THE HONG KONG POLYTECHNIC UNIVERSITY

Department of Electrical Engineering

Transient Stability-Constrained Optimal Power Flow Using Improved Differential Evolution and Parallel Computing

CAI HUA RONG

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

December 2007

CERTIFICATE OF ORINGNALITY

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

_____(Signed)

____Cai Huarong____(Name of student)

Abstract

Optimal Power Flow (OPF) is one of the most important tools in power system planning, operation and control. Its purpose is to determine the power system controls to find the delicate balance between economy and security. Due to the rapid increase of electricity demand and the deregulation of electricity markets, power systems tend to operate closer to stability boundaries and, as a consequence, resulting in serious damage to national economics and security. Thus, consideration of the transient stability limits in the OPF problem of power systems is becoming more and more imperative. It is, however, an open question as how to include the stability constraints into OPF since transient stability is a dynamic concept and differential equations are involved. Some conventional optimization methods such as Interior Point Method (IPM) have been attempted to incorporate the transient stability constraints into OPF mainly by approximating the differential equations to algebraic equations. However, conventional mathematical optimization methods are sensitive to the starting points and have convergence difficulties in handling nonlinear, non-convex problems. Besides, the discretizing scheme will lead to computational inaccuracy in solving the problem; and discrete control variable handling such as transformer tap settings is another problem for the conventional methods.

To address the above mentioned problems, this thesis is devoted to the development of an alternative approach in dealing with the OPF problem based on a new evolutionary algorithm Differential Evolution (DE). Each individual in the DE population is a candidate solution for the OPF problem. In particular, an improved version of DE with population re-initialization scheme called RDE is reported to ameliorate the premature problem of DE. Simulations on IEEE 14-, 30-, and 118-bus systems show the powerful ability of RDE in seeking the global optimal solution. As for transient stability constraints, a hybrid method which combines time domain simulation and transient energy function is employed to assess the transient stability of each individual with no limitation in system modeling. Stable individual has more chance to survive in the evolution process in seeking both secure and economic global solution. Since transient stability assessment is the most time-consuming part of the whole method, strategies called “stable-space push” and “fitness sorting” are also developed to reduce the searching space as well as the computation time. Besides the transient stability constraints, other non-convex and discontinuous practical constraints like generator valve-point effects, prohibited operation zones constraints that are difficult to handle by conventional methods are also considered into the OPF problem. The performance of the proposed algorithm has been tested on the WSCC 9-bus and New England 39-bus systems and compared with the reported results by conventional methods. The results show that the method developed in this thesis is very powerful in solving nonlinear, non-convex, discontinuous

complex optimization problem with both continuous and discrete control variables.

A parallel computation platform is also built in this thesis to speed up the proposed method. The parallel computation is implemented on a Beowulf PC-cluster using Message-Passing Interface (MPI) technology. Topologies like Master-Slave, Dual-Ring, and Hybrid Structure of the parallel computation are developed to optimize the computation. Case studies shows that parallelization does significantly improve the speed of DE; it is possible to realize online TSCOPF with moderate scale PC clusters and meet the real-world online application requirement.

The method developed in this thesis is found to be effective and powerful in finding the global solution which is economic and secure for the power system.

Acknowledgements

My precious days in PolyU as a PhD candidate were full of helps and impressions from lots of people. Without their support, dedications in this thesis would have not been possible. I treasure this chance here to express my deep thanks to all of them.

First and foremost, I would first like to express my most sincere gratitude to my two supervisors, Dr. C.Y. Chung and Pro. K. P. Wong. They were always available for me in spite of their busy schedules. Their knowledge and talents, their patience, their encouragement and considerations offer me so much. I have benefited so much from their invaluable guidance, suggestions, and advices.

Also, I would like to give my special thanks to all my friends, old and new. Their hearty supports always come to me without any reserve when needed. They made my life in Hong Kong more wonderful and meaningful.

Last but not the least, I would like to acknowledge the funding support of the Edward Sai Kim Hotung Funds; the helpful assistance from the Research Office in my PhD study years is gratefully appreciated.

Table of Contents

Chapter 1 Introduction

1.1. Background and Motivation

Optimal Power Flow (OPF) was firstly introduced by Carpentiers as a network constrained economic dispatch [Carpentiers, 1962] and formulated by Dommel and Tinny as optimal power flow [Dommel and Tinney, 1968]. The main purpose of OPF is to operate the system at the most economic state while satisfying specified security constraints. OPF has great meaning for power system operation and development especially in the modern society where human beings depend much more on electricity.

Power system security analysis is composed of both static and dynamic security analysis. However, most previous research works concern only about maintaining static security in the OPF problem; few can effectively deal with dynamic security constraints despite the recognition of its great importance.

Due to the rapid increase of electricity demand and the deregulation of electricity markets, power systems tend to operate more closely to stability boundaries and as a consequence, many instability problems occurred in many countries recently, for example, the power system blackout in large portions of the

Midwest and Northeast United States and Ontario, Canada on 14 August 2003 affected about 50 million people [U.S.-Canada Power System Outage Task Force, 2004]. Huge losses and expensive costs in these events give good evidences that dynamic stability under large disturbances is still the most serious threat for the development of modern power systems [Novosel, Begovic, and Madani, 2004]. Among various dynamic security analyses, transient stability is one of the most essential and important assessments. Huge attentions have been paid on power system transient stability analysis by engineers and researchers all these years. Therefore, consideration of transient stability constrained optimal power flow (TSCOPF) problem is becoming more and more imperative.

1.2. Present State of TSCOPF

1.2.1. Formulation of OPF

Currently, basic OPF problem is mathematically described as a nonlinear programming problem which aims to minimize an objective function while satisfying a serial of equality and inequality constraints.

$$\text{Minimize} \qquad f(\boldsymbol{u}, \boldsymbol{x}) \tag{1.1}$$

$$\text{Subject to} \qquad g(\boldsymbol{u}, \boldsymbol{x}) = 0 \tag{1.2}$$

$$h(\boldsymbol{u}, \boldsymbol{x}) \leq 0 \tag{1.3}$$

where $\boldsymbol{u}$ is the vector of control variables of the system, $\boldsymbol{x}$ is the vector of dependent variables of the system. Objective function f may be system

generating fuel cost, available transfer capability etc. Equality and inequality constraints (1.2) and (1.3) are system physical and operational constraints.

1.2.2. Transient Stability Analysis

Power system transient stability phenomena are associated with the operation of synchronous machines in parallel. From a physical viewpoint, transient stability is the ability of the power system to maintain synchronism when subjected to a severe transient disturbance such as a fault on transmission facilities, loss of generation, or loss of a large load. The system response to such disturbances involves large excursion of generator rotor angles, power flows, bus voltages, and other system variables [Pavella, Murthy, 1994]. If the resulting angular separation between the machines in the system remains within certain bounds, the system remains synchronism. From the system theory viewpoint, power system transient stability is a strongly nonlinear, high-dimensional problem. In large power system, transient stability may not always occur as first swing instability associated with a single mode; it could be a result of superposition of a slow inter-area swing mode and a local-plant swing mode causing a large excursion of rotor angle beyond the first swing [Kundur, 1994].

Stability is influenced by the nonlinear characteristics of the power system as well as the initial operating state and the severity of the disturbance. If the resulting angular separation between the machines in the system remains within certain bounds, the system remains synchronous. The period of interest of transient

stability is the transient period before the new steady-state conditions reached. If a system tends to go unstable by the loss of synchronism, relative angular velocities of some machines goes on increasing with respect to the rest of system machines. Loss of synchronism because of transient instability will usually be evident within 2 to 3 seconds after the initial disturbance.

Since transient stability has been in interests for long years, many advanced methods have been developed for its analysis. So far, methods for transient stability analysis can be categorized into three directions: time-domain simulation, direct methods, and hybrid methods.

Time domain simulation solves the differential-algebraic equations (DAEs) describing the dynamic responses of the power system step by step to get trajectories of the state variables such as generator rotor angles or active power outputs. Time domain simulation has been regarded as the best methods in terms of accuracy, reliability and modeling capability. Details of system dynamic responses could be provided even for large-scale power systems. Even very large systems can be analyzed by time-domain simulation with detail components modeling. However, time-domain methods suffer from the drawback that it cannot provide sufficient information about the degree of stability of the system.

Direct methods are originated from the control theory. They determine transient stability without explicitly solving the complex differential-algebraic system dynamic equations. Direct methods are classified into two directions: one is based on the transient energy functions (TEF) methods [Fouad and Vittal, 1992;

Pai, 1989], the other is based on extended equal area criterion (EEAC) methods [Xue, Van Custem, and Ribbens-Pavella, 1989]. The TEF based methods use transient energy functions described by system state variables to present system energy in different stages which is stimulated by the fault and accumulated in the fault-on stage. The research focuses include controlling unstable equilibrium point (CUEP) method, potential energy boundary surface (PEBS) method, and boundary of stability region based controlling unstable equilibrium point (BCU) method. Difficulties of this kind of methods are the definition and calculation of the energy functions. EEAC based methods equivalent the whole system into critical and system group, use the equal accelerating and decelerating area to determine the system stability; whereas, the discrimination of critical unit group is a crux. Compared with time-domain simulation, direct methods are capable of providing a quantitative measurement for the transient stability margin. However, the modeling capabilities of direct methods are limited and the accuracy and reliability are not guaranteed with the theoretic assumptions.

In order to overcome the drawbacks of these two methods, the hybrid methods which incorporate time domain simulation and TEF method have been developed in recent years [Maria and Tang, 1990; Fang and David, 2004]. The restriction on the application of the classical models has been removed and the problem of erratic nonlinearity of the transient energy margin that results in an unreliable prediction of the stability limits has also been overcome. The hybrid method will be used in this paper in the transient stability assessment.

1.2.3. Transient Stability Constrained Optimal Power Flow

Despite the importance of transient stability constrained OPF has been realized for years, few effective methods have yet been proposed till now. The reason lies in the fact that TSCOPF problem itself is a nonlinear, semi-infinite optimization problem with both algebraic and differential equations, which is hard to be solved even for small power systems. The main obstacles tackling this problem exist in: (1) how to incorporate the transient stability constraints into the OPF problem, namely, how to deal with the differential equations that represent the dynamic behavior of the system; and (2) how to deal with TSCOPF effectively and efficiently. Existing attempts on TSCOPF problem mainly go along two directions.

One direction is to solve OPF and the transient stability constraints in sequence separately. This kind of approach is much like a trial-and-error way. First a static OPF solution would be found, then sensitivities of transient stability index, such as energy margin, relative rotor angle, or critical clearing time (CCT), to system parameters, such as generator output or bus voltage, are used to reschedule the power flow of the system thus satisfying the transient stability limitation [Bettiol., Wehenkel, and Pavella, 1999; Nguyen and Pai, 2003; Daniel and Pavella, 2003; Shubhanga and Kulkarni, 2004]. However, instead of searching the solution space globally, only a conservative system configuration could be obtained.

The other direction is to incorporate transient stability constraints into OPF directly and solve it as a single problem. The transient stability constraints are

considered by converting the differential equations into numerical equivalent algebraic equations with specified time steps as inequality constraints such that conventional derivative-based optimization methods can be adopted [Gan, Thomas, and Zimmerman, 2000; Yuan, Kubkawa, and Sasaki, 2003]. The most common used nonlinear programming (NLP) model and interior point method (IPM) are briefly introduced here.

- Nonlinear Programming

Generally, nonlinear programming needs the utility of Langrange Multiplier and Karush-Kuhn-Tucker (KKT) conditions. The typical model of a nonlinear programming problem is described as below:

$$\begin{aligned} &\text{Minimize} && f(\boldsymbol{z}) \\ &\text{Subject to} && g(\boldsymbol{z}) = 0 \\ & && h(\boldsymbol{z}) \le 0 \end{aligned} \tag{1.4}$$

Supposing there exits multiplier vector $\boldsymbol{\xi}$ and $\boldsymbol{\mu}$, the following KKT conditions must be satisfied at the global optimal point of the problem:

$$\begin{cases} \dfrac{\partial f}{\partial \boldsymbol{z}^*} + \dfrac{\partial g^T}{\partial \boldsymbol{z}^*}\boldsymbol{\xi} + \dfrac{\partial f}{\partial \boldsymbol{z}^*}\boldsymbol{\mu} = 0 \\ g(\boldsymbol{z}^*) = 0 \end{cases} \tag{1.5}$$

$$\begin{cases} h(\boldsymbol{z}^*) \le 0 \\ \boldsymbol{\mu}^T h(\boldsymbol{z}^*) = 0 \\ \boldsymbol{\mu} \ge 0 \end{cases} \tag{1.6}$$

When solving nonlinear programming model, the inequality constraints can be converted to equality constraints through methods like relaxation. The converted optimization problem can be solved by algorithms like gradient

method, Newton method, and interior point method. The most up-to-date method is interior point method.

- Interior Point Method

Interior point method (IPM) was first proposed by Karmarkar of the Bell Laboratories in U.S.A. as an algorithm for solving linear programming problems. Instead of following a path on the boundary of the feasible region, IPM follows a path through the interior of the feasible region. So a feasible interior stating point is needed for IPM. This method is not sensitive to the scale of problem; therefore, it is more competitive in optimizing large convex non-linear problems, in particular, so-called geometric programs and semi-definite programs. As has been developed for more than twenty years, there is an enormous variety of branches most of which can be categorized into three classes: project scaling IPM, affine scaling IPM, and primal-dual IPM. So far, IPM has showed great vitality in power system optimization [Quintana, Torres, and Palomo, 2000].

The idea of discretizing differential equations into algebraic equations has been extended to consider multi-contingencies in [Yuan, Kubkawa, and Sasaki, 2003] and post-fault steady-state operating limit in [Dawn and Jeyasurya, 2004]. Reference [Chen, Tada, and Okamoto, etc., 2001] also proposed a functional transformation technique to convert the semi-infinite-dimensional TSCOPF problem to a solvable finite-dimensional programming problem. The transformed problem has the same variables as those in the static OPF problem so that the

efficiency could be improved. Although these works have made valuable contributions and improvement in TSCOPF problem solving, following cruxes are still open to overcome by the derivative-based conventional optimization methods.

1. The use of the discretising scheme may result in not only computation inaccuracy due to the approximation but also dramatic explosion of problem dimension with the introduction of a large number of variables corresponding to each time step which may cause convergence problem.
2. Handling of discrete control variables. Discrete variable may be generally treated as continuous variables and then be rounded off to the nearest discrete value after the optimization. However, when dispersion degree of the discrete variable is high, the objective value of final solution may become much worse or even dependent variables surpass their limitations and the solution becomes an infeasible one. Unfortunately, effective ways for this problem are still not found by derivative-based conventional optimization methods up to now.
3. Sensitive to starting points. Conventional methods search the optimal solution from a certain starting point and follow a path guided by the derivation information. However, OPF is a non-convex problem especially with the introduction of transient stability constraints. Therefore, the searching may converge to a nearest minimum theoretically.
4. Response to infeasible problem. When power system is too tight and in

deficiency of effective regulation measurements, there exits the possibility that no feasible solution could be obtained. It is important for an optimization algorithm not only to find out the global optimum, but also to distinguish the infeasible situation as early as possible and provide a sub-optimal solution. However, this problem is still unresolved by conventional methods.

1.3. Intelligent Evolutionary Algorithm

The derivative-based conventional optimization methods introduced above belongs to the category of classic mathematics. With the mutual osmosis of multi-field from 50s' last century, the modern mathematics has gained developments and improvements gradually. A serial of intelligent optimization algorithms or meta-heuristic algorithms has appeared; most of them were not proposed by mathematicians but scientists and engineers from all walks of industry with heuristics from their experiences [Bäck, Hammel, and Schwefel, 1997].

Evolutionary algorithm (EA) is a typical delegate of intelligent optimization algorithms [Dumitrescu, Lazzerini, and Jain, etc., 2000; Fogel, 2000]. It is a stochastic global search method that mimics the metaphor of natural biological evolution process. Generally, at the beginning of the optimization process, a group of candidate solutions are initialized randomly, each candidate solution is named

as individual, a group of individuals compose of a population, and the number of individuals in a population is called population size. A fitness value related to the optimization objective is assigned to each individual; the higher the fitness value, the better the candidate individuals. Operations such as crossover, recombination, mutation, selection etc. are employed to reproduce the next generation of population which has higher global fitness. These operations are different from specified algorithm. With the evolution of new generations, the global optimum is gradually approximated.

As a universal adaptive algorithm, EA has following remarkable advantages: (1) unlike conventional methods, EA searches the solution space from multi directions so that it has higher probability in finding the global optimum. (2) EA has few limitations on the objective function of the problem. The objective function may even be discontinuous or non-differentiable. (3) EA has good robustness. It is convenient for EA to handle discrete and complex nonlinear problem. (4) Moreover, EA methods can manage to provide sub-optimal solutions for infeasible problem. These features of EA have showed great competitiveness on conventional optimization methods. Therefore, EA has been widely used in the optimization problem in fields such as mechanics, chemistry, and computation etc. as well as the power systems [Miranda, Srinivasan, and Proenca, 1998].

Despite the superiority described above, EA has its disadvantage that the searching ability is constrained by the population size. The population size is decided by the dimension and complexity of the problem to be optimized. Higher

dimension needs larger population size, so does the complexity. If population size is not large enough, diversity of the population will be destroyed so that population will be trapped in a sub-optimum. This phenomenon is called premature convergence. Since each individual has to calculate its fitness function in the evolution process, when population size is large and the problem complexity is high, it has to take a plenty of time to obtain a satisfying solution.

Even though applications of EA in the study of static OPF are not new [Yuryevich & Wong, 1999; Bakirtzis & Biskas, 2002], research on the transient stability constrained OPF problem is few. Modeling of TSCOPF problem with EA method may be a main reason. The superiorities of EA to conventional derivative-based methods described above give itself strong reasons to be a promising way for solving the complex TSCOPF problem. This thesis aims to develop a novel way to deal with the transient stability constrained OPF problem using a new developed EA method called differential evolution (DE). Its optimization behaviors in both static OPF and transient stability constrained OPF problem are studied thoroughly. Transient stability performance of each individual (candidate solution) is assessed by hybrid method and incorporated into the DE based OPF as an index which has important influence on the evolution direction of DE. In particular, improvements have been made in DE's evolution scheme to ameliorate both the premature problem and long computation time caused by transient stability analysis of each individual. Besides the transient stability constraints, other non-convex and discontinuous constraints like generator

valve-point effects, prohibited operation zones constraints that are difficult to handle by conventional methods are also considered into the OPF problem to study its capability in practical application. Further more, a parallel computation platform is constructed for RDE to release the intense computational burden especially for large-scale systems.

1.4. Thesis Layout

Chapter 1 provides the background of transient stability constrained optimal power flow problem. Different methodologies are introduced and compared.

Chapter 2 introduces the mechanism of differential evolution (DE) algorithm. Performance and parameter setting of DE are studied through static OPF problem thoroughly. An improved version of DE is also proposed to address the premature problem of it.

Chapter 3 studies the performance of DE in handling continuous TSCOPF problem in detail. A hybrid method is employed to assess the transient stability of individuals. Strategies named stable-space push and fitness sorting are also developed to reduce the searching space as well as the computation time caused by the introduction of transient stability constraints. Comparisons with results obtained by conventional mathematical methods are also presented.

Chapter 4 focus on the ability of the proposed method in handling discontinuous and non-differentiable practical operation constraints such as discrete transformer tap settings, generators prohibitive operation zones (POZ),

and valve-point effects.

Chapter 5 investigates the parallel computation technologies and constructs a Beowulf pc-cluster platform for the proposed DE methods to release the computational burden.

Chapter 6 validates the effectiveness of parallel DE on the computation platform constructed in chapter 5 with both small and large systems. Different parallel topologies are studied. The performance of parallel DE is discussed in detail.

Chapter 7 presents overall conclusion of the work reported in this thesis and further possible directions.

1.5. List of Publications

Journal Paper:

1. H. R. Cai, C. Y. Chung, K. P. Wong, Application of Differential Evolution Algorithm for Transient Stability Constrained Optimal Power Flow, IEEE Transactions on Power Systems, Vol. 23, Issue 2, May 2008, Page:719-728.

2. H. R. Cai, C. Y. Chung, K. P. Wong, Parallel Multi-Constrained Optimal Power Flow with Discrete Control Variables Using Differential Evolution

Algorithm, to be submitted to the IEEE Transactions on Power Systems.

Conference Paper:

3. H. R. Cai, C. Y. Chung, K. P. Wong, Q. L. Liu, Parallel Differential Evolution Algorithm for Optimal Power Flow, The 6th International Conference on Power Transmission & Distribution Technology, Guangzhou, China, 10-12, Nov, 2007.

4. H. R. Cai, C. Y. Chung, K. P. Wong, Transient Stability Constrained Optimal Power Flow Using Differential Evolution Algorithm, Proceedings of International Conference on Electrical Engineering 2006 (ICEE2006), YongPyong Resort, Korea,9-13 July 2006.

Chapter 2 Differential Evolution

2.1 Introduction

Differential Evolution (DE) is a new kind of evolutionary optimization algorithm developed by Storn and Price. It is a simple yet powerful algorithm especially for problem with real-valued parameters [Storn & Price, 1995]. DE is proposed at the same year as Particle Swarm Optimization (PSO). Many researches [He, Wen, etc., 2004; Ahmed, Germano, 2005;] on OPF have shown that PSO is better than previously proposed EAs, such as Genetic Algorithm (GA), Evolutionary Programming (EP), and Evolution Strategy (ES). However, DE is paid much less attention than PSO, especially in the research field of OPF. In [Vesterstrom & Thomson, 2004], DE is compared with PSO based on 34 benchmark problems. The results have shown that DE is more efficient and robust. The optimization solutions are highly compromise with each other among different trials and barely influenced by the randomness of the initial population. Besides, DE has less parameter which means less dependence on parameter settings.

This chapter introduces the general mechanism of differential evolution

algorithm; applies it to the OPF problem to investigate the parameter setting and performance; and proposes a partial re-initialization strategy to improve the performance.

2.2 Basic DE Algorithm

DE seeks the solution of a problem by evolving a population composed of N_p individuals (solutions) $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{N_p}\}$ over a number of generations. Each individual is given a fitness function to indicate its optimality to the problem to be resolved. The evolution of the population is achieved through a reproduction process which creates new individuals from exiting ones. These new individuals will compete with their ancestors. Survivals compose the new generation for the next evolution. This iterative process moves on until a termination condition reached and the optimum solution is the best individual found ever. The basic process of DE is illustrated in Fig. 2.1 and the components are described in detail as below.

2.2.1 Individuals

An individual in the population is a candidate solution to the problem to be optimized, thus it principally takes the form of a *D*-dimensional real-valued vector. Each component in this vector is corresponding to a control variable of the problem. Suppose $\boldsymbol{u}_i[k]$ represents the *i*-th individual in generation *k*, $u_{i,j}[k]$

denotes the value of the *j*-th gene (control variable) of the *i*-th individual; the structure of an individual is as shown in Fig. 2.2.

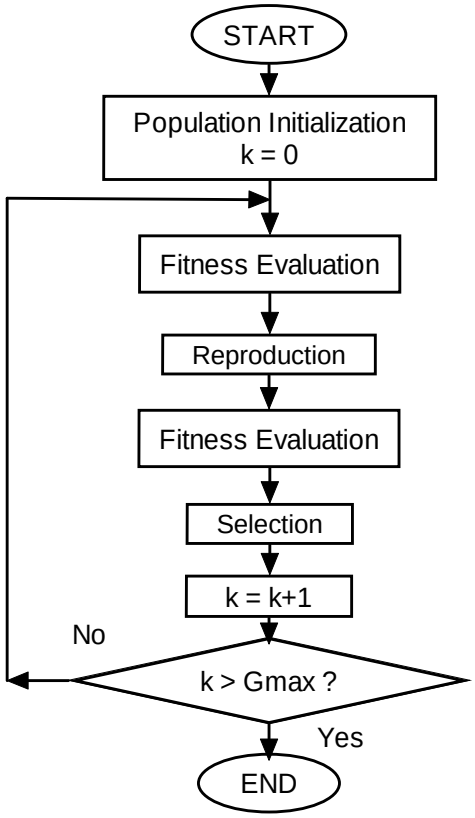

Fig. 2.1 Flowchart of differential evolution based OPF

$u_{i,1}[k]$	$u_{i,2}[k]$	$\cdots$	$u_{i,j}[k]$	$\cdots$	$u_{i,D}[k]$

Fig. 2.2 Structure of an individual

2.2.2 Initialization

All individuals in the population should be initialized at the beginning of the evolution process. The value of each control variable in an individual is obtained

as in equation (2.1).

$$u_{i,j}[0] = U[u_j^{\min}, u_j^{\max}] \qquad i = 1, \ldots, N_p, \quad j = 1, \ldots, D \tag{2.1}$$

where $U[u_j^{\min}, u_j^{\max}]$ is a uniform random number between the lower limit $u_j^{\min}$ and upper limit $u_j^{\max}$ of control variable *j*. For discrete variables, this random value is rounded off to the nearest valid value of the variable. In this way the initial population maintains a large diversity and hence provides an excellent starting point for a global search of the solution space.

2.2.3 Fitness

Each individual is given a fitness function $F(\boldsymbol{u}_i[k])$ to indicate its optimality to the problem to be resolved. Typically it is some function of the objective function of the problem.

2.2.4 Reproduction

Like other EAs, an offspring population of solutions of DE is produced from the existing population through a reproduction operator. The reproduction is driven by perturbing the value of each control variable in an individual using the difference between individuals selected randomly from the population; this is why the algorithm named as 'Differential Evolution'. The reproduction operator of DE combines the functions of mutation and crossover; two commonly used reproduction forms named *DE/current-to-rand/1* and *DE/current-to-best/1* are described here in equation (2.2) and (2.3).

DE/current-to-rand/1:

$$u'_{i,j}[k] = u_{i,j}[k] + \gamma(u_{r1,j} - u_{i,j}[k]) + \lambda(u_{r2,j}[k] - u_{r3,j}[k]) \quad (2.2)$$

DE/current-to-best/1:

$$u'_{i,j}[k] = u_{i,j}[k] + \gamma(u_{best,j} - u_{i,j}[k]) + \lambda(u_{r1,j}[k] - u_{r2,j}[k]) \quad (2.3)$$

where $u_{best,j}$ is the j-th control variable of the history best individual $\boldsymbol{u}_{best}$ found till the present generation; $r1 \neq r2 \neq r3 \neq i$ are integers randomly selected in range of [1, N_p]; the parameter γ and λ are parameters for contraction and diffusion. If $u'_{i,j}[k]$ is outside the feasible range of $[u_j^{\min}, u_j^{\max}]$, it is fixed to the limit $u_j^{\min}$ or $u_j^{\max}$. For discrete variables, $u'_{i,j}[k]$ is rounded off to the nearest valid value.

