Topics in Computer-Aided Design:

Part II. Synthesis of Optimal Heat Exchanger Networks by Tree Searching Algorithms

A compact matrix representation of an acyclic exchanger network is derived and used to generate a decision tree for the heat exchanger problem. The optimal network, which is imbedded in one of the tree nodes, is located by enumerating the entire tree using a search procedure. When this is not feasible, an alternate partial enumeration method is suggested. Sample problems ranging in size from 4 to 10 streams are used to illustrate the methods.

T. K. PHO
and
L. LAPIDUS

Department of Chemical Engineering
Princeton University
Princeton, New Jersey 08540

SCOPE

A common problem in process design is the synthesis of an heat exchanger network to transfer excess energy from a set of hot streams to streams which require heating. Since the network is unknown, the problem of finding an optimal network cannot be solved by ordinary optimization techniques which require a known problem structure. Attempts have been made, however, to solve this synthesis problem using combinatorial techniques and linearization (Kesler and Parker, 1969; Kobayashi et al. 1971; Nishida et al. 1972).

Perhaps the most feasible approach to the problem which can handle the associated nonlinearities directly is the branch and bound algorithm by Lee et al. (1970). In this method a stream process statement is used to describe all the possible matching sequences for each primary stream. A plausible network can be synthesized by selecting one process statement for each primary stream. Whenever feasible, this set of process statements will characterize completely the structure of a feasible network. The feasibility of the plausible network is tested by observing if any of the selected process statements specify multiple use of a primary stream. If the network feasibility is not met, a different set of process statements must be selected. The branch and bound method guides this selection by branching into smaller and more manageable bounding problems. Although the method guarantees optimality, there are two basic faults with this approach. First, since it must characterize all the possible matching sequences of each primary stream and there can be numerous possibilities, the process statement can be very lengthy and tedious to write. Further, the number of such statements can be very large (a total of 450 statements were reported for a six stream problem). Second, the formation of a stream process statement must be such that it satisfies a Stream Feasibility Criterion. This criterion specifies that a stream number must not appear more than once in a process statement. Unfortunately it will be shown here that this criterion is not always correct and can further complicate the synthesis algorithm.

In this paper, a compact synthesis matrix which can represent the structure of an exchanger network is derived. Based on this matrix a decision tree diagram whose nodes will encompass all the feasible networks is constructed. This reduces the synthesis problem into a tree searching problem where one seeks to locate a node with minimum cost. Two tree searching methods, a direct enumeration by a depth first technique and a partial enumeration using an evaluation function are described and illustrated on a number of sample examples.

CONCLUSIONS AND SIGNIFICANCE

It is shown that the synthesis matrix is a compact and efficient representation of an acyclic heat exchanger network. The matrix further includes networks which would otherwise be excluded from the stream process statement formulation of Lee et al. (1970). The decision tree approach to the synthesis problem requires minimal computer storage as every network can be constructed rapidly from the root node of the tree where it represents the network with no matching. An automatic synthesis of an optimal network is therefore possible by a direct enumeration of the entire tree. The present method, however, does not eliminate the basic combinatorial difficulty of the original problem because problems with more than ten streams can require excessive computation time. It is shown that the problem is less severe when the partial enumeration technique is used. It is therefore suggested that further work along this line be pursued.

As such, the design solution of the heat exchanger and other equivalent problems is now solvable in a simple and efficient manner.

PROBLEM STATEMENT

The synthesis problem to be considered has been defined in previous works (Masso and Rudd, 1969; Lee et al. 1970), and thus only the barest essentials are described here. There are a total of N_h hot streams h_i, $i = 1, 2, \ldots N_h$, to be cooled, and N_c cold streams c_j, $j = 1, 2, \ldots N_c$, to be heated. Associated with each stream are its input temperature T_i, output temperature T_i^0, and the heat capacity flow rate W_i. The synthesis problem is to create

a minimum cost network consisting of heat exchangers, coolers and/or heaters so that the output temperature of each primary stream is satisfied.

The synthesis task can begin by either exchanging as much heat as possible among the process streams (the capital cost is a maximum) or processing each stream through the auxiliary steam heaters and water coolers (the utility cost is a maximum). The desired exchanger network will be a balance between these two extremes depending on the fabrication cost of each exchanger specified by the heat transfer area $E = aA^b$ where a, b are constants, and the cost per pound of steam and water. The objective function for minimization is expressed by the following equation:

$$J = \delta\left\{\sum_i aA^b{}_{E_i} + \sum_i aA^b{}_{H_i} + \sum_i aA^b{}_{C_i}\right\} + U \tag{1}$$

where δ is the annual rate of return of the investment, A_E, A_H, A_C are the heat transfer areas for the exchanger, heater and cooler, respectively, and U is the cost of water and steam per year.

The examples worked out in this paper are denoted as 4SP1, 5SP1, 6SP1, 7SP1, 7SP2 and 10SP1, all except the 10SP1 are taken from the previous works of Lee et al. (1970) and Masso and Rudd (1969). The specifications for the problems 4SP1 and 10SP1 and the relevant data are listed in Tables 1 and 2, respectively. The 4SP1 problem involves 2 streams to be cooled and 2 to be heated; the 10SP1 problem has 5 streams to be cooled and 5 to be heated. As will become apparent in a later section, the size of a given problem is characterized by the number of entries in its synthesis matrix. The 10SP1 will have a 5×5 matrix and a total of 25 entries. This represents the maximum possible size for any ten streams problem. For example, 2 cold streams and 8 hot streams will have only 16 entries and is considerably easier to solve than the 10SP1. All other sample problems are also at their maximum possible size. The 10SP1 problem has not been solved before and will be used to illustrate the case where the combinatorial difficulty prohibits the solution by direct enumeration.

