2008 IEEE Swarm Intelligence Symposium
St. Louis MO USA, September 21-23, 2008

Particle Swarm Optimization with Spatially Meaningful Neighbours

James Lane, Andries Engelbrecht and James Gain

Abstract—**Neighbourhood topologies in particle swarm optimization (PSO) are typically random in terms of the spatial positions of connected neighbours. This study explores the use of spatially meaningful neighbours for PSO. An approach is designed which uses heuristics to leverage the natural neighbours computed with Delaunay triangulation. The approach is compared to standard PSO sociometries and fitness distance ratio approaches. Although intrinsic properties of Delaunay triangulation limit the practical application of this approach to low dimensions results show that it is a successful particle swarm optimizer.**

Index Terms—**Delaunay Triangulation, Neighbour Topology, Particle Swarm Optimization, Heuristics.**

I. Introduction

PARTICLE swarm optimization is a powerful, yet simple population based optimization strategy, particularly well suited for finding extrema in continuous non-linear functions [1]. The approach is derived in part from the interesting way flocks of birds and swarms in nature search for food. Kennedy and Eberhart, developed the approach by streamlining and adapting a simulation of flocking birds in 1995 [1].

In PSO a set of particles find an optimum through an iterative process in which particles sample a search space and then adjust their search directions to sample near to their fitter neighbours. Neighbours are those particles which can share information. The set of neighbour-connections between all of the particles forms the swarm's topology or sociometry [2] and affects the swarms exploitation and exploration behavior [3].

In standard PSO topologies there is no spatial significance between neighbouring particles as neighbours are random in terms of their relative positions. Neighbourhoods are also typically static, being computed once-off during initialization. This contributes to the standard PSO being a fast and simple high dimensional optimizer. Spatially meaningful topologies, on the other hand, have the additional overhead of computing neighbours, though they do present some significant advantages:

1) "Near neighbour interactions" introduce diversity in the Fitness Distance Ratio (FDR) PSO through recombination of nearby particles. This is helpful for avoiding premature convergence [4].
2) Sub-groups of particles near each other are able to find and explore multiple local peaks in multimodal problems, as demonstrated by the Fitness Euclidean Ratio (FER) PSO [5].
3) Dynamic neighbour connections are beneficial for introducing diversity [6].
4) Dynamic topologies are useful for tackling multiobjective optimization problems [7].
5) Spatial neighborhoods facilitate the formation of niches [8].

Manuscript received May 28, 2008; revised July 19, 2008. This work was supported in part by Edwin Blake's NRF Grantholders Bursary FA2007042600027.

James Lane and James Gain are with the Department of Computer Science at the University of Cape Town, and Andries Engelbrecht with the Department of Computer Science at the University of Pretoria, South Africa (email: {jlane, jgain}@cs.uct.ac.za, engel@cs.up.ac.za).

Current spatial approaches require quadratic time to find neighbours [5]. Delaunay triangulation (DT) presents a means of spatially subdividing a set of points in expected near linear time in low dimensions, 2D and 3D [9]. This research explores the use of Delaunay triangulation to achieve a spatial topology, by computing the closest surrounding neighbours for each particle. Our approach uses spatially meaningful heuristics to leverage the set of local Delaunay neighbours to explore diversely, work more immediately on common optima and as a swarm converge on the global best position. Our contributions include:

- researching Delaunay triangulation as a spatial sociometry in PSO and comparing it to standard approaches and other spatial approaches (FER and FDR PSO) in low dimensions (2D, 3D and 4D),
- heuristics which leverage Delaunay neighbours for accomplishing diversity, local exploitation and global convergence,
- a new low-dimensional dynamic-spatial PSO with directed connections and
- a classification schema for PSO sociometries.

A synopsis of DT is given next including a background of PSO with a focus on neighbourhood topologies, a classification schema for sociometries and related work. Our approach is presented in section III and results in section IV. Technicalities, limitations and application areas are then discussed. Conclusions are drawn and future work suggested.

II. Background

A. Delaunay Triangulation

Delaunay triangulation spatially sub-divides a set of points into triangles in 2D (tetrahedra in 3D and simplices in 4D), where the endpoints of the simplex (an n-dimensional equivalent of a triangle) edges lie on the circumference of the circumcircle (a circle with none of the other points inside it) [10]. Figure 1, Shows an example of a 2D DT. The Delaunay triangulation defines natural neighbours and is a useful spatial data structure for finding the nearest surrounding neighbours of a set of points. Delaunay triangulation has a worst case time complexity of $O(n^{\lceil \frac{d}{2} \rceil + 1})$, where d is the dimension of the points. In practice though, computing DT is significantly faster than this worst case which is experienced

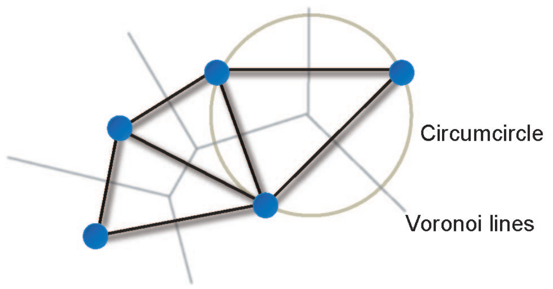

Fig. 1. Delaunay triangulation of a set of points.

for certain manufactured point sets [10]. In 2D the worst case time complexity is $O(n \log n)$ [9]. In 3D it is $O(n^2)$ though "for all practical purposes three-dimensional Delaunay triangulations appear to have linear complexity" [11]. In 4D there are algorithms which compute the DT in $O(n^3)$ [9].

