generate X sequences we have a probability of $1-(1-p)^X$ of obtaining the optimal assignment of tasks to workstations. If we are willing to assume a distribution for the objective over the possible sequences, we can even extend these results to make probabilistic statements concerning how close we are to the optimum by using order statistics.

At this point you may be wondering why we would bother to generate random sequences, since we may needlessly reexamine the same sequence and waste computational resources generating random numbers. Why not just order the set of possible sequences and try the first X unique sequences? Would this not give us a better chance of finding the optimum? In theory yes, it would for our simplistic model. In fact, we will shortly look at such an approach. In practice, however, there is likely to be a large number of optimal or near optimal solutions. Many of these solutions will have similar-looking workstations or sequences. By generating random sequences we may "jump around" the set of possible sequences and are likely to find at least one of these good ones. If we start exploring by always looking at a minimally different sequence each time, we will generate very similar sequences; perhaps we may find all good sequences, but it is just as likely we will find all bad sequences.

EXAMPLE 2.1

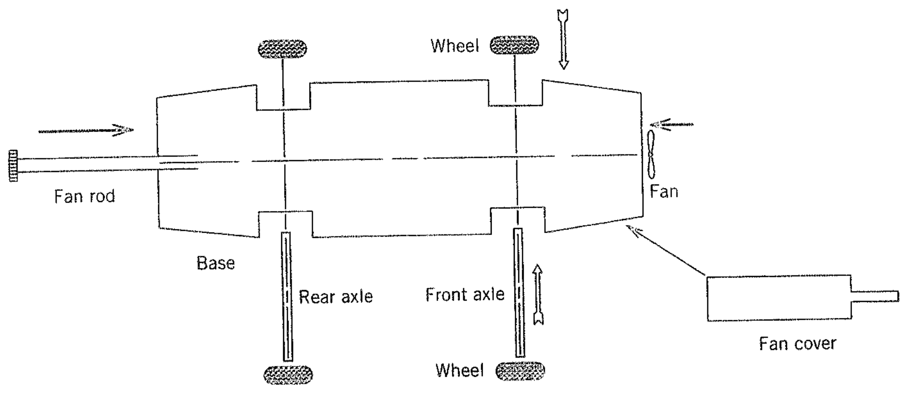
Consider the assembly of the spring-activated toy car shown in Figure 2.4. To take advantage of a large local labor pool of persons wishing part-time day employment and part-time second jobs in the evening, two 4 hour-shifts will be used, 4 days a week for assembly. Each shift receives two 10-minute breaks. Marketing estimates have been used to obtain a planned production rate of 1500 units per week. With holidays, vacations, and other events, the plant averages only 4 working days per week. The set of tasks, times, and precedence constraints are given in Table 2.2. No zoning constraints exist.

Table 2.2 Toy Car Assembly Tasks

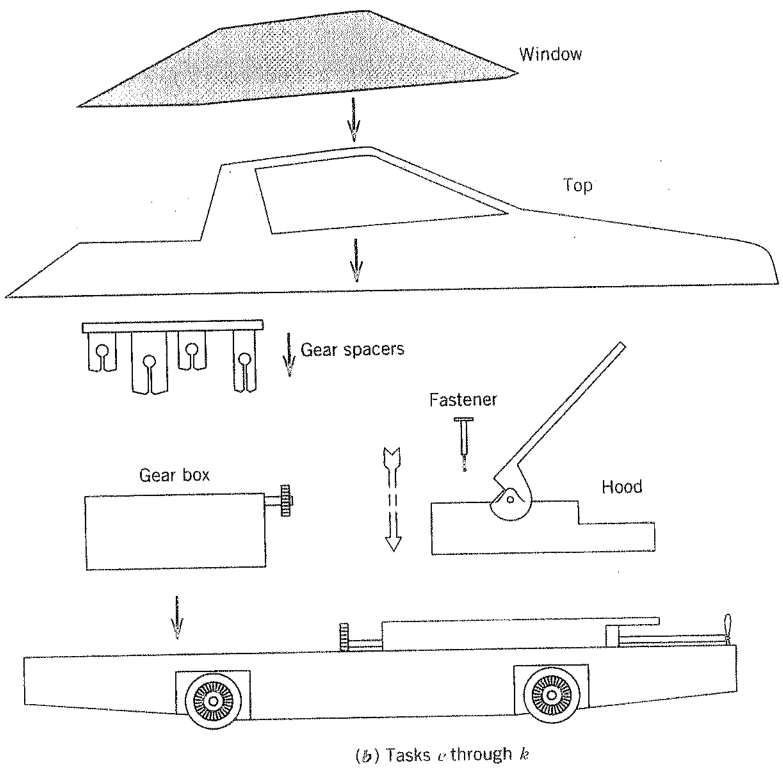
Task	Activity	Assembly Time (seconds)	Immediate Predecessors
a	Insert front axle/wheels	20	—
b	Insert fan rod	6	*a*
c	Insert fan rod cover	5	*b*
d	Insert rear axle/wheels	21	—
e	Insert hood to wheel frame	8	—
f	Glue windows to top	35	—
g	Insert gear assembly	15	*c,d*
h	Insert gear spacers	10	*g*
i	Secure front wheel frame	15	*e,h*
j	Insert engine	5	*c*
k	Attach top	46	*f,i,j*
l	Add decals	16	*k*

