APPENDIX 1

BIOGEOGRAPHY-BASED OPTIMIZATION

A1.1 GENERAL

Heuristic optimization is a new approach which can be used to solve complex problems. It can overcome many shortcomings of more traditional methods. The research area of heuristic optimization algorithms has been attracting researchers for the last 50 years, and numerous algorithms have been published. Some of them, such as genetic algorithms (GAs) and evolutionary strategy (ES) have been used to solve many problems, which are very difficult to solve using traditional optimization algorithms. The importance of heuristic algorithms is generally recognized by the engineering research community. More and more researchers choose heuristic algorithms for different kinds of hard-to-solve problems. Recently Dan Simon suggested Biogeography-based optimization (BBO), which is based on the science of biogeography, for solving complex optimization problems (Darwin 1995; Wallace 2005).

A1.2 BIOGEOGRAPHY

Biogeography is the study of the distribution of animals and plants over time and space. Its aim is to elucidate the reason of the changing distribution of all species in different environments over time. As early as the 19^{th} century, biogeography was first studied by (Wallace 2005; Darwin 1995). After that, more and more researchers began to pay attention on this area. Mathematical equations that govern the distribution of organisms were first discovered and developed during the 1960s. Mathematical models of biogeography describe how species migrate from one island to another, how new species arise, and how species become extinct. Geographical areas that are well suited as residences for biological species are said to have a high habitat suitability index (*HSI*). Features that correlate with HSI include such factors as rainfall, diversity of vegetation, diversity of topographic features, land area, and temperature.

Habitats with a high *HSI* tend to have a large number of species, while those with a low *HSI* have a small number of species. Habitats with a high *HSI* have many species that emigrate to nearby habitats, simply by virtue of the large number of species that they host. Habitats with a high *HSI* have a low species immigration rate because they are already nearly saturated with species. Therefore, high *HSI* habitats are more static in their species distribution than low *HSI* habitats. By the same token, high *HSI* habitats have a high emigration rate; the large number of species on high *HSI* islands has many opportunities to emigrate to neighboring habitats. This does not mean that an emigrating species completely disappears from its home habitat; only a few representatives emigrate, so an emigrating species remains extant in its home habitat, while at the same time migrating to a neighboring habitat. Habitats with a low *HSI* have a high species immigration rate because of their sparse populations. This immigration of new species to low *HSI* habitats may rise the *HSI* of the habitat, because the suitability of a habitat is proportional to its biological diversity. However if a habitat's *HSI* remains low, then the species that reside there will tend to go extinct, which will further open the way for additional immigration. Due to this, low *HSI* habitats are more dynamic in their species distribution than high *HSI* habitats.

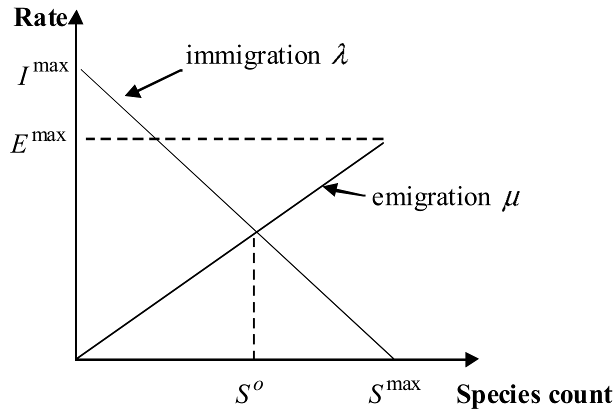

Figure A1.1 Species model of single habitat

The immigration and emigration process helps the species in the area with low HSI to gain good features from the species in the area with high HSI and makes the weak elements into strong. Besides it allows retaining good features of species in the area with high HSI. The rate of immigration (λ) and the emigration (μ) are the functions of the number of species in the habitat. Figure A1.1 shows the immigration and emigration curves indicating the movement of species in a single habitat.

A1.3 BBO

BBO is a bio-inspired stochastic optimization technique for solving multimodal optimization problems (Simon 2008). The environment of BBO corresponds to an archipelago, where every possible solution to the optimization problem is an island. The term “island” here is used descriptively rather than literally. That is, an island is any habitat that is geographically isolated from other habitats. The more generic term “habitat” is therefore used in this thesis rather than the term “island”. The problem variables that characterize habitability are called suitability index variables (SIV). The goodness of each solution is called it’s HSI, where a high HSI of an habitat means good performance on the optimization problem, and a low HSI means deprived performance on the optimization problem. The poor solutions in habitats with low HSI and accept a lot of new features from good solutions in habitats with high HSI and improve their quality. However, the shared features of the good solution still remain in the high HSI solutions. Improving the population is the way to solve problems in heuristic algorithms. SIV can be considered as the independent variables of the habitat, and HSI can be considered the dependent variable.

The method to generate the next generation in BBO is by immigrating solution features to other habitats, and receiving solution features by emigration from other habitats. The immigration and emigration curves shown in Figure A1.1, are straight lines for the case of $E=I$. Habitat with few species (low HSI, poor solution) has low μ and high λ, while habitat with more species (high HSI, good solution) high μ and low λ. Poor solutions accept more useful information from good solution, which improve the exploitation ability of algorithm. The mutation is also performed for the whole population in a manner similar to the mutation in genetic algorithms (GAs).