Table 1. Process Streams Specifications

Problem 4SP1

Stream	Capacity flow rate	Input temp.	Output temp.
c_1	1.445×10^4	140	320
c_2	1.153×10^4	240	500
h_1	1.667×10^4	320	200
h_2	2.000×10^4	480	280

Problem 10SP1

Stream	Capacity flow rate	Input temp.	Output temp.
c_1	1.445×10^4	140	320
c_2	1.153×10^4	240	431
c_3	1.600×10^4	100	430
c_4	3.276×10^4	180	350
c_5	2.635×10^4	200	400
h_1	1.667×10^4	320	200
h_2	2.000×10^4	480	280
h_3	2.800×10^4	440	150
h_4	2.380×10^4	520	300
h_5	3.360×10^4	390	150

Table 2. Design Data

Steam pressure (saturated)	962.5 lb./sq.in.abs. for problem 4SP1 450 lb./sq.in.abs. for all other problems
Cooling water temperature	100°F
Maximum water output temperature	180°F
Minimum allowable approach temperatures:	
Heat exchanger	20°F
Steam heater	25°F
Water cooler	20°F
Overall heat transfer coefficients:	
Heat exchanger	150 B.t.u./hr. ft.^{2} °F
Steam heater	200 B.t.u./hr. ft.^{2} °F
Water cooler	150 B.t.u./hr. ft.^{2} °F
Equipment down time	380 hr./yr. 260 hr./yr. for problems 7SP1, 7SP2 and 10SP1
Heat exchanger cost parameters	$a = 350$ $b = 0.6$
Annual rate of return	$\delta = 0.1$
Cooling water cost	5×10^{-5} \$/lb.
Steam cost	1×10^{-3} \$/lb.

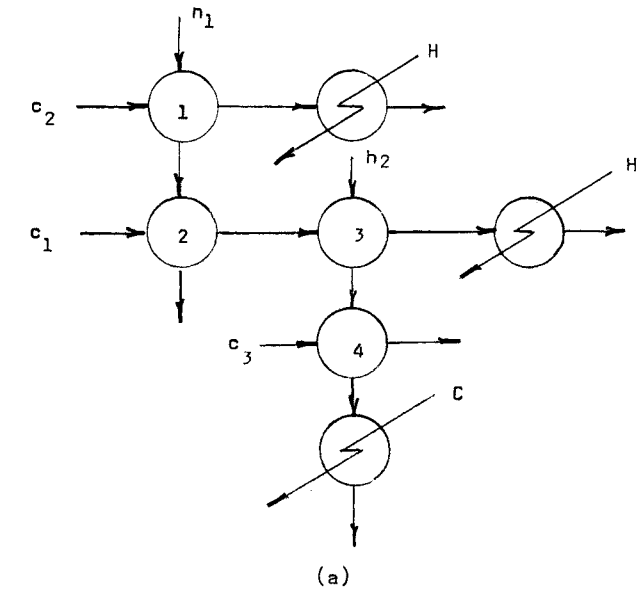
MATRIX REPRESENTATION OF EXCHANGER NETWORKS

As in the work of Lee et al., only networks which are of acyclic design, that is, a network with no recycle information flow, are considered here. For such a network, the synthesis can be performed by matching a pair of process streams sequentially. For example, the acyclic exchanger network shown in Figure 1a can be synthesized sequentially according to the following synthesis steps:

(i) Match cold stream c_2 and hot stream h_1 with the exchanger 1.
(ii) Match c_1 and residual of h_1 with the exchanger 2.
(iii) Match residual of c_1 and h_2 with the exchanger 3.
(iv) Match c_3 and residual of h_2 with the exchanger 4.

As the last synthesis step, any process streams which still have not reached their output temperatures will be processed through the auxiliary steam heaters and water coolers. In this example, two heaters are required to heat the residuals of c_1 and c_2, and one cooler to cool the residual of h_2. Note that the exchangers synthesized in the first four synthesis steps determined uniquely the structure of the overall network. The additions of heaters and coolers serve only to satisfy the design specifications. Following Lee et al., an additional assumption is also made that when two streams are matched by an exchanger, they must exchange as much heat as technically possible. As a consequence of this assumption, a given pair of process streams will be matched at most only once throughout the synthesis sequence. The above observations lead to the following derivation of a synthesis matrix to represent an arbitrary acyclic network.

We shall refer to an exchanger network with n heat exchangers as a n step network. At a given synthesis step a pair of process streams will be matched and identified by the (i, j) entry of a matrix A whose row and column denote the cold and hot streams, respectively. If at the nth synthesis step, the cold stream c_i is matched to the hot stream h_j, the (i, j) entry is set equal to n, or $a(i, j) = n$; otherwise the entry is left blank. A M step network will therefore have M nonzero entries whose values range from 1 to M. A convenient representation of the above concept, called the synthesis matrix, is shown in Figure 1b for the four-step network of Figure 1a. The first row and column of the synthesis matrix identify the cold and hot



	h_1	h_2	
c_1	2	3	H
c_2	1		H
c_3		4	T
	T	C	

(b)

	h_1	h_2	
c_1	2	3	H
c_2	1	5	H
c_3		4	T
	T	T	

(c)

Fig. 1. An acyclic exchanger network and its synthesis matrix.

streams. The last row and column denote whether the process streams require the service of heaters (H) or coolers (C) to meet their temperature specifications. Those streams which have met their terminal temperatures during the earlier synthesis and therefore require no service from the heaters and coolers in the final synthesis step are denoted by the symbol T. The remaining block of the synthesis matrix is the A-matrix defined earlier. The specification of this A-matrix will determine uniquely the structure of the overall exchanger network. The use of heaters and coolers serve only to satisfy the final output of each stream. Thus in what follows, an exchanger network will be specified by either its synthesis matrix or its A-matrix.

