Working Paper 12/02

LandParcelS: A Module for Automated Land Partitioning

Demetris Demetriou[1], John Stillwell[1] and Linda See[1,2]

[1]School of Geography
University of Leeds
Leeds LS2 9JT

[2]International Institute of Applied Systems Analysis (IIASA)
Schlossplatz 1
Laxenburg, A-2361 Austria

Email: demdeme@cynet.com.cy;

j.c.h.stillwell@leeds.ac.uk;

see@iiasa.ac.at

May 2012

Contents

Abstract

Land fragmentation is a widespread problem and schemes for consolidating land are required to improve agricultural efficiency. This paper explains the development of a module called *LandParcelS* (Land Parcelling System) that is part of an integrated planning and decision support system called LACONISS which is being developed to assist land consolidation planning in Cyprus. *LandParcelS* is the component of the system that automates the land partitioning process by optimising land parcels in terms of shape, size and value. The methodology employs a genetic algorithm and results are presented when treating the partitioning task as either a single or multi-objective problem.

List of Figures

LandParcelS: A Module for Optimum Land Partitioning

1 Introduction

Land consolidation planning is required when there is inefficient use of land for agricultural purposes because of fragmented land ownership. This paper explains the development of the module called *LandParcelS* (Land Parcelling System), part of an integrated planning and decision support system called LACONISS which is being developed to assist land consolidation planning. The purpose of the system and its theoretical framework are explained in Demetriou *et al.* (2012). LACONISS involves a suite of subsystems including a module to measure land fragmentation (*LandFragmentS*), a module to design and evaluate alternative land redistributions (*LandSpaCES)* and a module to generate a set of new parcels (*LandParcelS*) that represent the best land partitioning solution based on the optimum distribution of parcel centroids produced by *LandSpaCES*. The first two modules are explained in detail in Demetriou *et al.* (2010; 2011a; 2011b). This paper explains the structure of the land partitioning module and exemplifies its use with case study blocks of land from a land consolidation area in Cyprus.

The structure of the paper is as follows. Section 2 sets out the land partitioning problem as it is carried out in practice and discusses existing related studies. Thereafter, section 3 focuses on modelling the land partitioning process as a single and multi-objective optimisation problem. Section 4 deals with the design and the operation of a genetic algorithm in terms of its representation and the definition of genetic operators. Section 5 presents the module toolbar that operationalises the model in a GIS environment and section 6 reports an application of the model using two blocks of land in the case study area that treats land partitioning as either a single or a multi-objective problem. The performance of the algorithm is tested based on various combinations of optimisation parameters.

2 Land partitioning

2.1 The conventional process

The first step in the preparation of a land reallocation plan involves the subdivision of the consolidated area into blocks of land, where each land block is enclosed by roads and/or streams, canals or the external boundary of the study area. Then a process of land relocation is carried out, which involves two sub-processes, land redistribution and land partitioning, although in practice this is treated as a unified procedure. Land redistribution, the reallocation of properties, has been already automated through the Design and Evaluation *LandSpaCES* modules as elaborated in Demetriou *et al.* (2010). Land partitioning receives the outputs of the land redistribution exercise as inputs and involves the design of the subdivision of land into smaller 'sub-spaces', i.e. land parcels. This is conventionally a trial-and-error process on a block by block basis that relies on legislation, the existing land structure, empirical design criteria, physical and technical constraints and rules of thumb.

In particular, legislation provides the following constraints: the minimum area of a new parcel should be: 2 donums (0.27 ha) for permanently or seasonally irrigated land, plants and vines; 4 donums (0.54 ha) for land which is able to be seasonally irrigated; and 10 donums (1.34 ha) for dry and unplanted land. These minimum size figures can be reduced by half with the approval of the Head of the Land Consolidation Department (LCD). In addition, the legislation states that all the new parcels must be readily accessible via a rural road. Furthermore, existing boundaries especially if they are physical objects such as a stream, a river, a high stone wall, a series of trees or a wild plantation should be taken into account if possible. Other technical constraints are the existence of buildings (e.g. a farmstead) or other kinds of construction (e.g. fencing). Moreover, parcel shape criteria based on heuristics, i.e. the generation of parcels with regular shapes as discussed by Demetriou *et al.* (2011b), are set out as well.

The process is currently carried out by two land consolidation planners normally using a CAD system and/or GIS. Given the set of constraints outlined above, the planners try to design the new parcels based on a predefined total value of land for each owner. The land value of each parcel may vary in order to reach the overall value of land given to each owner. Thus, for each land block, planners attempt to create a predefined number of parcels (as a result of a preliminary study), each with as regular a shape as possible, approximating a desired land value (because the latter is the basic land reallocation criterion according to legislation) and size, subject to the limitation that the latter exceeds the minimum limits as noted earlier and that a parcel should have access via a road.

The land value of a parcel is calculated by overlaying a layer of parcel shapes with the land valuation thematic map that consists of various categories. As a result, a parcel may be divided into sub-areas since it might overlap with more than one category; hence the total land value of a parcel is computed by multiplying the size of each sub-area with the corresponding land value of each category and then summing these. Parcel shape, size and land value are interrelated variables which may conflict. An example of such a subdivision that has been carried out by land consolidation experts for land blocks with IDs B25 and B14 for the case study area is described in Demetriou *et al.* (2010) and illustrated in Figure 1. Both land blocks will be utilised later for applying LandParcelS. The number within each parcel represents the parcel ID.

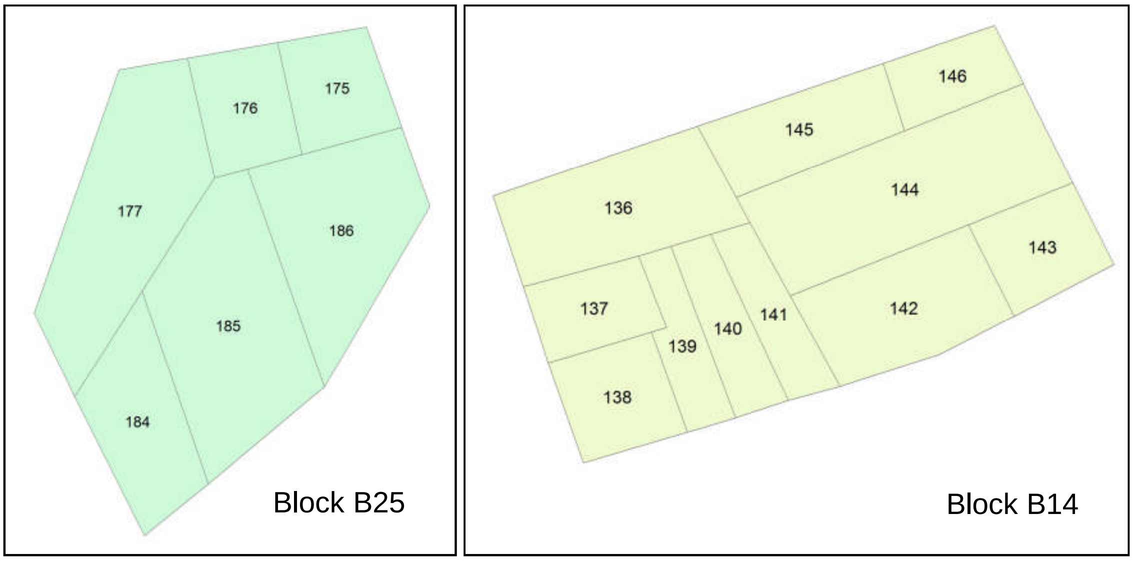

Figure 1: The subdivision of land blocks B25 and B14 as carried out by land consolidation experts

The conventional process described above suggests that, in computational terms, land partitioning is a design and optimisation problem (Tourino *et al.*, 2003) that focuses on how to optimally divide the space enclosed by a land block, given a set of points representing the centroids of new parcels so that the final parcels have as regular a shape as possible with a predefined size and land value. These centroids and their associated main characteristics such as size and land value are the basic output of the Design module of *LandSpaCES*. In addition, the Evaluation module of *LandSpaCES* identified the best alternative distribution of centroids based on seven evaluation criteria as demonstrated in the case study in section 8. An example of the output from this best alternative for the above two land blocks is shown in Figure 2 where both land blocks are enclosed by roads. Note that in one case in each block, a centroid has two parcel IDs, namely, 45/15 and 160/189 in blocks B25 and B14, respectively. This means that two co-landowners have been granted a separate parcel in approximately the same location. In practice, these parcels need to be adjacent to one another.

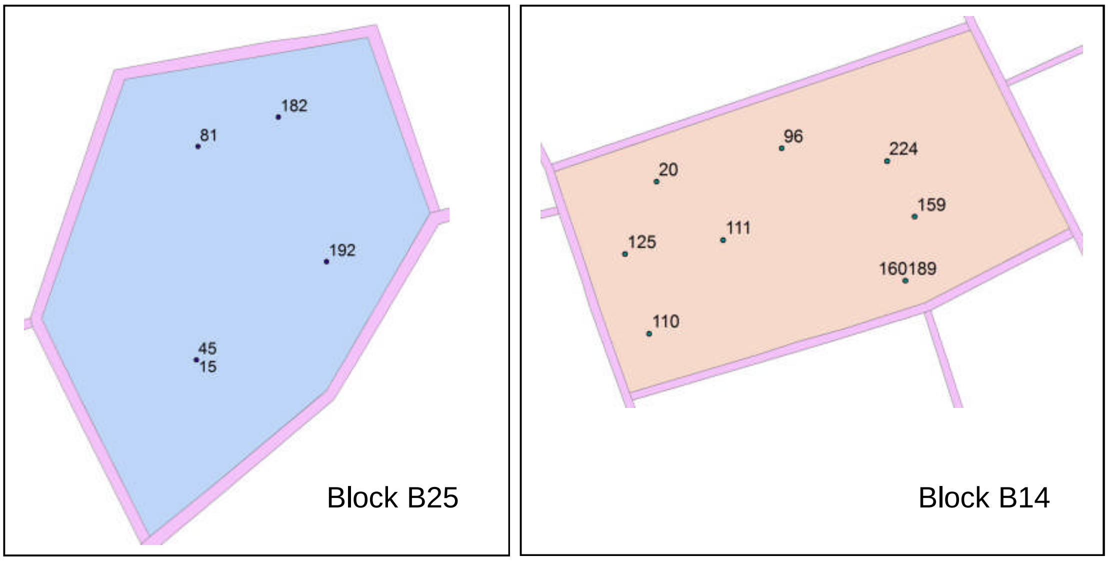

Figure 2. The centroids of new parcels in land blocks B25 and B14 resulting from *LandSpaCES*

Land partitioning is therefore an optimisation process that focuses on four main aspects of a parcel: shape, size, land value and road access. In particular, the main objective of the problem is to generate parcels with regular shapes subject to three main constraints: parcels should have access to a road and be of a predefined size and land value. Deb (2001) distinguishes hard and soft constraints. Whilst a hard constraint cannot be violated without making the solution infeasible, a soft constraint permits a range of variation within which a solution is feasible or alternatively, a maximum variation can be specified. Thus, road accessibility is a hard constraint whereas the size and land value of each parcel are soft constraints with an acceptable maximum variation employed in practice of ±10%. In addition, there are secondary but also important practical constraints such as: existing boundaries (e.g. a stone wall or an ecological line), buildings (a house, a farmstead) and other constructions (e.g. a fence, a well). As the primary constraints are the most important, they are the only ones involved in this version of the model.

2.2 Related work

There is very little literature on studies of land partitioning. Two studies by Buis and Vingerhoeds (1996) and by Tourino *et al.* (2003) both employed artificial intelligence techniques with GIS. In particular, Buis and Vingerhoeds (1996) used knowledge-based systems (KBS or ES) and GIS for the design of a new parcel layout in land consolidation projects, based on the principle that land partitioning *"is largely a hand crafted process, involving a balanced approach of optimization, handling constraints and experience"* (Buis and Vingerhoeds, 1996, p.308). The search for an optimal solution is obtained using a control strategy of steering the search process in a promising direction using heuristics and rules of thumb. It is a hierarchical generate-and-test process, in which solutions are incrementally constructed and tested to ensure each constraint holds so it is basically a kind of hill-climbing method. An advantage of this approach is that, further to agricultural efficiency issues (i.e. the shape and size of the parcels), ecological considerations are strongly taken into account. The basic limitations of this method are that it is semi-automated and also does not frame the problem as an optimization search process; hence, an optimum (or near optimum) solution cannot be obtained. Another system limitation is that the GIS and KBS are separate systems joined by loose coupling hence their communication is not efficient. In addition, this approach does not provide evaluation metrics regarding the performance of the algorithm in terms of finding the optimal solution or the computational time.

Tourino *et al.* (2003) built a complex GIS-based tool to support several of the tasks involved in land consolidation. The most sophisticated module is that which handles land partitioning and which has been automated by combining a region-growing algorithm and a simulated annealing optimization routine. An iterative seed-growing method is used to generate an initial distribution of the tessellated area among the domains (parcels). The area is divided into 'stands' (called land blocks in this research) and each stand is divided into square cells (pixels). Region-growing uses a heuristic flooding process, based on a linear objective function. It is a heuristic function comprised of six terms where each term is a constraint to guide the growth of the parcels. The planner may guide the process by weighting the importance of each term via a coefficient. The algorithm works separately for each stand, trying to obtain the most feasible set of parcels possible (in terms of shape). Region-growing may generate many alternative parcel partitions by changing the weights. Simulated annealing is then used to generate the new parcels (in terms of shape without changing their location) by minimising a non-linear objective function which is comprised of two terms; one to represent the objective to obtain parcels with regular shapes and the other as a constraint to maintain the score for each landowner (meaning the land value in this research).

The reason for trying to obtain the initial parcel shapes using the region-growing algorithm is that simulated annealing depends strongly on the starting solution. This is a disadvantage of this method as well as other classical optimization methods because the search for an optimum solution relies on one initial solution which iteratively may converge to an optimum or near optimum solution. As a result, even though simulated annealing is robust, fast and capable for solving large combinatorial problems, it does not guarantee the global optimum solution (Datta *et al.*, 2006). In addition, simulated annealing is not capable of providing a set of trade-off solutions in just one simulation run as in the case of MODM problems with conflicting objectives (which genetic algorithms can do). Another limitation of the system is that it does not take all the factors of the process into account, such as barriers (e.g. buildings, irrigation channels, wells) and pictorial elements (e.g. contour map, slope map).

System evaluation showed that the results were strongly influenced by the shape of the stand and the size of the original parcels. In addition, authors note that the final output was very different from that which experts would have produced.

The authors suggest that new optimization techniques should be considered and that the objective functions need to be improved. The research presented here follows on from these suggestions but genetic algorithms have been used to optimise the fit. Genetic algorithms use a population of solutions while searching for the optimum, instead of a single initial solution. Similar to Buis and Vingerhoeds (1996), Tourino *et al.* (2003) do not provide adequate evidence of evaluation regarding the performance of the algorithm in terms of finding the optimal solution and the computational time. In addition, the algorithm was applied to only one small block of five parcels.

3 Single and multi-objective land partitioning

3.1 Single-objective land partitioning

Based on the previous considerations, land partitioning can be modelled as a single objective minimisation problem subject to a set of constraints:

$$\min \sum_{i=1}^{N} \left| \frac{1}{16} - \frac{area(P_i)}{perimeter^2(P_i)} \right| \tag{1}$$

where *N* is the total number of parcels per land block and *i* represents one parcel. Equation 1, which has been employed by Tourino *et al.* (2003), is an area-perimeter ratio which results in a value for rectangular shapes with a length-breadth ratio 4:1. However, this formula, which in essence represents the compactness of shape, presents significant weaknesses for evaluating the shape of land parcels. Thus, a new index called the PSI (parcel shape index) has been developed (Demetriou *et al.,* 2011b) which outperforms other indices because it takes into account six parameters instead of two, namely: length of sides, acute angles, reflex angles, boundary points, compactness and regularity and, in addition, it considers the desired shape of parcels. The PSI takes values of between 0 and 1 representing the worst and optimum parcel shape, respectively. Therefore, the above objective can be restated using the PSI as follows:

$$min \sum_{i=1}^{N} 1 - PSI_i \qquad (2)$$

Ideally, the above function equals zero if all parcels of a block have the optimum shape, which is a rectangule with a breadth: length ratio 2:1.

If the aim is to generate parcels with regular shapes independently of their size and land value, then the above objective function can be used and the task could be completed using a single objective function optimisation and therefore a single optimum solution would exist. The definitions of an optimum or regular shape for a land consolidation plan and specification of the relevant PSI values have been discussed elsewhere (Demetriou *et al.,* 2011b). A shape is regular or near regular if it has a PSI of 0.7 to 0.9 and optimum or near optimum if the PSI is more than 0.9. Therefore, for optimisation purposes, any parcel with a PSI of more than 0.7 will be considered as acceptable with a gradual increase in terms of quality from near regular to optimum, meaning a PSI from 0.7 to 1.0. Thus, a kind of scaling is applied to PSI values that fall within this range. Namely, the term (1-PSI) is divided by 10, thereby favouring parcel shapes with PSI values between 0.7 and 1.0 and penalising parcel shapes with a PSI less than 0.7.

