Selection of Decompositions for Chemical Process Simulation

A method has been developed for selection of the set of torn streams in a process simulation that leads to convergence of a direct substitution calculation in the minimum number of iterations. It is based on the establishment of families of decompositions, the members of each of which have identical convergence behavior. Explicit criteria for choice among families have been found that lead, in almost every case, to a single optimum decomposition family. These results have been verified by application to several chemical process simulation examples.

RAVINDRA S. UPADHYE

and

EDWARD A. GRENS II

Department of Chemical Engineering
University of California
Berkeley, California 94720

SCOPE

Process simulation is a valuable and widely used technique in process development, design, and optimization. Simulations are computer implemented mathematical models for process plants operating at the steady state; they are composed of phenomenological or empirical models for the process units interlinked by flows of material and energy between units. The connections between units are the process streams and are described by stream variables (for example, component flows, enthalpies).

Since most chemical processes involve many recycle streams, that is, connections in the form of loops, the direct sequential solution of the mathematical models for the units in the process is not possible. In any attempt, unknown unit feed streams are encountered. The units must all be solved simultaneously, but this is not feasible for real nonlinear units. Thus an iterative solution procedure must be used. The most often used approach to simulation of recycle systems is that of decomposition. Values are estimated for the variables associated with a sufficient number of streams to allow calculation of all units to be made in some sequence; these streams are said to be torn. After calculation of all units the values calculated for these streams are compared with the estimates, and the process is repeated—that is, iterated—with corrected estimates until adequate agreement indicates convergence.

There are two important questions associated with this procedure: how should estimates be corrected, and which streams should be selected for tearing? There are a number of methods that can be used for correction, that is, convergence procedures. The most commonly used method is direct substitution—the use of values obtained for torn variables in one iteration as estimates of these variables for the next iteration. Direct substitution is used because of its simplicity and natural stability. More sophisticated methods, mainly independent acceleration of variables toward convergence or simultaneous Newton-Raphson iterations, are also used, but are often quite complicated and costly to apply.

The other question, that of selection of the decomposition, is what we are concerned with here. For a given convergence procedure, here that of direct substitution, some possible choices of torn streams will give simulations that converge much more rapidly than others. Means for finding these efficient decompositions are required. Methods that have been available for selection of decompositions usually have been based on tearing minimum streams or minimum variables (Sargent and Westerberg, 1964; Lee and Rudd, 1966; Christensen and Rudd, 1969). Unfortunately, these objectives are not consistent with a goal of minimum computational effort, at least for direct substitution iterations.

To minimize the computational cost of a process simulation using direct substitution, one must find the decomposition that converges, from a given starting estimate and to the desired precision, in the minimum number of iterations. Because of the very large number of decompositions possible for real processes an exhaustive examination of these possibilities is not practicable. Thus we have sought to avoid such a search by identification of classes of decompositions that can be eliminated, a priori, from consideration and by grouping decompositions with similar convergence properties. This work has been based on a fundamental examination of the properties of the directed graphs constituting the flow diagrams for the processes in question.

CONCLUSIONS AND SIGNIFICANCE

We have found that decompositions for direct substitution simulation can be grouped into families, the members of each family having identical convergence behavior. These decomposition families can be considered as a generalization and extension of the concept of unit ordering. The order of unit calculations (cyclicly permutable) characterizes some families, but no such ordering can represent the majority of families, and the restricted concept of ordering provides no basis for selection of the decomposition (or ordering) to be used.

The number of decomposition families is smaller than the number of decompositions by a factor of the number of process units. There are three types of families: those

Correspondence concerning this paper should be addressed to E. A. Grens II. R. S. Upadhye is with Air Products and Chemicals, Inc., Allentown, Pennsylvania 18105.

containing only nonredundant compositions, those containing only redundant decompositions, and those containing both. On the basis of a number of arguments we conclude that the convergence behavior of families with only nonredundant members should be superior to that of other types of families. Further, from extensive examination of real chemical processes we find that, for these processes, only one nonredundant decomposition family exists. From these results we have outlined a simple procedure to locate a decomposition in the nonredundant family, and, in turn, all decompositions in that family. Any of these will have optimum convergence properties in direct substitution simulation. These conclusions have been verified for several process examples, some taken from the literature. The procedure developed here thus allows selection of a decomposition on the rational basis of most rapid convergence and can lead to simulations with considerably better efficiency than those with decompositions selected arbitrarily or to minimize torn variables or streams.

