Dynamic Multiobjective Optimization of Power Plant Using PSO Techniques

J. S. Heo, K. Y. Lee, Fellow, IEEE and R. Garduno-Ramirez, Member, IEEE

Abstract—The Coordinated Control Scheme (CCS) requires references to provide control inputs to a power plant. The references are obtained by mapping the unit load demand to pressure set-point. In order to achieve the optimal power plant operation, the mapping should be optimized under a dynamic environment by considering the multiobjectives of the power system. In this paper, the multiobjective optimal power plant operation will be realized through the on-line optimal mapping between the dynamic unit load demand and pressure set-point using a modern heuristic method, the Particle Swarm Optimization (PSO). The multiobjective optimization is performed in the reference governor of a Fossil Fuel Power Unit (FFPU). Moreover, variations of the PSO technique, such as Hybrid PSO (HPSO), Evolutionary PSO (EPSO), and Constriction Factor Approach (CFA), will be introduced and the comparison will be made on the dynamic multiobjective optimization of a power plant.

Index Terms—Coordinated control, multiobjective optimization, power plant, dynamic load demand, pressure set-point scheduling, particle swarm optimization.

I. INTRODUCTION

In the past few decades, the control techniques in the boiler-turbine power system have been developed steadily. After examining those strategies, it has been shown that the coordinated control strategy has more stable and faster response to load changes than other control techniques, such as boiler-following control and turbine-following control [1]. In order to implement the coordinated control scheme, it requires a mapping between the unit load demand and pressure set-point for the references. Moreover, this mapping should be optimized for the optimal power plant operation under the dynamic environment by considering the multiobjectives as follows:

- Minimization of load tracking error
- Minimization of fuel consumption
- Minimization of pollutant emissions
- Maximization of duty of life for the equipment.

Recently, there has been growing interest in Particle Swarm Optimization (PSO) to search for the global optimum value in a wide range of engineering problems. A PSO can provide simple implementation, high quality solutions, and fast computation.

Many researchers have applied a basic PSO or the variations of PSO to their specific applications [2]-[13]. The basic PSO technique in general provides a better performance in computation and implementation compared with GA [2]. As the GA, the fundamental principle of PSO technique is based on a random search in the solution space. Moreover, modified PSO techniques provide advantages over the basic PSO with better convergence, stability, and performance [3]-[11]. The PSO techniques are also well suited for applications in power systems to optimize various performance indexes for optimal operation [8]-[10].

Since the PSO originated from the research of Eberhart and Kennedy [11], it has been applied in a broad area of optimization problems and much progress has been made in the theoretical foundation. Although much research has been devoted to the development of the basic PSO and variations of it, little information is available on the comparison of the PSO techniques in multiobjective optimization problems, and in dynamic applications in power systems.

In this paper, the multiobjective optimal power plant operation will be realized through the on-line optimal mapping between the dynamic unit load demand and pressure set-point using the Particle Swarm Optimization (PSO) technique. The multiobjective optimization procedure will be performed in the reference governor of a Fossil Fuel Power Unit (FFPU). Moreover, the variations of PSO technique, such as Hybrid PSO (HPSO), Evolutionary PSO (EPSO), and Constriction Factor Approach (CFA), will be introduced and the comparison will be made on the dynamic multiobjective optimization of the power plant.

Following the introduction, the power plant control system is described in Section 2. Section 3 describes particle swarm optimization techniques (PSO, HPSO, EPSO and CFA). Section 4 describes the multiobjective optimization using PSO techniques in the FFPU. Section 5 shows simulation results to demonstrate the feasibility of the proposed approach. The final section draws some conclusions.

II. POWER PLANT CONTROL SYSTEM

A. Control Structure

The Coordinated Control Scheme (CCS) requires references (or set-points) for both power demand ($E_d$) and...
pressure demand \( (P_d) \). The control structure of the CCS is developed in three main modules: reference governor, feedforward controller, and feedback controller as shown in Fig. 1. The multiobjective optimization is performed in the reference governor. The results of the multiobjective optimization are the set points for the power and pressure \( (E_d \text{ and } P_d) \) for the feedforward and feedback controllers.

