GA. So not only must we consider when to link two niches, but also when to sever those links. We must also consider how sharing is applied to the individuals within linked niches.

A niche may be linked to more than one niche. Two niches are termed to be *immediately* linked if a link exists between them. Two niches are described as *directly* linked if a route from one niche can be traced via immediate links to the other niche.

A - When and How to Link

In the APC mechanism described in [9], the decision to connect two centroids was made based on an estimation of the density of observations lying within a fixed distance from the line segment between the two centroids. For APC this estimation is made only once. *Niche Linkage*, on the other hand must check whether to link pairs of niches in every generation. To perform density estimations between pairs of niches in every generation would be computationally very expensive. So, to simplify the link decision, we only consider those niches that overlap to some degree. Figure 2 shows two suitable candidate niches in a 2-dimensional space. Here, x_{ij} indicates the amount of overlap between the two niches, *i* and *j*. The inner circles represent the inner niche radii. Given that $x_{ij} >= 0$, if the Hill-Valley function indicates that neither a peak nor a valley lies between the two midpoints, then the niches are linked. This decision to link is made in step 4) of the Generational Process (see Section II-C) after all merges have been completed.

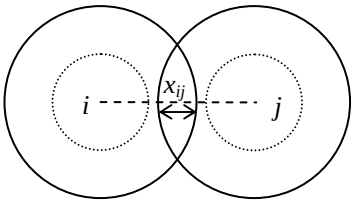

Figure 2. Minimum Niche Overlap for Niche Linkage

B - When and How to Sever Links

Links can be severed in a number of different ways. If a niche that is linked is deleted from the nicheset, *all* the immediate links between that niche and any other niche are immediately severed. If a linked niche is merged with another niche, then all the immediate links from the two original niches must be checked again for overlap and the presence of peaks or valleys. If either test fails, the link is severed. If a linked niche is split into two new niches by the *split* process, then all the immediate links from the original niche are severed. Finally, in the Generational Process step 5) (see Section II-C) where we check each niche in the nicheset, a further step is added. If a niche is linked, then each of the immediate links from that niche must be retested for overlap and the presence of peaks or valleys. Any links that fail these tests are severed.

C - Sharing in Linked Niches

The use of the triangular sharing function (see Section II-C) requires the calculation of the distance between the individual in question and the covering niches midpoint. However, with linked niches, there is no one single midpoint. So, if an individual is a member of a linked niche, the sharing function, m_i, is calculated as follows:

$$m_i = n_j \quad where\ ind.i \in niche\ j \tag{4}$$

Here, n_j is the niche count of niche j. So, the sharing function is the number of individuals within the niche. The use of a rectangular sharing is driven by the need to spread the individuals evenly over the ridge or plateau.

V. Niche Dynamics

There are a number of different dynamic quantities that can be determined within niches. The most obvious, and perhaps the most useful, is the rate of change of position of a niche's midpoint. This can be calculated as the Euclidean distance of the change in location from a niche's midpoint at the end of the previous generation, to the niche's final location after DNC in the current generation, i.e.

$$\frac{\Delta mid_j}{\Delta t} = \left\| mid_j(t) - mid_j(t-1) \right\| \tag{5}$$

Because niches are persistent, that is they are retained from one generation to the next, they will track a converging population, and as such, Δmid_j is a very good indicator for overall population convergence. Another good indicator is the number of niches per generation. There are some other dynamics that can be useful, for example, the rate of change of niche fitness, Δf_j, and the rate of change of niche count, Δn_j. Table 2 shows the average values of niche dynamics for DNC over De Jong F5 with population size 300.

Table 2. Niche Dynamics on De Jong F5

Generation	# niches	Δmid_j	Δf_j	Δn_j
0	153	5.72	-4.57	5.84
1	91	5.95	168.29	2.63
2	54	3.94	92.43	0.11
3	34	2.84	41.18	-1.08
4	31	1.34	26.97	-1.35
5	29	2.54	6.91	-1.24
6	30	0.88	30.51	0.46
7	26	0.41	6.31	0.15
8	32	0.4	9.93	-0.28
9	27	0.36	-9.77	0.26

Here, it is clear that as the population converges onto the peaks both the number of niches and the dynamics start to stabilise. Once converged, there is little movement of the niches, as can be seen from the low value of Δmid_j. The number of niches also remains stable. The small variations in the later generations can be attributed to stochastic sampling errors with the GA.

