5 Unit Commitment

5.1 INTRODUCTION

Because human activity follows cycles, most systems supplying services to a large population will experience cycles. This includes transportation systems, communication systems, as well as electric power systems. In the case of an electric power system, the total load on the system will generally be higher during the daytime and early evening when industrial loads are high, lights are on, and so forth, and lower during the late evening and early morning when most of the population is asleep. In addition, the use of electric power has a weekly cycle, the load being lower over weekend days than weekdays. But why is this a problem in the operation of an electric power system? Why not just simply commit enough units to cover the maximum system load and leave them running? Note that to "commit" a generating unit is to "turn it on;" that is, to bring the unit up to speed, synchronize it to the system, and connect it so it can deliver power to the network. The problem with "commit enough units and leave them on line" is one of economics. As will be shown in Example 5A, it is quite expensive to run too many generating units. A great deal of money can be saved by turning units off (decommitting them) when they are not needed.

EXAMPLE 5A

Suppose one had the three units given here:

Unit 1: \[
\begin{align*}
\text{Min} & = 150 \text{ MW} \\
\text{Max} & = 600 \text{ MW} \\
H_1 & = 510.0 + 7.2P_1 + 0.00142P_1^2 \text{ MBtu/h}
\end{align*}
\]

Unit 2:

\[
\begin{align*}
\text{Min} & = 100 \text{ MW} \\
\text{Max} & = 400 \text{ MW} \\
H_2 & = 310.0 + 7.85P_2 + 0.00194P_2^2 \text{ MBtu/h}
\end{align*}
\]

Unit 3:

\[
\begin{align*}
\text{Min} & = 50 \text{ MW} \\
\text{Max} & = 200 \text{ MW} \\
H_3 & = 78.0 + 7.97P_3 + 0.00482P_3^2 \text{ MBtu/h}
\end{align*}
\]
with fuel costs:

\[
\text{Fuel cost}_1 = 1.1 \text{ \(\$/\text{MBtu}\)}
\]

\[
\text{Fuel cost}_2 = 1.0 \text{ \(\$/\text{MBtu}\)}
\]

\[
\text{Fuel cost}_3 = 1.2 \text{ \(\$/\text{MBtu}\)}
\]

If we are to supply a load of 550 MW, what unit or combination of units should be used to supply this load most economically? To solve this problem, simply try all combinations of the three units. Some combinations will be infeasible if the sum of all maximum MW for the units committed is less than the load or if the sum of all minimum MW for the units committed is greater than the load. For each feasible combination, the units will be dispatched using the techniques of Chapter 3. The results are presented in Table 5.1.

Note that the least expensive way to supply the generation is not with all three units running, or even any combination involving two units. Rather, the optimum commitment is to only run unit 1, the most economic unit. By only running the most economic unit, the load can be supplied by that unit operating closer to its best efficiency. If another unit is committed, both unit 1 and the other unit will be loaded further from their best efficiency points such that the net cost is greater than unit 1 alone.

Suppose the load follows a simple "peak-valley" pattern as shown in Figure 5.1a. If the operation of the system is to be optimized, units must be shut down as the load goes down and then recommitted as it goes back up. We would like to know which units to drop and when. As we will show later, this problem is far from trivial when real generating units are considered. One approach to this solution is demonstrated in Example 5B, where a simple priority list scheme is developed.

<table>
<thead>
<tr>
<th>Unit 1</th>
<th>Unit 2</th>
<th>Unit 3</th>
<th>Max Generation</th>
<th>Min Generation</th>
<th>(P_1)</th>
<th>(P_2)</th>
<th>(P_3)</th>
<th>(F_1)</th>
<th>(F_2)</th>
<th>(F_3)</th>
<th>Total Cost (F_1 + F_2 + F_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Infeasible</td>
</tr>
<tr>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>200</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Infeasible</td>
</tr>
<tr>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>400</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Infeasible</td>
</tr>
<tr>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>600</td>
<td>150</td>
<td>400</td>
<td>150</td>
<td>0</td>
<td>3760</td>
<td>1658</td>
<td></td>
<td>5418</td>
</tr>
<tr>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>600</td>
<td>150</td>
<td>550</td>
<td>0</td>
<td>0</td>
<td>5389</td>
<td>0</td>
<td>0</td>
<td>5389</td>
</tr>
<tr>
<td>On</td>
<td>Off</td>
<td>On</td>
<td>800</td>
<td>200</td>
<td>500</td>
<td>0</td>
<td>50</td>
<td>4911</td>
<td>0</td>
<td>586</td>
<td>5497</td>
</tr>
<tr>
<td>On</td>
<td>On</td>
<td>Off</td>
<td>1000</td>
<td>250</td>
<td>295</td>
<td>255</td>
<td>0</td>
<td>3030</td>
<td>2440</td>
<td>0</td>
<td>5471</td>
</tr>
<tr>
<td>On</td>
<td>On</td>
<td>On</td>
<td>1200</td>
<td>300</td>
<td>267</td>
<td>233</td>
<td>50</td>
<td>2787</td>
<td>2244</td>
<td>586</td>
<td>5617</td>
</tr>
</tbody>
</table>
**EXAMPLE 5B**

Suppose we wish to know which units to drop as a function of system load. Let the units and fuel costs be the same as in Example 5A, with the load varying from a peak of 1200 MW to a valley of 500 MW. To obtain a “shut-down rule,” simply use a brute-force technique wherein all combinations of units will be tried (as in Example 5A) for each load value taken in steps of 50 MW from 1200 to 500. The results of applying this brute-force technique are given in Table 5.2. Our shut-down rule is quite simple.

When load is above 1000 MW, run all three units; between 1000 MW and 600 MW, run units 1 and 2; below 600 MW, run only unit 1.
TABLE 5.2 "Shut-down Rule" Derivation for Example 5B

<table>
<thead>
<tr>
<th>Load</th>
<th>Unit 1</th>
<th>Unit 2</th>
<th>Unit 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1200</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>1150</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>1100</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>1050</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>1000</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>950</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>900</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>850</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>800</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>750</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>700</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>650</td>
<td>On</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>600</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>550</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>500</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
</tbody>
</table>

Figure 5.1b shows the unit commitment schedule derived from this shut-down rule as applied to the load curve of Figure 5.1a.

So far, we have only obeyed one simple constraint: *Enough units will be committed to supply the load.* If this were all that was involved in the unit commitment problem—that is, just meeting the load—we could stop here and state that the problem was "solved." Unfortunately, other constraints and other phenomena must be taken into account in order to claim an optimum solution. These constraints will be discussed in the next section, followed by a description of some of the presently used methods of solution.

5.1.1 Constraints in Unit Commitment

Many constraints can be placed on the unit commitment problem. The list presented here is by no means exhaustive. Each individual power system, power pool, reliability council, and so forth, may impose different rules on the scheduling of units, depending on the generation makeup, load-curve characteristics, and such.

5.1.2 Spinning Reserve

*Spinning reserve* is the term used to describe the total amount of generation available from all units synchronized (i.e., spinning) on the system, minus the
INTRODUCTION

present load and losses being supplied. Spinning reserve must be carried so that the loss of one or more units does not cause too far a drop in system frequency (see Chapter 9). Quite simply, if one unit is lost, there must be ample reserve on the other units to make up for the loss in a specified time period.

Spinning reserve must be allocated to obey certain rules, usually set by regional reliability councils (in the United States) that specify how the reserve is to be allocated to various units. Typical rules specify that reserve must be a given percentage of forecasted peak demand, or that reserve must be capable of making up the loss of the most heavily loaded unit in a given period of time. Others calculate reserve requirements as a function of the probability of not having sufficient generation to meet the load.

Not only must the reserve be sufficient to make up for a generation-unit failure, but the reserves must be allocated among fast-responding units and slow-responding units. This allows the automatic generation control system (see Chapter 9) to restore frequency and interchange quickly in the event of a generating-unit outage.

Beyond spinning reserve, the unit commitment problem may involve various classes of “scheduled reserves” or “off-line” reserves. These include quick-start diesel or gas-turbine units as well as most hydro-units and pumped-storage hydro-units that can be brought on-line, synchronized, and brought up to full capacity quickly. As such, these units can be “counted” in the overall reserve assessment, as long as their time to come up to full capacity is taken into account.

Reserves, finally, must be spread around the power system to avoid transmission system limitations (often called “bottling” of reserves) and to allow various parts of the system to run as “islands,” should they become electrically disconnected.

EXAMPLE 5C

Suppose a power system consisted of two isolated regions: a western region and an eastern region. Five units, as shown in Figure 5.2, have been committed to supply 3090 MW. The two regions are separated by transmission tie lines that can together transfer a maximum of 550 MW in either direction. This is also shown in Figure 5.2. What can we say about the allocation of spinning reserve in this system?

The data for the system in Figure 5.2 are given in Table 5.3. With the exception of unit 4, the loss of any unit on this system can be covered by the spinning reserve on the remaining units. Unit 4 presents a problem, however. If unit 4 were to be lost and unit 5 were to be run to its maximum of 600 MW, the eastern region would still need 590 MW to cover the load in that region. The 590 MW would have to be transmitted over the tie lines from the western region, which can easily supply 590 MW from its reserves. However, the tie capacity of only 550 MW limits the transfer. Therefore, the loss of unit 4 cannot
be covered even though the entire system has ample reserves. The only solution to this problem is to commit more units to operate in the eastern region.

