Phase Balancing using Simulated Annealing

Jinxiang Zhu
Member, IEEE
Power Systems Energy Consulting
GE Power Systems
Schenectady, NY 12345

Griff Bilbro
Senior Member, IEEE

Mo-Yuen Chow
Senior Member, IEEE

Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27695-7911

ABSTRACT: Deregulation eliminates the boundary of the territory of the monopoly power industry. Competition forces utilities to improve power quality as well as to reduce investment and operation costs. Feeder imbalance describes a situation in which the voltages of a three-phase voltage source are not identical in magnitude, or the phase differences between them are not 120 electrical degrees, or both. It affects motors and other devices that depend upon a well-balanced three-phase voltage source. Phase balancing is to make the voltages balanced at each load point of the feeder. Phase swapping is a direct approach for phase balancing with the minimum cost. Phase balancing can enhance utilities' competitive capability by improving reliability, quality, and reducing costs. Therefore, *phase balancing* optimization is nowadays receiving more attention in the power industry, especially in today's deregulating environments. The non-linear effects, such as, voltage drops and energy losses, make the problem difficult to solve. This paper introduces Simulated Annealing as an effective method to solve a power distribution phase balancing problem with its non-linear effects.

Key Words: Phase balancing optimization, Feeder Reconfiguration, Phase Swapping, Power Quality, Deregulation, Simulated Annealing (SA), Candidate Set.

1. Introduction

In power distribution systems, single or two-phase laterals and unbalanced loading are common phenomena. Unbalanced feeders not only increase energy losses and the risk of overload situations, but they also affect system power quality and electricity price. Even a feeder system is designed as a balanced feeder based on the given load data, load prediction errors and unbalanced load growth will induce feeder imbalance. The purpose of phase balancing is to find the optimal phase swapping scheme to balance an unbalanced feeder system with minimum cost. Reference [1,2] has discussed the phase balancing problem in details.

Utilities pursue balanced operating conditions in their distribution systems. A balanced system has smaller peak load voltage drops and energy losses. Phase balancing improves voltage balance, security, reliability, utilization factors of existing facilities and defer feeder expansion projects. As a result, a utility can provide power service with higher quality and lower cost, which will enhance the utility's competitive edge in the deregulated environments.

PE-171-PWRS-0-10-1998 A paper recommended and approved by the IEEE Power System Planning and Implementation Committee of the IEEE Power Engineering Society for publication in the IEEE Transactions on Power Systems. Manuscript submitted April 21, 1998; made available for printing November 10, 1998.

There are two approaches for phase balancing. One is *feeder reconfiguration* at the system level; the other is *phase swapping* at the feeder level. Feeder reconfiguration has been extensively studied in the past several decades while phase swapping has been ignored. Since feeder reconfiguration is primarily designed for load balancing among the feeders, most researchers do not consider phase balancing as an objective in feeder reconfiguration. Only a few people incorporated phase balancing into feeder reconfiguration approaches based on the unbalanced feeder systems[3-6]. But they realized that feeder reconfiguration has limitation to reach phase balancing [3-6].

Since phase imbalance is usually not an emergency condition as long as it does not cause device overloading, extensive voltage drops, or trip ground relays, phase swapping is normally carried out during maintenance and restoration periods. Phase swapping is a direct and effective way to balance a feeder in terms of phases. For the time being, engineers already use phase swapping to balance phase loading based on their experiences and trial and error methods. The procedure is labor intensive, sub-optimal, and time-consuming.

