Journal of Heuristics, 9: 507–520, 2003

Solving Unit Commitment Problem Using Hybrid Particle Swarm Optimization

TIEW-ON TING, M.V.C. RAO, C.K. LOO AND S.S. NGU
Jalan Ayer keroh Lama, Multimedia University, 75450 Melaka, Malaysia
email: toting@mmu.edu.my

Submitted in August 2003 and Accepted by Anthony Cox in October 2003

Abstract

This paper presents a Hybrid Particle Swarm Optimization (HPSO) to solve the Unit Commitment (UC) problem. Problem formulation of the unit commitment takes into consideration the minimum up and down time constraints, start up cost and spinning reserve, which is defined as the minimization of the total objective function while satisfying all the associated constraints. Problem formulation, representation and the simulation results for a 10 generator-scheduling problem are presented. Results shown are acceptable at this early stage.

Key Words: Unit commitment, Hybrid Particle Swarm Optimization

I. Introduction

Unit commitment (UC) in a power system involves determining a start-up and shutdown schedule of units to meet the forecasted demand, over a short-term period (Wood and Wollenberg, 1996). In solving the unit commitment problem, generally two basic decisions are involved, namely the "unit commitment" decision and the "economic dispatch" decision. The "unit commitment" decision involves the determination of the generating units to be running during each hour of the planning horizon, considering system capacity requirements, including the reserve, and the constraints on the start up and shut down of units. The "economic dispatch" decision involves the allocation of the system demand and spinning reserve capacity among the operating units during each specific hour of operation.

UC problem has commonly been formulated as a nonlinear, large scale, mixed-integer combinatorial optimization problem with constraints. The exact solution to the problem can be obtained only by complete enumeration, often at the cost of a prohibitively computation time requirement for realistic power systems (Wood and Wollenberg, 1996). Research endeavours, therefore, have been focused on, efficient, near-optimal UC algorithms which can be applied to large-scale power systems and have reasonable storage and computation time requirements. A survey of literature on UC methods reveals that various numerical optimization techniques have been employed to approach the UC problems. Specifically, there are priority list methods (Burns and Gibson, 1975; Sheble, 1990), integer programming

(Dillon and Edwin, 1978; Garver, 1963), dynamic programming (Snyder, Powell and Rayburn, 1987; Lowery, 1983; Pang and Chen, 1976; Pang, Sheble and Albuyeh, 1981; Su and Hsu, 1991; Ouyang and Shahidehpour, 1991), branch-and-bound methods (Cohen and Yoshimura, 1983), mixed-integer programming (Muckstadt and Wilson, 1968), and Lagrangian relaxation methods (Merlin and Sandrin, 1983; Zhuang and Galiana, 1988). Among these methods, the priority list method is simple and fast but the quality of final solution is rough. Dynamic programming methods, which are based on priority lists are flexible but computationally expensive. Branch-and-bound adopts a linear function to represent the fuel consumption and time-dependent start cost, and obtains the required lower and upper bounds. The shortcoming of branch-and-bound is the exponential growth in the execution time with the size of the UC problem. The integer and mixed-integer methods adopt a linear programming technique to solve and check for an integer solution. These methods have only been applied to a small UC problem and have required major assumptions which limit the solution space. The Lagrangian relaxation method provides a fast solution but it may suffer from numerical convergence and solution quality problems.

Aside from the above methods, there is another class of numerical techniques applied to the UC problem. Specifically, there are artificial Neural Network (Ouyang and Shahidehpour, 1992; Sasaki et al., 1992), Simulated Annealing (SA) (Zhuang and Galiana, 1990), and Genetic Algorithms (GA's) (Dasgupta and McGregor, 1994; Huang et al., 1993; Sheble and Maifeld, 1994; Kazarlis, Bakirtzis, and Petridis, 1996). These methods can accommodate more complicated constraints and are claimed to have better solution quality. SA is a powerful, general-purpose stochastic optimization technique, which can theoretically converge asymptotically to a global optimum solution with probability 1. One main drawback, however, of SA is that it takes much CPU time to find the near-global minimum. GAs are a general-purpose stochastic and parallel search method based on the mechanics of natural selection and natural genetics. GAs are a search method having the potential to obtain near-global minimum.

In this paper, we apply the Hybrid Particle Swarm Optimization (HPSO) method in solving the UC problem. A description of HPSO method is presented in Section II. Then a detailed application of HPSO is given in Section III. The analysis of the HPSO method is given in Section IV. Numerical tests on two cases using HPSO, LR and GA's are compared in Section V. Finally, a conclusion is given in Section VI.

II. Particle Swarm Optimization

Particle Swarm Optimization (PSO) was introduced by Kennedy and Eberhart (1995) as an alternative to Genetic Algorithms. The PSO technique has ever since turned out to be a competitor in the field of numerical optimization. Similar to GA, a PSO consists of a population refining their knowledge of the given search space. PSO is inspired by particles moving around in the search space. The individuals in a PSO thus have their own positions and velocities. These individuals are denoted as particles. The PSO traditionally has no crossover between individuals, has no mutation and particles are never substituted by other individuals during the run. Instead the PSO refines its search by attracting the particles to positions with good solutions. Each particle remembers its own best position so far found in the exploration. This position is called personal best and is denoted by *pbest* in Eq. (1). Additionally, among these personal bests, there is only one which has the best fitness. The best among *pbest* is called the global best and is denoted by *gbest* in Eq. (1),

$$V_i = wV_i + \rho_1 * rand(\ ) * (gbest - X_i) + \rho_2 * rand(\ ) * (pbest - X_i) \tag{1}$$

where w is known as the inertia weight described in Shi and Eberhart (1998, 1999). The best found position for the given particle denoted by *pbest* and *gbest* is the best position known for all particles. The parameters ρ_1 and ρ_2 are set to a constant value 2, whereas rand ( ) is randomly generated value between 0 and 1. The position of each particle is updated every iteration. This is done by adding the velocity vector to the position vector, as described in Eq. (2) below:

$$X_i = X_i + V_i \tag{2}$$

It has been noticed that members of the group seem to share information between them, a fact that leads to increased cohesion or efficiency (e.g., in search of food) of the group. Some scientists suggest that knowledge is optimized by social interaction and thinking is not private but also interpersonal. Therefore, particle swarms have not only individual, but also a collective intelligence, simply by their social interactions.