5.1.3 Thermal Unit Constraints

Thermal units usually require a crew to operate them, especially when turned on and turned off. A thermal unit can undergo only gradual temperature changes, and this translates into a time period of some hours required to bring the unit on-line. As a result of such restrictions in the operation of a thermal plant, various constraints arise, such as:

- **Minimum up time:** once the unit is running, it should not be turned off immediately.
- **Minimum down time:** once the unit is decommitted, there is a minimum time before it can be recommitted.
- **Crew constraints:** if a plant consists of two or more units, they cannot both be turned on at the same time since there are not enough crew members to attend both units while starting up.

In addition, because the temperature and pressure of the thermal unit must be moved slowly, a certain amount of energy must be expended to bring the unit on-line. This energy does not result in any MW generation from the unit and is brought into the unit commitment problem as a *start-up cost*.

The start-up cost can vary from a maximum "cold-start" value to a much smaller value if the unit was only turned off recently and is still relatively close to operating temperature. There are two approaches to treating a thermal unit during its down period. The first allows the unit’s boiler to cool down and then heat back up to operating temperature in time for a scheduled turn on. The second (called *banking*) requires that sufficient energy be input to the boiler to just maintain operating temperature. The costs for the two can be compared so that, if possible, the best approach (cooling or banking) can be chosen.

\[
\text{Start-up cost when cooling} = C_c (1 - e^{-t/\alpha}) \times F + C_f
\]

where
\[
C_c = \text{cold-start cost (MBtu)}
\]
\[
F = \text{fuel cost}
\]
\[
C_f = \text{fixed cost (includes crew expense, maintenance expenses) (in } \text{R})
\]
\[
\alpha = \text{thermal time constant for the unit}
\]
\[
t = \text{time (h) the unit was cooled}
\]

\[
\text{Start-up cost when banking} = C_t \times t \times F + C_f
\]

where
\[
C_t = \text{cost (MBtu/h) of maintaining unit at operating temperature}
\]

Up to a certain number of hours, the cost of banking will be less than the cost of cooling, as is illustrated in Figure 5.3.

Finally, the capacity limits of thermal units may change frequently, due to maintenance or unscheduled outages of various equipment in the plant; this must also be taken into account in unit commitment.

### 5.1.4 Other Constraints

#### 5.1.4.1 Hydro-Constraints

Unit commitment cannot be completely separated from the scheduling of hydro-units. In this text, we will assume that the hydrothermal scheduling (or "coordination") problem can be separated from the unit commitment problem. We, of course, cannot assert flatly that our treatment in this fashion will always result in an optimal solution.
5.1.4.2 Must Run
Some units are given a must-run status during certain times of the year for reason of voltage support on the transmission network or for such purposes as supply of steam for uses outside the steam plant itself.

5.1.4.3 Fuel Constraints
We will treat the “fuel scheduling” problem briefly in Chapter 6. A system in which some units have limited fuel, or else have constraints that require them to burn a specified amount of fuel in a given time, presents a most challenging unit commitment problem.

5.2 UNIT COMMITMENT SOLUTION METHODS
The commitment problem can be very difficult. As a theoretical exercise, let us postulate the following situation.

- We must establish a loading pattern for $M$ periods.
- We have $N$ units to commit and dispatch.
- The $M$ load levels and operating limits on the $N$ units are such that any one unit can supply the individual loads and that any combination of units can also supply the loads.

Next, assume we are going to establish the commitment by enumeration (brute force). The total number of combinations we need to try each hour is,

$$C(N, 1) + C(N, 2) + \ldots + C(N, N - 1) + C(N, N) = 2^N - 1$$
where \( C(N, j) \) is the combination of \( N \) items taken \( j \) at a time. That is,

\[
C(N, j) = \left[ \frac{N!}{(N-j)! j!} \right] \\
= \frac{N!}{(N-j)! j!} = 1 \times 2 \times 3 \times \ldots \times j
\]

For the total period of \( M \) intervals, the maximum number of possible combinations is \((2^N - 1)^M\), which can become a horrid number to think about.

For example, take a 24-h period (e.g., 24 one-hour intervals) and consider systems with 5, 10, 20, and 40 units. The value of \((2^N - 1)^{24}\) becomes the following.

<table>
<thead>
<tr>
<th>( N )</th>
<th>((2^N - 1)^{24})</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>(6.2 \times 10^{35})</td>
</tr>
<tr>
<td>10</td>
<td>(1.73 \times 10^{72})</td>
</tr>
<tr>
<td>20</td>
<td>(3.12 \times 10^{144})</td>
</tr>
<tr>
<td>40</td>
<td>(Too big)</td>
</tr>
</tbody>
</table>

These very large numbers are the upper bounds for the number of enumerations required. Fortunately, the constraints on the units and the load-capacity relationships of typical utility systems are such that we do not approach these large numbers. Nevertheless, the real practical barrier in the optimized unit commitment problem is the high dimensionality of the possible solution space.

The most talked-about techniques for the solution of the unit commitment problem are:

- Priority-list schemes,
- Dynamic programming (DP),
- Lagrange relation (LR).

### 5.2.1 Priority-List Methods

The simplest unit commitment solution method consists of creating a priority list of units. As we saw in Example 5B, a simple shut-down rule or priority-list scheme could be obtained after an exhaustive enumeration of all unit combinations at each load level. The priority list of Example 5B could be obtained in a much simpler manner by noting the full-load average production cost of each unit, where the full-load average production cost is simply the net heat rate at full load multiplied by the fuel cost.
EXAMPLE 5D

Construct a priority list for the units of Example 5A. (Use the same fuel costs as in Example 5A.) First, the full-load average production cost will be calculated:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Full Load Average Production Cost ($/MWh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.79</td>
</tr>
<tr>
<td>2</td>
<td>9.48</td>
</tr>
<tr>
<td>3</td>
<td>11.188</td>
</tr>
</tbody>
</table>

A strict priority order for these units, based on the average production cost, would order them as follows:

<table>
<thead>
<tr>
<th>Unit</th>
<th>$$/MWh</th>
<th>Min MW</th>
<th>Max MW</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>9.48</td>
<td>100</td>
<td>400</td>
</tr>
<tr>
<td>1</td>
<td>9.79</td>
<td>150</td>
<td>600</td>
</tr>
<tr>
<td>3</td>
<td>11.188</td>
<td>50</td>
<td>200</td>
</tr>
</tbody>
</table>

and the commitment scheme would (ignoring min up/down time, start-up costs, etc.) simply use only the following combinations.

<table>
<thead>
<tr>
<th>Combination</th>
<th>Min MW from Combination</th>
<th>Max MW from Combination</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 + 1 + 3</td>
<td>300</td>
<td>1200</td>
</tr>
<tr>
<td>2 + 1</td>
<td>250</td>
<td>1000</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>400</td>
</tr>
</tbody>
</table>

Note that such a scheme would not completely parallel the shut-down sequence described in Example 5B, where unit 2 was shut down at 600 MW leaving unit 1. With the priority-list scheme, both units would be held on until load reached 400 MW, then unit 1 would be dropped.

Most priority-list schemes are built around a simple shut-down algorithm that might operate as follows.
At each hour when load is dropping, determine whether dropping the next unit on the priority list will leave sufficient generation to supply the load plus spinning-reserve requirements. If not, continue operating as is; if yes, go on to the next step.

- Determine the number of hours, $H$, before the unit will be needed again. That is, assuming that the load is dropping and will then go back up some hours later.
- If $H$ is less than the minimum shut-down time for the unit, keep commitment as is and go to last step; if not, go to next step.
- Calculate two costs. The first is the sum of the hourly production costs for the next $H$ hours with the unit up. Then recalculate the same sum for the unit down and add in the start-up cost for either cooling the unit or banking it, whichever is less expensive. If there is sufficient savings from shutting down the unit, it should be shut down, otherwise keep it on.
- Repeat this entire procedure for the next unit on the priority list. If it is also dropped, go to the next and so forth.

Various enhancements to the priority-list scheme can be made by grouping of units to ensure that various constraints are met. We will note later that dynamic-programming methods usually create the same type of priority list for use in the DP search.

5.2.2 Dynamic-Programming Solution

5.2.2.1 Introduction

Dynamic programming has many advantages over the enumeration scheme, the chief advantage being a reduction in the dimensionality of the problem. Suppose we have found units in a system and any combination of them could serve the (single) load. There would be a maximum of $2^4 - 1 = 15$ combinations to test. However, if a strict priority order is imposed, there are only four combinations to try:

- Priority 1 unit
- Priority 1 unit + Priority 2 unit
- Priority 1 unit + Priority 2 unit + Priority 3 unit
- Priority 1 unit + Priority 2 unit + Priority 3 unit + Priority 4 unit

The imposition of a priority list arranged in order of the full-load average-cost rate would result in a theoretically correct dispatch and commitment only if:
1. No load costs are zero.
2. Unit input–output characteristics are linear between zero output and full load.
3. There are no other restrictions.
4. Start-up costs are a fixed amount.