Phase swapping problem is an *Optimal Power Flow* problem. For example, the cost function is to minimize the energy losses, subject to load flow equations, voltage requirement, capacity constraints, and the phase balancing requirements. The control variables are the phase swapping options at each candidate node [2]. Phase swapping is a large-scale combinatorial optimization problem and is *NP-complete* (non-deterministic polynomial time complete). Thus its computing effort increases exponentially with the size of candidate set. We have formulated phase balancing problem into Mixed-Integer Programming (MIP) [1,2], which is suitable for the linear objective function. But in many cases, the linear function may not well represent some decision-making criteria. For example, minimizing energy losses will make the objective function as a non-linear integer function, which is difficult to be solved analytically. While intelligent computation methods, such as Fuzzy Logic, Genetic Algorithm, and Simulated Annealing, are powerful to solve nonlinear integer programming problems. In this paper, we will apply simulated annealing (SA), which is a heuristic optimization method for large-scale *NP-complete* problems, to solve the phase swapping problem.

SA can conquer the large-scale non-linear integer programming problems [10]. Even it cannot guarantee the optimality of the final solution, SA has the potential to avoid local optimal solutions and converge to the global optimal

solutions [10]. SA can provide an acceptable solution for a *NP-complete* problem within a reasonable computing time.

We will first briefly describe simulated annealing algorithm. Then, a non-linear objective function and detailed implementation of SA to solve phase balancing problem are presented. Finally, the SA is compared with quenching algorithm and greedy algorithm in terms of efficiency and optimality. The necessary and sufficient conditions for simulated annealing are also discussed. An example feeder is used to illustrate the proposed algorithm to obtain a global optimal solution.

2. Simulated Annealing Algorithm

SA is a simulation of annealing process of molten metals. Let's briefly describe matter physics, especially annealing process first. The number of atoms in samples of liquid or solid matter is of order 10^{23} per cubic centimeter, only the most probable behavior of the system in thermal equilibrium at a given temperature is observed in experiments. Therefore, *statistical mechanics* is the central discipline of condensed matter physics. The average behavior of the system is defined by the set of atomic positions $\{r_i\}$, which is weighted by its Boltzmann probability factor, $\exp\left(-E\left(\{r_i\}\right)/k_B T\right)$, where $E\left(\{r_i\}\right)$ is the system energy, k_B is Boltzmann constant, and T is temperature. In practice, low temperature is not a sufficient condition for forming a crystalline solid (with the lowest energy states). It is done by careful annealing, first melting the substance, and spending a long time at temperatures in the vicinity of the freezing point. Otherwise the resulting crystal will have many defects, or the substance may form a glass, with no crystalline order and only metastable, locally optimal structure.

Metropolis et al. introduced a simple algorithm that can be used to provide an efficient simulation of a collection of atoms in equilibrium at a given temperature [7]. In each step of the algorithm, an atom is given a small random displacement and resulting energy change, ΔE. If $\Delta E \le 0$, the displacement is accepted, and the configuration with the displaced atom is used as the starting point of the next step. The case $\Delta E > 0$ is treated probabilistically: the probability that the configuration is accepted is $\exp\left(-\Delta E/k_B T\right)$. By repeating the basic step many times, one simulates the thermal motion of atoms in thermal contact with a heat bath at temperature T.

Kirkpatrick et al. extended *Metropolis algorithm* to *Simulated Annealing* (SA) for approximate numerical simulation of the behavior of system at a finite temperature. SA provides a natural tool for bringing the techniques of statistical mechanics to bear on optimization [8]. It is a heuristic random search process to solve the combinatorial problems and non-linear / non-derivative problems, such as, integrated circuit design, traveling salesmen problem, etc. SA can find slightly better solution than most other heuristic methods, such as sequential algorithm [9].

SA allows perturbations to move deteriorated solution in a controlled fashion. It is essentially a simulation of annealing process of molten metals. The iteration number used in the SA is analogous to temperature level in the annealing process. A candidate solution is generated for each iteration. If this solution is a better one, then it is accepted and is used to generate the next candidate solution. If the solution is a deteriorated one, it is accepted when its probability of acceptance (Boltzmann distribution) is greater than a random number between 0 and 1. In the next iteration, temperature is reduced according to some function of system states. The deteriorated solution is accepted with less and less probability through iterations. The solution process continues until the maximum number of iteration is reached or the optimal solution is found. Because the deteriorated solution is acceptable, it is possible to jump out of local minima and potentially find the global minima. Since the deteriorated solution is accepted carefully, it can be proved that SA converges asymptotically [10].

