Optimization Approaches to Political Redistricting Problems

Dissertation

Presented in Partial Fulfillment of the Requirements for The Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Myung Jin Kim, M.A.

Graduate Program in Geography

The Ohio State University

2011

Dissertation Committee:

Ningchuan Xiao, Advisor

Morton O'Kelly

Mei-Po Kwan

Abstract

Political redistricting is the redrawing of political boundaries based on a set of criteria such as population equality, minority representation, contiguity, and compactness. Redistricting is a necessary process because population often changes over time and across space. From population shifts between states, each state may gain or lose seats. Also, population growth is different within a state. Based on census data, redistricting is normally taking place every 10 years. Also, the congressional district should be redrawn to make populations strictly equal. In the 2000 round of redistricting, 25 states require a redistricting plan with strict population equality, that is, a perfect plan with population deviation of 0.

These problems are difficult because the number of feasible redistricting plans is exponentially increased with the problem size. Also, several redistricting criteria must be satisfied at the same time, and it is difficult to formulate essential contiguity requirement in a mixed integer programming. A strict equal populated redistricting plan is intractable to solve.

The main purpose of this dissertation is to develop optimization approaches to political

redistricting focusing on a strict equal population and contiguity and is to compare them with the existing researches. This dissertation develops two types of exact optimization models to political redistricting based on recent advances in solving land acquisition problems. The new exact models successfully formulate contiguity requirement and satisfy a strict equal population. They are compared with the existing exact model. Computational experiments show that the exact models face computational challenges for large data even though contiguity and a strict equal population are successfully formulated in a mixed integer program. Then, this dissertation moves to implement two different heuristic optimization models, which efficiently finds high-quality solutions. They are evaluated using existing data sets for comparisons. Throughout computational experiments, it is clearly known that all of the heuristics efficiently find near-optimal solutions, and among them the Give-And-Take greedy algorithm shows such efficiency even for the large size problems. Also, all of the heuristics show higher population equality than the existing plan of Iowa in 2000 and Give-And-Take greedy algorithm finds a plan with the highest population equality.

Furthermore, this dissertation focuses on finding various redistricting plans which are different from the given plan but have similar or better population. These can be solved applying the Give-And-Take greedy algorithm, which starts from the given plan as an initial solution. Computational results have shown that for the given plan, the application of Give-And-Take greedy algorithm discovers lots of different spatial shapes but similar (better) or same population of the given plan.

Dedication

To my parents (Soon Ja Han and Hyung Doo Kim), my husband (Seok Kang) and my daughter (Dayoon Kang)

Acknowledgments

I would like to express my sincere gratitude to my advisor, Dr. Ningchuan Xiao for his endless guidance, assistance, encouragement, and consistent support throughout my graduate study at The Ohio State University. He understands that I have been doing my dissertation in Korea for a long time. Without his help and support, this dissertation would not be possible in my life. He has always given to me valuable instructions to make me get into the right track. He is now my role model in my life.

I would like to give my thanks to committee members, Dr. Mei-Po Kwan and Dr. Morton O'kelly. They give me valuable comments to complete my dissertation works. I also would like to thank Dr Larry Brown and Bryan Mark for the practice of the oral exam. I would like to thank Dr. Youngmin Lee in Social Studies at Ewha Womans University to keep continuing a work.

At last, I would like to give my special thanks to my husband (Seok Kang) and my daughter (Dayoon Kang) who support me and are the most precious in my life.

Vita

1999 ....B.A. Geography-Social Studies, Ewha Womans University

2002 ....M.A. Geography-Social Studies, Ewha Womans University

2003 to present ....Geography, The Ohio State University

Field of Study

Major Field: Geography

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

This chapter introduces issues of political redistricting problems. The first section includes research background of political redistricting problems are and discusses the research question of this dissertation. The second section describes research purpose and the third section explains research organization.

1.1 Research background

In the United States, representatives are elected at several levels of government (Morrill 1981). U.S. senators are elected at large from states and House representatives are elected from congressional districts of about equal population sizes. Within states, senators and representatives are elected to legislatures from a structure of districts unrelated to the congressional districts. Each state tries to redraw district boundaries based on the enacting plan created by a commission or a court. Redrawing of congressional district boundaries within the states normally takes place every 10 years on the basis of the population census (Mehrotra et al. 1998). In the 2000 elections, the U.S. House of Representatives are apportioned 435 seats, which were decided by the Reapportionment

Act of 1929, among the 50 states with an average population of 646,952[1]. The apportionment population includes resident population, U.S. Armed Forces personnel and federal civilian employees stationed outside the United States and their dependents[2].

Redistricting is necessary because of population shifts. Population changes between states and even within a state. From population shifts between states, each state may gain or lose seats. Also, population growth is different within a state. Therefore, the congressional district should be often redrawn (Wattson 2010). However, redistricting can be highly partisan because the drafters can be involved in the decision making where to draw political boundaries. If they have political interest, the redistricting outcome can be highly biased and unfair (Altman and McDonald 2010).

In order to redraw political districts, while the process and guidance vary from state to state, political redistricting plans must satisfy a set of demographic, geographic, and political criteria. Demographic criteria include equal population and minority representation and geographic criteria include contiguity and compactness. Political criteria include the respect of existing plans and avoidance of partisan gerrymandering. There are other criteria such as the respect of natural or administrative boundaries. These criteria are exhibited in law and identified in literature on political redistricting problems

[1] United States Code, Title 2, Chapter1, §2a and §2b, and Title 13, Chapter 5, Subchapter II, § 141.

[2] Washington D.C., the capital city, does not have a representative based on the U.S. Constitution since it is not a state but a district called the District of Columbia. The population of the District of Columbia is not included in the apportionment population. Instead, Washington D.C. elects one delegate, who has no voting right, to House of Representative.

(Williams 1995).

Political redistricting problems can be treated as combinatorial optimization problems because the number of feasible solutions is exponentially increased with the problem size (Altman 1998). These problems may be difficult to solve because of challenges in formulating their requirements in mathematical forms (Altman 1997; Altman 1998; Eagles et al. 2000), simultaneously satisfying several redistricting criteria (Williams 1995; Wei and Chai 2004), and the size of the solution space (Bação et al. 2005). A solution method for political redistricting problems should have both efficiency in terms of the computational time and effectiveness in terms of the ability to find high quality solutions.

The developments of computer technologies since 1960s have enabled computers to be used in political redistricting processes to manage large amounts of data, to draw political boundaries, and to analyze redistricting outcomes (Altman et al. 2005, Altman and McDonald 2010). Especially, in 1990s, GIS emergence in a computer technology had a tremendous effect on redistricting. Also, in 2000s, mapping in redistricting become more user friendly (Altman et al. 2005). The use of computers in political redistricting can remove several factors such as intentions of the decision makers or the majority in political views or race from the redistricting process, which tend to drive a highly biased and unfair outcome of redistricting. So, the redistricting process can eliminate human judgments replaced by a set of neutral criteria such as equal population, contiguity and compactness throughout computer technologies.

Existing optimization techniques for solving political redistricting problems may be divided into exact and heuristic methods. An exact approach (e.g., linear programming) can guarantee to find a global optimal solution by systematically examining all possible solutions. However, it may show high computational complexity and therefore may be impossible to solve large size problems. In order to overcome this issue, a heuristic approach can be designed to efficiently search for solutions to large size problems. A heuristic approach, however, cannot guarantee to achieve global optimality. Results from a heuristic algorithm are often near-optimal or optimal (Cooper 1964; Reeves 1993).

There are a few exact methods in the political redistricting. At first, an exact method for political redistricting was developed by Garfinkel and Nemhauser (1970). This method consists of two stages. The first stage was to exhaustively search for all possible districts with respect to equal population, contiguity and compactness. The second stage was to minimize the maximum deviation from the mean district population among all possible districts. However, this method may encounter computational difficulty when generating all possible districts. Another exact method for political redistricting has been developed by Shirabe (2009). The method formulates contiguity criterion based on recent advances in land acquisition problems. However, contiguity is not fully satisfied because it is still shown between different districts even though it should be really prevented. Furthermore, this method may not yields results with strict population equality.

Contiguity should be another challenge in political redistricting problems because of its difficulty to formulate. Garfinkel and Nemhauser (1970) enforce contiguity by the process of exhaustively searching for all possible districts. Though the requirement of contiguous districts has been traditionally difficult to formulate, some recent progress in solution approaches to land acquisition problems has showed success in addressing contiguity issues. To solve a land acquisition problem, a contiguous set of land parcels must be selected for a particular land use. Cova and Church (2000) developed contiguity constraints based on finding shortest path between a land parcel and a pre-selected root land parcel. Williams (2002) developed a contiguity model based on the constructing of a minimum spanning tree. More recently, Shirabe (2005) developed contiguity constraints based on finding network flows. These contiguity formulations in land acquisition problems can be applied to develop exact models to political redistricting.

Because of the computational challenge to exact methods, most approaches to political redistricting have been developed as heuristics (see chapter 2 section 2.4 for a more detailed review). However, most of the existing heuristics may show difficulty in finding a redistricting plan with strict population equality while most states require a strict population equality plan for congressional plans.

Existing literature does not provide comparative information of methods. Existing approaches have been developed for particular redistricting plans. There is no literature of evaluating approaches in order to get relative information such as the main idea of each method, how to efficiently find a high quality solution, and criteria used.

In the reality, most states have perfect plans where their population deviation is less than 1 person. There can be many plans with various spatial configurations that are perfect in the sense that it has either 0 or 1 person in population deviation. In addition, for the given plan, which is perfect or not, there can be different spatial shapes but similar or same population of the given plan.

1.2 Research purpose

The main purpose of the dissertation is to develop new optimization approaches to political redistricting problems. The dissertation develops two different exact optimization models to political redistricting based on recent advances in solving land acquisition problems. The first exact model is called a spanning tree model by constructing several districts with broken edges of a spanning tree to political redistricting. The second exact model is called a network flow model by making several sub-networks in political redistricting. The dissertation also implements two different heuristic optimization models to political redistricting. The first heuristic model is called a multi-scale simulated annealing heuristic which uses multiple cooling rates and the second heuristic model is called Give-And-Take greedy algorithm which exchanges population units between a district with the larger population than the ideal population and a district with the smaller population than the ideal population.

This dissertation has the second research objective, which is to evaluate with existing representative models in order to understand their relative performance. Two exact models will be compared with the exiting enumeration method of Garfinkel and Namhauser (1970). Two heuristic models will be compared from computational experiments with the existing heuristic of Xiao (2008).

Finally, the Give-And-Take greedy heuristic is expanded to address the third research objective, that is, to implement a heuristic to find different shapes but similar (better) or same population of the given plan.

1.3 Research organization

The dissertation is organized as follows. Chapter 2 reviews literature relevant to later chapters. The first section introduces political redistricting problems including several redistricting criteria. The second section discusses computational issues in political redistricting problems and the third section reviews how to consider contiguity in land acquisition problems, which can guide to develop new exact models. The fourth section reviews optimization approaches to political redistricting problems for exact models and heuristic models, and the fifth section discuss approaches to manipulate spatial configurations. The sixth section is political redistricting approaches under GIS environment.

Chapter 3 incorporates two exact optimization models for political redistricting. The first section introduces a spanning tree model and the second section discusses a network flow model. The third section explains the existing research of Garfinkel and Nemhauser (1970), and the fourth section compares the performance of these two new models, along with a third model of Garfinkel and Nemhauser (1970).

Chapter 4 implements two heuristic optimization models for political redistricting. The first section is a multi-scale simulated annealing, which is based on simulated annealing and the second section is Give-And-Take greedy algorithm, which include the main idea and the general process of the heuristic. The third section is experiments on the Give-And-Take greedy heuristic to find various spatial shapes similar to the given plan and the sensitivity analysis of the heuristic. The fourth section introduces computational experiments of the heuristic models. Computational experiments discuss comparisons of two new models along with the heuristic of Xiao (2008), and results of the application of Give-And-Take greedy heuristic and the sensitivity analysis.

Chapter 5 summarizes dissertation work, concludes and discusses computational results and contributions of the dissertation. Further, potential future work is also discussed.

Chapter 2 Literature review

This chapter provides a review of literature related to studies of political redistricting problems. The first section includes the political redistricting criteria. The second section discusses computational issues of political redistricting problems and the third section investigates contiguity requirement in land acquisition problems, which can be applied to solve political redistricting problems. The fourth section includes existing optimization approaches that can be employed to solve political redistricting problems and the fifth section reviews approaches to manipulate spatial configurations. The sixth section discusses political redistricting approaches relevant to GIS (Geographic Information System).

2.1 Political redistricting problems

A variety of redistricting criteria must be considered in order to evaluate political redistricting plans. Demographic criteria include equal population and minority representation. Geographic criteria include contiguity and compactness. Political criteria include partisan gerrymandering and the respect of existing plans.

2.1.1 Demographic criteria

The most common demographic criteria used in political redistricting are equal population and minority representation. The equal population criterion was an outcome of the “reapportionment revolution” of the 1960s. The Voting Right Act of 1965 and its amendments led to the minority representation criterion (Williams 1995).

2.1.1.1 Equal population

In political redistricting plans, all districts should have approximately the same number of voters in order to overcome malapportionment[3]and respect the “one-man, one-vote” principle which means that each district must have equal number of electors. With historical background, the Fourteenth Amendments to the United States Constitution includes Equal Protection Clauses, "no state shall… deny to any person within its jurisdiction the equal protection of the laws." In Baker v. Carr[4], the U.S. Supreme Court ruled that the federal courts should have jurisdiction to redistricting plans. In Baker, discrepancy from redistricting failed to have the equal protection by the Fourteenth Amendment. In Wesberry v. Sanders[5], congressional districts have to be approximately equal in population and “one-man, one-vote" was first applied. In Reynolds v. Sims[6], state legislature districts had to be roughly equal in population based on the principle of

3 Refers to inequality in the population size of districts.
4 Baker v. Carr, 369 U.S. 186 (1962).
5 Wesberry v. Sanders, 376 U.S.1 (1964).
6 Reynolds v. Sims, 377 U.S. 533 (1964)

“one-man, one vote”.

2.1.1.2 Minority representation

The Congress enacted the Voting Rights Act of 1965 in order to remedy the inequality of opportunity caused due to racial and ethnic minorities. It is possible to increase minority political participation such as African American, especially in the South due to the Voting Rights Act of 1965 and its amendments and extensions of 1970, 1975 and 1982 (Williams 1995). An outcome of the Voting Right Act is the creation of majority-minority districts. A majority-minority district refers to a district in which the majority of the population is either African American, Hispanic, Asian or Native American. Majority-minority districts can be formed proportionate to minority populations (Parker 1989; Williams 1995). Many of these majority-minority districts, however, were struck down as unconstitutional by court decisions in the 1990s because they can lead to racial gerrymandering with bizarre shapes (Eagles et al. 2000).

2.1.2 Geographic criteria

In addition to the demographic criteria, there are geographic criteria. Geographic criteria include contiguity and compactness. Contiguity is a legal requirement for political redistricting plans while compactness is not a legally required (Nagel 1972; Williams 1995).

2.1.2.1 Contiguity

District contiguity is a compelling criterion, not only because it is consistent with territorial political representation, but also because a contiguous district usually seems more rational or intuitive than a district formed from many disconnected pieces (Grofman 1985). A district is contiguous if one can go from any point in the district to any other point without leaving the district (Mills 1967; Nagel 1972; Grofman 1985). In political redistricting problems, contiguity has been maintained by exhaustively generating all possible districts (Garfinkel and Nemhauser 1970), by searching for a shortest path between a population unit and a pre-selected root population unit (Mehrotra *et al.* 1998), by swapping between adjacent population units (Nagel 1965; Kaiser 1966), or by adding adjacent population units to the seed population unit (Vickrey 1961; Harris 1964; Gearhart and Liittschwager 1969; Liittschwager 1973).

2.1.2.2 Compactness

A district is geographically compact if it has a circular or square shape (Garfinkel and Nemhauser 1970; Yong 1988; Niemi et al. 1990). Though a circular shape is an ideal as compactness measure for an individual district, it is impossible to have circular shapes for multiple districts. A square or a hexagonal shape can be an alternative for multiple districts (Niemi et al. 1990). District compactness is not a federal legal requirement for congressional (or state legislative) districts, although some states require that districts should be compact. Compactness have been supported as a neutral standard because compactness is used as to defend against gerrymandering in terms of the manipulation of the district boundary for a particular party, show internal cohesion in terms of making population units close together, and prohibit boundary irregularity (Morrill 1973; Morrill 1981; Grofman 1985; Baker 1990; Polsby and Popper 1991; Williams 1995; Mehrotra et al. 1998). Also, compactness standards will result in increased minority representation (Polsby and Popper 1993). However, there is little evidence that compactness can protect gerrymandering in the literature (Hacker 1964; Musgrove 1977; Young 1987; Altman 1998). Furthermore, compactness criteria have been controversial because it is measured in several different ways.

There are several compactness measures in the literature. Compactness can be broadly measured either relatively (relative measure) or absolutely (absolute measure) (Table 2.1). Relative measures decide district compactness by only shape regardless of area. When they are same shapes, a small area cannot be said to be more compact than a large area, and vice versa (Gibbs 1961; Reock 1961; Boyce and Clark 1964; Harris 1964; Kaiser 1966; Schwartzberg 1966; Yong 1988). The ratio of relative measures close to one indicates more compact. On the other hand, absolute measures determine district compactness based on area as well as on shape (Mills 1967). A small area is more compact than a large area even though two are same shapes (Weaver and Hess 1963; Hess et al. 1965; Adams 1977).

Relative Measure

1.District area compared with area of compact figure

- RDA_1: Ratio of the district area to the area of the minimum circumscribing circle (Roeck 1961).
- RDA_2: Ratio of district area to area of circle with diameter equal to district's longest axis (Gibbs 1961).
- RDA_3: Ratio of the district area to the area of the minimum circumscribing regular hexagon (Geisler 1985).
- RDA_4: Ratio of the district area to the area of the minimum convex shape that completely contains the district (Niemi, et al. 1991).

2.Length-Width test

- RLW_1: Ratio of the length (*L*) to the width (*W*), where length (*L*) and width (*W*) are those of the rectangle enclosing the district and touching it on all four sides for which the ratio of length to width is a maximum (Young 1988).
- RLW_2: Ratio of the length (L) to the width (W), where L is longest diameter and W is the maximum diameter perpendicular to L (Harris 1964).
- RLW_3: Ratio of the length (L) to the width (W), where W and L are that of the circumscribing rectangle with minimum perimeter (Niemi, et al. 1991).
- RLW_4: Ratio of the length (L) to the width (W), where L is longest axis and W and L are that of a rectangle enclosing the district and touching it on all four sides (Niemi, et al. 1991).
- RLW_5: Length (*L*) minus Width (*W*),where length (*L*) is the maximum length of any district and width (*W*) is the maximum width perpendicular to the length (Harris 1964).

3. The Perimeter Test

- RPT_1: Ratio of the district's perimeter length to the circumstance of a circle of equal area (Schwartzberg 1966).

4. The Perimeter-Area Comparisons

- RPA_1: Ratio of the district area to the area of a circle with the same perimeter (Cox 1927).
- RPA_2: 1- (Ratio of the district area to the area of a circle with the same perimeter)$^{1/2}$, PA2 = (PA1) $^{1/2}$ (Niemi, et al. 1991).
- RPA_3: Ratio of the perimeter of the district to the perimeter of a circle with an equal area (Horton 1932; Schwartzberg 1966).
- RPA_4: Perimeter of a district as a percentage of the minimum perimeter enclosing that area, $PA_4 = 100\ (PA_3)$

5. The Relative moment of inertia

-RMI_1: The relative moment of inertia – the variance of distances from all points in the district to the district's areal center of gravity which is normalized. Adjusted to range from 0 to 1 (Boyce and Clark 1964, Kaiser 1966).

Absolute Measure

1. Moment of inertia

- AMI_1: The population moment of inertia (Weaver and Hess 1963, Hess *et al.* 1965).

2.The perimeter Test

- APT_1: Sum of perimeter (Dixon 1968, Tyler and Wells 1971, Adams 1977, Wells 1982).

Table 2.1. Comparisons of compactness measures[7]

[7] Compactness measures are summarized based on the work of Young (1988), Niemi et al. (1990), Horn

2.1.3 Political criteria

Political redistricting problem also include political criteria. Political criteria include partisan gerrymandering and the respect of existing plans. Normally, political redistricting plans tend to minimize partisan gerrymandering and difference with the existing plans.

2.1.3.1 Partisan gerrymandering

Partisan gerrymandering is to draw district borders in a way that discriminates against a political party or a particular group. A partisan gerrymandered district can be formed by methods such as cracking or packing. Cracking is the process of diluting the power of voters for a particular party in a way that divides its supporters into many districts. Packing is the process of confining voters into one or a few district (Morrill 1981; Williams 1995). It is desirable that districts should have balances between political parties or particular groups.

2.1.3.2 The respect of existing plans

A new districting plan similar to the existing plan may have more social and political acceptance than the new districting plan totally different from the existing plan (Nagel 1965; Williams 1995). In the substantially changed new districting plan, it is necessary to

---

(1993) and Altman (1998).

reestablish a new set of interests by politicians and parties. From the point of view of politicians or parties, they may avoid running in the new district. Furthermore, a new districting plan minimally modifies the existing plan when a state's seat allocation has not been changed.

2.2 Computational issues in political redistricting problems

Political redistricting problem are difficult to solve because of the size of solution space. The lower bound of the number of partitions (S1) occurs when a population unit is connected to only two neighbors except that both ends have only one neighbor and can be calculated as (Keane 1975; Bação et al. 2005):

$$S_1(n,r) = \frac{(n-1)!}{(r-1)!(n-r)!}$$

where *n* is the number of population units, and *r* is the number of districts.

The upper bound of the number of partitions (S2) occurs when each population unit is adjacent to every other population unit and can be calculated as the Stirling number of the second kind (Altman 1998; Bação et al. 2005):

$$S_2(n,r) = \frac{1}{r!}\sum_{i=0}^{r}(-1)^i\left(\frac{r!}{(r-i)!i!}\right)(r-i)^n$$

The total number of possible partitions depends on the number of population units, the number of districts, and the connectivity among spatial units. A small problem may have a large number of possible ways. An example of S_1 (25, 4) would have 2024 ways and S_2

(25, 4) would have 4.677×10^{13} ways. Political redistricting problems are characterized as NP-complete[8] problems, which are computationally intractable.

2.3 Contiguity in land acquisition problems

Land acquisition problems search for continuous land parcels depending on the problem purpose. A cluster of acquired land parcels is contiguous if one can move from an acquired parcel to another without leaving the cluster. Spatial contiguity is one of the most frequently used criteria in land acquisition problems. Recent research has showed that spatial contiguity can be formulated in a mixed integer programming framework (Cova and Church 2000; Williams 2002, Shirabe 2005). Cova and Church (2000) formulated contiguity constraints with the concept of finding shortest path between a land parcel and a pre-selected root land parcel. The spatial contiguity is achieved by selecting a land parcel from a pre-selected root land parcel so that remaining land parcels are also connected. Williams (2002) developed a contiguity model based on the construction of a minimum spanning tree. Contiguity is satisfied by finding a minimum spanning tree with the number of edges equal to one less than the number of parcels needed in land parcels. More recently, Shirabe (2005) developed contiguity constraints based on finding network flows theory. Contiguity is designed as a connected sub-network, which represents fluid

---

[8] In computational complexity theory, the P problems are the problems solved in a polynomial time. The NP problems (non-deterministic polynomial time) are problems, which have no information whether or not problem is solved in polynomial time. NP-complete problem are the most difficult problems in NP problem and every problem in NP is reducible to it. NP-hard problems are the problems which are NP-complete and do not solved in a polynomial time.

movement from multiple sources to a single sink.