In the dynamic-programming approach that follows, we assume that:

1. A state consists of an array of units with specified units operating and the rest off-line.
2. The start-up cost of a unit is independent of the time it has been off-line (i.e., it is a fixed amount).
3. There are no costs for shutting down a unit.
4. There is a strict priority order, and in each interval a specified minimum amount of capacity must be operating.

A feasible state is one in which the committed units can supply the required load and that meets the minimum amount of capacity each period.

5.2.2.2 Forward DP Approach

One could set up a dynamic-programming algorithm to run backward in time starting from the final hour to be studied, back to the initial hour. Conversely, one could set up the algorithm to run forward in time from the initial hour to the final hour. The forward approach has distinct advantages in solving generator unit commitment. For example, if the start-up cost of a unit is a function of the time it has been off-line (i.e., its temperature), then a forward dynamic-program approach is more suitable since the previous history of the unit can be computed at each stage. There are other practical reasons for going forward. The initial conditions are easily specified and the computations can go forward in time as long as required. A forward dynamic-programming algorithm is shown by the flowchart in Figure 5.4.

The recursive algorithm to compute the minimum cost in hour $K$ with combination $I$ is,

$$F_{\text{cost}}(K, I) = \min_{|L|} [P_{\text{cost}}(K, I) + S_{\text{cost}}(K - 1, L; K, I) + F_{\text{cost}}(K - 1, L)]$$  \hspace{1cm} (5.1)

where

$$F_{\text{cost}}(K, I) = \text{least total cost to arrive at state } (K, I)$$
$$P_{\text{cost}}(K, I) = \text{production cost for state } (K, I)$$
$$S_{\text{cost}}(K - 1, L; K, I) = \text{transition cost from state } (K - 1, L) \text{ to state } (K, I)$$
State \((K, I)\) is the \(I^{th}\) combination in hour \(K\). For the forward dynamic-programming approach, we define a *strategy* as the transition, or path, from one state at a given hour to a state at the next hour.

Note that two new variables, \(X\) and \(N\), have been introduced in Figure 5.4.

\[
X = \text{number of states to search each period}
\]

\[
N = \text{number of strategies, or paths, to save at each step}
\]
Restricted search paths in DP algorithm with $N = 3$ and $X = 5$.

These variables allow control of the computational effort (see Figure 5.5). For complete enumeration, the maximum number of the value of $X$ or $N$ is $2^n - 1$.

For example, with a simple priority-list ordering, the upper bound on $X$ is $n$, the number of units. Reducing the number $N$ means that we are discarding the highest cost schedules at each time interval and saving only the lowest $N$ paths or strategies. There is no assurance that the theoretical optimal schedule will be found using a reduced number of strategies and search range (the $X$ value); only experimentation with a particular program will indicate the potential error associated with limiting the values of $X$ and $N$ below their upper bounds.

**EXAMPLE 5E**

For this example, the complete search range will be used and three cases will be studied. The first is a priority-list schedule, the second is the same example with complete enumeration. Both of the first two cases ignore hot-start costs and minimum up and down times. The third case includes the hot-start costs, as well as the minimum up and down times. Four units are to be committed to serve an 8-h load pattern. Data on the units and the load pattern are contained in Table 5.4.
### TABLE 5.4 Unit Characteristics, Load Pattern, and Initial Status for the Cases in Example 5E

<table>
<thead>
<tr>
<th>Unit</th>
<th>Max (MW)</th>
<th>Min (MW)</th>
<th>Incremental Heat Rate (Btu/kWh)</th>
<th>No-Load Cost (R/h)</th>
<th>Full-Load Ave. Cost (R/mWh)</th>
<th>Minimum Times (h)</th>
<th>Initial Conditions</th>
<th>Start-Up Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80</td>
<td>25</td>
<td>10440</td>
<td>213.00</td>
<td>23.54</td>
<td>4</td>
<td>-5</td>
<td>150</td>
</tr>
<tr>
<td>2</td>
<td>250</td>
<td>60</td>
<td>9000</td>
<td>585.62</td>
<td>20.34</td>
<td>5</td>
<td>8</td>
<td>170</td>
</tr>
<tr>
<td>3</td>
<td>300</td>
<td>75</td>
<td>8730</td>
<td>684.74</td>
<td>19.74</td>
<td>5</td>
<td>8</td>
<td>500</td>
</tr>
<tr>
<td>4</td>
<td>60</td>
<td>20</td>
<td>11900</td>
<td>252.00</td>
<td>28.00</td>
<td>1</td>
<td>-6</td>
<td>0</td>
</tr>
</tbody>
</table>

Initial Conditions:
- **Hours Off-Line (−)** or **On-Line (+)**

Start-Up Costs:
- **Hot** (R)
- **Cold** (R)
- **Cold Start** (h)

Load Pattern:

<table>
<thead>
<tr>
<th>Hour</th>
<th>Load (MW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>450</td>
</tr>
<tr>
<td>2</td>
<td>530</td>
</tr>
<tr>
<td>3</td>
<td>600</td>
</tr>
<tr>
<td>4</td>
<td>540</td>
</tr>
<tr>
<td>5</td>
<td>400</td>
</tr>
<tr>
<td>6</td>
<td>280</td>
</tr>
<tr>
<td>7</td>
<td>290</td>
</tr>
<tr>
<td>8</td>
<td>500</td>
</tr>
</tbody>
</table>

In order to make the required computations more efficiently, a simplified model of the unit characteristics is used. In practical applications, two- or three-section stepped incremental curves might be used, as shown in Figure 5.6. For our example, only a single step between minimum and the maximum power points is used. The units in this example have linear $F(P)$ functions:
The $F(P)$ function is:

$$F(P) = \text{No-load cost} + \text{Inc cost} \times P$$

Note, however, that the unit must operate within its limits. Start-up costs for the first two cases are taken as the cold-start costs. The priority order for the four units in the example is: unit 3, unit 2, unit 1, unit 4. For the first two cases, the minimum up and down times are taken as 1 h for all units.

In all three cases we will refer to the capacity ordering of the units. This is shown in Table 5.5, where the unit combinations or states are ordered by maximum net capacity for each combination.

**Case 1**

In Case 1, the units are scheduled according to a strict priority order. That is, units are committed in order until the load is satisfied. The total cost for the interval is the sum of the eight dispatch costs plus the transitional costs for starting any units. In this first case, a maximum of 24 dispatches must be considered.
TABLE 5.5 Capacity Ordering of the Units

<table>
<thead>
<tr>
<th>State</th>
<th>Unit Combination</th>
<th>Maximum Net Capacity for Combination</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1 1 1 1</td>
<td>690</td>
</tr>
<tr>
<td>14</td>
<td>1 1 1 0</td>
<td>630</td>
</tr>
<tr>
<td>13</td>
<td>0 1 1 1</td>
<td>610</td>
</tr>
<tr>
<td>12</td>
<td>0 1 1 0</td>
<td>550</td>
</tr>
<tr>
<td>11</td>
<td>1 0 1 1</td>
<td>440</td>
</tr>
<tr>
<td>10</td>
<td>1 1 0 1</td>
<td>390</td>
</tr>
<tr>
<td>9</td>
<td>1 0 1 0</td>
<td>380</td>
</tr>
<tr>
<td>8</td>
<td>0 0 1 1</td>
<td>360</td>
</tr>
<tr>
<td>7</td>
<td>1 1 0 0</td>
<td>330</td>
</tr>
<tr>
<td>6</td>
<td>0 1 0 1</td>
<td>310</td>
</tr>
<tr>
<td>5</td>
<td>0 0 1 0</td>
<td>300</td>
</tr>
<tr>
<td>4</td>
<td>0 1 0 0</td>
<td>250</td>
</tr>
<tr>
<td>3</td>
<td>1 0 0 1</td>
<td>140</td>
</tr>
<tr>
<td>2</td>
<td>1 0 0 0</td>
<td>80</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 1</td>
<td>60</td>
</tr>
<tr>
<td>0</td>
<td>0 0 0 0</td>
<td>0</td>
</tr>
</tbody>
</table>

Unit 1 2 3 4

*1 = Committed (unit operating).
0 = Uncommitted (unit shut down).

For Case 1, the only states examined each hour consist of:

<table>
<thead>
<tr>
<th>State No.</th>
<th>Unit Status</th>
<th>Capacity (MW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0 0 1 0</td>
<td>300</td>
</tr>
<tr>
<td>12</td>
<td>0 1 1 0</td>
<td>550</td>
</tr>
<tr>
<td>14</td>
<td>1 1 1 0</td>
<td>630</td>
</tr>
<tr>
<td>15</td>
<td>1 1 1 1</td>
<td>690</td>
</tr>
</tbody>
</table>

Note that this is the priority order; that is, state 5 = unit 3, state 12 = units 3 + 2, state 14 = unit 3 + 2 + 1, and state 15 = units 3 + 2 + 1 + 4. For the first 4 h, only the last three states are of interest. The sample calculations illustrate the technique. All possible commitments start from state 12 since this was given as the initial condition. For hour 1, the minimum cost is state 12, and so on. The results for the priority-ordered case are as follows.