What are the necessary and sufficient conditions for simulated annealing convergence? Simulated annealing can be modeled probabilistically as Markov chains. SA are (1) The Markov chain associated with $G(c_k)$, generation probability matrix at kth iteration, is irreducible, in another words, there is a positive probability of reaching state j from any state i in a finite number of transitions. (2) For any states j and i, i is reachable from j if and only if j is reachable from i. The optimal state ($X(k) \in \Re_{opt}$) must be reached from initial state when iteration k is large enough, that is, $\lim_{k\to\infty} \Pr\left(X(k) \in \Re_{opt}\right) = 1$ [10].

One step transition matrix:

$$P_{ij}(k-1,k) = \begin{cases} G_{ij}(c_k) \cdot A_{ij}(c_k) & \forall j \neq i \\ 1 - \sum_{l=1, l\neq i}^{R} G_{il}(c_k) \cdot A_{il}(c_k) & j = i \end{cases} \quad (1)$$

where: $A_{ij}(c_k) = \min\left\{1, \exp\left(\left(C(i) - C(j)\right)/c_k\right)\right\}$, acceptance probability

$G_{ij}(c_k) > 0$, generation probability of generating j from i

$C(i)$, cost function (objective) for state i.

c_k, the temperature at kth iteration. $\lim_{k\to\infty} c_k = 0$ and $c_k \ge c_{k+1}$

3. Objective Function and Constraints

To investigate the effects of voltage drops and/or losses, a full AC load flow model have to be considered in the phase swapping formulation. SA has advantages to solve problems with non-linear, non-derivative, discrete objective functions

and constraints. For phase swapping problems, the cost function of each candidate solution is easily evaluated after updating load flows (call AC load flow program or DC load flow program). For a radial feeder system, the voltages and its angles can be obtained by forward sweep and the losses can be obtained by backward sweep [11].

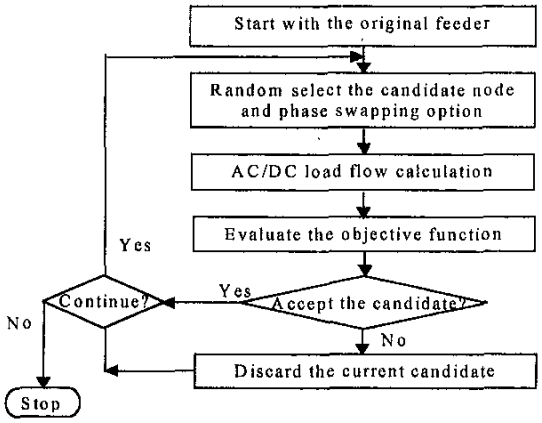

Figure 1. Flow Chart of Application of SA to the phase balancing problems

Load flow calculation is separated from the optimization process. That is, once a phase swapping candidate is generated, run AC load flow to update the load flows first, then check the validation constraints and evaluate the objective function based on the updated load flows. SA will decide whether the candidate is accepted or discarded. The process continues until the final temperature is reached.

SA gives us a freedom to define the objective function and constraints. Therefore the phase swapping formulation can be closer to decision-maker's expectation. For example, phase swapping cost is proportional to the number of phase swapping. But the benefits of phase swapping need to cover energy loss reduction, improve reliability and service quality, etc. The benefit function is difficult to be defined. But we realize that the smaller the flow is, the less possibility to be overloaded. For a radial feeder, the sum of the flow for each phase is a constant disregarding to phase swapping. Therefore the ideal flows are set at the mean of the flows for each phase. A penalty cost function, is proposed to coerce the flows towards as small as possible. Therefore each phase flow will converge to the mean of flows. We define some new terms as following:

