AN EXPERT SYSTEM APPROACH TO THE UNIT COMMITMENT PROBLEM

D. P. KOTHARI[1] and AIJAZ AHMAD[2]

[1]Centre for Energy Studies, Indian Institute of Technology, Delhi, Hauz Khas, New Delhi 110 016 and
[2]Electrical Engineering Department, Regional Engineering College, Srinagar, Kashmir, India

(Received 26 March 1994; received for publication 8 December 1994)

Abstract—An expert system plays a key role in the better conservation and management of electrical energy in a power station having a number of units that have to be committed for a given load. This paper presents a hybrid expert system dynamic programming approach to the unit commitment problem. Here, the scheduling output of the usual dynamic programming is enhanced by supplementing it with the rule based expert system. The proposed system limits the number of constraints and also checks the possible constraint violations in the generated schedule. The expert system communicates with the operator in a friendly manner, and hence, the various program parameters can be adjusted to have an optimal, operationally acceptable schedule.

Unit commitment Dynamic programming Expert system

1. INTRODUCTION

In order to select the most economical schedule of units in a power system, different program algorithms have been developed in the past. These algorithms are based on mathematical programming methods, such as dynamic programming (DP) [1, 7], Lagrangian relaxation [2], branch and bound etc. Experience with unit commitment (UC) using a DP technique has shown that, in order to obtain a good schedule, operator experience should be included in setting up the control parameters. This results in tuning the heuristic data and input parameters to produce a lower cost and operationally acceptable schedule. A hybrid Artificial Neural Network (ANN)–DP approach [3], has also been used.

This paper proposes a hybrid expert system (ES) dynamic programming approach to unit commitment. Here, the scheduling output of the usual DP is enhanced by supplementing it with the rule based ES. This ES combines the knowledge of the UC programmer and a set of rules. The main feature of the ES, thus developed using Turbo Prolog V2.0, includes making possible the use of the program by a non-expert user through its ability to guide the user through the important decision making processes in the problem set up period. The system guides the user in adjusting the marginal constraints, thereby ensuring a feasible solution within economical time limits.

The method has been applied to a 10 unit system, wherein the schedule has been obtained by the ES to ensure a feasible solution. It provides a flexible programme structure that is easily adaptable to changes in the system operation.

2. EXPERT SYSTEM DEVELOPMENT

In the past, the ES approach for UC has been applied by Mokhtari *et al.* [5]. Here, they have developed an ES to analyze and improve the DP based UC. The ES tool selected is based on the inference engine of the well known Mycin ES and has been developed for the combustion turbine cycling problem. Ouang and Shahidepour [4] have proposed a UC ES which applies the EASE + NEXPERT shell to obtain a commitment schedule by using heuristic rules. A user friendly interface and graphic utilities have also been set up.

In the work presented here, a rule based ES approach [6] for handling the UC problem is based on the application of an ES to supplement the DP technique so that a more appropriate solution is achieved.

2.1. Main features of the ES development

The main features of the ES developed here are:

(i) It combines the knowledge of the UC programmer with the rule base and, hence, can lead an inexperienced operator to a better unit schedule.
(ii) The knowledge base (KB) [9, 10], which is the data or knowledge used to make decisions, consists of two parts, i.e. the working memory and the rule base. The KB, containing the knowledge of both schedulers and mathematical programmers, is represented by if/then rule statements.
(iii) Each rule represents a piece of knowledge relevant to the problem and the rules can be added, removed and modified in the knowledge base conveniently.
(iv) Production rules are close to natural language and, therefore, easy to understand. This enables the user to communicate with the ES in a friendly manner in order to adjust various programme parameters, so that an optimal and acceptable solution is approached.
(v) It can give the steps that lead to the conclusions and user's queries about the main phase of the problem. The user can confirm or correct the conclusion by examining the explanations given by the inference engine.

The ES developed functions in the following main capacities:

(i) Preprocessor of the DP results.
(ii) KB and reasoning.
(iii) Postprocessor of the result.
(iv) User consultant.

2.2. Preprocessor of the DP results

There are many constraints in the UC problem which are too difficult to include in the DP algorithm because it increases the programm execution time exponentially. In order to adjust the input data properly, the operator must have a thorough knowledge of the UC programme. The constraints, like operational and complex, are accommodated in the ES as described below.

2.2.1. Inclusion of operational constraints. The operational constraints are included by the user who is guided by the preprocessor, and this is accomplished through an interactive question and answer session during which the user will enter data and answer questions concerning the constraints, such as spinning reserve requirements, unit minimum up and down time, must run status, unit maintenance schedules, unit minimum and maximum generating limits etc. The information regarding these constraints is requested by the system and has to be supplied by the user.

Take the case of the constraint "must-run-status". Here, it is assumed that, during any given hour, one or more units in any group may be designated as "must run", i.e. those units for which there will never be any doubt about the need to commit them either because of outstanding efficiency or exceptional capacity. There may also be certain units which will be known, in advance, to be unavailable due to scheduled or forced outage. Such units are designated as "must out". All such information is supplied by the user.