A synthesis matrix can serve to test if an existing network can be expanded to include an additional exchanger. This expansion process will generate a new network capable of achieving the same design specifications and is therefore a feasible solution. Consider the synthesis of a n step network for which the row i has a H in its last column, and the column j has a C in its last row. It is now possible to introduce an additional $(n+1)$th exchanger to perform the matching of stream c_i and h_j. Notice that the $a(i, j)$ entry will always be blank if the matching is feasible. The A-matrix for this $n+1$ step network will be the same as for the n step network except for the additional element $a(i, j) = n + 1$. For example, a possible five step network which can be expanded from the four step network of Figure 1a is shown in Figure 1c.

GENERATION OF A DECISION TREE DIAGRAM

The matrix representation of an exchanger network and its expansion process readily lends itself to the construction of a decision tree diagram. A decision tree diagram is a graph rooted at one starting node such that there exists only one path from this starting node to all other nodes. If each node represents a feasible exchanger network, then a decision tree diagram in a convenient representation of the whole feasible solution space. As shown earlier, either the synthesis matrix or its A-matrix can represent a feasible network and is thus an excellent choice for defining nodes in a decision tree.

As an illustration, we shall generate the decision tree diagram for the four streams problem 4SP1 (solved by Lee et al., 1970) using the A-matrix representation. To aid in interpretation, the blank entries in the rows and columns of the A-matrix whose corresponding streams have reached their final output temperatures will be denoted with the symbol asterisk *. These entries can no longer be used for further synthesis and the asterisks along the corresponding rows and columns will denote this infeasibility. The remaining blank entries of the A-matrix will therefore indicate possible future matchings. This forms the basis of the expansion process to be performed on an existing network. Since the A-matrix does not include the last row and column of the synthesis matrix which indicates the services of heaters and coolers, it is understood that any unsatisfied streams will automatically be processed through the heaters and coolers. The cost of each node represented by an A-matrix will include the cost of the exchangers plus the heaters and coolers cost as computed by (1).

For the starting node, we choose the exchanger network where all the streams are being processed directly through the steam heaters and water coolers. This is represented by the A-matrix where all its entries are blanks and is labeled as node 1 in the decision tree shown in Figure 2. The annual cost of this network is computed from (1) to

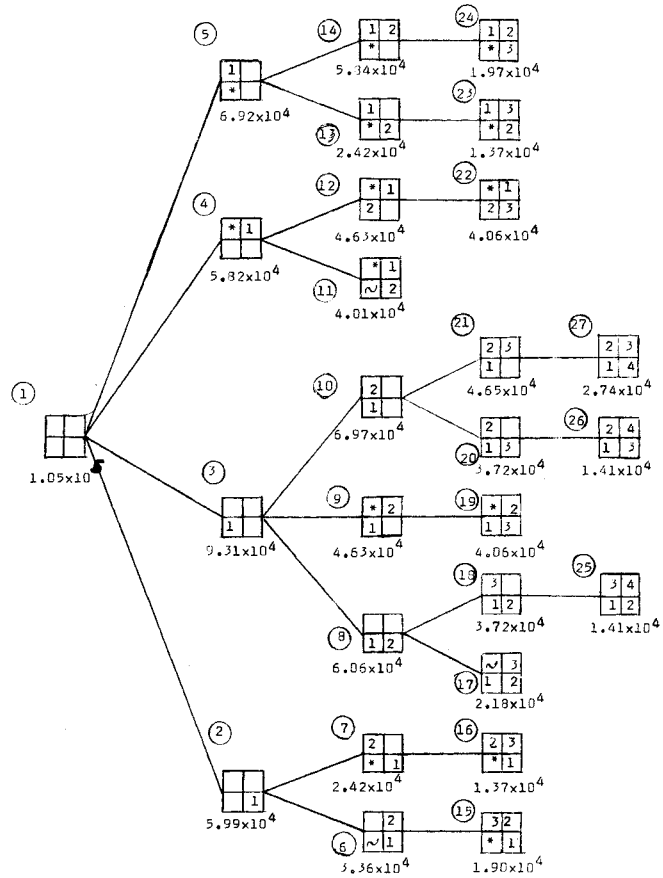


Fig. 2. Basic tree formation for Problem 4SP1.

be \$1.05 $\times$ 10^5/yr. and this is listed below the node. For the first synthesis step, any of the stream pairs, as represented by the four blank entries of node 1, is a possible candidate for matching. This can be indicated by assigning the value of one of each of the four blank entries. If the matching is feasible, its A-matrix will contain only one integer entry representing a new feasible one step network. If the matching is infeasible (for example, the inlet temperature of the hot stream is lower than the cold stream), the corresponding entry will be filled with the symbol $\sim$. In Figure 2, four one-step networks have been expanded from the first existing network node. They are labeled as nodes 2 through 5. For node 5, we notice that after the c_1 and h_1 match, the h_1 stream reaches its final temperature of 200°F, thus the symbol * is assigned the $a(2, 1)$ entry. Since node 5 has now two blank entries indicating that further synthesis is possible, the value 2 can be assigned to either entry $a(1, 2)$ or $a(2, 2)$, resulting in nodes 13 and 14. This expansion process can be repeated with all the other nodes until all the entries in the A-matrix are filled. A completely filled A-matrix indicates the completion of the synthesis steps and therefore forms the nodes at the end of each tree branch. The method results in a total of 27 nodes comprising all the feasible exchanger networks consistent with the data of 4SP1. We shall call Figure 2 the basic tree. From this basic tree, it is clear that node 23 represents the optimal network with an annual cost equal to \$13,700/yr. The same optimal network was also found by Lee et al. (1970).