Number of phase swapping:

$$H_1 = \sum_i \xi\left(T_i^k, T_i^0\right) \tag{2}$$

$$\text{where } \xi\left(T_i^k - T_i^0\right) = \begin{cases} 0 & \text{if } T_i^k = T_i^0 \\ 1 & \text{if } T_i^k \neq T_i^0 \end{cases} \tag{3}$$

The penalty cost of the flows:

$$H_2 = \sum_j w_j \cdot \sum_{ph} g\left(\frac{f_{j,ph}}{C_j}\right) \tag{4}$$

$$\text{where } g(x) = \begin{cases} 0 & \text{if } x < 0.3 \\ e^{x-1} - e^{-0.7} & \text{if } 0.3 \le x < 1.0 \\ e^{2(x-1)} - e^{-0.7} & \text{if } x \ge 1.0 \end{cases} \tag{5}$$

The overall index for phase swapping problem:

$$H = \delta \cdot H_1 + \tau \cdot H_2 . \tag{6}$$

where,

$f_{j,ph}$ is the phase flow on branch $j,ph = a, b, c$

T_i^0 is the original tapping scheme at node i,

T_i^k is the tapping scheme at node i in kth result candidate,

C_j is the capacity of jth line segment

w_j is the weighting factor of one segment j

δ is the cost for each node phase swapping

τ the coefficient for unbalanced penalty cost

H is the system balancing overall index

The number of phase swapping, H_1, is an indicator for the costs of phase swapping. *The penalty function,* H_2, is an indicator for the contra-benefit of phase balancing, i.e. the smaller the *penalty function is*, the more balanced the feeder is. It covers some special considerations, for example, there is no penalty to the light loaded feeders. The penalty grows fast if the flow is approaching its capacity, especially for overloaded situation. The weighting factor is assigned based on the importance of each branch. *The system balancing overall index, H,* is to compromise the cost and the benefits for phase swapping.

For the penalty cost function defined in (4), if the flow is far below the 30% of capacity, then the penalty cost is zero no matter how unbalanced it is. The penalty costs increase faster as flows increasing. Consequently, phase balancing focuses more on heavily loaded branches than the lightly loaded branches. When the phase swapping cost and penalty cost function are involved, phase swapping problem becomes a nonlinear integer-programming problem. The problem is difficult to solve by analytical methods. But heuristic methods can be good alternatives.

We will minimize the total cost (phase swapping cost and penalty cost) defined in (5). The objective function is a non-linear and non-derivative function that is difficult to be solved by traditional techniques. However, SA is easy to handle nonlinear and non-derivative objective function. Since the candidate of phase swapping is easy to be generated and the objective function is easy to be evaluated also, simulated annealing is good at solving the large-scale phase swapping

problems. At the same time, the constraint violation is checked in a straight forward manner. If the candidate violates the constraints, the candidate is discarded.

4. Implementation Procedures

For the phase balancing problem, the unbalanced node and load tapping are randomly selected to generate a candidate solution based on the current feeder tapping scheme. DC load flow is performed to update the system status. The index H in equ. (5) is then evaluated. If the H value increases (deteriorated solution), then the candidate is accepted when its probability of acceptance based on the Boltzmann distribution is greater than a uniform random number between 0 and 1. If H decreases (better solution), then the candidate is accepted. Once the candidate is accepted, the next candidate is generated from the new solution. As the iterations going on, the acceptance probability of the deteriorated solution is decreasing. The search process terminates when the temperature is cool enough.

SA for phase swapping problem needs four basic components [12]:

Configuration: a model of what a legal phase swapping is. One and only one load is assigned to phase line at a node.

Move Set: At each admissible unbalanced node / lateral, there are at most 6 phase tapping schemes. Each one is a valid move. If the candidate is identical with the current one, then generate another one. Each move will affect the upper stream flows.

Cost Function: The cost function can be unbalanced flow, balancing index, and the overall cost function defined in (6). In the following discussion, the overall cost function is to be minimized and therefore the feeder is balanced.

