Electrical Power and Energy Systems 33 (2011) 1457–1461

Contents lists available at ScienceDirect

Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Optimal power system operation using parallel processing system and PSO algorithm

Jong-Yul Kim [a,*], Kyeong-Jun Mun [b], Hyung-Su Kim [c], June Ho Park [a]

[a] Pusan National University, Geumjeong-gu, Busan, South Korea
[b] Korea Atomic Energy Research Institute, Daejeon, South Korea
[c] Namhae College, Gyeongnam, South Korea

ARTICLE INFO

Article history:
Received 2 December 2008
Received in revised form 15 February 2011
Accepted 1 June 2011
Available online 2 July 2011

Keywords:
Particle swarm optimization
Heuristic
Optimal power flow
PC cluster system
Parallel processing

ABSTRACT

In recent studies, PSO algorithm is applied to solve OPF problem. However, population based optimization method requires higher computing time to find optimal point. This shortcoming is overcome by a straightforward parallelization of PSO algorithm. The developed parallel PSO algorithm is implemented on a PC-cluster system with 8 Intel Pentium IV 2 GHz processors. The proposed approach has been tested on the test systems. The results showed that computing time of parallelized PSO algorithm can be reduced by parallel processing without losing the quality of solution.

1. Introduction

Optimal Power Flow (OPF) is a useful tool in planning and operation of a power system. The OPF problem can be described as the optimal allocation of power system controls to satisfy the specific objective function such as fuel cost, power loss, and bus voltage deviation. The control variables include the generator real powers, the generator bus voltages, the tap ratios of transformer and the reactive power generations of VAR sources. Therefore, the OPF problem is a large-scale highly constrained nonlinear non-convex optimization problem [1]. Recently, many heuristic optimization methods in [2–4] to overcome the limitations of the mathematical programming approaches have been investigated. Particle Swarm Optimization (PSO) is a newly proposed population based heuristic optimization algorithm [5]. Compared with other heuristic optimization methods, PSO has comparable or even superior search performance for some hard optimization problems in real power systems [6–8]. However, population based optimal research methods such as GA, EP and PSO require relatively higher computing time than conventional optimization techniques. In parallel processing, problems are divided into several sub problems, and allocated to each processor. This can reduce computing time and enhance computation efficiency [9]. In this paper, parallel PSO algorithm is proposed to improve the computing time and also PC-cluster system is developed to implement parallel PSO algorithm. To verify the usefulness of the proposed algorithm, parallel PSO algorithm has been tested and compared with standard PSO algorithm having with single processor. The standard IEEE 30 and 118-bus power systems have been employed to carry out the simulation study.

2. Optimal power flow problem formulation

The OPF problem can be formulated as a constrained optimization problem as follows:

$$\textit{Minimize} \quad f(x,u) \tag{1}$$

$$\textit{subject to} \quad g(x,u) = 0 \tag{2}$$

$$h(x,u) \leqslant 0 \tag{3}$$

where x is a set of state variables, and u is a set of controllable variables.

In this paper, the objective function of OPF is minimization of fuel cost for all generators which can be formulated as follows:

$$Min\, f(P_{gi}) = \sum_{i=1}^{N_g} \left(a_i + b_i P_{gi} + c_i P_{gi}^2\right) \tag{4}$$

where $f(P_{gi})$ is the total fuel cost ($/h) of all generators; P_{gi} is the active power output generated by the ith generator; a_i, b_i, c_i are fuel cost coefficients; and N_g is the total number of generators. The

* Corresponding author. Tel.: +82 55 280 1336; fax: +82 55 280 1339.
E-mail address: jykim@keri.re.kr (J.-Y. Kim).

0142-0615/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijepes.2011.06.026

equality constraints $g(x, u)$ are the nonlinear power flow equations which are formulated as follows:

$$P_{gi} - P_{di} - V_i \sum_{j=1}^{N_b} V_j Y_{ij} \cos(\theta_i - \theta_j - \varphi_{ij}) = 0 \quad i = 1, \ldots, N_g \tag{5}$$

$$Q_{gi} - Q_{di} - V_i \sum_{j=1}^{N_b} V_j Y_{ij} \sin(\theta_i - \theta_j - \varphi_{ij}) = 0 \quad i = 1, \ldots, N_g \tag{6}$$

where P_{gi} and Q_{gi} are the active and reactive power generations at bus i; P_{di} and Q_{di} are the active and reactive power demands at bus i; V_i and V_j are the voltage magnitude at bus i and j respectively; θ_i and θ_j are the voltage angles at buses i and j respectively; φ_{ij} is the admittance angle; Y_{ij} is the admittance magnitude; and N_b is the total number of buses.

