THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學
Pao Yue-kong Library
包玉剛圖書館

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

IMPORTANT

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

Pao Yue-kong Library, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

http://www.lib.polyu.edu.hk

ZONE DESIGN USING GRAPH PARTITIONING ALGORITHM WITH MULTIPLE CRITERIA, A EDA-BASED ZONE DESIGN

CHAN, HO-CHIU ELTON

Ph.D

The Hong Kong Polytechnic University

2017

The Hong Kong Polytechnic University

Department of Land Surveying & Geo-Informatics

Zone Design using Graph Partitioning Algorithm with Multiple Criteria, a EDA-based Zone Design

CHAN, Ho-chiu Elton

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

July 2016

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

(Signed)

____

CHAN, Ho-chiu Elton (Name of student)

____

Abstract

Zone design is a powerful concept in Geographic Information Analysis. It allows data analysts to develop a model not only for displaying zones by selected attributes but more interestingly for discovering hidden pattern within the extent of the zones. In brief, given a zonal data comprises a number of contiguous basic zonal units within a study area, zone design is to aggregate the basic spatial units into given number of regions subject to constraints and objective functions, whereas the objective functions is a series of pattern sensor, each can measure the presence of particular characteristics of interest in the zonal data and can be changed whenever required. This allows analyst to transform the zonal data to regions so that effects of the objective functions and constraints can be visualized on the map.

Unfortunately, zone design is a NP-complete problem that polynomial time algorithm is not available and exhaustive search is also impractical to solve the problem. Alternatively, heuristic approach is commonly adopted to tackle the problem within reasonable time. Common heuristic approaches include Hill-climbing, Tabu search and Simulated Annealing. In recent years, new approach like agglomerative clustering, which forms regions by cutting a minimum spanning tree is also introduced.

To tackle the zone design problem, this study investigate the use of Estimation of Distribution Algorithms (EDAs), and explore how it can be used to cope with the problem. EDAs are generic evolutionary searching methods that make use of probabilistic models to learn the structure and statistical parameters of a problem domain. Since the models in EDA can handle variables with multiple dependencies, it is common when dealing with variables in spatial data. By adding spatial handling procedure to EDA, this study presents a new way to handle zone design problem and at the same time, overcomes previous limitations in common heuristic approaches.

To test the EDA based zone design method, a zonal data based on Large Street Block Groups covering an entire Wan Chai district of Hong Kong is developed. Two encoding methods are used to model zone design solution for searching purpose: group-number encoding, and regional centre encoding. Meanwhile, GA-based zone design is developed for comparison purpose.

The EDA-based zone design is compared with the GA-based zone design for finding equal population regions. On the other hand, it is compared with SKATER (regionalization based on removal of edges from a minimum spanning tree) for finding homogenous regions with minimum internal variation. Test results show that the EDA-based zone design manages to generate solutions with better quality in homogeneity and compactness.

In reality, zone design problem is naturally a multiple-objective optimization problem. The difficulty to solve this kind of problem is that the objective functions are always conflicting with each other. Therefore, finding a single solution that can fit all the objectives simultaneously is an impossible task. Further to the EDA-based zone design, a variant based on the multiple-objective evolutionary paradigm is developed. Different from its predecessor that can find single optimum only, this variant is aimed to find the Pareto Optimal Set of the problem disregarding any subjective preference. The Pareto Optimal Set allows decision makers to decide which solution to go for that demonstrates a pragmatic way to tackle the zone design problem in reality.

Acknowledgements

My first and greatest gratitude goes to my supervisor Associate Professor Lilian PUN. You have given me helpful advice and comments during various phases of the work, and more importantly, support in overcoming numerous obstacles I have been facing through my research.

Besides my advisor, I would like to thank the rest of my Board of Examiners: Prof. Poh Chin LAI, Prof. Bo HUANG and Prof. Wu CHEN, for their insightful comments and challenging questions which have led me to widen my research from various perspectives.

The last but not the least, I would like to thank my wife Berlina and our little daughter Carina for their patience, love, and support during the years and many sleepless nights researching the topics and writing this thesis, without whom it would not have been possible to complete.

Hong Kong, 2017

CHAN Ho-chiu, Elton

Table of Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1 – Introduction and Motivation

1.1. Introduction

Zone design is of significant value for performing geographical analysis. Typical applications include census redistricting, electoral districting, environmental and health studies as well as location-allocation analysis. It is common that results of geographical analysis are typically presented by zones. Why are zones so appealing for analysis and presentation? In my opinion, it is due to simplicity and high expressiveness of zones that can intuitively describe notions of space and environs. An interesting characteristic of zones is that it is not just 2-dimensional, rather it can be organized as multiple nested levels when they are combined successively to form bigger zones or regions until all the zones are combined into one unit. The idea is the same when we consider a digital image that is originally composed of million pixels at high resolution, and it can be generalized to different levels of lower resolutions, in which each level has less number of pixels but bigger pixel size. However, such nested structure is not useful unless the characteristics or attribution of each zone are taken into account when combining the zones. In this connection, such aggregating operations on the zones can be further exploited to reveal patterns in the data as new type of spatial analysis tool in the Geographic Information Science.

To understand zone, we need to know how it is built. In general practice, data providers normally define zonal boundary in reference to existing topographic features such as roads, rivers and coastlines. To make it useful, the zones are referred as basic zonal units serving as the smallest building blocks within a study area. Further, these basic zonal units can be combined with each other to form larger zones or regions. For administrative convenience, these basic zonal units may follow existing administrative boundaries making it easy to cross-referencing with other data that has already adopted the same boundaries. This will save time and computational efforts especially when performing spatial analysis. When the zonal data of different themes having different boundaries unaligned with each other, it will be a disaster for a data analyst, who needs to modify the data so that all the boundaries can be aligned in a common frame before he/she can perform any data analysis.

However, delineation only defines the spatial part. To make the zone usable, each unit should be attached with representative qualitative or quantitative observations within its extent. Because such observations would normally be collected as a point based data, it can be simply transferred to respective zone and this is why zone or zonal data is normally classified as aggregated data.

To conduct zone design, we need to define proper criteria so that basic zonal units can be aggregated forming larger size regions to reveal what we want to see. It is expected that different criteria will generate different regions meeting specific objectives. For instance, equal population is an important criterion when conducting electoral districting because it ensures no single district has more influence due to there are larger number of voters and this also asserts voters has equally weighted voice in the election of representatives; homogeneity of social-economic characteristics is crucial for identifying spatial pattern in the community. When zonal units carrying similar characteristics of interest can be grouped together, it forms homogeneous clusters for easy visualization and further analysis such as finding the correlation with other factors trying to explain the formation of cluster. In short, the significance of zone design is not to build larger zones, it is a tool to display the criteria on the map facilitating us to understand the study area for conducting further activities and analysis.

However, zone design is not a trivial task. Because it is a well-known combinatorial optimization problem and has been proven to be NP-complete meaning exact solution cannot be found in polynomial time but it can be transformed to another known NP problem. Obviously, exhaustive enumeration is infeasible. As a result, different methods have been trying to solve the problem and two mainstream approaches are identified from the literature. The first approach is to treat the zone design problem as a clustering problem, in which each zonal unit is considered as a data point and clustering method is applied to grouping the points for forming regions. Another approach is using heuristic. Various traditional heuristic techniques can be identified and these includes Hill-Climbing, Simulated Annealing, and Tabu search. Heuristic is a common approach to solving combinatorial optimization problem because it can find an acceptable solution within reasonable time that requires to evaluate a small portion of possible solutions only. However, Hill-Climbing, Simulated Annealing and even Tabu search are heuristics that could be struck at local optimum easily. Another much advanced heuristic is evolutionary algorithm, which is also classified as meta-heuristic because it is not problem specific and it can be adapted to various problems easily. Representative example of Evolutionary algorithms is Genetic Algorithm (GA), which is a population-based optimization method and it will return a population of solutions rather than a single solution in each run. GA finds solutions by efficiently recombining and mutating components in a solution. Such solutions are not identified by random but controlled by objective function meaning solutions with higher fitness quality will have better chance to be selected and evolved to yield new solutions over generations until stopping criteria are met. However, when dealing with spatial data, GA is hard to maintain the topological relationship among zonal units. In particular, spatial continuity of a viable solution can be broken easily by the crossover operator.

To account for the above-mentioned issues, I propose to use a probabilistic graphical model-based evolutionary algorithm or Estimation of Distribution Algorithm (EDA). To find a better method, this study will review the zone design problem, investigate the EDA, and develop an EDA-based zone design algorithm as well as evaluate it to understand how good it is when compared with GA. Furthermore, based on the concept of Pareto dominance developed in multiple-objective evolutionary algorithm, this study also develops a variant of the EDA-based zone design to demonstrate a pragmatic approach for handling optimization problem with multiple conflicting objectives.

1.2. Contributions of the thesis

This study investigates the use of probabilistic graphical based evolutionary algorithm or EDA to solve the zone design problem. To our knowledge, this is the first-time EDA is applied to zone design problem that can generate regions by aggregating of basic spatial units subject to criteria and constraints within no human intervention. Although this study focuses on optimization of regions for homogeneity, the same idea can be applied to other criteria in the area of geographic information science, for example, criterion that maximizes the correlation between count of fire incidents and ratio of old buildings could reveal pattern of high risk regions. Besides, the proposed method is further enhanced to handle multiple objectives by finding the Pareto front.

1.3. Objectives

The objectives of this thesis are as follows:

i. Investigate and understand the use of EDA for zone design problem; and
ii. Find better solutions based on the probabilistic graphical network model and parameter learning technique.

1.4. Significance

Zone design is an crucial spatial data analysis tool, in particular for various applications such as electoral districting and spatial data analysis in the area of social science, image segmentation, school districting and police districting...etc. This study investigates how EDA can be used to cope with zone design problem. The proposed EDA based zone design method outperforms other approaches with higher quality measures. Inherited from the evolutionary algorithm, the proposed method can exploit existing identified solutions and explore new solution space for high-quality solutions at the same time. In fact, this study not only creates a new way to handle zone design problem, the way approach can be applied to any spatial optimization problem based on appropriate settings and selection of objective functions. Meanwhile, the proposed approach offers an automatic way to perform zone design that can avoid bias that is ubiquitous in manual process. More importantly, to my knowledge, this is the first trial to adopt EDA for solving multiple-objective zone design problem.

1.5. Approach

Approach of this study is as follows:

i. Review and define the zone design problem;
ii. Investigate and understand the design of EDA;
iii. Develop a EDA-based zone design algorithm;
iv. Identify and build test data;
v. Run the EDA-based zone design algorithm on the test data, compare and analyse results;
vi. Develop a variant of the algorithm and find the Pareto front of the problem;
vii. Give conclusions and recommendations.

1.6. Overview of the thesis

The remainder of this thesis is organized as follows:

- Chapter 2 provides literature review, understanding, and modelling issues of the zone design problem;
- Chapter 3 presents the concept of EDA;
- Chapter 4 provides the detailed description of each component and modelling issues of the EDA based zone design;
- Chapter 5 presents preparation of the test data, comparison of the EDA based zone design with other approaches and shows the approach to find Pareto optimal set; and
- Finally, in Chapter 6, reviews major contributions and concludes this study.

Chapter 2 – Zone Design Problem

2.1. Introduction

It is good to use zonal data or areal data as representation for performing efficient and effective spatial analysis of an area. It is notably true when each zone carries representative values indicating various notions such as distinctive geographical meaning, designated number of entities or dominated characteristics of interest. Furthermore, different colours can be assigned to the zone so as to reveal hidden pattern or to show extreme intensity areas.

Generally speaking, zonal data is naturally a result of generalization of point-based data. Therefore, position shifting and averaging the measurement of the original point-based data are expected. Fortunately, it can provide a better spatial context for easy understanding of the original data extent. This not only provides better visualization but also allows to build topological relationship among zones. These open many possibilities on spatial analysis. Think about the notion of zones, I can further identify three appealing characteristics making it excellent for spatial modelling and representation. First, it reduces data size and preserves confidentiality of the original data, which is a typical arrangement when compiling a large amount of point-based data such as census and health related data. Preservation of confidentiality is controversial because in some situations, analysts may want to understand the details by examining the original data but is impossible due to the generalization. This proves that zonal data is effective in preserving confidentiality. Second, zonal data minimizes data variation but preserves patterns and it can be manipulated to reveal hidden patterns by specifying suitable criteria and constraints. After all, the ultimate goal is to make the pattern visible. Third, zones are delineation of the concerned geographical area which is not only easy to understand but also promote easy communication.

For data collection and administrative convenience, it is a common practice that authority adopts existing administrative boundary, alignment of physical geographic features such as roads, rivers and coastlines, as well as political considerations to define zonal data, in which each zone can be referred as a basic zonal unit. It is considered that basic zonal unit is the smallest contiguous unit that can aggregate to form a study area. When the basic zonal units are grouped by specified attributes, each group forms a region. Further, all such regions form an exclusive partitioning of the study area, which also refers to an instance of zoning system. Therefore, the basic zonal units can be considered as a set of building blocks that can delineate a study area into different zoning systems fulfilling various tasks such as data collection, spatial analysis, visualization as well as data integration with other datasets. Further, it can be used for time series analysis, in which analysis involves repeated observations of the same variables related to each zone can be compared over a designated period of time to identify trend and ultimately predict future events.

Besides, zoning systems at different levels of granularity can form a nested hierarchical structure. See Figure 2-1 for an illustration that the lowest level comprises the basic zonal units only while the highest level simply refers to the entire study area. In between these two levels, there could be tremendous middle levels, each comprises regions that is the result of repeated aggregation of current zonal level forming the next higher zonal level with decreasing number of regions. On the other hand, when the number of regions is fixed, regions can still be formed in tremendous ways. It is the tremendous number of possibilities to form regions make zone design so appealing to be treated as a spatial analysis tool. However, how to find the zoning system fitting to our needs?

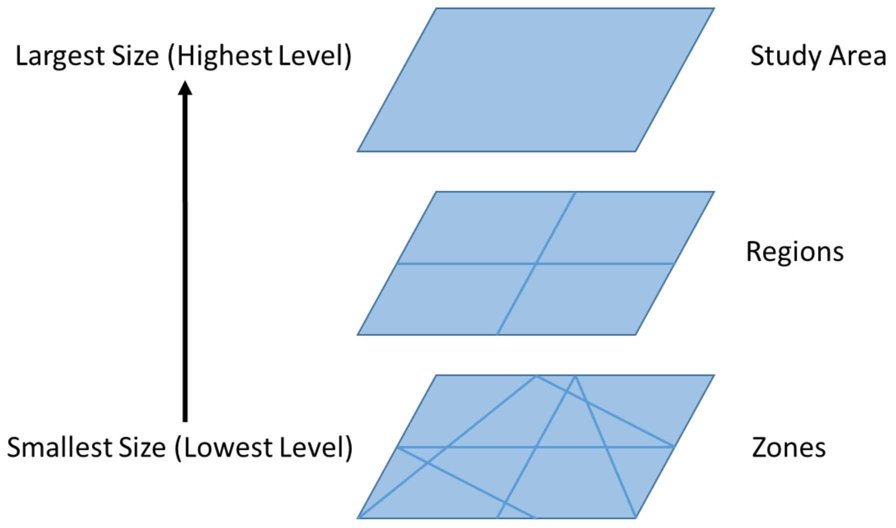


Figure 2-1 Hierarchy of Zones at different levels

In this chapter, I firstly give a review of the zone design and its applications, present the problem definition including its computation complexity and modelling issue when dealing with zonal data; then review existing common approaches and finally give a brief introduction to the use of EDA for solving the zone design problem.

2.2. A brief History

As proposed by Openshaw (1996) and Alvanides et al (2001), zone design can be regarded as a spatial analysis tool for spatial description, visualization, pattern capture, testing spatial hypotheses and spatial modelling. Zone design is also called regionalization problem (Openshaw 1977, Assunção et al. 2006, Guo, D. 2008, Li et al. 2013). In fact, there have been many research studies making use of the zone design for different analysis in various fields, for instance, spatial modelling (Openshaw 1976), health studies (Cockings and Martin 2005), electoral districting (Tasnádi, A. 2011; Bação et al. 2005), census districting (Openshaw and Rao 1995), spatial analysis (Guo and Wang 2011), and price analysis for public transport (Tavares-Pereira et al. 2007). Although these examples all refer to the same zone design problem, different approaches or methods have been proposed to solve it. This indicates that zonal data and zone design problem is of high practicality for solving real life problems. If regions can be intentionally optimized and shaped by maximizing or minimizing particular measures, it not only helps to answer "what if" type of questions but also allows data manipulation likes two sides of a coin.

The activity to carry out electoral districting by imposing artificial bias for election is called gerrymandering. This term was created in 1812 when Elbridge Gerry, the Governor of Massachusetts in United States attempted to manipulate the voting districts of his state, in which one of the districts was created with the shape of a salamander and it was the result of blending the words "Gerry" and "Salamander" (Mehrota et al. 1998; Tasnádi, A. 2011). Since this is an unfair act, researchers have been studying electoral districting for many years, including Bozkaya et al. (2003), Mehorota et al. (1998) and Hojati (1996) on how to perform zone design for electoral districting, while Vickrey (1961), Sherstyuk (1998), Altman (1997) have also given a very comprehensive analysis of gerrymandering.

2.3. Problem Definition

Given *n* basic and the smallest zonal units over a territory or study area, each zonal unit carries some quantities of interest such as population. The zone design problem is to aggregate the *n* basic zonal units into k regions where k < *n* subject to constraints and criteria. Constraints and criteria are two different concepts. Constraints serve as mandatory requirements or so-called hard constraints to which solution must conform; otherwise, the solution is considered invalid. If a solution is found violating the hard constraints, it will be classified as an invalid solution and the simplest handling approach is to discard it. Alternatively, one could try to repair the invalid solution. However, repairing a solution is a challenging task because there may be too many scenarios and once all scenarios are handled, the method may not be generic enough to apply to different problems. On the other hand, criteria are measures of concerned characteristics that can be either maximized or minimized subject to the optimization purpose. Such criteria is also called soft constraints that typically can be realized by defining proper objective functions. An objective function is primarily used for evaluation of each solution and as guidance of the searching algorithm to reach the optimal solution. By assigning a fitness value to each solution, this also facilitates comparison among different solutions.

2.4. Common Constraints and Criteria in Zone Design

Constraints in zone design, in general, focus on restrictions relating to the geometrical aspects of the solution regions. Common constraints for zone design (Grofman 1985, Haining et. al 2000, Bação et al. 2005, and Tavares et al 2004) are as follows:

i. Integrity – Each zonal unit can only be assigned to one and only one region at a time. This ensures that each zonal unit is exclusively assigned to one region preventing overlapping.

ii. Contiguity – The result regions must be contiguous without any hole or any isolated parts. Meanwhile, each zonal unit shall strictly conform to the same constraint.

iii. Number of regions – This is pre-defined in this study. However, it can be determined by cluster analysis method.

iv. Compatibility – This refers to the use of existing boundaries, e.g. administrative boundary or postal code boundary. This enables comparison between the resultant region and existing boundaries.

Furthermore, the followings are common criteria or objective functions for zone design.

i. Homogeneity (equality) – this aims at forming regions with respect to either maximizing or minimizing homogeneity of selected attribute of the basic zonal units. Homogeneity can be determined among zonal units or regions. For instance, population among regions in electoral districting would be minimized to achieve equal population and more importantly, this promotes unbiased election.

ii. Compactness – this aims at forming compact and circular-like regions, basically for aesthetic reasons.

Zone design problem is naturally subject to multiple and conflicting objectives. Different criteria can be introduced for shaping the regions. However, selection of objectives and how many objectives to be used are also highly dependent on the types of applications and the goal of the analysis.

Back to electoral districting, Altman (1997) proposed two additional criteria. First, "Creating fair electoral contests" aims at providing an impartial contest. Second, "Representational goals" or "respect for ethnic minorities" aims at protecting communities of interest and avoiding dilution of minority representation. Extensive discussion of constraints and criteria, in particular for electoral districting can be found in Grofman (1985), Lijphart (1989) Bozkaya et al. (2003), Grilli di Cortona et al. (1999), Kalcsics et al. (2005), and Ricca and Simeone (1997).

In this study, I consider applying homogeneity and compactness as criteria for performing zone design because these are the two basic and primitive conditions for constructing regions. The following sections define the objective functions for the two criteria. However, it would be easy to introduce additional constraints and criteria when required.

2.5. Selected Criteria

Generally speaking, homogeneity helps to identify homogenous patterns within the study area. Compactness tries to maintain the appearance of each region in a circular shape as much as possible.

2.6. Homogeneity

Homogeneity is one of the common objectives for various zone design related applications. It aims to achieve a state of identical values among regions so as to ensure sensible comparison. In general, there are two types of homogeneity: inter-regional homogeneity and intra-regional homogeneity. These two criteria can be identified in most of the research studies for solving the zone design problem. Because they are conflicting each other and thus will not be used at the same time. Regarding inter-regional homogeneity, the aim is to find a configuration that can minimize the difference in values among regions. In other words, this objective maximizes the similarity at the regional level. The reason to do that is to reduce the bias for pattern identification (Boscoe and Pickle 2003). In particular, this can also avoid the small numbers problem, which is a common concern in public health mapping that can cause unstable rates (Morris and Munasinghe 1993). The challenging part of finding regions of homogeneity is that regions are imaginary and they must be built by the basic zonal units. Therefore, it is necessary to ask two questions to find the configuration. The first question is which zonal units should be grouped together forming a configuration of regions. The second question is whether the configuration can achieve minimum difference among regions.

To define homogeneity mathematically, assuming the number of regions and the number of zones are denoted as *k* and *n* respectively are given, the sum of attribute values for each region should be as close as possible to the overall sum of attribute values divided by the number of regions or average sum of attribute values by region. As such, we can formulate the following objective function to achieve inter-regional homogeneity:

$$\text{Minimize} \sum_{i=1}^{k} \sum_{j=1}^{n} (A_{ij} - \overline{A})^2 \qquad (1)$$

In (1), A_{ij} denotes the attribute value of interest of zone j when it is assigned to region i. $\overline{A}$ denotes the overall average of the attribute values of interest among regions and it can be computed by (2).

$$\overline{A} = \frac{\sum_{j=1}^{n}(A_j)}{k} \qquad (2)$$

Further, the sum of square of the difference of attribute value carried by each region and the overall average is to ensure that the objective function is always non-negative. This objective function should be minimized, i.e. the smaller the better, so that all regions will carry almost the same attribute value of interest approaching to the overall average in order to realize homogeneity or more precisely equality.

On the other hand, intra-regional homogeneity refers to similarity of zonal units within each region. Such internal homogenous region is commonly called cluster, or sometimes refer to regionalization, which is useful for finding meaningful boundary representing minimum variation of concerned measures. Such measures could be extreme of variability of particular values exposing where they are using the zonal units. Given a set of zonal units, each zonal unit carries attribute values of interest, the objective function to seek for intra-regional homogeneity can be defined as follows:

$$\text{Minimize} \sum_{i=1}^{k} \sum_{j=1}^{n} (A_{ij} - \overline{A}_i)^2 \qquad (3)$$

To identify clusters using this objective function, all the zonal units should carry similar quantities or qualities of interest in each cluster. Therefore, $\overline{A}_i$ in (3) denotes the average attribute value of interest within region i only. In other words, this creates regions comprising of zonal units having the least variation of the attribute values. Obviously, inter-regional homogeneity and intra-regional homogeneity are two conflicting objectives and should be used separately in different applications.

There are other forms of objective function for homogeneity. As discussed in the previous section, Hess et al. (1965) proposed a location-allocation model. It is merely a demand and supply model, in which, it defines an objective function based on the ratio of demand and supply. However, one may need to aware that ratio may become unstable when the base size (e.g. population) is too small due to the so-called small-number problem (Goodchild and Gopal 1989). In fact, defining homogeneity can be very flexible but highly dependent on the underlying applications.

2.7. Compactness

Compactness enforces the zone design process forming regions with a circular shape. Li et al. (2013) analysed and pointed out three reasons explaining the importance of compactness measures in GISicence. First, a compact shape is likely to be homogeneous sharing common properties and has maximum accessibility. Second, representation through shape is essential to analyse and understand geographical phenomena. Third, compactness is an intrinsic property of zone, which is a common and natural representation for spatial objects. These explain why compactness has a long history being one of the common criteria for zone design.

Measure of compactness can be realized by many methods. Young (1998) and Maceachren, A.M. (1985) provided a detailed review of compactness measure, while Bação et al. (2005) applied compactness in zone design. In this study, the most direct method is adopted such that compactness is determined by the Euclidean distance between the centre of each zone and the centre of respective region, which can be formulated by (4), in which d_{ij} denotes the Euclidean distance and it is typically called Radial Compactness (Bação et al. 2005).

$$\sum_{i}^{k}\sum_{j}^{n}(d_{ij}) \tag{4}$$

There are other forms of compactness and the most classical one is to define compactness by a ratio shown in (5) (Bribiesca and Montero, 2009) where pr_i and a_i denote perimeter and area of region i respectively.

$$\sum_{i} \frac{\mathrm{pr}_i^2}{\mathrm{a}_i} \tag{5}$$

Bação et al. (2005) applied (5) to measure the compactness for a zone design solution and their results showed that compactness cannot be improved with increasing homogeneity. This indicates that they are conflicting with each other. Furthermore, compactness can also be determined by the ratio of the area of an inscribed circle to the area of the region as shown in (6). When the ratio almost equals to 1, the respective region is of high compactness and has a circle-like shape. Furthermore, Maceachren (1985) had provided a detailed analysis of several compactness measures including (6) and had found that it is sensitive to indentation and extrusion.

$$\frac{\text{Area of inscribed circle}}{\text{Area of a region}} \tag{6}$$

Compactness is highly dependent on the distribution of underlying data. If the size of the underlying zonal units is irregular, i.e. some are extremely large and others are very small and the distribution of attribute values is irregular at the same time, it will be very difficult to find regions with high degree of compactness. The reason is due to the uneven size of the basic zonal units, especially when some are extremely big when compared with others, will reduce the flexibility and make it more difficult to combine regions fitting to multiple criteria.

2.8. Computational Complexity

Understand the computational complexity of the zone design problem helps us to evaluate the difficulty and find out the right tool for solving the problem. In general, complexity of a computer algorithm is denoted by polynomial time, which is a term used in area of computational complexity theory in the field of computer science. The theory is devoted to analyse the difficulty of solving specified discrete problems using computers (Altman 1997). For in-depth detail and understanding of the theory, one can see Arora and Barak (2009).

In brief, computational complexity refers to how hard a problem can be solved by a computer algorithm. A problem is computable if a computer algorithm is available and it can find an exact solution (instead of approximate solution). To answer how hard a problem is, it can be measured by the number of instructions that the algorithm must be reached to provide an exact solution. And it can be represented by a time-complexity function (or time-complexity) with the problem size as input and number of instructions of the algorithm as output. Regarding zone design problem, problem size refers to the number of input zonal units and the number of expected regions. Time-complexity is independent of hardware or software in implementation, i.e. implementation independent.

When the time-complexity is a polynomial, corresponding algorithm is said to take polynomial time to solve the problem. If time-complexity is an exponential function, corresponding algorithm requires exponential time to solve the problem. Furthermore, a problem is computationally tractable if there exists an algorithm that can solve the problem within polynomial time. Otherwise, even there is an instance of the problem that cannot be solved within polynomial time, the problem is computationally intractable.

If algorithm does not exist for solving a problem within polynomial time, the problem is classified as Non-Deterministic Polynomial-time (NP). A problem is defined as NP-complete if it can be transformed to other known NP problems. In fact, computer researchers has identified a list of NP problem and provided proofs for the problem if it is intractable.

Regarding the zone design problem, because algorithm does not exist to solve it within polynomial time but it can be transformed to other known NP problems such as Weighted Set Partition, 3-Partition, Weak Partition, Integer Programming, Cut into Connected Components of Bounded Size and so on, it is a NP-complete problem Altman (1997). Although polynomial time algorithm is not available, computational complexity of a zone design problem can be estimated by Stirling Number of the Second Kind (Sharp 1968), denoted by $S(n0, r)$ as stated in (7), which counts the number of ways to combine *n* zones into r non-empty regions.

$$S(n, r) = \frac{1}{r!}\sum_{i=0}^{r}(-1)^{i}\binom{r}{i}(r-i)^{n} \qquad (7)$$

However, the Stirling Number of the Second Kind does not take any constraints and criteria into account, it just roughly indicates the number of ways for forming regions. Table 2-1 Number of possible solutions estimated by Equation 7 for different configuration of zones and regions

gives an idea of search space size and its growth rate with increasing number of zones and the number is tremendous.

It is obvious that the number of possible ways for the formation of regions increases exponentially with the number of zonal units. As a result, this clearly indicates that searching for optimal solution of a zone design problem by enumeration is infeasible even for small size problem.

No. of zonal units	No. of regions to be created	Number of possible ways
10	5	42,525
25	5	2,436,684,974,110,751
30	5	7,713,000,216,608,565,075
50	10	$26x10^{42}$

Table 2-1 Number of possible solutions estimated by Equation 7 for different configuration of zones and regions

2.9. Modelling Considerations

There are two modelling considerations when dealing with zonal data. First, a compact but flexible data structure is necessary to model the zonal data so that it can be configured to different solutions effectively. Second, spatial contiguity should be well-maintained so that regions formed by aggregration of the basic spatial units are all contiguous and without hole.

Regarding the first consideration, it is common to adopt graph-based representation for partitioning problem (Fjällström 1998, Kim et al. 2011) and the same has been adopted in many research studies in relation to zone design or regionalization (Assunção et al. 2006, Tavares-Pereira et al. 2007, Guo, D, 2008 and Guo and Wang, 2011). Given a set of zonal units covering a territory of interest, it can be modelled as a connected graph, in which each centre of the basic zonal unit denotes as a node while its adjacency relationship between two contiguous basic zonal units is represented by an edge connecting their respective centres. This approach is so appealing because the graph model can properly preserve the topological structure of the zonal units. Meanwhile, constraints such as integrity and contiguity can be maintained automatically without much difficulty. To ensure the model integrity, a proper method for encoding the structure is extremely crucial. Various methods are available to model the structure and the most popular method involves using an adjacency matrix and a node list. An adjacency matrix is usually preferable because it is more convenient for computation and manipulation.

Formally, a connected graph can be defined as $G = (V, E)$, where $V = \{1, 2, 3, \dots n\}$ represents a set of vertices, and $E = \{e_{ij}, where\ i, j = 1,2,3, \dots n;\ for\ i \neq j\}$ represents a set of edges with e_{ij} indicating the adjacency relationship between zonal units i and j. Each node can be associated with k characteristics represented by a vector $c_i = (c_{i1}, c_{i2}, \dots, c_{ik})$. Therefore, the zone design problem can be transformed to a partitioning problem of the graph G and the solution is to find a set $\{r_1, .. r_k\}$ that denotes k subsets of V and r_k is considered as the k partition of G and therefore, a zoning system can be denoted as $R = \{ r_1, r_2 \dots r_k\}$;

$$\forall i \in \{1, \dots, k\}, r_k \neq 0; \tag{8}$$

$$\forall i, j \in \{1, \dots, k\}, i \neq j,\ r_i \cap r_k = 0;\ and \tag{9}$$

$$\bigcup_{i=1}^{k} r_i = V \tag{10}$$

Based on the adjacency relationship between two different zones, we can add other useful information to the model. For instance, uncertainty of region assignment at particular vertex can be attributed by region assignment of other connecting vertices.

Regarding the second consideration, Queen and Rook contiguity are two common methods to determine the adjacency relationship or neighbours of a given zonal unit. For Queen Contiguity, adjacent relationship is established whenever two zonal units share a single point. For Rook Contiguity, the adjacent relationship is slightly stricter that two zonal units are adjacent only if they share at least two points. Figure 2-2 illustrates the scenarios of Queen and Rook Contiguity, in which there are 9 regular blocks.

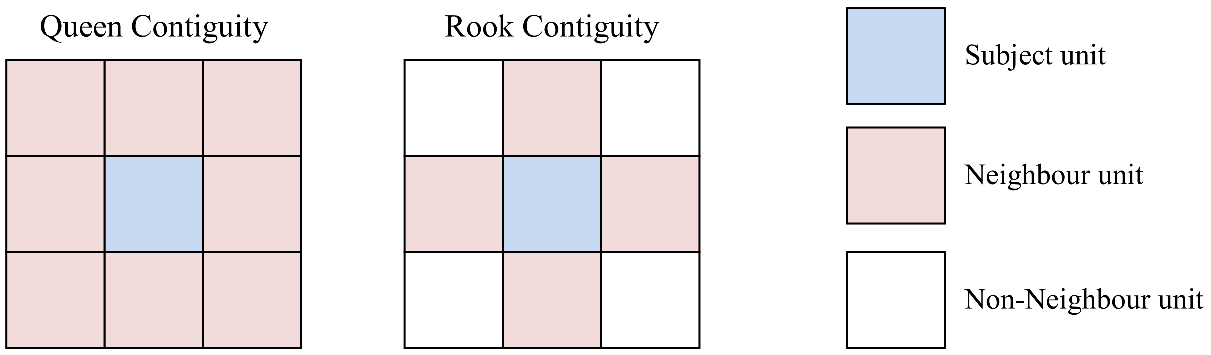


Figure 2-2 Queen and Rook Contiguity (Lloyd 2010)

According to Queen Contiguity, the block at the centre has neighbours in 9 directions. But in rook contiguity, the block has 4 neighbours only. Obviously, adjacency relationship of Queen Contiguity will be more than that of Rook Contiguity. This means more computing power is required to handle Queen Contiguity. On the other hand, rook contiguity requires less number of neighbouring relations and thus improves computational efficiency.

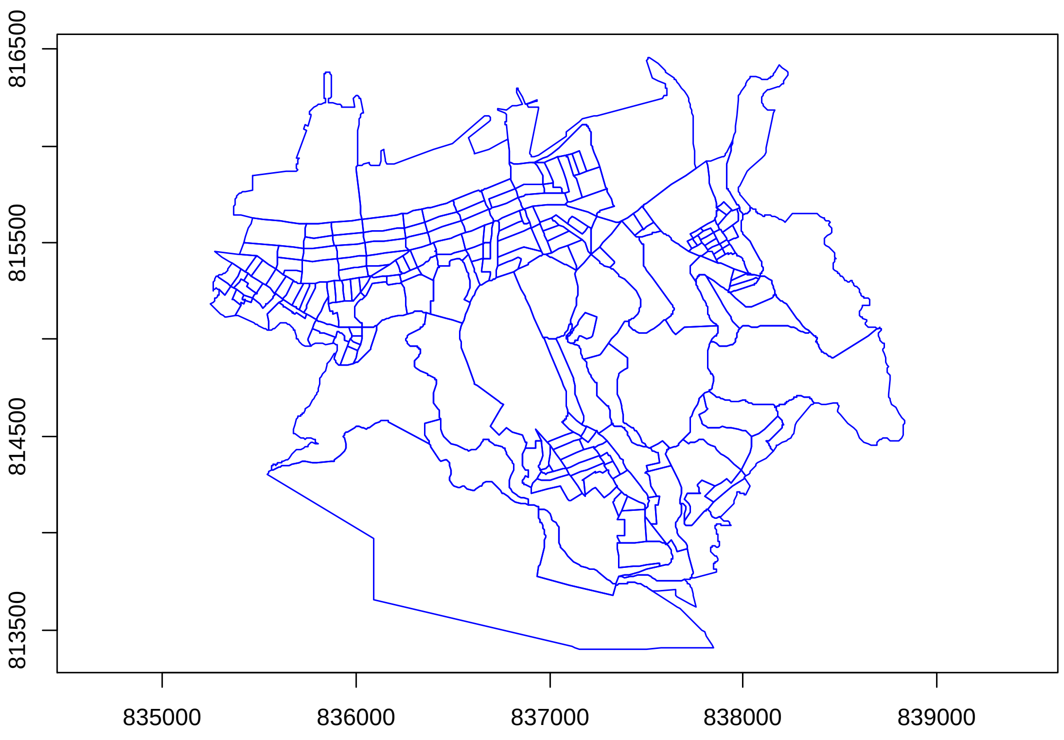


Figure 2-3 Street blocks of Wan Chai district, Hong Kong

Further, contiguity rule will affect formation of regions. Because Queen Contiguity allows regions forming from polygon with parts connecting by point. It will create region that violates the rule that any point shall reach another point within the same region without crossing another region. To further examine the two cases, two series of regions are generated with Queen Contiguity and rook contiguity respectively. It is shown that there is subtle difference in quality between the two contiguity settings. As expected, when the number of regions increases, solution quality with Queen Contiguity is slightly better than that with rook contiguity because the former gives relatively more flexibility to form regions attaining higher fitness values, however, it generates invalid region. As a result, it is considered that rook contiguity is much reasonable and natural for defining neighbourhood in zone design and rook contiguity is adopted in this study.

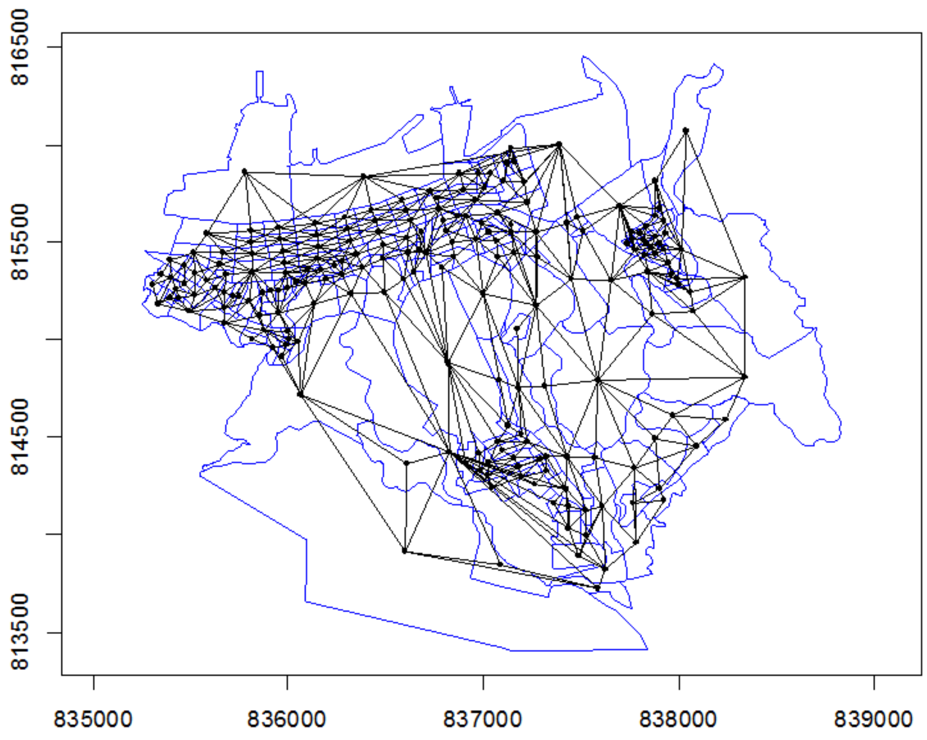


Figure 2-4 Zonal units and its graph model using Queen Contiguity

To illustrate how to convert the zonal units to graph model, a dataset is extracted from the Street Blocks available from the Planning Department of the Hong Kong SAR Government. The dataset shown in Figure 2.3 is Wan Chai District of Hong Kong. Graph models in Queen Contiguity and Rook Contiguity are shown in Figure 2.4 and Figure 2.5 respectively. In the figures, more adjacency relationships are created within the Wan Chai district using the Queen Contiguity method. Besides, the connecting edges created in Queen Contiguity will cross one another which will not happen in Rook Contiguity. Furthermore, other neighbouring definitions can be applied such as

Graph-based neighbours by triangulation, distance-based neighbours by choosing k nearest neighbour and higher-order neighbours by distance and order of contiguities (Roger et al. 2008).

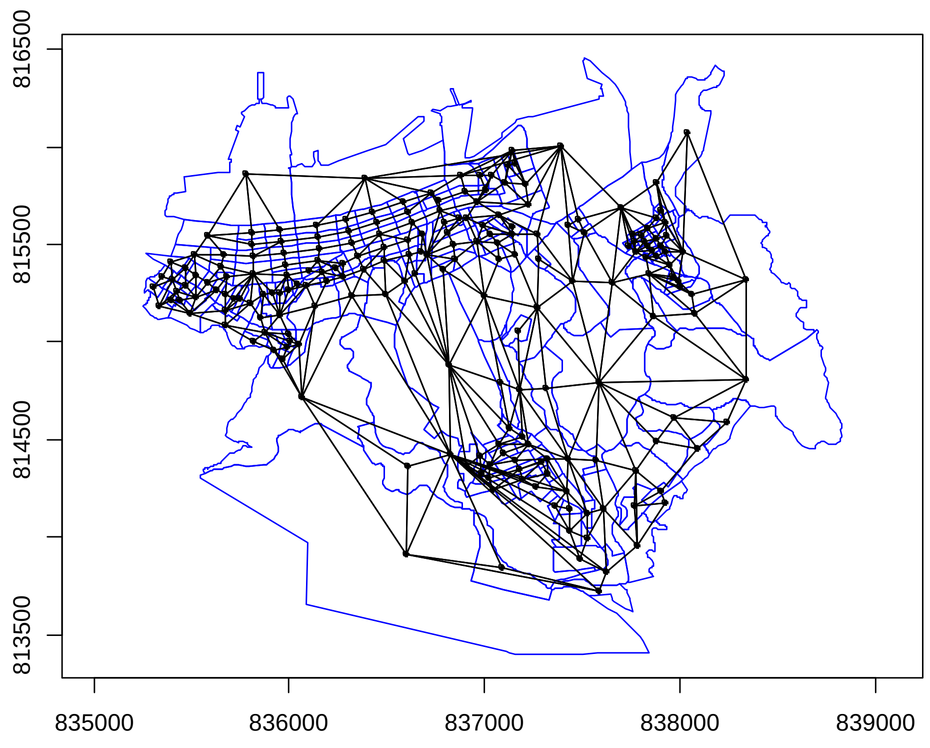


Figure 2-5 Zonal units and its graph model using Rook Contiguity

2.10.Approaches to Zone Design

As discussed in previous sections, the computation complexity of zone design problem has been proven to be NP-Complete. In other words, the search space is typically extremely large and infeasible to be solved within polynomial time. Enumerative approach will not be recommended to solve the problem. Heuristics, which can provide acceptable solution within reasonable time, has become a mainstream approach to the problem. The only drawback of this approach is that it can find a sub-optimal solution only. One of the first heuristic approaches for zone design is called Automated Zoning Procedure (AZP) developed by Openshaw (1977). The idea of AZP is very simple that it reassigns zonal units along the regional boundary iteratively to neighbouring regions hoping for the improvement of solution quality until particular stopping criterion is met. Zone design is a combinatorial optimization problem that has an extremely large search space, AZP can reach a very small portion only. Therefore, it cannot reach different areas of the search space and may easily be trapped in local optima. Besides, the degree of improvement is also highly relying on the initial configuration. Ricca et al. (2011) has reviewed different approaches in solving political redistricting problems. Among all the approaches, clustering such as the hierarchical clustering and partitional clustering, as well as evolutionary algorithm (meta-heuristics) are the mainstream for handling the problem.