The simplest version of PSO lets every individual move from a given point to a new one which is a weighted combination of the individual's best position ever found ("nostalgia"), and of the individual's best position, *pbest*. The choice of the PSO algorithm's parameters (such as the group's inertia) seems to be of utmost importance for the speed and efficiency of the algorithm.

III. HPSO approach to unit commitment

The original version of Particle Swarm Optimization (Kennedy and Eberhart, 1995) operates on real values. However, with a simple modification the particle swarm algorithm can be made to operate on binary problems, such as those traditionally optimized b genetic algorithms. In binary particle swarm, X_i and *pbest* can take on values of 0 or 1 only. The velocity, V_i will determine a probability threshold. If V_i is higher, the individual is more likely to choose 1, while lower values favour the 0 choice. Such a threshold needs to stay in the range [0.0, 1.0]. One straightforward function for accomplishing this is common in neural networks. The function is called sigmoid function, derived as follows:

$$s(V_i) = \frac{1}{1 + \exp(-V_i)}$$

The function squashes its input into the requisite range and has properties that make it agreeable to be used as a probability threshold. A random number (drawn from a uniform distribution between 0.0 and 1.0) is then generated, whereby X_i is set to 1 if the random number is less than the value from the sigmoid function as illustrated below.

If $rand(\ ) < s(Vi)$ then $U_i = 1$ else $U_i = 0$

In unit commitment problem, U_i represents the on or off state of the generator i. In order to ensure that there is always some chance of a bit flipping (on and off of generators), a constant $V_{\max}$ can be set at the start of a trial to limit the range of V_i. In practice, $V_{\max}$ is often set at ± 4.0, so that there is always at least a chance that a bit will change state. This is to limit V_i so that $s(V_i)$ does not approach 0.0 or 1.0 too closely. In this binary model, $V_{\max}$ functions similarly to mutation rate in genetic algorithms.

In solving the unit commitment problem, the real valued PSO and binary PSO are run in parallel, with each updated according to (3) and (4) separately. The real valued PSO will optimize the generated power, P_i in the vicinity of the on and off status, U_i, which is changed and optimized by binary PSO.

IV. Unit commitment problem formulation

In this section, we first formulate the UC problem, and then present a detailed HPSO algorithm for solving the UC problem.

The objective of the UC problem is the minimization of the total production costs over the scheduling horizon. Therefore, the objective function is expressed as the sum of fuel and start-up costs of the generating units. For N generators, the operation cost is defined mathematically as follows:

$$TPC_N = \sum_{i=1}^{N} \left[F_i(P_{ih}) + ST_i\left(1 - U_{i(h-1)}\right) \right] U_{ih} \tag{3}$$

The operating cost accumulates over the total number of operating hours, H, where $H = 24$ which represent 24 hours of operation for each unit of generator. Therefore Eq. (3) is rewritten as:

$$TPC_{HN} = \sum_{h=1}^{H} \sum_{i=1}^{N} \left[F_i(P_{ih}) + ST_i\left(1 - U_{i(h-1)}\right) \right] U_{ih} \tag{4}$$

In Eq. (3), TPC_N is the total production cost for N units of generator whereas TPC_{HN} in Eq. (4) denotes the total production cost for N units of generator over H number of operating hours. Due to the operational requirements, the minimization of the objective function is subjected to the following constraints:

(a) power balance constraint

$$\sum_{i=1}^{N} P_{ih} U_{ih} \geq D_h \tag{5}$$

(b) spinning reserve constraint

$$\sum_{i=1}^{N} P_{i(\max)} U_{ih} \geq D_h + R_h \tag{6}$$

(c) generation limit constraint

$$P_{i(\min)} \leq P_{ih} \leq P_{i(\max)} \tag{7}$$

(d) minimum up-time constraint

$$U_{ih} = 1 \quad \text{for} \quad \sum_{t=h-up_i}^{h-1} U_{it} < MU_i \tag{8}$$

(e) minimum down-time constraint

$$U_{ih} = 0 \quad \text{for} \quad \sum_{t=h-down_i}^{h-1} (1 - U_{it}) < MD_i \tag{9}$$

where the notations used are

TPC	Total production cost of the power generation,
$F_i(P_{ih})$	Fuel cost function of the ith unit with generation output, P_{ih}, at the hth hour. Usually, it is a quadratic polynomial with coefficients a_i, b_i and c_i as follows: $F_i(P_{ih}) = \alpha_i P_{ih}^2 + \beta_i P_{ih} + \gamma_i$
N	Number of generators,
H	Number of hours,
P_{ih}	The generation output of the ith unit at the hth hour,
ST_i	Start-up cost of the ith unit,
U_{ih}	The on/off status of the ith unit at the hth hour, and $U_{ih} = 0$ when off, $U_{ih} = 1$ when on,
D_h	Load demand at the hth hour,
Rh	Spinning reserve at the hth hour,
$P_{i(\min)}$	Minimum generation limit of ith unit,
$P_{i(\max)}$	Maximum generation limit of ith unit,
MU_i	Minimum up-time of ith unit,
MD_i	Minimum down-time of ith unit.

