Chapter 18

Automated Storage and Retrieval

18.1 Introduction

Much of the time that material is in a plant, it is being moved or stored. In this chapter, the dynamics of how an automated storage and retrieval system (AS/RS) organizes and maintains an inventory are examined. An AS/RS system provides the following benefits: space efficient storage of materials, high speed controlled transportation of materials, and real-time inventory control. Thus an AS/RS system helps reduce inventory, labor, floor space, and material control costs.

A typical AS/RS system has several components as is shown in Figure 18-1. A storage / retrieval (S/R) machine places pallets (or another standard carrier) having one or more standard sizes in a high rise rack system. A rack consists of a matrix of storage locations. Racks are separated by aisles. There is one S/R machine per aisle. An S/R machine moves in the horizontal direction on a track located in the floor of an aisle and rises vertically via an imbedded mechanism. Typically, vertical speed is about 1/3 of horizontal speed.

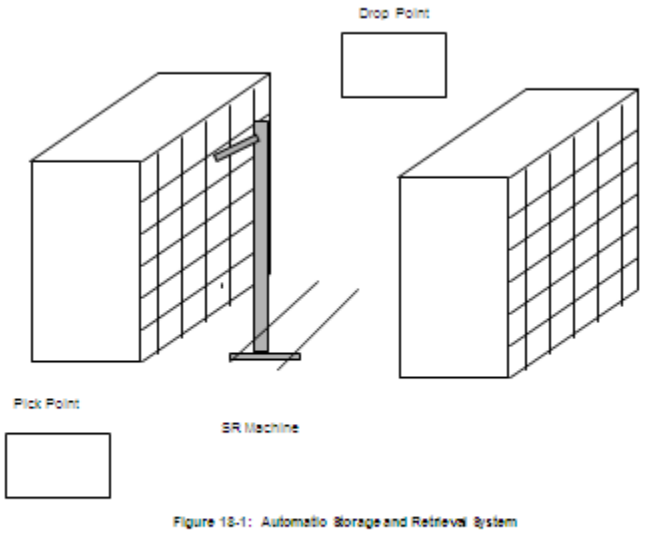

Figure 18-1: Automatic Storage and Retrieval System

Items to be stored arrive to a pick point. Retrieved items are transported by a the S/R machine to a drop point.

A computer-based control system is an important part of an AS/RS system. The computer keeps track of the exact location of all items in the racks. The control system directs the movement of the S/R machine by providing timely instructions concerning what items to retrieve or store in the racks. These instructions are in response to external requests for storage and retrieval.

The computer-based control system can be tested using simulation. Alternative rack sizes can be assessed. Various storage and retrieval strategies can be compared. In this way, movement of the S/R machine when it is empty, as well as the capital investment in racks, can be minimized.

18.2 Points Made in the Case Study

The control algorithm for an automated system, such as an AS/RS, can be included in a simulation model. The control algorithm may be coded in a general purpose programming language and interfaced within the model.

A simulation model can consist of multiple processes. These processes can share the same resources. In this application, an S/R machine, represented by a resource, is used by both an inventory storage process and by an inventory retrieval process.

A resource may have multiple BUSY states. Each BUSY state indicates that the resource is occupied in a unique way. In this application, each rack space is either empty or full of a particular type of item. BUSY states correspond to item types.

Sometimes it is necessary to select a resource to employ from a set of resources with similar or identical characteristics. A criteria for making the selection must be specified. The set of resources cannot be modeled as units of a single resource since the state of each individual resource must be tracked. In this application, the state of each individual rack storage space is important. The model must use the AS/RS control logic to select from among the storage spaces in the IDLE state when a carrier is stored. In the same way, the model must select from among items of the same type when a retrieval is required.

18.3 The Case Study

A particular manufacturing plant assembles finished goods from subassemblies that are produced in another area of the plant or delivered to the plant from external suppliers. A subassembly consists of component parts that have been joined together. Subassemblies arrive to the area preceding the final assembly operation as completed or as delivered.

Thus, a buffer before final assembly is required. The buffer is implemented using an AS/RS system. The storage area consists of two rectangular racks of bins with an aisle between them. Each bin holds one subassembly, which may be of one of four types. Subassemblies are delivered to a pick point where they are picked up one at a time by the S/R machine and placed in the nearest, with respect to S/R machine movement time, available bin.

