Proceedings of the 6th WSEAS Int. Conf. on EVOLUTIONARY COMPUTING, Lisbon, Portugal, June 16-18, 2005 (pp167-175)

Ant Colony Optimization for Economic Generator Scheduling and Load Dispatch

K. S. Swarup

Abstract—**Feasibility of application of Ant Colony Optimization to two case studies of economic load dispatch and generation scheduling are presented. Ant Colony Optimization (ACO) is a meta-heuristic approach for solving hard combinatorial optimization problems. The inspiring source of ACO is the pheromone trail laying and following behavior of real ants which use pheromones as a communication medium. In analogy to the biological example, ACO is based on the indirect communication of a colony of simple agents, called (artificial) ants, mediated by (artificial) pheromone trails. The pheromone trails in ACO serve as distributed, numerical information which the ants use to probabilistically construct solutions to the problem being solved and which the ants adapt during the algorithm's execution to reflect their search experience. The suitability of the ant colony optimization algorithm for economic dispatch was carried out for two systems consisting of 3 and 6 generating units. The method is further extended to generator scheduling for IEEE 14, 30 and 57 bus systems respectively.**

Index Terms - **Power Systems, Optimization, Meta-heuristic, Ant Colony Optimization, Economic Dispatch, Generation Scheduling.**

I. Introduction

Power system optimization is an important field in the operation, planning and control of power systems. Many modern heuristic techniques to the solution of complex power system optimization problems have been proposed, each differing in their method of representation, implementation and solution procedure. This paper presents an new meta heuristic approach to power system optimization problems namely economic dispatch and generation unit commitment.

II. Ant Colony Behavior

Real Ants

In many ants the visual system is very simple. Some species are completely blind. Communication between ants and between ants and their environment is often based on the use of chemical signals. Pheromones are produced by ants and they deposit them on trials when walking in search of food. By sensing the pheromone, the following ants can find food. Inspirational source of ant colony algorithms is the double bridge experiment described below.

K.S.Swarup is with the Department of Electrical Engineering , Indian Institute of Technology, Madras, INDIA. (e-mail: swarup@ee.iitm.ac.in).

Double bridge experiment

Fig 1. shows the double bridge experiment to illustrate the ant behaviour. Ant nest is connected to food source via two paths of differing length. Initially, ants move randomly and chose between shorter and longer path with equal probability. While walking, ants deposited pheromone. When choosing a path, ants chose with higher probability the path with the highest pheromone concentration. Ants choosing the short path will be first back with food. Therefore, trail on shorter path grows more quickly

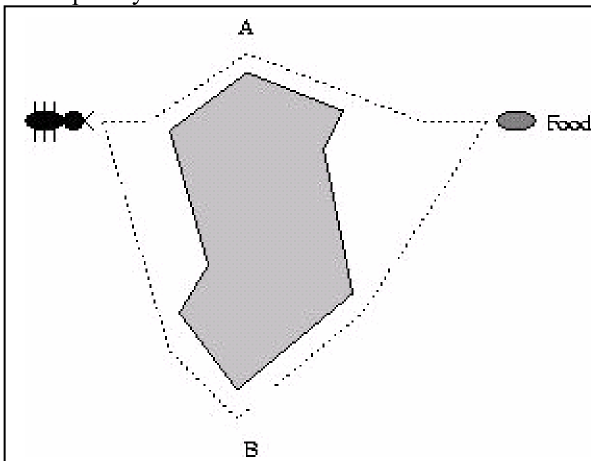

Fig 1. Double Bridge Experiment.

Auto Catalysis:

Positive feedback. It is a Self-reinforcing process If no *limiting* mechanism is in place, it leads to explosion. It is the central mechanism in ant algorithms. Probability of an ant choosing a path increases with the number of ants that chose the same path.

It is interesting to understand how ants, which are almost blind animals with very simple individual capabilities, act together in a colony and find the shortest route between the ant's nest and a source of food. They are also capable of adapting to changes in the environment, for example, finding a new shortest path once the old one is no longer feasible due to a new obstacle. The studies by ethnologists reveal that such capabilities that the ants have are essentially due to what is called "pheromone trails" that ants use to communicate information among individuals regarding path and decide where to go. Ants deposit a certain amount of pheromone while walking, and each ant probabilistically prefers to follow a direction rich in pheromone rather than a poorer one. In case of an obstacle in place, these ants that choose, by chance, the shorter path around the obstacle will more rapidly reconstitute the interrupted pheromone trail compared to those that choose the longer path. Hence the shorter path will receive a higher amount of pheromone in the time unit and this will in turn cause a higher number of ants to choose the shorter path (auto catalytic) process, very soon all the ants will choose the shorter path.