V. Constraints satisfaction

Recently, several methods for handling infeasible solutions for continuous numerical optimization problems have emerged (Michalewicz and Attia, 1994; Homaifar, Lai, and Qi, 1994; Powell and Skolnick, 1991; Schoenauer and Xanthakis, 1993). Some of them are based on penalty functions. They differ, however, in how the penalty function is designed and applied to infeasible solutions. They commonly use the cost function f to evaluate a feasible solution, i.e.,

$$\Phi_f(x) = f(x) \tag{10}$$

and the constraint violation measure $\Phi_u(x)$ for the $r + m$ constraints, usually defined as

$$\Phi_u(x) = \sum_{i=1}^{r} g_i^+(x) + \sum_{j=1}^{m} |h_j(x)| \tag{11}$$

or

$$\Phi_u(x) = \frac{1}{2}\left[\sum_{i=1}^{r} (g_i^+(x))^2 + \sum_{j=1}^{m} (h_j(x))^2\right] \tag{12}$$

where $g_i^+(x) = \max\{0, g_i(x)\}$. In other words, $g_i^+(x)$ is the magnitude of the violation of the ith constraints, where $1 \leq i \leq r$. Then the total evaluation of an individual x, which can be interpreted as the error (for a minimization problem) of an individual x is obtained as

$$\Phi(x) = \Phi_f(x) + s\Phi_u(x) \tag{13}$$

where s is a penalty parameter of a positive or negative constant for the minimization or maximization problem, respectively. By associating a penalty with all constraint violations, a constrained problem is transformed to an unconstrained problem such that we can deal with candidates that violate the constraints to generate potential solutions without considering the constraints.

A. Satisfying power demand and reserve constraints

With respect to the theoretical explanation above, we formulate the objective of the unit commitment problem as a combination of total production cost because the main objective with power balance and spinning reserve as inequality constraints, whereby $\Phi_f(x) = TPC_N$ (Eq. (3)) and $\Phi_u(x)$ is equivalent to the blend of power balance and spinning reserve constraints. Subsequently, the formulation of the objective function is

$$\Phi(x) = TPC_N + \frac{s}{2}\left[C_1\left(D_h - \sum_{i=1}^{N} P_{ih}U_{ih}\right)^2 + C_2\left(D_h + R_h - \sum_{i=1}^{N} P_{i(\max)}U_{ih}\right)^2\right] \tag{14}$$

As discussed, s is the penalty factor which is computed at the tth generation defined as $s = s_0 + \log(t + 1)$. The choice of s determines the accuracy and speed of convergence. From the experiment, greater value of s increases its speed and convergence rate. Due to this reason, we choose a value of 100 for its s_0. There are several methods for choosing s and each method establishes a family of intervals for every constraint that determines the appropriate values for s. The pressure on the infeasible solution can be increased with the number of generations as discussed in Kuhn-Tucker optimality theorem and penalty function theorem provide guidelines to choose the penalty term. In (14), C1 is set to 1 if violation to constraint (5) occurs, and C1 = 0 whenever (5) is not violated. Likewise, C2 is also set to 1 whenever violation of Eq. (6) is detected, and it remains 0 otherwise. The second term in the penalty factor is the reserve constraint, where R_h is 10% of the power demand D_h. Thus, Eq. (14) can also be written as:

$$\Phi(x) = TPC_N + \frac{s}{2}\left[C_1\left(D_h - \sum_{i=1}^{N} P_{ih}U_{ih}\right)^2 + C_2\left(1.1D_h - \sum_{i=1}^{N} P_{i(\max)}U_{ih}\right)^2\right] \tag{15}$$

By substituting Eq. (3) into (15),

$$\Phi(x) = \sum_{i=1}^{N}\left[F_i(P_{ih}) + ST_i\left(1 - X_{i(h-1)}\right)\right]U_{ih} + \frac{s}{2}\left[C_1\left(D_h - \sum_{i=1}^{N} P_{ih}U_{ih}\right)^2 + C_2\left(1.1D_h - \sum_{i=1}^{N} P_{i(\max)}U_{ih}\right)^2\right] \tag{16}$$

Equation (16) above is the fitness function for evaluating every particle in the population of PSO for an hour. For N hours, Eq. (16) is redefined as in (17)

$$\Phi(x) = \sum_{k=1}^{H}\left\{\sum_{i=1}^{N}\left[F_i(P_{ih}) + ST_i\left(1 - X_{i(h-1)}\right)\right]U_{ih} + \frac{s}{2}\left[C_1\left(D_h - \sum_{i=1}^{N} P_{ih}U_{ih}\right)^2 + C_2\left(1.1D_h - \sum_{i=1}^{N} P_{i(\max)}U_{ih}\right)^2\right]\right\} \tag{17}$$

To decrease the pressure of constraint violation error on the fitness function, $\Phi(x)$, a set of major feasible solutions that satisfy the power demand is generated before evaluation using (17) is considered. The pseudocode is given in figure 1.