The concept of immigration and emigration is mathematically represented by a probabilistic model, which relates the probability $P^s(t)$ that a habitat contains exactly S species at time t with that of the probability $P^s(t+\Delta t)$ at time $(t+\Delta t)$, as

$$P^s(t+\Delta t) = P^s(t)\,(1-\lambda_s\,\Delta t - \mu_s\,\Delta t) + P^{s-1}\,\lambda_{s-1}\,\Delta t + P^{s+1}\,\mu_{s+1}\,\Delta t \qquad \text{(A1.1)}$$

If time Δt is so small that the probability of more than one immigration or emigration can be ignored then taking the limit of Equation (A1.1) as $\Delta t \to 0$ gives the following equation

$$\dot{P}^{s} = \begin{cases} -(\lambda_s + \mu_s)P^s + P^{s+1}\,\mu_{s+1} & S = 0 \\ -(\lambda_s + \mu_s)P^s + P^{s+1}\,\mu_{s+1} + P_{s-1}\,\lambda_{s-1} & 1 \le S \le S^{\max} \\ -(\lambda_s + \mu_s)P^s + P^{s-1}\,\lambda_{s-1} & S = S^{\max} \end{cases} \quad \text{(A1.2)}$$

The equation for emigration rate μ_k and immigration rate λ_k for k -number of species is developed from Figure. A1.1 as

$$\mu_k = \frac{E^{\max}}{n} \quad \text{(A1.3)}$$

$$\lambda_k = I^{\max}\left(1 - \frac{k}{n}\right) \quad \text{(A1.4)}$$

when $E^{\max} = I^{\max}$, the immigration and emigration rates can be related as

$$\lambda_k + \mu_k = E^{\max} \quad \text{(A1.5)}$$

The process of BBO is based on the mechanisms of migration and mutation as discussed below.

A1.3.1 Migration

A population of candidate solutions can be represented as vectors of real numbers in BBO algorithm. Each real number in the array is considered as a SIV, which is then used to evaluate the fitness of each candidate solution, denoted by HSI. High HSI represents a better quality solution and low HSI denotes an inferior solution. The emigration and immigration rates of each solution are probabilistically used to control the sharing of features between habitats through a habitat modification probability, P^{mod}. If a given solution S_i is selected for modification, then its λ is used to probabilistically decide whether or not to modify each SIV in that solution. After selecting the *SIV* for modification, μ of other solutions are used to select which of the solutions among the habitat set will migrate randomly chosen *SIVs* to the selected solution S_i. Some kind of elitism, which retains the best habitat having highest *HSI* without performing migration operation, is used in order to prevent the best solutions from being corrupted. The migration operation is explained below:

Select H_i with probability $\alpha\ \lambda_i$

if H_i is selected

for j = 1 to P

Select H_j with probability $\alpha\ \mu_i$

if H_j is selected

Randomly select an *SIV* σ from H_j

Replace a random SIV in H_i with σ

end if

end for

end if

A1.3.2 Mutation

The cataclysmic events that drastically change the *HSI* of a habitat is represented by mutation of *SIV* and species count probabilities are used to determine mutation rates. The probability of each species count, P^s given by

Equation (A1.2), indicates the likelihood that it exists as a solution for a given problem. If the probability of a given solution is very low, then that solution is likely to mutate to some other solution. Similarly if the probability of some solution is high, then that solution has very little chance to mutate. So, it can be said that very high *HSI* solutions and very low *HSI* solutions have less chance to create more improved *SIV* in the later stage. But the medium *HSI* solutions have better chance to create much better solutions after mutation operation. Mutation rate of each set of solution can be calculated in terms of species count probability using the following equation:

$$m(S) = m^{\max}\left(\frac{1 - P^{s}}{P^{\max}}\right) \tag{A1.6}$$

In BBO, the mutation is used to increase the diversity of the population to get good solutions. Mutation operator modifies a habitat's SIV randomly based on mutation rate *m* for the case of E=I. The mutation operation is described as follows:

for j = 1 to N

Use λ_i and μi to compute the probability P_i

Select *SIV* $H_i(j)$ with probability α P_i

If $H_i(j)$ is selected

Replace $H_i(j)$ with a randomly generated *SIV*

end if

end for

A1.3.3 Algorithm

The basic procedure of BBO is as follows:

1. Define the habitat modification probability, mutation probability, and elitism parameter. Habitat modification

probability is similar to crossover probability in GA. Mutation probability and elitism parameters are the same as in GA.

2. Initialize the population (n habitats).

3. Calculate the immigration rate and emigration rate for each habitat. Good solutions have high emigration rates and low immigration rates. Bad solutions have low emigration rates and high immigration rates. Probabilistically choose the immigrating habitats based on the immigration rates.

4. Use roulette wheel selection based on the emigration rates to select the emigrating habitats.

5. Migrate randomly selected SIVs based on the selected habitats in the previous step.

6. Probabilistically perform mutation based on the mutation probability for each habitat.

7. Calculate the fitness of each individual habitat.

8. If the termination criterion is not met, go to step 3; otherwise, terminate.

A1.4 COMPARISON WITH OTHER EVOLUTIONARY ALGORITHMS

Although BBO is a population-based optimization algorithm it does not involve reproduction or the generation of “children”. This clearly distinguishes it from reproductive strategies such as GAs and evolutionary strategies. BBO also clearly differs from ACO, because ACO generates new set of solutions in all iteration. BBO, on the other hand, maintains its set of solutions from one iteration to the next, relying on migration to probabilistically adapt those solutions. BBO has the most common things with strategies such as PSO and DE. In those approaches, solutions are maintained from iteration to iteration, but each solution is able to learn from its neighbors and adapt itself as the algorithm progresses. PSO represents each solution as a point in space, and represents the change over time of each solution as a velocity vector. However, PSO solutions do not change directly; it is rather their *velocities* that change, and this indirectly results in position (solution) changes. DE changes its solutions directly, but changes in a particular DE solution are based on differences between other DE solutions. Also, DE is not biologically motivated. BBO can be contrasted with PSO and DE in that BBO solutions are changed directly via migration from other solutions (habitats). That is, BBO solutions directly share their attributes (SIVs) with other solutions. It is these differences between BBO and other population- based optimization methods that may prove to be its strength.

A1.5 SUMMARY

A brief introduction about the history and the theory of BBO has been overviewed in this chapter. The migration and mutation operations along with the simplified solution procedure have been described. The technique has also been compared with other evolutionary algorithms.

APPENDIX 2

PARTICLE SWARM OPTIMIZATION

A2.1 GENERAL

PSO is a population based probabilistic mechanism designed using swarm-intelligence. Swarm intelligence, also referred to as collective intelligence, is twined on socio-psychological principles and offers insights into social behaviour, in addition to contributing to engineering applications. It describes a coordinated behaviour of decentralized self-organised systems (Yu et al 2007; Amjady and Rezai 2010).