REPRESENTATION OF RECYCLE PROCESSES

A recycle process can be conveniently represented by a directed graph containing cycles or circuitous paths. The nodes (or vertices) of the graph represent process units, and the edges represent the process streams. Many methods exist to locate the cycles in such a directed graph (Norman, 1965; Tiernan, 1971). A simple cycle includes two or more streams and crosses no mode (unit) more than once. A stream may be included in more than one cycle; tearing a stream opens all cycles in which it is included. A valid decomposition is a set of torn streams that opens all cycles at least once. A redundant decomposition is a valid decomposition from which at least one stream can be removed without rendering the resulting decomposition invalid; a nonredundant decomposition has no such stream.

A simple process example is shown in Figure 1. The nature of the connections between the units of this process can be represented by the cycle/stream matrix shown in Table 1. Here the rows represent the cycles and the columns represent the streams. An entry in this matrix is unity if the corresponding stream is included in the cycle concerned and is otherwise blank. The last row in the matrix shows the number of variables in the respective streams.

In the simulation of a recycle process a decomposition is first selected, and initial values of the variables in the torn streams are estimated. (It will be assumed here that all variables in any stream are torn together.) Then the units in the process are calculated in turn, with their input variables having either the torn values or values yielded as outputs of previous units. The sequence of unit calculations is iterated until convergence is achieved.

CONVERGENCE BEHAVIOR OF DIRECT SUBSTITUTION SIMULATIONS

The method of direct substitution (DS) is the simplest, and therefore most commonly used, method for iterative calculation in process simulations. In this approach the set of values obtained for the torn variables as unit outputs during an iteration is used as the set of estimates for the torn variables for the next iteration. This procedure is repeated until changes in torn variables are satisfactorily small.

TABLE 1. CYCLE/STREAM MATRIX FOR PROCESS OF FIGURE 1

	Stream number					
Cycle	1	2	3	4	5	6
1			1			1
2	1	1			1	
3	1	1	1	1		
No. of variables	3	2	2	4	2	3

In concise form, this DS procedure can be written

$$\mathbf{x}^{(r+1)} = \varphi(\mathbf{x}^{(r)}) \tag{1}$$

where $\mathbf{x}^{(r)}$ and $\mathbf{x}^{(r+1)}$ are the values of the vector of torn variables at the end of iteration r and iteration $r+1$, respectively, and φ is a vector operator whose nature is determined by the process units, process arrangement, and the decomposition chosen. If $\mathbf{x}$ represents the true value of the torn variables, and the error vector at the end of iteration r is defined as

$$\boldsymbol{\epsilon}^{(r)} \equiv \mathbf{x}^{(r)} - \mathbf{x} \tag{2}$$

then for DS in the vicinity of the solution,

$$\boldsymbol{\epsilon}^{(r+1)} = \mathbf{J}\,\boldsymbol{\epsilon}^{(r)} \tag{3}$$

where $\mathbf{J}$ is the Jacobian matrix defined by

$$\mathbf{J} \equiv [J_{ij}] \equiv \left[\frac{\partial \phi_i}{\partial x_j}\right] \quad \text{for all } i, j \tag{4}$$

For the simulation to converge $\mathbf{J}$ must be a convergent matrix; this implies that the spectral radius of $\mathbf{J}$ must be less than unity. The spectral radius can also provide a measure of the rate of convergence: the smaller the spectral radius, the greater the rate of convergence.

φ is not an explicit function. Therefore, $\mathbf{J}$ must be determined indirectly from the unit sensitivities of the process units. If $\mathbf{y}$ and $\mathbf{z}$ are the vectors of input and output variables of a process unit, the relationship between $\mathbf{y}$ and $\mathbf{z}$ can be expressed as

$$\mathbf{z} = A(\mathbf{y}) \tag{5}$$

Then the unit sensitivity matrix $\mathbf{F}$ for this unit is defined as

$$\mathbf{F} \equiv [F_{ij}] \equiv \left[\frac{\partial \mathbf{z}}{\partial \mathbf{y}}\right] \tag{6}$$

evaluated at an estimate of the true value of $\mathbf{y}$ in the process.