VI. Test Functions

We describe here some simple 2D test functions that visibly demonstrate *Niche Linkage*'s ability to create niches of arbitrary shape and successfully identify ridges and plateaus within a fitness landscape.

$$g(d,r) = \begin{cases} 1 - \dfrac{2d^2}{r^2} & if\ d < \dfrac{r}{2} \\ \dfrac{2(d-r)^2}{r^2} & if\ \dfrac{r}{2} \le d < r \\ 0 & otherwise \end{cases} \tag{6}$$

$$Let\ \bar{v} = P_1 - P_0\ and\ \bar{w} = P - P_0$$

$$f_1(P, P_0, P_1, w) = \begin{cases} g\left(\left\|\bar{w}\right\|, w\right) & if\ \bar{w} \bullet \bar{v} \le 0 \\ g\left(\left\|P - P_1\right\|, w\right) & if\ \bar{v} \bullet \bar{v} \le \bar{w} \bullet \bar{v} \\ g\left(\left\| P - \left(P_0 + \dfrac{\bar{w} \bullet \bar{v}}{\bar{v} \bullet \bar{v}} \bar{v} \right) \right\|, w\right) & o/wise \end{cases} \tag{7}$$

$$f_2(x_i) = \begin{cases} 0 & if\ 10 - \sum_1^2 (3x_i)^4 \le 0 \\ 10 - \sum_1^2 (3x_i)^4 & otherwise \end{cases} \qquad x_i : [-1..1] \tag{8}$$

$$f_3(x,y) = \begin{cases} g(\sqrt{x^2 + (y-\pi)^2}, 2) & if\ y > \pi \\ g(\sqrt{x^2 + (y+\pi)^2}, 2) & if\ y < -\pi \\ g(|x - \pi\sin(y)|, 2) & if\ x - 2 < \pi\sin(y)\ \&\ x + 2 > \pi\sin(y) \\ 0 & otherwise \end{cases} \tag{9}$$

$$f_4(x,y) = 0.5 - \frac{(\sin(\sqrt{x^2+y^2}))^2 - 0.5}{(1+0.001(x^2+y^2))^2} \quad x,y:[-2\pi..2\pi] \quad (10)$$

g(d,r) describes a bell-shaped curve, where *d* is the Euclidean distance from the middle of the peak, and r is the radius of the peak. The curve is shown in Figure 3a, with *r=0.1*. Function f_1 describes a simple ridge with a bell-shaped cross-section of width *w* (described by *g(d,r)*), starting at point P_0 and ending at point P_1. For this paper, P_0*=(0.25,0.25)* and P_1*=(0.75,0.75)*, so the ridge is oriented along the *y=x* axis.. Function f_2 describes a simple plateau in the middle of the search space. Function f_3 describes an *'S'* shaped ridge with a bell-shaped cross-section of width, *w* (described by *g(d,r)*). Function f_4 is the Davis function [1]. The four test functions are shown in Figure 3b-e.

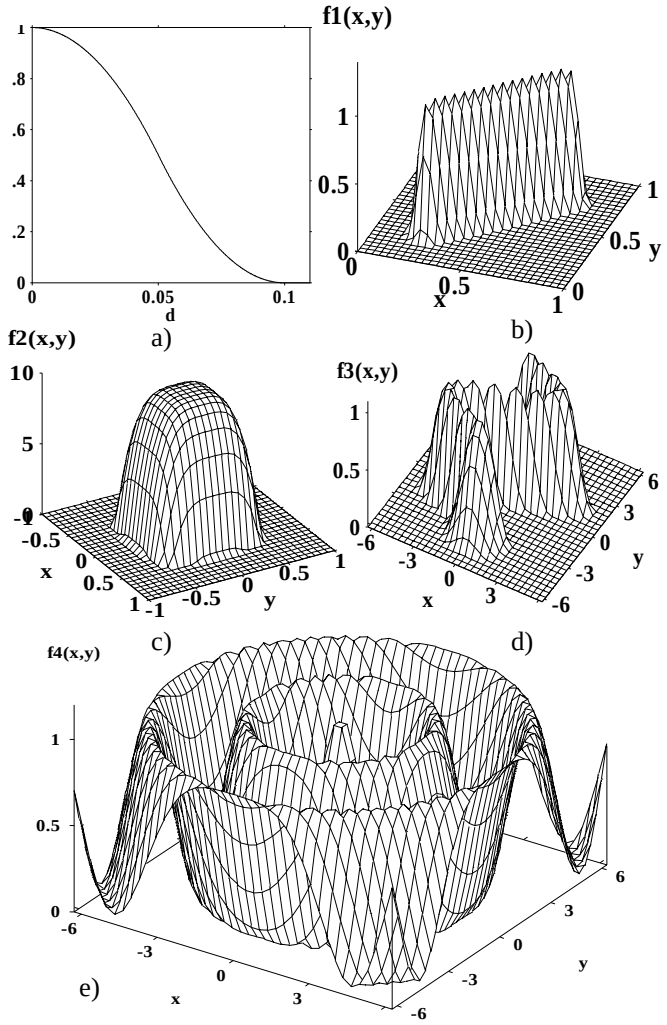

Figure 3. Ridge cross-section and Test Functions f_1-f_4

VII. RESULTS

We provide comparisons of basic DNC as reported in [5], and DNC with *Niche Linkage* as described in this paper. We also include the time results of standard fitness sharing, as described in [6], as a reference. The GA parameters used were as follows; p_m*=0.03*, p_c*=0.7* with 1 point crossover, population size for f_1-f_3 was 300, and 1000 for f_4, graycoding was used with 20 bits per parameter, and each run was for 50 generations with G_{gap}*=1*. The remainder stochastic sampling with replacement selection scheme was used with no elitism.