2.4 Optimization approaches to political redistricting problems

A variety of optimization solution approaches to political redistricting problems have been developed in the literature. They are into mainly two types of models; exact models and heuristic models.

2.4.1 Exact models

There are a few approaches based on exact optimization models. Garfinkel and Nemhauser (1970) propose an exact method for political redistricting with respect to three criteria such as population equality, contiguity and compactness. They first exhaustively generate all feasible districts and find a solution in a way that minimizes the maximum deviation from the mean district population in any chosen district. This method is an enumerative method, which considers all possible cases, however, becomes prohibitive in time. More recently, Shirabe (2009) develops an exact political redistricting model based on recent advances in land acquisition problems. The author formulates a contiguity requirement. However, contiguity is not fully satisfied because there still maintains unnecessary contiguity between different districts, which should be controlled.

2.4.2 Heuristic models

The first type of heuristic models is proposed by Weaver and Hess (1963) and Hess et al. (1965). They implement a heuristic optimization model by solving iterative transportation problems that allocates population units to legislative district centers with respect to compactness and population equality. They start with an arbitrary set of centers of districts and solve the transportation problem (the warehouse location problem) in a way that minimizes squared Euclidean distances between population units and districts in order to enforce compactness. They reunite any split population units to the center with the largest share of the population, calculate the center of gravity of each derived district and solve the transportation problem with these centers until there is no more change. Mills (1967) applies the recombining method of Hess et al. (1965) into districting problem with results that it can take a little time to reach the local optimum and take into account natural boundaries so that population units cannot be split in natural boundaries. Morrill (1973; 1976) redistricts the legislative seats on a basis of the method of Weaver and Hess (1963). Helbig et al. (1972) shows non-partisan empirical political redistricting results in Missouri using Hess et al. (1965). Plane (1982) reformulate the transportation model of Hess et al.(1965) into a quadratic integer program for considering inter- and intrazonal commuting of spatial interaction model for political redistricting problems. Robertson (1982) applies their approach to redistricting problems by taking into account currently existing features such as the social and topographic boundaries. The results show successful applicability of the location-allocation model to political redistricting problems, especially in urban areas. Hojati (1996) use Lagrangian relaxation to assign the district centers and capacitated transportation problems to resolve the split area problem. George et al. (1997) show flexibility to solve a large-scale problem based on a network-based optimization problem of transportation problem. More recently, Barkan et al. (2006) operate a location-allocation algorithm and propose a computational districting model known as a spatial decision support system (SDSS) that can be used to represent the problem, implement the model, and solve the problem and map solution based on location-allocation problem. These approaches have a disadvantage that initial selection of district centers may control the results because district centers are arbitrary chosen.

The second type of heuristic models is developed by Vickrey (1961). Vickrey (1961) develops a heuristic method based on a growth algorithm where a reference population unit is started, and a seed population unit far from a reference unit is selected for each district and the districts are growing by adding contiguous population units to the seed until all population units have been assigned. The results make it possible to have enclaves. Thoreson and Liittschwager (1967) expand the method of Vickrey (1961) in a way that uses different references units are explored for new solutions and try on a regular lattice with empirical results. Gearhart and Liittschwager (1969) and Liittschwager (1973) refine the method of Vickrey (1961) by adding the process of population equality and reducing the possibility of enclave formation. Harris (1964) develops a two-stage heuristic approach to districting. The first stage finds rectangular districts on a basis of growth algorithms that start with a seed and select adjacent population units to be contiguous, compact and equal populated districts. The second stage modifies districts so that district lines coincide with community or political boundaries. The growth algorithms satisfy population equality, however, also produce awkward shape if it overcomes the possibility of enclave.

The third type of heuristic models includes local search based methods by Nagel (1965) and Kaiser (1966). Nagel (1965) and Kaiser (1966) start with an existing districting plan and improve an existing districting plan by swapping population units from one district to another district. The obtained solution is dependent upon the initial assignment of population units to districts. Only improving solution will be accepted which means that it is possible to trap into local optimal solution. As a result, the solution is controlled by the original plan, which may produce minimum change.

The fourth type of heuristic models is spatial interaction modeling. Openshaw (1977a) designs zoning systems with four different spatial interaction models. The author recognizes the importance of zone design using aggregated data and shows the considerable role of the spatial interaction model in zone designs. Openshaw and Rao (1995) acknowledge the availability of GIS technology in zoning problem and provide three algorithms (tabu search, simulated annealing, parallel algorithms) with the objective based on spatial interaction using aggregated census data. Alvanides et al. (2000) upgrade previous approaches from originally Openshaw (1977a), Openshaw and Rao (1995) to Alvanides and Openshaw (1999) to solve zoning problem based on spatial interaction.

Alvanides et al. (2000) describe a zone design algorithm based on optimizing an objective function with constraints. A penalty function is added to the objective function to represent the 'cost' of violating the constraints.

The fifth type of heuristic models is set covering modeling or set partitioning model. Marsten (1974) presents an algorithm for the special linear program known as the set-partitioning problem so that such an algorithm can be applied to political redistricting problem. Nygreen (1988) considers the political districting problem for Wales to have as equal electorates as possible. These methods such as integer programming, set partitioning and implicit enumeration are used and compared. Mehrotra et al. (1998) propose a constrained graph partitioning heuristic with pre and post-processing steps building on the earlier work of Garfinkel and Nemhauser (1970). They consider many more districts by linear relaxation with respect to contiguity and population equality, and implemented a branch-and-price based heuristic method in order to yield the compact, contiguous, and equally populated districts.

The sixth type of heuristic models is based on metaheuristic approaches which can be applied to different optimizations. Metaheuristic approaches include simulated annealing, tabu search, and evolutionary algorithms. Browdy (1990) at first suggests the method of simulated annealing to the problem of drawing optimal districts with respect to population equality and contiguity, compactness. Macmillan and Pierce (1994) use simulated annealing for political redistricting problems called ANNEAL which provides the method of checking contiguity called the switching point method. The method considers topological relationships among population units and proves to be efficient because of focusing on the candidate area. Horn (1995) uses a hill-climbing technique for a large-scale electoral district configuration problem with the aim of getting compact, contiguous and equally populated districts. Alvanides (2000) employs a simulated annealing as an optimization technique to solve zoning problems. Macmillan (2001) also implements a model based on simulated annealing with respect to population equality and contiguity base on a contiguity checking procedure called the switching point method and show empirical evidence on the performance of the method. Bozkaya et al. (2003) apply tabu search and adaptive memory procedure to the political redistricting problem. They represent a multi-criteria approach including socio-economic homogeneity, similarity to the existing plan and integrity of communities in addition to criteria such population equality, contiguity and compactness that existing studies consider. Wei and Chai (2004) develop a multiobjective hybrid metaheuristic approach using tabu search and scatter search methods which approximate good non-dominated sets and explore solutions with respect to equal population and contiguity. Bação et al. (2005) propose a genetic algorithm to the political districting problem with respect to population equality, contiguity and compactness. More recently, Xiao (2008) implements an evolutionary algorithm to the political redistricting problem to have equally populated districts.

2.4.3 The limitation of existing optimization approaches

First, existing literature has shown that there are not various exact models. Garfinkel and Nemhauser (1970), and Shirabe (2009) produce exact optimization approaches to political redistricting. Garfinkel and Nemhauser (1970) show the difficulty to formulate contiguity and enforce it by the process of exhaustively searching for all possible districts, which is computationally very extensive. Shirabe (2009) formulates contiguity, but produces insufficient results and is still maintained between different districts even though it should be prohibited.

Second, existing approaches do not provide comparative information of these methods. Most of the existing research have been developed for a particular redistricting plan. There is no literature of evaluating approaches in order to get relative information. It is necessary to introduce comparative research in terms of efficiency in computing time and effectiveness in getting high-quality solutions, and main idea of each model as well as criteria used. Comparative research makes it possible to have objective judgments.

Third, most existing heuristics are still difficult in efficiently finding high quality solutions in population deviation. In reality, most states have perfect plans in terms that population deviation is less than one person. Most of the existing heuristics apply various heuristic methods such as simulated annealing, tabu search, and evolutionary algorithm to solving political redistrict. However, they still have the difficulty of finding a redistricting plan with strict population equality.

2.5 Approaches to manipulate spatial configurations

Manipulation of spatial configurations is mostly related with the compact shapes, one of redistricting principle, in political redistricting. The major objective finding compact spatial shapes in redistricting is to prevent gerrymandering, which is from the surname of Massachusetts Governor Elbridge Gerry and manipulate districts as the salamander shape of the district for political purpose (Stern 1974; Well 1982). However, manipulating districts as a more compact shape is important to prevent racial gerrymandering (Polsby and Popper 1993). There are several researches to show how affect the most compact shapes on redrawing of the district to diminish bias of the plan (Paddison 1976; Yong 1988; Niemi et al. 1990; Altman 1998; Johnston et al 1999; Johnston and Pattie 2000). Compact shapes can be manipulated by several methods such as the ratio of the district width to the district length, distance between population units and the district center and *etc*.

Another manipulation of spatial shapes is the foundation of different shapes. The reality is that most states require a perfect plan for a congressional plan, which represents a strict population equality plan with population deviation of 0. There will be many plans that are perfect in the sense that it has either 0 or 1 person deviation. Also, for the given plan which is perfect or not, there can be various redistricting plans with similar (better) or same population of the given plan. The given plan can be the official plan acknowledged by states in reality. The various redistricting plans can be proposed for a possible redistricting plan and the best redistricting plan can be selected among different redistricting plans.

Nagel (1965) and Kaiser (1966) manipulate spatial configurations by starting with an existing districting plan and improving an existing districting plan by swapping population units from one district to another district. Robertson (1982) takes the initial solution where locations of polling booths were fixed and electors were allocated to their nearest station, and illustrates gradually enhanced spatial configurations of districting by considering population equality, topographic and social boundaries, and adjustment. Trinidad and Smith (2000) show two different boundary configurations; the first takes into account only equality in population size and the second achieves both population equality and compactness.

2.6 Redistricting in the spatial analysis modeling with GIS environment

There are many definitions of GIS (Geographic Information System). Commonly, GIS refers to the integration system of hardware, and software for storage, mapping, management and analysis of geographic data which has topographic information (Macmillan and Pierce 1991, Folger, 2009). Furthermore, GIS makes it possible to understand, query, and visualize data as the form of maps, statistical reports and charts (Salling, 2010). GIS can be designed to support spatial decision making (Maguire 1991), thought GIS shows several shortcomings in that analytical modeling techniques or the decision maker's interaction with the solution process may not successfully be performed (Densham, 1991).

SDSS (Spatial Decision Support System) has been developed as the combined form of GIS and DSS (Decision Support System) since the late 1980s (Armstrong and P.J. Densham 1990, Densham 1991). DSS refers to interactive computer-based system which supports data managements and analytical model operations for decision makers in ill-structured problems (Gorry and Morton, 1971). The emergence of DSS makes it possible to support a variety of decision-makings (Geoffrion, 1983). Because spatial problems, for example facility location problems, have more complex natures, SDSS more easily can manage spatial data (the input and the output) as well as non-spatial data, represent complex spatial relations and include analytical modeling for spatial geographical analysis. (Densham, 1991). Under the environment of SDSS, decision makers iteratively generate a set of alternative solutions, participate in defining and analyzing the model problem, and evaluating the outcomes, and finally integrate the final outcomes with the quantitative data and the qualitative information of the model.

As applied political redistricting problems, GIS should be considered with SDSS. GIS aims to support people in making decision about redistricting (Altman and McDonald, 2010). GIS as SDSS sufficiently views and analyses demographic data such as census data, socio-information voting related as well as topographical information (Salling, 2010). Thus, GIS is an important redistricting tool to efficiently draw boundaries, build district plans, and evaluate alternative plans based on a set of criteria. GIS as SDSS provides redrawing of district boundaries and interaction with the mapping (or displaying) of district results which if necessary integrated with statistical measures of the redistricting criteria. Above all, GIS in political redistricting has an advantage to produce non-biased redrawing of political boundaries which shows fairness of redistricting results considering multiple criteria. However, in reality, commercial GIS software show limits to redraw political boundaries while user-requested modules or extension are added to GIS software easy to use. Other software such as optimization tools, programming tools or statistical tools can be used for political districting in an alternative way.

In the redistricting literature with optimization modeling approaches, GIS also can analyze a wide range of geographical systems, but also operationalize the problem associated with such systems. Macmillan and Pierce (1994) introduce the political redistricting optimization model which can be used in a GIS environment using TransCAD procedures. Macmillan (2001) proposes simulated annealing heuristic using the switching point method which can be employed into GIS framework by recognizing the arc-node topology.

There are several redistricting literatures to associate with GIS in political redistricting.

Openshaw and Rao (1995) acknowledge the availability of GIS technology in zoning problem that can use as a visualization tool. Openshaw (1996) gives the importance of GIS-based spatial analysis method in the zoning system. George et al. (1997) mention that the results are being incorporated into GIS which can be used as a practical tool to assist the decision-maker. Barkan et al. (2006) incorporate GIS into SDSS (spatial decision support system)–based electoral system which explores the spatial and the attribute characteristics of the solution.

More recently, according to the development in computer technologies, there is a web-based redistricting which enables the public to participate in the map drawing process by online (Altman and McDonald, 2010). In 2007, Chris Swain of the USC Game Innovation Lab released a "redistricting game"[9] so that many people can redraw political boundaries. The redistricting game is relatively a user-friendly game that novices can have an easier access in redistricting. Also, in 2009, throughout the support of GIS technologies, there is the Ohio Redistricting Competition[10] which is the project of Ohio's Secretary of State (SOS) to provide the public with an opportunity to participate fair and transparent redistricting process. This is an open competition that persons could build a new districting plan satisfying listed redistricting criteria by accessing software and data throughout the access to server via Internet. The winning plans show higher qualities than the current congressional district plan.

---

9 http://thecaucus.blogs.nytimes.com/2007/06/14/a-gamers-guide-to-redistricting/, http://redistrictinggame.org/

10 http://www.sos.state.oh.us/sos/redistricting.aspx

The web-based redistricting shows how GIS technologies proceed for next rounds. First step that GIS technologies have for next rounds is user-interface which non-expert can utilize in more easy ways. The use of GIS software sometimes gives novices troubles with the access because GIS has its own concept such as layers, spatial topology and mapping. There were trainings and consulting regarding GIS software in redistricting problems in Ohio competition, though non-experts still needs long time to acquire the knowledge of GIS and apply it to redistricting problems.

Second step that GIS has as a new direction is web-base GIS technologies. In Ohio competition, Web-based GIS technologies can make an important role in giving a possible direction in redistricting problems which the public can easily not only participate the redistricting process but also share or discuss the redistricting. Web-based GIS technologies in political redistricting show major steps so that public participation can be possible to produce the non-biased redistricting outcomes.

The last step that GIS should have is to build environment possible to manage or evaluate redistricting districting data. GIS software provides table forms which connect to spatial entities and include information such as coordinates, population and etc so that simple computations can be possible. However, it has still limitations that the results of the entire final plan have been a trouble with the displaying according to districting criteria, and there are no standard measures provided of redistricting criteria such as compactness.

Chapter 3 Exact approaches to political redistricting problems

Contiguity in political redistricting is complicated to represent as the formulation. However, recently contiguity has been successfully addressed as a mixed integer programming in land acquisition problems, which finds contiguous parcels for a particular purpose. So, the use of contiguity in land acquisition modeling can be designed to apply to political redistricting problems.

This chapter introduces two exact optimization models to political redistricting on a basis of recent development in land acquisition problems and also evaluates them with the existing approach. The first section introduces a spanning tree model and the second section produces a network flow model. The third section is a simple explanation of the existing research of Garfinkel and Nemhauser (1970) and the fourth section is computational experiments of the existing enumeration approach and two developed exact approaches.

New developed models can be developed using a graph representation of the problem. In this graph, there are n vertices, which can be exchangeably used with n nodes, each representing a spatial unit with a certain population, and edges meaning connections among vertices. Formally, a graph is a pair $G = (V, E)$ where V is a set of vertices and E is a set of all edges meaning connections among vertices.

The following is a list of variables that will be used in all the tree models.

N the total number of vertices in the graph;

i, j, I the indices and set of vertices in the graph, where $1 \le i, j \le N$;

D_i the set of vertices adjacent to vertex i;

P_i the population of the i-th spatial unit;

R the total number of districts to be divided;

r the indices of districts, $r = 1, 2, \ldots, R$;

$\overline{P}$ ideal population $\frac{\sum P_i}{R}$

3.1 A Spanning tree model

A planar graph is constructed in the Cartesian plane or on the surface of a sphere where edges only intersect at vertices. A spanning tree of the graph is a tree that connects all the n vertices of the graph with n-1 edges (Figure 3.1b). A subtree of the spanning tree consists of a subset of the vertices in the spanning tree and the land parcels represented by the vertices on a subtree from a contiguous set (see Figure 3.1c and 3.1d). Planar graphs have dual graphs, which is also a planar graph and can be created by placing a new vertex in each region enclosed by edges of the original graph (so-called the primal graph) and connecting these new vertices by new edges across every original edge. Note that the edges on the primal graph are intersecting with the edges in the dual graph (Figure 3.2a).

Contiguity can be enforced by searching for a spanning tree on a planar graph that no two edges intersect, except at vertices. In this way, cycling can be prevented and the valid spanning trees can be subsequently ensured because the complementarity between the primal graph and the dual graph can prevent a cycle in any of the two trees (Figure 3.2b). Williams (2002) utilized this feature to develop a contiguity model for land acquisition problems (see below constraints (2), (3) and (4) in the model).

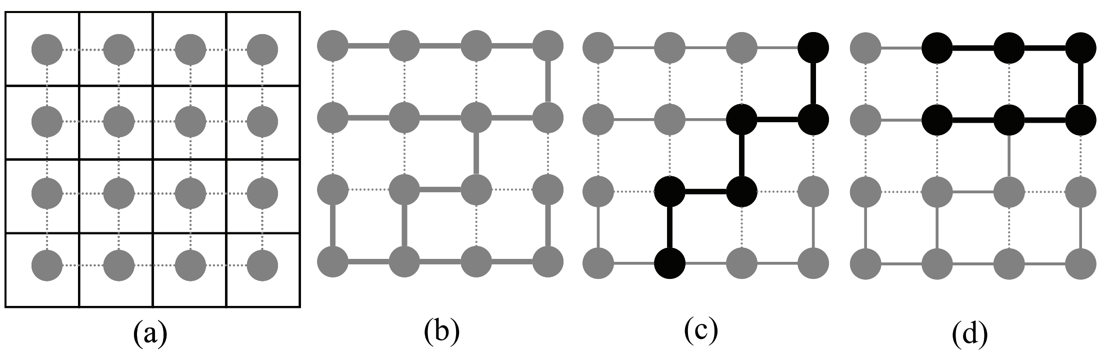


Figure 3.1. A graph representation of land parcels. (a) Each vertex is used to represent a regular land parcel and the edges (dashed lines) represent the connectivity between adjacent parcels. (b) A spanning tree where the solid lines represent edges on the spanning tree. (c) (d) A subtree of the spanning tree. Dark dots represent the vertices in the subtree and dark solid lines represent the edges on the subtree.

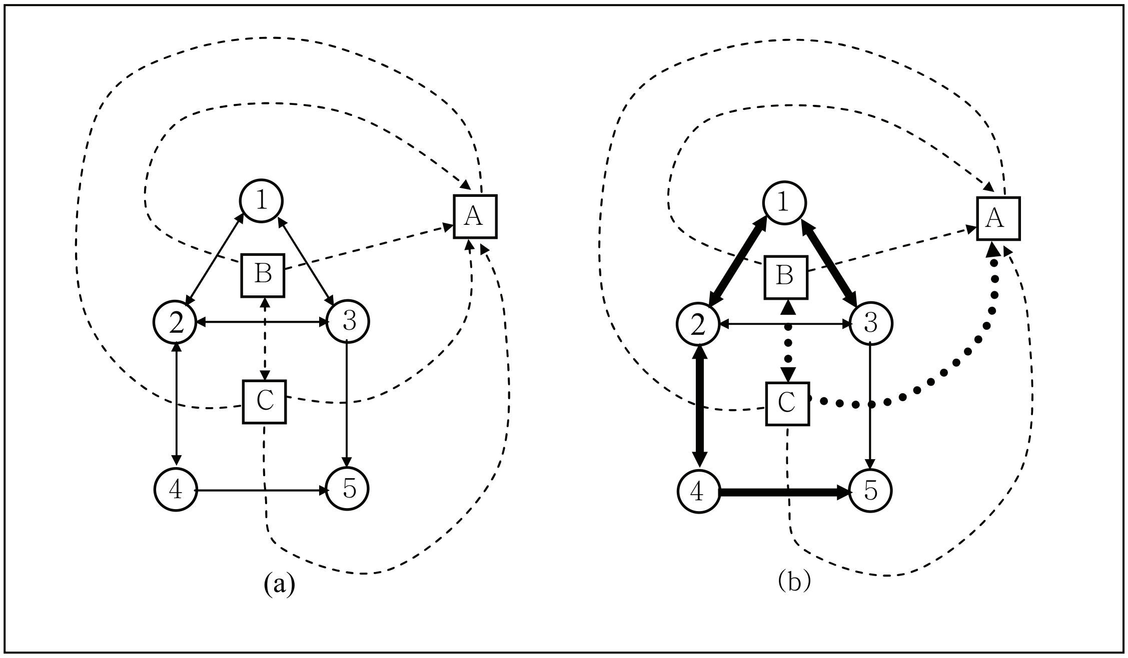


Figure 3.2. Primal and dual planar graphs (a), and complementary spanning trees (b). Here, circles represent vertices in the primal graph and vertex 5 is specified as the terminal vertex; squares represent vertices in the dual graph and vertex A is assigned to be the terminal vertex. Thick lines in (b) represent the arcs used to construct complementary spanning trees.

The same mechanism of ensuring contiguity which finds a complementary spanning tree both in the primal and dual graph can be applied to enforce contiguity in political redistricting. In political redistricting, contiguity is also ensured by searching for the complementarity of two spanning trees on the (primal and dual) graphs such that none of the edges in the primal tree intersects any edge of the dual tree. The construction of a (primal) spanning tree can keep contiguity in political redistricting problems (Figure 3.3a). In order to construct R districts, the total number of edges in the primal spanning tree is N - R. Similarly, $R-1$ arcs are broken in the primal spanning tree (referred as "broken arc") (Figure 3.3b).