B. Particle Swarm Optimization

Particle swarm optimization is a population based search strategy which finds an optimum by stochastically "flying" a set of particles through a search space. Particles iteratively sample a region between and beyond their own individual prior best position and the position of their most successful neighbour(s). In doing so, fitter positions may be found. Updating their individual best positions, the particles change their search directions to explore these new fitter positions. Through this process the particles converge on the maximum/minimum.

Equation (1) is the commonly used constriction factor velocity update equation for the standard (Canonical) PSO [12]. The equation causes a particle i, to oscillate around its individual best and neighbour best positions, dampening the velocity, and influence of these terms by a constriction factor χ. The velocity update moves a particle to a random position between and beyond its current position ($\mathbf{X}_i$), its previous individual best position ($\mathbf{P}_i$) and its most successful neighbour's best position ($\mathbf{P}_n$):

$$\mathbf{V}_i = \chi[\mathbf{V}_i + c_1\mathbf{r}_1(\mathbf{P}_i - \mathbf{X}_i) + c_2\mathbf{r}_2(\mathbf{P}_n - \mathbf{X}_i)] \quad (1)$$

where $\mathbf{X}_i$ is the particles current position and $\mathbf{V}_i$ is the particles velocity. Components are point-wise multiplied with each other. Typically χ is set to 0.729, in combination with $c_1 = c_2 = 2.05$ [12]. c_1, and c_2 scale the individual and neighbour contributions (which act as attractors) so that the particle searches around them. $\mathbf{r}_1$ and $\mathbf{r}_2$ are tuples of uniform random numbers in the range $[0;1]$, which introduce the stochastic component. A random number is computed for each dimension being point-wise multiplied. Figure 2 illustrates how this equation works. The neighbourhood best positions ($\mathbf{P}_n$) are computed each iteration before the velocity update step by running through the set of particles which comprise each particles neighbourhood and choosing the fittest of these. Individual bests are updated each iteration for each particle if the new position is fitter than the particle's previous best position.

The positions ($\mathbf{X}_i$) of the moving particles form an "explorer-swarm" responsible for exploring the search space.

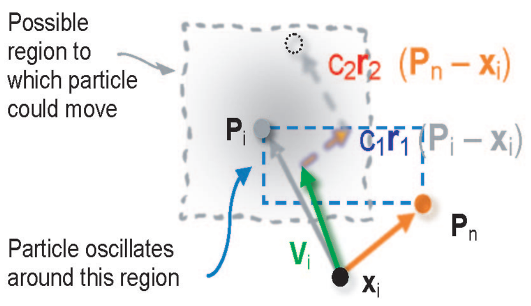
Possible region to which particle could move
$c_2\mathbf{r}_2$ $(\mathbf{P}_n - \mathbf{x}_i)$
$\mathbf{P}_i$
$c_1\mathbf{r}_1$ $(\mathbf{P}_i - \mathbf{x}_i)$
$\mathbf{P}_n$
Particle oscillates around this region
$\mathbf{V}_i$
$\mathbf{x}_i$

Fig. 2. A 2D illustration of the velocity update equation and the region to which it will move a particle. The scaled, shifted and constricted velocity results in a stochastic region to which a particle will move.

The personal bests ($\mathbf{P}_i$) of the particles may be thought of as a "memory-swarm" [5]. The memory swarm is significantly more stable than the explorer swarm, since it consists of the best points found so far by the individual explorer particles which are only updated if better points are found.

C. Neighbourhood Topologies

Figure 3 shows the most common topologies (neighbourhood structures) used by PSO: the star topology in which all particles are connected to all others, the ring neighbourhood for which each particle is connected to two others and the Von Neumann topology where each particle links to four others in a cubic-lattice type arrangement (this is essentially a ring topology but with four neighbours) and on the far right, Delaunay neighbours. Neighbourhood structure affects the

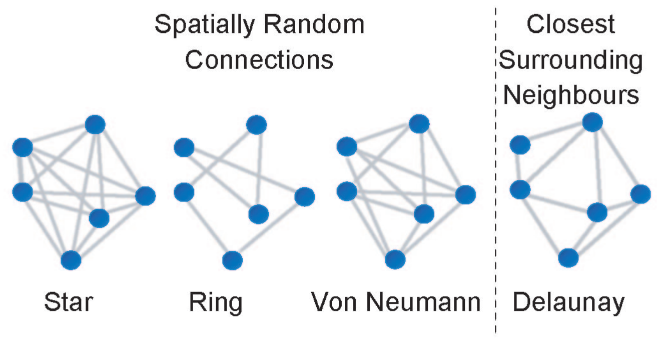

Fig. 3. The standard Random-Static topologies (Star,Ring,Von Neumann) and Spatial meaningful-Dynamic Delaunay neighbourhood structure.

performance and convergence of PSO significantly [13] since it determines the rate at which information propagates through the swarm. This greatly influences the swarm's exploitation and exploration behaviors. For instance, the fully connected star topology exhibits fast convergence with little exploration, best positions and fitness information being relayed directly to the entire swarm. Slow convergence with greater exploration is observed in the ring topology, which has few connected neighbours, since it takes longer (several iterations) for information to pass through the links to the other particles giving the swarm "more time" to explore. This makes a PSO using the ring topology less prone to being trapped in local extrema [14].