The Abstract Algorithm

- Colony of artificial ants build solutions to a given problem by moving on the problem's graph representation
- Each feasible path represents a solution to the problem
- They move by employing a probabilistic *local* decision rule that exploits pheromone trail values
- Once an ant has built a solution (or while the solution is being built), the ant evaluates the quality of the solution, and deposits pheromone on the components it used
- This directs the search of the ants in the future

Other Algorithmic Components:

- *Evaporation* and *daemon* actions
- Evaporation – process by which pheromone concentrations decrease over time
- Needed to avoid too-rapid convergence of the algorithm to a sub-optimal region
- Implements a form of "forgetting", favouring the exploration of new areas of the search space
- Daemon actions used to implement centralised activities that are not undertaken by single ants
- Example: deposit extra pheromone on the components used by the ant that built the best overall solution at the last iteration ("reward")

Formulation of the approach:

- Explicitly formulated in terms of computational *agents*
- Might, in principle, be possible to get rid of individual agents and concentrate on core mechanisms (reinforcement and evaporation)
- However, the agent-based formulation may be more flexible, and provide a useful aid to designing problem solving systems

III. A Simple Ant Colony Algorithm

Figure 2 shows the simple ant colony algorithm. The working of the can be described by means of the following.

1) ***Initialize A(t):*** The problem parameters are encoded as a real number. Before each run, the initial population (Nest) of the colony are generated randomly within the feasible region which will crawl to different directions at a radius not greater than R.
2) ***Evaluate A(t):*** The fitness of all ants are evaluated based on their objective function.
3) ***Add_trail:*** The trail quantity is added to the particular directions the ants have selected in proportion to the ants' fitness.
4) ***Send_ants A(t):*** According to the objective function, their performance will be weighted as a fitness value which directs influence to the level of trail quantity adding to the particular directions the ants have selected. Each ant chooses the next node to move taking into account two parameters: the visibility of the node and the trail intensity of the trail previously laid by other ants. The send_ants process sends ants by selecting directions using Tournament selection based on the two parameters.
5) ***Evaporate:*** finally, the pheromone trail secreted by an ant eventually evaporates and the starting point(nest) is updates with the best tour found.

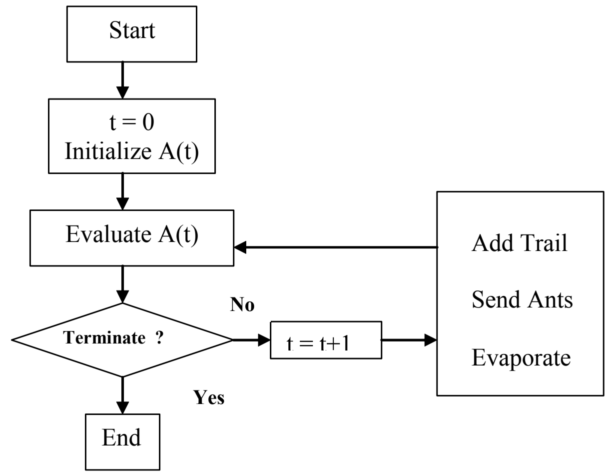

Fig 2. A Simple Ant Colony Algorithm

IV. Ant Colony Optimization

Ant Colony Optimization (ACO) is a recently proposed meta-heuristic approach for solving hard combinatorial optimization problems. The inspiring source of ACO is the pheromone trail laying and following behavior of real ants, which use pheromones as a communication medium. In analogy to the biological example, ACO is based on the indirect communication of a colony of simple agents, called (artificial) ants, mediated by (artificial) pheromone trails. The pheromone trails in ACO serve as distributed, numerical information, which the ants use to probabilistically construct solutions to the problem being solved, and which the ants adapt during the algorithm's execution to reflect their search experience. The first example of such an algorithm is Ant System (AS), which was proposed using as example application the well-known Traveling Salesman Problem (TSP). Despite encouraging initial results, AS could not compete with state-of-the-art algorithms for the TSP. Nevertheless, it had the important role of stimulating further research on algorithmic variants, which obtain much better computational performance, as well as on applications to a large variety of different problems. In fact there exists now a considerable amount of applications obtaining world class performance on problems like the quadratic assignment, vehicle routing, sequential ordering, scheduling, routing in Internet-like networks, and so on. Motivated by this success, the ACO meta-heuristic has been proposed as a common framework for the existing applications and algorithmic variants.

```
WHILE termination conditions not meet
DO
 ScheduleActivities
   AntBasedSolutionConstruction()
   PheromoneUpdate()
   DaemonActions() {optional}
 END ScheduleActivities
ENDWHILE
```

Fig 3. Pseudo Code for ACO

AntBasedSolutionConstruction(): An ant constructively builds a solution to the problem by moving through nodes of the construction graph G. Ants move by applying a stochastic local decision policy that makes use of the pheromone values and the heuristic values on components and/or connections of the construction graph. While moving, the ant keeps in memory the partial solution it has built in terms of the path it was walking on the construction graph.

PheromoneUpdate(): When adding a component ci to the current partial solution, an ant can update the values of the pheromone trails that where used for this construction step. This kind of pheromone update is called online step-by-step pheromone update. Once an ant has built a solution, it can (by using its memory) retrace the same path backward and update the pheromone trails of the used components and/or connections according to the quality of the solution it has built. This is called online delayed pheromone update. Another important concept in Ant Colony Optimization is pheromone evaporation. Pheromone evaporation is the process by means of which the pheromone trail intensity on the components decreases over time. From a practical point of view, pheromone evaporation is needed to avoid a too rapid convergence of the algorithm toward a sub-optimal region. It implements a useful form of forgetting, favoring the exploration of new areas in the search space.