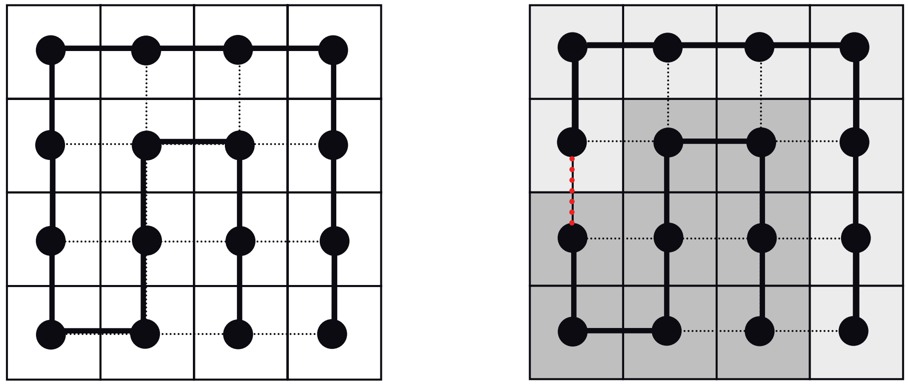


Figure 3.3. The method to model political redistricting problems in the spanning tree based approach. Circles represent vertices on the subtree and lines include edges on the subtree in the graph. Thick lines refer to edges on the spanning tree and red dot lines are broken arcs. Red dot lines should be prohibited edges in order to construct different districts.

Indices and parameters

M = the number of vertices in the dual graph;

k, l, K = the indices and set of dual vertices, where $k, l = 1, \ldots, M$;

D_k = the set of dual vertices that are adjacent to dual vertex k;

Decision variables

$$X_{ij} = \begin{cases} 1 & \text{if direct arc } (i, j) \text{ in the primal graph is selected for the primal spanning tree} \\ & \text{and is also selected for the subtree} \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1 & \text{if direct arc } (i, j) \text{ in the primal graph is selected for the primal spanning tree} \\ & \text{but is not selected for the subtree} \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{kl} = \begin{cases} 1 & \text{if direct arc } (k, l) \text{ in the dual graph is selected for the complemenary dual spanning tree} \\ 0 & \text{otherwise} \end{cases}$$

$$A_{ijr} = \begin{cases} 1 & \text{if primal vertices } i \text{ and } j \text{ belong to district } r \\ 0 & \text{otherwise} \end{cases}$$

$$V_{ir} = \begin{cases} 1 & \text{primal vertex } i \text{ is in districtc } r \\ 0 & \text{otherwise} \end{cases}$$

Model formulation

$$\text{Minimize} \max_{r=1}^{R} \left| \sum_{i=1} P_i V_{ir} - \overline{P} \right| \frac{100}{\overline{P}} \qquad (1)$$

Subject to:

$$\sum_{j \in D_i} X_{ij} + \sum_{j \in D_i} Y_{ij} = 1 \qquad \forall i = 1, \ldots, N-1 \qquad (2)$$

$$\sum_{l \in D_k} Z_{kl} = 1 \qquad \forall k = 1, \ldots, R-1 \qquad (3)$$

$$X_{ij} + Y_{ij} + X_{ji} + Y_{ji} + Z_{kl} + Z_{lk} = 1 \qquad \forall (i, j), i < j, \forall (k, l), k < l \qquad (4)$$

$$\sum_{r=1}^{R} V_{ir} = 1 \qquad \forall i \qquad (5)$$

$$\begin{cases} A_{ijr} \le V_{ir} \\ A_{ijr} \le V_{jr} \end{cases} \qquad \forall (i, j), i < j, \forall r \qquad (6)$$

$$X_{ij} + X_{ji} \le \sum_{r=1}^{R} A_{ijr} \qquad \forall (i, j), i < j \qquad (7)$$

$$\sum_{i \in I} \sum_{j \in D_i} X_{ij} = N - R \qquad (8)$$

$$X_{ij}, Y_{ij}, Z_{kl}, A_{ijr}, V_{ir}, W_{ij} \in \{0, 1\} \qquad \forall i, j, k, l, r \qquad (9)$$

The objective (1) is for equal population and is to minimize maximum deviation from ideal population. Constraints[11] (2), (3) and (4) are formulated by Williams (2001; 2002) specifying basic constructions of a complementary spanning tree in both primal and dual graph. Constraints (5) ensure that one vertex only belongs to one district. Constraints (6) and (7) satisfy district contiguity. Constraints (6) specify that $A_{ijr} = 1$ if and only if units i and j are in the same particular district r. Constraints (7) make sure that the subtree which contains units i and j belongs to the same district. $\sum_{r=1}^{R} A_{ijr}$ is 1 if and only if units i and j are in the same district (or the subgraph). Constraints (8) are developed in Williams (2002) to specify the number of districts. For given R districts, the total number of edges in the subtree is N - R. Instead of constraints (8), the summation of Y_{ij} (the total number of broken edges) equals to R-1 can be used ($\sum_{i \in I} \sum_{j \in D_i} Y_{ij} = R - 1$). Constraints (9) require X_{ij}, Y_{ij}, Z_{kl}, A_{ijr}, and V_{ir}, to be non-negative binary decision variables.

3.2 A network flow model

Shirabe (2005) developed contiguity constraints for land acquisition problems. The contiguity can be ensured by controlling flows in the network in a way that moves from multiple sources to a sink. There is one and only one sink for a contiguous set of spatial

---

[11] Constraints (2) specify that exactly one primal arc must be selected except for the terminal vertex in the primal graph. Constraints (3) indicate that exactly one dual arc must be selected except for the terminal vertex in the dual graph. Constraints (4) require the complementary relationship by ensuring that an arc in the primal graph does not intersect an arc in the dual graph.

units, and at least one flow is coming out from each spatial unit until to reach the arbitrary chosen sink (Figure 3.4).

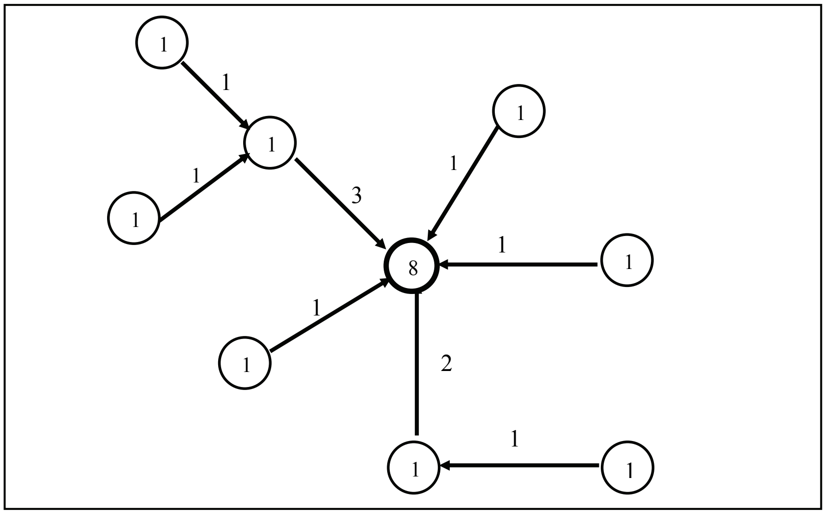


Figure 3.4. A network representation of contiguous spatial units. Circles represent nodes and the thick circle represents a sink in the network. An arrow represents non-negative flows in the network. At least one flow is coming out from each node.

The mechanism of enforcing contiguity which runs flows from sources to only one sink can be applied to ensure contiguity in districts in political redistricting. For political redistricting, contiguity will be enforced in such a way that at least one flow is coming out from each spatial unit in order to reach the arbitrary chosen sink within the district. Then, the supply sent from every source unit must arrive at the sink unit within the district (Figure 3.5a). In order to maintain contiguity within a district, it is also necessary to prevent the inflows from other districts (Figure 3.5b). If there are flows between different districts, real contiguity within the district is not fulfilled because one direction flow from a node can move from a district to another district. As a consequence, modeling contiguity within the district will be designed by not only finding the flow conversing to the sink node, but also controlling the inflows from different districts.

For the construction of several districts, several sub-networks are constructed (Figure 3.5b). The number of sub-networks to be required should be designed as equal number of districts to be needed. Furthermore, each sub-network contains one and only one sink, which means political redistricting problems have multiple sinks instead of a single sink. The number of multiples sinks is also as same as the number of sub-networks. In summary, several *n* districts are designed in a way that specifies several *n* sub-networks and requires several *n* sinks.

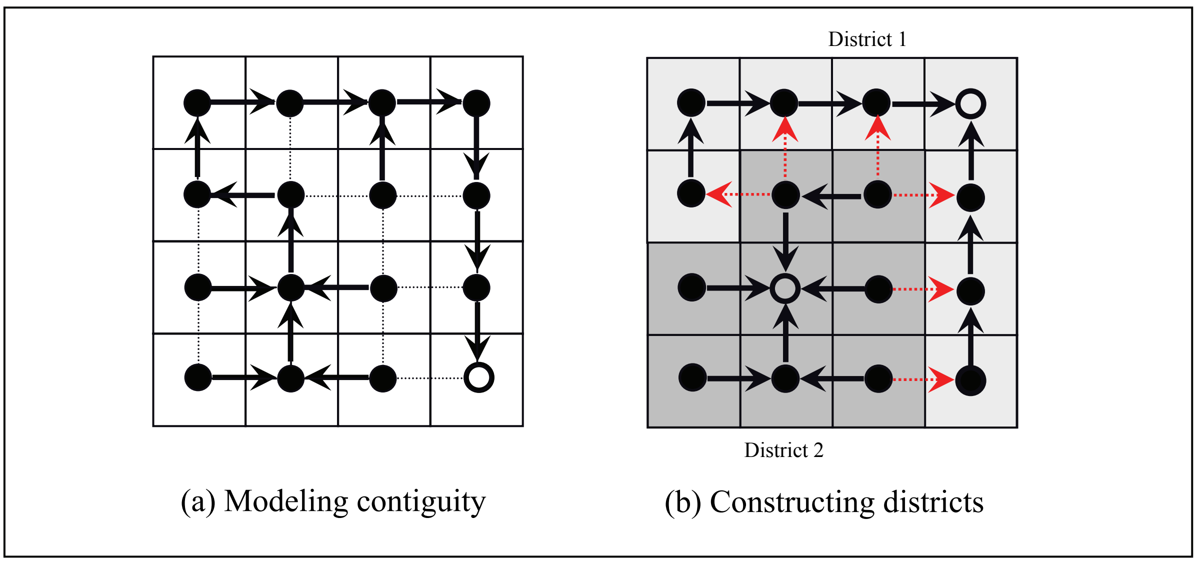


Figure 3.5. The method to model redistricting in the network flow-based approach. Circles represent nodes and lines refer to edges in the network. Thick arrow lines include the flow from a node to another node and red dot arrows are flows between the different districts. Red dot arrow lines should be controlled in the model.

Recently, Shirabe (2009) provides three kinds of models for political redistricting based on previous research of Shriabe (2005). The third model shows the most similarity with network flow model because the model only considers contiguity and population equality, and hubs (sinks) are not known. However, the model specifies the same number of population units in each district which cannot be possible in the reality for population equality. This condition disturbs the strict population equality. Furthermore, Shirabe (2009) introduces insufficient contiguity constraints. There are unnecessary flows between different districts when strict population equality is considered.

Indices and parameters

Q the maximum number of nodes to be chosen in a district ($N - R + 1$)

Decision variables

$$X_{ir} = \begin{cases} 1 & \text{if node } i \text{ is in district } r \\ 0 & \text{otherwise} \end{cases}$$

$$W_{ir} = \begin{cases} 1 & \text{if node } i \text{ is selected as a sink in district } r \\ 0 & \text{otherwise} \end{cases}$$

Y_{ijr} = the non-negative flow from node i to node j in a district r

Model formulation

$$\text{Minimize } \max_{r=1}^{R} \left| \sum_{i=1} P_i X_{ir} - \overline{P} \right| \frac{100}{\overline{P}} \tag{10}$$

Subject to:

$$\sum_{j \in D_i} Y_{ijr} - \sum_{j \in D_i} Y_{jir} \geq X_{ir} - QW_{ir} \qquad \forall i, r \qquad (11)$$

$$\sum_{r} X_{ir} = 1 \qquad \forall i \qquad (12)$$

$$\sum_{r} \sum_{i} W_{ir} = R \qquad (13)$$

$$\sum_{i} W_{ir} = 1 \qquad \forall r \qquad (14)$$

$$\sum_{j \in D_i} Y_{jir} \leq (Q-1) X_{ir} \qquad \forall i, r \qquad (15)$$

$$W_{ir} - X_{ir} \leq 0 \qquad \forall i, r \qquad (16)$$

$$\begin{cases} Y_{ijr} + Y_{jir} \leq (Q-1) X_{ir} \\ Y_{ijr} + Y_{jir} \leq (Q-1) X_{jr} \end{cases} \qquad \forall i, r \qquad (17)$$

$$X_{ir}, W_{ir} \in \{0,1\}, \quad Y_{ijr} \geq 0 \qquad \forall i, j, r \qquad (18)$$

Objective (10) is for equal population and is to minimize maximum deviation from ideal population. Constraints (11) specify the net outflow from each node. Left term respectively indicates the total outflow and total inflow of node *i* in district *r*. If node *i* is included but is not a sink in district *r*, we have $x_{ir} = 1$, $w_{ir} = 0$, and node *i* in district *r* must have a supply ≥ 1. If node *i* is included in district *r* and is a sink, we have $x_{ir} = 1$, $w_{ir} = 1$ and node *i* in district *r* can have a demand (negative net outflow) $\leq Q-1$. If node *i* is not included in district *r* and is not a sink, we have $x_{ir} = 0$, $w_{ir} = 0$ and node *i* in district *r* must have a supply ≥ 0. Finally, if node *i* is not included in district *r* but a sink ($x_{ir} = 0$, $w_{ir} = 1$), the rest of node *i* are forced to be 0, and no node *i* are selected for district *r*. Constraints (12) ensure that each node only belongs to one of districts. Constraints (13) specify the number of nodes that can be used as sinks. Constraints (14) ensure that each district must have only one sink. Constraints (15) ensure that there is no flow into any node i from the outside of district r (where $x_{ir} = 0$), and that the total inflow of any node in district r (where $x_{ir} = 1$) does not exceed $Q-1$. Constraints (16) make sure unless a node i is included in district r, the node i cannot be a sink in district r. Constraints (17) ensure that there is no flows (inflows and outflows) between different districts to ensure eligible contiguity. Constraints (18) require x_{ir} and w_{ir} to be a non-negative binary decision variable and y_{ijr} to be a non-negative decision variable.

How to se the maximum allowable number of units to constitute (Q) in a district is a big issue because depending on Q, the model adjusts the flows in the network, and diminishes redundant flows. Eventually, Q also can make contiguity constraints right working. In the net flow model, Q is set as $N-R+1$, where N is the total number of population units, R is the total number of districts to be divided. So, different data size data has different Q.

3.3 Garfinkel and Nemhauser (1970)

Garfinkel and Nemhauser (1970) develop the exact method of political redistricting problems by an implicit enumeration. The authors address three criteria such as population, contiguity and compactness. For equally populated districts, a district is feasible only if its population falls within a specified range centered on the mean district population. For contiguous districts, a district is contiguous if population units are connected within a district. In this paper, compactness is defined as geographical compactness with distance compactness and shape compactness. For distance compactness, a district is feasible only if the distance between population units must be less than a specified upper bound (an absolute measure). For shape compactness, a district is feasible only if the square of the distance's maximum diameter divided by the district's area must be less than another upper bound (a relative measure).

They developed an exact method in redistricting plans with two phases. Phase I exhaustively generates all feasible districts which satisfy equal population, contiguity and compactness based on tree search algorithm. First, a seed unit is randomly selected. Then adjacent units are combined until population range is the satisfied. In this process adding adjacent units, compactness and population are considered. The algorithm backtracks on the tree when populations are out of the upper bound of population range to check other units. The algorithm also has a checking process if the generation results in the form of enclave which cannot be feasible. In the below optimization equations, variable $\{a_{is}\}$is determined in the phase I of the model of feasible districts.

All enumerated feasible districts can be used for the optimization phase (Phase II). Phase II finds an optimal solution in a way that minimizes the maximum deviation from the mean district population in any chosen district. Phase II is mathematically expressed in a mixed integer programming as following;

Variables and indices

s, S = indices and set of feasible districts generated by Phase I of the model

$a_{is} = \begin{cases} 1 & \text{If spatial unit } i \text{ is in feasible district } s \\ 0 & \text{otherwise} \end{cases}$, which will be determined in the phase I

of the model.

$P_S^{'} = \sum_{i=1}^{n} a_{is} P_i$ is the total population of district s

$c_S = \left| P_S^{'} - \overline{P} \right|$ is the population deviation of district *s* from the ideal population

Decision variables

$T_s = \begin{cases} 1 & \text{if district } s \text{ is selected in the plan} \\ 0 & \text{Otherwise} \end{cases}$

Model formulation

$$\text{Minimize} \quad \max_{s=1}^{S} c_s T_s \frac{100}{\overline{P}} \tag{19}$$

Subject to:

$$\sum_{s=1}^{S} a_{is} T_s = 1 \qquad \forall i \tag{20}$$

$$\sum_{s=1}^{S} T_s = R \tag{21}$$

$$T_s \in \{0, 1\} \qquad \forall s \tag{22}$$

The objective (19) is about equal population and is to minimize the maximum deviation of any district population from ideal population. Constraints (20) ensure that each spatial unit is assigned to one and only one feasible district for the plan. Constraints (21) specify that *R* feasible districts will be selected. Constraints (22) require the binary decision variable.

3.4 Computational experiments

The experiments of the exact methods were conducted on a quad-core Xeon 2.8 GHz computer with 8 GB memory. A solver called CPLEX (version 11) is used to the problem; this is a parallel version with the deterministic mode and the opportunistic mode with threads in order to enhance CPU time. For the computational intensity of the exact models, the program is terminated if it cannot return an optimal solution within a day (1 day = 86400 seconds).

3.4.1 Test data description

Two types of data sets are used to test the models. The first type is a regular grid (5x5, 10x10, 10x5 and 25x40 data) in Figure 3.6. Each cell represents artificial population units and populations are randomly generated. The second type is an irregular grid which is real data, the 99 counties of Iowa where the counties will not be split by Iowa Constitution. For year 2000, five congressional districts are to be determined.

19	13	19	20	14
12	18	16	15	18
16	16	12	10	13
18	16	11	12	13
15	19	13	19	17

(a) 5x5 data

34	38	21	45	19
49	17	12	45	41
22	35	24	10	19
30	39	50	31	30
15	45	20	36	37
34	15	30	19	42
15	23	46	14	30
26	50	30	29	24
47	49	14	10	43
24	15	23	23	47

(b) 10x5 data

34	38	21	45	19	40	28	41	32	47
49	17	12	45	41	26	23	26	20	39
22	35	24	10	19	28	46	25	28	20
30	39	50	31	30	49	29	48	18	20
15	45	20	36	37	36	26	22	31	33
34	15	30	19	42	47	17	18	15	48
15	23	46	14	30	41	12	27	20	25
26	50	30	29	24	25	28	18	35	13
47	49	14	10	43	32	14	32	26	31
24	15	23	23	47	49	41	16	34	21

(c) 10x10 data

continued

Figure 3.6. Test data of a regular grid

Figure 3.6 continued

80	89	98	60	93	90	58	74	77	96	58	83	83	54	53	80	74	66	70	91	85	71	50	78	50	84	55	65	50	56	63	88	91	79	94	80	69	72	58	51
79	58	98	86	96	73	85	73	58	97	70	96	58	94	88	75	92	59	61	61	53	64	52	99	74	83	98	73	95	89	79	69	79	66	88	61	55	59	78	94
100	90	75	84	54	81	71	51	74	72	80	53	93	98	89	94	72	99	70	80	62	90	91	76	64	69	70	78	96	88	65	85	99	72	89	97	73	74	56	93
69	59	76	52	75	61	51	72	56	84	55	61	67	53	78	72	83	72	81	87	90	72	82	84	72	64	79	54	91	67	75	61	55	51	82	82	89	85	79	85
54	55	89	64	87	95	60	54	50	68	70	99	70	59	58	66	70	89	82	56	79	67	74	70	53	99	71	78	81	70	83	86	81	94	61	74	97	54	87	95
80	96	94	80	61	100	95	54	93	86	73	86	72	94	70	63	99	56	75	70	51	67	55	94	86	62	83	75	88	61	61	71	55	59	71	89	83	98	84	93
84	77	74	57	84	69	86	79	51	72	82	80	55	70	98	81	68	76	63	84	100	67	93	81	62	81	51	80	80	85	50	66	82	79	79	83	62	94	97	54
62	57	61	80	95	64	68	77	50	75	59	73	62	85	73	70	74	61	74	84	79	78	80	69	93	77	50	54	91	94	90	50	76	51	83	53	60	99	87	68
57	61	97	79	82	84	65	86	58	55	82	73	68	55	72	79	88	99	96	73	89	91	69	51	82	55	98	69	71	66	96	76	55	52	80	74	96	61	61	99
71	80	63	64	85	68	95	76	83	97	50	71	59	50	69	61	73	97	86	94	97	62	63	58	71	84	58	75	80	54	79	56	54	52	86	58	89	100	51	81
86	72	80	61	97	80	83	86	62	99	59	62	80	82	66	93	62	74	73	100	56	87	60	51	93	51	71	52	69	75	64	90	80	63	93	90	52	80	85	64
76	90	54	91	89	80	92	85	92	91	58	68	59	93	90	61	75	63	71	67	63	75	67	69	63	92	91	85	82	83	70	82	77	95	97	97	95	88	91	66
98	91	88	70	84	77	87	97	83	67	93	73	97	72	51	68	74	87	57	55	85	83	90	61	100	73	85	89	62	61	77	63	71	84	58	75	61	54	95	54
97	63	90	56	50	84	98	64	93	84	66	67	52	69	64	83	56	82	58	81	77	89	70	81	62	99	52	70	56	75	55	66	61	77	89	52	94	80	60	89
62	81	70	68	71	61	84	62	77	79	66	67	66	69	52	75	70	57	81	51	84	86	72	72	66	94	88	94	75	61	83	64	86	65	61	99	50	100	66	62
65	71	91	70	80	68	95	93	99	81	53	89	90	84	60	76	97	95	98	91	81	76	59	52	67	85	87	73	90	82	71	55	62	96	57	52	66	58	99	87
60	91	72	87	62	88	99	83	64	67	84	82	95	73	67	52	74	67	60	67	99	55	61	95	87	65	60	74	67	62	69	83	96	87	61	62	58	99	69	81
84	84	68	53	87	65	65	95	81	64	94	68	66	69	80	65	91	52	76	52	60	77	81	59	54	88	91	72	55	53	74	88	94	61	75	90	73	87	84	75
94	73	95	57	72	57	63	61	68	60	51	91	65	99	70	90	91	53	96	100	88	76	67	63	57	61	94	82	55	67	54	54	73	93	77	83	82	54	95	84
100	62	60	58	71	88	64	56	100	58	99	65	95	78	74	70	96	59	76	51	85	86	84	83	99	89	83	61	89	66	66	62	79	76	64	72	55	55	60	79
76	50	53	76	89	77	68	92	97	65	88	73	55	82	93	50	61	88	97	60	78	52	87	53	79	57	55	65	88	80	59	94	79	59	72	62	64	91	63	55
54	94	100	75	74	64	100	54	56	70	84	100	51	64	99	79	83	72	96	56	85	55	99	92	94	61	88	55	76	57	62	96	80	77	68	65	69	82	57	56
56	78	58	91	88	72	84	95	98	52	79	87	93	90	78	91	95	67	98	83	56	68	53	100	72	97	68	83	58	53	62	52	76	93	63	85	90	75	58	65
76	94	78	76	76	61	81	86	88	83	59	57	78	53	58	69	69	85	82	86	52	59	86	75	54	88	86	54	56	95	99	57	53	78	80	76	76	77	52	69
50	61	64	86	82	50	71	73	94	77	85	71	56	58	93	83	65	62	89	70	89	60	80	89	56	62	80	91	98	99	98	78	50	59	70	56	79	82	88	69

(d) 25x40 data

3.4.2 Test results

Phase I of the existing exact method is coded with short integer of C++ in order to consider computer memory. It has several factors to generate possible feasible districts (Table 3.1). The algorithm has *e* as 0 at first and increases or decreases it with an increment of 1 during the algorithm process. When *e* equals EN, the algorithm starts to check if there is an enclave. The program generates all feasible districts with respect to 1% population deviation and least compactness. In the exclusion matrix of the method, all units are candidates for generation (all units = 1) so that the algorithm can take care of all units (Figure 3.7).