We pause here to point out an incorrect statement made by Lee et al. (1970). In their approach, the exchanger network is represented by a set of process stream statements which characterize the matching sequences of each primary stream (it is simple to rewrite each of the nodes in Figure 2 in their equivalent process statements form). In particular, for the network represented by the node 26, the equivalent stream process statements can be written as follows:

Stream 1: ((1, [3, 2]), [(3, 2), 4], T
Stream 2: [1, [3, 2]], T
Stream 3: ([3, 2], 4), H
Stream 4: [(1, [3, 2]), [(3, 2), 4]], C

where streams 1 through 4 represent the streams c_1, h_1, c_2, and h_2, respectively. In their statement of Stream Feasibility Criterion, these authors stated that a stream process statement is feasible when a stream number does not appear more than once in that same statement. It has, however, been demonstrated here that the stream statements for the stream number 1 and 4 are feasible and yet both streams 3 and 2 appear twice in the same statements. Many similar examples for larger size problems may easily be visualized. The omission of these statements leads one to consider only a subset of the complete solution space. For example, the networks represented by nodes 26 and 27 in Figure 2 were not considered by Lee et al. In fact, node 26 represents the second lowest cost network for the 4SPI problem with a cost of \$14,100/yr. vs. \$19,700/yr. for node 24 if Lee et al.'s approach is used.

Finally, observe that though nodes 25 and 26 are distinct, they really represent the same exchanger network. This feature of nonuniqueness will be discussed in the next section.

REFINEMENT OF DECISION TREE DIAGRAM

If the synthesis sequence of an acyclic network is not unique, its matrix representation, which depends on the sequence chosen, is also not unique. For example, the exchanger network represented by the node 20 in Figure 2 can also be synthesized according to the sequence denoted by the node 18. For synthesis purpose, retaining any one of these nodes is sufficient, the other representations being redundant. For a relatively large size problem, the numerous redundant nodes which the tree can generate may cause difficulties. However, in Appendix I a rigorous method is derived in which all the redundant nodes can be deleted during the expansion of the tree, thereby achieving a reduction in the size of the basic tree. Here, a simpler and less rigorous method is described which is intended for problems of the type considered in the examples where the combinatorial difficulty is not too severe. While this simple method does not guarantee a unique representation of each network, it does provide for a very simple programming format on a computer while achieving a considerable reduction of the basic tree.

As the result of using the synthesis matrix to describe an exchanger network, the following properties may be noted. Let the entry which is assigned to the value n in the nth synthesis step be denoted by (i_n, j_n). Consider two consecutive entries (i_{n-1}, j_{n-1}) and (i_n, j_n). They are noninteracting if and only if $i_{n-1} \neq i_n$ and $j_{n-1} \neq j_n$. If any of these inequalities are violated, they are interacting. It then follows that

Theorem 1:

> "If the two entries (i_{n-1}, j_{n-1}) and (i_n, j_n) are noninteracting, the network with the assignments of $a(i_{n-1}, j_{n-1}) = n - 1$ and $a(i_n, j_n) = n$ and the network with $a(i_{n-1}, j_{n-1}) = n$ and $a(i_n, j_n) = n - 1$ are identical. But for the two consecutive entries which are interacting, the two assignments will result in two different networks."

The validity of the above theorem is most easily proved by rewriting the synthesis matrix into its network form. Further clarifications will be obvious as we proceed below.

Instead of identifying each entry in the synthesis matrix by its row i and column j, a transformation can be defined:

$$k = j + N_h\,(i - 1) \qquad \begin{matrix} i = 1, 2, \ldots N_c \\ j = 1, 2, \ldots N_h \end{matrix}$$

where N_c, N_h are the number of cold and hot streams, respectively. This will map each (i, j) entry into a unique number k which ranges from 1 to N, where $N = N_c \times N_h$. For example the (4, 3) entry in a 5 $\times$ 3 matrix is also the $3 + 3(4 - 1) = 12$ entry in the k-system. Thus each entry in the synthesis matrix may be denoted by either its (i, j) or its equivalent k number.

If k_n denotes the entry in which the nth synthesis step is assigned, then the convention of numbering the synthesis sequence is stated as follows:

> Two noninteracting entries (i_{n-1}, j_{n-1}) and (i_n, j_n) in the A-matrix are assigned the values of $n - 1$ and n, respectively, if and only if $k_n > k_{n-1}$.

Thus in Figure 2, the node 20 will be retained and the node 18 declared redundant since in the latter, $k_3 < k_2$, which violates the assignment convention.

As the synthesis task will be performed exclusively on a digital computer, the assignment convention can be extended to a more programmable format. Let the set of all feasible entries in the i_nth row and j_nth column be denoted by S_1^n. Then, by definition, any two consecutive entries k_{n-1} and k_n are interacting if and only if $k_n \in S_1^{n-1}$. Also let the set of all feasible entries whose k-values are larger than $k_n = j_n + N_h(i_n - 1)$ be denoted by S_2^n. Then a simple procedure of assigning the $(n + 1)$th synthesis

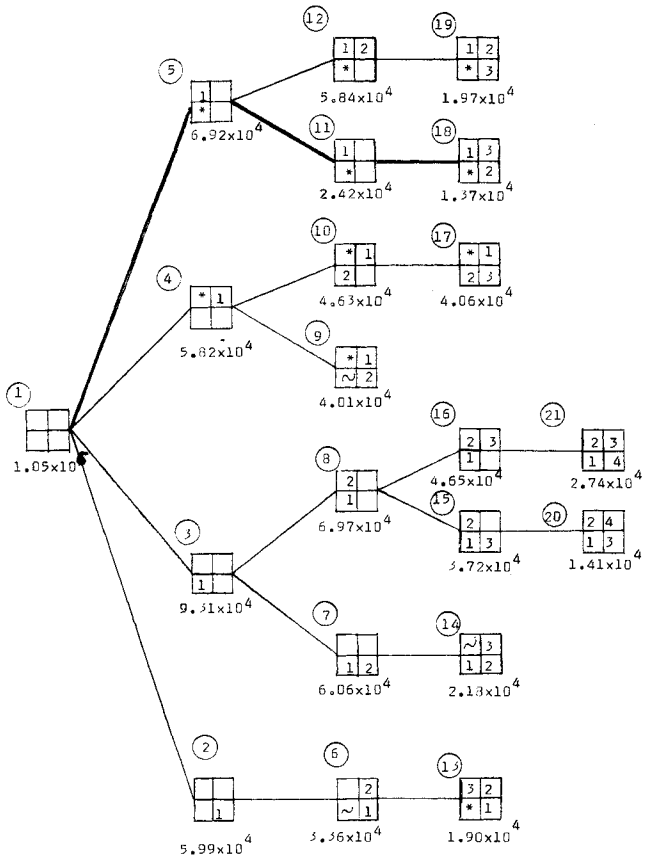


Fig. 3. A reduced decision tree for Problem 4SP1.