DaemonActions(): Daemon actions can be used to implement centralized actions which cannot be performed by single ants. Examples are the use of a local search procedure applied to the solutions built by the ants, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a non-local perspective. As a practical example, the daemon can observe the path found by each ant in the colony and choose to deposit extra pheromone on the components used by the ant that built the best solution. Pheromone updates performed by the daemon are called offline pheromone updates.

V. ECONOMIC DISPATCH

Economic dispatch in power system operation consists of minimizing the operation costs depending on demand and subject to certain constraints. It can be formulated as follows:

1) Objective function:

$$\textit{Minimize}\ \mathbf{Cost} = \sum_{i}^{Ng} Fi(Pi) \qquad (1)$$

where **Cost** is the operating cost of the power system. Ng is the number of units.

$Fi(Pi)$ is the cost function and Pi is the power output of the unit ***i***. $Fi(Pi)$ is usually approximated by a quadratic function of its power output Pi as:

$$F_i(P_i) = a_i P_i^2 + b_i P_i + c_i \qquad (2)$$

where a_i, b_i and c_i are the cost coefficients of the unit ***i***.

Wire drawing effect occurs when each steam admission valve in a turbine starts to open, and at the same time a rippling effect on the unit curve is produced. To model the effects of valve points a recurring rectified sinusoid contribution is added to the cost function.

The result is:

$$F_i(P_i) = a_i P_i^2 + b_i P_i + c_i + \left[g_i \sin\left\{ h_i (P_i - P_i^{\min}) \right\} \right] \qquad (3)$$

where g_i and h_i are valve-point coefficients. $P_i^{\min}$ is the lower generation limit of unit ***i***. d_i is the incremental cost curve value. Ignoring the valve point effects some inaccuracy would result in dispatch.

2) **Constraints:**

a. Unit operation constraints:

$$P_i^{\min} \le P_i \le P_i^{\max}$$
$$i = 1, 2, ..., N_g \qquad (4)$$

where $P_i^{\min}$, $P_i^{\max}$ are the lower and upper generation limit of unit ***i***.

b. Power Balance:

$$\sum_{i=1}^{Ng} P_i = P_L + P_D \qquad (5)$$

where P_D is the demand and P_L is transmission loss. The transmission loss can be calculated by the B coefficients method or power flow analysis.

B coefficients used in the power system is:

$$P_L = P^T B P + P^T B_o + B_{oo} \qquad (6)$$

where P is an Ng dimensional column vector of the power output of the units.

P^T is an associate matrix of $\boldsymbol{P}$. B is an Ng X Ng coefficient matrix.

B_o is an Ng dimensional coefficient column vector. B_{oo} is a coefficient. We can also get the transmission loss by power flow analysis. Line flow constraints and system stability constraints can be expressed as follows:

c. Line Flow Constraints:

$$|Lf_i| \le Lf_i^{\max} \qquad i = 1,2......N_L \tag{7}$$

where Lf_i is the MW line flow , $Lf_i^{\max}$ is the allowable maximum flow of line i (line capacity), and N_L is the number of transmission lines subject to line capacity constraints.

d. System Stability Constraints:

$$|\partial_i - \partial_j| \le \partial_{ij}^{\max} \qquad i, j = 1,2,....N_D and i \ne j \tag{8}$$

where ∂_i, ∂_j are voltage angle of bus i and j . $\partial_{ij}^{\max}$ is the allowable maximum voltage angle. N_D is the number of buses subject to system stability constraints.

Generalized Ant Colony Optimization (GACO)

It has the characteristics of positive feedback, distributed computation, and the use of constructive greedy heuristic, the GACO can be used to solve the non-convex, nonlinear constrained optimization problems.

When an objective function $f(X)$ is minimized in a compact set, it must be subject to linear/nonlinear, inequality/equality constraints. We can transform those constraints, which is difficult to be dealt with in feasible region by using the penalty function.

$$\min F(X) = f(X) + \sigma_1 \sum_{i=1}^{l_0} (h_i(X))^2 + \sigma_2 \sum_{j=1}^{u_0} \left[\max\{0, -g_j(X)\}\right]^2$$
$$h_i(X) = 0, i = 1,2,....,l - l_0 \tag{9}$$
$$g_j(X) \ge 0, j = 1,2,...,u - u_0$$

where $f(X)$ is the original objective function. $X = (x_1, x_2, ..., x_n)^T$ is a n-dimensional vector. l , u are the numbers of equality and inequality constraints of original problem. $\sigma 1$ and $\sigma 2$ are the penalty factors.

$$\sigma i(K) = \left(\frac{2}{1 + \exp -(\frac{aK}{T})} - 1 \right) \sigma_i^0 \quad i = 1,2 \tag{10}$$

where $\sigma_i(K)$ is the penalty factor made on iteration K. a is a positive parameter.

T is the upper limit of iterative times. σ_i^0 is the upper limit of $\sigma_i(K)$

VI. Economic Dispatch Using Ant Colony Optimization

Solution Coding:

Let $Xi = (x_1, x_2, ..., x_{Ng})$ be a vector denoting the i th individual of the ant colony, where Ng is the number of units and Xi is the generated power output of unit i. At initialization phrase, X_i is selected randomly from the selected region S.