Typically, neighbour particles are determined in the ring and Von Neumann topologies simply by using the different particle's indices (particles are connected as neighbours based solely on their array indices). This results in a spatially random topology, since there is no correlation between a particle's position in relation to its neighbour's positions. The randomness in terms of the related spatial layout between neighbours in the ring and Von Neumann topologies juxtaposed to the natural spatial neighbours found by Delaunay triangulation is evident in figure 3. The star, ring and Von Neumann topologies are static in that their neighbour connections are set at initialization and do not change throughout the search, even if the particles change position in relation to each other. Static topologies have minimal computational overhead since they do not require re-computation and only a single linear pass is needed to update neighbourhood bests.

D. Classifying Topologies

Figure 4 summarizes and illustrates classification criteria for PSO topologies. A topology structure may be static (neighbours remain fixed throughout iterations) or dynamic. Neighbouring particles are either spatially related or random in terms of their spatial layout. Spatial neighbourhoods are inherently dynamic because particles moving in relation to one another may move past each other or closer to other particles, resulting in topology changes. Another defining characteristic of a topology is whether the inter-particle connections are directed or undirected. Directed topologies allow one way information sharing, i.e. A→B means A can access B's information but not vice versa. The above classifications are helpful for logically

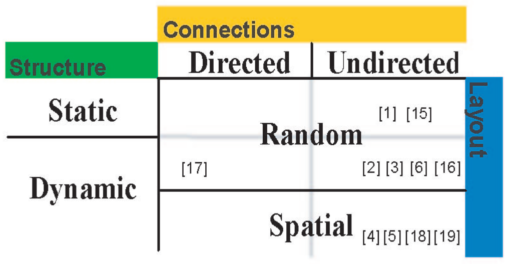

Fig. 4. Classification of PSO Topologies.

organizing and categorizing the vast related literature and approaches to neighbourhood structures in PSO. Our approach is an example of using a dynamic-spatial neighbourhood with directed connections.

E. Related Work

Static random topologies with undirected connections such as the star and ring neighbourhoods are the most commonly used in PSO implementations [6]. The Von Neumann topology has shown exceptional performance in the fully informed particle swarm (FIPS) PSO [15]. Directed connections have also been used with these static-random topologies. Experiments with random static topologies include the use of discrete random undirected graphs and acyclic random links [16][17].

Dynamic random topologies for both directed and undirected connections include variations such as: randomly increasing the number of undirected neighbour connections with successive iterations (moving the swarm from a state of exploration to one of exploitation) [2][18], randomly changing unconnected neighbours [14] and using random discrete structures and edge migrations for directed connections [17]. Experiments with different aspects of neighbourhoods and network connections including effects of out degree and the size of the population have been performed to help determine the properties of topologies that make for successful societies [14][6][3].

Dynamic spatial topologies in PSO are rare. Most likely because computing neighbours is an additional overhead and Euclidean distance is computationally expensive [18][14]. Examples of spatial neighbourhood approaches include: increasing the number of connected closest neighbours [18] and forming fully connected "clusters" after iterations based on particles search-space locations. [19]. The FDR (Fitness Distance Ratio) PSO computes a best neighbour position for each particle in the swarm by maximizing the ratio between the fitness difference of each particle for each dimension and the absolute value of the difference between the particles position in that dimension [4]. The Fitness Euclidean Ratio (FER) PSO [5] is a modification of the FDR approach that uses the Euclidean distance and memory swarm for the purpose of finding multiple extrema in multimodal problems. The FDR and FER are spatial approaches which parse the entire swarm for each particle when computing best neighbours whereas our approach uses the Delaunay neighbours and heuristics. The use of Delaunay triangulation to compute and maintain spatially meaningful neighbours is quite unlike current approaches and is to our knowledge the first time that spatial data structures are used to compute and manage neighbours for PSO.

Several miscellaneous spatial extensions have been proposed for PSO including collision avoidance [20], a spatial extension which causes particles to bounce off each other to avoid clustering [21]. Richards and Ventura [22] have used centroidal Voronoi tessellation for generating initial starting points for a swarm but do not use tessellation during the actual search.

III. Natural Neighbours Approach

The approach described below uses DT and heuristics to leverage near neighbours to work together on nearby common extrema. The heuristics and spanning property of DT are used to cause the swarm to progressively converge on the global extremum.

A. Finding Neighbours Using Delaunay Triangulation

DT is used as a first step in our approach to find a subset of closest surrounding neighbours for each particle. The Delaunay neighbours connect particles across the swarm so that each particle is either connected indirectly by a path through some set of other particles or directly to every other particle in the swarm. This is necessary, since at some point particles must be influenced by the global best for the swarm to ultimately converge upon it. DT plays a role in distributing the search in a spatially meaningful way by dividing space into Voronoi(the dual of DT) cells between particle positions in either the explorer or memory swarm. This is advantageous for exploration because it slows convergence on the global best, when there are sufficient particles and hence divisions through which information has to travel. Since Delaunay neighbours are the closest surrounding neighbours this means they may more immediately search local regions of the search space with other nearby neighbouring particles than random neighbours could.