step after the n step network has been synthesized is summarized by the following rule:

> "Given a synthesis sequence $k_1k_2 \ldots k_n$, the set of all entries for the feasible assignment of the $(n+1)$th synthesis step is given by $\{k_{n+1}\} = S_1^n \cup S_2^n$."

This in turn leads to

Theorem 2:

> "If the set $\{k_{n+1}\}$ with $n = 1, 2, \ldots, N-1$ are used in the generation of the decision tree for the exchanger problem, then the tree nodes will encompass all feasible networks."

Proof:

Let $\overline{k}_{n+1} \in \{k_{n+1}\}$ and consider a synthesis sequence

$$L = k_1k_2k_3 \ldots . k_{n-1}k_n\overline{k}_{n+1}$$

From the assignment rule it is apparent that k_n and $\overline{k}_{n+1}$ are noninteracting and $\overline{k}_{n+1} < k_n$; thus L violates the assignment convention and is a nonvalid representation. An equivalent valid representation to L would be $k_1k_2k_3 \ldots k_{n-1}\overline{k}_{n+1}k_n$ which would have been accounted for in the earlier synthesis.

Thus if a node is represented by a synthesis sequence $k_1k_2 \ldots k_{n-1}k_n$, only the last entry k_n is needed in order to assign the value of k_{n+1} from the set $\{k_{n+1}\}$. By Theorem 2, the above assignment rule not only will generate all feasible $n+1$ step networks, but will reduce substantially the number of redundant nodes. As applied to the 4SP1 problem, the rule results in the reduced tree shown in Figure 3. The tree consists of only 21 nodes and can be shown to be a subtree of Figure 2. In this particular example, it happens that each node is also the unique representation of the network it represents. It is interesting to note that, without the branching process, the basic method of Lee et al. must consider over 80 bounding problems in order to arrive at the optimal network. However, by the tree formulation approach, only 21 networks need to be considered.

SYNTHESIS BY ENUMERATION

An efficient method by which a decision tree for an exchanger synthesis problem can be generated has been described. As the optimal network is imbedded in one of the tree nodes, the most direct method of locating this optimal node is by enumerating each and every node on a computer. A method frequently used to enumerate completely all the nodes in a tree is by a blind-search procedure (Nilsson, 1971). This method offers a systematic path by which all the nodes are examined in a certain orderly manner. Depending on the order in which the nodes are to be generated, a blind-search can be done either breadth-first or depth-first (Nilsson, 1971).

In the breadth-first method the nodes are expanded in the order in which they are generated, whereas in the depth-first, only the most recently expanded nodes are generated first. As an illustration, let us consider the decision tree shown in Figure 3 and assume the immediate descendant nodes are expanded from top to bottom. Beginning from the starting node 1, the breadth-first method will enumerate nodes 5, 4, 3, 2 in that order of sequence. Since node 5 is first generated, it will be next expanded to yield nodes 12 and 11, then node 4 to yield nodes 10 and 9 and so on until the nodes 21 and 20 are enumerated last.

Using the depth-first method, the node 5 will be the only node generated from the starting node 1. Since it is the most recently generated node, node 5 is expanded to yield node 12. Again since node 12 is the most recently generated node, its expansion yields node 19. As node 19 has no descendant nodes, the method back tracks to node 12 to check if it has descendant nodes other than node 19. Since no such node can be found, the method back-tracks again to node 5 to check if descendant nodes other than node 12 exist. Now it generates node 11 and its expansion yields node 18. Again the back-tracking procedure leads to the enumeration of nodes 4, 10, 17, 9, 3, 8, and so on.

Notice that in the breadth-first method, the nodes in each level of the tree are generated sequentially until the last level of the tree is reached. No back-tracking procedure is required and, therefore, from a programing point of view, it is much simpler than the depth-first method. However, the breadth-first method requires all the nodes generated in each tree level be stored. This can cause severe storage problems because each node is characterized by a matrix. In contrast, the depth-first method requires the storage of at most only N_L nodes, where $N_L \leq N_c \times N_h$ is the number of tree levels in a problem which consists of N_c cold streams and N_h hot streams.

An enumeration program has been written in Fortran IV using the depth-first method to find the optimal networks for the problems 4SP1, 5SP1, 6SP1, 7SP1, and 7SP2. The results in terms of the synthesis matrix are shown in Table 3. The computer execution times (IBM 360/91) vary from 0.06 s for 4SP1 to 28 s for 7SP1. The problems 4SP1, 5SP1, and 6SP1 have also been solved by the branch and bound method (Lee et al., 1970). Since the optimal networks do not exhibit the multiple appearance of stream numbers when written in stream process statements form,