PSO belongs to the class of direct search methods, which are usually derivative-free, meaning that they depend only on the evaluation of the objective function. It is a flexible and well-balanced mechanism for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Hypothesis are plotted in this space and seeded with an initial velocity as well as a communication channel between particles. The particles thus move through the solution space and are evaluated according to some fitness criterion after each time step. The particles are accelerated over time towards those particles within their group, which generate fitness values. The advantage of such an approach over other global optimization strategies is that large numbers of members that make up the particle swarm facilitate the technique to be resilient to the problem of local minima.

Population of m-individuals called particles $X(t)$, is assumed as a candidate solution and is initialised with random guesses in the problem space.

The particles are allowed to fly around in a multidimensional search space with a velocity, $V(t)$ and communicate between two good positions, depending on the values of the objective function and adjust their own positions and velocity based on these good positions. The first is the best position it encompasses so far, called particle best, $X^{*}(t)$, while the second best value achieved so far by any particle, called global best, $X^{**}(t)$, resembles to publicized knowledge. These variables are defined as

Particle $X_j(t)$: It is a candidate solution of j-th particle represented by an m-dimensional vector. Where m is the no. of control variables.

Population $pop(t)$: It is a set of n-particles at instant-t

$$pop(t) = X(t) == \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \\ \vdots \\ X_n(t) \end{bmatrix} = \begin{bmatrix} x_{11}(t) & x_{12}(t) & \cdots & x_{1m}(t) \\ x_{21}(t) & x_{22}(t) & \cdots & x_{2m}(t) \\ x_{31}(t) & x_{32}(t) & \cdots & x_{3m}(t) \\ \vdots & \vdots & & \vdots \\ x_{n1}(t) & x_{n2}(t) & \cdots & x_{nm}(t) \end{bmatrix} \quad \text{(A2.1)}$$

Particle velocity $V(t)$: It is the velocity of the moving particles.

$$V(t) = \begin{bmatrix} V_1(t) \\ V_2(t) \\ V_3(t) \\ \vdots \\ V_n(t) \end{bmatrix} = \begin{bmatrix} v_{11}(t) & v_{12}(t) & \cdots & v_{1m}(t) \\ v_{21}(t) & v_{22}(t) & \cdots & v_{2m}(t) \\ v_{31}(t) & v_{32}(t) & \cdots & v_{3m}(t) \\ \vdots & \vdots & & \vdots \\ v_{n1}(t) & v_{n2}(t) & \cdots & v_{nm}(t) \end{bmatrix} \quad \text{(A2.2)}$$

Individual Best $X^*(t)$: It is the best position, each particle achieves so far at any instant- t

$$X^*(t)=\begin{bmatrix} X_1^*(t) \\ X_2^*(t) \\ X_3^*(t) \\ \vdots \\ X_n^*(t) \end{bmatrix}=\begin{bmatrix} x_{11}^*(t) & x_{12}^*(t) & \cdots & x_{1m}^*(t) \\ x_{21}^*(t) & x_{22}^*(t) & \cdots & x_{2m}^*(t) \\ x_{31}^*(t) & x_{32}^*(t) & \cdots & x_{3m}^*(t) \\ \vdots & \vdots & & \vdots \\ x_{n1}^*(t) & x_{n2}^*(t) & \cdots & x_{nm}^*(t) \end{bmatrix} \tag{A2.3}$$

Global Best $X^{**}(t)$: It is the best position among all individual best positions achieved thus far. i.e., it is the best position; the swarm is seen so far.

$$X^{**}(t)=\left\{x_{k1}^{**}(t) \quad x_{k2}^{**}(t) \quad \cdots \quad x_{km}^{**}(t)\right\} \tag{A2.4}$$

Upon finding the best values, the particles are expected to update their positions and velocities as

$$\begin{aligned} V_j(t) = w(t).V_j(t-1) + & \\ \sigma_1 C_1 \left\{X_j^*(t-1) - X_j(t-1)\right\} + & \qquad j=1,2,\cdots,n \\ \sigma_2 C_2 \left\{X^{**}(t-1) - X_j(t-1)\right\} & \end{aligned} \tag{A2.5}$$

$$X(t)=X(t-1)+V(t) \tag{A2.6}$$

During the iterative process, the inertia constant $w(t)$ is gradually decreased following the relation

$$w(t)=\eta \cdot w(t-1) \tag{A2.7}$$

The iterative procedure of updating the particle positions and velocities in tune with the objective function values is continued until the desired conditions are satisfied. However, the process can be terminated either if there is no appreciable change in the global best solution or after a specific number of iterations.

A2.1.1 Iterative steps

1. Choose suitable values for the following parameters
   - population size, n
   - c_1 & c_2
   - σ1 & σ2
   - $w(0)$
2. Set time counter $t = 0$.
3. <u>Initialisation</u>
   - Generate randomly n-particles $X(0)$. Each $x_{j,k}(0)$ in the population and is generated randomly with uniform probability in the range $[x_k^{\min}, x_k^{\max}]$
   - Generate random initial velocities of all particles $V(0)$. Each $v_{j,k}(0)$ is generated randomly with uniform probability in the range $[-v_k^{\max}, v_k^{\max}]$
   - Evaluate the objective function, Φ for each particle $X_j(0)$
   - Set $X_j^*(0) = X_j(0)$ for each particle- j

- Set $\Phi_j^*(0) = \Phi_j(0)$ for each particle- j

- Search for the best value $\Phi^{**}(0)$ among the objective function values $\Phi_j^*(0)$. Set the particle associated with the best value as $X^{**}(0) = X_j^*(0)$.

