Optimal Unit Commitment with Ant Colony Algorithm

Majid Alizadeh Moghadam*, Hasan Keshavarz Ziyarani

Electrical Engineering Department, University of Ghiaseddin Jamshid Kashani, Iran

*Corresponding Author: m.alizadehmoghadam@ghiaseddin.ac.ir

Copyright © 2014 Horizon Research Publishing All rights reserved.

Abstract This paper presents an improved ant colony search algorithm that is suitable for solving unit commitment (UC) problems. Ant colony algorithm (ACA) is a meta-heuristic technique for solving hard combinatorial optimization problems. It is a population-based approach that uses exploitation of positive feedback, distributed computation as well as constructive greedy heuristic. The ACA was inspired by the behavior of real ants that are capable of finding the shortest path from food sources to the nest without using visual cues. The constraints used in the solution of the UC problem using this approach are: real power balance, real power operating limits of generating units, spinning reserve, startup cost, and minimum up and down time constraints. The approach determines the units schedule followed by the consideration of unit transition related constraints. The proposed approach is expected to yield a better operational cost for the UC problem of production of 50 power plant units.

Keywords Ant Colony Search Algorithm, Optimization, and Unit Commitment

1. Introduction

The unit commitment problem is a hard combinatorial mixed integer optimization problem to determine the optimum schedule of generating units while satisfying a set of system and unit constraints. Finding a good solution to the unit commitment problem in a reasonable time is very critical, since it could mean significant annual financial savings in power generating costs.

Several solution techniques have been applied to solve the problem. These include deterministic, meta-heuristic, and hybrid approaches. Deterministic approaches include the priority list method, dynamic programming, Lagrangian Relaxation (LR). The priority list is the simplest and fastest but achieves poor final solutions. The LR method provides a fast solution but it may suffer from numerical convergence and solution quality problems.

Due to the mixed binary and continuous variable nature, of the UC problem, traditional optimization techniques may miss the optimal solution. In addition, dimensionality of the

UC problem limits the application of traditional optimization techniques to small size system. Recently, meta-heuristic approaches became popular in the effort to overcome shortcomings of traditional optimization techniques. Techniques such as genetic algorithms, evolutionary programming, simulated annealing, and tabu search have been widely investigated to solve the UC problem. These methods can accommodate more complicated constraints and are claimed to produce solutions of improved quality.

The ACA is a meta-heuristic technique proposed to find a near optimal solution to the UC problem. The ACA was inspired by the behavior of real ants which are capable of finding the shortest path from food sources to the nest without using visual cues. The technique has been tested successfully on diverse complex combinatorial optimization problems. Also, it has been shown that the ant colony method performs with little variability over problem diversity or random number seed. In addition, the ACA introduces an entirely new solution at each iteration, while other meta-heuristic optimization techniques are based on improvement of the solution or set of solutions obtained from the previous iteration. During the solution process of the UC problem using the ACA, the obtained solution is always feasible. Moreover, the ACA does not require major assumptions and approximations that limit the solution space.

The paper is organized as follows: Section 2 contains the Unit Commitment problem formulation; Section 3 introduces the paradigm of the ACA and includes a discussion on its application to solve the UC problem; Section 4 presents the performance and discussion of running the algorithms to solve the UC problem; and conclusions are presented in Section 5.

2. Problem Formulation

The objective of the unit commitment problem is to determine the output level of units in each time stage that yield the minimum total cost while satisfying the problem constraints. The UC problem is described as follows:

Subject to:

$$\text{Min } J = \sum \left[\sum v FC(P) + v(1-v)SC \right] \quad (1)$$

a) *Real power balance constraint*

$$\sum v P = P, J \in T \quad (2)$$

b) *Real power operating limits of generating unit*

$$v P \leq P \leq v P, J \in T; i \in N \quad (3)$$

c) *Spinning reserve constraint*

$$\begin{aligned} v &\geq I(1 \leq x \leq j-1), & j \in T; i \in N \\ v &\leq 1 - I(-j+1 \leq x \leq -1), & j \in T; i \in N \end{aligned} \quad (4)$$

Where:

N : number of generating units,

T : the total time horizon (hours),

$P_{i,j}$: real power output of unit i at stage j (MW),

$FC_{i,j}$: function representing the production cost of unit i

where $a_{0,i}$, $a_{1,i}$ and $a_{2,i}$ are the cost parameters:

$$(\$ / hr) : FC_{i,j}(P_{i,j}) = a_{0,i} + a_{1,i}P_{i,j} + a_{2,i}P_{i,j}^2 \quad (5)$$

PD_j : load at stage j (MW)

$v_{i,j}$: control variable, with

$$v_{i,j} = \begin{cases} 0 & \text{if unit } i \text{ is off at stage } j \\ 1 & \text{if unit } i \text{ is on at stage } j \end{cases}$$

$P_{res,j}$: power reserve at stage j (MW),

$I(x)$: logic function, 0 if x is false and 1 otherwise,

$x_{i,j}$: the consecutive time unit i has been up (+) or down (-) (hrs) from the initial stage to stage j ,

$SC_{i,j}$: the startup costs incurred during the transition of unit i from *off* to *on* during stage j

The system and unit capacity related constraints can be handled by solving the economic dispatch problem at each state. After the search space of multi-stage scheduling was determined, the unit transition related constraint and startup cost will be considered during the process of state transition by the ant colony search algorithm.

3. Ant Colony Behavior

The ant colony algorithm (ACA) mimics the behavior of real ants. It has been found suitable for solving UC problem that belong to a class of hard combinatorial optimization problems. The optimum paths followed by artificial ants (agents) are determined by their movements in a discrete time domain. While moving, agents lay some *pheromone* along their paths. Also, the agents are memory less. That is, regardless of the information acquired from their previous paths, the agents' decision to move from the present state r to the next state s is based on two measures. These are the length of the path which connects the present state to the next one, and the desirability measure (*pheromone level*). In brief, the paradigm of the ACA is such that each agent generates a

completed path (tour), by choosing the next states to move to according to a *probabilistic state transition rule*. This rule reflects the preference of agents to move on shorter paths that connect the current state to the next state and are also endowed with high level of pheromone.

In addition, while constructing its tour, an agent modifies the level of pheromone on the traversed paths by applying the *local updating rule*. The pheromone level on the traversed paths is lowered (*evaporated*) so that these paths become less attractive to other agents. This property gives agents a higher probability to explore different paths and find an improved solution. In other word, the role of the local updating rule is to make the desirability of paths change dynamically. This way, the agents can avoid being trapped in a local search based on the best previous tour.

Once all agents have reached the final state and have identified the best-tour-so-far based on the value of the objective function, they update the pheromone level on the paths that belong to the best tour by applying a *global pheromone updating rule*. This is intended to allocate a higher level of pheromone on the paths which belong to the best-tour-so-far. This rule is similar to a reinforcement learning scheme in which better solutions get a higher reinforcement. The rules that apply to agents while trying to find the best path solution are presented below.

3.1. State Transition Rule

Each agent builds a completed path to the end state, which occurs at the final stage. This takes place through the recurring application of the state transition rule. This rule calls for an agent located on state r at the current stage to move to state s at the next stage along a shorter path with a high amount of pheromone η_{rs} . This is achieved by a state transition rule that utilizes both the inverse of the length of the path η_{rs} and the amount of pheromone τ_{rs} .

In the UC problem, the transition cost from the present to the next hour TC_{rs} , simulates the length of the path taken by an agent to reach the next state s from the current state r .

$$s = \begin{cases} \arg \max_{u \in J_k(r)} \{[\tau_{ru} \eta_{ru}]^{\theta}\}, & \text{if } q \leq q_0 \\ \sigma & \text{otherwise} \end{cases} \quad (6)$$

$u(k)$: a set of states in the next stage that can be visited by agent k positioned on state r

$\eta_{rs} = \frac{1}{TC_{rs}}$: the inverse of the transition cost TC_{rs} with rs being the path from state r at the current stage to state s at the next stage

τ_{rs} : the pheromone level on the path connecting state r to state s ,

q : a uniformly distributed random number in $[0, 1]$,

q_0 : a decision parameter,

σ : a random variable selected according to the probability distribution given by :

$$P_{k,rs} = \begin{cases} \frac{[\tau_{rs}]^\theta [\eta_{rs}]^\theta}{\sum_{u \in J_k(r)} [\tau_{ru}]^\theta [\eta_{ru}]^\theta}, & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

with θ being a parameter which determines the relative importance of pheromone versus path length ($\theta > 0$).