Objective function and Feasible region:

In order to minimize the objective function of ED the constraints have to be obeyed. We can use penalty function to transform those constraints difficult to deal with in the feasible region including power balance, etc. the feasible region $\boldsymbol{S}$ is determined by the unit operation constraints.

Parameter Setting:

The parameters used here are $\boldsymbol{N}$=10-50, $\tau_0 = 1.0$, $\gamma_1 = \lambda_2 = 1$, $\boldsymbol{T}$=400-1000, $\boldsymbol{a}$=10-50, $\boldsymbol{r}$=0.08-0.5, $\rho = 0.9$, $\boldsymbol{NI}$=10. Upper limit of visibility is **500** and penalty factor is **150.**

VII. ACO Algorithm for Economic Dispatch

The Ant Colony Optimization Algorithm for economic dispatch consists of the following steps.

Step 1: Initialization: An initial population of ant colony individuals is selected randomly from the feasible region S. Typically; the distribution of initial trails is uniform. Visibility is defined and this quantity is modified during the run of the program. At the beginning, the ants can search on a large scale. With the running of the program the visibility decreases and the exactitude of the search increases gradually.

Step 2: Ai set is defined. If Ai is not equal to *phi*, which is an empty set, then go to Step 3, else go to Step 4.

Step 3: Let m be the quantity of elements in Ai and transitional probability is defined. Po is the probability of the neighborhood search. If the selection result is Pij then update rule 1 is carried out.

Update rule 1: Moving an ant from point i to j . Go to Step 5
If the selection result is Po then update rule 2 is carried out.

Update rule 2: Carrying out a search in the neighborhood of X. Go to Step 5.

Step 4: Searching in neighborhood. Let the result be Y.

Step 5: Updating the trail intensity matrix.

Step 6: After iteration all the ants have completed one move, calculate the results.

1) If convergence is not achieved, cancel the result from step 2 to step 4 and go to step 2.
2) If the results are not changed after *NI* iterations, disturb the ant colony by increasing the visibility and neighborhood search. *NI* is a coefficient.

VIII. Simulation Results for Ant Colony Optimization

Case Study I: 3 Generator System.

A computer program implementing the proposed algorithm was first prepared and run for a 3 generator system. A comparison with Lambda method and Genetic Algorithm (GA) is provided in table 2. In ACO the following parameters are chosen heuristically. No of ants=50, No of cycles=10, Alpha=0.5, Beta=0.05, Forget factor=0.9, Q=50 Cost coefficients and power range for calculating the operation cost is given in table 1.

TABLE 1: GENERATOR COST COEFFICIENTS FOR A 3 UNIT SYSTEM

Unit	Pmax (MW)	Pmin (MW)	ai	bi	ci	di	gi	hi
1	600	100	.001562	7.92	561	.003124	300	.0315
2	400	100	.00194	7.85	310	.00388	200	.042
3	200	50	.00482	7.97	78	.00964	150	.063

The load demand here is 850MW w/o losses.

TABLE 2: RESULTS FOR 850MW LOAD FOR A 3UNIT SYSTEM

Case Study	Unit1 (MW)	Unit2 (MW)	Unit3 (MW)	Cost ($/h)
Lambda Method	393.1698	334.6038	122.2264	8194.37
ACO	**394.11**	**333.12**	**122.3**	**8195.1**
GA	300	400	150	8237.6

The B-Loss coefficients for computing the losses are given below

$$[B] = \begin{pmatrix} 0.0676 & 0.00953 & -0.00507 \\ 0.00953 & 0.05210 & 0.00901 \\ -0.00507 & 0.00901 & 0.29400 \end{pmatrix}$$

$$B_0 = \begin{bmatrix} -0.07660 \\ -0.00342 \\ 0.01890 \end{bmatrix}$$

$$B_{00} = 0.040357$$

Table 3 shows the comparison of the ACO for 3 generating unit system with other methods like Genetic Algorithm, and constrained optimization.

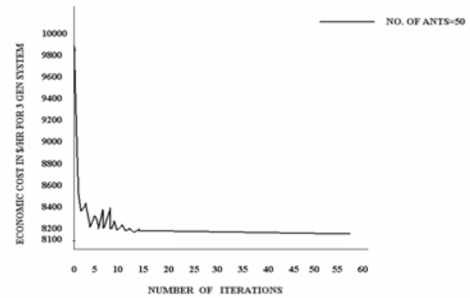

Fig. 4. Cost optimization for 3 generator system

TABLE 3: COMPARISON OF RESULTS FOR 500MW LOAD FOR A 3 UNIT SYSTEM CONSIDERING LOSSES.

Case Study	Unit1 (MW)	Unit2 (MW)	Unit3 (MW)	Losses(MW)	Cost ($/h)
GA	300.01	170.2	100.1	70.99	5745.11
ACO	**299.46**	**171.93**	**99.84**	**71.22**	**5735.74**
CO	299.46	172	98.84	70.24	5735.93

Case Study II: 6 Generator System.

Table 4 shows the Generator cost coefficients for a 6 unit system. Load demand is 1800MW. Transmission Losses are ignored.