Parameters	Setting Value
EN	5
ALPA (α)	0.01
BETA (β)	10000

EN: The constant to say how often the an enclave check is performed

α : The constant to decide the range of population deviation ($0 \le \alpha \le 1$), α close to 0 refers to strict population deviation.

β : The constant to refer to the compactness range ($0 \le \beta \le \infty$), β close to 0 is the most compact shape

Table 3.1. The parameter setting in Phase I of Garfinkel and Nemhauser (1970)

$$\boldsymbol{Z} = \mathbf{1} \qquad \Rightarrow \qquad \boldsymbol{a}_{ij} = \mathbf{1} \qquad \forall \boldsymbol{i}, \boldsymbol{j}$$

Figure 3.7. Exclusion Matrix

All exact methods find optimal solutions using the sequential optimizer, and the parallel optimizer (the deterministic mode and the opportunistic mode); the sequential optimizer runs on a single central processing unit (CPU) and the parallel optimizer simultaneously uses multiple CPUs to save CPU time, and provide concurrency. For the parallel optimizer, the opportunistic mode with eight threads and the deterministic mode with eight threads are used. The deterministic mode repeatedly finds the solution of the model with the same parameter settings on the same computing platform producing the same level of performance. The opportunistic mode differently finds the solution, however, yielding better performance.

Table 3.2 lists results of all models using parallel optimizer using opportunistic mode with eight threads for the three exact methods, which shows efficiency in performance. All methods find optimal solutions for 5x5 data. However, it is clearly known that the exact models show computational difficulty to find optimal solutions for many problems (10x5, 10x10 and 25x40 data); for 10x10 data with 4 districts, only deterministic mode finds the optimal solution while opportunistic mode cannot. For Iowa data, all exact methods do not solve the problem. As a consequence, all exact methods are inefficient to find optimal solutions. Figure 3.8 to Figure 3.10 show spatial configuration results of all three models. All results from the use of the sequential optimizer and the parallel optimizer (two modes) are reported. New approaches produce more different spatial shapes than the existing research.

The existing method takes less time to generate all feasible districts for large districts (e.g. 4 districts in 5x5 data) and much time to do for small districts problem (e.g. 2 districts in 5x5 data) and also computing time of optimization phases has same tendencies; In the phase of feasible district generations, CPU time of 4districts in 5x5 data is 5.85 seconds and CPU time of 2districts in 5x5 data is 184.39 seconds. It is because the existing method is based on feasible district generations and the generation complexity for small districts is much more than that for the large districts (Table 3.3). However, newly proposed models (spanning tree based methods and network flow based method) show the opposite tendencies which it needs less time to solve small districts problem (e.g. 2 districts in 5x5 data) and much time for large districts problem (e.g. 4 districts in 5x5 data).

In summary, two exact models to political redistricting are developed on a basis of using the solution approach of land acquisition problems, especially for focus on contiguity formulation. It is meaningful that the formulation of the redistricting problem as an optimization model contributes to our understanding of spatial organization and represents a fruitful GIScience research area. In our experiments, two exact methods are compared with the existing method of Garfinkel and Nemhauser (1970). Though all methods may find optimal solutions for small data, all exact models are complicated to find the redistricting plans for real cases. To overcome computing issue to exact methods, it is natural that a significant amount of efforts should have been conducted to develop heuristic methods which can be used to efficiently search for high-quality solution.

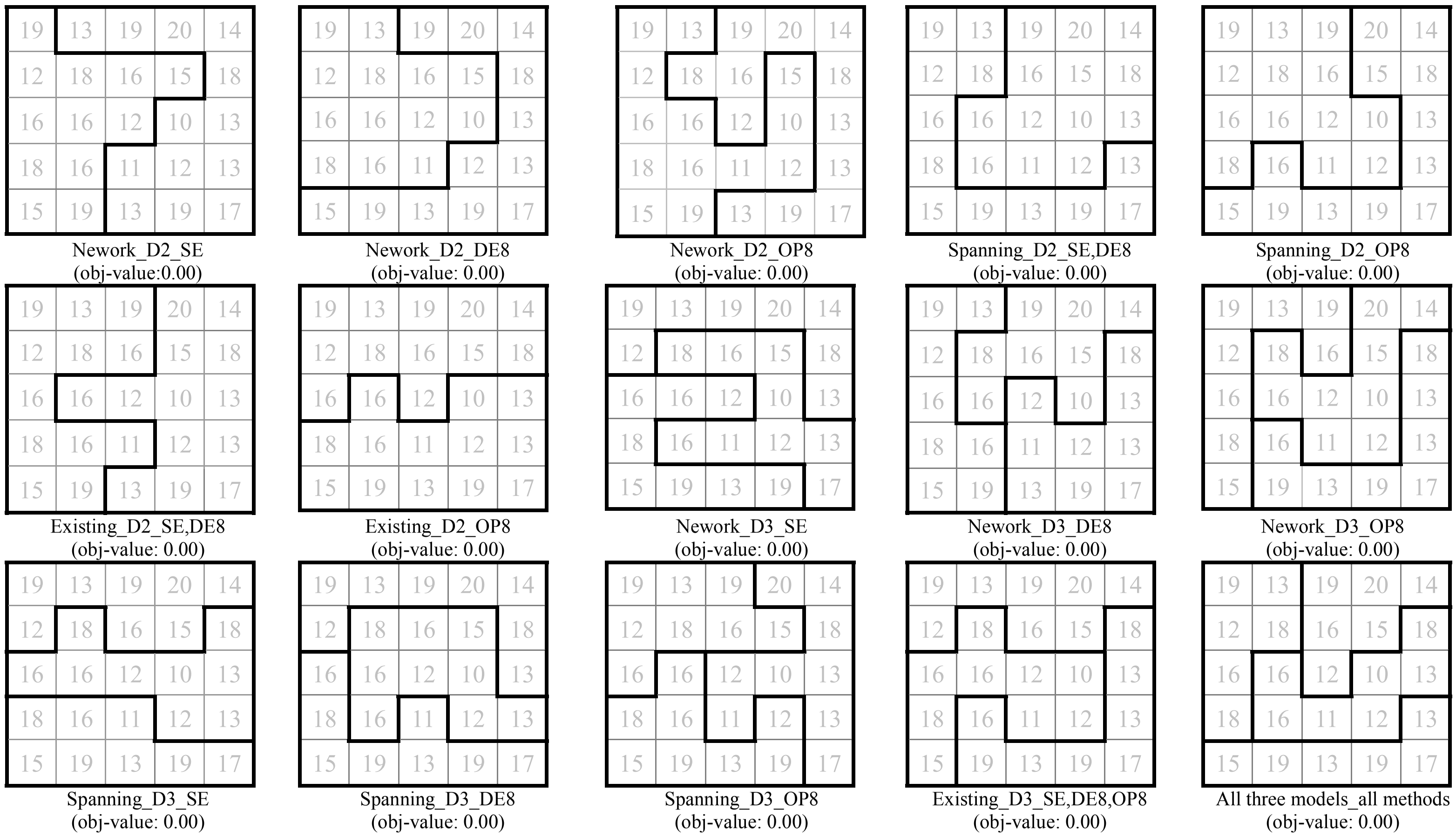


Figure 3.8. Redistricting results of 25 units in 5 by 5 data with 2, 3 and 4 districts for each exact model
Notation Example:
Spanning_D3_SE - spanning tree based approach with 3 districts using the sequential optimizer.
All three models_all methods – spanning tree, network flow and existing models with the sequential and parallel optimizers.

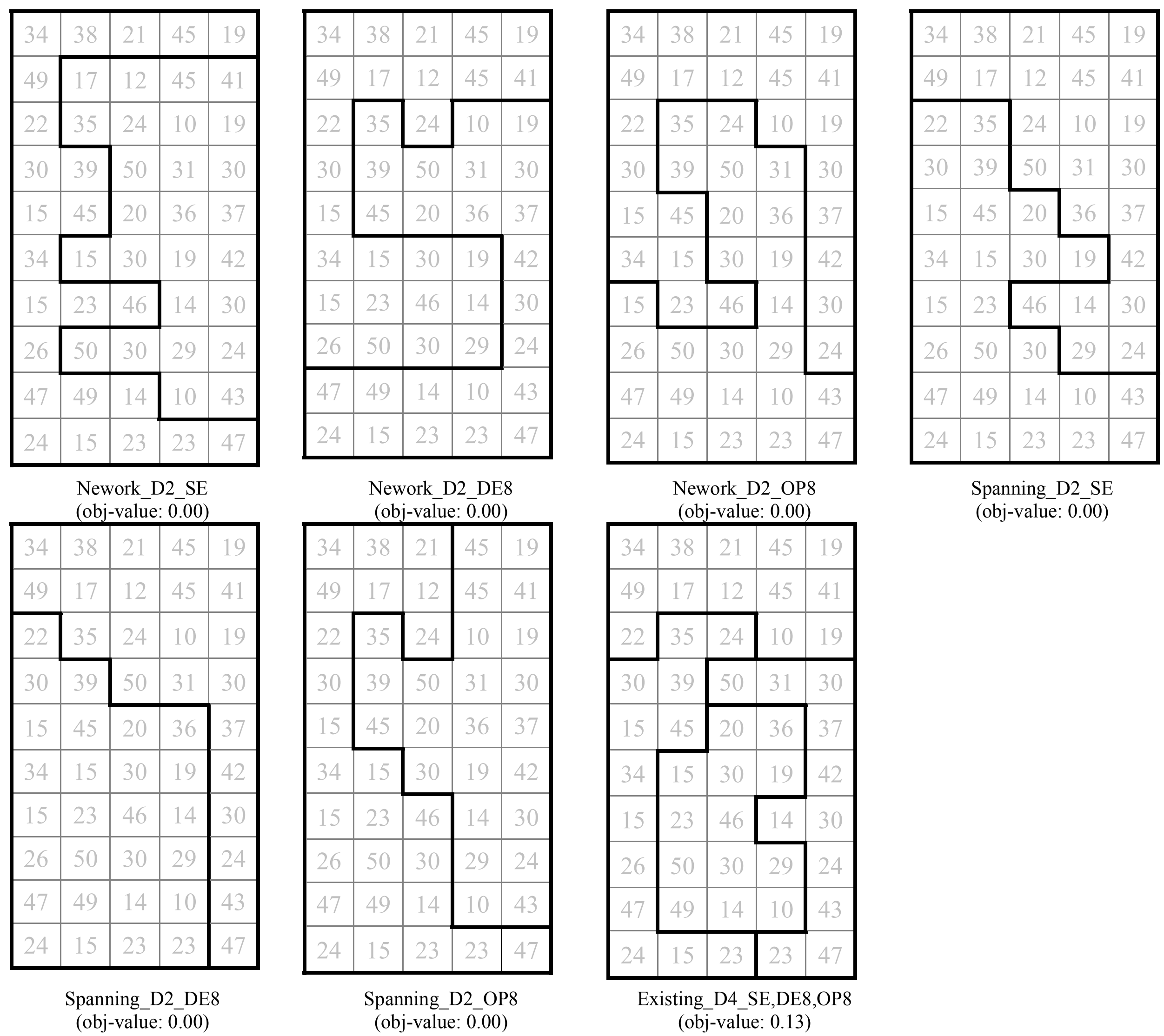


Figure 3.9. Redistricting results of 50 units in 10 by 5 data with 2 and 4 districts for each exact model

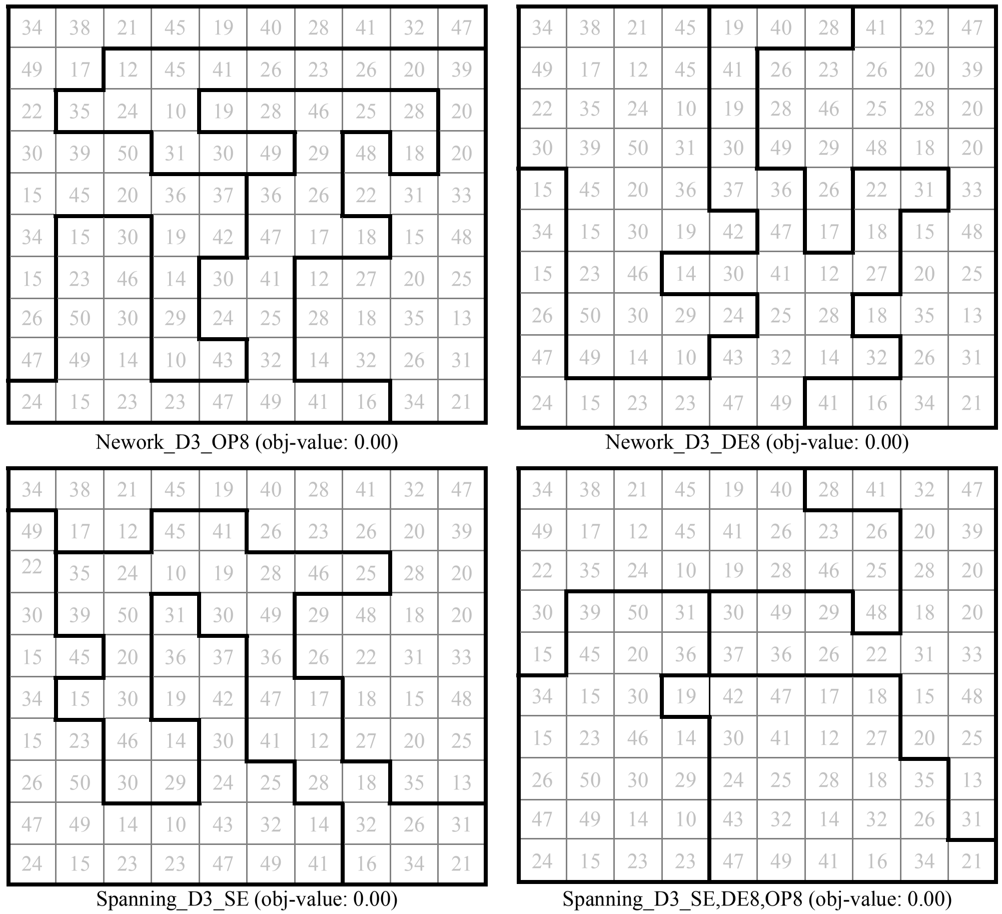


Figure 3.10. Redistricting results of 100 units in 10 by 10 data with $r = 3$ for each exact model.
Notation Example: Spanning_D3_DE8 - spanning tree based approach with 3 districts using the parallel optimizer (deterministic mode with 8 threads).
Spanning_D3_OP8-spanning tree based approach with 3 districts using the parallel optimizer (opportunistic mode with 8 threads).

n[a]	r[b]	Network based model				Spanning tree based model				Garfinkel and Nemhauser (1970)			
		(a) The first feasible solution		(b) The last feasible solution		(a) The first feasible solution		(b) The last feasible solution		Phase I**		Phase II**	
		obj-value	Time (sec.)	obj-value	Time (sec.)	obj-value	Time (sec.)	obj-value	Time (sec.)	#ofDis.[c]	Time(sec.)	obj-value	Time (sec.)
5x5	2	91.15	0.01	0.00	0.39	7.29	0.03	0.00	0.26	497	184.39	0.00	0.96
	3	177.34	0.12	0.00	1.90	36.72	0.33	0.00	17.56	507	40.13	0.00	0.31
	4	251.04	0.62	0.00	834.24	50.00	0.28	0.00	17199.96	135	5.85	0.00	0.79
10x5	2	93.67	0.01	0.00	3.43	84.93	0.08	0.00	0.04	*472084*	*185816.00*	-	-
	3	190.11	0.28	*0.13*	*86495.11*	57.20	0.41	*0.13*	*86472.55*	*598062*	*210561.00*	-	-
	4	271.74	0.51	*0.13*	*86482.08*	225.17	0.75	*0.13*	*86520.39*	551165	95734.30	0.13	24097.08
10x10	3	196.65	1.14	0.00	26298.05	194.41	1.88	0.00	1022.56	*207127*	*173683.00*	-	-
	4	290.51	2.68	*7.45*	*89495.38*	278.86	2.86	0.00**	152.47**	*247067*	*173688.00*	-	-
	5	380.69	37.86	*0.78*	*87394.41*	191.16	5.58	*28.73*	*87217.49*	*288792*	*173609.00*	-	-
25x40	5	397.70	3914.70	*384.17*	*106877.02*	0.18	0.84	*0.00*	*87044.44*	*45*	*174156.00*	-	-
	10	890.97	24637.74	*875.95*	*86400.00*	0.38	1.60	*0.04*	*96048.77*	*185*	*172929.00*	-	-
	20	1859.47	5768.09	*1814.04*	*86400.00*	1.61	4.75	*0.11*	*87916.00*	*634*	*173097.00*	-	-
Iowa	5	335.29	9.38	*335.29*	*87771.81*	307.31	1.15	*0.09*	*86444.29*	*443269*	*173700.00*	-	-

a: data type
b: the number of districts
c: the number of districts to be generated
**: results from deterministic model with 8 threads
Note: Italic numbers reflect intermediate results not final optimal results. As the program is running in a day, it reports intermediate results.

Table 3.2. Results from the three exact models

The total number of units	The number of district	The possible number of units assigned in each district	Complexity**
12	2	6	924
	3	4	495
	4	3	220

**: population of each unit is uniform and contiguity is not considered.

Table 3.3. The complexity of feasible district generations in Phase I of Garfinkel and Nemhauser (1970)

Chapter 4 Heuristic optimization models to political redistricting problems

From previous computational experiments using exact methods in Chapter 3, it is clearly known that exact methods show computational challenges and it is necessary to incorporate a heuristic optimization technique. Heuristic approaches cannot guarantee that solutions can reach the global optimum. Nevertheless, recent approaches have demonstrated that heuristics can be used to find near-optimal or optimal solution (Cooper, 1964). There are many meta-heuristics based on simulated annealing (Kirkpatrick et al. 1983), evolutionary algorithms (Back et al., 1997) and tabu search (Glover, 1977). The new heuristic should be both efficient in computing time and effective in finding high-quality solution.

This chapter implements two heuristic optimization approaches and compares them with the existing approach using evolutionary algorithms by Xiao (2008). The first section introduces a multi-scale simulated annealing heuristic approach and the second section discusses the development of a greedy algorithm called Give-And-Take greedy algorithm. The third section describes the applications of the Give-And-Take greedy algorithm to find various spatial shapes similar to the given plan, and explains the sensitivity analysis of the Give-And-Take greedy algorithm. The fourth section shows experiments of both heuristic models and the applications of Give-And-Take greedy algorithm.

4.1 Multi-scale simulated annealing (MSA algorithm)

Simulated annealing is known as not only a search technique that can be useful for combinatorial optimization problems, but also a popular tool where mathematical programming formulations are intractable to solve. Simulated annealing algorithms are search procedures based upon the thermodynamic process[12] of annealing metals (Kirkpatrick et al. 1983; Cerny 1985; Laarhoven and Aarts 1987). These algorithms may accept non-improving solutions in order to escape local optimal solutions (Ulungu et al. 1999).

4.1.1 A simulated annealing

Simulated annealing (SA) is one of metaheuristic methods (Kirkpatrick et al. 1983), which can be used to find optimal or near-optimal solutions to complex optimization problems (see David et al. 1989). In general, the algorithm maintains a current solution and, based on that, generates a neighborhood of the current solution from which a new

[12] This process heats the metal to a high temperature so that atoms can move relatively freely. The temperature of the metal is slowly lowered so that at each temperature the atoms can move enough to begin adopting the most stable orientation. If the metal is cooled slowly enough, the atoms are able to relax into the most stable orientation. This slow cooling process is known as annealing, and so their method is known as Simulated Annealing.

solution is selected. To improve the solutions found, any better solutions from the neighbourhood will be accepted. However, the algorithm also accepts a worse solution with a probability, which is diminishing during the process, in order to escape the local optimal solution. The algorithm stops when no solutions can be accepted (Figure 4.1).

Algorithm SA {*General procedure*}
1. Initialize a solution x
2. $T_0 = T$, where T is temperature
3. **repeat until** a stop condition is satisfied
4. Construct a new solution $y \in V(x)$, $V(x)$ is the neighborhood of x
5. Evaluate solutions x and y
5.1. If y is better than x, then $x := y$
5.2. Else $x: = y$ with probability $P(x, y, T)$
6. Decrease T

Figure 4.1. The general process of simulated annealing

This dissertation develops simulated annealing heuristic using a multi-scale decreasing method to political redistricting. The algorithm is called a multi-scale simulated annealing. The multi-scale simulated annealing for political redistricting (Algorithm MSA) is described in Figure 4.2. For the initial solution, the algorithm generates a solution by the process of randomly selecting a seed unit (step 1.3) and adding adjacent units to the seed unit until no more units are added (step 1.4 and step 1.5). A new solution can be founded by selecting population units from eligible candidate adjacent units, which can be possibly moved from a district to any other district without breaking contiguity (step 3.4 and step 3.5), rechecking contiguity among the selected population units (step 3.6), and moving the selected population units from a district to another district (step 3.7). For evaluations, improved new solutions will be accepted. However, the algorithm also accepts non-improved solutions with an acceptance probability in order to escape local optimal solutions (step 3.10). When the algorithm decreases temperature, a multi-scale decreasing method is used in this dissertation and the detailed explanations are described in section 4.1.2.3 (step 3.11).The algorithm stops when terminal conditions can be accepted.

Algorithm MSA {*Multi-scale simulated annealing to political redistricting*}
Input: Adjacent matrix, population, T (temperature)
Output: y (new solution)
1. Initial solution x
  1.1. $i = 0$
  1.2. **repeat until** $i = r - 1$
  1.3. Randomly select a seed unit
  1.4. Find adjacent units from the seed unit
  1.5. Add adjacent units to the seed unit until there are no units to be added
  1.6. Calculate objective function value of x
  1.7. $i := i + 1$
2. $T_0 := T$
3. **repeat until** a user specified stop condition is fulfilled
  3.1. Construct the new solution y
  3.2. $i = 0$
  3.3. **repeat until** $i = r - 1$
  3.4. Find eligible candidate adjacent units to a district [r] (ECA units) using Algorithm PI
  3.5. Randomly select the number of population units to be specified from ECA units
  3.6. Recheck contiguity among the selected population units using Algorithm P2
  3.7. Moved the selected population units to the district [r]
  3.8. $i = i + 1$
  3.9. Calculate objective function value of the new solution y
  3.10. Evaluate the current solution x and the new solution y
  3.10.1. If y is better than x, then $x := y$ (accept y)
  3.10.2 Else $x := y$ (accept y) with probability P (x, y, T)
  3.11. Decrease T using multiple cooling rates

Figure 4.2. The process of multi-scale simulated annealing

4.1.2 Cooling schedule in the MSA algorithm

There are four essential factors to constitute an annealing cooling schedule;

(1) the initial temperature

(2) the number of iterations to be performed at each temperature called Markrov chain length

(3) temperature change called the decreasing rate, and

(4) the stopping conditions such as final temperature or maximum number of iterations without improvement (Ulungu *et al.* 1999; Eglese 1990).