In the pseudocode above, P_g is the total power generated where $P_g = P_i + P_{i+2} + P_{i+3} \ldots + P_N$. P_i is the power generated by generator i. The maximum limit of P_i is specified by $P_{i(\max)}$. N is the total number of operating generators. D_h is the power demand to be satisfied and rand( ) is the random number generator between 0 to 1.

```
Do while ((Pg < Dh) and (Count < 100))
        Count = Count +1
        i = (Count modulus N) + 1

                If generator i is off then on it
                        Update total generated power, Pg = Pg + Pi
                Else if generator i if on then
                        (1) Minus the relevant power  of unit i, Pg = Pg – Pi
                        (2) Reinitialize, Pi = Pi + rand( ) * (Pi(max) – Pi)
                        (3) Update total generated power, Pg, Pg = Pg + Pi
                End if

                /* Note that (2) accelerates Dh to be satisfied*/
Loop
```

Figure 1. Generating feasible candidates within 100 iterations.

```
If Pi > Pi(max) then
        reinitialize, Pi = Pi(min) + rand( ) * (Pi(max) – Pi(min))
End if

If Pi < Pi(min) then
        reinitialize, Pi = Pi(min) + rand( ) * (Pi(max) – Pi(min))
End if
```

Figure 2. Reinitialize when candidate solutions exceed boundary.

B. Satisfying the generation limit constraints

As particles explore the searching space which is bounded by power limit as derived in (7), they do encounter cases whereby the power generated exceeds the boundary and therefore violates the constraint in (7). To avoid this, we reinitialize the value whenever it is greater than the maximum or is smaller than the minimum. Thus is the pseudocode (figure 2).

C. Satisfying the minimum up and down time constraints

The technique used to satisfy the minimum up and down time in this experiment is extremely simple. As the solution is based upon the best particle (*gbest*) in the history of the entire population, constraints are taken care of by forcing the binary value in *gbest* to change its state whenever either the MU_i or MD_i constraint is violated. However, this may change the fitness value evaluated using Eq. (17). It implies that the current *gbest* might no longer be the best among all the other particles. To correct this error; the *gbest* will be revaluated using the same formula (17).

Table 1. Generator system operator data.

	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6	Unit 7	Unit 8	Unit 9	Unit 10
Pmax (MW)	455	455	130	130	162	80	85	55	55	55
Pmin (MW)	150	150	20	20	25	20	25	10	10	10
α (\$/h)	1000	970	700	680	450	370	480	660	665	670
β (\$/MWh)	16.19	17.26	16.60	16.50	19.70	22.26	27.74	25.92	27.27	27.79
γ (\$/MWh^2)	0.00048	0.00031	0.002	0.00211	0.00398	0.00712	0.0079	0.00413	0.00222	0.00173
Min Up (h)	8	8	5	5	6	3	3	1	1	1
Min Down (h)	8	8	5	5	6	3	3	1	1	1
Hot start cost ($)	4500	5000	550	560	900	170	260	30	30	30
Cold start cost ($)	9000	10000	1100	1120	1800	340	520	60	60	60
Cold start hrs (h)	5	5	4	4	4	2	2	0	0	0
Initial status (h)	8	8	−5	−5	−6	−3	−3	−1	−1	−1

VI. Numerical simulations

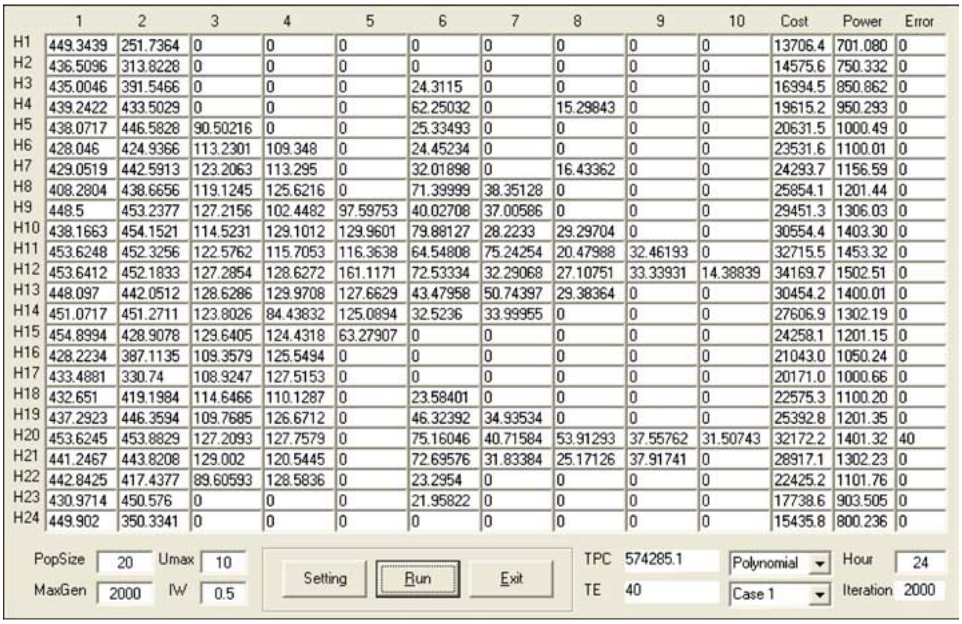
The PSO program was implemented in Visual Basic language and the simulation was carried out on a 10-generator system, the data is given in Tables 1 and 2. The setting for HPSO is as follows:

Population size = 20
Maximum iteration = 2000
Dimension = Number of generator, N ($U_{\max}$) = 10
Maximum velocity, $V_{\max} = P_{i(\max)} - P_{i(\min)}$
Inertia weight, $w = 0.5$