DE/current-to-best/1 can be viewed as a greedier version of *DE/current-to-rand/1*, since it exploits the information of the best individual $\boldsymbol{u}_{best}$ to guide the following search. By experience, the searching abilities of *DE/current-to-best/1* and *DE/current-to-rand/1* differ not much in case parameter λ is selected little larger in *DE/current-to-best/1* than in *DE/current-to-rand/1*. Since *DE/current-to-best/1* needs one less random numbers when reproducing an individual, we chose *DE/current-to-best/1* as the reproduction operator in this work.

The mechanism of *DE/current-to-best/1* can be illustrated with Fig. 2.3. The effect of item $\gamma(u_{best,j} - u_{i,j}[k])$ is accordant with the idea of 'population acceleration' [Wong & Li, 2002], it drives an individual $\boldsymbol{u}_i[k]$ to move towards $\boldsymbol{u}_{best}$ and reaches point M as shown in Fig. 2.3 (0< γ <1); while term

$\lambda(u_{r1,j}[k]-u_{r2,j}[k])$ gives a disturbance at point *M* to diverge the individual away from $\boldsymbol{u}_{best}$, the strength and direction of this disturbance are decided by the difference of two individuals $\boldsymbol{u}_{r1}[k]$ and $\boldsymbol{u}_{r2}[k]$ randomly selected from the population. These two items finally decide the position of the offspring individual $\boldsymbol{u}_i'[k]$. Though the contractive effect introduced by the acceleration may lead to premature of the population, it can be balanced by the diffusion effect provided by $\lambda(u_{r1,j}[k]-u_{r2,j}[k])$. Thus DE can still maintain a strong global searching ability when λ and γ are carefully selected.

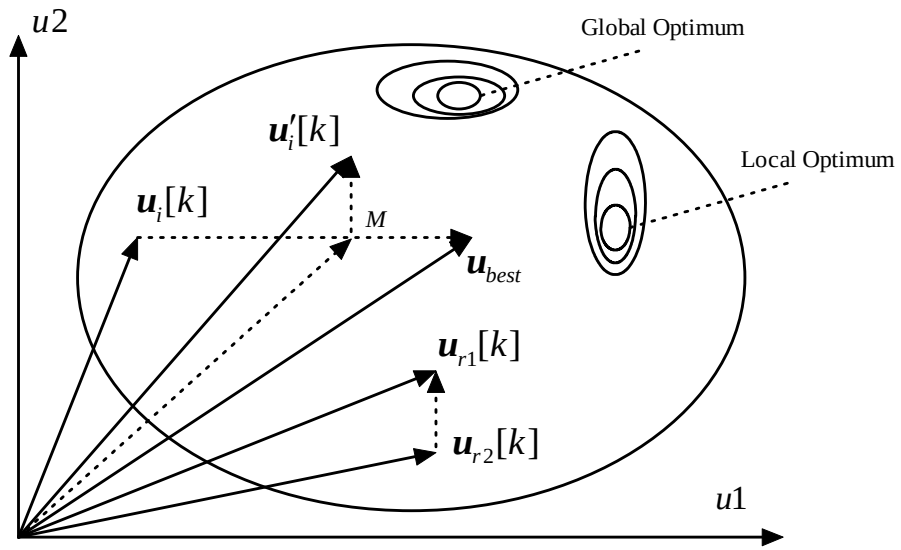

Fig. 2.3 Illustration of the mechanism of *DE/current-to-best/1*

2.2.5 Selection

The original population and the population produced by the reproduction operator will compete with each other and survivors build up the new generation

for the next evolution. Generally, DE uses the Binary Tournament Selection One-to-one Selection scheme. In other word, each offspring individual $\boldsymbol{u}_i'[k]$ is compared with its parent individual $\boldsymbol{u}_i[k]$ and winner is selected to be $\boldsymbol{u}_i[k+1]$ in the next generation as shown in (2.4).

$$\boldsymbol{u}_i[k+1]=\begin{cases}\boldsymbol{u}_i'[k] & if \quad F(\boldsymbol{u}_i'[k])\geq F(\boldsymbol{u}_i[k])\\ \boldsymbol{u}_i[k] & Otherwise\end{cases} \qquad i=1,\ldots,N_p \tag{2.4}$$

Practically, there are two ways to implement the selection operation [Lampinen & Storn, 2004]:

(1) The selection operation is implemented after all offspring individuals have been produced. The offspring individuals do not participate in the reproduction procedure. Each offspring individual is compared with his corresponding father one by one;

(2) Each time when a father individual produces his offspring individual, these two competes with each other and survivor substitutes the old one in the population immediately. These survivors will participate in the reproduction operation for the following individuals in the population. Thus the reproduction and selection process will interact with each other.

The latter way is greedier than the former since new individuals participate in the evolution earlier. High greedy may help the population converge faster; however, it may also lead the population to premature. Moreover, the second way introduces an unbalance into the population since individuals have different lifetimes according to their positions in the population. Therefore, the first way of selection is used in this work.

2.2.6 Termination Criteria

The commonly used termination criterion that stops the evolution when a preset maximum generation $G_{\max}$ reached is adopted in this work. Also criterion like stops the evolution when there is no appreciable change in the best fitness within the population for a number of iterations can be adopted.

2.3 Basic DE Optimal Power Flow

The optimal power flow problem is described in section 1.2. In this section, the basic OPF problem is formulated based on DE to investigate the parameter settings and performances of DE. For convenience, the detail mathematical model of OPF is formulated here. The optimization object in this work is to minimize the total generating fuel cost; control variables are generator active output (P_g), generator voltage magnitude (V_g), and transformer tap settings (T); dependent variables are generator reactive output (Q_g), and PQ-node voltage magnitude (V_{pq}).

$$\text{Min:} \quad f = \sum_{i=1}^{N_g} C_i \tag{2.5}$$

$$\text{s.t.} \quad P_{gi} - P_{di} - V_i \sum_{j=1}^{N} V_j (G_{ij} \cos\alpha_{ij} + B_{ij} \sin\alpha_{ij}) = 0, \qquad i = 1,\ldots,N \tag{2.6}$$

$$Q_{gi} - Q_{di} - V_i \sum_{j=1}^{N} V_j (G_{ij} \sin\alpha_{ij} - B_{ij} \cos\alpha_{ij}) = 0, \qquad i = 1,\ldots,N \tag{2.7}$$

$$P_{gi}^{\min} \le P_{gi} \le P_{gi}^{\max}, \qquad i = 1,\ldots,N_g \tag{2.8}$$

$$V_{gi}^{\min} \le V_{gi} \le V_{gi}^{\max}, \qquad i = 1,\ldots,N_g \tag{2.9}$$

$$T_i^{\min} \le T_i \le T_i^{\max}, \qquad i = 1,\ldots,N_t \tag{2.10}$$

$$Q_{gi}^{\min} \le Q_{gi} \le Q_{gi}^{\max}, \qquad i = 1,\ldots,N_g \tag{2.11}$$

$$V_{pqi}^{\min} \le V_{pqi} \le V_{pqi}^{\max}, \qquad i = 1,\ldots,N_{pq} \tag{2.12}$$

where C_i in (2.5) is the fuel cost of generator i, N is the total number of system buses, N_g is the total number of generating units, N_{pq} is the total number of PQ-node, N_l is the total number of branches, and N_t is the total number of transformers. P_{di} and Q_{di} are the active and reactive power loads of bus i; α_{ij} is the voltage angle difference between bus i and j; G_{ij} and B_{ij} are transfer admittance between bus i and j; T_i is the tap settings of transformer i.

Equation (2.6) and (2.7) are load flow equality constraints; (2.8) to (2.10) are control variable inequality constraints; and (2.11) to (2.12) are dependent variable inequality constraints. The major components of DE that related to the optimal power flow problem are described below.

2.3.1 Individuals

An individual within the population represents a candidate optimal power flow solution. The elements within the individual are control variables of the problem as stated above. It has to be mentioned here that slack unit active generation is not a control variable here since its value is determined by load flow equations. Detail composition of an individual is illustrated in Fig. 2.4.

P_{g2}	$\cdots$	P_{gN_g}	V_{g1}	$\cdots$	V_{gN_g}	T_1	$\cdots$	T_{N_t}

Fig. 2.4 Control variables encoded in DE individuals

2.3.2 Constrained Initialization

Generally, all control variables in an individual are initialized randomly within their limits. To help to meet the satisfaction of the slack bus active power constraints and power flow balance, a constrained initialization are introduced here for the generator active power outputs initialization. Suppose the total active load of the system is P_L, total power loss is P_{loss}, sum of the power that has been dispatched to all generators excluding the slack unit is P_{ds}, lower and upper limits of slack bus active power are $P_{slack}^{\min}$ and $P_{slack}^{\max}$.

(a) If $P_{ds} + P_{slack}^{\max} < P_L$, the candidate is inevitably an infeasible solution, it will be abandoned and a new one will be re-initialized;

(b) If $P_L + P_{loss} < P_{ds} + P_{slack}^{\min}$, here we set P_{loss} as 10% of P_L, the candidate will not be accepted and also a new candidate will be generated, because we think the candidate is not a good one since the power loss rate is too high; These constraints will help to reduce the searching space with little time consumption.

2.3.3 Fitness of Individual

As we have mentioned, a fitness value F is assigned to each individual to

measure the quality of this individual in DE. In the OPF problem, as well as minimizing the system generating fuel cost, we have to ensure the feasibility of the solutions. Violations of variable constraints are treated as punishment in the fitness function. Thus, the fitness function of individual *i* is formulated as below:

$$F_i = 1/(f_i + K_V F_{Vi} + K_Q F_{Qi} + K_{PS} F_{PS}) \tag{2.13}$$

$$F_{Vi} = \sum_{j=1}^{N_{pq}} (|V_{pqij} - V_{pqij}^{\lim}|)/(V_{pqij}^{\max} - V_{pqij}^{\min}) \tag{2.14}$$

$$F_{Qi} = \sum_{j=1}^{N_g} (|Q_{gij} - Q_{gij}^{\lim}|)/(Q_{gij}^{\max} - Q_{gij}^{\min}) \tag{2.15}$$

$$F_{PS} = (P_{slack} - P_{slack}^{\lim})/(P_{slack}^{\max} - P_{slack}^{\min}) \tag{2.16}$$

where f_i is the total generating fuel cost, F_{Vi} and F_{Qi} denote the sum of the normalized violations of PQ-bus voltages and generator reactive power outputs of individual *i*, respectively; $V_{pqij}^{\lim}$ and $Q_{gij}^{\lim}$ denote the violated lower or upper limits of the load-bus voltages $(V_{pqij}^{\min}, V_{pqij}^{\max})$ and the generators' reactive power outputs $(Q_{gij}^{\min}, Q_{gij}^{\max})$ respectively; F_{PS} denotes the violation of slack bus active power limitation of individual *i*, $P_{slack}^{\min}$ and $P_{slack}^{\max}$ are the lower and upper limits of it. K_Q, K_V, and K_{PS} are the corresponding penalty coefficients. An individual is deemed better if its fitness value is higher.

2.4 Parameter Selection of DE

Similar to other EAs, proper parameter selection is important for the performance of DE. Parameters in the DE algorithm are much less than other EAs,

therefore, it is relatively easier to determine. The parameters in DE include reproduction coefficients γ and λ in (2.3) and the population size N_p. Although the parameter settings are problem-specified, in DE, they normally lie within a close range.

The selection of N_p is largely dependent on the control variable dimension D. For real-world engineering problems, $N_p \geq 20D$ may probably be more than enough and it will lead to heavy computation burden, while it is generally difficult to obtain the optimal solution for $N_p < 2D$ [Corne, Dorigo, & Glover, 1999]. Other experiments suggest to address N_p in [3D, 8D] [Gamperle, Muller & Koumoutsakos, 2002]. As a rough principle, $N_p = 5D$ is a good choice for a first try because the optimal solutions are often possible to be obtained.

The reproduction scheme in (2.3) combines two parts of operations. The term $\gamma(u_{best,j} - u_{i,j}[k])$ contracts an individual to move to the present optimal point $\boldsymbol{u}_{best}$, while term $\lambda(u_{r1,j}[k] - u_{r2,j}[k])$ tries to give it a random deflection to escape from the moving direction to maintain population diversity. It is obvious that γ controls the strength of contractive pressure and λ controls the strength of diffusive pressure. High ratio of γ to λ may result in premature convergence, while, low ratio of it may slow down the convergence speed. Therefore, the contractive and diffusive effective should be well balanced. K. V. Price pointed out that for DE, when using reproduction scheme in (2.3), choosing γ randomly from the range of [0, 1] per individual per generation is frequently very effective, which is the scheme to be used in this paper. For parameter λ, when it is lower than 0.5 may

lead to premature convergence, while greater than 1.0 tend to slow down the convergence speed. So we range λ in [0.5, 0.9], and the critical value will be determined by the simulation tests.

2.4.1 Test cases

The IEEE 14- and 30-bus test systems are employed to investigate the setting of DE parameters. System bus and branch data are available on [http://www.ee.washington.edu/research/pstca/]; generator cost data and control variable limits are the same as in [Zimmerman & Gan, 1997, MATPOWER] for IEEE 14-bus system and in [Alsac & Scott, 1974] for IEEE 30-bus system. The IEEE 14-bus system has 12 control variables including 4 generator active power outputs, 5 generator bus voltages, and 3 transformer tap settings; the IEEE 30-bus system has 15 control variables including 5 generator active power outputs, 6 generator bus voltages, and 4 transformer tap settings. Thus the population size for each system is 60 and 75, respectively. The tests are run on a 2.66GHz Pentium IV PC. Totally 20 trails are performed for each test case to see the average performance.

2.4.2 Simulation results of DE with different λ values

In this part, simulations are conducted with different λ values from 0.5 to 0.9 on IEEE 14 and 30-bus systems. To eliminate the influence of population size, N_p is set as large as $5D$ for all λ, the maximum generation number $G_{\max}$ is set to

100. the minimum, maximum, and average fuel cost corresponding to different λ values among those 20 trials for each case are listed in Table 2.1 and Table 2.2. Similar conclusions can be drawn that considering both the fuel cost solutions and the standard deviations of all trails, the best setting for λ should be 0.7. Therefore, λ is set to 0.7 for the rest of the simulation studies in this paper.

Table 2.1 Simulation results with different λ for IEEE 14-bus system

λ	0.5	0.6	0.7	0.8	0.9
Minimum Cost ($/h)	8076.83	8076.79	8076.75	8077.10	8078.49
Maximum Cost ($/h)	8085.97	8077.54	8077.06	8079.11	8084.18
Average Cost ($/h)	8078.79	8076.89	8076.81	8077.59	8080.66
Standard Deviation	2.09	0.16	0.07	0.47	1.63

Table 2.2 Simulation results with different λ for IEEE 30-bus system

λ	0.5	0.6	0.7	0.8	0.9
Minimum Cost ($/h)	802.11	802.02	802.03	802.08	802.30
Maximum Cost ($/h)	806.57	802.62	802.09	802.31	803.41
Average Cost ($/h)	803.03	802.13	802.05	802.16	802.54
Standard Deviation	1.13	0.16	0.01	0.06	0.24

2.4.3 Basic DE performance

Since λ has been set, different population sizes of $N_p > 5D$ have been tried for these two systems but no significant improvements made. It can be concluded that $N_p = 5D$ is large enough. The best system control variable settings with

minimum fuel costs are listed in Table 2.3 and Table 2.4 for IEEE 14-bus and IEEE 30-bus systems, respectively. Fig. 2.5 and Fig. 2.6 show the average convergence curves over those 20 trails for these two test systems. In the beginning parts of the convergence curves, the fuel costs may increase during several generations. This phenomenon indicates the process of eliminating system variables' violations. For both systems, the convergence curves drop rapidly which proves the effectiveness of DE. Moreover, the slight differences between the minimum cost and maximum cost also prove the robustness of DE.

DE took 1.0 s to evaluate 100 generations for the IEEE 14-bus system and 3.2 s for the IEEE 30-bus system; however, since the algorithm converges rapidly, it took only 50 generations of DE to find a good enough solution for both systems, which means half of the running time.

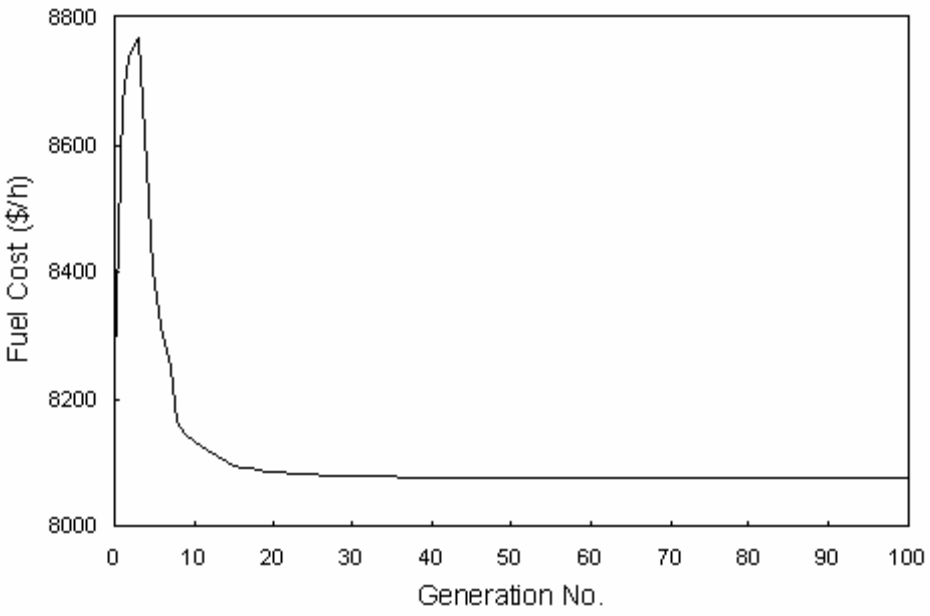

Fig. 2.5 Average convergence curve for IEEE 14-bus system

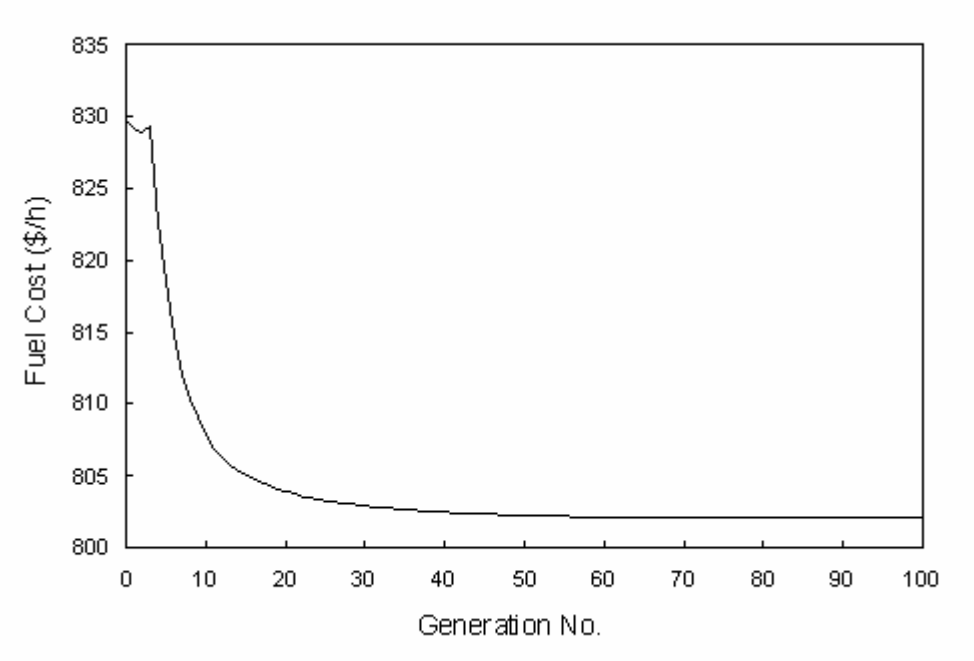

Fig. 2.6 Average convergence curve for IEEE 30-bus system

Table 2.3 Best system control variable settings for IEEE 14-bus system

Bus	Pg (MW)	Vg (p.u.)	Branch	Transformer Tap
1	194.77	1.060	4 – 7	1.01
2	36.78	1.039	4 – 9	0.90
3	28.82	1.016	5 – 6	0.98
6	0.00	1.054		
8	7.83	1.060		
Fuel Cost ($/h)			8076.75	
Initial Cost ($/h)			8172.00	
Cost Saving			1.17%	

Table 2.4 Best system control variable settings for IEEE 30-bus system

Bus	Pg (MW)	Vg (p.u.)	Branch	Transformer Tap

1	176.46	1.050	6 – 9	1.00
2	48.71	1.037	6 – 10	0.96
5	21.53	1.013	4 – 12	1.00
8	22.06	1.020	28 – 27	0.94
11	12.01	1.087		
13	12.03	1.085		
Fuel Cost ($/h)			802.03	
Initial Cost ($/h)			872.29	
Cost Saving			8.05%	

In order to investigate the robustness of DE, the simulation results on the IEEE 30-bus system are compared with those reported in literatures: the gradient based optimal power flow method in [Alsac & Scott, 1974], the EP based OPF combined with gradient acceleration in [Yuryevich & Wong, 1999], the best known result obtained in [Bakirtzis & Biskas, 2002] using enhanced GA, and improved PSO method in [He, Wen, Prempain, etc., 2004]. All simulations on these methods have the same system data, control variable limits and initial conditions. These simulation results are summarized in Table 2.5. Obviously, the minimum fuel cost obtained by DE is lower than those obtained by gradient method and EP. For GA and PSO, the results are close to DE, however, GA needed a population of 80 individuals over 200 generations; and PSO needed a population of 50 individuals over 500 generations to obtain these results. While for DE, it used 75 individuals iterating over 100 generation (which means much less load flow calculation times) to get a better solution of 802.03$/h. These results

demonstrate that DE converges fast with excellent optimization capability.

Table 2.5 Simulation results comparison for IEEE 30-bus system

	Gradient	EP	GA	PSO	DE
Minimum Cost ($/h)	802.40	802.62	802.06	802.0477	802.03
Population size	\	20	80	50	75
Generation	\	50	200	500	100
Load flow times	\	1000	16000	25000	7500

2.5 Reinitialized Differential Evolution

2.5.1 Method Introduction

Despite the simplicity, effectiveness, and robustness of DE, it requires relatively large population size to avoid premature convergence. The reason for this disadvantage exits in the reproduction scheme of DE. From (2.3), we can see that unlike GA or EP, which has a randomly mutation scheme in reproduction, any offspring individual of DE is actually a linear combination of the initial population space. When the population size is small, the information embedded in the individuals to share and spread over is finite. This limitation on possible combinations may impede the global search ability of DE and lead to premature especially for high dimensional problem. Larger population size may not only increase current searching points but also bring more combination possibilities;

however, computational burden also becomes heavier with the population size increasing especially for large systems. To address this disadvantage, an improved version of DE named reinitialized DE (RDE) is proposed here.

The main idea is after iteration of L generations; we do not reproduce population but reinitialize it. The reinitialized individuals act as fresh blood from external family which can help much to maintain the population diversity and stretch the searching space. The history best individual $\boldsymbol{u}_{best}$ is kept recorded and acts as seed in the reproduction of new generations, the convergence characteristics will not be influenced by the reinitialization. Since repeated load flow calculation is the most time consuming part in DE based OPF, time for reinitialization is almost negligible; therefore, the whole computation time will reduce since smaller population size is need by RDE.

The main process of RDE is illustrated in Fig. 2.7 below.

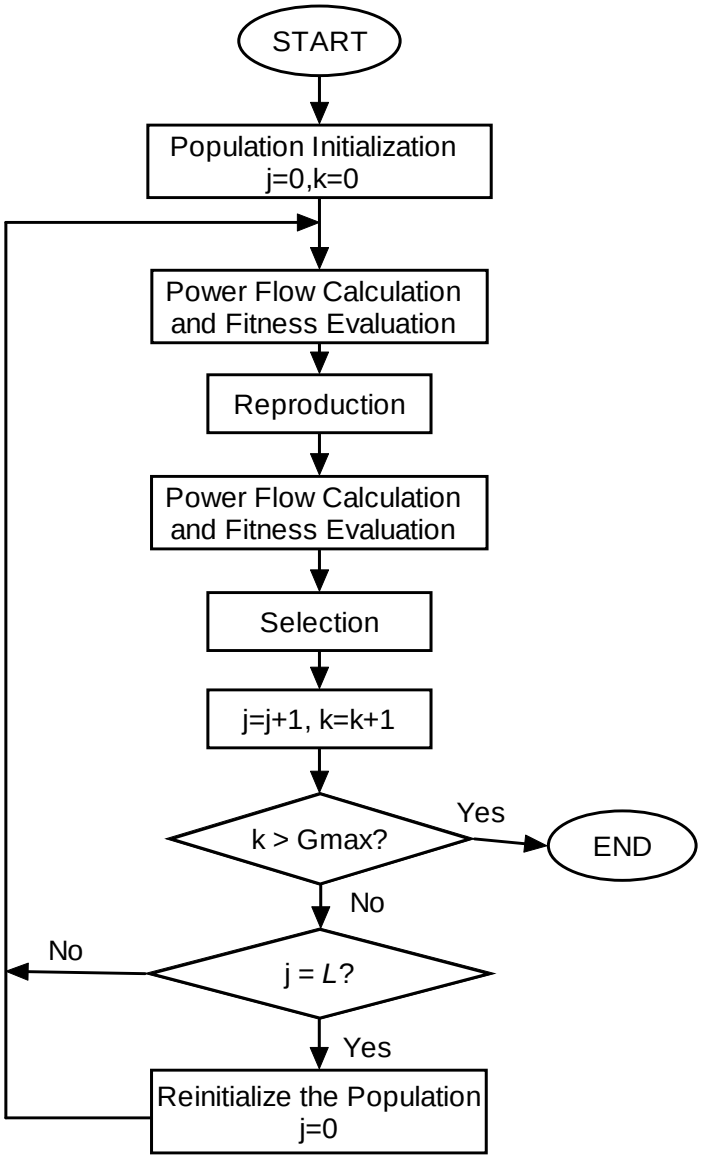

Fig. 2.7 Flowchart of reinitialized differential evolution (RDE)

2.5.2 RDE Performance Study

The IEEE 118-bus system is employed here to testify the effect of the proposed RDE method. System bus and branch data are available on [http://www.ee.washington.edu/research/pstca/]; generator cost data and control variable limits are the same as in [Zimmerman & Gan, 1997, MATPOWER]. This

system has totally 116 control variables including 53 unit active power outputs, 54 generator voltage magnitudes, and 9 transformer tap-settings. For basic DE, we set the population size from about 3*D* to 5*D*, that is, from 360 to 600. For RDE, we set the population size at 360 (3*D*) and change *L*, the cycle of reinitialization, to examine the performance of RDE. Both DE and RDE are run over 1000 generations for 20 trails. Simulations results for these two methods are listed in Table 2.6 and 2.7.