The initial temperature is desirable at sufficiently high temperature so that all transitions are accepted, and the typical value regarding decreasing rate is less than 1 and between 0.95 and 0.99. The number of iterations is related with the theory of Markrov chains and is determined by a sufficient number of transitions so that all neighbors of the current solutions can be investigated at each value of temperature. Other simple schemes keep the number of iterations constant or take a multiple of the average size of a neighborhood. Lastly, it is important to have the stopping condition so that the algorithm may not be stopped too early (Pirlot 1996; Eglese 1990).

4.1.2.1 Initial temperature

In the political redistricting problem, the initial temperature is first decided by the method of White (1984) who proposed that the initial temperature is just within one standard deviation of the mean cost. The cost is usually an objective function value and the mean cost is the mean value of the objective function values. Initial temperature is decided by randomly generating 10000 feasible solutions, calculating the mean objective function value and standard deviation from solutions for the use of the initial temperature (Table 4.1).

n^a	r^b	Mean[c]	SD[d]	Mean-SD	Mean+SD	Initial Temperature
5by5	2	7.53	8.28	-0.75	15.81	15.81
	3	15.31	10.74	4.57	26.05	26.05
	4	20.09	11.47	8.62	31.56	31.56
10by10	3	12.27	10.33	1.94	22.60	1.94
	4	16.50	10.86	5.64	27.37	5.64
	5	19.11	11.03	8.08	30.14	8.08
25by40	5	16.25	10.26	5.99	16.51	5.99
	10	22.47	9.98	12.50	32.45	32.45
	20	25.28	8.65	16.63	33.94	16.63
Iowa	5	43.12	12.19	30.93	55.31	55.31

a: data type *b*: the number of districts
c: the mean value of the objective function value *d*: standard deviation

Table 4.1. Initial temperature in the MSA algorithm

4.1.2.2 Markrov chain length

The number of iterations to be performed at each temperature is related with the theory of Markrov chain and called Markrov chain length. For simplicity in the dissertation, a constant Markrov chain length of 1 is used in Algorithm MSA.

4.1.2.3 Multiple cooling rates

A great variety of cooling schedules have been used in the literature. Especially, Abramson et al. (1999) use six different cooling schedules of simulated annealing for solving the school time tabling problem; the basic geometric cooling rate (single cooling rate), multiple cooling rates, geometric reheating, enhanced geometric reheating, non-monotonic cooling, and reheating as a function of cost. Experimental results show that the use of multiple cooling rates gives better quality solutions in less time than that of a single cooling rate.

The single cooling rate is described by a proportional temperature function such as $T_{t+1} = \alpha T_t$, where α is a constant less than 1. Therefore, at high temperature all exchanged solutions are accepted even though most of them are not valuable. This simple cooling scheme may not search the solution space in an effective way depending on the problem difficulty. The use of different cooling rates may take less time at high temperature by fast cooling, and take more time to explore solution at low temperature by slow cooling.

In order to use different rates, it is necessary to decide the phase transition (or temperature transition) by calculating the specific heat of the substance. Specific heat is described by Laarhoven and Aarts (1988) and Abramson et al. (1999). When the specific heat is maximal, this starts to reorder the state. In order to compute the specific heat temperature, we observe $\frac{\sigma_c^2}{T}$ by calculating the variance of the objective function (σ_c^2)

over the number of trials at a particular temperature, T (Figure 4.3).

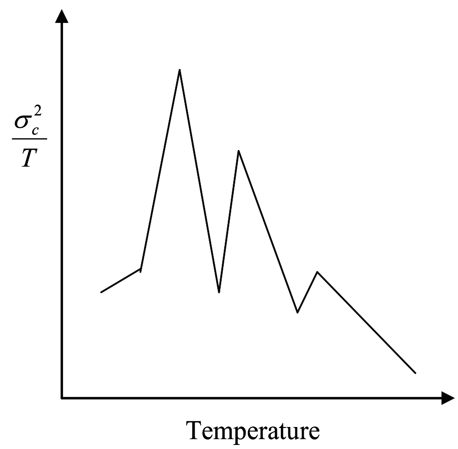


Figure 4.3. Specific heat over temperature

The method of multiple cooling rates can be described below.

$$T_{k+1} = \alpha T_k \quad \text{if } T_k > T_{msp}$$

$$T_{k+1} = \beta T_k \quad \text{if } T_k \leq T_{msp}$$

Where T_{msp} is the temperature at the maximum specific heat occurs, and α and β are constants less than 1 ($\alpha < \beta$). Temperatures are also decreased in the algorithm process using two different cooling rates. Two different cooling rates α can be 0.5 for fast cooling and β can be between 0.95 and 0.99 for slow cooling.

The specific heat in this dissertation is decided by calculating the variances of the objective function with the number of trials equal to 3 at each temperature T. After getting the specific heat, the decreasing rate of α is set as 0.5 for fast cooling and the decreasing rates of β is applied from 0.95 to 0.985 with the increment of 0.05 in a various way for slower cooling. Thought several trials using different β s, the most appropriate β to reach high quality solutions are decided (Please see Table 4.2).

4.1.2.4 The stopping condition

Stopping temperature is used for stopping condition in this dissertation. It is decided by an empirical analysis, which solves many trials to find appropriate cooling schedules (Table 4.2). Stopping temperature of political redistricting problems is set from 1.E-07 to 0.1 with the increment of 0.1 for 5 by 5, and 10 by 10 data, and very low values such as 1.E-21 for Iowa data, and 1.83E-50 for 25 by 40 data in order to find high quality solutions.

4.1.3 Ensuring contiguity in the MSA algorithm

The basic idea of ensuring contiguity in the MSA algorithm heuristic is to find movable units among adjacent units. Normally, adjacent units to each district include both non-movable units and movable units. Non-movable units refer to units, which break contiguity when they are chosen for the new solution. Movable units include units, which still maintain contiguity at all districts after adding from a district to any other districts.

During each iteration in an algorithm, two procedures are used to find movable units. The first procedure is to find eligible candidate adjacent units from adjacent units, which can be possibly moved from a district to any other districts. The second procedure is called contiguity rechecking procedure, which examines if the selected population units are really movable. So, MSA algorithm finds eligible candidate units using Algorithm P1 and selected a certain number of population units can be selected from eligible candidate adjacent units. Then MSA algorithm rechecks if the selected population units are really movable using Algorithm P2.

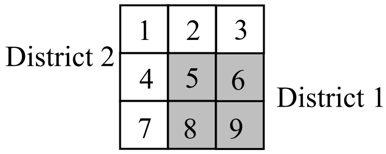
4.1.3.1 Finding eligible candidate adjacent units

The first procedure finds eligible candidate adjacent units using Algorithm P1 (Figure 4.4). Eligible candidate adjacent units refer to units which can be possible selected from adjacent units for a new solution. First, the algorithm finds adjacent units that have a common border to a district from adjacency matrix (step 3). Second, the algorithm selects non-movable units and excludes them from adjacent units (step 5). Thereby the algorithm finds eligible candidate adjacent units, which can be possibly selected for a new solution, by remove non-movable units from adjacent units (step 6).

In the algorithm process, non-movable units can be found from sub-adjacent matrix, which is consisting of population units in a district, extracted from adjacent matrix. In the sub-adjacent matrix, the algorithm finds the unit that has only one and one adjacent unit in the certain district (denoted as "the referenced unit"). The unit right adjacent to the referenced unit is a non-movable unit. Non-movable units can be founded while there is referenced unit and the number of unit except the referenced unit >2 (step 5). For example, there is a 3 by 3 regular grid with 2 districts (Figure 4.5a) Adjacent units to District 1 are unit 2, 3, 4, and 7. In order to know non-movable units and eligible candidate adjacent units among these adjacent units, in the sub-matrix of District 2, reference units are unit 3 and 7. Non-movable units are unit 2 and 4 right next to the reference units since they break contiguity of District 2 when they are moved (Figure 4.5b). Amongst these adjacent units, eligible candidate units are unit 3 and 7 that can be taken into account for the new solution (Figure 4.4 and Figure 4.5).

```
Algorithm P1 {Finding eligible candidate adjacent units}
Input: Adjacent matrix
Output: Array of eligible candidate adjacent units
1. i := 0, j = 0
2. Repeat until i = r-1
3.     Find adjacent units to district [i]
4.     Repeat until j = l, where l is not equal to i
5          If (the number of adjacent units > 2)
                 While(# of the referenced unit >0) and (# of units except the referenced unit > 2)
                       Find non-movable unit
6.     Remove non-movable units from adjacent units.
7. i = i + 1
```

Figure 4.4. The procedure of PI to find eligible candidate adjacent units



(a) two district example

unit	1	**2**	3	**4**	7	SumOfRow
1	0	1	0	1	0	2
2	1	0	1	0	0	2
3	0	1	0	0	0	1
4	1	0	0	0	1	2
7	0	0	0	1	0	1

(b) sub-adjacent matrix of District2

unit	1	2	3	4	5	6	7	8	9
1	0	1	0	1	0	0	0	0	0
2	1	0	1	0	1	0	0	0	0
3	0	1	0	0	0	1	0	0	0
4	1	0	0	0	1	0	1	0	0
5	0	1	0	1	0	1	0	1	0
6	0	0	1	0	1	0	0	0	1
7	0	0	0	1	0	0	0	1	0
8	0	0	0	0	1	0	1	0	1
9	0	0	0	0	0	1	0	1	0

(c) adjacent matrix of all units

Figure 4.5. A hypothetical example of a two-district case.
(a) Two district case, (b) sub-adjacency matrix consisting of units in District 2 and (c) adjacent matrix of all units; adjacent units to District 1 is unit 2, 3, 4, and 7 and non-movable units are unit 2 and unit 4. Movable units are unit 3 and unit 7. Sub-adjacent matrix of units in District 2 (b) is from adjacent matrix for all units (c).

When eligible candidate units are decided, the algorithm selects the number of population units to be specified among eligible candidate units from Algorithm P1. If the number of population units is specified as 3, then the algorithm randomly selects population units between 1, 2, and 3. The selected population units among eligible candidate units can be considered for the second procedure of ensuring contiguity.

4.1.3.2 Contiguity rechecking procedure

The second procedure is a contiguity checking procedure using Algorithm P2, which rechecks if the selected population units among eligible candidate adjacent units are really “movable” (Figure 4.6). The method of checking contiguity is performed by four steps:

(1) The first step randomly selects a seed unit amongst the selected population units from eligible candidate adjacent units (step 3),

(2) The second step finds adjacent units to the seed unit, and randomly choosing only one and one unit among them (denoted as “the fruit unit”) (step 4.and step 5),

(3) The third step adds adjacent units to the fruit unit except the seed unit until there is no unit to be appended (step 6 and step 7),

(4) The fourth step calculates the total number of units to be added except the seed unit (step 8). If the total number of units is equal to one less than the total number of units in the district (the total number of units in the district – 1), the seed unit can be “movable”. Otherwise the seed unit is “non-movable” which can be removed from eligible candidate adjacent units (step 9).

Algorithm P2 {*Contiguity rechecking procedure*}
Input: *n* (the number of the selected units), the set of selected units
Output: true or false
1. $i = 0$
2. **Repeat until** $i = n$
3. Randomly select a seed unit from selected units
4. Find adjacent units to the seed unit
5. Randomly select a fruit unit among adjacent units
6. **Repeat until** there is no unit to be selected except the seed unit
7. Add adjacent units to the fruit unit except the seed unit
8. Calculating the total number of units to be added except the seed unit
9. If the total number of units = the number of units in the district – 1,
   Then the seed unit is “movable unit” and is maintained.
   Else the seed unit is “non-movable unit” and is subtracted.
10. $i = i + 1$

Figure 4.6. Contiguity rechecking procedure

For example, there is a 5 by 5 data with 3 districts (Figure 4.7). Unit 13 is decided as “eligible candidate adjacent units” from the above first step of Algorithm P1. So, unit 13 can possibly be moved from District 2 to District 1 and possibly be selected for a new solution. Unit 13 can be judged throughout the second procedure of rechecking the contiguity using Algorithm P2. From the first procedure, eligible candidate adjacent units to District 1 are unit 11, 12, 13, 14, and 15 which are judged as eligible candidate adjacent units for possibly being moved. Amongst them, the algorithm checks if unit 13 are really “movable”. The algorithm (1) randomly selects unit 13 as a seed unit, (2) finds adjacent units (unit 12 and 14) to the seed unit, and randomly selects unit 12 as the fruit unit. Except the seed unit (unit 13), the algorithm (3) adds adjacent units (unit 11 and 17) to the fruit unit (unit 12), and continuously selects adjacent units (unit 16) until there is no unit to be selected. The total number of units to be added to the fruit unit except the seed unit is 4 (unit 12, 11, 17 and 16). The seed unit 13 is “non-movable” because the total number of units (4; unit 12, 11, 17 and 16) is not equal to one less than the total number of units in District 2 (the total number of units in District 2 – 1 which is 8 (unit 11, 12, 14, 15, 16, 17, 19 and 20). The checking procedure proceeds with another seed unit among the selected units from eligible candidate adjacent units to District 1.

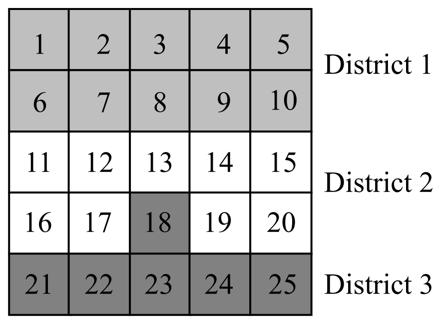
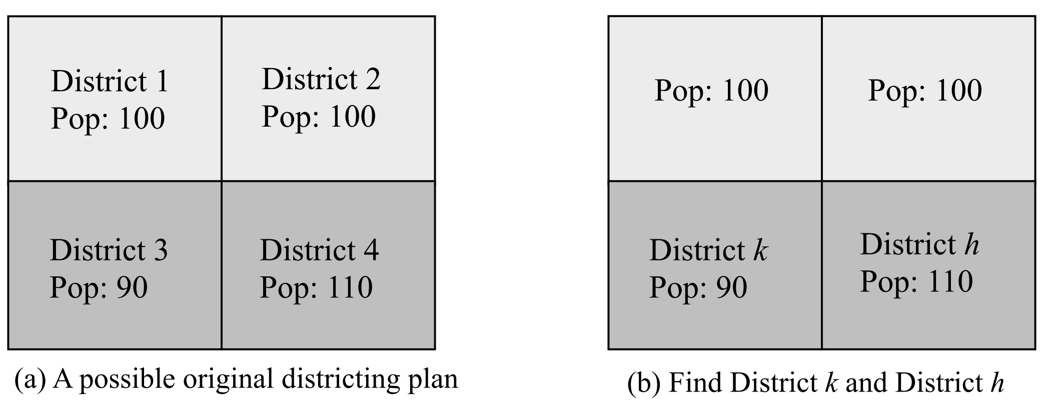


Figure 4.7. The exceptional case
Unit 13 is considered as an eligible candidate adjacent unit in the first procedure of ensuring contiguity in section 4.1.3.1 (Algorithm P1). However, it can be realized as a non-movable unit from rechecking contiguity procedure in section 4.1.3.2 (Algorithm P2).

4.2 Give-And-Take greedy algorithm (GAT algorithm)

This dissertation develops another heuristic to political redistricting problems called Give-And-Take greedy algorithm (GAT algorithm). Greedy algorithms are a local search algorithm to find better solutions throughout several iterations with the hope of reaching the global optimal (Cormen et al. 2001). Greedy algorithms do not guarantee to reach optimal solutions, but is quite powerful and works well for a wide range of problems. The idea of greedy algorithms is to keep only best solution at a time. From an initial solution, greedy algorithm explores neighborhoods and creates candidate solutions in neighborhoods, and then selects the best candidate solutions among candidate solutions. The algorithm determines if a candidate solution improves the objective function, thereby indicates the best solution.

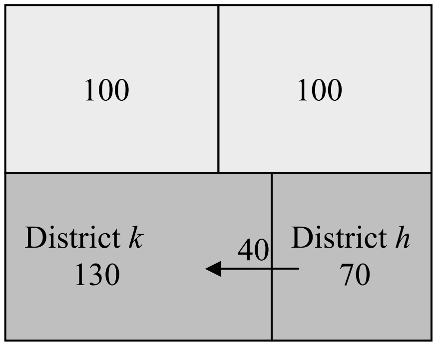
Figure 4.8 shows the central idea of the Give-And-Take greedy algorithm (GAT algorithm). The algorithm finds population units in a district with a larger population district than the ideal population and then population units in an adjacent district with a smaller population than the ideal population (Figure 4.8b). By giving population units and taking them between these districts, the algorithm tries to minimize the population difference (Figure 4.8c and Figure 4.8d).



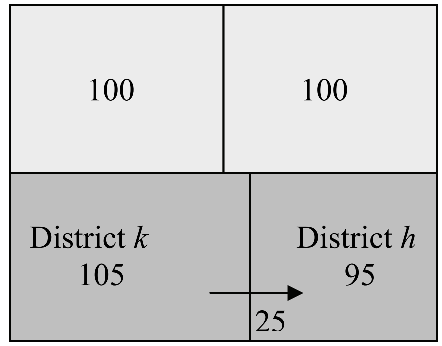
Continued

Figure 4.8. The main idea of Give-And-Take greedy algorithm

Figure 4.8 continued



(c) Give pop 40 from District *h* to District *k*



(d) Take pop 25 District *k* to District *h*

4.2.1 The general process of the heuristic

The Give-And-Take greedy algorithm is outlined in Figure 4.9 (Algorithm GAT). The initial solution is obtained from the same process of the multi-scale simulated annealing, which randomly selects *r* seed units, finds adjacent units from each seed unit, and adds adjacent units to the seed unit until there are no units to be added. For a new solution, the Give-And-Take greedy algorithm finds a solution using two algorithms: Algorithm APS and Algorithm ACA. Algorithm APS is to find all possible solutions by searching for diverse configurations which is evolved from the initial plan. Algorithm ACA is to check all candidate adjacent units by finding similar configurations to the initial plan, which possibly visits all of adjacent units to the initial plan. The Give-And-Take algorithm finds a solution using algorithm ACA and also find a solution using algorithm APS. By comparing two solutions, the Give-And-Take algorithm reports a better solution.

Algorithm APS and Algorithm ACA have broadly a same mechanism called Algorithm GAT-kernal in Figure 4.9. The algorithm first finds the district, which has a smaller population than the ideal population (District *k*). If the district exists, the algorithm finds candidate districts, which have larger population than the ideal population, and randomly selects one from the candidate adjacent districts (District *h*). The algorithm finds adjacent units in the selected larger population district, which have common border to the district with a smaller population the ideal population (Uh). Then the algorithm randomly selects the maximum number of population units to be swapped (Uh1), and gives them to the smaller population district *k*. The algorithm also finds adjacent units in the smaller population district which can be moved to the larger population district (Uk). It randomly keeps selecting a population unit and adding (or taking) it to the larger population district under the condition (Please see the next paragraph). In the above described way, the algorithm swaps population units between these. And then the algorithm checks if the swapped population units are movable. If it is true, the algorithm creates a new solution and calculates the objective function (return 0). Finally, the new solution, which is the better than or equal to the current solution, is always accepted, thereby the objective function value finally can be minimized. The algorithm can stop when the algorithm continually fails to exchange population units between these districts at specified number of times (nf = NF).

For the swapping (or giving and taking) condition, two differences (Difference 1 and Difference 2) used in the algorithm as below.

$$\text{Difference1} = \text{the population}\,(h) - \text{the population of district}\,(k)$$
$$\text{Difference2} = \text{the total population in Uh1} - \text{the total population inUk1}$$

where *h* is the district which has a larger population than the ideal population in the original plan, *k* is the district which has a smaller population than the ideal population in the original plan, Uh1 is the maximum number of population units moved from district *h* to district *k* and Uk1 is the number of reassigned population units from district *k* and district *h*. Difference 1 is the difference between the population in the district which has larger population than ideal population in the original plan and the population in the district which has a smaller population district than ideal population in the original plan. Difference 2 is the difference of really moved population units between two districts. The population units can be swapped while Difference 1 $\geq$ Difference 2. Among them, population units can also be swapped while Difference 2 ≥ 0. Under the evaluation condition, the algorithm can minimize population deviation between districts. For example, in Figure 4.8a and Figure 4.9b, Difference 1 is 20, which is the population in District *h* in the original plan – the population in District *k* in the original plan (110 – 90 = 20) (Figure 4.8b). Difference 2 is 15, which is the real number of swapped units (40 – 25 = 15) (Figure 4.8c and Figure 4.8d).

Note: each district is represented as an array and there are a total of *r* arrays for a specific problem.

Notation

A solution is denoted as s, which contains *r* arrays referred to as s[0], s[1], s[2], ..., s[*r*]. Each of the *r* arrays in s contains a set of unique identification numbers of the units that are assigned to that array. For example, if s[0] includes elements {0, 1, 13, 60}, it means that spatial units 0, 1, 13, and 60 are assigned to district 0.

Algorithm GAT {Give-And-Take greedy algorithm to political redistricting}
Input: Adjacent matrix, population of each population unit
Output: *y*

Generating a random initial solution
Algorithm APS
Algorithm ACA
Evaluation APS and ACA

Generating a random initial solution

```
s := ø
U := a set of all unassigned units
Randomly select r seed units from U
Add each of the seed units to one of the r arrays in s
Remove the r seeds from U
Repeat until |U| = 0
        For each district i
                u := random unassigned unit that is adjacent to the district
                add u to district s[i]
                remove u from U
Return s
```

Algorithm APS

```
Input: s, NF (number of failed trials)
s0 := s
nf := 0
Repeat until nf = NF
        For each district k in s0
                p := GAT-kernel(k, s0)
                if (p > 0)
                        nf := nf + p
                if (p = 0)
                        s := s0
return s
```

Continued

Figure 4.9. The process of Give-And-Take greedy algorithm

Figure 4.9 continued

Algorithm ACA

```
Input: s, NF (number of failed trials)
s0 := s
nf := 0
Repeat until nf = NF
        s0 := s
        For each district k in s0
                p := GAT-kernel(k, s0)
                if (p > 0)
                        nf := nf + p
                if (p = 0)
                        report s0
```

Algorithm GAT-kernel(k, s)

```
If the population of district k in s is greater than the ideal population, Then return -1
h := a random district with a population greater than the ideal population
d1 := population(h) - population(k)
Uh := a set of units in h and are adjacent to district k
Uh1 := nn units randomly selected from Uh (nn = rand_range(1, NN)
Add units in Uh1 to district k
Uk := a set of border units in district k adjacent to h
Uk1 := empty
For each unit j in Uk
        D2 := population(Uh) – population(j)
        if (Difference 1 >= D2)
                assign j to district h
                add j to Uk1
        Difference 2 := total population in Uh1 – total population of units in Uk1
        if (Difference 2 >= 0)
                if districts k and h are contiguous using CONTIGUITY_P2
                        return 0
                else
                        return 1
        else
                return 1
```

4.2.2 Ensuring contiguity in the GAT algorithm

The GAT algorithm uses contiguity rechecking procedure in a more simple way in above section 4.1.3.2 because the only usage of contiguity rechecking procedure not only keeps sufficiently checking contiguity among units, but also saves computing time. After exchanging population units between districts, the algorithm checks if exchanged units are really movable for the choice of next solution with three procedures (Figure 4.10).

