8-2014

A Stochastic Model for Self-scheduling Problem

Lili Zhang
University of Tennessee - Knoxville, lzhang42@utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Lili Zhang entitled "A Stochastic Model for Self-scheduling Problem." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Industrial Engineering.

Mingzhou Jin, James Ostrowski, Major Professor

We have read this thesis and recommend its acceptance:

Tsewei Wang

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

8-2014

A Stochastic Model for Self-scheduling Problem

Lili Zhang
University of Tennessee - Knoxville, lzhang42@utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Lili Zhang entitled "A Stochastic Model for Self-scheduling Problem." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Industrial Engineering.

Mingzhou Jin, James Ostrowski, Major Professor

We have read this thesis and recommend its acceptance:

Tsewei Wang

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

A Stochastic Model for Self-scheduling Problem

A Thesis Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Lili Zhang

August 2014

To my parents, uncle, and aunt

Acknowledgements

I would like to dedicate my sincere gratitude to my advisors Dr. Mingzhou Jin and Dr. James Ostrowski for their continuous support of my graduate study and research. Thanks to their instructions, the scope of my knowledge has been deepened and broadened, and my perspective of life and world becomes far-reaching. Besides my advisors, I would like to thank Dr. Tsewei Wang for being my thesis committee member and all other professors whose course I have taken for strengthening my understanding of specific topics, Dr. Garcia, Dr. Keyser, Dr. Li, Dr. Simonton, and Dr. Shylo. I also want to thank Dr. Kobza especially. My thanks also go to my fellow labmates, Nelson Granda-Marulanda, Michael Hare, Bernard Knueven, Jinglu Song, and Ying Zhang. It has been a joy to discuss problems with them. Last but not least, I would like to give thanks to Yang Tong for encouraging me in the past one year and a half.

Abstract

The unit commitment (UC) problem is a typical application of optimization techniques in the power generation and operation. Given a planning horizon, the UC problem is to find an optimal schedule of generating units, including on/off status and production level of each generating unit at each time period, in order to minimize operational costs, subject to a series of technical constraints. Because technical constraints depend on the characteristics of energy systems, the formulations of the UC problem vary with energy systems. The self-scheduling problem is a variant of the UC problem for the power generating companies to maximize their profits in a deregulated energy market. The deterministic self-scheduling UC problem is known to be polynomial-time solvable using dynamic programming. In this thesis, a stochastic model for the self-scheduling UC problem is presented and an efficient dynamic programming algorithm for the deterministic model is extended to solve the stochastic model. Solutions are compared to those obtained by traditional mixed integer programming method, in terms of the solution time and solution quality. Computational results show that the extended algorithm can obtain an optimal solution faster than Gurobi mixed-integer quadratic solver when solving a stochastic self-scheduling UC problem with a large number of scenarios. Furthermore, the results of a simulation experiment show that solutions based on a large number of scenarios can generate more average revenue or less average loss.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The energy market has undergone deregulation in the past decades. Briefly speaking, the generating companies (GENCOs) sell their electricity via auction to retailers, also known as Independent System Operator (ISO), and then retailers sell the electricity to customers [20]. Deregulation improves resource utilization and also increases competition among GENCOs. In auction, GENCOs derive their prices and bid against others. Hence, price becomes a very important uncertain parameter when GENCOs decide their production plan in order to maximize their profits. Optimization techniques have a long history of application in the area of power generation, operation, and control. The scheduling problem faced by GENCOs for the maximization of their profit is referred to as stochastic self-scheduling UC problem with the uncertainty of price.

Moreover, renewable energies such as hydro, solar and wind have been incorporated into the energy system in order to reduce the green house emissions produced by fossil fuels. The supply of these renewable resources can be influenced greatly by natural conditions such as weather which is hard to forecast accurately, thus intensifying the uncertainty of power planning. Both the deregulation of electricity market

and the integration of renewable energies have made demand/supply increasingly unpredictable. The stochastic self-scheduling UC problem proposed in this thesis can be used as a subproblem to address the UC problem with unknown demand by using Lagrangian relaxation and decomposing the original multi-unit commitment problem into subproblems with individual units.

1.2 Mathematical Preliminaries

1.2.1 Integer Programming

Integer Programming (IP) is a type of mathematical programming that requires decision variables to be positive integers. If only some decision variables are to be integers, it is referred to as the *Mixed Integer Programming* (MIP). The function of integer variables arise in many settings. For example, in the classical knapsack problem, each item is associated with a binary variable. If an item is put in the knapsack, its corresponding binary variable value will be 1; otherwise, it will be 0. In this thesis, in the mixed integer quadratic programming, three types of binary variables will be associated with the generating unit to formulate the on/off status, start-up status, and shut-down status. More information about IP and combinatorial optimization can be found in [19].

A general IP problem can be formulated as

$$
\begin{aligned}
&\max\ c^T x \\
&\text{s.t.} \\
&\qquad Ax = b \\
&\qquad x \geq 0, x \in Z,
\end{aligned}
$$

where $c \in \Re^n, b \in \Re^m$, A is a $m \times n$ matrix.

As we know, the feasible region of a linear programming (LP) problem is a convex set. If an LP has an optimal solution, then an optimal solution must be one of the extreme points of its feasible region. The methods for solving an LP like the Simplex method can find an optimal solution by searching to improve the solution along the edges. The feasible region of an IP is comprised of discrete points. IP and MIP are in the scope of NP-hard, which means that not all or none of IP and MIP problems can be solved in polynomial time by a known method. General methods for solving IP and MIP are linear relaxation, branch and bound, and cutting plane. It can also be solved by decomposition techniques such Bender's decomposition and heuristic methods like genetic algorithm. Optimization software such as Gurobi contain all these algorithms in their solvers. Before a problem is solved, it is usually preprocessed such as discarding some redundant constraints and deleting variables with fixed values in order to reduce the size of the problem.

1.2.2 Stochastic Programming

Stochastic Programming (SP) is a framework that deals with the uncertainty of parameters in the area of optimization. In many situations, we have to make a decision first before we know what is exactly going to happen and then after the event is realized, we take make-up actions. For example, a news vendor has to order papers in the morning but he doesn't know what the demand of that day will be. If the demand is larger than what he ordered in the morning, he will have to pay a penalty for a back order. Otherwise, he will have to pay for a holding cost. In this thesis, we treat the price of the electricity as an uncertain parameter. The review is focused on two-stage SP, but both formulations and solution methods of two-stage SP can be extended to multi-stage SP. More information about SP can be found in [4].

Uncertain parameters in a problem are formulated as random variables. These random variables can be continuous or discrete. When the distribution of a random

variable is discrete, it is represented by a finite number of realizations, also known as scenarios, with a particular probability associating with each scenario, where the sum of all probabilities is equal to 1. When the distribution of a random variable is continuous, SP will take advantage of its probability distribution.

SP formulates problems with different characterizations in different categories. Based on the way uncertainty is addressed, SP can be classified into expected-value SP or chance-constrained SP. In this thesis, we adopt the expected-value model because we will use a finite number of scenarios to formulate the the uncertainty of the parameter, which is price in this thesis. The idea of expected-value model is to put the expected value of second-stage objectives of all scenarios into the objective function.

We solve a SP problem by approximation, transforming it to its deterministic equivalent problem and solving its deterministic equivalent problem. To implement it, a finite number of scenarios must be constructed, either based on historical data or experts' opinions. Monte Carlo technique is usually used to reduce the size of the scenario set. The idea is to generate a sample $\{\xi_1, \xi_2, \cdots, \xi_N\}$ of N scenarios with corresponding possibility p_i for each scenario i, $i = 1, \cdots, N$. An application of this method on two-stage SP is illustrated below.

$$\begin{aligned} \min_{x \in \Re^n} \ & c^T x + E[Q(x, \xi)] \\ \text{s.t. } & Ax = b \\ & x \geq 0 \end{aligned}$$

where $Q(x, \xi)$ is the optimal value of the second-stage problem.

$$\begin{aligned} \min_{y \in \Re^m} \ & Q(x, \xi) = q(\xi)^T y \\ \text{s.t. } & T(\xi)x + Wy = h(\xi) \\ & y \geq 0 \end{aligned}$$

Using Monte Carlo sampling method, the expectation function $E[Q(x, \xi)]$ can be approximated by

$$\sum_{i=1}^{i=N} p_i Q(x, \xi_i)$$

In a multi-stage process, we have to add a nonanticipativity constraint to ensure the effectiveness of scenarios. To illustrate nonanticipativity constraint, let us consider the scenario tree in Figure 1.1. A scenario is a realization of uncertainties along the time horizon. So in this tree, each path denotes a scenario and there are three scenarios in this case. To make scenarios applicable, two conditions must be satisfied:

- The decisions made for scenario 1, 2, 3 at time 0, 1, 2 must be the same.
- The decisions made for scenario 2, 3 at time 3 must be the same.

The above two conditions are called nonanticipativity constraint.

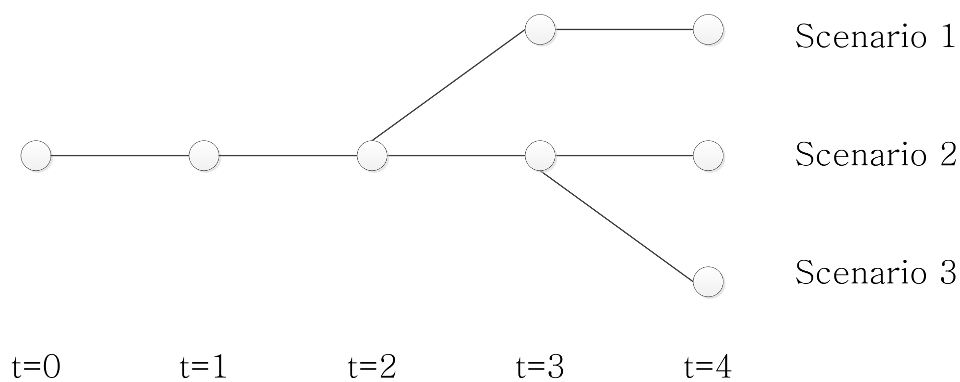


Figure 1.1: Scenario Tree

With many scenarios, this problem can become very large. It would be more efficient to explore the special structure of a SP problem. Its deterministic equivalent problem can be linear or mixed-integer programming problem. Methods for large-scale problems such as Lagrangian relaxation and Bender's decomposition can be applied.