The final assembly process requests subassemblies one at time. Each request specifies a particular type of subassembly. The S/R machine retrieves the nearest, with respect to its movement time, subassembly of the requested type and places it at the drop point. The subassembly is subsequently moved from the drop point to the final assembly area.

To minimize unproductive movements, the S/R machine remains at the bin in which it last placed a subassembly or at the drop point when it completes a task and becomes idle.

Subassemblies arrive from 6:00 A.M. to 2:00 P.M. each day. The final assembly process operates from 8:00 A.M. to 4:00 P.M. each day or until all of the subassemblies in the AS/RS have been consumed.

18.3.1 Define the Issues and Solution Objective

A fundamental issue in the AS/RS control algorithm is into what free bin to store a subassembly and from what occupied bin to retrieve a subassembly. The algorithm to select a bin is an intrinsic component of the operation of the AS/RS system and must be included in the simulation model. Each bin is in one of nine states:

1. Idle
2-5. Occupied with a subassembly of type one, two , three, or four
6-9. Occupied with a subassembly of type one, two, three, or four that is committed to the final assembly process

The selected bin is the one in the specified state that requires the least travel time for the S/R machine. The idle S/R machine waits at the pick point or at the last bin in which a subassembly was stored.

The S/R machine moves 6 feet per second horizontally and 2 foot per second vertically. Each bin is 1 foot square including the rack structure. Thus, the time to reach any bin is the sum of the number of bins traversed horizontally * 1/6 second per bin and the number traversed vertically * 1/2 second per bin. This sum is illustrated for a rack 8 bins high and 7 bins long in Figure 18-2 assuming the S/R machine starts at the pick point which is to the left of the bin structure on the floor level.

The search for a bin is performed by the AS/RS control software. Bins are searched in order of the movement time values shown in Figure 18-2, least to greatest until a bin in the state desired is located. Among bins with the same value, those closer to the floor are preferred.

The same search strategy can be applied if the S/R machine is waiting at a particular bin. The control algorithm searches in four directions, one at a time. These directions are:

1. Right and up from the current location, as shown in Figure 18-2.
2. Right and down from the current location.
3. Left and up from the current location.
4. Left and down from the current location.

After all the searches have been completed, the storage location nearest the S/R machine with respect to movement time is chosen.

The search strategy is worthy of discussion. Consider the movement time of $7/6^{th}$ second. This is the movement time to the seventh bin in the first row, the fourth bin in the second row, and the first bin in the third row. Thus, the bin search order for the time $7/6^{th}$ second is as listed previously.

Consider searching up and right from the current SR machine location in general. Bins are examined in order of movement time, least to greatest, until a bin in the desired state is found. Bins with equal movement times are searched as follows. The search begins at the bin to the right of the current location and procedes to the bin in the next higher row and three columns preceding (since the vertical movement time is three times the horizontal movement time). This part of the search stops when either a bin in the desired state is found or the next bin to be examined would be to the left of the current location of the SR machine or the next bin to be examined does not exist.

It takes the S/R machine 6 seconds to store or retrieve a subassembly from a bin. The time between requests to store a subassembly is 20 seconds, exponentially distributed, as is the time between requests to retrieve a subassembly.

Two configurations of the AS/RS system have been proposed. In the first, each rack has 180 bins, 10 bins high and 18 bins long. In the other, each rack has 225 bins, 9 bins high and 25 bins long. Thus, extra storage space requires more floor space. The problem is to select between these two alternatives.