Table 2.6 Simulation results of basic DE with different population size for IEEE 118-bus test system

N_p	Minimum Cost ($/h)	Maximum Cost ($/h)	Average Cost ($/h)	STD %	ACT (s)
360	131495.72	134326.64	132568.10	0.61%	1088.15
480	130414.08	132501.33	131134.27	0.36%	1446.87
600	130265.52	131142.27	130634.69	0.19%	1818.46

STD: Standard Deviation. ACT: Average Computational Time.

Table 2.7 Simulation results of RDE with different re-initialization cycle for IEEE 118-bus test system

L	Minimum Cost ($/h)	Maximum Cost ($/h)	Average Cost ($/h)	STD %	ACT (s)
500	131045.02	134248.66	131938.60	0.52%	1088.65
300	130614.29	133049.00	131652.62	0.50%	1088.73
250	130585.87	132958.91	131473.57	0.43%	1091.28
200	130533.61	132585.44	131359.55	0.42%	1092.05
100	130204.93	131574.06	130636.57	0.21%	1094.35

Compare Table 2.6 and Table 2.7, it is no doubt that RDE can obtain better solution than basic DE with the same population size, moreover, when the reinitialization cycle L reaches 100, that is, reinitialize the population per 100 generations during the 1000 iterations, the solution obtained by RDE is comparable to basic DE with population size as large as 5D. The computation time needed for basic DE is 1818.46 seconds, for RDE is 1094.35 seconds which differs not much with that of basic DE with N_p=3D. Thus, conclusion can be dawn that RDE do help to rescue the population from trap of local minimum with less population size and computation time.

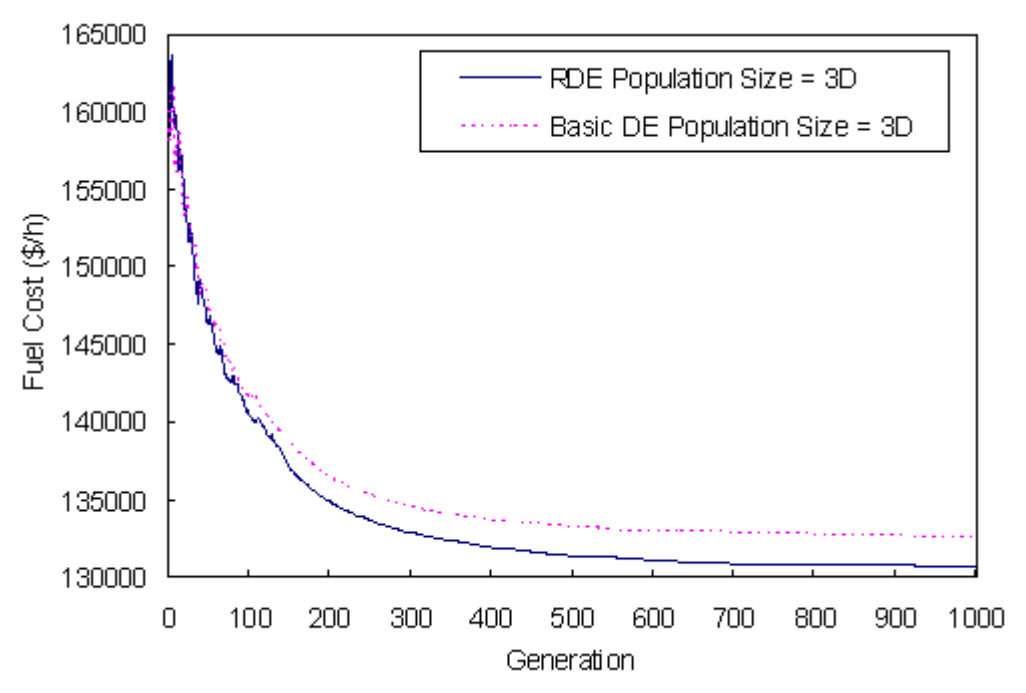

Fig. 2.8 Average convergence curve for IEEE 118-bus system

The average converge curves of basic DE and RDE based OPF for IEEE 118-bus system with population size = 360 (3D) are shown in Fig. 2.8. Obviously

we can see that RDE converges better than basic DE. The best system control variable settings obtained by RDE method with N_p=360, L=100 are listed in Table 2.8.

Table 2.8 Best solution of RDE based OPF of IEEE 118-bus system

Bus	Pg (MW)	Vg (p.u.)	Bus	Pg (MW)	Vg (p.u.)
69	431.54	1.0549	62	2.62	1.0532
1	1.11	1.0163	65	354.75	1.0596
4	0.00	1.0599	1.0599	343.88	1.0599
6	21.58	1.0448	1.0448	2.79	1.0301
8	24.71	1.0234	1.0234	0.26	1.0597
10	401.72	1.0594	1.0594	2.30	1.0356
12	86.53	1.0463	1.0463	6.10	1.0061
15	0.00	1.005	1.005	25.15	0.9923
18	2.19	1.009	1.009	0.26	1.0265
19	17.59	0.9983	0.9983	42.09	1.0396
24	0.08	1.029	1.029	2.20	1.0345
25	20.77	1.0366	1.0366	3.63	1.0533
26	27.01	1.0514	1.0514	490.89	1.0577
27	1.17	1.0027	1.0027	0.48	1.0216
31	6.55	1.0237	1.0237	0.00	1.0046
32	28.5	1.0175	1.0175	0.18	1.0348
34	25.84	0.9808	0.9808	8.58	1.0385
36	0.00	0.9755	0.9755	226.57	1.042
40	36.81	0.9629	0.9629	36.58	1.0333
42	34.18	0.9437	0.9437	0.00	1.0204
46	17.47	1.0331	1.0331	0.00	1.0223

49	201.53	1.0596	1.0596	98.87	1.0362
54	50.87	1.038	1.038	0.05	1.0108
55	49.31	1.0357	1.0357	32.36	1.0109
56	0.15	1.0353	1.0353	39.11	1.0169
59	160.41	1.0394	1.0394	0.32	1.0397
61	143.80	1.0573	1.0573	7.16	1.0191
Branch	Transformer Tap		Branch	Transformer Tap	
8 – 5	0.95		64 – 61	1.00	
26 – 25	1.05		65 – 66	1.00	
30 – 17	1.02		68 – 1	1.00	
38 – 37	1.05		81 – 88	0.95	
65 – 59	0.98				
Fuel Cost ($/h)			130204.93		

2.6 Summary

In this chapter, basic DE algorithm is introduced and applied to solve the OPF problem. The mechanism of DE has been analyzed in detail; the parameters of DE is discussed and decided by simulation. So far, the simulation results showed that although DE needs a relatively large population size, it manages to provide better results than those obtained from other optimization techniques in terms of accuracy and convergence speed due to DE's simple reproduction scheme and robustness. Based on basic DE, a reinitialized differential evolution (RDE) algorithm is proposed to address the premature problem of it. By introduction of the re-initialization, population diversity can be optimized which help to reduce

the population size as well as speed the optimization. Numerical experiments carried out on the IEEE 118-bus system verified that RDE can save the computational time significantly.

Chapter 3 Continuous Transient Stability Constrained OPF

3.1 Introduction

From the discussion in Chapter 1, it is of increasingly importance to consider transient stability constraints in Optimal Power Flow (OPF) problems since modern power systems tend to operate closer to stability boundaries due to the rapid increase of electricity demand and the deregulation of electricity markets. New and robust algorithm should be developed for the transient stability constrained OPF (TSCOPF) problem which overcomes the difficulties that conventional mathematical methods are limited to handle. This chapter develops a TSCOPF method based on the RDE algorithm introduced in Chapter 2. The transient stability constraints are embedded into the OPF problem as a stability index of the individuals which has an impact on the evolution direction of the population. It removes the difficulties associated with the handling of transient differential equations. To reduce the computational burden, several strategies are further proposed in the transient assessment and selection of solution individuals in evolution process of RDE.

As mentioned in Chapter 1, it is difficult for the present mathematical methods to handle discrete optimal problems. To compare the performance with conventional methods reported in the literature, only continuous control variables are considered in this chapter.

3.2 Transient Stability Assessment

Transient stability assessment (TSA) is the evaluation of the stability of a power system to withstand specified contingencies by surviving the subsequent transient events and arrive at an acceptable steady state operating condition [Fouad and Vittal, 1992]. Many advanced methods have been developed for transient stability assessment. These methods include time-domain simulation, transient energy function (TEF) methods [Fouad and Vittal, 1992; Pai, 1989], and hybrid methods [Maria, Tang, and Kim, 1990; Pavella, 1998].

Most of the researches in TSCOPF problems consider the transient stability constraints through time domain simulation and constrain the relative rotor angle within a predefined limit, for example 100 degree [Gan, Thomas, and Zimmerman, 2000; Yuan, Kubkawa, and Sasaki, 2003] or π *rad* [Nguye and Pai, 2003]. However, these thresholds for different systems may vary and also cannot be defined easily. Besides, although TEF methods can produce a Transient Stability Index (TSI) to measure the relative stability of the system, these methods are suffering from convergence problems in the calculation of the controlling unstable

equilibrium point, limited modeling capability and difficulty to identify the critical machine group. To avoid these problems, the hybrid method, which combines time-domain simulation and TEF method, is used in this study. Time-domain simulation is first performed to calculate the generator rotor angles and then transient energies are calculated to determine the system stability based on the results of time-domain simulation, in which the detailed models can be incorporated. This section will describe the representation of transient stability constraints and the procedure of transient stability assessment in detail.

The transient behavior of a N_g -generator power system is described by a set of differential and algebraic equations as follows:

$$M_i \frac{d^2\delta_i}{dt^2} = P_{mi} - P_{ei} \tag{3.1}$$

$$\dot{\delta}_i = \omega_i \tag{3.2}$$

where $\dot{\delta}_i$ and ω_i are rotor angle and angular speed of generator *i*; P_{mi} and P_{ei} are the mechanical power input and electrical power output of generator *i*; and M_i is the moment of inertia of generator *i*.

Since in transient stability studies, relative movements of rotor angles have to be studied, a frame of reference must be set up. Usually, the center of inertia (COI) coordinate is used. It can be represented by a linear combination of all generator rotor angles:

$$\delta_{COI} = \frac{1}{M_T} \sum_{i=1}^{N_g} M_i \delta_i \tag{3.3}$$

where M_T is the inertia center and it is defined as:

$$M_T = \sum_{i=1}^{N_g} M_i \tag{3.4}$$

Then we have the rotor angle and speed in COI frame in (3.5) and (3.6) respectively as below:

$$\theta_i = \delta_i - \delta_{COI} \tag{3.5}$$

$$\dot{\theta}_i = \tilde{\omega}_i \tag{3.6}$$

Thus the system equations with respect to the COI frame are denoted here as:

$$M_i \dot{\tilde{\omega}}_i = P_{mi} - P_{ei} - \frac{M_i}{M_T} P_{COI} \equiv PAC_i \tag{3.7}$$

$$P_{COI} = \sum_{i=1}^{N_g} (P_{mi} - P_{ei}) \tag{3.8}$$

where PAC_i is the accelerating power of generator *i*.

The transient energy function of a power system modeled above is defined as:

$$TEF = KE + PE \tag{3.9}$$

$$KE = \frac{1}{2} \sum_{i=1}^{N_g} M_i \tilde{\omega}_i^2 \tag{3.10}$$

$$PE = -\sum_{i=1}^{N_g} \int_{\theta_i^{SEP}}^{\theta_i} PAC_i^P \, d\theta_i \tag{3.11}$$

where *KE* is the system kinetic energy; *PE* is the system potential energy; θ_i^{SEP} represents the rotor angle of the post-fault system's stable equilibrium point; and PAC_i^P is the accelerating power of the post-fault systems.

According to the transient energy function theory [Fouad and Vittal, 1992], the *TEF* remains constant in the post-fault period if the system damping is neglected; and this is called the *TEF* conservation property. If a system is stable after a contingency, *KE* and *PE* are both positive and keep smaller than *TEF*;

otherwise, *KE* increases and *PE* drops rapidly. Fig. 3.1 and 3.2 illustrate the variation of *KE*, *PE*, and *TEF* along typical stable and unstable trajectories respectively.

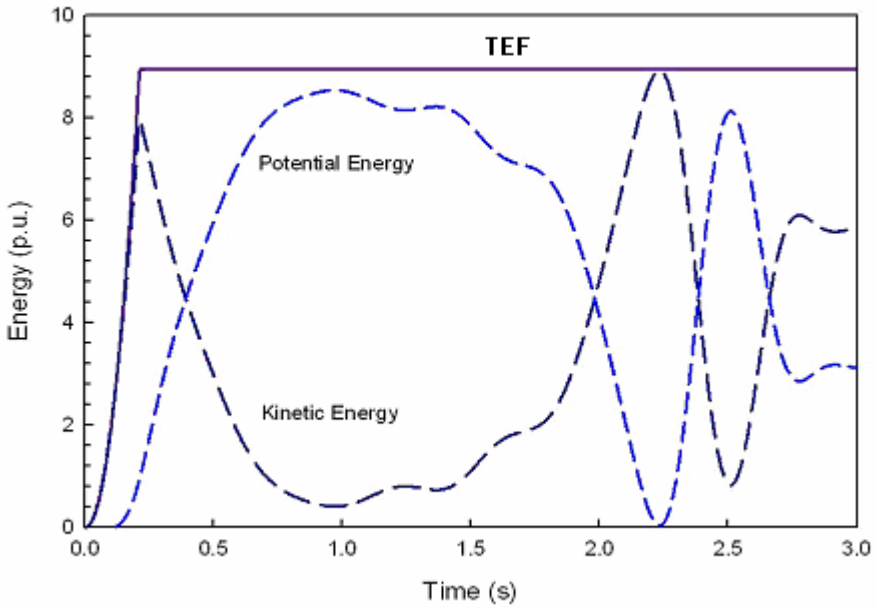

Fig. 3.1 Variations of KE, PE, and TEF along a stable trajectory

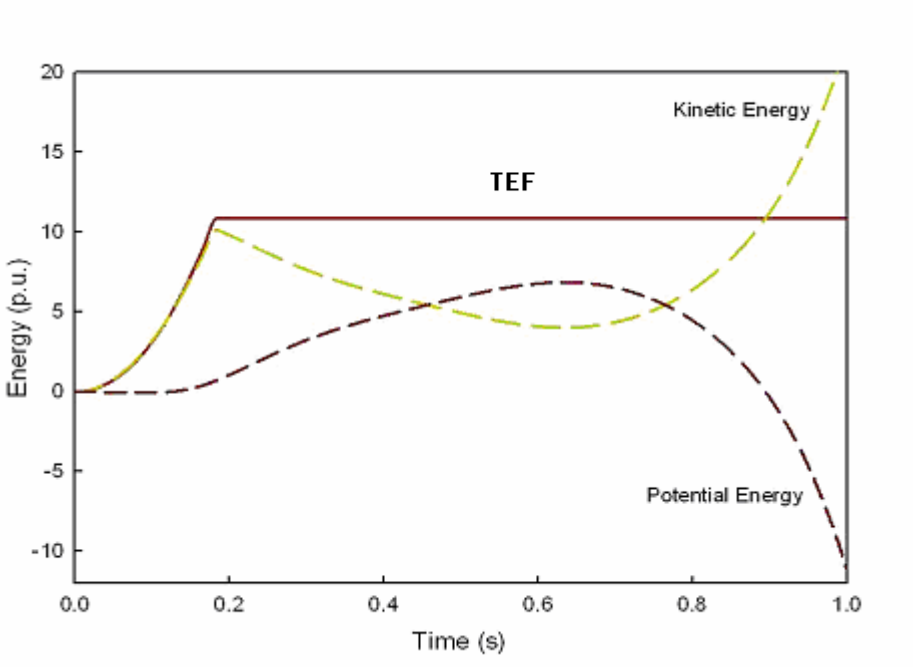

Fig. 3.2 Variations of KE, PE, and TEF along an unstable trajectory

From (3.10), it can be observed that *KE* is in proportion to the square of the angular speed ω, so the increasing rate of *KE* is very large. Since the *TEF* keeps almost constant, *KE* will surpass the limitation of the *TEF* and become much larger than it when the system goes unstable. We can make use of these characteristics to determine whether the system is transiently stable or not by the following procedures:

Step 1: Perform a time-domain simulation till the contingency is cleared; and calculate the value of *TEF* at this moment and denote it as TEF_{cl};

Step 2: Continue the time-domain simulation in Step 1 till the predefined simulation period reached; and then calculate *KE* of the last time step;

Step 3: Check whether *KE* is bounded within the value of TEF_{cl}. If yes, the system is stable; otherwise, the system is unstable.

3.3 Differential Evolution Algorithm for TSCOPF

Unlike conventional derivative based mathematical TSCOPF methods, RDE based TSCOPF needs not to combine the differential dynamic equations directly with basic OPF equations. Like the fitness function, we assign an index Stability to each individual in RDE. After fitness evaluation of each individual, transient stability assessment is carried out. If Stability = YES, the individual is able to keep the system stable after the contingency. Otherwise, this individual is not a one

that satisfies the transient stability requirements. Thus the relationship between OPF equations and dynamic equations are relaxed. The transient stability is not included into the fitness function, since transient stability here is a condition to be met, not an objective to optimize. This transient stability index will take its affection in the evolution procedure of RDE.

3.3.1 Stability Constrained Selection

The global best individual $\boldsymbol{u}_{best}$ denotes the best individual obtained from all former generations and it is also the final solution of the problem. For the first generation, it is just the best one in the initial population. To find $\boldsymbol{u}_{best}$, individuals needs to compete with one another. When two individuals $\boldsymbol{u}_a$ and $\boldsymbol{u}_b$ compete, we define that $\boldsymbol{u}_a$ is better than $\boldsymbol{u}_b$ if one of the following conditions is matched:

Condition (i): If both of them are stable, $\boldsymbol{u}_a$ has higher fitness value;

Condition (ii): If both of them are unstable, $\boldsymbol{u}_a$ has higher fitness value;

Condition (ii): If $\boldsymbol{u}_a$ is stable, while $\boldsymbol{u}_b$ is unstable.

This “stable space push” strategy helps to induce the searching direction towards space where the system can maintain its stability.

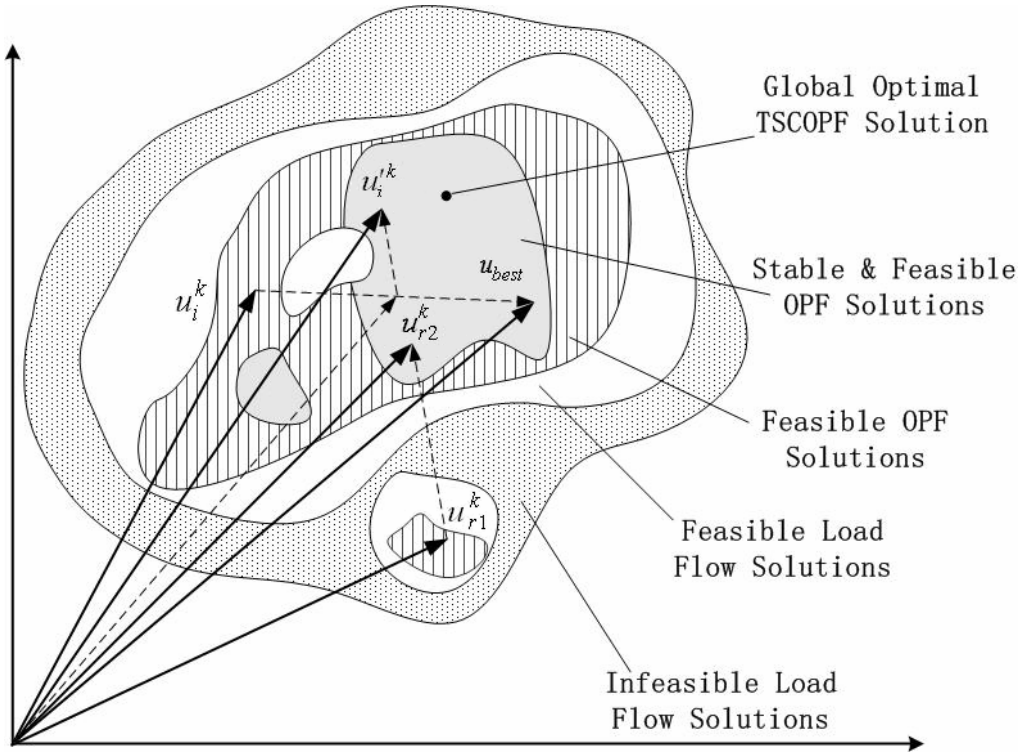

Fig. 3.3 Illustration of the reproduction mechanism of DE in TSCOPF searching space

The reproduction mechanism of DE for only two control variables is illustrated in Fig. 3.3. Obviously, the survival priority of stable individuals will devote to push the whole population to a stable space as well as finding the optimal fuel cost solution.

3.3.2 Partial Transient Stability Assessment

Since the searching space is very huge for TSCOPF optimization problems, and the transient stability assessment is the most time-consuming part, it is necessary to improve the efficiency for the practical use. Two measures are proposed here to improve the speed of DE iteration:

Strategy 1: After fitness evaluation of a generation, the whole population will be sorted by fitness from the best to the worst. It aims at finding out the elite part of the whole individuals, stable ones among these elite can help to lead the converge direction.

Strategy 2: Instead of all individuals in the population, only certain percentages of the population with better fitness will undergo the TSA calculation to find out some stable seeds used to push the population converging to a feasible and stable space. This can help to release the computational burden without deteriorating the reproduction characteristics of the evolution. The impacts of different percentages on the whole TSCOPF performance will be studied by following simulations.

The flowchart of the proposed TSCOPF is illustrated in Fig. 3.4.

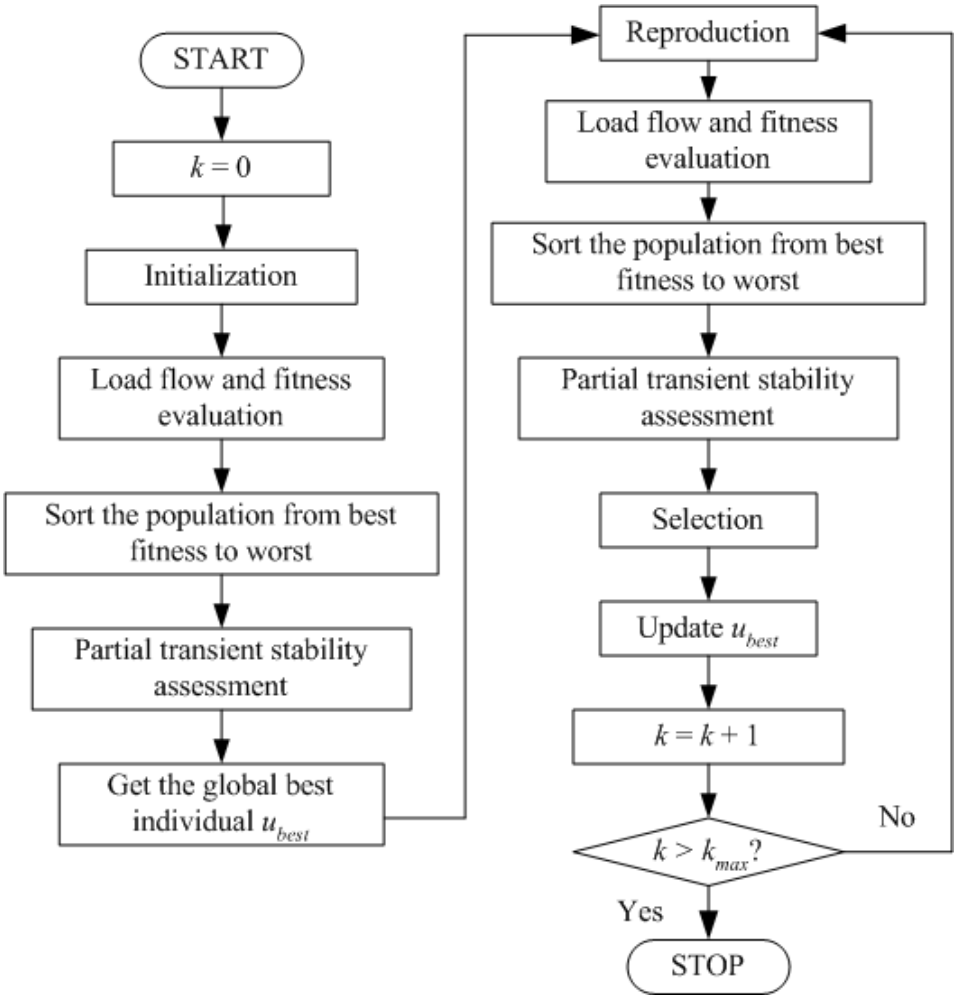

Fig. 3.4 Illustration of the DE based TSCOPF flowchart

3.4 Case Studies

The proposed RDE based TSCOPF method is tested on the WSCC 3-generator, 9-bus system and New England 10-generator, 39-bus system. In both test systems, the classical generator model is used for synchronous generator and loads are modeled as constant impedances. Integration time step is 0.01 second for transient stability simulation, the whole simulation period is 3.0 second. Besides, Critical Clearing Time (CCT), which is determined by repeating the transient time

domain simulation with different fault clearing times, is used to compare the stability performance of the optimal solutions obtained. For each test case, totally 20 trials are performed to verify the robustness of the proposed method. Program is run on a computer with Intel Pentium IV 2.66G CPU.

3.4.1 Simulation on the WSCC 3-generator, 9-bus System

The single line diagram of this system is shown in Fig. 3.5; and the system data are available in [Sauer, Pai, 1998]. The fuel cost parameters and the rating of generators are taken from [Nguyen, Pai, 2003]. The lower and upper limits of all bus voltage magnitudes are set at 0.95 p.u. and 1.05 p.u., respectively. There are five control variables including 2 generator active power outputs and 3 generator node voltages. The population size is set as 20 (i.e. four times of the number of control variables); and the maximum generation number is 100. The population is reinitialized per 10 generations. As a rough try for a simple system, half of the population will undergo the transient stability assessment. Two contingency cases are studied:

Case A: A three phase to ground fault at bus 7 and cleared by tripping line 7-5 at 350 ms, which is greater than the initial CCT 162 ms.