The following seven constraints relate to parcel: size, land value, length of frontage side, number of corners, linearity of boundary sides, how close to perpendicular are the frontage border lines to a road and cell adjacency in the case of a raster representation. As noted earlier, the PSI depends on six parameters; three represent corresponding constraints, namely, length of frontage side, number of corners, and how perpendicular the frontage border lines are to a road. In other words, the violation of these constraints is inherently penalised by the calculation of the PSI. Thus, no further consideration is required for these constraints since they are inherently included in the objective function. In addition, the constraint regarding the linearity of boundary sides is always fulfilled inherently by the Thiessen polygons method for the creation of parcel shape. Furthermore, the cell continuity constraint is not relevant because creation of Thiessen polygons is a vector-based process.

Based on the above analysis, three constraints must be taken into account in the optimisation process: size, land value and the accessibility of a parcel from a road. As noted, the first two are soft constraints and the latter is hard. However, these constraints are manageable in the context of optimisation only if a mechanism for generating feasible solutions is available, that is, solutions which do not violate any constraint. In such cases, the optimisation process only needs to find the optimum or near optimum solution in terms of parcel shape. Unfortunately, this is not the case and the problem is more complicated since the generation of parcels with a predefined size and/or land value is part of the problem. As a result, both parameters must be incorporated into the optimisation process. In other words, the two soft constraints can be treated as objective functions (Deb, 2001); hence land partitioning is converted from a single to a multi-objective optimisation problem as outlined in the next section.

3.2 Multi-objective land partitioning

Land partitioning can be formulated as a multi-objective problem with three objective functions representing shape, size and land value as follows:

$$minimise \left(\sum_{i=1}^{N} (1 - PSI_i) * w_1 + \sum_{i=1}^{N} |dArea|_i * w_2 + \sum_{i=1}^{N} |dValue|_i * w_3 \right) \tag{3}$$

subject to the following constraint:

$$\sum_{i}^{N} R_i = 0 \tag{4}$$

where *dArea* and *dValue* are the percentage differences between the desired and designed size and land value of a parcel, respectively and w_1, w_2, w_3 are the weights for each objective function that sum up to 1. The function R equals 0 or 1 when a parcel has access to a road or not, respectively. This is an equality constraint, that is, a hard constraint which, if not fulfilled, renders the solution infeasible. This constraint can be used as a penalty function that equals the number of parcels without accessibility in a land block and it can be added to the overall fitness function. The use of a penalty function in order to penalise solutions that violate one or more objectives is a popular constraint handling strategy although it may distort the objective function and hence lead to a sub-optimal solution (Deb, 2001).

Based on the above, an overall fitness function can be generated by combining the above two equations that compose four functions: F1, F2, F3 and R. Ideally the sum of the fitness will equal zero if all the parcels included in a land block have an optimum shape (F1) with the desired size (F2), land value (F3) and access from a road (R):

$$Fitness = \left(\sum_{i=1}^{N}(1 - PSI_i) * w_1 + \sum_{i=1}^{N}|dArea|_i * w_2 + \sum_{i=1}^{N}|dValue|_i * w_3\right) + \sum_{i}^{N} R_i \qquad (5)$$

In contrast to single-objective optimisation that involves a unique optimum solution, multi-objective problems with conflicting objectives involve a different optimum solution for each objective. In addition, there is not a single optimum solution which simultaneously optimises all objectives. As a result, the outcome is a set of solutions that are all optimal in varying degrees of trade off between the objectives. Graphically, these optimal solutions lie on a curve called the Pareto-optimal front. In particular, if all objective functions are to be minimised, this front lies close to the bottom-left corner of the search space. In principle, in multi-objective problems, there exists at least one solution in the Pareto-optimal set which will be better than any other non-Pareto optimal solution (Deb, 2001).

Therefore, the task in multi-objective optimisation is to find the Pareto-optimal solutions which are also called non-dominated solutions because none of these solutions is the best with respect to all objectives unless the importance of each objective can be defined. Thus, in the case where there is a confidence regarding the weights of objective functions, there is no reason to find other trade-off solutions (Deb, 2001) and the multi-objective problem can then be converted to a single-objective solution by utilising an appropriate vector of weights for objective functions. This requirement of multi-objective optimisation which focuses on finding multiple optimal solutions in one single simulation run is what makes genetic algorithms a unique method for this purpose. This is the principal difference between genetic algorithms and classical or other stochastic search optimisation algorithms that use a point-by-point approach where one solution in each iteration is modified to a different, and hopefully better, single optimised solution. In contrast, genetic algorithms use a population of solutions in each iteration. It should also be noted that if the objectives of an optimisation problem are not conflicting independent of the number of objectives, then there is a single optimum solution and all members of the population are expected to converge to that optimum solution.

4 Genetic algorithms

Genetic algorithms have the potential to solve the land partitioning problem as described in this section of the paper.

4.1 Representation

Use Vector or raster structure?

The land partitioning problem can be represented using both available GIS data structures, namely, raster and vector, although the latter is the normal structure used to represent the problem in CAD systems. Thus, a critical question regarding representation of the problem is whether to employ a raster or a vector data structure. A raster representation was initially adopted as discussed but this effort was abandoned early on in the research because the process of crossover (which is the fundamental GA operator) between two raster solutions presented significant weaknesses when executed on a pixel by pixel basis as explained below.

Firstly, it was inherently very time consuming and, in addition, extra time was needed for the calculation of the various parameters involved in the fitness function (the six factors of the PSI noted earlier and the land value of a parcel) which required the conversion from raster to vector because of the limitation of the former structure. Secondly, the crossover operator was tested, and resulted in completely infeasible solutions at times in terms of the number of parcels generated. This is because parcels with the same ID in the crossed solutions may have not any common pixel so these parcel(s) were not included in the child solution. Thirdly, crossover resulted in parcels with non-linear boundary sides and thus an additional constraint would have been a necessary addition to the process. Moreover, many parcels have numerous boundary edges with completely irregular shapes so the consequence was that much more time was required for optimisation. Fourthly, the accuracy of a vector representation is much higher than the raster (centimetres vs metres). Finally, the vector format is fundamentally the representation utilised in CAD systems where the actual task of land partitioning is currently carried out.

The only advantage of using a raster representation for the particular problem is that it would be easier to reach the desired size and land value of a parcel because it provides a detailed cell based representation of space in contrast to a vector structure that does not divide space into cells. Thus, taking into account the above considerations, it was decided that a vector based structure would be utilised for representing the land partitioning problem.

The evolutionary structure

The conventional process of land partitioning is carried out on a block by block basis until the whole plan is completed. The optimisation process follows the same rationale, suggesting that optimisation is carried out separately on a block by block basis, because no reallocation changes between blocks are permitted at this stage, since the relevant decisions have been already taken in *LandSpaCES*. In addition, the complexity of the problem and the computational time are significantly reduced. Thus, the land block is the basic unit on which evolution is undertaken.

In particular, in evolutionary terms, a land block represents an *individual* (or organism) which is evolved during the optimisation process. A land block is divided into parcels representing *chromosomes*. A *chromosome* encodes the characteristics that define an individual such as: shape, size, land value, accessibility to a road. In addition, shape is further represented through the PSI by the six aforementioned features. Each chromosome has a core *gene,* namely, a centroid which is defined by its X,Y coordinates. A set of individuals compose a *population*. Summarising the above, the genetic algorithm has the following hierarchical vector-based structure: population-individuals-chromosomes-genes; representing respectively: a set of subdivision solutions for a land block; one subdivision solution for a land block; land parcels; and centroids of parcels. A graphical representation of this structure is illustrated in Figure 3. The attribute table of an individual including all the relevant parameters is illustrated in Figure 4.

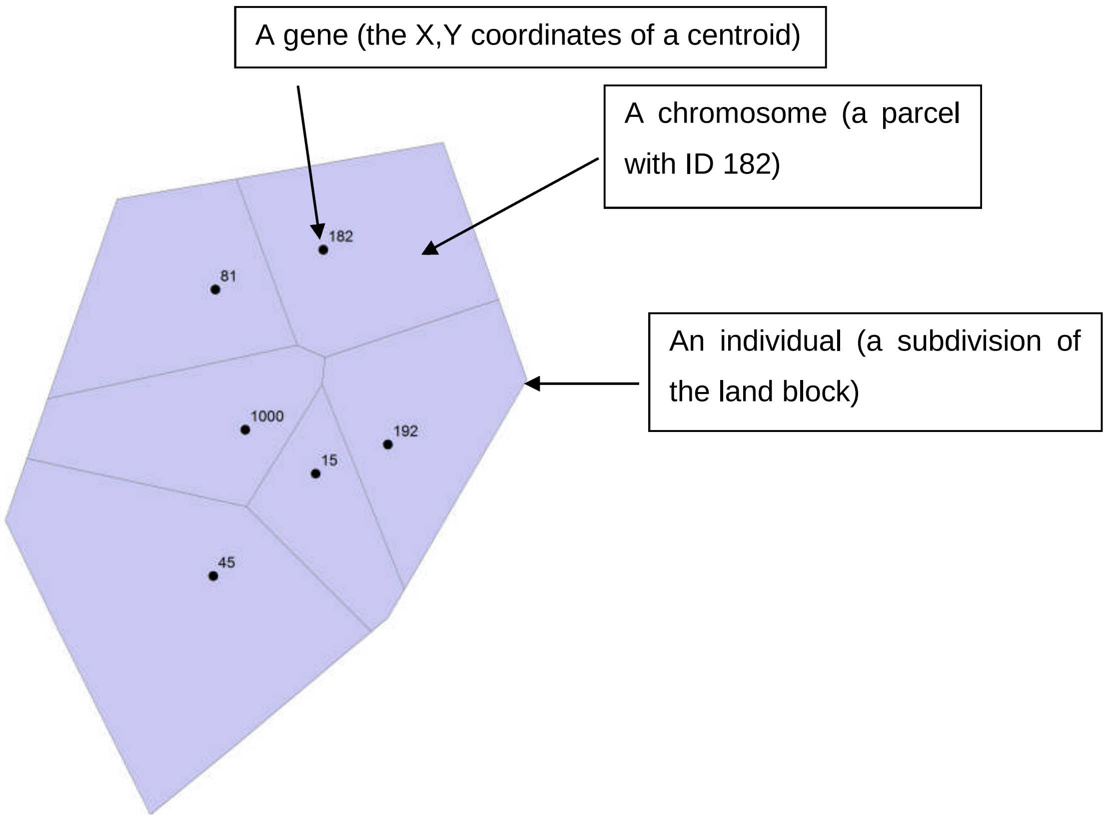

Figure 3: A graphical representation of the GA structure

Attributes of VectorB25_1

FID	Shape	Id	Input_FID	Parcel_ID	Owner_ID	Area	Value	PSI	PA2Ratio	NminLenght	NumAcute
0	Polygon	0	0	15	42	5881,8	481,07	0,639403	0,049574	2	1
1	Polygon	0	2	192	212	9947,01	1280,91	0,67366	0,055063	1	1
2	Polygon	0	4	182	16	3418,45	466,74	0,662753	0,063265	1	1
3	Polygon	0	5	1000	1111	2762,091014	0	0,747903	0,052723	2	0
4	Polygon	0	1	45	121	5881,8	481,07	0,669203	0,05926	1	1
5	Polygon	0	3	81	2	2676	317	0,659197	0,057426	2	1

FID	NumReflex	NumPoints	SDRadials	dArea	dValue	F1	F2	F3	CenX	CenY	R1
0	0	5	5,698429	-3424,989011	273,68794	0,360597	0,582303	1	144908,59	351940,53	0
1	0	5	10,697014	-5508,554413	892,376518	0,32634	0,55379	1	144945,7	351959	0
2	0	5	7,135679	1779,517728	-203,16871	0,337247	1	0,303278	144924,31	352029,95	0
3	0	6	6,932808	1324,164197	-352,321222	0,02521	0,479406	1	144852,71	351962,14	0
4	0	5	14,614233	3815,938276	-290,845598	0,330797	1	0,376784	144857,33	351884,57	0
5	0	4	10,562623	2013,92353	-315,185615	0,340803	1	0,498565	144851,38	352018,28	0

Record: 1 Show: All Selected Records (0 out of 6 Selected) Options

Figure 4: The attribute table of an individual

<u>The genetic process</u>

The genetic process is illustrated by the following sequence:

Create initial population

Evaluate initial population

Do

- select individuals for mating
- create offspring by crossover with a probability P_c
- mutate selected individuals with a probability P_m
- evaluate new individuals
- terminate process if a certain criterion is satisfied
- keep a percentage e of individuals from previous population for mating

Loop

End of the process

Initially, a random population of individuals is created and evaluated using all the terms or a combination of terms in equation 5. Then an iterative process begins whereby an iteration is called a generation. In particular, a selection method is employed to fill the mating pool with the same number of individuals as found in the initial population based on their fitness value. Afterwards, new individuals (offspring) are created by applying the genetic operators to parent individuals. In particular, crossover combines the genetic code of two randomly selected parent individuals from the mating pool. Then, changes are introduced into the genetic code of an individual by mutation. Eventually, new offspring are evaluated using the fitness measure, and if the termination criterion is met, then the iterative process ends (exits loop) and hopefully the best solution is returned. Otherwise, a new population is created by keeping a percentage (e%) of best individuals from the previous population to be put directly into the new mating pool and selecting from the rest of the individuals (using the employed selection method) to complete the mating pool with the equal number of individuals as the initial population. The basic steps of the process are discussed further below.

4.2 Generation of a random population

This initialisation operation is used to generate a random population consisting of a defined number of individuals. As noted by Deb (2001), the size of the population is a critical point in GAs in terms of the success of the algorithm for finding the global optimum and not a sub-optimal solution. The size of the population depends on the complexity of the problem concerned. This initialisation operator is based on a well-known GIS mechanism for sub-dividing space into smaller sub-spaces based on a given set of points called Thiessen polygons or Voronoi diagrams/network. This paper uses the former term. The use of the Thiessen polygon method is based on the fact that the input in land partitioning is the output of *LandSpaCES* which, as shown in Figure 2, is a set of points for each block representing the approximate centroid of each new parcel accompanied by a set of attributes.

Creating Thiessen polygons is a method for dividing the 2-D Euclidean space into a number of regions equal to the number of points provided (Figure 3). Thiessen polygons have the unique property that each polygon contains only one input point, and any location within a polygon is closer to its associated point than to the point of any other polygon (Chrisman, 2002). The concept of Thiessen polygons has been widely applied for space partitioning problems in a variety of disciplines (Dong, 2008) and it has gained a lot of attention for modelling spatial related problems (Gong *et al.*, 2011). In ArcGIS (with an ArcInfo license), Thiessen polygons can be generated either in a vector form using the Create Thiessen Polygons geoprocessing tool or in a raster form using the Euclidean Allocation tool in the Spatial Analyst Extension. In the former case, all points are represented as a triangulated irregular network (TIN) that meets the Delaunay criterion (Chrisman, 2002). Then, the perpendicular bisectors for each triangle edge are generated, forming the edges of the Thiessen polygons. The locations at which the bisectors intersect determine the locations of the Thiessen polygon vertices. In the latter case, every cell in the Euclidean allocation output raster is assigned the value of the source to which it is closest, as determined by the Euclidean distance algorithm.

Although Thiessen polygons provide a ready to use tool for space partitioning, the only criterion used for this purpose is the nearest-neighbour rule between points, without taking into account any associated value, e.g. any attributes of the points. Dong (2008) and Gong *et al*. (2011) have constructed algorithms for the generation of weighted Thiessen polygons based on an attribute of the points using a raster and a vector based structure, respectively. Although both methods have drawbacks, a wider range of spatial situations can be modelled compared to those using ordinary Thiessen polygons. However, both methods are not appropriate for land partitioning, e.g. by employing parcel size or the land value as weights, because the constructed shapes tend to be cyclical and not regular. As a result, ordinary Thiessen polygons are utilised for randomly generating alternative sub-divisions of a land block. The aim to steer Thiessen polygons towards the creation of parcels with regular shapes and a predefined size and/or land value with access to roads is left to the genetic algorithm.

The random generation of different subdivisions is based on the simple logic that: if the initial layout of centroids within a land block provided by *LandSpaCES* is randomly moved to new locations, then a new solution can be created and so on. This random movement (R_m) of centroids can be any distance in a 0.5m step, in any direction and may reach a $maxR_m$ which is calculated as the square root of the desired size (A_i) of the associated parcel with the centroid concerned, multiplying by a constant (c) which is capable of varying (increasing or decreasing) the searching distance of the algorithm as shown in equation 6.

$$maxR_m = c\sqrt{A_i} \quad (6)$$

This figure is based on the assumption that, in the case of a square shaped parcel which is very close to the optimum parcel shape, the maximum movement equals the length of the side of the square. As a result, the movement is limited to a circle with radius 2*$maxR_m$.

