UTILIZING GENETIC ALGORITHM TO DETECT COLLUSIVE OPPORTUNITIES IN DEREGULATED ENERGY MARKETS

by
Barış Esen

Submitted to the Graduate School of
Engineering and Natural Sciences
in fulfillment of the requirements for the degree of
Master of Science

Sabancı University,
January 2019

Utilizing Genetic Algorithm to Detect Collusive Opportunities in Deregulated Electricity Market

APPROVED BY:

Assoc. Prof. Dr. Güvenç Şahin..
(Thesis Supervisor)

Asst. Prof. Dr.Emre Çelebi..

Asst. Prof. Dr.Tevhide Altekin..

DATE OF APPROVAL: 03.01.2019

ACKNOWLEDGEMENTS

I would like to express my feelings to everyone who supported me throughout my thesis. First and foremost, I really want to thank my supervisor Assoc. Prof. Dr. Güvenç Şahin for his everlasting support, guidance, and feedback along the way to finish my thesis. He always provided me with the energy to finish up my works and gave me a positive perspective on whatever I do.

I would particularly want to thank Dr. Danial Esmaeili Aliabadi who assisted me to develop my algorithm. He always welcomed me in his office and helped me during the other parts of my thesis. He never hesitated to share information and knowledge.

I would like to thank my thesis committee for spending their precious time on my thesis dissertation.

I owe my greatest gratitude to parents. My father Prof. Dr. Ramazan Esen is always inspiring and leading. I always admired his academic life and works. My mother Ayşe Esen always supports me in every challenge that I encounter. I also want to thank my sister Duygu Esen Özel. Even though she is away, she always stood by me.

My completion of this thesis couldn’t have been accomplished without the endless support of Zeynep Selçuk and Mustafa Kemal Taş. I am grateful for their endless belief in my work.

Finally, I want to thank people and organization that have established Sabanci University.

UTILIZING GENETIC ALGORITHM TO DETECT

COLLUSIVE OPPORTUNITIES IN DEREGULATED ENERGY MARKETS

Barış Esen

Industrial Engineering, Master's Thesis, 2019

Thesis Supervisor: Assoc. Prof. Dr. Güvenç Şahin

Keywords: Deregulated electricity markets, genetic algorithms, parameter estimating, collusion opportunities, independent system operator

Abstract

Deregulated electricity markets allow competition over the electricity price among the power companies. However, in an oligopolistic environment, the strategic behavior of the power companies in the electricity market may lead to collusive opportunities. The independent system operator (ISO) is an authorized entity which is responsible for administrating the electricity market. Therefore, ISO shall be able to detect and avoid collusive opportunities among generators. In this study, we propose a metaheuristics approach to assist ISO in the decision-making process to prevent collusions. We develop a method, based on principles of genetic algorithm to detect the collusive opportunities in deregulated electricity markets. We test our algorithm on three problems of varying size. Our results are promising in terms of both speed and accuracy. For the large-scale problem, our algorithm works much faster than the existing alternatives in the literature.

GENETİK ALGORİTMASI KULLANARAK SERBESTLEŞMİŞ ELEKTRİK PİYASALARINDA GİZLİ ANLAŞMALARI TESPİT ETME

Barış Esen

Endüstri Mühendisliği, Yüksek Lisans Tezi, 2019

Tez Danışmanı: Doç. Dr. Güvenç Şahin

Anahtar Kelimeler: Serbestleşmiş elektrik piyasası, genetik algoritması, parametre tahmini, gizli anlaşma olasılıkları, bağımsız sistem yöneticisi

Özet

Serbestleşmiş elektrik piyasaları, elektrik şirketleri arasındaki elektrik fiyatı rekabetine olanak sağlar. Ancak, az sayıda elektrik şirketinden oluşan bir piyasada, şirketlerin stratejik yaklaşımları gizli anlaşmalara yol açabilir. Bağımsız bir sistem işletmecisi, elektrik piyasasını idare etmekten sorumlu yetkili bir kuruluştur. Bu nedenle, elektrik şirketleri arasında oluşabilecek gizli anlaşmaları tespit edip önleyebilecek niteliğe sahip olmalıdır. Bu çalışmada, gizli anlaşmaların önlenmesi için karar alma sürecinde bağımsız sistem işletmecisine yardımcı olacak meta-sezgisel bir yaklaşım öneriyoruz. Oluşturduğumuz meta-sezgisel yöntem, genetik algoritma prensiplerine dayanmaktadır. Algoritmamızı değişken büyüklükteki üç örnek problem üzerinde test ediyoruz. Sonuçlarımız, hem hız hem de doğruluk açısından umut vericidir. Büyük ölçekli bir problem karşısında, algoritmamız literatürdeki mevcut alternatifinden çok daha hızlı çalışmaktadır.

Table of Contents

LIST OF FIGURES

Chapter 1

1. Introduction

The electricity price is one of the major concerns of every individual and organization on a daily basis. Electricity consumers expect fair and stabilized electricity prices to manage their electricity consumption according to their economic incentives. To meet these expectations of electricity consumers, the proper electricity pricing is crucial. The electricity pricing (sometimes referred to as electricity tariff) is the process of determining the electricity price. The electricity price is determined in a system called the electricity market. The electricity market enables the trading of electricity purchases through bids to buy and sales through offers to sell. The power companies compete in the electricity market to obtain market share to make profits. On the other hand, governments play role in the regulation of the electricity market.

The electricity markets are divided into two sections in terms of regulation: (i) regulated electricity markets and (ii) deregulated electricity markets. Regulated electricity markets contain a utility company that establishes a monopoly with complete control over the market. The utility company that owns the regulated electricity market, have the right to set electricity prices for consumers. The consumers have no other option to choose and they need to abide by the designated electricity prices.

Deregulated electricity markets, however, allows competition over the electricity price which benefits consumers by allowing them to compare prices of each power company. Power companies compete to obtain partial or complete control over the deregulated electricity market with an auction mechanism. In the auction, each company declares their power generation capacity and also bid a unit price for the electricity production.

The decision maker of the auction mechanism is the Independent System Operator (ISO). ISO is an organization that maximizes the social welfare through the electricity markets. In electricity pricing auction, ISO solves a decision-making problem to allocate a share of the market demand to each company based on bid-offers and the production capacities. According to obtained results from the decision-making problem, ISO distributes the market share of each power company. The process of solving the decision-making problem and the distribution of the market share is called the market clearing process.

The strategic behavior of the power companies in auctions of the deregulated electricity market may lead to collusion opportunities. The collusion among power companies can be defined as explicit or implicit (tacit) non-competitive agreement to increase the bids to obtain higher market shares and also to increase prices. Sweeting (2007) showed that power companies might be engaged in tacit collusion in the UK. Similarly, Fabra and Toro (2005) studied on collusion in the Spanish electricity market and showed the collusive situations may exist.

If there exists some type of collusion in the market, the ISO should be able to identify. However, this task is not easy hence the effect of the collusion is hard to recognize without knowing any previous agreements among the power companies.

Aliabadi *et al*. (2016) showed that identification of collusion in the deregulated electricity market can be achieved when sufficient conditions exist. In order to identify the collusion opportunities, they proposed an algorithm based on a mathematical programming problem formulation of the market clearing process and the behavior of the power companies. However, the proposed algorithm is not computationally efficient to obtain exact solutions. Therefore, we attempted to attack this problem to find better alternatives to obtain solutions faster.

Contribution of thesis study can be summarized as follows:

- We develop an algorithm to show that the collusive behavior of power producers can be identified with a metaheuristic approach.
- We compare our algorithm with the existing exact algorithm in the literature in terms of accuracy and speed.
- We work with three different problems to measure the performance of the algorithm.

The remainder of the thesis is organized as follows: Chapter 2 presents the problem environment with a literature review, problem definition, and related mathematical notation, respectively. In Chapter 3, we explain our metaheuristic approach. We present our case studies and computational results in Chapter 4. Finally, in Chapter 5, we conclude the thesis with ultimate remarks and future research directions.

Chapter 2

2. Problem Environment

In this chapter, the problem environment is discussed. First, we present a literature review on collusive opportunities in the deregulated electricity markets. Next, we discuss our problem definition and our notation. Finally, we present the mathematical model in Aliabadi *et al.* (2016).

2.1. Literature Review

Collusion opportunities in the deregulated electricity markets have not been studied broadly in the literature. However, studies regarding the strategic behavior of the power companies may give insights about collusion opportunities. For example, such strategic behaviors are explained through Conjectural Variation models in Song (2004), Ruiz (2010), and Ruiz (2012). As an alternative to these models, Wang (2009) and Botterud (2007) utilized simulation models to study strategic behaviors of the power companies in electricity markets.

In recent years, with the enlightenment of studies conducted on behaviors of the power companies, researchers have developed mechanisms to detect and prevent collusion opportunities in the deregulated electricity markets. Liu and Hobbs (2013) are the first researchers who studied modeling tacit collusion in a repeated game setting.

Aliabadi *et al.* (2016) presented an analytical model to determine collusion opportunities in deregulated electricity markets. To the best of our knowledge, this is the first study regarding the collusion opportunity detection in deregulated electricity markets. We discuss this work later in more detail.

2.2. Problem Definition and Notation

We consider that the ISO runs a day-ahead market for each hour of the next day. ISO regulates the transmission grid under a settlement system based on an auction mechanism. This auction mechanism is utilized to manage the deregulated electricity market. For each hour of the day-ahead market, power company-i offers its bid price (b_i) and available production capacity to the ISO. The bid-offer options for each company vary between the upper and the lower bound on the electricity selling price to the market. Lower bound must be bigger than the production cost to make a profit. On the other hand, the upper bound must be reasonable to stay in the competition for the market share. For simplicity, we assume a single bid price for a company and neglect the flexible or block bid prices. Then ISO solves a decision-making problem for each hour of the day-ahead market to clear the market bids while maximizing the social welfare. The solution for market clearing process determines the power injected by each GenCo (P_i), the voltage-angle (θ_i), and the unit price of electricity on each node of the network. The unit price at each node is known as locational marginal price (LMP_i).

The interconnected network for delivering the electricity from power companies to electricity consumers is called electricity grid. The electricity grid consists of many power companies connected to each other by transmission lines. For modelling purposes, the transmission grid is represented as a network.

Figure 1 shows a graphical network representation of an example that we use in our experiments. Each node in the graph represents a power company with a maximum power generation level (P_i^{max}) and a demand center with required power injection level (D_i). The transmission line between two nodes, i and j, could afford to transmit only up to a certain level of electricity (F_{ij}^{max}).