Case B: A three phase to ground fault at bus 9 and cleared by tripping line 9-6 at 300 ms, which is greater than the initial CCT 214 ms.

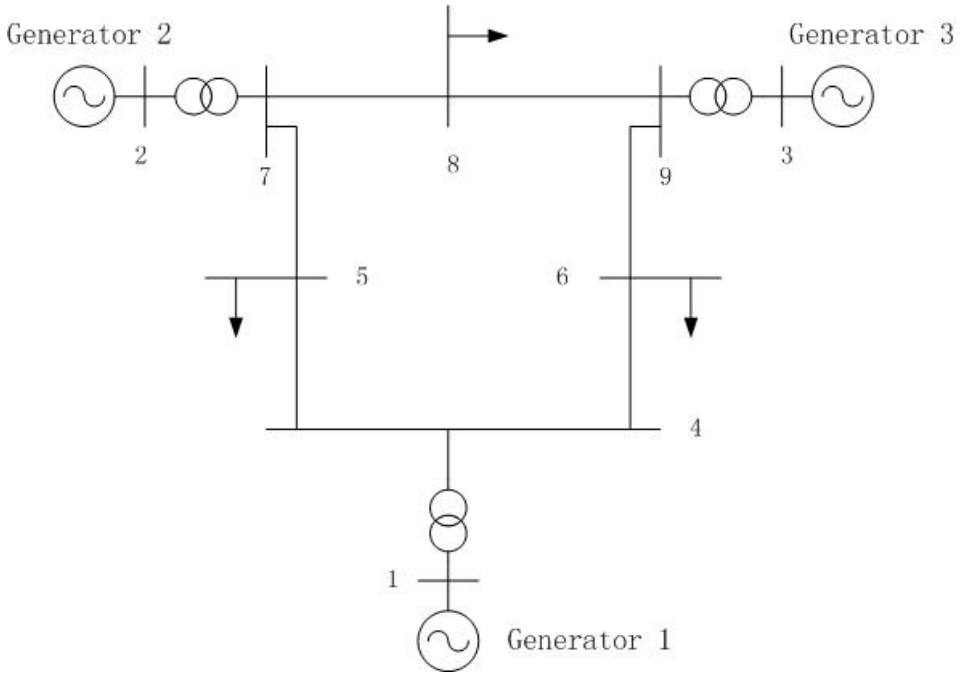

Fig. 3.5 WSCC 3-generator, 9-bus test system

The minimum, maximum, and average fuel costs of the final solutions among all 20 trials for the base OPF case, Case A, and Case B are listed in Table 3.1. Here, the base OPF case means the OPF solution without transient stability constraints, and it is also solved by the RDE algorithm. It can be found that the proposed method is very robust because the differences between the minimum costs and the maximum costs are very slight and also the standard deviations for all trials are fairly small. The average fuel costs for Case A and Case B are \$1140.65 and \$1148.58 respectively, which are rather smaller than the corresponding costs reported in [Nguyen, Pai, 2003] as \$1191.56 and \$1179.95 respectively. The costs of all TSCOPF cases are higher than that of the base OPF as expected because the economy of the system operation is sacrificed for the enhancement of the stability performance. It is important to note that the proposed method can ensure the

system stability in all 20 trials (100% stable) while the base OPF cannot (0% stable) as shown in Table 3.1. Besides, the average convergence curves for different cases in Fig. 3.6 clearly show the fast convergence property of the proposed method and a rapid drop of the fuel costs in the first 20 RDE generations.

The best system solutions (i.e. with minimum fuel cost among all trials) for each case are listed in Table 3.2; the CCTs for different conditions and contingencies are compared in Table 3.3. The CCT of the best solution obtained by the base OPF is only 0.293s for Case A and 0.219s for Case B respectively. Both are smaller than the fault clearing time (FCT), which means that the system will go unstable when the corresponding contingencies occur. While the proposed TSCOPF method can schedule the system with CCT larger than the FCT as well as minimize the generating cost. Besides, Fig. 3.7 and Fig. 3.8 show the stable trajectories of rotor angles with respect to COI coordinate of best solutions for Case A and Case B with best system solutions in Table 3.2 respectively.

Table 3.1 Simulation results of the WSCC 3-generator, 9-bus system

Case	Base OPF	A	B
Minimum Cost ($/h)	1132.30	1140.06	1147.77
Maximum Cost ($/h)	1132.71	1141.57	1151.37
Average Cost ($/h)	1132.32	1140.65	1148.58
Standard Deviation (%)	0.001	0.04	0.07
Feasible (%)	100	100	100
Stable (%)	0	100	100

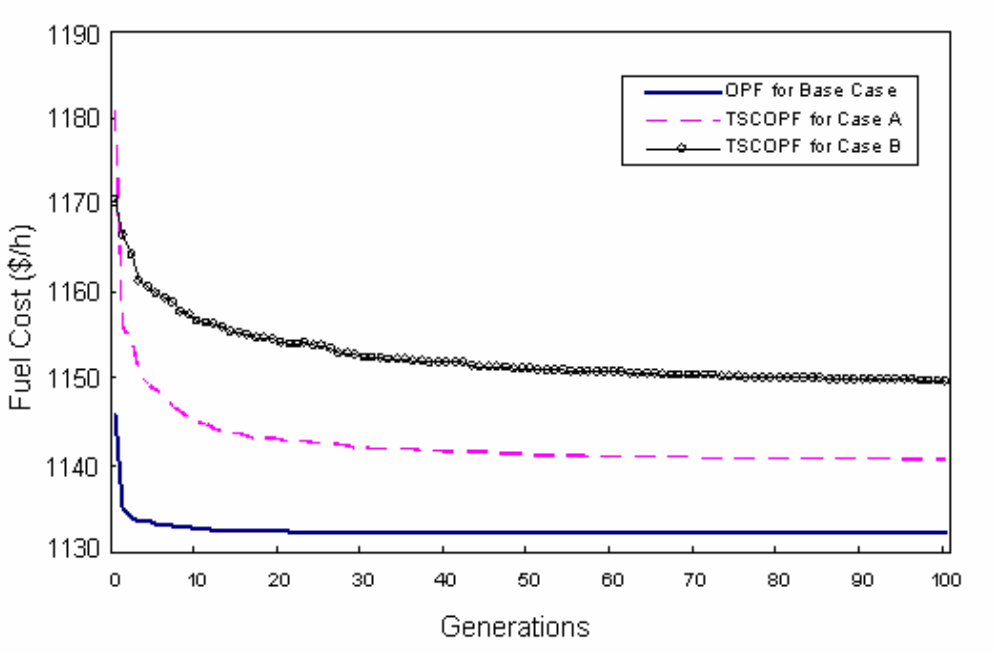

Fig. 3.6 Average convergence curves of the WSCC 3-generator, 9-bus system

Table 3.2 Best solutions of the WSCC 3-generator, 9-bus system

Case	Base OPF	A	B
G1 (MVA)	105.94+j17.14	130.94-j9.63	130.01+j19.39
G2 (MVA)	113.04+j4.92	94.46+j9.22	127.17+j7.34
G3 (MVA)	99.29-j15.31	93.09+j24.77	60.72-j18.34
V1 (p.u.)	1.050	0.9590	1.0495
V2 (p.u.)	1.050	1.0139	1.0481
V3 (p.u.)	1.040	1.0467	1.0327
Cost ($/h)	1132.30	1140.06	1148.58

Table 3.3 Comparison of CCT for cases A and B

Case	A	B
Fault Clearing Time (ms)	350	300
CCT of Initial Condition (ms)	162	214
CCT of Base OPF Solution (ms)	293	219

CCT of TSCOPF Solution (ms)	398	376

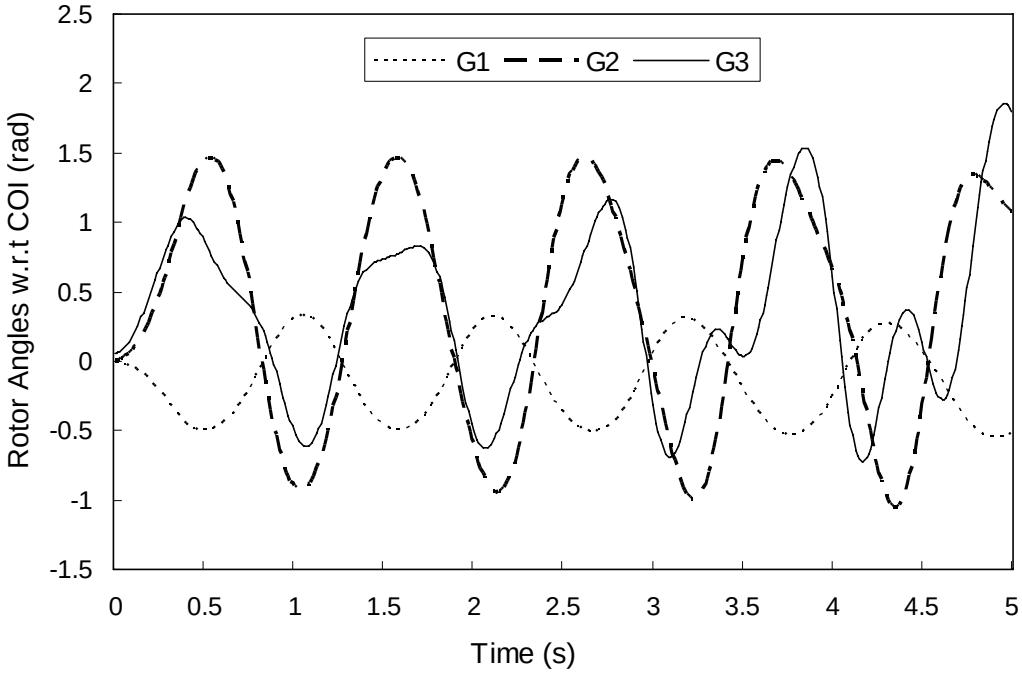

Fig. 3.7 Stable trajectory of rotor angles of TSCOPF solution for case A

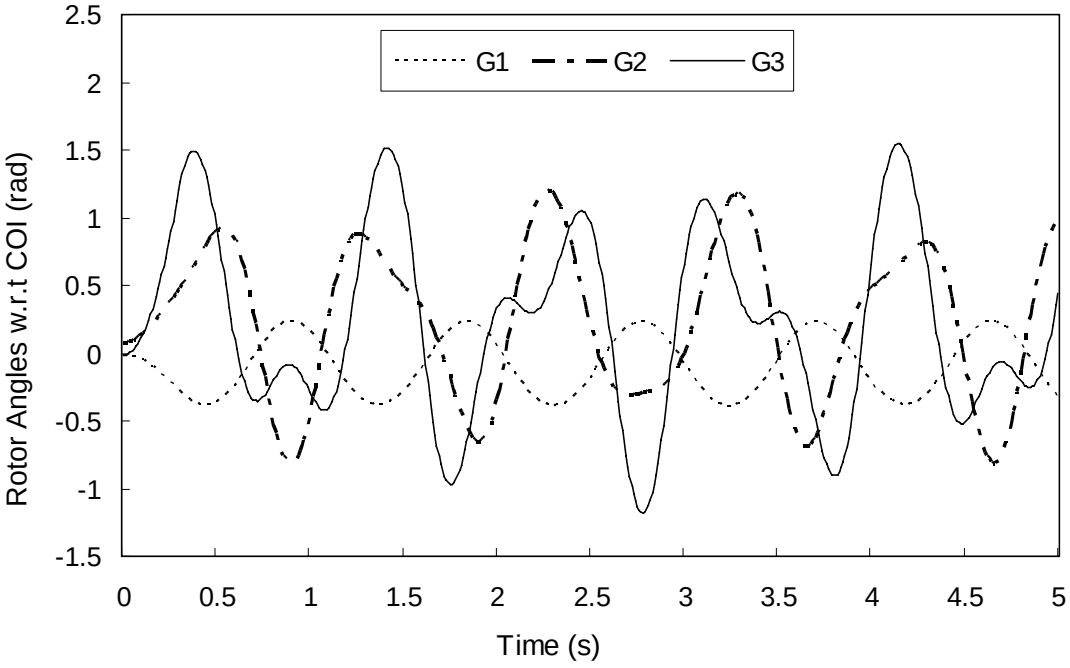

Fig. 3.8 Stable trajectory of rotor angles of TSCOPF solution for case B

3.5 Simulation on the New England 10-generator, 39-bus System

The single line diagram of this system is shown in Fig. 3.9. System data is available in [Sauer, Pai, 1998]. The fuel cost parameters and the ratings of generators are taken from [Nguyen, Pai, 2003]. The lower and upper limits of all bus voltage magnitudes are as those in the MATPOWER software package [Zimmerman, Gan, 1997]. There are 19 control variables, including 9 generator active power outputs and 10 generator voltages. We set the population size as 60 (i.e. almost three times of the number of control variables); and the maximum generation number is 100. The following two contingency cases are studied:

Case C: A three phase to ground fault at bus 4 and cleared by tripping line 4-5 at 250 ms, which is greater than the initial CCT 222 ms.

Case D: A three phase to ground fault at bus 21 and cleared by tripping line 21-22 at 160 ms, which is greater than the initial CCT 144 ms.

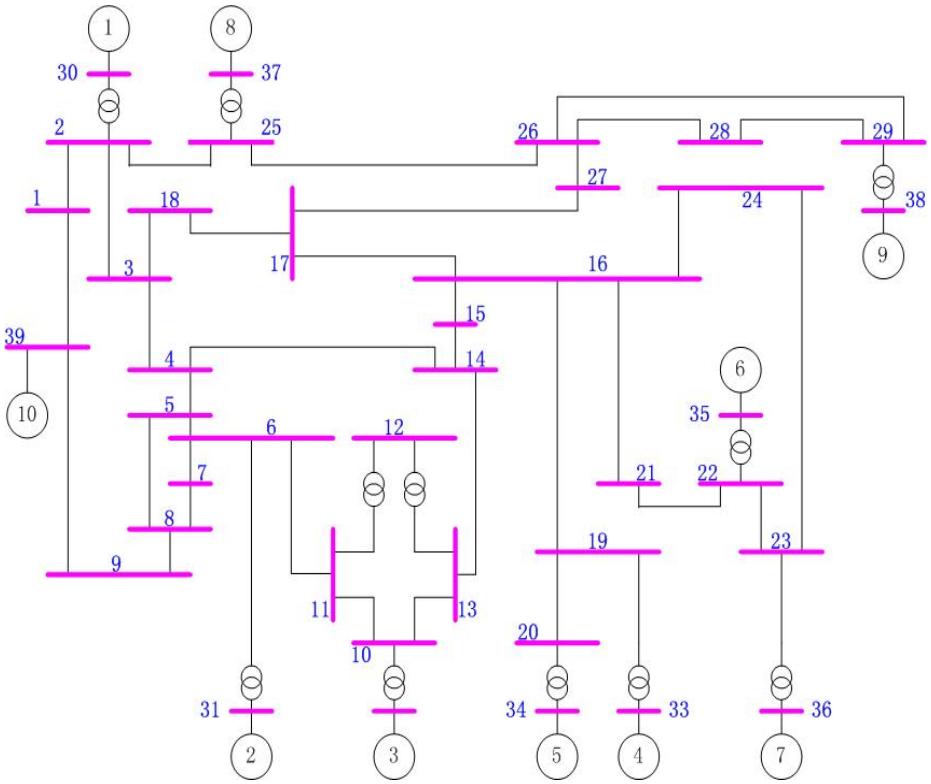

Fig. 3.9 New England 10-generator, 39-bus test system

In this study, the effectiveness of the proposed TSCOPF method under different percentages of the population selected for the TSA is studied. The average convergence curves corresponding to Case C are shown in Fig. 3.10 as an example. The fuel costs and computation times for different percentages of population for TSA are shown in Fig. 3.11. It can be observed that the proposed method converges very well for different percentages and it can find out stable solutions even only 20% of the whole population is selected to undergo TSA. Obviously, higher percentage can find a better system condition, but this effect is reduced when the percentage becomes large enough. Meanwhile, the computation time is almost linear to the percentage since TSA is the most time-consuming part of the program. Therefore, a good compromise between the cost and computation

time can be achieved according to their relationship. In this study, 40% percentage of the population for TSA is selected.

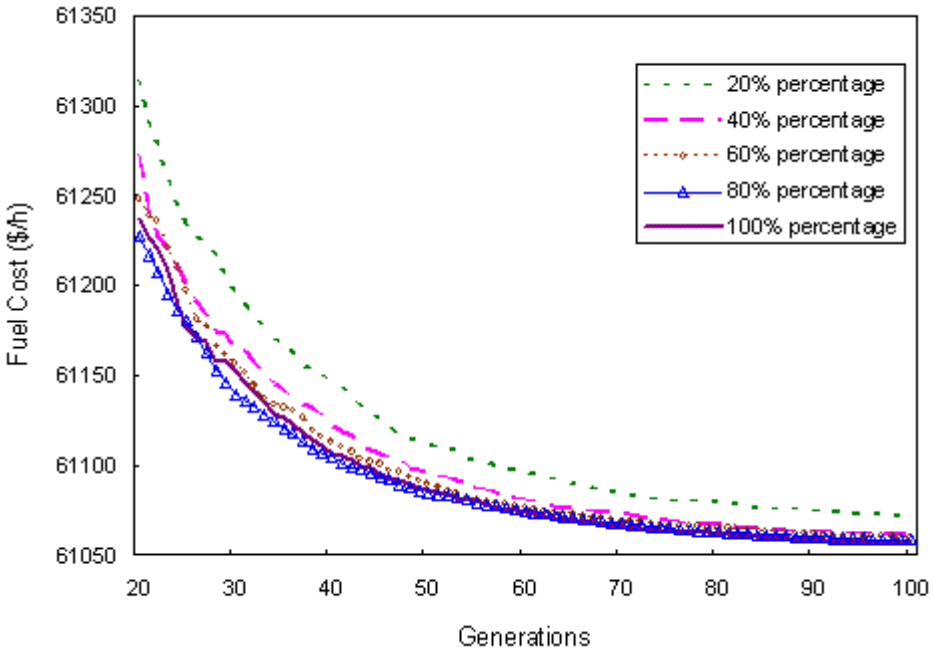

Fig. 3.10 Average convergence curves for different percentages of population undergoing TSA in case C

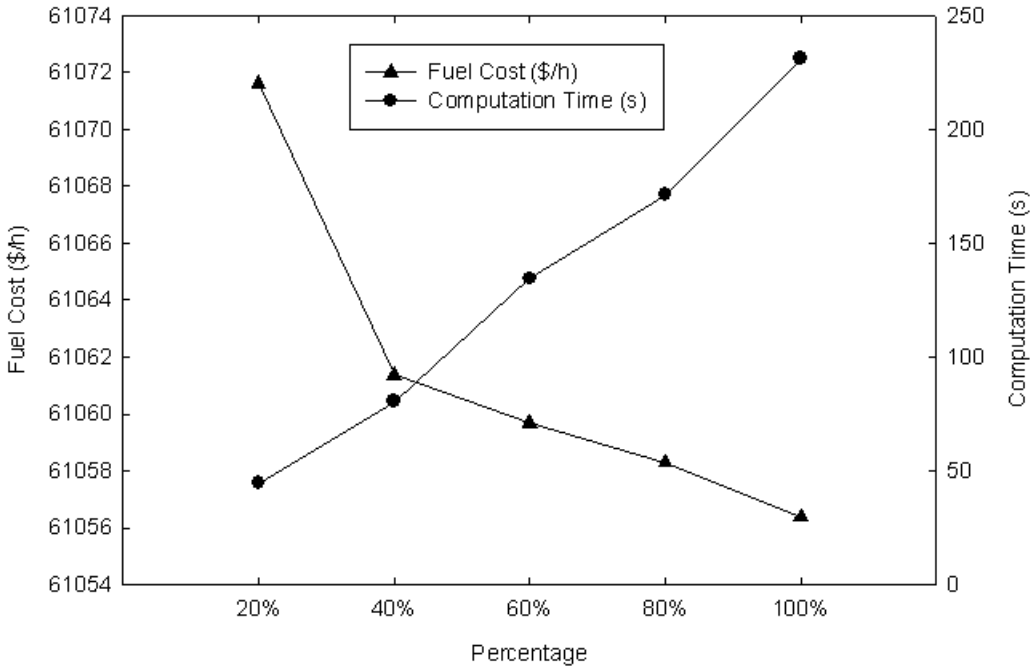

Fig. 3.11 Fuel cost and computation time versus different percentages of population for TSA in case C

The average convergence curves of Cases C and D are shown in Fig. 3.12, from which we can observe that both cases converge very well. In the beginning part of the converge curves, the fuel costs may increase during several generations as shown in Fig. 3.12 and it is a consequence of the DE activity to eliminate the violations and insecurities.

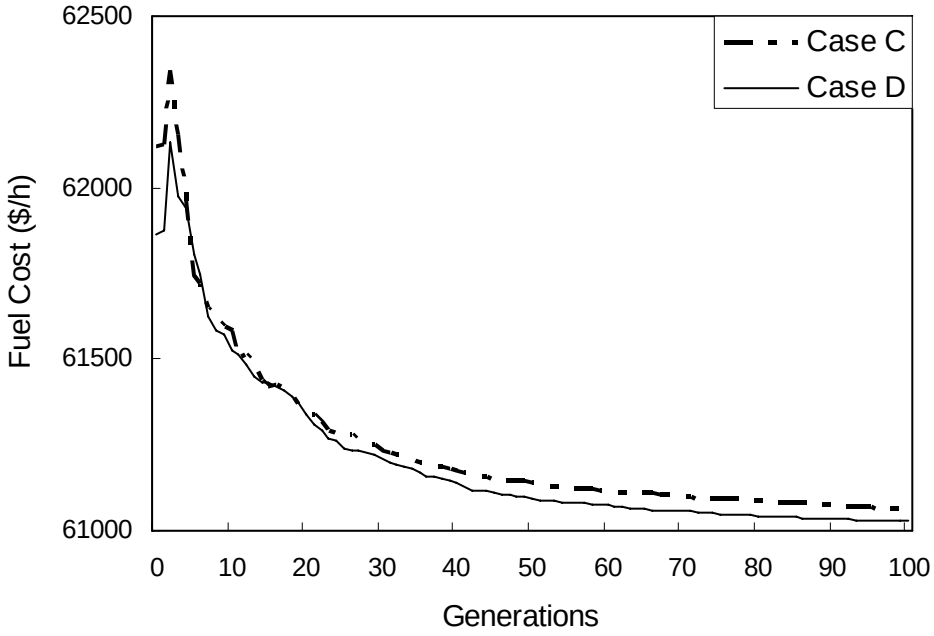

Fig. 3.12 Average convergence curves of the New England 39-bus system

TSCOPF simulation results are summarized in Table 3.4. Comparisons of the fuel costs of the best solutions for different cases are listed in Table 3.5. The system conditions of the best solutions for different cases are listed in Table 3.6. Small standard deviation of different cases has verified the robustness of the proposed methods again.

For the Reported Results in Table 3.5, Cases C is the result reported in [Nguyen, Pai, 2003] which used transient sensitivities to reschedule the generator outputs to maintain system stability; Cases D is the result reported in [Dawn, Jeyasurya, 2004], which converted the dynamic differential equations into equivalent algebraic equations to solve the TSCOPF problem. Obviously, the stable solutions obtained by DE method are much economical than the conventional methods. As we know, conventional methods are local search

methods and only able to search the local optimal solution, which meets the transient stability constraints, in local region; while DE searches the solution in the global search space and avoids to be trapped by a local optimum. The simulation results thus have verified that RDE has a much stronger searching ability than conventional mathematical methods.

Table 3.4 Simulation results of the New England 39-bus system

Case	C	D
Minimum Cost ($/h)	61021.04	60988.25
Maximum Cost ($/h)	61107.62	61068.87
Average Cost ($/h)	61061.35	61027.57
Average Time (s)	86.61	91.86
Standard Deviation (%)	0.05	0.04
Feasible (%)	100	100
Stable (%)	100	100

Table 3.5 Comparisons of fuel cost for cases C and D

Case	Base OPF ($/h)	TSCOPF ($/h)	Reported Results ($/h)
C	60936.51	61021.04	61826.53
D	60936.51	60988.25	62263.00

Table 3.6 Best solutions of the New England 10-generator, 39-bus system

Case	C	D	Case	C	D
G1 (MW)	251.97	237.06	V1 (p.u.)	0.9881	0.9925
G2 (MW)	540.70	587.35	V2 (p.u.)	1.0457	1.0284

G3 (MW)	587.96	668.64	V3 (p.u.)	1.0292	1.0139
G4 (MW)	643.11	634.92	V4 (p.u.)	0.9734	0.9876
G5 (MW)	509.93	493.77	V5 (p.u.)	1.0164	1.0301
G6 (MW)	639.09	619.79	V6 (p.u.)	1.0600	1.0456
G7 (MW)	540.30	514.00	V7 (p.u.)	1.0009	1.0455
G8 (MW)	565.09	542.87	V8 (p.u.)	0.9974	1.0391
G9 (MW)	839.99	837.03	V9 (p.u.)	1.0284	1.0249
G10 (MW)	1023.96	1003.93	V10 (p.u.)	1.0389	0.9836
Cost ($/h)	61021.04	60988.25			

Comparisons of CCT for different cases are listed in Table 3.7. Once again, the base OPF solution can schedule the system in a more economical condition but with weak stability performance, while the proposed DE based TSCOPF method is able to not only ensure the system stability, but also schedule the system more economically than other reported TSCOPF methods. Stable trajectories of rotor angles with respect to COI coordinate of the best solutions for Cases C and D are shown in Fig. 3.13 and Fig. 3.14, respectively.

Table 3.7 Comparisons of CCT for cases C and D

Case	C	D
Fault Clearing Time (ms)	250	160
CCT of Initial Condition (ms)	222	144
CCT of Base OPF Solution (ms)	224	145
CCT of TSCOPF Solution (ms)	260	169

The relationship between the fuel cost and the requirement of the FCT is also studied. Curves of optimal fuel cost versus different FCTs for different cases are shown in Fig. 3.15. It is found that an optimal solution can be found when FCT is reasonable. As the requirement of FCT increases and becomes critical, the fuel cost increases rapidly. However, the system will have no stable solution up to certain level of FCT requirement and this is represented by the dashed line in the figure.