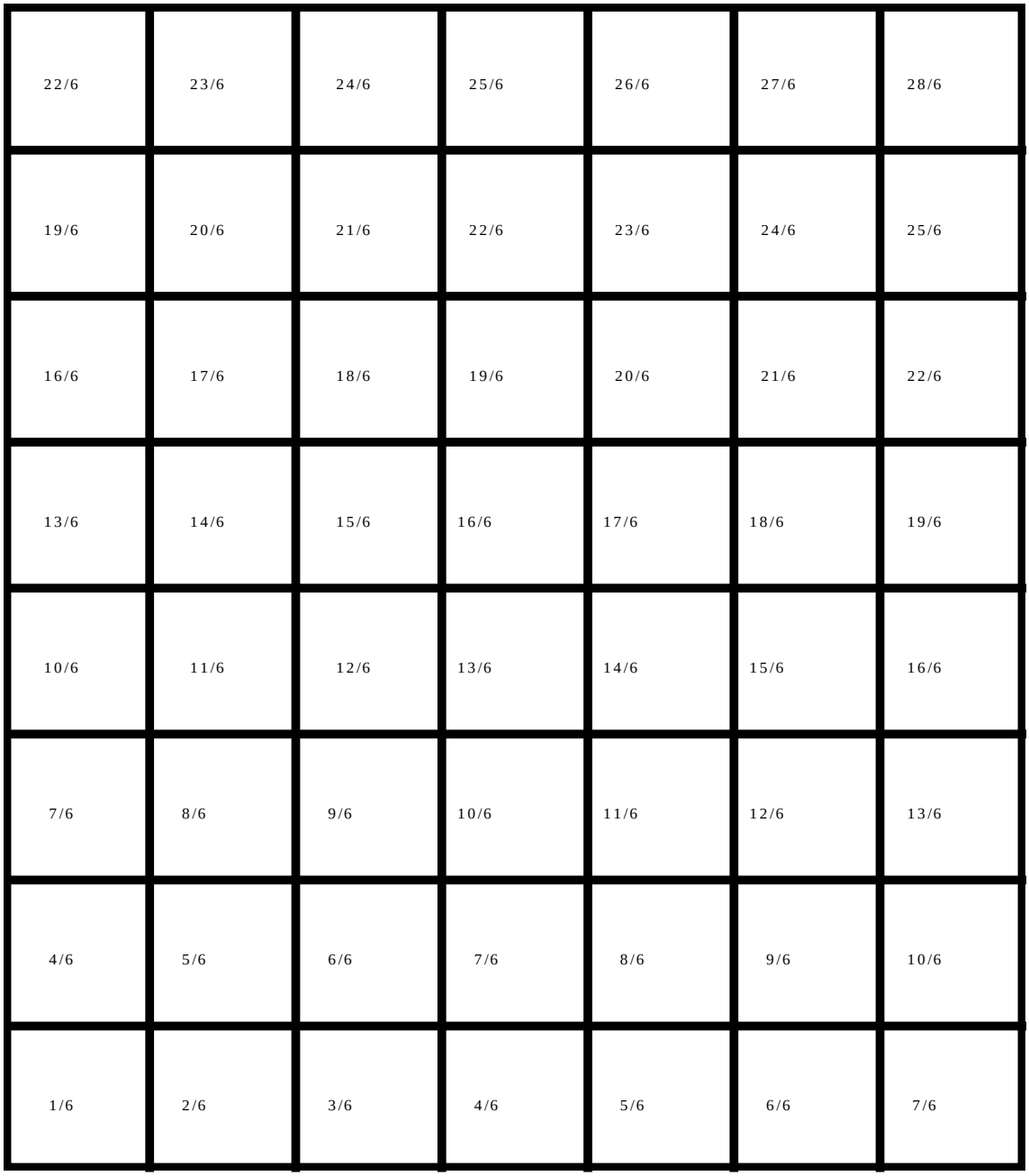

Figure 18-2: S/R Machine Movement Time (Seconds)

18.3.2 Build Models

The two operations performed by the AS/RS system are modeled as two separate processes. The first operation stores a subassembly in a bin. The second retrieves a subassembly from a bin.

Entities represent subassemblies to be stored or retreived and have five attributes:

Type =	Type of subassembly: 1, 2, 3, 4.
ArriveTime =	Time of arrival to the AS/RS system.
Rack =	Rack in which to store the subassembly: 1 or 2.
Row =	Horizontal position of the bin in which to store the subassembly.
Column =	Vertical position of the bin in which to store the subassembly.

A state variable is used to track the state of each bin:: idle, filled with a subassembly of a particular type, or filled with a subassembly of a particular type that is committed to the second manufacturing process. In addition, there is a state variable for each subassembly type modeling the number of units of that type in the racks.

A resource represents the S/R machine. The resource modeling the S/R machine has two attributes indicating its Row and Column location in the rack structure.

The storage of an arriving subassembly is handled as follows. The subassembly waits at the pick point until at least one bin is idle. Which particular idle bin to use is determined by the AS/RS control algorithm which is implemented in the model. Information identifying the location of the bin is recorded in the attributes (Rack, Row, Column) of the subassembly entity.