In the following sections, both the clustering technique and the Genetic Algorithm (GA), as well as relevant past works in relation to the zone design problem will be reviewed. By understanding the shortcomings of these approaches, the use of an advanced evolutionary algorithm called Estimation of Distribution Algorithm will be introduced, in which it will be the foundation in the proposed solution in solving the zone design problem.

2.11.Clustering

Clustering is a common data analysis method by mean of grouping or classifying objects into clusters so that similarity of objects within a cluster is higher than that of other clusters. In general, number of clusters should be given before conducting clustering analysis and it can be defined by user requirement or determined by empirical results. To identify the optimal result, evaluation criteria to determine quality of cluster are required. According to Jain and Dubes (1988), clustering can be divided into hierarchical clustering and partitional clustering, in which both require a proximity matrix to define pairwise similarity between objects being considered.

2.12.Hierarchical Clustering

Hierarchical clustering creates clusters by nested sequence of clustering operations. Although our focus is the clustering result, the nested structure created in the process is also useful for some applications, especially in the fields of social science and biology that have interested on taxonomy. The nested sequence could be either “agglomerative” or “divisive”. The agglomerative approach starts with each object as a single cluster and gradually merges objects based on proximity measures until all objects are assigned with one and only one cluster. On the other hand, divisive approach starts with all objects in a single cluster and sub-dividing it sequentially into smaller clusters until expected the number of clusters is reached. Instead of creating a single cluster, hierarchical clustering (i.e. either agglomerative or divisive.) not only creates nested clusters but also show how objects are merged or split at different levels. Hence, a partition can be identified by choosing a cluster level. Given a set of data points, basic algorithm of hierarchical clustering is shown in Algorithm 2-1.

To compute and update a proximity matrix, a well-defined definition of distance between two clusters is necessary. Euclidean distance is a typical distance being used in reality. Other distance measures such as Manhattan distance (i.e. sum of absolute difference of two points), and “sup” distance (i.e. maximum of absolute difference of two points) could be used. Here the proximity matrix is not restricted to geographical distance and it can be defined as the difference of particular attribute values between the clusters. In addition, a method to determine the merging or dividing actions of two clusters is also required. There are several methods can be identified in the literature (Jain and Dubes 1988) and these include single link, complete link, average link and Ward’s method. No matter which method is used, two clusters will be merged if they have the least dissimilarity. Therefore, the method to determine the dissimilarity will greatly affect the clustering action.

```
1. Compute a proximity matrix;
2. Assume each data point be an individual cluster for agglomerative approach, or assume all
   data points are in a single cluster for divisive approach;
3. Merge or divide cluster according to the proximity matrix and selected clustering method;
4. Update the proximity matrix;
5. Repeat steps 3 and 4 until all data points are exhausted.
```

Algorithm 2-1 Hierarchical clustering algorithm (Jain and Dubes, 1988)

Consider single link, dissimilarity of two clusters is determined by two data points in two adjacent clusters having minimum dissimilarity. For complete link, on the contrary, it uses the farthest data points (i.e. maximum dissimilarity) in the opposite clusters to determine the dissimilarity. For average link, dissimilarity of two clusters is determined by average dissimilarity over all data points in the opposite clusters. And finally, Ward’s method is also called Sum of Square Error (SSE), and it determines the dissimilarity by the variation of SSE before and after merging of two clusters. In brief, different notions of dissimilarity have different limitations. For example, the single-link method uses the closest data points only and is suffered from a so-called chaining effect making the output clusters not compact enough and far apart. In the same sense, the complete-link method is suffered from crowding effect making the output clusters too compact. Furthermore, average-link method and the Ward's method are two relatively balanced methods. Details of the clustering methods can be found in many clustering related textbooks such as Jain and Dubes (1988). Due to the hierarchical nature of this clustering method, its results can be presented by dendrogram, which is a tree diagram for showing arrangement of clusters.

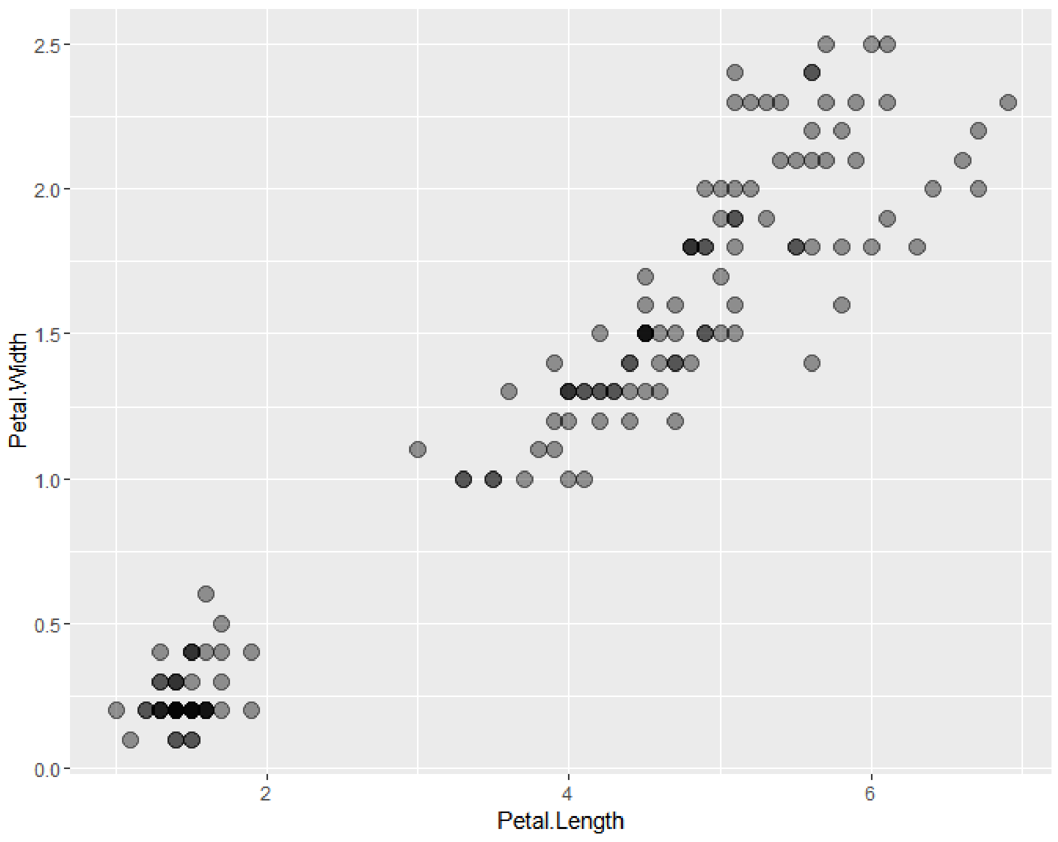


Figure 2-6 IRIS dataset (UCI 2006), overlapping points are represented by darker grey colour.

The well-known IRIS dataset in pattern recognition (UCI 2016) is used to demonstrate how hierarchical clustering works. In the IRIS dataset, there are 150 instances, which can be divided three classes. Each class contains 50 instances referring to a type of iris plant.

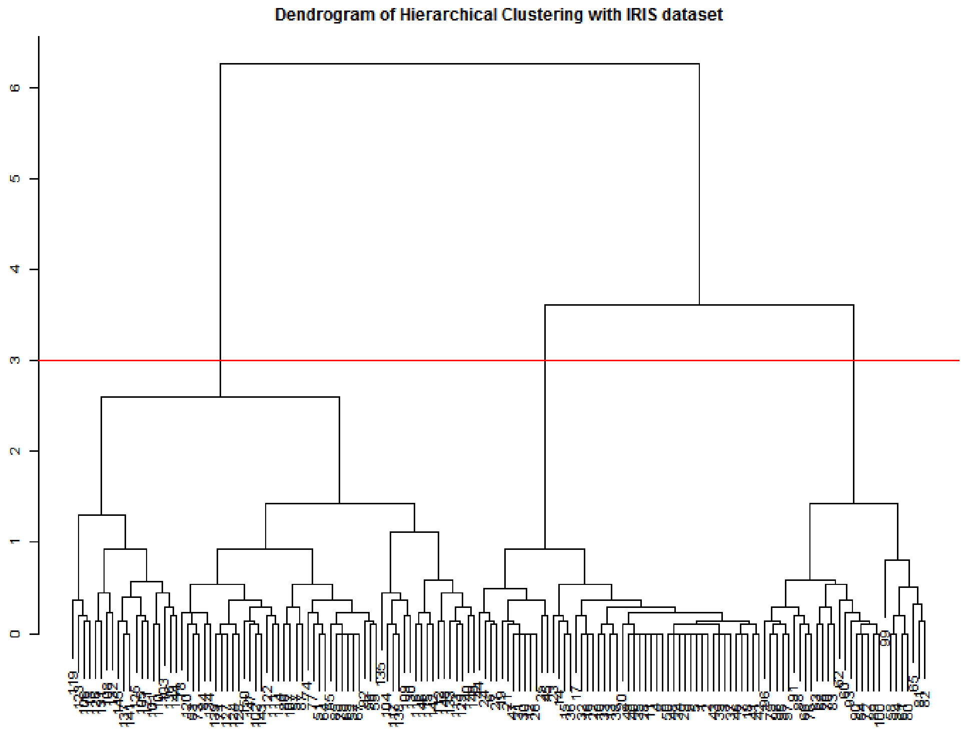


Figure 2-7 Dendrogram of IRIS dataset, hierarchical clustering with agglomerative approach and Complete Link applied. The red line indicates possible tree cut for identifying 3 clusters.

Figure 2-6 shows the original IRIS dataset. To apply hierarchical clustering with agglomerative approach, a distance matrix can be defined with the variables: Petal.Width and Petal.Length in the dataset and apply Complete Link for defining the similarity. Figure 2-7 shows the dendrogram of the dataset. A red line is added to cut the dendrogram into three parts (i.e. parts below the red line), each part representing an individual cluster.

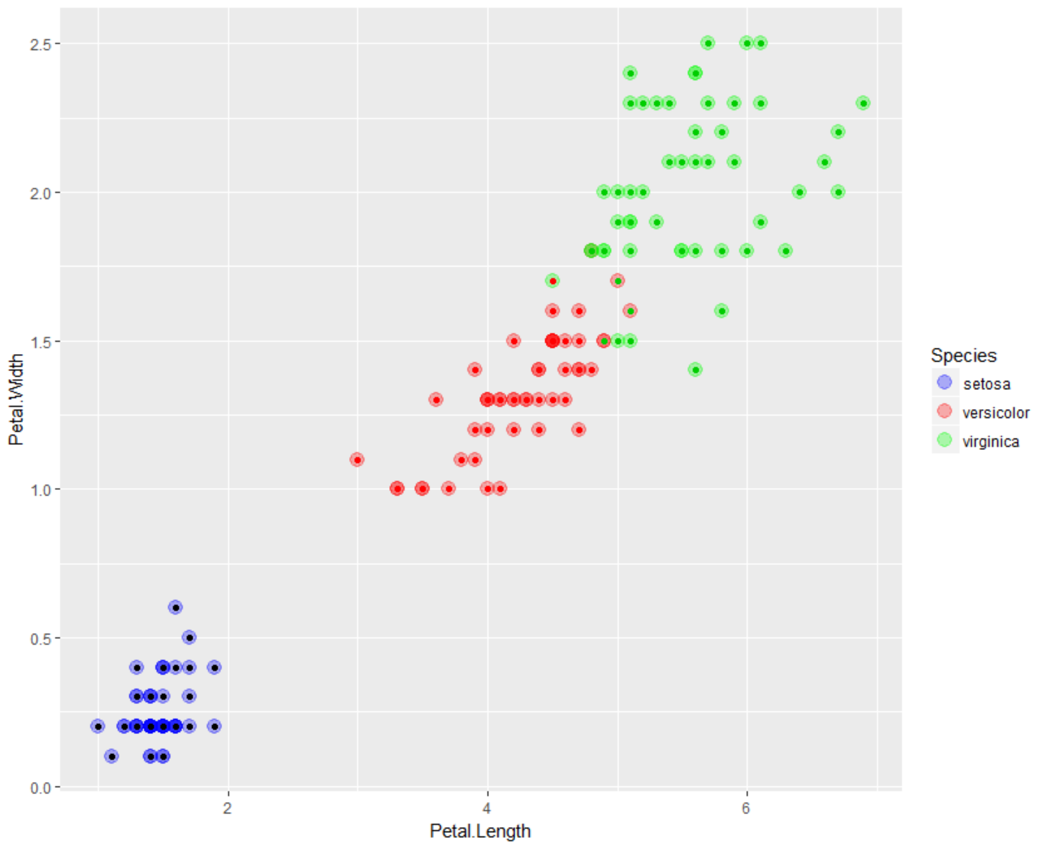


Figure 2-8 Clusters identified using the Hierarchical Clustering with Agglomerative Approach and Complete Link.

Therefore, three clusters can be identified in the dataset as shown in Figure 2-8, and in fact some data points in between two clusters may not be classified correctly. As shown in the figure, if data points are correctly classified, colour of the inside dot will be same as the colour of the outside ring.

Regarding zone design, Assuncao et al. (2006) and Guo (2008) have proposed the use of hierarchical clustering for building regions based on basic zonal units. In particular, Guo has reviewed three similarity measures (i.e. Single Link, Average Link and Complete Link) in conjunction with two contiguity constraints (i.e. consider only edges in the minimum spanning tree versus consider all the edges). However, their approaches strictly rely on a Minimum Spanning Tree (MST) generated from the data objects, which serves as a base for on-going clustering process. Clusters are formed by progressively cutting away vertices from the MST. This optimization process therefore focuses on each merging or dividing step, but not all the objects globally. In other words, the use of MST restricts the search as a local optimization and the partition it found is unlikely the global optimum. A practical advantage of hierarchical clustering is that the hierarchical structure can be visualized by a dendrogram and this is useful for some applications that can make use of the underlying structure found. To seek for global optimal, the use of partitional clustering will be reviewed in the following section.

2.13.Partitional Clustering

The concept of partitional clustering is different from that of hierarchical clustering in term of how they identify partition. In Partitional Clustering or sometimes called Exclusive Clustering, clusters are created without the hierarchical and nested structure. As we discussed in previous section, the formation of hierarchical structure and the clusters at different levels are primarily governed by specific similarity measures in both divisive and agglomerative approaches. In fact, the clustering process aims to finding some generalized or representative quality of the given data objects creating clusters by mean of merging or dividing action. For example, in Single Link, data carried by zone that has the least distance with its neighbouring clusters is used to represent the cluster it belongs to. Definitely, it will ignore the farthest parts in the cluster but it also gains simplicity for better computational performance. In most of the cases, Average Link is the best method because it summarizes the data rather than picking a single data to represent respective cluster. However, Partitional Clustering does not emphasize such localized searching and generalization process. It looks for an optimal partition of the entire solution space globally. Obviously, there will be tremendous ways for forming clusters. Partitional Clustering is a kind of global optimization method and one of the well-known examples is k-means.

In brief, k-means finds the best selected centroid of clusters by two-step iteration. K-means requires initialization by randomly assigning sufficient number of existing data points as cluster centres. The first step of the iteration is to assign data points to the closet centroid. Second step of the iteration is to update the centroids based on the data point assignment in the first step. These two steps are repeated until either the maximum number of iteration is reached or there is no further updating of the position of centroids.

To run partitional clustering, it is essential to define criteria, in which proximity matrix is a must, for identifying the fitness of each prototype partition. In theory, optimal partition can only be found by evaluating all possible solutions against the criteria and this is the major difficulty in Partitional Clustering. Owing to the limitation of resources and time, it is infeasible to do this by enumeration in reality. To solve this problem, a common approach is to use heuristic, which aims to find the optimal with the least number of evaluations in reasonable time. Unfortunately, simple heuristics may not be effective enough for solving the problem because it could be easily trapped in local optimum. I believe that heuristic is unlikely to handle complicated problem except if it has learning capability. Therefore, use of learning algorithm is a sensible approach and next Chapter details how it can be implemented in the zone design problem.

2.14.Evolutionary Algorithm

Different from traditional heuristic approaches such as hill-climbing, tabu search and Simulated Annealing, which can provide single iterated solution at a time only, GA is a population based evolutionary algorithm meaning that can provide multiple solutions but not a single solution in each generation. Although traditional heuristic has capabilities to improve solution quality, it is easy to be trapped in local optimum. With population based evolutionary algorithm like GA, it can easily jump from/to different areas in the solution space and search for the global optimum. To start with GA, an encoding method must be defined to encode all possible solutions so that it can be evolved by the genetic operators. Besides, objective functions are required to evaluate each solution and allow ranking of the solution for elitism selection. In general, there are two common genetic operators, i.e. crossover and mutation. Given at least two encoded solutions, the crossover operator would split the solution into two parts and then recombine them with one part from the first encoded solution and another part from the second encoded solution. On the other hand, mutation operator changes some components in the encoded solution and normally this will be done at certain low probability only because it does not always generate better results. The power of mutation operator is that it can help to explore the area containing new solutions quickly. Therefore, the two genetic operators are essential to evolve solution in GA and the evolving process should be repeated until termination criteria are met. Algorithm 2-2 shows a standard workflow of GA.

```
1. Generate initial population;
2. Repeat until termination criteria are met;
3. Evaluate fitness for each candidate in the current generation;
4. Select candidates;
5. Perform crossover until sufficient new candidates are created;
6. Draw candidate according to the probability of mutation and perform mutation;
7. Select best fit candidates and keep them in the next generation.
```

Algorithm 2-2 Genetic algorithm

However, the main drawback of GA is the difficulty of choosing appropriate parameters for a given problem (Roure et al. 2002). More precisely, the performance is highly dependent on the design of the encoding and the two genetic operators. Further, standard genetic operators are inadequate for handling two-dimensional spatial problems. Therefore, special genetic operators highly customized for recombining and mutating spatial data are essential to cope with spatial problems. (Xiao et al. 2002; Tavares-Pereira et al. 2008; Xiao 2008). Because there are very limited or even no study on the performance of these specially developed operators, the actual performance may not be well-understood or available. It is well known that GA will be greatly degraded when handling complex problem such as deceptive function. More precisely, fitness landscape would mislead the evolution process giving solution away from the global optimum and there would be high dependency among components in the solution (Pelikan 2008; Sangkavichitr & Chongstitvatana 2010). To address these limitations of GA, probabilistic based evolutionary algorithm was then developed and this further evolved so as to lead to the development of a new generation evolutionary algorithm known as Estimation of Distribution Algorithm (EDA). (Pelikan et al. 2002; Pelikan 2005)

In this study, EDA is adopted as the foundation to solve the zone design problem and the main difference between EDA and other evolutionary algorithms such as GA is that EDA relies on probabilistic model to generate new solution while GA requires specially designed genetic operators. Obviously, the concept of EDA is more intuitive.

2.15.Conclusion

In this chapter, I have defined and reviewed the zone design problem, its common constraints and criteria that can be used to generate regions, and also discussed its computational complexity. Furthermore, I have discussed the modelling issues and suggested that graph structure and Rook Contiguity are the most appropriate settings for modelling the zonal data for zone design.

Zone design is a well-known combinatorial optimization problem in geographic information science. Over the years, much research effort has been trying to solve the problem with different approaches. According to past experience, although exhaustive enumeration is a possible solution, it is infeasible to solve real problem. Meanwhile, heuristic is still the most popular approach to solve the problem because it can find solution with acceptance quality within reasonable time. In particular, the idea of partitional clustering method is one of the good candidates to cope with the zone design problem. However, traditional heuristic is easy to be trapped in local optima making it difficult to reach the global optima in the solution space. Therefore, evolutionary algorithms may be the last resort to tackle the problem. However, GA has its own limitations in particular when it is used to handle spatial problem. In particular, it requires to develop highly customized genetic operators while the understanding on how GA works is still very limited and not easy to explain. As the performance of GA would greatly degraded when handling solution with high dependency among its components, it may be much promising if there exists a solution searching method that can model inter-dependency among components. Therefore, it seems EDA is a right selection to tackle the problem because it has the probabilistic model for modelling the problem structure and it finds solutions by evolutionary approach. To our knowledge, this is the first-time EDA is applied in zone design problem. In the next chapter, I will provide a review on EDA and relevant different dependency models.

Chapter 3 – Estimation of Distribution Algorithms (EDA)

3.1. Introduction

This chapter is devoted to the introduction to Estimation of Distribution Algorithms (EDA) or Probabilistic Model-Building Genetic Algorithms (PMBGA), which is a heuristic approach aiming to tackle hard optimization problems (Pelikan 2005). Because the assignment of zonal units to regions in the zone design problem is usually based on discrete variables, this chapter will focus on EDA in discrete domain only.

3.2. Heuristic

Heuristic approach is an effective technique to tackle difficult problems, in particular, if the problems have the following properties: very large size of search space such that enumeration is not feasible and non-differentiable such that analytical method cannot be used. In brief, heuristic is realized by repeated refinement of existing best available solution, in which each refinement is usually obtained by educated guess. Although heuristic does not guarantee to find the optimal, it can get a sub-optimal solution within a reasonable time. Suppose if you encountered a very hard problem with limit time resource, heuristic would be the last resort to try. One of the popular heuristic approaches is Genetic Algorithm (GA), which is an evolutionary and population-based algorithm. Because GA is not problem-specific, it is also referred as a meta-heuristic approach. There are two pre-requisites to start GA. First, a proper encoding method that transforms candidate solutions to an evolvable form is required. Second, a set of genetic operators commonly known as crossover and mutation should be defined to evolve the encoded solutions. The crossover and mutation operators play different roles in GA. Crossover operator is a kind of exploitation search that creates a solution by recombining existing pair of solutions. On the other hand, mutation operator plays a role of exploration meaning it could create new solution in different areas in the search space by modifying an existing solution. Regarding encoding methods, there are common encodings such as binary encoding or real-valued encoding. Standard encoding is good to use because it always comes with a set of standard genetic operators. Both the standard encoding method and standard genetic operators have been studied by researchers for many years and therefore, their performance and capability are usually well-understood, and therefore development effort is minimal if they are applied in real problem. However, these standard methods are not suitable to all problems. In some situations, specially designed encoding and genetic operators are necessary.

In fact, GA has been applied in many spatial related applications. Bação et al. (2005) has demonstrated the use of GA to solve zone design problem, in which each zone design solution is represented by a list of identifiers. Each identifier represents a regional centre but not the actual assignment to each zonal unit. Each regional centre refers to either a centroid of existing zonal unit or any coordinates within the study area. Given the regional centres, each zonal unit can be assigned to the nearest one and consequently, zonal units belong to the same regional centre form a region. To evolve the encoded solution, tournament selection, uniform crossover and mutation with rate at 0.001 are used.

Tavares-Pereira et al (2001) also applied GA in the study of districting problem of a public transportation network pricing system in Paris. In their implementation, they use an explicit list of partitions to encode the solution. For example, to present a solution having five zonal units {1,2,3,4,5} in three partitions, a possible solution can be represented as {{1,2,},{3,5},{4}}. Meanwhile, special genetic operators for crossover and mutation that make use of the graph connectivity are developed to evolve the solutions. To further improve quality of the solutions, they also introduced a local search procedure.

Another GA example can be found in Correa et al. (2001) for solving a p-median problem. P-median problem aims to find P supply points to serve N demand points with minimum sum of distance, where both P and N are integer, and P must be smaller than N. Although this is not exactly a zone design problem, Hess et al. (1965) applied the same concept to solve the zone design problem. Nevertheless, it has been shown that GA works well on problem with tightly combined string representation such as the onemax problem. Onemax is a research problem for understanding how GA works. In brief, it is a maximization problem that aims to maximize the number of ones in a bit string. Bit string is composed of "0" or "1" only and each position of the bit string can serve as a building block of the optimal solution. In case when the building block spreads all over the solutions, performance of GA to combine them correctly would become poor because those generic crossover operators would not mix the building blocks properly and may eventually break it during the evolution process (Pelikan et el. 1999). GA is even worse in solving deceptive problems, in which the performance of GA would be lower than that of other heuristic approaches because objective function of the deceptive problems can mislead GA so as to get away from maxima and can be stuck in local minima. To cope with the limitations of GA, EDA is developed as a new breed of evolutionary algorithm (Pelikan et el. 1999; Pelikan et al 2002; Pelikan 2005). Different from the GA that relies on genetic operators, searching of optimum solution in EDA is realized by learning the probabilistic model of the promising solutions, exploiting the learnt probabilistic models, and exploring new solutions by sampling.

In the reminder of this Chapter, the basics of EDA and related probabilistic models for the discrete domain will be reviewed.

3.3. What is EDA?

General workflow of EDA can be described in several steps. First, population initialization is necessary to generate first set of solution candidates and it can be generated by a uniform probability distribution when there is no prior knowledge on the problem, and/or simply provided by prior knowledge such that existing best available solution can be included in the initial set of solution. However, when stochastic approach is used to generate population, it will not guarantee that all the candidates created are feasible solutions. In case impaired solution is created, it could be either repaired or simply discarded. It will proceed to next step unless sufficient number of feasible candidate solutions are generated. Next, all feasible candidate solutions should be evaluated according to the objective functions so that they can be ranked by fitness value. Subsequently, it will enter the following 5-step recursive process:

i. A set of promising solutions is selected according to a selection method ordered by fitness values;

ii. Given a probabilistic model, probability distribution is learnt from the promising solutions;

iii. New solutions are created by sampling technique based on learnt probabilistic model;

iv. New solutions, which are found infeasible, are either repaired or discarded;

v. Remaining feasible solutions with higher fitness value are selected and incorporated into the current population and a new generation of population is created.

This recursive process will be terminated until either fitness value is converged or number of maximum generation is reached. The basic procedure of EDA is outlined in Algorithm 3-1:

```
1    Start generation 0, t=0;
2    Generate a random population of size N.
3    While termination criteria not met do
     3.1   Select population of N promising solutions C(t);
     3.2   Learn probability distribution D(t) and structure S(t) of C(t);
     3.3   Generate new candidate solution N(t) according to D(t) and S(t);
     3.4   t = t+1;
     3.5   Integrate the newly created candidate solution to generate P(t);
4    End while
```

Algorithm 3-1 Estimation of distribution (Pelikan 2005; Hauschild and Pelikan 2011)

To illustrate the entire process, Figure 3-1 shows steps to perform EDA for finding the maximum of a 2-dimensional objective function i.e.$f(x, y) = x^2 + y^2$, whereas x and y are identical and independent random variables. Assuming the two variables are normally distributed, statistical parameters i.e. mean and variance, are learnt from the solution population recursively until termination criteria is met. At the beginning, a population is randomly initialized, in which its distribution is much diverged and in normal bell-shape. Quality of the population is evaluated and those solutions with high fitness quality are selected. Based on the promising population, statistical parameters are learnt again and used to generate a new population. After several iterations of selection, evaluation, and learning, the variance of distribution is approaching to zero. Finally, solution is converged and hence solution of the objective function is found.

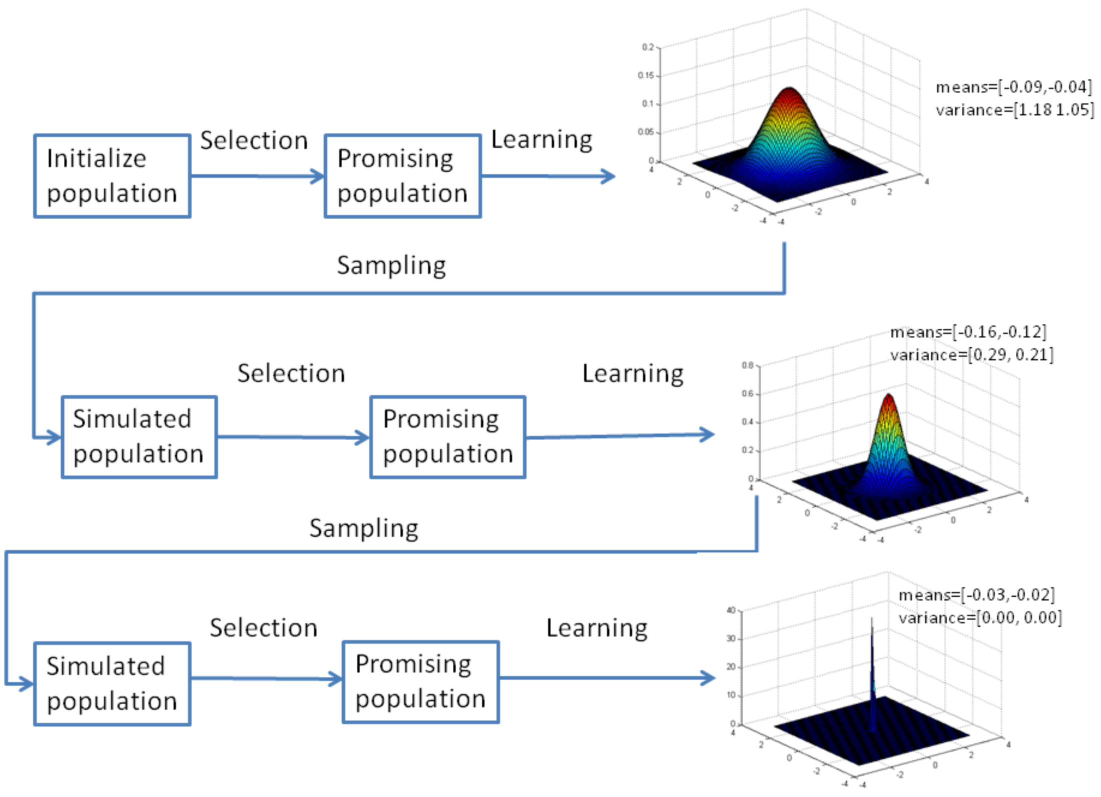


Figure 3-1 Procedure of EDA to find maxima of a 2-dimensional objective function

To improve the efficiency of the optimization process, a local search for improving the quality of each candidate solution can be performed before learning the probabilistic model. The two key steps to realize EDA are: which model should be used to describe the structural and statistical characteristics of the solutions; how to do sampling so that new solutions can be generated accordingly. Regarding the model, because it should handle multiple variables and their interaction, it should have two parts: interaction that the model can handle and the model parameters. As discussed, a solution is composed of variables. If a model that can properly describe interaction among the variables is found, it can be used to sample a large amount of new solutions efficiently and effectively. Further, model parameters can tell how likely the interaction will be realized by given data. In complex system, more interactions among variables are expected. Because typical probability model is not capable of handling both model parameters and interaction seamlessly. Graphical Probabilistic Models, which is a mixture of graph and probability is well-fit technique applying to abovementioned situation. In the following sections, I will give a review on Graphical Probabilistic Models including Bayesian Networks, Markov Networks and its sampling method, which are widely used in conjunction with EDA.

3.4. Graphical Probabilistic Models

The type of probabilistic model commonly applied to EDA for modelling complex interaction is called Graphical Probabilistic Models. In general, graphical probabilistic model is composed of two parts: (1) a set of model parameter; and (2) a graph structure. The set of parameters refers to the characteristics of probability distribution for the variables. On the other hand, the graph is a useful representation for modelling interaction (dependencies and independencies) among the variables. Two major graphical probabilistic models can be identified in the literature: Bayesian Networks and Markov Networks. The primary difference between these two models is that Bayesian Networks rely on Directed Acyclic Graph (DAG), which are useful for modelling causality, while Markov Networks rely on undirected graph, which are useful for modelling interaction without direction. Markov Networks are more appropriated for modelling problem of image analysis or spatial data (Koller and Friedman 2009). With the graphical probabilistic model, it can be used to perform conditional probability query and to find the most probable value of the model. The task to performing conditional probability query is to find out the probability of all possible assignment to the variables in the graph given some known or imaginative conditions. In other words, this is similar to the "What-if" analysis and the result is not just a "Yes" or "No" statement but also accompanies with the degree of certainty for the event. To find the most probable value, it is to identify the most likely assignment out of all possible assignments to the variables indicating which assignment is expected to happen at high certainty.

3.5. Bayesian Networks

Given $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ be a DAG, where V denotes a set of vertices and E denotes a set of edges of G, we define $\mathrm{X} = (X_1, \dots, X_n)$ be a set of random variables representing V while x_i represents the value of X_i, in which the random variables can either be discrete or continuous depending on the application. In respect to the zone design problem, I consider each zonal unit is a variable and the value assigned to it indicates which region it belongs to, and thus the variables are discrete values. In Bayesian Networks, vertices are connected by directed edges representing dependencies but no cycle is allowed. Instantiation of X defines an instance of solution and one can call X as Bayesian Network with respect to G, which has joint probability distribution that can be factorized as follows:

$$\mathrm{P}(X) = \mathrm{P}(X_1, \dots, X_n) = \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P}(\mathrm{X_i} | \Pi_i) \qquad (11)$$

Where $\Pi_i \subseteq \{X_1, \dots, X_n\}/X_i$ denotes a set of parents of $\mathrm{X_i}$, in other words, $\mathrm{X_i}$ has dependency to variables in Π_i. If $\mathrm{X_i}$ has no parent, then $\Pi_i = \emptyset$. As a result, the representation of a Bayesian Network is a pair (B_S, B_P), where B_S denotes structure of the DAG or G that determines the conditional dependencies of X, while B_P is a set of conditional probability distributions corresponding to the structure. Structure of DAG can be represented by an $\mathrm{n} \times \mathrm{n}$ adjacency matrix A where $A_{ij} = 1\, if\ edge\ i \rightarrow j\ exists, otherwise, A_{ij} = 0$. Bayesian Networks are commonly used for modelling the cause and effect relationships among events. The Bayesian Optimization Algorithm (BOA) developed by Pelikan et al. (1999) is an example to use Bayesian Networks for modelling data in EDA.

Nevertheless, is it possible to realize zone design with the Bayesian network? In general, the acyclic property of Bayesian Networks naturally do not fit to model the unidirectional interconnected relationship among zonal units. Of course, one could transform the graph of zonal units to a tree structure to fulfil the acyclic requirement. However, this would omit some interactions and therefore cannot model the original graph completely.

3.6. Sampling of Bayesian Networks

After a probabilistic model of a population (i.e. structure and parameters for a Bayesian Network) is learnt, representative solutions can be generated for building population of the next generation. This process is called sampling and can be done by a method called Probabilistic Logic Sampling (PLS) (Henrion 1988).

```
1. Input the network structure, i.e adjacency matrix;
2. Find an ancestral ordering ($\pi$) of the nodes of variables in the Bayesian Network;
3. For i=1,..,$|\pi|$, generate a value for $X_i$ from $P(X_i|\pi_i)$.
```

Algorithm 3-2 Probabilistic Logic Sampling (PLS) (Henrion 1988)

Given an ancestor-descendent relationship structure of the Bayesian network, the idea of PLS is that it samples variables only when all its parents have been sampled. To ensure parents are properly created, PLS initially computes an ancestral ordering of all the variables such that a child variable must be preceded by their respective parent variables. Next, sampling of each variable can be generated according to corresponding conditional probability. Pseudo code of PLS is shown in Algorithm 3-2.

3.7. Markov Networks

Markov Networks are another type of graphical probabilistic model. Similar to the Bayesian Networks, it is a pair (M_S, M_P), where M_S is structure of an undirected graph and M_P is set of parameter of the graph. Similarly, each node of the graph represents a variable whilst each edge represents a joint dependency between the variables. Instantiation of X defines an instance of solution and one can call X a Markov network with respect to M_S, which has global property that the joint probability distribution of X can be defined by the cliques of the undirected graph as follows:

$$\mathrm{P}(X) = \mathrm{P}(X_1, \dots, X_n) = \frac{1}{Z}\prod_{\mathrm{i=1}}^{\mathrm{m}} \Psi_i(c_i) \tag{12}$$

Each clique is a complete sub-graph of neighbouring nodes drawn from an undirected graph. Union of all the cliques is equivalent to the undirected graph. In equation (12), $\Psi_i(c_i)$ is the potential function on clique i where $c_i \in X$ and m denotes number of cliques in the graph. Further, Z is a partition function for normalization to ensure P(X) be a proper probability such that $0 \le \mathrm{P(X)} \le 1$ for any combination of X. EDA that makes use of the global property includes Distribution Estimation using Markov Networks (DEUM) (Shakya and McCall 2007), Markov Network Estimation of Distribution Algorithm (MN-EDA) (Santana 2005) and Markov Network Factorized Distributing Algorithm (MN-FDA) (Santana 2003). Besides, there is a local property in Markov Networks and it is defined as follows:

$$\mathrm{P}(X_i | X - \{X_i\}) = \mathrm{P}(X_i | N_i) \tag{13}$$

This is known as Markovianity (Besag 1974; Shakya and Santana 2012) and it indicates that the conditional probability of node X_i given other nodes in X is equivalent to the conditional probability of node X_i given the states of its neighbours N_i. The neighbours of X are also known as Markov Blanket for X_i. Obviously, modelling using the local property of Markov Network is rather straight forward because the conditional probability at each variable can be determined by its neighbours rather than all the variables. Examples of local property based EDA include Markovianity based Optimization Algorithm (MOA) and Markovian Learning Estimation of Distribution Algorithm (MARLEDA) (Shakya and Santana 2012).

3.8. Sampling of Markov Networks

In Bayesian statistics, posterior distribution can be estimated by sampling from the product of likelihood and prior distribution. However, in EDA, sampling is used to generate new and improved solution searching for best solutions. Regarding Markov Networks, sampling can be done by Gibbs sampler. The Gibbs sampler belongs to the class of Markov Chain Monte Carlo (MCMC) method and it has been used in EDA based Markov Networks. Different from PLS, Gibbs sampling does not require ordering of the variables, and sampling can be drawn directly from a conditional probability of each variable given the current state of remaining variables. Besides, Gibbs sampling is an iterative process so that number of cycles should be specified to run the process. An illustration of Gibbs sampler is as follows:

```
1   Assign a starting values S for the solution vector Θ:
2   For k times, do the following steps
    2.1   Sample $(\theta_1^j | \theta_1^{j-1}, \theta_2^{j-1} \cdots \theta_k^{j-1})$
    2.2   Sample $(\theta_2^j | \theta_1^j, \theta_3^{j-1} \cdots \theta_k^{j-1})$
    2.3   Sample $(\theta_3^j | \theta_1^j, \theta_2^j, \theta_4^{j-1} \cdots \theta_k^{j-1})$
    2.4           ⋮          ⋮
    2.5   Sample $(\theta_k^j | \theta_1^j, \theta_2^j \cdots \theta_{k-1}^j, \theta_{k+1}^j, \cdots)$
3   Sample based on given parameters is found.
```

Algorithm 3-3 Gibbs Sampler (Lynch 2007)

As conditional probability p(x|y) is proportional to its joint probability f(x, y) at fixed state of y and thus p(y) is a constant. The constant can be omitted to simplify the Gibbs sampling for most problems and therefore one can compute the joint probability only (Lynch 2007).

3.9. Types of EDA for Discrete Variables

Types of EDA can be classified by the inter-dependencies among variables. In general, there are three inter-dependencies scenarios: model without dependency, model with pairwise dependency and model with multivariate dependencies. These dependency models induce different models in form of a set of nodes, a chain, a tree or a group of connected nodes respectively as illustrated in Figure 3-2. The following sections will provide a brief description of each model with respect to discrete domain.