For a process where the torn variables do not interact, the off-diagonal elements of $\mathbf{J}$ are zero. It then follows from Equation (3) that

$$\epsilon_j^{(r+1)} = J_{jj}\,\epsilon_j^{(r)} \quad \text{for all } j \tag{7}$$

where J_{jj} is the diagonal term in $\mathbf{J}$ for the variable x_j. If one considers only the magnitude of ϵ_j, repeated applications of Equation (7) yield

$$\epsilon_j^{(r)} = (J_{jj})^r\,\epsilon_j^{(0)} \tag{8}$$

and

$$\ln \epsilon_j^{(r)} = r \ln J_{jj} + \ln \epsilon_j^{(0)} \tag{9}$$

Therefore, the graph of $\ln \epsilon_j^{(r)}$ against r in this case would be a straight line. The slope of this line would be a measure of the rate of convergence of the variable x_j in the particular decomposition. If the off-diagonal terms of $\mathbf{J}$ are

not zero but small compared to the diagonal terms, such a graph would still be reasonably linear. When interactions are more significant, Equation (9) is a less satisfactory approximation and the graph of $\ln \epsilon_j^{(r)}$ vs. r may deviate considerably from linearity. In case of severe interactions, the contribution of $\epsilon_j^{(r)}$ *to* $\epsilon_j^{(r+1)}$ may not dominate, and the plot ceases to be useful except for process systems whose units have linear or nearly linear behavior.

CLASSIFICATION OF DECOMPOSITIONS ACCORDING TO CONVERGENCE BEHAVIOR

In general, different decompositions converge differently under direct substitution. Some lead to calculations that approach the final value uniformly while others oscillate. Some decompositions lead to faster convergence than others, and there may be decompositions that preclude convergence.

Unit Orderings

Frequently in specification of a decomposition for direct substitution only unit orderings are considered. Once a process computation is started, the particular unit ordering (or one of its cyclic permutations) repeats itself during successive iterations. There is, thus, no difference, as far as convergence of the process computations is concerned, between the cyclic permutations of any unit ordering. This behavior pattern can be illustrated by a simple example. For the process shown in Figure 1, the convergence of three different decompositions, {1, 6}, {2, 3} and {2, 5} is compared in Tables 2a, b, and c, respectively, by representation of successive symbolic values for stream variables (one per stream). All of these decompositions are started with arbitrary initial values. The units here are expressed as algebraic transformation operators; no assumptions about linearity are made. Note that the + signs in the arguments of the unit operators are not arithmetic addition signs; they only serve to separate the different inputs to a particular unit. Also the subscripts on unit operators serve to distinguish different unit outputs.

It is seen from Tables 2a, b, and c that given suitable starting values, the decompositions {1, 6} and {3, 5} yield identical values in subsequent iterations and must therefore converge identically. On the other hand, it is seen that {2, 3} does not exhibit this property even though corresponding initial values are used. In fact, no possible choice for the initial values of streams 2 and 3 will change this behavior; whatever values one starts with, at some stage (and thereafter)the decomposition {2, 3} will differ from the other two decompositions. Similar examination of {2, 6} and {4, 5, 6} shows that they exhibit the same behavior as shown by {1, 6}, and thus have the same convergence properties. Examination of {1, 3} and {2, 4, 6} shows that they are similar to {2, 3}. The reasons for this behavior superficially can be attributed to the unit orderings corresponding to these decompositions. The decompositions {1, 6}, {2, 6}, {3, 5}, and {4, 5, 6} all have the unit orderings that are cyclic permutations of ABCD. On the other hand, the decompositions {2, 3}, {1, 3}, and {2, 4, 6} have unit orderings which derive from ABDC or ADBC and thus have different convergence properties from those of {1, 6} and its related decompositions. Any unit ordering (or associated group of orderings) has both a characteristic convergence behavior and many corresponding decompositions. We may look upon these unit orderings as families of decompositions having the same convergence behavior. Such an attribute for a unit ordering is to be expected, and such orderings are thus often considered instead of decompositions.

However, there exist decompositions that cannot be represented by a unit ordering. For the process of Figure 1, {3, 5, 6} is such a decomposition. The only unit that can be calculated initially is D; but that creates a new value for stream 6 before its existing value is used in the calculation of unit C. Similar situations arise for many other redundant decompositions. Only nonredundant decompositions are necessarily characterized by unit orderings although the orderings usually give rise to many more redundant than nonredundant decompositions.

