Population Density and Spatially Constrained Selection in Evolutionary Computation

G. DICK, P. WHIGHAM
Department of Information Science
University of Otago
P.O. Box 56, Dunedin
NEW ZEALAND
{gdick,pwhigham}@infoscience.otago.ac.nz

Abstract: - To be completed...

Key-Words: - Evolutionary computation, selection, spatial patterns

1 Introduction

Evolutionary computation (EC) uses the basic operations of biological systems (selection, reproduction and recombination) as a tool for optimisation and search. While the paradigm has worked well, the correlation of the operations to their real-world counterparts is often unrealistic. For example, a population of individuals is frequently represented as a "bucket" in which there are no restrictions on which individuals can be paired for mating. This phenomenon rarely occurs in nature. Usually, a population is organised in some way that imposes a restriction on the possible mating pairs.

This paper extends previous work on spatially constrained selection [1]. The spatially constrained selection paradigm places individuals on a plane and uses this space to restrict the way that individual pairs can be selected for reproduction. This mating restriction emulates some of fundamental theories explored in population genetics [2].

This paper begins by introducing the spatially constrained EC paradigm. A brief introduction into fields of similar research is also presented. The paper continues by presenting the results of experiments used to test the spatially constrained EC paradigm. Finally, conclusions will be made and future work will be described.

2 Spatially-Driven Selection

Most populations in natural systems are distributed over an area that is many magnitudes of size larger than their members. It is very rare for individuals to come into contact with every other member of the population to which they belong. Although this feature may at first appear to be detrimental to natural systems, it is likely that this phenomenon helps to promote speciation and enhance a population's ability to adapt to its environment [2].

Traditional EC systems use populations that have no spatial structure. Individuals are able to see every member of the population and are able to pair with anyone for mating, assuming they are selected. This model has the benefit of being relatively simple both in terms of implementation and mathematical representation. The downside of this simplicity is that they may not search the problem domain as efficiently as systems that use a spatially structured population. The use of population models from population genetics in evolutionary computation is a field that is being extensively studied. Cantú-Paz provides an excellent summary of this research [3].

2.1 Similar systems

There has been a large amount of research into partitioning a population spatially within an EC system. The most common examples are the parallel-executing evolutionary computation models, more commonly referred to as the fine-grained and coarse-grained models.

2.1.1 Fine-grained evolutionary computation

Fine-grained models of EC contain a single population that is distributed over a grid [4]. The shape of the grid is often a two-dimensional torus, however other topologies have been explored [5, 6]. Selection in this model is constrained by limiting mating to individuals that are within neighbourhoods. These neighbourhoods act as multiple subpopulations within the larger population. Genetic material of the individuals is spread through the population due to the fact that an individual can belong to more than one neighbourhood. Fine-grained EC is generally more efficient than a sequential model at searching problems spaces due to the constraining effect that the neighbourhoods have on the selection process. However, fine-grained EC methods are limited in the fact that they required expensive hardware in order to execute properly [3].

2.1.2 Island-based evolutionary computation

Island models of EC take several sequential systems and execute them in parallel [7]. Selection in this model is constrained by limiting mating to individuals that are within the same subpopulation. By dividing the individuals into smaller populations, the system can more efficiently search different parts of the fitness landscape. However, because the subpopulations are smaller, they tend to converge at a less than optimal solution.

To counter the premature convergence problem, island-based models incorporate migration. Migration shifts a portion of a population into a different one. This injects new genetic information into each subpopulation, which increases the diversity and allows the search of the problem domain to continue.

While island-based models are essentially an extension of sequential algorithms, they introduce a considerable amount of complexity. Island-based models have a topology which defines the direction that migrating individuals move within the system. The topology of these systems – that is, the number and size of the populations involved and how they are connected – can vary greatly, currently there is no well-defined "best" topology [8, 9].

Another issue that complicates island-based models is the migration strategy. The frequency of migration, the number of individuals to migrate and the selection of individuals to migrate are all parameters that need to be considered when setting up an island-based EC system [9].

2.2 Spatially constrained selection

In this section, we introduce the spatially constrained selection method. This selection scheme attempts to model the natural phenomena described in §2. The spatial model used to hold the members of the population is a Euclidean space in two dimensions. The space is infinite and continuous[1]. The selection scheme places individuals on a two dimensional plane and allows them to breed only with individuals on the plane that are "visible" to them. Spatially constrained selection cannot be used in asexual EC systems (for example Evolutionary Programming). This should be obvious, as there can be no concept of distance when only one parent is selected.