The set of neighbours DT provides is merely a point of departure for our approach, since using all of the Delaunay neighbours may result in a nearly fully connected swarm which could lead to particles converging too quickly on a local optimum. Figure 5 illustrates this problem in which the DT neighbours form a topology very similar to a star topology. The particles in the illustration will be drawn into the center (local extremum) in the next iteration before the particles have a chance to explore their own local regions, causing the swarm to miss the global optimum. Particle $k(P_k)$, which is very close to the global optimum needs some time or help to search locally. A heuristic is required to meaningful break connections.

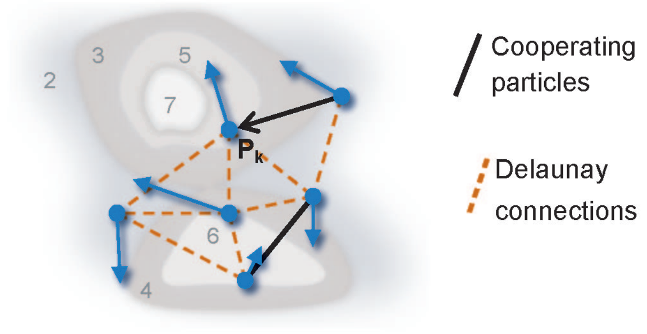

Fig. 5. Contour map with Delaunay neighbours forming an almost fully connected topology.

B. Dynamic Connections and Heuristics

Dynamic connections present a means of introducing diversity and are used to overcome the problem of over-connection encountered when using all of the Delaunay neighbours. A sociometry composed of natural neighbours undergirds a framework (spatial context) which allows for the design of meaningful dynamism. Our rules for choosing connections aim to select neighbours from among the set of natural neighbours to search together locally in common spatial regions near to each other and yet ultimately tend towards the global optimum. Spatially meaningful heuristics are used to accomplish this by modulating connections.

1) Choosing Locally Cooperating Neighbours: Given a set of Delaunay neighbours, only the connections between particles which are cooperating to find a common local optimum are desired. The following rules are used to decide which particles are working together:

1) if a particle $\mathbf{P}_1$ is following behind another particle $\mathbf{P}_2$ then a directed connection is made from $\mathbf{P}_1$ to $\mathbf{P}_2$. This represents particles heading in the same general direction for which the trailing particle is connected to the leading one. Figure 6(left) illustrates this case.
2) If two particles, $\mathbf{P}_1$ and $\mathbf{P}_2$, are heading towards each other (but not past one another) they are considered to be cooperating and an undirected connection is made between the two. This case is shown in Figure 6 (right).

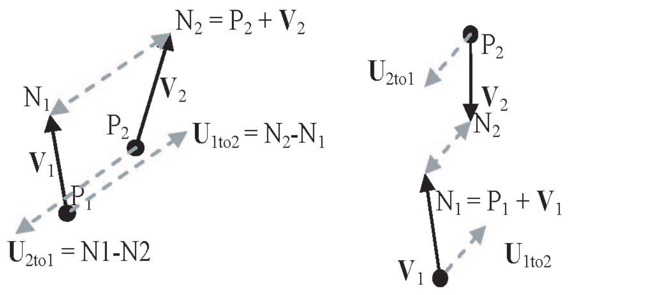

Fig. 6. A particle is connected to a neighbouring particle if it is following or heading towards its neighbour. This is the case when $\mathbf{V}_1 \cdot \mathbf{U}_{1to2} > 0$.

These two heuristics are implemented by testing when:

$$\mathbf{V}_1 \cdot \mathbf{U}_{1to2} > 0 \tag{2}$$

In equation (2) $\mathbf{V}_1$ is $\mathbf{P}_1$'s velocity and $\mathbf{U}_{1to2}$ is the offset vector from $\mathbf{P}_1$ to $\mathbf{P}_2$ after a move (velocity update). Similarly, this rule may be applied to test if $\mathbf{P}_2$ is working with $\mathbf{P}_1$. The black lines in figure 5 illustrate the subset of Delaunay neighbours that these heuristics would choose.

Another meaningful heuristic for maintaining connections between cooperating particles, is described immediately below: Figure 7 shows the stochastic region of overlap for two neighbouring particles. If this region is significantly greater than a selected percentage threshold of the two combined regions of motion, then the neighbouring particles are highly likely to be working together in the same region. An undirected connection is maintained between these neighbours in this case. Alternatively the region between a particle's personal best and neighbour best, around which a particle oscillates, may be used in this test, see figure 2. In our experiments the stochastic region was used rather than the region of oscillation. These heuristics reduce and vary the connections in

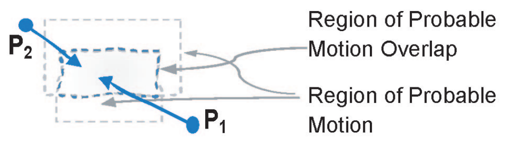

Fig. 7. Stochastic region of overlap.

the swarm. After their application there may be particles with no connections. Such particles are connected to the closest fittest neighbour amongst their original set of Delaunay neighbours so that no particles are left unconnected. Alternatively, unconnected particles may be left to perform hill-climbing in their immediate region.