The reason for defining this maximum distance of movement is to limit the region in which a new parcel is created so as to be as close as possible to the original centroid while on the other hand not limiting the diversity of the population because this prevents the convergence of the algorithm to a global optimum solution. Thus, the constant *c* largely determines the diversity of the initial population and the search performance during the evolution of the algorithm. Another critical issue related to the $maxR_m$ is to check if a new location of a centroid falls outside the land block concerned; in such a case a new random location is provided until the new point is located within the block.

Another issue is that some centroids provided by *LandSpaCES* are in a common location because the original parcels in that location are owned by two or more co-owners (Figure 2). In practice, these owners are usually allocated new parcels in the neighbourhood. Thus, before beginning the creation of the polygons, these common centroids should be separated. This task is done randomly using the $maxR_m$ distance with a small constant *c*.

Another problem tackled was how to represent the residual area of a block that has not been allocated to any landowner. This 'unallocated area', which is allocated later before completing the final plan or may be allocated by the LCD, should be represented as a parcel; hence the creation of an extra centroid for each block was needed. Common sense says that this unallocated parcel area should be located in the least dense part of the block. In other words, the new centroid should be the point that has the longest distances from all existing centroids.

For this purpose the Euclidean Distance tool of Spatial Analyst was employed, which provides an output raster that contains the measured distance from every cell to the nearest source, that is, the existing centroids. The distances are measured as the crow flies (Euclidean distance) in the projection units of the raster, such as feet or metres and are computed from cell centre to cell centre. Then the cell with the largest value (having the furthest distance) is the one that will be searched for and its coordinates are used to create the new centroid. An example for land block B14 is illustrated in Figure 5 in which the centroid with ID

1000 is the new point having the farthest distance from all of the other points. Cells with the longest distances are represented by a deep brown colour. The algorithm reads this maximum distance and identifies the X,Y coordinates of the relevant cell in order to create a new centroid in this location.

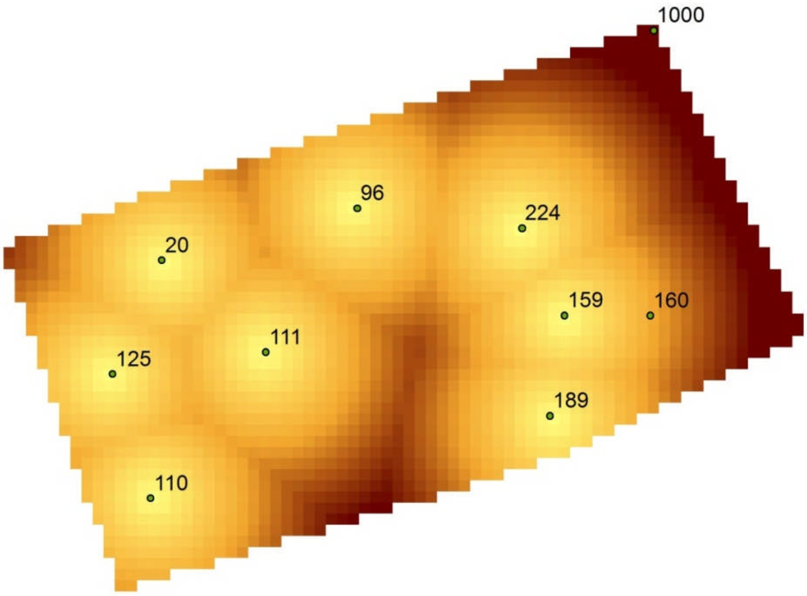

Figure 5: The outcome of the Euclidean Distance tool to identify the furthest point (with ID 1000) from existing centroids for land block B14

Once the common centroids are separated, the new centroid representing the unallocated part of the block is created and the initialisation algorithm randomly moves the initial location of the centroids, for a number of times defined by the user, to create a random population, including different subdivisions of the block using Thiessen polygons. This is the starting solution of the model from which the GA begins its search. An example showing six different random solutions for land block B25 is illustrated in Figure 6. The first three solutions are feasible whilst the other three are not. In addition to the initial operator utilised for the generation of a random population, the following five evolutionary operators are involved in the process: fitness function, selection-mating, crossover, mutation and an elite preserving operator which is discussed in the sequent section.

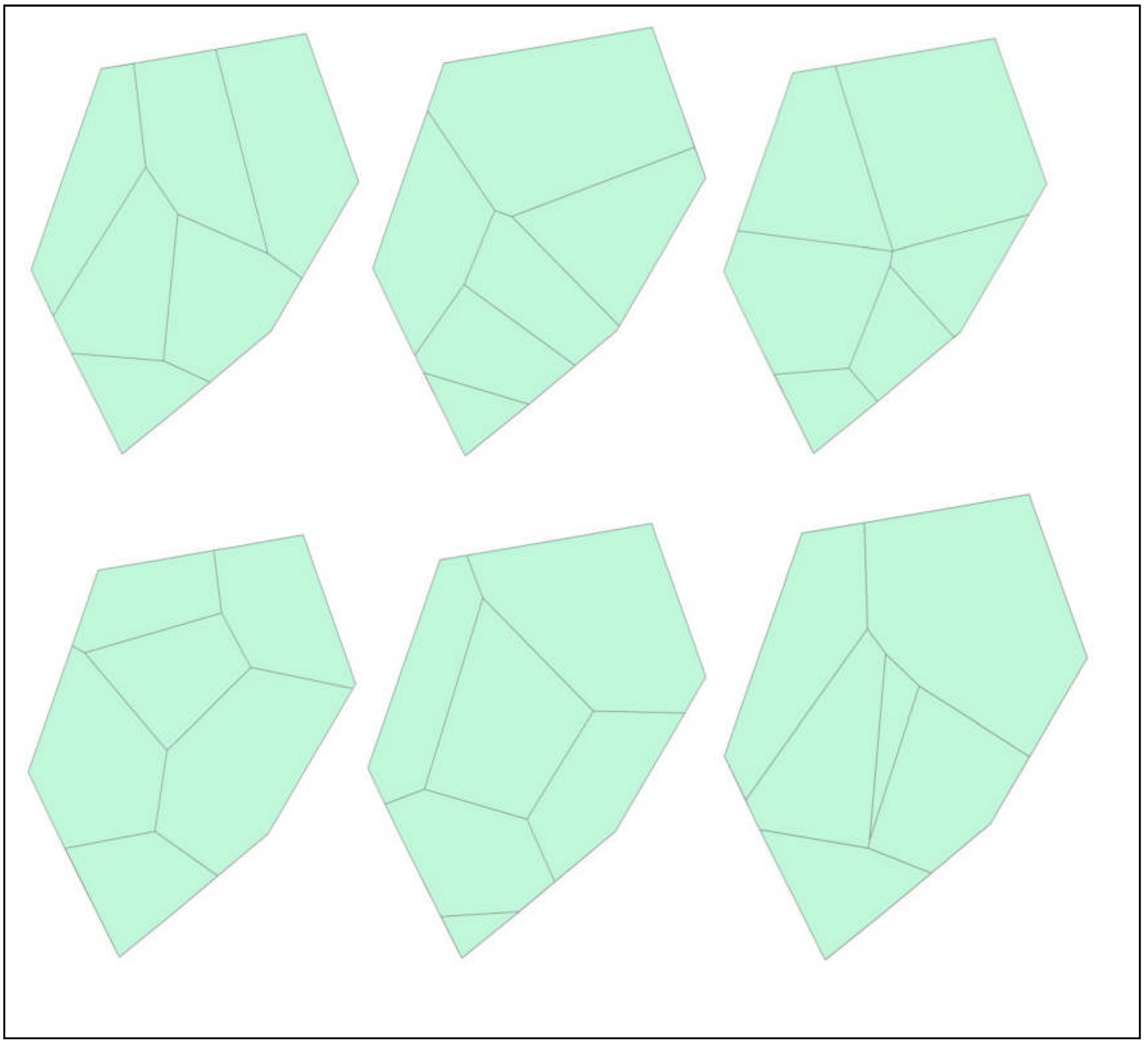

Figure 6: A set of six random individuals for land block B25 picked up from a population

4.3 Genetic operators

Fitness function

The fitness function is applied at two levels: focal and zonal operators referring to the parcel (chromosome) and block (individual) levels respectively. When the initial population is created, a fitness function is used to measure the quality of each individual with respect to the model objectives. All model objectives are set out in the fitness function shown in equation 5. However, the fitness function may vary depending on the number of terms (objective functions) included in it, defining land partitioning as a single or multi-objective problem. Thus, taking into account the land consolidation practice, three different multi-objective optimisation schemes can be defined for investigation: shape and size; shape and land value; and shape, size and land value. Initially the fitness of each chromosome (parcel) is calculated and then the overall fitness of the individual is simply the average fitness of all chromosomes that comprise that individual. The way this is undertaken is analogous to focal and zonal statistics in ArcGIS. A focal operator is a measure over the sub-space of a zone centred over a focal point while the zonal operator is a metric of the whole zone.

Selection-mating

A selection operator selects which individuals will be involved in the reproduction process. The main aim of this operator is to make multiple copies of good solutions based on their fitness score and to eliminate bad solutions from a new population, while keeping the population size stable. A number of methods exist for doing this such as tournament selection, proportionate selection, roulette wheel selection and ranking selection (Deb, 2001). Goldberg and Deb (1991) showed that tournament selection outperforms or is at least equivalent both in terms of the convergence and computational time compared to any other selection method that exists in the literature. Therefore, tournament selection was chosen for this evolutionary model as used in other similar spatial problems (Delahaye, 2001; Krzanowski and Raper, 2001; Van Dijk, *et al.*, 2002; Bacao *et al.*, 2005; Datta, *et al.*, 2006).

In tournament selection, two solutions are randomly selected from the current population and the best between the two is placed in the mating pool. Then, two other solutions are selected (excluding those already selected) and the best one fills the mating pool. The process is carried out systematically so that each solution participates in two tournaments and hence the best solutions always win in both rounds. Two copies of these solutions are then transferred to the mating pool. In contrast, the worst solutions will lose in both rounds and will thus not participate in the mating pool while bad solutions may have only one copy in the mating pool. With this iterative process and combined with the fact that the crossover operator has many possibilities to produce a better solution than its parents, an improved population in terms of fitness is gradually created from generation to generation, until a convergence of the algorithm to the global optimum solution will be achieved.

Crossover

Crossover is the process of the mating of two individuals (parents) by exchanging or combining genetic material (genes) to create a new individual (an offspring). In particular, this model's crossover involves the combination of genes (X,Y coordinates of a centroid) between two corresponding chromosomes (land parcels) that belong in two parents. Whilst the most popular GA encoding is the binary string system (0 and 1), the genes of this GA are represented by real numbers, i.e. X,Y coordinates, and thus a real parameter crossover encoding is utilised. This is the most common encoding utilised in complex problems (Renner and Ekart, 2003; Datta, *et al.*, 2006) when objective functions include real valued parameters because it avoids extra processing associated with decoding the consequents, involving the so called 'Hamming cliff problem' (Krzanowski and Raper, 2001).

As noted earlier, initially a raster based, cell by cell crossover was tried but it did not work properly. Then a crossover was used that was based on the average X,Y coordinates of two parents which was also used by Krzanowski and Raper (2001). This crossover led to the premature convergence of the algorithm in a sub-optimal solution because the diversity of the population quickly reduced towards the mean centre of the centroids of the initial population. This is due to the fact that the crossover search performance is low since it is limited to the coordinates of the line that connects the centroids of the two parents. Thus, the exploitation (selecting best solutions) and exploration (find better solutions) performance of a GA, which is represented by the selection and crossover operator respectively, are critical to the success of the algorithm.

After examining a plethora of real-parameter operators found in the literature and presented in Gwiazda (2006) as well as the relevant suggestions noted by Deb (2001), the BLX-a crossover operator introduced by Eshelman and Schaffer (1993) was utilised in the model. An advantage of this operator is that it may search outside the line that connects the centroid of the two parent solutions. It also has the property that the location of the offspring depends on the difference between the parent solutions. Thus, this operator follows an adaptive search strategy which involves searching the entire space early on while also maintaining a focused search when the population tends to convergence in some region of the search space (Deb, 2001). As a result, this operator enhances the diversity of a population which is reduced by the selection operator to avoid premature convergence. It is called the Blend Crossover operator and the way it works for the X coordinate is shown in Figure 7. Similarly, it can be applied for the Y coordinate.

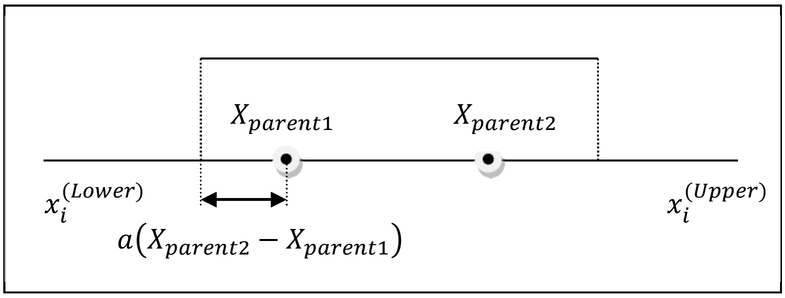

Figure 7: The BLX-a crossover operator for the X coordinate (adapted from Deb, 2001)

The offspring values for X_{new} and Y_{new} are calculated based on equations 7 and 8, respectively:

$$X_{new} = (1 - \gamma_i) \times X_{parent1} + (\gamma_i \times X_{parent2}) \quad (7)$$

$$Y_{new} = (1 - \gamma_i) \times Y_{parent1} + (\gamma_i \times Y_{parent2}) \quad (8)$$

where

$$\gamma_{i=(1+2a)u_i - a} \quad (9)$$

u_i = a random number between 0 and 1

a = 0.5

Investigations showed that this operator works best with a equal to 0.5 (Deb, 2001), and hence this value was used. A crossover operator can be applied based on a probability *Pc* which is usually between 0.7 and 1. Taking into account that the algorithm involves an elitist operator (discussed later), which directly transfers a percentage of best parents into the next generation, there is no reason to adopt a *Pc* value of less than 1. In addition, this option enhances the searching power of the algorithm and maintains the diversity of the population (Van Dijk *et al.*, 2002).

Mutation

Mutation, which is rare in nature, involves a random change to the genetic material (to the gene) of an individual. Although fitness may be worse than before mutation, the process is necessary to maintain diversity in the population and avoid premature convergence to local optima. In the case of binary coding, it involves the random flipping of a selected gene (e.g. from 1 to 0 or vice versa) in a chromosome. Similarly, in our case, it involves a random change or displacement of a gene (X, Y coordinates) of a chromosome (a parcel) of an individual (land block) in a new location. It can be applied at two levels: parcel based or in just one chromosome of an individual which is randomly selected, or block based where all the chromosomes of an individual are randomly selected and subject to the mutation operator.

Krzanowski and Raper (2001) defined a ‘small’ and a ‘big’ mutation affecting an individual and the whole population, respectively. The reason for utilising parcel based and block based mutation is that the former type of mutation only affects the fitness of an individual slightly, especially when an individual consists of many chromosomes. This is due to the fact the fitness of an individual is the average fitness of all the chromosomes that belong to it; hence, even in a positive change to the fitness of one chromosome, the expected overall change will be small. This remark is aligned with the finding of Krzanowski and Raper (2001) that the mutation operator (both ‘small’ and ‘big’) has no effect in the evolution process for solving spatial problems, which has also been confirmed by further studies (Krzanowski, 1997).

As for crossover operators, after examining a plethora of real-parameter mutation operators found in the literature and presented in Gwiazda (2007), a random mutation scheme was used for parcel based and block based levels involving a random displacement in any direction of the current centroid(s) location of a parcel or all parcels, respectively. The maximum displacement bound of the centroids set at $maxR_m$ (equation 6). A similar mutation operator has been applied in Delahaye (2001) for airspace sectoring. No reference is made for the effects of this operator in the evolution process. When a mutation probability P_m of 0.05 is used, two individuals from a population of 40 are subject to the mutation operator.

Elite - preserving operator

A way of speeding up the convergence of a GA is by utilising an elite preserving operator that enhances the possibilities for creating better offspring. Such an operator favours the best or elite members of a population, which are automatically transferred into the next generation. Although it has been proved that an elitist operator is important in the success of a GA (Rudolph, 1996), it is not clear as to what degree the operator should be used. In its simplest form application is determined by a percentage (*e*) of members of the current population. However, attention should be paid to defining *e*% because if it is too small, then the influence of elite members will not have a positive effect on the next population. If it is too large, the population may lose its diversity and premature convergence is then possible. Deb (2001) suggests trial and error to define *e*% for a given problem although a commonly used value is 10%. Thus, some initial runs to monitor the improvement in the mean fitness may assist in determining an appropriate *e*%.