1.2.3 Graph Theory

A graph or network is made up of nodes and arcs connecting its nodes. It can be used as a media to study many problem solutions such as shortest path, maximum capacity, and minimum-cost. Natural networks exist in many areas such as transportation, communication, and manufacturing. Moreover, many problems that do not have an obvious structure of natural network can be transformed into network problems and then be solved easily. The self-scheduling UC problem in this thesis will be transformed into a shortest path problem in the dynamic programming procedure. *Graph Theory* studies problems that use networks as their objects. More information about graph theory can be found in [2].

First of all, a network must be represented and stored in computer storage in some format (data structure) before we can use it as an object to analyze a problem. Basically there are two ways: adjacency matrix and adjacency list. Note that a graph is denoted by $G(N, E)$, where N is the set of all nodes in the graph, and E is the set of all edges in the graph.

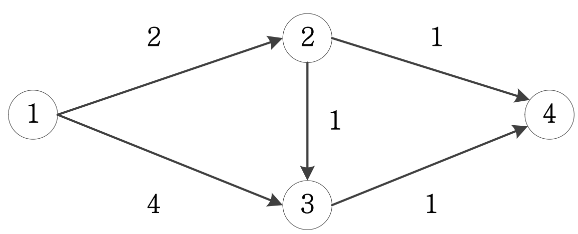


Figure 1.2: Graph Example

In Figure 1.3, the graph is represented as an $|N| \times |N|$ matrix (2-dimensional array). If there is an edge from node i to node j and the weight on the edge is w, the ijth element of the matrix will be w; otherwise, it is 0. It can be cast in a straightforward way, but regular solution approaches are not very efficient when the matrix is sparse.

In Figure 1.4, the graph is represented as an array of single linked list with the size $|N|$. The adjacency list of v, $\forall v \in G$, is denoted as $Adj[v]$ in this thesis. In a

	1	2	3	4
1	0	2	4	0
2	0	0	1	1
3	0	0	0	1
4	0	0	0	0

Figure 1.3: Adjacency Matrix

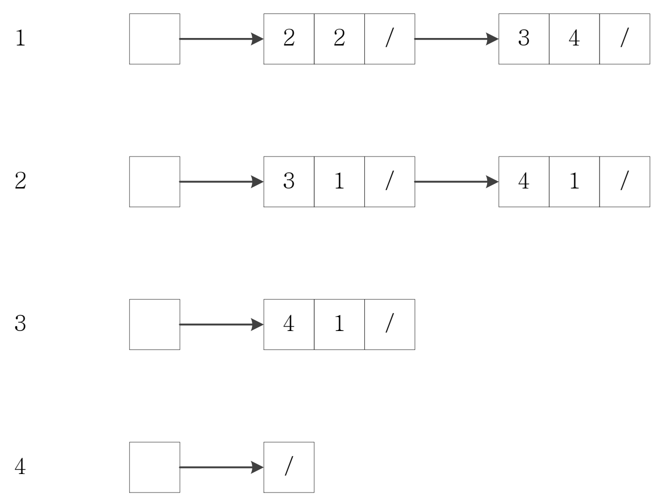


Figure 1.4: Adjacency List

single linked list, there are two types of fields, *data* and *link*. Data field stores node information. Pointer field stores the pointer which points to its next adjacency node. Double linked list can be used in order to sweep backward. Comparing to single linked list, double linked list has one more link field that stores a pointer pointing to the

previous adjacent node. In general, single linked list is enough to explore a structure of a network problem. Compared with adjacency matrix, it is harder to implement, but more space efficient. We adopt this representation in the code.

Sometimes we need to do some transformations on the original network for computational convenience. For example, in some graphs, there are also weights on nodes (see Figure 1.5). In this case, the weight of a node must be transferred to its adjacent edge in order to implement some algorithms such as the shortest-path algorithm. The weight transformation can be easily done.

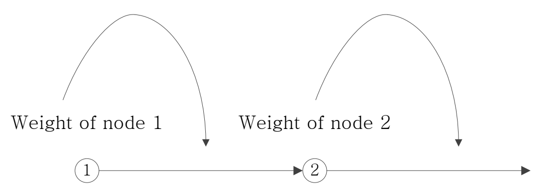


Figure 1.5: An Example of Graph Transformation

Single-source shortest path problem is a typical application in graph theory. General algorithms for computing shortest path problems are Dijsktra's algorithm and Bellman-Ford algorithm. The time complexity of Dijkstra's algorithm is $O(|E|+|N|log|N|)$ with Fibonacci heap implementation [2], but it cannot deal with a network with negative edge weight or negative cycle. The time complexity of Bellman-Ford algorithm is $O(|E||N|)$ [2]. It allows the network to have negative edge weights, but no negative cycle. There is a special group of graphs called directed acylic graphs (DAG). The algorithm for computing the shortest path in a DAG can do better. The idea is that we topologically sort a graph to get a linear representation of a graph before we compute its shortest path. Topological sorting is to rearrange a graph in a linear ordering such that every node is processed before all the nodes to which it points. As an example, Figure 1.7 shows the topological sorting of the graph in Figure 1.6.

The single-source shortest paths in a DAG is solved by Algorithm 1. The computation procedure for graph in Figure 1.7 is described from Figure 1.8, where

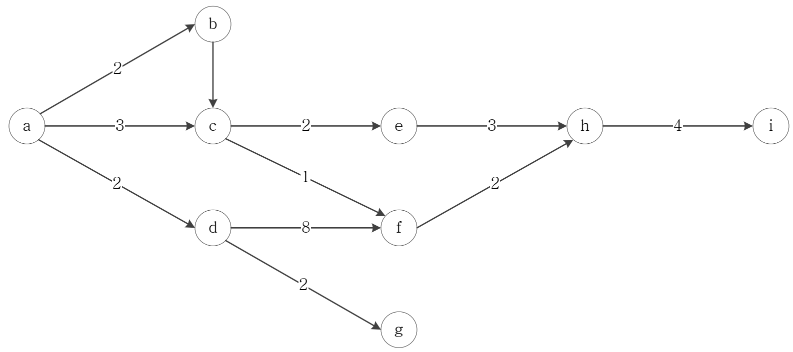


Figure 1.6: Graph before Topological Sort

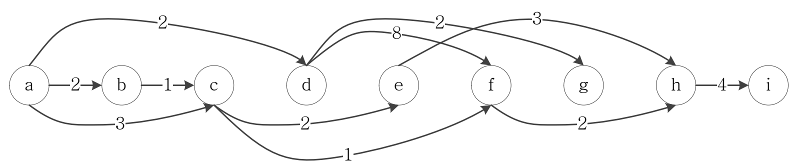


Figure 1.7: Graph after Topological Sort

the source node is node a and the number on each node denotes the shortest distance from node a to that node. The concrete procedure is as follows, where the notation of node is based on Figure 1.7.

Step 1 Initialize the shortest distance from the source node to itself to be 0 and from the source node to each other node to be infinity.

Step 2 Compute the shortest distance from node a to its adjacent nodes, which are node b, c, d. The shortest distance from node a to node b, c, d is updated from infinity to $2, 3, 2$, respectively.

Step 3 Compute the shortest distance to adjacent nodes of node b, c, d, respectively. For node b, its adjacent node is node c. The distance along the path $a-b-c$ is $2+1=3$, while the current shortest path is 3 along the path $a-c$. Hence the shortest path to node c doesn't have to be updated. For node c, its adjacent nodes include node e and f. The shortest distance to node e, f is updated from infinity to 5 and 4 along the path $a-c-e$ and $a-c-f$, respectively. For

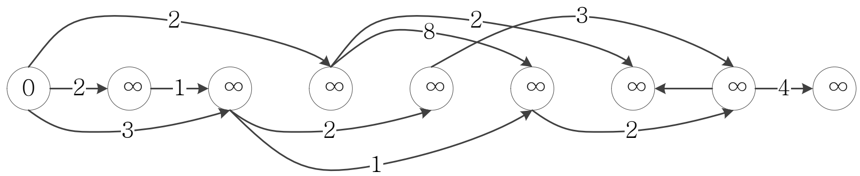
node d, its adjacent nodes include node f and g. The distance to node f along the path $a-d-f$ is 10, while the current shortest distance along the path $a-c-f$ is 4. So the shortest distance to node f doesn't have to be updated. The shortest distance to node g is updated from infinity to $2+2=4$ along the path $a-d-g$.

Step 4 Compute the shortest distance to adjacent nodes of node e, f, g, respectively. For node e, its adjacent node includes node h. The shortest distance to node h is updated from infinity to 8 along the path $a-c-e-h$. For node f, its adjacent node includes node h. The distance along the path $a-c-f-h$ is 6, which is smaller than the current shortest distance. The shortest distance is updated from 6 to 8. For node g, there is no adjacent node.

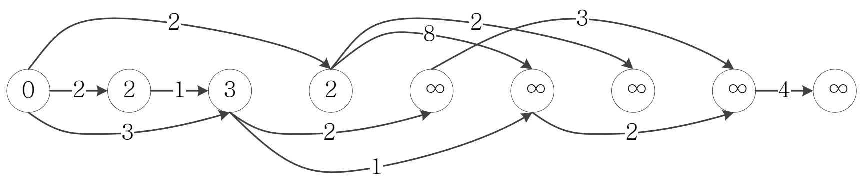
Step 5 Compute the shortest distance to adjacent nodes of node h. For node h, its adjacent node include i. The shortest distance is updated from infinity to $6+4=10$ along the path $a-c-f-h-i$.