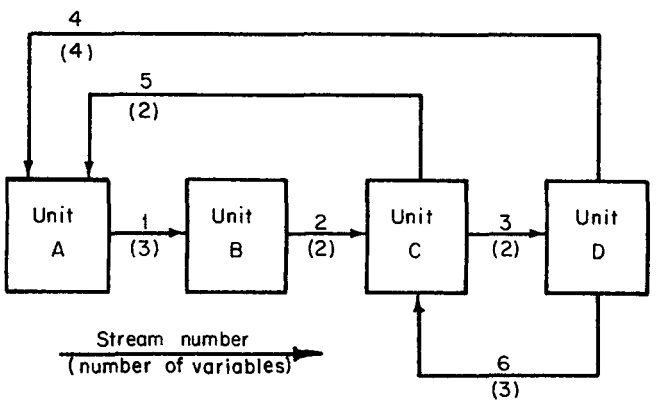

Fig. 1. Flow diagram for simple process.

Table 2. Behavior of Decompositions for Process of Figure 1

(a) Decomposition {1, 6}

$$\left.\begin{aligned} S_6 &= y \\ S_1 &= x \end{aligned}\right\} \text{initial values}$$
$$S_2 = Bx$$
$$S_3 = C_1(Bx + y)$$
$$S_5 = C_2(Bx + y)$$
$$S_4 = D_1C_1(Bx + y)$$
$$S_6 = D_2C_1(Bx + y)$$
$$S_1 = A(D_1C_1(Bx + y) + C_2(Bx + y))$$
$$S_2 = BA(D_1C_1(Bx + y) + C_2(Bx + y))$$
$$S_3 = C_1(BA(D_1C_1(Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$
$$S_5 = C_2(BA(D_1C_1(Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$
$$S_4 = D_1C_1(BA(D_1C_1Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$

(b) Decomposition {2, 3}

$$\left.\begin{aligned} S_3 &= C_1(Bx + y) \\ S_2 &= Bx \end{aligned}\right\} \text{initial values}$$
$$S_4 = D_1C_1(Bx + y)$$
$$S_6 = D_2C_1(Bx + y)$$
$$S_5 = C_2(Bx + D_2C_1(Bx + y))$$
$$S_1 = A(D_1C_1(Bx + y) + C_2(Bx + D_2C_1(Bx + y)))$$

(c) Decomposition {3, 5}

$$\left.\begin{aligned} S_3 &= C_1(Bx + y) \\ S_5 &= C_2(Bx + y) \end{aligned}\right\} \text{initial values}$$
$$S_4 = D_1C_1(Bx + y)$$
$$S_6 = D_2C_1(Bx + y)$$
$$S_1 = A(D_1C_1(Bx + y) + C_2(Bx + y))$$
$$S_2 = BA(D_1C_1(Bx + y) + C_2(Bx + y))$$
$$S_3 = C_1(BA(D_1C_1(Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$
$$S_5 = C_2(BA(D_1C_1(Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$
$$S_4 = D_1C_1(BA(D_1C_1(Bx + y) + C_2(Bx + y)) + D_2C_1(Bx + y))$$

Decomposition Families

It is, therefore, desirable to replace the severely limited concept of families of decompositions characterized by

unit orderings with a concept that is unambiguous and includes all possible valid decompositions. Such a formulation has been found possible; it is based on the following theorem.

Theorem 1: (Replacement Rule): Let $\{D_1\}$ be a valid decomposition. Let A_i be a unit such that all its inputs are included in the set $\{D_1\}$. (At least one such unit must exist, otherwise $\{D_1\}$ cannot be valid.) Replace all the inputs of A_i in $\{D_1\}$ by all the outputs of A_i. Let the new decomposition be $\{D_2\}$. Then:

(a) $\{D_2\}$ is also a valid decomposition, and
(b) $\{D_2\}$ and $\{D_1\}$ have the same convergence properties as far as calculations by direct substitutions are concerned.

A proof of Theorem 1 is given in Appendix A.

For the process shown in Figure 1, {2, 6} and {3, 5} have the same convergence properties, according to Theorem 1, because the inputs of unit C (streams 2 and 6) are replaced by the outputs of the same unit (streams 3 and 5). Similarly, {1, 6} and {4, 5, 6} have the same convergence behavior. In this way a family of decompositions can be defined as a set of decompositions linked by this Replacement Rule. Thus the decompositions {1, 6}, {2, 6}, {3, 5}, and {4, 5, 6} constitute one family, whereas {1, 3}, {2, 3}, {2, 4, 5} and {3, 4, 5} constitute another family.