2) Local Exploitation: Another useful spatial heuristic is to attract particles to their “closest-fitter" neighbour. We aim to cause particles nearby one another to work together towards their closest peak, rather than their fittest peak. This slows the rate at which the global best is passed through the swarm and presents a way of getting local particles to work together to improve a solution in their local vicinity. Figure 8 illustrates this: $\mathbf{P}_3$ will move towards "closer fitter" particle $\mathbf{P}_4$ working locally with it, rather than being drawn away to a more distant peak by $\mathbf{P}_2$, even though this is the fittest neighbour. $\mathbf{P}_2$ and $\mathbf{P}_1$ are responsible for exploring their common local peak. This heuristic takes advantage of particles being spread across space with interleaved sections between them. However, if there are many particles in the swarm and a rugged function landscape, this rule may slow the rate of convergence on the global optimum (more iterations will be required to find the global best).

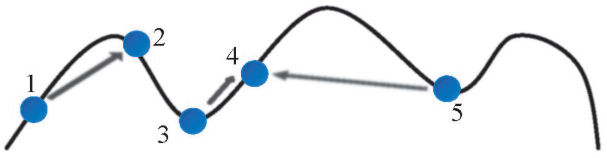

Fig. 8. The "closest-fitter" heuristic will draw P3 towards P4 even though P2 is P3’s fittest neighbour.

3) Convergence on the Global Best: Though particles should investigate local extrema, they must ultimately progress towards the global optimum. A meaningful measure for deciding when to pull a particle away from a local peak is the ratio of the distances between the "closest-fitter" and "fittest" neighbours. It is also a measure of how well a peak has been exploited. This is because particles which converge locally on their "closest-fitter" neighbour, exploiting a local peak, will get closer and closer to each other. This distance will become significantly smaller than the distance to the local fittest particle in cases where a fittest neighbour is on a different higher neighbouring peak. Figure 9 illustrates this. Particles $\mathbf{P}_1$ and $\mathbf{P}_2$ will converge on each other. As they do, the distance to $\mathbf{P}_2$’s closest fitter neighbour becomes significantly small in relation to its distance to $\mathbf{P}_3$, its fittest neighbour. Incorporating the swarm’s diameter into this test

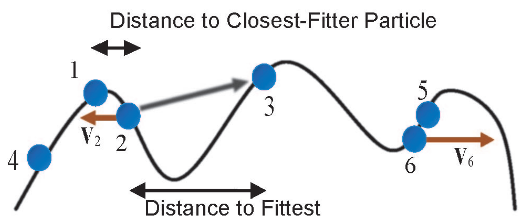

Fig. 9. The distance between closest fitter particles becomes increasingly small in relation to the distance to the fittest neighbour if there is a fitter neighbour on a higher peak.

allows particles to dynamically adapt the depth to which they search as the swarm contracts. This is desirable because as the swarm progresses towards the global optimum, peaks should be examined more closely. A local exploitation ratio threshold may be set to some factor of the swarm’s sparseness ($n^{\frac{1}{d}}$, where n is the number of particles and d is the dimension). Alternatively and more simply, the local exploitation ratio threshold may be set to a fraction of the diameter. In our tests we let particles explore to one hundredth of the swarms radius. Additionally we test if the distance to the closest fitter particle is less than the local exploitation ratio. This is also an indicator of a peak being sufficiently exploited.

When the ratio is below the local exploitation ratio threshold the fittest neighbour is used rather than the closest fitter neighbour only if the velocity of the particle is at most twice the distance to the closest neighbour. This is to prevent arbitrary particles which land nearby the local peak from disrupting a local search (see particle $\mathbf{P}_5$ and $\mathbf{P}_6$). Only those particles which are sampling the local peak with a small step size should be allowed to move onto the fittest peak. This rule and the spanning property of the DT (their is some path from every neighbour to every other neighbour in the DT) results in particles at some point converging on the global best particle. The rate at which the particles tend to this point is slowed by all of the rules and the spatial separation between the particles resulting in greater exploration.

C. Integration into the Standard PSO Algorithm

Algorithm 1 Pseudo code for the PSO Natural Neighbours algorithm.

```
Randomly generate initial population
Repeat
    $N$ = compute_delaunay($X_1$ to population_size)
    for $i$ = 1 to population_size do
        if $f(X_i) < f(P_i)$ then
            $P_i$ = $X_i$
        $P_n$ = chooseBestNeighbour( $N_i$ )
        for $d$ = 1 to dimensions do
            velocity_update()
            position_update()
        end
    end
until termination criterion is met.
```

Algorithm 1 shows how the standard PSO algorithm is modified to use natural neighbours and our heuristics. A new step, “compute_delaunay", is added which returns the Delaunay neighbours, N, for the positions, $\mathbf{X}_i$ of the particles, in the swarm. In this work we concentrate on finding the DT of the explorer swarm. An alternative would be to compute the DT of the memory swarm and let explorer points contribute to improving their closest memory swarm points.

Our heuristics are integrated into the "chooseBestNeighbour" procedure, which returns a neighbouring best particle ($\mathbf{P}_n$) for particle i, from i’s set of Delaunay neighbours. Algorithm 2 shows pseudo code for determining the best neighbour using the heuristics. $\mathbf{P}_f$ is the position of the fittest neighbour and $\mathbf{P}_c$ the position of the closest fitter neighbour individual bests are used rather than explorer positions.