### B. Power Unit Model

The FFPU under study is a 160MW oil-fired drum-type boiler-turbine generator unit. It is represented by a third order Multiple Input-Multiple Output (MIMO) nonlinear model with three inputs and three outputs [13]. The inputs are positions of valve actuators that control the mass flow rates of fuel \( (u_1 \text{ in pu}) \), steam to the turbine \( (u_2 \text{ in pu}) \), and feedwater to the drum \( (u_3 \text{ in pu}) \). The outputs are electric power \( (E \text{ in pu}) \), drum steam pressure \( (P \text{ in kg/cm}^2) \), and drum water level deviation \( (L \text{ in m}) \). The state variables are electric power \( (E) \), drum steam pressure \( (P) \), and fluid (steam-water) density \( (\rho_f) \). The state equations are:

\[
\begin{align*}
\frac{dP}{dt} &= 0.9u_1 - 0.0018u_2 P^{9/8} - 0.15u_3 \\
\frac{dE}{dt} &= ((0.73u_2 - 0.16)P^{9/8} - E)/10 \\
\frac{d\rho_f}{dt} &= (14u_3 - (1.1u_2 - 0.19)P)/85
\end{align*}
\]

The drum water level output is calculated using the following algebraic equations:

\[
\begin{align*}
q_e &= (0.85u_2 - 0.14)P + 45.59u_1 - 2.51u_3 - 2.09 \\
\alpha_s &= (1/\rho_f - 0.0015)/(1/(0.8P - 25.6) - 0.0015) \\
L &= 50(0.13\rho_f + 60\alpha_s + 0.11q_e - 65.5)
\end{align*}
\]

where \( \alpha_s \) is the steam quality, and \( q_e \) is the evaporation rate \( (\text{kg/sec}) \). Positions of valve actuators are constrained to \([0,1] \), and their rates of change \( (\text{pu/sec}) \) are limited to:

\[
\begin{align*}
-0.007 &\leq du_1/dt \leq 0.007 \\
-2.0 &\leq du_2/dt \leq 0.02 \\
-0.05 &\leq du_3/dt \leq 0.05
\end{align*}
\]

### C. Operating Windows

In order to get optimal solution, the solution space must be predefined from the given model and constraints, (1)-(3). The solution space is obtained using power-input operating windows which are driven by the inverse static equations:

\[
\begin{align*}
u_1 &= (0.0018u_2 P^{9/8} + 0.15u_3)/0.9 \\
u_2 &= (0.16P^{9/8} + E)/0.73P^{9/8} \\
u_3 &= (1.1u_2 - 0.19)P)/141
\end{align*}
\]

However, the inverse static equations require the relationship between power and pressure. For this relationship, the equilibrium points need to be found from the given model and constraints. First, solve the given model (1) by setting the derivatives equal to zero and find all possible and meaningful equilibrium points. The resulting power-pressure operating window is shown in Fig. 2, which is represented by upper and lower limits. Secondly, determine the input operating windows by solving the inverse static model (4) for all points in the power-pressure operating window. Fig. 2 shows also the power-input operating windows corresponding to various power ranges in the solution space for the optimization.

### III. PARTICLE SWARM OPTIMIZATION

#### A. The Basic PSO

Basically, the PSO is developed through simulation of birds flocking in two-dimensional space [12]. The position of each bird (called agent) is represented by a point in the X-Y coordinates and also the velocity is similarly defined. Bird flocking assumed to optimize a certain objective function. Each agent knows its best value so far \((pbest)\) and its current position. This information is an analogy of personal experience of an agent. Moreover, each agent knows the best value so far in the group \((gbest)\) among \(pbests\) of all agents. This information is an analogy of an agent knowing how other agents around it have performed. Each agent tries to modify its position using the concept of velocity. The velocity of each agent can be updated by the following equation:
is weighting function, 1

the PSO. The initial

strongly on

mechanism. Since search procedure by the PSO depends

basic mechanism of the PSO and the natural selection

procedure by the PSO depends

differences between EPSO and the basic PSO are an explicit

parameters. Instead of moving the agents to find an optimal

selection procedure and self-adapting properties for its

parameters. Instead of moving the agents to find an optimal

solution in solution space, EPSO reproduces the agents with

the movement rule of PSO and the mutation rule of

Evolutionary Strategy (ES). Then, the best agents are selected

by stochastic tournament through the evaluation. The general

scheme of EPSO is replication, mutation, reproduction with

movement rule, evaluation, and selection. The movement rule

for EPSO is the following:

\[ v_{i}^{\text{new}} = w_{i}^{*} v_{i} + w_{r1}^{*} (pbest_{i} - s_{i}) + w_{r2}^{*} (gbest^{*} - s_{i}) \]  

(8.a)

\[ w_{i}^{*} = w_{i0} + \tau \cdot N(0,1), \quad gbest^{*} = gbest + \tau' \cdot N(0,1) \]  

(8.b)

\[ s_{i} = s_{i} + v_{i}^{\text{new}} \]  

(8.c)

where \( w_{i}^{*} \) are the weights which undergo mutation, \( gbest^{*} \) is the \( gbest \) distributed randomly, \( \tau, \tau' \) are learning parameters (either fixed or treated as strategic parameters and therefore subject to mutation), and \( N(0,1) \) is a Gaussian random variable with 0 mean and variance 1. Furthermore, EPSO can also be classified as a self-adaptive algorithm, because it relies on the mutation and selection of strategic parameters, just as any SA—SA evolution strategy [5]. However, EPSO has a drawback of requiring 2 evaluations per agent per iteration.