2.2.2. Inclusion of complex constraints. The inclusion of complex constraints in the problem not only makes the problem difficult but increases the program execution time also. This is particularly true of constraints like "unit minimum up and down time", which are time dependent. Here, it is assumed that such a constraint is not violated frequently, so that it can be relaxed and handled external to the UC program itself by adjusting the input data which is fed to it by the preprocessor. Detections for constraint violations can be made by including rules in the knowledge.

Also, constraints like fuel constrained units, crew constraints and pollution constraints are present in the actual system operation, and these can be generalized to form production rules that would be added to the knowledge base. Here, in the present study, although it has been tried to include as many constraints as possible, yet these constraints could not be included to avoid the complexity of the problem.

2.3. Knowledge base and reasoning

All the knowledge in solving the UC problem is presented by either a property list or a production rule, i.e. facts and rules. The property list has a form of object-attribute-value. For instance, if the maximum capacity of unit-1 is 500 MW, this knowledge is stored in the knowledge base by a triple: (Unit-1, maximum capacity, 500). Similarly, if the value of cost parameter A of unit 10 is Rs. 2500, this knowledge is stored as: (Unit-10, cost A, 2500).

Other facts and system constraints are also represented in the same manner. The value portion of the triple can be update if required.

Another form of knowledge is a production rule. A rule expresses the relationship between facts and has a form of (if clause then clause). Such rule based representation allows the expert system to approach a problem in a way similar to a human expert. An example of knowledge representation is given below:

If the unit start up cost is low,
And the efficiency is high,
And the unit cost parameter values are low,
Then the unit has 'must-run-status'.

Such a rule is represented in Turbo Prolog as follows:

Hypothesis (must-run-status),
condition (start-up-cost-low),
condition (efficiency high),
condition (cost-parameter-values-low).

The inference engine examines the first predicate of the rule. If the start up cost is low, the engine goes to check the second predicate, if the efficiency is high, then it goes to check the third predicate, and it the cost parameter values are low, then the engine concludes that the unit has must-run-status.

The facts for all units, like minimum and maximum generating limits, cost curve parameters, start up times etc. and the load curve for the period are stored in the data base. This corresponds to the prolog's static database. The preceding rule about must-run-status, for example, is a part of the rule base/static database.

2.3.1. Production rules in the knowledge base. In order to have a fast solution, a production rule based and priority list based heuristics are implemented. The rules, here used, use the priority list as the starting point with additional heuristics leading to the optimal commitment decisions. The decisions in the ES developed here are made according to the following set of rules:

Rule 1: If a particular unit which is on the top of the priority list of available uncommitted units but has a higher minimum up-time (say, 4 h) than of the other units (say 1 h) and the spinning reserve requirement is insufficient only for say 1 h, then it is more economical to commit the second unit for 1 h rather than the first unit for at least the minimum up-time of 4 h.

Rule 2: If, at any stage, there is an outage associated with the normally committed units, then commit the next most economical unit.

Rule 3: Assign the value of maximum tolerable insecurity level (MTIL) to be 0.000445. If, due to some forced outage, this value of MTIL is not satisfied, then commit the units to the next nearest higher value of MTIL.

Rule 4: If the load is much less than expected, then the "must run" unit (if suggested by the user) should be one of the committed units in such a case.

Rule 5: If a unit is committed, decommitted and committed again and the "off" status in between is close or equal to the minimum down time of the unit, commit the unit continuously at the required output.

Rule 6: At any instant, if some committed units of smaller capacity can be replaced in an "off" state, then do such replacement, preserving the spinning reserve constraint.

Rule 7: All the committed units must operate at least at their minimum output conditions.

Rule 8: If the load demand is not met at any instant, first let the most efficient unit generate more power until either its maximum output is reached or the load demand is fulfilled.

Rule 9: At any stage, if the sum of the maximum generating capacity of all committed units is greater than the load demand, then decommit only that unit which has the lowest efficiency.

2.4. Postprocessor

The function of the postprocessor is to guide the user in analyzing the schedule and also make suggestions to the user about adjustments to program parameters to improve the overall schedule cost. The parameters adjusted here include minimum up and down time, unit initial conditions and priority ordering. Another function of the ES postprocessor is to detect constraint violations, which is done by including rules in the knowledge base. These rules can deduce logically a violation condition from the schedule output and, subsequently, advise the user and recommend corrections for the problem. We have seen that minimum up and down times are particularly difficult to model and cannot be incorporated directly in a DP routine. So, to detect such violations, a rule can be introduced in the knowledge base. For example, one of such rules is:

Check constraint (up-time-violated):

condition (unit on),
condition (unit off),
condition (dur-less-than-1 h).

This rule states that, if the unit is committed, then decommitted and the duration between on and off is less than 1 h, then the unit up time is violated. The condition "up-time-violated" will not be a fact in the database. There are a number of such rules which have been used in the programme for checking constraint violations.