In this experiment four test cases are considered,

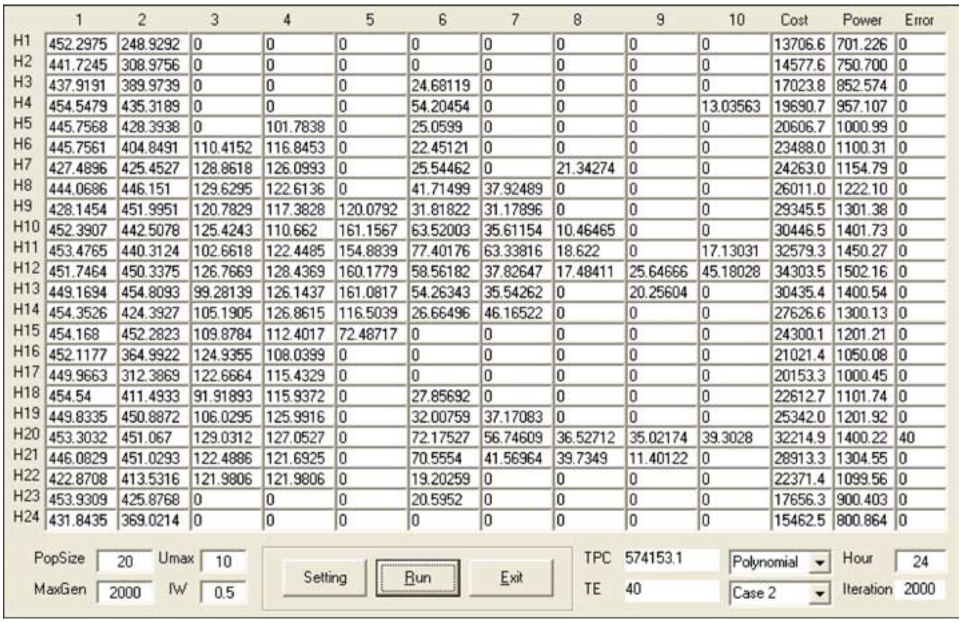
Case 1: Using standard PSO.
Case 2: Using PSO + differential mutation. The differential mutation is the best method out of the ten operators used in Ting, Rao, and Loo (2002). Results show a tremendous improvement when applying to 10 benchmark problems in terms of speed and accuracy.
Case 3: Using PSO with linear decreasing inertia weight, w. The method used here is adopted from Shi and Eberhart (1998, 1999), where it is claimed to obtain a better result as compared to using static w.
Case 4: Same as in *Case* 1, but instead of reinitializing the values of particles when violating constraint (4), the values are set to its relevant $P_{i(\max)}$ when exceeding $P_{i(\max)}$ and is set to $P_{i(\min)}$ when values are below $P_{i(\min)}$.

The simulations of the relevant test case are shown in figures 3–6. The results show a slight violation of reserve constraint at the 20th hour for all test cases. The total error is shown after the label *TE*, where it displays a value of 40. This violation to the reserve constraint is considered acceptable. However, in all four test cases, other constraints considered in (5), (7–9)



	1	2	3	4	5	6	7	8	9	10	Cost	Power	Error
H1	449.3439	251.7364	0	0	0	0	0	0	0	0	13706.4	701.080	0
H2	436.5096	313.8228	0	0	0	0	0	0	0	0	14575.6	750.332	0
H3	435.0046	391.5466	0	0	0	24.3115	0	0	0	0	16994.5	850.862	0
H4	439.2422	433.5029	0	0	0	62.25032	0	15.29843	0	0	19615.2	950.293	0
H5	438.0717	446.5828	90.50216	0	0	25.33493	0	0	0	0	20631.5	1000.49	0
H6	428.046	424.9366	113.2301	109.348	0	24.45234	0	0	0	0	23531.6	1100.01	0
H7	429.0519	442.5913	123.2063	113.295	0	32.01898	0	16.43362	0	0	24293.7	1156.59	0
H8	408.2804	438.6656	119.1245	125.6216	0	71.39999	38.35128	0	0	0	25854.1	1201.44	0
H9	448.5	453.2377	127.2156	102.4482	97.59753	40.02708	37.00586	0	0	0	29451.3	1306.03	0
H10	438.1663	454.1521	114.5231	129.1012	129.9601	79.88127	28.2233	29.29704	0	0	30554.4	1403.30	0
H11	453.6248	452.3256	122.5762	115.7053	116.3638	64.54808	75.24254	20.47988	32.46193	0	32715.5	1453.32	0
H12	453.6412	452.1833	127.2854	128.6272	161.1171	72.53334	32.29068	27.10751	33.33931	14.38839	34169.7	1502.51	0
H13	448.097	442.0512	128.6286	129.9708	127.6629	43.47958	50.74397	29.38364	0	0	30454.2	1400.01	0
H14	451.0717	451.2711	123.8026	84.43832	125.0894	32.5236	33.99955	0	0	0	27606.9	1302.19	0
H15	454.8994	428.9078	129.6405	124.4318	63.27907	0	0	0	0	0	24258.1	1201.15	0
H16	428.2234	387.1135	109.3579	125.5494	0	0	0	0	0	0	21043.0	1050.24	0
H17	433.4881	330.74	108.9247	127.5153	0	0	0	0	0	0	20171.0	1000.66	0
H18	432.651	419.1984	114.6466	110.1287	0	23.58401	0	0	0	0	22575.3	1100.20	0
H19	437.2923	446.3594	109.7685	126.6712	0	46.32392	34.93534	0	0	0	25392.8	1201.35	0
H20	453.6245	453.8829	127.2093	127.7579	0	75.16046	40.71584	53.91293	37.55762	31.50743	32172.2	1401.32	40
H21	441.2467	443.8208	129.002	120.5445	0	72.69576	31.83384	25.17126	37.91741	0	28917.1	1302.23	0
H22	442.8425	417.4377	89.60593	128.5836	0	23.2954	0	0	0	0	22425.2	1101.76	0
H23	430.9714	450.576	0	0	0	21.95822	0	0	0	0	17738.6	903.505	0
H24	449.902	350.3341	0	0	0	0	0	0	0	0	15435.8	800.236	0

Figure 3. Solution for test case 1.