Termination criterion

A termination criterion was not used when applying the GA as the goal was to find the best performance balanced by allowing the GA to run for a reasonable amount of computational time.

5 Module toolbar

The *LandParcelS* module is operationalised as a toolbar (Figure 8) consisting of four icons: ‘Optimisation parameters’; ‘Generate Population’; ‘GA Run’; and ‘Outputs’. Each icon launches a separate window with one or more functionalities. Icons appear on the toolbar in the order in which they must be executed. The functionality of each icon is described below.

Figure 8: The toolbar of *LandParcelS*

The first icon launches the window shown in Figure 9. If only one input parameter is to be optimised, then the weights are not required. If more than one parameter is selected, then weights need to be entered that sum to 1. The penalty function is an added term in the fitness function that penalizes infeasible solutions.

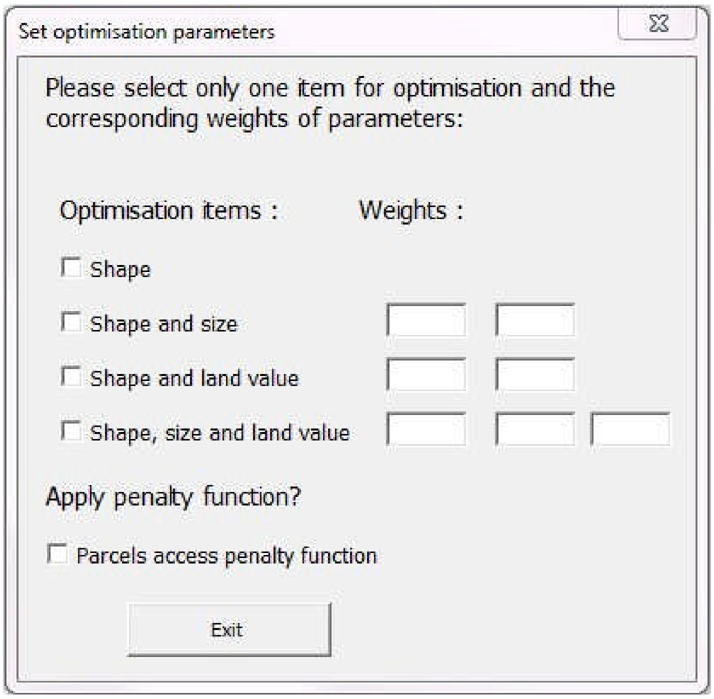

Figure 9: The 'Optimisation parameters' window

Once the optimisation parameters have been defined, the initial step in the evolutionary process is the generation of a random population of solutions by defining which land block will be partitioned and the size of the population, using the relevant icon that launches the window as shown in Figure 10.

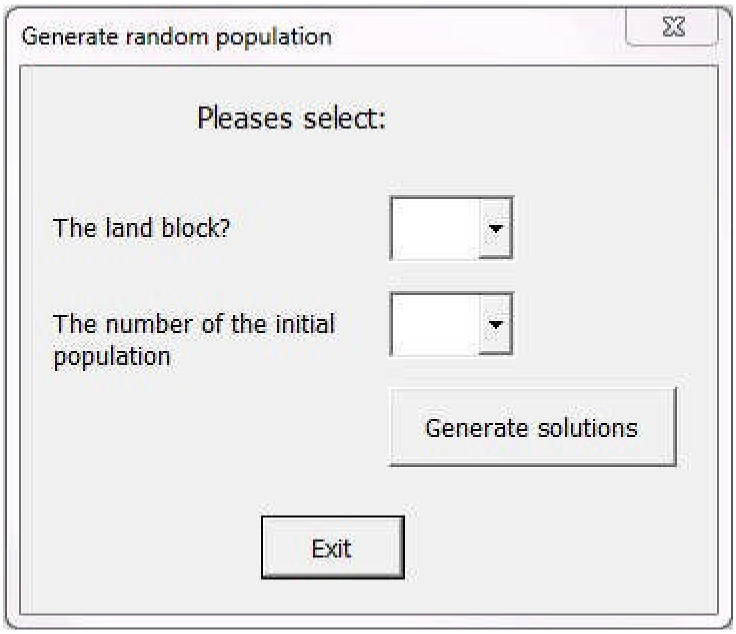

Figure 10: The ‘Generate random population’ window

The GA is then run by defining the number of generations and the elitist factor as explained earlier in the window illustrated in Figure 11, which is launched by the ‘GA Run’ icon.

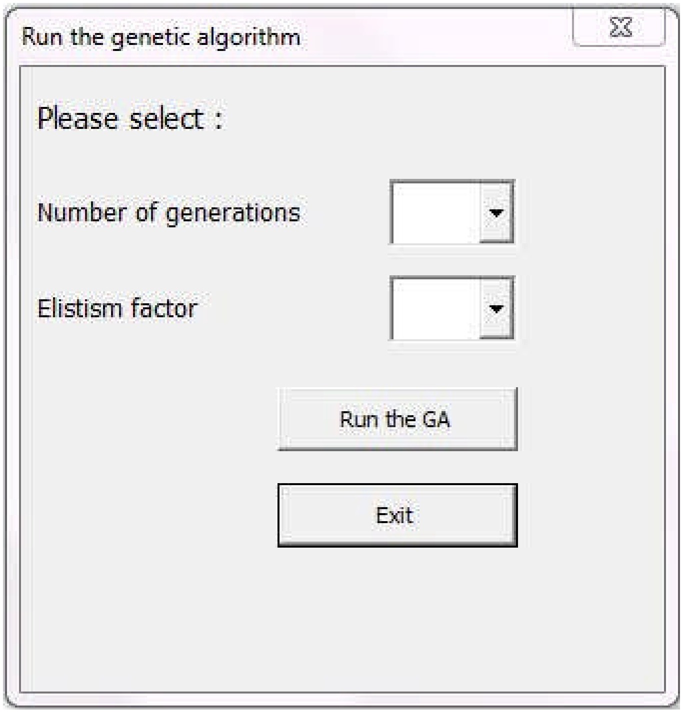

Figure 11: The ‘Run the genetic algorithm’ window

The Outputs icon launches the window illustrated in Figure 12, which displays two database tables: the OFitness table (Figure 13) and the GFitness table (Figure 14). Both tables contain useful information regarding the evolution of the process for each generation. In particular, the OFitness table presents evolutionary statistics for each generation including minimum, maximum and mean values for each objective function (F1, F2 and F3) and the minimum and mean overall fitness. The GFitness table lists the mean value of the objectives functions F1, F2 and F3 and the overall fitness for each solution for the current generation. The graphical outputs consisting of the block subdivision design can be seen by utilising simple functions within the GIS environment. Eventually, the user may store all outputs in a separate folder using e.g. ArcCatalog and delete the solutions by selecting the relevant buttons in the 'Outputs' window.

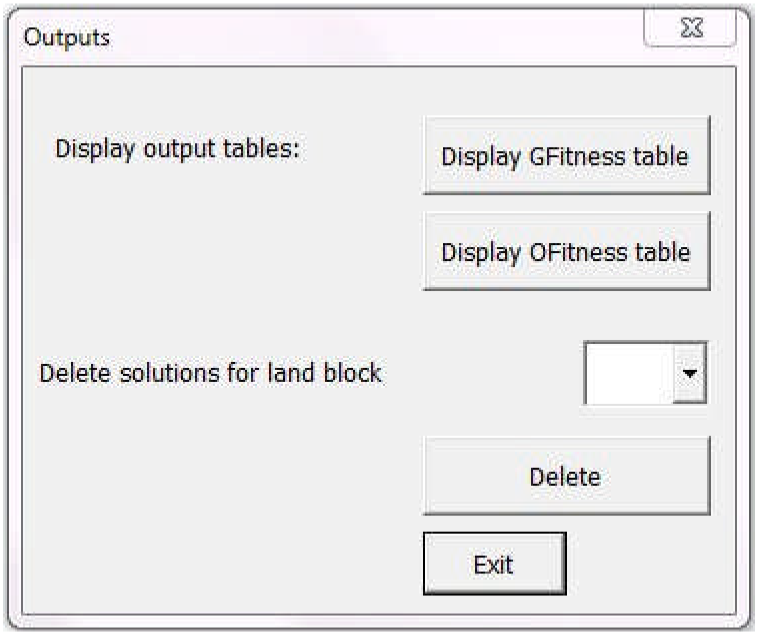

Figure 12: The 'Outputs' window

Attributes of OFitnessTable

OID	PopNo	GenNo	minF1	maxF1	meanF1	minF2	maxF2	meanF2	minF3	maxF3	meanF3	minFitness	Fitness
0	40	0	0,074233	0,387066	0,238912	0,29271	0,989392	0,575359	0,40076	0,800218	0,596825	0,429691	1,333069
1	0	1	0,071402	0,408553	0,242745	0,318427	0,783191	0,55804	0,394321	0,773833	0,592332	0,304602	0,919981
2	0	2	0,075828	0,341667	0,228239	0,326669	0,802181	0,590186	0,414161	0,864735	0,624018	0,301025	0,892796
3	0	3	0,025082	0,322889	0,210387	0,339738	0,967754	0,617994	0,453264	0,767216	0,584024	0,333121	0,886472
4	0	4	0,12082	0,374828	0,230719	0,209616	0,810522	0,574479	0,304027	0,815369	0,571955	0,217449	0,705727
5	0	5	0,073648	0,36788	0,222432	0,334711	0,84385	0,573452	0,419286	0,801238	0,592545	0,305182	0,728248
6	0	1	0,072704	0,369458	0,226046	0,265277	0,909881	0,565925	0,418173	0,846227	0,609527	0,280139	0,697949
7	0	2	0,019548	0,387594	0,221009	0,328412	0,865226	0,575177	0,385619	0,799278	0,563258	0,298775	0,754343
8	0	3	0,084558	0,402845	0,248702	0,3275	0,869341	0,561412	0,322344	0,749105	0,564121	0,321475	0,69887
9	0	4	0,074449	0,405647	0,236937	0,320055	0,752867	0,502896	0,384044	0,732292	0,545648	0,29396	0,799704
10	0	5	0,122211	0,375706	0,21427	0,28822	0,748374	0,511603	0,391969	0,7275	0,537782	0,26936	0,677136
11	0	6	0,075829	0,373572	0,222974	0,309239	0,807521	0,527702	0,408001	0,759978	0,552661	0,317896	0,716757
12	0	7	0,079877	0,389065	0,217886	0,258875	0,889737	0,551171	0,389019	0,700518	0,543584	0,269193	0,784514
13	0	8	0,120164	0,372848	0,235055	0,316248	0,874295	0,559562	0,379542	0,728221	0,543877	0,281474	0,619661
14	0	9	0,076651	0,390053	0,234961	0,267848	0,998132	0,5688	0,432662	0,827853	0,576484	0,292289	0,852032
15	0	10	0,075012	0,32109	0,20511	0,289001	0,995623	0,572392	0,41507	0,791979	0,551766	0,271968	0,773936
16	0	1	0,070287	0,403125	0,204814	0,293008	0,873496	0,533026	0,407641	0,725539	0,518146	0,274498	0,692384
17	0	2	0,077885	0,3575	0,199247	0,257809	0,746581	0,530557	0,405792	0,668164	0,510204	0,246124	0,589295
18	0	3	0,121977	0,382544	0,222348	0,242675	0,830611	0,462583	0,335155	0,777349	0,525004	0,234319	0,614536
19	0	4	0,072561	0,367778	0,210474	0,322349	0,728221	0,47256	0,41542	0,67944	0,52857	0,297757	0,570143
20	0	5	0,023841	0,360216	0,210385	0,290871	0,626867	0,455973	0,400518	0,720178	0,51694	0,269377	0,631855
21	0	6	0,071915	0,369513	0,197556	0,27512	0,68433	0,475904	0,415805	0,642767	0,496061	0,26988	0,545234
22	0	7	0,124482	0,341708	0,207317	0,325184	0,74609	0,457572	0,401268	0,652414	0,499071	0,295755	0,457521
23	0	8	0,079194	0,30641	0,19387	0,266844	0,668004	0,431118	0,373741	0,649547	0,484432	0,258396	0,483669

Record: 1 Show: All Selected Records (0 out of 24 Selected) Options

Figure 13: The OFitness table

Attributes of GFitnessTable

OID	PopNo	SolNo	F1	F2	F3	Fitness
0	8	1	0,168031	0,370139	0,460834	0,329718
1	8	2	0,132458	0,580765	0,584776	1,491103
2	8	3	0,251616	0,479253	0,649547	0,433726
3	8	4	0,095416	0,558062	0,634382	0,465532
4	8	5	0,086253	0,450092	0,482535	1,377324
5	8	6	0,208428	0,462772	0,46096	0,411904
6	8	7	0,249916	0,339977	0,455389	0,321965
7	8	8	0,200543	0,472992	0,54608	0,418502
8	8	9	0,249085	0,315155	0,447229	0,301941
9	8	10	0,079194	0,570359	0,52521	0,472126
10	8	11	0,123388	0,668004	0,483039	0,559081
11	8	12	0,17778	0,366851	0,623058	0,329037
12	8	13	0,186304	0,365781	0,457656	0,329886
13	8	14	0,174739	0,387021	0,468585	1,344564
14	8	15	0,235386	0,458533	0,420346	0,413904
15	8	16	0,203349	0,439581	0,423627	0,392335
16	8	17	0,185219	0,471769	0,454169	0,414459
17	8	18	0,252562	0,461034	0,625246	0,41934
18	8	19	0,122987	0,620135	0,501141	0,520706
19	8	20	0,126281	0,444982	0,432649	0,381242
20	8	21	0,189027	0,341771	0,458446	0,311222
21	8	22	0,231403	0,338546	0,498293	0,317117
22	8	23	0,182962	0,277254	0,466555	0,258396
23	8	24	0,213556	0,491743	0,445967	1,436106
24	8	25	0,250628	0,266844	0,464393	0,263601
25	8	26	0,30641	0,511432	0,609653	0,470427

Record: 1 Show: All Selected ecords

Figure 14: The GFitness table

6 Application of the model

For the application of the model, we selected two typical land blocks (they are surrounded by roads and they have quite regular shapes) of the case study land consolidation area presented in Demetriou *et al.* (2010), reflecting a different complexity of land partitioning. Problem complexity is defined by three main factors: the number of parcels that need to be created, the size of the search space and the shape of the block. In particular, block B25 was selected, which involves six parcels and its size equals about 3 ha, and block B14, which involves 10 parcels with a size of around 5 ha. The tests that follow reveal the behaviour of the algorithm for solving these problems for different optimisation cases in the context of both single and multi-objective approaches.

6.1 Single-objective land partitioning

Shape optimisation for land block B25

When partitioning is carried out by utilising the Thiessen polygons tool without any optimisation process, the result is shown in Figure 15a. This solution and similarly for land block B14 is referred to as the initial subdivision where the parcel shapes are defined according to the principles of Thiessen polygons. As a result, parcel shape depends entirely upon the layout of the centroids and is therefore neither necessarily regular nor optimum. The relevant metrics for the three objective functions are the following: F1 (0.264), F2 (0.634), F3 (0.621) and R (0).

On the other hand, if land partitioning is treated as a single optimisation problem aimed at generating parcels with regular shapes, then the best subdivision is illustrated in Figure 15b. Further to the considerable visual improvement of parcel shape, the overall fitness outcome is very close to zero (0.073), which represents an improvement of 72.4% compared to that of the initial subdivision. In particular, all parcels have a PSI greater than 0.7 with the exception of parcel with ID 15 which has a slightly lower PSI, namely, 0.673. As the external shape of the land block is not involved in the optimisation process, it cannot change, and therefore the parcel shapes limit the amount of regularity achievable.