2.5. User interface

A natural language permits the user to communicate with the ES during the consultation process using a human language. The main feature of the user interface, here, is that it works as an explanatory system which permits the user to answer a question with why. The system then responds with any information regarding the commitment of a particular unit at a particular time and so helps the user to understand the reasoning process. Any question raised by the operator concerning the result is answered by the ES, and if the results seemed to be unsatisfactory, the KB was modified and the entire process repeated until satisfactory good results were obtained.

3. SAMPLE SYSTEM

The rule based ES developed here is applied to analyze and improve the DP results of the sample system of Ref. [8]. From various consultation sessions with the ES, it was seen that the system is quite helpful in improving the results. A number of UC runs were made to test the capability and friendliness of the ES. The program has been run on an IBM-PC-AT. The consultation sessions provide almost all information about the various UC schedules given at different values of MTIL. The system also gives an explanation about any decisions taken by it. It was seen that, in order to increase the effectiveness of the system, a considerable database is required. As the rules included here in the KB have been obtained mainly from a thorough survey of the literature and are based on heuristics, therefore, it is expected that, if the rules obtained from field experts are included here, the system will obviously pave the way for handling the practical UC problem.

The sample system consists of 10 thermal generating units with different generating capacities and start up times. The time span is 24 h.

It was pointed out by the ES not to put unit 2 off for 1 h and then restart it again. It suggested that the unit must be kept on line from the fourth to the sixth hour. Further, it suggested that the value of MTIL at which the earlier schedule was computed by DP can be increased so that the overall schedule will be more economical. The final UC schedule is depicted in Table 1.

Table 1

Hour	Load (MW)	Unit status 1	2	3	4	5	6	7	8	9	10
1	2000	0	1	0	1	1	1	1	1	1	1
2	1980	0	1	0	1	1	1	1	1	1	1
3	1940	0	1	0	1	1	1	1	1	1	1
4	1900	0	1	1	1	1	1	1	1	1	1
5	1840	0	0	0	1	1	1	1	1	1	1
6	1870	0	1	0	1	1	1	1	1	1	1
7	1820	0	0	0	1	1	1	1	1	1	1
8	1700	0	0	0	1	1	1	1	1	1	1
9	1510	0	0	0	1	1	0	1	1	1	1
10	1410	0	0	0	0	1	0	1	1	1	1
11	1320	0	0	0	0	1	0	1	1	1	1
12	1260	0	0	0	0	1	0	1	1	1	1
13	1200	0	0	0	0	0	0	1	1	1	1
14	1160	0	0	0	0	0	0	1	1	1	1
15	1140	0	0	0	0	0	0	1	1	1	1
16	1160	0	0	0	0	0	0	1	1	1	1
17	1260	0	0	0	0	0	0	1	1	1	1
18	1380	0	0	1	0	1	0	1	1	1	1
19	1560	0	0	1	1	1	0	1	1	1	1
20	1700	0	0	1	1	1	1	1	1	1	1
21	1820	0	0	1	1	1	1	1	1	1	1
22	1900	0	1	1	1	1	1	1	1	1	1
23	1950	0	1	1	1	1	1	1	1	1	1
24	1990	0	1	1	1	1	1	1	1	1	1

When one compares the schedule obtained here to that obtained purely by DP [8], there is not much variation except the points discussed above, but the main feature is the capability of the ES to approach the operator in a friendly manner. A better solution may be achieved if the knowledge of a good number of field experts is included in the database.

4. CONCLUSION

An expert system and dynamic programming hybrid has been presented for solving the unit commitment problem. The dynamic programming solution of a sample system of 10 units was supplemented by the expert system, resulting in better management of the units and, hence, better energy conservation.

REFERENCES

1. P. G. Lowery, *IEEE Trans. Power Apparatus Systems* **PAS-85,** (1966).
2. Slobodan Ruzic and Nikola Rajakovic, *IEEE Trans. Power Systems* **6,** No. 1 (1991).
3. Z. Ouang and S. M. Shahidepour, *IEEE Trans. Power Systems* **6,** No. 3 (1991).
4. Z. Ouang and S. M. Shahidepour, *Electric Power System Res.* **20,** No. 3 (1990).
5. Sasan Mokhtari, Jagjit Singh and Bruce Wollenberg, *IEEE Trans. Power Systems* **3,** No. 1 (1988).
6. S. K. Tong *et al.*, *IEEE Trans. Power Systems* **6,** No. 3 (1991).
7. C. K. Pang and H. C. Chen, *IEEE Trans. Power Systems* **PAS-95,** No. 4 (1976).
8. A. K. Ayub and A. D. Patton, *IEEE Trans. Power Systems* **PAS-90,** (1971).
9. T. Sakaguchi *et al.*, *IEEE Trans. Power Systems* **PAS-102,** No. 2 (1983).
10. Tim Taylor *et al.*, *IEEE Trans. Power Systems* **4,** No. 1 (1989).
11. I. J. Nagrath and D. P. Kothari, *Power System Engineering.* McGraw–Hill, New Delhi (1994).