Algorithm 1 Shortest Path Algorithm for DAG

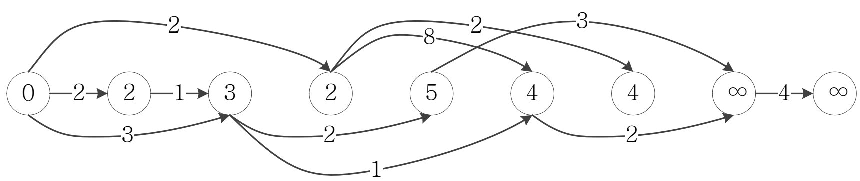
```
procedure TOPOLOGICAL SORT(G)
    Mark all nodes in Graph G unvisited
    visit v ∈ G and mark v visited
    for u ∈ Adj[v] and u is marked unvisited do
        visit u and mark u visited
    end for
end procedure
dist[] is an array that stores the shortest distance from source s to all nodes
procedure SHORTEST PATH FOR DAG(G, s)
    Topological Sort G
    Initialize dist[] = [0, inf, ··· , inf]
    for u ∈ Adj[v], ∀v ∈ G do
        if dist[u] > dist[v] + weight(u, v) then
            dist[u] = dist[v] + weight(u, v)
        end if
    end for
end procedure
```



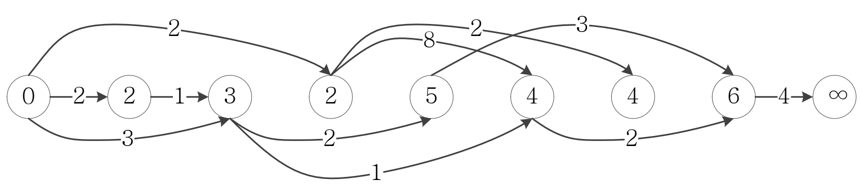
(a) Step 1



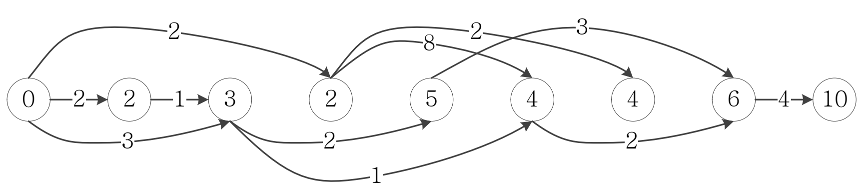
(b) Step 2



(c) Step 3



(d) Step 4



(e) Step 5

Figure 1.8: Computation of Shortest Path

1.2.4 Dynamic Programming

Dynamic programing (DP) represents a mathematical framework for computing solution to modeling problems where information and decisions evolve over time. The idea behind DP is that we first solve different parts of the problem (subproblems), then combine the solutions of the subproblems to reach an overall solution. DP is broadly applied in many settings. A shortest path problem is one of the best known applications of DP. Moreover, shortest path problems arise in a variety of settings that do not have natural network structures, based on the fact that many problems that do not have physical network structure can be represented by networks. Hence, many applications of dynamic programming can be reduced to finding the shortest (or longest) path that joins two nodes in a constructed network [43]. More information can be found in [28].

1.3 Contribution

This thesis extends an efficient DP algorithm [10] that solves the deterministic self-scheduling problem in polynomial time to deal with uncertain price and compares it with traditional MIP method both in solution quality and time efficiency. The numerical results show that the method proposed in this thesis performs better than the traditional MIP method when solving stochastic self-scheduling UC problems with a large number of scenarios. Furthermore, the results of the simulation experiment show that decisions based on more scenarios can yield more profit or less loss in average. Hence, the work in this thesis caters to the deregulation of energy market and has economic significance for GENCOs.

What's more, the proposed approach can also be used to solve the subproblem of a multiple unit commitment problem with uncertain demand, which can be decomposed into subproblems for each generating unit after Lagrangian relaxation. The resulting

model will help reduce the total production cost and ultimately the consumption of natural resources in a whole, which have been demonstrated by many references.

The self-scheduling UC problem is a variant of the UC problem , a classical problem in the power generation, operations, and control. Chapter 2 gives a review of formulations and solution methods of deterministic unit commitment problems. It also gives literature review on the stochastic UC problems and the stochastic self-scheduling UC problems. In Chapter 3, we propose a stochastic model for stochastic self-scheduling problem, present the DP solving procedure, illustrate computational experiments comparing DP method and traditional MIP method, and analyze the computational results. The conclusions and future work can be found in Chapter 4.

Chapter 2

Literature Review of Unit Commitment Problem

Unit commitment (UC) problem is a typical problem in electricity generation and operation. Given a planning horizon, its purpose is to find an optimal schedule of generating units, including on/off status and production amount of each generator at each time instance, in order to minimize operational costs, subject to a series of operational constraints. Because of the complexity of the nature of electricity generation procedure, different literature address different aspects of UC problems. Different emphasis of UC problem will lead to different formulations and probably different solution techniques. Section 2.1.1 sketches variants of formulations of UC problem that exist in the literature. Section 2.1.2, 2.2, and 2.3 discuss the development of solution techniques of deterministic UC problem, stochastic UC problem, and stochastic self-scheduling UC problem, respectively.

2.1 Unit Commitment Problem

2.1.1 Fomulations

The variants of UC problem in the literature are roughly summarized as below according to the number of units, planning horizon, energy resources, technical constraints, demand satisfactions, and uncertainty of parameters considered in the UC problems..

Number of generating units Some literature only consider one generating unit, referred to as single UC problem [10]. Other literature consider multiple generating units [23]. In most of the literature, the UC problem considers more than one generating unit. Because multiple UC problems can be decomposed into subproblems that only involve one generating unit by some decomposition techniques such as Lagrangian relaxation and Bender's decomposition, to be able to solve single UC problem efficiently will help significantly solve multiple UC problems efficiently. UC problems usually refer to multiple UC problems in the literature.

Planning Horizon The planning horizon can range from one day to several years [29]. Long-term planning, lasting more than two years, emphasizes the study of the capacity expansion decisions such as the location of new generating units [18] [46]. Medium-term planning, lasting from one month to two years, results from long-term planning and generates schedules in smaller time units such as weekly [13]. Short-term planning, lasting from several hours to one week, generates schedules in hours [26]. Most literature studied in this thesis focus on short-term UC problem.

Energy Resources Some literature only consider thermal unit or hydro unit [5] [21] [8]. Some literature consider hydro-thermal units [11] [30]. And other literature integrate some renewable energy such as wind and solar [40] [42]. Different

resources manifest themselves as different objective function and constraints in UC problems.

Technical Constraints Technical constraints depend on unit characteristics. Typical constraints are general capacity limits, minimum up-time, minimum down-time, maximum ramp-up rate, maximum ramp-down rate, and time-dependent start-up cost.

Demand Satisfaction In a regulated environment, the generation of electricity must meet the expected demand, which is referred to as the Security Constrained Unit Commitment problem (SCUC) [46]. In deregulated environment, GENCOs sell power to ISO and ISO is in charge of meeting the demand. So for GENCOs, they don't have to meet the demand. Instead the price is important for maximizing their profits. This is refered to as the Price Based Unit Commitment problem (PBUC) [16], which is also known as self-scheduling problem, so described in this thesis.

Uncertainty of Parameters Some literature assume that all parameters in UC problem are known. Other literature incorporate the uncertainties of some parameters such as demand and price into the problem formulation and solution. This part will be elaborated and illustrated in 2.2.

2.1.2 Solution Methods

Mathematically speaking, UC problem is a complex problem. It is very hard to get an optimal solution. Obviously enumeration can assure us an optimal solution. But the computational time and space will increase exponentially as the problem size increases [44]. Solution methods proposed in the literature are listed below.

Mixed Integer Programming It is a rigorous method and can reach a better solution than other methods. But it is restricted by the size of the problem. With the advance of computational techniques such as MIP solvers and parallel

computing, it becomes attractive to solve large-scale UC problem by MIP. Ostrowski [23] introduces a tight MIP formulation for UC problem by including a new class of inequalities regarding minimum up- and down-time that tightens the feasible region, improving the solution quality and computational time. Aghaei [1] proposes MIP for generalized hydro-thermal self-scheduling problem consisting of practical constraints such as prohibited zones, and the effectiveness has been demonstrated by case studies. In [32], a modified branch-and-bound method is developed, which branches binary variables based on their difference from bounds, and no decomposition approach is required in the solving procedure. As the model incorporates more constraints or generating units, the solution space will become very complex. In many cases, the combination of decomposition techniques such as Lagrangian relaxation and Benders decomposition is applied.

Lagrangian and Benders Decomposition The adoption of decomposition techniques is based on the structure of the problem. By exploiting UC problem, we can easily find one way of decomposition. All constraints express the characterizations of each individual unit except the demand constraint or some other reserve requirements. Under this case, Lagrangian relaxation (LR) can be applied [41]. The bundle constraint or some other hard constraints, such as the demand constraint, are moved to the objective function associated with a Lagrangian multiplier, which will cause a penalty by the violation of bundle constraints. The problem can be decomposed into subproblems for each individual unit. Given a set of Lagrangian multipliers, we can get a lower bound to the original problem by solving the subproblems. The dual problem obtained from LR is easier to solve than the original UC problem, but the gap between the dual optimal solution and the original optimal solution is a weakness. However, the computational results in many literature show that this method converge rapidly and the solution is satisfactory. Discussions in

the literature also include how to update Lagrangian multipliers, how to solve the Lagrangian function, which is usually a nondifferential function, and how to tackle the duality gap. Subgradient method is adopted to handle the dual of a subproblem in [3]. Lauer [15] tackled the duality gap problem using a constrained Netwon's method. Virmani [41] illustrates the practical aspects of LR for solving thermal UC problem. Frangioni [11] uses LR method to solve large-scale hydro-thermal UC problem with ramp constraints on thermal units. Finardi [9] presented a comparative analysis of dual problems based on LR in the hydro UC.