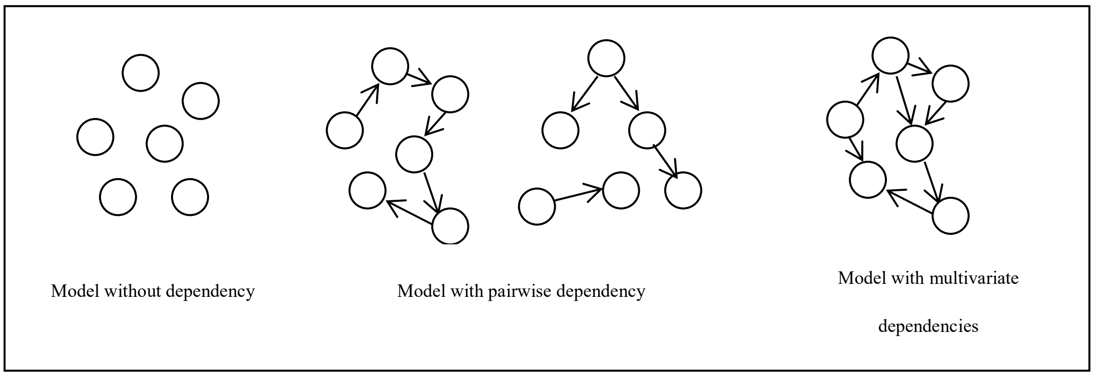

Model without dependency
Model with pairwise dependency
Model with multivariate dependencies

Figure 3-2 Graphical representation of different approaches for modelling dependencies (Pelikan 2005)

3.10.Models without dependencies

This is the simplest graphical model that assumes all variables are independent. Graphically, it can be depicted by separated nodes, in which each node represents an independent variable X. The joint distribution probability of a solution can be formulated as follows:

$$\mathrm{P}(X) = \prod_{\mathrm{i}=1}^{\mathrm{n}} P(\mathrm{x_i}) \tag{14}$$

In (14), $\mathrm{x_i} \in \{1, .., \mathrm{k}\}$ denotes all possible values of $\mathrm{x_i}$ where $i \in \{1, .., n\}$ and $\mathrm{k} \leq \mathrm{n}$, as such $P(\mathrm{x_i} = k)$ or $P(k)$ denotes the probability of X when it is equal to k. The probability density function or marginal probability when $\mathrm{x_i} = k$ can be defined as follows:

$$P(\mathrm{x_i} = k) = \frac{m(\mathrm{x_i} = k)}{n} \tag{15}$$

In equation (15), $m(\mathrm{x_i} = k)$ denotes the number of instances when the ith variable is assigned with the kth value. Each variable can be sampled by (15) forming a solution. It has been shown that this modelling approach work well for linear problem (Mühlenbein 1997). Several EDAs have been developed based on this model. The simplest one is the Population-Based Incremental Learning (PBIL) (Baluja 1994), which encodes solution as a fixed-length binary string and the population of solution is generated by a probability vector that is initially defined by uniform probability (i.e. 0.5). Best solutions are selected to create new generation. After several generations, probability would be shifted to indicate the best solution. Similar to PBIL, compact Genetic Algorithm (cGA) (Harik et al. 1998) is initialized by a probability vector with uniform probability. Two parent solutions are selected for each generation to generate two new solutions by a bit-by-bit tournament selection at each position of the solution string. In addition, probability at each position of the winning solution would be increased by 1/(population size) while losing solution will be decreased by 1/(population size). The Univariate Marginal Distribution Algorithm (UMDA)

(Mühlenbein 1998) is another example that uses equation (14) to compute distribution of the best solution. The probability distribution is then used to generate new solutions replacing the old ones until termination criteria are met. Algorithm 3-4 shows the pseudo codes of a generic EDA using Model without dependency that the solution of a new generation is constructed by the marginal probability at each position of the solution based on the population of current generation.

```
1    Start generation 0, t=0;
2    Generate a random population of size M.
3    While termination criteria are not met do
     3.1    Select population of $N \leq M$ promising solutions C(t);
     3.2    Learn the joint probability distribution $D_{t-1}$ of C(t);
                $P_t(x|D_{t-1}) = \prod_i \frac{\sum_{j=1}^{N}(X_i = x_i|D_{t-1})}{N}$
     3.3    Generate new candidate solution N(t) by sampling with $D_{t-1}$;
     3.4    t = t+1;
4    End while
```

Algorithm 3-4 Pseudo code of EDA with Model without dependency

3.11.Models with pairwise dependencies

Models with pairwise dependencies assume dependencies exist within pairs of variables. Graphically, pairwise dependencies can be illustrated by a chain. To build the model, the challenge is to find the optimal pairwise structure for a given set of selected promising solutions. Following Sections 3.12 and 3.13 will give a brief review of two EDAs based on model with pairwise dependencies namely Mutual-Information-

Maximizing Input Clustering (MIMIC) algorithm (de Bonet et al. 1997) and Bivariate Marginal Distribution Algorithm (BMDA) (Pelikan and Muehlenbein 1999).

3.12.Mutual-Information-Maximizing Input Clustering (MIMIC)

Mutual-information-maximizing input clustering (MIMIC) algorithm adopts a chain model, in which the starting variable follows a univariate probability while the remaining variables follow conditional probabilities given the variable at previous position in the chain. Based on the chain model as illustrated in Figure 3-3 Graphical representation of a chain model illustrating the pairwise dependencies, joint probability over a set of variables, X= $\{x_1, x_2, \dots, x_n\}$ can be defined as follows:

$$\mathrm{P(X)} = \mathrm{p}(x_1|x_2)\mathrm{p}(x_2|x_3), \dots, \mathrm{p}(x_{n-1}|x_n)\mathrm{p}(x_n) \tag{16}$$

It is assumed that there exists a true joint probability distribution over the set of variables and it requires a permutation to give an optimal structure. As such, the goal of MIMIC is to seek for a permutation of the variables that can generate samples matching with the true joint distribution probability as close as possible.

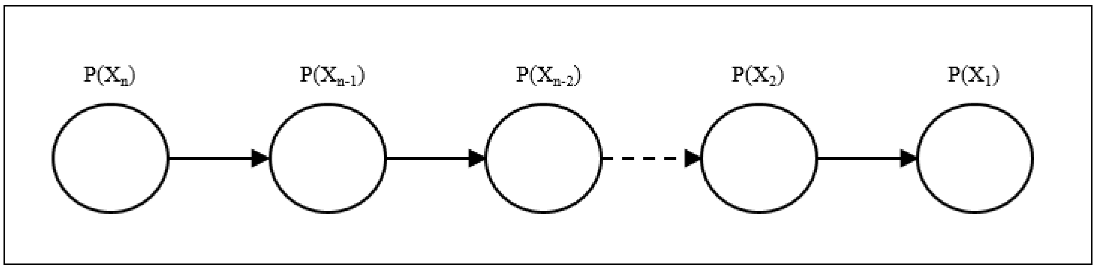


Figure 3-3 Graphical representation of a chain model illustrating the pairwise dependencies

Consider a permutation of variables $\pi = (i_1, i_2, \dots, i_n)$ that represents the sample structure of a solution set, a sample joint probability $\mathrm{P}_{\pi}(\mathrm{X})$ can be defined as follows:

$$\mathrm{P}_{\pi}(\mathrm{X}) = \mathrm{p}_{\pi}(\mathrm{X}_{i_1}|\mathrm{X}_{i_2}) p_{\pi}(\mathrm{X}_{i_2}|\mathrm{X}_{i_3}), \dots, p_{\pi}(\mathrm{X}_{i_{n-1}}|\mathrm{X}_{i_n}) p_{\pi}(\mathrm{X}_{i_n}) \tag{17}$$

The goal is to minimize the difference between the true joint distribution probability $\mathrm{P}(\mathrm{X})$ and the sample joint distribution probability $\mathrm{P}_{\pi}(\mathrm{X})$. To realize this, Kullback-Liebler (KL) divergence (Shlens 2014) is used, which measures the agreement between the two distributions. KL divergence can be expressed as the expectation of logarithmic difference between the distributions as follows:

$$\mathrm{D}_{KL}\big(P(X)||P_{\pi}(X)\big) = E_{P(X)}[log \frac{P(X)}{P_{\pi}(X)}] \tag{18}$$

$$= E_{P(X)}[logP(X)] - E_{P(X)}[logP_{\pi}(X)] \tag{19}$$

$$= -h(X) - E_{P(X)}\big[log\mathrm{p}_{\pi}(\mathrm{X}_{i_1}|\mathrm{X}_{i_2}), \dots, p_{\pi}(\mathrm{X}_{i_{n-1}}|\mathrm{X}_{i_n}) p_{\pi}(\mathrm{X}_{i_n})\big] \tag{20}$$

$$= -h(X) + h(\mathrm{X}_{i_1}|\mathrm{X}_{i_2}) +, \dots, + h(\mathrm{X}_{i_{n-1}}|\mathrm{X}_{i_n}) + h(\mathrm{X}_{i_n}) \tag{21}$$

In (20) and (21), $h(X)$ denotes the unconditional entropy of variable X or mean uncertainty of X and can be defined in (22)

$$h(X) = -E_{P(X)}[\log(P(X))] = -\sum_{x_i \in X} P(x_i) \log P(x_i) \tag{22}$$

Besides, $h(X|Y)$ denotes the conditional entropy of variable X given Y or mean uncertainty of X given Y and can be defined in (23).

$$h(X|Y) = \sum_{y} h(X|Y)P_y(y) \tag{23}$$

According to properties of KL divergence, it is always non-negative. When it is equal to zero, the two concerned distributions are identical. In this case, because the first term of the KL divergence i. e. $E_{P(X)}[logP(X)]$ does not depend on the permutation π, it is negligible and one can consider the remaining part only. Therefore, the remaining part can be defined as a cost function $J_{\pi}(x)$ to be minimized for finding the optimal structure as follows:

$$J_{\pi}(x)= h\big(X_{i_1}|X_{i_2}\big)+,\dots,+h(X_{i_{n-1}}|X_{i_n}) + h(X_{i_n}) \tag{24}$$

In other words, an optimal permutation π of the variables will give the least entropy with respect to the true distribution. Unfortunately, if there are n variables, the number of permutations is equal to Factorial of n (n!). Therefore, the search space is too big for exhaustive enumeration. To solve this problem, MIMIC employs a greedy algorithm that firstly finds the variable with the least entropy (or least uncertainty), then construct a chain by pairing up remaining variables based on the least conditional entropy recursively until all the variables are exhausted. Pseudo code of the greedy algorithm is shown in Algorithm 3-5.

Once a full chain model is constructed, the conditional probability distribution can be used for generating new solution candidates and the process is repeated until criteria are met.

```
1   For each selected solution set, identify a variable that provides the smallest entropy and
    assign it as $X_{i_n}$;
2   For i=n to 1
    2.1   Find variables Xj that gives the minimum among all possible conditional entropies:
          $h(X_{n-1}|X_k), h(X_{n-2}|X_k), \cdots, h(X_2|X_k), h(X_1|X_k)$
3   End for
4   The result is a list of $X_i$
```

Algorithm 3-5 MIMIC- Greedy Algorithm (de Bonet et al. 1997)

3.13. Bivariate Marginal Distribution Algorithm

Another EDA that makes use of pairwise model is called Bivariate Marginal Distribution Algorithm (BMDA) (Pelikan and Muehlenbein 1999). This EDA measures dependency of two variables at position i and j using conditional probability. Conditional probability of variables i and j can be calculated as in (25).

$$p_{i,j}(x_i|x_j) = \frac{p(x_i, x_j)}{p(x_j)} \tag{25}$$

To test the independence of two random variables, it uses Pearson's chi-square statistics as shown in (26).

$$\chi^2_{i,j} = \sum_{i,j} N \frac{(p(x_i, x_j) - p(x_i)p(x_j))^2}{p(x_i)p(x_j)} \tag{26}$$

According to product rule of probability, x_i and x_j are independent if they satisfy (27)

$$p(x_i, x_j) = p(x_i)p(x_j) \tag{27}$$

A statistical hypothesis test or Pearson's chi-square test is conducted to test for the independence of the variables, i.e. whether null hypothesis assuming the independence can be rejected. Consider 95% level of significance, the independence is true if $\chi^2_{i,j}$ < 3.84 with the degree of freedom is 1, i.e. p-value = 0.001. Otherwise, the hypothesis is rejected and an edge will be added for the pair indicating the dependency.

According to the dependency test, a graph can be constructed by adding edges for every pair of variables whenever the Pearson's chi-square test rejects the null hypothesis. By applying the test for each pair of variables, edges with sufficient dependency can be identified and form a model structure of the variables. The construction of the dependency graph can be briefly described as in Algorithm 3-6.

```
1   Initialize with a set of existing vertices V;
2   Set E = 0;
3   Set A = V;
4   Set R = 0; Node from each connected component of the graph
5   Select an node from A as v;
6   Add v to R;
7   Remove v from A;
8   If A is empty then exit;   Check if all the vertices have been visited.
9   Else Compute $\chi^2_{i,j}$ between v and each of v in V\A;
10  End if
11  If no dependency is found then go to 4;
12  Else Add edge (v,v') to E if $\chi^2_{i,j}$ between x and x' is maximized, replace v with v' and goto
    7;
```

Algorithm 3-6 Construction of dependency graph algorithm (Pelikan and Muehlenbein 1999)

New candidates can be created by sampling according to the dependency graph G=(V,E,R). The sampling has two steps. First, it computes the univariate marginal probabilities for each root node of connected components stored in R. Second, given structure of the dependency graph, it computes the conditional probabilities of remaining positions. Algorithm 3-7 shows the sampling method for BMDA.

```
1. Given G(V,E,R)
2. Identify a value for $X_{i_n}$ based on corresponding probability $p(X_{i_n})$;
3. For k = n-1 to 1, choose value for $X_{i_k}$ based on probability $p(X_{i_k}|X_{i_{k+1}})$.
```

Algorithm 3-7 Sampling for BMDA (Pelikan and Muehlenbein 1999)

According to Pelikan (2005), there are two drawbacks of using the pairwise model. First, it allows dependency limited to pairwise positions only. Second, there is no known algorithm for learning the best model distribution in polynomial time. Despite these disadvantages, EDA with pairwise model provides better performance and beats the ordinary genetic algorithm when the problem has dependencies of order two.

3.14.Models with Multivariate Dependencies

Different from the univariate models and pairwise models, multivariate models allows higher flexibility for modelling complex interaction among variables so that problems of bounded difficulty can be solved quickly, accurately and reliably. Nevertheless, it is inevitable that the more interaction among variables, the much computation resources are required. There are several EDAs falling in this category. Factorized Distribution Algorithm (FDA) (Muhlenbein et al., 1999) is specially designed to optimize Additively Decomposed Functions (ADF) with assumption that underlying dependency structure should be known in advance but not learnt from the distribution. However, this approach in general omits the concept of structure learning in EDA. The following

Sections 3.15, 3.16 and 3.19 will provide a brief description of two EDAs with directed graph including Extended Compact Genetic Algorithm (ECGA) and Bayesian Optimization Algorithm (BOA) for directed graph, as well as Markovianity Optimization Algorithm (MOA) for undirected graph respectively.

3.15. Extended Compact Genetic Algorithm (ECGA)

Extended Compact Genetic Algorithm (ECGA) makes use of the concept of Marginal Product Model (MPM) that is the product of marginal distributions according to partition of the variables. This is based on the idea of linkage learning, in which partitions are formed by grouping of several variables where linkages exist. Each identified partition is represented by an independent variable and will be subsequently processed by UMDA. To illustrate the concept of MPM, one can consider a 4-bit problem and three partitions can be defined such that: [1], [2,3], [4] in which the value within each square bracket denotes position along a text string. In the example, the 1st and 4th positions are independent, while 2th and 3rd positions are dependent. Therefore, each position can be assigned with either 0 or 1 while a 2-bit value {00, 01, 10, 11} to a partition which has two positions. Example showing the marginal distribution of possible value for each partition is given in Figure 3-4. It shows that for any individual, 0000 or 0110 is possible but 1101 or 1011 is not.

Partition	Probability distribution			
[1]	p(0)=1.0	p(1)=0		
[2,3]	p(00)=0.5	p(10)=0	p(01)=0	p(11)=0.5
[4]	p(0)=1.0	p(1)=0		

Figure 3-4 Marginal distribution model of the 4-bit problem example

To compose a proper MPM, ECGA uses a model selection method called Minimum Description Length (MDL) metric. Given different models of the same data, MDL metric selects the model having the minimum size so that higher ability to be compressed can be achieved. To account for model in binary encoding, the MDL metric is equal to the sum of the model complexity C_m and the compressed population complexity C_p denoting optimization measures as follows:

$$\begin{aligned}&\text{Minimize} && \text{MDL metric } C_c = C_m + C_p \\ &\text{Subject to} && 2^{S[I]} \leq N \ \forall i \in [1, S[I]]\end{aligned} \tag{28}$$

It is considered that the I^{th} partition of a MPM is of size S[I], and the sum of all S[I] is of length L. For binary encoding, each subset of size S requires $2^{S[I]}$ permutations, which is equal to the number of marginal probabilities that the model can represent. Take into account for all the MPMs, it is required to sum over all the $2^{S[I]}$ permutations. Moreover, suppose N is the population size, $log_2(N)$ bits are required for representing each of the marginal probability of MPM. As a result, the model complexity C_m, in term of total number of number of bits to store all marginal probabilities can be given by:

$$C_m = log_2(N) \sum_I 2^{S[I]} \tag{29}$$

On the other hand, let M_I be the marginal distribution of the I^{th} partition, the entropy[1] of this distribution is E(M_I). The entropy is defined as the average number of bits

[1] If the i^{th} event occurs with probability $p_i \in [0,1]$ the minimum number of bits required and entropy (average amount of information contained in each event) are given by:
$-log_2 p_i$ *and* $\sum p_i(-log_2 p_i)$ respectively.

required to represent the Ith partition in the population and thus it can be used to estimate the distribution's compression of the population (Harik 1999). This value is always smaller than S[I], i.e. length of the Ith partition. Therefore, the compressed population complexity C_p can be given by (30).

$$C_p = N \sum_{I} E(M_I) \tag{30}$$

Based on above measures, one can evaluate the efficacy of any given MPM and its structure.

To find a good MPM, ECGA uses a greedy algorithm starting with each variable representing one partition. In brief, two selected partitions will be merged if the overall MPM can improve the MDL metric. Otherwise, current model is used. Main steps of the ECGA are shown in Algorithm 3-8.

Although ECGA works for identifying linkage between variables, it does not allow overlapping linkage that limits the modelling capability.

```
1. Generate a random population of size N.
2. Undergo tournament selection at a rate S.
3. Model the population using a greedy MPM search.
4. If the model is converged, stop.
5. Generate a new population using the given model.
6. Return to step 2.
```

Algorithm 3-8 Extended Compact Genetic Algorithm (ECGA) (Harik 1999)

3.16.Bayesian Optimization Algorithm (BOA)

Bayesian Optimization Algorithm (BOA) is another ED. It uses multivariate dependencies for modelling underlying data. Given a set of promising solutions, BOA firstly builds a Bayesian network by learning the structure and parameters that are represented by a DAG and conditional probabilities among variables respectively. Given a set of variables X_i and a Bayesian Network, a joint probability of an instantiation can be defined as follows:

$$p(\boldsymbol{X}) = \prod_{i=1}^{n} p(x_i|\Pi_i) \tag{31}$$

In (31),$\boldsymbol{X} = (x_1, x_2, \dots, x_{n-1}, x_n)$ $\Pi_i$denotes the parent variables of x_i that is indicated by existence of directed edge from x_j to x_i meaning x_j is ancestor of x_i this also means that there is dependency between x_j and x_i. The entire Bayesian Network has to be acyclic. Otherwise, the conditional probability of a variable cannot be determined properly. To reduce the complexity, usually number of incoming edges (k) to each node is bounded by a pre-defined number. Once the network structure is learnt, parameters of each variable can be determined, and subsequently it can be used to generate new solutions by sampling. The process is repeated until some termination criteria are met. Main steps of BOA can be found in Algorithm 3-9.

```
1    Set t=0;
2    Generate initial population P(0);
3    Evaluate the population P(0) based on the fitness equation;
4    While stop criteria not met
     4.1    Select population of promising solutions from P(t) as S(t);
     4.2    Build Bayesian Network B(t) for S(t);
     4.3    Sample B(t) and generate N(t);
     4.4    Incorporate the new solutions N(t) into P(t) by a replacement method;
     4.5    Set t = t+1;
5    End while
```

Algorithm 3-9 Bayesian Optimization Algorithm (BOA) (Pelikan 2005)

BOA follows most of the EDA steps to evolve solutions, except BOA has its own structure learning method to distinguish the best structure from others. In BOA, two different approaches have been introduced to formulate scoring metric of a Bayesian network. This includes Minimum Description Length (MDL) based scoring metrics and Bayesian scoring metric. A brief description of the two approaches will be provided in the following sections.

3.17.Bayesian Scoring Metric

In BOA, the Bayesian-Dirichlet equivalent (BDe) (Heckerman et al., 1994) can be used to determine the quality of network structure. In order to determine a proper network structure, it considers the posterior probability of a Bayesian network B given data D based on the Bayes theorem as shown in (32).

$$p(B|D) = \frac{p(D|B)p(B)}{p(D)} \tag{32}$$

In (32), p(B|D) is the posterior probability of the Bayesian Network given data that is what should be learnt from the data, p(D|B) is a likelihood function. It is not a probability but indicates how likely the data is obtained with the given Bayesian network, p(B) is prior probability of the Bayesian Network and p(D) is the probability of the data. P(D) can be determined by summing across all the parameters as shown in (33).

$$p(D) = \sum_{p(B)} p(D|B)p(B) \tag{33}$$

The posterior probability p(B|D) indicates the probability of the Bayesian Network given the data. The higher the posterior probability indicates the higher certainty of the network generating from the data. Given the generated data and a Bayesian Network, the posterior probability serves as an indicator for the goodness of the network. As the p(D) is a constant for all possible networks, the posterior probability can be written as (34) for scoring the network structure.

$$p(B|D) = p(D|B)p(B) \tag{34}$$

In order to compute *p(B,D)* in closed form, the following seven assumptions have been made on the *p(B,D)*. For details, pleases refer to Heckerman et al. (1994)

i. The database D is a multinomial sample;
ii. Given the network structure, all cases are independent;
iii. All databases are complete meaning there is no variables with missing data;
iv. Parameter modularity and thus parents determine parameters;

v. Probability distribution of the parameters follows Dirichlet;

vi. All complete structures are possible;

vii. Likelihood equivalence meaning scores of equivalent networks are equal.

In view of the above assumptions and considering Z be a set of n discrete variables, the scoring function can be given in (35).

$$p(B,D) = p(B)\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{(\mathrm{r_i}-1)!}{\left(N_{ij}+\mathrm{r_i}-1\right)!}\prod_{k=1}^{r_i}N_{ijk}! \tag{35}$$

In (35), each variable $\mathrm{x_i}$ in Z has $\mathrm{r_i}$ possible states, i.e. $\left(x_{i1},\dots x_{ir_i}\right)$ and $q_i = \prod_{x_l \in \Pi_{x_i}} r_l$ is the number of unique configurations of the parent set Π_{x_i} of $\mathrm{x_i}$. Let D be a database of m cases where each case follows the possible states of variables $\mathrm{x_i}$. Let w_{ij} denotes the jth unit configuration of Π_{x_i}. Furthermore, N_{ijk} is the number of cases found in D, in which the variable $\mathrm{x_i}$ is at kth state (i.e. x_{ik}) and at the jth configuration of its parent set and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ denotes the number of cases found in D for all variables $\mathrm{x_i}$ for $\mathrm{i} \in (1,\dots,\mathrm{n})$ and its parent set Π_{x_i}takes the jth unit configuration. As $p(B)$ is fixed. The following function $\mathrm{g}(\mathrm{i},\Pi_i)$ should be be maximized to obtain maximum $p(B,D)$:

$$\mathrm{g}(\mathrm{i},\Pi_i) = \prod_{j=1}^{q_i}\frac{(\mathrm{r_i}-1)!}{\left(N_{ij}+\mathrm{r_i}-1\right)!}\prod_{k=1}^{r_i}N_{ijk}! \tag{36}$$

Furthermore, the following is the general form of the scoring metric.

$$p(B,D) = p(B)\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{\Gamma\left(N_{ij}'\right)}{\Gamma\left(N_{ij}+N_{ij}'\right)}\prod_{k=1}^{r_i}\frac{\Gamma\left(N_{ijk}+N_{ijk}'\right)}{\Gamma\left(N_{ijk}'\right)} \tag{37}$$

In (37), Γ is the Gamma function, i.e. $\Gamma(n) = (n-1)!$; $N'_{ij} = \sum_k N_{ijk}{}'$ which is a prior probability distribution of the parameters given the network structure. This is similar to p(B),which represents the prior probability establishing a preference bias for the network structure . If $N'_{ijk} = 1$, this metric is reduced to the K2 metric. Cooper and Herskovits 1992 have introduced a heuristic algorithm called K2 algorithm to find the best Bayesian Network.

```
1     {Input: A set of nodes, an ordering on the nodes, an upper bound u on the number of parents
      a node may have, and a database D containing m cases.}
2     {Output: For each node, a printout of the parents of the node.}
3     For i:=1 to n do
      3.1    $\Pi_i = \emptyset$;
      3.2    $P_{old} = g(i, \Pi_i)$;
      3.3    OKToProceed := true
      3.4    While OKToProceed and | $\Pi_i$ |<u do
             3.4.1  let z be the node in Pred($x_i$)- $\Pi_i$ that maximizes $g(i, \Pi_i \cup \{z\})$;
             3.4.2  $P_{new} = g(i, \Pi_i \cup \{z\})$;
             3.4.3  if $P_{new} > P_{old}$ then
             3.4.4      $P_{old} = P_{new}$ ;
             3.4.5      $\Pi_i = \Pi_i \cup \{z\}$;
             3.4.6  else OKToProceed := false;
      3.5    End {while}
      3.6    Write("Node:", "parents of this nodes :", $\Pi_i$);
4     End {for}
5     End {K2}
```

Algorithm 3-10 K2 Algorithm (Cooper and Herskovits 1992)

1

vertices are not allowed to be a descendent of succeeding vertices. It uses a greedy algorithm searching for the maxima of the scoring function by iteratively adding, removing or reversing edges. The construction is terminated when improvement is not further possible. Next, new population is generated by sampling using the structure and parameters of the Bayesian Network learnt in two steps. First step computes an ancestral ordering of variables. Second step computes the values of the variables according to the ancestral ordering such that parents of each variable can be generated prior to itself. Since Bayesian Networks can encode more complex dependencies and independencies, they are applicable to problems with overlapping dependencies.

3.18.Scoring metric using Minimum Description Length

As discussed in the previous sections, MDL is based on the concept of information theory. In particular, it measures the model quality by determining the size of model encoding method and the size of encoded data according to the encoding model. Therefore, the best model would use least data size for describing subject data. There are different approaches to the design of MDL metrics and the two-part approach is commonly adopted for structure learning. The two parts refer to data description given the model and description of the model respectively. Regarding the first part, it equals to the log likelihood function, $\log\big(p(D|S,\hat{\theta})\big)$ of the data respect to the structure S. This value can be maximized by a maximum likelihood estimate $\hat{\theta}$. Using the same notation of the K2 metric, the log likelihood function can be derived as follows:

$$\log \mathrm{L}(\mathrm{D}|\mathrm{S},\theta) = \log p(D|S,\theta) \tag{38}$$

$$= log \prod_{w=1}^{N} p(x_w|S,\theta) \tag{39}$$

$$= log \prod_{w=1}^{N} \prod_{i=1}^{n} p(x_{w,i}\,|\Pi_i^S,\theta_i) \tag{40}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \log(\theta_{ijk})^{N_{ijk}} \tag{41}$$

Since the maximum likelihood estimate $\hat{\theta}$ for θ_{ijk} is given by $\frac{N_{ijk}}{N_{ij}}$, the log maximum likelihood function can be written as follows:

$$\log p(D|S,\hat{\theta}) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} log\left(\frac{N_{ijk}}{N_{ij}}\right) \tag{42}$$

Regarding the second part, it refers to the number of parameters of the factorized joint probability distribution with a non-negative penalty function $f(N)$. In general, the number of parameters is given by $\prod_{i=1}^{n} q_i(r_i - 1)$.

As a result, this leads to the following MDL scoring function.

$$\text{MDL}(D|S,\hat{\theta}) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} log\left(\frac{N_{ijk}}{N_{ij}}\right) - f(N) \prod_{i=1}^{n} q_i(r_i - 1) \tag{43}$$

There are three possible values to the penalty function as follows:

- If $f(N) = 1$, the metric known as Akaike's Information Criterion (AIC) (Akaike, 1974);
- If $f(N) = \frac{1}{2} logN$, the metric known as Bayesian Information Criterion (BIC) (Schwarz, 1978);
- If $f(N) = 0$, the metric is simply a maximum likelihood and the best network with these criteria is a complete network, which has edges for every two possible nodes. However, it will violate the acyclic requirement of a Bayesian network.

In addition, the log-likelihood function can be represented by the conditional entropy $\text{H}(\text{X}_\text{i}|\Pi_i)$ of X_i given parent Π_i as follows:

$$\log \mathrm{L}\big(\mathrm{D}\big|\mathrm{S},\hat{\theta}\big) = -\sum_{i=1}^{n} \mathrm{H}(\mathrm{X_i}|\Pi_i)N \tag{44}$$

In (44), $\mathrm{H}(\mathrm{X_i}|\Pi_i) = -\sum_{x_i,\pi_i} p(x_i,\pi_i)\log p(x_i|\pi_i)$; $p(x_i,\pi_i)$ is the joint probability with $\mathrm{X_i} = x_i\ and\ \Pi_i = \pi_i$, and $p(x_i|\pi_i)$ is the conditional probability of $\mathrm{X_i} = x_i\ given\ \Pi_i = \pi_i$. The conditional entropy denotes the length of description for respective value of $\mathrm{X_i}$ given Π_i. This metric favours model with variables, which can give as much information as possible about its parents so as to compress the values of the variable, which therefore gives a higher value of the metric. For further reading of Minimum Description Length, please see Grünwald (2007)

According to Pelikan (2005), the Bayesian metrics would capture unnecessary dependencies because it tends to be sensitive to noise of the data. Order of interactions is usually restricted to lower the complexity of the model. Besides, MDL metric favours simple model and it requires a large amount of data in order to learn a model so as to capture all necessary dependencies.

3.19.Markovianity based Optimization Algorithm (MOA)

As discussed in previous sections, Markov Networks rely on undirected graph for modelling dependencies among variables. MOA is an example of EDA applying Markov Networks by exploiting the local Markov property.

```
1. Generate a random population of size N.
2. Select a set of M best solution candidates, where M<N;
3. Estimate the Markov Network structure from the set;
4. Compute the local conditional probability for each variable;
5. Generate a new population of size N based on the Markov network and its parameters.
6. Return to step 2 until termination criteria are met.
```

Algorithm 3-11 Markovianity based Optimization Algorithm (Shakya and Santana 2012)

Similar to ordinary EDA, MOA is starting with a population of solutions, which could be initialized by random generation or previous known solutions. After identifying the best solution candidates, they would be used to estimate the undirected structure. In MOA, structure learning is achieved by determining the Mutual Information (MI) between every possible pair of variables. In brief, MI uses entropy (i.e. also called Shannon Entropy that indicates amount of information the variable carries.) of each variables and entropy of the pair to measure their mutual dependence. Given two variables A and B, MI between the variables can be determined by (45) as follows:

$$MI(A,B) = \sum_{A,B} p(a,b) \log(\frac{p(a,b)}{p(a) \cdot p(b)}) \tag{45}$$

In (45), p(a,b) denotes the joint probability of A and B when A=a and B=b. By applying the equation to each pair of variables in the selected population, a matrix of mutual information can be created. Next, when the MI is greater than a threshold, an edge will be added between two corresponding variables, in which the threshold is defined as follows:

$$\text{Threshold to add a new edge} = \text{Mean(MI)} * \text{significant parameter} \tag{46}$$

In Santana's paper (Shakya and Santana 2012), the significance parameter is set to 1.5. Further, in order to limit the complexity, the number of edges connecting to a variable is also limited to a certain number except the case when the excessive edge has the highest mutual information.

After the structure is learnt, conditional probability of each variable or the local property of the Markov Network is determined based on the population and this also completes the entire probabilistic graphical model. The next step is to generate a new generation of population using sampling technique. Gibbs sampler, which is a class of MCMC method, is used in MOA. To start a Gibbs sampler, a solution candidate should be given. Then, for a fixed number of iteration, a sample or new solution candidate can be created by successive sampling of each variable based on the conditional probability in the given solution candidate. The conditional probability of each variable given its neighbour can be determined by (47) as follows:

$$P(x_i|N_i) = \frac{P(x_i, N_i)}{\sum_{x_i} P(x_i, N_i)} \tag{47}$$

In (47), $\mathrm{x}_i \in \{0,1\}$ if binary variable is used. MOA also defines a temperature based Gibbs sampler as follows:

$$P(x_i|N) = \frac{e^{P(x_i,N_i)/_T}}{\sum_{x_i} e^{P(x_i,N_i)/_T}} \tag{48}$$

In (48), T denotes temperature, which is used to control the convergence of the probability. Similar to the temperature in Simulated Annealing, it allows the search to change from exploration of the new regions of search space to exploitation of search space slowly in order to allow extensive search for global optima. In MOA, T is defined as $\frac{1}{g\times CR}$, in which g denotes the current generation and CR denotes a cooling rate parameter. When the CR is set with a high value, the conditional probabilities will have quick convergence but less exploration. On the contrary, when the CR is set with a small value, convergence of the conditional probabilities becomes slower but with more exploration. List of steps of the Gibbs sampler is as follows:

```
1. Randomly select a solution candidate from the current population;
2. For r iterations, do
    a. Randomly select a variable $x_i$ from the given solution candidate;
    b. Using the Markov network model (i.e. structure and parameters) to compute the
       conditional probability $P(x_i|N_i)$ of $x_i$ as Gibbs probability;
    c. Sample $x_i$ using $P(x_i|N_i)$
    d. Go to i. until all variables are exhausted.
3. Terminate with a new sample of solution candidate.
```

Algorithm 3-12 Gibbs sampler for MOA (Shakya and Santana 2012)

3.20.Conclusion

This chapter has introduced the concept of probabilistic graphical model as well as EDA and more importantly, how the probabilistic graphical model is used in EDA. In fact, probabilistic graphical model realizes data learning in the EDA process, in which it greatly improves the capability for searching the best solution in the search space. There are two common classes of probabilistic graphical model: Bayesian Networks and Markov Networks. Bayesian Networks represent variables and their inter-dependency via directed acyclic graph, in which it is commonly used to model causality relationships in term of probability. Given some observations or data, Bayesian Networks can be used to model the causal relationship among variables with conditional probability showing how likely particular values will be assigned to corresponding variables. The typical real-life example application of Bayesian Networks is modelling of diseases and its symptoms. As such, it can be used to compute how likely the presence of the diseases. On the other hand, Markov Networks represent variables and their inter-relationship via undirected graph, in which cyclic dependence is allowed. Due to the characteristics of Markov Networks, it can be used to model lattice data and spatial data naturally. In contrast with the ancestor and descendant relationship, the non-directed characteristics of dependency in Markov Networks can be used to model the adjacency relationship or spatial relationship in the spatial data model.

Moreover, three dependency relationships that can be employed in EDA for modelling variables have been reviewed. They are “no dependency”, “pairwise dependency” and “multiple dependency”. In general, the more complex the relationship among the variables, the higher order of dependency should be used. Dependency between variables can be determined by different approaches such as mutual information, statistical test and so on. However, the major challenge is to scrutinize all possible dependencies relationship from the tremendous data and identify the most probable one adding to the model. This is an intractable problem. To make the problem can be solved within a reasonable time, various measures are applied to limit the problem size.

Based on the understanding of the zone design problem, EDA as well as the Markov property, I will present the detailed construction of EDA-based zone design in next chapter.

Chapter 4 – EDA based Zone Design

4.1. Introduction

This chapter provides details on how to apply EDA to solve the zone design problem. In general, this is developed based on the steps outlined in Section 3.3. Different from optimization problems that discussed in previous Chapter that variables are expected to be implicitly connected with each other and various methods, for example, use of hypothesis test and measure of mutual information, are proposed to find the dependency. In zone design, each zonal unit is represented by a variable, and the collection of it is fundamentally and explicitly connected to each other according to the spatial topology. As a result, additional handling is a must to ensure spatial continuity accordingly for each solution found when EDA is running. Otherwise, it will be very hard for EDA to identify a configuration conforming to the contiguity constraint no matter how high fitness the solution can achieve because spatial topology is extremely difficulty to be re-constructed and it can be easily be violated by statistical sampling.

As discussed, graphical probabilistic model in EDA is composed of two parts: inter-dependency structure and model parameters. To solve the zone design problem using EDA, the topological structure given in the zonal data would be directly adopted to define the model structure. This not only ensures the right spatial topology over the evolutionary process, but also greatly reduces computational effort. On the other hand, according to the local property of Markov Networks, conditional probability of each variable given the population can be determined at each generation as the model parameters. In the following sections, I discuss major components of the EDA based Zone Design algorithm, and subsequently analyse its computational complexity.

4.2. Initial population

In this study, group-number encoding is adopted. In brief, each solution of zone design must be encoded as a solution string of size n where n denotes the number of basic zonal units within the study area. Each position of the solution string represents a zonal unit and is treated as a variable. It accepts values within the range of possible region numbers that is expected to create in a zone design problem. Suppose R denotes the possible values that could be assigned to each basic zonal unit. Disregarding the continuity constraint, the solution size of the zone design problem is equal to $|R|^n$ because each variable has R possible values. Figure 4-1 shows an example of an encoded zone design solution. In the figure, first row shows the identifiers of the basic zonal units while second row represents the region assignment, e.g. Zone 1 is assigned to region 3. Collectively, the entire string gives configuration or a possible zone design solution.

Zone	1	2	3	4	5	6	7	8	9	10	…
Region	3	3	2	2	3	1	1	3	2	3	…

Figure 4-1 Encoding of a zone design solution

To start EDA, an initial population of specified size is created, in which each solution have to comply with the hard constraint of contiguity and is evaluated by the objective function. In normal circumstance, when there is no a priori information, initial population can be generated by sampling of each variable at random based on uniform distribution.

Solution diversity is very crucial in the initial population because it can greatly increase the chances of reaching various potential areas in the search space where optimal solution can be found. In other words, to win at the starting line, diversity matters. Otherwise, it would be a great challenge to test the exploration capability of the searching mechanism and this will be discussed in Section 4.5. To increase solution diversity, one simple method is to increase the population size. However, it is always good to include high-quality known solutions in the initial population because they can speed up the exploration process.

It must be noted that when solution for spatial problem is encoded, e.g. the zone design, it is impossible to ensure all the basic spatial units or variables collectively complying with correct topological structure using bit-by-bit random assignment. The reason is that each zonal unit is in fact dependent on each other geographically. According to the first law of geography, everything is related to everything else, but near things are more related than distant things (Tobler 1970). Randomly assigning a region to each zonal unit disregarding the spatial relationship among the zonal units will violate the contiguity constraint and this will not work for zone design. However, it there a way to generate solutions using random assignment efficiently? I suggest that it can be done in a reverse manner that each basic zonal unit should be assigned to a group of given regions rather than finding a set of basic zonal units forming a region. This can ensure that spatial contiguity can be maintained naturally due to region-centric thinking.

To realize generation of feasible solutions that can properly group basic zonal units into expected number of regions during population initialization, the following three methods are considered: (1) Partitioning by cutting a minimum spanning tree; (2)

Partitional clustering method using k-means; and (3) Random assignment of regional centres.

Use of partitioning by cutting minimum spanning tree can be found in Assunção et al. 2006. To apply this approach, the basic zonal units and their spatial relationship should be initially modelled as a graph, whereas a node represents a zonal unit and an edge connecting a pair of zonal units denotes the existence of adjacency relationship. Further, a proximity measure should be defined to build a minimum spanning tree. The use of minimum spanning tree aims to provide a structure connected by homogenous nodes. In zone design, dissimilarity of characteristics of interest such as population and any socio-economic attributes of interest, between a pair of connecting nodes can be used to define the proximity. Suppose there are two connecting nodes, i and j, the characteristic of interest denoted as c, can be defined as follows:

$$\mathrm{Dissimilarity}_{\mathrm{ij}} = \sum_{\mathrm{c}} (\mathrm{x}_{\mathrm{i}}^{\mathrm{c}} - \mathrm{x}_{\mathrm{j}}^{\mathrm{c}})^2 \tag{49}$$

Subsequently, a minimum spanning tree of the graph model with respect to the dissimilarity can be created, such that only nodes having the least similarity are connected. This model is very appealing because it not only preserves the natural topological connectivity but also reveals the homogeneity structure of the underlying zonal data. Given the number of partitions or regions k, k-1 edges can be identified and then removed from the minimum spanning tree forming k sub-trees. Each sub-tree represents a single partition/region. For the purpose of initialization, the k-1 to-be-removed edges can be selected from the pool of edges randomly. Providing the selected edges do not share any node, this assures the correct number of partitions/regions can be created. Algorithm 4-1 provides necessary steps for the population initialization by cutting minimum spanning tree.

```
1. Given a set of contiguous zonal data (in the form of topological-correct polygons) and number
   of partitions/regions k to be created;
2. Create a graph based on a set of contiguous zonal data; in which each zone carries the
   characteristics of interest;
3. Assign a measure of similarity to each edge;
4. Create minimum spanning tree respecting the similarity;
5. Remove k-1 edges from the tree; and
6. Return a set of sub-trees in term of a solution string, each represents a partition or region.
```

Algorithm 4-1 Population Initialization using cutting Minimum Spanning Tree approach

Although the abovementioned approach can provide solution with correct number of regions and meet the hard constraint of contiguity, it will create regions with unbalanced number of zonal units. This is obvious when the expected number of regions is small because dissimilarity of a single zonal unit is always zero resulting a single-node region. This is undesirable as it will make an unbalanced result. To handle this problem, in the work of Assunção et al. 2006, they added a lower bound for selected attribute values of each region so that it will be rejected when it violates the constraint.

Second approach refers to the k-means clustering method for generating population. As discussed in Section 2.13, k-means is a popular and simple partitional clustering method. Given an expected number of regions k and coordinates of n basic zonal units, k-means can partition the n zonal units into k regions. In short, each zonal unit will be assigned to the nearest regional centre. This method solely relies on proximity but not the topological structure of the zonal units. Usually, Euclidean distance will be used. One of the advantages of this method is that it ensures all the solutions conforming to the contiguity constraint and has a good balance of number of zonal units among regions.