Table 4: Data for 6 generator system

Unit	Pmax (MW)	Pmin (MW)	ai	bi	ci	di	gi	hi
1	600	100	.001562	7.92	561	.003124	300	.0315
2	400	100	.00194	7.85	310	.00388	200	.042
3	200	50	.00482	7.97	78	.00964	150	.063
4	590	140	.00139	7.06	500	.00278	200	.054
5	440	110	.00184	7.46	295	.00368	250	.062
6	440	110	.00184	7.46	295	.00368	250	.062

The following parameters are chosen heuristically for the ACO. No of ants=50; No of cycles=20; Alpha=0.5; Beta=0.05; Forget factor=0.9; Q=50. Table 5 and figure 5 show the result and the cost optimization for the 6 generator unit system

TABLE 5: RESULTS FOR 6 GENERATOR SYSTEM

Case Study	UNIT1 (MW)	UNIT2 (MW)	UNIT3 (MW)	UNIT4 (MW)	UNIT5 (MW)	UNIT6 (MW)	Fuel Cost ($/h)
NEWTON	184	166.2	54.4	590	402.7	402.7	16609.57
ACO	**248.27**	**217.36**	**74.94**	**588.37**	**335.78**	**335.28**	**16579.33**
GA	250.49	215.43	109.9	572.84	325.66	325.66	16585.85

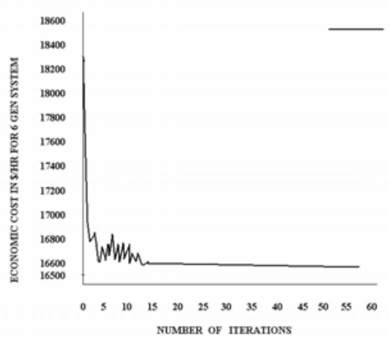

Fig. 5. Cost optimization for 6 generator system

The ACO approach to economic dispatch ahs been further tested for the IEEE 14, 30 and 57 bus systems respectively. The comparisons of the results are shown in table 6.

TABLE 6: RESULTS FOR IEEE 14, 30, 57 BUS SYSTEMS

Case study	Base case generation (MW)	Optimum generation	Load (MW)	Optimum Cost (in $/h)
IEEE 14 Bus System	P1=210.660 P2=40 P6=20 Losses=11.66	P1=159.44 P2=67.89 P6=39.96 Losses=8.6	259.0	1134.98
IEEE 30 Bus System	P1=238.48 P2=40 P11=20 Losses=15.08	P1=164.233 P2=73.98 P11=54.234 Losses=9.9	283.4	1244.76
IEEE 57 Bus System	P1=478.6570 P3=40 P8=450 P12=310 Losses=27.857	P1=411.125 P3=99.56 P8=405.678 P12=358.45 Losses=20.8	1250.8	6517.876

Tables 7 8 and 9 show the comparison of the ACO with conventional methods for the standard IEEE 14, 30 and 57 bus systems.

TABLE 7: COST OF GENERATION OBTAINED BY DIFFERENT TECHNIQUES FOR IEEE 14 BUS SYSTEM:

S. no	Techniques used	Total cost of generation (in $/h)
1	QP method	1134.277
2	GA	1136.64
3	**ACO**	**1134.98**

TABLE 8: COST OF GENERATION OBTAINED BY DIFFERENT TECHNIQUES FOR IEEE 30 BUS SYSTEM:

S. no	Techniques used	Total cost of generation (in $/h)
1	QP method	1244.426
2	GA	1245.81
3	**ACO**	**1244.76**

TABLE 9: COST OF GENERATION OBTAINED BY DIFFERENT TECHNIQUES FOR IEEE 57 BUS SYSTEM:

S. no	Techniques used	Total cost of generation (in $/h)
1	QP	6523.47
2	**ACO**	**6517.876**

Figures 6, 7 and 8 show the Fuel cost optimization of the ACO for the standard IEEE 14, 30 and 57 bus systems.

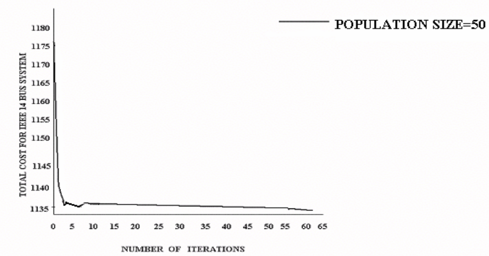

Fig. 6. Fuel Cost Optimization for IEEE 14 Bus system

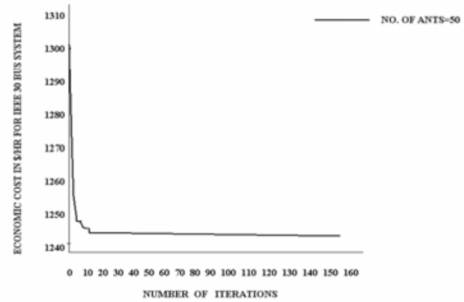

Fig. 7. Fuel Cost Optimization for IEEE 30 Bus system

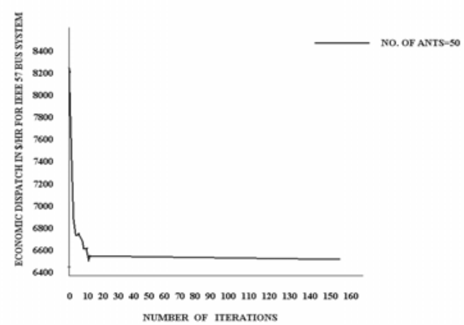

Fig. 8. Fuel Cost Optimization for IEEE 57 Bus system

IX. SHORT TERM GENERATION SCHEDULING USING ANT COLONY SEARCH ALGORITHM (ACSA)

To supply a high quality of electric energy to the consumer in a secure and economic manner, electric utilities face many economical and technical problems in operation, planning and control of electric energy systems. One of the most important problems is to determine the most economic and secure way of short-term generation scheduling and dispatch such that the constraints are satisfied simultaneously.