It is possible to identify the decomposition families by the decompositions they contain. This, however, becomes cumbersome for large processes. One would also like to avoid the need for tracing from one decomposition to another by repeated applications of the Replacement Rule to establish whether the two given decompositions fall within the same family. For these reasons, it is convenient to establish an identification parameter for a family based on the cycle/stream matrix of the process. If $\{D_1\}$ and $\{D_2\}$ are two decompositions in a family, and if the cycle i is opened p times in D_1, then D_2 also opens this cycle p times. Thus if the cycle vector for a decomposition is defined as the vector (dimension a, where a is the number of cycles) whose ith element is equal to the number of times cycle i has been opened by the decomposition, this cycle vector can serve to identify the decomposition family. For example, in the process of Figure 1 and for the decomposition {1, 6}, stream 1 opens cycles 2 and 3, and stream 6 opens cycle 1. The cycle vector for {1, 6} is thus given by $\mathbf{c}_{\{1,6\}} = (1, 1, 1)^T$, and this will also correspond to other decompositions in the family with {1, 6}. Similarly, the cycle vector for {2, 3} and other decompositions of the same family is given by $(1, 1, 2)^T$.

A more formal definition of the cycle vector is possible. Let $\mathbf{w}_{\{D\}}$ be a vector, here called the decomposition vector for the decomposition $\{D\}$, the ith element of which is equal to the number of times stream i is torn in the decomposition $\{D\}$. Let $\mathbf{L}$ be the cycle/stream matrix for the process. Then the cycle vector for the decomposition $\{D\}$ is defined as

$$\mathbf{c}_{\{D\}} \equiv \mathbf{L}\,\mathbf{w}_{\{D\}} \tag{10}$$

Not only do all decompositions in a family have the same cycle vector, but it also appears (based on extensive examination of cases) that the vector uniquely corresponds to a single family.

There are several interesting implications of the existence of decomposition families. First, as far as the rate of convergence of direct substitution is concerned, the number of torn streams or variables is not important. Secondly, some redundant decompositions converge identically as certain nonredundant decompositions. For example, {2, 3} and {2, 4, 6} are in the same family, even though they are nonredundant and redundant decompositions, respectively. In fact, decomposition families can be grouped into three classes. A nonredundant family is one containing no redundant decomposition. A mixed family contains both redundant and nonredundant decompositions. A redundant family contains only redundant decompositions. For any process there must be at least one nonredundant family of decompositions (see Appendix B, Theorems 2 and 3).

Having defined decomposition families, we can consider the consequences of this grouping for the problem of selection of best decompositions. Since decompositions can be grouped into families, each having its own convergence behavior, it is only necessary to select among families rather than among decompositions. This reduces the dimensions of the problem of selection of decompositions by at least a factor of the number of units (u) in the process (see Appendix B, Theorem 4). The choice between redundant, nonredundant, and mixed families can then be considered. Regardless of this choice, the grouping of decompositions solely by unit orderings is not adequate. If redundant decompositions are desirable, unit orderings omit most of these. If only nonredundant decompositions are desirable, the unit orderings do not distinguish those in nonredundant families from those in mixed families (which have convergence equivalent to some redundant decompositions). Furthermore, the convergence behavior of several unit orderings may be identical (they correspond to members of the same family). The concept of decomposition families developed here, on the other hand, provides a general and valuable framework for the selection of decompositions for use in direct substitution.

In this regard, straightforward application of the Replacement Rule leads to further interesting and useful results. If one generates all the decompositions in a mixed or redundant family, some of them will be found to contain one or more streams multiple times. For example, for the process shown in Figure 1, the family corresponding to the decomposition {1, 3} will consist of {1, 3}, {2, 3}, {2, 4, 6}, {3, 4, 5}, and {4, 4, 5, 6}. The last decomposition contains stream 4 twice and is here called a *double-tear decomposition*. It is a special case of a redundant decomposition. Note that {4, 4, 5, 6} is different from {4, 5, 6} and that they have different convergence characteristics (belong to different decomposition families). Double-tear decompositions are quite legitimate and can be implemented strictly in accordance with the rules of direct substitution although they are frequently excluded from consideration. Table 3a shows the implementation of {4, 4, 5, 6} and confirms that it does indeed behave identically with {2, 4, 6} (shown in Table 3b), and hence with {2, 3}, which is a nonredundant decomposition. A double-tear may be regarded as a one iteration lag in the connection for the variables for the