	1	2	3	4	5	6	7	8	9	10	Cost	Power	Error
H1	452.2975	248.9292	0	0	0	0	0	0	0	0	13706.6	701.226	0
H2	441.7245	308.9756	0	0	0	0	0	0	0	0	14577.6	750.700	0
H3	437.9191	389.9739	0	0	0	24.68119	0	0	0	0	17023.8	852.574	0
H4	454.5479	435.3189	0	0	0	54.20454	0	0	0	13.03563	19690.7	957.107	0
H5	445.7568	428.3938	0	101.7838	0	25.0599	0	0	0	0	20606.7	1000.99	0
H6	445.7561	404.8491	110.4152	116.8453	0	22.45121	0	0	0	0	23488.0	1100.31	0
H7	427.4896	425.4527	128.8618	126.0993	0	25.54462	0	21.34274	0	0	24263.0	1154.79	0
H8	444.0686	446.151	129.6295	122.6136	0	41.71499	37.92489	0	0	0	26011.0	1222.10	0
H9	428.1454	451.9951	120.7829	117.3828	120.0792	31.81822	31.17896	0	0	0	29345.5	1301.38	0
H10	452.3907	442.5078	125.4243	110.662	161.1567	63.52003	35.61154	10.46465	0	0	30446.5	1401.73	0
H11	453.4765	440.3124	102.6618	122.4485	154.8839	77.40176	63.33816	18.622	0	17.13031	32579.3	1450.27	0
H12	451.7464	450.3375	126.7669	128.4369	160.1779	58.56182	37.82647	17.48411	25.64666	45.18028	34303.5	1502.16	0
H13	449.1694	454.8093	99.28139	126.1437	161.0817	54.26343	35.54262	0	20.25604	0	30435.4	1400.54	0
H14	454.3526	424.3927	105.1905	126.8615	116.5039	26.66496	46.16522	0	0	0	27626.6	1300.13	0
H15	454.168	452.2823	109.8784	112.4017	72.48717	0	0	0	0	0	24300.1	1201.21	0
H16	452.1177	364.9922	124.9355	108.0399	0	0	0	0	0	0	21021.4	1050.08	0
H17	449.9663	312.3869	122.6664	115.4329	0	0	0	0	0	0	20153.3	1000.45	0
H18	454.54	411.4933	91.91893	115.9372	0	27.85692	0	0	0	0	22612.7	1101.74	0
H19	449.8335	450.8872	106.0295	125.9916	0	32.00759	37.17083	0	0	0	25342.0	1201.92	0
H20	453.3032	451.067	129.0312	127.0527	0	72.17527	56.74609	36.52712	35.02174	39.3028	32214.9	1400.22	40
H21	446.0829	451.0293	122.4886	121.6925	0	70.5554	41.56964	39.7349	11.40122	0	28913.3	1304.55	0
H22	422.8708	413.5316	121.9806	121.9806	0	19.20259	0	0	0	0	22371.4	1099.56	0
H23	453.9309	425.8768	0	0	0	20.5952	0	0	0	0	17656.3	900.403	0
H24	431.8435	369.0214	0	0	0	0	0	0	0	0	15462.5	800.864	0

Figure 4. Solution for test case 2.

	1	2	3	4	5	6	7	8	9	10	Cost	Power	Error
H1	454.7033	246.0677	0	0	0	0	0	0	0	0	13696.7	700.771	0
H2	449.5299	300.6722	0	0	0	0	0	0	0	0	14562.5	750.202	0
H3	447.2657	376.0712	0	0	0	27.27711	0	0	0	0	16994.5	850.614	0
H4	418.1398	454.0831	0	0	0	39.92398	0	38.0997	0	0	19708.6	950.246	0
H5	444.96	404.2514	121.0091	0	0	31.46665	0	0	0	0	20662.5	1001.68	0
H6	446.4999	400.4336	100.7168	122.1584	0	31.03901	0	0	0	0	23562.7	1100.84	0
H7	448.2352	443.5595	89.33803	128.1454	0	32.29975	0	0	13.00478	0	24244.6	1154.58	0
H8	454.9307	427.0196	127.9679	108.2592	0	65.0676	27.32445	0	0	0	25822.0	1210.56	0
H9	440.9735	447.2252	117.1076	123.6284	0	49.5894	62.37415	35.23718	0	25.04621	29205.3	1301.18	0
H10	443.7815	451.4257	129.3096	123.637	143.3348	43.01076	32.62489	35.25083	0	0	32204.7	1402.37	0
H11	451.3739	448.9481	120.5873	126.3435	156.573	21.93787	70.75922	39.94422	16.6754	0	32574.5	1453.14	0
H12	454.6332	453.5735	121.5305	128.5936	161.1201	29.59755	49.8786	52.95373	37.30904	12.7298	34365.2	1501.92	0
H13	451.9563	447.8981	100.0724	129.5013	126.5134	67.96997	30.54271	46.03267	0	0	30529.9	1400.48	0
H14	432.8499	448.4104	115.7843	121.4228	84.98415	73.6957	25.61602	0	0	0	27630.9	1302.76	0
H15	442.3194	453.9587	127.0775	113.6366	65.87752	0	0	0	0	0	24313.3	1202.87	0
H16	442.5332	356.6079	129.3045	121.7121	0	0	0	0	0	0	21022.4	1050.15	0
H17	426.5648	318.935	125.8772	128.6638	0	0	0	0	0	0	20158.9	1000.04	0
H18	439.4057	398.9806	115.0723	122.3281	0	25.21452	0	0	0	0	22584.8	1101.00	0
H19	444.708	451.3772	128.6097	108.9719	0	39.91804	28.7763	0	0	0	25304.4	1202.36	0
H20	451.4222	454.4952	128.8916	129.4877	0	71.4073	65.27321	48.32902	18.4883	33.60454	32207.9	1401.39	40
H21	447.2304	447.5722	104.9924	129.6861	0	66.83609	40.17138	11.17801	53.9651	0	29004.1	1301.63	0
H22	429.9172	423.7518	122.1212	103.0433	0	22.58399	0	0	0	0	22424.7	1101.41	0
H23	452.8515	409.7876	0	0	0	38.29157	0	0	0	0	17757.9	900.930	0
H24	442.0292	358.121	0	0	0	0	0	0	0	0	15441.1	800.150	0

PopSize 20 Umax 10 | Setting | Run | Exit | TPC 575985.2 | Polynomial | Hour 24
MaxGen 2000 IW 0.5 | TE 40 | Case 3 | Iteration 2000

Figure 5. Solution for case 3.