2.2.1 Individual Placement

An individual in a spatially constrained EC system has a pair of values that represent its location on a two dimensional surface. The individuals in the initial population have their coordinates randomly assigned so that they fit into a bounded area of unit size. Figure 1(a) shows a typical distribution of individuals on the plane.

2.2.2 Selection of Mates

In a spatially constrained selection method, the first parent is selected from the entire population using whichever selection method is being used by the EC system. A second parent is then selected from the deme that is visible to the initial parent. This concept of visibility essentially creates subpopulations based on the location of individuals on the surface. Figure 1(b) shows the selected individual and highlights its visibility radius. Figure 1(c) shows two individuals that could possibly be selected for breeding.

It is important to note that spatially constrained selection does not alter the actual mechanism for selecting an individual (for example. tournament, rank-based or roulette-wheel). The only difference that spatially constrained selection imposes is that the second parent is selected from a sub-population which is determined when the first parent is selected.

2.2.3 Offspring Placement

When new individuals are created in the system, they are placed in a location that maintains the relationship with their parents' coordinates. To introduce new offspring into the population, one parent is chosen randomly and the new individual is placed randomly at coordinates which fall within that parent's visibility radius.

Initial experiments placed an additional constraint on placement by only allowing individuals to be placed on the surface within the boundaries imposed on the initial population. It has now been determined that this leads the population to organise itself so that all of its members can see each other. Figure 2 shows a graph of the average proportion of the population an individual can see versus the number of iterations completed when using bounded space. By approximately 20% of completed iterations, the population is organised such that it is entirely visible by all individuals.

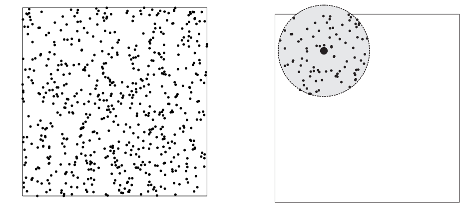
For this paper, the boundaries that constrain the population are removed. This allows a new individual to be placed anywhere within the full extent of its parents' visibility radii as shown in Figure 1(d).

2.2.4 Additional Parameters

Initial experiments using spatially constrained selection required two new parameters to be set in order to adjust the new algorithms used. These parameters are as follows:

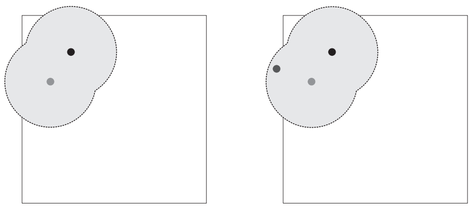
- *Side length:* This parameter was used to determine the boundaries of the plane that the individuals will "live" on. This parameter was used primarily in the placement of newly created individuals, maintaining a closed area in which the population was allowed to live on. This parameter is optional now that the selection process has been modified to permit a surface that has no boundaries. This parameter is automatically assigned the value of 1 when an unbounded space is required.

[1] Of course representing space in a digital form always implies a discrete representation, however for our purposes this limitation can be ignored.



(a) Placement of Individuals on Plane

(b) Visibility Radius of Initially Selected Individual and Available Deme



(c) The Selected Parents, With Available Area for Offspring

(d) Possible Location of Offspring Placement

Figure 1: Constraining selection with space.

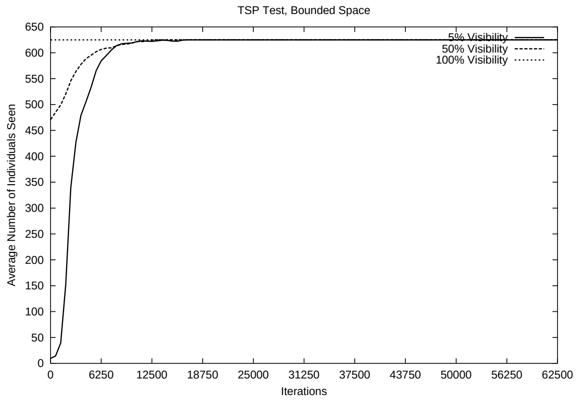


Figure 2: Population density over time using bounded space.