<table>
<thead>
<tr>
<th>Hour</th>
<th>State with Min Total Cost</th>
<th>Pointer for Previous Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12 (9208)</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>12 (19857)</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>14 (32472)</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>12 (43300)</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note that state 13 is not reachable in this strict priority ordering.
Sample Calculations for Case 1

\[ F_{\text{cost}}(J, K) = \min_{\{L\}} [P_{\text{cost}}(J, K) + S_{\text{cost}}(J - 1, L; J, K) + F_{\text{cost}}(J - 1, L)] \]

Allowable states are

\[ \{ \} = \{0010, 0110, 1110, 1111\} = \{5, 12, 14, 15\} \]

In hour 0\{L\} = \{12\}, initial condition.

\( J = 1: \) 1st hour

\[
\begin{align*}
F_{\text{cost}}(1, 15) &= P_{\text{cost}}(1, 15) + S_{\text{cost}}(0, 12; 1, 15) \\
&= 9861 + 350 = 10211 \\
F_{\text{cost}}(1, 14) &= 9493 + 350 = 9843 \\
F_{\text{cost}}(1, 12) &= 9208 + 0 = 9208
\end{align*}
\]

\( J = 2: \) 2nd hour

Feasible states are \{12, 14, 15\} = \{K\}, so \( X = 3 \). Suppose two strategies are saved at each stage, so \( N = 2 \), and \{L\} = \{12, 14\},

\[
\begin{align*}
F_{\text{cost}}(2, 15) &= \min_{\{12, 14\}} [P_{\text{cost}}(2, 15) + S_{\text{cost}}(1, L; 2, 15) + F_{\text{cost}}(1, L)] \\
&= 11301 + \min \left[ \frac{(350 + 9208)}{(0 + 9843)} \right] = 20859
\end{align*}
\]

and so on.

Case 2

In Case 2, complete enumeration is tried with a limit of \( (2^4 - 1) = 15 \) dispatches each of the eight hours, so that there is a theoretical maximum of \( 15^8 = 2.56 \cdot 10^9 \) possibilities. Fortunately, most of these are not feasible because they do not supply sufficient capacity, and can be discarded with little analysis required.

Figure 5.7 illustrates the computational process for the first 4 h for Case 2. On the figure itself, the circles denote states each hour. The numbers within the circles are the “pointers.” That is, they denote the state number in the previous hour that provides the path to that particular state in the current hour. For example, in hour 2, the minimum costs for states 12, 13, 14, and 15, all result from transitions from state 12 in hour 1. Costs shown on the connections are the start-up costs. At each state, the figures shown are the hourly cost/total cost.
In Case 2, the true optimal commitment is found. That is, it is less expensive to turn on the less efficient peaking unit, number 4, for hour 3, than to start up the more efficient unit 1 for that period. By hour 3, the difference in total cost is \( \$165 \), or \( \$0.104/\text{MWh} \). This is not an insignificant amount when compared with the fuel cost per MWh for an average thermal unit with a net heat rate of 10,000 Btu/kWh and a fuel cost of \( \$2.00 \) MBtu. A savings of \( \$165 \) every 3 h is equivalent to \( \$481,800/\text{yr} \).

The total 8-h trajectories for Cases 1 and 2 are shown in Figure 5.8. The neglecting of start-up and shut-down restrictions in these two cases permits the shutting down of all but unit 3 in hours 6 and 7. The only difference in the two trajectories occurs in hour 3, as discussed in the previous paragraph.

Case 3

In case 3, the original unit data are used so that the minimum shut-down and operating times are observed. The forward dynamic-programming algorithm was repeated for the same 8-h period. Complete enumeration was used. That is, the upper bound on \( X \) shown in the flowchart was 15. Three different values for \( N \), the number of strategies saved at each stage, were taken as 4, 8, and 10. The same trajectory was found for values of 8 and 10. This trajectory is shown in Figure 5.9. However, when only four strategies were saved, the procedure flounders (i.e., fails to find a feasible path) in

---

**FIG. 5.7** Example 5E, Cases 1 and 2 (first 4 h).
<table>
<thead>
<tr>
<th>Status number</th>
<th>Unit status</th>
<th>Total capacity</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1111</td>
<td>690</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>630</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>13</td>
<td>0111</td>
<td>610</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>12</td>
<td>0110</td>
<td>550</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>440</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>10</td>
<td>1101</td>
<td>390</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9</td>
<td>1010</td>
<td>380</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>8</td>
<td>0011</td>
<td>360</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td>1100</td>
<td>330</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>6</td>
<td>0101</td>
<td>310</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>5</td>
<td>0010</td>
<td>300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>250</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>1001</td>
<td>140</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>80</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>60</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

**FIG. 5.8** Example 5E, Cases 1 and 2 (complete solutions).
hour 8, because the lowest cost strategies in hour 7 have shut down units that cannot be restarted in hour 8 because of minimum unit downtime rules.

The practical remedy for this deficiency in the method shown in Figure 5.4 is to return to a period prior to the low-load hours and temporarily keep more (i.e., higher cost) strategies. This will permit keeping a nominal number of strategies at each stage. The other alternative is, of course, the method used here: run the entire period with more strategies saved.

These cases can be summarized in terms of the total costs found for the 8-h period, as shown in Table 5.6. These cases illustrate the forward dynamic-programming method and also point out the problems involved in the practical application of the method.

![FIG. 5.9 Example 5E, Case 3.](image)

<table>
<thead>
<tr>
<th>State number</th>
<th>Unit status</th>
<th>Total capacity</th>
<th>Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1111</td>
<td>690</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>630</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>0111</td>
<td>610</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>0110</td>
<td>550</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>440</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>1101</td>
<td>390</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>1010</td>
<td>380</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 5.6 Summary of Cases 1–3**

<table>
<thead>
<tr>
<th>Case</th>
<th>Conditions</th>
<th>Total Cost (R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Priority order. Up and down times neglected</td>
<td>73439</td>
</tr>
<tr>
<td>2</td>
<td>Enumeration ((X \leq 15)) with 4 strategies ((N)) saved. Up and down times neglected</td>
<td>73274</td>
</tr>
<tr>
<td>3</td>
<td>(X \leq 15). Up and down times observed</td>
<td>No solution</td>
</tr>
<tr>
<td></td>
<td>(N = 4) strategies</td>
<td>74110</td>
</tr>
<tr>
<td></td>
<td>(B = 8) strategies</td>
<td>74110</td>
</tr>
<tr>
<td></td>
<td>(N = 10) strategies</td>
<td>74110</td>
</tr>
</tbody>
</table>
5.2.3 Lagrange Relaxation Solution

The dynamic-programming method of solution of the unit commitment problem has many disadvantages for large power systems with many generating units. This is because of the necessity of forcing the dynamic-programming solution to search over a small number of commitment states to reduce the number of combinations that must be tested in each time period.

In the Lagrange relaxation technique these disadvantages disappear (although other technical problems arise and must be addressed, as we shall see). This method is based on a dual optimization approach as introduced in Appendix 3A and further expanded in the appendix to this chapter. (The reader should be familiar with both of these appendices before proceeding further.)

We start by defining the variable $U_i^t$ as:

$$
U_i^t = 0 \text{ if unit } i \text{ is off-line during period } t
$$

$$
U_i^t = 1 \text{ if unit } i \text{ is on-line during period } t
$$

We shall now define several constraints and the objective function of the unit commitment problem:

1. Loading constraints:

$$
P_{\text{load}}^t - \sum_{i=1}^{N} p_i^t U_i^t = 0 \quad \text{for } t = 1 \ldots T
$$

2. Unit limits:

$$
U_i^t P_i^\text{min} \leq P_i^t \leq U_i^t P_i^\text{max} \quad \text{for } i = 1 \ldots N \text{ and } t = 1 \ldots T
$$

3. Unit minimum up- and down-time constraints. Note that other constraints can easily be formulated and added to the unit commitment problem. These include transmission security constraints (see Chapter 11), generator fuel limit constraints, and system air quality constraints in the form of limits on emissions from fossil-fired plants, spinning reserve constraints, etc.

4. The objective function is:

$$
\sum_{t=1}^{T} \sum_{i=1}^{N} [f_i(P_i^t) + \text{Start up cost}_{i,t}] U_i^t = F(P_i^t, U_i^t)
$$

We can then form the Lagrange function similar to the way we did in the economic dispatch problem:

$$
\mathcal{L}(P, U, \lambda) = F(P_i^t, U_i^t) + \sum_{t=1}^{T} \lambda^t \left( p_i^t \text{load} - \sum_{i=1}^{N} p_i^t U_i^t \right)
$$

The unit commitment problem requires that we minimize the Lagrange function
above, subject to the local unit constraints 2 and 3, which can be applied to each unit separately. Note:

1. The cost function, $F(P_i^t, U_i^t)$, together with constraints 2 and 3 are each separable over units. That is, what is done with one unit does not affect the cost of running another unit, as far as the cost function and the unit limits (constraint 2) and the unit up- and down-time (constraint 3) are concerned.

2. Constraints 1 are coupling constraints across the units so that what we do to one unit affects what will happen on other units if the coupling constraints are to be met.