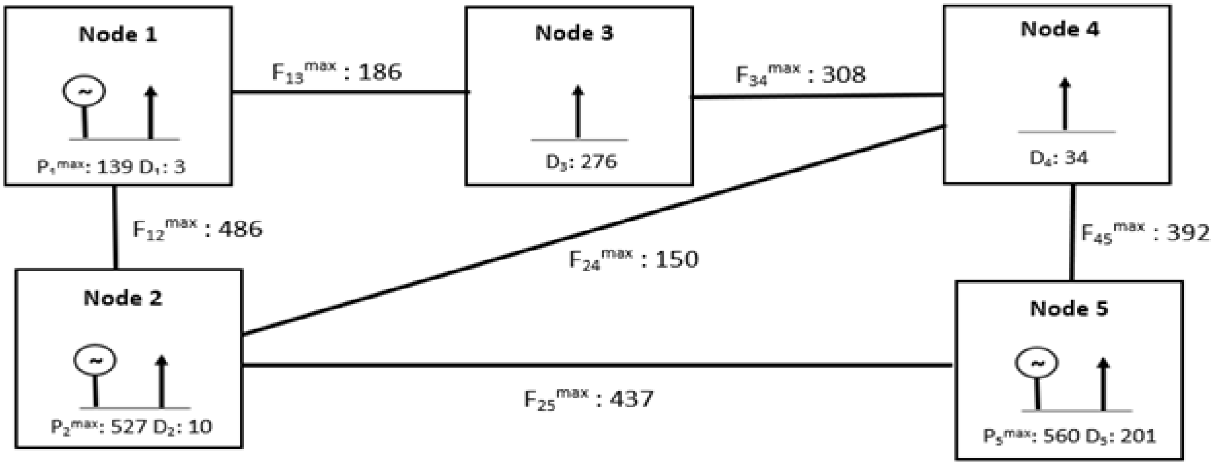

Figure 1. Graphical representation of the transmission network.

2.3. Mathematical Model of Market Clearing Process

The mathematical model of our study is primarily based on electricity market clearing process. Alternating Current Optimal Power Flow (AC-OPF) problem and Direct Current Optimal Power Flow (DC-OPF) problem are the acknowledged formulation approaches that can be utilized for the market clearing process.

AC-OPF problem is a non-convex mixed integer linear problem that is based on minimizing the total variable generation costs. AC-OPF problems are generally approximated by the DC-OPF formulation in a linearized form. Hence DC-OPF is more tractable, we conducted our experiments with the DC-OPF formulation so that our collusion detection approach is also tractable. The DC-OPF problem formulation can be given as follows:

$$\text{Minimize}_{P_i,\, \theta_i} \quad z = \sum_i b_i P_i, \tag{1}$$

$$\text{subject to} \quad P_i - D_i = \sum_{ij \in BR} y_{ij}\,(\theta_i - \theta_j) \qquad (\mathrm{LMP}_i)\,\forall i \tag{2}$$

$$P_i \le P_i^{max} \qquad (\emptyset_i^{high})\ \forall i \tag{3}$$

$$P_i \geq 0 \qquad (\emptyset_i^{low})\ \forall i \qquad (4)$$

$$\left|y_{ij}\left(\theta_i - \theta_j\right)\right| \leq F_{ij}^{max} \qquad \forall ij\ \in BR \qquad (5)$$

$$\theta_1 = 0 \qquad (6)$$

where decision variables are P_i denoting power injection level and θ_i denoting voltage angle. The parameters D_i, P_i^{max}, and F_{ij}^{max} are already defined in Section 2.2. The set of all available distinct transmission lines is denoted by BR while the susceptance of the line between node i and node j is shown as y_{ij}.

In the mathematical formulation (1)-(6), the objective function in (1) minimizes the total cost of produced electricity for all power companies. Constraint (2) is the flow balance constraint which ensures the transmission of the excessive generated power in a node towards the other connected nodes. Constraint (3) limits the power injection level up to the capacity of the corresponding power producer at each node. Constraint (4) ensures that the amount of the power production should be nonnegative. Constraint (5) restricts the power transmission on each transmission line with a certain level of electricity. Constraint (6) sets the voltage angle at the first node arbitrarily as 0 to serve as an angular reference for the remaining nodes.

According to an optimal solution of the model in (1)-(6), the profits of power companies (r_i) are calculated as

$$r_i = P_i(LMP_i - c_i) \qquad (7)$$

where c_i is the adjusted production cost of electricity by power company i and $(LMP_i - c_i)$ is the cost of producing a unit power ($/MW).

2.4. Existing Work on Determining Collusion Opportunities

Aliabadi *et al*. (2016) developed game-theoretic understanding of collusion among power generators in electricity markets. They defined the set of submitted bids by all power companies

($b_1 \in B_1, \ldots, b_n \in B_n$) as the “state” of the game and utilized DC-OPF to calculate the payoffs of the power companies. Thereon, they defined two types of collusive equilibrium state:

- The *strong collusive state* is defined as the equilibrium state where all power companies have no incentive to deviate. Every power company has benefits to stay in this state which is similar to a Nash equilibrium state in game theory. However, the profits of all companies with strong collusion are greater than the Nash equilibrium without any collusion.
- The *weak collusive state* is defined as the equilibrium state where a power company may have some incentive to leave the collusive state before the game reaches a Nash equilibrium. These short-term deviations in power company strategies may increase the short-term profit of the company choosing to move out of the collusive state. However, the game will eventually get to the Nash equilibrium when no such other equilibrium exists.

Aliabadi *et al.* (2016) proposed an algorithm to detect weak and strong collusive equilibrium states. The algorithm is initiated by finding all Nash equilibria so that the Nash payoffs for all companies can be calculated. Thereon, they enumerate all distinct bid sets to detect those with collusive characteristics. Both finding all Nash equilibria and enumerating all possible auction states are computationally very expensive and indeed theoretically intractable since the computational complexity is $O(2^n)$ while n is the number of power companies. In this study, we attempt to reduce the number of calculated payoff profiles by utilizing a metaheuristic approach.

Chapter 3

3. A Meta-Heuristic Approach

In this chapter, we explain our meta heuristic approach to detect collusive states in an oligopolistic deregulated market structure. First, we present a literature review on the genetic algorithm. Next, we explain settings and finally the components of our version of the genetic algorithm.

3.1. A Literature Review on Genetic Algorithm

Genetic algorithm is well known and a commonly used metaheuristic in operations research and computer science. The idea of the algorithm was derived from mimicking the natural selection in biological environments. The origins of the genetic algorithm can be traced back to the early 1950s. Evolution computing was initially developed by Nils Aall Baricelli (1954) and Bremermann (1958). Bremermann showed that solving optimization problems with the genetic algorithm was possible. Subsequent necessary elements of the genetic algorithm were described in books by Fraser and Burnell (1970), and Crosby (1973). In their work, the concept of mutation, cross-over, and the selection was explicitly described. The genetic algorithm was popularized by John Holland's work on the computer simulation of evolution. In late 1970's he introduced "Schema Theorem" which also was called the fundamental theorem of the genetic algorithm and then extended by Goldberg (1989). The schema theorem claims that low order schemata with high fitness increase exponentially in frequency in successive generations. The concept of "fitness function" has drawn more attention after schema theorem was presented. Consequently, Schaffer (1985) has introduced the multi-objective evolution algorithm which created another area of focus.

The following studies over the genetic algorithms have deemphasized the schema theory since it has failed to make predictions about the population composition and speed of population convergence. Vose and Liepins (1991) introduced the complete geometric picture of genetic algorithm's behavior with an exact model. Nix and Vose (1992) have attempted and failed to apply Markov Chain analysis on stochastic models due to high dimensions and non-linearities. John Koza (1992) has used a genetic algorithm to evolve programs to perform certain tasks thus he called "Genetic Programming (GP)". In the late 1990s, self-adaptatively and multi-objective genetic algorithms have studied by researches. Smith and Fogarty (1996) expanded this idea by dynamically changing mutation rates in a genetic algorithm.

In recent years, many researchers have focused on using genetic algorithms as a tool. In this respect, studies focus on solving well-known problems such as the Traveling Salesman Problem and Flexible Job Shop Scheduling Problems. In our study, we develop a method, based on principles of the genetic algorithm to detect the collusive opportunities in deregulated electricity markets.

3.2. Settings for Genetic Algorithm

As explained in Section 2.4, the algorithm proposed in Aliabadi *et al*. (2016) cannot be utilized for the problems where many payoff profiles exist either due to the number of power companies or the size of the possible bid offers. Therefore, we develop an efficient search algorithm to detect collusive opportunities. We choose to perform this search by utilizing a genetic algorithm since the genetic algorithm is easy to implement and modify when the target is not well defined.

The algorithm initializes the search with a population of randomly generated individuals. Subsequently, the algorithm selects the individuals to increase survivability. The survivability is measured by the fitness function. To converge the initial population to better-qualified individuals, we utilize cross-over operation. On the other hand, we utilize mutation operation in order to sustain divergence. To terminate the search procedure, we use the trivial maximum number of generations rule. A flow chart of our genetic algorithm is shown in Figure 2. We will explain these components later in detail.