The subassembly continues to wait until the S/R machine is idle. The S/R machine moves from its current location to the pick point. The S/R machine picks up the subassembly, moves to the selected idle bin, and stores the subassembly in that bin. The S/R machine waits at that bin for its next assignment.

Finally, the state of the system updated. The state of the bin is changed to the type of subassembly stored in the bin. The location of the S/R machine is recorded in its Row and Column attributes. The number of subassemblies of the type just stored is incremented by one.

The process of retrieving a subassembly from a bin is similar to the storage process just described. The request for a subassembly of a particular type waits until there is a subassembly of that type in the AS/RS. The AS/RS system control algorithm selects the bin closest to the current location of the S/R machine containing a subassembly of the desired type. The S/R machines moves to that bin, retrieves the subassembly and procedes to the drop point. The S/R machine becomes idle and remains at the drop point.

Again, the state of the system is updated. The state of the bin from which the subassembly was retrieved is changed to idle. The number of subassemblies of the type just retrieved is decremented by one. The location of the SR machine is recorded.

When it becomes idle, the SR machine resource may need to choose between two jobs: storing a subassembly or retrieving a previously stored one. Management decided that it was most important to keep the second manufacturing process working. Thus, priority is given to requests to retrieve previously stored subassemblies.

The AS/RS control algorithm is shown in Figures 18-3 a, b, and c.

```
function SearchOne
/* routine to search in a given direction for a bin in a given state
   inputs:
     RowDir      = Row Direction (1 or -1)
     ColDir      = Column Direction (1 or -1)
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     Rack        = Rack ID number (1 or 2)
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     Row = Row of required bin
     Col = Column of required bin  */
/* Search Equivalent Bins */
    Row = 0
    Col = 0
    RowIndex = RowStart
    ColIndex = ColStart
    RowBase  = RowStart
    ColBase  = ColStart
/* Stay within the boundaries of a rack */
   while RowIndex <= RowMax and RowIndex > 0 do
   begin
      while ColIndex <= ColMax and ColIndex > 0 do
      begin
/* Stay within the boundaries of the search direction from the
   starting point */
       while (RowIndex <= RowMax and RowIndex > 0)   and
                    (ColIndex <= ColMax and ColIndex > 0)   and
                    ((RowIndex >= RowStart and RowDir > 0)   or
                     (RowIndex <= RowStart and RowDir < 0)) and
                    ((ColIndex >= ColStart and ColDir > 0)   or
                     (ColIndex <= ColStart and ColDir < 0))  do
       begin
                  if(LocState(Rack,RowIndex,ColIndex) = TargetState) then
                  begin
                 /* Bin in desired state found.  Set attributes */
                     Row = RowIndex
                     Col = ColIndex
                     return
                  end
                 /* Go to next bin having same movement time */
                  RowIndex = RowIndex + RowDir
                  ColIndex = ColIndex - ColDir*ColDiff
       end
   /* Go to bin with next smallest movement time in the initial row */
          RowIndex = RowBase
          ColBase  = ColBase + ColDir
          ColIndex = ColBase
      end
  /* Go to bin in the next row with the next smallest movement time */
      RowBase  = RowBase + RowDir
      RowIndex = RowBase
      ColBase  = ColStart + ColDir*ColDiff
      ColIndex = ColBase
   end
end
```

Figure 18-3a: Control Algorithm Function for Searching in One Direction

```
begin S_SearchRack
/* routine to search for a storage state in a given state in a given
direction in each of two racks
   inputs:
     RowDir      = Row Direction (1 or -1)
     ColDir      = Column Direction (1 or -1)
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     RackA = Rack ID of the required bin
     Row   = Row of required bin
     Col   = Column of required bin  */
/* Search the first rack */
     Rack  = 1
     RackA = 1
     call S_SearchOne
     RowTemp = Row
     ColTemp = Col
/* Seach the second rack */
     Rack  = 2
     RackA = 2
     call S_SearchOne
/* See if the location in the first rack is closer than the one in the
second rack */
/* return second rack info if no bin was found in the first rack */
     if(RowTemp = 0 or ColTemp = 0) then return
/* return first rack info if no bin was found in the second rack */
     if(Row     = 0 or Col     = 0) then
     begin
       Row   = RowTemp
       Col   = ColTemp
       RackA = 1
       return
     end
/* bin found in both racks; return info for bin with shorter movement
time */
     if(abs(RowTemp-RowStart)*RowSpeed+abs(ColTemp-ColStart)*ColSpeed<
        abs(Row    -RowStart)*RowSpeed+abs(Col    -ColStart)*ColSpeed)
     then
     begin
/* movement to rack one bin is shorter */
       Row   = RowTemp
       Col   = ColTemp
       RackA = 1
       return
     end
end
```

