TABLE 16.3
Characteristics of batch scheduling and planning problems

DETERMINE	GIVEN
What Product amounts: lot sizes, batch sizes	**Product requirements** Horizon, demands, starting and ending inventories
When Timing of specific operations, run lengths	**Operational steps** Precedence order Resource utilization
Where Sites, units, equipment items	**Production facilities** Types, capacities
How Resource types and amounts	**Resource limitations** Types, amounts, rates

Source: Pekny and Reklaritis (1998).

2. The bottom-up approach, which develops detailed plant simulation and optimization models, optimizes them, and translates the results from the simulations and optimization into practical operating heuristics. This approach often leads to large models with many variables and equations that are difficult to solve quickly using rigorous optimization algorithms.

Table 16.3 categorizes the typical problem statement for the manufacturing scheduling and planning problem. In a batch campaign or run, comprising smaller runs called lots, several batches of product may be produced using the same recipe. To optimize the production process, you need to determine

1. The recipe that satisfies product quality requirements.
2. The production rates needed to fulfill the timing requirements, including any precedence constraints.
3. Operating variables for plant equipment that are subject to constraints.
4. Availability of raw material inventories.
5. Availability of product storage.
6. The run schedule.
7. Penalties on completing a production step too soon or too late.

EXAMPLE 16.2 MULTIPRODUCT BATCH PLANT SCHEDULING

Batch operations such as drying, mixing, distillation, and reaction are widely used in producing food, pharmaceuticals, and specialty products (e.g., polymers). Scheduling of operations as described in Table 16.3 is crucial in such plants. A principal feature of batch plants (Ku and Karimi, 1987) is the production of multiple products using the

same set of equipment. Good industrial case studies of plant scheduling include those by Bunch et al. (1998), McDonald (1998), and Schulz et al. (1998). For example, Schulz et al. described a polymer plant that involved four process steps (preparation, reaction, mixing, and finishing) using different equipment in each step. When products are similar in nature, they require the same processing steps and hence pass through the same series of processing units; often the batches are produced sequentially. Such plants are called multiproduct plants. Because of different processing time requirements, the total time required to produce a set of batches (also called the makespan or cycle time) depends on the sequence in which they are produced. To maximize plant productivity, the batches should be produced in a sequence that minimizes the makespan. The plant schedule corresponding to such a sequence can then be represented graphically in the form of a Gantt chart (see the following discussion and Figure E16.2b). The Gantt chart provides a timetable of plant operations showing which products are produced by which units and at what times. Chapter 10 discusses a single-unit sequencing problem.

In this example we consider four products ($p1, p2, p3, p4$) that are to be produced as a series of batches in a multiproduct plant consisting of three batch reactors in series (Ku and Karimi, 1992); see Figure E16.2a. The processing times for each batch reactor and each product are given in Table E16.2. Assume that no intermediate storage is available between the processing units. If a product finishes its processing on unit k and unit $k + 1$ is not free because it is still processing a previous product, then the completed product must be kept in unit k, until unit $k + 1$ becomes free. As an example, product $p1$ must be held in unit 1 until unit 2 finishes processing $p3$. When a product finishes processing on the last unit, it is sent immediately to product storage. Assume that the times required to transfer products from one unit to another are negligible compared with the processing times.

The problem for this example is to determine the time sequence for producing the four products so as to minimize the makespan. Assume that all the units are initially

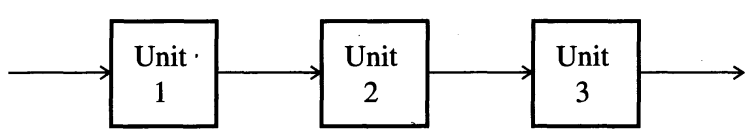

FIGURE E16.2a
Multiproduct plant.