Algorithm 2 Pseudo code for finding a particles best neighbours.

```
input: $N_i$ Particle $i$'s closest neighbours
output: $\mathbf{P}_n$ the best neighbour
Procedure chooseBestNeighbour ( $N_i$ )
    hasConnectedNeighbours = false
    $\mathbf{P}_f$ = $min(N_k)$
    for $k$ = 1 to neighbourset_size do
        if $working\_together(\mathbf{P}_i, P_k) and$
            $dist(\mathbf{X}_i - \mathbf{P}_c) < dist(X_i - \mathbf{P}_k) and$
            $f(\mathbf{P}_k) < f(\mathbf{X}_i)$ then
            $\mathbf{P}_c = \mathbf{P}_k$
            hasConnectedNeighbours = true
        end if
    end
    $localExploitationRatio = swarm.diameter/200$
    if $hasConnectedNeighbours$ and
        $distance(\mathbf{X}_i - \mathbf{P}_c)/distance(\mathbf{X}_i - \mathbf{P}_f) >$
        $localExploitationRatio$ and
        $V_i < 2 * distance(\mathbf{X}_i - \mathbf{P}_c)$ then
        $\mathbf{P}_n = \mathbf{P}_c$
    else
        $\mathbf{P}_n = \mathbf{P}_f$
    Return $\mathbf{P}_n$
end Procedure
```

IV. Results

Internal tests comparing DT without heuristics, heuristics with a fully connected swarm and a combination of DT with heuristics showed that DT found solutions using the least amount of iterations but was the least successful at finding the global best. Using heuristics with a fully connected swarm was comparative to DT with heuristics. It found solutions in slightly fewer iterations but performed marginally worse at finding the global extremum (more connections implies faster convergence and less exploration).

The Delaunay approach with heuristics (DTH) was evaluated against the star (GB), Ring (LB2) and Von Neumann (LB4) static topologies as well as the FER and FDR (112) fitness ratio approaches. FDR (112) is used in our experiments, as this was the best performer amongst the FDR variations as reported by Veermachaneni et al [4]. Tests were run on five of the most commonly used benchmark test functions for testing neighbourhood structures [6][13][4], The commonly used sphere function was omitted from our test bed, since it is too simple in low dimensions, approaches always find the global best. Tests were run in 2D, 3D and 4D. Thirty trials were run for each topology on each of the test functions for swarms of size 10, 20 and 30 particles. Trials were terminated after 10000 iterations. Table I shows the functions used, the initialization domain and the terminating criteria. The reader is referred to [14] for a detailed description of these functions. The terminating criterion serves as the finishing-line, it is a value for a specific test function, which if reached indicates that the swarm is on the global peak. All functions were tested in 2D, 3D and 4D except for Schaffer which is a 2D function.

TABLE I
FUNCTIONS, STOP CRITERIA AND DOMAINS

Function	Domain	Criterion
Schaffer	[-100;100]	0.00001
$0.5 + \frac{(sin\sqrt{(x_1^2+x_2^2)})^2-0.5)}{(1+0.001(x_1^2+x_2^2))^2}$		
Rastrigin	[-5.12;5.12]	0.01
$\sum_{i=1}^{n} x_i^2 + 10 - 10cos(2\pi x_i)$		
Rosenbrock	[-30;30]	100
$\sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$		
Griewanck	[-600;600]	0.05
frac14000$\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})x_i + 1$		
Ackley	[-32;32]	0.01
$20 + e - 20e^{-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}} - e^{\frac{\sqrt{\sum_{i=1}^{n} cos(2\pi x_i)}}{n}}$		

"Success rate" indicates the number of times an approach reaches the criteria. It is chosen as the most significant measure for evaluating the approaches, since it shows an approach's ability to find the global extremum [13].

"Number of iterations to reach the criteria" is a significant independent measure of an approaches performance; the median of these values is used for successful trials (see [13]). Table 2 shows success rates and the median number of iterations to success. A -1 indicates that 50% or more of the trials were unsuccessful. In our tests, initial velocities are random with magnitude at most half the search space diameter. We also execute the update of the individual bests before moving particles and after adjusting velocities in order to help maintain variation between individual bests and current position for all the approaches. In 4D, DT computation occasionally fails (possibly due to degenerate point sets) in which case the fully connected neighbour graph is used.

Time tests were performed. The DTH approach, in 2D and 3D took a few seconds longer to find solutions than the other approaches which typically finished in under a second. The approach in 4D depending on the number of iterations-took from a few seconds to several minutes to find solutions. It must be taken into account that the approach was implemented for proof of concept rather than optimized execution speed.

The results in Table 2 show that DTH and LB2 are in terms of success-rate either as good or better than the other approaches, with DTH performing better in 2D on the Schaffer function and LB2 doing the best on Rastrigrin in 3D and 4D for 10 and 20 particles. LB4 and FER are close contenders.

In terms of iterations to success, FDR strangely converges the fastest with GB. This is possibly due to it making velocity updates using not only the global best but also a neighbour best which for low-dimensions is possibly very close to the global best, giving each particle a greater weighting towards the global best than towards its personal best position, hence causing premature convergence. FER is the fastest of the more successful approaches. Depending on the function, DTH and LB2 (the slowest of the approaches) seem to be on par in 3D and 4D with DTH being faster in 2D.