4. Update the time counter $t = t + 1$

5. Update the inertia weight $w(t) = \alpha\, w(t-1)$

6. Update the particle velocity $V(t)$. Using the global best, $X^{**}(t-1)$, and the particle best, $X^*(t-1)$, the j^{th} particle velocity is updated by

$$V_j(t) = w(t).V_j(t-1) + c_1\sigma_1\left\{X_j^*(t-1) - X_j(t-1)\right\} + c_2\sigma_2\left\{X^{**}(t-1) - X_j(t-1)\right\}$$

7. Check for velocity limit violation. If there is any limit violation, set the velocity to its respective limit.

8. Update the particle position.

$$X(t) = X(t-1) + V(t)$$

9. Check for particle position limits. If there is any limit violation, set the position to its respective limit.

10. Evaluate the objective function $\Phi_j(t)$ of each particle $X_j(t)$.

11. Update the particle best.

For each particle - j,

If $\Phi_j(t)$ is better than $\Phi_j^*(t-1)$, then

Set $X_j^*(t) = X_j(t)$

Set $\Phi_j^*(t) = \Phi_j(t)$

Else

Set $X_j^*(t) = X_j^*(t-1)$

Set $\Phi_j^*(t) = \Phi_j*(t-1)$

End

12. Update the global best.

Search for the best value among the objective function values $\Phi_j^*(t)$.

If $best\{\Phi^*(t)\}$ is better than $\Phi^{**}(t-1)$, then

Set $\Phi^{**}(t) = best\{\Phi^*(t)\}$

Set the particle associated with the best value as $X^{**}(t) = X_j^*(t)$.

Else

Set $X^{**}(t) = X^{**}(t-1)$

Set $\Phi^{**}(t) = \Phi^{**}(t-1)$

End

13. Check for convergence. i.e., check whether any one of the following criteria is satisfied.

- the number of iterations since the last change of the best solution is greater than a pre specified number or
- Number of iterations reaches the maximum allowable number.

If converged, stop; else, go to step (4)

APPENDIX 3

GENETIC ALGORITHM

A3.1 GENERAL

The GA developed (Holland 1975) is a heuristic search that mimics the process of natural evolution based on the mechanics of natural selection and natural genetics. A population of chromosomes, which encode candidate solutions to an optimization problem, migrates towards better solutions by simulating "*the survival of the fittest*" criterion of Darwinian evolution among chromosome structures. The evolution usually starts from a population of randomly generated individuals and proceeds in generations. In each generation, the fitness of every individual in the population is evaluated, after which multiple individuals are stochastically selected from the current population and modified to a new population. Though the search process is based on randomised information exchange, it efficiently uses historical information to speculate on new search points with the expected improved performance. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations is produced, or a satisfactory fitness level is reached for the population.

GA is inherently a parallel mechanism as it simultaneously evaluates many points in the search space. Besides no restrictions on the solution space are made during the search process. GA is considered to be a robust approach in light of the fact that it is more likely to converge to the global best and carries with it only a remote possibility of settling at local optimum (Goldberg 2000).

A3.1.1 Genetic Representation of Solutions

A set of binary bits, called sub-string, is used to represent each of the decision variables. The number of bits in the sub-string can be chosen arbitrarily subject to a minimum of two bits. However the bit length decides the number of numerical values that are generated within the range of the lower and upper limits of the decision variable. The resolution of the solution depends upon how many bits are used to represent each decision variable. The higher the number of bits used, the finer is the resolution. However, higher the number of encoding bits, slower appears to be the rate of convergence. The bit length thus influences the accuracy and convergence rate of the optimisation problem. For example, a problem with three variables can be encoded by five binary bits as

G_1	G_2	G_3
0 0 1 0 1	1 1 0 0 1	0 1 1 1 1
5 bits	5 bits	5 bits

Figure A3.1 Representation of decision variables

The whole string of binary bits, which represents a single solution point in the search space of an optimisation problem, is called a chromosome. This representation implicitly takes care of the minimum and maximum limits of each variable. Each sub-string is decoded to obtain the actual value using the expression given below:

$$G_i = G_i^{\min} + \frac{G_i^{\max} - G_i^{\min}}{2^L - 1} * D_i \qquad \text{(A3.1)}$$

A3.1.2 Initialisation

A group of chromosomes, called population, is randomly chosen. Each chromosome in the population represents a solution point in the problem search space. The number of chromosomes in the population is known as population size, which depends on the nature of the problem. There is no hard and fast rule in deciding the population size. Initially many individual solutions are heuristically generated to cover the entire search space to form an initial population.

A3.1.3 Fitness Function

The fitness function is defined over the genetic representation and measures the *quality* of the represented solution. The fitness function is always problem dependent and obtained by suitably augmenting the objective function and constraints. It gives a raw measure of the amount of fit of each chromosome in the problem space and contributes to the selection process. The objective function, Φ, is considered as the fitness function, FF, in a maximisation problem while in a minimisation problem, the fitness function is taken as

$$FF = \frac{1}{1+\Phi} \qquad \text{(A3.2)}$$

Among a number of available measures, the most common method of handling explicit constraints is the penalising strategy. It converts the constrained optimisation problem into an unconstrained problem by adding or multiplying a certain penalty to the objective function for any violation of the constraint. These terms reduce the fitness of the chromosome depending on the magnitude of the violation. The main advantage of penalty method compared to other approaches is that it does not disregard infeasible solutions; instead it uses these solutions in such a way as to aid the search process. Sometimes, these infeasible solutions may provide much more useful information about the optimum than the feasible solutions.

A3.1.4 Genetic Operators

There are three prime operators associated with GA. They are Reproduction, Crossover and Mutation. These operators coordinate together to explore and exploit various areas of the search space in order to locate the global optimum.

Reproduction: A proportion of the existing population is selected as parents to breed a new generation. Usually certain chromosomes are reproduced in or copied to a mating pool. Only the individuals that end up in the mating pool stand a chance of becoming parents. Chromosomes are selected through a fitness-based process, where fitter solutions are typically more likely to be selected for the mating pool.

Certain selection methods rate the fitness of each solution and preferentially select the best solutions, while other methods rate only a random sample of the population. Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps to keep the diversity of the population large, preventing premature convergence on poor solutions. Roulette Wheel Selection is often used as it chooses the individuals on the basis of performance with respect to other individuals in the population. Reproduction is simply an operation whereby an old chromosome is copied into a mating pool according to its fitness value.