3) Constriction Factor approach (CFA) [6], [7]. The CFA applies the velocity of the constriction factor approach in the basic PSO as follows, instead of (5) and (6):

\[ v_{i}^{k+1} = K \left[ v_{i}^{k} + c_{1} \cdot rand_{1} \times (pbest_{i} - s_{i}^{k}) \right. \]

\[ \left. + c_{2} \cdot rand_{2} \times (gbest - s_{i}^{k}) \right] \]  

(9.a)

\[ K = \frac{2}{\phi^2 - \sqrt{\phi^2 - 4\phi}}, \quad \text{where} \quad \phi = c_{1} + c_{2}, \quad \phi > 4 \]  

(9.b)

It has been observed here and also in other papers [4] that \( \phi = 4.1 \) leads to good solutions. The CFA ensures the convergence of the search procedure. Therefore, the CFA can generate higher quality solutions than the conventional PSO approach.

However, the constriction factor only considers dynamic behavior of each agent and the effect of the interaction among agents. Namely, the equations were developed with a fixed set of best positions (\( pbest \) and \( gbest \)), although \( pbest \) and \( gbest \) can be changed during search procedure in the basic PSO equation.

IV. MULTIOBJECTIVE OPTIMIZATION

A. Formulation of Multiobjective Optimization Problem

The following objective functions can be described for minimization:

\[ J_{1}(u) = |E_{uid} - E_{ss}| \]  

(10.a)

\[ J_{2}(u) = u_{1} \]  

(10.b)

\[ J_{3}(u) = -u_{2} \]  

(10.c)

\[ J_{4}(u) = -u_{3} \]  

(10.d)

where \( E_{uid} \) is the unit load demand (MW), and \( E_{ss} \) is the corresponding generation (MW) as provided by the steady-
state equation:

\[ E_{ss} = \frac{(0.73u_2 - 0.16)/0.0018u_2}{(0.9u_1 - 0.15u_3)} \]  

(11)

The objective functions are described as following: \( J_1(u) \) is the cost function for power generation error, \( J_2(u) \) is to account for fuel consumption through the fuel valve position, and \( J_3(u) \) is for loss due to pressure drop across the steam valve. Since the pressure drop increases as the valve closes, it is desired to keep it open as wide as possible, thus it is desired to maximize \( u_2 \), or equivalently minimize \(-u_2\). Similarly, \( J_4(u) \) is for pressure drop in the feedwater control valve. Thus, the multiobjective optimization is to be performed to minimize the above objective functions.

**B. PSO for Multiobjective Optimization in the FFPU**

1) **Initialization.** The first step of the PSO for the FFPU is random generation of the agents in the solution space. The agents represent the search points in the solution space, which are expressed by controls \( u_1 \), \( u_2 \), and \( u_3 \). Moreover, the initial velocities are also generated randomly within the same space. In order to speed up the search for an optimal solution, \( c_1 \) and \( c_2 \) are set to 2, \( w_{max} = 0.8 \), and \( w_{min} = 0.3 \) in the PSO. These values are obtained from experimental results. The number of agents is 40 and the iteration is 130 for the basic PSO and its variations. The initial \( pbest \)s are equal to the current search points and \( gbest \) is found by comparing the \( pbest \)s among the agents.