Although k-means is starting from random points whereas distribution of points may not be well-balanced, solution will be converged when there is no further movement of the regional centres. To generate solution for further generation, provision of the converged solution becomes a limitation that population will be at very low diversity or may always converge to a single solution. This will definitely increase the difficult for exploring new solutions.

Similar to the second approach, third approach is relatively simpler that makes use of random selection of zonal units as cluster centres to develop partition only. With the selected zonal units as cluster centres, remaining zonal units will then be assigned to the nearest cluster centre forming new solution without conducting any further refinement. Quality of the newly generated solution may not be good but this is not the concern at the initial population generation stage. More importantly, this can increase diversity of the population and the very hard part of ensuring contiguity can be achieved naturally. Since this approach relies on Euclidean distance between zonal unit and each regional centre, it is very sensitive to their positions. As the zonal unit is represented by a point usually refers to its centroid as its position. Any small movement of the position will generate different result. In particular, it will generate region with disjoint parts when zonal unit is too large and connecting to many small-size zonal units. Therefore, although this approach can generate contiguity regions, it does not guarantee all the samples are conforming to the contiguity constraint.

In this study, I will adopt the third approach for population initialization, not only because it is the simplest one but also it is among the three approaches that requires the least computation efforts with higher chance to obtain balanced partitions. To avoid disjointed regions at the initialization stage, validation check is required to ensure the data integrity of each created region. Of course, the most direct method to avoid this issue is to use basic zonal unit data of even and similar size. Furthermore, I will discuss the group-number encoding in Section 5.2 with details in implementation.

4.3. Selection

Once a population of valid solutions is created, subset of the population will be selected for learning the statistical model. In general, selection is a biased logic that favours solution with higher fitness score. In this implementation, roulette wheel selection or proportional selection with replacement (Lipowski and Lipowska 2011) is adopted. In brief, these methods will select particular solution based on probability in proportional to its fitness score. Characteristic of these selection methods is that it preserves diversity because solution with low fitness score has certain chance to be selected, no matter how low the probability is. And these low fitness solutions may help to explore the high fitness landscape of the solution space. Definitely, solution with higher fitness score has a higher probability to be selected. However, allowing too many low fitness solutions surviving in the population will result in slow converging rate. Therefore, it is very important to control and balance of good and poor-quality solutions in the population and this can be achieved by adjusting the parameters such as population size, rate of applying genetic operators or the distribution of selection and so on.

4.4. Probabilistic Model

As discussed in Chapter 2, zone design is a well-known combinatorial optimization problem and graph model is well suited to present the zonal data and its topological relationship naturally. Therefore, it is proposed to use Markov Networks to model the zonal data by a set of categorical variables $\{X_1, X_2 \cdots X_{n-1}, X_n\} \in X$ and its topological relationship. For each variable, it can be assigned with a value from a finite set $\{p_1, p_2 \cdots p_k\} \in P \; and \; k < n$ denoting which partition or region that respective zone is belonging to. As mentioned in Chapter 3 that a probabilistic graphical model comprises model structure and model parameters. Given a set of variables and observations, structure learning is the most difficult part in the modelling process because potential relationships among variables is vast and observations available may not be sufficient to find a correct relationship. Furthermore, structure of the zone design problem refers to the interconnection among zonal units and even if two distanced zonal units with no direct topological connection has indication of causal relationship when examine their attribute values, it cannot break the topological structure. Otherwise, a region may comprise scattered zonal units in the study area and it will never generate any valid and meaningful solution. Therefore, to ensure meaningful solution can be generated, and the fact that neighbour connectivity is inherent and explicit in the zonal data, structure learning becomes unnecessary in the zone design problem. Instead, topological structure of the zonal data is adopted directly. To avoid confusion, the zonal data that I am referring to is spatial data in polygonal form and the data has correct topological structure describing the basic spatial partitioning of a study area.

On the other hand, parameter estimation is straight forward and it can be achieved by applying the local property of Markov Networks. Recalling the local property, for each variable in X, it has conditional probability denoted as $P(x_i|N_i)$, where N_i denotes neighbours of x_i , where $x_i \notin N_i$ and there exists an adjacent relationship between x_i and all members in N_i forming a one-to-many relationship. To obtain conditional probability of each variable, it can be done by counting the number of instances for each possible assignment given fixed state of its neighbours in the population. Here, the fixed state means the assignment of its neighbours is fixed and is treated as a condition for yielding a probability of assignment for subject variable/node. For example, consider a zone design problem aiming to group underlying zonal units into 3 regions, the conditional probability of the variables is defined according to Bayes' Theorem as follows:

$$P(x_i|N_i) = \frac{P(x_i, N_i)}{P(N_i)} = \frac{P(x_i, N_i)}{\sum_x P(x_i, N_i)} \tag{50}$$

In (50), assuming $\mathrm{x_i} \ni \{1, 2, 3\}$, denoting the possible assignment to each region, the conditional probability for $\mathrm{x_i} = 1$ is given as follows:

$$P(x_i = 1|N_i) = \frac{P(x_i = 1, N_i)}{P(x_i = 1, N_i) + P(x_i = 2, N_i) + P(x_i = 3, N_i)} \tag{51}$$

Since all the joint probabilities are just counting from the population, it can easily be determined.

4.5. Sampling

To generate new population, I adopt a sampling method using a Markov Chain Monte Carlo (MCMC) process. Normally, a so-called “burn-in” period, which refers to the initial iterations, is required until the probability distribution becomes stable. It is considered that the minimum number of iteration “r” as the number of random selection with replacement so that each variable would be chosen at least once[2], and it can be defined as follows:

$$r = n \times \ln(n) \tag{52}$$

In (52), n denotes the length of the solution string. In the work of Shakya & Santana 2012b, r is further multiplied by an Iteration Coefficient to increases the number of iterations.

The conditional probability of particular variable or zonal unit can be derived from a simple joint probability as mentioned in Equation (50) or Gibbs distribution. The reason to use Gibbs distribution is that it can make use of a temperature coefficient, T to control the exploration rate to improve the opportunity escaping from local optima. T has a positive effect on exploring the search space (i.e. $T=1/(g\times CR)$). As discussed in Section 3.19, g is the current generation and CR is the cooling rate parameter, in which the larger the CR parameter of lower temperature, the quicker the convergence rate but less exploration. On the contrary, the lower the CR parameter of higher temperature,

[2] This refers to the coupon collector’s problem.

the slower the convergence rate but more exploration for higher quality solution. In our case, CR is set to 1.1.

```
1. Randomly select a solution candidate from the current population;
2. For r iterations, do
    a. Randomly select a variable $x_i$ from the given solution candidate;
    b. Retrieve the conditional probability $P(x_i|N_i)$ from the Markov network model based
       on the local Markov property derived from the joint probability (53) or Gibbs
       distribution (54);
```

$$P(x_i|N_i) = \frac{P(x_i, N_i)}{\sum_x P(x_i, N_i)} \tag{53}$$

$$P(x_i|N_i) = \frac{e^{P(x_i,N_i)/T}}{\sum_x e^{P(x_i,N_i)/T}} \tag{54}$$

```
    c. Sample $x_i$ using $P(x_i|N_i)$
3. Terminate with a new sample of solution candidate.
```

Algorithm 4-2 Sampling process in the EDA for zone design

The goal is to compute the conditional probability at each fixed state meaning the neighbours are assigned with same set of regions. Since the denominator becomes a constant, it can be ignored to simplify the computation.

Algorithm 4-2 shows the entire sampling process. Each process will return one new solution only and it can be repeated until sufficient number of new solutions is obtained. Nevertheless, in order to ensure proper and meaningful solution to be generated, additional measures are introduced to ensure valid solution for the zone design problem to be generated. These measures include relaxation of sampling probability, infeasible solution vector and detection of cut-node.

4.6. Relaxation of sampling probability

At each generation, the population being used in the algorithm is just a sampling of the potential solutions only at a relatively small number. Obviously, the sampling will not generates all the feasible configurations. In case particular configuration is not available during the sampling process, it will return an undefined or zero conditional probability. This situation will adversely affect the entire evolutionary process because the calculation will be terminated by the invalid probability value. To cope with this problem, a minimum percentage of particular configuration can be maintained consciously in the population by assuming that percentage is negligible for calculating the joint probabilities. Meanwhile, zero probability has to be avoided. To do this, in case particular configuration is not available in the sample population, calculation of the joint probability will be relaxed by ignoring the neighbours of the subject variable. For example, the following frequency table is node "82" and its neighbours that is identified in a sample population.

```
$`82`           Neighbours
        82      34      73      80      81      83      92        Freq    Prob
1       1       1       1       1       2       2       1         9       0.18
2       1       1       1       1       3       3       1         2       0.04
3       2       2       2       2       1       1       2         4       0.08
4       2       2       2       2       3       3       2         18      0.36
5       3       3       3       3       1       1       3         8       0.16
6       3       3       3       3       2       2       3         9       0.18
```

Figure 4-2 Frequency table of node "82" and its neighbours

As indicated in the table, nodes "34", "73", "80", "81", "83" and "92" are neighbours of node "82". Each row shows a configuration of possible region assignment, its frequency and occupying percentage in the sample population. Assuming a state of neighbours “1,1,1,2,2,1” is drawn, i.e. row number 1, it can be determined that it has 18% of chance to be appeared in the population. However, if state of the neighbours, say “1,1,1,3,2,1”, is drawn, it is undefined in the table and therefore corresponding probability cannot be determined. In this case, it can be replaced by the marginal probability of subject node, i.e. “82” or a minimum probability avoiding zero probability. For instance, suppose zero count for $(x_i = 1|N_i)$ is identified, it can be replaced by $P(x_i = 1)$ or a pre-defined minimum probability for further sampling.

In fact, Alden (2007) implemented similar approach to handle the problem. However, he suggested relaxing the conditional probability by nearby well-defined value assignment. In other words, neighbour would be ignored sequentially until a valid probability can be identified in the table. In the worst case, a marginal probability would be used, which is equivalent to the univariate EDA.

4.7. Feasible solution vector

In zone design, contiguity constraint must always be maintained although it is almost impossible to create a conforming solution by chance because the configuration drawn from the probabilistic model does not consider spatial topological structure. Therefore, without proper restriction, it is unavoidable that sampling will generate invalid solutions.

To ensure all the new solutions are valid regions, i.e. seamless without any holes, a special handling that restricts the region assignment for each variable is imposed in the sampling process. To realize this, a feasible solution vector that determines which region assignment value can be applied to the selected variable in a given solution during the sampling process is introduced. The feasible solution vector is a binary vector, in which each position represents a possible assignment value, i.e. “0” denotes corresponding region assignment value is infeasible to be used, while “1” denotes corresponding region assignment value is feasible to be used. For instance, a vector (1, 1, 1) means subject variable can be assigned to all the 3 possible regions to maintain conforming solution. The feasibility to assign which region to the selected variable is determined by regions already assigned to its neighbours in the selected solution. To illustrate the idea, consider 3 scenarios. Scenario 1 is shown in Figure 4-3, in which a variable is assigned to region 1 and has 6 neighbours in three different regions, i.e. 1, 2, or 3. In this case, a feasible solution vector can be formulated to indicate that the variable can be assigned with any regions, i.e. 1, 2, and 3 represented by a vector (1, 1, 1) in order to maintain conforming solution.

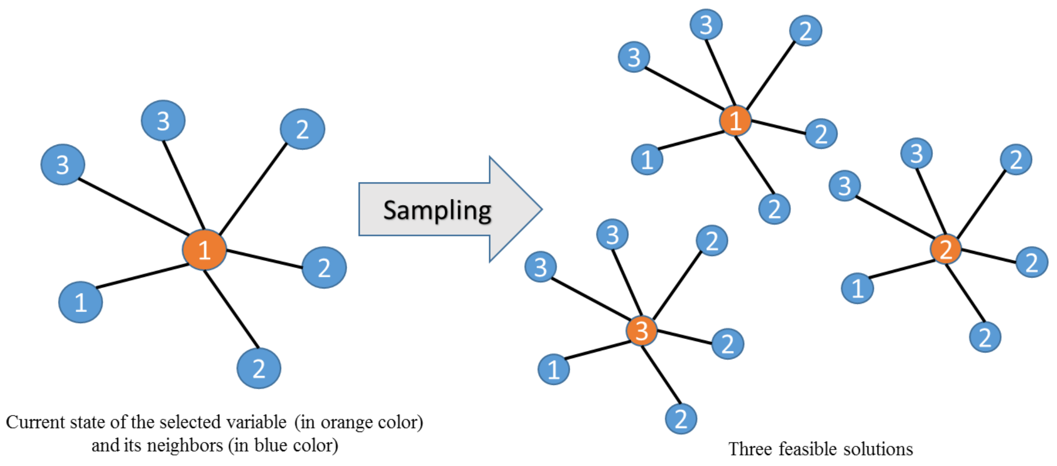

Sampling
Current state of the selected variable (in orange color) and its neighbors (in blue color)
Three feasible solutions

Figure 4-3 Scenario 1 shows sampling of variable and forming feasible solution vector (1, 1, 1)

In Scenario 2, the same variables and neighbours similar to Scenario 1 are presented. However, subject variable and all its neighbours are now in region 3. Therefore, if the variable is assigned to region other than 3, it will create a single-zonal unit region and also a hole for region 3 resulting an invalid solution. To avoid region assignment other than 3, a feasible solution vector (0, 0, 1) is created to force the region assignment to subject variable be region 3 only.

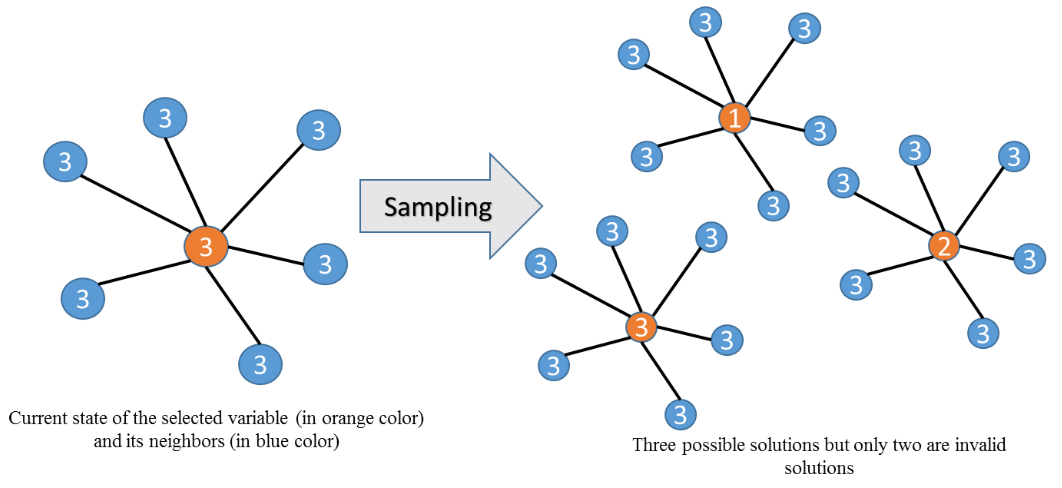

Sampling
Current state of the selected variable (in orange color) and its neighbors (in blue color)
Three possible solutions but only two are invalid solutions

Figure 4-4 Scenario 2 showing sampling of variable and forming feasible solution vector (0, 0, 1)

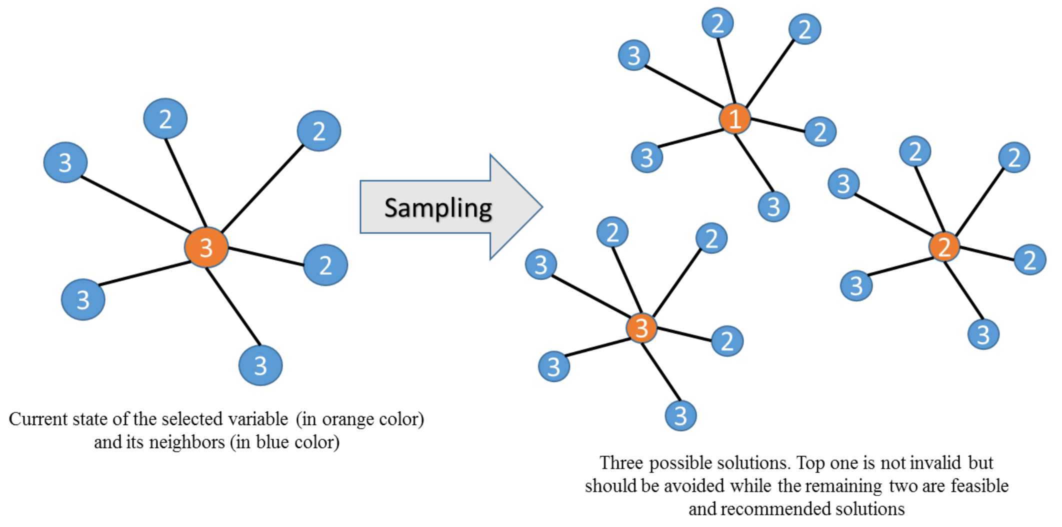

Sampling
Current state of the selected variable (in orange color) and its neighbors (in blue color)
Three possible solutions. Top one is not invalid but should be avoided while the remaining two are feasible and recommended solutions

Figure 4-5 Scenario 3 showing sampling of variable and forming feasible solution vector (0, 1, 1)

In Scenario 3, neighbours of subject variables are assigned to two different regions only, i.e. regions 2 and 3. In this circumstance, subject variable can be assigned with either one of the regions. Otherwise, a single-zonal unit region will be created similar to case 2. Therefore, a feasible solution vector (0 1 1) is generated.

This feasible solution vector is applied for each zonal unit when sampling. It operates just like a masking so that it will prohibit invalid region assignment. Because this will adjust the overall probability of each variable, probability of remaining possible value should be normalized. Of course, before the sampling, all the values have been normalized to form a correct probability, i.e. sum to 1. With the use of feasible solution vector, this ensures all the solutions generated by the sampling process conforming to the hard constraint of contiguity making the EDA based zone design keep running properly.

4.8. Detection of Cut-node

Cut-node refers to the node that will cut a region into two parts when the node is removed from the region. In the sampling process, nodes are randomly drawn from the graph for changing its value. As discussed in previous section, restriction stated in the feasible solution vector will be imposed in each move. However, even if feasible values are determined for the node, there is a chance that the selected node will cut an existing intact region and turn out making the entire solution becomes infeasible. A complete review on verification of graph contiguity can be found in King et al. (2014).

Ricca and Simeone 2008 has discussed a simple mechanism to maintain contiguity for districting problem. They have introduced the following two conditions to make sure feasible solution can be maintained after moving the nodes, in which it refers to change the value of subject zonal unit in the zone design problem. As such, the moving node must conform to the following two conditions:

i. must be a boundary node connecting to two different regions; and
ii. must not be a cut-node for the original region.

The first condition can be handled by the feasible solution vector. The second condition, can be realized by checking the reachability among neighbours of the moving node pretending the moving node is removed. This means if the neighbours of the moving node cannot be reached from each another without passing through the moving node, the moving node is identified as a cut-node.

```
1    Given an adjacency list and list of given nodes (neighbours of the moving node)
2    Mark all nodes as unvisited
3    Choose some starting node x
4    Mark x as visited
5    Create a list L
6    Push x to L
7    While L is not empty
     7.1    Select the first node in L
     7.2    Remove the first node from L
     7.3    Get a list of neighbours of the first node from the adjacency list
     7.4    For each neighbour
            7.4.1  If it is not visited and in the list of given nodes
            7.4.2  Mark the unvisited node as visited and push it into L
8    End while
9    If all given nodes are visited, subject moving node is not a cut node
10   If any given node are unvisited, subject moving node is a cut node
```

Algorithm 4-3 Steps to detect cut-node

In do this, identification of cut-node is based on adjacent list using Breath First Search (BFS) algorithm. To save the processing time, it will check against the relevant neighbouring nodes rather than the entire graph. Algorithm 4-3 shows the steps to detect cut-node.

Therefore, whenever a moving node is detected as a cut-node, it will be ignored until a non-cut-node is identified in order to run subsequent sampling process.

4.9. Replacement

Replacement is an optional step in EDA and it defines how many solutions will be replaced by solutions in the new generation. The idea of replacement aims to improve solution diversity. This can be realized by a niching technique so that it will try to reach more than one local optima in the population.

Restricted Tournament Replacement (RTR) (Harik 1995) is a well-known niching scheme in EDA. It is different from full replacement that replaces current population entirely with all new population or elite replacement that replaces selected worst solutions from the current population with the best solutions from the new population. Replacement using RTR is restricted to selected solutions only. RTR introduces an additional source of selection pressure that has empirically been proven with advantageous effects. Furthermore, RTR can preserve diversity and converge with less population size and number of evaluations (Lima et al 2007).

In RTR, new solutions will replace selected solutions in the current population according to similarity and fitness comparison. To be more precise, a subset of solutions with size w called replacement window is selected from the current population and the same number of new solutions is also created by sampling. Then, for each of the w new solutions, the closet solution is identified from the replacement window in terms of genotypic distance. Genotypic distance refers to the difference between two encoded solutions and therefore this is a measurement within the search space. In this study, genotypic distance of two candidates can be defined by their difference of regional assignment.

If the new solution has higher fitness score than the nearest solution, replacement is performed in the current population. Otherwise, the new solution is discarded and the process moves to the next one until all the new solutions are exhausted. The genotypic distance is defined by the number of difference of value at each string position. Steps in the RTR can be found in Algorithm 4-4.

```
1. Define a replacement window w;
2. Select w solutions from the current population and from the new population respectively;
3. For each new solution in the replacement window of the current population, do
    a. Identify the most similar new and current solution pair in term of genotypic distance;
    b. If the new solution has higher fitness score, replace corresponding current solution in
       the current population; Otherwise, discard the new solution;
4. A new population is created.
```

Algorithm 4-4 Restricted Tournament Replacement

As mentioned above, the idea of RTR is to discourage convergence to single optima. This will be good for finding Pareto front in multiple criteria optimization.

4.10.Computational Time Complexity

Computational time complexity of the EDA based zone design is mainly governed by the parameter learning procedure and the sampling process. Regarding the parameter learning, retrieving a list of neighbours for each variable is required. Then corresponding instances are retrieved from the current population to compute the conditional probability of the variable for each particular value given its neighbours at fixed values. This leads to a computational complexity of O(number of variables*number of neighbours*number of zones). Regarding the sampling process, each randomly chosen solution is required to run r iterations for sampling a single solution. To complete a population of size popSize, the computational complexity of O(r*popSize) is required.

4.11.Conclusion

In this chapter, all the essential components required to implement the EDA based zone design algorithm have been discussed in detail. The algorithm is using EDA and Markov Networks as foundation for processing and modelling the zonal data. As zonal data itself has incorporated a topological structure, structure learning usually advocated in probabilistic graphical model is no longer required. Instead, the topographical structure is adopted as the structure of the probabilistic model. This chapter has gone through each process to run the EDA. In population initialization, it is recommended to use random assignment of cluster centres to generate the population for the first generation. Next, a model learning process is straight forward that relies on counting number instance of each node and its neighbours having fixed states in the population. Consequently, to generate new solutions in the sampling process, Gibbs sampling technique is adopted. Because not all combinations can be found in the population, relaxation of the conditional probability to marginal probability is considered necessary to ensure meaningful regions can be generated. Besides, various measures such as feasible solution vector and cut-node detection are introduced to ensure new solution will always conform to the contiguity constraint. Therefore, the entire process can obtain valid solution based on the statistical learning from the population. Because the searching mechanism is based on the local property of Markov network, it can provide a global search rather than incremental local improvement. Therefore, this approach is expected to provide better performance and effectiveness when compared with other approaches such as traditional heuristics for solving zone design problem.

Chapter 5 – Experiments and Results

5.1. Introduction

This chapter is devoted to making the zone design problem be solvable by the EDA based Zone Deign introduced in Chapter 4. Since zone design is working in discrete domain, discussion focuses on discrete representation only. To see how the EDA-based Zone Design works and its output regions, a test data is developed based on Tertiary Planning Unit (TPU) and the Large Street Block data of the Hong Kong census data 2011. Two scenarios are developed to demonstrate the capability of EDA based Zone Design algorithm. First scenario will find regions of intra-homogeneity population or equal population. The results are compared with those found by a GA based Zone Design implementation. Second scenario finds inter-homogenous regions or so-called clusters from the test data. The results are compared with those generated by SKATER (Assunção et al. 2006), which is a regionalization algorithm aiming to identify homogenous regions based on single linkage clustering procedure. Test results of these two scenarios show that the EDA based Zone Design outperforms both the GA implementation and SKATER.

5.2. Encoding Methods

As discussed in Section 2.9, graph is the most appropriate representation for modelling basic zonal units. More importantly, it has flexibility to form all possible zone design solutions respecting to the topological structure. Meanwhile, Rook Contiguity is adopted to define whether two adjacent basic zonal units have neighbouring relationship that ensures valid regions can always be generated.

In order to solve the zone design problem using EDA, an encoding method is required. Encoding method converts solutions from the solution space to the search space so that it can be evolved facilitating creation of new and favourable encoded solutions. Because the encoding method will influence the efficiency of solution searching process, it is crucial to design or select an appropriate encoding method. Although the graph model is good for modelling zonal units, a data structure is required to provide an easy way presenting the zone design solution and refining value of particular zonal unit at the same time.

To do this and as discussed in Section 4.2, a potential solution i can be represented by a character string $X^i = (x_1^i, x_2^i \dots x_n^i)$ where x_k^i denotes a variable representing zonal unit k of the solution i. Each variable is assigned with a value indicating which partition or region it is belonging to. This is the most natural and straight forward approach to encode zoning or graph partitioning solutions. This is a direct encoding because decoding is not necessary to retrieve the actual solution. Since each zonal unit represents a fragment of a region, collection of zonal units having the same regional assignment can be retrieved directly to form a complete region without complicate computation. This encoding method is also referred as group-number encoding or vertex-to-cluster encoding in the literature. However, this encoding method just records which value is assigned to which zonal unit only. To maintain the topological relationship, a neighbour list is used. Neighbour list is generated by the graph model and it can be used to retrieve list of neighbours given a subject zonal unit. Therefore, group-number encoding and neighbour list are used together to fully encapsulate a graph model so that it can be operating in the EDA paradigm.

Nevertheless, Kim et al 2011 and Rothlauf 2006 has shown that the group-number encoding method has the problem of redundancy. In brief, it is possible to have more than one encoding representing the same solution. For example, for a 5-bit bipartition problem, a solution could be encoded as “01110”, which is identical to another solution encoded as “10001”. Because for each string position, it can be either “1” or “0” for a bipartition problem. Although the size of solution space is 2^5 (i.e. 32), the actual number of distinct solutions should be 16 only. Of course, when the number of possible values for each position increases, solution space would also increase exponentially. Rothlauf 2006 has provided a detailed study on encoding method indicating that redundancy undermines performance of evolutionary algorithm. This means that some solutions appear to be different in the search space are actually the same in the solution space. To address this problem, maintaining good diversity of solution may help and that could be realized by different measures such as the RTR and the Gibbs sampling with temperature parameter as discussed in Chapter 4. Nevertheless, group-number encoding is a primary encoding method being adopted in the EDA-zone design algorithm.

Moreover, I also adopt a simpler encoding method as proposed by Bação et al. 2005. Besides, Correa et al. (2001) also used the same encoding to solve a p-median problem. This encoding method abstracts regions by its centre only. For example, suppose there is a zone design problem requiring to aggregate n zonal units to 3 regions, a solution can be presented as a character string $X = (x_1, x_2, x_3)$, in which each variable x_i can be assigned to a zonal unit representing centre of a region. Due to lesser number of variables, this encoding method is much efficient and faster for computation. Because it is not a direct encoding, it must be decoded to obtain an actual solution for fitness evaluation. Decoding can be achieved by associating remaining zonal units to the nearest regional centre based on Euclidean distance. This is the same when performing k-mean clustering without iteration. The major benefit of this approach is that the regions created are in general contiguous. However, it is highly sensitive to the position of each zonal unit, in which it usually refers to the centroid.

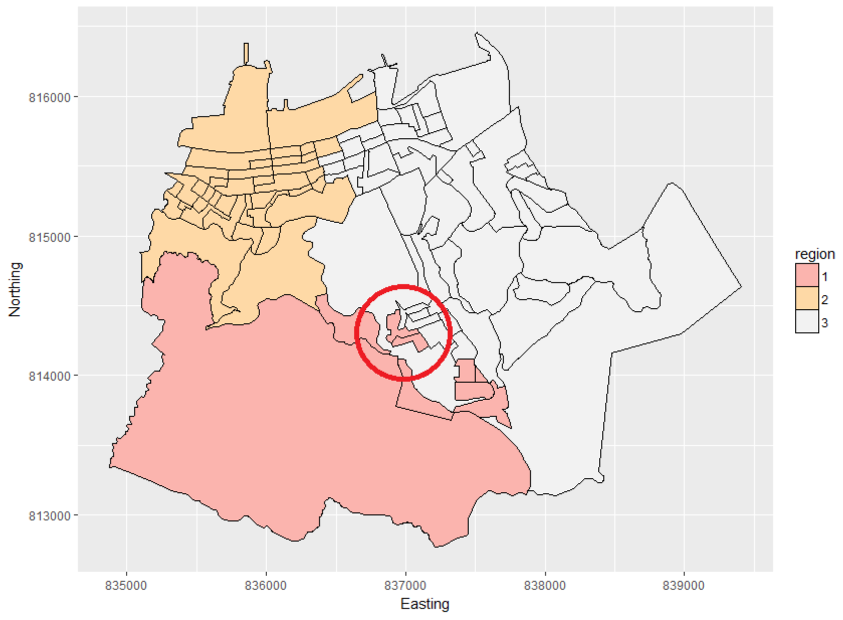


Figure 5-1 Example of Parts of Region (isolated part in red circle) created by the Regional-centre encoding approach

When the zonal data is composed of both small and big zonal units meaning the size of zonal units is uneven, it can generate regions with isolated parts. Figure 5 1illustrates a case of regions with isolated parts. As shown in the figure, isolated parts of region 1 is highlighted by red circle. By observing the adjacent zonal units, it is obvious that the very large zonal unit next to the isolated part creates the problem. Regional centre encoding is another method being used for testing the algorithm.

5.3. Objective Functions

An objective function is primarily used for evaluating the fitness of solution, a typical application is to find optimal values of a mathematical function. However, it can be used to identify characteristic of underlying data if such characteristic can be formulated as an equation and this can be applied to zone design. In this connection, characteristics that I consider for shaping the zone design are as follows:

(a) Equality or Homogeneity;
(b) Compactness; and
(c) Lower limits for selected attributes of the region.

It is believed that these are the most common characteristics and sufficient for demonstrating how the EDA-based Zone Design algorithm works. Besides, most of the past studies are using the similar set of criteria for conducting zone design and regionalization (Altman, 1997; Bação et al. 2005; Assunção et al. 2006). Regarding equality or inter-regional homogeneity, it is one of the essential conditions for census districting. It can be realized by the sum of squared difference between each zonal unit and overall average population in the study area. Measure of equality is minimized so that variation among regions can be minimized as well to achieve the goal of equal population. In contrast to the concept of inter-regional homogeneity, intra-regional homogeneity demands zones having similar quality within the same region and it can be realized by the sum of squared difference between each zonal unit and within-region average. Likewise, this measure should be minimized. Regarding the conditions for compactness and minimum number of zones region, these are aesthetic measures that govern the shape of the regions/partitions. In normal situation, circular shape is preferable because it is much compact and good for internal accessibility. Details of the objective functions for equality, homogeneity, and compactness have been discussed in Chapter 2.

Regarding the lower limit of concerned attributes within a region, it is a hard constraint to avoid generating unbalanced regions. The limit can be determined from the given zonal data. Normally, it is better to set the lower limit greater than the minimum value of the concerned attribute. If solution fails to meet this hard constraint, fitness value will be set to “NA” meaning the solution will then be discarded. Because there is more than one objective function, an overall fitness is defined by their simple summation. It is considered that all the criteria are of equal importance and therefore no weighting is defined.

5.4. Test Data

The test data is comprised of two components. The first component is the Tertiary Planning Units (TPU) and Street Blocks of Hong Kong released in 2011, in which Street Blocks give the definition of basic zonal units forming TPUs by aggregation. Meanwhile, the TPUs serve as reference showing how it can be formed by the Street Blocks and these will be used for comparison with EDA-based Zone Design later in this Chapter. Because the TPU data set does not contain any useful attributes for zone design leading to the second component for the provision of useful attributes facilitating analysis. The second component refers to the census data 2011, which provides different theme of socio-economic related data. In this study, population data is used. Owing to these are two independent datasets, data integration work is necessary to combine them as a single data for analysis. In the following sections, I will give a brief description of the two datasets and relevant integration works to run zone design.

5.5. Tertiary Planning Units (TPU)

According to the metadata of the TPU data set (LandsD 2016), TPU boundaries are mainly delineated by the nature of the geographic features of the area such as roads, railway lines, coastlines, contours, waterways, lot boundaries or zoning boundaries of town plans. The TPU data set contains boundary of both TPUs and Street Blocks, in which each TPU can be formed by aggregation of respective Street Blocks. The primary purpose of TPU is for town planning. Due to administrative convenience, TPUs and Street Blocks are used as the common reference for compilation of statistical data such as census/by-census.

Regarding the organization of the data, the whole territory of Hong Kong is divided into planning units at different levels in a hierarchical structure. Hierarchy of the town planning units is comprised of Street Block, TPU, Secondary Planning Unit (SPU), and Primary Planning Unit (PPU). Street Blocks are the smallest units while PPU is the largest units in term of size. Each level has a nested relationship with its immediate higher level. Each TPU can be identified by a unique three-digit number. The first digit indicates which PPU the TPU belonging to while the first and second digits together indicates corresponding SPU. For the identification of street block, additional digits will be added after the TPU number delimited by a “/” symbol, e.g. 131/17. The following table shows the number of records found at different levels in the dataset released in 2011:

Item	Planning Units	No. of Records
1.	Street Block	2,627
2.	Tertiary Planning Unit (TPU)	289
3.	Secondary Planning Unit (SPU)	52
4.	Primary Planning Unit (PPU)	9

Table 5-1 No. of Records at different levels of Planning Units in the 2011 TPU data set

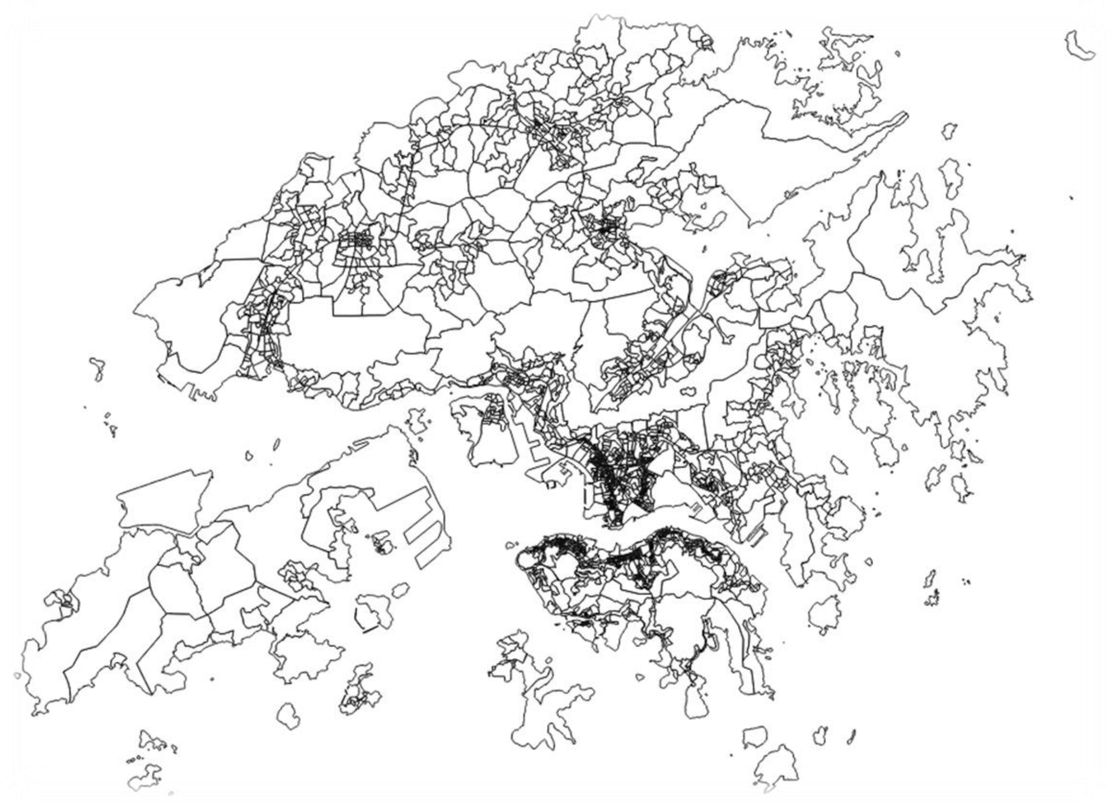


Figure 5-2 Map showing 2,627 Street Blocks released in 2011

Figures 5-2 and 5-3 show the appearance of Street Blocks and TPUs respectively. As shown in the figures, it seems that the size of Street Blocks is relatively small in urban and major built-up area, while it becomes relatively bigger in rural or more precisely hilly areas. This is expected because Street Blocks are basically delineated by roads and streets, which are the indication of level of development. Therefore, areas at North side of Hong Kong Island, Kowloon peninsula and some new towns in the New Territories are having high density of Street Blocks. When looking at the TPU boundaries, it is likely that population is one of the considerations for defining it from the underlying Street Blocks.

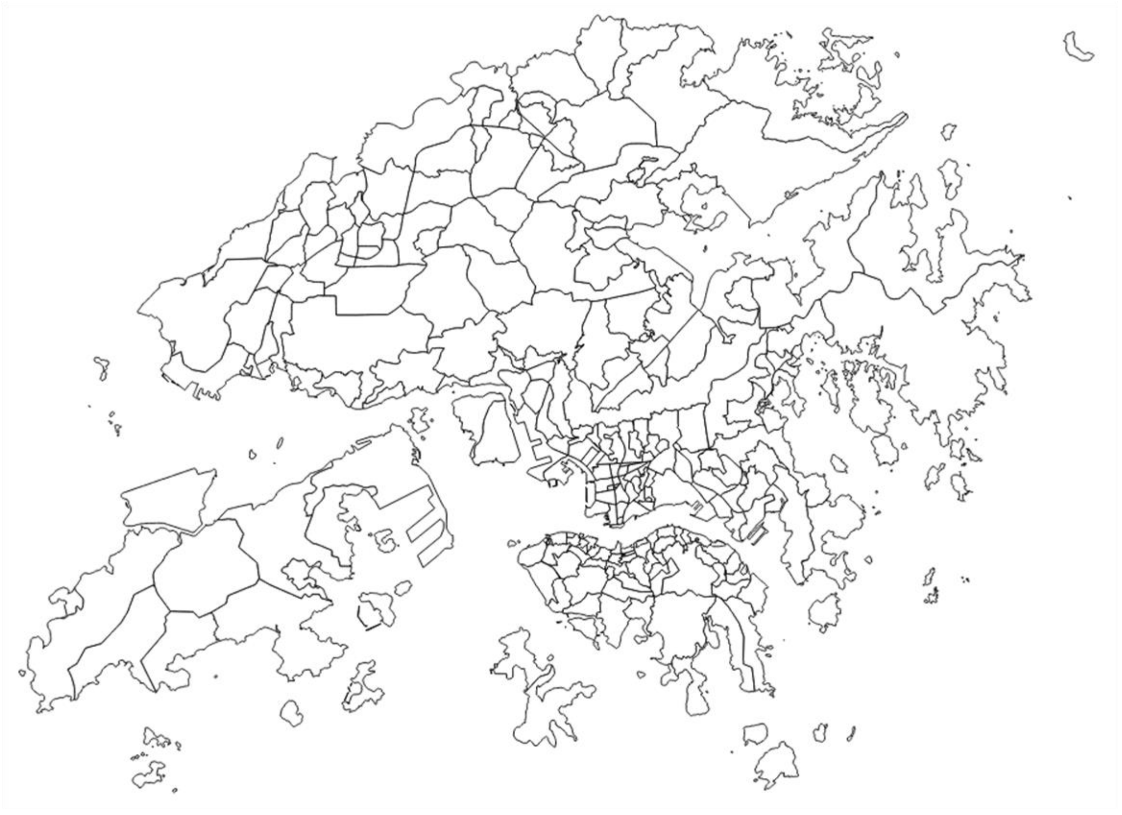


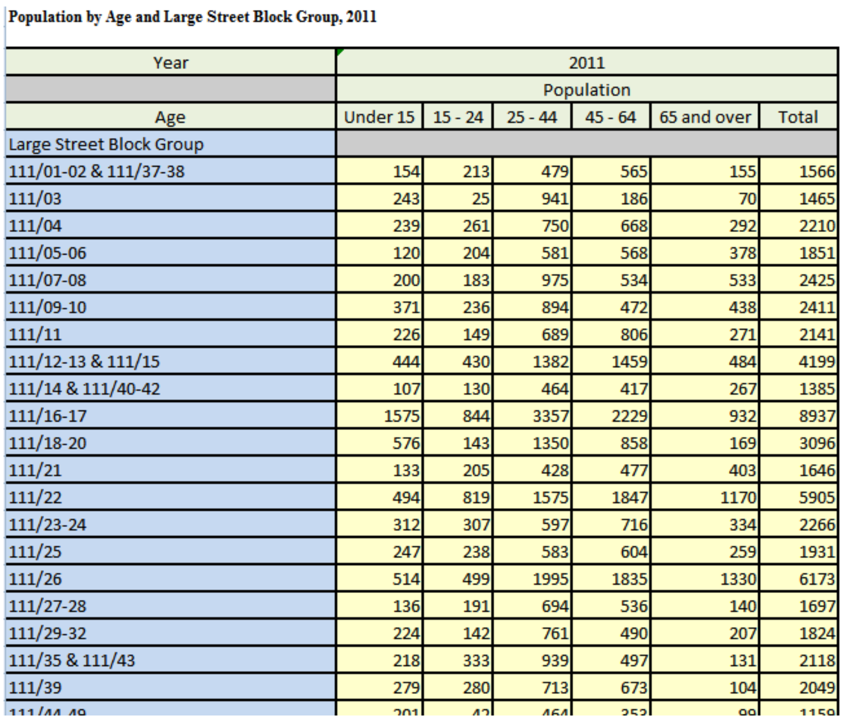
Figure 5-3 289 TPUs released in 2011

5.6. Census Data

Census data of Hong Kong is available in the web site at http://www.censtatd.gov.hk/. To a certain extent, the census data can be associated to the town planning units at the level of TPUs and Street Blocks. Of course, data that can associate to Street Blocks is preferable for analysis but this is solely depending on its availability. In this study, the most detailed data that can be identified in census 2011 through their webpage is called Large Street Block Group, a level in between Street Block and TPU. After comparing the data from the Large Street Blocks and Street Blocks, it revealed that a record in the Large Street Blocks Group can be related to multiple Street Blocks. This indicates that the census data has been generalized intentionally and is not at its original level of details. Because those affected areas are close to or on the hillside having less population, it is believed that relevant generalization by combining Street Blocks is for administrative purpose for conducting census. Although this is not desirable for data analysis, it is the most available data in the public domain.