TABLE 3. OPTIMAL NETWORK FOR SAMPLE PROBLEMS

Problem	Optimal network					Cost, $/yr.	Computer exec. time, s
		h_1	h_2				
	c_1	1	3	T			
4SP1	c_2		2	H		1.3685×10^4	0.06
		T	C				
		h_1	h_2				
	c_1	4	2	H			
5SP1	c_2		1	T		3.8268×10^4	0.31
	c_3	3		T			
		T	T				
		h_1	h_2	h_3			
	c_1		2	1	T		
6SP1	c_2	3			T	3.5659×10^4	2.01
	c_3		5	4	T		
		C	C	C			
		h_1	h_2	h_3			
	c_1		5	2	T		
7SP1	c_2			1	T	3.0433×10^4	28.08
	c_3	3			T		
	c_4		4		T		
		C	C	C			
		h_1	h_2	h_3			
	c_1			1	T		
7SP2	c_2	3		2	T	2.8518×10^4	22.94
	c_3	4			T		
	c_4	6	5		H		
		T	T	T			

identical results should be achieved as the present method. However, only the 4SP1 and 5SP1 results are actually the same. The optimal network for 6SP1 as found by Lee et al. (1970) is shown in Figure 4 with the cost computed to be $35,714/yr. vs. the present optimal network cost of $35,657/yr. The discrepancy is traced to a computational error in Lee's work (the first residual of c_2 should be 243° instead of 230°F). The seven streams problem 7SP1 and 7SP2 have also been solved by Masso and Rudd (1969) using their heuristic structuring algorithm. The network costs they found for 7SP1 and 7SP2 are $34,376/yr. and $28,628/yr., respectively. This is compared to the present optimal costs of $30,433/yr. and $28,518/yr., respectively. Thus in these two cases, improved networks have been found by the present algorithm. Notice that the problem 7SP2 displays the special feature that nearly all the available process heat is recoverable and that total recovery is achievable by many configurations. Computationally,

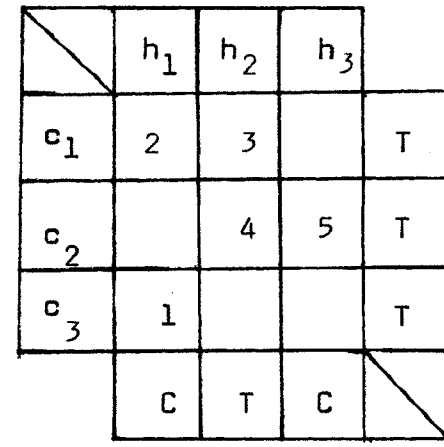


Fig. 4. Lee *et al.*'s optimal network for Problem 6SP1.

this helps reduce the number of tree branches in the decision tree and therefore reduces the computation time.

SYNTHESIS BY PARTIAL ENUMERATION

It is apparent that the depth-first method, when allowed to search each branch to the end of the decision tree, is equivalent to the direct enumeration of the entire solution space. In principle the method can provide a solution to all sizes of the exchanger synthesis problem. However, as the decision tree grows large, the computation time for the direct enumeration can become prohibitive. The same problem is also encountered in the chess playing program and other related areas of artificial intelligence (Samuel, 1967; Nilsson, 1971). A possible solution for this difficulty is to define an evaluation function to each of the nodes generated during the synthesis; such a function evaluates the quality of each node. It can be used to pull the search toward the optimal node by first generating the most promising node, while the remaining nodes are either retained for future expansion or simply deleted from further consideration.

The evaluation function can be exact or merely heuristic; in the first case optimality is guaranteed while in the second there is no guarantee of finding the optimal node. However, the heuristic procedure would be expected to be much easier to compute. Thus the purpose of the evaluation function is to generate a partial enumeration to the entire decision tree while still providing for the possibility that the optimal node can be located.

Here a method which employs the heuristic approach will be detailed for the synthesis of optimal exchanger network. Define an operator $\Gamma_n(\cdot)$ which when operating on a node y will generate all descendant nodes of depth zero to n (n look-ahead steps) from node y. The descendant nodes of depth zero from the node y is defined as y itself. Thus in Figure 3, $\Gamma_2(5) = \{5, 12, 11, 19, 18\}$. Notice that the set $\Gamma_n(y)$ forms itself a subtree rooted at node y; thus the enumeration program as described in the previous section can be used to enumerate all the nodes in this subtree. The minimum cost then serves as the evalution function for the node y.

Beginning at the starting node 1, the exchanger synthesizer program would examine all the evaluation functions of its immediate descendant nodes and move to the

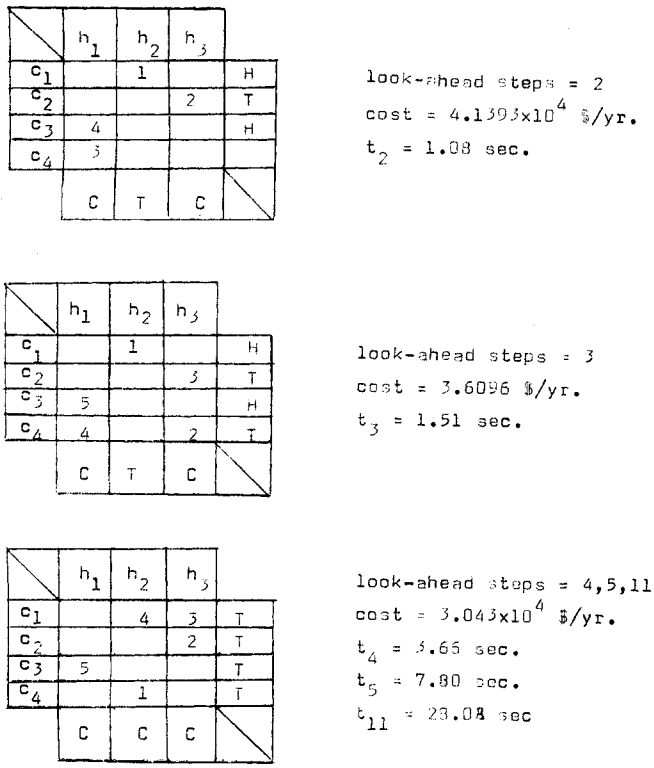


Fig. 5. Synthesis of exchanger network for Problem 7SP1 using various look-ahead steps.

node with the smallest evaluation function; at the same time all the other generated nodes with greater evaluations will be discarded. If the cost of this newly arrived node is equal to its evaluation function, the search terminates and this node is taken as being the local optimal node. Otherwise the search is continued by expanding the newly arrived node and computing the evaluation functions of its descendant nodes.