TABLE II
RESULTS - SUCCESS RATE & PERFORMANCE

		Success rate %						Iterations to criterion					
	n	DTH	GB	LB2	LB4	FDR	FER	DTH	GB	LB2	LB4	FDR	FER
		2D						2D					
Schaffer	10	93	20	90	43	23	57	526	-1	2028	-1	-1	496
	20	100	50	93	93	17	97	352	462	802	323	-1	482
	30	100	90	97	97	57	97	273	201	723	245	100	287
Rastrigin	10	100	77	100	97	63	93	78	59	94	75	41	71
	20	100	100	100	100	97	100	71	47	76	57	33	54
	30	100	100	100	100	97	100	64	38	66	54	24	49
Rosenbrock	10	100	100	100	100	100	100	6	5	6	8	5	7
	20	100	100	100	100	100	100	2	3	4	4	3	4
	30	100	100	100	100	100	100	3	2	3	3	3	3
Griewank	10	100	100	100	100	93	100	66	50	70	61	34	68
	20	100	100	100	100	97	100	67	33	67	42	22	30
	30	100	100	100	100	100	100	51	38	39	46	18	37
Ackley	10	100	100	100	100	100	100	60	51	76	67	40	62
	20	100	100	100	100	100	100	52	40	71	58	31	51
	30	100	100	100	100	100	100	45	37	65	51	28	44
		3D						3D					
Rastrigin	10	80	53	87	83	30	63	223	97	214	165	-1	195
	20	97	77	100	100	73	100	224	111	177	148	61	129
	30	100	97	100	100	77	97	204	83	129	114	49	105
Rosenbrock	10	100	100	100	100	100	100	22	20	29	23	14	24
	20	100	100	100	100	100	100	16	12	23	21	11	18
	30	100	100	100	100	100	100	13	10	23	17	10	14
Griewank	10	100	90	100	100	73	100	300	103	161	131	96	178
	20	100	100	100	100	87	100	250	97	154	122	51	132
	30	100	100	100	100	97	100	173	84	124	100	47	122
Ackley	10	100	97	100	100	100	100	92	74	122	92	55	95
	20	100	100	100	100	100	100	76	60	111	86	42	74
	30	100	100	100	100	100	100	70	53	101	81	38	64
		4D						4D					
Rastrigin	10	60	33	67	33	17	40	581	-1	576	-1	-1	-1
	20	100	63	100	80	33	80	606	164	382	216	-1	279
	30	100	73	100	93	50	87	865	163	373	204	72	222
Rosenbrock	10	100	100	100	100	100	100	37	36	53	46	25	41
	20	100	100	100	100	100	100	28	28	45	37	16	31
	30	100	100	100	100	100	100	26	21	40	32	14	26
Griewank	10	97	83	97	93	37	90	490	172	294	442	188	380
	20	100	90	100	100	83	100	815	145	231	235	117	209
	30	100	100	100	100	67	100	347	151	249	200	75	200
Ackley	10	100	93	100	100	100	100	115	98	155	122	68	120
	20	100	100	100	100	100	100	102	77	156	109	52	92
	30	100	100	100	100	100	100	95	69	146	113	45	81

V. Discussion

A. Limitations and Drawbacks

The very Delaunay Triangulation which is so useful for the approach becomes the obstacle to extending it to higher dimensions. The approach is theoretically bound by its worst case time and space complexity, making it computationally practical only for low dimensions. Further computing Delaunay Triangulation in 4D and higher is commonly done by finding the convex hull, which for degenerate point sets can capriciously malfunction if sufficient numeric precision is not used. The CGAL framework [23] used to compute Delaunay triangulations in the implementation of this research proved to be robust and very helpful. It supports LEDA [23], a library of efficient data types and algorithms which handles exact precision computation.

Though it may be possible to use approximate Voronoi diagrams or linear programming (which may be used to find Voronoi cell neighbours rather than compute the exact Voronoi Diagram) to speed up computation of the Delaunay triangulation and extend the approach to higher dimensions, there is another issue: natural neighbours may only be meaningful in higher-dimensions where the number of particles is significant compared to the dimension. As dimension increases for a fixed number of uniformly randomly distributed particles, the particles become increasingly sparse. This means that, for a small set of points as the problem dimensionality increases, the Delaunay Triangulations will become more fully connected tending towards a star topology. For example we counted Delaunay neighbours for ten randomly distributed particles in increasing dimensions: in 2D there were 21 neighbours, in 3D-34, 4D-39, 5D-40 and by 6D the swarm was fully connected with 45 neighbours.

However, any high-dimensional problem may be solved by splitting it into many smaller dimensional problems as is done for the cooperative PSO, provided that there are not interdependencies among the dimensions [24].

B. Faster Neighbours

The time complexity of computing the Delaunay triangulation in low dimensions is $O(n \log n)$ in 2D and 3D. This is an improvement and no worse than the fitness distance ratio methods which are $O(n^2)$ though time tests suggest the comparison is not this straightforward since our approach takes longer (in seconds) per iteration for small numbers of particles. This may be partly due to the approach's heuristic tests which require a pass through all of the neighbour connections.

A kinetic Delaunay data structure[23], could also be used to significantly reduce the number of times the triangulation has to be repaired. Locality is an important ingredient for successful kinetic data structures (geometric data structures designed to cater for motion) which our approach satisfies, with its use of locally constrained motion and the idea of particles working together locally.

Using the DT of the memory swarm, rather than the explorer swarm could also cut computations since the DT would be updated less often and extensively, only when fitter positions are found.