Cross over: Crossover is the primary genetic operator, which promotes the exploration of new regions in the search space. It is a structured, yet randomised mechanism of exchanging information between chromosomes. This operation is like two scientists exchanging information. Crossover begins first by randomly selecting any two members previously placed in the *mating pool* during reproduction. A crossover point is then selected at random and information up to the crossover point of one parent is exchanged with the other member, thus creating two new members for the next generation. An example of the crossover operation is shown in Figure A3.2.

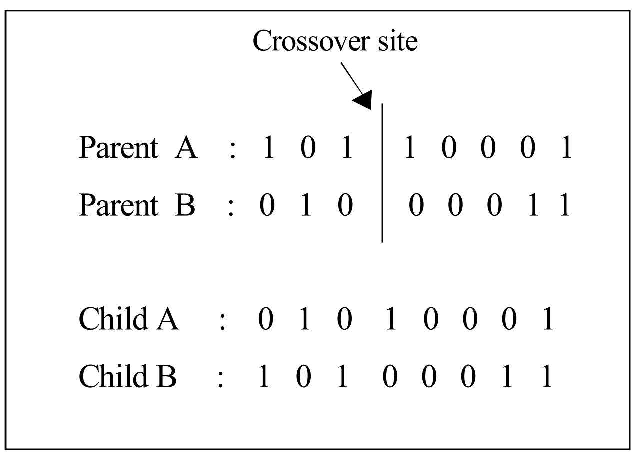

Figure A3.2 Crossover operation

Mutation: Although reproduction and crossover effectively search and recombine the existing chromosomes, they do not create any new genetic material in the population. Mutation is capable of overcoming this shortcoming. It is an occasional (with small probability) random alternation of one or more bits of a chromosome as shown in Figure A3.3. This occasionally introduces beneficial materials, which help to diversify the population and consequently allows the GA to search in new regions of the search space.

Child A	:	0	1	0	1	0	0	0	1
							↓		
New Child A	:	0	1	0	1	0	1	0	1

Figure A3.3 Mutation operation

A3.1.5 Stopping Criteria

The process of generating new chromosomes through reproduction, crossover and mutation are continued until the desired stopping conditions are satisfied. It can be terminated either after a fixed number of generations, or if there is further significant improvement in the global best solution.

A3.1.6 GA Parameters

There are certain parameters that are to be arbitrarily chosen by a trial and error process in such a way to obtain the global solution.

Population size n - It represents the number of chromosomes or the number of solution points considered in one generation.

Probability of Crossover Pc - It controls the frequency with which the crossover operator is applied.

Probability of Mutation Pm - This factor decides the number of mutation process that takes place during generation of the child population.

A3.1.7 Genetic Iteration

The process of generating a parent population through selection and reproduction from an initial population and then generating a child population through crossover and mutation from the parent population may be called as a genetic iteration. The genetic iterations may be continued by taking the child population obtained in the previous iteration as the initial population for the next iteration. Child population of chromosomes that is different from parent population results ultimately in all iteration process. Generally the average fitness increases through this procedure for the population, since only the best organisms are selected for breeding along with a small proportion of less fit solutions. At every generation, the chromosome that acquires the maximum fitness is stored along with its objective function. The chromosome with maximum fitness is taken as the global solution of the given optimisation problem on the completion of a specific number of genetic iterations. The steps explaining the GA iterative process is summarized through the following nine steps.

1. Construct an initial population of chromosome by a random process.
2. Evaluate fitness of each chromosome.
3. Generate mating pool based on fitness function values.
4. Select mating pair of chromosomes, called parent chromosomes, from mating pool.
5. Create two child chromosomes from the parent chromosomes by applying genetic operators.

6. Retrace steps 4 - 5, till the child population of size m is generated.

7. Store the chromosome having the maximum fitness and also the corresponding objective function.

8. Repeat steps 2 - 7, until the specified number of genetic iterations are completed.

9. Return the chromosome with highest fitness function as the solution.

APPENDIX 4

BUS DETAILS

Table A4.1 Bus data of the IEEE 14 bus system

Bus No	VM (per unit)	P_L (MW)	Q_L (MVAR)	P_G (MW)	Q^{min} (MVAR)	Q^{max} (MVAR)	Shunt Capacitor
1	1.06	0.0	0.0	0.0	0.0	0.0	---
2	1.045	21.7	12.7	40.0	-40.0	50.0	---
3	1.010	94.2	19.0	0.0	0	40.0	---
4	---	47.8	3.9	---	---	---	---
5	---	7.6	1.6	---	---	---	---
6	1.070	11.2	7.5	0.0	-6	24.0	---
7	---	0.0	0.0	---	---	---	---
8	1.090	0.0	0.0	0.0	-6	24.0	---
9	---	29.5	16.6	---	---	---	0.190
10	---	9.0	5.8	---	---	---	---
11	---	3.5	1.8	---	---	---	---
12	---	6.1	1.6	---	---	---	---
13	---	13.5	5.8	---	---	---	---
14	---	14.9	5.0	---	---	---	---

Table A4.2 Line data of the IEEE 14 bus system

Between Buses	R_L	X_L	$B_C^{1/2}$	TTS
1-2	0.01938	0.05917	0.02640	---
2-3	0.04699	0.19797	0.02190	---
2-4	0.05811	0.17632	0.01870	---
1-5	0.05403	0.22304	0.02460	---
2-5	0.05695	0.17388	0.01700	---
3-4	0.06701	0.17103	0.01730	---
4-5	0.01335	0.04211	0.0064	---
5-6	0.0	0.25202	0.0	0.932
4-7	0.0	0.20912	0.0	0.978
7-8	0.0	0.17615	0.0	---
4-9	0.0	0.55618	0.0	0.969
7-9	0.0	0.11001	0.0	---
9-10	0.03181	0.08450	0.0	---
6-11	0.09498	0.19890	0.0	---
6-12	0.12291	0.25581	0.0	---
6-13	0.06615	0.13027	0.0	---
9-14	0.012711	0.27038	0.0	---
10-11	0.08205	0.19207	0.0	---
12-13	0.22092	0.19988	0.0	---
13-14	0.17093	0.34802	0.0	---