Cooling Schedule: is the ***key*** to ensure that SA is convergent to optimal solution. Specially, we need a starting hot temperature and rules to determine when the temperature should be lowered, how much the temperature should be lowered, and when annealing should be terminated. In summary, the cooling schedule is defined in the four terms: initial temperature, final temperature, number of iterations, and cooling rate.

Initial Temperature ($t_{initial}$): The initial temperature is determined in such a way that practically all possible phase swapping could be made. The initial temperature selected here is the biggest difference of objective function for any two phase swapping schemes.

Final Temperature (t_{final}): The final temperature is determined in such a way that at the optimal point the expected improvements in the objective function become negligible.

Number of iterations (N): The number of iterations is determined by the k ($k = 10 \sim 50$) times of the number of possible options. In the example, k is *10*. The number of options is dependent on the number of nodes in the feeder and the number of the phases at each node.

Cooling Rate ($r_{cooling}$): The temperature cooling rate is about 0.50-0.99. It is the function of the initial temperature, the final temperature, and the number of iterations. Since only one candidate is performed at each temperature level, the cooling rate is determined by

$$r_{cooling} = \sqrt[N]{\frac{t_{final}}{t_{initial}}} \tag{7}$$

For example, initial temperature is 450, final temperature is 0.1, and the number of iterations is 300. Then the cooling rate is 0.9723.

Annealing process requires to spend a long time at the temperatures in the vicinity of the freezing point so that bigger crystal is resulted. For simulated annealing, the temperature goes down at the beginning faster than at the end. For example, the cooling rate is 0.95 at the first 50 iterations, then it changes to 0.99 afterwards. Spending a long time enough at temperature in the vicinity of the freezing point ensures the optimal solution. This will save the computation efforts by 100 iterations while the optimality will not be affected at all.

5. Comparison with Greedy Algorithm, Quenching Algorithm, and Simulated Annealing

Greedy algorithm is to randomly pick up a variable and set it to the optimal value with the others unchanged. This process continues until the 20 consecutive iterations can not improve. Greedy algorithm is highly dependent on the initial status and the random numbers. For the phase swapping problem, the greedy method is to randomly select a node in the feeder and evaluate all the possible tapping options at this node. The best tapping scheme is taken for each candidate node. The new system is at least not a deteriorated solution. If a node is selected more than once consecutively, phase swapping evaluation is processed only once. The process will stop when the pre-defined epoch limit is reached or when system objective cannot improve in 20 consecutive iterations. The greedy algorithm cannot avoid the local minimum, thus does not guarantee to get the optimal phase balancing scheme.

Quenching algorithm is the same as simulated annealing algorithm except that it only accepts the candidate when the objective function is improved. In quenching algorithm, starting with the current system status, we try to perturb the known solution to improve the objective. For the phase swapping problem, a random phase swapping scheme for a randomly selected node is generated. Then the system

objective index is evaluated. If the objective is improved, then the change is accepted; otherwise the change is discarded. The process stops when the improvements cannot be found in 100 consecutive iterations. The result is highly dependent on the initial status and the random number series. Quenching algorithm, each new perturbation moves the system to a restrictedly improved solution from the previous one, may be trapped in a local minima in many cases. Therefore quenching algorithm is carried out several times with the different starting point and different random series, and the best solution is saved as the global optimum.

Simulated Annealing accepts the deteriorated solution probabilistically [10]. For the phase swapping problem, a random phase swapping scheme for a randomly selected node is generated. Then the system balance index is evaluated. If the objective is improved, then the change is accepted. If the new solution is a deteriorated solution, then it is accepted if the acceptance rate (calculated based on Boltzmann distribution function) is greater than a uniform random number between 0 and 1; otherwise it is rejected. The process continues until the final temperature is reached. SA is not sensitive to the initial status of the system and the random number series. It can find the global optima in most cases.