The Lagrange relaxation procedure solves the unit commitment problem by "relaxing" or temporarily ignoring the coupling constraints and solving the problem as if they did not exist. This is done through the dual optimization procedure as explained in the appendix of this chapter. The dual procedure attempts to reach the constrained optimum by maximizing the Lagrangian with respect to the Lagrange multipliers, while minimizing with respect to the other variables in the problem; that is:

$$q^*(\lambda) = \max_{\lambda^i} q(\lambda)$$

where

$$q(\lambda) = \min_{P_i^t, U_i^t} \mathcal{L}(P, U, \lambda)$$

This is done in two basic steps:

**Step 1** Find a value for each $\lambda^i$ which moves $q(\lambda)$ toward a larger value.

**Step 2** Assuming that the $\lambda^i$ found in step 1 are now fixed, find the minimum of $\mathcal{L}$ by adjusting the values of $P_i^t$ and $U_i^t$.

The adjustment of the $\lambda^i$ values will be dealt with at a later time in this section; assume then that a value has been chosen for all the $\lambda^i$ and that they are now to be treated as fixed numbers. We shall minimize the Lagrangian as follows.

First, we rewrite the Lagrangian as:

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{i=1}^{N} \left[ F_i(P_i^t) + \text{Start up cost}_{i,t} \right] U_i^t + \sum_{t=1}^{T} \lambda^i \left( P_{\text{load}}^t - \sum_{i=1}^{N} P_i^t U_i^t \right)$$

This is now rewritten as:

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{i=1}^{N} \left[ F_i(P_i^t) + \text{Start up cost}_{i,t} \right] U_i^t + \sum_{t=1}^{T} \lambda^i P_{\text{load}}^t - \sum_{i=1}^{T} \sum_{i=1}^{N} \lambda^i P_i^t U_i^t$$
The second term above is constant and can be dropped (since the $\lambda^t$ are fixed). Finally, we write the Lagrange function as:

$$L = \sum_{i=1}^{N} \left( \sum_{t=1}^{T} \left[ F_i(P_i^t) + \text{Start up cost}_{i,t} \right] U_i^t - \lambda^t P_i^t U_i^t \right)$$  \hspace{1cm} (5.10)

Here, we have achieved our goal of separating the units from one another. The term inside the outer brackets; that is:

$$\sum_{t=1}^{T} \left[ F_i(P_i^t) + \text{Start up cost}_{i,t} \right] U_i^t - \lambda^t P_i^t U_i^t$$

can be solved separately for each generating unit, without regard for what is happening on the other generating units. The minimum of the Lagrangian is found by solving for the minimum for each generating unit over all time periods; that is:

$$\min q(\lambda) = \sum_{i=1}^{N} \min \sum_{t=1}^{T} \left[ F_i(P_i^t) + \text{Start up cost}_{i,t} \right] U_i^t - \lambda^t P_i^t U_i^t$$  \hspace{1cm} (5.11)

Subject to

$$U_i^t P_i^{\min} \leq P_i^t \leq U_i^t P_i^{\max} \text{ for } t = 1 \ldots T$$

and the up- and down-time constraints. This is easily solved as a dynamic-programming problem in one variable. This can be visualized in the figure below, which shows the only two possible states for unit $i$ (i.e., $U_i^t = 0$ or 1):

where $S_i$ is the start-up cost for unit $i$.

At the $U_i^t = 0$ state, the value of the function to minimized is trivial (i.e., it equals zero); at the state where $U_i^t = 1$, the function to be minimized is (the start-up cost is dropped here since the minimization is with respect to $P_i^t$):

$$\min \left[ F_i(P_i) - \lambda^t P_i^t \right]$$  \hspace{1cm} (5.12)

The minimum of this function is found by taking the first derivative:

$$\frac{d}{dP_i^t} \left[ F_i(P_i) - \lambda^t P_i^t \right] = \frac{d}{dP_i^t} F_i(P_i) - \lambda^t = 0$$  \hspace{1cm} (5.13)
The solution to this equation is

\[ \frac{d}{dP_i} F_i(P_i^{\text{opt}}) = \lambda^t \] (5.14)

There are three cases to be concerned with depending on the relation of $P_i^{\text{opt}}$ and the unit limits:

1. If $P_i^{\text{opt}} \leq P_i^{\min}$, then:

\[ \min [F_i(P_i) - \lambda^t P_i^t] = F_i(P_i^{\min}) - \lambda^t P_i^{\min} \] (5.15a)

2. If $P_i^{\min} \leq P_i^{\text{opt}} \leq P_i^{\max}$, then:

\[ \min [F_i(P_i) - \lambda^t P_i^t] = F_i(P_i^{\text{opt}}) - \lambda^t P_i^{\text{opt}} \] (5.15b)

3. If $P_i^{\text{opt}} \geq P_i^{\max}$, then:

\[ \min [F_i(P_i) - \lambda^t P_i^t] = F_i(P_i^{\max}) - \lambda^t P_i^{\max} \] (5.15c)

The solution of the two-state dynamic program for each unit proceeds in the normal manner as was done for the forward dynamic-programming solution of the unit commitment problem itself. Note that since we seek to minimize $[F_i(P_i) - \lambda^t P_i^t]$ at each stage and that when $U^t = 0$ this value goes to zero, then the only way to get a value lower is to have

\[ [F_i(P_i) - \lambda^t P_i^t] < 0 \]

The dynamic program should take into account all the start-up costs, $S_i$, for each unit, as well as the minimum up and down time for the generator. Since we are solving for each generator independently, however, we have avoided the dimensionality problems that affect the dynamic-programming solution.

5.2.3.1 Adjusting $\lambda$

So far, we have shown how to schedule generating units with fixed values of $\lambda^t$ for each time period. As shown in the appendix to this chapter, the adjustment of $\lambda^t$ must be done carefully so as to maximize $q(\lambda)$. Most references to work on the Lagrange relaxation procedure use a combination of gradient search and various heuristics to achieve a rapid solution. Note that unlike in the appendix, the $\lambda$ here is a vector of values, each of which must be adjusted. Much research in recent years has been aimed at ways to speed the search for the correct values of $\lambda$ for each hour. In Example 5D, we shall use the same technique of adjusting $\lambda$ for each hour that is used in the appendix. For the unit commitment problem solved in Example 5D, however, the $\lambda$ adjustment factors are different:

\[ \lambda^t = \lambda^t + \left[ \frac{d}{d\lambda} q(\lambda) \right] x \] (5.16)
where

\[ x = 0.01 \quad \text{when} \quad \frac{d}{d\lambda} q(\lambda) \text{ is positive} \quad (5.17) \]

and

\[ x = 0.002 \quad \text{when} \quad \frac{d}{d\lambda} q(\lambda) \text{ is negative} \quad (5.18) \]

Each \( \lambda' \) is treated separately. The reader should consult the references listed at the end of this chapter for more efficient methods of adjusting the \( \lambda \) values. The overall Lagrange relaxation unit commitment algorithm is shown in Figure 5.10.

Reference 15 introduces the use of what this text called the "relative duality gap" or \((J^* - q^*)/q^*\). The relative duality gap is used in Example 5D as a measure of the closeness to the solution. Reference 15 points out several useful things about dual optimization applied to the unit commitment problem.

1. For large, real-sized, power-system unit commitment calculations, the duality gap does become quite small as the dual optimization proceeds, and its size can be used as a stopping criterion. The larger the problem (larger number of generating units), the smaller the gap.
2. The convergence is unstable at the end, meaning that some units are being switched in and out, and the process never comes to a definite end.
3. There is no guarantee that when the dual solution is stopped, it will be at a feasible solution.

All of the above are demonstrated in Example 5D. The duality gap is large at the beginning and becomes progressively smaller as the iterations progress. The solution reaches a commitment schedule when at least enough generation is committed so that an economic dispatch can be run, and further iterations only result in switching marginal units on and off. Finally, the loading constraints are not met by the dual solution when the iterations are stopped.

Many of the Lagrange relaxation unit commitment programs use a few iterations of a dynamic-programming algorithm to get a good starting point, then run the dual optimization iterations, and finally, at the end, they use heuristic logic or restricted dynamic programming to get to a final solution. The result is a solution that is not limited to search windows, such as had to be done in strict application of dynamic programming.

**EXAMPLE 5D**

In this example, a three-generator, four-hour unit commitment problem will be solved. The data for this problem are as follows. Given the three generating
Pick starting $\lambda^k$ for $t=1...T$

$k = 0$

for each unit $i$

Build dynamic program having two states, and $T$ stages and solve for:

$P_t^i$ and $U_t^i$ for all $t = 1...T$

No

last unit done

Yes

Solve for the dual value $q^i(\lambda^k)$

Using the $U^i$ calculate the primal value $J^*$

that is, solve an economic dispatch for each hour using the units that have been committed for that hour

$\frac{f^*}{q^*}$

calculate the relative duality gap

relative duality gap sufficiently small

update $\lambda^i$ for all $t$

update final adjustments to unit commitment schedule to achieve feasibility

done

FIG. 5.10 Lagrange relaxation procedure for unit commitment.

units below:

$$F_1(P_1) = 500 + 10P_1 + 0.002P_1^2 \quad \text{and} \quad 100 < P_1 < 600$$

$$F_2(P_2) = 300 + 8P_2 + 0.0025P_2^2 \quad \text{and} \quad 100 < P_2 < 400$$

$$F_3(P_3) = 100 + 6P_3 + 0.005P_3^2 \quad \text{and} \quad 50 < P_3 < 200$$
No start-up costs, no minimum up- or down-time constraints.