The parameter q_0 determines the relative importance of choosing the more preferable (*exploitation*) versus less desirable (*exploration*) paths. Every time an agent in a current state r has to choose a next state s , it samples a random number $0 < q < 1$. According to (1), if $q \leq q_0$, the best path is an exploitation one, otherwise an exploration path is chosen according to (1) and (2).

3.2. Local Updating

During the establishment of its path, an agent changes the pheromone level on the traversed path (local updating) by applying the local updating rule (3). This rule has the effect of lowering the pheromone level (evaporation) on the traversed paths so that these become less attractive to other agents.

$$\tau_{rs} = (1 - \rho) \cdot \tau_{rs} + \rho \cdot \tau_0 \quad (8)$$

ρ : decay/evaporation coefficient ($0 < \rho < 1$),

τ_0 : initial pheromone level, $\tau_0 = \frac{1}{J}$ where J is a rough approximation of the optimum value of the cost function .

Using the sum of the initial minimum transition costs to estimate J and keeping it throughout the solution process was found to be a good approach.

3.3. Global Updating

After all the agents have built their individual solutions, a global pheromone updating rule is applied only to the path that labeled the best-tour-so-far. It allocates a higher pheromone level to this tour. This is equivalent to a reinforcement learning scheme in which better solutions get a higher reinforcement. Global pheromone updating is performed by applying (9)

$$\tau_{rs} = (1 - \alpha) \cdot \tau_{rs} + \alpha \cdot \Delta \tau_{rs} \quad (9)$$

$$\Delta \tau_{rs} = \begin{cases} (J^*)^{-1}, & \text{if } (r,s) \in \text{global-best tour} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

α : pheromone decay parameter ($0 < \alpha < 1$)

J^* : the best value of the objective function reached so far.

4. Ant Colony Algorithm

Figure 1 illustrates the computational flow of the ACA to solve the unit commitment problem. The main computational processes for the traditional ACA are discussed as follows

Step 1. Initialization

- i. Setup the parameters ($q_0, \theta, \alpha, \dots$, etc) and capture the system data (load, reserve, minimum up and down time, and different cost components).
- ii. Build the unit status table.

Step 2. Search space definition and economic dispatch

- i. Establishing the multi-stage search space. All the possible permutations form this search space.
- ii. Economic dispatch

Step 3. The ACA cycle

- i. Place the agents at the initial state.
- ii. Select the next state that each agent will move to by applying the state transition rules (7) and (8).
- iii. Update the local pheromone levels using (9).
- iv. Update the cost functions for each path considering the transition cost.
- v. When all agents have reached the final stage (i.e., each has generated a completed path or tour), apply the global updating rule (10) to the best-tour-so-far.
- vi. One full cycle is now completed. The process ends if no or minor improvement in the objective function value is observed; otherwise steps i-vi are repeated.

In the ACA as described above, all agents are moving together from the initial stage to the final stage. The pheromone level is updated locally after all agents move from one stage to the other. In the proposed ACA, one agent moves at a time from the initial to the final stage. During the movement of each agent the pheromone level is updated locally. Moving one agent at a time will give all the other agents the opportunity to exploit the experience of the previous agents. For example, in the first cycle of the solution of unit commitment problems, all agents start with same initial pheromone level (in this problem this value is taken 0.01). The first agent will move from the initial stage to the final stage. Movement decision is based on the initial pheromone level and the transition cost. During the movement of the first agent the pheromone level for the selected path by the agent will be updated. Now, when the second agent moves from the initial to the final stage, it uses the information (pheromone level)

Left by the previous agent as part of decision making process. The process repeated. For example, the third agent will use the information left from the previous two agents.

Figure 1. Algorithm of problem solving

5. Results

In this section we present the results. We have attempted to compare the results in different states to obtain a better view about solving the problem.

5.1. Production Cost

Following running the program, production planning is implemented for a time period of 24 hours through ant colony algorithm. In general, production rate of each unit, data of unit hourly, which should be placed into the power system and production cost, are expected to be the optimum output of an exploitation planning. Table 1 demonstrates the hourly production and its cost. As it is shown, cost has a direct relationship with production so with the production

increase the cost also increase. However, production increase is not the sole factor through the exploitation planning process, and exploitation constraints and power plants constraint will also cause cost raise. The total exploitation cost of units in 24 hours equals 6328800 dollars. This cost includes the transient and fixed cost of power plants and the extra cost imposed by the constraints to the system.