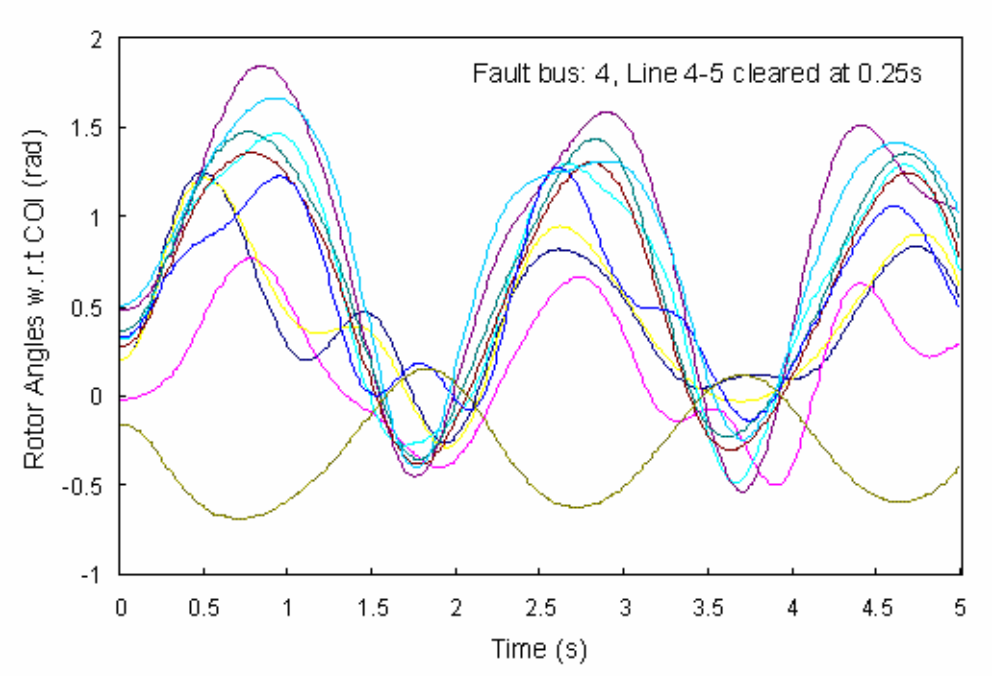

Fig. 3.13 Stable trajectory of rotor angles for case C

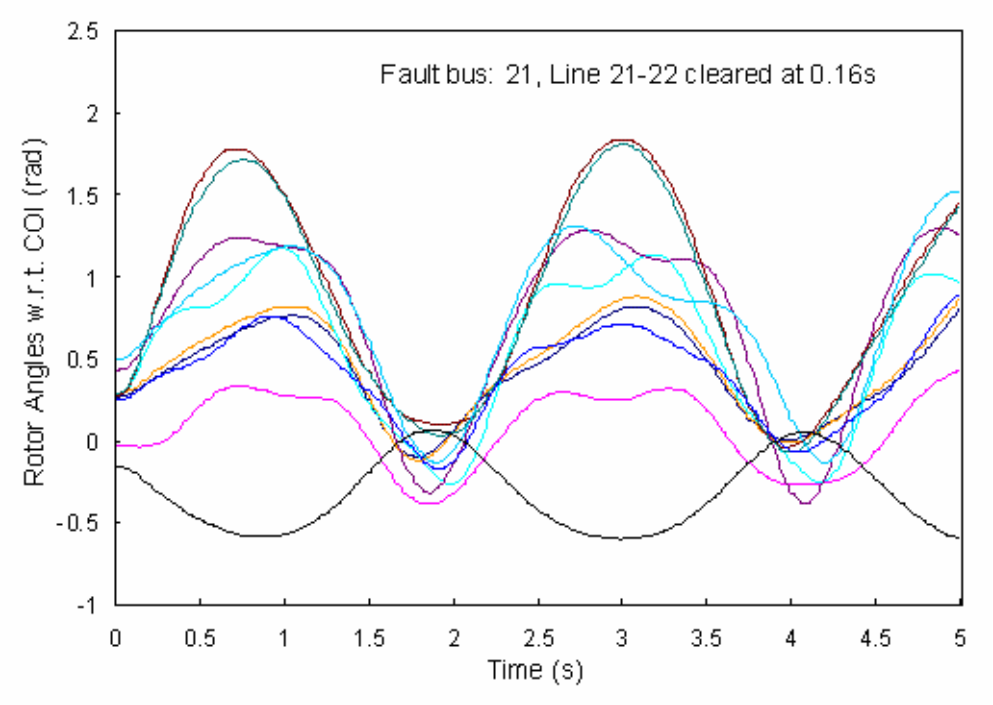

Fig. 3.14 Stable trajectory of rotor angles for case D

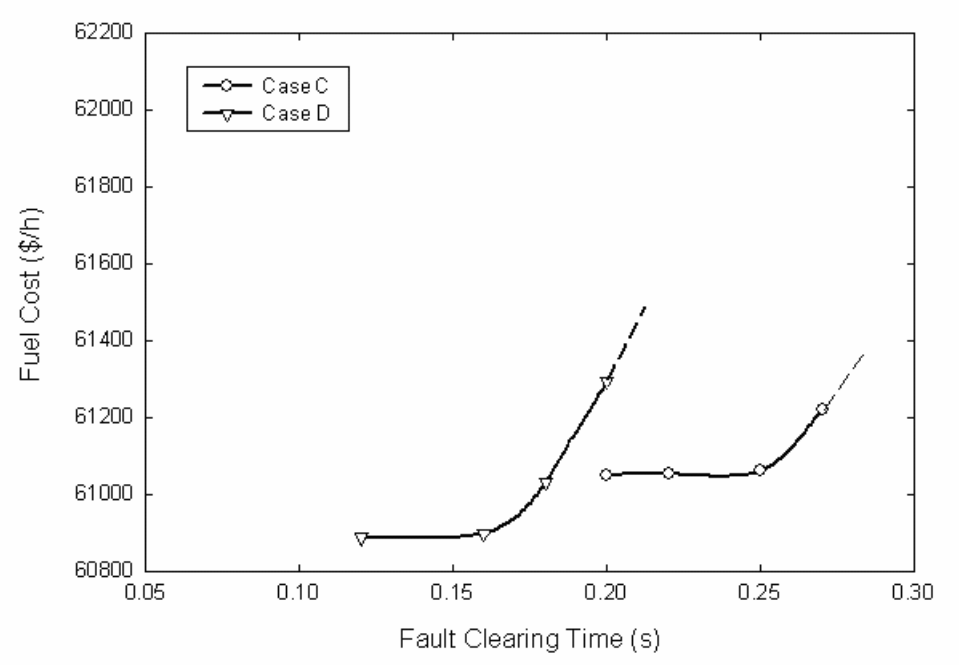

Fig. 3.15 TSCOPF solution versus different fault clearing time

3.6 Summary

In this chapter, the proposed RDE is applied to solve the TSCOPF problem. It is found that the proposed method is not only able to ensure the transient stability performance of the system, but also determine a lower fuel cost solution compared with other reported results in the literature. The robustness and effectiveness of the proposed method have also been verified based on the simulation results. Besides, impact of the fault clearing time requirement on the total system fuel cost is investigated by this new powerful tool.

Chapter 4 Multi-Constrained Discrete TSCOPF

4.1 Introduction

The researches in last chapter have investigated the strong ability of DE in continuous transient stability constrained optimal power flow problem. However, the power systems can not be adjusted smoothly in practice as expected. Some control variables such as the transformer tap settings are discrete; generators have prohibitive operation zones (POZ), valve-point effects. The continuous TSCOPF problem itself is a nonlinear and non-convex problem which is hard to be handled by conventional methods, these practical operation constraints make it more complicated. With these constraints, the decision space of the TSCOPF problem is divided into disjoint sub-regions which make the problem numerically discontinuous and non-differentiable. As a result, the conventional numerical approach cannot be applied directly. Applications of EAs in solving economic dispatch problems with all kinds of generator constraints have proved their power in handling nonlinear, non-convex, non-smooth, non-differential and discontinuous problems [Gaing, 2003; Park, Lee, etc., 2005; Chiang, 2005; Liu

and Cai, 2006; Coelho and Mariani, 2006].

In this chapter, the TSCOPF problem with multi practical operation constraints is resolved using DE algorithm to examine its behavior in handling discontinuous and non-differentiable optimization problems.

4.2 Practical Operation Constraints

Practical operation constraints considered in this chapter are introduced as below.

(1) Valve Point Effects of Generator

In most of research works, the cost function of each unit has been approximately represented by a single quadratic cost function. Conventional methods based on derivation need these simplicities to obtain solutions. However, in reality, large steam turbines have steam admission valves, the valve-point effect is the rippling effect added to the generating unit curve when each steam admission valve in a turbine starts to open. Thus, the input–output characteristics of the generating units will become non-convex and furthermore they may generate multiple local optimum points in the cost function. Therefore, the cost function is normally described as the superposition of a sinusoidal function and a quadratic function as below.

$$f_i = a_i + b_i P_{gi} + c_i P_{gi}^2 + \left| d_i \sin(e_i (P_i^{\min} - P_i)) \right| \tag{4.1}$$

where a_i, b_i, and c_i are the fuel consumption cost coefficients of unit i, d_i and

e_i are fuel cost coefficients of the *i*th unit with valve-point effects.

(2) Prohibited Operating Zones of Generator

For convenience, the unit generation output is usually assumed to be adjusted smoothly. Practically, there are prohibited operating zones in the input-output curve of generator due to steam valve operation or vibration in a shaft bearing. In practical operation, adjusting the generation output P_{gi} of a unit must avoid these prohibited zones. The feasible operating zones of unit i can be described as follows:

$$\begin{aligned} & P_{gi}^{\min} \le P_{gi} \le P_{gi,1}^{l} \\ & P_{gi,j}^{u} \le P_{gi} \le P_{gi,j+1}^{l}, \qquad j = 1,2,...,(N_z - 1) \\ & P_{gi,N_z}^{u} \le P_{gi} \le P_{gi}^{\max} \end{aligned} \tag{4.2}$$

where $P_{gi,j}^{u}$ and $P_{gi,j}^{l}$ are upper and lower bounder of the *j*th prohibited zone of unit *i*; N_z is the total number of prohibited zones in unit *i*.

(3) Discrete Control Region of Transformers

Adjusting transformer tap settings is a common way in power system control. Most transformers in power systems are discontinuously variable transformer, values of the transformer ratios are within their available sets as below:

$$T_i \in \{T_i^{\min}, \cdots, T_{ik}, \cdots, T_i^{\max}\}, \qquad i = 1, \ldots, N_t \tag{4.3}$$

where T_i is the tap setting of transformer i, $T_i^{\min}$ and $T_i^{\max}$ are minimum and maximum tap settings, T_{ik} is one of the valid tap settings, and N_t is the total number of adjustable transformers.

(4) Static Thermal Constraints

$$\left|S_i\right| \le S_i^{\max}, \qquad i = 1,\ldots,N_l \tag{4.4}$$

where N_l is the total number of branches; and S_i is the apparent power flow in branch i, $S_i^{\max}$ is the upper limit of branch flow apparent power.

4.3 Multi-constrained discrete TSCOPF

The implementation of the multi-constrained TSCOPF problem differs not much with that of the continuous TSCOPF studied in last chapter. However, a few questions have to be mentioned here.

4.3.1 Handling of Discrete Variables

As introduced above, control variables like transformer tap settings and generator active power outputs are discrete variables now. During the optimization process, we have to take some measures to handle this. Fortunately, discrete control variables influence only the initialization and reproduction stages of DE. When these two operations occur, discrete variables are firstly treated as continuous variables within their lower and upper boundaries, and then will be rounded off the nearest possible values.

4.3.2 Fitness Evaluation Function

The fitness function has to make some modifications in the multi-constrained TSCOPF. Violation of static thermal constraint of individual is added into

punishment items in the fitness function.

$$F_i = 1/(f_i + K_V F_{Vi} + K_Q F_{Qi} + K_{PS} F_{PS} + K_{BF} F_{BF}) \tag{4.5}$$

$$F_{BFi} = \sum_{j=1}^{N_l} (|S_{ij} - S_j^{\max}|) / S_j^{\max} \tag{4.6}$$

where F_{BFi} is the sum of violation of branch flow limitations of individual i, and K_{BF} is the corresponding penalty coefficient.

4.4 Case Studies

The DE based multi-constrained TSCOPF method is tested on the New England 10-machine, 39-bus system. Lower and upper limits for transformer tap settings are 0.90 and 1.10 p.u., respectively, and the adjustment step is 0.01 p.u.. Branch flow apparent power limits is set at 800MVA for each line, and branch 29-38 has a flow of 852.61MVA which violates the limits. There are 31 control variables, including 9 generator active power outputs, 10 generator voltage magnitudes and 12 transformer tap settings. A three phase to ground fault is applied at bus 21 and cleared by tripping line 21-22 at 0.16s. Transient simulation time step is set as 0.01 second, the whole simulation period is 3.0 second. CCT of the test system under original system configuration is 144ms which is less than the fault clearing time. The population size is set at 90, and the maximum generation number is 100. Re-initialization period is set as 10 generations. Totally 20 trial runs are performed.

4.4.1 Case A: Discrete Optimization Validation

Before all the TSCOPF simulations, basic OPF is run on both MATPOWER [Zimmerman and Gan, 1997] software without transformer tap adjustment and DE method with transformer tap adjustment. Temporarily, generator valve-point effects, generator POZ constraints, and branch flow constraints are all not considered, since MATPOWER solves OPF by conventional methods, and discrete constraints can not be handled. The results are compared to verify the ability of DE in handling discrete control variables.

The minimum fuel cost obtained by MATPOWER is 61761.68$/h. DE simulation results are listed in Table 4.1. The best system solutions are listed in Table 4.2. The average fuel cost obtained from the 20 trials is 60916.11$/h, which saved 1.37% more fuel cost than the MATPOWER solution. It is obviously that DE has stronger ability to handle discrete control variables and converges to more optimal solution than conventional method embedded in the MATPOWER. Moreover, DE takes 3.95 seconds to reach the solution while MATPOWER takes 4.78 seconds under the same computing conditions.

Table 4.1 Simulation results for case A

Minimum Cost ($/h)	60907.65
Maximum Cost ($/h)	60928.04
Average Cost ($/h)	60916.11
Standard Deviation (%)	0.007
Feasible (%)	100

Stable (%)	100
ACT (s)	3.95

Table 4.2 Best system solutions for case A

Unit	Generation (MW)	Unit	Voltage (p.u.)
G30	234.30	G30	1.0443
G31	563.77	G31	1.0164
G32	634.69	G32	0.9993
G33	628.97	G33	1.0023
G34	509.35	G34	1.0197
G35	654.36	G35	1.0441
G36	562.58	G36	1.0539
G37	536.52	G37	1.0592
G38	830.45	G38	1.0594
G39	983.94	G39	1.0485
Branch	Tap Settings (p.u.)	Branch	Tap Settings (p.u.)
12-11	1.02	22-35	1.03
12-13	1.01	23-36	1.03
6-1	1.10	25-37	1.00
10-32	1.10	2-30	1.00
19-33	1.08	29-38	1.00
20-34	0.99	19-20	1.06
Fuel Cost ($/h)		60907.65	
CCT (ms)		127	

4.4.2 Case B: TSCOPF with Valve-point Effects

In this section, valve-point effects are included into the TSCOPF problem. The fuel cost coefficients with valve-point effects are listed in Table 4.3.

Table 4.3 Unit fuel cost coefficients with valve-point effects

Unit	*a*	*b*	*c*	*d*	*e*
30	0.0193	6.900	0.000	100.000	0.084
31	0.0111	3.700	0.000	150.000	0.063
32	0.0104	2.800	0.000	200.000	0.042
33	0.0088	4.700	0.000	150.000	0.063
34	0.0128	2.800	0.000	150.000	0.063
35	0.0094	3.700	0.000	150.000	0.063
36	0.0099	4.800	0.000	150.000	0.063
37	0.0113	3.600	0.000	150.000	0.063
38	0.0071	3.700	0.000	200.000	0.042
39	0.0064	3.900	0.000	250.000	0.036

Table 4.4 shows the minimum, maximum and average fuel costs obtained among all 20trials. The best system solutions are listed in Table 4.5. The average convergence curve is shown in Fig. 4.1. As seen in Table 4.4, all trials are managed to converge to a feasible and stable solution. Standard deviation of all solutions is small which proves the robustness of the proposed method. The best solution with a minimum cost of 61514.18 $/h has a CCT of 197 ms which is longer than the fault clearing time. Fig. 4.2 shows the stable trajectory of generator rotor angles

under COI coordinate. Branch 29-38 has a branch flow of 865.03MVA which is large than 800MVA, other branches are within the limits.

Table 4.4 Simulation results for case B

Minimum Cost ($/h)	61514.18
Maximum Cost ($/h)	61757.21
Average Cost ($/h)	61598.48
Standard Deviation (%)	0.13
Feasible (%)	100
Stable (%)	100
ACT (s)	136.39

Table 4.5 Best System Solutions for Case B

Unit	Generation (MW)	Unit	Voltage (p.u.)
G30	295.57	G30	1.0084
G31	600.87	G31	0.9922
G32	672.96	G32	1.0204
G33	600.15	G33	0.9930
G34	495.44	G34	1.0018
G35	557.80	G35	0.9737
G36	502.54	G36	1.0366
G37	546.13	G37	1.0132
G38	823.64	G38	0.9647
G39	1045.99	G39	1.0409
Branch	Tap Settings (p.u.)	Branch	Tap Settings (p.u.)
12-11	1.02	22-35	1.04
12-13	0.98	23-36	1.05

6-1	0.99	25-37	1.02
10-32	1.04	2-30	1.02
19-33	0.97	29-38	1.04
20-34	1.00	19-20	0.98
Fuel Cost ($/h)		61514.18	
CCT (ms)		197	

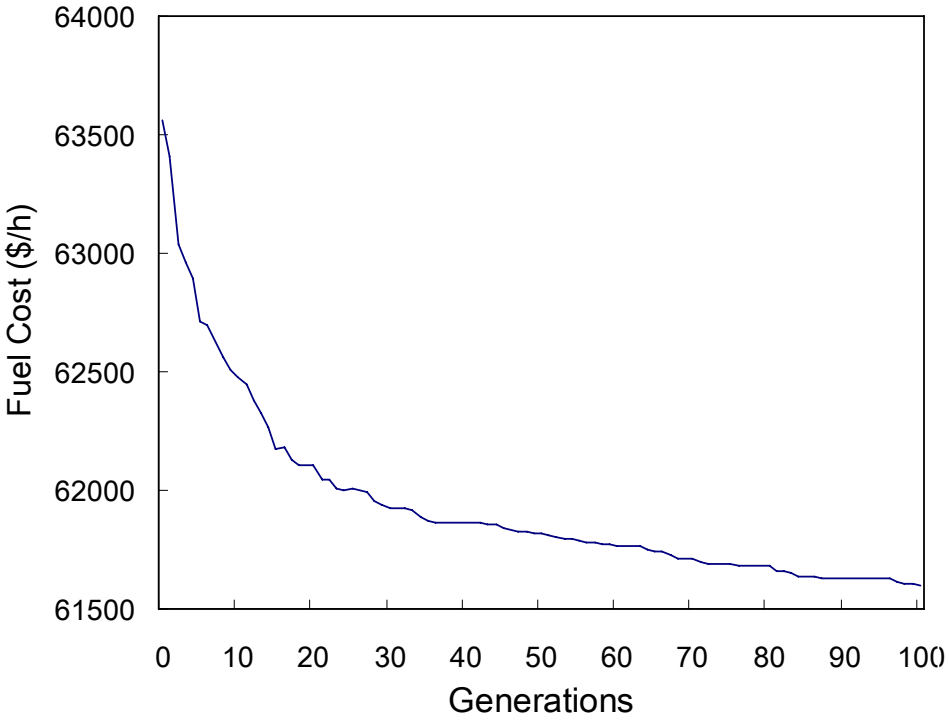

Fig. 4.1 Average converge curve for case B

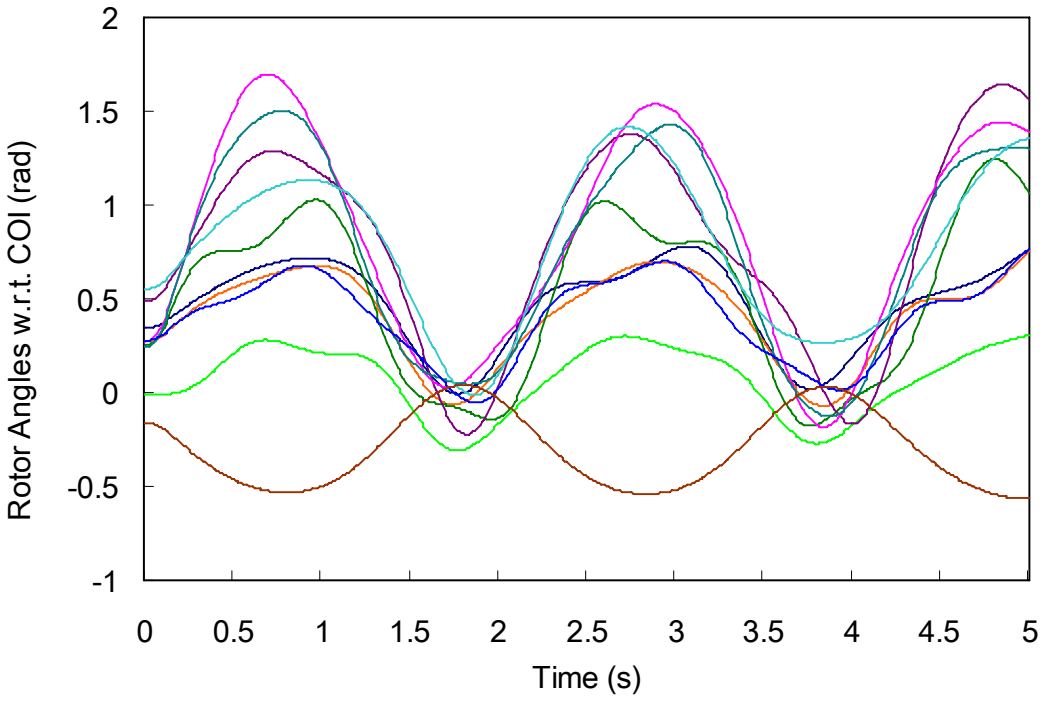

Fig. 4.2 Stable trajectories of rotor angles for case B

4.4.3 Case C: Multi-Constrained OPF

All practical operation constraints mentioned in this chapter are finally incorporated to check the adaptability of the proposed method. Three units are selected to have POZs. The generator prohibited operating zones data are listed in Table 4.6. These POZs are designed to locate the unit active power output solutions in last section into the POZ regions.

Table 4.6 Prohibited operating zones of New England 39-bus system

Unit	$P_i^{\min}$ (MW)	$P_i^{\max}$ (MW)	Prohibited Zones (MW)

30	0	350	[200, 300]
32	0	800	[400, 500] [600, 700]
39	0	1200	[700, 800] [900, 1100]

Simulation results are summarized in Table 4.7 and best system solutions are listed in Table 4.8. Fig. 4.3 and Fig. 4.4 illustrate the average converge curve and stable rotor angle trajectory of case C.

As seen from the results, generation power solutions of those units have POZ have been moved away from the POZ to feasible regions. Branch flow in line 29-38 has been decreased to 752.53MVA without any other branch flow violations. Though the mean fuel cost is slightly large than that in Case B, solutions obtained in this case are more meaningful to practical operations.

Table 4.7 Simulation results for case C

Minimum Cost ($/h)	61527.30
Maximum Cost ($/h)	61696.31
Average Cost ($/h)	61639.44
Standard Deviation (%)	0.11
Feasible (%)	100
Stable (%)	100
ACT (s)	138.56

Table 4.8 Best system solutions for case C

Unit	Generation (MW)	Unit	Voltage (p.u.)

G30	300.51	G30	1.0252
G31	599.42	G31	1.0462
G32	598.96	G32	1.0187
G33	599.24	G33	0.9754
G34	501.32	G34	1.0234
G35	600.33	G35	0.9982
G36	503.17	G36	1.0236
G37	550.05	G37	0.9835
G38	74.84	G38	0.9835
G39	1135.44	G39	0.9867
Branch	Tap Settings (p.u.)	Branch	Tap Settings (p.u.)
12-11	1.02	22-35	1.01
12-13	0.96	23-36	1.05
6-1	1.07	25-37	1.06
10-32	0.98	2-30	0.96
19-33	1.00	29-38	1.01
20-34	0.97	19-20	0.97
Fuel Cost ($/h)		61527.30	
CCT (ms)		180	

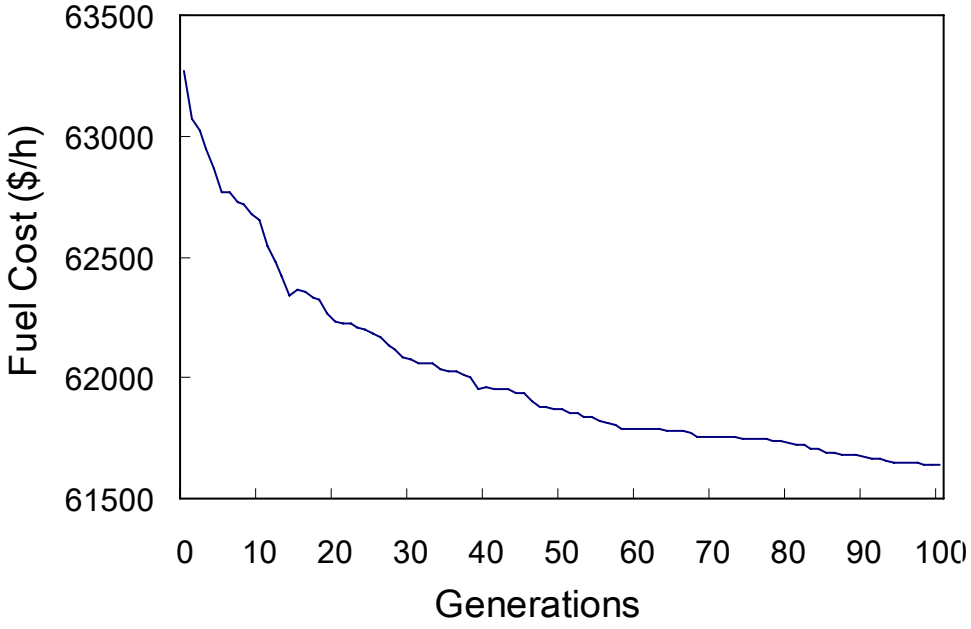

Fig. 4.3 Average converge curve for case C

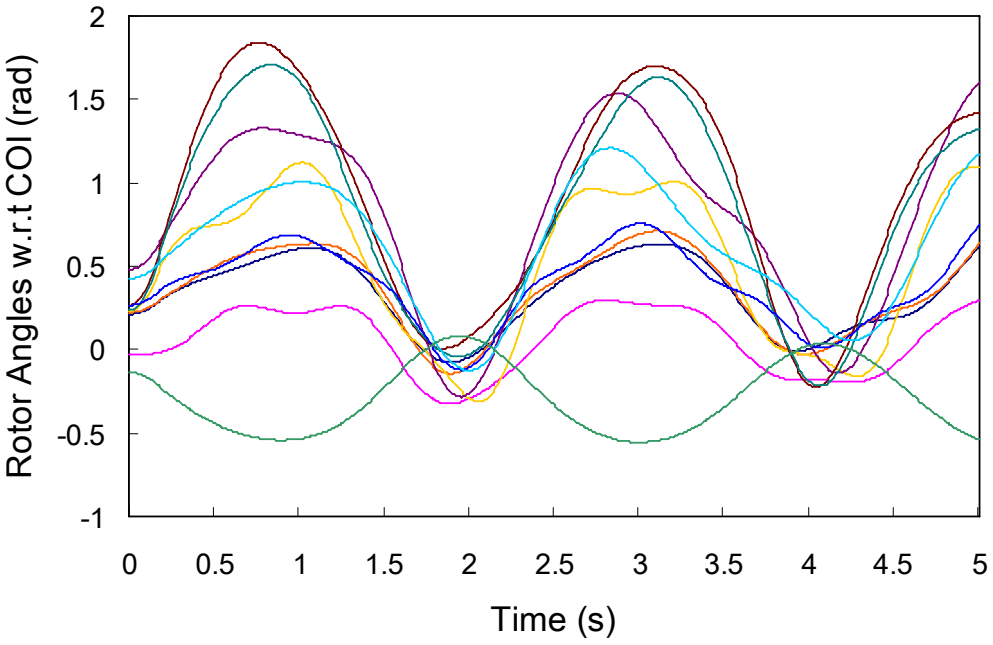

Fig. 4.4 Stable trajectories of rotor angles for case C

4.5 Summary

This chapter extends and improves continuous TSCOPF to multi-constrained OPF problem. Transient stability constraints, generator prohibited operating zones, and other practical operation constraints are all included as constraints to the OPF problem which makes the problem a particularly complex one. Fortunately, simulation results show the proposed method is very effective and robust in handling this nonlinear, non-smooth, non-convex, and non-differentiable optimization problem.