As shown in Figure 5-4, to associate the census data, Street Blocks are combined according to definition of the Large Street Blocks Group. Therefore, size of basic zonal units for zone design becomes larger while number of basic zonal units available for forming immediate higher level zonal units becomes smaller.

This situation is common in census data, i.e. reduces data size and preserves confidentiality. Unfortunately, this also induces the Modifiable Area Unit Problem (MAUP) and makes it hard for further data analysis. In brief, MAUP reveals a phenomenon when dealing with aggregating data, statistical property of associated data such as correlation of attributes, will become non-stationary at different scales and configurations. However, this is out of our scope and interested reader can find the details of MAUP from Openshaw (1983).


Population by Age and Large Street Block Group, 2011

Year	2011					
	Population					
Age	Under 15	15 - 24	25 - 44	45 - 64	65 and over	Total
Large Street Block Group						
111/01-02 & 111/37-38	154	213	479	565	155	1566
111/03	243	25	941	186	70	1465
111/04	239	261	750	668	292	2210
111/05-06	120	204	581	568	378	1851
111/07-08	200	183	975	534	533	2425
111/09-10	371	236	894	472	438	2411
111/11	226	149	689	806	271	2141
111/12-13 & 111/15	444	430	1382	1459	484	4199
111/14 & 111/40-42	107	130	464	417	267	1385
111/16-17	1575	844	3357	2229	932	8937
111/18-20	576	143	1350	858	169	3096
111/21	133	205	428	477	403	1646
111/22	494	819	1575	1847	1170	5905
111/23-24	312	307	597	716	334	2266
111/25	247	238	583	604	259	1931
111/26	514	499	1995	1835	1330	6173
111/27-28	136	191	694	536	140	1697
111/29-32	224	142	761	490	207	1824
111/35 & 111/43	218	333	939	497	131	2118
111/39	279	280	713	673	104	2049

Figure 5-4 Part of the population data at Street Block level (Large Street Block Group) from Population Census 2011 (See http://www.census2011.gov.hk/)

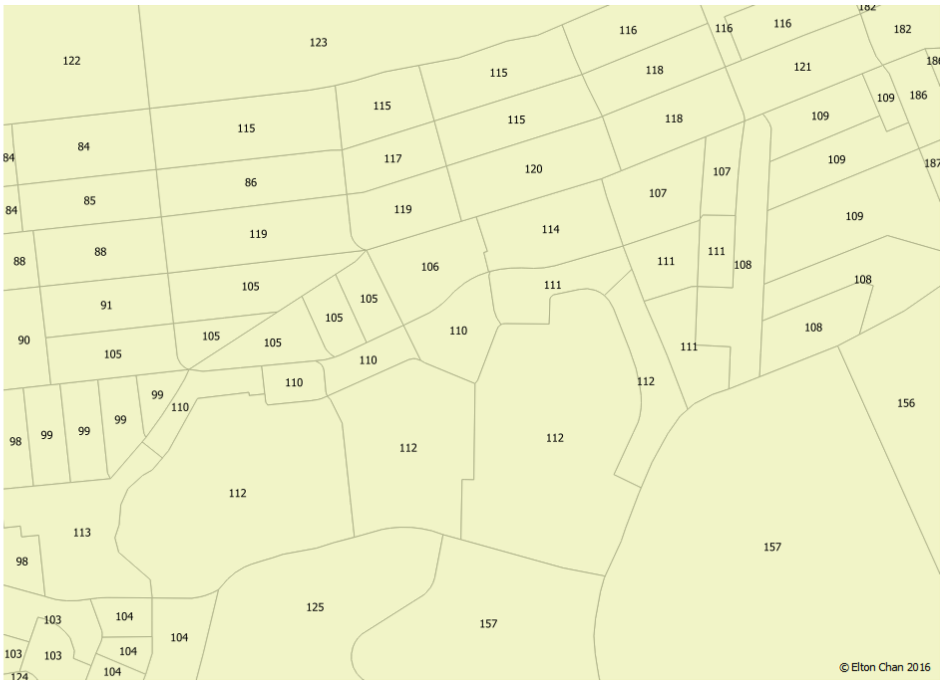


Figure 5-5 Street Blocks labelled with the same number are in the same Large Street Blocks Group according to the Population Census 2011

5.7. Wan Chai District

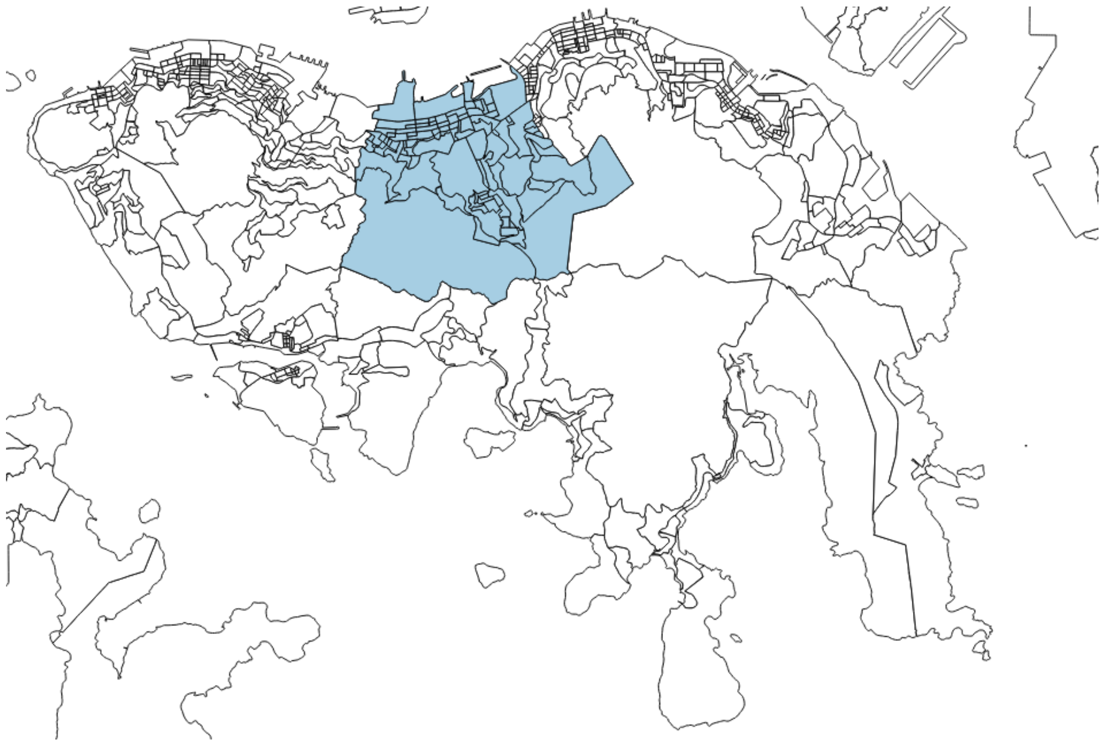


Figure 5-6 Overview of Hong Kong Island and the Wan Chai District (Area shaded in blue colour). The boundary represents aggregated Street Blocks due to Large Street Block Group

In this study, Street Blocks within Wan Chai District is used as test data. Wan Chai District is one of the 18 administrative districts of Hong Kong. According to current practice, demarcation of the districts, and daily administration matters are under purview of Home and Affairs Department (HAD) of the HKSAR Government. Figure 5-5 shows the location of Wan Chai District. Geographically, it is located at the centre of the Hong Kong Island.

As shown in Figure 5-6, the configuration of Wan Chai district is composed of street blocks and TPUs with irregular shape and various size. Some regular and small-size

Street Blocks can still be found at the North because these areas are mainly covered by reclaimed lands. On the other hand, irregular Street Blocks with larger covering area can be found at the South. These areas are usually near or at hillside and may be within the country park. This implies that population in areas at South is relatively less than that at North. This is a typical example of spatial partitioning in Hong Kong. However, when the size of Street Blocks becomes larger, it is a challenge to find a good solution for forming regions with good compactness. The reason is due to less flexibility for combining and re-combining the shape from the basic zonal units. Descriptive statistics of population, area and perimeter of the zonal units within the study are shown in Table 5-2.

According to the statistic, area of the zonal units varies significantly within the study area. Although there is variation of population as well, it is relatively less significant. However, when the population density is considered, it reveals that areas with the highest population density are located near the coastline and keep decreasing when moving to the hilly areas at the southern part of the Hong Kong Island. Typically, such hilly areas have relatively poor accessibility.

	Population	Area (sq.m)	Perimeter (m)
Minimum	1,004	3,941	322
Maximum	7,464	3,367,366	10,677
Range	6,460	3,363,425	10,355
Sum	165,803	10,066,664	119,350
Median	1,612	22,839	771
Mean	1,802	107,092	1,270
Variance	905,275	143,301,722,071	2,389,081
Standard Deviation	951	378,552	1,546

Table 5-2 Descriptive statistics of population, area and perimeter of street blocks within the study area

This unbalance configuration creates a great challenge to find the best partition meeting the criteria of equality and compactness.

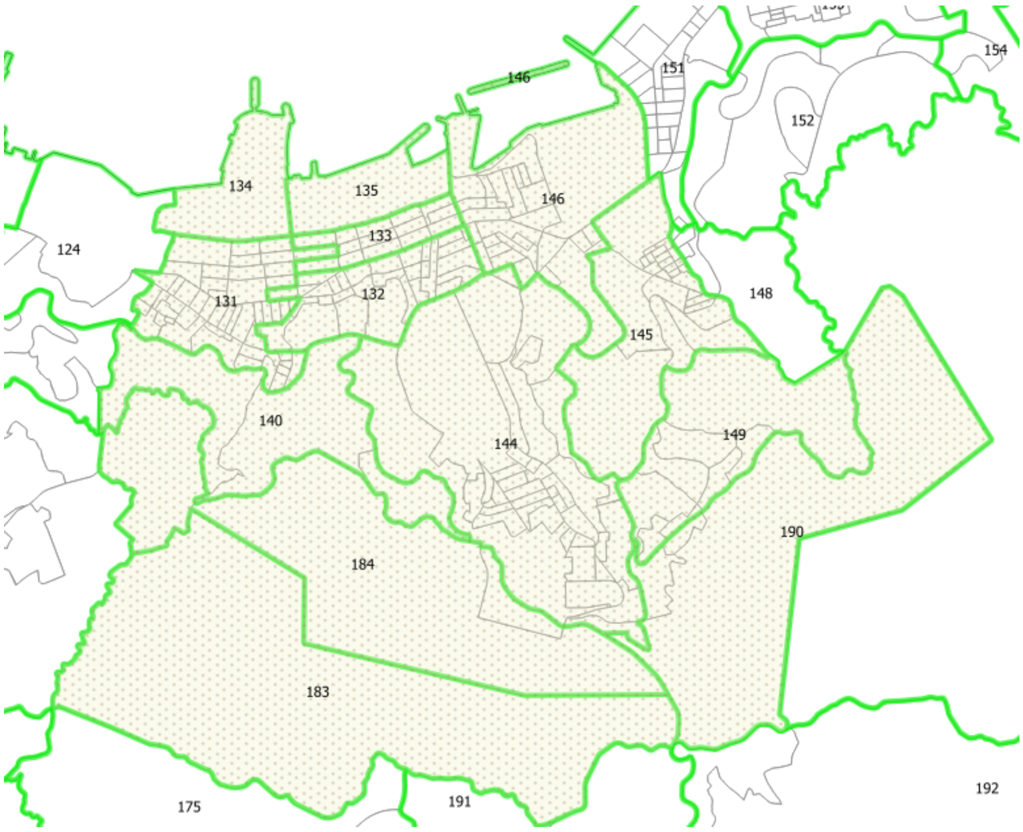


Figure 5-7 Existing official TPUs (outlined in green colour) and Street Blocks (outlined in grey colour)

Zone design is performed by aggregating the Street Blocks at the level of Large Street Block Group in accordance with criteria of equal population and compactness in this study. Of course, the zone design solution should also respect the constraints discussed in the previous chapter including contiguity and expected number of regions. Otherwise, it would not be recognized as valid solution.

According to the information provided by the Census & Statistic Department (C&SD) available from their Census 2011 web site, Wan Chai District is comprised of TPUs with identifiers 131, 132, 133, 134, 135, 140, 144, 145, 146, 149, 183, 184 and 190. In other words, Wan Chai District is composed of 13 TPUs or 204 Street Blocks. Since the population census data is organized by Large Street Blocks Group rather than Street Blocks, zonal data conforming to Large Street Block Group should be created by merging respective Street Blocks. The result is a set of Large Street Blocks that has lesser number of zonal units and different data organization when compared with the original Street Blocks.

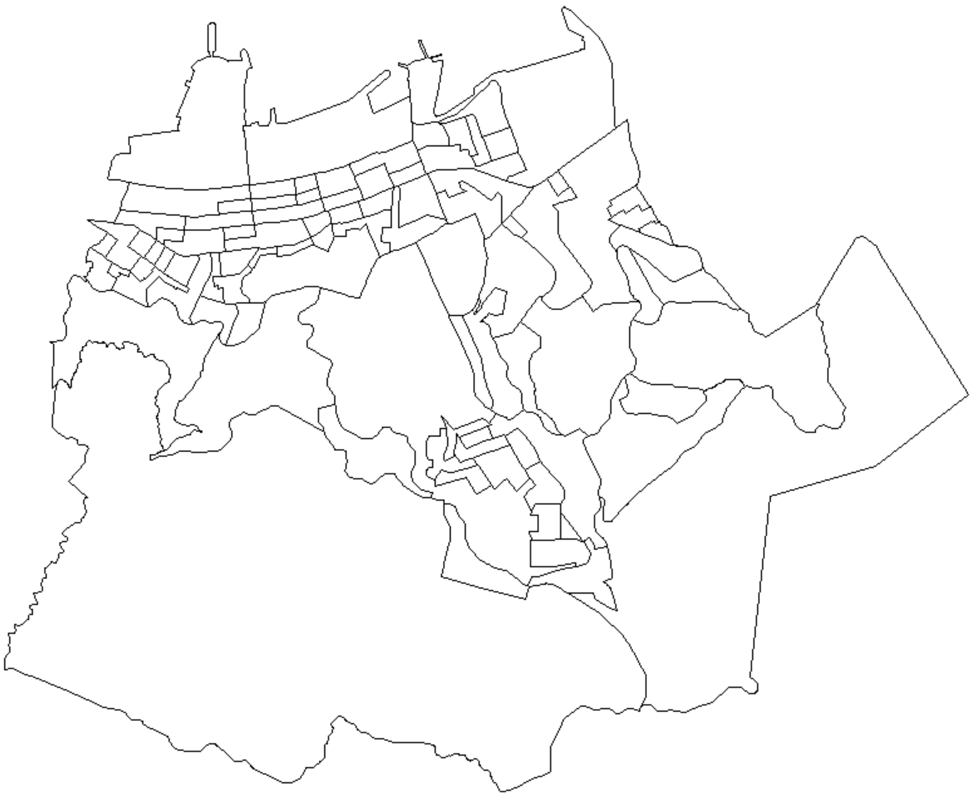


Figure 5-8 Zonal units of the test data

After merging, the number of basic zonal units within the study area is reduced to 94 from 204, which greatly reduces the size of viable solutions. However, the problem size or number of possible solutions is still extremely large as reference to Equation (7) in

Chapter 2. Same case when considering the search space. Regarding the group-number encoding, the number of possible encoded solutions is〖 r〗^n, in which r and n represent number of expected regions and number of zonal units respectively. Regarding the regional centre encoding, the number of possible encoded solutions is〖 n〗^r. Of course, this estimation disregards the constraint of contiguity. Another characteristic of the Large Street Blocks is that size of zonal units becomes more uneven. Large-size zonal units are surrounded by many small-size zonal units and this limits the flexibility during aggregation generating additional challenge fitting to the objective functions. It is considered that the Large Street Blocks complied in this study have certain representative level because it reflects real situation and characteristics in the reality. Overview of the basic zonal units of the test data in this study is shown in Figure 5 8.

5.8. Experiment Design

The experiment is to find the optimal partitioning of TPUs within the Wan Chai District according to the street blocks respecting the Large Street Blocks Group given in the Population Census 2011. This is a typical zone design or districting problem. The objective is to compare the EDA-based Zone Design algorithm with alterative algorithms and investigate if it can provide better solutions.

Three alterative algorithms are developed for the experiment. This includes a GA and two EDAs with different encoding methods. Regarding the GA implementation, it is made reference to the technique reported in Bação et al. (2005). Same as a typical GA, encoding method, population initialization method, and genetic operators for evolving solutions should be properly defined. Regional centre encoding as discussed in Section 5.2 is applied used, in which a zonal unit identifier is assigned to each position of a string vector indicating the centre of the to-be-generated region. The advantage of this encoding is its simplicity because no significant effort is required for maintaining the contiguity when generating region.

According to GA, several steps should be defined including population initialization, genetic operators, i.e. selection, crossover and mutation, and elitism technique.

Regarding the population initialization method, it can be done by sampling the existing zonal unit with a uniform probability. This means each zonal unit has the same chance to be a centre of the output region, but this is only limited to the initialization stage. Subsequently, on-going configuration will be guided by probability of regional assignment for each individual zonal unit according to fitness of solutions in each generation.

Regarding the genetic operators, three types of operator, namely selection operator, crossover operator and mutation operator are used. Tournament selection with 3 candidates is adopted for selection of high quality solutions. Besides, uniform crossover is used as crossover operator, in which each position can have the equal probability to be exchanged between two parent solutions. The rate of crossover and mutation are set to 0.8 and 0.2 respectively, which are determined by empirical test indicating best fitness can be achieved. Finally, elitism is also applied to make sure the best solution can be retained in the next generation of population. These steps are repeated until termination criteria are met, i.e. number of generations = 10. Steps of GA implementation in this study are shown in Algorithm 5-1.

```
1.   Generate initial population using random sampling of sufficient number of regions based on
     uniform distribution;
2.   Repeat while termination criteria is not met;
2.1.   Fitness evaluation for each candidate in the current generation. This requires converting the
       candidates from the search space to the solution space; Penalty will be given to candidates
       that cannot generate feasible solution; (i.e. contiguity)
2.2.   Tournament selection with 3 candidates are used to select candidates of good quality.
2.3.   Perform uniform crossover of 2 randomly selected candidates at rate 0.8;
2.4.   Perform mutation at rate 0. 1;
2.5.   Apply elitism to select the best fitness candidates and keep them in the next generation.
3.   End while.
```

Algorithm 5-1 Genetic Algorithm for Zone Design

Regarding details of the EDA based zone design, please refer to Chapter 4. Two different encoding methods are used in the experiment. The first EDA based zone design adopts the group-number encoding method. Characteristic of this encoding method is that it models the zone design solution directly without conversion between the search space and the solution space. Meanwhile, it allows manipulation at the level of zonal units. This means the evolution process can add or remove particular units from the region. On the other hand, a probability model is built or learnt by selected candidates of high quality, which are used to generate new population using sampling technique rather than genetic operators in GA. Although the sampling process will create new candidates at each generation, to ensure all newly created candidates are conforming to maintain contiguity, additional spatial data handling and checking procedures are necessary.

The second EDA based zone design adopts the regional centre encoding method same as the GA implementation. Marginal probability of each zonal assignment is learnt from the population having high fitness value. Again, stopping criteria is defined by number of generations.

Therefore, in the experiment, both regional centre method and group-number method are used in the EDA implementation while GA uses regional centre method only. The reason not to implement group-number method in GA because this requires a new set of crossover and mutation operators to proper evolve the solutions. Such operators should be capable of evolving solution at the regional level meaning it should be realized by a region-to-region crossover but not the zonal unit level operations as implemented in the EDA with the two encoding methods and even the regional centre method in GA. Example of regional level crossover and mutation can be found in Tavares-Pereira (2007), in which graph-based crossover and mutation operators was developed to evolve solutions. However, in this study, ordinary crossover and mutation operators can be used directly. Therefore, to keep the simplicity in implementation, only regional centre method is considered in GA.

Furthermore, all the computer program and development in this study are based on the GA package in R (Scrucca 2013). R is a popular computer language for statistical computing in recent years (R 2016a), mainly for data mining, statistical analysis and machine learning. Although R is easy to program, performance of some functions can be very slow when compared with other programming languages. In order to improve the computational performance, selected functions that require intensive computation such as the learning and sampling functions are written in C++ with Rcpp (Eddelbuettel & Francois 2011; Eddelbuettel 2013). In general, Rcpp provides R functions as well as C++ classes which offer a seamless integration of R and C++ (R 2016c).

5.9. Operations and Parameters

To allow correct comparison among the optimization methods, this section will describe common operations and parameters whenever applicable among them. Key parameters and settings of the optimization methods are as follows:

Parameters/Methods	**GA**	**EDA1**	**EDA2**
Population Size:	1,000	1,000	1,000
Encoding method	A list of unordered zonal unit identifiers, each represent a centre of output region.	A list of ordered region assignment to each zonal unit	A list of unordered zonal unit identifiers, each represent a centre of output region.
Population method	Random sampling of zonal units at regional centre and create clusters by finding closet zonal units.	Random sampling of zonal units at regional centre and create clusters by finding closet zonal units.	Random sampling of zonal units at regional centre and create clusters by finding closet zonal units.
Selection method	Tournament Selection	Tournament Selection	Tournament Selection
Crossover method	Uniform crossover	N/A	N/A
Mutation method	Simple mutation at rate 0.2 by random assignment of selected position of a solution string selected by random.	Simple mutation at rate 0.2 by random assignment of selected position of a solution string selected by random.	Simple mutation at rate 0.2 by random assignment of selected position of a solution string selected by random.
Learning method	N/A	- Model structure adopted from the spatial data - Conditional probability of each variable given its neighbours is learnt from the solution	- Model structure adopted from the spatial data - Marginal probability of each variable is learnt.

Parameters/Methods	**GA**	**EDA1**	**EDA2**
		population	
Sampling method	N/A	Gibbs Sampler	Simple sampling
Replacement method	N/A	N/A	N/A
Stopping Criteria	Number of iteration reached, i.e. 100 generations.	Number of iteration reached, i.e. 100 generations.	Number of iteration reached, i.e. 100 generations.

Table 5-3 Key parameters applied for running GA and EDAs

5.10.SKATER

SKATER stands for Spatial 'K'luster Analysis by Tree Edge Removal. As revealed by the name, it creates cluster by removing edge from a tree. To do this, SKATER firstly converts a zonal data into a graph model. Based on selected attributes of interest and the graph model, it generates a minimum spanning tree based on similarity. This means only edges having least similarity will be retained, in which similarity is defined by the difference of attribute values between the connecting nodes. With the minimum spanning tree, it will then identify k-1 edges that can maximize its objective function and remove them in order to create k clusters.

In this process, the steps to create the minimum spanning tree is very crucial because it governs how the spanning tree is formed and also subsequent optimization for edge removal. However, this is problem specific solely for finding homogenous clusters. Using the same approach, it cannot solve the equal districting problem by re-defining the objective function. This is because it is running on the minimum spanning tree, which is determinative or even pre-emptive to find least difference cut that does not help to find conditions such as inter-regional homogeneous or equal regions not necessary related to minimum difference. Algorithm 5-2 shows the major steps in SKATER. In the original design, heuristic is proposed for faster tree partitioning. However, it is skipped to find appropriate edges by exhaustive search. This ensures the best solution can be identified.

```
1. Given a graph representing a study area;
2. Build a minimum spanning tree based on selected attributes so that only edges with least
   dissimilarity will be retained;
3. Cut the tree into two parts by maximizing the Sum of Square Difference (SSD) before and
   after the cut, i.e. fitness = $SDD_{before} - (SSD_{part1} + SSD_{part2})$
4. Add the parts to a list L;
5. While parts<expected number of regions
     a. Identify the part and edge to cut with maximum fitness;
     b. Add the parts to list L;
6. End while.
```

Algorithm 5-2 Steps in SKATER

On the other hand, in the EDA implementation, the zone design is adjusted to find intra-homogenous zones with or without radial compactness as criterion is also applied.

5.11. Results and Comparison

There are two parts in this section. First part focuses on comparison between the GA and EDA implementations for zone design problem finding equal districts. Second part compares SKATER and the EDA implementation, for finding regions with least internal variation of selected attribute value.

5.12.Compare with GA for Inter-Homogeneity Regions

This section shows the performance of the GA and the two EDAs for finding inter-homogeneity regions, which involves two objective functions including equal population and radial compactness. For each algorithm, at most 100 generations were performed. Figure 5-9 shows the fitness values of each algorithm versus number of regions.

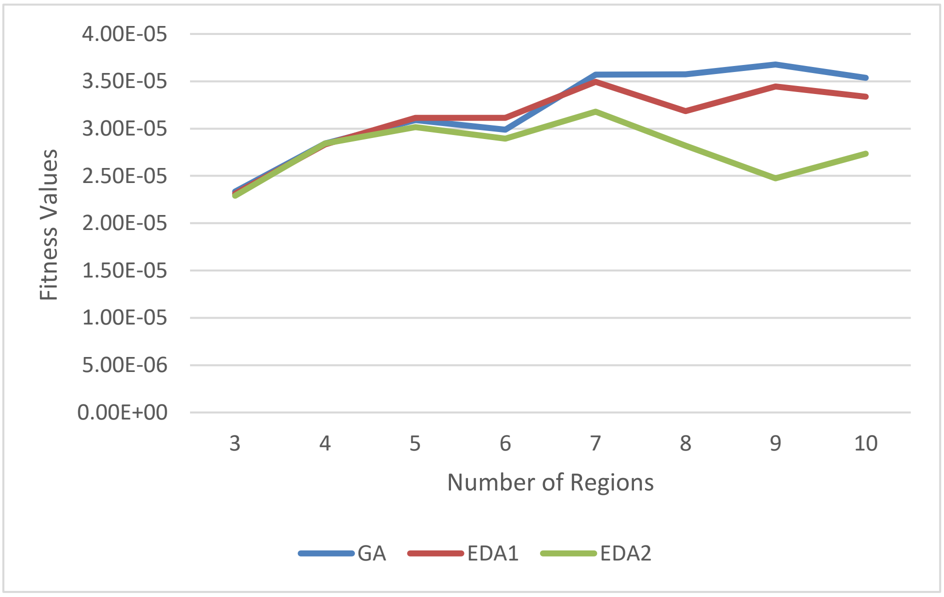


Figure 5-9 Fitness variation of GA, EDA1 and EDA2 against number of regions

According to the figure, fitness values of the solutions generated by GA and EDA1 are comparable. Although difference between the fitness values increases with the increasing number of regions, the magnitude is very tiny in the range from 1.18E-07 to 4.19E-05. Fitness graphs and corresponding best configuration of the zonal units generated by GA and EDA1 are shown in Figure 5-9, Figure 5-13, Figure 5-15 and Figure 5-17 respectively.

In general, fitness graph shows the change of fitness values over generations during the evolutionary process. By observing the following characteristics of the graph, it can reveal the searching behaviours for better understanding whether it is working or further modification of relevant parameters is required.

i. Convergence: If the trend of fitness values is increasing over generations, the search is on the right track looking for the optimal solutions. Otherwise, operators and its parameters are needed to be reviewed and adjusted.

ii. Maximum, Mean and Median of the fitness values: In each generation, the algorithm finds a population of candidate solutions, in which maximum fitness shows the best solution so far it can identify while mean and median fitness indicate the central tendency of the population. Further, mean fitness over generations also shows an asymptotic convergence toward the optimum achievable by the algorithm. However, mean value is only useful when the fitness values of the population are in normal distribution. If the distribution of the fitness values becomes skewed, median fitness should be observed instead. Furthermore, when the median fitness is closed to the maximum fitness, it indicates that the entire population tends to be homogenous and usually this happens when the solutions are almost converged. If the mean fitness is far from the maximum fitness, it indicates that the population maintains certain level of diversity.

iii. Shape of the fitness curve – Steepness of the fitness curve indicates how fast the algorithm can identify high quality solutions. Normally, a very steep fitness curve will be shown at the initial generations because the process is normally initialized by random solutions having relatively lower quality. Afterwards, the curve will become less steep and even flat at later generations when solution is converged.

In addition, termination of the algorithm is usually defined by number of maximum generations or difference of the best fitness between generations is less than a pre-defined limit.

By observing the fitness graphs of GA, mean fitness and fitness of best solutions increases gradually until certain number of generations, e.g. about 50 generations for 3 and 4 regions, about 60 generations for 5 and 6 regions. Bigger gap between the mean fitness and the fitness of best solutions indicates that GA maintains relatively high diversity within the population. The green shaded area of the fitness growth graph shows the median of the fitness values in each generation of population. When it is far from the mean fitness, it indicates the distribution is skewed meaning particular solutions are dominating the population. On the other hand, it is observed that a fast convergence for the case of EDA1 that requires up to 20 generations to reach steady maximal for small number of regions, and requires up to 40-50 generations for higher number of regions showing the effectiveness of the use of probabilistic model in the EDA1.

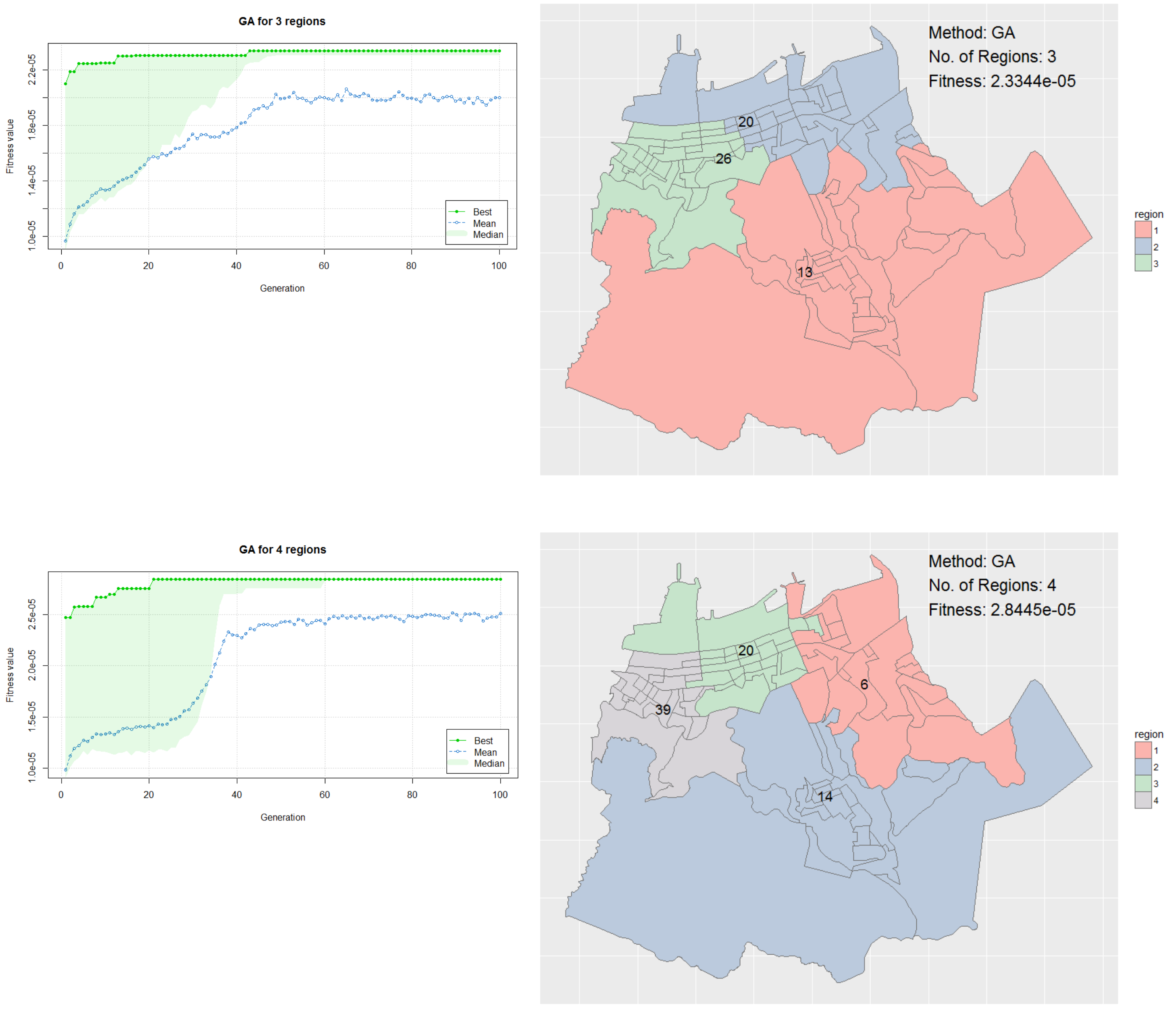


Figure 5-10 Fitness growth graphs and best solutions for 3&4 regions using GA

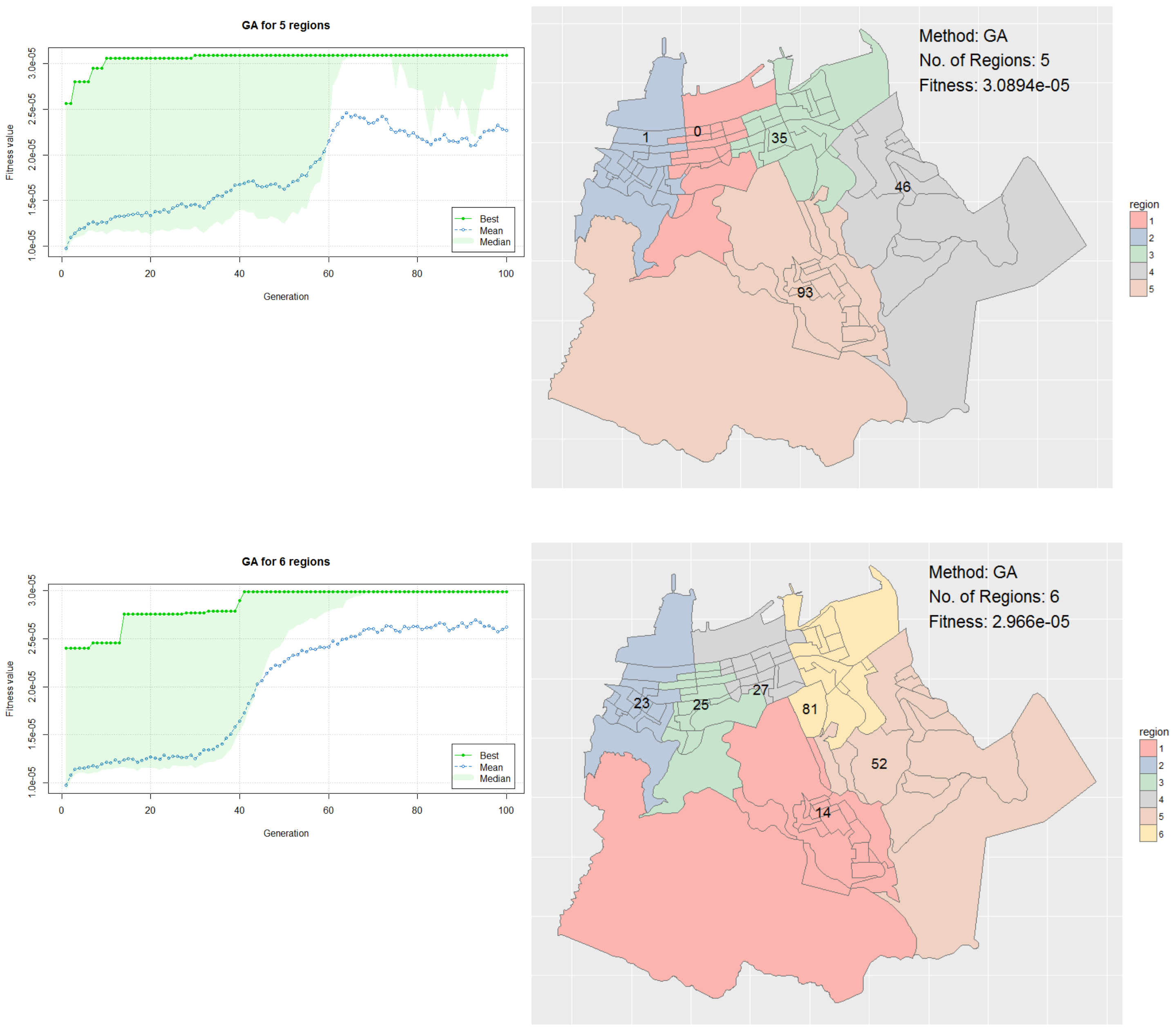


Figure 5-11 Fitness growth graphs and best solutions for 5&6 regions using GA

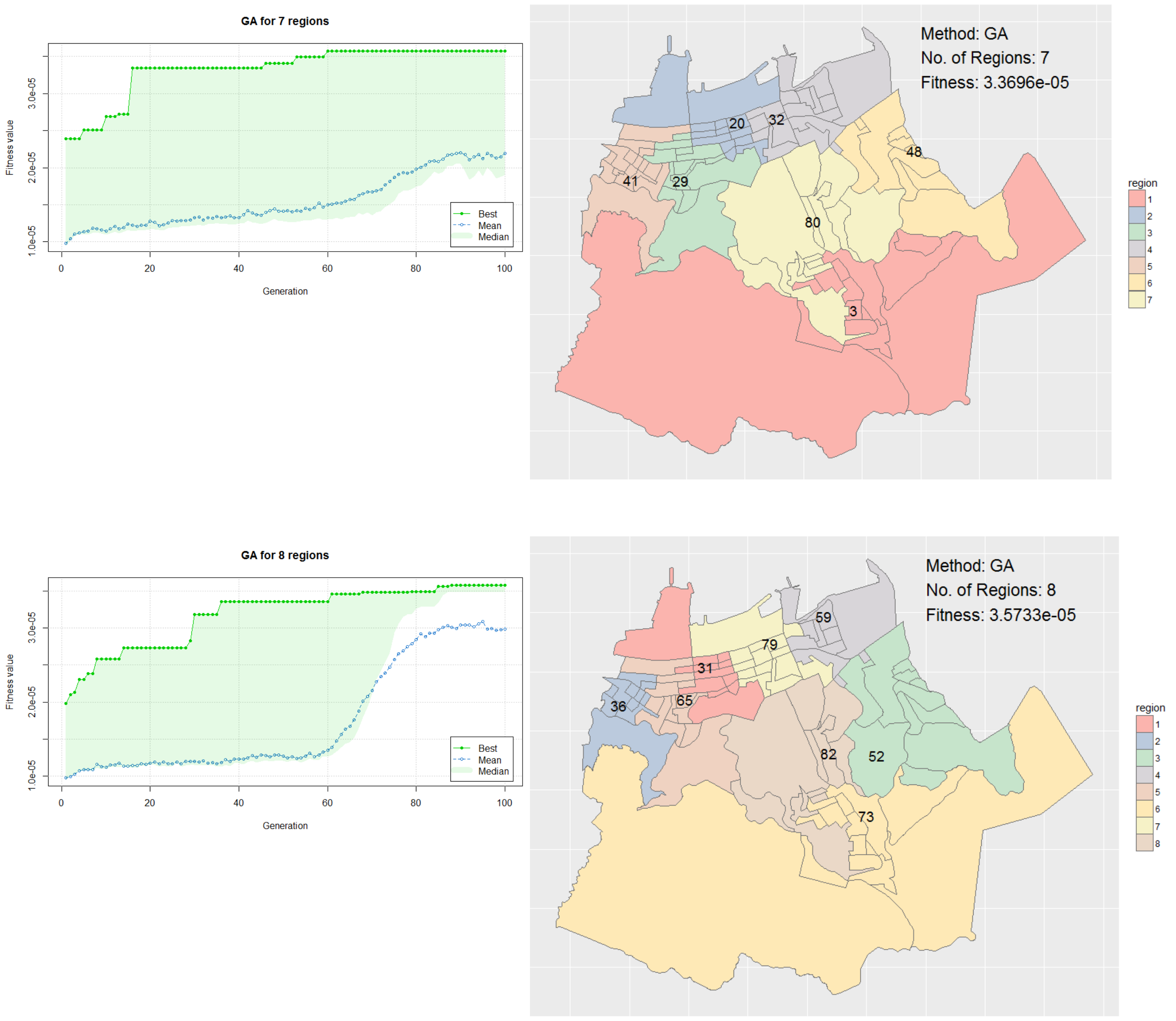
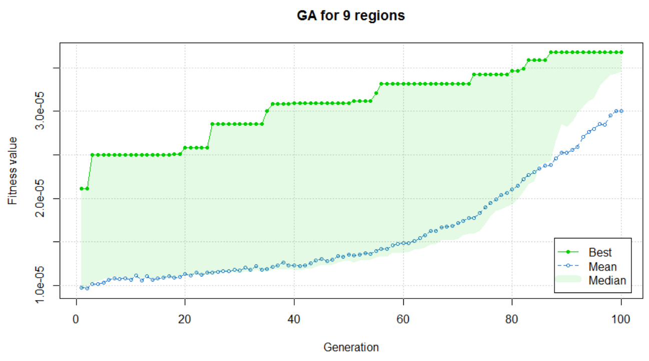
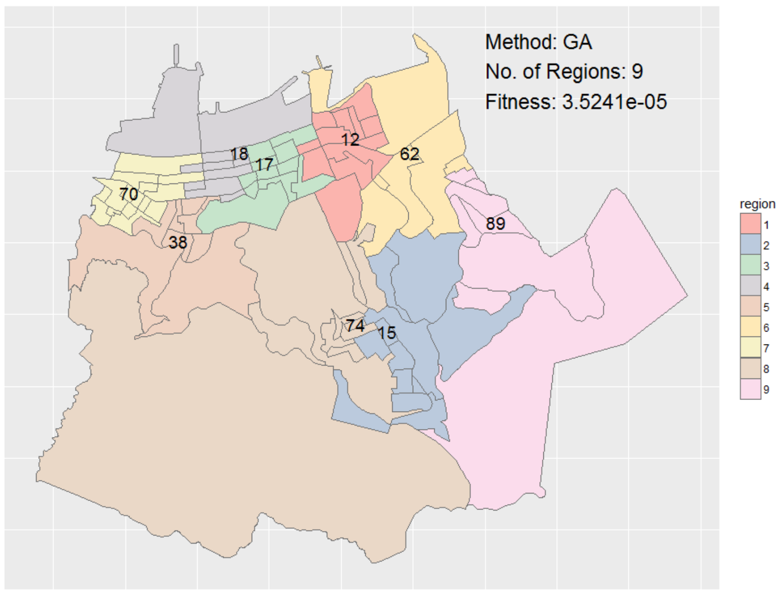
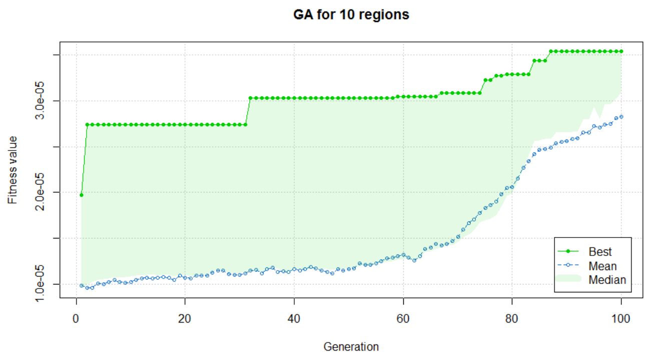