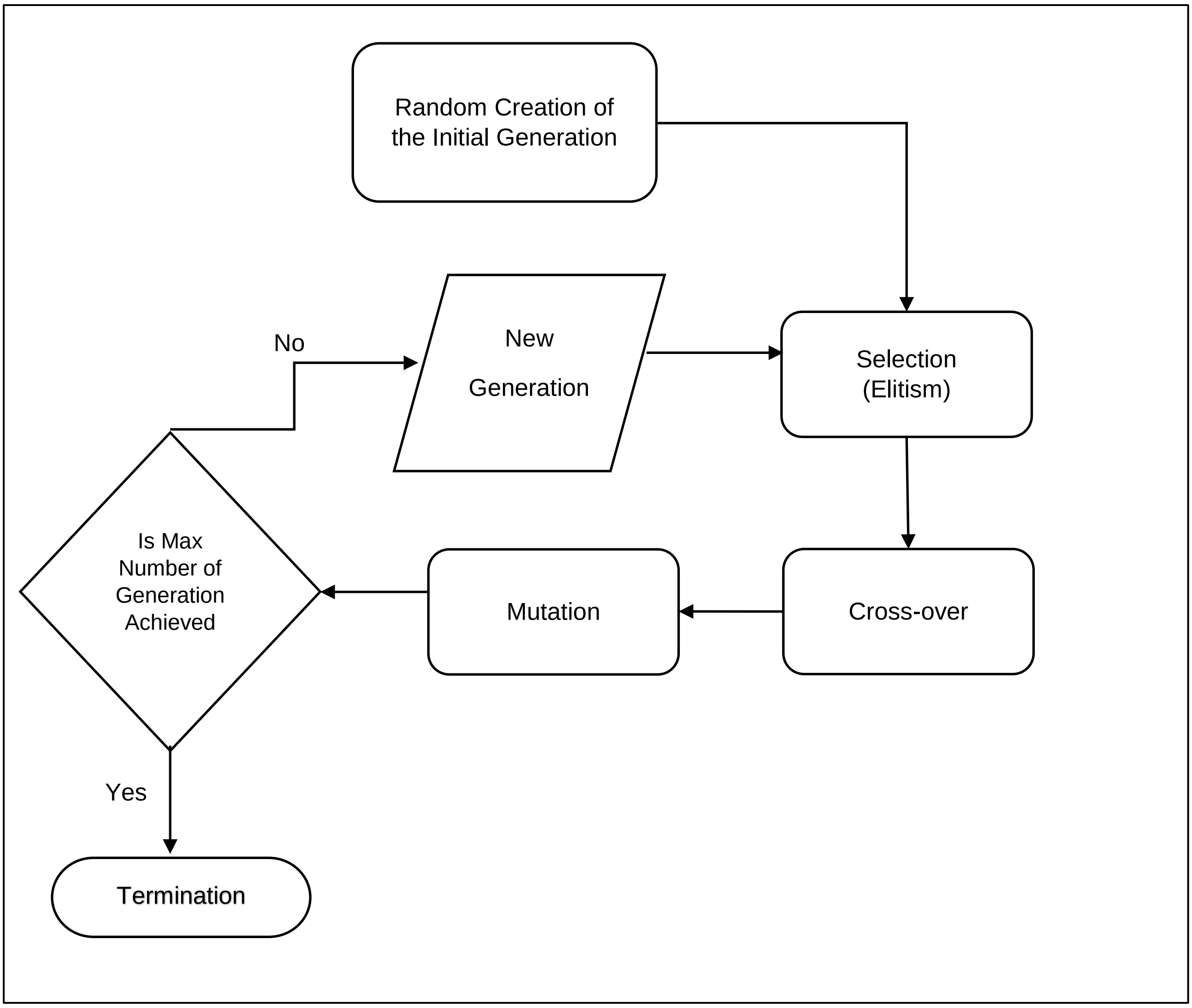

Figure 2. Flow chart of the genetic algorithm.

3.3. Components of Genetic Algorithm

In this section, we describe the general search mechanism and each component of the genetic algorithm implementation in our study. In this respect, we explicitly define the components and the required parameters of the general search mechanism.

3.3.1. Chromosome and Individual Representation

To implement any genetic algorithm, we need to define the chromosome and individual representation of the solution. In the present work, power companies have their bid-offers for a unit megawatt price of power (MW/$). Each bid-offer corresponds to a chromosome in our representation. Consequently, the bid set is a set of bid-offers for all companies in the market which forms an individual representation. The individuals are defined as a string of integers. The example of an individual with n power companies is shown in Figure 3.

b_1	b_2	b_3	$\dots$	b_n

Figure 3. Representation of an individual with n chromosomes

3.3.2. Initial Population

To obtain the diversification in the first generation, we randomly select bid-offers between marginal cost and price cap of each power company. The size of the population corresponds to one of the parameters that we tune to obtain a better solution.

3.3.3. Fitness Function

The fitness function is the most important component of the genetic algorithm. It evaluates all the individuals inside the population then assigns a value to an individual which we called fitness score. If an individual has high fitness score then the survivability of this individual is high, if an individual has low fitness score, it is the other way around. Fitness functions are problem dependent. In this study, we make an extra diligent effort to define the most appropriate fitness function to obtain fast and concrete results.

3.3.4. Selection

Selection methods are utilized to randomly select individuals according to their fitness scores. Two most commonly used selection methods for the genetic algorithm are roulette wheel selection and tournament selection. In our setting, we utilize roulette wheel selection since it is the most common selection mechanism.

The basis for the selection mechanisms dates back to the 1970s. In the aforementioned study of Holland (1975), the proportionate selection method was developed to examine the regions to find promising sections. According to their fitness score, each individual has survival probabilities towards the next generation. If f_i denotes fitness score corresponding to an individual i, then its selection probability is $p_{i=}\frac{f_i}{\sum_{j=1}^{N} f_i}$ where N is the number of individuals.

In the roulette wheel method, we divide all individuals in the population proportional to their fitness scores. For example, if an individual A has a fitness score of 10 and all other individuals in the population have a combined fitness score of 90, then we set this individual A's fitness ratio as 0.1. Subsequently, we allocated each individual's fitness ratio in the population by dividing a wheel into portions we spin the roulette wheel, if the wheel stops at 10 percent subsection which is allocated for individual A, then we select this individual to perform cross-over and other mechanisms. Figure 4 illustrates the roulette wheel selection.

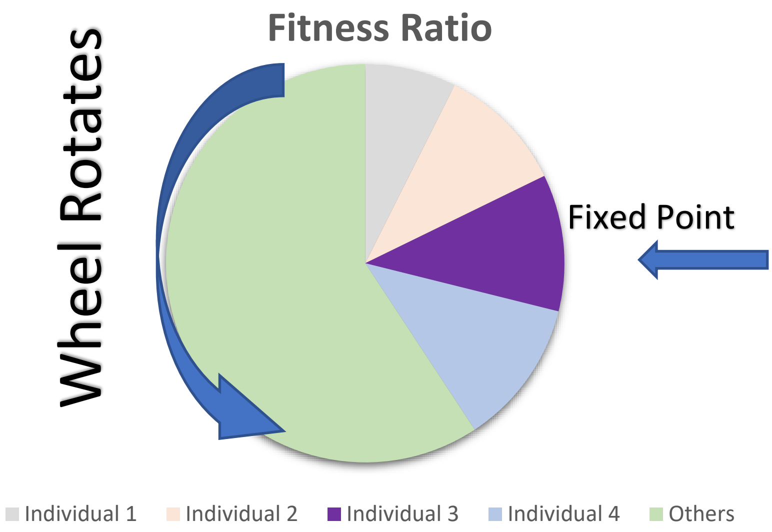

Figure 4. Roulette wheel selection method.

3.3.5. Elitism

Elitism in the genetic algorithm is the concept of preventing the random destruction of good genetic information. In this strategy, a small proportion of individuals with the best fitness scores are chosen in the current generation to pass directly to the next generation avoiding crossover and mutation operations. These individuals are marked as an "elite". One of the parameters of the algorithm to tune designates the proportion of elite individuals. However, this tuning effort could increase the complexity of our study drastically. Therefore, for simplicity, we select the 10 percent of the best-fitted individuals of each population as elitism proportion.

3.3.6. Crossover Operation

Cross-over is an operation to combine the genetic information of two chromosomes to generate new chromosomes. The combined chromosomes are called as parent chromosomes and offspring is called as child chromosome. Cross-over operations are utilized to increase genetic variations.

The initial step of cross-over operation is to select individuals to pair. Each pair consists of two individuals. In our experiments, we use the roulette wheel selection method to select from the population, discarding the elite individuals. Subsequently, we utilize the roulette wheel selection method to select its pair from the remaining population.

The second step involves determining the pairs to cross-over. Each pair has a chance of cross over with P_c. To determine the pairs to crossover, we randomly generate a random number between 0 and 1. If the generated random number is larger than the cross-over probability threshold, we do not perform the cross-over and we pass individuals of that particular pair to the very next generation. However, if the generated number is less than the cross-over probability we perform the cross-over for the corresponding pair.

The third step performs the cross-over. The most common methods for cross-over operation are single-point and two-point crossover methods in the literature. In our setting, we use the two-point crossover. In this operation, parent chromosomes are divided into three parts by randomly selected two points. Then they exchange genetic information between these two points to generate child chromosomes as illustrated in Figure 5.

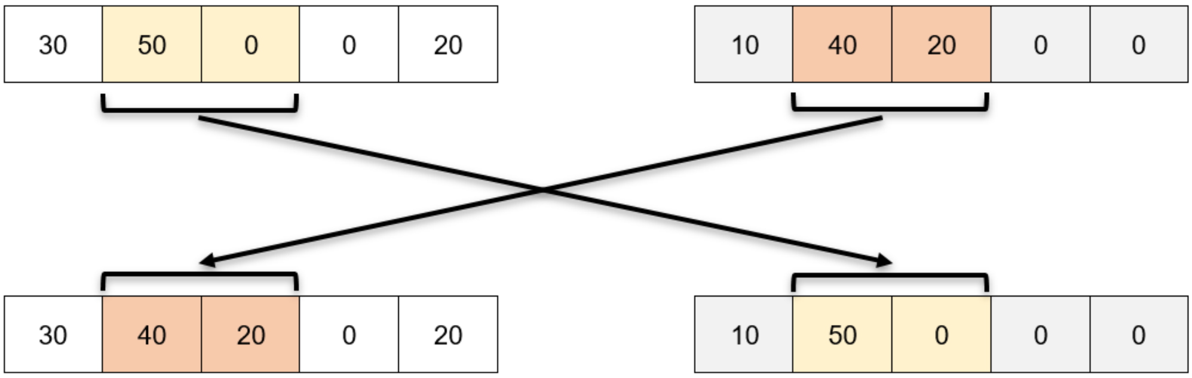

Figure 5. Cross-over operation.

The final step is to replace the parents with the offspring to maintain the stability in the number of individuals in the population.

3.3.7. Mutation Operation

Mutation operation occurs after the cross-over terminates. If an individual is not marked as an elite in the current population it has a chance of mutating (P_m). Mutation operation alters one or many chromosomes of the individual to maintain genetic diversity. A common method of implementing the mutation operation involves determining a single chromosome with generating the random number. This method is called single-point mutation. Other types are inversion and floating-point mutation. In our study, we use single-point mutation.

The initial step of single-point mutation is to determine the chromosome to be selected. In our experiments, we index the chromosomes of individuals starting from 1. Then, we generate a random number in a range of the number of chromosomes. The generated random number indicates the index of chromosome to mutate.

Once we determine the chromosome to mutate, we perform the mutation operation on a chromosome. Since the chromosome represents a single bid-offer of the particular power company, it can get values in bid-offer options to the corresponding power company. If the bid-offer option size is larger than a single bid-offer, we create a list of potential new values for that chromosome while discarding its initial value. Otherwise, we select another chromosome to mutate. If the potential new values list for the corresponding chromosome is not empty, we randomly select a value from that list and assign to that chromosome. Therefore, we generate a mutated individual. Figure 6 illustrates the mutation operation according to our individual representation.