The computational time is acceptable for the New-England 39-bus system, however, the searching process may become slow in practical system where more contingencies have to be considered and the system size is much larger. To improve the proposed method to practical utility, parallel computation technology will be introduced in the next chapter.

Chapter 5 Parallel Computation

5.1 Introduction

The studies in last two chapters show that DE is effective and robust in solving the TSCOPF problem. However, transient stability assessment is a CPU intensive task especially when system becoming large. Fortunately, the development in the world of modern computing offers the possible way of parallel computation to decrease the computation time of engineering problem and meet the need of practical application.

As one of the evolutionary algorithms, DE is intrinsically a parallel searching algorithm. Thus, it is very suitable for parallel computation. So far, researches about parallel evolutionary algorithm on OPF problem are inadequate. This chapter will parallelize the RDE based TSCOPF problem on a Beowulf PC-cluster platform. The construction of parallel platform is firstly introduced; details about parallelization of RDE-TSCOPF are described consequently; and the characteristics and performance of parallel RDE are investigated through OPF simulations on IEEE 118-node system.

5.2 Parallel Computation and PC-cluster

5.2.1 Structures of parallel computer systems

Parallel computation can be utilized in different levels in computer systems, such as task level, instruction level, or hardware level [Alba & Tomassini, 2002]. According to the number of instruction streams and data streams, Michael Flynn divided computer systems into four kinds of architecture in 1966: Single Instruction Single Data Stream system (SISD), Single Instruction Multiple Data Stream system (SIMD), Multiple Instruction Single Data Stream system (MISD), and Multiple Instruction Multiple Data Stream system (MIMD). Although there is great development in computer systems, this kind of division is still widely accepted today. SISD refers to serial computer system corresponds to ordinary computer or station with single CPU. The other three architectures belong to parallel architecture. In reality, MISD is seldom used; while SIMD and MIMD are most used parallel architectures. The structure of SIMD and MIMD systems are illustrated in Fig. 5.1 and Fig. 5.2.

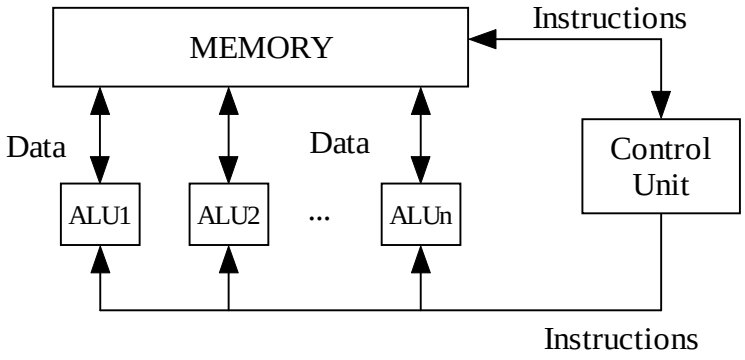

Fig. 5.1 Illustration of the SIMD architecture

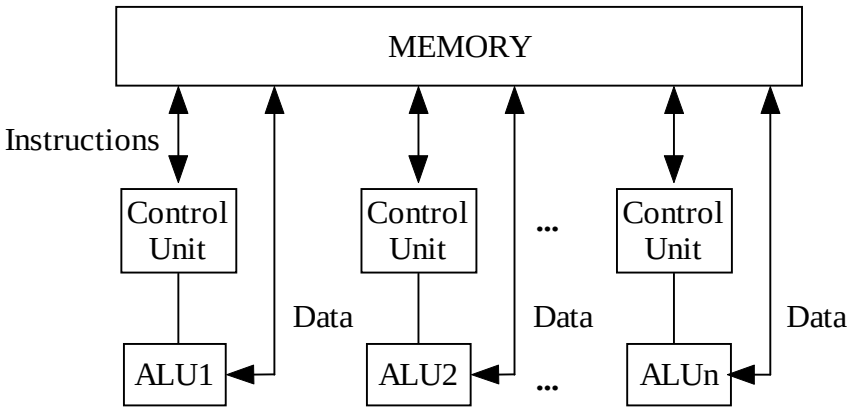

Fig. 5.2 Illustration of the MIMD architecture

In SIMD system, there is only one control unit and several algorithm logic units (ALU) as slaves. In each instruction cycle, the control unit will broadcast an instruction to all slave ALUs; each ALU may execute the instruction or idle for new instruction. The SIMD system runs in a precise synchronous mode. When data structure is inerratic, this parallel system has the advantage of easy coding; however, when data structure is not regular or the program has many branches, the whole efficiency of this parallel system will slow down with many ALUs' idling.

In MIMD system, each CPU is autonomous with both control unit and ALU. Each CPU can run the program in its own pace unless the program demands them to run synchronistically. Thus, the MIMD system is synchronous system. MIMD system is more flexible than SIMD system and is the most widely used system. Generally, there are two kinds of MIMD architectures: shared memory architecture and distributed architecture as shown in Fig. 5.3 and 5.4.

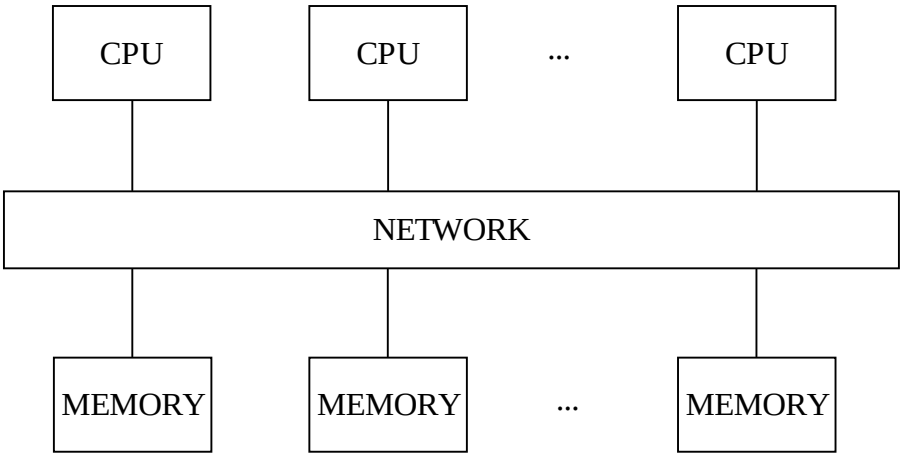

Fig. 5.3 Shared memory MIMD architecture

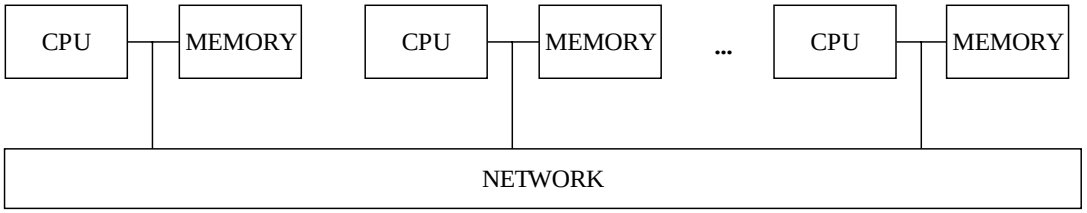

Fig. 5.4 Distributed memory MIMD architecture

5.2.2 Beowulf PC-cluster

Cluster is a powerful concept and technique for driving extended capabilities from exiting classed of component. In the field of computing system, cluster are ensembles of independently operational computers integrated by means of an interconnection network and supporting user-accessible software for organizing and controlling concurrent computing tasks that may cooperate on a common

application program or workload [Sterling, 2002]. Nowadays, it is among the first class computer system architecture techniques for achieving significant improvements in overall performance. Obviously, it belongs to the distributed memory MIMD structure.

Beowulf is a class of cluster that may be implemented by the end users themselves from available components. It exploits mass-market PC hardware and software in conjunction with cost-effective commercial network technology. Key advantages of this approach are high performance for low price, system scalability, and rapid adjustment to new technology advances. Nowadays, Beowulf cluster are becoming the main platform for many scientific, engineering, and commercial applications.

5.3 Build a Beowulf PC-cluster

5.3.1 Hardware Construction

Hardware construction of a Beowulf cluster is simple and convenient as demonstrated in Fig. 5.5. A host node (computer) is used to control the cluster, write and run programs, etc. Work nodes are connected to the host node via one or more Ethernet switches according to the work node number and switch port number.

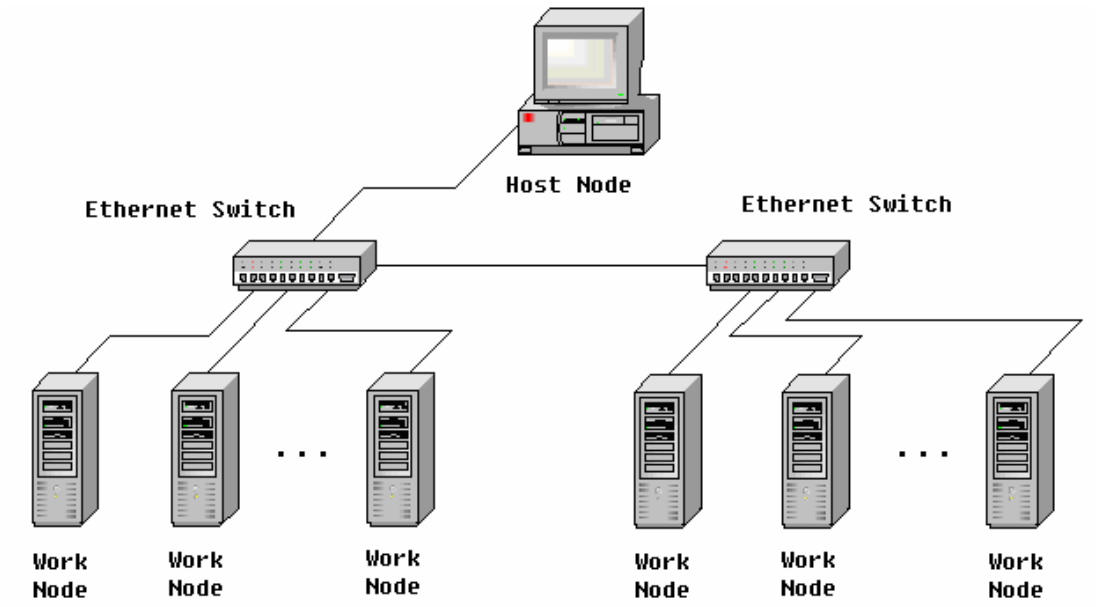

Fig. 5.5 Configuration of the Beowulf cluster used in this research

5.3.2 Network configurations

Before the cluster starts to work, configurations have to be made on networks before the cluster starting to work. These configurations mainly include protocol settings and security settings.

(1) IP address setting

After operating system Linux (Red Hat 9.0) installation, assign unique IP addresses and hostnames for each computer in the cluster. Suppose there are totally *HW* nodes included in the cluster, we named our nodes here from *ciarlab00* to *ciarlabHW*, using the *ciarlab00* as our administrative node. Our cluster is private, so theoretically we could assign any valid IP addresses to our nodes as long as each had a unique value. We used IP addresses 192.168.100.200 for the

host node and added 1 for each work node (192.168.100.201, etc). Also, the hostname for each node is *ciarlabXX.cluster.net*, where '*XX*' is from 00 to *HW* and the DNS name is *cluster.net* with net mask: 255.255.255.0. These settings can be made from the X-windows: *System Settings* → *Network* → *eth0 (Gigabit Ethernet adapter)* → *DNS.*

Then, modify the file */etc/hosts* on each node to include all the nodes in the cluster as follows:

```
192.168.100.200     ciarlab00.cluster.net     ciarlab00
192.168.100.201     ciarlab01.cluster.net     ciarlab01
192.168.100.202     ciarlab02.cluster.net     ciarlab02
        .                    .                    .
        .                    .                    .
        .                    .                    .
192.168.100.2HW     ciarlab40.cluster.net     ciarlabHW
127.0.0.1           localhost.localdomain     localhost
```

Since the computers in our Lab have two Ethernet adapters, Gigabit (eth0) and 100Mbps (eth1), in order to minimize the startup time of the Linux operating system, one can optionally turn off the 100Mbps Ethernet adapter (eth1) by editing the following file: */etc/sysconfig/networking/devices/ifcfg-eth1,* modifying the line ON BOOT = yes to ON BOOT = no.

(2) Network Services

In order to have remote login functions, certain network services should be enabled. At */etc/xinetd.d* directory, modified *rsh*, *rlogin*, *rexec*, *telnet*, *nfs*, *rsync*, *ntpd* ⇒ disabled = yes changed to disabled = no. Then type the command: *xinetd –restart* or reboot the computer in order to activate those functions. Also, turn off

kudzu function (automatically detect new hardware) in the master node only. All these tasks can also be performed under the X-Windows graphic user's interface, chose *System Settings* → *Server Settings* → *Services*, and then select the appropriate items.

(3) Remote Shell Login (RSH)

Create *.rhosts* file in the user directory (*/home/ciarlab*) and *hosts.equiv* file in the */etc* directory. Then, use the command *chmod 644 .rhosts* to change the ownership of the *.rhosts* file. The two files are distributed to all nodes in the cluster. The files are shown as follows:

```
ciarlab00.cluster.net     ciarlab
ciarlab01.cluster.net     ciarlab
ciarlab02.cluster.net     ciarlab
          .                  .
          .                  .
          .                  .
ciarlabHW.cluster.net     ciarlab
```

(4) NFS Security Settings

Disable the firewall under X-Windows, *System Settings* → *Network*. Also, create files *hosts.allow* and *hosts.deny* in the */etc* directory for all nodes.

In file *hosts.allow*, write in:

```
portmap:    192.168.100.200, 192.168.100.201, # , 192.168.100.2HW
lockd:      192.168.100.200, 192.168.100.201, # , 192.168.100.2HW
mountd:     192.168.100.200, 192.168.100.201, # , 192.168.100.2HW
rquotad:    192.168.100.200, 192.168.100.201, # , 192.168.100.2HW
statd:      192.168.100.200, 192.168.100.201, # , 192.168.100.2HW
```

In file *hosts.deny*, write in:

```
portmap:    ALL
lockd:      ALL
mountd:     ALL
rquotad:    ALL
statd:      ALL
```

(5) Clock Synchronization (NTP Setup)

The Network Time Protocol (NTP) provides the clock synchronization. The following shows the reader how to setup our own NTP servers for other clients which are behind the firewalls. Open file */etc/ntp.conf* on each node and edit it (suggestion: backup the original one and then create a new one):

For master node:

```
server          127.127.1.0         #local clock
fudge           127.127.1.0         stratum 10
driftfile/etc/ntp/drift
restrict        default ignore
restrict        127.0.0.1
restrict        192.168.100.201     nomodify
restrict        192.168.100.202     nomodify
...
restrict        192.168.100.2HW     nomodify
authenticate    no
```

For slave nodes:

```
server          192.168.100.200
restrict        default ignore
restrict        127.0.0.1
restrict        192.168.100.200     nomodify
driftfile/etc/ntp/drift
authenticate    no
```

Then, create a file */etc/rc2.d/S168ntp* for all nodes, add the line */usr/sbin/ntpd –c /etc/ntp.conf* to the newly created file. Change the ownership by typing: *chmod 777 S168ntp*. After rebooting, one can check the status at: */usr/sbin/ntpq –p*.

(6) Secure Shell Login (SSH)

Ssh is a secure clone of *rsh* with RSA encryption based authentication. The basis of using *ssh* without typing your password is public key based authentication. A pair of public/private keys is needed to be generated for this.

Firstly, generate user's public/private keys in the */home/ciarlab* directory by typing the command: *ssh-keygen –t rsa*. This will generate user's *id_rsa* and *id_rsa.pub* in the *.ssh* directory in the */home/ciarlab/* directory.

Secondly, copy the *id_rsa.pub* to the *.ssh* directory of the remote hosts that one wants to logon to as authorized_keys2.

Thirdly, change the ownership (*chmod*) of the *ciarlab* and *.ssh* to 755. Then copy the *.ssh* to all nodes: *scp –r .ssh ciarlabXX:/home/ciarlab* (where XX is from 01 to 36).

5.3.3 Software Environment and Tools

Parallel computation on a Beowulf is accomplished by dividing a

computation into parts and making use of multiple processes, each executing on a separate processor, to carry out these parts. Sometimes an ordinary program can be used by all the process, thus no communication occurs among the separate tasks. However, when a parallel computer system is needed to attack a large problem with a more complex structure, such communication is necessary. One of the most straightforward approaches to communication is to have the processes coordinate their activities by sending and receiving messages, much like a group of people cooperate to perform a complex task. This approach to achieve parallelism is called Message Passing [Sterling, 2002].

In computing world, message is defined as a group of information organized in specified format that can be understood by both dispatchers and receivers. Messages provide the data exchange and cooperation among different process. Generally, a message passing tool should be employed to carry out the communication in parallel programming. Basic message communication tools such as sockets and more sophisticated environments, such as Parallel Virtual Machine (PVM) and Message Passing Interface (MPI) are discussed here.

The socket interface is a widely available message passing programming tool which defines a set of data structures and C functions for programmers to establish the full-duplex connection among different computers through TCP protocol. Synchronous and asynchronous parallel programs can be developed with the socket application programmer's interface (API), with the added benefits of common availability, high standardization, and complete control over the

communication primitives. But programming with sockets is error-prone and requires understanding low level characteristics of the network. Also, it does not include any process management, fault tolerance, task migration, security options, or other features usually requested by modern parallel applications [Comer and Stevens, 1993].

PVM is the first widely used software system for message passing developed by Oak Ridge National Laboratory (ORNL), University of Tennessee, Emory University etc. [Sunderam, 1990]. It permits the utilization of a heterogeneous network as a single general and flexible concurrent computational resource. The advantages of PVM are its wide acceptability and its heterogeneous computing facilities, including fault-tolerance issues and interoperability. However, its flexibility is at the cost of capability [Buyya, Vol. 2, 1999]. PVM has recently begun to be unsupported and no further releases come out. Users are shifting from PVM to more efficient paradigms, such as MPI.

MPI is a message passing library specification established by MPI Forum, a group of parallel computer vendors, computer scientists, and users who came together to cooperatively work out a community standard. [http://www.mpi-forum.org/index.htm]. The first version was released in 1994 [William, Vol. 1, 1998], the second version was released in 1997 as the extension of the first one [William, Vol. 2, 1998]. Similar to PVM, MPI is a library of message passing routines but more complete. Users can call corresponding MPI functions to accomplish the message passing efficiently with programs coding in

C/C++ or FORTRAN languages. The MPI functions support process-to-process communication, group communication, setting up and managing communication groups, and interacting with the environment. The communication speed is much fast with the implementation of MPI. As a result, the MPI specification is accepted by more and more computer vendors, such as DEC, HITACH, HP, IBM, and SUN, etc. MPI has now become a defacto standard and several implementations exist. The most common used are MPICH [MPICH home page, http://www-unix.mcs.anl.gov/mpi/mpich/] and LAM/MPI [LAM/MPI parallel computing home page, http://www.lam-mpi.org/]. In this research, LAM/MPI 7.0.2 in C language is employed and installed in each node of the Beowulf cluster.

5.4 MPI Programming

5.4.1 MPI Commands

MPI is a set of API functions enabling programmers to write high-performance parallel programs that pass messages between serial processes to make up an overall parallel job. MPI library provides users full-fledged function definitions, so it is fare convenient for MPI programming. A typical MPI parallel program in C language is shown in Fig. 5.6 as below.

Header file *mpi.h* should be included into the program like ordinary C libraries before call functions in MPI library. All MPI library functions are labeled with MPI_ at the beginning of function name.

An MPI parallel program is composed of multiple processes that communicate with one another. All processes that on task are ranked from 0 and they group into a community which is named as *MPI_COMM_WORLD*.

Function *MPI_Init()* starts up the MPI, and it must be called before any other MPI function.

Function *MPI_Finalize()* shuts down the MPI, all activities related to MPI will be finalized, for example, release the memory allocated by MPI. No MPI function may be called after this function is called.

Function *MPI_Comm_rank()* returns the rank of the calling process and stores it in variable *rank*; function *MPI_Comm_size()* returns the total number of processes and store it in variable *size*.

Condition-branch statement *if…else…* is the main part to realize parallel computation. It assigns different processes to execute different tasks. Tasks that doesn’t been assigned to a specified process will be executed by all processes.

```
        ⋮
# include "mpi.h"
        ⋮

main(int argc, char* argv[])
{
    int rank, size;
        ⋮

    MPI_Init(&argc, &argv);     // No MPI function should be called before here.
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
        ⋮

    if(rank==0)
    {
        ⋮
     }
     else if(rank==1)
    {
        ⋮
     }
     else
    {
        ⋮
     }
     MPI_Finalize();     // No MPI functions should be called after here.

     return 0;
}
```

Fig. 5.6 A typical MPI program

There are different ways for processes to communicate through different MPI functions. Most commonly used is point-to-point sending and receiving (*MPI_Send(), MPI_Recv()*) or point-to-all broadcasting (*MPI_Bcast()*). They are interpreted below:

int MPI_Send(void message, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)*

It sends *count* copies of data stored in memory block *message* to the process with

rank *dest*.

int MPI_Recv(void message, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status* status)*

It receives *count* copies of data sent by process *source* and store it into memory block *message*.

int MPI_Bcast(void message, int count, MPI_Datatype datatype, int root, MPI_Comm comm)*

It sends *count* copies of data stored in memory block *message* in the process with rank *root* to every process (including *root*) in *comm*.

All of the above functions return an integer value which is used as an echo code. If an error is detected in data transfer, the program will make response accordingly. In default situation, the program will exit with an error code.

Parameter *tag* is used to pair up the sending and receiving activity. An *MPI_Recv()* can only receive the data from an *MPI_Send()* with the same tag.

Parameter *datatype* describe the data type of the message defined by MPI. These data types may be basic types corresponding to C language, or special data type of MPI, such as MPI_BYTE and MPI_PACKED, even data types constructed by programmers. Table 5.1 listed the data types corresponding to basic C data types.

Table 5.1 MPI data types corresponding to fundamental C data types

MPI DATA TYPE	C DATA TYPE
MPI_CHAR	signed char
MPI_SHORT	signed short int
MPI_INT	signed int
MPI_LONG	signed long int
MPI_UNSIGNED_CHAR	unsigned char
MPI_UNSIGEND_SHORT	unsigned short int
MPI_UNSIGEND	unsigned int
MPI_UNSIGNED_LONG	unsigned long int
MPI_FLOAT	float
MPI_DOUBLE	double
MPI_LONG_DOUBLE	long double

Besides these basic functions introduced above, other MPI functions used for timing, counting, and cooperating are listed below:

double MPI_Wtime(void)

It returns a double precision number representing the number of seconds that have elapsed since some time in the past.

int MPI_Reduce(void operand, void* result, int count, MPI_Datatype datatype, MPI_Op operator, int root, MPI_Comm, comm)*

It combines the contents of each process's operand using the operation *operator*. Store the result on process with rank *root* only.

int MPI_Waitall(int array_size, MPI_Request requests[], MPI_Status statuses[])

It waits for all the operations associated to elements of the *array* requests to complete.

int MPI_Barrier(MPI_Comm comm)

It blocks the calling process until all processes in *comm* have entered the function (for synchronization).

Details of the usage of these functions and other advanced MPI functions please refer to [Pacheco, 1997].

5.4.2 How to Run MPI Program

In order to run MPI program, every computational nodes need to install the LAM/MPI software. LAM/MPI is a high-performance, freely available, open source implementation of the MPI standard. LAM/MPI is not only a library that implements the mandated MPI API, but also the LAM run-time environment: a user-level, daemon-based run-time environment that provides many of the services required by MPI programs. Details on the configuration and usage of the LAM/MPI can be found in its installation guide and getting started manual respectively.

(1) Compiling MPI Programs

LAM/MPI provides wrapper compilers to add the correct compiler/linker flags and then invoke the underlying compiler to actually perform the compilation/link. The wrapper compiler is named *mpicc* for C programs. The usage of *mpicc* is:

mpicc mpi_program1.c mpi_program2.c –o my_mpi_program

where *mpi_program1.c* and *mpi_program2.c* is the programs to be compiled/linked, *my_mpi_program* is the output executable file which is ready to run in the LAM/MPI run-time environment.