Another decomposition technique commonly used in large-scale UC problems in recent years is Benders decomposition (BD) [22]. In BD, the original problem is reformulated into a relaxed master problem and a set of subproblems. By decomposing UC problem, the relaxed master problem only involves integer variables which model the unit's on/off status, while subproblems involve continuous variables which model the generating level. The relaxed master problem and suproblems interact iteratively until an optimal solution is found. Because the relaxed subproblem only includes a subset of constraints, subproblems are solved to examine the optimal solution of the relaxed master problem and to find if any constraint is violated. There are two types of constraints added. One is called feasibility cut, while the other is called optimality cut. In each iteration, if suproblems are feasible, the relaxed master problem with less constraints provides a lower bound to the original problem, while the upper bound of original problem is obtained by calculating a new objective function using solutions of that relaxed master problem. Finally the upper bound and lower bound will converge, and the optimal solution of the original problem is obtained. The application of BD in UC has been discussed recently. Shahidehopour [33] reviews the application of BD in power system. A

BD method is extended for two-stage SCUC in [17]. And an improved scheme of DB is proposed for network-constrained UC in [45].

Dynamic Programming DP is not efficient to solve UC problem independently because of the curse of the high dimensionality in UC. It is usually applied with other methods. Hobbs [14] combined priority list with DP. Nowadays, it is mostly used to solve subproblems disaggregated by decomposition techniques such as Lagrangian relaxation, where the subproblem only deals with a single generating unit. Fan [7] proposed an efficient DP algorithm for UC problem considering ramp constraints and time-dependent start-up cost with a piecewise linear function describing production cost. The production cost function is extended to an arbitrary convex function in [10].

Heuristic methods Because of the complexity of UC problem, heuristic methods such as priority list [37] and genetic algorithm [6] have been applied in solving a UC problem. The main drawback of this type of method is that an optimal solution is not guaranteed. Because no heuristic method is involved with this thesis, the application of heuristics in UC problems will not be discussed extensively here.

Summaries of optimization methods applied in UC problem can be found in [31][24][25][47][22].

2.2 Stochastic Unit Commitment Problem

Electric load, or demand, fluctuates over time. For example, on a typical day, the load requirement during the daytime is usually bigger than that at later night. And the transformation of energy market from regulation to deregulation has increased the uncertainty of energy system. More and literature in UC problems include uncertainty of parameters in the model. With the advance of computation techniques, many researchers made contributions to formulating and solving stochastic UC problem.

In 1996, Takriti, Birge, and Long [38] incorporated the uncertainty of demand in the UC problem. This is the first paper that has incorporated the uncertainty in the unit commitment problem. The uncertainty was modeled by a finite number of scenarios, and each scenario is associated with a corresponding probability. Please refer to Figure 1.1 in Section 1.2.2 Stochastic Programming for more details. The Lagrangian relaxation technique was then applied and the problem was decomposed into suproblems of each generating unit. The DP method was used to solve the subproblem. In 2012, Shiina [34] used the same method, the combination of Lagrangian relaxation and DP method, to solve the UC problem with uncertain demand, but developed an algorithm that combined the lambda iteration and golden section to update dual multiplier method. The formulation and solution procedure in this paper is presented as follows. Notations and decision variables used in the formulation are below.

Notations:

T Number of time periods.

I Number of generating units.

S Number of scenarios of demand.

d_t^s Demand at time t under scenario s

P Feasible region constructed by minimum up-time, minimum down-time, capacity, and bundle constraints.

Decision Variables:

u_{it} $u_{it} = 1$, if the unit i is on at time t; otherwise, $u_{it} = 0$.

x_{it}^s The generating amount of the unit i at time t under scenario s.

The formulation of the stochastic UC problem is presented below.

$$\min_{x_{it}^s, u_{it}} \sum_{s=1}^{S} p_s \sum_{t=1}^{T} \sum_{i=1}^{I} \{f_i(x_{it}^s) u_{it} + g_i(u_{i,t-1}, u_{it})\} \tag{2.1}$$

$$\text{s.t.} \sum_{i=1}^{I} x_{it}^s \geq d_t^s, t = 1, \cdots, T, s = 1, \cdots, S \tag{2.2}$$

$$u_{it}, x_{it}^s \in P \tag{2.3}$$

where $f_i(x_{it}^s)u_{it}$ is a fuel cost function, a convex quadratic function of x_{it}^s, and $g_i(u_{i,t-1}, u_{it})$ is a start-up cost function. Let $\lambda_t^s(\geq 0)$ be Lagrange multipliers associated with the demand constraints 2.2. The Lagrangian Relaxation Problem is as follows.

$$L(\lambda) = \min_{x_{it}^s, u_{it}} \sum_{s=1}^{S} p_s \sum_{t=1}^{T} \sum_{i=1}^{I} \{f_i(x_{it}^s) u_{it} + g_i(u_{i,t-1}, u_{it})\} - \sum_{s=1}^{S} \sum_{t=1}^{T} \lambda_t^s (\sum_{i=1}^{I} x_{it}^s - d_t^s) \tag{2.4}$$

$$\text{s.t.} u_{it}, x_{it}^s \in P \tag{2.5}$$

The objective function 2.4 can be written as follows.

$$L(\lambda) = \min \sum_{i=1}^{I} \sum_{t=1}^{T} \sum_{s=1}^{S} [p_s \{f_i(x_{it}^s) u_{it}\} - \lambda_t^s x_{it}^s] + \sum_{i=1}^{I} \sum_{t=1}^{T} g_i(u_{i,t-1}, u_{it}) + \sum_{s=1}^{S} \sum_{t=1}^{T} \lambda_t^s d_t^s \tag{2.6}$$

In 2.6, the term $\sum_{s=1}^{S} \sum_{t=1}^{T} \lambda_t^s d_t^s$ is a constant. So the problem after Lagrangian relaxation can be decomposed into subproblems for each generating unit. And then the problem is solved by DP procedure on the scenario tree. The production level x_{it}^s is obtained first by solving the below problem.

$$\min_{x_{it}^s} \sum_{s'' \in B(s,t)} p_{s''} \{f_i(x_{it}^s)\} - \lambda_t^s x_{it}^s \tag{2.7}$$

$$\text{s.t.} \quad \textit{capacity} \quad \textit{constraint} \tag{2.8}$$

And then on/off status u_{it} are determined by the calculation of DP, which is done by the a series of recursive equations. When solving DP problem by recursive equations, the curse of high dimensionality of DP cannot be avoided. Frangioni [10] presented an efficient dynamic programming (DP) algorithm for solving single deterministic UC problem with ramp constraints and arbitrary convex cost function. That algorithm is one of the most efficient ways of solving this kind of problem. We extended that algorithm to solve the subproblems for each generating unit under the uncertain price. Basically the definitions of stage and state are different. In [38] and [34], each time instance in the planning horizon is a stage, and the number of potential states of a stage is dependent on minimum up-time and down-time. For example, there are 24 time periods in the planning horizon, and the minimum up- and down-time is 6, 4, respectively. Based on the DP method in these papers, there are 24 stages and each stage has 10 potential states. A recursive equation is then defined to solve this problem. However, the stage of the DP method presented in this thesis is a valid up-time interval that satisfies minimum-up time. So there is only one state at each stage, which is "on" during that time interval. It is easier to implement and more efficient, because the problem is transformed to compute the shortest path in a directed acyclic network, avoiding the curse of high dimensionality of DP . The stage of the DP method presented in this thesis is a valid up-time interval that satisfies minimum-up time. So there is only one state at each stage, which is "on" during that time interval. It is easier to implement and more efficient, because the problem is transformed to compute the shortest path in a directed acyclic network, avoiding the curse of high dimensionality of DP .

In addition to the deregulation of energy market, as the renewable resources are integrated into the generating systems, the uncertainty of electricity generating system has been more broadly considered. The availability of renewable resources such as wind and solar is controlled by weather conditions to a significant extent. Wang [42] and Tuohy [40] studied wind power forecasting uncertainty and demonstrated the effectiveness of stochastic modeling in incorporating wind power.

A detailed literature review for large-scale unit commitment under uncertainty can be found in [22].

2.3 Stochastic Self-scheduling Problem

As mentioned earlier, the self-scheduling UC problem is a variant of UC problems. The main difference between self-scheduling problem and UC problem is that self-scheduling problem is to find an optimal schedule at the standpoint of the GENCOs in order to maximize their profits [27]. In deregulated environment, the GENCOs decide generating schedules by themselves, which is what the self-scheduling means. In addition to constraints in UC problems, self-scheduling problem may also consider the interaction between GENCO and the rest of the system and include conditions particular to a GENCO, such as risk management and bilateral agreements, which makes the model very complex , but it is reasonable to only consider the simplest case by assuming that the GENCO is the price taker when GENCO is small and its behavior doesn't influence the market price [29]. In many literature [36] [12], self-scheduling problem is used by a GENCO that may own one or more generating units. In this thesis, under the assumption that GENCO is the price taker and the restriction of one generating unit, we present a stochastic model for self-scheduling problem, considering the fact of auction when GENCOs sell electricity to ISO, which makes price uncertain.

2.4 Review Summary

UC problem is to find an optimal production schedule of available generating units subject to a series of operational constraints. Because of varieties of characterizations of electricity system worldwide, UC problem is not a well-defined problem. People incorporate different considerations in the formulation of UC problem in different situations. Among all variants, the self-scheduling UC problem is one that addresses

the profit of GENCOs. The elements considered in the model in this thesis is listed in Table 2.1.