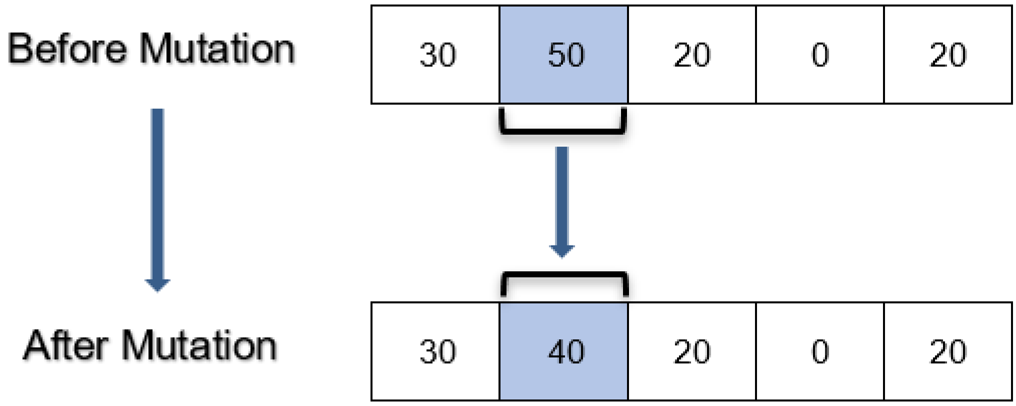

Figure 6. Example for mutation operation on a single chromosome.

3.3.8. Population Size

Population size is a parameter that controls the number of individuals in each generation. We replace parent individuals with child individuals. Hence, the size of the population is constant throughout the generations. Tuning this parameter is crucial because having large population may increase computational burden while a small population may not be sufficient to obtain good results.

3.3.9. Maximum number of Generations (Epoch)

Epoch denotes the maximum number of generations to be created. It also provides the termination criterion. For instance, if the maximum number of generations is 5, then after generating 5 new population, our algorithm stops.

3.3.10. Number of Replications

In a genetic algorithm, the probability is an important factor. Therefore, conducting experiments in a single run are not sufficient to obtain all possible results from a parameter set.

In our study, we conduct many experiments with the same parameters to obtain as many as possible results. However, determining the number of replications while performing other experiments on parameters is difficult. This task increases the computational burden exponentially. In this respect, we set the number of replications as 20. Therefore, the results for each parameter set consists of distinct results obtained by 20 replications.

Chapter 4

4. Computational Experiments

In this chapter, we explain the computational experiments with our meta heuristic approach. First, we present problems used in the computational study. Next, we present and explain the results of computational experiments.

4.1. Problems for Computational Study

In order to test the accuracy and efficiency of our algorithmic approach, we generate test problems. However, the first and foremost challenge in this task is to ensure that problems demonstrate a collusive market structure. Yet, it is another challenging task itself to create such problem settings. In this respect, we work with two transmission grid settings that are created artificially based on characteristics of real-life settings. Using the second grid, we generated two problem instances where the second is a larger problem due to the larger number of bids from generators.

4.1.1. Small Problem

The small example is the case study in Aliabadi *et al*. (2016) for which the collusive states have been exactly identified already. The problem setting is shown in Table 1 and the Figure 7. P_i^{max} refers to maximum power in megawatts (MW) that can be produced in GenCo-i with a cost of c_i. Finally, B_i is the set of bid-offer options b_i by each GenCo.

ID	P_i^{max}(MW)	c_i ($/MW)	B_i ($/MW)
GenCo-1	139	20	{20,25,30,35,40,45,50}
GenCo-2	527	20	{20,25,30,35,40,45,50}
GenCo-5	560	30	{30,35,40,45,50}

Table 1. Small problem demand load data.

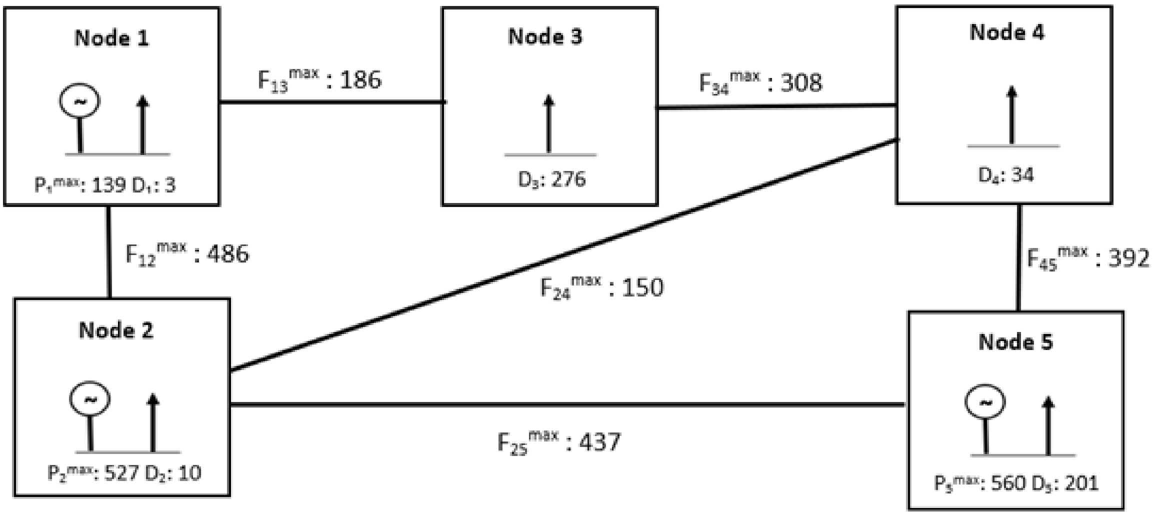

Figure 7. Small problem network structure.

In this transmission grid network, there are five nodes with six transmission lines. Each node has one power company and one demand center. The first two power companies (GenCo-1 and GenCo-2) have seven distinct bid-offer options and fifth power company has five different bid-offer options. In total, we have 7*7*5=245 bid-offer states in the market and this problem is relatively easy to solve with Aliabadi *et al*.'s algorithm to find collusive states.

4.1.2. Medium Problem

To increase the size of the problem, we add additional two more nodes to the network. The demand load data and the network structure of the problem are given in Table 2 and Figure 8, respectively. In this problem, there are 1225 bid-offer states in total in the market.

ID	P_i^{max}(MW)	c_i ($/MW)	B_i ($/MW)
GenCo-1	36	20	{21,26,31,36,41,46,51}
GenCo-2	34	20	{22,27,32,37,42,47,52}
GenCo-5	30	30	{33,38,43,48,53}
GenCo-6	31	10	{14,24,34,44,54}

Table 2. Medium problem demand load data.

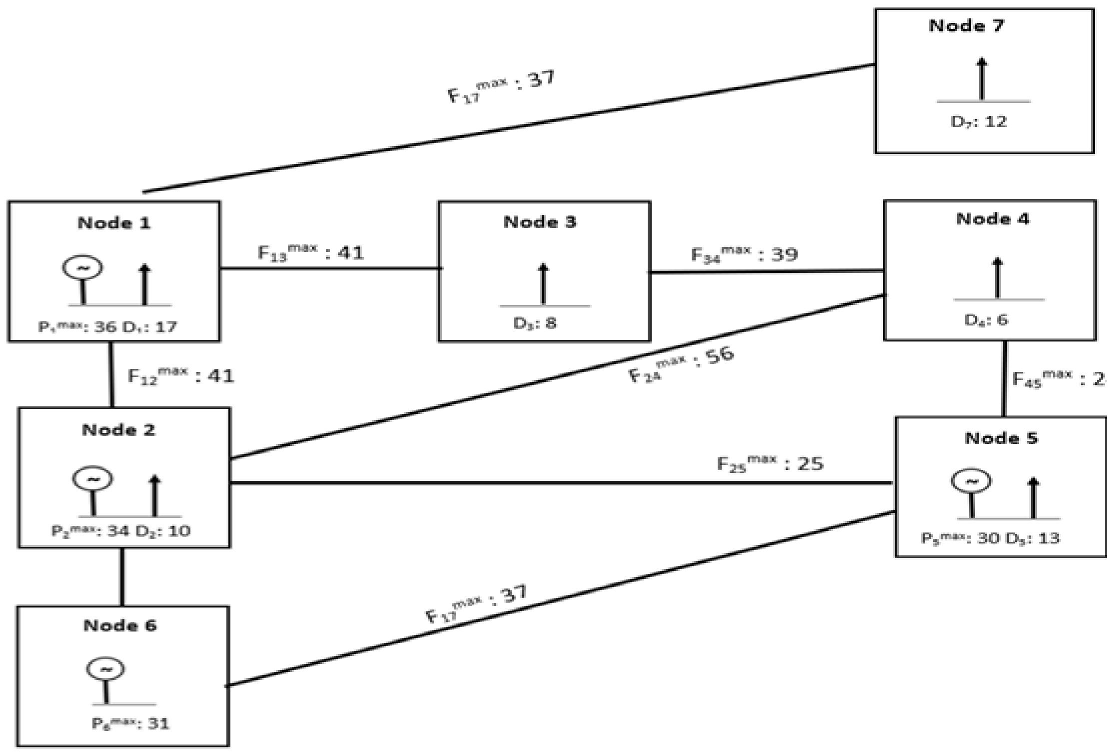

Figure 8. Medium problem network structure.

4.1.3. Big Problem

To further increase the size of the problem, we utilize the network structure of the medium problem. However, we increase the number of bid options for every GenCo. The demand load data is given in Table 3. This problem has 45056 bid-offer states. We use this problem setting to compare our results only in terms of speed with the existing algorithm in Aliabadi *et al.* (2016).

ID	P_i^{max}(MW)	c_i ($/MW)	B_i ($/MW)
GenCo-1	36	20	{21,23,25,27,29,31,33,35,37,39,41,43,45,47,49,51}
GenCo-2	34	20	{22,24,26,28,30,32,34,36,38,40,42,44,46,48,50,52}
GenCo-5	30	30	{33,35,37,39,41,43,45,47,49,51,53}
GenCo-6	31	10	{14,16,18,20,22,24,26,28,30,32,34,36,38,40,42,44}

Table 3. Big problem demand load data.

4.2. Computational Results

The implementation of the genetic algorithm comes with various parameter values to set. These parameters are the components of the genetic algorithm discussed in Chapter 3. The evaluation of each parameter is crucial to find the optimal or hopefully a good parameter set. Therefore, unlike the traditional algorithm evaluations, we evaluate our algorithm while setting the parameters. In this respect, we define a set of problem-specific performance measures to describe the accuracy of the algorithm.

4.2.1. Performance Measures

The developed genetic algorithm finds "suspicious" solutions instead of exact solutions. Therefore, to measure the performance of the algorithm, we compare our results with exact solutions in Aliabadi *et al.* (2016). In this respect, we utilize two performance measures; ratio of found collusive state and ratio of the collusive state coverage.