This example is solved using the Lagrange relaxation technique. Shown below are the results of several iterations, starting from an initial condition where all the $\lambda_i$ values are set to zero. An economic dispatch is run for each hour, provided there is sufficient generation committed that hour. If there is not enough generation committed, the total cost for that hour is set arbitrarily to 10,000. Once each hour has enough generation committed, the primal value $J^*$ simply represents the total generation cost summed over all hours as calculated by the economic dispatch.

The dynamic program for each unit with a $\lambda_i = 0$ for each hour will always result in all generating units off-line.

### Iteration 1

<table>
<thead>
<tr>
<th>Hour</th>
<th>$\lambda$</th>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$u_3$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_{\text{load}}$ - $\sum_{i=1}^{N} P_i^t U_i^t$</th>
<th>$p_{\text{dc}}^{\text{p}1}$</th>
<th>$p_{\text{dc}}^{\text{p}2}$</th>
<th>$p_{\text{dc}}^{\text{p}3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>170</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>520</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>330</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$q(\lambda) = 0.0, \quad J^* = 40,000, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = \text{undefined}$

In the next iteration, the $\lambda_i$ values have been increased. To illustrate the use of dynamic programming to schedule each generator, we will detail the DP steps for unit 3:

$\lambda = 1.7 \quad 5.2 \quad 11.0 \quad 3.3$

$F(P) - \lambda P = 327.5 \quad 152.5 \quad -700.0 \quad 247.5$

$P_3^1 = p_3^{\text{min}} \quad P_3^2 = p_3^{\text{min}} \quad P_3^3 = p_3^{\text{max}} \quad P_3^4 = p_3^{\text{min}}$

$U_3 = 1 \quad \bullet \quad \bullet \quad \bullet \quad \bullet$

$U_3 = 0 \quad \bullet \quad \bullet \quad \bullet \quad \bullet$

$\text{min cost: } -700$
The result is to schedule unit 3 off during hours 1, 2, and 4 and on during hour 3. Further, unit 3 is scheduled to be at its maximum of 200 MW during hour 3. The results, after all the units have been scheduled by DP, are as follows.

### Iteration 2

<table>
<thead>
<tr>
<th>Hour</th>
<th>( \lambda )</th>
<th>( u_1 )</th>
<th>( u_2 )</th>
<th>( u_3 )</th>
<th>( P_1 )</th>
<th>( P_2 )</th>
<th>( P_3 )</th>
<th>( P_{\text{load}} - \sum_{i=1}^{N} P_i U_i )</th>
<th>( P_{1\text{dc}} )</th>
<th>( P_{2\text{dc}} )</th>
<th>( P_{3\text{dc}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>170</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>5.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>520</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>11.0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>400</td>
<td>200</td>
<td>500</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>330</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ q(\lambda) = 14,982, \quad J^* = 40,000, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = 1.67 \]

### Iteration 3

<table>
<thead>
<tr>
<th>Hour</th>
<th>( \lambda )</th>
<th>( u_1 )</th>
<th>( u_2 )</th>
<th>( u_3 )</th>
<th>( P_1 )</th>
<th>( P_2 )</th>
<th>( P_3 )</th>
<th>( P_{\text{load}} - \sum_{i=1}^{N} P_i U_i )</th>
<th>( P_{1\text{dc}} )</th>
<th>( P_{2\text{dc}} )</th>
<th>( P_{3\text{dc}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>170</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10.4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>400</td>
<td>200</td>
<td>-80</td>
<td>0</td>
<td>320</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>16.0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>600</td>
<td>400</td>
<td>200</td>
<td>-100</td>
<td>500</td>
<td>400</td>
<td>200</td>
</tr>
<tr>
<td>4</td>
<td>6.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>330</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ q(\lambda) = 18,344, \quad J^* = 36,024, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = 0.965 \]

### Iteration 4

<table>
<thead>
<tr>
<th>Hour</th>
<th>( \lambda )</th>
<th>( u_1 )</th>
<th>( u_2 )</th>
<th>( u_3 )</th>
<th>( P_1 )</th>
<th>( P_2 )</th>
<th>( P_3 )</th>
<th>( P_{\text{load}} - \sum_{i=1}^{N} P_i U_i )</th>
<th>( P_{1\text{dc}} )</th>
<th>( P_{2\text{dc}} )</th>
<th>( P_{3\text{dc}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>170</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10.24</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>400</td>
<td>200</td>
<td>-80</td>
<td>0</td>
<td>320</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>15.8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>600</td>
<td>400</td>
<td>200</td>
<td>-100</td>
<td>500</td>
<td>400</td>
<td>200</td>
</tr>
<tr>
<td>4</td>
<td>9.9</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>380</td>
<td>200</td>
<td>-250</td>
<td>0</td>
<td>130</td>
<td>200</td>
</tr>
</tbody>
</table>

\[ q(\lambda) = 19,214, \quad J^* = 28,906, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = 0.502 \]
Iteration 5

<table>
<thead>
<tr>
<th>Hour</th>
<th>$\lambda$</th>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$u_3$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_{\text{load}} - \sum_{i=1}^{N} P_i^U U_i$</th>
<th>$P_{\text{dc}}$</th>
<th>$P_{\text{dc}}$</th>
<th>$P_{\text{dc}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>170</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10.08</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>400 200</td>
<td>-80</td>
<td>320</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>15.6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>600</td>
<td>400</td>
<td>200</td>
<td>-100 500 400 200</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>9.4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>130</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$q(\lambda) = 19.532, \quad J^* = 36,024, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = 0.844$

Iteration 6

<table>
<thead>
<tr>
<th>Hour</th>
<th>$\lambda$</th>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$u_3$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_{\text{load}} - \sum_{i=1}^{N} P_i^U U_i$</th>
<th>$P_{\text{dc}}$</th>
<th>$P_{\text{dc}}$</th>
<th>$P_{\text{dc}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>-30</td>
<td>0</td>
<td>0</td>
<td>170</td>
</tr>
<tr>
<td>2</td>
<td>9.92</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>384</td>
<td>200</td>
<td>-64</td>
<td>320</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>15.4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>600</td>
<td>400</td>
<td>200</td>
<td>-100 500 400 200</td>
<td>0</td>
<td>0</td>
<td>200</td>
</tr>
<tr>
<td>4</td>
<td>10.7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>400</td>
<td>200</td>
<td>-270</td>
<td>130</td>
<td>200</td>
</tr>
</tbody>
</table>

$q(\lambda) = 19.442, \quad J^* = 20,170, \quad \text{and} \quad \frac{J^* - q^*}{q^*} = 0.037$

The commitment schedule does not change significantly with further iterations, although it is not by any means stable. Further iterations do reduce the duality gap somewhat, but these iterations are unstable in that unit 2 is on the borderline between being committed and not being committed, and is switched in and out with no final convergence. After 10 iterations, $q(\lambda) = 19,485, \quad J^* = 20,017, \quad \text{and} \quad (J^* - q^*)/q^* = 0.027$. This latter value will not go to zero, nor will the solution settle down to a final value; therefore, the algorithm must stop when $(J^* - q^*)/q^*$ is sufficiently small (e.g., less than 0.05 in this case).

**APPENDIX**

**Dual Optimization on a Nonconvex Problem**

We introduced the concept of dual optimization in Appendix 3A and pointed out that when the function to be optimized is convex, and the variables are continuous, then the maximization of the dual function gives the identical result as minimizing the primal function. Dual optimization is also used in solving the unit commitment problem. However, in the unit commitment problem there are variables that must be restricted to two values: 1 or 0. These 1–0 variables
cause a great deal of trouble and are the reason for the difficulty in solving the unit commitment problem.

The application of the dual optimization technique to the unit commitment problem has been given the name "Lagrange relaxation" and the formulation of the unit commitment problem using this method is shown in the text in Section 5.2.3. In this appendix, we illustrate this technique with a simple geometric problem. The problem is structured with 1–0 variables which makes it clearly nonconvex. Its form is generally similar to the form of the unit commitment problems, but that is incidental for now.

The sample problem to be solved is given below. It illustrates the ability of the dual optimization technique to solve the unit commitment problem. Given:

\[
J(x_1, x_2, u_1, u_2) = (0.25x_1^2 + 15)u_1 + (0.255x_2^2 + 15)u_2 
\]

subject to:

\[
\omega = 5 - x_1u_1 - x_2u_2 
\]

and

\[
0 \leq x_1 \leq 10 
\]

\[
0 \leq x_2 \leq 10 
\]

where \(x_1\) and \(x_2\) are continuous real numbers, and:

\[
u_1 = 1 \text{ or } 0 \\
u_2 = 1 \text{ or } 0
\]

Note that in this problem we have two functions, one in \(x_1\) and the other in \(x_2\). The functions were chosen to demonstrate certain phenomena in a dual optimization. Note that the functions are numerically close and only differ by a small, constant amount. Each of these functions is multiplied by a 1–0 variable and combined into the overall objective function. There is also a constraint that combines the \(x_1\) and \(x_2\) variables again with the 1–0 variables. There are four possible solutions.