TABLE E16.2
Processing times (h) of products

	Products			
Units	*p1*	*p2*	*p3*	*p4*
1	3.5	4.0	3.5	12.0
2	4.3	5.5	7.5	3.5
3	8.7	3.5	6.0	8.0

empty (initialized) at time zero and the manufacture of any product can be delayed an arbitrary amount of time by holding it in the previous unit.

Solution. Let N be the number of products and M be the number of units in the plant. Let $C_{j,k}$ (called completion time) be the "clock" time at which the jth product in the sequence leaves unit k after completion of its processing, and let $\tau_{j,k}$ be the time required to process the jth product in the sequence on unit k (See Table E16.2). The first product goes into unit 1 at time zero, so $C_{1,0} = 0$. The index j in $\tau_{j,k}$ and $C_{j,k}$ denotes the position of a product in the sequence. Hence $C_{N,M}$ is the time at which the last product leaves the last unit and is the makespan to be minimized. Next, we derive the set of constraints (Ku and Karimi, 1988; 1990) that interrelate the $C_{j,k}$. First, the jth product in the sequence cannot leave unit k until it is processed, and in order to be processed on unit k, it must have left unit $k-1$. Therefore the clock time at which it leaves unit k (i.e., $C_{j,k}$) must be equal to or after the time at which it leaves unit $k-1$ plus the processing time in k. Thus the first set of constraints in the formulation is

$$C_{j,k} \geq C_{j,k-1} + \tau_{j,k} \quad j = 1, \ldots, N \quad k = 2, \ldots, M \tag{a}$$

Similarly, the jth product cannot leave unit k until product $(j-1)$ has been processed and transferred:

$$C_{j,k} \geq C_{j-1,k} + \tau_{j,k} \quad j = 1, \ldots, N \quad k = 1, \ldots, M \tag{b}$$

Set $C_{0,k} = 0$. Finally the jth product in the sequence cannot leave unit k until the downstream unit $k+1$ is free [i.e., product $(j-1)$ has left]. Therefore

$$C_{j,k} \geq C_{j-1,k+1} \quad j = 1, \ldots, N \quad k = 1, \ldots, M-1 \tag{c}$$

Although Equations (a)–(c) represent the complete set of constraints, some of them are redundant. From Equation (a) $C_{j,k} \geq C_{j,k-1} + \tau_{i,k}$ for $k \geq 2$. But from Equation (c), $C_{j,k-1} \geq C_{j-1,k}$, hence $C_{j,k} \geq C_{j-1,k} + \tau_{i,k}$ for $k = 2, M$. In essence, Equations (a) and (c) imply Equations (b) for $k = 2, M$, so Equations (b) for $k = 2, M$ are redundant.

Having derived the constraints for completion times, we next determine the sequence of operations. In contrast to the $C_{j,k}$, the decision variables here are discrete (binary). Define $X_{i,j}$ as follows. $X_{i,j} = 1$ if product i (product with label pi) is in slot j of the sequence, otherwise it is zero. So $X_{3,2} = 1$ means that product $p3$ is second in the production sequence, and $X_{3,2} = 0$ means that it is not in the second position. The overall integer constraint is

$$X_{1,j} + X_{2,j} + X_{3,j} + X_{4,j} + \cdots + X_{N,j} = 1 \quad j = 1, \ldots, \tag{d}$$

Similarly every product should occupy only one slot in the sequence:

$$X_{i,1} + X_{i,2} + X_{i,3} + X_{i,4} + \cdots + X_{i,N} = 1 \quad i = 1, \ldots, N \tag{e}$$

The $X_{i,j}$ that satisfy Equations (d) and (e) always give a meaningful sequence. Now we must determine the clock times $t_{i,k}$ for any given set of $X_{i,j}$. If product pi is in slot j, then $t_{j,k}$ must be $\tau_{i,k}$ and $X_{i,j} = 1$ and $X_{i,1} = X_{i,2} = \cdots = X_{i,j-1} = X_{i,j+1} = \cdots = X_{i,N} = 0$, therefore we can use $X_{i,j}$ to pick the right processing time representing $t_{j,k}$ by imposing the constraint.

$$\tau_{j,k} = X_{1,j}t_{i,k} + X_{2,j}t_{2,k} + X_{3,j}t_{3,k} + \cdots + X_{N,j}t_{N,k} \qquad j = 1, \ldots, N \qquad k = 1, \ldots, M \tag{f}$$

To reduce the number of constraints, we substitute $\tau_{j,k}$ from Equation (f) into Equations (a) and (b) to obtain the following formulation (Ku and Karimi, 1988).