Figure 18-3b: Control Algorithm Function for Determining the Shortest Move Time Among Two Racks

The control algorithm is implemented as three functions. The first searches from the current location of the SR machine given by RowStart and ColStart in any of the four directions given above as specified by RowDir and ColDir. Each of these two variables takes on the values –1 or +1 to define the search direction. The dimensions of a rack are specified in the variables RowMax and ColMax. Which of the two racks to search is specified in the variable Rack which has the value 1 or 2. The variable TargetState gives the state of interest, zero for idle or 1, 2, 3, or 4 for a subassembly of that type. The state of each bin is stored in the three dimensional array LocState(Rack, Row, Col). The variable ColDiff stores the ratio of the horizontal speed to the vertical speed of the SR machine which is three in this case.

The entity attributes Row and Col store the location of the bin in the desired state. If no such bin is found both Row and Col have the value zero.

The other two functions use the same variables as SearchOne. The function SearchRack, shown in Figure 18-3b, searches each rack in one of the directions listed above for a bin and returns the location of the bin that is closest with respect to movement time to the current position of the SR machine. The rack ID number (1 or 2) is returned in the entity attribute RackA.

The function SearchAll, shown in Figure 18-3c, searches in all four directions from the current SR machine location to find the nearest bin in the desired state. The directions are searched one at a time using function SearchRack. After each search, the nearer location so far is determined. The nearest location is returned using the entity attributes Row, Col, and RackA.

The model also contains two processes, Arrival and Retrieval, whose steps were previously described. Pseudo-code for the two processes follows. The same variables defined above for the function SearchOne are also used in the processes. Note that in the Arrival process, the function SearchRack is used instead of SearchAll since the only search direction is up and right of the pick point.

```
begin SearchAll
/* routine to search for a storage state in each of two racks in all
directions
   inputs:
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     RackA = Rack ID of the required bin
     Row = Row of required bin
     Col = Column of required bin  */
/* Search right and up */
     RowDir = 1
     ColDir = 1
     call S_SearchRack
     RowTemp1 = Row
     ColTemp1 = Col
     RackTemp = RackA
/* Search right and down */
     RowDir = 1
     ColDir = -1
     call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
    end
```

Figure 18-3c: Control Algorithm Function for Determining the Shortest Move Time in Any Direction

```
/* Search left and up */
    RowDir = -1
    ColDir = 1
    call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
    end
/* Search left and down */
     RowDir = -1
     ColDir = -1
     call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
     end
/* return closest location */
     Row   = RowTemp1
     Col   = ColTemp1
     RackA = RackTemp
end
```