Here a novel co-operating agents approach, ant colony search algorithm(ACSA)- based scheme, for solving a short-term generation scheduling problem of thermal power systems. In the ACSA, the state transition rule, global and local updating rules are also introduced to ensure an optimal solution. Once all the ants have completed their tours, a global pheromone updating rule is then applied and the process is iterated until the stop condition is satisfied. The effectiveness of the proposed scheme has been demonstrated on the daily generation scheduling problem of model power systems.

Constraints Considered

Spinning reserve constraints:

$$\left(\sum_{i=1}^{G} u_{ij}P_i^{\max} - P_{Dj}\right) / P_{Dj} \geq 0.1,\ j \in T \qquad (11)$$

where, u_{ij} is the status index of the i unit at the j stage(1 for up and 0 for down)

Minimum up time of units:

$$\left(u_{ij} - u_{ij-1}\right)\left(w_{ij-1} - \tau h_i\right) \leq 0, i \in G, j \in T \qquad (12)$$

where, τh_i is the minimum up time of the i unit and $w_i = u_{ij}\left(w_{ij-1} + 1\right)$

Minimum down time of units:

$$\left(u_{ij} - u_{ij-1}\right)\left(q_{ij-1} - \tau l_i\right) \geq 0, i \in G, j \in T \qquad (13)$$

where, τl_i is the minimum down time of the i unit and $q_{ij} = \left(1 - u_{ij}\right)\left(q_{ij-1} + 1\right)$

Maximum operating time of units:

$$u_{ij,}\left(v_{ij-1} - \tau u_i\right) \leq 0, i \in G, j \in T \qquad (14)$$

where τu_i is the maximum operating time of i th unit and $v_{ij} = u_{ij}\left(v_{ij-1} + 1\right)$

The objective function to be minimized is given as

$$f(\pi) = \sum_{i=1}^{n} tc\left(s_{\pi(i)} s_{\pi(i+1)}\right) \qquad (15)$$

where $tc\left(s_i, s_j\right)$ is the total transition cost between state i and state j that is given and

$\pi(i)$ for i=1, n defines a permutation. Let m be the number of ants, then

$$m = \sum_{i=1}^{n} b_i(t) \qquad (16)$$

where $b_i(t)$ is the number of ants in state i at time t.

There is also a global structure that represents the nest neighborhood. In terms of GSP it represents the transition cost between each pair of states and the trail left by the ants in the course of the algorithm execution. When the ant system, is applied to symmetric instances of he traveling salesman problem, each ant generates a complete tour by choosing the cities according to a probabilistic state transition rule to build a solution and a local pheromone updating rule will be followed. Once all the ants have completed their tours a global pheromone updating rule is then applied and the process is iterated until the end condition is satisfied.

The state transition rule used by the ant system is called random proportional rule given by

$$p_k(i,j) = \frac{\left[\tau(i,j)\right]\left[\eta(i,j)\right]^{\beta}}{\sum_{u \in J_k(i)} \left[\tau(i,u)\right]\left[\eta(i,u)\right]^{\beta}},\ j \in J_k(i) \qquad (17)$$

$= 0$ otherwise

which gives the probability with which ant k in city, i chooses to move to the state j. Here τ is the pheromone , $=1/\partial$ is the inverse of the distance. While constructing its tour, an ant also modifies the amount of pheromone on the visited edges by applying the local updating rule given by

$$\tau(i,j) \leftarrow (1-\rho)\tau(i,j) + \rho\Delta\tau(i,j) \qquad (18)$$

where 0< ρ <1 is a parameter. The global updating rule is finally implemented as follows. Once all the ants have built their tours, the pheromone is updated on all the edges according to

$$\tau(i,j) \leftarrow (1-\alpha)\tau(i,j) + \sum_{k=1}^{m} \Delta\tau_k(i,j) \qquad (19)$$

where 0< α <1 is a pheromone decay parameter. m is number of ants.

ACSA ALGORITHM:

The ACSA algorithm shown in figure 9 consists of the following important steps.

1) Form the search space.
2) *m* ants are initially positioned in n states.
3) Each ant builds a tour by repeatedly applying the state transition rule
4) By applying the local updating rule, amount of pheromone is changed. Once all ants have terminated their tour, the amount of pheromone is modified again by applying global rule.
5) Seek the best tour using the solution process.
6) Pheromone updating rules are so designed so that they give more pheromone to edges which should be visited by ants.
7) The overall flow of the ACSA based technique is given in the algorithm below.