Table 2.1: Elements considered in the Model

No. of Unit	Planning Horizon	Energy Resources	Technical Constraints	Demand Satisfaction	Uncertain Parameter
1	Short-term	Thermal	General Capacity	No	Price
			Minimum up-time		
			Minimum down-time		
			Maximum ramp-up rate		
			Maximum ramp-down rate		
			Time dependent start-up cost		

The complexity of operational restrictions leads UC problem to a large-scale and non-convex problem. Driven by the real-time operation and economic interest, many researchers have devoted to solving UC problem faster. The optimization techniques have been broadly used to tackle UC problem, especially methods for large-scale problems, such as mixed integer programming, Lagrangian relaxation, Bender's decomposition, and heuristics.

The incorporation of uncertainty in the model makes the size of the problem much larger. Researchers usually use a finite number of scenarios to formulate the uncertainty of such elements as demand and price in the model. The literature shows that Lagrangian relaxation is a popular way of solving the problem. The hard constraints such as demand and nonanticitipativity constraints are moved to the objective and then the dual problem can be decomposed by generating units. Furthermore, the DP method is a favorable way of solving the subproblem for each generating unit resulting from relaxation and decomposition. We present a different DP procedure to solve such a subproblem, which transforms to compute a shortest path in a directed acyclic network, avoiding the curse of high dimensionality, which may result from the DP methods that are used in papers that solve stochastic UC problems.

Chapter 3

Problem Solving

3.1 Problem Formulation

The stochastic self-scheduling UC problem in this thesis is described as follows. A thermal generating unit is given. We want to generate an optimal schedule in planning horizon T in order to maximize the profit resulted from selling generated electricity. The characterization of the unit is that it has maximum output level $\widehat{P}$ and minimum output level $\underline{P}$, which is its capacity. There are two types of cost associated with the electricity generation process. One is generation cost, while the other is start-up cost. The generating cost is formulated by a quadratic function in production level. The start-up cost depends on how long the unit has been off before it is turned on. The start-up cost is formulated by a piece-wise constant function. When the down-time periods surpass the cold-start moment $\widehat{TC}$, it will be cold start-up cost C; otherwise, it is warm start-up cost W (see Figure 3.1). The revenue function is the product of production level and unit price. The price r is uncertain and treated as a discrete random variable. It can be represented by a set of scenarios S with a corresponding probability. The price in scenario s at time t is denoted by r_t^s, where $s \in S, t \in T$. During the time periods when the unit is on, beside the capacity restriction, because of mechanical inertia, the maximum increase from a time instant to the next cannot

exceed the maximum ramp-up rate $\widehat{RU}$, and the maximum decrease from a time instant to the next cannot exceed the maximum ramp-down rate $\widehat{RD}$. At the time instant when the generating unit is turned on, the maximum start-up output level is $\widehat{SU}$ at that time instant. At the time instant when the generating unit is turned off, the maximum shut-down output level is $\widehat{SD}$ at that time instant. Because mechanical restrictions, the unit can only be turned on at least after a minimum down-time period $\widehat{DT}$ once it is turned off, and the unit can only be turned off at least after a minimum up-time $\widehat{UT}$ once it is turned on. Assume that the unit is available over the whole planning horizon. The objective is to find an optimal schedule that maximizes the profit of selling electricity generated by the unit. The schedule should include not only the on/off decisions of the unit at each time instant but also the production level of the unit at time periods when the unit is on.

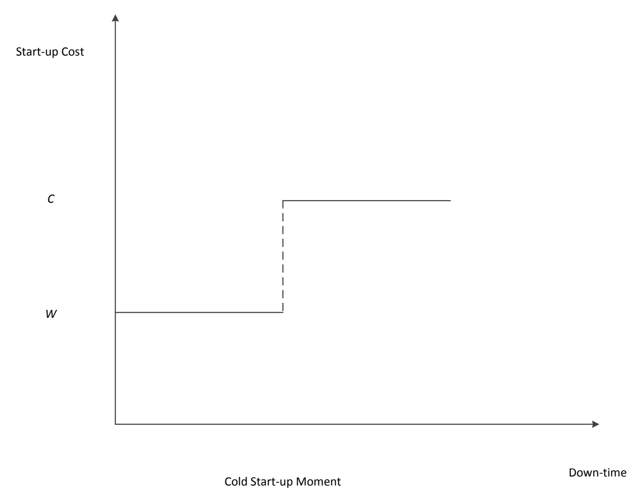


Figure 3.1: Start-up Cost

3.2 Scenario Generation

As it is mentioned in Section 3.1, the uncertainty of price is formulated by a finite number of scenarios. Scenarios are generated based on a scenario tree with a given initial price, where the price at the next time instant can go up or down by a certain percentage (e.g. 10%) with the same probability. The implementation is listed below. We first use the random number generator rand() to generate a random matrix with entries 0 or 1. If a[i][j] is 0, the price will go down by 10%. If a[i][j] is 1, the price will go up by 10%.

```
int a[Scenario][Time]; //Declare random matrix
int r[Scenario][Time]; //Declare price matrix
srand(1); // Change seed of random number generator
for (int i=0; i < Scenario; i++)
  for (int j=0; j < Time; j++)
    a[i][j] = rand() % 2; //Generate 0 or 1
for (int i = 0; i < Scenario; i++)
  r[i][0] = 104;
for (int i=0; i < Scenario; i++)
  for (int j=1;j < Time; j++)
   r[i][j] = r[i][j-1] - 0.1*r[i][j-1] + 0.2*a[i][j]*r[i][j-1];
```

In this case, the probability of each scenario $\pi_s = \frac{1}{|S|}, s \in S$.

3.3 Mixed Integer Quadratic Programming Procedure

As known, a good formulation is indispensable for solving a problem successfully and efficiently. Ostrowski [23] presented a tight MIP formulation for the UC problem, which is one of the most computationally efficient formulations. It is not only used

for UC problem but also used for self-scheduling problem. To formulate the stochastic self-scheduling problem in this thesis, we also adopt three binary variables to model the status of the unit same as those presented in [23].

Decision Variables Four decision variables are introduced:

- p_t^s Electricity level produced at time t in scenario s, $t \in T, s \in S$.
- c_t Start-up cost of the unit at time t, $t \in T$.
- v_t On/off status of the unit at time t, $t \in T$. If the unit is on at time t, $v_t = 1$; otherwise, $v_t = 0$.
- y_t Start-up status of the unit at time t, $t \in T$. If the unit is turned on at time t, $y_t = 1$; otherwise, $y_t = 0$.
- z_t Shutdown status of the unit at time t, $t \in T$. If the unit is turned off at time t, $z_t = 1$; otherwise, $z_t = 0$.

Based on the problem statement in Section 3.1, we come up with the following mathematical model. The illustration of this model is also presented as follows.

$$\max_{p_t^s, c_t, v_t, y_t, z_t} \sum_{s \in S} \pi_s \sum_{t \in T} [r_t^s p_t^s - c_t - c(p_t^s)] \tag{3.1}$$

$$\text{s.t.} \tag{3.2}$$

$$c_t \geq C y_t - \sum\nolimits_{i=t-\widehat{TC}+1, i \geq 1}^{t} (C - W) z_i, \quad \forall t \in T \tag{3.3}$$

$$\sum\nolimits_{i=t-\underline{UT}+1, i \geq 1}^{t} y_i \leq v_t, \quad \forall t \in T \tag{3.4}$$

$$v_t + \sum\nolimits_{i=t-\underline{DT}+1, i \geq 1}^{t} z_i \leq 1, \quad \forall t \in T \tag{3.5}$$

$$\underline{P} v_t \leq p_t^s \leq \widehat{P} v_t, \quad \forall t \in T \quad \forall s \in S \tag{3.6}$$

$$p_{t+1}^s \leq p_t^s + \widehat{RU} v_t + \widehat{SU} y_{t+1}, \quad t \in [1, |T| - 1] \quad \forall s \in S \tag{3.7}$$

$$p_t^s \leq p_{t+1}^s + \widehat{RD} v_{t+1} + \widehat{SD} z_{t+1}, \quad t \in [1, |T| - 1] \quad \forall s \in S \tag{3.8}$$

$$v_t - v_{t+1} + y_{t+1} - z_{t+1} = 0, \quad t \in [1, |T| - 1] \tag{3.9}$$

$$B(s_1, t) = B(s_2, t) \Rightarrow p_t^{s_1} = p_t^{s_2}, \quad \forall (s_1, s_2) \in S^2 \quad \forall t \in T \tag{3.10}$$

Objective Function The revenue is denoted by $r_t^s p_t^s$. When the unit is turned on, it causes start-up cost c_t, which is constrained in 3.3. As long as the unit is in the committed status, it generates a generating cost $c(p_t^s)$. The generation cost in the objective is the expected value of all scenarios since the price is formulated by a finite number of scenarios. The objective is to maximize the profit.

Start-up Cost Constraint Constraint 3.3 models the start-up cost presented in Figure 3.1. If the down-time surpass $\widehat{TC}$, the start-up cost is cold start-up cost C when the unit is turned on; otherwise, the start-up cost is warm start-up cost W.

Minimum up- and down-time Constraints Constraint 3.4 models minimum up-time. If the the unit is turned on at time instant t, it must remain committed until $t + \widehat{UT} - 1$. Constraint 3.5 model minimum down-time. If the unit is turned off at time instant t, it must remain uncommitted until $t + \widehat{DT} - 1$. Note that these constraints must be restricted by the logical constraint 3.9.

Capacity Constraint Constraint 3.6 expresses the maximum and minimum output the unit can generate when it is committed.