- *Visibility radius:* This parameter determines the maximum range that two individuals can see each other when selection takes place. Only two individuals within this radius can mate. The parameter is defined as a percentage of $\sqrt{2} \times l$, where l is the side length used for the initial population boundaries. When the visibility radius is set to 100% and the system is using bounded space, the entire population is visible by each individual and the population effectively has no spatial distribution.

As will be seen in §4.1, it is possible that both of these parameters can be removed from the system.

2.2.5 Computational Overheads

The spatially constrained selection process requires additional overheads maintain the visibility descriptions. In order to maintain a relationship between visible individuals it is necessary to calculate the distance between them. This calculation can be performed at either the selection stage, or when an new individual is created. For this paper, we chose the latter, as is simplifies the population density counting that is performed in §3.2.

In §3.1, it will be demonstrated that the computational overhead required by spatially constrained selection is offset by the reduction in the number of iterations required to find an equivalent solution when using selection methods that do not incorporate a spatially structured population.

3 Experimental Results

The test problem run for this paper was the Travelling Salesman Problem (TSP). The system parameters as described in Table 1. The tests were run with visibility settings of 5%, 25%, 50% and 100%. The tests were run at each visibility setting 100 times and the final results were averaged out over these runs. The tests were run using a steady-state genetic algorithm. The choice of a steady-state model rather than a generational one is discussed in §5.1.2.

Population Size	625.
Visibility	5%, 25%, 50% and 100%.
Base Selection	Tournament Selection with a tournament size of 6.
Iterations	62500.
Cities	30.

Table 1: The parameters used in the tests

3.1 Fitness Comparisons

Figure 3 shows the average fitness of the individuals against the number of completed iterations. The "control" is the plot of the results produced by a genetic algorithm that does not incorporate spatially constrained selection. A lower fitness value indicates that the individuals of the population have been more successful in solving the problem. As can be seen, the population is slightly more successful at finding fitter solutions when spatially constrained selection is used. More importantly, the population requires far fewer iterations to achieve the equivalent level of fitness than when a non-spatial parent selection mechanism is used. Examining Figure 3 it can be seen that after approximately 1/3 of the iterations for the non-spatial approach, the average fitness using spatial selection have reached to within 95% of the final result. This statistically significant improvement for convergence to a solution highlights the benefits of spatial selection EC. Although this is for a single example, similar improvements have been observed for a number of other, unpublished, examples.

It is interesting to note that adjusting the visibility parameter does not appear to alter the behaviour substantially. This observation will be discussed in §4.

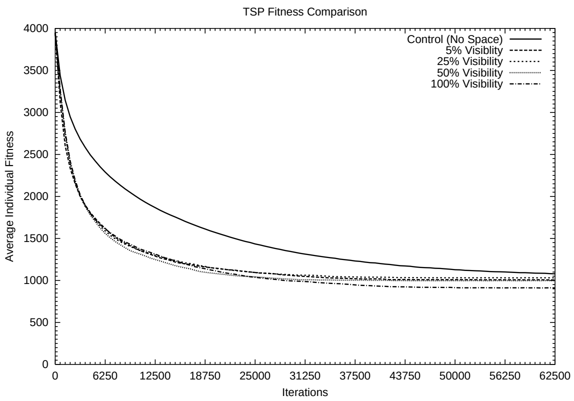


Figure 3: Average fitness results of experimentation.

3.2 Population Density

Figure 4 shows the average proportion of the population an individual can see against the number of iterations completed. There is a marked change in behaviour when compared with the results shown in Figure 2. The population converges to a point where its members can see approximately 25% of the other individuals. The population remains in this state until approximately 15000 iterations are complete, at which point there is a slow decay in the average number of individuals seen by the population's members.

The population density results show a similar trend to the results seen in §3.1. The value that the visibility parameter is set to has very little impact on the population density that is observed. An explanation for this effect will be offered in §4.