(2) Booting LAM/MPI

Before any MPI programs can be executed, the LAM/MPI run-time environment must be launched. This is typically called booting LAM/MPI.A successfully boot process creates an instance of the LAM/MPI run-time environment commonly referred to as the LAM/MPI universe. When booting LAM/MPI, a boot schema file listing the hosts on which to launch the LAM/MPI run-time environment is needed. This file is typically referred to as a *hostfile*. For example, if the contents of *hostfile* is:

```
ciarlab00       cpu=2
ciarlab01       cpu=2
    ⋮
ciarlab20       cpu=2
```

It tells the LAM/MPI that totally 21 nodes are specified and 2 MPI processes can

be launched on each node. It is important to note that the number of CPUs specified in here has no correlation to the physical number of CPUs in the machine. The *lamboot* command is used to launch the LAM run-time environment. For example:

lamboot –v hostfile

Note that it is not necessary to have LAM/MPI booted to compile MPI programs.

(3) Running MPI Programs

Before running MPI programs, make sure every slave nodes in the cluster has a copy of the executable (in this case, *my_mpi_program*) in the same directory as in the master node. A batch file which includes a serial of remote copy commands (*scp*) is usually employed to take this mission.

Now the parallel program can be run with the command of *mpirun*. Mainly there are three modes to run a program as listed below:

mpirun C my_mpi_program

mpirun N my_mpi_program

mpirun –np X my_mpi_program

where item *C* means to create a process on each CPU; item *N* means to create a process on each node in the LAM/MPI universe; and item *–np X* means to create *X* copies of processes of *my_mpi_program*, LAM/MPI will schedule how many copies of *my_mpi_program* will be run in a round-robin fashion on each node by how many CPUs were listed in the boot schema file or *hosftile*.

(4) Shutdown LAM/MPI

When complete the parallel computation, use command *lamhalt* to shutdown the LAM/MPI run-time environment.

5.5 Parallel Topologies

The most well-know topologies for parallel evolutionary algorithms are Global Parallel topology, Decomposition topology, and Hybrid topology.

5.5.1 Global (Master-Slave) Parallel Topology

The Global Parallel topology which is also called Master-Slave topology is shown in Fig. 5.7. A master processor is always employed to coordinate all other processors (slaves) equally. This method maintains a single population and most of the steps in this loop (evaluation, selection of the individuals) can be executed in parallel. The procedure for Master-Slave DE algorithm can be described as follows.

1. Master processor initializes the whole population of individuals and then divides it evenly among the other slaves.
2. After receiving the sub-populations, the slave processors execute fitness evaluation independently. Then each slave processor sends its result back to the master for the reproduction or reinitialization.
3. The reproduced offspring population is then divided evenly and sent to

slave processors again for fitness evaluation. After fitness evaluation, the offspring individuals compete with their corresponding father, the survivals are sent back to the master for the next reproduction and selection for the best fitness individual.

4. The stopping rule is checked. If it is not satisfied, return to Step 3.

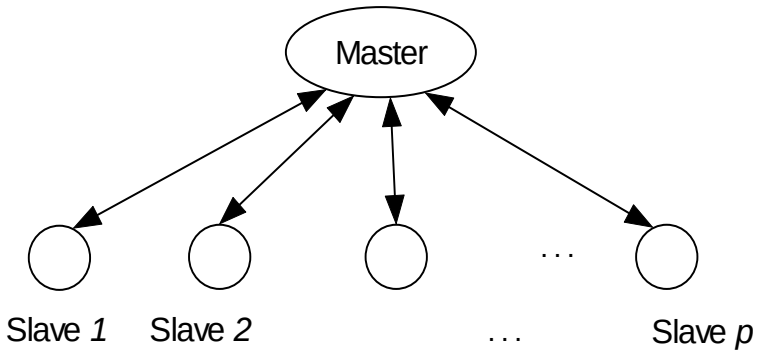

Fig. 5.7 Global (Master-Slave) topology for parallel DE

The optimization ability of this structure has no difference to that of the sequential DE while the computational speed can be improved greatly.

5.5.2 Decomposition Topology

Instead of performing the evolution on a single population, in decomposition topology, the population are partitioned (usually equally) into several sub-populations to evolve. The most popular decomposition methods are Coarse-Grain topology (Fig. 5.8) and Fine-Grain topology (Fig. 5.9).

(1) Coarse Grain Topology

The Coarse-Grain topology is also called Island topology. Each sub-population is assigned to a different processor (island). Each processor runs a sequential DE independently and in parallel on its own sub-population. Sparse exchanges of individuals among different sub-populations will be performed periodically; this is called migration. Individuals may migrate randomly from one sub-population to another or only to geographically nearby sub-populations. A migration policy controls the kind of island being used. Additional controlling parameters have to be decided when migration occurs and how migrants are selected/incorporated from/to the source/target islands are needed, including which other processors to exchange individuals with, how often processor exchange individuals (migration interval), the number of individuals that processors exchange with each other (migration size).

(2) Fine Grain Topology

Relatively larger amount of sub-populations exits in a Fine-Grain topology (Cellular topology or 2D-mesh topology) which also means the small amount or even one individual in each sub-population. Individuals in the sub-population are allowed to mate only with a neighborhood. The shape of a neighborhood may be a cross, square, line etc. Other critical parameters are the radius of the neighborhood to the size of the underlying grid (neighborhood size) and the individual replacement scheme.

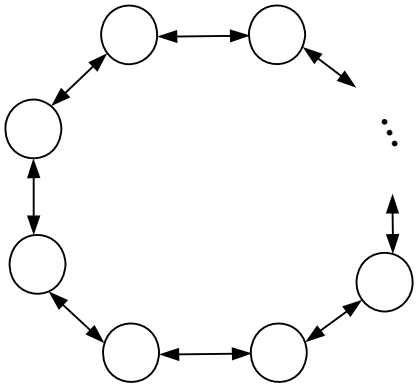

Fig. 5.8 Coarse-Grain parallel topology

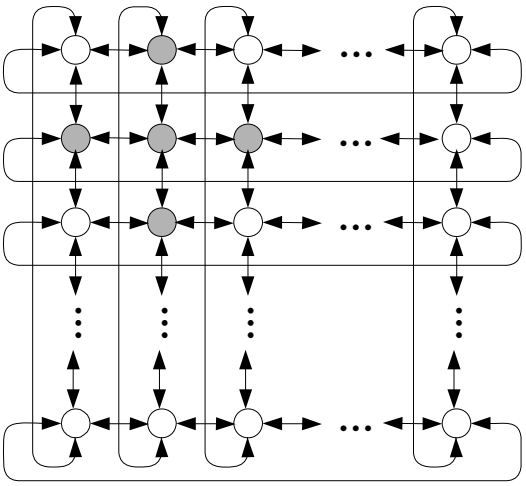

Fig. 5.9 Fine-Grain parallel topology

Actually a dual-direction Island topology is a special case of 2D-mesh topology which only has a single row of processors connected together. Whatever decomposition strategy chosen, additional parameters that control the optimization behavior of the algorithm have to be decided. These algorithm-specified decisions may improve the optimization performance but also complicate the situation. General procedure for a decomposition parallel DE is

summarized as follows.

1. Host processor initialized the whole population and divides it evenly among *p* processors.
2. Each processor executes a sequential DE after receiving the sub-population from the host.
3. After some generation, each processor sends part of its individuals to other processors and replaces part of its own individuals after receiving the individuals from others processors.
4. Check the stopping rule, if it is not satisfied, return to Step 2. Otherwise, each processor sends back its result to host for sorting out the best individual among the sub-populations.

5.5.3 Hybrid Topology

Combination of above topologies results in hybrid topology. These hybrids may use a Master-Slave parallelization on each island of a Coarse-Grain (Fig. 5.10), or have a Coarse-Grain at upper level and Fine-Grain at the lower (Fig. 5.11), or two-level Coarse-Grained parallelization (Fig. 5.12), and etc. Some of these hybrids may keep a same complexity as one of their components or add a new degree of complexity.

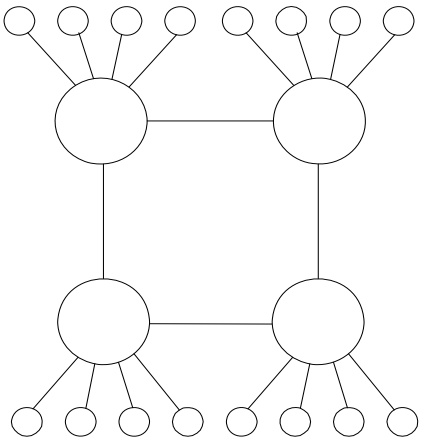

Fig. 5.10 Hybrid Coarse-Grain and Master-Slave Parallel topology

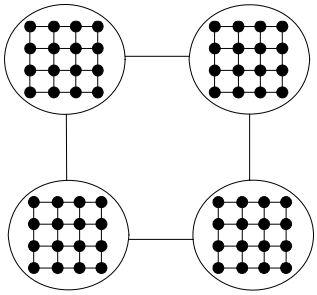

Fig. 5.11 Hybrid Coarse and Fine-Grain topology

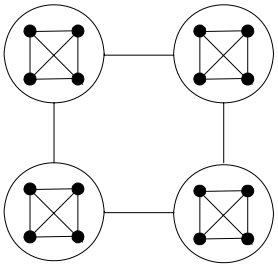

Fig. 5.12 Two Level Coarse-Grain topology

5.6 Summary

As a stochastic searching method based on population, DE is intrinsically

suitable for parallel computation. This chapter introduced modern parallel computation technology to speed up the searching process of DE. The construction of Beowulf PC-cluster, coding, compiling, and running of parallel program are described in detail. Different topologies of parallel DE are analyzed and implemented to study the performance. With the help of parallel computation, it is possible for DE to realize optimization with moderate population size to obtain better solutions.

Chapter 6 Validation of the Parallel DE Based TSCOPF

6.1 Introduction

Chapter 5 constructed the Beowulf pc-cluster parallel computation platform. The speeding up effect is now validated in this chapter on standard test systems. There are totally 31 computers in our cluster including one host node with a monitor and 30 computers act as work nodes without monitors. They are connected via two 24-port Gigabit Ethernet switches. The host node is configured with a single Intel Pentium IV 3GHz CPU; each work node is configured with dual Intel Xeon 2.66GHz CPUs. So there are maximum 61 processors on call.

6.2 Topologies Studies

The basic OPF problem on IEEE 118-bus systems with quadratic generation cost curves for minimizing the total system active power generation cost is employed here to investigate the optimization behavior of different parallel topologies described in last chapter. All system data are the same as that in Chapter

2. For all topologies, the population size was set at 600 for even distribution of sub-populations; the total number of generations was 1000 and run for 20 trials.

For Coarse-Grain topology, the migration strategy is dual-ring as illustrated in Fig. 5.8. The sub-population size is of 10 individuals. The best 20% of the sub-population will migrate to the adjacent 2 processors and replace the worst with individuals received from other sub-population each generation.

For Fine-Grain topology, the 2D-mesh is constructed as 10 rows and 6 columns. Each sub-population will exchange individuals with their 4 neighbors that next to it from up, down, left, and right as the dark circles shown in Fig. 5.9. The best 20% individuals of each elementary sub-population are migrated within the neighborhood and the worst one is replaced by new coming individual.

For Hybrid topology, a combination of the Master-Slave and Coarse-Grain parallel topologies has been implemented as shown in Fig. 5.10. At the upper-level, the hybrid topology is ring-liked which evolves several populations independently. At the lower-level, the Master-Slave topology is adopted in each of the populations (processors). Migration of the best individual occurs between adjacent processors as in the Coarse-Grain topology. The migration size is also set at 20% of the sub-population. 4 processors are appointed to upper-level, and each island has 15 processors to construct the lower-level Master-Slave topology. To improve the availability, unlike global Master-Slave implementation, the master processor of the lower-level in the Hybrid topology has to take the evolution task of one sub-population besides the reproduction work, and the sub-population can be

distributed evenly among the lower-level processors.

For comparison, the sequential simulation results with same population size are listed with the parallel ones in Table 6.1 together. Convergence characteristics of different topologies are shown in Fig. 6.1. The average, minimum and maximum costs of solutions are found to be very similar for sequential and Master-Slave DE implementations. However, optimization performances of other three topologies are much worse than the Master-Slave one. The reason lies in that DE is a kind of evolutionary algorithm without actual mutation operation in its evolution, the reproduction of new individual relies on the difference of other individuals in the population. Therefore, it needs relatively large population to prevent premature. The Master-Slave implementation reproduces its offspring based on the whole population; however, the Coarse-Grain, Fine-Grain, and Hybrid parallel DE evolve independently on the sub-populations whose sizes are much smaller than the whole population, therefore, overall fitness of these implementations is decreased by the small sub-population which deteriorates the diversity of the optimal solutions. This conclusion is also told by the relatively better performance of Hybrid implementation which has larger sub-population size than the Coarse-Grain and Fine-Grain implementations.

Table 6.1 Simulation results of parallel OPF on different topologies for IEEE 118-bus test system

Topology	Minimum	Maximum	Average	STD %

	Cost ($/h)	Cost ($/h)	Cost ($/h)	
Sequential	130019.73	130905.36	130304.51	0.19%
Master-Slave	130025.80	130898.02	130299.05	0.18%
Coarse-Grain	136258.42	146688.36	141268.01	2.03%
Fine-Grain	136026.41	145972.06	140687.62	1.60%
Hybrid	131063.70	134422.92	132703.50	0.67%

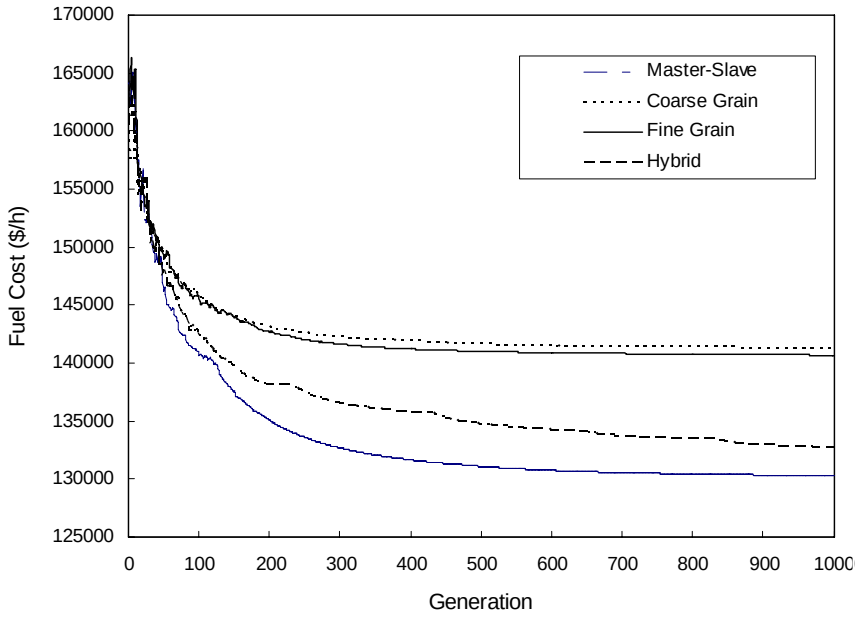

Fig. 6.1 Average convergence curve for IEEE 118-bus system

From the above, Master-Slave is the best topology for parallel DE implementation theoretically and numerically. It speeds up the computation without sacrifice the optimization performance. Especially, the implementation of Master-Slave topology is the most convenient one. Therefore, Master-Slave is chosen implementation in the following studies in this thesis.

6.3 Computation Time Analysis

According to the Amdahl's law [Amdahl, 1967], the ideal speedup ratio with constant problem size is equal to the number of processors used in the parallel computation. The experimental speedup ratio ψ is computed as the ratio of the sequential execution time to the parallel execution time. Assuming that a program is perfectly parallelized and taking the communication time into account, the speedup is given by,

$$\psi = \frac{T_{comp}^{s}}{T_{comp}^{p}} = \frac{T_{comp}^{s}}{(T_{comp}^{s} / p) + T_{comm}} \leq \frac{T_{comp}^{s}}{(T_{comp}^{s} / p_{\max}) + T_{comm}} = \psi_{\max} \qquad (6.1)$$

where p is the number of processors, T_{comm} is the total communication time that used to transfer information among different processors, T_{comp}^{s} is the serial execution time, and T_{comp}^{p} is the parallel execution time. Obviously, the experimental speedup is less than the ideal speedup due to the inclusion of communication time. When $p = p_{\max}$, the speedup upper bound $\psi_{\max}$ is achieved.

In order to investigate the practical computational time speedup performance of the proposed parallel methods, simulations with different number of slave processes engaged in the computation are re-performed on two cases:

A. Basic OPF problem on the IEEE 118-bus system

B. TSCOPF problem on the New England 39-bus system

Since parallelization may significantly improve the speed of DE for solving TSCOPF problems; it is possible to realize the OPF problem on the PC-clusters

with moderate population size. For the IEEE 118-bus system, the simulation conditions are the same as in last section. For the New England 39-bus system, the system parameters are the same as the multi-constrained discrete TSCOPF case studied in chapter 4. The population size of case B in this testing is enlarged to 120 (4 times of the control variable size) and the maximum generation number is extended to 200. All individuals in the population will take participant into the transient stability assessment so as to obtain better solutions. The average computational time (ACT) and speedup ratios corresponding to different slave numbers among the total 20 trials for case A and B are listed in Table 6.2 and 6.3, respectively. Where slave number = 1 represents the sequential computation case. The speedup ratio curves of both cases are plotted in Fig. 6.2.

Table 6.2 Computational time for parallel DE with different number of slave processes for IEEE 118-bus system (population size = 600)

Slave Number	1	2	4	6	8	10
ACT (s)	1820.52	1089.48	580.34	430.79	328.44	260.48
Speedup Ratio	1.000	1.671	3.137	4.226	5.543	6.989
Slave Number	12	15	20	30	40	60
ACT (s)	223.07	176.17	141.61	129.25	124.11	120.16
Speedup Ratio	8.161	10.334	12.856	14.085	14.668	15.151

Table 6.3 Computational time for parallel DE with different number of slave processes for New England 39-bus system (population size = 120)

Slave Number	1	2	4	8	10	12

ACT (s)	575.5	304.59	174.32	118.43	90.65	74.42
Speedup Ratio	1.000	1.823	3.185	4.688	6.125	7.460
Slave Number	15	20	24	30	40	60
ACT (s)	66.2	52.75	39.60	28.44	21.60	15.03
Speedup Ratio	8.387	10.526	14.020	19.522	25.704	36.939

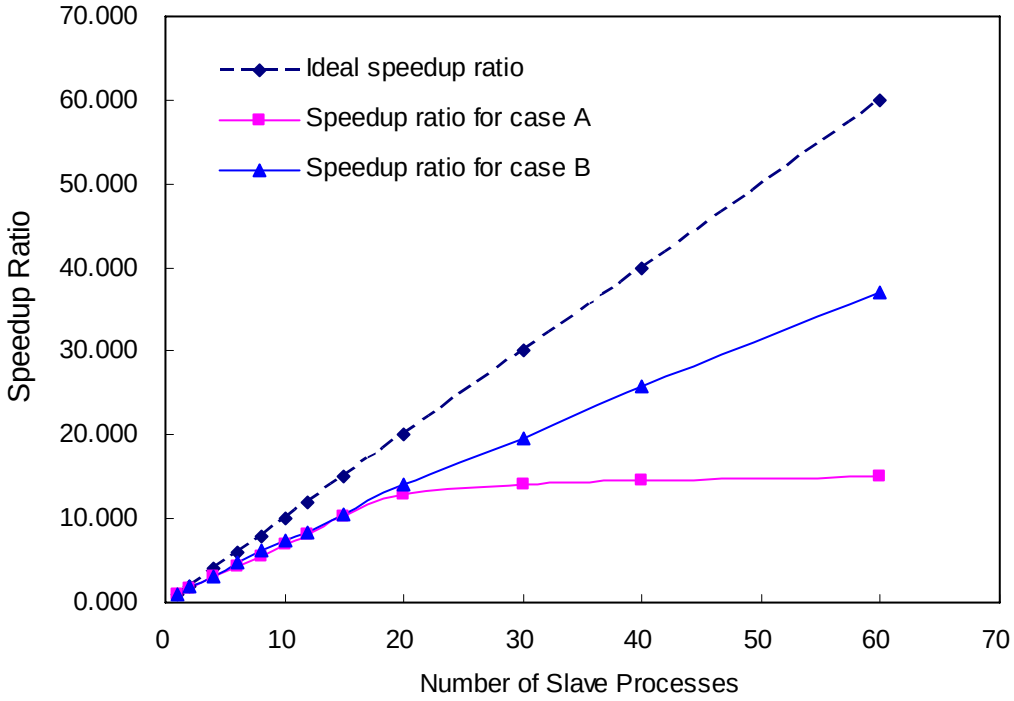

Fig. 6.2 Speedup ratios correponding to different slave processe numbers

ACTs listed in Table 6.2 and 6.3 show that parallelization does significantly improve the speed of DE for solving both basic OPF and transient stability constrained OPF problems. The performance of the parallel implementation can be reflected by the speedup ratio. As expected, the more is the number of processors, the higher is the speedup ratio, until $\psi_{\max}$ is reached when $p = p_{\max}$ (60 processors).

However, it is also found that although the computational time keeps decreasing with the increase of the number of slave processes, the speedup ratios saturate gradually (comparing to the ideal speedup ratio line in Fig. 6.2), which means that the efficiency of the PC-cluster drops with the increase of the slave processors. Besides, speedup ratio of case A saturates much more quickly than that of case B. The reason for this phenomenon exits in the sequential nature of data communications between master and slaves processors (via the only one data channel). For Master-Slave topology, when problem and its population size are given, the communication time T_{comm} keeps almost the same despite the different slave process numbers. When p is small, the communication time T_{comm} is negligible to the total execution time; whereas, with the number of slave processes becoming large, the calculating time T_{comp}^{s} / p decreases rapidly. When T_{comp}^{s} / p is small enough and comparable with the communication time T_{comm}, the speedup ratio starts to be saturated. The computational time for one load flow calculation is much shorter than transient stability assessment for an individual, therefore, the speedup ratio of basic OPF saturates more early than that of TSCOPF. The speedup ratio saturation is an important item considering both the efficiency and the economy in parallel computation. In this research, 20 slave processes is a good balance for basic OPF problem; while, all available processes should be employed when solving the TSCOPF problem.

6.4 Multi-contingency TSCOPF

Power system operating conditions are classified into five states: Normal, Alert, Emergency, In Extremis, and Restorative states [Kundur, 1994]. The main goal of system operators is to operate and maintain power systems in normal secure state with time-varied operating conditions. Practically, the supervisory control and data acquisition (SCADA) system in collects the power system data at a specified scan rate frequency. Energy management system (EMS) estimates the system status based on these data and make corresponding preventive or operational decisions. Preventive control is carried out in the normal or alert state of the operation, which is before the occurrence of contingencies. The number of possible contingencies could be huge in a real power system. Only the most dangerous contingencies are considered for dispatch. Therefore, a fast on-line contingency assessment tool is usually embedded in the EMS to rank and filter out those credible ones time window by time window. Preventive strategies based on TSCOPF are then expected to re-dispatch the system and maintain the system stable in most credible contingencies. Thus TSCOPF with respect to single contingency might deteriorate the security level in other contingencies. Multi-contingency cases should be considered for the improvement of the overall security level. Consider multi-contingency constraints in TSCOPF may be a rather rough task for traditional sequential computation because of the huge CPU intensity.

In this study, the definition of "multi-" or "single-" contingency in preventive

control is according to [Yuan, Kubkawa, and Sasaki, 2003]. "Single contingency" is defined as one fault (with or without enclosure) or two faults (simultaneous or cascading), etc. Multi-contingency, such as contingency (A+B), is defined as either contingency A or contingency B will occur at the same operating point. In other word, for preventive control based on transient stability constrained OPF, the system remains stable no matter which contingency A or B occurs.

In the following study, a three-phase fault is applied at the sending end of all lines of the New England 39-bus system and is then cleared with the faulty line tripped simultaneous at the ends of the line. Three contingencies are screened out and listed as below:

1. Contingency A: A three-phase-to-ground fault at bus 21 and cleared by tripping line 21-22 at 160 ms, which is greater than the initial CCT 144 ms
2. Contingency B: A three-phase-to-ground fault at bus 17 and cleared by tripping line 17-18 at 200 ms, which is greater than the initial CCT 167 ms
3. Contingency C: A three-phase-to-ground fault at bus 4 and cleared by tripping line 4-5 at 250 ms, which is greater than the initial CCT 222 ms

Simulations are conducted with 60 slave processes and all system data and parameters are the same as last section. Contingency combinations (A), (A+B), and (A+B+C) are applied to the TSCOPF problem and simulation results are summarized in Table 6.4. The convergence curves of each situation are plotted in Fig. 6.3.

All cases converge well as shown in Fig. 6.3. As has expected, the system has

to sacrifice more economy when consider more security. The average system cost is 61428.65 $/h when only contingency A is considered and increases to 61515.60 $/h when all three contingencies included. Especially, two infeasible solutions occurred among the 20 trials for case (A+B+C) which shows the solution is very close the economy and security boundary.