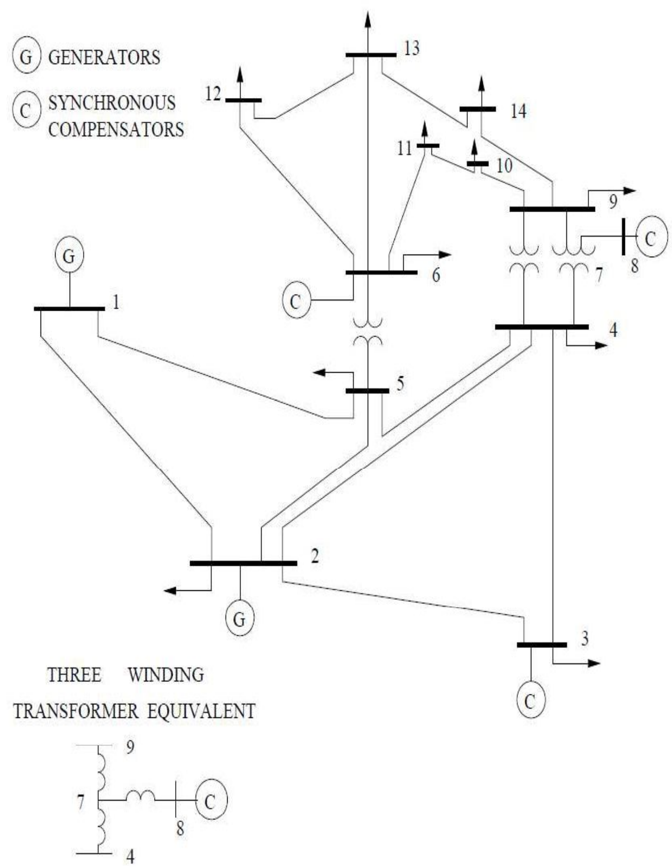

Figure A4.1 Single line diagram of IEEE 14 bus system

Table A4.3 Bus data of the IEEE 30 bus system

Bus No	VM (per unit)	P_L (MW)	Q_L (MVAR)	P_G (MW)	Q^{min} (MVAR)	Q^{max} (MVAR)	Shunt Capacitor
1	1.05	---	---	---	---	---	---
2	1.0338	21.70	12.7	57.56	-20	60	---
3	---	2.4	1.2	---	---	---	---
4	---	7.6	1.6	---	---	---	---
5	1.0058	94.2	19.0	24.56	-15	62.5	---
6	---	0.0	0.0	---	---	---	---
7	---	22.8	10.9	---	---	---	---
8	1.0230	30.0	30.0	35.0	-15	50	---
9	---	0.0	0.0	---	---	---	---
10	---	5.8	2.0	---	---	---	0.19
11	1.0913	0.0	0.0	17.93	-10	40	---
12	---	11.2	7.5	---	---	---	---
13	1.0883	0.0	0.0	16.91	-15	45	---
14	---	6.2	1.6	---	---	---	---
15	---	8.2	2.5	---	---	---	---
16	---	3.5	1.8	---	---	---	---
17	---	9.0	5.8	---	---	---	---
18	---	3.2	0.9	---	---	---	---
19	---	9.5	3.4	---	---	---	---
20	---	2.2	0.7	---	---	---	---
21	---	17.5	11.2	---	---	---	---
22	---	0.0	0.0	---	---	---	---
23	---	3.2	1.6	---	---	---	---
24	---	8.7	6.7	---	---	---	0.04
25	---	0.0	0.0	---	---	---	---
26	---	3.5	2.3	---	---	---	---
27	---	0.0	0.0	---	---	---	---
28	---	0.0	0.0	---	---	---	---
29	---	2.4	0.9	---	---	---	---
30	---	10.6	1.9	---	---	---	---

Table A4.4 Line data of the IEEE 30 bus system

Between Buses	R_L	X_L	$B_C^{1/2}$	TTS
1-2	0.0192	0.0575	0.0264	---
1-3	0.0452	0.1852	0.0204	---
2-4	0.0570	0.1737	0.0184	---
3-4	0.0132	0.0379	0.0042	---
2-5	0.0472	0.1983	0.0209	---
2-6	0.0581	0.1763	0.0187	---
4-6	0.0119	0.0414	0.0045	---
5-7	0.0460	0.1160	0.0102	---
6-7	0.0267	0.0820	0.0085	---
6-8	0.0120	0.0420	0.0045	---
6-9	0.0	0.2080	0.0	1.0155
6-10	0.0	0.5560	0.0	0.9629
9-11	0.0	0.2080	0.0	---
9-10	0.0	0.1100	0.0	---
4-12	0.0	0.2560	0.0	1.0129
12-13	0.0	0.1400	0.0	---
12-14	0.1231	0.2559	0.0	---
12-15	0.0662	0.1304	0.0	---
12-16	0.0945	0.1987	0.0	---
14-15	0.2210	0.1997	0.0	---
16-17	0.0824	0.1923	0.0	---

Table A4.4 (Continued)

Between Buses	R_L	X_L	$B_C^{1/2}$	TTS
15-18	0.1070	0.2185	0.0	---
18-19	0.0639	0.1292	0.0	---
19-20	0.0340	0.0680	0.0	---
10-20	0.0936	0.2090	0.0	---
10-17	0.0324	0.0845	0.0	---
10-21	0.0348	0.0749	0.0	---
10-22	0.0727	0.1499	0.0	---
21-22	0.0116	0.0236	0.0	---
15-23	0.1000	0.2020	0.0	---
22-24	0.1150	0.1790	0.0	---
23-24	0.1320	0.2700	0.0	---
24-25	0.1885	0.3292	0.0	---
25-26	0.2544	0.3800	0.0	---
25-27	0.1093	0.2087	0.0	---
28-27	0.0	0.3960	0.0	0.9581
27-29	0.2198	0.4153	0.0	---
27-30	0.3202	0.6027	0.0	---
29-30	0.2399	0.4533	0.0	---
8-28	0.0636	0.2000	0.0214	---
6-28	0.0169	0.0599	0.0650	---