	1	2	3	4	5	6	7	8	9	10	Cost	Power	Error
H1	455	455	0	0	0	0	0	0	0	0	17353.3	910	0
H2	455	455	0	0	0	0	0	0	0	0	17353.3	910	0
H3	455	455	0	0	0	0	0	55	0	0	19511.3	965	0
H4	455	455	0	0	0	80	0	55	0	0	21987.7	1045	0
H5	455	455	0	130	0	80	0	0	0	0	22970.3	1120	0
H6	455	455	130	130	0	80	0	0	0	0	26402.1	1250	0
H7	455	455	130	130	0	80	0	55	0	0	27460.2	1305	0
H8	455	455	130	130	162	0	0	0	0	0	28651.6	1332	0
H9	455	455	130	130	162	0	0	55	55	0	31241.2	1442	0
H10	455	455	130	130	162	0	85	55	55	55	36799.9	1582	0
H11	455	455	130	130	162	80	85	55	55	0	36552.6	1607	0
H12	455	455	130	130	162	80	85	55	55	55	38476.3	1662	0
H13	455	455	130	130	162	80	85	55	0	0	34041.0	1552	0
H14	455	455	130	130	162	80	0	55	0	0	31146.0	1467	0
H15	455	455	130	130	162	0	0	0	0	0	26851.6	1332	0
H16	455	455	130	130	0	0	0	0	0	0	23105.7	1170	0
H17	455	455	130	130	0	0	0	0	0	0	23105.7	1170	0
H18	455	455	130	130	0	0	0	0	0	55	25369.4	1225	0
H19	455	455	130	130	0	80	85	0	0	0	28627.1	1335	0
H20	455	455	130	130	0	80	85	55	55	55	34850.4	1500	40
H21	455	455	130	130	0	80	85	0	55	55	32572.3	1445	0
H22	455	455	130	130	0	0	0	0	55	0	25277.3	1225	0
H23	455	455	130	0	0	0	0	0	0	0	20245.1	1040	0
H24	455	455	0	0	0	0	0	0	0	0	17353.3	910	0

PopSize 20 Umax 10 | Setting | Run | Exit | TPC 647305.6 | Polynomial | Hour 24
MaxGen 2000 IW 0.5 | TE 40 | Case 4 | Iteration 2000

Figure 6. Solution for case 4.

Table 2. Load demands for 24 hours.

Hour	D_h	Hour	D_h
1	700	13	1400
2	750	14	1300
3	850	15	1200
4	950	16	1050
5	1000	17	1000
6	1100	18	1100
7	1150	19	1200
8	1200	20	1400
9	1300	21	1300
10	1400	22	1100
11	1450	23	900
12	1500	24	800

are all satisfied. By observing the generated power as in figures 3–5, we can conclude that the generated power for each hour is very close to the power demand, D_h as depicted in Table 2. One major advantage of HPSO is its simplicity of the algorithm as compared to any other techniques in existence so far. HPSO is much faster than other techniques such as genetic algorithm approach. The best result is simulated from test case 2, where total production cost obtained is \$574153.1. This is a close solution to the known global optimum of \$565825 as reported in Cheng, Liu, and Liu (2000) and Kazarlis, Bakirtzis, and Petridis (1996). The worst result, as displayed in figure 6 shows a very expensive solution. This result shows that setting generated power to its minimum or maximum limit is not a good strategy to satisfy constraint (7). By re-initialization, PSO has more chances to created better and better candidate solutions to the economic dispatch problem, optimized by real valued PSO in HPSO.

VII. Conclusion

Application of HPSO is a new approach in solving the Unit Commitment problem. Results demonstrated that HPSO is a competent method to solve the UC problem. The total objective is the sum of objectives and constraints, which are fuel cost, start up cost, spinning reserve and power demand. For better solution, powers generated by N unit of generators are constantly checked so that feasible particles that meet the power demand are always generated. This reduces the pressure of the constraint violation of the total objective function. The minimum up and down time are treated separately by forcing the generator to turn on or off in order to fulfil this constraint. Four test cases are considered for simulation and results obtained are acceptable at this prior stage.

References

Burns, R.M. and C.A. Gibson. (1975). "Optimization of Priority Lists for A Unit Commitment Program." IEEE/PES 1975 Summer Meeting, Paper A 75 453-1.

Cheng, C.-P., C.-W. Liu, and C.-C. Liu. (2000). "Unit Commitment by Lagrangian Relaxation and Genetic Algorithms." *IEEE Trans. on Power Systems* 1(2).

Cohen, A.I. and M. Yoshimura. (1983). "A Branch-and-Bound Algorithm for Unit Commitment." *IEEE Trans. on Power Systems* PAS-102(2), 444–451.

Dasgupta, D. and D.R. McGregor. (1994). "Thermal Unit Commitment Using Genetic Algorithms," *IEE Proc. C, Gener. Transm. Distrib.* 141(5), 459–465.