As a typical illustration of this partial enumeration algorithm, the 4SP1 problem has been solved using $n = 2$ (2 look-ahead steps). The result is the heavy line path shown in Figure 3; in this case optimality is achieved. For $n = 1$, it is interesting to note that the search will always move to the next node with the minimum cost; this is analogous to employing a method of steepest descent. Of course when $n = N_c \times N_h - 1$, the method reduces to complete enumeration of the entire tree.

Problems 7SP1 and 7SP2 have been solved using the heuristic evaluation function by varying the number of look-ahead steps n until the optimal node is located. The results are tabulated in Figures 5 and 6 where t_n denotes the computer execution time for the particular choice n. The computer time for complete enumeration is given as t_{11}. For 7SP1, only 4 steps of look-ahead is required to locate the optimum network. The execution time is 3.66 s vs. 28.08 s for complete enumeration. The 5 look-ahead steps also locate the same optimal network. Only 3 steps look-ahead is required for the 7SP2 with an execution time of 1.54 s vs. 22.94 s for complete enumeration. Using 4 and 5 look-ahead steps produce the same result. Thus the heuristic evaluation function appears to be an extremely powerful technique for solving the exchanger network problem when direct enumeration proves to be prohibitive. The ten streams 10SP1 is one such problem and it has here been solved by using $n = 5$. The resultant network, which may be optimal or may only be suboptimal, is shown in Figure 7.

EXTENSIONS

Although the decision tree formulation to the exchanger problem provides an efficient means of representing the entire feasible network, the basic combinatorial difficulty of the problem has not been eliminated. This difficulty is also inherent in all other methods such as those of Lee et al. (1970). Since all the possible matchings of each primary stream must be considered prior to applying the method of branch and bound, the first part of their formulation has essentially enumerated all possible networks. Thus the saving that is brought about by applying the branch and bound technique to locate the optimal network will be impossible to realize if the first part of their formulation cannot be enumerated successfully. This suggests that only a synthesis by the partial enumeration technique can provide the computational efficiency that is required for solving practical applications.

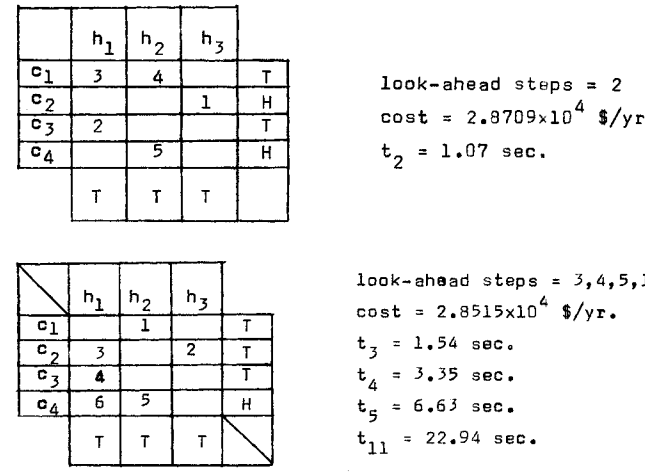


Fig. 6. Synthesis of exchanger network for Problem 7SP2 using various look-ahead steps.

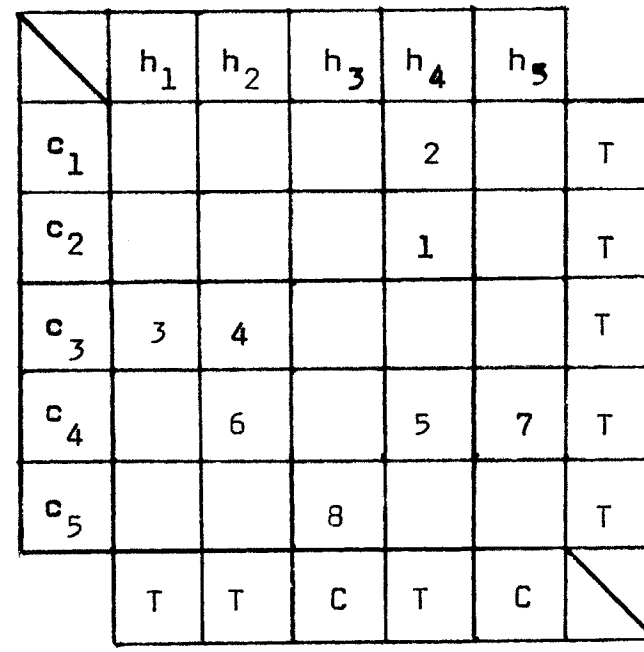


Fig. 7. Synthesis of exchanger network for Problem 10SP1 using 5 look-ahead steps.

In the partial enumeration method, only the nodes with the smallest evaluation function are retained and expanded. The algorithm is equivalent to enumerating all the nodes along a most promising tree branch. If the optimal node lies on this tree branch, it will, in general, be located by this method. However because of the heuristic nature of the evaluation function, it is not possible to guarantee that the optimal node will lie on this tree branch. Therefore, it may prove advantageous to examine more than just one promising branch. This can be achieved rather easily by always retaining the nodes with the first m lowest evaluation functions and expanding these nodes in the order in which they are generated. In this way as many as m^N tree branches will be examined, $m > 1$ but probably $m < 5$, rather than the single most promising one.