Solution

The precedence relations are easier to visualize in graphical form. Figure 2.5 summarizes the table. Each task is represented by a node. Associated task times



(*a*) Tasks *a* through *d*



(*b*) Tasks *e* through *k*

Figure 2.4 Toy car for Example 2.1.

are shown with each node. Directed arcs between nodes indicate immediate predecessor relations.

To illustrate the COMSOAL approach, we will generate a random sequence. First, we must determine the cycle time.

$$C = \frac{1 \text{ week}}{1500 \text{ units}} \times 4\frac{\text{days}}{\text{week}} \times 2\frac{\text{shifts}}{\text{day}} \times 220\frac{\text{minutes}}{\text{shift}} = 1.17\frac{\text{minutes}}{\text{unit}}$$

To meet demand we will use $C = 70$ seconds. Reference to Figure 2.5 indicates four potential first tasks (a, d, e, or f). We generate a random number uniformly distributed between 0 and 1 ($U[0, 1]$). Our outcome is $R = .34$. Thus, since R is

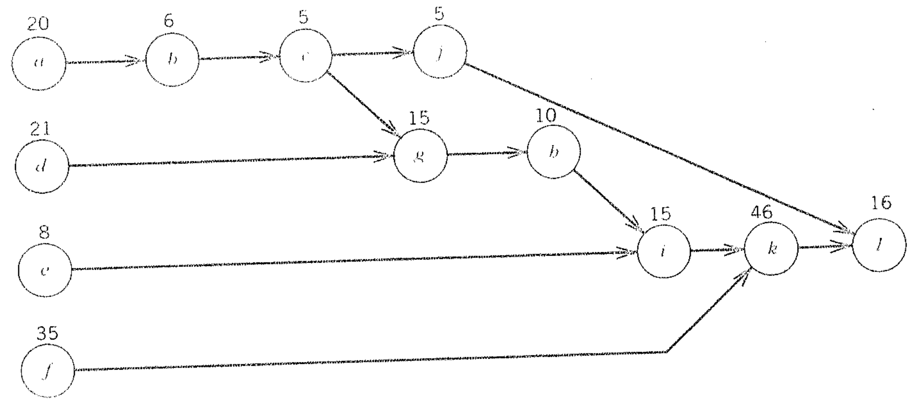


Figure 2.5 Model car precedence structure.

in the second quartile and d is the second task, d is placed first. We can continue in this fashion until a schedule is complete. Table 2.3 summarizes the results of our random-sequence generation. Each row represents the assignment of one task or the opening of a new station because list F is empty. After assigning all 12 tasks, we obtain a feasible solution. As indicated in Table 2.3, our first solution requires four stations. A quick check shows our lower bound to be $K^0 = \lceil \sum_{r=a}^{l} t_r / C \rceil = \lceil 202/70 \rceil = 3$. Thus, better solutions may exist. ▩

After reading the next two sections the reader may wish to consider how the basic COMSOAL procedure could be improved. Several end-of-chapter exercises explore this issue.

Table 2.3 Single COMSOAL Sequence Results

Step	List A	List B	List F	U (0,1)	Selected Task	Station (idle time)
1	*a* through *l*	*a,d,e,f*	*a,d,e,f*	.34	*d*	1(49)
2	*a* through *l*, -*d*	*a,e,f*	*a,e,f*	.83	*f*	1(14)
3	*a,b,c,e,g,h,i,j,k,l*	*a,e*	*e*	—	*e*	1(6)
4	*a,b,c,g,h,i,j,k,l*	*a*	—	Open station		
4	*a,b,c,g,h,i,j,k,l*	*a*	*a*	—	*a*	2(50)
5	*b,c,g,h,i,j,k,l*	*b*	*b*	—	*b*	2(44)
6	*c,g,h,i,j,k,l*	*c*	*c*	—	*c*	2(39)
7	*g,h,i,j,k,l*	*g,j*	*g,j*	.21	*g*	2(24)
8	*h,i,j,k,l*	*j,h*	*h,j*	.42	*h*	2(14)
9	*i,j,k,l*	*i,j*	*j*	—	*j*	2(9)
10	*i,k,l*	*i*	—	Open station		
10	*i,k,l*	*i*	*i*	—	*i*	3(55)
11	*k,l*	*k*	*k*	—	*k*	3(9)
12	*l*	*l*	—	Open station		
12	*l*	*l*	*l*	—	*l*	4(54)