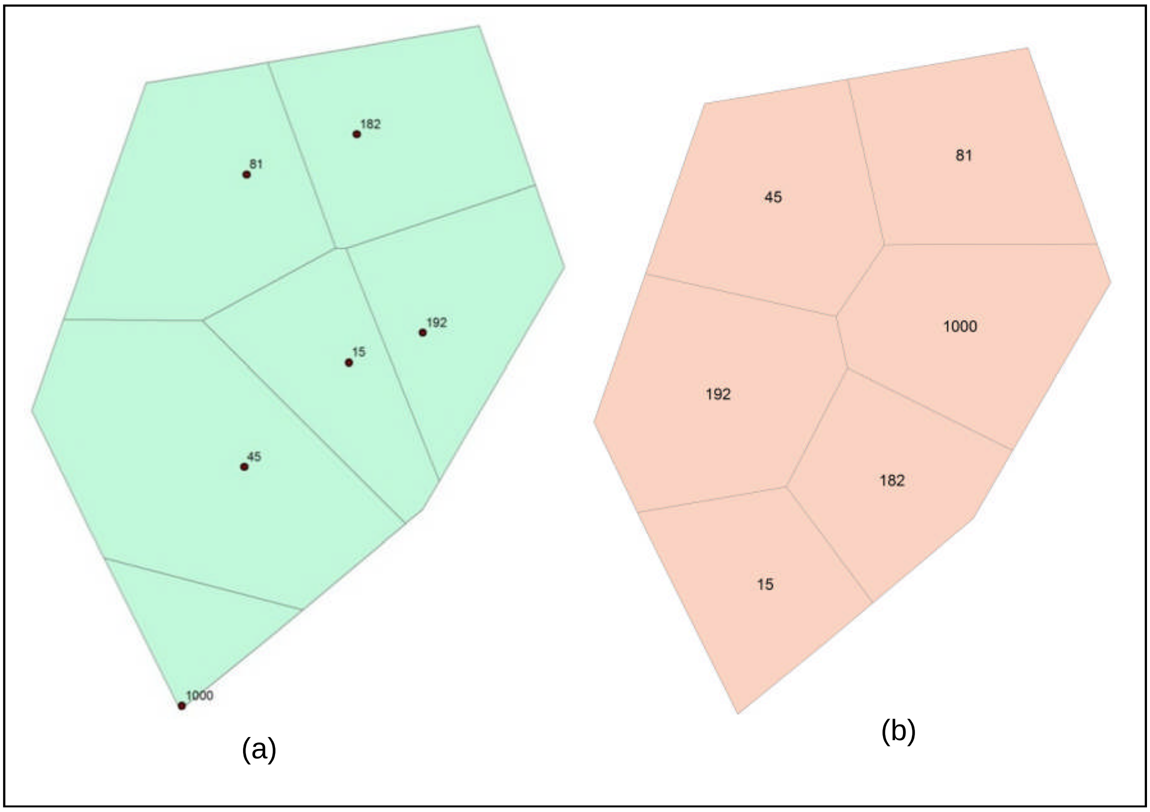

Figure 15: Subdivision of land block B25 by utilising (a) regular Thiessen polygons and (b) the optimisation algorithm

In addition, the way in which Thiessen polygons work may prevent the absolute optimality of shapes. For example, an expert would draw a straight line (Figure 15b) from the internal point of the parcel with ID 15 to the internal point of the parcel with ID 1000, to separate parcels with IDs 45, 81 and 192, 182. This is compared to a polyline with three segments and four points as the optimisation outcome. To avoid this weakness in the algorithm, appropriate modifications to the code of the Thiessen polygon process would be necessary. For example, forcing a new polygon to have the same corners as those of an existing adjacent polygon(s) on a polygon by polygon design process would be one way of solving this. However, this means not using Thiessen polygons for this process in the original manner in which it was designed. However, it is not possible to modify the internal code for generating Thiessen polygons in ArcGIS so a new algorithm would need to be written from scratch. However, we have demonstrated that the algorithm may successfully use Thiessen polygons to create polygons with regular shapes.

Furthermore, in order to undertake a more in-depth investigation of the behaviour of the algorithm and the effects of its main operators in the evolutionary process, the program ran for six different sets of parameters as follows:

Case I: no elitist, block based mutation, no penalty function

Case II: e=10%, no mutation, no penalty function

Case III: e=10%, block based mutation, no penalty function

Case IV: e=10%, parcel based mutation, no penalty function

Case V: e=10-40%, block based mutation, with penalty function

Case VI: e=40%, parcel based mutation, no penalty function

For all cases, the population size is set to 40. The population takes 10 minutes to be created and consists of 1/3 feasible and 2/3 infeasible solutions. It should be noted that before defining the size of the initial population to 40, some trials with a smaller population size, e.g. 20, and larger size, e.g. 60 and 80, were carried out. The former is too small and hence the algorithm cannot converge whilst the latter increased the computational time too much. Therefore, the population size was set to 40 for all optimisation cases presented here. It is worthwhile mentioning that a similar population (with 50 members) has been used by Datta *et al.* (2006) for a similar spatial problem (land use management). In addition, Krzanowski and Raper (2001) suggest a population size of 40 to 80 for spatial problems.

A detailed representation of the behaviour of the GA for the cases I to V is illustrated in Figures 16-20, respectively, showing four evolutionary statistics, namely, minimum, maximum, mean values of F1 and the overall fitness for each generation. The latter is involved only when the penalty function is added to the fitness measure. Otherwise the overall fitness equals the mean F1.

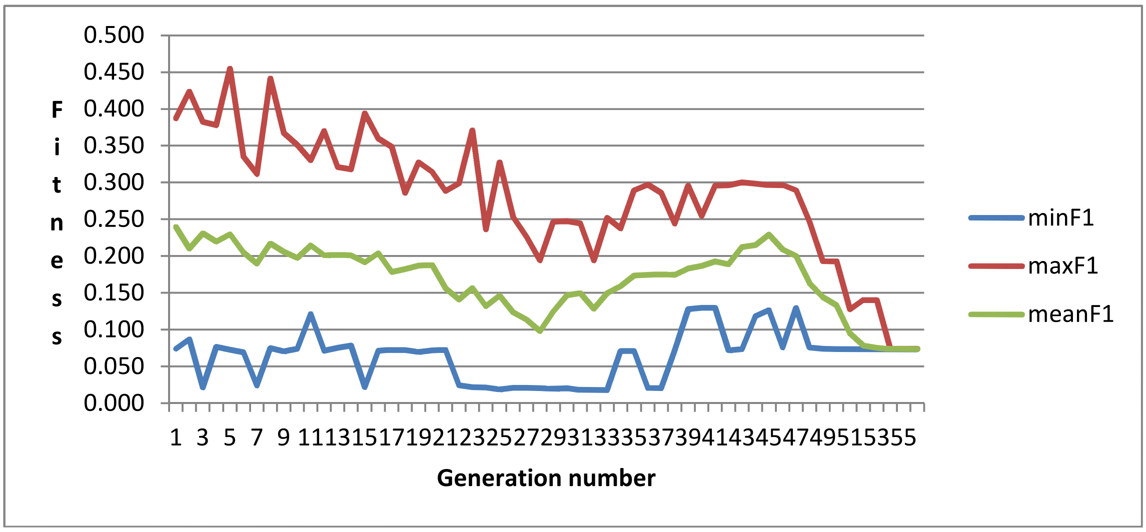

Figure 16: The evolutionary statistics for case I

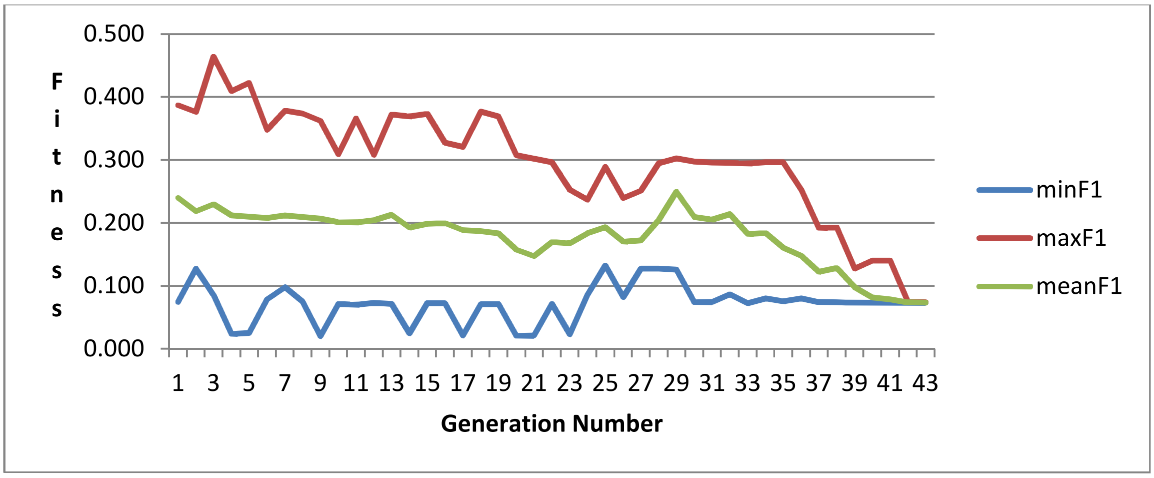

Figure 17: The evolutionary statistics for case II

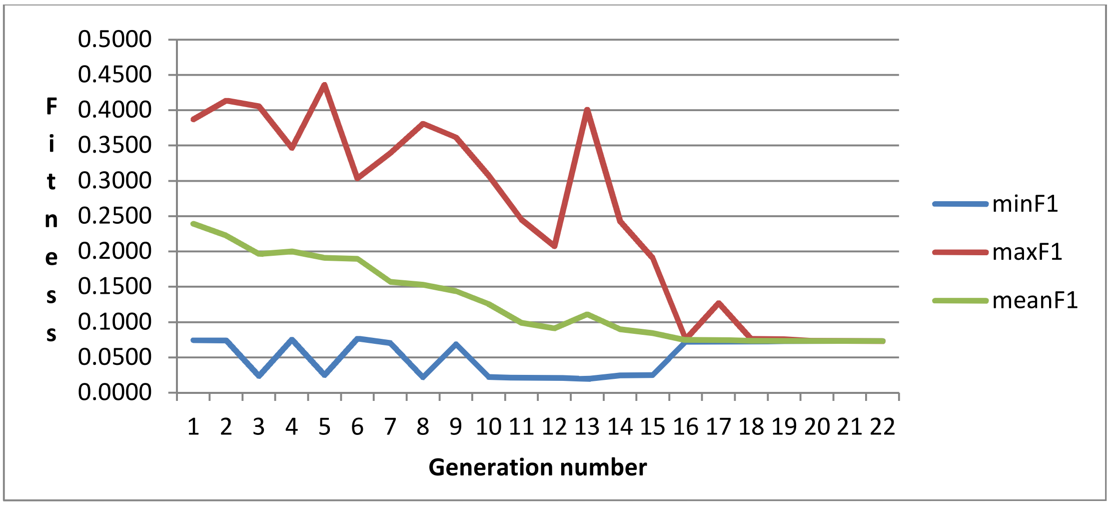

Figure 18: The evolutionary statistics for case III

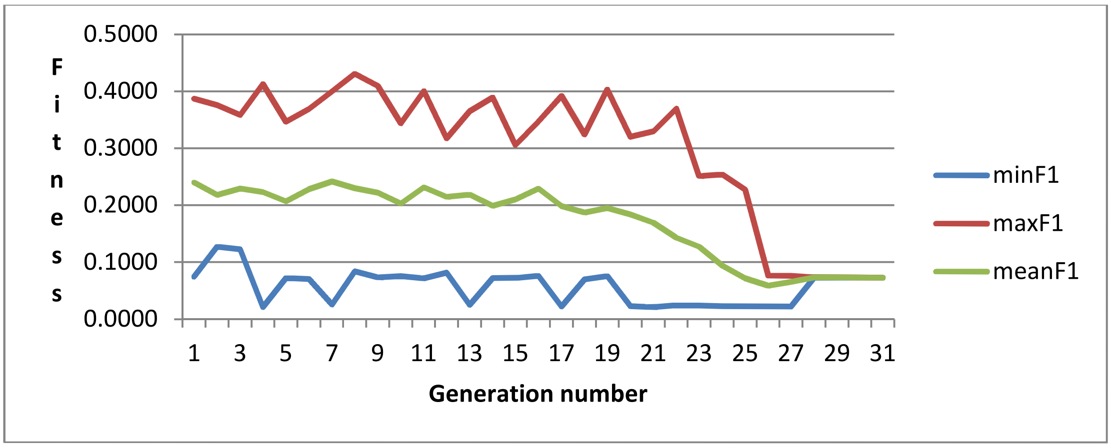

Figure 19: The evolutionary statistics for case IV

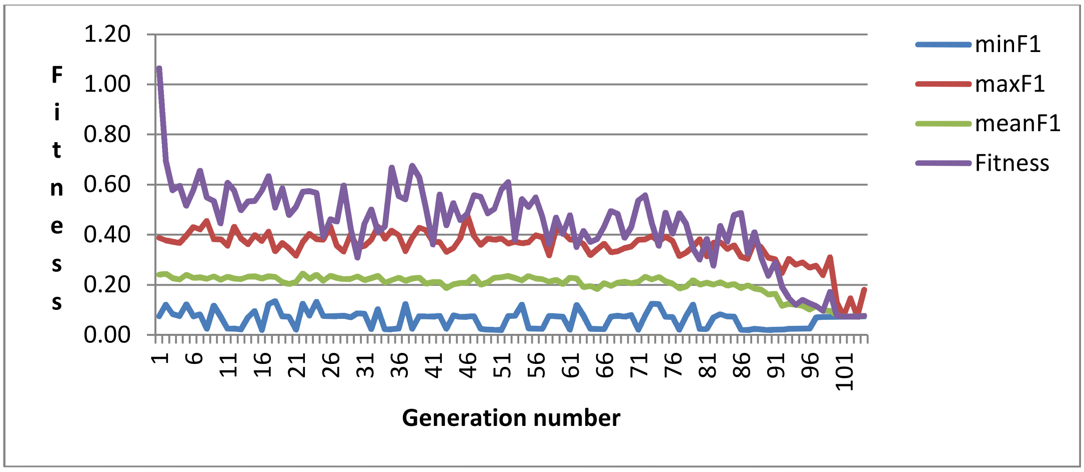

Figure 20: The evolutionary statistics for case V

Some interesting findings can be extracted from these tests. In particular, the fastest convergence of the algorithm was achieved in case III (Figure 18), in the 18th generation in 4.8 hours which uses a 10% application rate for the elitist block based mutation and no penalty function. Initially the mean fitness score is 0.24 but gradually falls to an optimum of 0.07. It is only the mean F1 line (among the above figures) that presents an almost continuous improvement until the optimum solution is found. This improvement is combined with the largest fluctuations of maxF1 that have been caused by the block based mutation, and which may create considerably worse solutions than the existing ones compared to what parcel based mutation may cause (Figure 19). In parallel, the smallest minF1 value has risen very fast (in generation 10) which is a local optimum non-feasible solution (as we saw in the subdivision design of that solution) and remains stable for several generations. Then the search moves out of this local optimum through the crossover operator and quickly reaches the optimum feasible solution. In the other three cases (I, IV and V) the smallest minF1 value rises later, (in the 21st, 20th and 86th generations respectively) while this is not observed in case II where no mutation is applied.

As expected, in such a single-objective problem, all members of the population converge to one optimum solution and the minF1, meanF1 and fitness level off (Figure 18) after 18 generations. In addition, it is notable that case I, which involves the same parameters as those of case III except for the use of the elitist operator, took a longer time to converge than the latter (in the 55th generation which lasted 14.67 hours). This finding highlights the importance of the elitist operator for speeding up the process, in this example by 9.87 hours. However, the elitist operator should be introduced very carefully because if it is high (e.g. in case IV), it may lead to premature convergence in a non-optimum solution since the diversity of the population is lost. Furthermore, it is worthwhile mentioning that case II, which involves a block based mutation, converged faster by nine generations compared with case III, which involved a parcel based mutation. However, it was ranked as having the second fastest convergence (in 27 generations which lasted 7.2 hours). Despite this, it seems that the block based mutation converges faster; however, this finding is not repeated in the second example involving block B14. Moreover, it is notable that convergence can be also achieved even without mutation but in a much longer time as shown in case II (in 42 generations that took 11.2 hours). This happens, because the BLX-a crossover operator is powerful in maintaining the diversity of the population without the involvement of a mutation operator, although the time is significantly increased.

Another noticeable outcome is that when the penalty function was included in the fitness, the convergence of the algorithm was considerably extended, (in 102 generations that lasted 27.2 hours), achieving the maximum computational time (hence the worst case) for this test. This is due to the fact that a penalty function distorts the fitness measure through penalising non-feasible solutions by adding a number to the overall fitness. As a result, the overall fitness and the other metrics as well present continuous fluctuations of varying degrees (Figure 20) until all solutions become feasible and eventually converge to the optimum. Therefore, penalty functions need to be treated carefully in terms of the value(s) of the constraint violation(s) so as to steer the search towards the feasible region (Deb, 2001) and should be used only if it is really necessary. For example, in this situation the penalty function was not necessary because the algorithm was able to lead the search into the feasible region without it.

Shape optimisation for land block B14

The initial subdivision for land block B14 without optimisation is shown in Figure 21a. The relevant metrics for the three objective functions are the following: F1 (0.221), F2 (0.957) and F3 (0.610) and R=1 meaning that the solution is not feasible since the parcel with ID 159 has no access from a road.