To reduce the computation time it is possible to make use of the problem constraints as generated by previous experiences or physical limitations due to specific layout of the process streams. As an example, the following simplifications can be useful in reducing the tree size:

1. Exclude the set of nodes which specify a vapor to vapor match. The low heat transfer efficiency of this matching suggests this feature.
2. Avoid matching streams that can cause start-up and control problems, for example, reflux and feed streams in distillation columns.
3. Avoid matching streams that are physically too far apart.

By identifying these infeasible matches, the symbol * can be assigned to their appropriate entries prior to construction of the decision tree.

ACKNOWLEDGMENT

The authors wish to acknowledge the support of this work from National Science Foundation Grant NSF-GP-24730. Furthermore, this work made use of the Princeton University Computer facilities supported in part by National Science Foundation Grant NSF-GJ-34 and GU-3157.

NOTATION

$a(i, j)$ = (i, j)th entry of A-matrix
c_i = cold stream i
C = water cooler
h_j = hot stream j
H = stream heater
k_n = $j_n + N_h(i_n - 1)$
N = $N_c \times N_h$
N_c = number of cold streams
N_h = number of hot streams
S_1^n = set feasible entries in i_nth row and j_nth column
S_2^n = set of feasible entries whose k values are larger than $j_n + N_h(i_n - 1)$
T = symbol indicates output temperature of process stream has met specification and requires no utility service

LITERATURE CITED

Hwa, C. S., "Mathematical Formulation and Optimization of Heat Exchanger Networks Using Separable Programming," *AIChE-Intern. Chem. Eng. Symp.* Ser., No. 4 (1965).

Kesler, M. G., and R. O. Parker, "Optimal Networks of Heat Exchanger," *Chem. Eng. Progr. Symp. Ser. No. 92*, **65** (1969).

Kobayashi, S., T. Umeda, and A. Ichikawa, "Synthesis of Optimal Heat Exchange Systems—An Approach by the Optimal Assignment Problem in Linear Programming," *Chem. Eng. Sci.*, **26**, 1367 (1971).

Lee, K. F., A. H. Masso, and D. F. Rudd, "Branch and Bound Synthesis of Integrated Process Designs," *Ind. Eng. Chem. Fundamentals*, **9**, 48 (1970).

Nilsson, N. J., *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, New York (1971).

Masso, A. H., and D. F. Rudd, "The Synthesis of System Designs—II Heuristic Structuring," *AIChE J.*, **15**, 10 (1969).

Nishida, N., S. Kobayashi, and A. Ichikawa, "Optimal Synthesis of Heat Exchange Systems—Necessary Conditions for Minimum Heat Transfer Area and their Application to System Synthesis," *Chem. Eng. Sci.*, **27**, 1408 (1972).

Samuel, A., "Some Studies in Machine Learning Using the Game of Checkers II. Recent Progress," *IBM J. Res. Develop.*, **11**, 601 (1967).

APPENDIX I

Given a synthesis sequence $k_1k_2 \ldots k_n$ representing uniquely a n step network, the problem is to find the set $\{k_{n+1}\}$ so that the sequence $k_1k_2 \ldots k_nk_{n+1}$ where $k_{n+1} \in \{k_{n+1}\}$ is also the unique representation of a $n + 1$ step network.

Let a $n + 1$ step network be denoted by $L_1 = k_1k_2 \ldots k_nk_{n+1}$. Since the sequence $k_1k_2 \ldots k_n$ represents uniquely a n step network, this means the relative position of each entry with respect to the other entries is fixed. Therefore the only other possible representation of L_1 is by placing k_{n+1} in between two consecutive entries, that is, $L_2 = k_1k_2 \ldots k_{i-1}k_{n+1}k_i \ldots k_n$. We examine two properties of k_{n+1} which can make this possible.

Case 1:

If k_{n+1} is noninteracting with all k_i, $i = 1, 2, \ldots n$, then L_1, L_2 represent the same network. Let this set of k_{n+1} be denoted by $\overline{Q}$.

Case 2:

If k_{n+1} is interacting with k_{i-1} but noninteracting with all $k_i, k_{i+1}, \ldots k_n$, then L_1, L_2 represent the same network. Let this set of k_{n+1} be denoted by p^{i-1}.

The assignment rules for defining the set $\{k_{n+1}\}$ is described as follows: In Case 1, if k_{n+1} from the set $\overline{Q}$ is always chosen so that it is larger than all the entries k_i, $i = 1, 2, \ldots n$, then the representation L_2 can never occur during the expansion of the decision tree. This is because in L_2 we have $k_{n+1} > k_i$ which violates the assignment rule. Similarly in Case 2, if we choose from the set p^{i-1} only those k_{n+1} whose values are larger than $k_i, k_{i+1}, \ldots k_n$, then the occurrence of L_2 can be prevented. Notice that for those entries not chosen as k_{n+1}, there exists a valid representation which has been accounted for in the earlier synthesis. This follows from the same proof as Theorem 2.

The determination of $\overline{Q}$ and p^i are derived as follows: Since S_1^i denotes the set of all feasible entries in the A-matrix which is interacting with k_i, it is clear that p^i is a subset of S_1^i. The set of all entries in S_1^i which are not interacting with $k_{i+1}, k_{i+2}, \ldots k_n$ will be taken as p^i. The set of all feasible entries interacting with at least one k_i, $i = 1, 2, \ldots n$ is given by the union of all S_1^i, $i = 1, 2, \ldots n$, that is, $Q = \bigcup_{i=1}^{n} S_1^i$. Thus its complement will give $\overline{Q}$. Notice that the ring sum of both Q and $\overline{Q}$ includes all the feasible entries in the A-matrix. This shows that the assignment rules for determining the set $\{k_{n+1}\}$ will encompass all feasible $n + 1$ step networks.

Since the representation to any one step network is always unique, the assignment rules can be applied recursively with n varying from 1 to $N - 1$, where $N = N_c \times N_h$. This procedure will result in a tree where each node is the unique representation of the network it represents.

Manuscript received February 7, 1973, and accepted July 9, 1973.