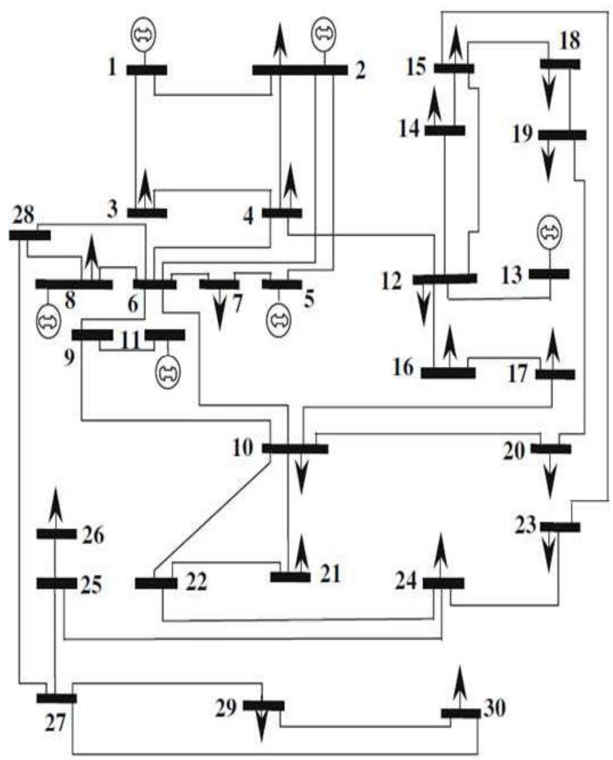

Figure A4.2 Single line diagram of IEEE 30 bus system

Table A4.5 Bus data of the IEEE 57 bus system

Bus No	VM (per unit)	P_L (MW)	Q_L (MVAR)	P_G (MW)	Q^{min} (MVAR)	Q^{max} (MVAR)	Shunt Capacitor
1	1.040	---	---	---	---	---	---
2	1.01	3	88	0	-17	50	---
3	0.985	41	21	40	-10	60	---
4	---	0	0	---	---	---	---
5	---	13	4	---	---	---	---
6	0.98	75	2	0	-8	25	---
7	---	0	0	---	---	---	---
8	1.005	150	22	450	-140	200	---
9	0.98	121	26	0	-3	9	---
10	---	5	2	---	---	---	---
11	---	0	0	---	---	---	---
12	1.015	377	24	310	-50	155	---
13	---	18	2.3000	---	---	---	---
14	---	10.5000	5.3000	---	---	---	---
15	---	22	5	---	---	---	---
16	---	43	3	---	---	---	---
17	---	42	8	---	---	---	---
18	---	27.2000	9.8000	---	---	---	0.10
19	---	3.3000	0.6000	---	---	---	---
20	---	2.3000	1	---	---	---	---
21	---	0	0	---	---	---	---
22	---	0	0	---	---	---	---

Table A4.5 (Continued)

23	---	6.3000	2.1000	---	---	---	---
24	---	0	0	---	---	---	---
25	---	6.3000	3.2000	---	---	---	0.059
26	---	0	0	---	---	---	---
27	---	9.3000	0.5000	---	---	---	---
28	---	4.6000	2.3000	---	---	---	---
29	---	17	2.6000	---	---	---	---
30	---	3.6000	1.8000	---	---	---	---
31	---	5.8000	2.9000	---	---	---	---
32	---	1.6000	0.8000	---	---	---	---
33	---	3.8000	1.9000	---	---	---	---
34	---	0	0	---	---	---	---
35	---	6	3	---	---	---	---
36	---	0	0	---	---	---	---
37	---	0	0	---	---	---	---
38	---	14	7	---	---	---	---
39	---	0	0	---	---	---	---
40	---	0	0	---	---	---	---
41	---	6.3000	3	---	---	---	---
42	---	7.1000	4.000	---	---	---	---
43	---	2	1	---	---	---	---
44	---	12	1.8000	---	---	---	---
45	---	0	0	---	---	---	---
46	---	0	0	---	---	---	---
47	---	29.7000	11.6000	---	---	---	---

Table A4.5 (Continued)

48	---	0	0	---	---	---	---
49	---	18	8.5000	---	---	---	---
50	---	21	10.5000	---	---	---	---
51	---	18	5.3000	---	---	---	---
52	---	4.9000	2.2000	---	---	---	---
53	---	20	10	---	---	---	0.63
54	---	4.1000	1.4000	---	---	---	---
55	---	6.8000	3.4000	---	---	---	---
56	---	7.6000	2.2000	---	---	---	---
57	---	6.7000	2	---	---	---	---

Table A4.6 Line data of the IEEE 57 bus system

Between Buses	R_L	X_L	$B_C^{1/2}$	*TTS*
1-2	0.0083	0.0280	0.0645	---
2-3	0.0298	0.0850	0.0409	---
3-4	0.0112	0.0366	0.0190	---
4-5	0.0625	0.1320	0.0129	---
4-6	0.0430	0.1480	0.0174	---
6-7	0.0200	0.1020	0.0138	---
6-8	0.0339	0.1730	0.0235	---
8-9	0.0099	0.0505	0.0274	---
9-10	0.0369	0.1679	0.0220	---

Table A4.6 (Continued)