In this section, three approaches are proposed and compare to find the optimal solution to the phase balancing problem, which is a nonlinear combinatory problem. The only differences among these algorithms lie in the acceptance rules and randomness of the candidate. For example, greedy algorithm only randomly select a node and pick up the best phase swapping. Quenching algorithm randomly pick a node and a phase swapping scheme but only the better solution is acceptable. Simulated annealing not only randomly pick up a node and a phase swapping scheme but also accept the deteriorated solution probabilistically. Theoretically SA will find the global optima in most cases but it takes longer time to solve the problem. Greedy algorithm is the fastest method but the optimality is not as good as the other two methods. When we select the approach, we need to compromise the optimality and the efficiency of the algorithm.

6. Illustrative Example

A five-node system in Figure 2 is used to illustrate the Simulated Annealing algorithm for phase swapping problem. At same time, the quenching algorithm and greedy algorithm are also applied to the example.

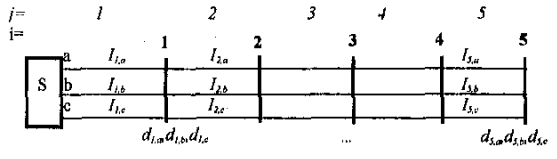

Figure 2: A 5-node feeder system

The cost function for this phase swapping problem is H (6). It is a nonlinear and non-derivative integer function. This optimization problem cannot be solved by conventional mathematical methods, while can be solved by Simulated Annealing, Quenching Algorithm, and Greedy Algorithm. The results of the five-node system for 100 runs are listed in Table 1. The number 79 means greedy algorithm converges to optima (the minimum cost is 438) 79 times of 100 runs. The most runs are convergent to the optima. The best approach is quenching algorithm method for this problem. If the system is larger, then SA performance will be better.

Table 1. Comparison on optimality

Solution / Method	Optima (438)	Near-optima 1 (444)	Near-optima 2 (448)
Greedy	79	0	21
Quenching	83	3	14
SA	52	29	19

To compare the optimality and efficiency of the proposed approaches, many tests have been done on from 10 to 30-candidate systems with random unbalanced loading. For each system, each approach runs 20 times, which is large enough to provide statistical significant results, to get the average cost value and the average CPU times. Every approach pursues convergence for each individual run.

Table 2. Comparison on computation time

# Candidate / Method	10	15	20	25	30
Greedy	17	32	61	77	179
Quenching	14	41	69	105	184
SA	63	102	207	334	546

Table 3. Comparison on optimal cost

# Candidate / Method	10	15	20	25	30
Greedy	661	539	991	931	1645
Quenching	661	539	991	931	1645
SA	660	539	991	928	1643

Table 4. Comparison on average cost

# Candidate / Method	10	15	20	25	30
Greedy	668	550	1001	943.	1660
Quenching	665	547	1000	939	1659
SA	672	543	997	943	1658

The results in Table 2-4 indicate that annealing can find the better solution than other methods in most cases.

However SA is a time-consuming approach for phase balancing problems than other methods. Quenching algorithm is faster and its solution is reasonable good. Greedy algorithm is slightly worse than Quenching algorithm from optimality point of view. Quenching algorithm and Greedy algorithm are also good methods to solve large-scale phase balancing optimization problems if the optimality is not a critical issue. Even SA, quenching algorithm, and greedy algorithm do not guarantee the optimality but they provide solutions that are close to the optimal one. But only Simulated Annealing has potential to avoid the local minima.

7. Conclusion

Growing pressures on competition have forced utilities to seek any opportunity to reduce cost and to improve quality. Phase swapping is one of effective way to reach their goal. But if phase balancing problems are modeled as non-linear integer programming, then it is difficult to solve by the traditional methods such as Mixed Integer Programming. Simulated Annealing is a promising approach to solve the large-scale feeder systems. It can not guarantee the optimal solution but it has potential to avoid local minimum. It is time-consuming method but it can provide a better solution than other heuristic methods (e.g. Greedy algorithm, Quenching algorithm). It can solve the phase swapping problems considering voltage drops and energy losses, which optimization methods are not afford to solve large scale phase balancing problems.