The ratio of found collusive state is the number of real collusive states divided by the number of suspicious states found. This performance measure evaluates the precision of our algorithm; "real collusive bid states" demonstrates the true positive detection while "suspicious states found" illustrates the true positive detection and the false positive detection combined.

$$Ratio\ of\ Found\ Collusive\ State = \frac{\#\ of\ Real\ Collusive\ States\ Found}{\#\ of\ Suspicious\ States\ Found}$$

The ratio of collusive state coverage is the number of collusive states found divided by the number of real collusive states. This performance measure evaluates the sensitivity or the hit ratio of the algorithm.

$$Ratio\ of\ Collusive\ State\ Coverage = \frac{\#\ of\ Collusive\ States\ Found}{\#\ of\ Real\ Collusive\ States}$$

4.2.2. Parameter Setting

In order to discuss the accuracy and efficiency of our approach, we need to ensure that parameter values associated with components of the algorithm are set to correct values. We select the small problem to obtain "the best possible parameter settings" since it has the smallest number of possible bid states and the collusive states are known. In each subsection, we discuss the results of the preliminary experiments for one of the parameters that may play a crucial role on the performance of the algorithm.

The algorithm is coded with Python 3.6. Optimization problems are solved by GUROBI 8.0. Computational experiments are conducted on an Intel Core i5 7600k quad-core processor with 3.80 GHz speed and 12 Gb RAM, with 64-bit Windows 10 operating system.

4.2.2.1. Fitness Function

The fitness function is the most crucial parameter in the genetic algorithm. According to our problem setting, there is no obvious function to assume as the fitness function. Therefore, initially, we embed as many components as possible into the potential fitness function. Then, we determine the best fitness function according to the performance of the components. The payoff of each power producer (r_i) is a crucial component that needs to be considered. Since the aim of this study is to detect the collusions, the minimum payoff of all active power companies can give some insight about the collusions. Therefore, we add $minr = \min_i\{r_i\}$ as the possible component of the fitness function.

As we mentioned in previous sections, LMP_i, b_i, and P_i are other components that play an important role in our problem definition. Since if there is collusion among power companies, they intend to increase the unit price of electricity for all companies in the collusion. Therefore, we need to consider $minb = \min_i\{b_i\}$ and $minLMP = \min_i\{LMP_i\}$ as other possible components of the fitness function.

We also take into consideration the value of $P_i(b_i - c_i)$ as a possible component of the fitness function. Hence LMP_i shouldn't be smaller than b_i then $P_i(b_i - c_i)$ forms lower bound to payoffs of power generators. Similar to the discussion of $minr,$ we decide $minPbc =$

$\min_i\{P_i\,(b_i - c_i)\}$ can give insight about collusion and we add this component into fitness function. Then the form of the fitness function is as following,

$$w1 * minr + \ w2 * \ \ minb + w3 * minLMP + w4 * minPbc$$

where $\sum_{i=1}^{4} w_i = 1$. To evaluate the fitness function, we constructed a test parameter set. In this set, we had 50 distinct parameters set option for all possible weight combinations corresponding to different values of the component parameters of genetic algorithm. We run the algorithm with each parameter option and calculate the "ratio of found collusive states" and the "ratio of the collusive state coverage" for each weight combination to determine the performance. The parameter options and results are presented in **Appendix A** and **Appendix B**, respectively. The ratio of found collusive states and ratio of collusive state coverage are maximized with $w_1 = 0, w_2 = 0.3, w_3 = 0$, and $w_4 = 0.7$. Therefore, our fitness function is formed as,

$$Fitness\ Function = \ 0.3 * \ minb + 0.7 * minPbc$$

This fitness function is utilized in the parameter tuning experiments for all parameters.

4.2.2.2. Mutation Rate (Pm)

Possible values for mutation rate are between 0 and 1; we consider increment size of 0.1. For each mutation rate, we run the algorithm 50 times with the parameter values presented in Appendix C. In terms of mutation rates, we conclude that any value in the range of 0.2 to 0.8 could be considered as a good rate. We decide to set the mutation rate as 0.2. The results are presented in Figure 9.

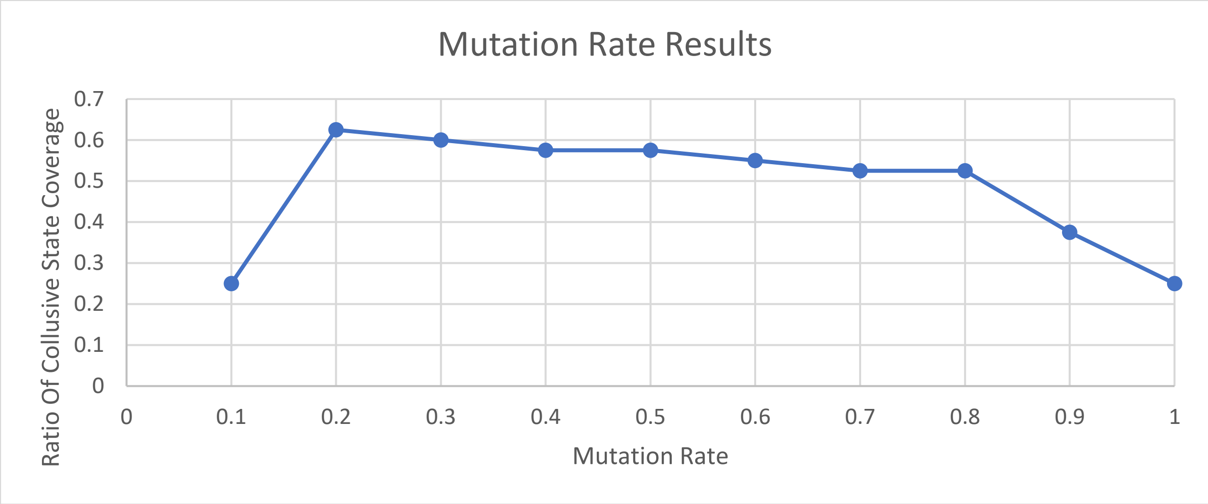

Figure 9. The ratio of collusive state coverages for different mutation rates.

As in the mutation rate analysis, we run the algorithm 50 times with the parameter values presented in Appendix D. As results illustrates in Figure 10, the best-obtained parameter setting for cross-over is 0.4.

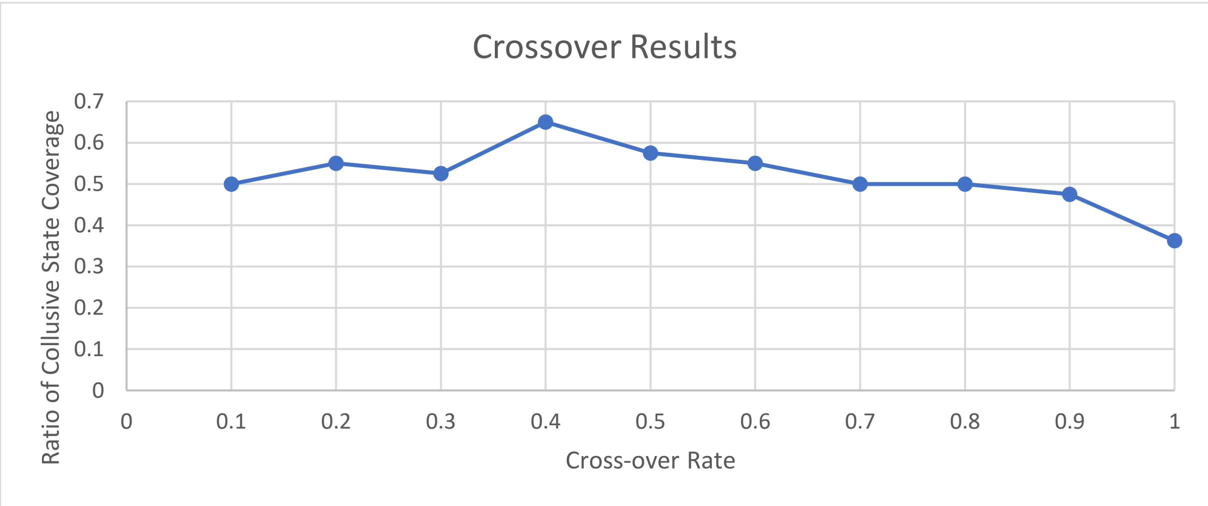

Figure 10. The ratio of collusive state coverages for different crossover rates.

4.2.2.3. Population Size

For this parameter, we set the other parameters as their best-obtained value. Then, we run the algorithm for various values of the size of the population. We found that the collusive state coverage is directly proportionate to population size. The result of population size experiment is presented in Figure 11.

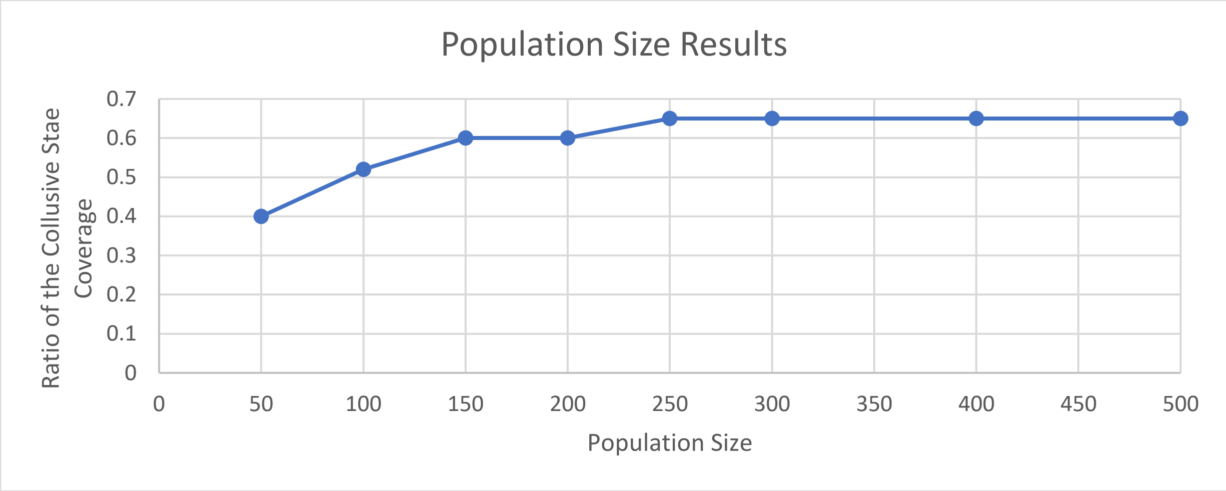

Figure 11. The ratio of collusive state coverage for different population size.

4.2.2.4. Maximum Number of Generations (Epoch)

As in the population size analysis, we found that as we increase the number of the maximum number of generations, the collusive state coverage ratio increases as well. The result of the maximum number of generations is presented in Figure 12.