1. If \(u_1\) and \(u_2\) are both zero, the problem cannot have a solution since the equality constraint cannot be satisfied.
2. If \(u_1 = 1\) and \(u_2 = 0\), we have the trivial solution that \(x_1 = 5\) and \(x_2\) does not enter into the problem anymore. The objective function is 21.25.
3. If \(u_1 = 0\) and \(u_2 = 1\), then we have the trivial result that \(x_2 = 5\) and \(x_1\) does not enter into the problem. The objective function is 21.375.
4. If \(u_1 = 1\) and \(u_2 = 1\), we have a simple Lagrange function of:

\[
L(x_1, x_2, \lambda) = (0.25x_1^2 + 15) + (0.255x_2^2 + 15) + \lambda(5 - x_1 - x_2) 
\]

The resulting optimum is at \(x_1 = 2.5248\), \(x_2 = 2.4752\), and \(\lambda = 1.2642\), with an
objective function value of 33.1559. Therefore, we know the optimum value for this problem; namely, \( u_1 = 1, u_2 = 0, \) and \( x_1 = 5. \)

What we have done, of course, is to enumerate all possible combinations of the 1–0 variables and then optimize over the continuous variables. When there are more than a few 1–0 variables, this cannot be done because of the large number of possible combinations. However, there is a systematic way to solve this problem using the dual formulation.

The Lagrange relaxation method solves problems such as the one above, as follows. Define the Lagrange function as:

\[
\mathcal{L}(x_1, x_2, u_1, u_2, \lambda) = (0.25x_1^2 + 15)u_1 + (0.255x_2^2 + 15)u_2 + \lambda(5 - x_1u_1 - x_2u_2)
\]

As shown in Appendix 3A, we define \( q(\lambda) \) as:

\[
q(\lambda) = \min_{x_1, x_2, u_1, u_2} \mathcal{L}
\]

where \( x_1, x_2, u_1, u_2 \) obey the limits and the 1–0 conditions as before. The dual problem is then to find

\[
q^*(\lambda) = \max_{\lambda \geq 0} q(\lambda)
\]

This is different from the dual optimization approach used in the Appendix 3A because of the presence of the 1–0 variables. Because of the presence of the 1–0 variables we cannot eliminate variables; therefore, we keep all the variables in the problem and proceed in alternating steps as shown in the Appendix 3A.

**Step 1** Pick a value for \( \lambda^k \) and consider it fixed. Now the Lagrangian function can be minimized. This is much simpler than the situation we had before since we are trying to minimize

\[
(0.25x_1^2 + 15)u_1 + (0.255x_2^2 + 15)u_2 + \lambda^k(5 - x_1u_1 - x_2u_2)
\]

where the value of \( \lambda^k \) is fixed.

We can then rearrange the equation above as:

\[
(0.25x_1^2 + 15 - x_1\lambda^k)u_1 + (0.255x_2^2 + 15 - x_2\lambda^k)u_2 + \lambda^k 5
\]

The last term above is fixed and we can ignore it. The other terms are now given in such a way that the minimization of this function is relatively easy. Note that the minimization is now over two terms, each being multiplied by a 1–0 variable. Since these two terms are summed in the Lagrangian, we can minimize the entire function by
minimizing each term separately. Since each term is the product of a function in $x$ and $\lambda$ (which is fixed), and these are all multiplied by the 1–0 variable $u$, then the minimum will be zero (that is with $u = 0$) or it will be negative, with $u = 1$ and the value of $x$ set so that the term inside the parentheses is negative. Looking at the first term, the optimum value of $x_1$ is found by (ignore $u_1$ for a moment):

$$\frac{d}{dx_1} (0.25x_1^2 + 15 - x_1\lambda^k) = 0$$

(5A.9)

If the value of $x_1$ which satisfies the above falls outside the limits of 0 and 10 for $x_1$, we force $x_1$ to the limit violated. If the term in the first brackets

$$(0.25x_1^2 + 15 - x_1\lambda^k)$$

is positive, then we can minimize the Lagrangian by merely setting $u_1 = 0$; otherwise $u_1 = 1$.

Looking at the second term, the optimum value of $x_2$ is found by (again, ignore $u_2$):

$$\frac{d}{dx_2} (0.255x_2^2 + 15 - x_2\lambda^k) = 0$$

(5A.10)

and if the value of $x_2$ which satisfies the above value falls outside the 0 to 10 limits on $x_2$, we set it to the violated limit. Similarly, the term in the second brackets

$$(0.255x_2^2 + 15 - x_2\lambda^k)$$

is evaluated. If it is positive, then we minimize the Lagrangian by making $u_2 = 0$; otherwise $u_2 = 1$. We have now found the minimum value of $\mathcal{L}$ with a specified fixed value of $\lambda^k$.

**Step 2** Assume that the variables $x_1$, $x_2$, $u_1$, $u_2$ found in step 1 are fixed and find a value for $\lambda$ that maximizes the dual function. In this case, we cannot solve for the maximum since $q(\lambda)$ is unbounded with respect to $\lambda$. Instead, we form the gradient of $q(\lambda)$ with respect to $\lambda$ and we adjust $\lambda$ so as to move in the direction of increasing $q(\lambda)$. That is, given

$$\nabla q = \frac{dq}{d\lambda}$$

(5A.11)

which for our problem is

$$\frac{dq}{d\lambda} = 5 - x_1u_1 - x_2u_2$$

(5A.12)
we adjust $\lambda$ according to

$$\lambda^{k+1} = \lambda^k + \frac{dq}{d\lambda} \alpha$$ (5A.13)

where $\alpha$ is a multiplier chosen to move $\lambda$ only a short distance. (This is simply a gradient search method as was introduced in Chapter 3). Note also, that if both $u_1$ and $u_2$ are zero, the gradient will be 5, indicating a positive value telling us to increase $\lambda$. Eventually, increasing $\lambda$ will result in a negative value for

$$(0.25x_1^2 + 15 - x_1\lambda^k)$$

or for

$$(0.255x_2^2 + 15 - x_2\lambda^k)$$

or for both, and this will cause $u_1$ or $u_2$, or both, to be set to 1. Once the value of $\lambda$ is increased, we go back to step 1 and find the new values for $x_1$, $x_2$, $u_1$, $u_2$ again.

The real difficulty here is in not increasing $\lambda$ by too much. In the example presented above, the following scheme was imposed on the adjustment of $\lambda$:

- If $\frac{dq}{d\lambda}$ is positive, then use $\alpha = 0.2$.
- If $\frac{dq}{d\lambda}$ is negative, then use $\alpha = 0.005$.

This lets $\lambda$ approach the solution slowly, and if it overshoots, it backs up very slowly. This is a common technique to make a gradient “behave.”

We must also note that, given the few variables we have, and given the fact that two of them are 1-0 variables, the value of $\lambda$ will not converge to the value needed to minimize the Lagrangian. In fact, it is seldom possible to find a $\lambda^k$ that will make the problem feasible with respect to the equality constraint. However, when we have found the values for $u_1$ and $u_2$ at any iteration, we can then calculate the minimum of $J(x_1, x_2, u_1, u_2)$ by solving for the minimum of

$$[(0.25x_1^2 + 15)u_1 + (0.255x_2^2 + 15)u_2 + \lambda(5 - x_1u_1 - x_2u_2)]$$

using the techniques in Appendix 3A (since the $u_1$ and $u_2$ variables are now known).

The solution to this minimum will be at $x_1 = \overline{x}_1$, $x_2 = \overline{x}_2$ and $\lambda = \overline{\lambda}$. For the case where $u_1$ and $u_2$ are both zero, we shall arbitrarily set this value to a large value (here we set it to 50). We shall call this minimum value
<table>
<thead>
<tr>
<th>Iteration</th>
<th>$\hat{\lambda}$</th>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$q^*$</th>
<th>$\omega$</th>
<th>$\hat{\lambda}$</th>
<th>$\bar{x}_1$</th>
<th>$\bar{x}_2$</th>
<th>$J^*$</th>
<th>$J^* - q^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5.0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>50.0</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
<td>0</td>
<td>0</td>
<td>2.0</td>
<td>1.9608</td>
<td>5.0</td>
<td>5.0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>50.0</td>
<td>9.0</td>
</tr>
<tr>
<td>3</td>
<td>2.0</td>
<td>0</td>
<td>0</td>
<td>4.0</td>
<td>3.9216</td>
<td>10.0</td>
<td>5.0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>50.0</td>
<td>4.0</td>
</tr>
<tr>
<td>4</td>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>6.0</td>
<td>5.8824</td>
<td>15.0</td>
<td>5.0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>50.0</td>
<td>2.33</td>
</tr>
<tr>
<td>5</td>
<td>4.0</td>
<td>1</td>
<td>1</td>
<td>8.0</td>
<td>7.8431</td>
<td>18.3137</td>
<td>—10.8431</td>
<td>1.2624</td>
<td>2.5248</td>
<td>2.4752</td>
<td>33.1559</td>
<td>0.8104</td>
</tr>
<tr>
<td>6</td>
<td>3.9458</td>
<td>1</td>
<td>1</td>
<td>7.8916</td>
<td>7.7368</td>
<td>18.8958</td>
<td>—10.6284</td>
<td>1.2624</td>
<td>2.5248</td>
<td>2.4752</td>
<td>33.1559</td>
<td>0.7546</td>
</tr>
<tr>
<td>7</td>
<td>3.8926</td>
<td>1</td>
<td>0</td>
<td>7.7853</td>
<td>7.6326</td>
<td>19.3105</td>
<td>—2.7853</td>
<td>2.5</td>
<td>5.0</td>
<td>—</td>
<td>21.25</td>
<td>0.1004</td>
</tr>
<tr>
<td>8</td>
<td>3.8787</td>
<td>1</td>
<td>0</td>
<td>7.7574</td>
<td>7.6053</td>
<td>19.3491</td>
<td>—2.7574</td>
<td>2.5</td>
<td>5.0</td>
<td>—</td>
<td>21.25</td>
<td>0.0982</td>
</tr>
</tbody>
</table>
\( J^*(x_1, x_2, u_1, u_2) \) and we shall observe that it starts out with a large value, and decreases, while the dual value \( q^*(\lambda) \) starts out with a value of zero, and increases. Since there are 1–0 variables in this problem, the primal values and the dual values never become equal. The value \( J^* - q^* \) is called the duality gap and we shall call the value

\[
\frac{J^* - q^*}{q^*}
\]

the relative duality gap.