8. References

[1] J. Zhu, M.-Y. Chow, and F. Zhang, "Phase Swapping to Balance a Radial Feeder System," presented in *IEEE PES, Winter Meeting'98*.

[2] J. Zhu, M.-Y. Chow, and K. Hoffman, "Phase Balancing Optimization for a Large-Scale Feeder System," *submitted to IEEE Transactions on Power Systems*, 1997.

[3] Y.-Y. Hsu, Y. Jwo-Hwa, S. S. Liu, Y. W. Chen, H. C. Feng, and Y. M. Lee, "Transformer and Feeder Load balance Using a Heuristic Search Approach," *IEEE Transactions on Power Systems*, vol. 8, pp. 184-90, 1993.

[4] W.-M. Lin and H.-C. Chin, "Optimal Switching for Feeder Contingencies in Distribution Systems with Fuzzy Set Algorithm," *IEEE*, 1996.

[5] J.-C. Wang, H.-D. Chiang, and G. R. Darling, "An Efficient Algorithm for Real-Time Network Reconfiguration in Large Scale Unbalanced Distribution System," *IEEE Transactions on Power Systems*, vol. 11, pp. 511-7, 1996.

[6] V. Borozan, "Minimum Loss Reconfiguration of Unbalanced Distribution Networks," *IEEE winter meeting*, vol. 96 WM 343-4 PWRD, 1996.

[7] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Journal of Chem. Physics," *Journal of Chem. Physics*, vol. 21, pp. 1087, 1953.

[8] S. Kirkpatrick, C. D. Gelatt, and J. M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671-80, 1983.

[9] R. Romero, R. A. Gallego, and A. Monticelli, "Transmission System Expansion Planning by Simulated Annealing," *IEEE Transactions on Power Systems*, vol. 11, pp. 364-369, 1996.

[10] P. J. M. V. Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Application*: D. Reidel Publishing Company, 1987.

[11] F. Zhang and C. S. Cheng, "A Modified Newton Method for Radial Distribution System Power Flow Analysis," *IEEE Transactions on Power Systems*, 1996.

[12] R. Rutenbar, "Simulated Annealing Algorithms: An Overview," *IEEE Circuits and Devices Magazine*, pp. 19-26, 1989.

Jinxiang Zhu received his B.S. degree from Tsinghua University in Beijing PRC, and his M.S. and Ph.D. degree from North Carolina State University in Electrical Engineering, respectively in 1990, 1994, and 1997. His research interests include power system operation and planning, optimization techniques, fuzzy logic, and neural networks. He is a member of IEEE. Currently he is an application engineer at General Electric Company. Dr. Zhu is working on Market Simulation and Analysis for the deregulated power systems.

Griff Billbro is IEEE senior member. B.S. Physics, Case Western Reserve University, 1973; M.S. Physics, University of Illinois, Urbaba, 1975; Ph.D. University of Illinois, Urbana, 1977. He currently holds Associate Prfessor of Electrical and Computer Engineering, North Carolina State University. Dr. Billbro" current research interests include developing global optimization algorithms for communications and signal processing, modeling electron devices in silicon carbide and diamond, and implementing non-linear algorithms in analog integrated circuit.

Mo-yuen Chow earned his B.S. degree at the University of Wisconsin-Madison (1982); M. Eng. degree (1983) and Ph.D. degrees at Cornell University (1987), all in Electrical Engineering. Upon completion of his Ph.D., Dr. Chow joined the faculty of North Carolina State University in Raleigh, NC, where he is presently an Associate Professor in the Department of Electrical and Computer Engineering. Since 1987, Dr. Chow has been working as a Principal Investigator in several projects in the areas of system monitoring, fault detection and control, applications of artificial neural network and fuzzy logic to power engineering. He is currently an Associate Editor of the *IEEE Transactions on Industrial Electronics*. He is also listed in *Who's Who in Asian Americans*.