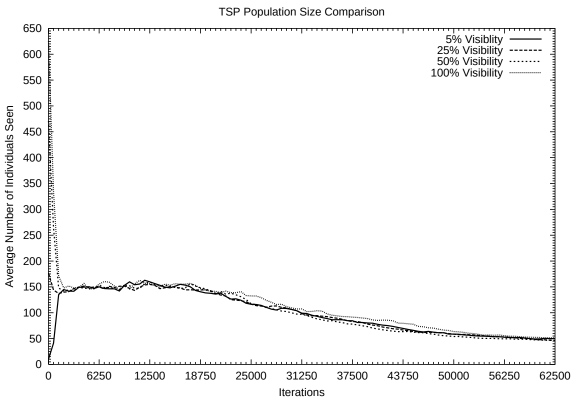


Figure 4: Population density results of experimentation.

4 Discussion

The results in §3 demonstrate that spatially constrained selection alters the behaviour of an EC system. The use of space as a constraining mechanism allows the population to more efficiently search the problem domain, finding better solutions on average and in fewer iterations. In addition to the improvement in searching the fitness landscape, there are two two notable properties of the system that are discussed below.

4.1 Visibility and Scale

The value of the visibility parameter appears to have little demonstratable effect on the results shown in §3 and §4. In both cases, the population quickly adapts to a uniform state that was independent of the visibility setting. This appears to be a side-effect of using unbounded space. Earlier experimentation used bounded space and the problems explored with this configuration favoured particular visibility settings [1].

It is possible that using unbounded space effectively creates a "scaleless" population. When using unbounded space, an initial population (when number of iterations = 0) that uses visibility parameter set to 5% may have a similar structure to the same population if it had used a larger visibility setting and was allowed to run for a number of iterations.

Since adjusting the visibility parameter appears to have little effect on the behaviour of the system, it would be reasonable to eliminate this parameter and have visibility based on an arbitrary value.

4.2 Space and Selection Pressures

The results in §3.2 shows a steady decrease in population density that begins after approximately 25% into a run. This is similar to the results observed in §3.1, where the average fitness of the population levels out in approximately the same amount of time. When the average fitness reaches this state, the selection of parents would be more evenly distributed over the individuals of the entire population, therefore the selection pressure will reduce. This could indicate that there is a relationship between the selection pressure present in the system and the density of the population.

Figure 5 shows the comparison of population densities when a selection pressure is present and when parents are picked at random in the absence of a selection pressure. When there is no selection pressure present, the population density steadily adapts itself into a position where the individuals of the population can see 4–5% of the other members. This occurs regardless of what value is used to set the visibility parameter.

When selection takes an individual's fitness into consideration, a selection pressure is produced. The behaviour of the population changes when this happens. The population quickly adapts to a structure that allows individuals to see approximately 25% of the other members. This happens regardless of the value used for the visibility parameter. The number of iterations required for the population to adjust itself also appears to be independent of the visibility parameter.

The increase in the average number of visible individuals for the population occurs when there are variations in fitness between members of the population, and in particular when individuals are improving in fitness. Once the population has converged the selection pressure decreases, and the population's spatial structure acts as if selection is random (see Figure 5).

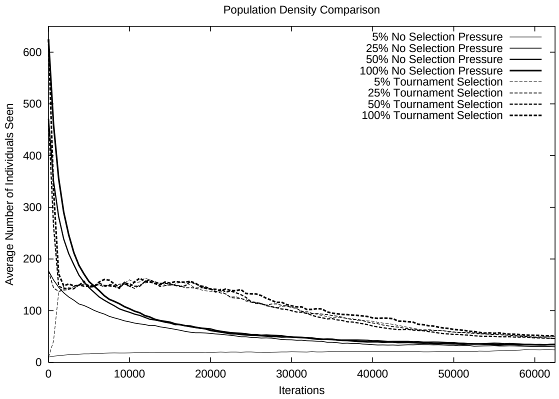


Figure 5: Population density with differing selection pressures.

5 Conclusion

The selection scheme plays an important role in evolutionary computation systems. It needs to efficiently bias selection towards fitter individuals while ensuring that sufficient diversity to solve the problem remains in the population. This paper presented work on extending the spatially constrained selection paradigm. The experimental results indicate that using this technique helps to improve the efficiency of evolutionary computation systems.

5.1 Future work

This paper has introduced some extensions to the spatially constrained selection technique. The results presented indicate that these extensions have improved spatially constrained selection and its ability to enhance EC systems. The following section describes possible directions that future work in this field could take to further improve the spatially constrained selection paradigm.