Figure 18-3c: Concluded

```
Define Resources
        SRMach              // Storage / retrieval (S/R) machine

Define Attributes
        ArriveTime          // Time of arrival of subassembly
        Type                // Type of subassembly

Define Variables
        Bin                 // Bins currently available
        InvSA1              // Subassemblies of type 1
        InvSA2              // Subassemblies of type 2
        InvSA3              // Subassemblies of type 3
        InvSA4              // Subassemblies of type 4
        RowStart            // Current location of S/R machine – row
        ColumnStart         // Current location of S/R machine – column
        RowDirection        // Direction of row search
        ColumnDirection     // Direction of column search
        TargetState         // State of requested bin
        Rack                // Rack with bin in requested state
        Row                 // Row of bin in requested state
        Column              // Column of bin in requested state
        VerticalSpeed       // Vertical (column) speed of S/R Machine
        HorizontalSpeed     // Horizontal (row) speed of S/R Machine
        LocState            // State of each bin
        ColumnMax           // Number of columns in a rack

Process SubAssembly_Arrivals
Define Arrivals
        Time of first arrival:   0
        Time between arrivals:   Exponential 20 seconds
        Number of arrivals:      Infinite
Begin
        Set ArriveTime = Clock
        Set Type = Integer Uniform (1, 4)
        Wait until Bin > 0
        Increment Bin by 1
        Wait until SRMachine is IDLE
        Make SRMachine BUSY
        Wait for RowStart*VerticalSpeed + ColumnStart*HorizontalSpeed
                // Move SRMachine to Pick Point
        Set RowStart = 1
        Set ColumnStart = 1
        Set RowDirection = 1
        Set ColumnDirection = 1
        Set TargetState = 0
        Call SearchRack returning Rack, Row, Column
        Wait for Row*VerticalSpeed + Column*HorizontalSpeed
                // Move SRMachine to Selected Bin
        Wait for 6 seconds      // Store Carrier in Bin
        Make SRMach IDLE
        LocState (Rack, Row, Column) = Type
        Increment InvSA<Type> by 1
End
```

```
Process SubAssembly_Retrievals
Define Arrivals
        Time of first arrival:      2 hours
        Time between arrivals:  Exponential 20 seconds
        Number of arrivals:         Infinite
Begin
        Set ArriveTime = Clock
        Set Type = Integer Uniform (1, 4)
        Wait until InvSA<Type> >0
        Decrement InvSA<Type> by 1
        Set TargetState = Type
        Call SearchAll returning Rack, Row, Column
        Set LocState (Rack, Row, Column) = Type +4
        Wait until SRMachine is IDLE
        Make SRMachine BUSY
        Wait for abs ((Row – RowStart)*VerticalSpeed) +
                abs ((Column-ColumnStart)*HorizontalSpeed
                // Move SRMachine to Select Bin
        Wait for 6 seconds                  // Remove Carrier from Bin
        Wait for abs (Row-1)*VerticalSpeed + abs (Column-ColumnMax)*HorizontalSpeed
                // Move SRMachine to drop point
        Make SRMach IDLE
        Set RowStart = 1
        Set ColumnStart = ColumnMax
        Set LocState(Rack, Row, Column) = 0
End
```

–

18.3.3 Identify Root Causes and Assess Initial Alternatives

Table 18-1 gives the design for the AS/RS system simulation experiment. The final assembly process consumes all subassemblies stored in the rack each day. Thus, a terminating experiment with the simulated time interval equal to the time each day that the subassemblies arrive to the AS/RS system is appropriate. The dynamics of how the final assembly process consumes the subassemblies remaining in the storage racks after all subassemblies have arrived will not affect the choice of configurations. Thus, this part of the system need not be included in the experiment.

Table 18-1: Simulation Experiment Design for the AS/RS System

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	Rack configuration -- (10X18 and 9X25)
Performance Measures	1. Number of bins in non-IDLE states 2. Time subassemblies wait for a bin 3. Time subassemblies and final process requests wait for the SR machine.
Random Number Streams	1. Type of subassembly to store 2. Time between arrivals of subassemblies to the AS/RS system 3. Type of subassembly requested by final assembly 4. Time between arrivals of requests from final assembly
Initial Conditions	The bins empty and the SR machine idle
Number of Replicates	20
Simulation End Time	One eight hour day (time in seconds)

The initial conditions are the daily start-up conditions for the system: all bins empty and the SR machine idle. Twenty replicates will comprise the experiment. There are four random number streams, one each to determine the type of subassembly delivery to the AS/RS and requested by the final assembly process as well as one each for the time between arrivals of subassemblies and requests from the final assembly process.

The model parameter is the rack configuration with the two alternatives proposed by management tested. Performance measures have to do with the utilization of bins, subassembly waiting time for an empty bin, and waiting time for the SR machine to move subassemblies.

Results of this experiment are shown in Table 18-2. The average percent of bins occupied is 16% less for the 9 X 25 rack configuration with an approximate 95% confidence interval of 15% to 18% for the true percent difference. The 9 X 25 rack is 25% bigger than the 10 X 18 rack. Thus, some use is made of the extra bin space. This is reflected in the fact that there is no waiting for an empty bin when the larger rack is used. However, the average waiting time for the smaller rack is only 2.5 seconds with an approximate 95% confidence interval of 0.7 to 4.3 seconds for the true mean waiting time. The average waiting time for the SR machine increases when the larger rack size is used, though the average difference is only 2.4 seconds.