The presence of the 1–0 variables causes the algorithm to oscillate around a solution with one or more of the 1–0 variables jumping from 1 to 0 to 1, etc. In such cases, the user of the Lagrange relaxation algorithm must stop the algorithm, based on the value of the relative duality gap.

The iterations starting from \( \lambda = 0 \) are shown in Table 5.7. The table shows eight iterations and illustrates the slow approach of \( \lambda \) toward the threshold when both of the 1–0 variables flip from 0 to 1. Also note that \( \omega \) became negative and the value of \( \lambda \) must now be decreased. Eventually, the optimal solution is reached and the relative duality gap becomes small. However, as is typical with the dual optimization on a problem with 1–0 variables, the solution is not stable and if iterated further it exhibits further changes in the 1–0 variables as \( \lambda \) is adjusted. Both the \( q^* \) and \( J^* \) values and the relative duality gap are shown in Table 5.7.

**PROBLEMS**

5.1 Given the unit data in Tables 5.8 and 5.9, use forward dynamic-programming to find the optimum unit commitment schedules covering the 8-h period. Table 5.9 gives all the combinations you need, as well as the operating cost for each at the loads in the load data. A "\( \times \)" indicates that a combination cannot supply the load. The starting conditions are: at the beginning of the first period units 1 and 2 are up, units 3 and 4 are down and have been down for 8 h.

5.2 Table 5.10 presents the unit characteristics and load pattern for a five-unit, four-time-period problem. Each time period is 2 h long. The input–output characteristics are approximated by a straight line from min to max generation, so that the incremental heat rate is constant. Unit no-load and start-up costs are given in terms of heat energy requirements.

a. Develop the priority list for these units and solve for the optimum unit commitment. Use a strict priority list with a search range of three (\( X = 3 \)) and save no more than three strategies (\( N = 3 \)). Ignore min up-/min down-times for units.

b. Solve the same commitment problem using the strict priority list with \( X = 3 \) and \( N = 3 \) as in part a, but obey the min up/min down time rules.
TABLE 5.8 Unit Commitment Data for Problem 5.1

<table>
<thead>
<tr>
<th>Unit</th>
<th>Max (MW)</th>
<th>Min (MW)</th>
<th>Incremental Heat Rate (Btu/kWh)</th>
<th>No-Load Energy Input (MBtu/h)</th>
<th>Start-Up Energy (MBtu)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500</td>
<td>70</td>
<td>9950</td>
<td>300</td>
<td>800</td>
</tr>
<tr>
<td>2</td>
<td>250</td>
<td>40</td>
<td>10200</td>
<td>210</td>
<td>380</td>
</tr>
<tr>
<td>3</td>
<td>150</td>
<td>30</td>
<td>11000</td>
<td>120</td>
<td>110</td>
</tr>
<tr>
<td>4</td>
<td>150</td>
<td>30</td>
<td>11000</td>
<td>120</td>
<td>110</td>
</tr>
</tbody>
</table>

Load data (all time periods = 2 h):

<table>
<thead>
<tr>
<th>Time Period</th>
<th>Load (MW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>600</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
</tr>
<tr>
<td>3</td>
<td>700</td>
</tr>
<tr>
<td>4</td>
<td>950</td>
</tr>
</tbody>
</table>

Start-up and shut-down rules:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Minimum Up Time (h)</th>
<th>Minimum Down Time (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Fuel cost = 1.00 R/MBtu.

TABLE 5.9 Unit Combinations and Operating Cost for Problem 5.1

<table>
<thead>
<tr>
<th>Combination</th>
<th>Operating Cost (R/h)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Load 600 MW</td>
</tr>
<tr>
<td>A</td>
<td>6505</td>
</tr>
<tr>
<td>B</td>
<td>6649</td>
</tr>
<tr>
<td>C</td>
<td>6793</td>
</tr>
</tbody>
</table>

1 = up; 0 = down.

c. (Optional) Find the optimum unit commitment without use of a strict priority list (i.e., all 32 unit on/off combinations are valid). Restrict the search range to decrease your effort. Obey the min up-/min down-time rules.

When using a dynamic-programming method to solve a unit commitment problem with minimum up- and down-time rules, one must save an additional piece of information at each state, each hour. This information
### TABLE 5.10 The Unit Characteristic and Load Pattern for Problem 5.2

<table>
<thead>
<tr>
<th>Unit</th>
<th>Net Full-Load Heat Rate (MBtu/h)</th>
<th>Incremental Heat Rate Min (MBtu)</th>
<th>No-Load Cost Min (MBtu)</th>
<th>Start-Up Cost Min (MBtu)</th>
<th>Up/Down Time (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>9900</td>
<td>50</td>
<td>220</td>
<td>400</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>10100</td>
<td>15</td>
<td>80</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>10800</td>
<td>15</td>
<td>60</td>
<td>105</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>11900</td>
<td>5</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>12140</td>
<td>5</td>
<td>34</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Load Pattern

<table>
<thead>
<tr>
<th>Hours</th>
<th>Load (MW)</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2</td>
<td>250</td>
<td>1. Initially (prior to hour 1), only unit 1 is on and has been on for 4 h.</td>
</tr>
<tr>
<td>3-4</td>
<td>320</td>
<td>2. Ignore losses, spinning reserve, etc. The only requirement is that the generation be able to supply the load.</td>
</tr>
<tr>
<td>5-6</td>
<td>110</td>
<td>3. Fuel costs for all units may be taken as 1.40 $/MBtu</td>
</tr>
<tr>
<td>7-8</td>
<td>75</td>
<td></td>
</tr>
</tbody>
</table>

simply tells us whether any units are ineligible to be shut down or started up at that state. If such units exist at a particular state, the transition cost, \( S_{\text{cost}} \), to a state that violates the start-up/shut-down rules should be given a value of infinity.

### 5.3 Lagrange Relaxation Problem

Given the three generating units below:

\[
F_1(P_1) = 30 + 10P_1 + 0.002P_1^2 \quad \text{and} \quad 100 < P_1 < 600
\]
\[
F_2(P_2) = 20 + 8P_2 + 0.0025P_2^2 \quad \text{and} \quad 100 < P_2 < 400
\]
\[
F_3(P_3) = 10 + 6P_3 + 0.005P_3^2 \quad \text{and} \quad 50 < P_3 < 200
\]

#### Load:

<table>
<thead>
<tr>
<th>( t )</th>
<th>( P_{\text{load}} ) (MW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>300</td>
</tr>
<tr>
<td>2</td>
<td>500</td>
</tr>
<tr>
<td>3</td>
<td>1100</td>
</tr>
<tr>
<td>4</td>
<td>400</td>
</tr>
</tbody>
</table>

No start-up costs, no minimum up- or down-time constraints.
a. Solve for the unit commitment by conventional dynamic programming.

b. Set up and carry out four iterations of the Lagrange relaxation method. Let the initial values of $\lambda^t$ be zero for $t = 1 \ldots 4$.

c. Resolve with the added condition that the third generator has a minimum up time of 2 h.

FURTHER READING

Some good introductory references to the unit commitment problem are found in references 1–3. A survey of the state-of-the-art (as of 1975) of unit commitment solutions is found in reference 4. References 5 and 6 provide a good look at two commercial unit commitment programs in present use.

References 7–11 deal with unit commitment as an integer-programming problem. Much of the pioneering work in this area was done by Garver (reference 7), who also sounded a note of pessimism in a discussion of reference 8, written together with Happ in 1968. Further research (references 9–11) has refined the unit commitment solution by integer programming but has never really overcome the Garver–Happ limitations presented in the 1968 discussion, thus leaving dynamic programming and Lagrange relaxation as the only viable solution techniques to large-scale unit commitment problems.

The reader should see references 12 and 13 for a discussion of valve-point loading and for a thorough development of economic dispatch via dynamic programming.

Reference 14 provides the reader with a good overview of unit commitment scheduling. References 15, 16, and 17 are recommended for an understanding of the Lagrange relaxation method, while references 18–21 cover some of the special problems encountered in unit commitment scheduling.

7. Garver, L. L., “Power Generation Scheduling by Integer Programming—