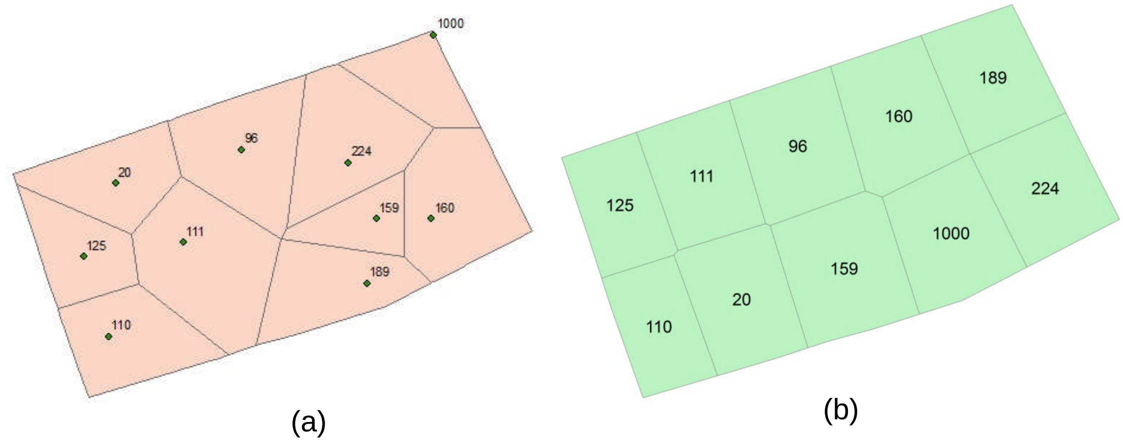

Figure 21: Subdivision of land block B14 by utilising: (a) regular Thiessen polygons and (b) the optimisation algorithm

On the other hand, the best subdivision generated by optimisation is illustrated in Figure 21b for which F1 equals 0.019, which represents an improvement by 91.4% compared to the initial subdivision, further to the considerable visual improvement. In particular, all parcels have a PSI greater than 0.7. This subdivision is exactly what a human expert would do for designing parcels with regular shapes, i.e. symmetrically divide the block with a line in the middle and then design the parcels vertically to each road side. However, as in the previous example with block B25, the algorithm fails to draw a perfect straight line in the middle of the block. In particular, the algorithm presents a weakness to do that so the joins of parcels are not identical. This limitation suggests again that the process utilised for generating polygons through Thiessen polygons needs additional guidance. Despite this limitation, it is clear again that the algorithm is able to reach a near optimum solution for even a more complex land partitioning problem. In this case, the convergence achieved is better than that for block B25 and reached very close to the absolute optimum, because in contrast to block B25, the boundary of block B14 is almost rectangular.

For a more in-depth investigation of the performance of the algorithm, the model ran for the following five cases representing different sets of parameters:
Case I: e=10-40%; block based mutation, no penalty function
Case II: e=10-40%; no mutation, with penalty function
Case III: e=10-40%; parcel based mutation, with penalty function
Case IV: e=10-40%; block based mutation, with penalty function
Case V: no elitist, block based mutation, with penalty function

Initially, the algorithm ran (case I) without the penalty function as we did with block B25 but a feasible solution could not be created even after 50 generations. Thus, the penalty function was introduced in the fitness measure with F1 for the next three cases. In addition, taking into account the previous experience gained from the behaviour of the algorithm for case V that includes a penalty function, a varying elitist factor from 10 to 40% was included, to speed up the process as much as possible whilst avoiding premature convergence. Therefore, in cases I to IV, the elitist factor begins with a value of 10% and gradually increases to 20%, 30% and 40% when the number of feasible solutions exceeds a certain percentage. For instance, when the number of feasible solutions in a population of 40 members exceeds 20%, e.g. 8 out of 40 solutions are feasible, then the elitist factor is set to 20% and so on. This trick was also necessary because the initial random population had only a few feasible solutions, and hence a strategy to preserve feasible solutions in the next generations was necessary.

A detailed representation of the behaviour of the GA for cases II, III, IV and V is illustrated in Figures 22-25 respectively. The initial population (40 random solutions), which took 23 minutes to be created, consisted of only a few feasible solutions (3-6) out of 40 and as a result, the initial mean fitness score is very high for all cases (around 2.5). The best outcome obtained in case III occurred after 42 generations (12.0 hours) although the other two cases (case II and case IV) converged shortly afterwards with 43 (12.3 hours) and 46 (13.2 hours) generations, respectively. This finding is sharply in contrast to the previous example for block B25, which showed that block based mutation considerably speeded up the process compared with that involving parcel based mutation or even more with no mutation at all.

Thus, this case agrees with the findings of Krzanowski and Raper (2001) that the mutation operator has no effect on the evolutionary search in spatial problems. However, it is not possible to apply this statement to all spatial problems and there may be situations where the mutation operator has a considerable influence on the evolutionary process depending on the particular features of a problem and the other optimisation parameters set out. In addition, the mutation operator is always useful for maintaining the diversity of a population from generation to generation, especially if the crossover used does not have this ability.

In terms of the evolutionary statistics, all of the cases shown in Figures 22-25 present a very similar picture regarding the minimum, maximum and mean values of F1. In particular, they remain stable across the whole evolution with very small fluctuations until the last few generations before their convergence to the optimum solution. This is in contrast with what happened in cases with similar parameters for land block B25, where in many cases they present significant fluctuations. It seems that this phenomenon is due to the shape of the land block B14 which is almost rectangular, hence not favouring considerably worse or better solutions.

Another interesting finding is that without the use of an elitist operator, convergence was achieved after 62 generations (17.77 hours), whilst after the introduction of a varying elitist factor, the evolutionary process was speeded up by 32.3%, again indicating the importance of this operator.

The computational time needed to achieve the convergence (12.0 hours and 4.8 hours in the two examples) is very high compared to the time a human expert could design near optimum subdivisions in terms of parcel shape. The reason is that the human brain can easily perceive a rectangular shape from an irregular shape or symmetrical shapes (Delahaye, 2001) but for a computer, this remains a difficult problem. This happens for many complex problems related to spatial planning or engineering design because the evaluation of the fitness function is time consuming (Renner and Ekart, 2003, Stewart *et al*., 2004). For example, in a land use management multi-objective problem (Datta *et al.,* 2006), the algorithm needed 5000 generations and took 3.82 days to converge. In addition, the simulated annealing algorithm of Tourino *et al*. (2003) needed 10,000 stages (time is not noted) for solving a land partitioning with a block involving five parcels. The computational time could be considerably reduced by employing parallel computing that permits the parallel evaluation of individuals and the other computations so as to solve large and difficult problems in a reasonable time (Renner and Ekart, 2003).

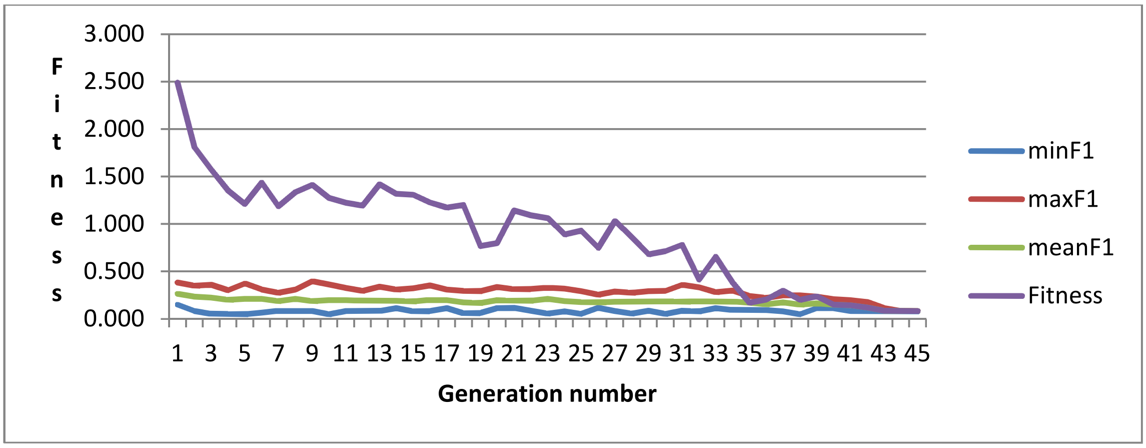

Figure 22: The evolutionary statistics for case II

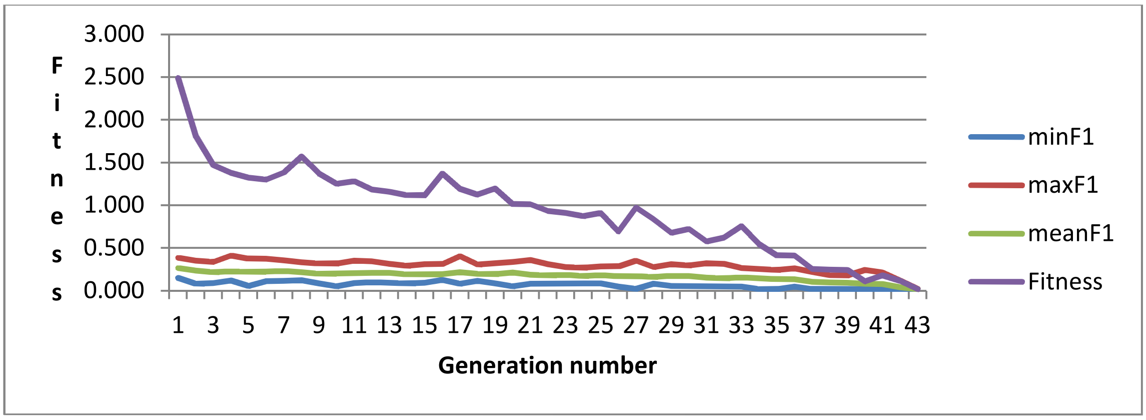

Figure 23: The evolutionary statistics for case III

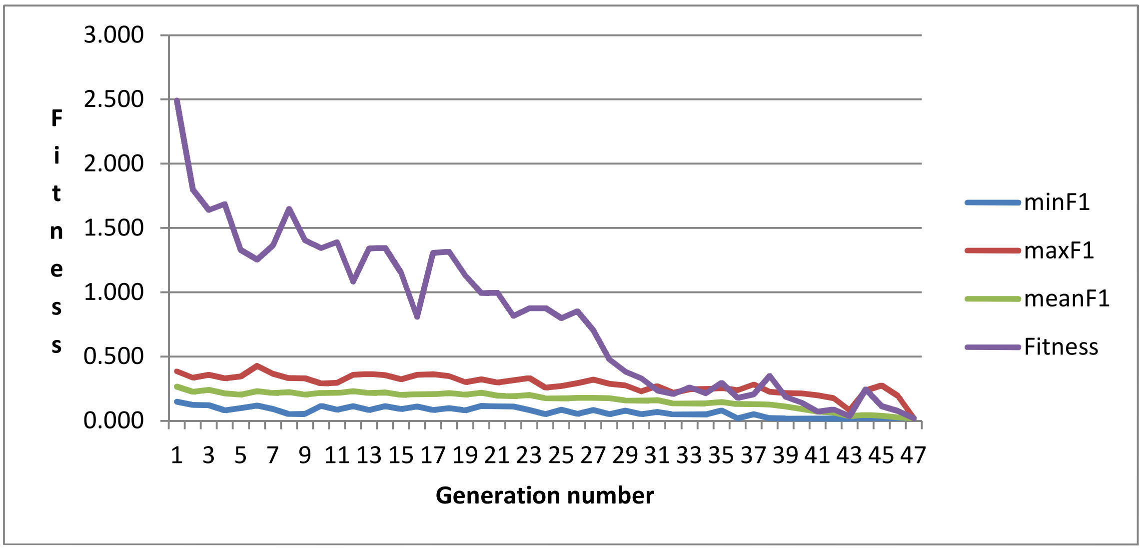

Figure 24: The evolutionary statistics for case IV

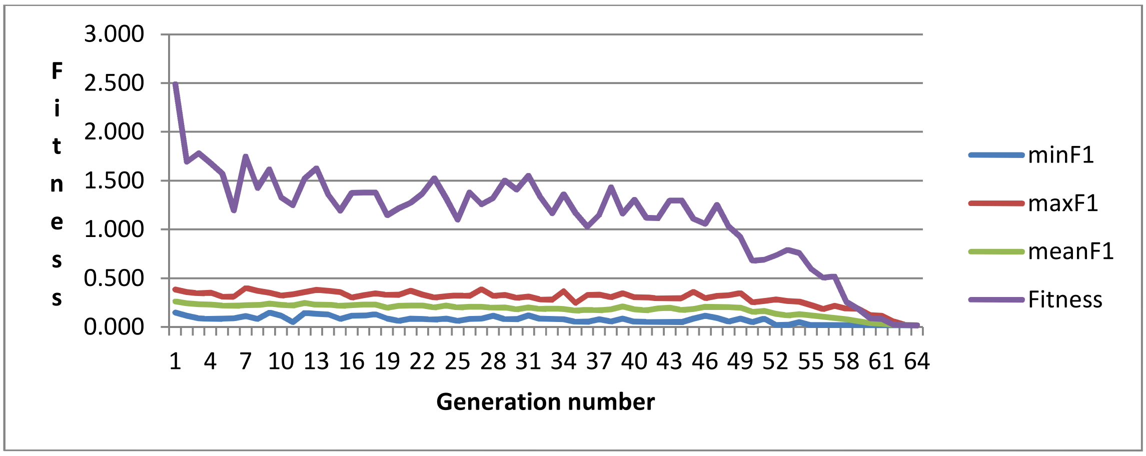

Figure 25: The evolutionary statistics for case V

6.2 Multi-objective land partitioning

As already noted, in a multi-objective problem with conflicting objectives there is not a single optimum solution. Instead, there are a number of optimal solutions that lie on a curve called the Pareto-optimal front. None of these solutions is the best in all the objectives involved and thus the planner may decide which solution is best only if he/she defines the importance of each objective. We treated the land partitioning problem based on three optimisation cases: shape and size (F1 and F2); shape and land value (F1 and F3) and shape, size and land value (F1, F2 and F3). These cases have been applied to blocks B25 (the best outcomes shown in Figure 26) and B14 (the best outcomes shown in Figure 27). The Pareto-optimal front for each case is presented in Figures 28-32. These include the solution with the minimum overall fitness (final population) and a few other selected populations having a fitness value close to the minimum.

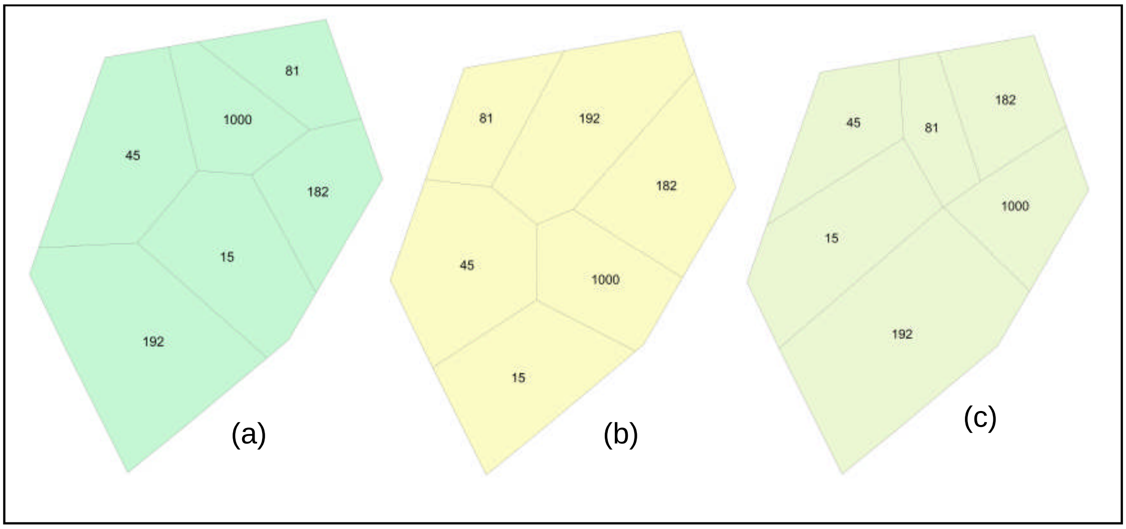

Figure 26: Multi-objective optimisation outcomes for land block B25: (a) shape and size; (b) shape and land value; (c) shape, size and land value