9-11	0.0258	0.0848	0.0109	---
9-12	0.0648	0.2950	0.0386	---
9-13	0.0481	0.1580	0.0203	---
13-14	0.0132	0.0434	0.0055	---
13-15	0.0269	0.0869	0.0115	---
1-15	0.0178	0.0910	0.0494	---
1-16	0.0454	0.2060	0.0273	---
1-17	0.0238	0.1080	0.0143	---
3-15	0.0162	0.0530	0.0272	---
4-18	0	0.5550	0	0.9700
4-18	0	0.4300	0	0.9780
5-6	0.0302	0.0641	0.0062	---
7-8	0.0139	0.0712	0.0097	---
10-12	0.0277	0.1262	0.0164	---
11-13	0.0223	0.0732	0.0094	---
12-13	0.0178	0.0580	0.0302	---
12-16	0.0180	0.0813	0.0108	---
12-17	0.0397	0.1790	0.0238	---
14-15	0.0171	0.0547	0.0074	---
18-19	0.4610	0.6850	0	---
19-20	0.2830	0.4340	0	---
20-21	0	0.7767	0	1.0430
21-22	0.0736	0.1170	0	---
22-23	0.0099	0.0152	0	---
23-24	0.1660	0.2560	0.0042	---

Table A4.6 (Continued)

24-25	0	1.1820	0	1.0
24-25	0	1.2300	0	1.0
24-26	0	0.0473	0	1.0430
26-27	0.1650	0.2540	0	---
27-28	0.0618	0.0954	0	---
28-29	0.0418	0.0587	0	---
7-29	0	0.0648	0	0.9670
25-30	0.1350	0.2020	0	---
30-31	0.3260	0.4970	0	---
31-32	0.5070	0.7550	0	---
32-33	0.0392	0.0360	0	---
32-34	0	0.9530	0	0.9750
34-35	0.0520	0.0780	0.0016	---
35-36	0.0430	0.0537	0.0008	---
36-37	0.0290	0.0366	0	---
37-38	0.0651	0.1009	0.0010	---
37-39	0.0239	0.0379	0	---
36-40	0.0300	0.0466	0	---
22-38	0.0192	0.0295	0	---
11-41	0	0.7490	0	0.9550
41-42	0.2070	0.3520	0	---
41-43	0	0.4120	0	---
38-44	0.0289	0.0585	0.0010	---
15-45	0	0.1042	0	0.9550
14-46	0	0.0735	0	0.9000

Table A4.6 (Continued)

46-47	0.0230	0.0680	0.0016	---
47-48	0.0182	0.0233	0	---
48-49	0.0834	0.1290	0.0024	---
49-50	0.0801	0.1280	0	---
50-51	0.1386	0.2200	0	---
10-51	0	0.0712	0	0.9300
13-49	0	0.1910	0	0.8950
29-52	0.1442	0.1870	0	---
52-53	0.0762	0.0984	0	---
53-54	0.1878	0.2320	0	---
54-55	0.1732	0.2265	0	---
11-43	0	0.1530	0	0.9580
44-45	0.0624	0.1242	0.0020	---
40-56	0	1.1950	0	0.9580
56-41	0.5530	0.5490	0	---
56-42	0.2125	0.3540	0	---
39-57	0	1.3550	0	0.9800
57-56	0.1740	0.2600	0	---
38-49	0.1150	0.1770	0.0015	---
38-48	0.0312	0.0482	0	---
9-55	0	0.1205	0	0.9400

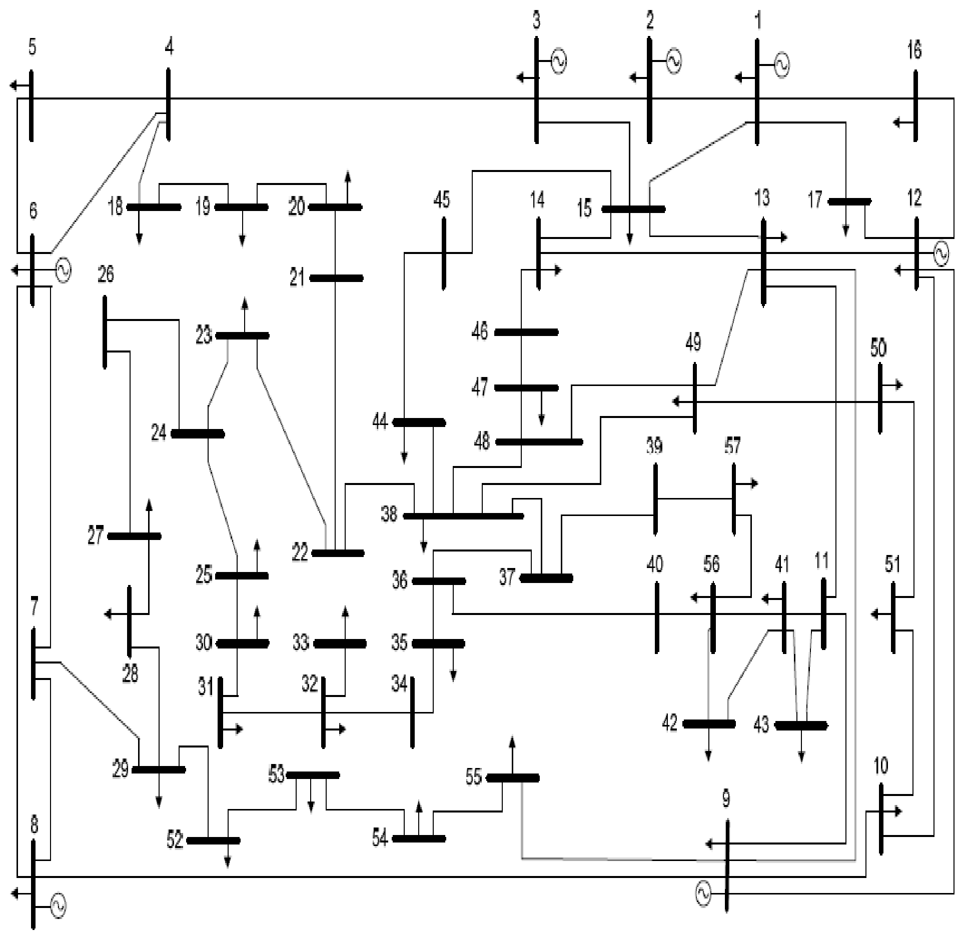

Figure A4.3 one line diagram of IEEE 57 bus system