Minimize: C_{NM}

Subject to: Equations (c), (d), (e) and

$$C_{i,k} \geq C_{i,k-1} + \sum_{j=1}^{N} X_{j,i} t_{j,k} \quad i = 1, \ldots, N \quad k = 2, \ldots, M \qquad (g)$$

$$C_{i,1} \geq C_{i-1,1} + \sum_{j=1}^{N} X_{j,i} t_{j,i} \quad i = 1, \ldots, N \qquad (h)$$

$C_{i,k} \geq 0$ and $X_{i,j}$ binary

Because the preceding formulation involves binary ($X_{i,j}$) as well as continuous variables ($C_{i,k}$) and has no nonlinear functions, it is a mixed-integer linear programming (MILP) problem and can be solved using the GAMS MIP solver.

Solving for the optimal sequence using Table E16.2, we obtain $X_{1,1} = X_{2,4} = X_{3,2} = X_{4,3} = 1$. This means that $p1$ is in the first position in the optimal production sequence, $p2$ in the fourth, $p3$ in the second, and $p4$ in the third. In other words, the optimal sequence is in the order $p1$-$p3$-$p4$-$p2$. In contrast to the $X_{i,j}$, we must be careful in interpreting the $C_{i,k}$ from the GAMS output, because $C_{j,k}$ really means the time at which the jth product in the sequence (and not product pi) leaves unit k. Therefore $C_{2,3} = 23.3$ means that the second product (i.e., $p3$) leaves unit 3 at 23.3 h. Interpreting the others in this way, the schedule corresponding to this production sequence is conveniently displayed in form of a Gantt chart in Figure E16.2b, which shows the status of the units at different times. For instance, unit 1 is processing $p1$ during [0, 3.5] h. When $p1$ leaves unit 1 at $t = 3.5$ h, it starts processing $p3$. It processes $p3$ during [3.5, 7] h. But as seen from the chart, it is unable to discharge $p3$ to unit 2, because unit 2 is still processing $p1$. So unit 1 holds $p3$ during [7, 7.8] h. When unit 2 discharges $p3$