5.2. The Effect of Number of Agents (ants)

In ants colony algorithm, each ant as an agent searches for different states and figures out the optimum results. There is no special method for determining the number of agents and this number is identified experimentally. It may seem that by increasing the number of agents and consequently the number of iterations the result will be more rewarding, but in

this section we will see this is not the case. In this section within two scenarios the production cost for the first hour has been calculated, one in which the number of agents is 25 being under 50 agents for the main problem solution and the other in which the number of agents is 100, being over the number of problem solving agents.

Table 1. Amount & cost of production by hour

hour	production(<i>MW</i>)	Production cost (\$)
1	6852	155020
2	9136	183290
3	9136	179570
4	11420	242650
5	11420	222680
6	10278	203860
7	7994	166610
8	9136	190160
9	13704	265420
10	17130	336680
11	11420	335540
12	11420	226080
13	14704	263800
14	14846	296190
15	12562	247160
16	17130	338620
17	18272	359060
18	20840	406310
19	20556	403570
20	18272	348510
21	17130	334290
22	17130	327570
23	13704	263900
24	6852	142230
Total		6328800

5.2.1. The First Scenario: The Number of agents 25

Twenty five agents have been utilized to search the problem space. As shown in table 2, the best answer lies in the 4th iteration and equals 141016.1 MW, which is higher by 2.97% compared to the best problem solving result with 50 agents (136822.7 MW), showing the error of 2.97% regarding a more accurate answer to the problem. Thus, if the number of agents is under usual rate, the possibility of error incidence will increase.

Table 2. Production costs by hour for 25 agents

Iteration	cost (\$)	iteration	cost (\$)
1	146769.2	14	144629.1
2	148321.3	15	145029.7
3	142222.7	16	141936.4
4	141016.1	17	146678.2
5	144982.1	18	141761.5
6	145061.6	19	146434.2
7	142156	20	140322.4
8	142287	21	146034.3
9	146359.9	22	144710.7
10	143798.1	23	145377.4
11	142654.2	24	146316.6
12	143568	25	144629.1
13	143097.5		

5.2.3. The Problem of Local Optimization and Lack of Convergence

The use of meta-heuristic methods to solve a problem increases the probability of lack of convergence and trapping in local optimization. As stated, one of the methods to prevent this problem is to cause excitation in search space of the agents. In this project this method is used to solve the problem of local optimization and lack of convergence of algorithm. In most cases, of course, after frequent iterations convergence acquired, However, in this case the answer is either local optimization or a long period of time is required to receive the answer due to the iteration increment. Figure 2 indicates an example of the increase in the number of iterations due to the search space limitation. In this chart, total units production power is plotted in terms of different iterations. It can be observed that following 3081 iterations optimum answer has been obtained. As demonstrated in the figure, the red circles indicate trapping in non-convergent iterations in which the number of iterations reaches 1000, causing elongation of production planning process.

To solve this problem and to shorten the required time for obtaining the optimum answer, a constraint is added to the ant's colony algorithm. This constraint will lead to exciting in search space in case the number of iterations exceeds 70. As it is shown in figure 3 from the beginning to the seventieth iteration, algorithm is situated in non-convergent iterations loop (the part shown with red circle), However, with exciting from the seventieth iteration on, the algorithm is removed from the non-convergent loops trap, and the number of iterations compared to the alike previous state is reduced to 143, and the problem of extra iterations will no longer exist and a greater space will be achieved. This will reduce the calculation time which is quite essential in real time planning.