The OPF inequality constraints, $h(x, u)$, represent limits of control variables and state variables. The system operation constraints consist of the transmission line loadings, load bus voltages, reactive power generations of generator, and active power generation of slack generator. These variables should be within the set lower and upper limits.

$$S_i \leqslant S_i \leqslant S_i^{max} \quad i = 1, 2, \ldots, N_l \tag{7}$$

$$V_i^{min} \leqslant V_i \leqslant V_i^{max} \quad i = 1, 2, \ldots, N_b \tag{8}$$

$$Q_{gi}^{min} \leqslant Q_{gi} \leqslant Q_{gi}^{max} \quad i = 1, 2, \ldots, N_g \tag{9}$$

$$P_{gs}^{min} \leqslant P_{gs} \leqslant P_{gs}^{max} \tag{10}$$

Concerning control variables, active power output and voltage of generators, transformers tap ratio, and shunt capacitors are restricted by lower and upper limits as follows:

$$P_{gi}^{min} \leqslant P_{gi} \leqslant P_{gi}^{max} \quad i = 1, 2, \ldots, N_g - 1 \tag{11}$$

$$V_{gi}^{min} \leqslant V_{gi} \leqslant V_{gi}^{max} \quad i = 1, 2, \ldots, N_g \tag{12}$$

$$t_i^{min} \leqslant t_i \leqslant t_i^{max} \quad i = 1, 2, \ldots, N_t \tag{13}$$

$$sh_i^{min} \leqslant sh_i \leqslant sh_i^{max} \quad i = 1, 2, \ldots, N_{sh} \tag{14}$$

3. Parallel computation of PSO algorithm using PC clustering

3.1. PC cluster system

After mid 1980, high performance computers have been needed according to the development of large scale science and engineering. Since supercomputers are expensive, cluster systems replaced supercomputers because it has the availability of inexpensive high performance PCs, and high speed networks, and development of integrated circuits. PC cluster system provides higher availability as well as greater performance by lower cost with interconnecting several PCs or workstations. PC cluster system is very competitive with parallel machine in terms of a ratio of cost to performance because clustering is one of the types of parallel or distributed processing system, which is composed of a collection of interconnected low cost PCs working together as single and integrated computing resources. Also, it is easy to add nodes that construct the PC cluster. A basic construction diagram for PC cluster is shown in Fig. 1.

The performance of the PC cluster system depends on the quality of message passing system, libraries, and compilers for parallel programming and performance of individual nodes. Therefore, it is important to select each component described above properly to obtain better performance. The PC cluster system implemented in this paper is composed of eight nodes based on fast Ethernet with Ethernet switch. For operating system, master node uses Windows 2000 server, and slave nodes use Windows 2000 pro. To connect each node, fast Ethernet card and switching hub were used. In data communication, MPI library was used, which is effective for parallel application by using message-passing method through TCP/IP over Internet. Symantec PC anywhere was used for remote control of each node, and MS visual C++ 6.0 was used for compilers of parallel programming. Table 1 shows the picture and the specification of the PC cluster system developed in this paper.

3.2. Parallel computing of PSO algorithm

The PSO is basically developed through the simulation of bird flocking in two-dimensional space. In PSO, each particle i ($i = 1, \ldots, N$) in the population is characterized by three vectors (x_i, v_i, p_i) which represent their temporal position, velocity, and the best position. The fitness of each particle is given by the function value $f(x_i)$. Each particle stores its best position p_i called personal best, p-best, which gives the best fitness in memory. They can also consult their neighbor's best position. Most simply, the neighbor is the whole population (fully connected topology), and therefore, the neighbor's best is the best position among personal bests of the whole population. Hence, the position p_g is called global best. Now each particle i moves around the search space, and renews its velocity using its past experience (personal best) and the population's experience (global best) as follows:

$$v_i = \omega v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i) \tag{15}$$

The parameter c_1 and c_2 are the acceleration constant, r_1 and r_2 are the uniform random numbers within the range [0, 1]. If v_i is larger than a predefined velocity v_{max} called maximum velocity, it is set to v_{max}. Similarly, if it is smaller than $-v_{max}$, it is fixed to $-v_{max}$. The parameter ω is called inertia weight [10], which controls the exploration (global search)–exploitation (local search) tradeoff.

$$\omega = (\omega_{ini} - \omega_{fin}) \times (MAX_{iteration} - Iteration)/MAX_{iteration} + \omega_{fin} \tag{16}$$

Then the particle changes its position by the "equation of motion":

$$x_i = x_i + v_i \tag{17}$$

The population size is one of the key factors that will affect the search performance of the PSO algorithm for seeking the optimal solution. The larger population size can guarantee the higher chance of obtaining the optimal solution. However, it is obvious that more computing time is needed. To reduce the computing time with same quality of solution, parallel PSO algorithm is proposed and paralleled by the PC cluster system. The most important issue of parallelizing PSO algorithm is exchange model of evolution information. Different ways will result in different performances. The proposed configuration is a kind of parallel algorithm based on coarse grain model, in which the population is divided into some sub-populations evolving independently.

Each sub-population exchanges require information only between two neighboring sub-populations connected by arrowed lines as shown in Fig. 2. Each sub-population is allocated in each processor that involves in parallel computing. With each processor that can communicate with the neighboring sub-populations, the best solution of each processor is transferred to the neighboring processors by migration operation every generation. The flowchart