Figure 5-12 Fitness growth graphs and best solutions for 7&8 regions using GA







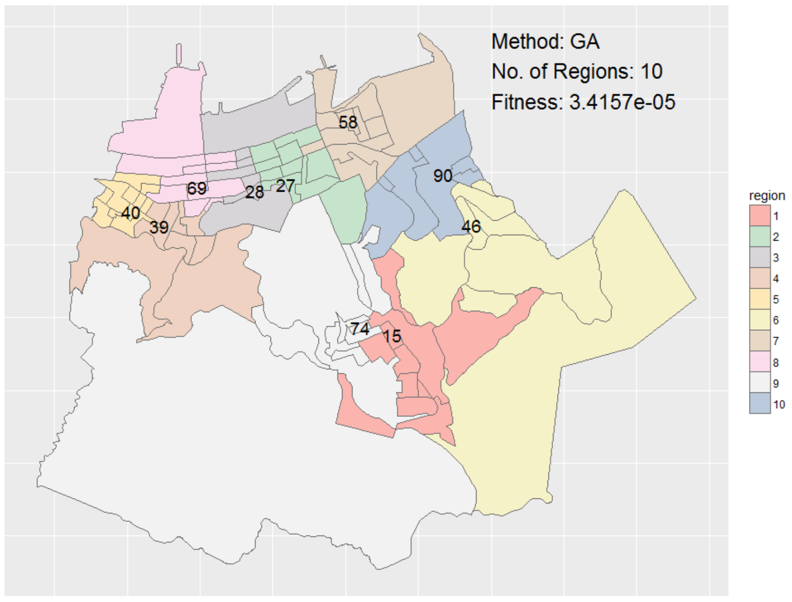
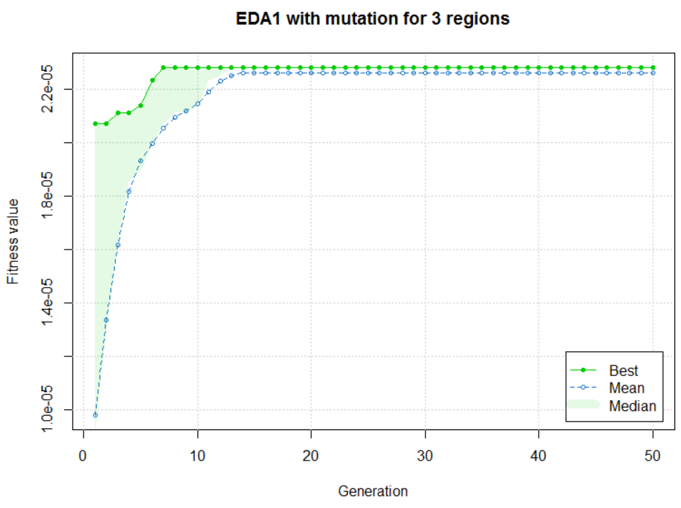
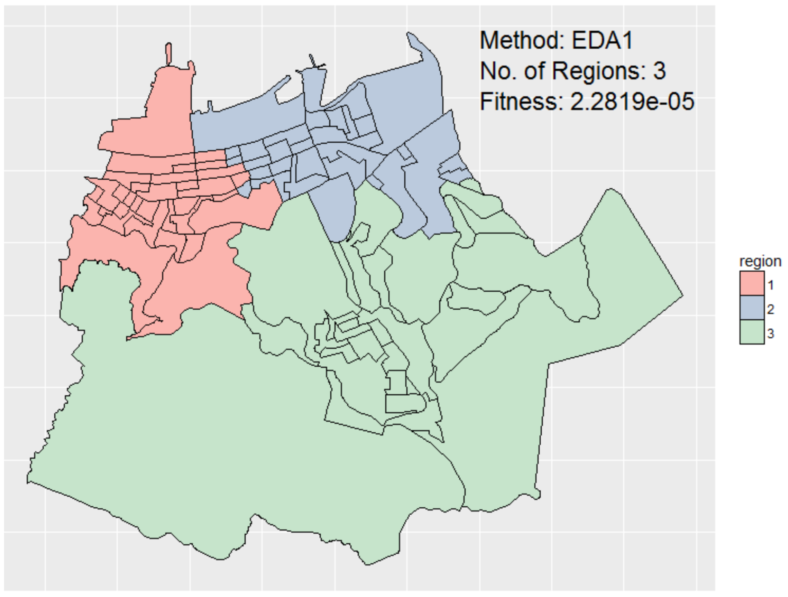
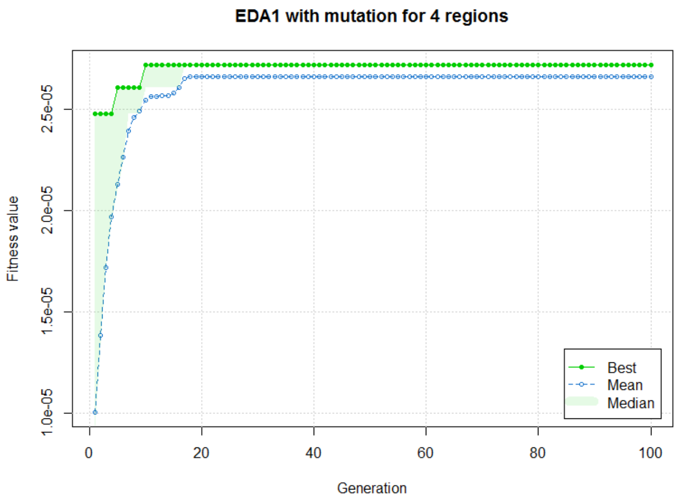


Figure 5-13 Fitness growth graphs and best solutions for 9&10 regions using GA







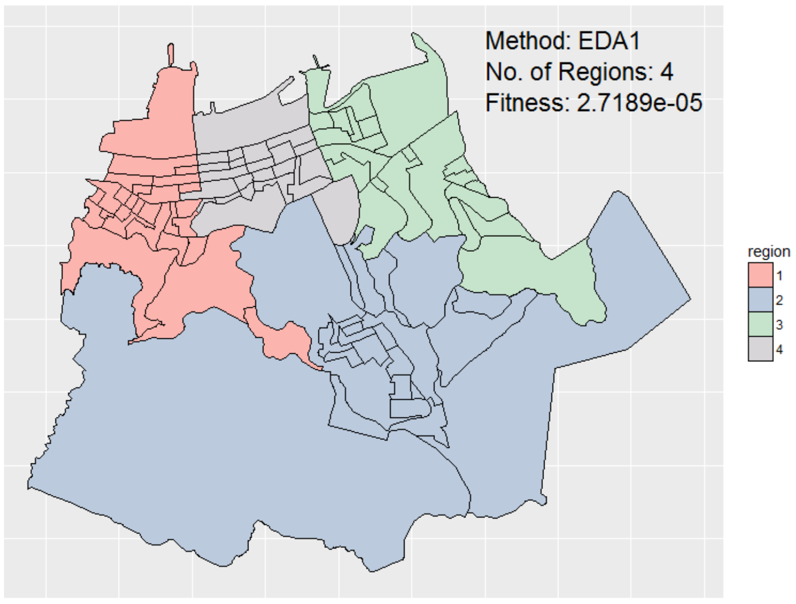
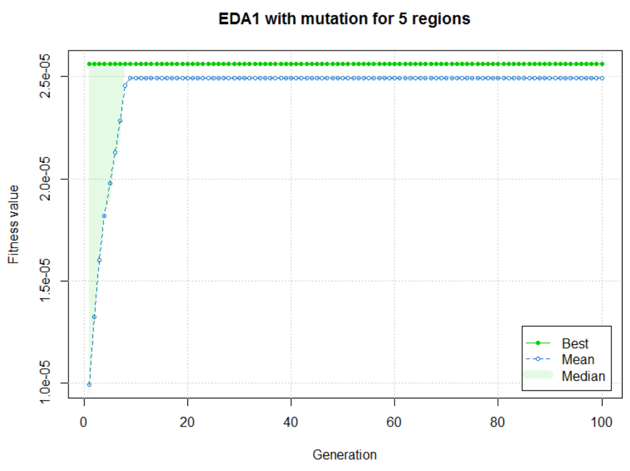
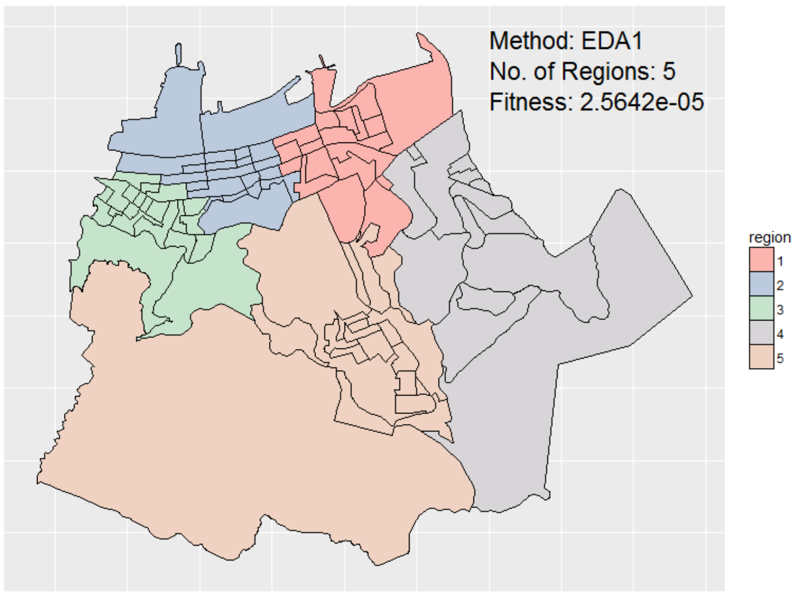
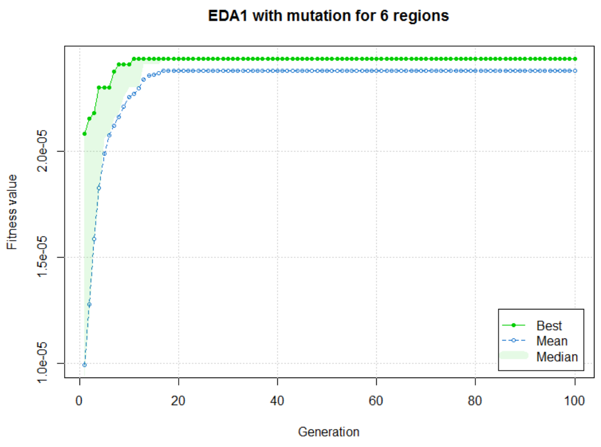


Figure 5-14 Fitness growth graph and best solution for 3&4 regions using EDA1







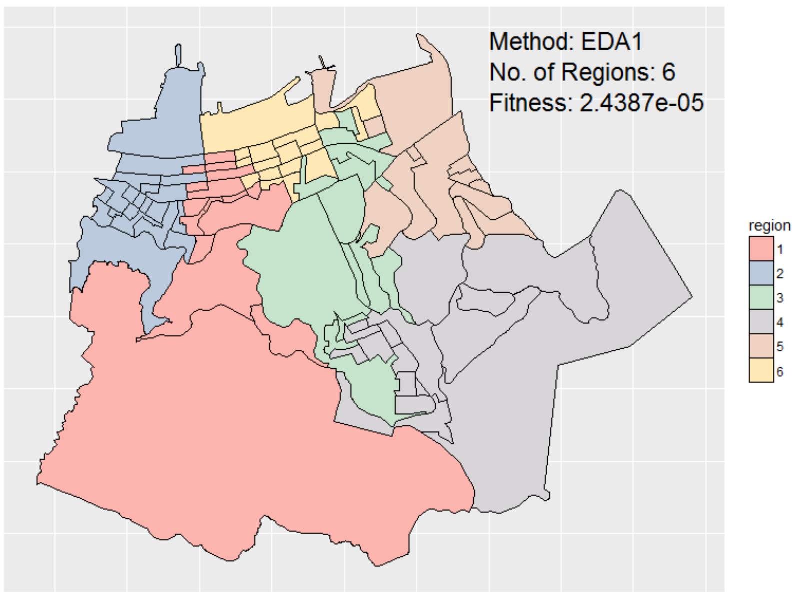
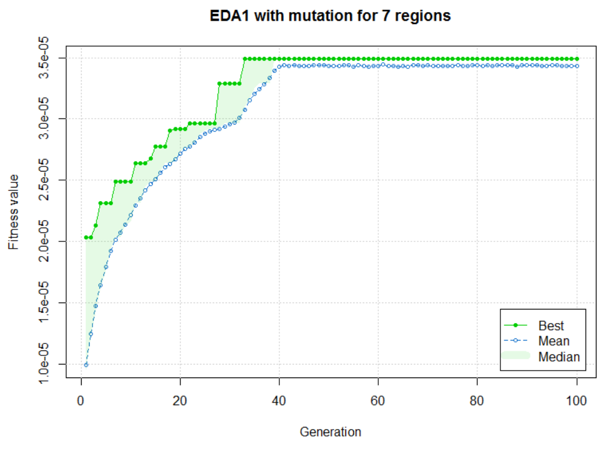
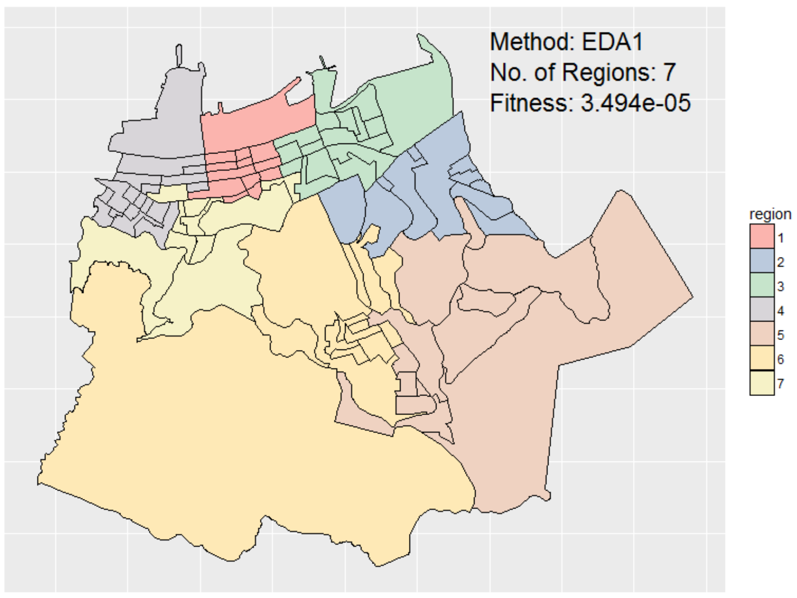
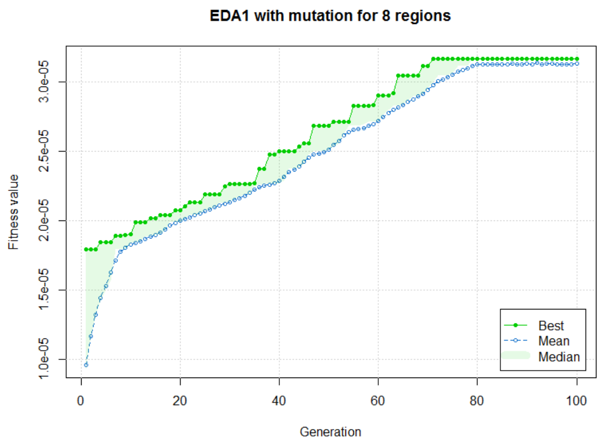


Figure 5-15 Fitness growth graph and best solution for 5&6 regions using EDA1







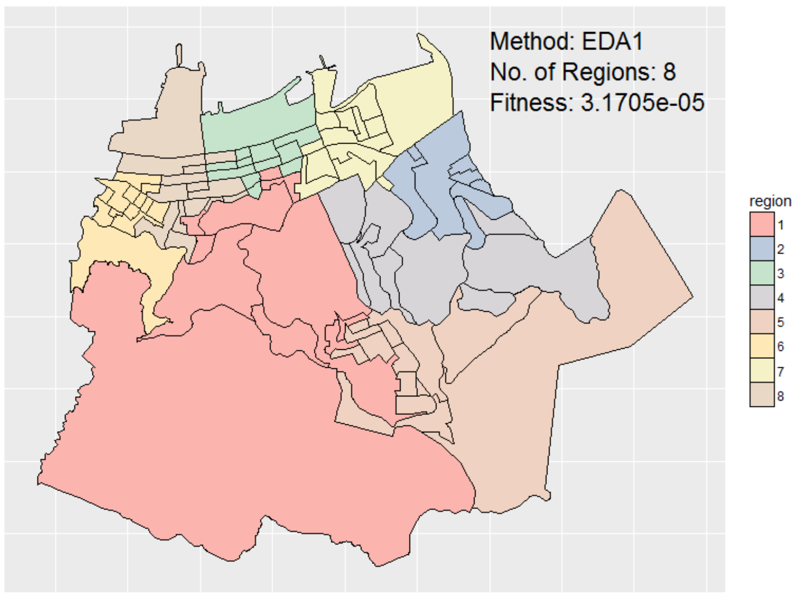
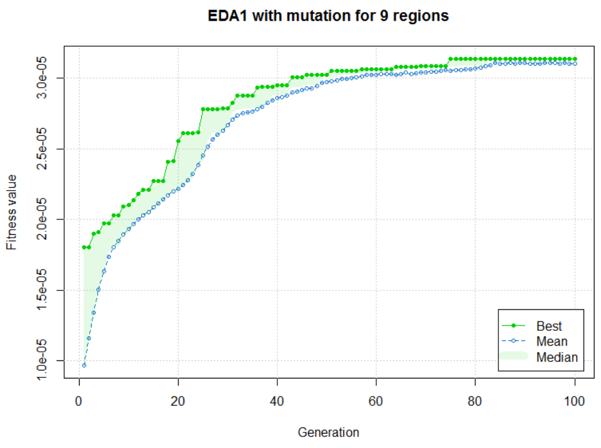
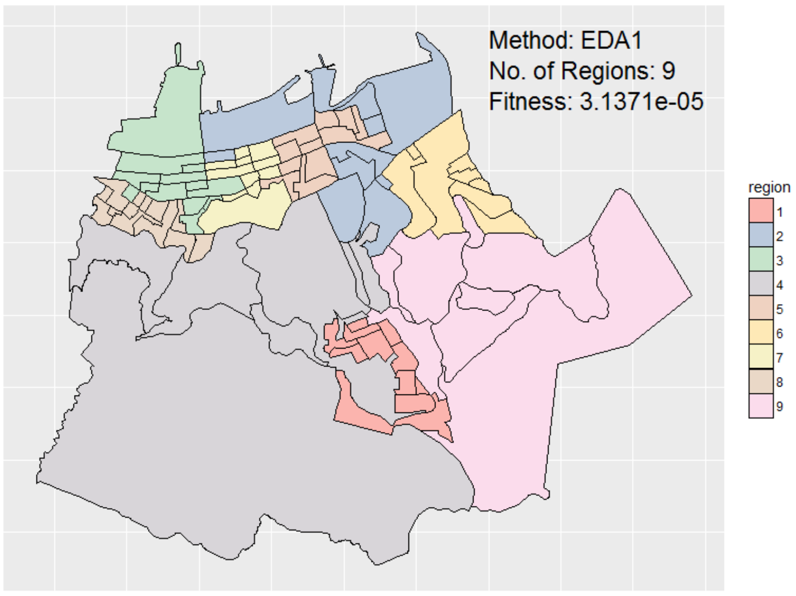
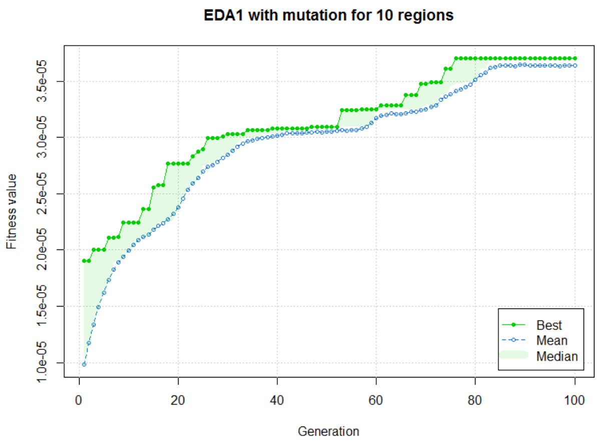


Figure 5-16 Fitness growth graph and best solution for 7&8 regions using EDA1







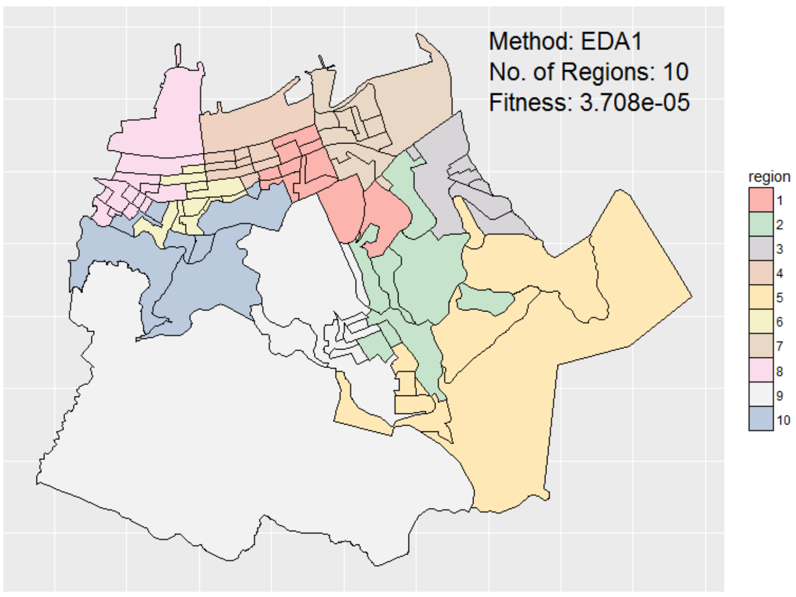


Figure 5-17 Fitness growth graph and best solution for 9&10 regions using EDA1

Regarding the implementation of GA and EDA1, the major difference is the use of different solution encoding methods, in which GA uses the regional-centre encoding that gives it better performance for evolving solutions because its simplicity. However, it is less flexible for composing a solution that the group-number encoding adopted in EDA1 can introduce minor refinement at the zonal unit level to reach higher fitness value. To better understanding the effect of the solution encoding, EDA2 employs the same encoding of GA giving a head-to-head comparison.

Besides, EDA1 is running with mutation and it is a very crucial factor to improve solution quality over generations. It is found that when EDA1 is hard to maintain its exploring capability. Therefore, a mutation operator is added into EDA1 trying to improve the diversity. Figure 5-18 shows the fitness values achieved by EDA1 with and without mutation.

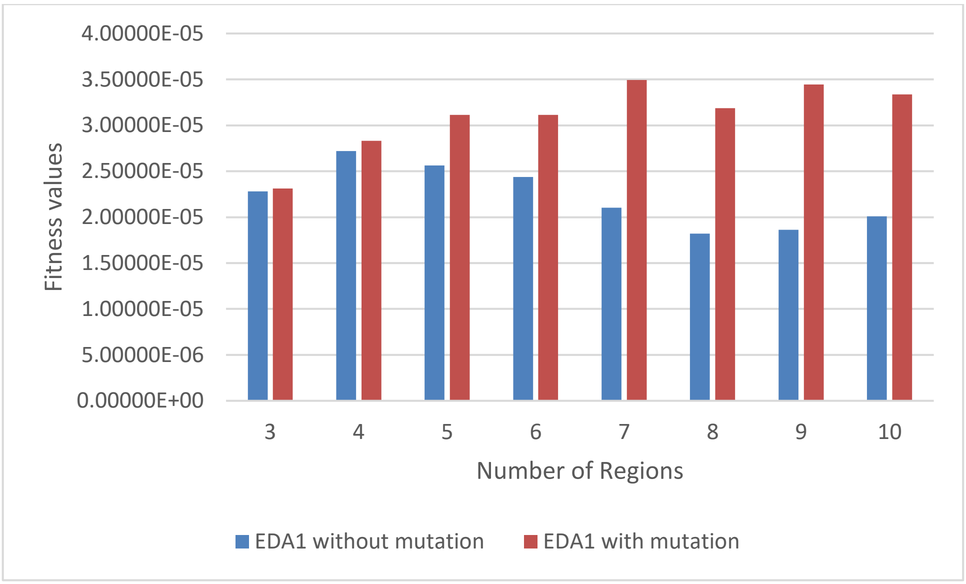


Figure 5-18 Fitness Values of EDA1 with and without mutation

	GA		EDA1		EDA2	
Population Size	Mean	Best	Mean	Best	Mean	Best
1000	2.67E-05	2.94E-05	3.06E-05	**3.12E-05**	2.85E-05	3.09E-05
2000	2.53E-05	3.08E-05	3.06E-05	**3.11E-05**	2.75E-05	2.94E-05
3000	2.51E-05	3.08E-05	3.06E-05	**3.11E-05**	2.73E-05	2.97E-05
4000	2.37E-05	2.99E-05	3.06E-05	**3.12E-05**	2.73E-05	2.97E-05

	GA		EDA1		EDA2	
5000	2.46E-05	3.08E-05	3.20E-05	**3.29E-05**	2.72E-05	2.94E-05
6000	1.71E-05	3.08E-05	3.20E-05	**3.26E-05**	2.82E-05	3.08E-05

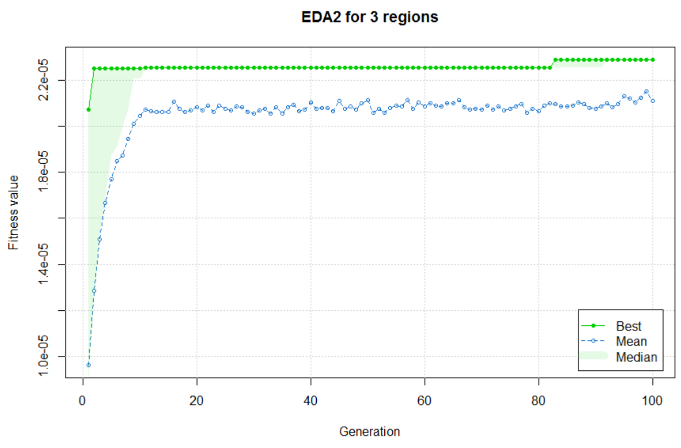
Table 5-4 Fitness Value of GA and EDA2 with increasing population size for a 6-region problem with best values in bold

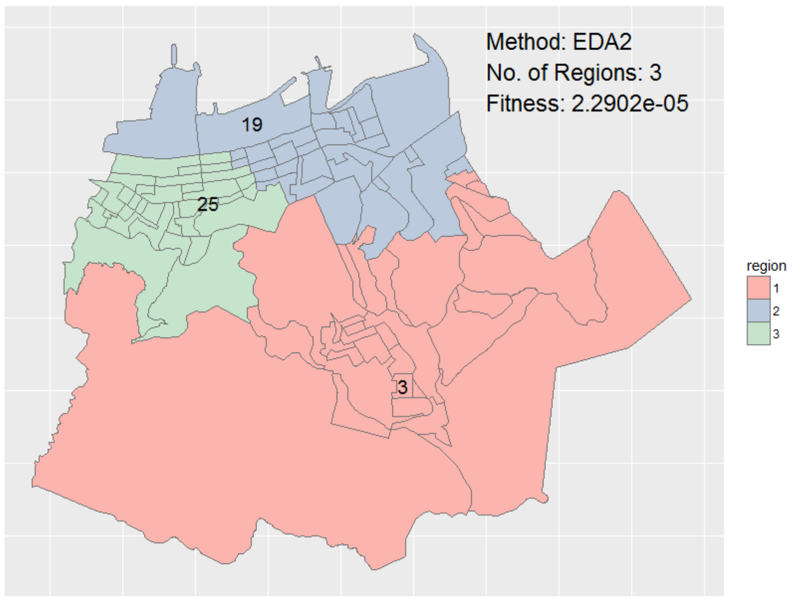
As shown in Figure 5-18, it is obvious that mutation provides a positive effect that can improve fitness of the solutions, especially when dealing with higher number of regions. Without mutation, probabilistic model learnt from the population is converged prematurely making it hard to explore new solution. As a result, when mutation is used, it takes the role for exploration. To further investigate the effect, several tests on GA, EDA1 and EDA2 with increasing population size for a 6-region problem have been performed and the results are shown in Table 5-4. As expected, after increasing the population size, EDA1 outperforms both GA and EDA2 indicating that probabilistic approach requires a larger population size to learn a more effective model and generate higher quality solutions. The reason to acquire larger population size is highly dependent on the use of probabilistic model, in which EDA2 employs marginal probability (i.e. variables with no dependency) for modelling variables so that it can performs well even with small population size. On the other hand, larger population size is required for EDA1 so that it can obtain workable parameters from its multiple dependency model.

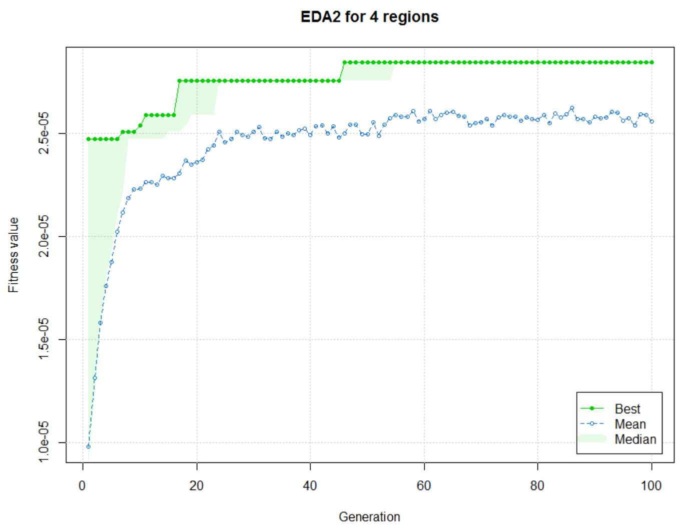
Furthermore, EDA1 and EDA2 are using different encoding methods, the comparison indicates that encoding method plays a crucial factor affecting the achievable solution quality that algorithm can reach. This is more obvious when the number of regions increases. As expected, by using the same encoding, difference of the best fitness values attained by both GA and EDA2 are close to zero and thus negligible, i.e. difference of their fitness values is ranging from 0.00 to 1.50-E06 only.

In fact, regional-centre encoding method is less flexible for improving solution quality because solution is basically determined by a set of regional centres, which does not allow zonal-level refinement intrinsically. To further improve solution quality generated by EDA2, besides further increase the population size and mutation rate, a local optimization process could be introduced after the evolution process is terminated. Fitness growth graphs and corresponding best configuration of the zonal units generated by GA and EDA2 are shown in Figure 5-20 and Figure 5-22 respectively.

Although the difference of fitness values among GA, EDA1 and EDA2 is so small, the result maps look quite different. The reason may due to the uneven size of zonal units, in particular, when it carries disproportionate quantity of interest, i.e. population in this study. Observing the fitness growth graph of GA, it is noted that there is a relatively larger difference between the mean fitness and the best fitness indicating the population maintains a relatively higher diversity of solutions. Regarding the EDAs, the mean fitness is closed to the best fitness indicating the population is highly converged. Because EDAs create new generation of population by its probabilistic model learnt from the present population, the model carries characteristics of the population, e.g. the probability distribution of each variable/zonal unit that could be used to perform inference finding the most probable assignment of each zonal unit.







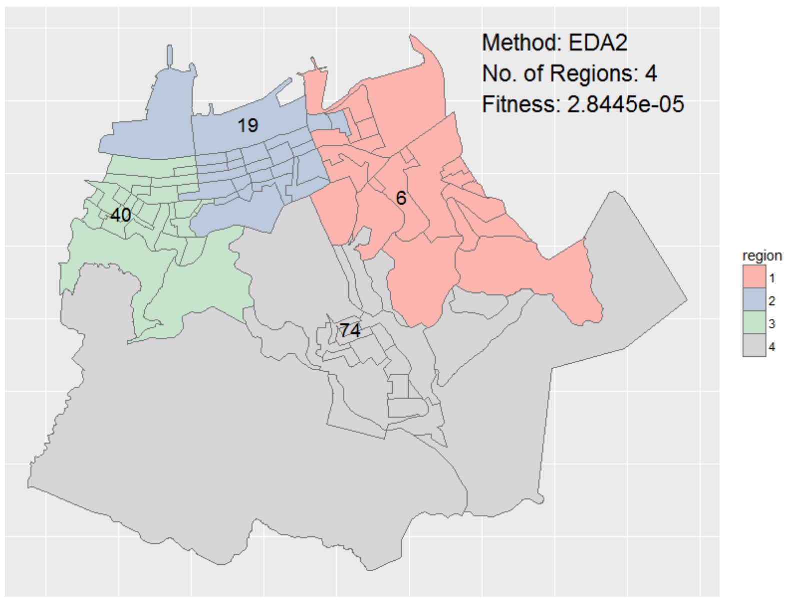


Figure 5-19 Fitness growth graph and best solution for 3&4 regions using EDA2

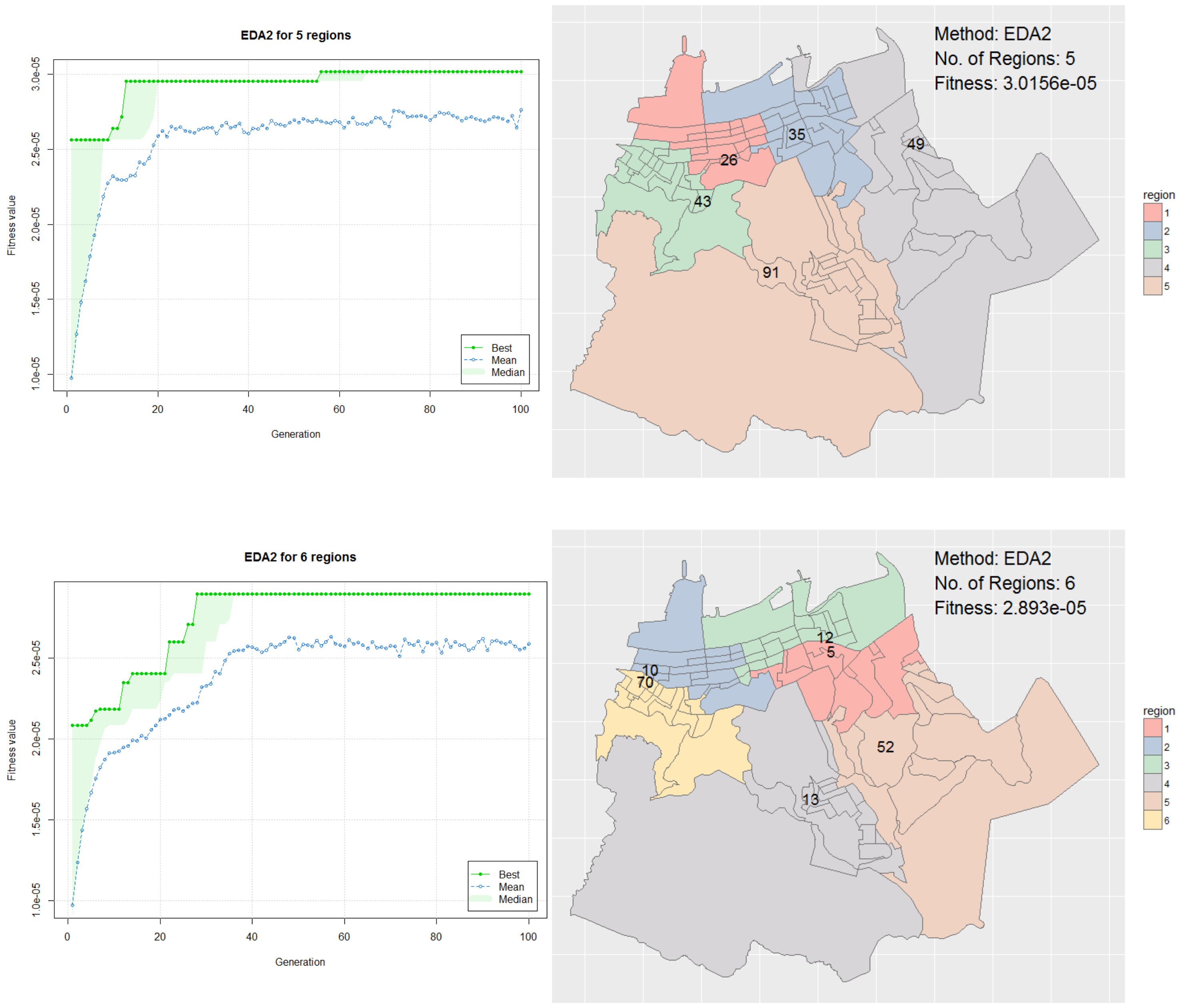
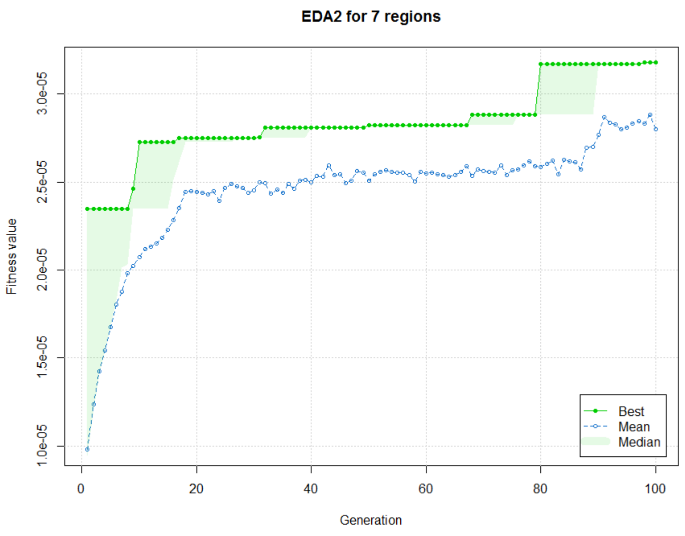
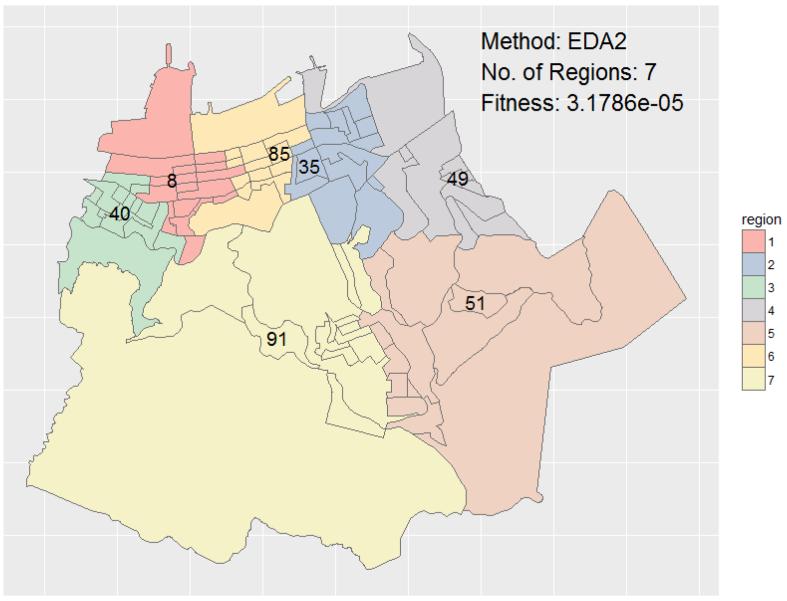
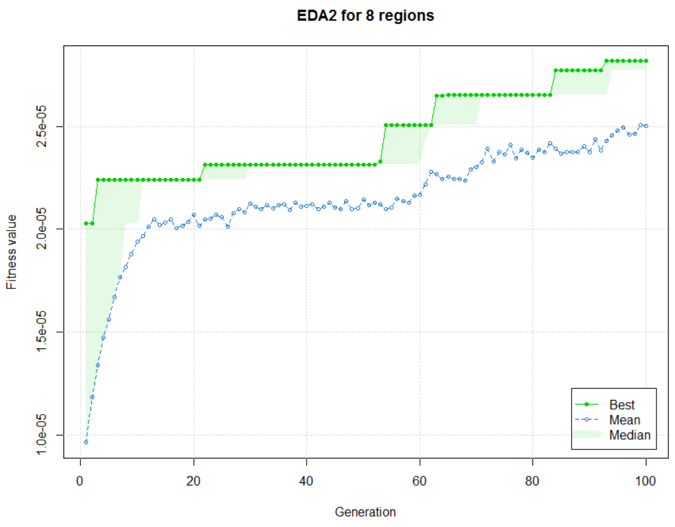