2) **Performance Evaluation.** The evaluation of search point for each agent is performed by using the deviation of each objective function from its possible minimum value, which is then weighted with a preference value. In the multiobjective optimization, the objective functions are often in conflict to each other when performing the optimization. Thus, it is proposed to minimize the maximum deviation of the objective functions instead of directly minimizing the multiobjective functions [14], [15]. The maximum deviation of the multiobjective functions is defined as following:

\[
\delta_m = \max_{i=1,\ldots,k} \delta_{pi} \quad \delta_{pi} \geq 0 
\]

\[
\delta_{pi} = \beta_i (J_i(u) - J_i(u)^*) \quad i = 1,2,\ldots,k \quad u \in \Omega \n
\]

\[
J_i^* = \min_{u \in \Omega} J_i(u) \quad i = 1,2,\ldots,k
\]

(12.a)

(12.b)

(12.c)

where \( \delta_m \) is the maximum deviation of the multiobjective functions, \( \delta_{pi} \) is weighed deviation, \( \beta_i \) is the preference value, \( J_i^* \) is the minimum possible value of the objective function \( J_i \), and \( \Omega \) is the solution space. The preference values give the relative priorities of the objectives in searching for the optimal solution. For each agent, if the new position of the agent has better performance than the current \( pbest \), the \( pbest \) is replaced by the new position. If the best new position among all \( pbests \) is better than the current \( gbest \), the \( gbest \) is replaced by the best new \( pbest \) and the agent number with the best \( pbest \) is stored.

**C. Modifications of the PSO**

1) **Basic PSO method.** The modification of current search point is performed by (5), (6), and (7) in every iteration.

2) **Hybrid PSO (HPSO).** The natural selection mechanism is achieved by forming subgroups from the entire group in the solution space. Then, each agent is evaluated and the best agent is found in each subgroup. Finally, the agents with low performance are replaced by the agent with the best performance in the subgroup. The current points, however, are evaluated with the original \( pbests \) and \( gbest \). Therefore, the HPSO method realizes more intensive search nearby the best agents [8].

3) **Evolutionary PSO (EPSO).** The EPSO is performed with (8). First, each agent is replicated twice. Then, \( w \) and \( gbest \) are mutated with fixed learning parameters \( \tau \) and \( \tau' \), respectively, as shown in (8). Using (8.a) which is called “Movement rule” in EPSO, each agent generates an offspring. Finally, the better agent is selected between the two descendants, which have different \( w \) and \( gbest \).

4) **Constriction factor approach (CFA).** The CFA performs with (9.a) and (9.b) instead of (5) and (6) in the basic PSO. For the best performance, the values of \( c_1 \) and \( c_2 \) are set to 2.1 and 2.0, respectively, in the “Initialization”.

**D. Set-point Scheduler**

The configuration of the reference governor for FFPU is shown in Fig. 4, where the result of the PSO procedure gives a set of optimal solution for the given unit load demand. The set-point scheduler maps the optimal solution \((u_1^*,u_2^*,u_3^*)\) into set-points, demand power \((E_d)\) and pressure \((P_d)\), by direct static equations:

\[
Ed = \frac{(0.73u_2^* - 0.16)/(0.0018u_2^*)}{(0.9u_1^* - 0.15u_3^*)} \quad (13.a)
\]

\[
Pd = 14u_3^* / (1.1u_2^* - 0.19) \quad (13.b)
\]

**V. SIMULATION RESULTS**

In the following simulations, only the results by the basic PSO technique will be shown due to the limitation of space. However, the comparison between the basic PSO and its three variations will be shown at the end of the simulation. Simulations deal with three different cases: Case 1: \( I \)-objective; Case 2: \( 2 \)-objectives; and Case 3: \( 4 \)-objectives. The objective functions are given in (10) and a vector of preference values is given as \( \beta = [1, 0.5, 1, 0] \). This means that \( \beta_1 = 1 \) is for \( J_1(u) \), \( \beta_2 = 0.5 \) is for \( J_2(u) \), \( \beta_3 = 1 \) is for \( J_3(u) \), and \( \beta_4 = 0 \) is for \( J_4(u) \). First, the case “\( I \)-objective”
means minimization of \( J_1(u) \) alone, the case “2-objectives” is for minimization of two objective functions \( J_1(u) \) and \( J_2(u) \), and the case “4-objectives” implies to include all four objective functions.

Fig. 5 shows a dynamic unit load demand and the solution space which is obtained from the operating windows (Fig. 2). The gaps between upper and lower limits are the solution space for the optimization process.

Next step is to perform the PSO for the multiobjective optimization with predefined objective functions and preference values. Fig. 6 show the optimal input trajectories that are optimal solutions found by the PSO in the solution space. In Fig. 6, fuel efficiency is improved as the number of objectives is increased, where an objective function (10.b) was included to minimize \( u_1 \) in reducing the consumption of fuel for Cases 2 & 3. The valve opening \( u_2 \) is increased as the number of objectives is increased, which is also desirable since the pressure valve was required to keep open as wide as possible in Case 3. On the other hand, the results of \( u_3 \) are about the same for all cases. This is because the solution space for the feedwater valve is very small as shown in Fig. 5. All simulation results are improved as the number of objectives is increased. These optimal solutions are the values of \( g_{best} \) that are found though the PSO technique.