Table 18-2: Results of the AS/RS System Simulation Experiment

	Percentage of Full Bins			Average Time Waiting for an Empty Bin (Seconds)			Average Time Waiting for the SR Machine (Seconds)		
Replicate	**10 X 18**	**9 X 25**	**Diff**	**10 X 18**	**9 X 25**	**Diff**	**10 X 18**	**9 X 25**	**Diff**
1	70%	92%	22%	0	8.3	8.3	9.4	6.8	2.6
2	72%	92%	20%	0	8.8	8.8	9.4	6.7	2.7
3	71%	92%	20%	0	8.2	8.2	9.5	6.7	2.8
4	67%	93%	27%	0	8.7	8.7	9.6	6.7	2.9
5	65%	92%	27%	0	8.5	8.5	9.3	6.7	2.6
6	72%	92%	19%	0	8.2	8.2	9.4	6.7	2.7
7	68%	94%	26%	0	8.8	8.8	9.5	6.7	2.7
8	72%	93%	21%	0	8.8	8.8	9.5	6.7	2.9
9	73%	91%	18%	0	8.9	8.9	9.5	6.7	2.9
10	70%	93%	23%	0	8.5	8.5	9.5	6.7	2.8
11	68%	92%	25%	0	8.8	8.8	9.6	6.6	3.0
12	69%	92%	24%	0	8.4	8.4	9.5	6.8	2.7
13	77%	91%	14%	0	8.2	8.2	9.4	6.7	2.7
14	68%	92%	24%	0	8.3	8.3	9.5	6.7	2.8
15	63%	94%	31%	0	9.0	9.0	9.6	6.7	2.9
16	70%	92%	22%	0	8.2	8.2	9.6	6.7	2.9
17	68%	93%	25%	0	8.9	8.9	9.5	6.7	2.9
18	74%	93%	18%	0	8.5	8.5	9.3	6.7	2.6
19	71%	92%	22%	0	8.7	8.7	9.4	6.7	2.8
20	65%	92%	27%	0	8.3	8.3	9.6	6.7	2.9
Average	70%	92%	23%	0	8.5	8.5	9.5	6.7	2.8
Std. Dev.	3%	1%	3.9%	0	0.28	0.28	0.088	0.043	0.10
99% C.I. Lower Bound	67%	92%	20%	0	8.4	8.4	9.4	6.7	2.7
99% C.I. Upper Bound	72%	93%	25%	0	8.7	8.7	9.5	6.7	2.8

18.3.4 Review and Extend Previous Work

The smaller rack configuration seems preferable since it would be less costly and require less floor space as long as system performance is not significantly improved by using the larger rack. While the larger rack does eliminate waiting for an empty bin, it increases waiting time for the SR machine. However, no waiting time is very long for either configuration. The bin utilization is relatively high for both rack sizes (92% versus 70%).

18.3.5 Implement the Selected Solution and Evaluate

The AS/RS system with the 10 X 18 configuration of two racks of bins will be implemented. Subassembly waiting at the pick point will be monitored and sufficient buffer space provided as needed.

18.4 Summary

This case study shows how system operational algorithms can be included in models. The use of modeler defined resource states is included. Inventories and other resources are shared between processes in the model. The simulation experiment compares alternative system configurations.

Problems

1. Based on the process steps in the simulation model, tell why the bin states: occupied with a subassembly of type one, two, three, or four that is committed to the final assembly process are necessary.

2. Validate the function SearchOne by searching from rack location row 2 column 2 as shown in Figure 18-2 in the right and up direction. List the first ten values of RowIndex and ColIndex computed in SearchOne, including the infeasible bin location values that cause the loops in SearchOne to end.

3. In the Arrival process model, why is it not necessary to check if the function SearchRack located a bin in the IDLE state?

4. Tell why the quantity: Number of final assembly process requests waiting for a subassembly is not an effective performance measure for the simulation experiment in this chapter.

5. What impact would running the simulation experiment until all subassemblies had been moved to the final assembly process have on the validity of the performance measure estimates?

6. Explain why the average waiting time for the SR machine increases when the larger rack size is used especially considering that there is no waiting for an empty bin.

7. Would you expect the utilization of the SR machine to increase or decrease when the larger rack size is used? Justify your answer.

8. Use Little's Law to estimate the average number of subassemblies waiting at the pick point. How much buffer space would you use at the pick point?