In order to provide a meaningful performance metric we considered the ratio of the area of the ridges covered by linked niches, and the actual area of the ridge. The closer this value is to 1.0, the more ideal the niche coverage. We also include the number of linked niches and the values of niche dynamics in the final generation along with the time taken to process 50 generations. All values are averaged over 20 different GA runs.

Table 3. Performance Comparison

	DNC with Niche Linkage				DNC without Niche Linkage			
	f_1	f_2	f_3	f_4	f_1	f_2	f_3	f_4
Ratio	1.017	0.975	1.15	0.865	0.96	0.925	0.089	0.758
#Linked	1	1	1.1	9.3	-	-	-	-
Δ#niches	3.1	2.1	4.5	16.5	3.3	1.2	7.7	18.2
Δmid$_j$	0.017	0.025	0.19	0.28	0.029	0.047	0.28	0.32

Table 4. Time Comparison (seconds)

	f_1	f_2	f_3	f_4
DNC with Niche Linkage	23.6	21.6	36.7	2114
DNC without Niche Linkage	21.1	20	30.4	1874
Fitness Sharing	468.9	458.9	471.3	6291

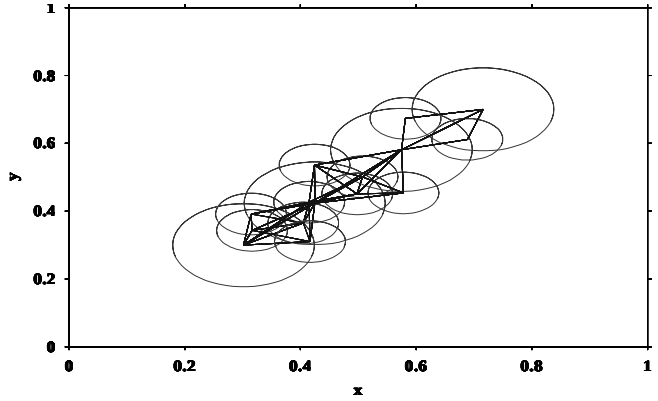

Figure 4. Nicheset of Generation 50 on f_1

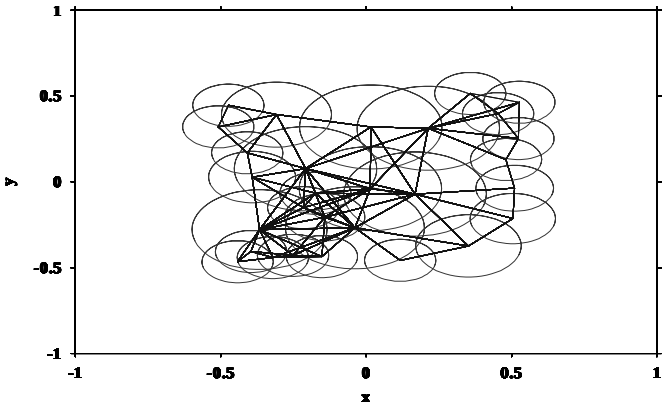

Figure 5. Nicheset of Generation 50 on f_2

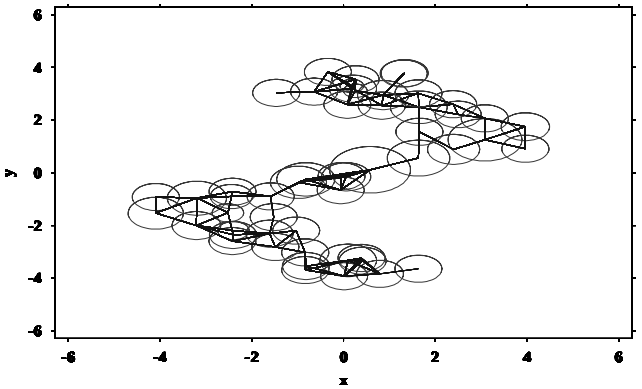

Figure 6. Nicheset of Generation 50 on f_3

As is clear from Table 3, DNC with *Niche Linkage* consistently outperforms DNC on all four test functions. It is also evident from the niche dynamics data that without *Niche Linkage*, DNC suffers from the migrating phantom peaks problem described in Section III. This is suggested by the higher value of Δmid_j, indicating that there is significant movement of niches even though the low value of Δ*#niche* shows that the number of niches is stable. Table 4 shows that there is comparatively little additional computational expense incurred for including *Niche Linkage* in DNC. It also shows that even with population sizes of 1000, DNC is an order of magnitude faster than standard fitness sharing.