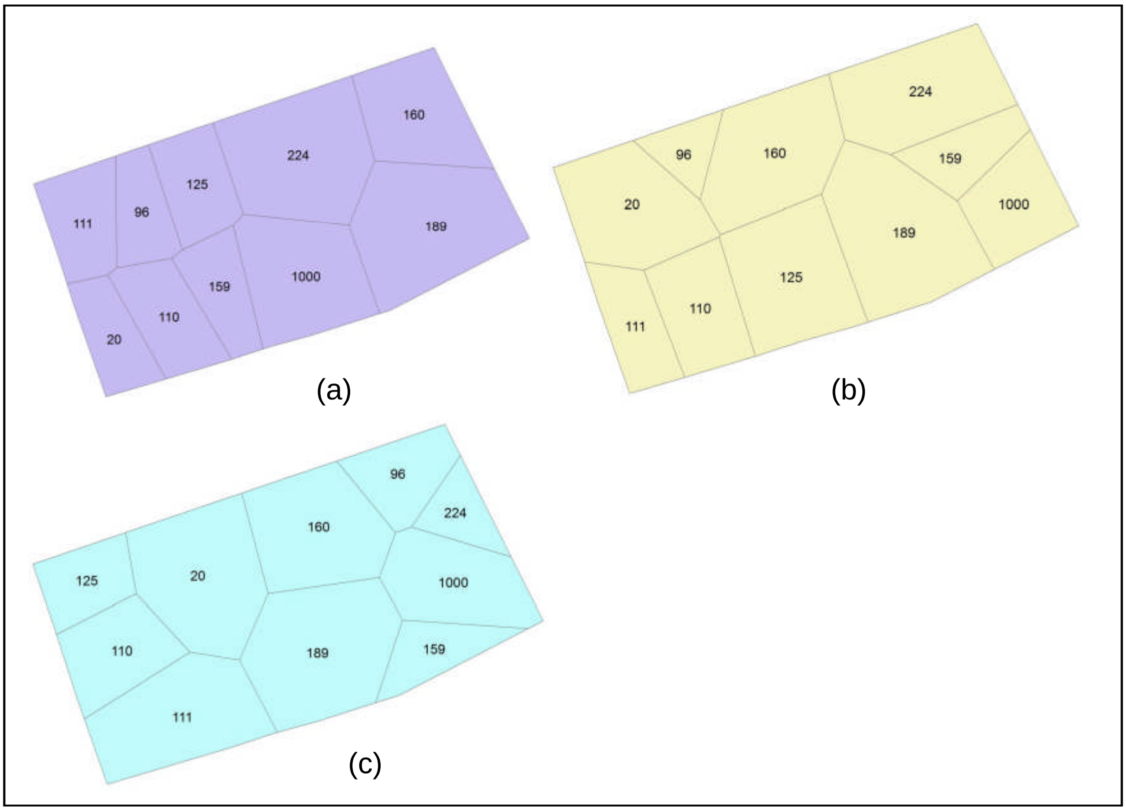

Figure 27: Multi-objective optimisation outcomes for land block B14:
(a) shape and size; (b) shape and land value; (c) shape, size and land value

Minimise shape and size (F1 and F2)

In this case we consider two conflicting objectives: minimise F1 and F2. Figures 28 and 29 shows a set of trade off solutions between the two objectives for land block B25 and B14 and the Pareto-optimal front, respectively. The proof that the two objectives are conflicting is the existence of Pareto-optimal solutions. Solutions that lie on the Pareto-optimal front are all feasible whilst solutions that fall within the non-Pareto optimum front region can be either feasible or infeasible. Taking into account that the earlier results showed that the algorithm can satisfactorily produce regular shapes, we wish to test the performance of the algorithm in minimising objective F2 and therefore assign (in equation 5) a high weight value of 0.8 to F2 and a low weight of 0.2 to F1 whilst ignoring F3. The penalty function for infeasible solutions is also involved.

In the case of block B25 the best solution (Figure 26a), which is the solution that dominates all the others based on a certain weighting scheme, is that with the lowest F2 value; this is marked with a triangle and falls on the Pareto-optimal front, which is marked as a dashed line (Figure 28). All the other solutions belong in the non-Pareto-optimal set. The best solution resulted in F1 of 0.181, F2 of 0.094 and overall fitness of 0.112 meaning that: F1 has been improved by 31.44% and F2 by 85.17% compared to the initial subdivision. Furthermore, the parcel shape (F1) and size (F2) are on average only 18.1% and 9.4% from the optimum, respectively. These are very encouraging results because the PSI is on average 0.819 and the variation of parcel size is within the acceptable range in practice (±10%) suggesting that if a guidance operator was utilised to create the parcels, then the Pareto-optimal front will be shifted even closer to the origin point of the two axes, hence to the optimum solution.

In the case of block B14, the results are slightly worse compared to those of block B25 because of the higher complexity of the former block. In particular, the best solution (Figure 27a) is shown again with a triangle (Figure 29) that falls at the bottom of the Pareto-optimal front and has an overall fitness of 0.298, F1 of 0.089 and F2 of 0.35. This represents an improvement of 59.73% and 63.43% in F1 and F2 respectively, compared with the initial subdivision. It can be also said that parcel shape (F1) and size (F2) are on average far from the optimum by 8.9% and 35.0%, respectively. The latter outcome regarding the size of the parcels exceeds the desirable variation noted above emphasising the need for improving the performance of the algorithm for more complex land partitioning problems.

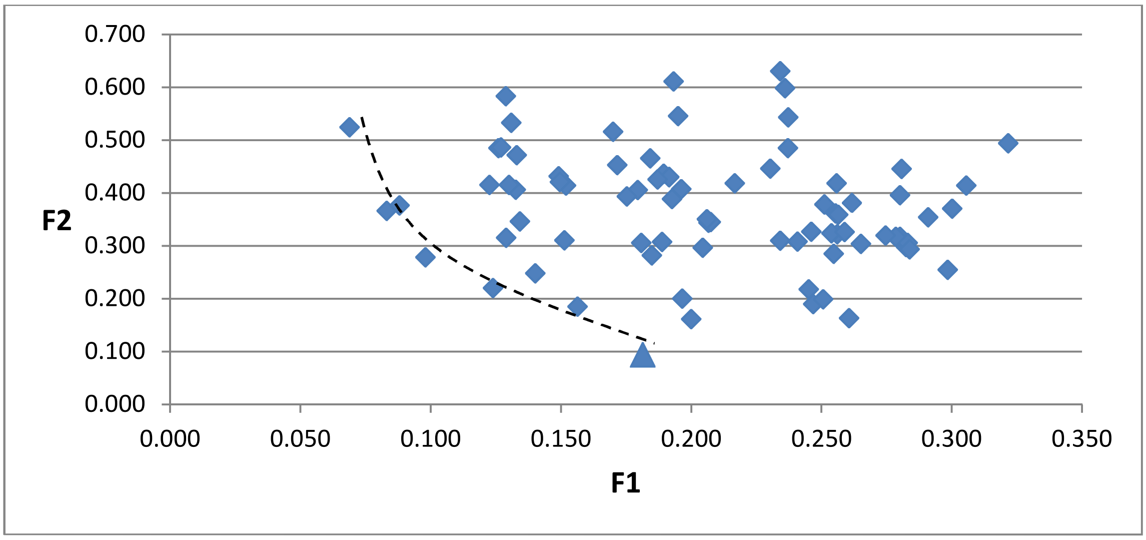

Figure 28: A set of solutions and the Pareto-optimal front for land block B25

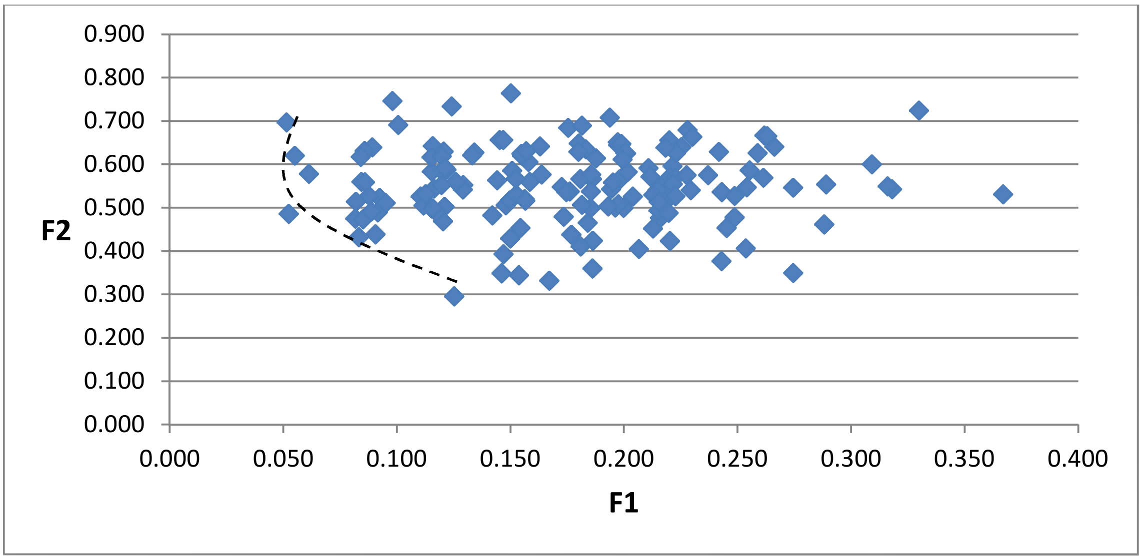

Figure 29: A set of solutions and the Pareto-optimal front for land block B14

Minimise shape and land value (F1 and F3)

Similar to the outcome for minimising the shape and the size of the parcels, the results for minimising the shape and the land value for land block B25 are encouraging. In particular, the best solution (Figure 26b) marked in the Pareto-optimal front (Figure 30) has a fitness of 0.130, F1 of 0.079 and a F3 of 0.142 meaning that F1 has been improved by 70.07% and F3 by 77.13% compared to the initial subdivision. The parcel shape (F1) and land value (F3) are on average far from the optimum by 7.9% and 14.2%, respectively.

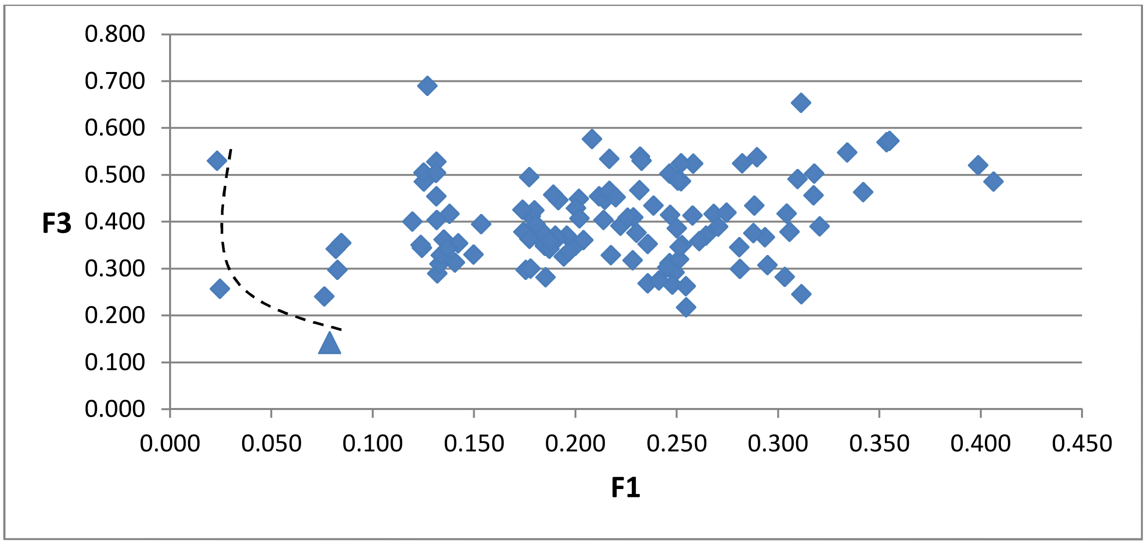

Figure 30: A set of solutions and the Pareto-optimal front for land block B25

As expected, the outcome for land block B14 is not as good as that for land block B25. In particular, the best solution (Figure 27b) marked in the Pareto-optimal front (Figure 31) has an overall fitness of 0.281, F1 of 0.128 and F3 of 0.319. This indicates that F1 has been improved by 42.1% and F3 by 47.7% compared to the initial subdivision. Moreover, the parcel shape (F1) and land value (F3) are on average far from the optimum by 12.8% and 31.9%, respectively.

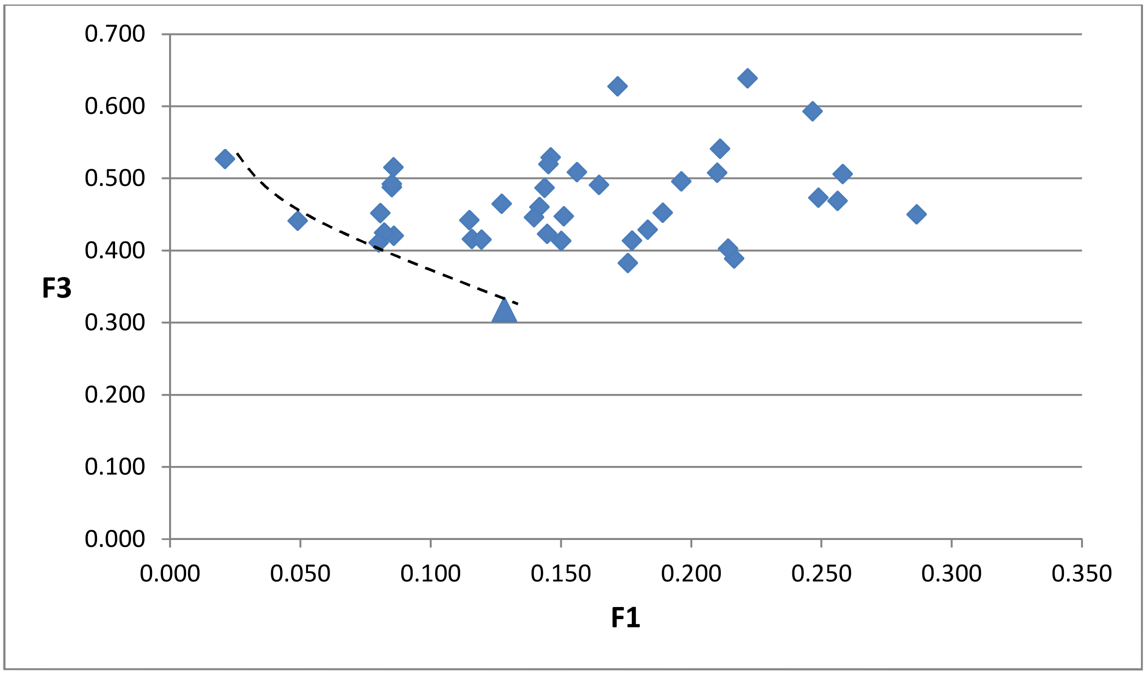

Figure 31: A set of solutions and the Pareto-optimal front for land block B14

Minimise shape, size and land value (F1, F2 and F3)

The best solutions for simultaneously optimising shape, size and land value of parcels for block B25 and B14 are shown in Figures 26c and 27c, respectively. In addition, Figures 32 and 33, show in a 3D plane the projection of a set of solutions with respect to the three objective functions, F1, F2 and F3 for both blocks B25 and B14, respectively. The best solution for land block B25 (marked with a triangle in Figure 32) resulted in the following metrics: Fitness (0.143), F1 (0.138), F2 (0.176) and F3 (0.113) involving an average improvement of F1, F2 and F3 by 47.7%, 72.2% and 81.8%, respectively. In other words, F1, F2 and F3 are on average far from the absolute optimum by 13.8%, 17.6% and 11.3%, respectively. Despite the complexity of simultaneously optimising three objectives, the results are very encouraging. It is obvious from Figure 32 that several trade-off solutions from the cloud of points representing the overall fitness (large points) are close to the origin of the three axes that reflect the optimum solution. It should also be noted that in this case, the size (F2) and the land value (F3) are not conflicting but correlated as shown below.