(1) The first procedure selects a seed unit among the exchanged units (step 1),

(2) The second procedure adds adjacent units to the seed unit until there is no unit to be appended (step 2 and step 3.1),

(3) The third procedure calculates the total number of units to be added to the seed unit (step 3.2). If the total number of units is equal to the total number of units in each district, the seed unit can be movable. Otherwise the seed unit is non-movable (step 4).

Algorithm CONTIGUITY_P2 {*Contiguity rechecking procedure of the GAT algorithm*}
Input: *n* (the number of the exchanged units), the set of selected units
Output: true or false
1. Select a seed unit from exchanged units
2. Find adjacent units to the seed unit
3. **Repeat until** there is no unit to be selected
3.1. Add adjacent units to the seed unit
3.2. Calculate the total number of units to be added
4. If the total number of units = the total number of units in the district,
Then the seed unit is "movable unit" and is maintained
Else the seed unit is "non-movable unit" and is subtracted

Figure 4.10. Contiguity procedure of the Give-And-Take algorithm

4.3 Experiments on the Give-And-Take greedy algorithm

There are two kinds of experiments for the Give-And-Take greedy algorithm. The first experiment extends the algorithm in order to find various spatial configurations with similar (or better) population of the given plan. The second experiment is a sensitivity analysis to see how the algorithm performs under different parameters.

4.3.1 Finding various spatial configurations similar to the given plan

For the given redistricting plan, is it possible to find different spatial shapes with similar population to the given plan? The question can be solved using Give-And-Take greedy algorithm, which starts from the given plan as the initial solution. The algorithm also gives and takes population units between a district with the larger population with the ideal population and a district with the smaller population with the ideal population. The algorithm would allow population units to be switched between districts if doing so does not increase population deviation.

4.3.2 Sensitivity analysis of the heuristic

There are two parameters to be explored for sensitivity analysis of the Give-And-Take greedy algorithm: the maximum number of units to be swapped (step 2.1.4.5 in Figure

4.9), and the specified number of failures (step 2.1.2 and step 2.2.1 in Figure 4.9). The maximum number of units to be swapped units is the number of units that are randomly selected for adding to the smaller population district. For example, when the maximum number of units to be swapped is set to 3, the algorithm randomly decides 1, 2 or 3 units to be selected for swapping. The specified number of failures is about the terminal condition of the algorithm and the total number of degraded trials where the population units between the larger population district and the smaller population district continually fail to be exchanged. The algorithm exchanges the population units between the smaller population district and the larger population district until the algorithm continually fails to exchange them specified number of times, for example, 100.

4.4 Computational experiments of the heuristic models

Heuristic tests for political redistricting problems are performed on a computer system of Intel(R) Core™ 2 CPU E7400, 2.80GHz, 4 GB RAM. All algorithms utilize the same objective function and test data set which can be comparable. The objective function is to minimize the population deviation of each district from the ideal population. So, we have a following objective function.

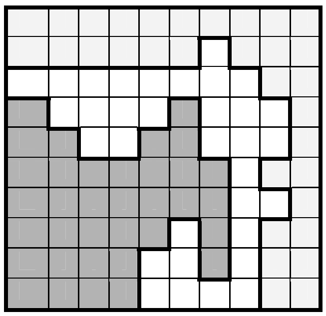
$$\text{Min } 100 \times \frac{1}{P} \times \sum_{j=1}^{r} \left| p_j - P^* \right|$$

Where P is the total population, r is the number of districts, P^* is the ideal population computed as the rounded integer value of P/r, and p_j is the population of the j district.

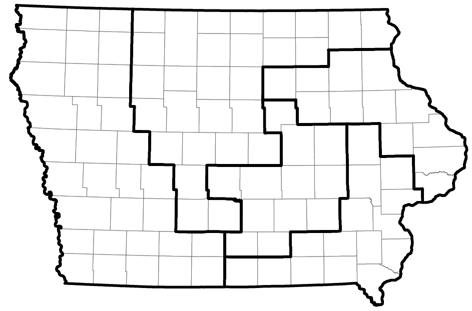
The same types of data sets used in the previous Chapter 3 here are used here for performance test and comparisons of two heuristics: regular data (5 by 5, 10 by 10 and 25 by 40 type of data), and the Iowa data. For the experiments on the Give-And-Take greedy algorithm, above the 10 by 10 data and Iowa congressional redistrict data are used. (Figure 4.11*a* and Figure 4.11*b*).

The dissertation also applied the Give-And-Take greedy algorithm on another data set, which is the large size of 8374 VTDs (Voting Tabulation District) data for Texas. VTD refers to election units that State and local governments create for elections. VTD is a term adopted by the Bureau of the Census in order to relate election data to census data. VTDs and their demographic data are obtained from the American Fact Finder in Census Bureau[13]. The given plan of Texas is 107th Congressional Districts (Jan. 2001 - Jan. 2003), which shows a good match to the shape file of VTDs data. 107th congressional district has 30 districts with an objective function value of 9.341 (Figure 4.11*c*).

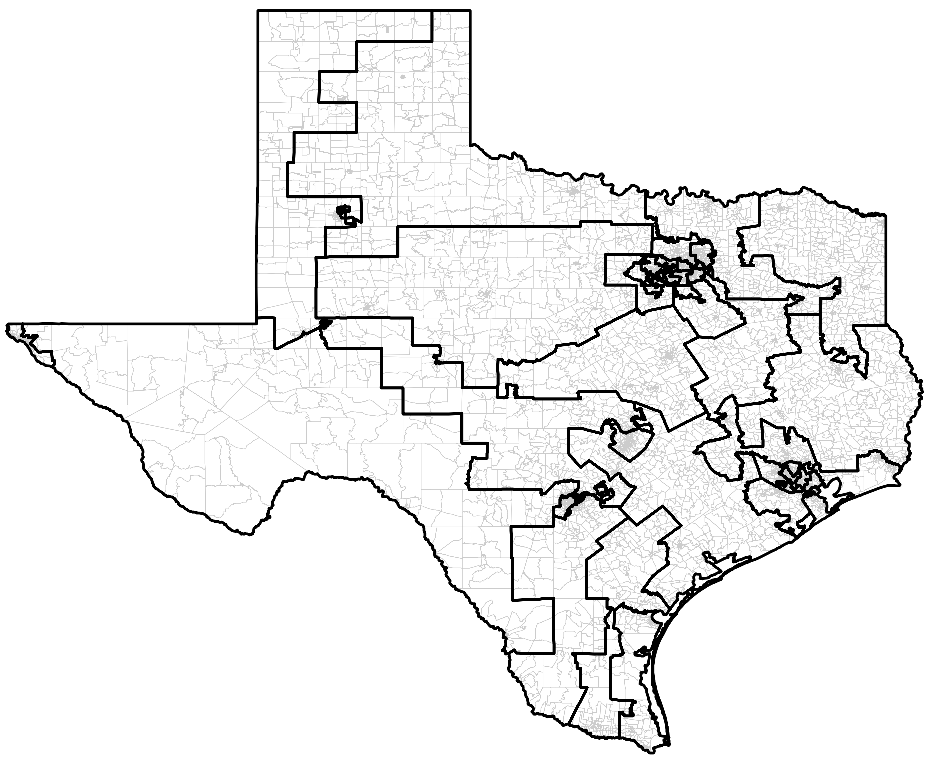
[13] http://factfinder2.census.gov



(a) “Official” plan of 10 by 10 data: 1.29
(39 persons)



(b) Iowa official plan: 0.0080
(235 persons)



(c) Official plan of Texas 107th congressional district: 9.341
(1947759 persons)

Figure 4.11. Test data with the given plan

4.4.1 Test results of heuristic models

The multi-scale simulated annealing heuristic and the Give-And-Take greedy heuristic find a solution at every iteration and are executed 500 times. So, the total number of solutions generated is as same as 500 for both heuristics. Parameter settings for the best solution founded by both algorithms are listed in Table 4.2 and Table 4.3, respectively. For the Give-And-Take greedy algorithm, the maximum number of units to be swapped is not the real number to be swapped. When the maximum number of units to be swapped is decided, for example 4, the heuristic randomly decides 1, 2, 3 and 4 units to be selected for swapping, for example 3.

n[a]	*r*[b]	Ini_Temp.[c]	Alpa-rate[d]	Beta-rate[e]	MSP[f]	Stop_Temp.[g]
5 by 5	2	15.81	0.5	0.95	2.91	0.01
	3	26.05	0.5	0.965	6.49	0.001
	4	31.56	0.5	0.96	3.34	0.00001
10 by 10	3	1.94	0.5	0.965	0.93	0.00001
	4	5.64	0.5	0.96	2.10	0.000001
	5	8.08	0.5	0.96	1.97	0.0001
25 by 40	5	5.99	0.5	0.96	6.76E-13	1.0E-22
	10	32.45	0.5	0.98	1.70E-64	1.1E-65
	20	16.63	0.5	0.98	3.46E-49	1.8E-50
Iowa	5	55.31	0.5	0.95	16.15	1.0E-22

a: data type
b: the number of districts
c: Initial temperature
d: Alpa for multiple decreasing rates
e: Beta for multiple decreasing rates
f: the temperature at the maximum specific heat occurs
g: Stop temperature

Table 4.2. Parameter settings of the multi-scale simulated annealing

n^a	r^b	The certain number of units to be swapped	The specified number of failures
5 by 5	2	5	100
	3	8	10000
	4	2	1000
10 by 10	3	2	1000
	4	8	1000
	5	3	10000
25 by 40	5	5	1000
	10	7	1000
	20	8	1000
Iowa	5	5	10000

Table 4.3. Parameter settings of the Give-And-Take greedy algorithm

Table 4.4 includes computational results of two heuristics. For comparison, the computational results of Xiao (2008) are utilized. The first column is data set from 5 by 5, 10 by 10, and 25 by 40 data to Iowa data. The second column is the number of districts, and the third column is theoretical optimal solution. The global optimal solution is not known in heuristic models, but for many cases of test data (5 by 5 data, 10 by 10 data) the global optimal solution can be used from the test results of exact models in Chapter 3 (Table 3.2). For other cases not known from exact models, theoretical optimal solution is calculated with the assumption that all population units are connected to any other population units. If the total population can be evenly divided by the number of districts, the theoretical optimal solution has 0 as the objective function value. The minimum, median, maximum objective function values and CPU time for each heuristic including the result of Xiao (2008) are reported.

n^a	r^b	Theoretical Optimal Solution[c]	Multi-scale simulated annealing				Give-And-Take algorithm				Xiao (2008)			
			Min	Median	Max	Time[d] (sec.)	Min	Median	Max	Time[d] (sec.)	Min	Median	Max	Time (sec.)
5by5	2	0.00	0.00	0.00	2.60	0.06	0.00	0.00	16.15	0.01	0.00	0.00	0.00	7.00
	3	0.00	0.00	0.52	3.65	0.17	0.00	2.08	23.44	0.17	0.00	0.00	1.56	6.00
	4	0.00	0.00	1.56	6.25	0.31	0.00	10.94	33.85	0.03	0.00	0.52	1.56	6.00
10by10	3	0.00	0.00	0.14	0.54	1.22	0.00	0.00	39.16	0.16	0.00	0.00	0.00	73.00
	4	0.00	0.00	0.34	1.02	1.72	0.00	0.47	31.10	0.16	0.00	0.00	0.07	66.00
	5	0.07	0.07	0.75	1.76	1.53	0.07	3.18	29.40	0.47	0.07	0.11	0.14	64.00
25by40	5	0.0027	0.004	0.076	19.326	161.50	0.004	0.007	13.122	2.312	0.0027	0.0043	0.0113	19496
	10	0.004	0.111	4.159	22.31	321.56	0.004	4.397	27.163	26.417	0.0174	0.0308	0.004	18901
	20	0.004	0.674	8.313	25.669	223.59	0.090	11.378	29.813	22.986	0.0710	0.0978	0.2404	19832
Iowa*	5	0.0003	0.0066 (192 persons)				0.00075 (22 persons)				0.0045 (131 persons)			

a: Data type
b: The number of districts
c: Theoretical optimal solution $100\times\frac{1}{P}\times|P-r\times P^*|$
d: average time in seconds after 10 times trials with the computer system of Intel(R) Core™ 2 CPU E7400, 2.80GHz, 4 GB RAM
*: The official plan of Iowa in 2000: 0.0080

Table 4.4. Results of all heuristic models

From comparisons between two new heuristics, the Give-And-Take greedy algorithm is more efficient than the multi-scale simulated annealing heuristic, and even the large data such as 25 by 40 data show such efficiency in CPU time. This is because the algorithm uses a simple contiguity checking process to evaluate if the swapped population units are really “movable” and accepts only a better solution unlike a multi-scale simulated annealing to often accept worse solutions. The Give-And-Take greedy algorithm can reach theoretical optimal solution for small data set such as 5 by 5 and 10 by 10 data as well as for large data set such as 25 by 40 data ($r = 4$). For other data set, the Give-And-Take algorithm reaches near-optimal solutions. However, in terms of the ranges of solutions such as minimum, median and maximum objective values, multi-scale simulated annealing shows higher quality solutions because the results of Give-And-Take greedy algorithm absolutely depends on random initial solutions. Multi-scale simulated annealing may not reach theoretical optimal solutions for 25 by 40 data but reach near-optimal solutions.

From the comparison with Xiao (2008), two new heuristics (the Give-And-Take algorithm and the multi-scale simulated annealing) are more efficient than the existing heuristic of Xiao (2008). The Give-And-Take greedy algorithm shows excellent efficiency to find solutions while Xiao (2008) and the multi-scale simulated annealing show higher effectiveness. For the large size data such as 25 by 40 data, all of the heuristic can reach near-optimal solutions. Especially, the Give-And-Take greedy algorithm and the evolutionary algorithm of Xiao (2008) arrive at theoretical optimal solutions in one case, $r = 10$ and $r = 5$ respectively. The maps of the best solution founded in Figure 4.12 to Figure 4.14 when they are represented with a strict population equality and contiguity. New heuristics finds various kinds of optimal solutions in the same objective function value for each problem. Furthermore, two new heuristics find different redistricting plans in the same strict equal population.

Each algorithm finds different spatial configurations in the strict equality redistricting plan (Table 4.5). For 5 by 5 data, the multi-scale simulated annealing finds 100 different kinds of results for $r = 2$, 18 different kinds of spatial configuration for $r = 3$, and 1 solution for $r = 4$. The Give-And-Take greedy algorithm finds 254 various spatial shapes for $r = 2$, 61 various kinds of shapes for $r = 3$ and 41 shapes for $r = 4$. For 10 by 10 data, the multi-scale simulated annealing finds 51 distinctive spatial configurations for $r = 3$, 2 different kinds of spatial configuration for $r = 4$, and 2 solutions for $r = 5$. The Give-And-Take greedy algorithm finds 254 various spatial shapes for $r = 3$, 61 various kinds of shapes for $r = 4$ and 41 shapes for $r = 5$. For 25 by 40 data, the Give-And-Take greedy algorithm finds 2 different shapes for $r = 10$. The Give-And-Take greedy algorithm shows advantages finding different spatial configurations in the same population when the data sizes are larger because the algorithm not only visits all adjacent units to the initial solution (Algorithm ACA) but also checks all units as possible evolved from the initial solution (Algorithm APS).

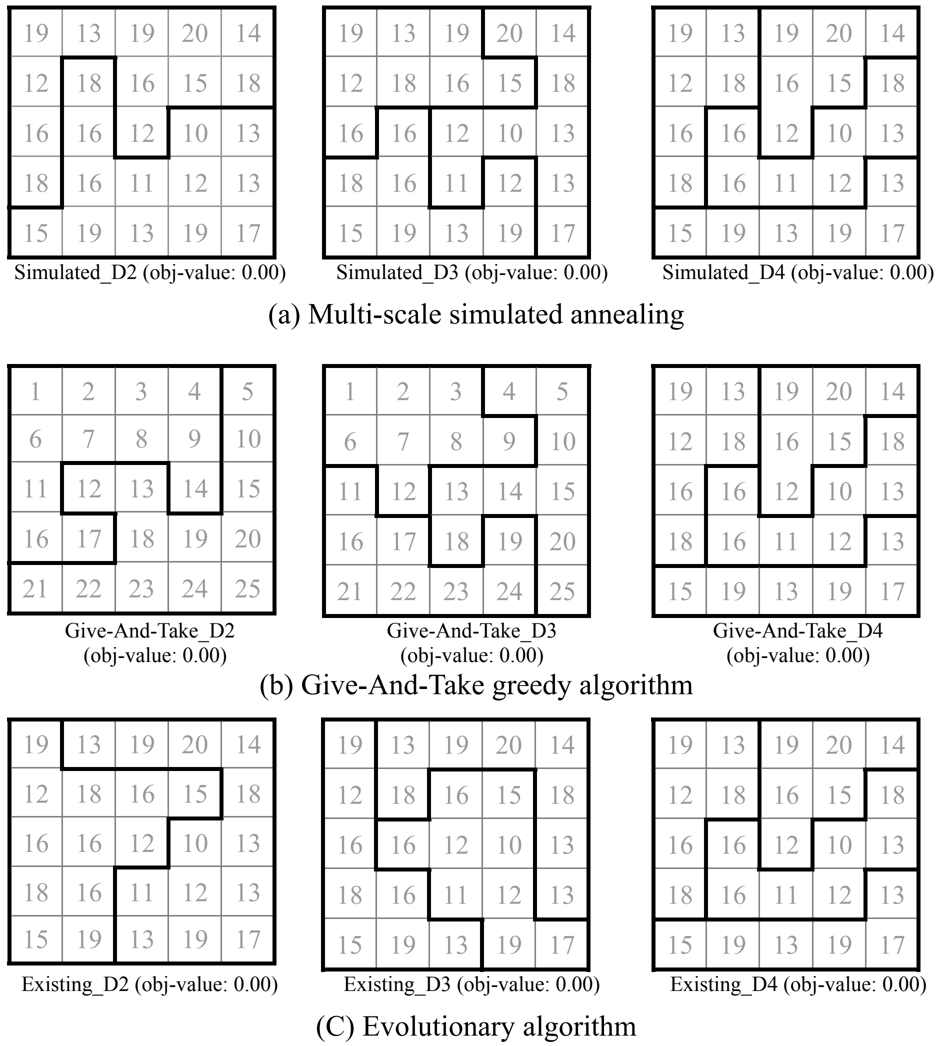


Figure 4.12. Redistricting results of 25 units in 5 by 5 data with 2, 3 and 4 districts for each heuristic.

Thick lines represent the district boundaries. Multi-scale simulated annealing finds 100 different kinds of results for r = 2, 18 different kinds of spatial configuration for r = 3, and 1 solution for r = 4. Give-And-Take greedy algorithm finds 254 various spatial shapes for r = 2, 61 various kinds of shapes for r = 3 and 41 shapes for r = 4.

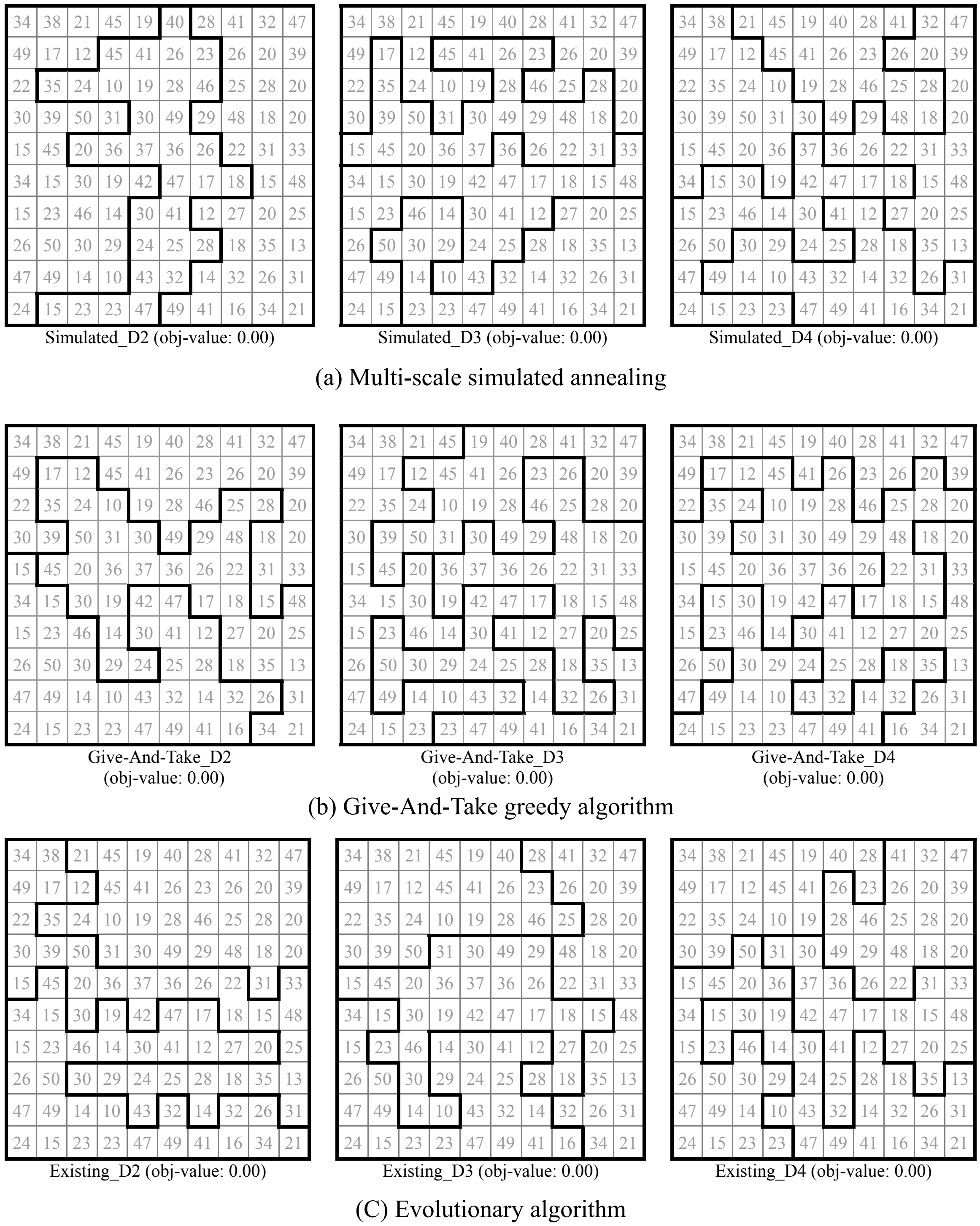


Figure 4.13. Redistricting results of 100 units in 10 by 10 data with 3, 4, and 5 districts.