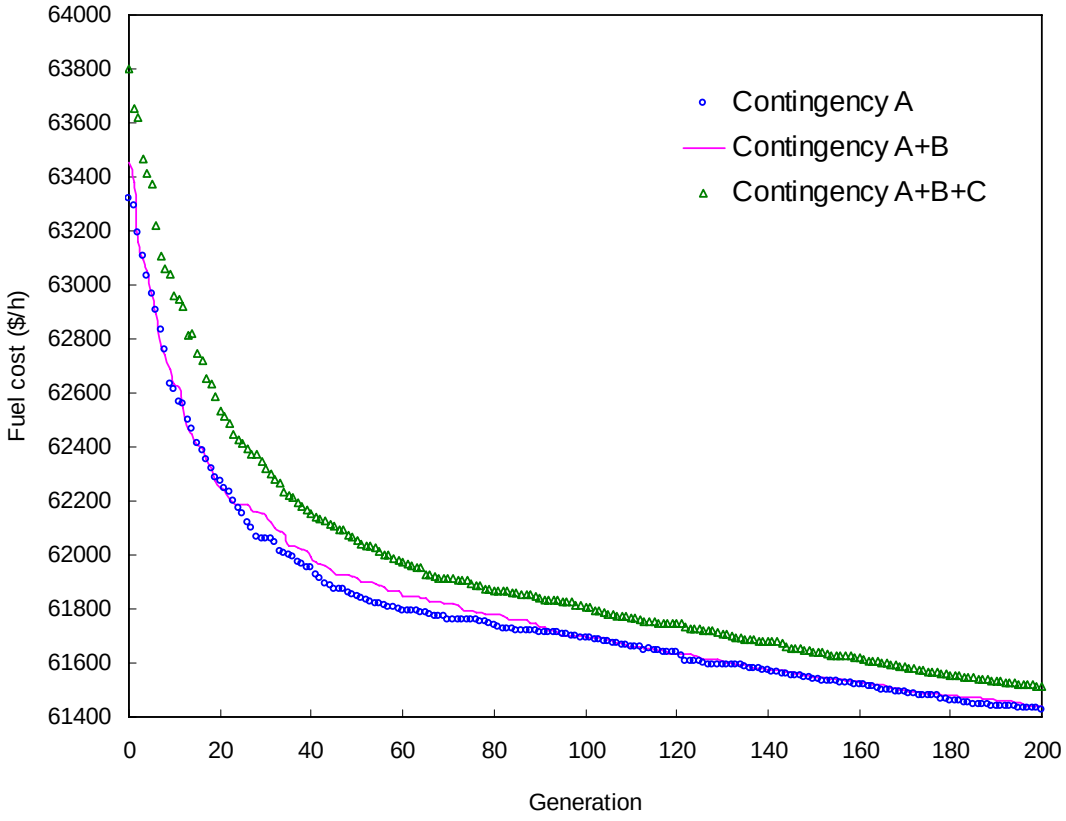

Fig. 6.3 Average converge curves for different contingency combinations

Table 6.4 Simulation results for different contingency combinations

Contingency	A	A+B	A+B+C

Minimum Cost ($/h)	61360.66	61349.63	61389.01
Maximum Cost ($/h)	61604.05	61725.02	61808.31
Average Cost ($/h)	61428.65	61438.36	61515.60
Standard Deviation (%)	0.09	0.15	0.18
Feasible (%)	100	100	90
Stable (%)	100	100	100
ACT (s)	15.03	29.95	47.17

Unlike conventional optimization methods whose CPU time increases exponentially with the contingency size (since the number of variables increases with that of contingencies), the average CPU time for different cases bears a linear relationship to the number of contingencies.

6.5 TSCOPF on Dynamic 17-generator System

Above simulations of TSCOPF problems are carried out on a small test system. The dynamic 17-generator system is employed in this section to investigate the ability of the proposed parallel method on large system. This system has 162 bus and 284 branches. System data is available in [Power System Test Case Archieve: http://www.ee.washington.edu/research/pstca/] and all negative generator real power at each bus is treated as positive load. Ten transformers are selected as adjustable with regulation range from 0.90 p.u. to 1.10 p.u.. The regulation interval is set as 0.01 p. u.. The fuel cost parameters and the rating of generators are listed in Table 6.5. The lower and upper limits of all bus

voltage magnitudes are set at 0.94 and 1.06, respectively. There are 43 control variables, including 16 generator active power outputs, 17 generator voltages and 10 discrete transformer tap settings. The population size is set as 180 (i.e. each slave processor needs to handle 3 individuals per generation); and the maximum generation number is 200. A three-phase to ground fault occurs at bus 1 and cleared by tripping line 1-4 at 240 ms, which is greater than the initial CCT 220 ms. The initial operating state has a fuel cost of 29416.48 $/h. All individuals are participated in the transient stability assessment.

Table 6.5 Generator data of the 17-generator, 162-bus system

Bus No.	P_{min}	P_{max}	a	b	c
3	1000	2300	0.00064	0.50	0.0
6	500	1094	0.00098	0.30	0.0
15	1000	1800	0.00076	0.50	0.0
27	1000	1800	0.00076	0.50	0.0
73	200	747	0.00150	0.20	0.0
76	500	1355	0.00088	0.30	0.0
99	0	450	0.00200	0.40	0.0
101	0	382	0.00200	0.40	0.0
108	0	1200	0.00084	0.30	0.0
114	0	431	0.00200	0.40	0.0
118	0	473	0.00200	0.40	0.0
121	200	920	0.00150	0.30	0.0
124	1000	2851	0.00640	0.52	0.0
125	1000	2688	0.00640	0.67	0.0
126	1000	2767	0.00640	0.42	0.0

130	200	755	0.00150	0.30	0.0
131	200	875	0.00150	0.30	0.0

Simulation results are listed in Table 6.6. All violations have been eliminated at the end of the evolution. The average fuel cost among all 20 trials is 26520.39 $/h and the cost reduction is 9.85% when compared with that of the original operating state. The small standard deviation of 0.07% reveals the robustness of DE based TSCOPF method even for large systems. All solutions obtained can maintain the system stable after the contingency. Convergence curve in Fig. 6.4 shows that DE converges well.

Table 6.6 Simulation results of the 17-generator, 162-bus system

Minimum Cost ($/h)	26480.28
Maximum Cost ($/h)	26566.08
Average Cost ($/h)	26520.39
Standard Deviation (%)	0.07
Feasible (%)	100
Stable (%)	100
ACT (s)	598.73

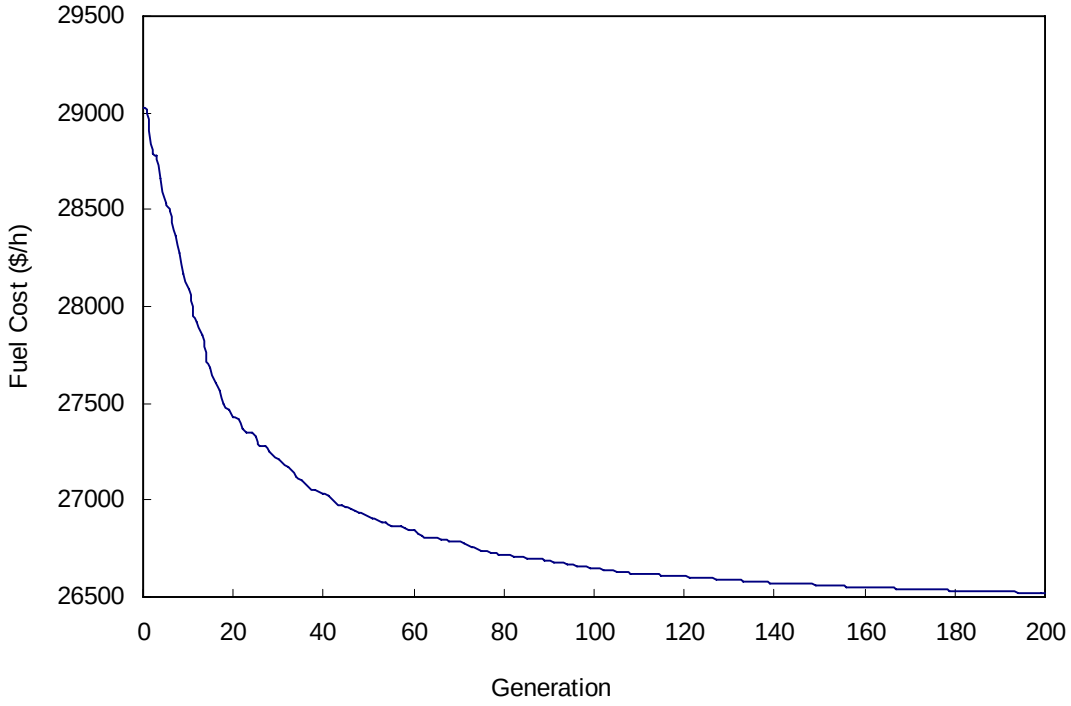

Fig. 6.4 Average converge curves of the dynamic 17-generator system

6.6 Summary

This chapter makes use of the parallel computation platform constructed in chapter 5 to resolve the TSCOPF problem. Master-Slave topology is selected as the most suitable implementation for DE method numerically and theoretically. Simulations results on both basic OPF and TSCOPF problems have proven the strong accelerating ability of parallelization. Speedup ratio studies provide a criterion for slave process size selection and balance of the efficiency and economy. With parallel computation, multi-contingency and large system TSCOPF problem could be resolved in acceptable time. As a result, parallel DE based TSCOPF provides a promising way to realize on-line economic system

operation with dynamic security insurance.

Chapter 7 Conclusion and Future Work

7.1 Conclusion

The electric power industry has undergone fundamental changes over the past two decades. Modern deregulated competitive environments have pushed the system to operate much closer to security boundaries than ever before. Operating the system in a most economical way as well as maintain its stability especially for credible disturbances becomes an imperative problem. However, it is practically an extremely difficult task since the stability constrained OPF is mathematically a non-linear, non-convex, and discontinuous problem with both differential and algebraic constraints. Conventional derivative-based methods always suffer from these non-differentiable features of the TSCOPF problem. This thesis has done some pioneer work in solving the TSCOPF problem using the new-fashioned Differential Evolution algorithm and parallel computation technology. The main work and conclusions can be summarized as follows:

1) Firstly introduces the differential evolution (DE) algorithm to resolve the TSCOPF problem. The mechanism and parameter settings of DE are thoroughly discussed. Performance studies showed that DE is an excellent optimization method. Since DE needs a relatively large population size to avoid premature, an

improved version of DE which re-initializes the population at certain generation interval has also been proposed to address this problem.

2) Transient stability constraint is denoted as an index of individual which has decision in the evolving process of DE. This problem modeling way relaxes the binding of OPF equations and transient stability DAEs. Hybrid method which combines time domain simulation and transient energy function is employed to assess the transient stability of each individual with no limitation in system modeling. Stable individual has more chance to survive in the evolution process in seeking both secure and economy global solution. Since transient stability assessment is the most time-consuming part of the whole method, strategies called "stable-space push" and "fitness sorting" are also developed to reduce the searching space as well as the computation time. With the flexibility and robustness of DE, more practical discontinuous and non-differentiable constraints such as generator prohibited operating zones, generator valve-point effects could have been considered in the TSCOPF problem. Despite the complexity, DE showed its strong optimization ability. Comparison of the simulations results revealed the high superiority of DE to conventional methods.

3) Since DE is intrinsically easy to be parallelized, parallel computation platform is constructed in this thesis to speedup the searching process of DE. The parallel computation is implemented on a Beowulf PC-cluster using Message-Passing Interface (MPI) technology. Different parallel topologies and the speedup ratios are studied in detail. Case studies shows that parallelization does

significantly improve the speed of DE; Moreover, unlike conventional optimization methods whose CPU time increases exponentially with the contingency size, the average CPU time for different cases bears a linear relationship to the number of contingencies. Therefore, it is possible to realize online TSCOPF with moderate scale PC clusters and meet the real-world online application requirement.

7.2 Future Work

The research work presented in this thesis is an attempt to develop new methodologies for maintaining transient stability of the power system as well as its economical operation. With the progress made in this research work, the following issues are expected to be further explored in the future:

1) Transient stability is one of the most important concerns of power system security and it is the only dynamic constraint considered in this research. However, power systems face various disturbances in practice. Other security problem such as voltage stability constraints could also be integrated in.

2) Although parallel computation speeds up the computation dramatically, improvements of DE by methods such as constructing hybrid algorithms or incorporating OPF related knowledge are highly preferred to reduce the required population size to further speedup the computation or enable the use of smaller scale PC-clusters for economy consideration.

Reference

Ahmed A. A. Esmin, Germano lambert-Torres, “A hybrid particle swarm optimization applied to loss power minimization,” IEEE Transactions on Power Systems, Vol. 20, No. 5, pp. 859-866, Oct. 2005

Alba E., Tomassini M., “Parallelism and evolutionary algorithms,” IEEE Transactions on evolutionary computation, vol. 6, no. 5, pp. 443-462, Oct. 2002

Alsac O., & Scott B., “Optimal load flow with steady state security,” IEEE Transaction on Power Apparatus. Syst., PAS-93, pp. 745-751, May-June, 1974

Amdahl Gene, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities", AFIPS Conference Proceedings, (30), pp. 483-485, 1967

Anderson P. M., Fouad A. A., “Power system control and stability”, Iowa State University Press, 1977

Bäck T., Hammel U., and Schwefel H. P., “Evolutionary computation: comments on the history and current state,” IEEE Transactions on Evolutionary Computation,

1997, 1(1): 3-17

Bakirtzis A. G. , Biskas P. N. , etc., “Optimal Power Flow by Enhanced Genetic Algorithm,” IEEE Trans. Power Systems, vol. 17, pp. 229-236, May 2002

Bettiol A.L., Wehenkel L., and Pavella M., “Transient stability-constrained maximum allowable transfer,” IEEE Trans. Power Syst., Vol. 14, No. 2, pp. 654-659, May 1999

Bjelogrlić M. R., Ćalović M. S., Babić B. S., and et al, “Application of Newton's optimal power flow in voltage/reactive power control,” IEEE Transactions on Power Systems, vol. 5, no. 4, pp. 1447-1454, Nov. 1990

Buyya R., “High performance cluster computing: Architectures and Systems, Volume 1”, Upper Saddle River, N.J. : Prentice Hall PTR, c1999

Buyya R., “High performance cluster computing: Programming and Applications, Volume 2”, Upper Saddle River, N.J. : Prentice Hall PTR, c1999

Carpentiers, J.: “Contibution a. l’etude du dispatching economique,” Bull. Soc. Francaise Elect. 1962, 3, pp. 431–447

Chang Y. P. and Wu C. J., “Optimal multiobjective planning of large-scale passive harmonic filters using hybrid differential evolution method considering parameter and loading uncertainty,” IEEE Transactions on Power Delivery, vol. 20, no. 1, pp. 408-416, Jan. 2005

Chen L., Tada Y., Okamoto H., Tanabe R., and Ono A., “Optimal operation solutions of power systems with transient stability constraints,” IEEE Trans. Circuits and System I, vol. 48, pp. 327–339, Mar. 2001

Chiang C. L., “Improved genetic algorithm for power economic dispatch of units with valve-point effects and multiple fuels,” IEEE Transactions on Power Systems, Vol. 20, No. 4, Nov. 2005, pp. 1690-1699

Coelho L. S., and Mariani V. C., “Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect,” IEEE Transactions on Power Systems, Vol. 21, No. 2, May. 2006, pp. 989-995

Comer D. E., and Stevens D. L., “Internet working with TCP/IP (Volume III)”, Englewood, Cliffs, NJ: Prentice-Hall, 1993

Corne D., Dorigo M., and Glover F., “New ideas in optimization”, London:

McGraw-Hill Education, 1999, pp. 79-108

Daniel R. V. and Pavella M., “A comprehensive approach to transient stability control: part I - near otimal preventive control,” IEEE Trans. Power Syst., Vol. 18, No. 4, pp. 1446-1452, Nov. 2003

Dawn Layden, Jeyasurya B., “Integrating security constraints in optimal power flow studies,” IEEE Power Engineering Society General Meeting, Vol. 1, June 2004

Dommel H. W. and Tinney W. F., “Optimal power flow solutions,” IEEE Trans. PAS-87, Vol.87, No.5, pp. 1866-1876, 1968

Dumitrescu D., Lazzerini B., Jain L. C., et al. “Evolutionary Computation”, New York: CRC Press, 2000

Ernst D., Ruiz-Vega D., Pavella M., Hirsch P. M., and Sobajic D., “A unified approach to transient stability contingency filtering, ranking and assessment,” IEEE Trans. Power Syst., 2001, 16, (3), pp. 435-443

Fang D. Z. and David A. K., “A Normalized Energy Function for Fast Transient Stability Assessment,” Electric power System Research, 69 (2004), pp 287-293

Fogel D. B., “Evolutionary computation: toward a new philosophy of machine intelligence”, 2nd edition. New York: IEEE Press, 2000

Fouad A. A., and Vittal V., “Power System Transient Stability Analysis Using the Transient Energy Function Method”, Englewood Cliffs, new Jersey: Prentice Hall, USA, 1992

Gaing Z. L., “Particle swarm optimization to solving the economic dispatch considering the generator constraints,” IEEE Transactions on Power Systems, Vol. 18, No. 3, Aug. 2003, pp. 1187-1195

Gamperle R., Muller S., & Koumoutsakos P., “A parameter study for differential evolution,” Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, WSEAS Press, pp. 293-298, 2002. Available: http://www.icos.ethz.ch/cse/research/publications/proceedings/wseas02.pdf

Gan Deqiang, Thomas Robert J., and Zimmerman Ray D., “Stability-constrained optimal power flow,” IEEE Trans. Power Systems, vol. 15 pp. 535-540, May 2000

Geist A., Beguelin A. and Dongarra J., “PVM: Parallel Virtual Machine”, Cambridge, MA: MIT Press, 1994

Gropp W., Lusk E. and Skjellum A., “Using MPI: Portable Parallel Programming with the Message-Passing Interface”, Cambridge, MA: MIT Press, 1994

He S., Wen J. Y., Prempain E., Wu Q. H., Fitch J., & Mann S., “An improved particle swarm optimization for optimal power flow,” International Conference on Power System Technology - POWERCON, Singapore, Nov., 2004, 1633-1637

Hollman J. A., Marti J. R., “Real time network simulation with PC-cluster,” IEEE Transactions on Power Systems, 2003, 18(2): 563-569

Kundur P., Paserba J., Ajjarapu V., Andersson G., Bose A., Canizares C., Hatziargyriou N., Hill D., Stankovic A., Taylor C., Van Cutsem T., and Vittal V., “Definition and Classification of Power System Stability,” IEEE/CIGRE Joint Task Force on Stability Terms and Definitions Report, IEEE Trans. Power Syst., Vol. 19, No. 3, pp.1387-1401, Aug. 2004

Kundur P., “Power system stability and control”, McGraw-Hill, 1994

LAM/MPI parallel computing home page, http://www.lam-mpi.org/

Lampinen J., “A Bibliography of Differential Evolution Algorithm, Technical

report", Laboratory of Information Processing, Department of Information Technology, Lappeenranta University of Technology, Finland, 2001

Lee K. Y. and Yang F. F., "Optimal reactive power planning using evolutionary algorithms: a comparison study of evolutionary programming, evolutionary strategy, genetic algorithm, and linear programming," IEEE Transactions on Power Systems, vol. 13, no. 1, pp. 101-108, February 1998

Liu D., and Cai Y., "Taguchi Method for Solving the economic dispatch problem with nonsmooth cost function," IEEE Transactions on Power Systems, Vol. 20, No. 4, Nov. 2005, pp. 2006-2014

Maria G. A., Tang C., and Kim J., "Hybrid transient stability analysis," IEEE Trans, Power Syst., 1990, 5, (2), pp. 384-393

Message Passing Interface Forum home page, http:// www.mpi-forum.org/index.htm

Miranda V., Srinivasan D., Proenca L. M., "Evolutionary computation in power systems," International Journal of Electrical Power & Energy Systems, 1998, 20(2): 89-98

Momoh J. A., El-Hawary M. E., and Adapa R., “A review of selected optimal power flow literature to 1993 I: Nonlinear and quadratic programming approaches,” IEEE Transaction on Power Systems, Vol. 14(1), pp. 96-104, 1999

Momoh J. A., El-Hawary M. E., and Adapa R., “A review of selected optimal power flow literature to 1993 II: Newton, linear programming and interior point methods,” IEEE Transaction on Power Systems, Vol. 14(1), pp. 105-111, 1999

MPI: A Message-Passing Interface Standard, 19945, 1995

MPICH home page. http://www-unix.mcs.anl.gov/mpi/mpich/

Nguyen T., Pai M., “Dynamic Security-Constrained Rescheduling of Power Systems Using Trajectory Sensitivities,” IEEE Trans. on Power Systems, Vol.18, No.2, May 2003, pp. 848-854

Novosel D., Begovic M. M., and Madani V., “Shedding light on blackouts,” IEEE Power and Energy Magazine, Vol. 2, No. 1, pp. 32- 43, Jan./Feb. 2004

Ohta Y., Chen L., Hamada H., and Yokoyama R., “Functional transformation cobined with modified SQP for transient stability-constrained OPF,” in Proc. 2005, IEEE Power Engineering Society Transmission and Distribution Conference

Pacheco P. S., “Parallel programming with MPI”, San Francisco: Morgan Kaufmann Publishers, 1997

Pai M. A., “Energy Function Analysis for Power System Stability”, Norwell, MA: Kluwer, 1989

Park J. B., Lee K. S., Shin J. R., and Lee K. Y., “A partical swarm optimization for economic dispatch with nonsmooth cost function,” IEEE Transactions on Power Systems, Vol. 20, No. 1, Feb. 2005, pp. 34-42

Pavella M., “Generalized one-machine equivalents in transient stability studies,” IEEE Power Eng. Rev., 1998, 18, (1), pp. 804-810

Pavella M., Murthy P.G., “Transient stability of power systems: theory and practice”, Chichester, New York, Wiley, 1994

Petcu D. and Zaharie D., “Parallel implementation of multi-population differential evolution, in Concurrent Information Processing and Computing,” D. Grigoras and A. Nicolau (Eds), IOS Press, vol. 195 NATO Science Series: Computer & Systems Series, May 2005, pp. 223-232

Power System Test Case Archieve, [Online]. Available: http://www.ee.washington.edu/research/pstca/

Quintana V. H. and Santos-Nieto M., “Reactive-power dispatch by successive quadratic programming,” IEEE Transactions on Energy Conversion, vol. 4, no. 3, pp. 425-435, Sept. 1989

Quintana V. H., Torres G. L., Palomo J. M., “Interior-point methods and their applications to power systems-a classification of publications and software codes,” IEEE Transactions on Power Systems, 2000, 15(1): 170-176

Ramos J. L. M., Expósito A. G. and Quintana V. H., “Transmission power loss reduction by interior-point methods: implementation issues and practical experience,” IEE Proceedings-Generation, Transmission and Distribution, vol. 152, no. 1, pp. 90-98, Jan. 2005

Rezania E. and Shahidehpour S. M., “Real power loss minimization using interior point method,” International Journal of Electrical Power & Energy Systems, vol. 23, no. 1, pp. 45-56, Jan. 2001

Ristanovic R., “Successive linear programming based OPF solution, Optimal Power Flow: Solution Techniques, Requirements and Challenges,” IEEE Power

Engineering Society, 1996, pp. 1-9

Sauer P. W., Pai M. A., “Power System Dynamics and Stability”, Englewood Cliffs, NJ: Prentice-Hall, 1998

Scala La M., Trovato M., and Antonelli C., “On-line dynamic preventive control: An algorithm for transient security dispatch,” IEEE Trans. Power Syst., vol. 13, pp. 601–610, May 1998

Shahidehpour S. M. and Ramesh V. C., “Nonlinear programming algorithm and decomposition strategies for OPF, Optimal Power Flow: Solution Techniques, Requirements and Challenges,” IEEE Power Engineering Society, 1996, pp. 10-24

Shubhanga K. N., and Kulkarni A. M., “Stability-constrained generation rescheduling using energy margin sensitivities,” IEEE Trans. Power Syst., Vol. 19, No.3, August, 2004

Spears W. M., “Evolutionary algorithms: the role of mutation and recombination”, Berlin: Springer-Verlag Press, 2000, pp. 205-206

Sterling T., “Beowulf Cluster Computing with Linux”, Cambridge, MA, MIT Press, 2002

Storn R., Price K., “Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces”, Technical report, International Computational Science Institute, Berkeley, 1995

Su C. T. and Lee C. S., “Network reconfiguration of distribution systems using improved mixed-integer hybrid differential evolution,” IEEE Transactions on Power Delivery, vol. 18, no. 3, pp. 1022-1027, July 2003

Sunderam V. S., “PVM: A frame for parallel distributed computing”, J. Concur. Practice and Experience, Vol. 2, No. 4, pp 315-339, 1990

U.S.-Canada Power System Outage Task Force, “Final report on the August 14, 2003 blackout in the United States and Canada: causes and recommendations,” http://www.nerc.com/~filez/blackout.html, 2004

Vesterstrom J. and Thomsen R., “A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems,” In Congress on Evolutionary Computation, June 19-23, 2004, Portland, Oregon, USA, vol. 2, pp. 1980-1987

William G., “MPI-the complete reference, vol. 1, the MPI core”, Cambridge,

Massachusetts: MIT Press, 1998, 2nd edition

William G., “MPI-the complete reference, vol. 2, the MPI-2 extensions”, Cambridge, Massachusetts: MIT Press, 1998

Wong K. P., Li A., “Virtual population and acceleration techniques for evolutionary power flow calculation in power systems,” Invited paper in Evolutionary Optimization, R. Sarker, M. Mohammadian and X. Yao (Eds), Boston: Kluwer Academic Publishers, 2002, 329-345

Xue, Y., Van Custem, T., and Ribbens-Pavella M., “Extended equal area criterion justifications, generalizations, applications,” IEEE Transactions on Power Systems, vol. 4, issue 1, pp. 44-52, Feb. 1989

Yan W., Lu S., and Yu D. C., “A novel optimal reactive power dispatch method based on an improved hybrid evolutionary programming technique,” IEEE Transactions on Power Systems, vol. 19, no. 2, pp. 913-918, May 2004

Yuan Y., Kubkawa J., an Sasaki H., “A Solution of Optimal Power Flow With Multicontingency Transient Stability Constraints,” IEEE Trans. On Power Systems, Vol.18, No.3, August 2003, pp. 1094-1102

Yuryevich J. and Wong K. P., “Evolutionary programming based optimal power flow algorithm,” IEEE Trans. Power Systems, vol. 14, pp. 1245-1250, Nov. 1999

Zhang X., Dunn R. W., and Li F., “Stability constrained optimal power flow for the balancing market using genetic algorithms,” IEEE Power Engineering Society General Meeting, Vol. 2, July 2003

Zimmerman R. D., Gan D., “MATPOWER: A MATLAB Power System Simulation Package,” Power Systems Engineering Research Center, Cornell University, 1997. http://www.pserc.cornell.edu/matpower

Appendix: List of Abbreviation

ACT	Average Computational Time
ALU	AlgorithmLogic Unit
API	Application Programmer’s Interface
CCT	Critical Clearing Time
COI	Center of Inertia
DAEs	Differential-Algebraic Equations
DE	Differential Evolution
EA	Evolutionary Algorithm
EP	Evolutionary Programming
GA	Genetic Algorithm
IPM	Interior Point Method
KE	Kinetic Energy
LP	Linear Programming
MIMD	Multiple Instruction Multiple Data Stream
MISD	Multiple Instruction Single Data Stream
MPI	Message Passing Interface
NLP	Non-linear Programming
NTP	Network Time Protocol
OPF	Optimal Power Flow

ORNL	Oak Ridge National Laboratory
PE	Potential Energy
POZ	Prohibited Operating Zones
PSO	Particle Swarm Optimization
PVM	Parallel Virtual Machine
QP	Quadratic Programming
RDE	Re-initialized Differential Evolution
RSL	Remote Shell Login
SIMD	Single Instruction Multiple Data Stream
SISD	Single Instruction Single Data Stream
SSH	Secure Shell Login
STD	Standard Deviation
TEF	Transient Energy Function
TSCOPF	Transient Stability Constrained Optimal Power Flow