Generator Scheduling Data:

The method discussed in the previous section deals with 24h generation scheduling or allocation problem. The general and fuel characteristics of units of the model system (6UNITS) are given below in table 10 below.

TABLE 10: FUEL CHARACTERISTICS OF 6 UNIT SYSTEM

No	Name	Pmax	Pmin	a	b	c	Cost	PF	$\lambda^{\min}$	$\lambda^{\max}$
1	UT01	15	60	.00510	2.2034	15	11	1	25.92	30.97
2	UT02	20	70	.00396	1.9161	25	11	1	22.82	27.18
3	UT03	23	60	.00482	1.6966	12	11	1	21.102	25.03
4	UT04	25	90	.00261	1.5354	72	11	1	18.325	22.06
5	UT05	50	190	.00193	1.0818	10	9	1	11.473	16.36
6	UT06	60	200	.00171	1.0543	16	9	1	11.335	15.65

The numeric parameters are set to the following values. $\alpha = \rho = 0.1, \beta = 2, \tau_0 = \frac{1}{L_{nn}}$, where L_{nn} is the tour length produced by the nearest neighbor heuristic and *n* is the number of states, and the number of ant is heuristically chosen to be 50.

Proceedings of the 6th WSEAS Int. Conf. on EVOLUTIONARY COMPUTING, Lisbon, Portugal, June 16-18, 2005 (pp167-175)

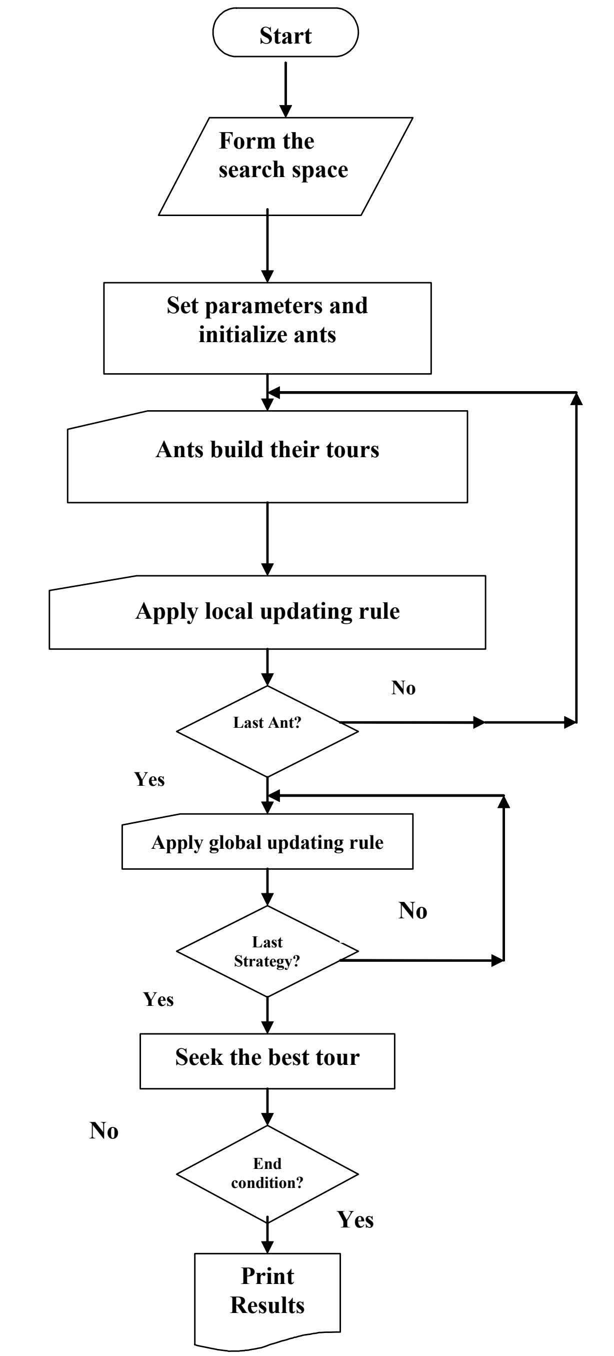

Fig 9. Flow Chart of Generation Scheduling using ACSA

X. Scheduling Results

Table 11 shows the scheduling of units obtained by the ACO. In the generation scheduling we have decided depending on the cost of generation that which units should be switched on and which should be kept off in the 24 hr period according to the varying load. In ED the units were kept on all the time and the generation was decided to minimize the cost. Hence by using this technique we should see a difference in the cost incurred in generation. Figure 10 shows the generation scheduling obtained by the ACO.

This method gives us \$ 184840.3 whereas just ED gives us \$ 187125.9 for the same case. The study results indicate that, in terms of both economy and optimality, the proposed ACSA-based optimization technique is applicable to the short term generation problem of thermal power systems.

Further studies should be done to investigate the feasibility of the algorithm in large systems with more complicated constraints.