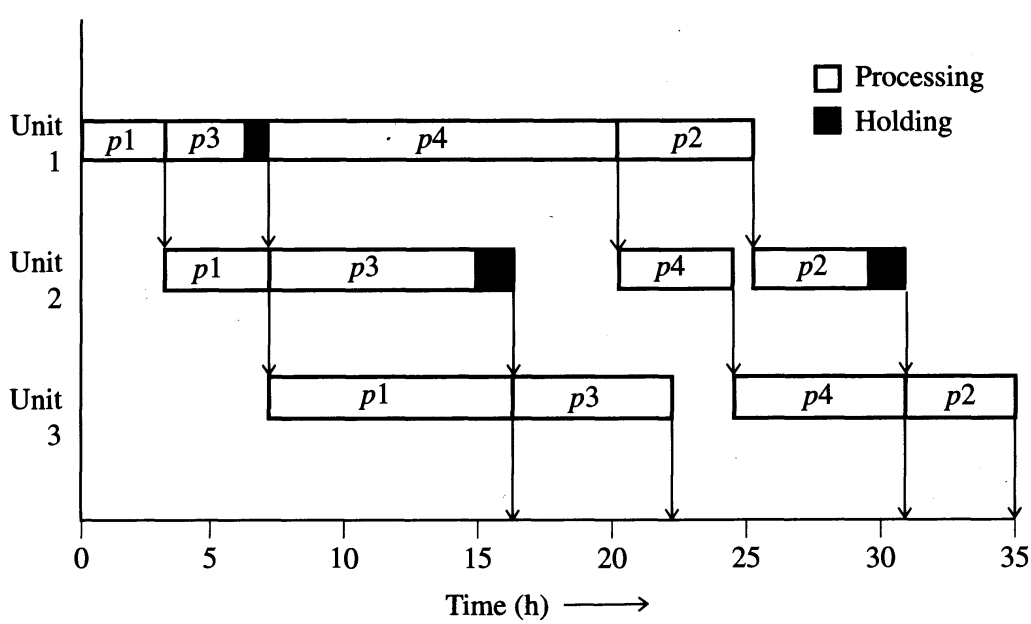

FIGURE E16.2b
Gantt chart for the optimal multiproduct plant schedule.

to unit 3 at 16.5 h, unit 1 is still processing $p4$, therefore unit 2 remains idle during [16.5, 19.8] h. It is common in batch plants to have units blocked due to busy downstream units or units waiting for upstream units to finish. This happens because the processing times vary from unit to unit and from product to product, reducing the time utilization of units in a batch plant. The finished batches of $p1$, $p3$, $p4$, and $p2$ are completed at times 16.5 h, 23.3 h, 31.3 h, and 34.8 h. The minimum makespan in 34.8 h.

This problem can also be solved by a search method (see Chapter 10). Because the order of products cannot be changed once they start through the sequence of units, we need only determine the order in which the products are processed. This is the same problem as considered in Section 10.5.2, to illustrate the workings of tabu search. Using the notation of that section, let

$$\mathbf{P} = (p(1), p(2), \ldots, p(N))$$

be a permutation or sequence in which to process the jobs, where $p(j)$ is the index of the product in position j of the sequence. To evaluate the makespan of a sequence, we proceed as in Equations (a)–(c) of the mixed-integer programming version of the problem. Let $C_{j,k}$ be the completion time of product $p(j)$ on unit k. If product $p(j)$ does not have to wait for product $p(j-1)$ to finish its processing on unit k, then

$$C_{j,k} = C_{j,k-1} + t_{p(j),k} \qquad (i)$$

If it does have to wait, then

$$C_{j,k} = C_{j-1,k} + t_{p(j),k} \qquad (j)$$

Hence $C_{j,k}$ is the larger of these two values:

$$C_{j,k} = \max(C_{j-1,k} + t_{p(j),k}, C_{j,k-1} + t_{p(j),k}) \qquad (k)$$

This equation is solved first for $C_{1,k}$ for $k = 1, \ldots, M$, then for $C_{2,k}$ for $k = 1, 2, \ldots, M$, and so on. The objective function is simply the completion time of the last job:

$$f(\mathbf{P}) = C_{N,M} \qquad (l)$$

In a four-product problem, there are only $4! = 24$ possible sequences, so you can easily write a simple FORTRAN or C program to evaluate the makespan for an arbitrary sequence, and then call it 24 times and choose the sequence with the smallest makespan. For larger values of N, one can apply the tabu search algorithm described in Section 10.5.2. Other search procedures (e.g., evolutionary algorithms or simulated annealing), can also be developed for this problem. Of course, these algorithms do not guarantee that an optimal solution will be found. On the other hand, the time required to solve the mixed-integer programming formulation grows rapidly with N, so that approach eventually becomes impractical. This illustrates that you may be able to develop a simple but effective search method yourself, and eliminate the need for MILP optimization software.

The classical solution to a scheduling problem assumes that the required information is known at the time the schedule is generated and that this a priori scheduling remains fixed for a planning period and is implemented on the plant equipment. Although this methodology does not compensate for the many external disturbances and internal disruptions that occur in a real plant, it is still the strategy most commonly found in industrial practice. Demand fluctuations, process devia-