5.1.1 Population dynamics

5.1.2 Parallel implementation

The work presented in this paper was executed using a serial genetic algorithm. The spatially constrained selection algorithm used in this paper distributed the individuals at a conceptual level. This algorithm lends itself well to extensions that would additionally distribute the individuals at the physical/logical level. A parallel implementation of spatially constrained selection would require a steady-state algorithm. This is the justification for using steady-state genetic algorithms for this paper. Initially, it was thought that a parallel implementation would require a bounded population so that the population could be distributed evenly over the processors [1]. An alternative implementation is to organise the processors radially around a point and have each processor maintain the individuals that lie within the segment that it owns. Figure 6 shows an example of how the system may work with ten processors controlling the population.

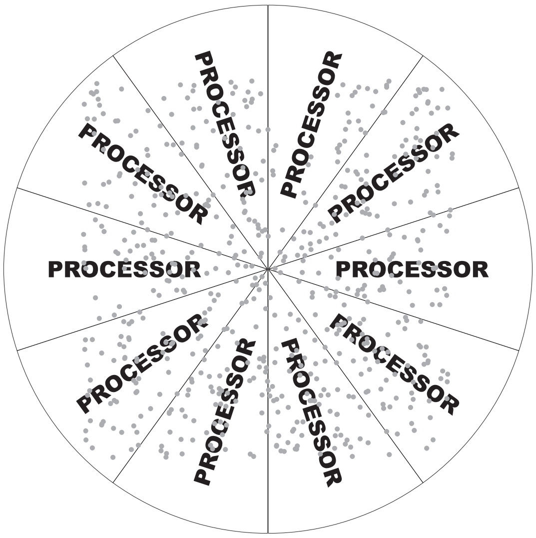


Figure 6: Possible method for partitioning population onto processors in a parallel implementation.

In addition to the processors controlling the population, there would need to be a master controller that would know the location of each individual in the system. The master's primary role would be to oversee the selection process, while the other evolutionary processes (crossover, mutation, reproduction…) would be handled by the processors that are managing the the population plane.

A parallel version of of the spatially constrained selection method could essentially produce a parallel EC paradigm that introduces no additional parameters over a standard sequential EC system[2].

References

[1] G. Dick and P. Whigham. Implementation of genetic algorithms on various interconnection networks. In *Australisia-Japan Joint Workshop on Intelligent and Evolving Systems*, 2002. To appear.

[2] Sewall Wright. *Evolution and the genetics of pop-*

[2] Technically, the number of processors in the system could be considered a parameter. Since the number of processors would be known to the user, it would be trivial to set its value. For this reason, we do not consider it to be a true parameter and thus can be ignored when considering the system as a whole.

ulations, volume 2, chapter 12, pages 290–344. University of Chicago Press, Chicago, 1969.

[3] E. Cantú-Paz. A survey of parallel genetic algorithms. Technical Report 97003, Illinois Genetic Algorithms Laboratory(IlliGAL), 1997.

[4] George G. Robertson. Parallel Implementation of Genetic Algorithms in a Classifier System. In John J. Grefenstette, editor, *Proceedings of the 2nd International Conference on Genetic Algorithms (ICGA87)*, pages 140–147, Cambridge, MA, July 1987. Lawrence Erlbaum Associates. Also Technical Report TR-159 RL87-5 Thinking Machines Corporation.

[5] M. Schwehm. Implementation of genetic algorithms on various interconnection networks. In M. Valero, E. Onate, M. Jane, J. L. Larriba, and B. Suarez, editors, *Parallel Computing and Transputer Applications*, pages 195–203, Amsterdam, 1992. IOS Press.

[6] Markus Schwehm. Parallel Population Models for Genetic Algorithms. Technical report, Universität Erlangen-Nürnberg, IMMD VII, 1996.

[7] John Grefenstette. Parallel adaptive algorithms for function optimization. Technical Report CS-81-19, Vanderbilt University, Nashville, TN, 1981.

[8] D. E. Goldberg, H. Kargupta, J. Horn, and E. Cantu-Paz. Critical deme size for serial and parallel genetic algorithms. IlliGAL Report 95002, University of Illinois, Urbana-Champaign, 1995.

[9] Erick Cantu-Paz. Topologies, migration rates, and multi-population parallel genetic algorithms. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 91–98, Orlando, Florida, USA, 13-17 July 1999. Morgan Kaufmann.