Table 3. Production costs by hour for 100 agents

iteration	cost (\$)	iteration	cost (\$)
1	145921.4	51	145658.1
2	143015.7	52	145921.6
3	141333.3	53	141595
4	140813.9	54	138641
5	143729.1	55	144254.9
6	143241.5	56	140810.9
7	146346.3	57	145601.9
8	142437.2	58	142725.1
9	141374.9	59	142865.7
10	142702.4	60	142779.3
11	140408.3	61	144642.2
12	143489.1	62	142225.7
13	148220.3	63	142278.9
14	140720.2	64	142927.5
15	142423.2	65	143282.7
16	149137.9	66	143868
17	136612.4	67	140350.4
18	139153.3	68	140017.3
19	145520.7	69	144929.7
20	144855.5	70	146735
21	142974.1	71	143641
22	147111.8	72	141415.4
23	146575.4	73	146898.5
24	140103.8	74	138803.1
25	145624.6	75	148271.1
26	144847.7	76	146301.3
27	145642	77	143706.5
28	142570.7	78	141334.6
29	147279.6	79	147322.3
30	142711.5	80	143216.9
31	138652.9	81	142590.4
32	145224.5	82	138561.9
33	141544.8	83	143073.5
34	144421.2	84	143744.3
35	140333.2	85	145243
36	143758.3	86	143611
37	146670.1	87	145245.7
38	145463.9	88	137857.1
39	144633.1	89	138923
40	142320.1	90	144161.7
41	142871.1	91	139302.7
42	140098.3	92	145033.1
43	141598.1	93	146962.3
44	143022.1	94	141993.6
45	143216.7	95	140151.1
46	143615.4	96	142410.7
47	146661.6	97	143173.2
48	142031.4	98	141385.3
49	144415	99	142619.4
50	139620.7	100	141692.8

Figure 2. Increment of iterations due to limitation of search space

Figure 3. Reduction of iterations due to the access to wider search space

6. Conclusions

This paper deals with the unit commitment problem using ant colony algorithm. To solve this problem two scenarios were taken into account in the first place. The first scenario was choosing 25 agents (ants) and the second scenario was choosing 100 agents. The results of the two scenarios were compared with each other, and ultimately 50 agents was determined which seems a more reasonable number and the results cover a wide range of search space. The results obtained from ant's colony algorithm to solve the problem of unit commitment indicates the efficiency of this algorithm in solving the problem.

REFERENCES

- [1] M. Y. El-Sharkh, N. S. Sisworahardjo, A. Rahman, M. S. Alam, "An Improved Ant Colony Search Algorithm for Unit Commitment Application," IEEE PSCE, 2006.
- [2] N.S.Sisworahardjo, A. A. EL-Keib, "Unit Commitment Using the Ant Colony Search Algorithm," IEEE. Large Engineering Systems conference on Power Engineering, 2002.
- [3] A. J. Wood and B. F. Wollenberg, "Power generation operation and control," 2nd Ed., John Wiley and Sons, New York, 1996.
- [4] C. K. Pang and H. C. Chen, "Optimal short-term thermal unit commitment," IEEE Transaction on Poiver Systems, PAS-95, pp. 1336-1341, 1976.
- [5] S. J. Wang, S. M. Shahidehpour, D. S. Kirschen, I. S. Mokhtar, and G.D. Irisarri, "Short-term generation scheduling with transmission and environmental constraints using an augmented Lagrangian relaxation," IEEE Transactions on Power Systems, vol. 10, no.3, pp. 1294-1301, Aug. 1995.
- [6] M. Aizadeh. Moghadam, S. Jalilzadeh, "Control of AGC in Interconnected Power System with Diverse Sources of Power Generation" Universal Journal of Electrical and Electronic Engineering, vol. 2, no. 7, pp. 259-269, 2014.
- [7] R. H. Lasseter, A. Akhil, C. Marnay, J. Stephens, J. Dagle, R. Guttromson, A. Meliopoulos, R. Yinger, and J. Eto, The CERTS Microgrid Concept, white paper for Transmission Reliability Program, Office of Power Technologies, U.S. Department of Energy, Apr. 2002.
- [8] P. Piagi and R. H. Lasseter, "Autonomous control of microgrids," in Proc. IEEE Power Eng. Soc. General Meeting, 2006, 2006, 8 pp.
- [9] M. C. Chandorkar, D. M. Divan, and R. Adapa, "Control of parallel connected inverters in standalone AC supply systems," IEEE Trans. Ind. Appl., vol. 29, no. 1, pp. 136-143, Jan./Feb. 1993.
- [10] N. Pogaku, M. Prodanovic, and T. C. Green, "Modeling, analysis and testing of autonomous operation of an inverter-based microgrid," IEEE Trans. Power Electron., vol. 22, no. 2, pp. 613-625, 2007.