Figure 5-20 Fitness growth graph and best solution for 5&6 regions using EDA2







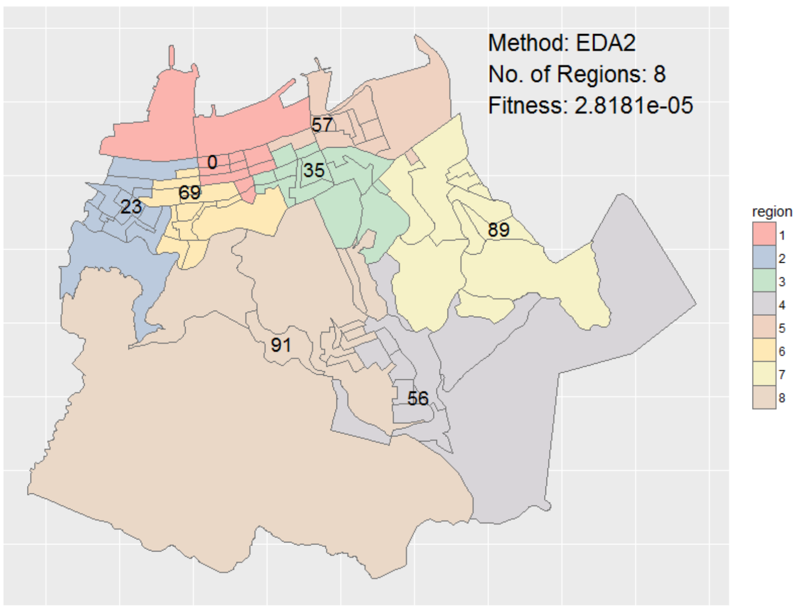
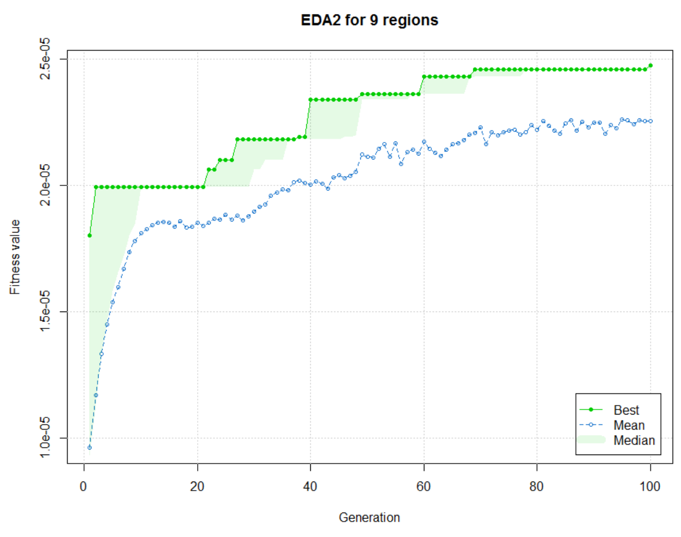
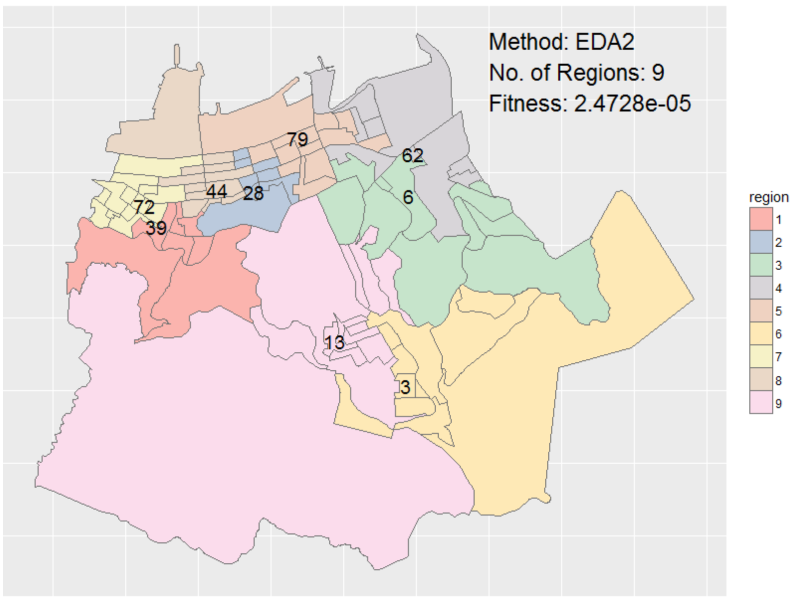
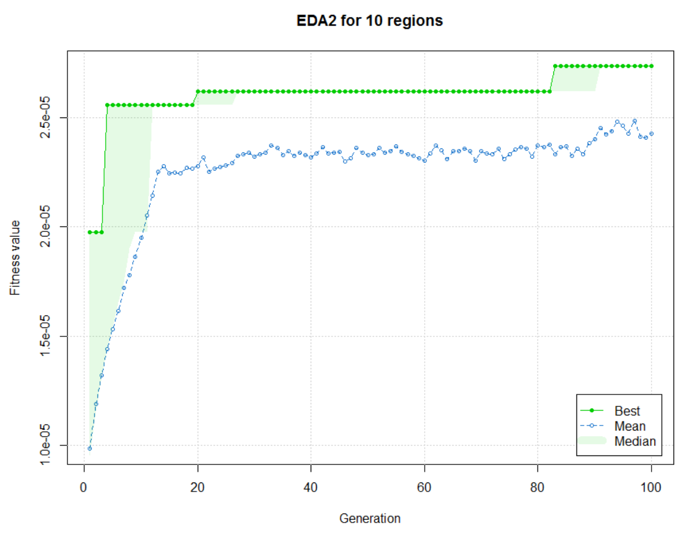


Figure 5-21 Fitness growth graph and best solution for 7-10 regions using EDA2







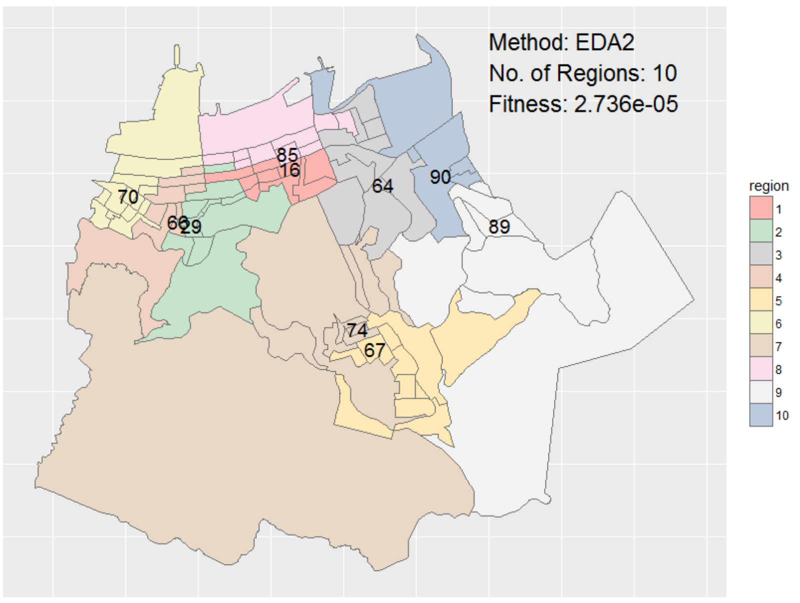


Figure 5-22 Fitness growth graph and best solution for 9&10 regions using EDA2

5.13. Compare with SKATER for Intra-Homogeneity Regions

This section compares the intra-homogenous clusters generated by SKATER and EDA1.

To run SKATER, a lower bound of population should be defined to ensure it can generate balanced and reasonable regions. Otherwise, solution found will contain a single-zone region because the fitness value of a single-zone region always equals to zero making the search stopped to find solution by combining zonal units. To determine the lower bound, characteristics of the zonal data whether there is irregular distribution of concerned attribute values and very high variation in the size of the zonal unit should be taken into account and it should also be inversely proportional to the number of expected regions. If the lower bound is too small, a single-zone region will more likely be generated as illustrated in Figure 5-23. However, if the lower bound is too large, no solution can satisfy the constraint and the algorithm will stop prematurely and therefore sufficient number of regions cannot be obtained. In this study, the lower bound is set to 1.5 times of the average population for each region.

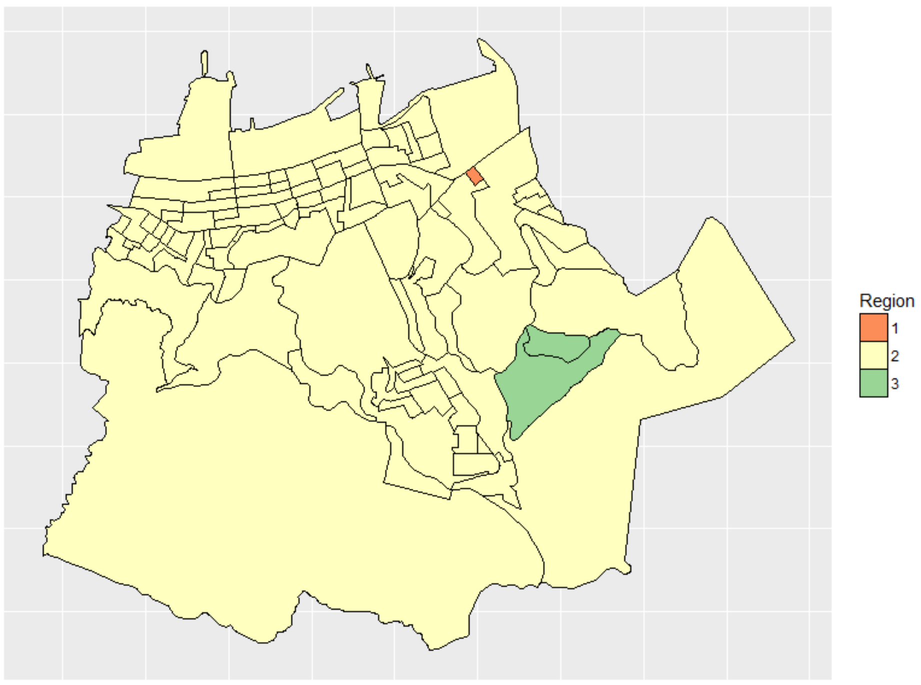


Figure 5-23 An example showing unbalanced solution when lower bound is set to a small value

Although SKATER is designed to find intra-homogeneous regions, it is a single objective optimization method. In order to allow comparison, in addition to set the same condition for configuring EDA1, EDA1 is configured to optimize both intra-homogeneity and compactness, which should be difficult than applying single objective. Meanwhile, fitness values in both homogeneity and compactness achieved by SKATER are also measured for comparison purpose.

As a result, several tests with increasing number of regions were performed. Results showing the fitness values attained for homogeneity, as well as homogeneity and compactness with increasing number of regions using SKATER and EDA1 respectively can be found in Table 5-5.

	SKATER		**EDA1**	
No. of Regions	homogeneity	homogeneity and compactness	homogeneity	homogeneity and compactness
3	6.00E-06	4.02E-06	6.32E-06	4.94E-06
4	6.00E-06	4.02E-06	6.44E-06	5.17E-06
5	6.00E-06	4.05E-06	6.51E-06	5.35E-06
6	6.00E-06	4.14E-06	6.64E-06	5.49E-06
7	6.01E-06	4.21E-06	6.70E-06	5.62E-06
8	6.01E-06	4.27E-06	6.88E-06	5.76E-06
9	6.01E-06	4.31E-06	6.98E-06	5.86E-06
10	6.01E-06	4.34E-06	7.02E-06	6.00E-06

Table 5-5 Fitness values of SKATER and EDA1 with increasing number of regions. (Higher values mean better)

In the table, fitness values are classified by homogeneity, and homogeneity and compactness for both SKATER and EDA1. Although compactness is not one of the optimization criteria in SKATER, it is also measured for each generated solution based on the objective functions to observe an overall quality of regions it can achieve. On the other hand, both homogeneity and compactness are criteria being optimized in EDA1. According to the results, EDA1 outperforms SKATER consistently. This can be explained by the high flexibility of the encoding method that can compose different solutions easily in EDA1. In SKATER, it is believed that generation of new solutions are restricted by the minimum spanning tree.

Actual zone design generated by SKATER and the best results generated by EDA1 with intra-homogeneity and compactness for 3-region and 10-region corresponding to Table 5-5 are shown in **Error! Reference source not found.**.

As shown by the appearance of regions formed by SKATER, several regions are composed of two zonal units only. This is because SKATER remove edges from the MST once the region is within the bounding limits until sufficient number of regions is created. This case also indicates that the lower bound may still be too small. However, with the same bounding condition, the EDA1 can provide a better appearance and this indicates that results in EDA1 is driven by the objective functions correctly. Another observation is that results of SKATER would create regions within region. This happens when edge connecting to leave node within 1st or 2nd order separation is selected for testing the SSD and subsequently be removed disregarding surrounding region assignment. Since the number of edges of this kind is relatively higher, this scenario is of higher chance to happen in SKATER. As shown in Figure 5 25, if the lower bound is set to a larger value, the result of SKATER will look much balanced. However, it cannot maintain a good compactness as indicated by the fitness of the result at about 6.00E-06 when considering homogeneity solely, and at about 4.02E-06 when considering both homogeneity and compactness. Both fitness values are lower than that provided by EDA1.

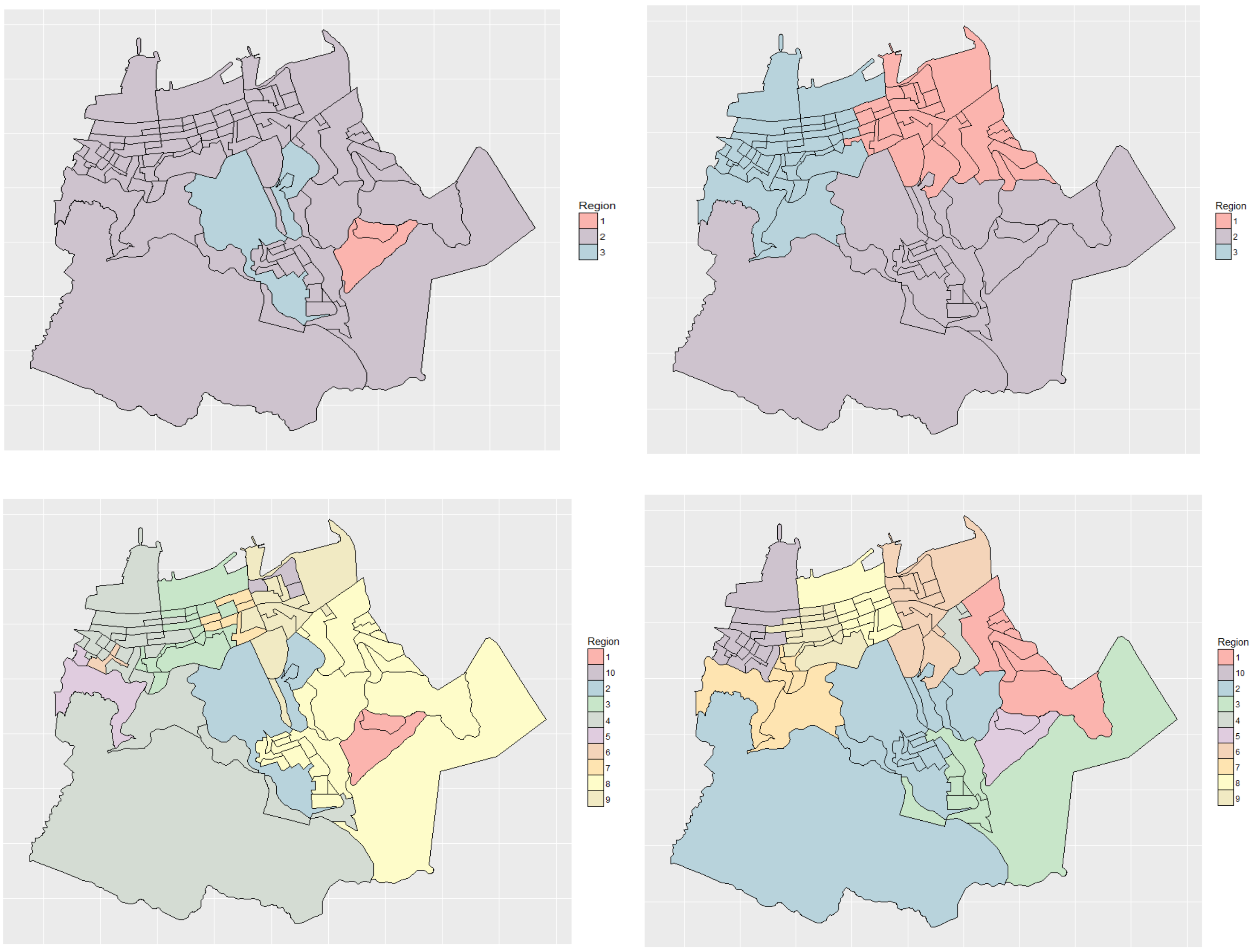


Figure 5-24 Regionalization results generated by SKATER (Left) and EDA1 (Right) for 3-region (Above) and 10-region (Below)

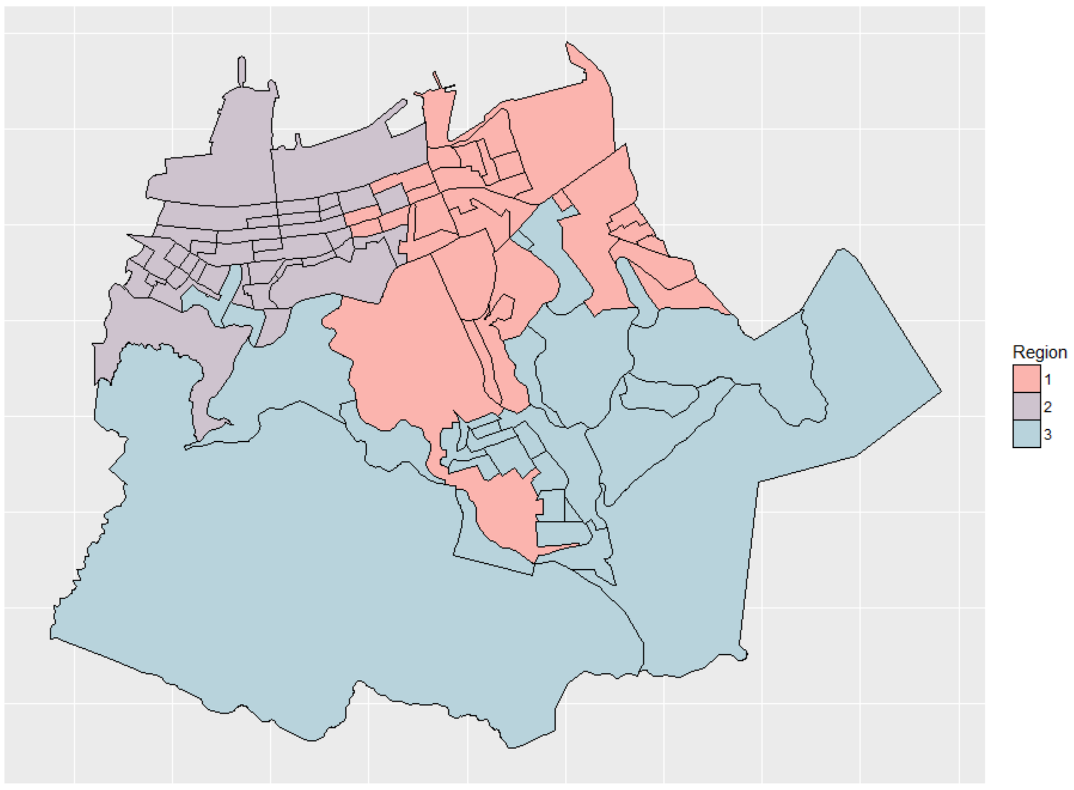


Figure 5-25 Result of SKATER for a 3-region problem with a higher lower bound

5.14.Generate TPUs using EDA based Zone Design

As mentioned previously, TPU is for town planning purpose and it can be formed by aggregation of Street Blocks. In fact, applications of TPU and Street Block are not limited to town planning and it would also be used for socio-economic analysis or presenting analysis results. In some circumstances and for convenience, TPU may be adopted for presenting underlying socio-economic data. To do this, it is reasonable that formation of regions by aggregating Street Block should respect criteria of equal population and compactness as discussed in Chapter 2. Otherwise, it will be difficult to ensure meaningful aggregated results and avoid bias among regions especially when dealing with ratio and percentage, which could hit the small numbers problem

(Choynowsk 1959; Cromley & McLafferty 2012). In this section, same number of TPUs that covers the test area will be generated by EDA1 subject to the two objective functions for comparison. It is not expected to generate regions having the same configuration as the existing TPUs. However, this comparison can provide an idea how likely the existing TPUs are conforming to the two criteria and what will the regions look like when they are conforming to the two criteria.

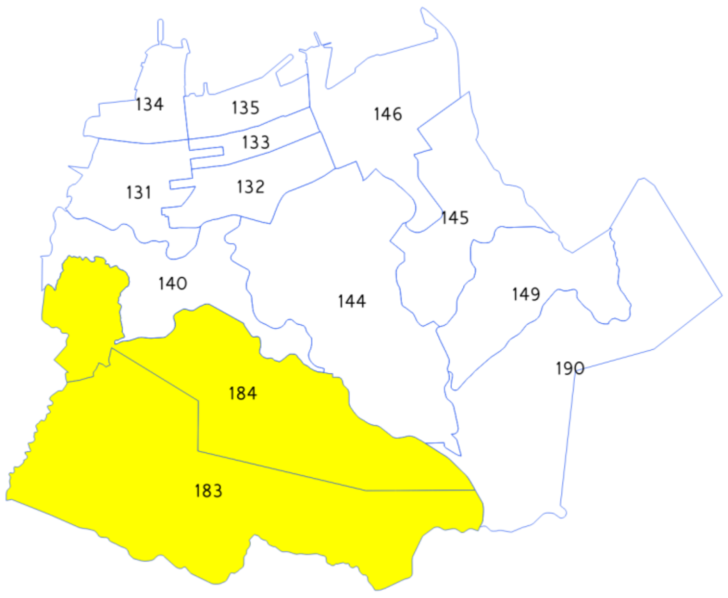


Figure 5-26 TPUs with yellow highlighted indicating 2 units to be removed

To ensure meaningful comparison, the test data should be able to re-construct the TPUs. However, due to the test data is based originated from the Large Street Blocks, the TPUs 183 and 184 are merged in the test data and there is no chance to re-generate these two TPUs using the test data. Original TPUs and the test data can be found in Figure 5 7 and Figure 5 8 respectively. To resolve this situation, spatial units covered by TPUs 183 and 184 should be removed from the TPUs and test data. As a result, 2 Large Street Blocks should be removed from the test data. Therefore, there are remaining 92 Large

Street Blocks in the revised test data, which will be used to aggregate 11 regions. Figure 5-26 shows the existing TPUs of Wan Chai District with yellow highlighted indicating the parts to be removed.

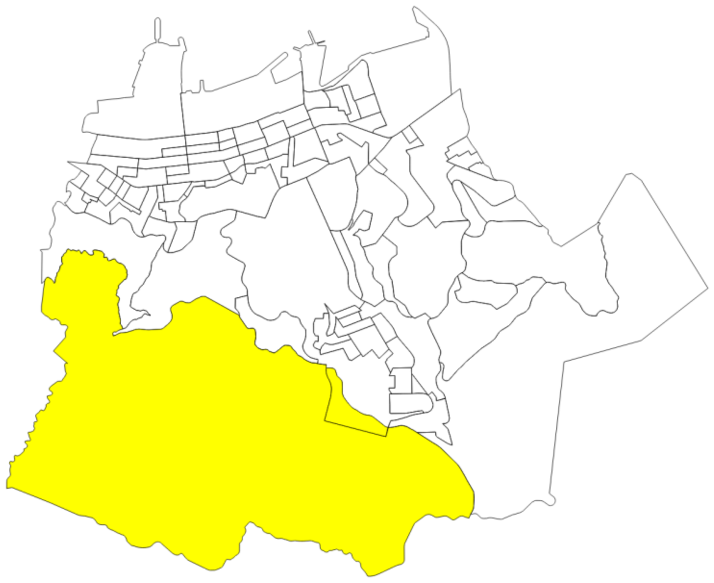


Figure 5-27 Test Data with yellow highlighted indicating the 2 units to be removed

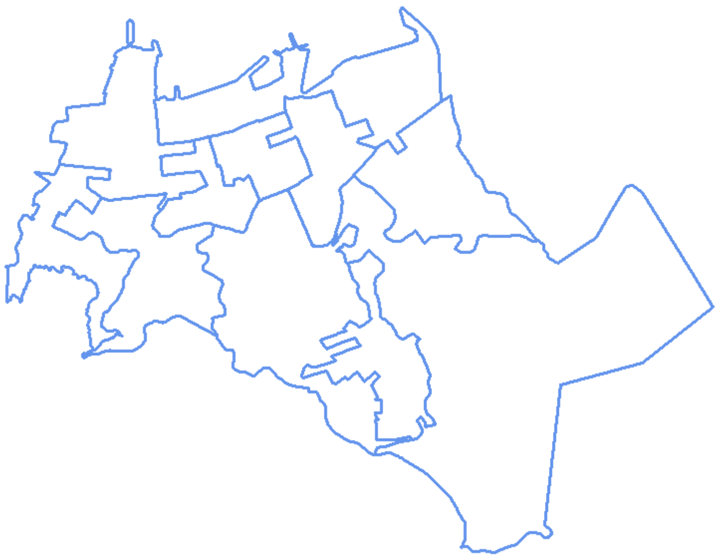


Figure 5-28 11 Regions generated by EDA1

As a result, given the two criteria of equal population and radial compactness, Figure 5-28 shows the 11 regions generated by EDA1. A visual comparison of the appearance with the TPUs as depicted in Figure 5-26 indicates that EDA1 has generated a different configuration of regions. To understand the difference, it can be revealed by comparing the fitness values they attained, position of the regional centres and their neighbours, and distribution of the population data.

Regarding the fitness values, Table 5-6 shows the overall fitness values and its components in equal population and compactness. It clearly shows that regions generated by EDA1 has achieved very small value in equal population, which is about 5% of TPU. Meanwhile, both solutions have similar measures on compactness but regions generated by EDA1 is slightly better than that of TPU. The fitness values of equal population and compactness are computed by Equations 1 and Equation 4 of Chapter 1 respectively. Obviously, quality of EDA1 is better than that of TPU in accordance to the two objective functions.

	Equal Population	**Compactness**	**Overall Fitness**
TPU	108,362	24,611	132,973
EDA1	6,136	22,863	28,999 (best value)

Table 5-6 Fitness values of TPU and 11 regions generated by EDA1

Next, another way to show the difference is to compare the difference of regional centres and it neighbours between the original TPU and the EDA1 generated regions. Regional centres and their neighbours of the 11 regions and the TPU are shown in Figure 5-29 and Figure 5-30 respectively.

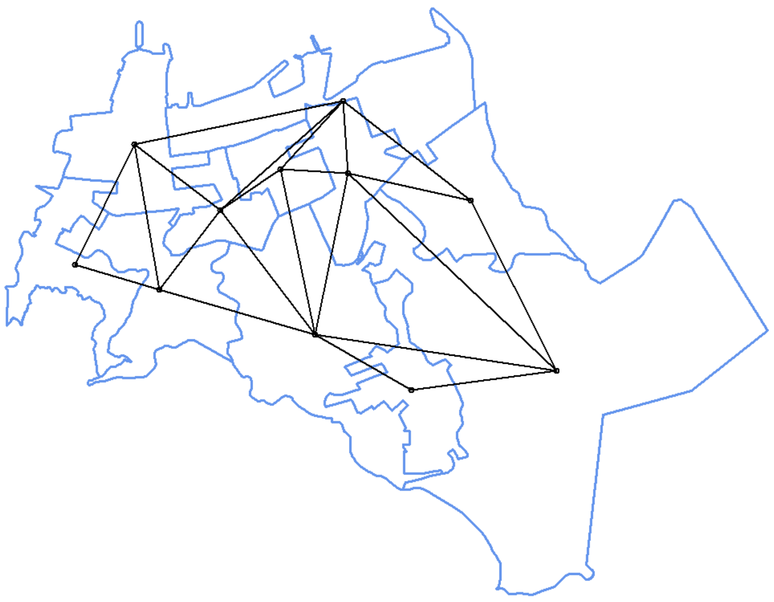


Figure 5-29 Regional centres and their neighbours in the 11 regions generated by EDA1

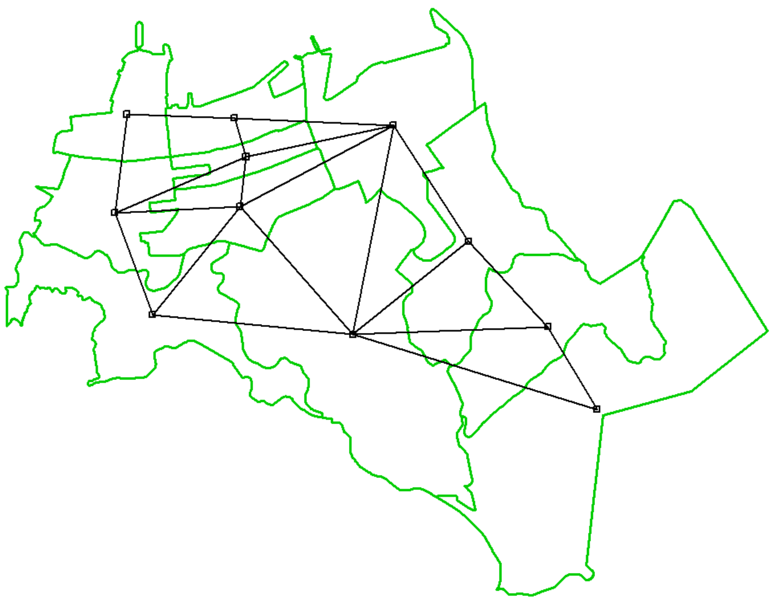


Figure 5-30 Regional centres and their neighbours of the TPU

As indicated in the figures, although the regions generated by EDA1 is different from that of TPU, the configuration as depicted by their centres and neighbours looks similar and this is consistent with the fitness values of compactness that the difference in term of the configuration is not significant.

Furthermore, descriptive statistics of the original large street block, TPU and the regions generated by EDA1 show that data distribution generated by EDA1 has smaller variation (indicated by the standard deviation of the population data) when compared with that of TPU and even smaller than that of the original Large Street Bock. Variation of statistics due to aggregation of zonal data is due to Modifiable Areal Unit Problem (MAUP) (Openshaw 1976), which aims to understand the scale and aggregation effects in region-building. Scale effect refers to uncertainty on the number of regions required in a study, while aggregation effect refers to the question in how to aggregate zonal data to form given number of regions. The significance of MAUP is that both scale and aggregation effects would change the model parameters, awareness should be raised when performing the two operations in areal data or otherwise, incorrect results and decision could be made. However, when the region-building or the aggregation action is under controlled, such action can be transformed to a spatial analysis tool.

Table 5-7 shows the comparison of the key statistics among the three datasets. It shows that regions generated by EDA1 provides the least variation in the population data, which is preferable for further data analysis.

Dataset	Minimum	1st Q	Median	Mean	3rd Q	Maximum	S.D
Original Large Street Blocks	500	1,205	1,568	1,774	2,021	7,464	960
Regions generated by EDA1	13,016	14,602	14,969	14,886	15,468	15,806	776
TPU	1,049	5,046	13,825	14,886	23,864	33,222	11,961

Table 5-7 Comparison of statistics of population data among Large Street Block, regions generated by EDA1 and TPU

5.15.Finding Pareto Front

As mentioned in previous sections, zone design problem is an optimization problem always subject to multiple and conflicting objective functions especially when dealing with real problem. Unfortunately, it is also impossible to obtain a solution that can reach the highest scores of all the conflicting objective functions simultaneously. Typical workaround is to transform the multiple objective functions to a single objective function, which is used in previous chapters. In this chapter, the approach to handle multiple objectives advocated in multiobjective evolutionary algorithms (MOEAs) is introduced. In brief, instead of finding an optimal solution or even a set of optimal solutions, this approach aims to obtain solution set by considering trade-off among the conflicting objective functions without subjective preference. This means even if a solution poorly fits to a particular objective function, it can still obtain a very good overall fitness to the optimization problem as long as it could perfectly fit to other objective functions. The solution set that each solution can give an overall best fitness disregarding how good it can fit to particular objective function is called the Pareto set and its graphical representation is called the Pareto front. In this section, EDA-based algorithm (i.e. EDA1) will be modified by incorporating the above concept of MOEAs, find the Pareto front and rankings of the entire solution. Different from general evolutionary algorithm, MOEAs has special design not only to identify solutions with high fitness value but also maintain the solution diversity along the Pareto front. Furthermore, this section also reveals the behaviour of the EDA based zone design algorithm by visualizing distribution of solutions and its ranking on the landscape of objective functions over generations.

5.16.Combining multiple objective functions

In general, there are two options of either summation of objective functions or product of objective functions to combine multiple objective functions to a single objective functions during the optimization process:

$$g(x) = \sum_i w_i f_i(x) \quad \text{or} \tag{55}$$

$$g(x) = \prod_i f_i(x) \tag{56}$$

In the previous chapter, summation approach is used to combine the values of objective functions because product of the values may create a very big number that may hit the data overflow problem. Besides, it is considered that each objective is of same significance and therefore, no specified weightings are assigned to the objective functions. However, it is noted that there are two difficulties for this approach for handling values of multiple objective functions. Firstly, defining a proper weighting is a very difficult task and may introduce unexpected bias distorting the solution landscape. Further investigation is necessary to understand its effect to the optimization process. Secondly, even if the relative importance among the objective functions is known, changing of weighting may not be consistently changing fitness value of the solution meaning they may not be linearly proportional. In order to determine proper weighting, performing testing with different combination of weightings may be the only method and unavoidable. As a result, this creates another optimization problem but now the target is to optimize weightings.

Again, zone design problem is naturally multiple objective problem. In the view of above, it is good to handle the problem using another perspective that is the concept of Pareto dominance.

5.17.Multi-objective Optimization

Multi-objective optimization aims at finding solutions to optimize k objective functions simultaneously subject to a set of constraints, in which it considers trade-off among the k functions or technically speaking, it allows any possible combination of maximization and minimization of these k functions.

A multiple-objective problem is defined as optimizing (minimize or maximize) a vector of objective functions and it can be described as follows:

$$\mathrm{f(x)} = \left(f_1(x), f_2(x), \dots, f_k(x)\right)\ subject\ to\ g_i(x) \tag{57}$$

As shown in above equation, $x \in \mathbb{Z}^n$ is a solution represented by a n-dimensional vector $x = (x_1, . . ., x_n)$ in a decision space containing all possible x that can satisfy the evaluation of f(x). Besides, $g_i(x)$ represents a set of constraints that defines the set of feasible solutions in the decision space. By applying k objective functions that may be linear or nonlinear, continuous or discrete, quality of the solution vector can be shown in the objective space. Considering a maximization problem, given two solution vectors x and y, when y strongly dominates x or y is said to be Pareto optimal, it means x does not exist such that $f_i (x) \geq f_i (y)$ and it can be expressed as follows:

$$y \prec x \text{ iff } \forall i \, f_i(x) > f_i(y) \tag{58}$$

On the other hand, x is said to be weakly dominated by y if there is at least one of the objective functions that value of y is larger than that of x and it can be expressed as follows:

$$y \preccurlyeq x \text{ iff } \forall i \, f_i(y) \geq f_i(x), \exists i \; f_i(y) > f_i(x) \tag{59}$$

Collection of the Pareto optimal can be plotted on the graph showing a shape of the set is so called Pareto front. Figure 5-31 illustrates the decision space and objective space and their relationship. In the figure, decision space is constructed by the value of the solutions, in which just a small sub set represents feasible solutions. On the other hand, each feasible solutions can be mapped to the objective space and corresponding feasible objective space. The Pareto Optimal set or the Pareto Front are located on the edge of the feasible objective space. Depending on the nature of the optimization problem, location of the Pareto Front is different. In general, when dealing with a minimization optimization problem, the Pareto Front is at the lower left of the feasible objective space. However, when dealing with a maximization optimization problem, the Pareto Front is at the upper right of the feasible objective space.

The Pareto front indicates how the trade-off working among the objective functions. As indicated in the figure, solution x inside the feasible objective space is dominated by y, which is a Pareto optimal. Pareto Optimal Set can be represented by a curve along the boundary of the feasible objective space. Finding it is not only by identifying solutions with highest fitness but also by collecting key and representative solutions that can shape the curve completely on the graph.