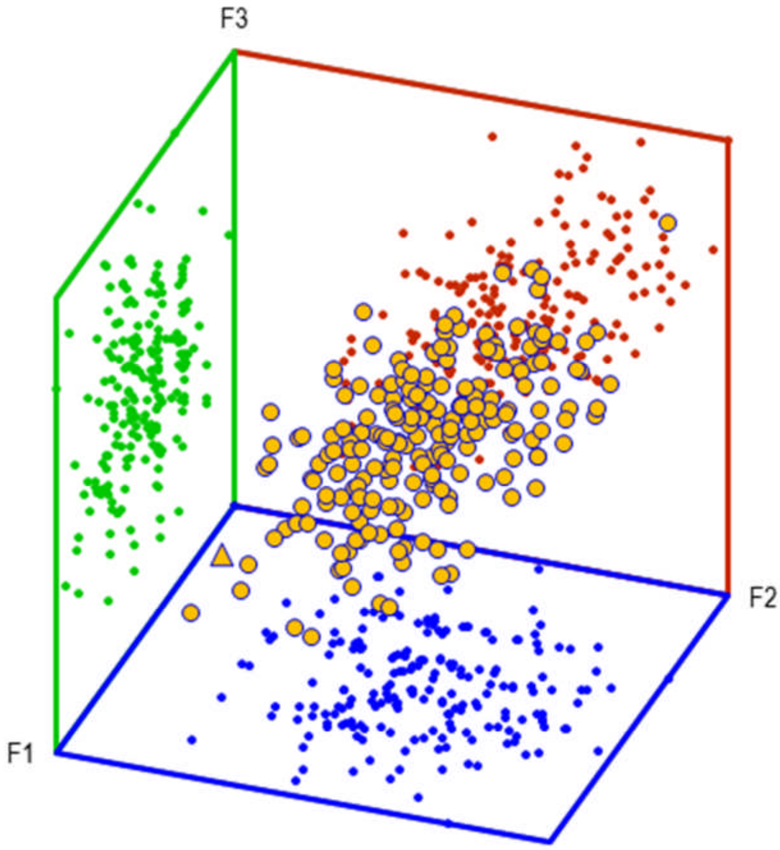

Figure 32: A set of solutions for simultaneous optimisation of parcel shape, size and land value for land block B25

In the case of block B14, the outcome is worse as expected although it is in general moderate. In particular, the best solution gave the following results: overall Fitness (0.332), F1 (0.193), F2 (0.355) and F3 (0.378). This represents an improvement in each optimisation parameter compared with the initial subdivision of 12.7%, 62.9% and 38.0%, respectively. In other words, F1, F2 and F3 are on average far from the absolute optimum by 19.3%, 35.5% and 37.8%, respectively. This result is reflected graphically in Figure 33 where the cloud of points representing the overall fitness is quite far from the origin of the three axes that reflect the optimum solution.

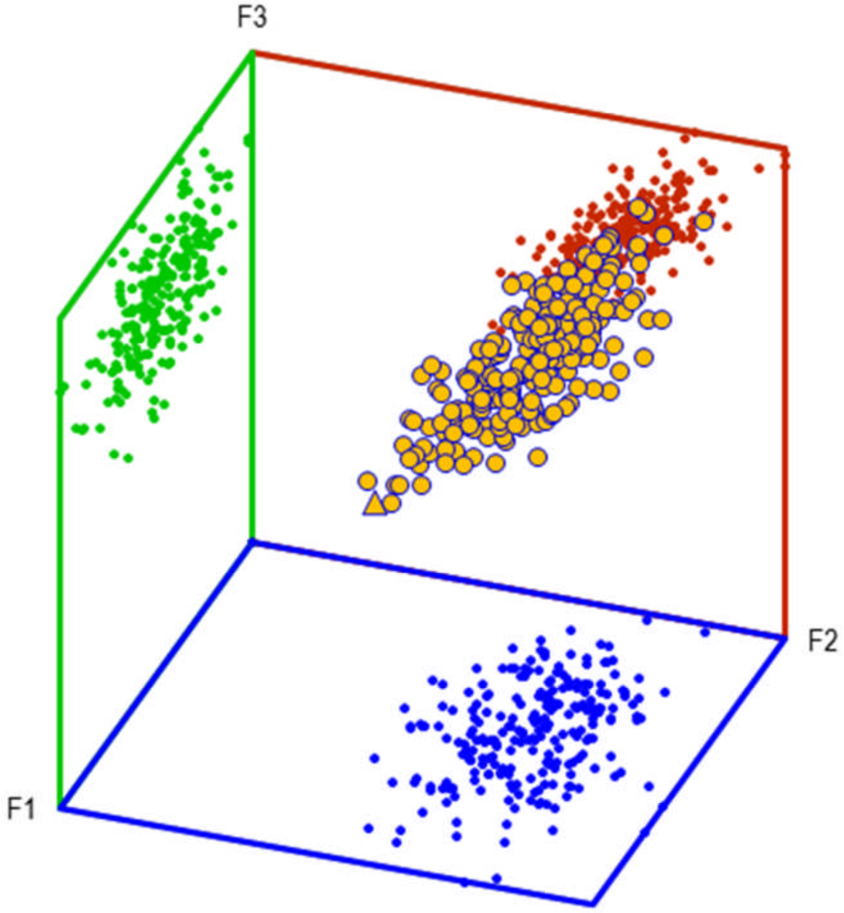

Figure 33: A set of solutions for simultaneous optimisation of parcel shape, size and land value for land block B14

7 Conclusions

This paper discusses the design, development and evaluation of a new model called *LandParcelS*, which integrates GIS and genetic algorithms for automating the land partitioning problem. Land partitioning is a very complex spatial multi-objective problem that involves the optimisation of three parameters, i.e. parcel shape, size and land value which are represented by separate minimization objective functions. In addition, the problem is subject to the constraint that each new parcel should have access from a road. To date this problem has not been solved satisfactorily in an automated fashion.

The model was applied to two land blocks of different complexity. These examples were chosen from a real land consolidation case study area and the land partitioning was treated as both a single and multi-objective problem. The evaluation of the performance of the algorithm was based on metrics that addressed three critical questions: How far is the obtained solution(s) from the absolute optimum? How much improvement has been achieved compared to the original solution, that is, the partition which is automatically obtained by the regular Thiessen polygons process? How much time is needed to arrive at a reasonable solution? Thus, the evaluation of the model, which considered these critical questions, suggests the following findings.

In the case of single optimisation, involving optimising the shape of parcels, the results are near optimum, that is, close to zero (0.07 and 0.019) for both land blocks. Therefore, the algorithm may successfully steer the Thiessen polygon process to generate polygons with regular shapes. This may have relevance to spatial problems. Thus, the PSI has been shown to be an efficient and reliable index for evaluating parcel shapes.

In the case of multi-objective optimisation with two objectives, namely, the shape and size or shape and land value results, present a different picture depending on the complexity of the block. In particular, for the block with the lower complexity, the outcome is fairly close to the optimum, namely the optimisation of the shape and the size of the parcels are on average far from the absolute optimum by 18.1% and 9.4%, respectively; and for the second case, optimisation of the shape and the land value of the parcels, are on average far from the absolute optimum by 7.9% and 14.2%, respectively. In contrast, for the block with the higher complexity, the outcomes are further from the optimum in the case of size and land value. In particular, in the case of optimisation of shape and size of parcels are on average far from the absolute optimum by 8.9% and 35.0%, respectively and for the case of optimisation of shape and land value of parcels are on average far from the absolute optimum by 12.8% and 31.9%, respectively. Similarly, in the case of multi-objective optimisation with three objectives (shape, size and land value), the results present a different picture depending on the complexity of the block. In particular, for the easier block, optimisation of shape, size and land value of parcels are on average far from the absolute optimum by 13.8%, 17.6% and 11.3%, respectively, whilst for the more difficult block the results are 19.3%, 35.5% and 37.8%, respectively.

The above outcomes are encouraging taking into account the complexity of the problem although undoubtedly there is a need for improving the performance of the algorithm for reaching optimum solution(s) for both single and multi-objective land partitioning, especially for the latter case. The weaknesses of the algorithm stem from the fact that optimisation treats the problem through a generic mechanism for space partitioning (Thiessen polygons). Hence the genotype of the algorithm involves only the two input parameters (the X and Y coordinates of the centroid of each parcel) in the optimisation process, on which the other parameters of the problem i.e. shape, size and land value are defined. Therefore, the improvement of the performance of the algorithm can be achieved either by developing a new generic space partitioning algorithm or by introducing a so called guidance (or learning or local optimiser) within the Thiessen polygons process. In the former case, the algorithm will take as input parameters the geometric features of shapes through the PSI and the size/land value of parcels that will then be optimized through *LandParcelS.* In the latter case, size and land value will be considered as constraints and the guidance operator will try to satisfy them during both the initialization and optimisation process.

Another limitation of the algorithm is that the computational time is quite long for both single and multi-objective land partitioning compared to what a planner would expect from such a sophisticated planning system. Two potential solutions to decrease the computation times are: firstly, parallel computing that permits the simultaneous processing of various functionalities of the algorithm and secondly the use of a more powerful/efficient programming language.

Further to the evaluation of the results, several interesting findings have been draw on by investigating the behaviour of the algorithm in various changes to its parameters. In particular, the elitist operator is necessary to significantly speed up the process since a test carried out showed a decrease in computational time by 32.3% and 67.3% for both examples in single optimisation. Although the algorithm may converge without the introduction of a mutation operator, in the one example (land block B25), it speeded up the process by 58% while for the other example (land block B14) it did not have any significant positive or negative effect. This suggests that the mutation operator may not always be necessary for all spatial problems although its use may benefit the performance. Similarly, block based mutation may benefit the performance of the algorithm compared with the parcel based mutation or they may both have no considerable influence in the process. In addition, the introduction of a penalty function definitely extends computational time but on the other hand it is necessary sometimes to steer solutions in the feasible region. For instance, the example for land block B14 could not converge without the penalty function. Moreover, it seems that a population size of around 40 fits for several spatial problems including land partitioning.

This paper proved that the integration of genetic algorithms with GIS may satisfactorily solve the land partitioning problem although further efforts are needed to improve the algorithm. This research also contributes to the broader field of spatial planning, especially for those disciplines that focus on space partitioning.

References

Bacao, F., Lobo, V., Painho, M., 2005. Applying genetic algorithms to zone design. *Soft computing*, 9, 341-348.

Bennett, D., Xiao, N., Armstrong, M., 2004. Exploring the geographic consequences of public policies using evolutionary algorithms*. Annals of the Association of American Geographers,* 94(4), 827-847.

Brooks, C., 1997. A parameterized region-growing programme for site allocation on raster suitability maps. *International Journal of Geographical Information Science*, 11(4), 375-396.

Brooks, C., 2001. A genetic algorithm to design optimal patch configurations using raster data structures. In: Krzanowski, R. and Raper, J. (eds.), *Spatial Evolutionary Modelling*. Oxford University Press. New York, 142-157.

Buis, A., and Vingerhoeds, R., 1996. Knowledge-based systems in the design of a new parcelling. *Knowledge-based Systems,* 9, 307-317.

Chrisman, N., 1999. *Exploring Geographic Information Systems*. John Wiley & Sons, Inc. New York.

Datta, D., Deb, K., Fonseca, C., Lobo, F., Condado, P., 2006. Multi-objective evolutionary algorithm for land use management problem. KanGAL, Department of Mechanical Engineering, Report Number 2006005.

Deb, K., 2001. *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley, West Sussex, 1-46, 81-163, 447-478

Deb, K., 2008. Optimisation. Lecture notes. Helsinki School of Economics, Finland.

Deb, K., 2009. Evolutionary computing methodologies for single and multi-objective optimization. Lecture notes. University of Jyvaskyla, Finland.

Delahaye, D., 2001. Airspace sectoring by evolutionary computation. In: Krzanowski and Raper (eds.), 2001. *Spatial Evolutionary Modelling.* Oxford University Press. New York, 203-233.

Demetriou, D., Stillwell, J., See, L., 2010. *LandSpaceS*: A design module for land consolidation: Method and application, *Working Paper 10/07*, School of Geography, University of Leeds, Leeds.

Demetriou, D., Stillwell, J., See, L., 2011a.A multi-attribute decision-making module for the evaluation of alternative land consolidation plans, *Working Paper 11/02*, School of Geography, University of Leeds, Leeds.

Demetriou, D., Stillwell, J., See, L., 2011b. The development and evaluation of a new model for measuring land fragmentation, *Working Paper 11/05*, School of Geography, University of Leeds, Leeds.

Demetriou, D., Stillwell, J., See, L., 2012. Land consolidation in Cyprus why is an Integrated Planning and Decision Support System required? *Land Use Policy.* 29, 131-142.

Dong, P., 2008. Generating and updating multiplicatively weighted Voronoi diagrams for point, line and polygon features in GIS. *Computers & Geosciences*, 34(4), 411-421.

Eshelman, L., Schaffer, D., 1993. Real coded genetic algorithms and interval schemata. In *Foundations of Genetic Algorithms II*, 187-202.

Gea, H., and Jain, S., 1998. Two-dimensional packing problems using genetic algorithms. *Engineering with Computers*, 14, 206-213.

Goldberg, D., 1989. *Genetic Algorithms: in Search, Optimisation and Machine Learning.* Addison Wesley, Crawfordsville.

Goldberg, D., Deb, D., 1991. A comparison of selection schemes used in genetic algorithms. In: *Foundations of Genetic Algorithms*, I, 69-93.

Gong, Y., Li, G., Tian, Y., lin, Y., Liou, Y., 2011. A vector-based algorithm to generate and update multiplicatively weighted Voronoi diagrams for points, polyline and polygons. *Computers and Geosciences,* doi:101016/j.cageo.2011.09.003

Gwiazda, T., 2006. *Genetic Algorithms Reference*. *Volume I: Crossover for Single Objective Numerical Optimisation Problems.* Tomaszgiazda, Poland.

Gwiazda, T., 2007. *Genetic Algorithms Reference*. *Volume II: Mutation Operator for Numerical Optimisation Problems.* Tomaszgiazda, Poland.

Hamda, H., and Schoenauer, M., 2002.Topological optimum design with evolutionary algorithms. *Journal of Convex Analysis*, 9(2), 503-517.

Jain, S., Gea, H., 1998. Two-dimensional packing problems using genetic algorithms. *Engineering with Computers,* 14, 206-213.

Jha, M., McCall, P., Schonfeld, P., 2001. Using GIS, Genetic algorithms and visualization in highway development. *Computer-aided Civil and Infrastructure Engineering,* 16(6), 399-414.

Keeney, R., Raiffa, H., 1993. *Decision with Multiple Objectives: Preferences and Value Trade-offs*. Cambridge University Press, 592pp.

Krzanowski, R., 1997. Evaluation of spatial evolutionary algorithm for spatial modeling. PhD thesis, University of London, London.

Krzanowski, R., and Raper, J., 2001. *Spatial Evolutionary Modelling*. Oxford University Press. New York, 1-63.

Li, X., Yeh, G., 2005. Integration of genetic algorithms and GIS for optimal location search. *International Journal of Geographical Information Science*, 19 (5), 581-601.

Malczewski, J., 1999. *GIS and Multicriteria Decision Analysis*. John Wiley & Sons, INC, New York, 392pp.

Malczewski, J., 2006. GIS-based multicriteria decision analysis: a survey of the literature. *International Journal of Geographical information Science*, 20(7), 703-726.

Matthews, K., Craw, S., MacKenzie, I., Elder, S., Sibbald, A., 1999. Applying genetic algorithms to land use planning. Proceedings of the 18th Workshop of the UK planning and Scheduling Special Interest Group, University of Salford, UK.

Matthews, K., Sibbald, A., Craw, S., 1999. Implementation of a spatial decision support system for rural land use planning : integrating geographic information system and environmental models with search and optimisation algorithms. *Computers and Electronics in Agriculture*, 23, 9-26.

Negnevitsky, M., 2002. *Artificial Intelligence: A Guide to Intelligent Systems*. Addison Wesley, Second edition, Essex, 415pp.

Openshaw, S., and Openshaw, C., 1997. *Artificial Intelligence in Geography*. John Wiley & Sons Ltd., West Sussex, 201-234.

Padhy, N., 2005. *Artificial Intelligence and Intelligent Systems*. Oxford University Press, New Delhi, 614 pp.

Renner, G., and Ekart, A., 2003. Genetic algorithms in computer-aided design. *Computer Aided Design,* 35(8), 709-726.

Rudolph, G., 1996. Convergence of evolutionary algorithms in general search spaces. Proceedings of the third IEEE conference on evolutionary computation, 50-54.

Sharifi, A., Boerboom, L., Zucca, A., 2007. Spatial Decision Support Systems. Lecture Notes. ITC, International Institute for Geo-Information Science and Earth Observation, The Netherlands.

Sharifi, A., Herwijnen, M., Toorn, W., 2004. *Spatial Decision Support Systems*. ITC, International Institute for Geo-Information Science and Earth Observation, The Netherlands.

Stewart, T., Janssen, R., Herwijnen, M., 2004. A genetic algorithm approach to multi-objective land use planning. *Computers and Operations Research,* 31, 2293-2313.

Tourino, J., Boullon, M., Gonzalez, X., 2003. A GIS-embedded system to support land consolidation plans in Galicia. *International Journal of Geographical Information Science*, 17(4), 377-396.

Van Dijk, S., Thierens, D., Berg, M., 2001. Design genetic algorithms to solve GIS problems. In: Krzanowski, R. and Raper, J. (eds.), *Spatial Evolutionary Modelling.* Oxford University Press. New York, 158-179.

Van Dijk, S., Thierens, D., Berg, M., 2002. Using genetic algorithms for solving hard problems in GIS. *Geoinfomatica*, 6(4), 381-413.

Wyatt, R., 2006. Evolutionary computing for optimising a region's distribution of agricultural production. *Applied GIS*, 2(3), 21.1-21.22