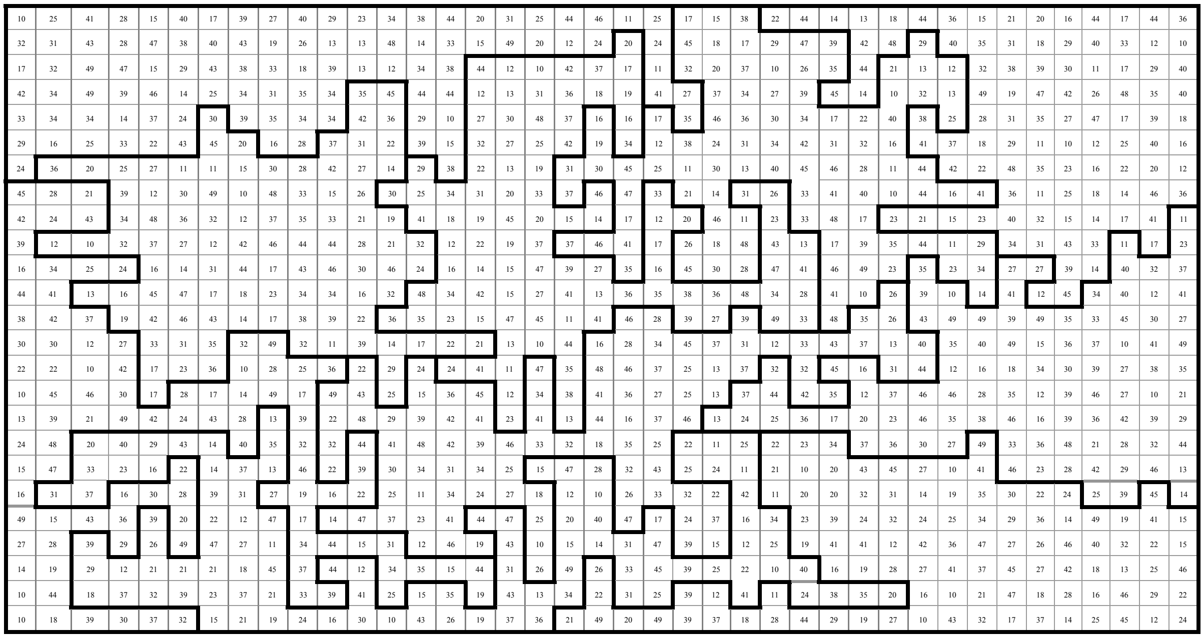
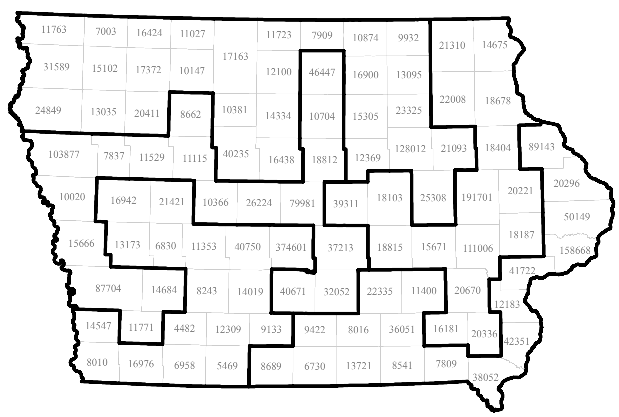


Figure 4.14. Redistricting results of 1000 units in 25 by 40 data with r = 10 for Give-And-Take greedy algorithm

For the Iowa redistricting example, the total population of year 2000 is 2,926,324, and the theoretical global optimal solution is 0.0003. The official plan adopted by Iowa in 2000 has an objective function value of 0.0080, and the total absolute deviation from the ideal population is 235 persons (Figure 4.11*b*). The best solution by Give-And-Take greedy algorithm shows higher population equality than that by the multi-scale simulated annealing heuristic. The multi-scale simulated annealing heuristic has the best solution with an objective function value of 0.0066, and 192 persons as the total absolute deviation from the ideal population (Figure 4.15*a*), the Give-And-Take greedy algorithm has as an objective value of 0.00075, which is 22 persons as the total absolute deviation from the ideal population (Figure 4.15*b*). Guo and Jin (2011) report 33 persons as the total absolute deviation from the ideal population[14]. The best plan by the Give-And-Take algorithm using Iowa 2010 census data is also tested. The total population of year 2010 is 3,046,355 and the Give-And-Take greedy algorithm has an objective value of 0.00023, which is 7 persons as the total absolute deviation from the ideal population (Figure 4.16). It can be concluded that the all of the heuristics find redistricting plans with higher population equality than the official plan. Among them the Give-And-Take greedy algorithm has the most strict population equality redistricting plan.



(a) The best solution by the multi-scale simulated annealing: 0.0066 (192 persons)

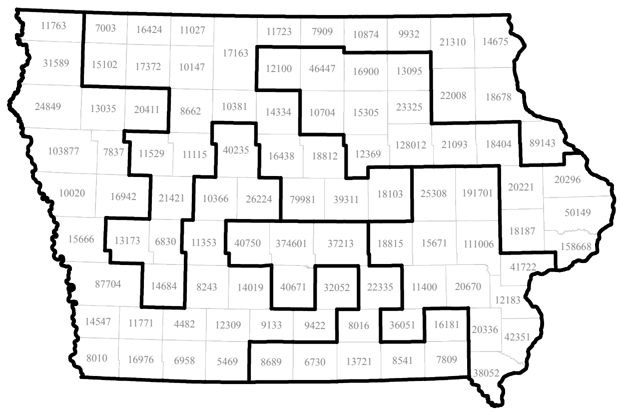
continued

Figure 4.15. Redistricting for Iowa where 99 counties with 5 congressional districts (2000)

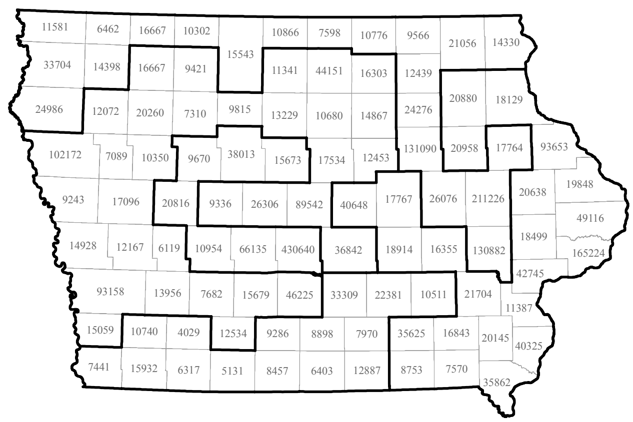
---

[14] http://www.spatialdatamining.org/iRedistrict

Figure 4.15. continued



(b) The best solution by the Give-And-Take algorithm: 0.00075 (22 persons)



The best solution by the Give-And-Take algorithm: 0.00023 (7 persons)

Figure 4.16. Redistricting for Iowa where 99 counties with 4 congressional districts (2010)

n^a	r^b	Multi-scale simulated annealing	Give-And-Take algorithm
5 by 5	2	100	75
	3	18	8
	4	1	1
10 by 10	3	51	254
	4	2	61
	5	2	41
25 by 40	5	-	-
	10	-	2
	20	-	-

a: Data type
b: The number of districts

Table 4.5. The number of different shapes in the same objective function values representing the strict population equality

4.4.2 Results from the experiments of the Give-And-Take greedy algorithm

According to the above tests, the Give-And-Take greedy algorithm can be applied to find different spatial configurations but similar or better population of the given plan because the algorithm can not only reach high quality solutions in efficient amount of time even for the large data, but also find lots of different spatial configurations. Algorithm APS is to find all possible solutions by searching for diverse configurations which is evolved from the initial plan. The algorithm finds different kinds of spatial configurations starting from the given plan (Figure 4.9 and section 4.3.1). Parameters for the application of the algorithm are decided in Table 4.6.

Data type	n^a	r^b	G^c	S^d	The specified number of failure
10by10	100	3	1.29	3	100
Iowa	99	5	0.0080	3	100
Texas	8374	30	9.34	3	100

a: The number of population units
c: Given initial solutions
b: The number of districts
d: The certain number of units to be swapped

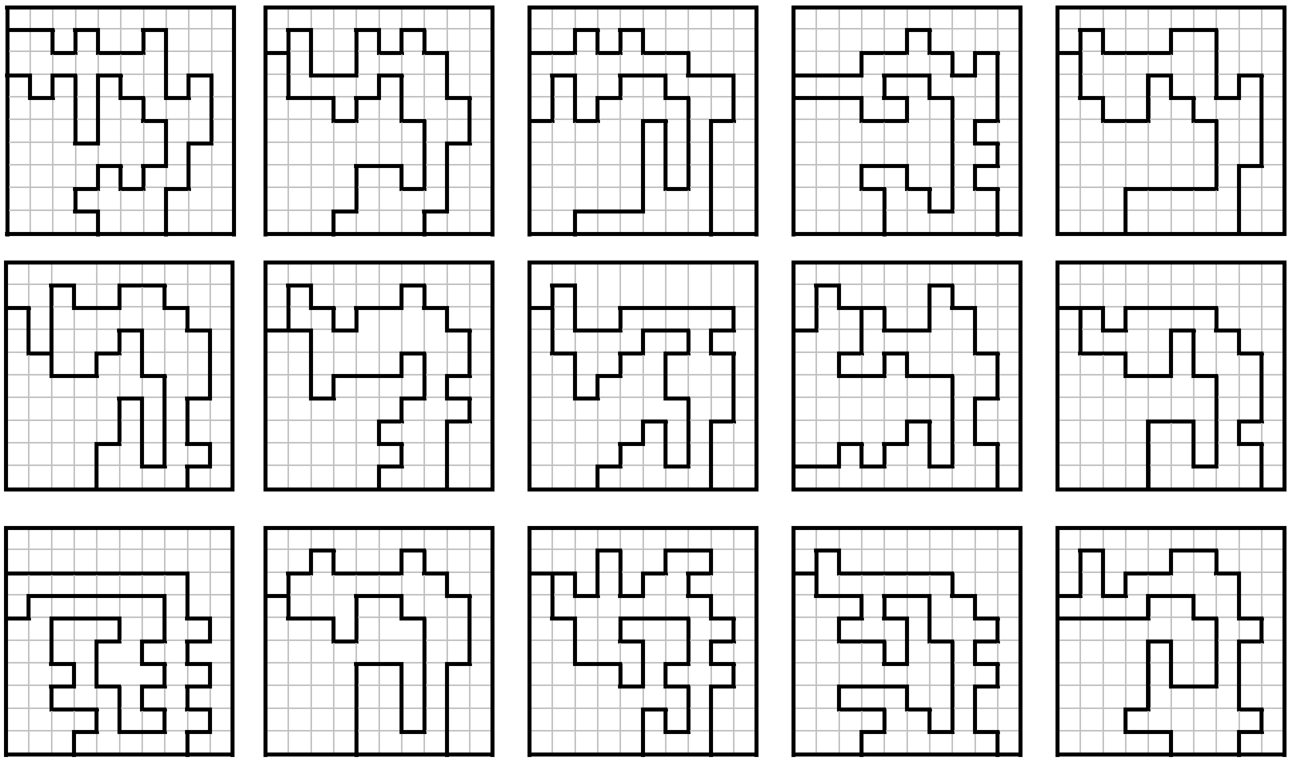
Table 4.6. Parameter setting of the Give-And-Take greedy algorithm for finding various shapes

n	*r*	G	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
10by10	3	1.29	1.016	1.016	1.220	0.005	0.000	0.271	1.220	0.027
Iowa	5	0.008	0.0053	0.0059	0.0073	0.009	0.0053	0.0059	0.0073	0.004
Texas	30	9.341	9.014	9.256	9.340	405.301	0.433	2.430	9.339	1280.92

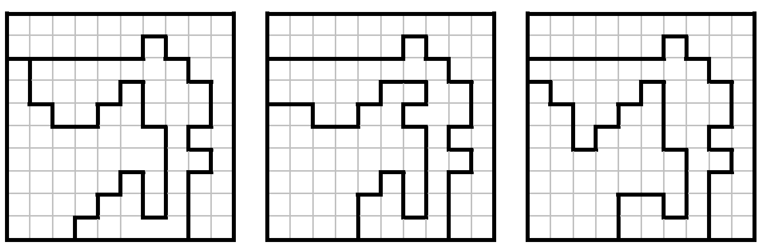
Table 4.7. Computational results of finding different redistricting plan of the Give-And-Take greedy algorithm

Table 4.7 shows computational results of finding various redistricting plans similar to the given plan. Min, Median and Max are the objective function values starting from the given plan. Solutions are divided into all possible solution units check using algorithm APS and all candidate adjacent unit checks using algorithm ACA (Please see Figure 4.9). Computational experiments show that CPU Times are very short, even the large data such as Texas with 8374 VTDs show such efficiency in CPU time because the algorithm only uses a simple contiguity procedure. Figure 4.17 to Figure 4.19 show different spatial results of 10 by 10 data, Iowa and Texas. Computational results show that the Give-And-

Take greedy algorithm find lots of different redistricting plans but similar (better) or same population of the given plan because the algorithm not only visits all of adjacent units to the given plan, but also searches for diverse spatial configurations evolved from the given plan.

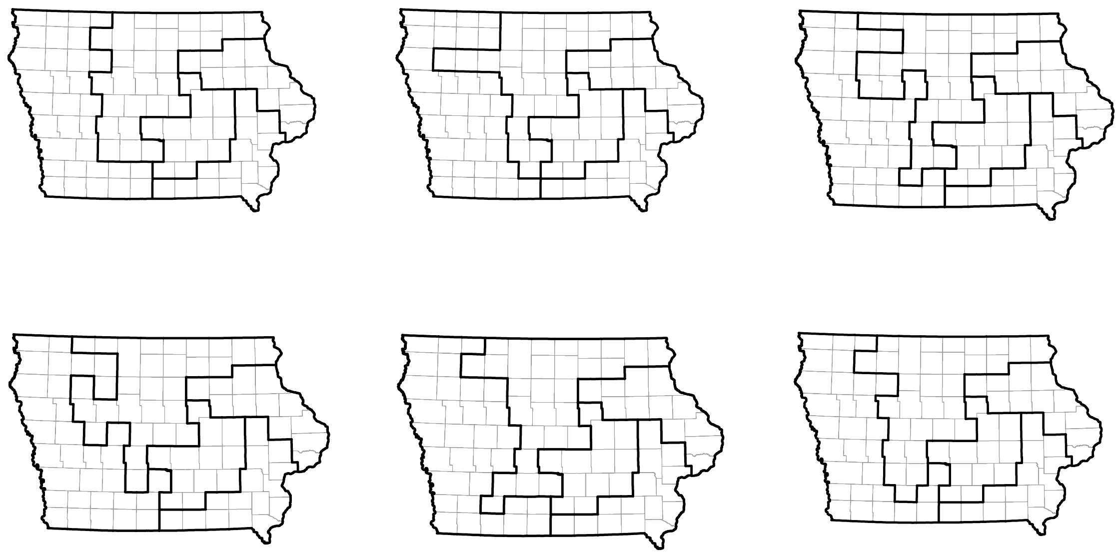


(a) The minimum objective function value of 0

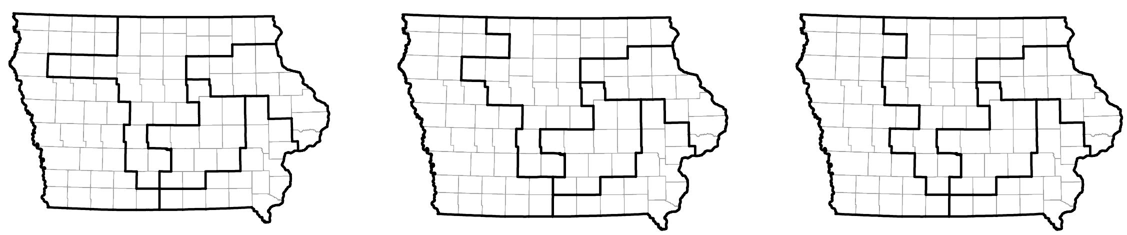


(b) The maximum objective function value of 1.219

Figure 4.17. Various spatial configurations results of 10 by 10 data

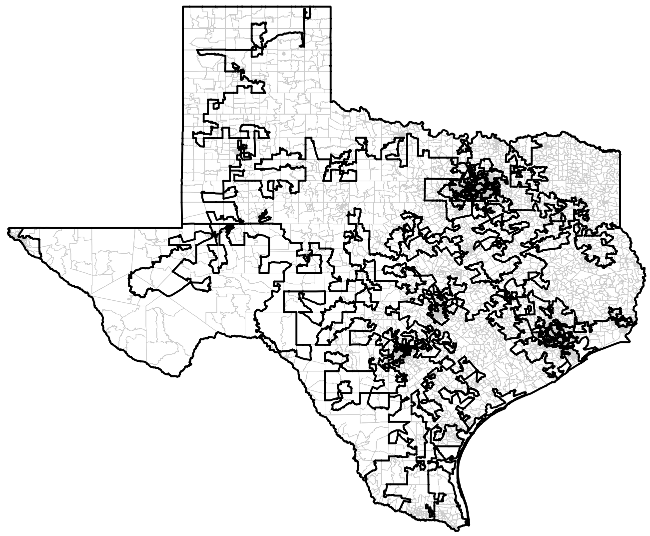


(a) the objective function value: 0.0053

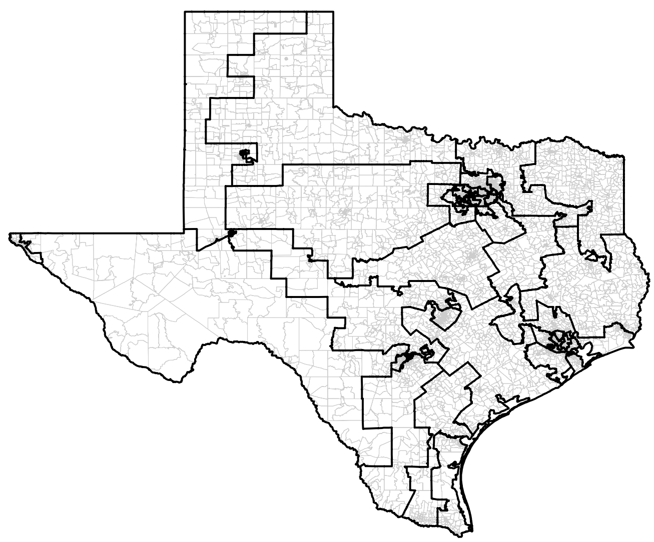


(b) obj-value: 0.0054 (c) obj-value: 0.0059 (d) obj-value: 0.0062

Figure 4.18. Various spatial configurations in Iowa



(a) Minimum objective function value: 0.60



(b) Maximum objective function value: 9.31

Figure 4.19. Various spatial configurations in Texas

4.4.3 Test results for sensitivity analysis of the Give-And-Take greedy algorithm

Parameter settings for sensitivity analysis are listed in the second column of Table 4.8 and Table 4.9. The maximum number of units to be swapped is set from 1 to 10 with an increment of 1. The number of failures for terminating the program is set from 10, 50…to 10000. The algorithm is executed 100 times for 10 by 10 data and Iowa data, and is executed 10 times for Texas data due to the relatively long CPU Time. There are two types of experiments. The first experiment is to explore the effect of the certain number of units to be swapped under the fixed number of failures (100 for all data). The second experiment is to observe results when the algorithm changes the number of failures within the fixed number of units to be swapped (3 for all data).

Table 4.8 and Table 4.9 include computational results of the algorithm. Min, Median and Max are the objective function values from the algorithm starting from the given plan. Solutions are divided into all possible solution units check using algorithm APS and all candidate adjacent unit checks using algorithm ACA (Please see Figure 4.9). Algorithm APS finding all possible solutions finds redistricting plans with higher population equality because the algorithm searches for diverse spatial configurations evolved from the given plan. Computational results show that different parameters have not a strong influence on reaching good solutions for 10 by 10 data and Iowa data. The Give-And-Take greedy algorithm often fails to find solutions in some cases of Iowa data because the given plan of Iowa shows higher population equality. Texas data have shown that the solution qualities become worse when the number of failures is so small (e.g. 10) or when the number of units to be swapped is so small (e.g.1). Texas data also have that the solution qualities, especially minimum solution qualities, are better in case of the higher number of trials (≥ 500). Several computational experiments can generally be concluded that parameters in the algorithm are not especially sensitive and CPU time gets long in the large number of failures.

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
1.29	1	100	1.016	1.016	1.220	0.013	0.000	0.407	1.220	0.018
	2	100	1.016	1.016	1.220	0.019	0.000	0.339	1.220	0.026
	3	100	1.016	1.016	1.220	0.019	0.000	0.407	1.220	0.021
	4	100	1.016	1.016	1.220	0.018	0.000	0.271	1.220	0.017
	5	100	1.016	1.016	1.220	0.019	0.000	0.237	1.220	0.016
	6	100	1.016	1.016	1.220	0.019	0.000	0.474	1.220	0.018
	7	100	1.016	1.016	1.220	0.017	0.000	0.203	1.220	0.012
	8	100	1.016	1.016	1.220	0.016	0.000	0.474	1.220	0.011
	9	100	1.016	1.016	1.220	0.015	0.000	0.813	1.220	0.007
	10	100	1.016	1.016	1.220	0.015	0.000	0.474	1.152	0.010

(a)10 by 10 data

continued

Table 4.8. Computational results depending on the different numbers of units to be swapped

Table 4.8 continued

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
0.008	*1*	*100*								
	2	100	0.006	0.006	0.006	0.003	0.005	0.006	0.006	0.051
	3	100	0.005	0.006	0.006	0.002	0.005	0.006	0.006	0.029
	4	100	0.006	0.006	0.006	0.003	0.005	0.005	0.006	0.024
	5	100	0.006	0.006	0.006	0.002	0.005	0.006	0.006	0.018
	6	100	0.005	0.006	0.006	0.003	0.005	0.005	0.006	0.010
	7	100	0.005	0.006	0.006	0.002	0.005	0.006	0.006	0.006
	8	*100*								
	9	*100*					0.005	0.006	0.006	0.016
	10	*100*					0.005	0.006	0.006	0.007

(b) Iowa data; gray columns fail to find the objective function values and solutions

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
9.308	1	100	9.001	9.240	9.308	49.460	1.122	3.279	9.308	1962.660
	2	100	8.854	9.225	9.308	51.515	0.450	2.320	9.306	1332.920
	3	100	8.866	9.206	9.308	49.874	0.449	2.118	9.283	1164.160
	4	100	8.697	9.176	9.307	45.960	0.148	1.693	9.308	1405.290
	5	100	8.806	9.161	9.305	44.236	0.130	1.690	9.308	1262.330
	6	100	8.789	9.135	9.306	44.416	0.449	2.000	9.288	1308.020
	7	100	8.633	9.182	9.308	43.043	0.131	2.032	9.306	1375.380
	8	100	8.760	9.193	9.303	42.375	0.130	1.934	9.301	1391.050
	9	100	8.555	9.196	9.307	42.909	0.483	2.332	9.308	1012.420
	10	100	8.776	9.184	9.307	42.901	0.467	2.526	9.308	1083.470

(c) Texas data

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
1.29	3	10	1.016	1.016	1.220	0.021	0.000	0.610	1.220	0.009
	3	50	1.016	1.016	1.220	0.018	0.000	0.474	1.220	0.019
	3	100	1.016	1.016	1.220	0.019	0.000	0.407	1.220	0.021
	3	200	1.016	1.016	1.220	0.018	0.000	0.339	1.220	0.032
	3	500	1.016	1.016	1.220	0.015	0.000	0.339	1.287	0.017
	3	1000	1.016	1.016	1.220	0.014	0.000	0.271	1.220	0.045
	3	5000	1.016	1.016	1.287	0.017	0.000	0.203	1.287	0.072
	3	10000	1.016	1.016	1.287	0.016	0.000	0.339	1.220	0.246

(a) 10by10 data

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
0.008	3	10	0.006	0.006	0.006	0.187	0.005	0.005	0.006	0.221
	3	50	0.005	0.006	0.006	0.207	0.006	0.006	0.006	0.194
	3	100	0.006	0.006	0.006	0.172	0.005	0.006	0.006	0.432
	3	500	0.006	0.006	0.006	0.177	0.005	0.005	0.006	1.445
	3	1000	0.006	0.006	0.006	0.174	0.005	0.005	0.006	2.800
	3	5000	0.005	0.006	0.006	0.197	0.005	0.005	0.006	11.581
	3	10000	0.006	0.006	0.006	0.184	0.005	0.006	0.006	15.040

(b) Iowa data

Given initial Solutions	The certain number of units	The number of failure	Algorithm ACA {*All candidate adjacent units check*}				Algorithm APS {*All possible solution units check*}			
			Min	Median	Max	CPU Time	Min	Median	Max	CPU Time
9.308	10	3	8.923	9.194	9.308	49.121	4.351	7.584	9.304	106.725
	50	3	8.895	9.194	9.307	51.641	0.449	2.420	9.303	887.861
	100	3	8.866	9.206	9.308	49.874	0.449	2.118	9.283	1164.160
	500	3	8.783	9.193	9.306	50.972	0.130	1.818	9.308	1728.880
	1000	3	8.936	9.191	9.308	42.606	0.148	1.641	9.303	1719.870
	5000	3	8.787	9.180	9.307	42.562	0.130	1.874	9.307	2134.840
	10000	3	8.778	9.190	9.307	42.442	0.136	1.858	9.308	3007.210

(c) Texas data

Table 4.9. Computational results depending on the different numbers of failures

Chapter 5 Discussion and conclusion

This chapter summarizes the dissertation and discusses possible future work. The first section presents the objectives of the dissertation and summarizes computational results. The second section provides several variations or extensions for future work.