Finally, the power and pressure set-points are obtained by set-point scheduler (13) as shown in Fig. 7. The demand power set-point (\( E_d \)) is almost the same for all cases as the unit load demand. The demand pressure set-points (\( P_d \)) mapped for different number of objective functions are shown in Fig. 7. This is because the power-pressure operating window is quite large and the same amount of power can be produced on a wide range of pressure as shown in Fig. 2.

Table I shows comparison of the performance of the four different PSO techniques: the basic PSO, CFA, HPSO, and EPSO. The performance is given numerically by integrating the objective functions over one load cycle (120 min.) for all cases. For Case 1, the first column is highlighted since only the first objective function, \( J_1(u) \), is included in the optimization. The remaining columns are not optimized. Similarly, in Case 2 the first two columns are highlighted since both objective functions, \( J_1(u) \) and \( J_2(u) \), are included in the optimization. However, in Case 3 all columns are highlighted since all four objective functions are considered in the optimization. As we examine the first columns in all three cases, it is clear that HPSO performs the best among all four PSO techniques. The second columns show that EPSO still performs the best for Case 2, and as good as EPSO for Case 3. However, as more objective functions are added in Case 3, EPSO performs slightly better than HPSO in columns 3 and 4. It shows that HPSO performs the best in Cases 1 and 2, and as good as EPSO in Case 3. In view of the computational complexity and the efficiency, HPSO is preferred over EPSO since EPSO needs 2 evaluations of the performance per agent in each iteration. Nevertheless, all simulation results show that PSO techniques can be accommodated well in the multiobjective optimization problem. They also can be adopted for on-line implementation since the pressure set-point needs to be updated only when the unit load demand is changed during the load cycle, and since the computing time is reasonably short for on-line implementation in the FFPU.

VI. CONCLUSION

The Particle Swarm Optimization (PSO) technique is...
presented as an alternative optimization technique for solving a multiobjective optimization problem. The feasibility of using the PSO is demonstrated in the design of optimal set-points for the multiobjective optimal power plant operation. The optimal mapping between the unit load demand and pressure set-point is realized and furthermore, the mapping can also be realized for the dynamic unit load demand. This paper shows that improvements can be made to the basic PSO technique in order to solve the multiobjective optimization problem effectively. It has been demonstrated that the Hybrid PSO (HPSO) technique improves the convergence and problem effectively. It has been demonstrated that the Hybrid PSO (HPSO) technique improves the convergence and performs the best among all PSO techniques in FFPU. Finally, the feasibility of on-line implementation is discussed for the case when the dynamic unit load demand is given in advance.

VII. REFERENCES


VII. REFERENCES


Jin S. Heo received his B.S. and M.S. degrees in Electronics Engineering from Inje University, Korea, in 1999 and 2001, respectively. As a candidate, he is currently pursuing the Ph.D. degree in Electrical Engineering at the Pennsylvania State University. His interests are multiobjective optimization in control system, intelligent distributed control, multiagents system, modeling and control of fuel cell power plants, and real-time embedded system.

Kwang Y. Lee received his B.S. degree in Electrical Engineering from Seoul National University, Korea, in 1964. M.S. degree in Electrical Engineering from North Dakota State University, Fargo, in 1968, and Ph.D. degree in System Science from Michigan State University, East Lansing, in 1971. He has been with Michigan State, Oregon State, Univ. of Houston, and the Pennsylvania State University, where he is a Professor of Electrical Engineering and Director of Power Systems Control Laboratory. His interests include power system control, operation and planning, and intelligent system applications to power systems. Dr. Lee is a Fellow of IEEE, Associate Editor of IEEE Transactions on Neural Networks, and Editor of IEEE Transactions on Energy Conversion. He is also a registered Professional Engineer.

Raul Garduno-Ramirez received his B.S. degree in Electrical Engineering from the National Polytechnic Institute (IPN), Mexico, in 1985, his M.S. degree in Electrical Engineering from Advanced Research Centre of the IPN, Mexico, in 1987, and his Ph.D. degree in Electrical Engineering at the Pennsylvania State University in 2000. He was a Fulbright Fellow during his Ph.D. program at Penn State. During 1986 he stayed at the National Mechanical Laboratory, Tsukuba, Japan. Since 1987 he has been with the Electric Research Institute, Mexico, where he has been involved in the development of control systems for power plants. His current interest is in control software development and intelligent control.