Ramping Constraints Constraint 3.7 and 3.8 illustrate maximum ramp-up rate, maximum ramp-down rate, maximum start-up rate, and maximum shut-down rate. If the unit is committed at time instant t and $t+1$ and the the production level at time t is p, then the production level at time $t+1$ should fall in $[p + \widehat{RU}, p - \widehat{RD}]$. If the unit is uncommitted at time t and eligibly turned on at time $t+1$, the production level at time $t+1$ should be no more than $\widehat{SU}$. If the unit is committed at time t and eligibly turned down at time $t+1$, the production level at time t should be no more than $\widehat{SD}$.

Logical Constraint Constraint 3.9 formulates the relationship among three binary variables. This ensures y_t and z_t to take appropriate values when the unit changes its status, either turned on or turned off.

Nonanticipativity Constraint Since the uncertainty of price is represented by a set of scenarios, constraint 3.10 is included to make scenarios applicable. If two scenarios s_1 and s_2 are indistinguishiable at time t, then the decision made for these two scenarios must be the same at time t, namely $p_t^{s_1} = p_t^{s_2}$ in this problem, where $t \in T$ and $s_1, s_2 \in S$.

3.4 Dynamic Programming Procedure

The basic idea of DP method is to construct nodes and arcs of a network based on minimum up-time $\underline{UT}$ and minimum down-time $\underline{DT}$ and add costs and profits as weights of nodes or arcs. Every path from the source to the sink will be a feasible solution to the problem, the objective is to minimize the cost. After the network is constructed, this problem is reduced to compute the shortest path on a directed acyclic network. The detailed procedure is presented below with a flow chart of the extended algorithm in Figure 3.2.

1. Node Construction

   (a) Find all time intervals that satisfy the minimum up-time constraint and represent them by nodes. So each qualified on-time interval is denoted by a node. Moreover, add a source node s and a sink node d. The time interval $[a, b]$ denoted by each node means that the unit is turned on at time instant a, and turned off at time instant b. Note that the unit is uncommitted at time instant $a - 1$ and $b + 1$, where $a,\ b \in T, a \leq b$.

   (b) Find time instances t satisfying $t + \underline{UT} > T$ and represent time intervals $[t, T]$ as nodes. For example, given $T = 24, \underline{UT}$, time intervals $[14, 24], [15, 24], [16, 24], [17, 24], [18, 24], [19, 24], [20, 24], [21, 24], [22, 24], [23, 24]$ are also eligible.

   (c) Add corresponding weights to nodes. For each node denoting time interval $[a, b]$, its weight represents the minimal cost generated by the unit in that

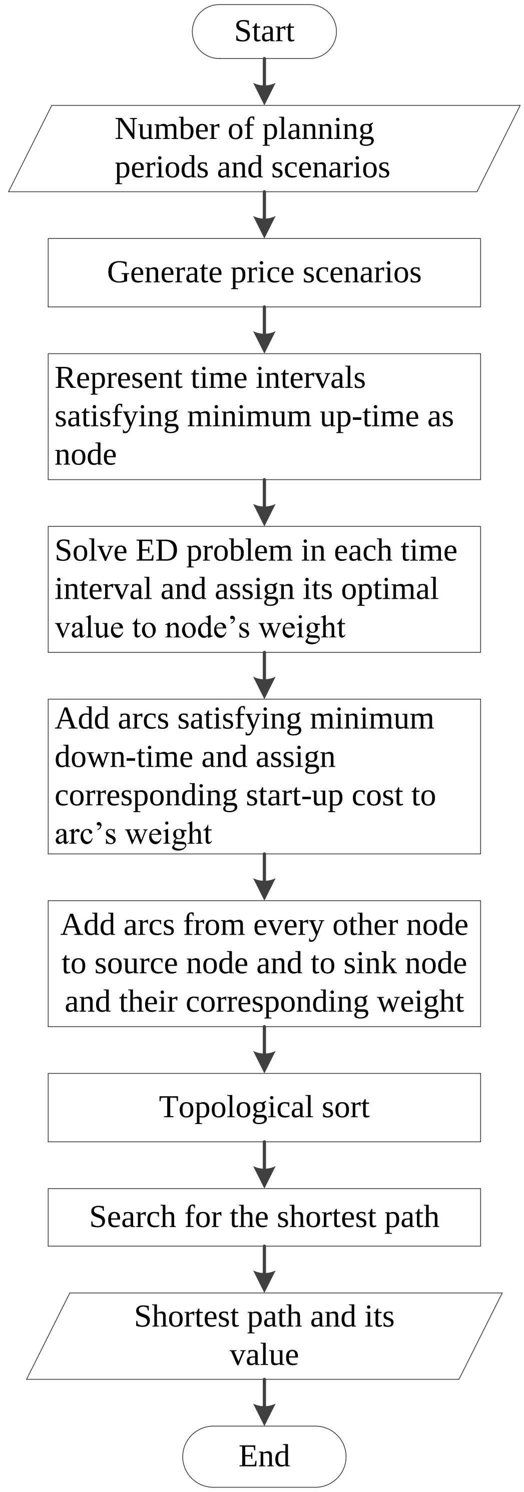


Figure 3.2: Flow Chart of Dynamic Programming Procedure

time interval, which can be obtained by solving the economic dispatch (ED) problem. The ED problem for time interval $[a, b]$ is formulated as follows.

$$\min \sum_{s \in S} \pi_s \sum_{t \in T} [c(p_t^s) - r_t^s p_t^s] \tag{3.11}$$

$$\text{s.t.} \tag{3.12}$$

$$\underline{P} \leq p_t^s \leq \widehat{P}, \quad t \in [a, b] \quad \forall s \in S \tag{3.13}$$

$$p_{t+1}^s \leq p_t^s + \widehat{RU}, \quad t \in [a, b-1] \quad \forall s \in S \tag{3.14}$$

$$p_t^s \leq p_{t+1}^s + \widehat{RD}, \quad t \in [a, b-1] \quad \forall s \in S \tag{3.15}$$

$$p_a^s \leq \widehat{SU}, \quad \forall s \in S \tag{3.16}$$

$$p_b^s \leq \widehat{SD}, \quad \forall s \in S \tag{3.17}$$

$$B(s_1, t) = B(s_2, t) \Rightarrow p_t^{s_1} = p_t^{s_2}, \quad t \in [a, b] \quad \forall (s_1, s_2) \in S^2 \tag{3.18}$$

Notice the difference between ED and UC problem. ED problem only decides the production level given a time interval when the unit is known for sure to be committed. UC problem not only decides the production level but also the on/off status of the unit at each time instance over a planning horizon.

2. Arc Construction

   (a) Add arcs between nodes where the minimum down-time is satisfied. For example, an arc is added from node $[a, b]$ to node $[c, d]$, if $c - b + 1 \geq \widehat{DT}$.

   (b) Add corresponding weight to arcs added above. Each arc is associated with start-up cost caused by turning on the unit at time instant t. The start-up cost depends on how long it has been uncommitted before it is turned on. For example, the unit is turned off at time instant b before it is turned on at time t, the start-up cost is C when $t - b \geq \widehat{TC}$; otherwise, the start-up cost is W.

(c) Add arcs from the source node s to every other node and arcs' corresponding weights, either corresponding time-dependent start-up cost or zero cost, depending on the time periods when the unit has been uncommitted before it is turned on. Take the arc from s to $[a, b]$ as the example. The start-up cost is C when $a \geq \widehat{TC}$; otherwise, the start-up cost is W.

(d) Add arcs from every other node to the sink node d with zero weight.

(e) Transform the weights of nodes to their adjacent arcs.

3. Shortest-path Computation

   Every $s - d$ path in the network represents a feasible solution. And the cost of the path is the negative objective value of 3.1. Note that the minimal cost is equal to the maximal profit. The network is traversed in its topological order when searching the shortest path. The shortest path algorithm for directed acyclic graph is adopted in the code.

4. A Concrete Example of Network Construction

   Assume $T = 6$, $\underline{UT} = 3$, $\underline{DT} = 1$, $\widehat{TC} = 3$. The resulting graph is in Figure 3.3.

   (a) Node Construction

   Based on $\underline{UT}$, we represent valid up-time intervals as nodes. Notice that [4,5] doesn't satisfy the minimum up-time but is also eligible. And then we add a source and sink node. The weights of these nodes can be obtained by solving corresponding ED problems.

   (b) Arc Construction

   Based on $\underline{DT}$, we can add an arc from [0,2] to [4,5]. We also need to add an arc from the source to every other node, and from every other node to the sink. Because $\widehat{TC} = 3$, the weight on the arc from the source to [4,5] is the cold start-up cost, and the weights on the arcs from the source to

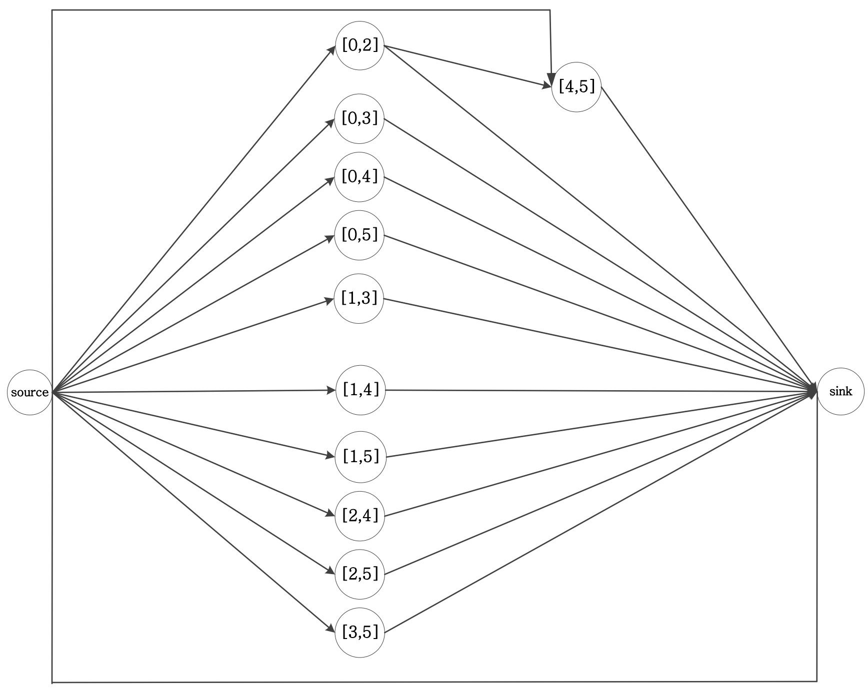


Figure 3.3: A Concrete Example of Network Construction

other nodes are the warm start-up cost. And the weight on the arcs from every other node to the sink are 0s. .