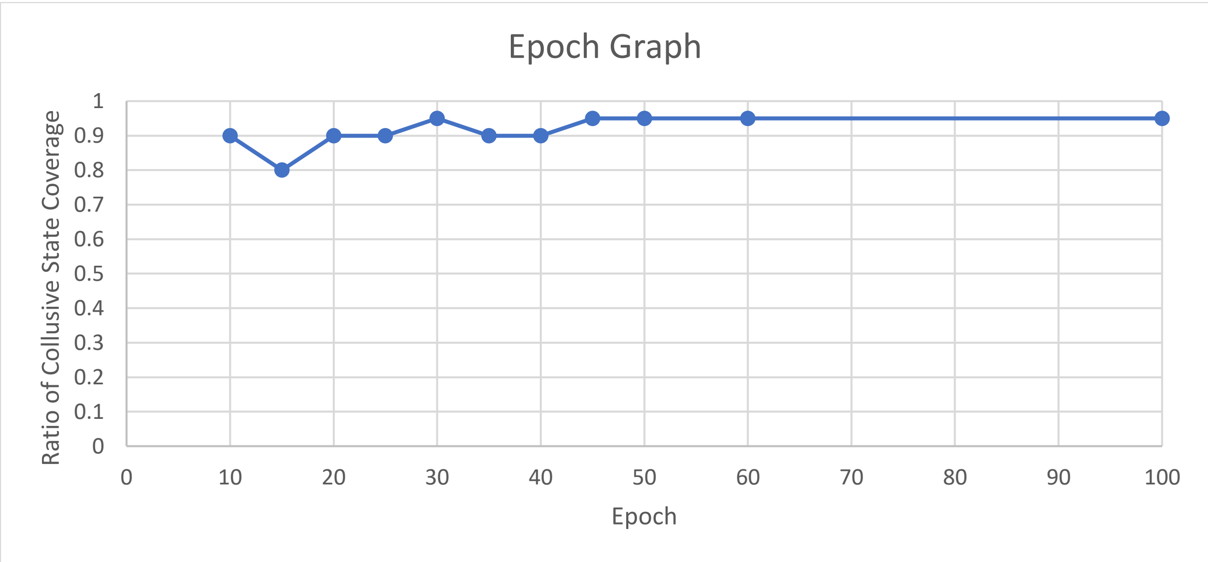

Figure 12. The ratio of collusive state coverage for different maximum number of generations.

4.2.3. Medium Problem

We found the exact solution corresponding to all collusive states for the medium problem with the algorithm in Aliabadi *et al*. (2016). Then, we measure the accuracy of our results according to the exact solutions. The results for the medium size problem for different values of the population size are shown in Figure 13.

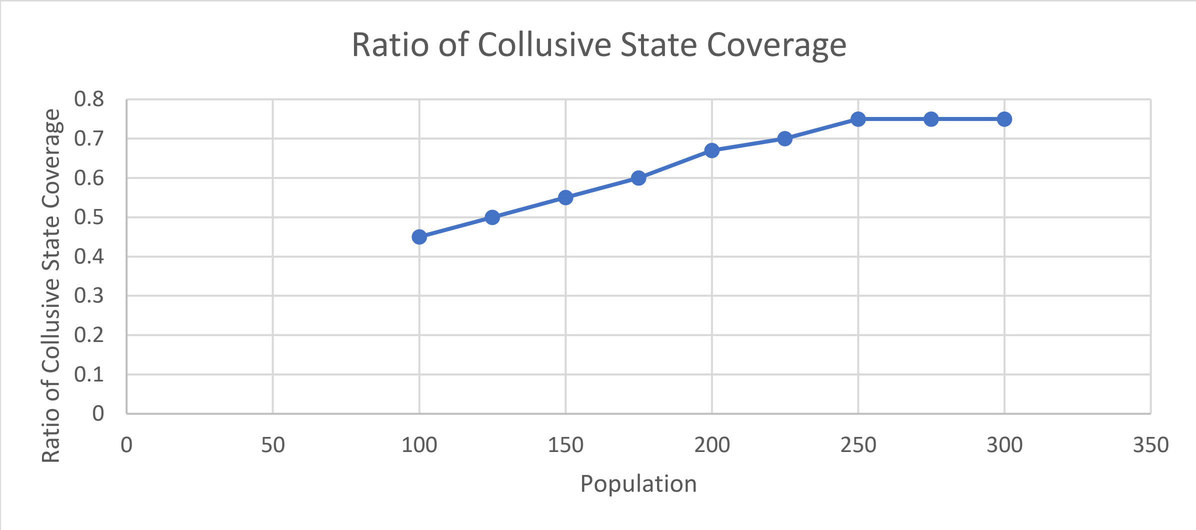

Figure 13. The ratio of collusive state coverage for different maximum number of generations.

4.2.4. Big Problem

We compare our algorithm with the algorithm in Aliabadi *et al.* (2016) in terms of execution time. We conduct experiments on both algorithms with the same computing power as for the small problem and the medium problem. Their algorithm failed to obtain the exact solutions for big problem in more than two weeks since the number of states is too large. However, our algorithm finds the result in less than a hundred seconds. The suspicious bid-offers (b_i) and corresponding payoffs for each power company are shown in Table 4.

Suspicious States ($b_1, b_2, b_3, b_4, b_5, b_6, b_7$)	**Payoffs**
(46, 47, 0, 0, 38, 39, 0)	{52, 81, 0, 0, 300, 930, 0}
(51, 52, 0, 0, 43, 29, 0)	{62, 96, 0, 0, 450, 1085, 0}
(46, 47, 0, 0, 33, 29, 0)	{52, 81, 0, 0, 300, 930, 0}
(51, 52, 0, 0, 38, 34, 0)	{62, 96, 0, 0, 450, 1085, 0}
(41, 42, 0, 0, 33, 29, 0)	{43, 66, 0, 0, 150, 775, 0}
(46, 47, 0, 0, 38, 24, 0)	{52, 81, 0, 0, 300, 930, 0}
(46, 47, 0, 0, 38, 34, 0)	{52, 81, 0, 0, 300, 930, 0}

Table 4. Suspicious states found for big problem.

Chapter 5

5. Conclusion and Future Research

We present a metaheuristic method to detect collusion opportunities in oligopolistic deregulated electricity markets. We created artificial problems representing real-life situations closely to test the performance of our method against the existing ones.

The first problem is used to determine the most promising parameter setting for the algorithm, i.e. the parameter tuning study. This problem was taken from Aliabadi *et. al* (2016). According to the performance evaluations, the most promising values for each parameter were determined. Then using the obtained parameters, we conducted experiments on the other two larger problems. The experiments on the medium and the big problem were utilized to determine the performance of our algorithm in terms of speed and accuracy.

This is the first study to use a heuristic approach to detect collusion in deregulated electricity markets. In the context of heuristic solution, we introduce the notion of "suspicious" collusive states in order to interpret the results obtained with the genetic algorithm. The quality of solutions and performance of the algorithm is measured by the closeness between the set of suspicious states and the actual collusive states as defined by Aliabadi *et al.* (2016). The performance of the search algorithm was found admissibly good according to the computational experiments. Moreover; hence, the algorithm narrows down the large size of the possible bid sets into the small set of suspicious states, this heuristic approach allowed us to solve the problem much faster than the algorithm in Aliabadi *et al.* (2016). Therefore, this approach may guide the decision maker (ISO) to detect the collusive opportunities and to counteract accordingly.

This work can be extended with a utilization of the different methods to determine the best parameter settings for the genetic algorithm. Moreover, we considered simplified power systems in this study. One may want to study this approach on more complex networks or more operational-level problems.

Another future work area might be involved with consideration of the different metaheuristics. Therefore, the obtained results in this study can be compared with other metaheuristics in terms of accuracy and speed to determine the best approach for detecting collusion opportunities in the large-scale problems.

REFERENCES

1. Aliabadi, D. E., Kaya, M., & Şahin, G. (2016). Determining collusion opportunities in deregulated electricity markets. *Electric Power Systems Research,141*, 432-441. doi:10.1016/j.epsr.2016.08.014
2. Barricelli, N. (1954). *Esempi numerici di processi di evoluzione*. Methodos.
3. Barricelli, N. A. (1957). *Symbiogenetic evolution processes realized by artificial methods*. Methodos.
4. Botterud, A., Mahalik, M. R., Veselka, T. D., Ryu, H., & Sohn, K. (2007). Multi-Agent Simulation of Generation Expansion in Electricity Markets. *2007 IEEE Power Engineering Society General Meeting*,1-8. doi:10.1109/pes.2007.385566
5. Bremermann, H. J. (1958). *The evolution of intelligence. The nervous system as a model of its environment* (Tech. No. 1). Seattle, Washington: Dept. Mathematics, Univ. Washington.
6. Crosby, J. L. (1973). *Computer simulation in genetics*. London: Wiley.
7. Fabra, N., & Toro, J. (2005). Price wars and collusion in the Spanish electricity market. *International Journal of Industrial Organization,23*(3-4), 155-181. doi:10.1016/j.ijindorg.2005.01.004
8. Fraser, A., & Burnell, D. (1970). *Computer models in genetics*. New York: McGraw-Hill.
9. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading (Massachusetts): Addison-Wesley Pub.
10. Holland, J. H. (1962). Outline for a Logical Theory of Adaptive Systems. *Journal of the ACM,9*(3), 297-314. doi:10.1145/321127.321128
11. Holland, J. H. (1968). *Hierarchical descriptions of universal spaces and adaptive systems, ORA Projects 01252 and 08226*(Rep.). Michigan.