C. Improving the Approach

DT has the potential for implementing dynamic velocity updates: if each particle adjusted its velocity so that it searches within its own Voronoi cell neighbourhood, it could result in a more distributed and adaptive coverage of the search space. Also, as particles converge, neighbourhood regions will naturally contract and particles will slow down, performing a finer search, while particles on the outskirts of the swarm would search more broadly.

VI. Applications

The additional overheads and complexity for computing the DT are likely to preclude the approach to specialized low-dimensional problems such as Mobile Robotics. One of our aims is to use the approach for the scientific visualization of geoscience data (typically 2D or 3D) to find and track multiple extrema. In such applications the additional computational cost of computing the Delaunay triangulation is a non-issue since such spatial data structures often have to be computed in any event. Currently we are using a memory swarm variant to find multiple spatially distributed silhouette points.

VII. Conclusions and Future Work

This research explored using Delaunay neighbours as a spatial topology for PSO. Such a topology on its own results in particles which converge too quickly. The spatial nature of this topology however does facilitate meaningful spatial heuristics which modulate the connections to accomplish local searching, diverse exploration and overall convergence. Our approach is comparatively successful to the Standard Ring and Von Neumann topologies in 2D, 3D and 4D (though significantly slower in 4D). The use of Delaunay Triangulation limits the approach to low dimensions.

Future research should explore ways of leveraging spatial topologies, including the use of FIPS PSO, which may perform even better. Graph spanners may present an alternative to DT for computing a subset of spatial neighbours. This or the use of heuristics on their own may be a way of extending the approach to higher-dimensions. Exploring the use of spatial neighbours for multimodal and dynamic problems may also prove to be fruitful.

Acknowledgments

The author wishes to thank his parents and friends for their prayers and support.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of the IEEE International Joint Conference on Neural Networks, IEEE Press*, vol. 8, no. 3, pp. 1943–1948, 1995.

[2] M. Richards and D. Ventura, "Dynamic sociometry in particle swarm optimization," *Proceedings of the Joint Conference on Information Sciences*, pp. 1557–1560, September 2003.

[3] R. Mendes and J. Neves, "What makes a successful society? experiments with population topologies in particle swarms," in *SBIA*, 2004, pp. 346–355.

[4] K. Veeramachaneni, T. Peram, C. Mohan, and L. Osadciw, "Optimization using particle swarm with near neighbor interactions," *Genetic and Evolutionary Computation Conference*, July 2003.

[5] X. Li, "A multimodal particle swarm optimizer based on fitness euclidean-distance ratio," in *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2007, pp. 78–85.

[6] A. S. Mohais, R. Mendes, C. Ward, and C. Posthoff, "Neighborhood re-structuring in particle swarm optimization." in *Australian Conference on Artificial Intelligence*, ser. Lecture Notes in Computer Science, S. Zhang and R. Jarvis, Eds., vol. 3809. Springer, 2005, pp. 776–785.

[7] X. Hu and R. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," May 2002.

[8] R. Britz, A. Engelbrecht, and F. V. den Bergh, "Locating multiple optima using particle swarm optimization," *Applied Mathematics and Computation*, 2007.

[9] E. Aganj, J.-P. Pons, F. Segonne, and R. Keriven, "Spatio-temporal shape from silhouette using four-dimensional delaunay meshing," *Computer Vision, ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, October 2007.

[10] P. Cignoniz, C. Montaniz, and R. Scopigno, "Dewall a fast divide and conquer delaunay triangulation algorithm in ed," *Computer-Aided Design 30*, pp. 333–341, April 1997.

[11] J. Erickson, "Dense point sets have sparse delaunay triangulations"," in *SODA '02: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2002, pp. 125–134.

[12] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation 6*, pp. 58–73, 2002.

[13] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better." *IEEE Trans. Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[14] R. Mendes, *Population Topologies and Their Influence in Particle Swarm Performance*. University of Minho, April 2004.

[15] J. Kennedy and R. Mendes, "Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms," *Soft Computing in Industrial Applications, Proceedings of the 2003 IEEE International Workshop*, pp. 45–50, June 2003.

[16] ——, "Population structure and particle swarm performance," in *CEC 02: Proceedings of the Evolutionary Computation on 2002. Proceedings of the 2002 Congress*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 1671–1676.

[17] A. Mohais, C. Ward, and C. Posthoff, "Randomized directed neighborhoods with edge migration in particle swarm optimization," *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 548–555, 2004.

[18] P. Suganthan, "Particle swarm optimiser with neighbourhood operator," *Proceedings of the Congress on Evolutionary Computation*, pp. 1958–1962, 1999.

[19] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," *Evolutionary Computation . CEC 99. Proceedings of the 1999 Congress on*, vol. 3, 1999.

[20] T. M. Blackwell and P. Bentley, "Don't push me! collision-avoiding swarms," in *CEC '02: Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 1691–1696.

[21] T. Krink, J. Vesterstrøm, and J. Riget, "Particle swarm optimisation with spatial particle extension," *Proceedings of the Congress on Evolutionary Computation*, pp. 1474–1479, 2002.

[22] M. Richards and D. Ventura, "Choosing a starting configuration for particle swarm optimization," *Proceedings of the International Joint Conference on Neural Networks*, pp. 2309–2312, July 2004.

[23] D. Russel, *Kinetic Data Structures in Practice, PhD*. Stanford University, March 2007.

[24] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, June 2004.