3.5 Computational Experiments

To test the solution time and solution quality of the proposed method, we design two experiments. The design of experiments and analysis of results of experiments are presented below. All experiments are solved with Gurobi 5.6.0 on a server with 2 octacore xeon processors with 256gb of ram. The code is compiled with g++ version 4.7.2.

To test the time efficiency of proposed DP method, we do the following experiment.

- Solve 10-scenario, 100-scenario, and 1000-scenario problems each under 24, 48, 72 time periods, using two methods presented in the thesis. Compute each instance 10 times by changing initial price and seed. The initial price for each instance is given in Appendix. Given an initial price, the instance is computed two times using seed 1 and 2. The parameter values set-up is listed in Table 3.1. The initial price for each instance is listed in Table 3.2.

Table 3.1: Parameter Values

$\underline{DT}$	Minimum down-time	6
$\underline{UT}$	Minimum up-time	12
$\widehat{P}$	Maximum power output of a unit	250
$\underline{P}$	Minimum power output of a unit	100
$\widehat{RD}$	Maximum ramp-down rate of a unit	15
$\widehat{RU}$	Maximum ramp-up rate of a unit	15
$\widehat{SU}$	Maximum shutdown rate of a unit	100
$\widehat{SD}$	Maximum startup rate of a unit	100
C	Cold start-up cost	40
W	Warm start-up cost	20
$\widehat{TC}$	Cold start hours	3

Table 3.2: Initial Price

Periods	Scenarios	Initial Price
24	10	90, 100, 110, 150, 200
24	100	90, 100, 113, 115, 150
24	1000	90, 100, 113, 150, 200
48	10	90, 100, 110, 150, 200
48	100	90, 100, 110, 150, 200
48	1000	90, 100, 110, 150, 200
72	10	90, 100, 110, 150, 200
72	100	90, 100, 110, 150, 200
72	1000	90, 118, 120, 150, 200

The comparison of computational results of those two methods is presented in Table 3.3(in seconds).

Table 3.3: Computational Results of the DP vs. Gurobi solver

Instance		DP			Gurobi		
Periods	Scenarios	time	std. dev.	Gap(%)	time	std.dev.	Gap(%)
24	10	0.92	0.05	0	0.67	0.25	0
	100	4.92	0.08	0	6.78	1.23	0
	1000	156.654	2.22	0	1424.86	159.76	0
48	10	5.32	0.11	0	2.38	0.74	0
	100	42.53	1.91	0	41.98	34.52	0
	1000	1185.993	20.82	0	4200		[0,1253.94]
72	10	20.74	1.81	0	9.758	10.52	0
	100	155.45	8.77	0	529.387	384.12	0
	1000	3818.05	19.23	0	4200		[3.75,1194.40]

From Table 3.3, we can see that for 1000-scenario problems, DP method is more efficient than Gurobi MIQP solver both in solution time and solution quality, because Gurobi MIQP solver cannot always give an optimal solution within time limit, as shown in the column gap, representing the gap from the current solution to the optimal solution. We can also notice that for 10-scenario problems, the average solution time of the Gurobi MIQP solver is smaller than that of DP method, while for 100-scenario problems, the average solution time of DP method is mostly better than that of Gurobi MIQP solver. However, the standard deviation of solution time of Gurobi MIQP solver is rather large, as well as the gap range when using Gurobi MIQP solver to solve 1000-scenario problems under 48 and 72 hours. That means Gurobi MIQP solver solves some instances relatively quickly, while it solves other instances rather slowly, mainly depending on optimal solution situations of the problem. Gurobi solves the problems with the optimal solution that keeps the generating unit committed over the whole planning horizon much faster than solving problems with other optimal solution situations and also solves the problems with the optimal solution that keeps the generating unit uncommitted over the whole planning

horizon faster in some situations (Gurobi converges very slowly in some problems with the latter kind of optimal solution), which results in the big standard deviation and large gap range. The solution time of DP method and Gurobi for 100-scenarios problems under 72 hours is presented in Table 3.4 in seconds. And the solution time and gap range of DP method and Gurobi for 1000-scenario problems under 48 and 72 hours are presented in Table 3.5 in seconds. In these two tables, *Null* means the unit will be off over the whole planning horizon, *All* means the unit will be on over the whole planning horizon, and the *Solution* means optimal on-time intervals.

Table 3.4: Solution Time of 100-scenario Problems under 72 Hours

Periods	Initial Price	DP Time	Gurobi Time	Solution
72	90	158.87	409	Null
		169.17	407.5	Null
	118	164.22	657.12	[0.43]
		164.11	679.92	[0,27]
	120	154.32	1137.44	[0,49]
		141.66	1135.31	[0,28]
	150	146.72	404.36	All
		147.47	392.88	All
	200	153.62	32.77	All
		159.41	37.57	All

Table 3.5: Solution Time and Gap Range of 1000-scenario Problems

Periods	Initial Price	DP Time	Gurobi Time	Gurobi Gap (%)	Solution
48	110	1167.4	4200	1253.9	[0, 15]
	200	1215.8	4200	0	All
72	118	3800.6	4200	1194.40	[0,45]
	200	3811.4	4200	3.75	All

Based on the analysis of results in Table 3.3, we can conclude that DP has great advantage when solving a large-scale stochastic self-scheduling problem with a large number of scenarios. A simulation experiment is conduction to show that the solution

based on the model with more scenarios can yield better objective value, which is revenue, in average.

- Solve 10-scenario and 100-scenario problems, substitute their optimal first-stage solutions to 1000-scenarios problems, and solve 1000-scenario problems, respectively. Use the same seed and all the same parameters for 10-scenario problem, 100-scenario problem, and 1000-scenario problem in each run. Run the experiment 100 times by using the seed $1, 2, \cdots, 100$, respectively. Compare the expectations of objective values of 1000-scenario problems based on optimal solutions of 10-scenario and 100-scenario problems.

Of these 100 replications, the expectation ($E1$) of objective values of 1000-scenario problems based on optimal first-stage solutions of 10-scenario problems is -1883.45, while the expectation ($E2$) of objective values of 1000-scenario problems based on optimal first-stage solutions of 100-scenario problems is -622.38. We also observe that 76% of objective values of 1000-scenario problems based on optimal first-stage solutions of 100-scenario problems are larger than or the same as objective values of 1000-scenario problems based on optimal first-stage solutions of 10-scenario problems. Back to the meaning of objective value, which is revenue, we can conclude that decisions based on 100-scenario problems can bring more profit or less loss than decisions based on 10-scenario problems in average.

Besides the above main results, we also find that the computational time of DP procedure is greatly dependent on the time spent on solving the economic dispatch problems of time intervals in the network, in other words, the weights of nodes during the network construction. The computational time of shortest path in a network is closely associated with the number of nodes and paths. And in this problem, the number of nodes depends on the minimum up-time and down-time rate of the generating unit, because the minimum up-time rate decides the number of nodes and the minimum down-time rate decide the number of arcs. So the computational time of DP procedure is very problem-specific.

Chapter 4

Conclusions and Future Work

The self-scheduling UC problem is a typical application of optimization in the area of power generation. The problem is to find an optimal schedule of generating units in the planning horizon in order to maximize the profit of GENCO subject to a series of operation restrictions. The uncertainty parameters such as price and demand have become critical, because of the deregulation of energy market and incorporation of renewable energy resources.

A stochastic model for the self-scheduling UC problem is presented in this thesis. The model considers a single thermal generating unit with certain characteristics. The price uncertainty is represented by a finite number of scenarios. The objective is to maximize the revenue over a planning horizon. The production cost is formulated as a quadratic function. Typical constraints of thermal generating units such as capacity, minimum up- and down-time, maximum ramp up- and down-rate, and time dependent start-up cost, are considered into the model.

An efficient dynamic programming algorithm presented by Frangioni for solving nonlinear single-unit commitment problems with ramping constraints is extended to solve the self-scheduling problem. Computational results show that this algorithm is more efficient than Gurobi MIQP solver 5.6.0 when there are a large number of scenarios and time periods in the planning horizon. Furthermore, the results of

the simulation experiment show that decisions based on more scenarios can yield more average profit or less average loss. Hence the work in this thesis caters to the deregulation of energy market and has significant economic significance for GENCO.

The proposed approach can also be used to solve the subproblem of a multiple unit commitment problem with uncertain demand, which can be decomposed into subproblems for each generating unit after Lagrangian relaxation. We can update the dual multiplier by updating the price. The resulting model will help reduce the total production cost and ultimately the consumption of natural resources in a whole, which have been demonstrated by many references, such as [38], [39], [35], and [34].

Self-scheduling or UC problem is a very complex problem, even not well-defined, because of complexities of electricity operations. We only considered a simplified case of stochastic self-scheduling problem in this thesis. In the future, the model can be improved in the following aspects. And the DP algorithm presented in this thesis should be able to be adapted to improved models correspondingly.

- Optimize the code. The performance of an algorithm doesn't only depend on on the design of algorithm itself, but also the data structures used to implement that algorithm. The computation time varies with the usages of different data structures.

- Revoke some assumptions and adapt the model to more real operation context. We have made following assumptions for simplifying the model.

  - The company is the price taker, so the price will not be influenced by the market.