12. Holland, J. H. (1975). An introductory analysis with applications to biology, control, and artificial intelligence. In *Adaptation in natural and artificial systems*. Michigan: University of Michigan Press.
13. Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing,4*(2). doi:10.1007/bf00175355
14. Liu, A. L., & Hobbs, B. F. (2013). Tacit collusion games in pool-based electricity markets under transmission constraints. *Mathematical Programming,140*(2), 351-379. doi:10.1007/s10107-013-0693-5
15. Nix, A. E., & Vose, M. D. (1992). Modeling genetic algorithms with Markov Chains. *Annals of Mathematics and Artificial Intelligence,5*(1), 77-88. doi:10.1007/BF01530781
16. Ruiz, C., Conejo, A., & Arcos, R. (2010). Some analytical results on conjectural variation models for short-term electricity markets. *IET Generation, Transmission & Distribution,4*(2), 257-267. doi:10.1049/iet-gtd.2009.0049
17. Ruiz, C., Kazempour, S. J., & Conejo, A. J. (2012). Equilibria in futures and spot electricity markets. *Electric Power Systems Research,84*(1), 1-9. doi:10.1016/j.epsr.2011.09.019
18. Schaffer, J. D. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*(pp. 93-100). Hillsdale, NJ: L. Erlbaum Associates.
19. Smith, J., & Fogarty, T. C. (1996). Self adaptation of mutation rates in a steady state genetic algorithm. *Proceedings of IEEE International Conference on Evolutionary Computation,*318-323. doi:10.1109/icec.1996.542382
20. Song, Y., Ni, Y., Wen, F., & Wu, F. F. (2004). Conjectural variation based learning model of strategic bidding in spot market. *International Journal of Electrical Power & Energy Systems,26*(10), 797-804. doi:10.1016/j.ijepes.2004.07.001

21. Sweeting, A. (2007). Market Power in the England and Wales Wholesale Electricity Market 1995–2000. *The Economic Journal,117*(520), 654-685. doi:10.1111/j.1468-0297.2007.02045.x

22. Vose, M. D. (1991). Punctuated Equilibria in Genetic Search. *Complex Systems,5*(1).

23. Wang, J. (2009). Conjectural Variation-Based Bidding Strategies with Q-Learning in Electricity Markets. *2009 42nd Hawaii International Conference on System Sciences*. doi:10.1109/hicss.2009.128

Appendix A: The Test Parameter Set for the Fitness Function

Parameter Set ID	Mutation Rate	Crossover Rate	Population Size	Epoch
1	0.44	0.75	100	30
2	0.68	0.70	300	50
3	0.72	0.74	500	40
4	0.76	0.59	300	20
5	0.63	0.41	400	30
6	0.70	0.39	200	40
7	0.17	0.67	300	60
8	0.01	0.91	100	50
9	0.14	0.06	300	70
10	0.59	0.62	400	90
11	0.93	0.22	500	20
12	0.42	0.19	200	10
13	0.26	0.86	200	30
14	0.98	0.95	300	50
15	0.16	0.12	400	10
16	0.86	0.18	100	60
17	0.23	0.80	500	40
18	0.31	0.08	500	20
19	0.88	0.65	300	30
20	0.79	0.62	100	50
21	0.94	0.66	200	10
22	0.44	0.47	500	20
23	0.37	0.95	400	60
24	0.90	0.38	300	80
25	0.27	0.77	100	90
26	0.38	0.93	200	30
27	0.99	0.11	300	80
28	0.54	0.75	400	10
29	0.46	0.35	300	20
30	0.02	0.82	100	90
31	0.76	0.60	300	30
32	0.20	0.49	100	50
33	0.42	0.60	500	10
34	0.87	0.48	200	50
35	0.80	0.32	400	60
36	0.91	0.42	100	70
37	0.26	0.93	400	20
38	0.38	0.84	500	30
39	0.49	0.95	200	80
40	0.98	0.63	300	30
41	0.24	0.32	100	50

42	0.09	0.97	500	40
43	0.53	0.39	100	20
44	0.15	0.22	400	40
45	0.15	0.80	400	20
46	0.24	0.48	300	90
47	0.29	0.85	200	30
48	0.05	0.77	500	40
49	0.06	0.73	100	50
50	0.46	0.74	200	90

Appendix B: The Test Results for the Fitness Function

w1	w2	w3	w4	Ratio of Found Collusive States	Ratio of Collusive State Coverage
0	0.3	0	0.7	0.552147239	0.77
0	0.5	0	0.5	0.534375	0.76
0	0.6	0	0.4	0.525	0.7
0	0.9	0	0.1	0.535087719	0.677777778
0	0.2	0	0.8	0.41322314	0.555555556
0	0.4	0	0.6	0.49500998	0.551111111
0	0.7	0	0.3	0.591687042	0.537777778
0	0.8	0	0.2	0.404761905	0.453333333
0	0	0	1	0.022891915	0.351111111
0	0	0.1	0.9	0.503311258	0.337777778
0	0	0.6	0.4	0.511864407	0.335555556
0	0.1	0	0.9	0.332594235	0.333333333
0	0.1	0.5	0.4	0.562264151	0.331111111
0	0.7	0.1	0.2	0.42	0.326666667
0	0.7	0.3	0	0.584	0.324444444
0.1	0.4	0.3	0.2	0.478688525	0.324444444
0.2	0	0	0.8	0.475409836	0.322222222
0.2	0	0.2	0.6	0.478405316	0.32
0.2	0.2	0.2	0.4	0.397222222	0.317777778
0.2	0.2	0.4	0.2	0.552123552	0.317777778
0.2	0.4	0.1	0.3	0.546153846	0.315555556
0.2	0.4	0.2	0.2	0.321995465	0.315555556
0.3	0	0.2	0.5	0.557312253	0.313333333
0.3	0.2	0.5	0	0.532075472	0.313333333
0.4	0	0.3	0.3	0.463576159	0.311111111
0.4	0	0.6	0	0.454248366	0.308888889
0.4	0.1	0.3	0.2	0.516981132	0.304444444
0.4	0.2	0.4	0	0.54	0.3
0.5	0	0.3	0.2	0.558333333	0.297777778
0.5	0	0.4	0.1	0.532	0.295555556
0.5	0.1	0	0.4	0.494339623	0.291111111
0.6	0.4	0	0	0.437710438	0.288888889
0.7	0	0	0.3	0.433333333	0.288888889
0	0	0.2	0.8	0.437288136	0.286666667
0	0	0.4	0.6	0.483018868	0.284444444
0	0	0.5	0.5	0.350684932	0.284444444
0	0.1	0.1	0.8	0.484732824	0.282222222
0	0.1	0.6	0.3	0.390625	0.277777778
0	0.1	0.8	0.1	0.625	0.277777778
0	0.2	0.3	0.5	0.482625483	0.277777778
0	0.3	0.1	0.6	0.46969697	0.275555556

0	0.3	0.7	0	0.60591133	0.273333333
0	0.4	0.3	0.3	0.415540541	0.273333333
0	0.4	0.4	0.2	0.495934959	0.271111111
0	0.4	0.6	0	0.463601533	0.268888889
0	0.5	0.5	0	0.5	0.266666667
0	0.6	0.2	0.2	0.53125	0.264444444
0	0.7	0.2	0.1	0.1973466	0.264444444
0	0.8	0.1	0.1	0.292079208	0.262222222
0	0.9	0.1	0	0.229862475	0.26
0	1	0	0	0.291044776	0.26
0.1	0	0	0.9	0.28606357	0.26
0.1	0	0.2	0.7	0.504347826	0.257777778
0.1	0.1	0.3	0.5	0.56	0.248888889
0.1	0.1	0.6	0.2	0.482608696	0.246666667
0.1	0.2	0.5	0.2	0.426923077	0.246666667
0.1	0.2	0.6	0.1	0.313390313	0.244444444
0.1	0.3	0.3	0.3	0.438247012	0.244444444
0.1	0.3	0.6	0	0.5215311	0.242222222
0.1	0.4	0.1	0.4	0.360927152	0.242222222
0.1	0.5	0.3	0.1	0.355263158	0.24
0.1	0.7	0.2	0	0.213572854	0.237777778
0.1	0.8	0.1	0	0.177152318	0.237777778
0.2	0	0.8	0	0.347402597	0.237777778
0.2	0.1	0.3	0.4	0.351973684	0.237777778
0.2	0.1	0.6	0.1	0.263681592	0.235555556
0.2	0.1	0.7	0	0.339805825	0.233333333
0.2	0.2	0.3	0.3	0.342105263	0.231111111
0.2	0.3	0.4	0.1	0.341059603	0.228888889
0.2	0.3	0.5	0	0.443478261	0.226666667
0.2	0.4	0.3	0.1	0.280555556	0.224444444
0.2	0.5	0.1	0.2	0.259067358	0.222222222
0.2	0.6	0.1	0.1	0.220750552	0.222222222
0.2	0.7	0.1	0	0.276243094	0.222222222
0.2	0.8	0	0	0.31152648	0.222222222
0.3	0	0	0.7	0.153609831	0.222222222
0.3	0.1	0	0.6	0.160392799	0.217777778
0.3	0.1	0.2	0.4	0.153724247	0.215555556
0.3	0.1	0.3	0.3	0.389344262	0.211111111
0.3	0.1	0.5	0.1	0.143730887	0.208888889
0.3	0.1	0.6	0	0.366141732	0.206666667
0.3	0.2	0.1	0.4	0.465	0.206666667
0.3	0.2	0.2	0.3	0.62	0.206666667
0.3	0.3	0.2	0.2	0.455	0.202222222
0.3	0.4	0.1	0.2	0.347328244	0.202222222
0.3	0.4	0.2	0.1	0.182186235	0.2

0.4	0	0.1	0.5	0.245856354	0.197777778
0.4	0	0.4	0.2	0.586666667	0.195555556
0.4	0.1	0.1	0.4	0.564102564	0.195555556
0.4	0.1	0.2	0.3	0.286666667	0.191111111
0.4	0.2	0.1	0.3	0.419512195	0.191111111
0.4	0.2	0.2	0.2	0.416666667	0.188888889
0.4	0.2	0.3	0.1	0.425	0.188888889
0.4	0.3	0.2	0.1	0.418367347	0.182222222
0.4	0.4	0.1	0.1	0.422680412	0.182222222
0.4	0.4	0.2	0	0.26557377	0.18
0.5	0	0.5	0	0.397058824	0.18
0.5	0.1	0.2	0.2	0.5	0.177777778
0.5	0.2	0	0.3	0.564285714	0.175555556
0.5	0.2	0.1	0.2	0.545454545	0.173333333
0.5	0.2	0.3	0	0.382352941	0.173333333
0.5	0.3	0	0.2	0.37254902	0.168888889
0.6	0	0	0.4	0.345454545	0.168888889
0.6	0.1	0	0.3	0.5	0.166666667
0.6	0.1	0.2	0.1	0.493333333	0.164444444
0.6	0.2	0	0.2	0.365	0.162222222
0.6	0.3	0	0.1	0.48	0.16
0.8	0	0	0.2	0.473333333	0.157777778
0.8	0	0.1	0.1	0.466666667	0.155555556
0.8	0	0.2	0	0.34	0.151111111
0.8	0.1	0.1	0	0.453333333	0.151111111
0.8	0.2	0	0	0.453333333	0.151111111
0	0	0.7	0.3	0.255725191	0.148888889
0	0	0.8	0.2	0.253112033	0.135555556
0	0	0.9	0.1	0.406666667	0.135555556
0	0.1	0.3	0.6	0.393333333	0.131111111
0	0.1	0.4	0.5	0.55	0.122222222
0	0.1	0.9	0	0.2125	0.113333333
0	0.2	0.1	0.7	0.121428571	0.113333333
0	0.2	0.2	0.6	0.414634146	0.113333333
0	0.2	0.4	0.4	0.099609375	0.113333333
0	0.2	0.7	0.1	0.09922179	0.113333333
0	0.3	0.3	0.4	0.212765957	0.111111111
0	0.3	0.4	0.3	0.25	0.111111111
0	0.3	0.5	0.2	0.298013245	0.1
0	0.3	0.6	0.1	0.325925926	0.097777778
0	0.4	0.2	0.4	0.162878788	0.095555556
0	0.4	0.5	0.1	0.055555556	0.093333333
0	0.5	0.1	0.4	0.048406139	0.091111111
0	0.5	0.2	0.3	0.238095238	0.088888889
0	0.5	0.3	0.2	0.045086705	0.086666667