Dillon, T.S. and K.W. Edwin. (1978). "Integer Programming Approach to The Problem of Optimal Unit Commitment with Probabilistic Reserve Determination." *IEEE Trans. on Power Systems* PAS-97(6), 2154–2166,

Garver, L.L. (1963). "Power Generation Scheduling by Integer Programming Development of Theory." *IEEE Trans. on Power Systems* 102, 730–735.

Homaifar, A.S., H.Y. Lai, and X. Qi. (1994). "Constrained Optimization via Genetic Algorithms." *Simulation* 62, 242–254.

Huang, C.L., J.S. Tzeng, P.C. Yang, and H.T. Yang. (1993). "Implementation of Genetic Algorithm for Unit Commitment." In *1993 Proceedings of The 14th Symposium on Electrical Power Engineering*, Taiwan, R.O.C., pp. 439–446.

Kazarlis, S.A., A.G. Bakirtzis, and V. Petridis. (1996). "A Genetic Algorithm Solution to the Unit Commitment Problem." *IEEE Trans. on Power Systems* 11(1), 83–92.

Kennedy, J. and R. Eberhart. (1995). "Particle Swarm Optimization." In *Proc. IEEE Int. Conf. Neural Networks* [Online], pp. 1942–1948. Available: http://www.engr.iupui.edu/∼shi/Conference/psopap4.html

Lowery, P.G. (1983). "Generation Unit Commitment by Dynamic Programming." *IEEE Trans. on Power Systems* 102, 1218–1225.

Merlin, A. and P. Sandrin. (1983). "A New Method for Unit Commitment at Electricite De France." *IEEE Trans on Power Systems* 102, 1218–1255.

Michalewicz, Z. and N. Attia. (1994). "Evolutionary Optimization of Constrained Problems." in A.V. Sebald and L.J. Fogel (eds.), *Proc. 3rd Annu. Conf. Evolutionary Programming*. River Edge, NJ: World Scientific, pp. 98–108.

Muckstadt, J.A. and R.C. Wilson. (1968). "An Application of Mixed-Integer Programming Duality to Scheduling Thermal Generating Systems." *IEEE Trans. on Power Systems* 1968–1978.

Ouyang, Z. and S.M. Shahidehpour. (1991). "An Intelligent Dynamic Programming for Unit Commitment Application." *IEEE Trans on Power Systems* 6(3), pp. 1203–1209.

Ouyang, Z. and S.M. Shahidehpour. (1992). "A Hybrid Artificial Neural Network/Dynamic Programming Approach to Unit Commitment." *IEEE Trans. on Power Systems* 7(1), 236–242.

Pang, C.K. and H.C. Chen. (1976). "Optimal Short-Term Thermal Unit Commitment." *IEEE Trans. on Power Systems* 95(4), 1336–1246.

Pang, C.K., G.B. Sheble, and F. Albuyeh. (1981). "Evaluation of Dynamic Programming Based Methods and Multiple Area Representation for Thermal Unit Commitment." *IEEE Trans. on Power Systems* PAS-100(3), 1212–1218.

Powell, D. and M.M. Skolnick. (1991). "Using Genetic Algorithm in Engineering Design Optimization with Nonlinear Constraints." In S. Forrest (ed.), *Proc. 5th Int. Conf. Genetic Algorithms*, Los Altos, CA: Morgan Kaufmann, pp. 151–157.

Sasaki, H., M. Watabable, J. Kubokawa, N. Yorino, and R. Yokoyama. (1992). "A Solution Method of Unit Commitment by Artificial Neural Networks." *IEEE Trans. on Power Systems* 7(1), 974–985.

Schoenauer, M. and S. Xanthakis. (1993). "Constrained GA Optimization." In *Proc. 5th Int. Conf. Genetic Algorithms*, Los Altos, CA: Morgan Kaufmann, pp. 573–580.

Sheble, G.B. (1990). "Solution of the Unit Commitment Problem by the Method of Unit Periods." *IEEE Trans. on Power Systems* 5(1), 257–260.

Sheble, G.B. and T.T. Maifeld. (1994). "Unit Commitment by Genetic Algorithm and Expert System." *Electric Power System Research* 30, 115–121.

Shi, Y.H. and R.C. Eberhart. (1998). "A Modified Particle Swarm Optimizer." In *Proceedings of the IEEE International Conference on Evolutionary Computation*, Anchorage, Alaska, May 4–9.

Shi, Y.H. and R.C. Eberhart. (1999). "Empirical Study of Particle Swarm Optimization." *1999 Congress on Evolutionary Computation*, Washington, DC, USA, July 6–9.

Snyder, W.L. Jr., H.D. Powell Jr., and J.C. Rayburn. (1987). "Dynamic Programming Approach to Unit Commitment." *IEEE Trans. On Power Systems* 2, 339–350.

Su, C.C. and Y.Y. Hsu. (1991). "Fuzzy Dynamic Programming: An Application to Unit Commitment." *IEEE Trans. on Power Systems* 6(3), 1231–1237.

Ting, T.O., M.V.C. Rao, and C.K. Loo. (submitted in 2002). "On the Superb Particle Swarm Optimization." *IEEE Trans. on Evolutionary Computation (under revision).*

Wood, A. and B. Wollenberg. (1996). *Power Generation Operation and Control*, 2nd ed., New York: Wiley.

Zhuang, F. and F.D. Galiana. (1988). "Toward a more Rigorous and Practical Unit Commitment by Lagrangian Relaxation." *IEEE Trans. on Power Systems* 3(2), 763–772.

Zhuang, F. and F.D. Galiana. (1990). "Unit Commitment by Simulated Annealing." *IEEE Trans. on Power Systems* 5(1), 311–317.