Figures 4, 5 and 6 show the final nicheset in generation 50 for functions f_1, f_2 and f_3 respectively. Niches are shown as circles and links between niches are shown by line segments between their midpoints. Only those niches that are directly linked are shown. In Figure 4, the ridge feature aligned along the $y=x$ axis is clearly identified by a single multiply-linked niche. In Figure 5, the plateau is identified by the interconnected mesh of niches in the centre of the search space. In Figure 6, the '*S*' shaped curve is also clearly identified by a single multiply-linked niche, however there are a number of redundant niches present.

VIII. CONCLUSION

In this paper, we have shown that the introduction of *Niche Linkage* frees DNC of the assumption that the peaks must be hyperspherical in shape, which in turn allows DNC to create niches of arbitrary shape and therefore more accurately model the fitness landscape. This provides improved feature extraction and the ability to identify ridges and plateaus within the search space. We have also shown that the computational cost of adding *Niche Linkage* to DNC is relatively small, especially when compared to the cost of using standard fitness sharing. However, *Niche Linkage* has yet to be tested on high-dimensional functions or problems where the ridges or plateaus are of non-equal fitness.

The use of the Hill-Valley function is critical for DNC to operate properly. However, there are some drawbacks to employing it. The Hill-Valley function only works in Euclidean space and is highly sensitive to noise. A noise spike can lead the Hill-Valley function to report a valley lying between two points when in fact it is just noise. This can lead to incorrect decisions being made with regards to the merging, linking and splitting of niches. A further problem is that the interpolated samples are always the same. So if the two endpoints have not changed from one generation to the next, no new information is gathered with regards to the presence of valleys.

We also introduced the concept of *niche dynamics* as an indicator of overall population convergence. Because there is no direct link from the individuals in one generation to the population of a subsequent generation, there is no simple way to determine whether a population has converged. Due to the fact that niches are persistent from one generation to the next and will reliably track individuals on optima, it is very simple to determine population convergence based on the niche dynamics. There are many potential uses for niche dynamics, both as a convergence measure and as an additional decision tool. For example, once the nicheset has converged the amount of exploration could be increased by dynamically increasing the population size. A further use of *niche dynamics* is at a local level. Once a niche has converged, an analysis of the contents of the niche could be made. If the spread of individuals within the niche is very small and confined to within the inner niche radius, the niche may actually be too large. A gradual reduction of that niche's niche radius could then produce a much more accurate model of the peak. This would, in turn, allow for the creation of a truly dynamic, self-tuning, niching evolutionary algorithm.

REFERENCES

[1] Davis, L. (ed.) "Handbook of Genetic Algorithms", Van Nostrand Reinhold, New York, 1991

[2] Deb, K. & Goldberg, D. "An Investigation of Niche and Species Formation in Genetic Optimization", In Proc. 3rd Inter. Conf. on Genetic Algorithms, pp42-50, 1989

[3] Gan, J. & Warwick, K. "Dynamic Niche Clustering: A Fuzzy Variable Radius Niching Technique for Multimodal Optimisation in GAs", In Proc. of Congress on Evolutionary Computation (CEC), Vol 1, pp215-222, 2001

[4] Gan, J. & Warwick, K. "A Variable Radius Niching Technique for Speciation in Genetic Algorithms", in Proc. Genetic and Evolutionary Computational COnference (GECCO), pp96-103, Morgan-Kaufmann, 2000

[5] Gan, J. & Warwick, K. "A Genetic Algorithm with Dynamic Niche Clustering for Multimodal Function Optimisation", in Proc. 4th Inter. Conf. on Artificial Neural Networks and Genetic Algorithms (ICANNGA), pp248-255, Springer-Wien New York, 1999

[6] Goldberg, D. "Genetic Algorithms, Search, Optimization & Machine Learning", Addison-Wesley, 1989

[7] Goldberg, D. & Richardson, J. "Genetic Algorithms with Sharing for Multimodal Function Optimization", In Proc. 2nd Inter. Conf. on Genetic Algorithms, pp41-49, 1987

[8] Ursem, R. "Multi-national Evolutionary Algorithms", In Proc. of Congress on Evolutionary Computation (CEC), Vol 3, pp1633-1640, 1999

[9] Tyree, E.W. & Long, J.A. "Modelling Clusters of Arbitrary Shape with Agglomerative Partitional Clustering", In Proc. 1st Inter. Workshop on Statistical Techniques in Pattern Recognition, Prague, 1997