0	0.5	0.4	0.1	0.045130641	0.084444444
0	0.6	0.1	0.3	0.016135881	0.084444444
0.1	0	0.1	0.8	0.066793893	0.077777778
0.1	0	0.6	0.3	0.142276423	0.077777778
0.1	0	0.7	0.2	0.00808946	0.075555556
0.1	0	0.9	0	0.22962963	0.068888889
0.1	0.1	0	0.8	0.055045872	0.066666667
0.1	0.1	0.1	0.7	0.220588235	0.066666667
0.1	0.1	0.4	0.4	0.183006536	0.062222222
0.1	0.1	0.5	0.3	0.01752109	0.06
0.1	0.1	0.7	0.1	0.211382114	0.057777778
0.1	0.1	0.8	0	0.26	0.057777778
0.1	0.2	0.1	0.6	0.103305785	0.055555556
0.1	0.2	0.2	0.5	0.074766355	0.053333333
0.1	0.2	0.3	0.4	0.195121951	0.053333333
0.1	0.2	0.4	0.3	0.043071161	0.051111111
0.1	0.3	0.2	0.4	0.22	0.048888889
0.1	0.3	0.4	0.2	0.42	0.046666667
0.1	0.3	0.5	0.1	0.034246575	0.044444444
0.1	0.4	0	0.5	0.033500838	0.044444444
0.1	0.4	0.2	0.3	0.38	0.042222222
0.1	0.4	0.5	0	0.038356164	0.031111111
0.1	0.5	0	0.4	0.029598309	0.031111111
0.1	0.5	0.1	0.3	0.028508772	0.028888889
0.1	0.5	0.2	0.2	0.015625	0.026666667
0.1	0.5	0.4	0	0.11	0.024444444
0.1	0.6	0	0.3	0.043103448	0.022222222
0.1	0.6	0.1	0.2	0.086206897	0.022222222
0.1	0.6	0.2	0.1	0.031578947	0.02
0.1	0.6	0.3	0	0.050314465	0.017777778
0.1	0.9	0	0	0.028985507	0.017777778
0.2	0	0.1	0.7	0.077669903	0.017777778
0.2	0	0.3	0.5	0.057553957	0.017777778
0.2	0	0.4	0.4	0.064220183	0.015555556
0.2	0	0.6	0.2	0.033653846	0.015555556
0.2	0.1	0.1	0.6	0.026217228	0.015555556
0.2	0.1	0.4	0.3	0.044025157	0.015555556
0.2	0.1	0.5	0.2	0.030456853	0.013333333
0.2	0.2	0	0.6	0.025773196	0.011111111
0.2	0.2	0.5	0.1	0.015384615	0.008888889
0.2	0.2	0.6	0	0.027027027	0.006666667
0.2	0.3	0	0.5	0.012552301	0.006666667
0.2	0.3	0.2	0.3	0.012096774	0.006666667
0.2	0.4	0	0.4	0.007692308	0.004444444
0.2	0.5	0	0.3	0.007142857	0.004444444

0.2	0.5	0.2	0.1	0.01980198	0.004444444
0.2	0.5	0.3	0	0.003472222	0.002222222
0.2	0.6	0	0.2	0.006711409	0.002222222
0.2	0.6	0.2	0	0.009259259	0.002222222
0.2	0.7	0	0.1	0.009174312	0.002222222
0.3	0	0.1	0.6	0.007092199	0.002222222
0.3	0	0.4	0.3	0	0
0.3	0	0.5	0.2	0	0
0.3	0	0.6	0.1	0	0
0.3	0	0.7	0	0	0
0.3	0.2	0	0.5	0	0
0.3	0.2	0.4	0.1	0	0
0.3	0.3	0.1	0.3	0	0
0.3	0.3	0.3	0.1		0
0.3	0.3	0.4	0	0	0
0.3	0.4	0	0.3	0	0
0.3	0.4	0.3	0	0	0
0.3	0.5	0	0.2	0	0
0.3	0.5	0.1	0.1	0	0
0.3	0.5	0.2	0	0	0
0.3	0.6	0	0.1	0	0
0.3	0.6	0.1	0	0	0
0.4	0	0.2	0.4	0	0
0.4	0	0.5	0.1	0	0
0.4	0.1	0	0.5	0	0
0.4	0.1	0.4	0.1	0	0
0.4	0.1	0.5	0	0	0
0.4	0.3	0.1	0.2	0	0
0.4	0.4	0	0.2	0	0
0.4	0.5	0	0.1	0	0
0.4	0.6	0	0	0	0
0.5	0	0	0.5	0	0
0.5	0	0.2	0.3	0	0
0.5	0.1	0.1	0.3	0	0
0.5	0.1	0.3	0.1	0	0
0.5	0.1	0.4	0	0	0
0.5	0.2	0.2	0.1	0	0
0.5	0.3	0.1	0.1	0	0
0.5	0.3	0.2	0	0	0
0.5	0.4	0	0.1	0	0
0.5	0.5	0	0	0	0
0.6	0	0.1	0.3	0	0
0.6	0	0.3	0.1	0	0
0.6	0.2	0.1	0.1	0	0
0.6	0.2	0.2	0	0	0

0.7	0	0.1	0.2	0	0
0.7	0	0.2	0.1	0	0
0.7	0.1	0.1	0.1	0	0
0.7	0.1	0.2	0	0	0
0.7	0.2	0	0.1	0	0
0.7	0.2	0.1	0	0	0
0.7	0.3	0	0	0	0
0.8	0.1	0	0.1	0	0
0.9	0	0.1	0	0	0
0.9	0.1	0	0	0	0
0	0	0.3	0.7	0	0
0	0	1	0	0	0
0	0.1	0.2	0.7	0	0
0	0.1	0.7	0.2	0	0
0	0.2	0.5	0.3	0	0
0	0.2	0.6	0.2	0	0
0	0.2	0.8	0	0	0
0	0.3	0.2	0.5	0	0
0	0.4	0.1	0.5	0	0
0	0.6	0.3	0.1	0	0
0	0.6	0.4	0	0	0
0	0.8	0.2	0	0	0
0.1	0	0.3	0.6	0	0
0.1	0	0.4	0.5	0	0
0.1	0	0.5	0.4	0	0
0.1	0	0.8	0.1	0	0
0.1	0.1	0.2	0.6	0	0
0.1	0.2	0	0.7	0	0
0.1	0.2	0.7	0	0	0
0.1	0.3	0	0.6	0	0
0.1	0.3	0.1	0.5	0	0
0.1	0.4	0.4	0.1	0	0
0.1	0.7	0	0.2	0	0
0.1	0.7	0.1	0.1	0	0
0.1	0.8	0	0.1	0	0
0.2	0	0.5	0.3	0	0
0.2	0	0.7	0.1	0	0
0.2	0.1	0	0.7	0	0
0.2	0.1	0.2	0.5	0	0
0.2	0.2	0.1	0.5	0	0
0.2	0.3	0.1	0.4	0	0
0.2	0.3	0.3	0.2	0	0
0.2	0.4	0.4	0	0	0
0.3	0	0.3	0.4	0	0
0.3	0.1	0.1	0.5	0	0

0.3	0.1	0.4	0.2	0	0
0.3	0.2	0.3	0.2	0	0
0.3	0.3	0	0.4	0	0
0.3	0.7	0	0	0	0
0.4	0	0	0.6	0	0
0.4	0.2	0	0.4	0	0
0.4	0.3	0	0.3	0	0
0.4	0.3	0.3	0	0	0
0.4	0.5	0.1	0	0	0
0.5	0	0.1	0.4	0	0
0.5	0.4	0.1	0	0	0
0.6	0	0.2	0.2	0	0
0.6	0	0.4	0	0	0
0.6	0.1	0.1	0.2	0	0
0.6	0.1	0.3	0	0	0
0.6	0.3	0.1	0	0	0
0.7	0	0.3	0	0	0
0.7	0.1	0	0.2	0	0
0.9	0	0	0.1	0	0
1	0	0	0	0	0

Appendix C: The Test Parameter Set for the Mutation Operation

Parameter Set ID	Crossover Rate	Population Size	Epoch
1	0.36	200	30
2	0.92	300	50
3	0.91	300	30
4	0.19	300	30
5	0.31	100	20
6	0.41	300	50
7	0.58	200	40
8	0.48	500	20
9	0.69	200	30
10	0.02	400	10
11	0.56	400	10
12	0.92	100	40
13	0.71	300	30
14	0.86	200	40
15	0.43	400	40
16	0.19	200	20
17	0.3	400	50
18	0.92	500	10
19	0.67	400	30
20	0.58	500	10
21	0.27	200	10
22	0.8	100	40
23	0.26	300	40
24	0.65	200	20
25	0.81	200	30
26	0.95	100	10
27	0.51	200	50
28	0.07	500	40
29	0.79	100	20
30	0.04	300	10
31	0.09	200	40
32	0.82	200	10
33	0.35	100	30
34	0.97	400	30
35	0.01	400	10
36	0.61	300	30
37	0.75	500	40
38	0.51	200	50
39	0.05	400	10
40	0.71	300	20
41	0.19	200	30

42	0	500	10
43	0.92	300	10
44	0	100	50
45	0.74	100	50
46	0.66	100	50
47	0.13	100	50
48	0.12	500	50
49	0.33	400	50
50	0.81	400	50

Appendix D: The Test Parameter Set for the Crossover Operation

Parameter Set ID	Mutation Rate	Population Size	Epoch
1	0.82	400	20
2	0.33	500	50
3	0.73	400	30
4	0.38	400	40
5	0.79	300	40
6	0.43	500	30
7	0.53	500	10
8	0.96	300	40
9	0.70	500	10
10	1.00	200	40
11	0.49	400	30
12	0.03	300	10
13	0.63	100	30
14	0.60	500	10
15	0.26	400	50
16	0.42	400	10
17	0.50	300	10
18	0.38	500	50
19	0.06	100	50
20	0.26	200	30
21	0.95	500	30
22	0.76	400	50
23	0.72	300	40
24	0.82	100	10
25	0.37	300	20
26	0.37	500	30
27	0.43	400	40
28	0.74	200	20
29	0.23	500	10
30	0.90	400	50
31	0.98	100	20
32	0.74	300	30
33	0.47	500	20
34	0.60	100	30
35	0.73	300	20
36	0.32	100	30
37	0.99	400	40
38	0.99	400	40
39	0.15	300	10
40	0.60	200	20

41	0.46	300	20
42	0.13	500	30
43	0.51	100	40
44	0.13	500	30
45	0.49	300	40
46	0.11	200	20
47	0.75	300	30
48	0.41	300	30
49	0.97	200	20
50	0.17	200	30