  - The generating unit has been off long enough, so it can be turned on at the very beginning of planning horizon.

- Incorporate multiple generating units into the model.

- Integrate renewable resources such as hydro units and wind into the model.

- More research are needed to construct scenarios that reflects real price.

Bibliography

[1] Aghaei, J., Ahmadi, A., Shayanfar, H., and Rabiee, A. (2012). Mixed integer programming of generalized hydro-thermal self-scheduling of generating units. *Electrical Engineering*, 95(2):109–125. 17

[2] Ahuja, R., Magnanti, T., and Orlin, J. (1993). *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc. 6, 8

[3] Bard, J. F. (1988). Short-term scheduling of thermal-electric generators using Lagrangian Relaxation. *Operations Research*, 36(5):756–766. 18

[4] Birge, J. and Louveaux, F. (1997). *Introduction to Stochastic Programming*. Springer Veriag. 3

[5] Carrion, M. and Arroyo, J. (2006). A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. *Power Systems, IEEE Transactions on*, 21(3):1371–1378. 15

[6] Damousis, I., Bakirtzis, A., and Dokopoulos, P. (2004). A solution to the unit-commitment problem using integer-coded genetic algorithm. *Power Systems, IEEE Transactions on*, 19(2):1165–1172. 19

[7] Fan, W., Guan, X., and Zhai, Q. (2002). A new method for unit commitment with ramping constraints. *Electric Power Systems Research*, 62(3):215–224. 19

[8] Finardi, E. C. and Da Silva, E. L. (2006). Solving the hydro unit commitment problem via dual decomposition and sequential quadratic programming. *Power Systems, IEEE Transactions on*, 21(2):835–844. 15

[9] Finardi, E. C. and Scuzziato, M. R. (2014). A comparative analysis of different dual problems in the Lagrangian Relaxation context for solving the Hydro Unit Commitment problem . *Electric Power Systems Research*, 107(0):221 – 229. 18

[10] Frangioni, A. and Gentile, C. (2006). Solving nonlinear single-unit commitment problems with ramping constraints. *Operations Research*, 54(4):767–775. 12, 15, 19, 22

[11] Frangioni, A., Gentile, C., and Lacalandra, F. (2008). Solving unit commitment problems with general ramp constraints. *International Journal of Electrical Power & Energy Systems*, 30(5):316 – 326. 15, 18

[12] Haghighat, H., Seifi, H., and Kian, A. R. (2008). On the self-scheduling of a power producer in uncertain trading environments. *Electric Power Systems Research*, 78(3):311–317. 23

[13] Hanscom, M. A., Lafond, L., Lasdon, L., and Pronovost, G. (1980). Modeling and resolution of the medium term energy generation planning problem for a large hydro-electric system. *Management Science*, 26(7):659–668. 15

[14] Hobbs, W., Hermon, G., Warner, S., and Shelbe, G. (1988). An enhanced dynamic programming approach for unit commitment. *Power Systems, IEEE Transactions on*, 3(3):1201–1205. 19

[15] Lauer, G., Sandell, N., Bertsekas, D., and Posbergh, T. (1982). Solution of large-scale optimal unit commitment problems. *Power Apparatus and Systems, IEEE Transactions on*, PAS-101(1):79–86. 18

[16] Li, T. and Shahidehpour, M. (2005). Price-based unit commitment: a case of Lagrangian Relaxation versus mixed integer programming. *Power Systems, IEEE Transactions on*, 20(4):2015–2025. 16

[17] Liu, C., Shahidehpour, M., and Wu, L. (2010). Extended Benders decomposition for two-stage SCUC. *Power Systems, IEEE Transactions on*, 25(2):1192–1194. 19

[18] Meza, J. L. C., Yildirim, M. B., and Masud, A. S. (2007). A model for the multiperiod multiobjective power generation expansion problem. *Power Systems, IEEE Transactions on*, 22(2):871–878. 15

[19] Nemhauser, G. and Wolsey, L. (1999). *Integer and Combinatorial Optimization*. John Wiley & Sons, Inc. 2

[20] Ni, E., Luh, P., and Rourke, S. (2004). Optimal integrated generation bidding and scheduling with risk management under a deregulated power market. *Power Systems, IEEE Transactions on*, 19(1):600–609. 1

[21] Niknam, T., Khodaei, A., and Fallahi, F. (2009). A new decomposition approach for the thermal unit commitment problem. *Applied Energy*, 86(9):1667–1674. 15

[22] Osiris, E. (2014). Large-scale unit commitment under uncertainty: a literature survey. 18, 19, 23

[23] Ostrowski, J., Anjos, M., and Vannelli, A. (2012). Tight mixed integer linear programming formulations for the unit commitment problem. *Power Systems, IEEE Transactions on*, 27(1):39–46. 15, 17, 27, 28

[24] Padhy, N. P. (2003). Unit commitment problem under deregulated environment-a review. In *Power Engineering Society General Meeting, 2003, IEEE*, volume 2. IEEE. 19

[25] Padhy, N. P. (2004). Unit commitment-a bibliographical survey. *Power Systems, IEEE Transactions on*, 19(2):1196–1205. 19

[26] Pang, C. and Chen, H. (1976). Optimal short-term thermal unit commitment. *Power Apparatus and Systems, IEEE Transactions on*, 95(4):1336–1346. 15

[27] Pindoriya, N., Singh, S. N., et al. (2009). Day-ahead self-scheduling of thermal generator in competitive electricity market using hybrid pso. In *Intelligent System Applications to Power Systems, 2009. ISAP'09. 15th International Conference on*, pages 1–6. IEEE. 23

[28] Powell, W. B. (2010). *Approximate Dynamic Programming: Sloving the Curse of Dimensionality*. John Wiley & Sons, Inc. 12

[29] Prina, J. (2012). Stochastic unit commitment and self-scheduling: A review considering co_2 emission modeling. In *Handbook of CO_2 in Power Systems*, pages 311–326. Springer. 15, 23

[30] Rajan, A. and Christober, C. (2011). Hydro-thermal unit commitment problem using simulated annealing embedded evolutionary programming approach. *International Journal of Electrical Power & Energy Systems*, 33(4):939–946. 15

[31] Saravanan, B., Das, S., Sikri, S., and Kothari, D. (2013). A solution to the unit commitment problem–a review. *Frontiers in Energy*, 7(2):223–236. 19

[32] Shafie-Khah, M., Moghaddam, M. and Sheikh-El-Eslami, M.K. (2011). Unified solution of a non-convex SCUC problem using combination of modified branch-and-bound method with quadratic programming. *Energy Conversion and Management*, 52:3425–3432. 17

[33] Shahidehopour, M. and Fu, Y. (2005). Benders decomposition: applying Benders decomposition to power systems. *Power and Energy Magazine, IEEE*, 3(2):20–21. 18

[34] Shiina, T. (2012). Unit commitment problem with stochastic demand. *Journal of Computations & Modelling*, 2(2):35–53. 20, 22, 40

[35] Shiina, T. and Birge, J. R. (2004). Stochastic unit commitment problem. *International Transactions in Operational Research*, 11(1):19–32. 40

[36] Shrestha, G., Kai, S., and Goel, L. (2004). An efficient stochastic self-scheduling technique for power producers in the deregulated power market. *Electric Power Systems Research*, 71(1):91–98. 23

[37] Srinivasan, D. and Chazelas, J. (2004). A priority list-based evolutionary algorithm to solve large scale unit commitment problem. In *Power System Technology, 2004. PowerCon 2004. 2004 International Conference on*, volume 2, pages 1746–1751. 19

[38] Takriti, S., Birge, J. R., and Long, E. (1996). A stochastic model for the unit commitment problem. *Power Systems, IEEE Transactions on*, 11(3):1497–1508. 20, 22, 40

[39] Takriti, S., Krasenbrink, B., and Wu, L. S.-Y. (2000). Incorporating fuel constraints and electricity spot prices into the stochastic unit commitment problem. *Operations Research*, 48(2):268–280. 40

[40] Tuohy, A., Meibom, P., Denny, E., and O'Malley, M. (2009). Unit commitment for systems with significant wind penetration. *Power Systems, IEEE Transactions on*, 24(2):592–601. 15, 22

[41] Virmani, S., Adrian, E. C., Imhof, K., and Mukherjee, S. (1989). Implementation of a Lagrangian Relaxation based unit commitment problem. *Power Systems, IEEE Transactions on*, 4(4):1373–1380. 17, 18

[42] Wang, Q., Guan, Y., and Wang, J. (2012). A chance-constrained two-stage stochastic program for unit commitment with uncertain wind power output. *Power Systems, IEEE Transactions on*, 27(1):206–215. 15, 22

[43] Winston, W. L. (2003). *Operations Research: Applications and Algorithms*. Cengage Learning. 12

[44] Wood, A. J., Wollenberg, B. F., and Sheblé, G. B. (2013). *Power Generation, Operation, and Control*. John Wiley & Sons, Inc. 16

[45] Wu, L. (2013). An improved decomposition framework for accelerating lsf and bd based methods for network-constrained uc problems. *Power Systems, IEEE Transactions on*, 28(4):3977–3986. 19

[46] Wu, L., Shahidehpour, M., and Li, T. (2007). Stochastic security-constrained unit commitment. *Power Systems, IEEE Transactions on*, 22(2):800–811. 15, 16

[47] Yamin, H. (2004). Review on methods of generation scheduling in electric power systems. *Electric Power Systems Research*, 69(2):227–248. 19

Vita

Lili Zhang was born in Shandong province, China. She graduated from Central South University in 2011 with bachelor degree in Electronic Information Science and Technology. After that, Lili found her passion in Operations Research. She officially started her graduate study in the Department of Industrial and Systems Engineering at the University of Tennessee, Knoxville in 2013 spring. Because of excellent academic performance, Lili is invited to be a member of the honor society of Phi Kappa Phi.