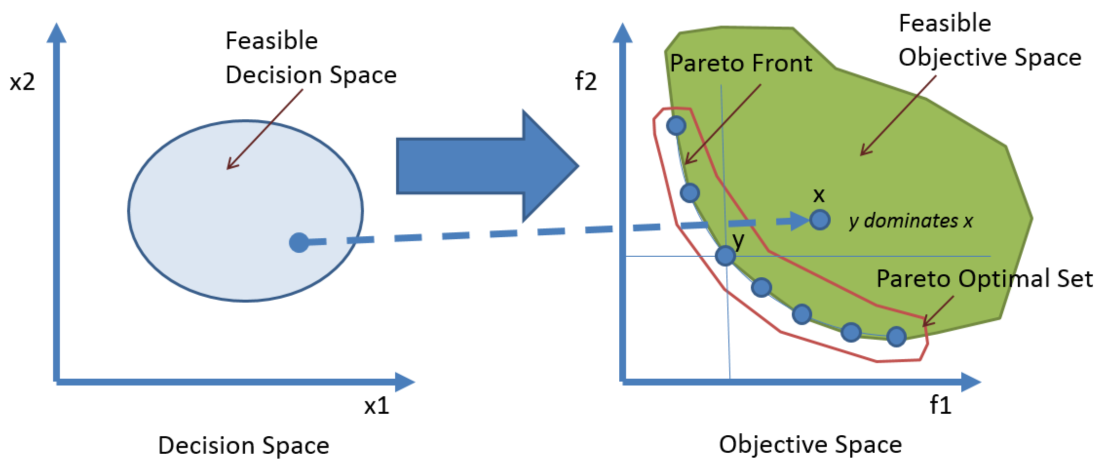


Figure 5-31 Decision Space and Objective Space of a multiple-objective problem

5.18.Non-dominated Sorting Genetic Algorithm II (NSGA-II)

Non-dominated Sorting Genetic Algorithm (NSGA) is one of the well-known evolutionary algorithms specially designed for solving multi-objective optimization problem proposed by Srinivas and Deb (1995). In general, multi-objective evolutionary algorithm returns a set of Pareto optimal for each single generation and is realized by two common features including non-dominated sorting and diversity preservation of the selected non-dominated solutions. Subsequently, Deb et al. (2002) proposed NSGA-II, which is an improved version of NSGA. According to Deb et al. (2002), NSGA-II not only can maintain better spread and thus diversity of the solution, it can also provide better convergence in the obtained non-dominated front. NSGA-II has two major components: fast non-dominated sorting and density estimation of solutions in the population or called crowding distance. Fast non-dominated storing refers to a procedure of sorting solutions in a population into different non-dominated level or ranking. In brief, for each solution, it will be assigned with a domination count, i.e. number of solutions which dominates subject solution, and also the set of solutions that subject solution dominates. Of course, if the domination count is zero, it means it is dominated by nothing and hence subject solution must be a Pareto optimal. As a result, once the sorting based on dominance is done, Pareto optimal can also be identified. Further, density estimation is crucial in the multi-objective optimization and it has a role to maintain and preserve solution diversity, especially when the solutions are in the Pareto Optimal Set. In NSGA, a fitness sharing parameter is required to determine the extent of sharing fitness in the problem for maintaining solution diversity. More precisely, the parameter defines a distance, in which any two solutions within the distance can share each other's fitness. This parameter is no longer required in NSGA-II and is replaced by a crowding distance and a crowding-comparison operator. Given a subject solution, a crowding distance refers to absolute normalized fitness value difference of two adjacent solutions, in which the adjacency is determined by order of solution in each ranking. Normally, the larger crowding distance is preferable because it means the solution is less crowded and more diverse. After assignment of the crowding distance to each solution, a crowding-distance-comparison operator is used to select the solution for running further evolutionary process. With the two pieces of information, i.e. crowding distance and ranking collected from the population, a solution will be selected primarily when it has a higher rank. If two solutions have the same rank, the one with greater crowding distance will be selected.

```
1    For each solution (say p) in the population
     1.1    Define $S_p$ be set of solution which is dominated by solution p
     1.2    Define $n_p$ be the number of solutions which dominates solution p
     1.3    For each solution q in the population but not p
            1.3.1  If p dominates q then add q to $S_p$;
            1.3.2  Else if q dominates p, then $n_p=n_p+1$
2    If $n_p=0$ then rank of p = 1
3    Add p to Front $F_1$
4    End for loop
5    Set i = 1
6    While $F_i \neq 0$
     6.1    $Q = 0$
     6.2    For each solution p in $F_i$
            6.2.1  For each solution q in $S_p$
            6.2.2  $n_q = n_q-1$
            6.2.3  if $n_q = 0$ then rank of q = i + 1
            6.2.4  Add q to Q
     6.3    $i = i + 1$
     6.4    Set $F_i$ equal to Q
7    End while loop
```

Algorithm 5-3 Fast non-dominated sort in NSGA-II

The procedure for the fast non-dominated sorting, crowding distance calculation and overall evolutionary procedure are illustrated in Algorithm 5-3, Algorithm 5-4 and Algorithm 5-5 respectively. In fact, NSGA-II relies on both solution ranking and crowding distance for selecting solutions so that it can be retained in population for next generation and subsequent evolution process.

```
1   Compute number of solutions (len) belonged to the rank
2   For each solution i in each rank, set $I[i]_{distance} = 0$
3   For each objective m
    3.1   Sort I based on m
    3.2   Set $I[1]_{distance} = I[len]_{distance} = inf$ so that lower and upper points will always be selected
    3.3   For 2 to len-1
          3.3.1   Compute $I[i]_{distance} = I[i]_{distance} + (I(i+1).m – I[i-1].m/(max(f.m)-min(f.m))$
4   End for loop of m
5   End for loop of i
```

Algorithm 5-4 Crowding distance assignment in NSGA-II

```
1   Define $R_t = P_t \cup Q_t$, in which $P_t$ and $Q_t$ refer to past population and current offspring
    respectively. Invalid solutions must be removed from $P_t$ and $Q_t$ to avoid growing of the
    solutions
2   Get Front F by fast non-dominated sort ($R_t$)
3   While $|P_{t+1}|+|F_i|\leq N$
    3.1   Compute Crowding distance assignment for solution in $F_i$
    3.2   Include solutions in $F_i$ to $P_{t+1}$
    3.3   $i = i + 1$
4   End while loop
5   Sort $F_i$ in descending order using crowding-distance-comparison operator
6   Include $1:(N-|P_{t+1}|)$ solutions in $F_i$ to $P_{t+1}$
7   Create $Q_{t+1}$ using remaining evolutionary operations
```

Algorithm 5-5 Overall evolutionary process of in NSGA-II

In addition, another point that is worth to know about NSGA-II is that both current and past populations are used in each generation as solution candidates. This double-size population ensures sufficient amount of solutions for finding the Pareto set. This is significant because number of feasible solutions may be limited, especially when it is generated by random process in the initialization stage, in which infeasible solutions should be removed from the population to prevent misleading the evolutionary process.

5.19.Applying NSGA-II to Zone Design Problem

According to NSGA-II, a variant of EDA1 is developed by adding the niching processes that discussed in previous section. These processes are including fast non-dominated sort and assignment of crowding distance, that would be added before the model learning and sampling steps of EDA1. Same as the original EDA1, two criteria of equal population and compactness are used to solve a 6-region zone design problem, in which it is considered that the problem size is sufficient enough to demonstrate how the NSGA-II works among the 3-region to 10-region problem sets. Due to the additional procedures, the variant will take longer time to complete the same number of generations when compared with the original EDA1. The major components of NSGA-II are a series of sorting, elitist selection and ranking procedures. To allow observation of the change of population in term of ranking and fitness values over the generations, the algorithm is terminated at specified generation, i.e. 3 generations and 10 generations. With different number of generations, all tests will use the same population size, i.e. 1,000, and rate of mutation, i.e. 0.2.

5.20.Results

Figure 5-32 shows the results of the first test with 3 generations. Each point represents a single solution and its colour represents which rank it belongs to. In this case, the algorithm can identify four different ranks of solution. Since this is a maximizing problem, optimal points should be at the upper right area of the graph that is far away from the origin. The red points are the solution having the highest fitness values among others forming the Pareto Optimal Set, while the red line connecting them represents the Pareto Front.

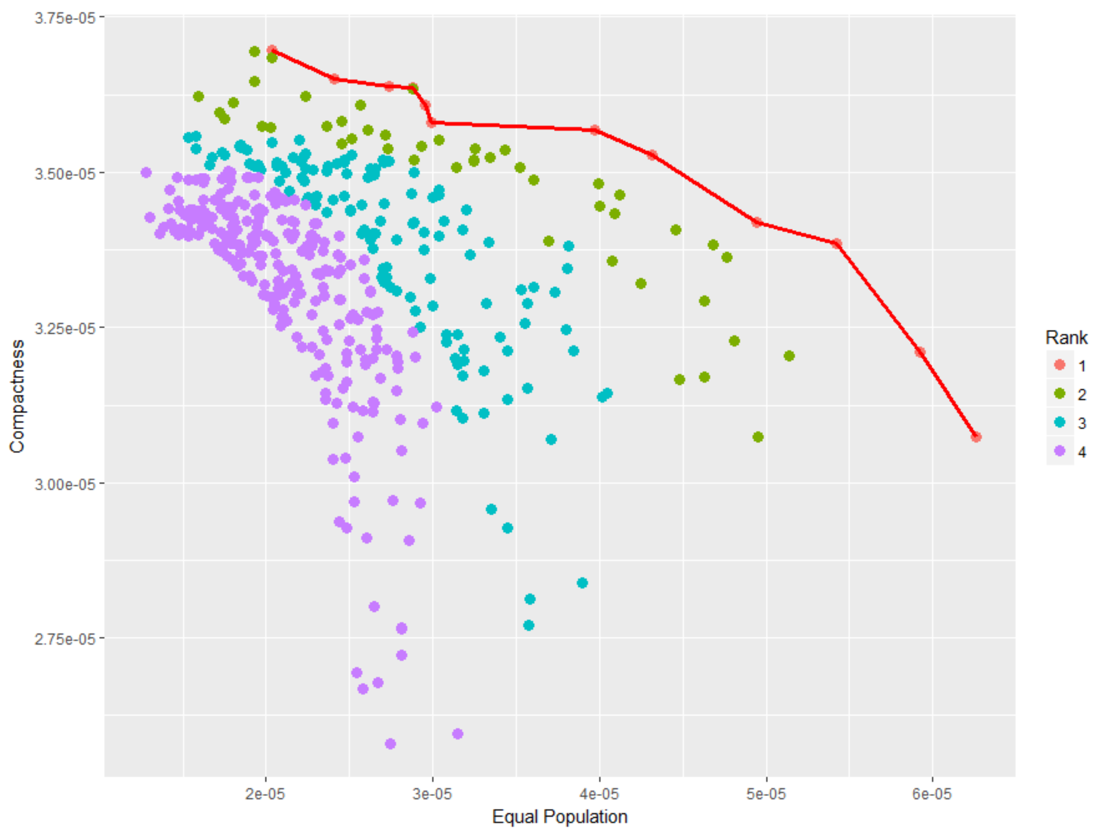


Figure 5-32 Results of a 6-region Zone Design Problem solved by variant of EDA1 after 3 iterations

It is observed that density of the solutions at all ranks are relatively high at area of high compactness. The reason is due to the use of distance-based approach for population initialization. However, this situation should be changed after several generations subject to the objective functions. Besides, diversity among highest fitness solutions (i.e. solutions along the red line) has a good distribution between the two conflicting criteria. This shows that trade-off between the criteria is working properly. In this test, as expected, there are many identical solutions created in each generation. After the three generations, 12 unique solutions can be identified in the Pareto Optimal Set. Appearance of the zone designs at the starting point, middle point and ending point of the Pareto front are shown in Figure 5-33, Figure 5-34 and Figure 5-35 respectively.

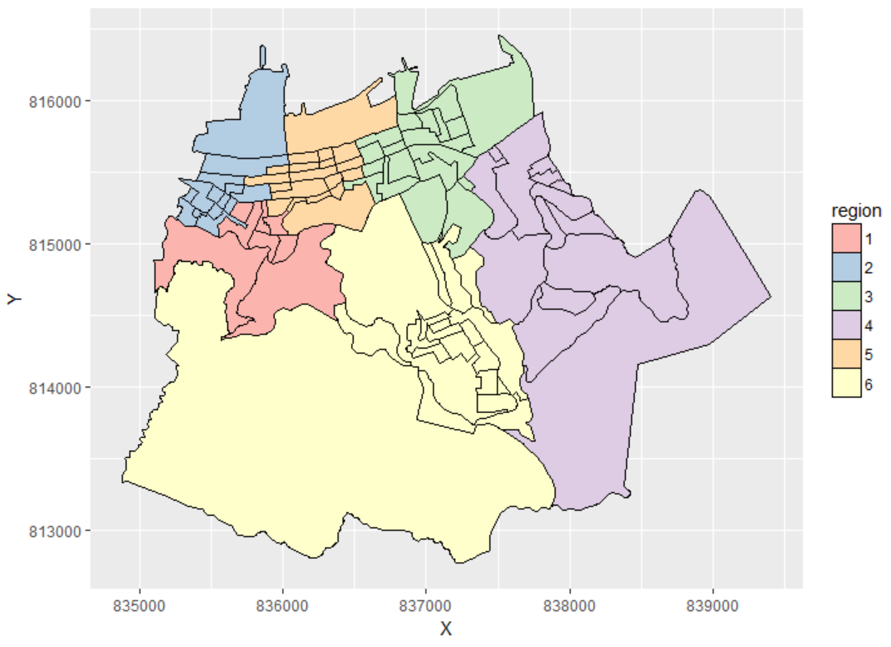


Figure 5-33 Solution with the highest value of compactness but lowest value of equal populaton

As shown in Figure 5-33, although each region is of different sizes, shape of the regions is considerably balanced and circular without much elongated appearance making it has a highest value of compactness. It is also because of the smaller size of the zonal units in the Northern portion of the test area, it provides more flexibility forming compact region. To maintain good compactness, two larger regions are formed at South. On the other hand, fitness to achieve equal population is compromised making it can only attain the lowest value of equal population among the solution on the Pareto Front. However, this is still one of the best solutions found.

Figure 5-34 shows the balanced solution between compactness and equal population. It is obviously that some regions are slightly elongated when compared with Figure 5-33 and in normal situation, this is the best of the best solutions.

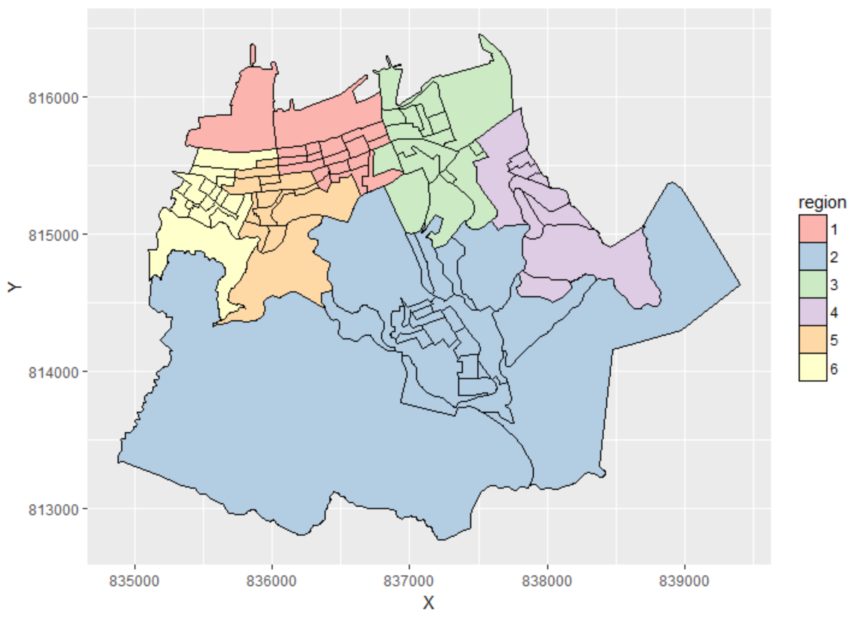


Figure 5-34 Solution with the highest value of compactness and highest value of equal population

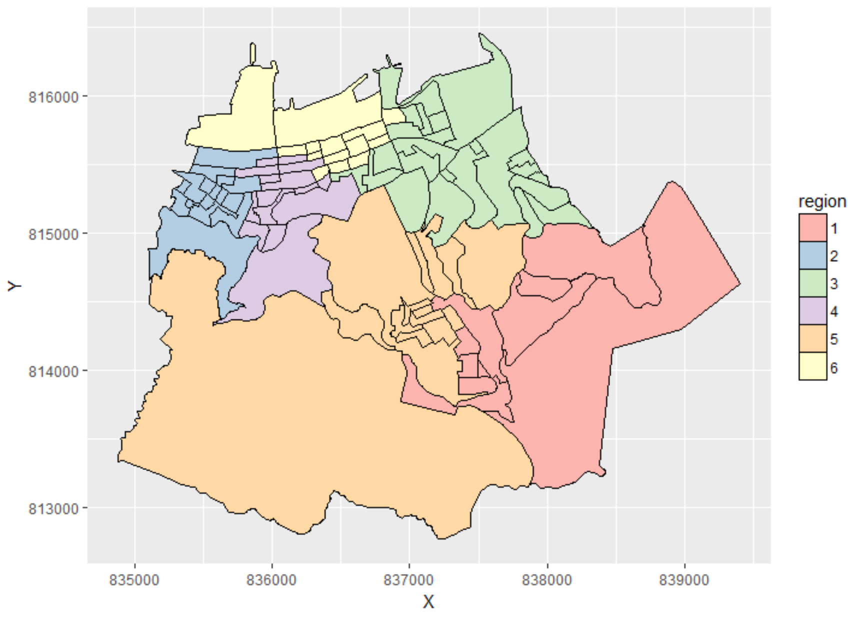


Figure 5-35 Solution with the lowest value of compactness and highest value of equal population

Figure 5-35 shows the result with the lowest compactness but higher quality of equal population. Shape of the regions at Northern part is slightly elongated. Except an small odd extension found at the region at Southeast, the overall configuration is similar to the first case. When the number of generations is increased, solutions converge and number of ranks decrease. Figure 5-36 shows the Pareto Front (in red colour) after 10 generations of the 6-region zone design problem. There are only two ranks and the entire population is shifted to the area with relatively higher value of equal population. The Pareto Front after three generations are shown in blue colour for easy reference. According to the result and comparison, overall solutions are improved and moved to higher fitness area but less compactness.

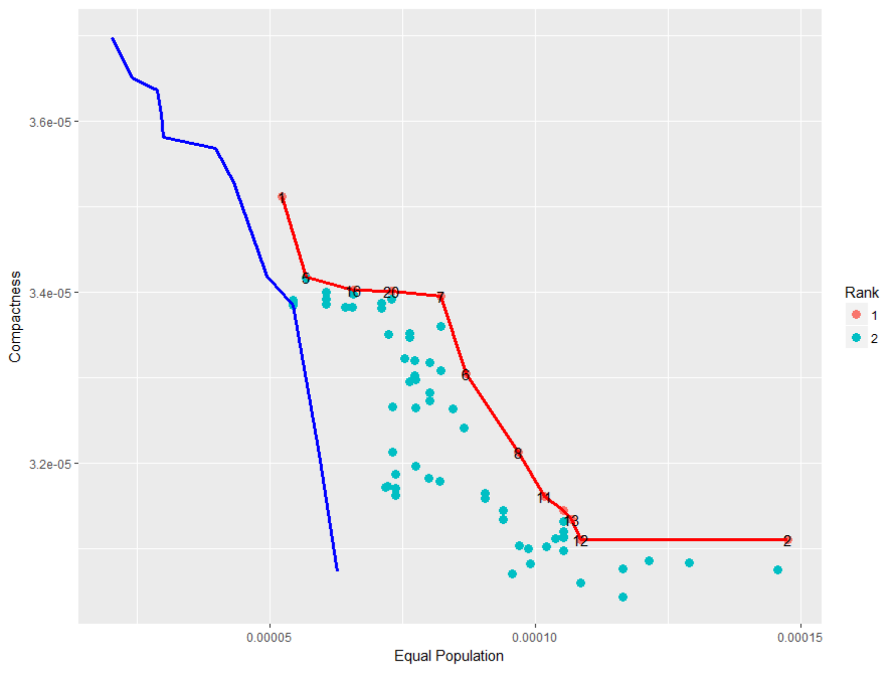


Figure 5-36 Results of a 6-region Zone Design Problem solved by variant of EDA1 after 10 generations. Blue line and red line represents the Pareto Front after 3 generations and 10 generations respectively

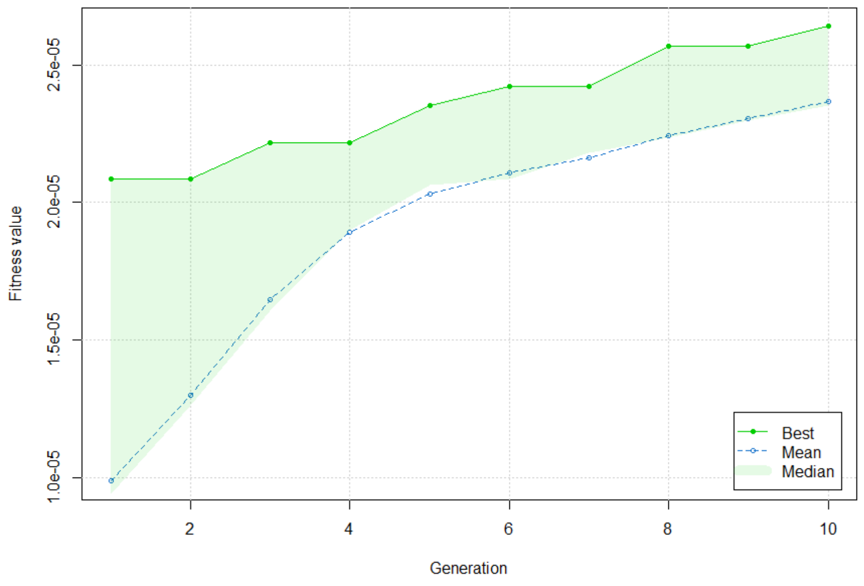


Figure 5-37 Fitness growth of the 6-region zone design problem

By observing the fitness growth of the zone design problem as shown in Figure 5-37, overall fitness including best solution, mean and median of the population is gradually increased. Meanwhile, it seems that overall fitness is yet to be converged after 10 generations so that better solutions could be generated by further generations. Due to the preservation of solution diversity, there is a big gap maintained between the best solution and the mean quality solution.

Based on the results obtained from the Pareto Optimal Set as shown in previous figures, it allows decision makers to see all the best options giving an opportunity to better understand the solution with different scenarios. When performing optimization of more than two criteria, the same approach can be adopted. However, showing high dimensional data using graph may be difficult to interpret and understand. One approach could be to present pairs of criteria at a time. This is a pragmatic approach for finding solution for a multiple-objective optimization problem, simply by showing and understanding all the options, then choose the appropriate one to go for.

5.21.Conclusion

This chapter has demonstrated the use of EDA based zone design algorithm with two different solution encodings as well as a GA based zone design for comparison purpose. EDA is a powerful optimization method based on statistical model to realize its learning capability.

As discussed, EDA is a generic optimization and it can be applied to various optimization problems. However, for handling spatial data and relevant spatial modelling problem, additional considerations on how to handle spatial relationship are essential. Otherwise, it will create infeasible solutions that can greatly degrade the search performance making it very hard to identify feasible solution during the evolution process.

With the foundation built for handling spatial data, this spatial-enabled EDA can solve different zone design problems by applying different sets of objective function. In this study, it has demonstrated to use objective functions of equality (inter-homogeneity) and homogeneity (intra-homogeneity) as well as radial compactness to solve a typical zone design problem. Next, EDA is also applied to find cluster with quality measures on homogeneity (intra-homogeneity) and radial compactness.

In the tests, group-number encoding is adopted, in which it is a direct encoding method for modelling solution of the zone design problem. Although this is easy to understand, it has a drawback that the search space is extremely large and its size can grow exponentially with proportional to the number of zonal units. In addition, the encoding suffers from redundancy. Due to these concerns, an alternative encoding called region-centre encoding method is used for comparison. Altogether, three different implementations are presented, i.e. EDA with group-number encoding, EDA with regional centre encoding and GA with regional centre encoding. Besides, it is shown that mutation is an essential ingredient for improving the EDA searching for higher quality solutions. The main reason is that mutation enables the search jumping to different areas within the solution space for global optimums. More importantly, this works effectively even after the statistical model learnt are converged.

Tests in this study also indicated that results generated by EDA are comparable with the GA implementation with the region-centre encoding method in term of solution quality. This is largely due to the flexibility of the direct encoding method (i.e. group-number encoding) that can model the solution very well with high flexibility. All the test results also indicate that quality of solution generated by EDA is consistently better than that generated by EDA with the region-centre encoding method.

However, when processing time is considered, EDA will not beat GA because computational complexity in EDA are definitely higher, in which both the learning and sampling functions would draw most of the computational time during processing.

Although GA is also a kind of population based evolutionary optimization algorithm, the way to evolve solutions is very different when compared with EDA. In GA, it treats solution individually and it will not abstract any information from the population but relaying on the magic of crossover and mutation to create new solutions hoping to improve it over generations. In particular, it is difficult to understand and explain how the crossover operator works so that overall solution quality can be improved. However, GA requires relatively less computational power for its searching mechanism and in general it can coverage solution in short time. This may due to the standard operations in the genetic operators that does not consider the inter-dependency among variables in the solution during searching. On the other hand, EDA relies on a probabilistic model to work, which can improve iteratively by summarising data obtained from the population and the concept is more intuitive.

By adjusting the objective functions, performance between EDA with direct encoding method and SKATER are compared. Test results indicate that EDA outperforms SKATER consistently with and without consideration of compactness. It is believed that this is due to the minimum spanning tree adopted in SKATER that restricts the flexibility in later grouping of zonal units. To improve the performance of SKATER, it could generate a new minimum spanning tree whenever a region is found so that it can truly reflect the configuration at all time.

Finally, way to apply advanced multiple-objective evolutionary algorithm to the EDA based zone design for finding the Pareto front is presented. With this technique, measure of multiple objectives is no longer consolidated to a single value but a set of individual fitness values that could be presented in high dimensional graph. A multiple-objective evolutionary algorithm called NSGA-II is adopted. NSGA-II is composed of three major components namely, the fast non-dominated sort, crowding distance assignment and the crowding-comparison operator. The key of this approach is that all non-dominated solutions can be identified disregarding any subjective preference while the crowding distance is parameterless and it can maintain good diversity of solution along the Pareto Front. It is demonstrated that EDA-based zone design can easily be adopted to the multiple-objective approach. Therefore, applying EDA to spatial problem is not difficult. By using Markov Networks for modelling the multiple dependency in spatial data, spatial analysis can be realized by selecting an encoding method that can fully model the solution and applying appropriate objective functions.

Chapter 6 – Conclusions

6.1. Concluding Remarks

To my knowledge, this study is the first time to apply EDA to tackle zone design problem. EDA is an advanced evolutionary algorithm that makes use of graphical probabilistic model and sampling technique to learn and tackle hard optimization problems. This study has shown that EDA based Zone Design can be used to solve spatial optimization problem and it is capable of finding best solutions that outperforms GA, which is a well-known evolutionary algorithm, and "SKATER", which is a deterministic algorithm specially designed for regionalization and finding intra-homogenous regions.

It is believed that zone design has very broad application area. Recalling the idea of zone design, it is merely an aggregation process of zonal data subject to certain criteria and the result is a set of regions revealing the visual appearance of the criteria. It is similar to the idea of image recognition. However, zone design is much difficult because it is an unsupervised learning. Although zone design is a simple idea, it is very powerful and the significance is that it gives flexibility to the analyst so that he/she can adjust what to reveal from the data. This is completely difference from ordinary spatial analysis using spatial operators for selecting data. The objective function in zone design is more statistical oriented focusing on evaluation of spatial property ranging from relationship between an individual zonal unit and its neighbourhood to qualifier at regional level.

This study further confirmed that zone design can be used as a spatial analysis tool by applying EDA. To do this, this study has fully reviewed the definition of zone design, formation of zonal data, discussed its computational complexity and common approaches to cope with the problem. Next, population-based evolutionary algorithms including GA and EDA are reviewed. The reason to review GA because it is the predecessor of EDA and most of the data flow in relation to evolving solutions is developed from GA. Furthermore, this study has also provided an in-depth review on various dependency models that could be used in conjunction with EDA for modelling variables in given dataset. As a result, an EDA based zone design algorithm is developed. Different from typical EDA, the EDA based zone design algorithm finds optimal solutions with respect to spatial continuity. To realize this, local Markov property is used to do sampling of new solution. Furthermore, feasible solution vector and detection of cut-node are proposed to ensure spatial continuity of each region. Solution encoding is a must in evolutionary algorithm. In this connection, two different encoding methods are developed. One is the group-number encoding that provides maximum flexibility for composing a zone design because components of this encoding are at zonal unit level but it more complex due to larger number of variables and suffers from redundancy. Another is the regional centre method. It's simplicity is gained from using regional centres to abstract a zone design solution instance. However, decoding is necessary to form an actual solution using Euclidean distance as proximity measure for matching zonal units to the nearest regional centre. It is noted that proximity measure is sensitive to the size of zonal units. Uneven size of zonal units could generate region that comprises isolated parts.

To test the EDA based Zone Design algorithm, Street Block data covering Wan Chai district of Hong Kong and respective census data are developed. The test data is finally based on Large Street Block Group defined by the census data and this the finest data available in the public domain. For comparison purpose, a GA based zone design algorithm is developed for comparing the best solution that it can find with that provided by the EDA based zone design. On the other hand, after modifying the objective function, the EDA based zone design is also compared with a regionalization algorithm called SKATER for a clustering problem. It can be concluded that encoding method affects the search performance greatly. It is further confirmed that selection of encoding method is crucial in evolutionary algorithm because right encoding method should be capable of generating every instance of feasible solutions so that it will not limit the exploration capability of the search. Meanwhile, it is also noted that EDA has good exploitation capability such that average fitness of the population can be improved gradually even though a complex encoding is adopted. In short, high quality solution can be obtained by the EDA based zone design algorithm and respective solution quality outperforms that found by GA and SKATER.

Furthermore, a variant of the EDA based zone design is developed to cope with multiple criteria optimization problem by finding the Pareto Front of the solutions. This is a pragmatic approach for handling multiple-objective problem that allows decision makers to view all the best solutions and then choose one out of the promising solution set. This is completely different from traditional approach that only a single result will be presented as the optimal answer.

Besides finding the optimal solution, after each run of the EDA-based zone design algorithm, a probabilistic model showing the how likely each zonal unit would be assigned to which region is obtained. This model could be used to predict the most probable region that a given zonal unit could be assigned to. In addition, one would interest to know if there is any zonal unit having high variation of the regional assignment, and therefore further investigation can be conducted to find out the cause.

Nevertheless, EDA has shortcomings. The most obvious one is its high complexity and computational requirement, in particular, in the parts that learn relationship among variables from the data and generate new solutions using sampling techniques. Inevitably, EDA takes longer time to converge solutions when compared with that of GA simply because GA does not learn. However, it is believed that this issue could be solved with advent of computer technologies.

Therefore, it is recommended that further advancement can be investigated in the following areas:

i. Use parallel processing to improve performance – EDA is computation intensive method. It will be beneficial from parallel computing by distributing solution generation and evaluation simultaneously in different processors.
ii. Introduce machine learning technique such as Conditional Random Field to detect characteristics of spatial data, and alternatively, adopt interactive learning approach to improve the solution quality. However, how to model the spatial problem adopting to the technique is a challenging question.

iii. Cloud computing – It is a popular technology in recent years that makes use of almost unlimited computing resources on the Internet. That unlimited resources refers to memory, computing power or even data storage. In view of the computation intensive nature of the EDA method and complexity of spatial data in general, using cloud computing will be a future trend because computing resources may not be the major concern and we can focus on modelling and analysing of the underlying data.

Although this study has demonstrated a new way to realize zone design as a spatial analysis tool, without the proper zonal data, it is still hard to reach meaningful results in real life applications. It is believed that machine learning or approach with learning capability could be a new trend in analysing spatial data. Ultimately, it is expected that EDA or similar approach with learning capability could be further developed and become a standard analysis tool in geographic information science.

References

Alden, M.E. 2007. *MARLEDA: Effective Distribution Estimation Through Markov Random Fields,* PhD Dissertation, The University of Texas at Austin.

Altman, M. 1997. The computational complexity of automated redistributing: Is automation the answer? *Rutgers Computer & Tech. L.J.* 23(1): 81-142.

Alvanides, S., Openshaw, S. and Macgill, J. 2001. Zone design as a spatial analysis tool. *Modelling Scale in Geographical Information Science* edited by Tate, N.J. and Atkinson, P.M. Chichester: Wiley, 2001. 141-157.

Arora, S. and Barak, B. 2009 *Computational Complexity: A Modern Approach*. Cambridge University Press. New York, NY, USA.

Assunção, R.M., Neves, M.C., Câmara, G. and Da Costa Freitas, C. 2006. Efficient regionalization techniques for socio - economic geographical units using minimum spanning trees. *International Journal of Geographical Information Science* 20 (7): 797–811.

Bação, F., Lobo, V., and Painho, M. 2005. Applying genetic algorithms to zone design. *Soft computing*. 2005(9): 341-348.

Baluja, S. 1994. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning, *Technical Report No. CMU-CS-94-163*, Carnegie Mellon University, Pittsburgh, PA, 1994.

Boscoe, F.P. and Pickle, L.W. 2003. Choosing geographical units for choropleth rate maps, with an emphasis on public health applications, *Cartography and Geographic Information Science*, 30 (2003):237-248.

Bribiesca, E. and Montero, R.S. 2009. State of the art of compactness and circularity measures, *International Mathematical Forum*, vol. 4, no. 27, 1305–1335, 2009.

Casella, G. and George, E.I. 1992. Explaining the Gibbs Sampler, *The American Statistician*, vol. 46, Issue 3, 167-174, 1992.

Cockings, S and Martin, D. 2005. Zone design for environment and health studies using pre-aggregated data, *Social Science & Medicine* 60 (2005) 2729-2742.

Correa E.S., Steiner, M.T.A., Freitas, A.A., Carnieri, C. 2001. A genetic algorithm for the P-median problem. *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-2001*, 1268-1275. Morgan Kaufmann.

Cooper, G.F., and Herskovits, E. 1992. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning*, 9, 309-347.

De Bonet, J.S., Isbell, C.L. and Viola, P. 1997. MIMIC: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, Mozer, M.C., Jordan, M.I. and Petsche, T. (Eds.), 1997, vol. 9, 424.

Deb, K., Pratap, A., Agarwal, and Meyarivan, T. 2002. A Fast and Elitist Mulitobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, Vol.6, No.2, April 2002.

Eddelbuettel, D. 2013. *Seamless R and C++ Integration with Rcpp,* UseR! Series, Springer, June 2013, 220 pages.

Eddelbuettel, D. and Francois, R. 2011. Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, Volume 40, Issue 8, April 2011.

Fjällström, P.O. 1998. Algorithms for graph partitioning: A survey. *Linköping Electronic Articles in Computer and Information Science*, 3(10), 1998.

Goodchild, M. and Gopal, S. 1989. *Accuracy of Spatial Database*, Taylor & Francis, 1989

Grünwald, P.D. 2007. *The Minimum Description Length Principle*, MIT Press, June 2007.

Guo, D. 2008. Regionalization with dynamically constrained agglomerative clustering and partition (REDCAP). *International Journal of Geographical Information System* Vol.22, No.7, July 2008, 801-823.

Guo, D. and Wang, H. 2011. Automatic Region Building for Spatial Analysis. *Transactions in GIS*, 2011 15(s1) 29-45.

Haining, R., Wise, S., and Ma, J. 2000. Designing and Implementing Software for Spatial Statistical Analysis in GIS Environment. *Journal of Geographical Systems,* (2000) 2:257-286.

Harik, G., Lobo, F.G., and Goldberg, D.E. 1998. The compact genetic algorithm. *Proceedings of the International Conference on Evolutionary Computation (ICEC'98)*, Piscataway, NJ, 1998, 523–528.

Harik, G. 1999. Linkage learning via probabilistic modeling in the ECGA. University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, *IlliGAL Report No. 99010*, 1999.

Hauschild, M., and Pelikan, M. 2011. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation* 1(3): 111-128(2011)

Heckerman, D., Geiger, D. and Chickering, M. 1994. Learning Bayesian networks: The combination of knowledge and statistical data. Microsoft Research, Redmond, WA, *Technical Report MSR-TR-94-09*, 1994.

Henrion, M. 1988. Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling, Lemmer, J. F. and Kanal, L. N. (Editors), *Uncertainty in Artificial Intelligence 2*, North-Holland, 149-163.

Hess, S.W., Siegfeldt, J.B., Whelan, J.N., & Zitlau, P.A. 1965. Nonpartisan political redistricting by computer. *Operations Research*, 13(6), 998–1006.

Jain, A.K. and Dubes, R.C. 1988. *Algorithms for clustering data Prentice-Hall, Inc.* Upper Saddle River, NJ, USA.

Kim, J., Hwang, I., Kim, Y.H and Moon, B. 2011. Genetic Approaches for Graph Partitioning: A Survey. *Genetic and Evolutionary Conference Computation Conference 2011*, July 12–16, 2011, Dublin, Ireland.

King, D.M., Jacobson, S. H., and Sewell, E.C. 2014. Efficient geo-graph contiguity and hole algorithms for geographic zoning and dynamic plane graph partitioning. *Mathematical Programming*, 149(1-2), 425-457.

Koller, D. and Friedman, N., (2009) *Probabilistic Graphical Models: Principles and Techniques*, MIT Press.
LandsD (2016)
URL:http://www.landsd.gov.hk/mapping/en/metadata/metadata/web/data/others/pland_tpu.html, Retrieved in May 2016.

Li, W.; Goodchild, M., and Church, R. 2013. An Efficient Measure Of Compactness For Two-Dimensional Shapes And Its Application In Regionalization Problems. *International Journal of Geographical Information Science* 2013, 27, 1227-1250.

Lipowski, A. and Lipowska, D. 2011. Roulette-wheel selection via stochastic acceptance, *Physica A* 391 (2012):2193-2196.

Lloyd, C. 2010. *Spatial data analysis: an introduction for GIS users*. Oxford University Press.

Lynch, S.M. 2007. *Introduction to Applied Bayesian Statistics and Estimation for Social Scientists*, Springer.

Maceachren, A.M. 1985. Compactness of Geographic Shape: Comparison and Evaluation of Measures. *Geografiska Annaler, Series B, Human Geography*, Vol.67, No.1, 53-67.

Morris, R.D. and Munasinghe, R.L. 1993. Aggregation of existing geographic regions to diminish spurious variability of disease rates. *Statistics in Medicine*, 12 (1993): 1915-1929.

Mühlenbein, H. 1997. The equation for response to selection and its use for prediction. *Evolutionary Computation*, vol. 5, no. 3, 303–346.

Mühlenbein, H., Mahnig, T. and Rodriguez, A.O. 1999. Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics*, vol. 5, pp. 215–247, 1999.

Openshaw, S. 1977. Optimal zoning systems for spatial interaction models. *Environment & Planning A*, 9, 169-184.

Openshaw, S. 1976. A geographical solution to scale and aggregation problems in region-building, partitioning and spatial modeling. *Transactions of the Institute of British Geographers*, New Series, 2(4): 459-472.

Openshaw, S. 1983. The Modifiable Areal Unit Problem, Concepts and Techniques in *Modern Geography No.38, Geo Books*. Norwich.

Openshaw, S. and Rao L. 1995. Algorithms for reengineering 1991 census geography. *Environment & Planning A*, 27, 425-446.

Pelikan, M. 2005. *Hierarchical Bayesian Optimization Algorithm. Toward a New Generation of Evolutionary Algorithms.* Studies in Fuzziness and Soft Computing. Springer, 2005.

Pelikan, M., Goldberg, D.E., and Cantu-Paz, E. 1999. BOA: The Bayesian optimization algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-1999*, volume I, 525-532, Morgan Kaufmann.

Pelikan, M., Goldberg, D.E., and Lobo., F. 2002. A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, 21(1):5-20.

Pelikan, M. and H. Mühlenbein, H. 1999. The bivariate marginal distribution algorithm. *Advances in Soft Computing—Engineering Design and Manufacturing,* London, Roy, R., Furuhashi, T., and Chawdhry, P.K. (Eds.), 1999, 521–535.

R 2016a. https://www.r-project.org/ retrieved in May 2016.

R 2016b. https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/mtcars.html retrieved in May 2016.

R 2016c. https://cran.r-project.org/web/packages/Rcpp/Rcpp.pdf, retrieved in July 2016.

Ricca, F., Scozzari, A., and Simeone, B. 2011. Political districting: From classical models to recent approaches. *Annals of Operations Research*, 9(3):223-254.

Ricca, F., & Simeone, B. 2008. Local search algorithms for political districting. *European Journal of Operational Research* 189 (2008):1409-1426.

Roger, S.B., Edzer, J.P., and Gómez-Rubio, V. 2008. *Spatial Data Analysis with R*, Springer. August 2008.

Rothlauf, F. 2006. *Representations for Genetic and Evolutionary Algorithms*, Springer.

Roure, J., Larranaga. P., and Sanguesa, R. 2002. An Empirical Comparison Between K-Means, GAs and EDAs in Partitional Clustering. *Estimation of distribution algorithms : a new tool for evolutionary computation* / edited by Pedro Larranaga, Jose A. Lozano. Boston : Kluwer Academic Publishers, c2002.

Sangkavichitr, C., and Chongstitvatana, P. 2010. Fragment as a Small Evidence of the Building Blocks Existence, *Exploitation of Linkage Learning in Evolutionary Algorithms.* Chen, Ying-ping (Ed.), Berlin : Springer, c2010.

Santana, R. 2003. A Markov Network Based Factorized Distribution Algorithm for Optimization, in 14th European Conference on Machine Learning, Cavtat-Dubrovnik, Croatia, September 22-26, 2003, *Proceedings. LNCS (LNAI)*, Lavrac, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.), vol. 2837, 337-348, Springer, Heidelberg.

Santana, R. 2005. Estimation of distribution algorithm with Kikuchi approximations. *Evolutionary Computation* 13(1): 67-97.

Santana, R., and Shakya, S. 2012a. Probabilistic Graphical Models and Markov Networks. *Markov Networks in Evolutionary Computation*, Siddhartha Shakya and Roberto Santana (Eds.), 2012, 3-17.

Santana, R., and Shakya, S. 2012b. MOA - Markovian Optimisation Algorithm in *Markov Networks in Evolutionary Computation*, Siddhartha Shakya and Roberto Santana (Eds.), 2012, 39-51.

Scrucca, L. 2013. GA: A Package for Genetic Algorithms in R. *Journal of Statistical Software*, [S.l.], v. 53, Issue 4, 1 - 37, Apr. 2013.

Shakya, S. and McCall, J. 2007. Optimization by Estimation of Distribution with DEUM framework based on Markov Random Fields. *International Journal of Automation and Computing* 4, 262-272.

Shlens, J. 2014. *Notes on Kullback-Leibler Divergence and Likelihood Theory*. URL: http://arxiv.org/pdf/1404.2000v1.pdf

Srinivas, N. and Deb, K. 1995. Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3): 221-248.

Tasnádi, A. 2011. The Political Districting Problem: A Survey*. *Society and Economy* 33(2011)

Tavares-Pereira, F., Figueira, J.R., Mousseau, V., and Roy, B. 2007. Multiple criteria districting problems – The public transportation network pricing system of the Paris region. *Annals of operations research*. 2007(154): 69-92.

Tobler W. 1970. A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46(2): 234-240.

UCI 2016. *Machine Learning Repository, Center for Machine Learning and Intelligent Systems*, http://archive.ics.uci.edu/ml/datasets/Iris, Retrieved in May 2016.

Xiao, N. 2008. A Unified Conceptual Framework for Geographical Optimization using Evolutionary Algorithms. *Annals of the Association of American Geographers*, 98(4), 795-817.

Xiao, N., Bennet, D.A., Armstrong, M.P. 2002. Using evolutionary algorithms to generate alternatives for multiobjective site-search problems. *Environment and Planning A*, 2002 (34), 639-656.

Young, H.P. 1998 Measuring the compactness of Legislative District, *Legislative Studies Quarterly*, Vol.13, No.1 (Feb., 1998), 105-115.