5.1 Research summary

The main objective of this dissertation is the development of new optimization approaches to political redistricting problems with respect to essential political redistricting criteria of a strict equal population and contiguity for the use of congressional plans. The second objective is the evaluation with other representative research to get relative performance. The third objective is the foundation of various redistricting plans similar to or the same as the given plan in population, for the given plan.

This dissertation develops two exact models by formulating contiguity requirement based on the recent developments of land acquisition problems. The first model is called a spanning tree model and the second model is called a network flow model. Two models can successfully address contiguity formulation and a strict equal population in a mixed integer program while the existing approach of Garfinkel and Nemhauser (1970) ensures contiguity by the process of generating all feasible districts. Computational results show that all exact methods can find optimal solutions for small data. However, the method of Garfinkel and Nemhauser (1970) shows computational challenges in generating all feasible districts. All exact models are inefficient to find the redistricting plans for real cases. So, it can be concluded that it is necessary to develop efficient and effective heuristics.

This dissertation also implements two heuristic models in order to efficiently find high quality solutions. The first model is called a multi-scale simulated annealing and the second model is called Give-And-Take greedy algorithm. The multi-scale simulated annealing allows non-improving solutions in order to escape the local optimal solution. Give-And-Take greedy algorithm exchanges population units between a district with smaller population than the ideal population and a district with larger population than the ideal population by using a more simple contiguity procedure. Computational experiments show that Give-And-Take greedy algorithm is much more efficient for most of regular cases (5by5, 10by10, and 25by40 data). Even for the large data such as 25by40, Give-And-Take greedy algorithm shows such efficiency and also finds more various spatial shapes in the same population deviation because the algorithm visits all adjacent units and checks all possible candidate population units. For the large size of data set such as 25by 40 data, all of the heuristics can reach near-optimal solutions, and especially

The Give-And-Take greedy algorithm and the evolutionary algorithm of Xiao (2008) finds theoretical optimal solutions in $r = 10$ and $r = 5$, respectively. For Iowa congressional redistricting, all of the heuristics find redistricting plans with higher population equality than the official plan of 0.0080, which is 235 persons as the total absolute deviation from the ideal population. Among all of the heuristics, the best redistricting plan by Give-And-Take greedy algorithm shows the highest population equality with the objective function value of 0.00075, which is 22 persons as the total absolute deviation from the ideal population. The best solution by the multi-scale simulated annealing 0.0066, which is192 persons as the total absolute deviation from the ideal population and the best solution by the existing heuristic of Xiao (2008) is 0.0045, which is 131 persons as the total absolute deviation from the ideal population.

The Give-And-Take greedy algorithm can be applied to find different redistricting plans, which have same or similar population of the given plan because the algorithm efficiently finds a high quality solution even for the large size problem. Starting from the given plan as an initial solution, Give-And-Take greedy algorithm exchanges population units between a district with a larger population than the ideal population and a district with a smaller population than the ideal population, and accepts better or same plans to the given plan. Results from the application of the Give-And-Take greedy algorithm are great varieties of spatial configurations similar to or same as the given plan in population deviation.

5.2 Future work

Several variations or extensions are necessary to investigate for future work. The first extension is the addition of other political redistricting criteria to the current models. In the reality, political redistricting is a complex and time consuming task because there are many criteria considered such as population equality, minority population, contiguity, compactness, communities of interest, competitiveness and *etc*. These criteria can be dealt with either constraints or another objective function value in the current developed optimization research.

The second extension is the development of a multiobjective heuristic to political redistricting. Really, political redistricting is a multiobjective problem because several conflicting objectives should be considered at the same time. The purpose of solving a multiobjective problem is to generate compromised (non-dominated) solutions (Cohon 1978; Steuer 1986). However, most of the existing researches have been developed as a single objective problem. So, it is necessary to develop a multiobjective optimization approaches to political redistricting in order to find lots of compromised solutions. Among them, the best compromised redistricting plan can be selected. In the context of political redistricting, an effective and efficient multiobjective algorithm should be developed.

The third extension is the study of how optimization algorithms to political redistricting have been integrated into GIS. Interactive type of GUI (Graphic User Interface) composes button-click options including several heuristics so that planners can use these heuristics for redistricting problems without being an expert in operations research. Web-based GIS system with optimization algorithms can also be considered. Throughout the easy access to data and software by online, the public can have the possibility for greater participation in redistricting processes.

The fourth extension is the implementation of more effective and efficient heuristic. The Give-And-Take heuristic shows that maximum values of objective function values are worse and the ranges of solutions such as minimum values, median values and maximum values are large. More robust heuristic should be considered in have good solution ranges. A new heuristic can be implemented by using another metaheuristic or the different method to generate an initial solution. For example, a spanning tree based heuristic algorithm can be implemented.

The last extension is the focus on other district problems. There are sales districting (Hess and Samuels 1971; Zoltners and Sinha 1983; Fleischmann and Paraschis 1988), school districting (Schoepfle and Church, 1989; Ferland and Gu´enette 1990) and emergency service territories (Baker et al. 1989; D'Amico et al. 2002) and electrical power districting (Bergey et al. 2003). Different district problems have their own specific characteristics and consider their corresponding criteria. For example, sales districting much resemble political redistricting in that geographical criteria such as contiguity, compactness are major criteria. However, there are other criteria considered such as the number of territories, locations of sales representatives, balance, and maximizing profit. Based on the particular characteristics and criteria of each districting problem, creative models can be developed.

Reference

Abramson, D., Krishnamoorthy, M. and Dang, H. Simulated annealing cooling schedules for the school timetabling problem. Asia-Pacific Journal of Operational Research, 16 1-22, 1999.

Adams, B. A Model State Reapportionment Process: The Continuing Quest for 'Fair and Effective Representation. Harvard Journal on Legislation 14: 825-904, 1977.

Aerts, J. C. J. H., et al. Using Linear Integer Programming for Multi-Site Land-Use Allocation. Geographical Analysis 35(2): 148-169, 2003.

Agrell, P. J., et al. Interactive multiobjective agro-ecological land use planning: The Bungoma region in Kenya. European Journal of Operational Research 158: 194-217, 2004.

Altman, M. Is automation the answer:—The computational complexity of automated redistricting. Rutgers Computer and Law Technology Journal 23(1): 81-142, 1997.

Altman, M. Modeling the effect of mandatory district compactness on partisan gerrymandering. Political Geography 17(8): 989-1012, 1998.

Altman, M., McDonald, K. and McDonald, M. P., From Crayons to Computers: The Evolution of Computer Use in Redistricting, Social Science Computer Review 23(3): 334-346, 2005.

Altman, M. and McDonald, M.P. The Promise and Perils of Computers in Redistricting. Duke J. Constitutional Law and Public Policy 5: 69-112, 2010.

Alvanides, S. and S. Openshaw Zone design for planning and policy analysis. Geographical Information and Planning. S. J., G. S. and S. Openshaw. Berlin, Springer-Verlag: 299-315, 1999.

Alvanides, S., et al. Designing zoning systems for flow data. GIS and Geocomputation: Innovations in GIS. P. Atkinson and D. Martin. London, Taylor and Francis: 115-134, 2000.

Alvanides, S. Zone Design Methods for Application in Human Geography. PhD thesis, School of Geography, University of Leeds, 2000.

Armstrong, M.P. and P.J. Densham, Database Organization Strategies for Spatial Decision Support Systems. International Journal of Geographical Information Systems, 4(1), 3-20, 1990.

Bação, F., et al. Applying genetic algorithms to zone design. Soft Computing - A Fusion of Foundations, Methodologies and Applications 9(5): 341–348, 2005.

Back, T., D. B. Fogel, and Z. Michalewicz (Eds.) Handbook of Evolutionary Computation. New York, Osford University Press/IOP, 1997.

Baker, G. E. The totaliry of circumstances approach, New York: Agathon Press, 1990.

Baker, J.R., Clayton, E.R., and Moore, L.J.. Redesign of primary response areas for county ambulance services. European Journal of Operational Research, 41:23–32, 1989.

Barkan, J. D., et al. Space matters: Designing better electoral systems for emerging democracies. American Journal Of Political Science 50(4): 926-939, 2006.

Bergey, P. K., C. T. Ragsdale and M. Hoskote. A simulated annealing genetic algorithm for the electrical power districting problem. Annals of Operations Research 121: 33-55, 2003.

Boyce, R. R. and W. A. V. Clark The Concept of Shape in Geography. Geographical Review 54: 561-572, 1964.

Bozkaya, B., et al. A tabu search heuristic and adaptive memory procedure for political districting. European Journal of Operational Research 144(1): 12-26, 2003.

Browdy, M. Simulated Annealing: An Improved Computer Model for Political Redistricting. Yale Law and Policy Review 8: 163-179, 1990.

Cerny, V. A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications 45: 41-51, 1985.

Cohon, J. L. Multiobjective Programming and Planning, New York, 1978.

Congressional Quarterly Staff. Congressional Quarterly's Guide to 1990 Redistricting. Washington, D.C., Congressional Quarterly, Inc. 1993.

Cooper, L. Heuristic Methods for location-allocation problems. SIAM Review 6: 37-54, 1964.

Cormen, T. H.,C.E. Leiserson, et al Introduction to Algorithms, The MIT Press, 2001

Cova, T. J. and R. L. Church. Contiguity Constraints for Single-Region Site Search Problems. Geographical Analysis 32(4): 306-329, 2000.

D'Amico, S.J., Wang, S.-J., Batta, R., and Rump, C.M.. A simulated annealing approach to police district design. Computers and Operations Research, 29:667–684, 2002

David S. Johnson, Cecilia R. Aragon, Lyle A. McGeoch and Catherine Schevon. Optimization by simulated annealing: an experimental evaluation. Part I, graph partitioning Operations Research 37(6): 865-892, 1989.

Densham, P. Spatial Decision Support Systems, in D.J. Maguire, M.F. Goodchild, and D.W. Rhind (Ed.) Geographical Information Systems: Principles and Applications, Longman, London, 403-412, 1991.

Densham, P. J. and G. Rushton. Strategies for solving large location-allocation prolem by heuristic methods. Environment and Planning A 24: 289-304, 1992.

Diamond, J. T. and J. R. Wright. Efficient land allocation. Journal of Urban Planning and Development 115(2): 81-96, 1989.

Eagles, M., et al. Controversies in political redistricting: GIS, geography, and society. Political Geography 19(2): 135-139, 2000.

Ehrgott, M. and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. OR Spektrum 22: 425-460, 2000.

Ferland, J.A. and Gu´enette, G.. Decision support system for a school districting problem. Operations Research, 38(6):15–21, 1990.

Fleischmann B., and Paraschis J.N. Solving a large scale districting problem: A case report. Computers and Operations Research, 15(6):521–533, 1988.

Garfinkel, R. S. and G. L. Nemhauser Optimal Political Districting by Implicit Enumeration Techniques. Management Science 16(8): B495-B508, 1970.

Gearhart, B. C. and J. M. Liittschwager Legislative districting by computer. Behavioral Science 14: 404-417, 1969.

George, J., et al. Political district determination using large-scale network optimization. SOCIO-ECONOMIC PLANNING SCIENCES 31(1): 11-28, 1997.

Gibbs, J. P. Urban Research Methods. New York, Van Nostrand, 1961.

Glover, F. Heuristic for integer programming using surrogate constraints. Decision Sciences 8: 156-166, 1977.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley.

Gorry G A, Morton M S A framework for management information systems, Sloan Management Review 13; 56-70, 1971.

Groffrion A M. Can OR/MS evolve fast enough? Interfaces 13:10-25, 1983.

Grofman, B. Criteria for redistricting: A social science perspective. UCLA Law Review 33: 77-184, 1985.

Guo D. and H. Jin. iRedistrict: Geovisual Analytics for Redistricting Optimization, Journal of Visual Languages and Computing, 2011.

Hacker, Andrew. Congressional Districting - The Issue of Equal Representation. Washington D.C.: Brookings Institution, 1964.

Harris, C. C., Jr A Scientific Method of Districting. Behavioral Science 9: 219-225, 1964.

Helbig, R. E., et al. Political redistricting by computer. Communications of the ACM 15(8): 735–741, 1972.

Hess, S.W. and Samuels, S.A. Experiences with a sales districting model: criteria and implementation. Management Science, 18:41–54, 1971.

Hess, S. W., et al. Non-partisan political redistricting by computer. Operations Research 13: 998-1006, 1965.

Hojati, M. Optimal political districting. Computers & Operations Research 23(12): 1147-

1161, 1996.

Horn, D. L., et al. Practical application of district compactness. Political Geography 12(2): 103-120, 1993.

Horn, M.E.T., Solution techniques for large regional partitioning problems, Geographical Analysis, 27(3); 230-248, 1995.

Kaiser, H. An objective method for establishing legislative districts. Midwest Journal of Political Science 10: 200-213, 1966.

Keane, M. The size of the region-building problem. Environment and Planning A 7: 575-577, 1975.

Kirkpatrick, S., et al. Opimization by simulated annealing. Science 220: 671-680, 1983.

Laarhoven, P. J. M. and E. H. L. Aarts Simulated Annealing: Theory and Applications, Dordrecht: Reidel, 1987.

Liittschwager, J. M. The Iowa redistricting system. Annals of the New York Academy of Sciences 219: 221-235, 1973.

Macmillan, W. D. Redistricting in a GIS environment: An optimisation algorithm using switching-points. Journal of Geographical Systems 3(2): 167 - 180, 2001.

Macmillan, W. D. and T. Pierce. Spatial analysis and GIS. London; Bristol, PA, Taylor & Francis, 1994.

Macmillan, W. D. and T. Pierce. Active computer-assisted redistricting. USA: Dartmouth, Aldershot; Brookfield, Vt., 1996.

Maguire D. J., An overview and definition of GIS. In; Matuire D.J, Goodchild M F, Rhind D W (eds.), Geographical Information Systems; principles and applications. Longman, London; 9-20, 1991.

Marsten, R. E., An Algorithm for Large Set Partitioning Problems. Management Science 20(5): 774-787, 1974.

Mehrotra, A., et al. An optimization based heuristic for political districting. Management Science 44(8): 1100-1114, 1998.

Michael Falcone, A Gamer's Guild to Redistricting, The caucus, June 14, 2007

Miettinen, K. M., Nonlinear Multiobjective Optimization, Kluwer Academic, 1999.

Mills, G.., The determination of local government electoral boundaries. Operations Research Quarterly 18: 243-255, 1967.

Minor, S. D. and T. L. Jacobs. Optimal Land Allocation for Solid and and Hazardous Waste Landfill Siting. Journal of Environmental Engineering 120: 1095-1108, 1994.

Morrill, R. L. Ideal and reality in reapportionment. Annals of the Association of American Geographers 63: 463-477, 1973.

Morrill, R. L. Redistricting revisited. Annals of the Association of American Geographers 66: 548-556, 1976.

Morrill, R. L. Political Redistricting and Geographic Theory. Washington D.C., Association of American Geographers, 1981.

Musgrove, Phillip. The General Theory of Gerrymandering. U.S.A.: Sage Publications,Inc, 1977.

Nagel, S. S. Simplified bipartisan computer redistricting. Stanford Law Review 17: 863-899, 1965.

Nagel, S. S., Computers and the law and policitics of redistricting. Polity 5: 77-93, 1972.

Niemi, R. G., et al. Measuring Compactness and the Role of a Compactness Standard in a Test for Partisan and Racial Gerrymandering. The Journal of Politics 52(4): 1155-1181, 1990.

Nygreen, B. European assembly constituencies for Wales -comparing of methods for solving a political districting problem. Mathematical Programming 42: 159-169, 1988.

Openshaw, S. Optimal zoning systems for spatial interaction models. Environment and Planning A 9: 169-184, 1977a.

Openshaw, S. Spatial Analysis: Modeling in a GIS Environment, Cambridge: GeoInformation International, 1996.

Openshaw, S. and L. Rao. Algorithms For Reengineering 1991 Census Geography. Environment And Planning A 27(3): 425-446, 1995.

Parker, F. R. Racial gerrymandering and legislative reapportionment. Washington, D.C., Howard University Press, 1989.

Pirlot, M. General local search methods. European Journal of Operational Researh 92: 493-511, 1996.

Plane, D. A. Redistricting Reformulated - A Maximum Interaction Minimum Separation Objective. Socio-Economic Planning Sciences 16(6): 241-244, 1982.

Reeves, C. R., Ed. Modern heuristic techniques for combinatorial problems. Oxford, Blackwell Scientific Publications, 1993.

Reock, E. C., Jr. Measuring Compactness as a Requirement of Legislative Apportionment. Midwest Journal of Political Science 5: 70-74, 1961.

Robertson, I. M. L. The delimitation of local government electoral areas in Scotland: a semi-automated approach. The Journal of the Operational Research Society 33: 517-525, 1982.

Polsby, David and Robert Popper. The Third Criterion: Compactness as a Procedural Safeguard Against Partisan Gerrymandering. Yale Law & Policy Review 9: 301-353, 1991.

Polsby, David and Robert Popper. Ugly: An Inquiry into the Problem of Racial Gerrymandering Under the Voting Rights Act. Michigan Law Review 92: 588-651, 1993.

Ronan, R.. Sales territory alginment for sparse accounts. OMEGA The International Journal of Management Science, 11:501–505, 1983.

Ruzika, S. and M. M. Wiecek. Approximation Methods in Multiobjective Programming. Journal of Optimization Theory and Applications 126(3): 473-501, 2005.

Schoepfle O B, Church R L, A fast, network-based, hybrid heuristic for the assignment of students to schools Journal of Operations Research Society 40 1029–1040, 1989.

Schwartzberg, J. E. Reapportionment, Gerrymanders, and the Notion of Compactness. Minnesota Law Review 50: 443-452, 1966.

Shirabe, T. A Model of Contiguity for Spatial Unit Allocation. Geographical Analysis 37: 2-16, 2005.

Shirabe, T. Districting modeling with exact contiguity constraints. Environment and Planning B: Planning and Design, 36; 1053-1066, 2009

Stern, R. S. Political Gerrymandering: A Statutory Compactness Standard as an Antidote for Judicial Impotence. The University of Chicago Law Review 41: 398-416, 1974

Steuer, R. E. Multiple Criteria Optimization: Theory, Computation, and Application, New York:John Wiley, 1986.

Stewart, T. J., et al. A genetic algorithm approach to multiobjective landuse planning. Computers & Operations Research 31: 2293-2313, 2004.

Synder, S. and C. Revelle Multiobjective grid packing model: an application in forest management. Location Science 5(3): 165-180, 1997.

Thoreson, J. D. and J. M. Liittschwager. Legislative districting by computer simulation. Behavioral Science 12(3): 237-347, 1967.

Tornqvist, G., et al. Title Multiple location analysis, Lund, Gleerup, 1971.

Ulungu E.L., Teghem J., Fortemps Ph., Tuyttens D., MOSA method: a tool for solving multiobjective combinatorial optimization problems. Journal of Multi-Criteria Decision Analysis, 8, 221-236, 1999.

Vickrey, W. On the prevention of gerrymandering. Political Science Quarterly 76(1): 105-110, 1961.

Wattson, Peter S., How to Draw Redistricting Plans That Will Stand Up in Court, National Conference of State Legislatures National Redistricting Seminar, Providence, Rhode Island, September 26, 2010

Weaver, J. B. and S. W. Hess. A procedure for nonpartisan districting: development of computer techniques. The Yale Law Journal 72: 288-308, 1963.

Wei, B. C. and W. Y. Chai. A Multiobjective Hybrid Metaheuristic Approach for GIS-based Spatial Zoning Model. Journal of Mathematical Modelling and Algorithm 3: 245-261, 2004.

Well, D. Against A affirmative Gerrymandering. Representation and Redistricting Issues. B. Grofman, A. Lijphard, R. B. McKay and H. A. Scarrow. Lexington, Lexington Books, 1982.

White S. R. Concepts of Scale in Simulated Annealing. Proceedings IEEE International Conference on Computer Design, 648-651, 1984.

Williams, J. C. Political Redistricting: A Review. Papers in Regional Science: The journal of the RSAI 74 (1): 13-40, 1995.

Williams, J. C. A Zero-One Programming Model for Contiguous Land Acquisition. Geographical Analysis 34: 330-349, 2002.

Williams, J. C. and C. S. ReVelle. A 0-1 programming approach to delineating protected reserves. Environment and Planning B 23: 607-624, 1996.

Wright, J. R., et al. A Multiobjective integer programming model for the land acquisition problem. Regional Science and Urban Economics 13: 31-53, 1983.

Xiao, N. A unified conceptual framework for geographical optimization using evolutionary algorithms. Annals of the Association of American Geographers, 98(4): 795-817, 2008.

Yong, H. P. Measuring the compactness of legislative districts. Legislative Studies Quarterly 13: 105-115, 1988.

Zoltners, A. A. and Sinha, P. Sales territory alignment: a review and model. Management Science, 29:1237–1256, 1983.

Zionts, S. and J. Wallenius. An interactive programming method for solving the multiple criteria problem. Management Science 22(6): 652-663, 1976.