9. Compute the expected time to store a carrier in a bin after the SR machine is obtained.

10. Visit a manufacturing facility and observe the automated material handling equipment that is in use.

11. Make a list of the automated material handling equipment you have observed in the service systems you encounter regularly.

12. How much improvement is there in the AS/RS system if the speed of the SR machine increases by 100%.

13. How much improvement is there in the AS/RS system if the time between requests from the second manufacturing process is uniformly distributed between 10 and 30 seconds?

14. Perform additional simulation experiments to find the smallest difference between the starting time of the storage process (currently 6:00 A.M.) and the retrieval process (currently 8:00 A.M) for which the system can effectively operate.

15. The current rack configurations are about one story high. Suppose a two story high configuration was preferred, specifically 18 bins high and 10 bins wide. Compare system performance using this configuration to the 10 bins high and 18 bins wide configuration.

16. Embellish the model in this chapter with acceleration and deacceleration of the SR machine. Assume the acceleration (deacceleration) distance is one bin in either direction and the average time to traverse this bin is twice that of other bins.

Case Problem

The benefits of AS/RS technology have been effectively realized in libraries. The amount of floor space required for books and periodicals has been reduced by ten-fold or more. The number of librarians required was reduced as well. Reshelving errors were eliminated. The location of each item while in the library is known with certainty. Despite these benefits, it is estimated that a few (less than 12) mini-load AS/RS systems have been installed in libraries.

This case problem involves determining the saturation point for a mini-load AS/RS system installed in a particular library. This is done be creating a graph of the cycle time for retrieving a book or periodical versus the arrival rate for such requests. The arrival rate resulting in the longest acceptable retrieval time is the saturation point. The smallest arrival rate of interest is 10 requests per hour. Assume that the arrival rate for retrievals is the same as the arrival rate for returns.

The mini-load AS/RS system installed in one particular library has a capacity of 250,000 books and periodicals. There is a single aisle with identical racks on each side. The system is installed inside a secured vault for safety and security reasons.

Books and periodicals are stored in carriers that are 4 feet deep and 2 feet wide. Each carrier row is one of three heights: 10, 12, or 15 inches. Each item is stored in the shallowest carrier in which it can stand. Thus, vertical space is used most efficiently. Assume that the number of books and periodicals of each height is the same.

There are 36 carrier rows on each side of the single aisle. The height of the first row is 10 inches, the second 12 inches, the third 15 inches, the fourth 10 inches and so forth. There are 60 carriers in each row.

The S/R machine travels at a high rate of speed: 12.6 feet/second horizontally and 4.3 feet/second vertically. Assume that the S/R machine must travel either horizontally or vertical but not diagonally.

The process of retrieving a book or periodical is the following. A patron makes a request using the electronic library catalog system. The AS/RS fills one request at a time. The location of the

item is completely random. The S/R machine moves from its idle location to the required carrier, extracts the carrier in 3 seconds, and places the carrier in the pick and delivery station. A librarian must remove the desired item from the carrier and record its status in the information system. This takes 7 seconds. The S/R machine remains idle at the pick and delivery station.

Next the librarian determines whether any item that needs to be returned to storage is of the same size as the carrier. If so, the item's new carrier location is recorded in the information system and the item placed in the carrier. Both steps combined take 7 seconds.

Assume the library is open 16 hours per day, 7 days per week.

Embellishment: The AS/RS system tests the carrier for weight restrictions. One in 100 tests fail. In this case, the librarian must remove the item as well as the newly entered location from the information system in 7 seconds. In either case, the S/R machine replaces the carrier and returns empty to its idle location.

Embellishment: Find the saturation point when the following procedure is used. The S/R machine does not replace a carrier that is at a pick and delivery station until the next retrieval request is made. At that time, a carrier is first stored and then the next carrier retrieved.

Embellishment: Limit the number of carriers stored at the pickup/dropoff station to a total of three. When the fourth carrier arrives, it is immediate returned to the same storage location by the AS/RS machine.

Case Problem Issues:

1. How should carriers be modeled?

2. How should the location of the carrier containing the book or periodical requested be determined?

3. How should S/R machine travel time be computed?

4. Specify the process for book and periodical returns.

5. What are good initial conditions for this simulation experiment?

6. What performance measures, other than cycle time, would be of interest?

7. What is the expected utilization of the SR machine?

8. How should verification and validation evidence be obtained?