Table 11: Generation Scheduling over 24 hrs for the 6 generator system

Time	1	2	3	4	5	6	7	8	9	10	11	12
Unit1	0	0	0	0	0	0	15	16	15	15	15	16
Unit2	0	0	0	0	0	0	0	28	52	22	21	31
Unit3	0	0	0	0	25	25	34	46	60	26	38	50
Unit4	26	26	0	0	0	0	0	0	0	80	89	90
Unit5	166	155	160	153	157	167	189	188	190	191	190	190
Unit6	198	187	185	176	183	198	200	202	200	201	200	200

Time	13	14	15	16	17	18	19	20	21	22	23	24
Unit1	15	14	15	15	16	16	16	16	15	0	0	0
Unit2	21	28	20	21	54	38	20	33	45	31	20	0
Unit3	35	45	38	27	58	54	31	51	60	47	26	23
Unit4	89	90	82	0	0	0	0	0	0	0	0	0
Unit5	190	190	190	190	190	190	190	190	190	190	190	186
Unit6	200	200	200	200	200	200	200	200	200	200	200	200

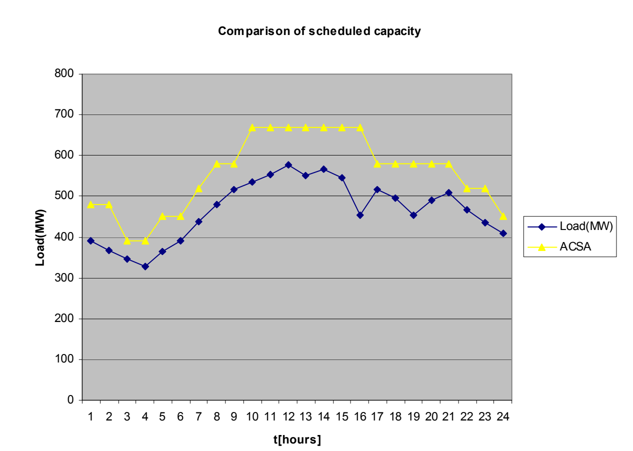

Fig. 10. Generation Scheduling obtained by ACO.

VII Conclusions

The field of ACO algorithms is very lively, as testified for example by the successful biannual workshop, where researchers meet to discuss the properties of ACO and other ant algorithms, both theoretically and experimentally.

From the theory side, researchers are trying either to extend the scope of existing theoretical results, or to find principled ways to set parameters values.

From the experimental side, most of the current research is in the direction of increasing the number of problems that are successfully solved by ACO algorithms, including real-world, industrial applications.

Currently, the great majority of problems attacked by ACO are static and well-defined combinatorial optimization problems, that is, problems for which all the necessary information is available and does not change during problem solution.

For this kind of problems ACO algorithms must compete with very well established algorithms, often specialized for the given problem. Also, very often the role played by local search is extremely important to obtain good results. Although rather successful on these problems, it is believed that ACO algorithms will really prove their strength when they will be systematically applied to "ill-structured" problems for which it is not clear how to apply local search, or to highly dynamic domains with only local information available.

A first step in this direction has already been done with the application to telecommunications networks routing, but more research is necessary. More refinement in infeasibility detection is required.

The problem can be extended to further dispatch problems with prohibiting operating zones and environmental constraints. It can also be applied to other large scale power system optimization problems like Optimal Power Flow, etc.

Important Contributions

In the case of Economic dispatch, GACO is able to solve complicated, non convex, nonlinear problems.

It achieves good convergence and provides accurate dispatch solutions in reasonable time.

The results show that GACO is robust, accurate and efficient.

Further work is required for searching the neighborhood, and present more efficacious sufficient conditions for convergence.

Various practical applications of new method wait for further development as well.

The effectiveness of the ACSA has been demonstrated on the daily generation scheduling problems of model power systems and in terms of both economy and optimality, it is applicable.

Further studies are being conducted to investigate the feasibility of the algorithm in large systems with more complicated constraints.

VIII REFERENCES

[1]M. Dorigo and G. Di Caro, 1999. The Ant Colony Optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover editors, New Ideas in Optimization}, pages 11-32. McGraw-Hill.

[2] M. Dorigo, G. Di Caro and L. M. Gambardella, 1999. Ant algorithms for discrete optimization. Artificial Life, 5, 2, pages 137-172.

[3] M. Dorigo, V. Maniezzo and A. Colorni, 1996. Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics - Part B, 26, 1, pages 29-41.

[4] M. Dorigo and T. Stützle, 2002. The ant colony optimization metaheuristic: Algorithms, applications and advances. In F. Glover and G. Kochenberger editors, Handbook of Metaheuristics, volume 57 of International Series in Operations Research & Management Science, pages 251-285. Kluwer Academic Publishers, Norwell, MA.

[5] M. Dorigo and T. Stützle, 2003. Ant Colony Optimization. MIT Press, Boston, MA.

[6] Coloni A, Dorigo M, Maniezzo V. An Investigation of some properties of Ant Algorithm. Proceedings of the parallel problem solving from nature 1992; Conference(PPSN92): 509-20.

[7] Modern optimization techniques in power systems. In: Song Y H, editor. Dordrecht: Kluwer, 1999

[8] Yu IK, Song Y H. Short term generation scheduling of thermal units with environmental constraints. Proceeding of IEE-Generation Transmission and Distribution, part C 1997; 144(5): 469-76

[9] Song Y H, Chou C S. Application of ant colony search algorithms in power system optimization. IEEE Power Engineering Review 1998;18(12):63-64.