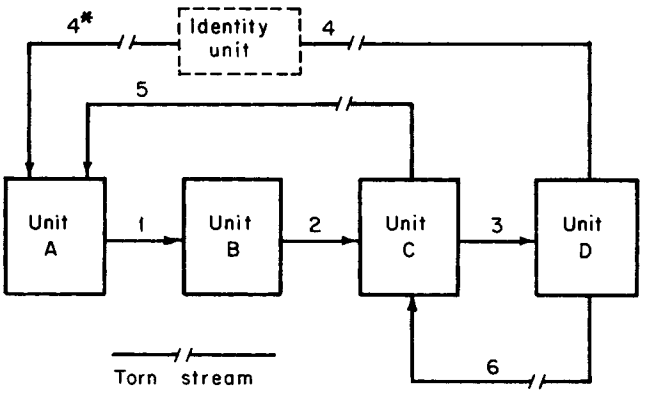

Fig. 2. Interpretation of double-tear decomposition {4,4*,5,6}.

TABLE 3. BEHAVIOR OF DOUBLE-TEAR DECOMPOSITION FOR PROCESS OF FIGURE 1

(a) Decomposition {4, 4, 5, 6}

$$
\begin{aligned}
S_5 &= C_2(x+z) \\
S_4^{\bullet} &= y \\
S_4 &= D_1C_1(x+z) \\
S_6 &= D_2C_1(x+z)
\end{aligned}
\Big\} \text{ initial values}
$$

$$
\begin{aligned}
S_1 &= A(y + C_2(x+z)) \\
S_2 &= BA(y + C_2(x+z)) \\
S_3 &= C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_5 &= C_2(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_4^{\bullet} &= S_4 = D_1C_1(x+z) \\
S_4 &= D_1C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_6 &= D_2C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_1 &= A(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) \\
S_2 &= BA(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) \\
S_3 &= C_1(BA(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) + D_2C_1(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z)))
\end{aligned}
$$

(b) Decomposition {2, 4, 6}

$$
\begin{aligned}
S_2 &= x \\
S_4 &= y \\
S_6 &= z
\end{aligned}
\Big\} \text{ initial values}
$$

NOTE: Decomposition {2,3} would have starting values: $S_2 = BA(y + C_2(x + z))$ and $S_1 = C_1(x+z)$ to give identical behavior

$$
\begin{aligned}
S_3 &= C_1(x+z) \\
S_5 &= C_2(x+z) \\
S_1 &= A(y + C_2(x+z)) \\
S_2 &= BA(y + C_2(x+z)) \\
S_4 &= D_1C_1(x+z) \\
S_6 &= D_2C_1(x+z) \\
S_3 &= C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_5 &= C_2(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_1 &= A(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) \\
S_2 &= BA(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) \\
S_4 &= D_1C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_6 &= D_2C_1(BA(y + C_2(x+z)) + D_2C_1(x+z)) \\
S_3 &= C_1(BA(D_1C_1(x+z) + C_2(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z))) + D_2C_1(BA(y + C_2(x+z)) \\
&\quad + D_2C_1(x+z)))
\end{aligned}
$$

doubly-torn stream.* In general, multiple-tears (that is, triple-tear, quadruple-tear etc.) can be regarded in a similar manner.

Any family that contains a redundant decomposition must also contain at least one double-tear decomposition (see Appendix B, Theorem 2). Thus any redundant decomposition must have the same convergence behavior as some double-tear decomposition. Therefore, decomposition families can be viewed as falling into two major classes: those containing no double-tear decomposition and those containing at least one double-tear decomposition.

SELECTION OF EFFICIENT DECOMPOSITIONS FOR DS SIMULATION

Since the presence or absence of double-tears is one major factor distinguishing between decomposition families, it is appropriate to investigate the convergence properties of decompositions with double-tears in comparison with corresponding decompositions with a single-tear in place of the double-tear. If double-tearing a stream could be shown, in general, to be undesirable, it would follow that all families containing decompositions with double-tears would also be undesirable. This is indeed the case, at least as far as chemical processes are concerned.

It was mentioned earlier that the spectral radius of the Jacobian of a decomposition is a measure of the rate of convergence of that decomposition. For any given Jacobian, if the derived Jacobian resulting from double-tearing an already torn stream had a larger spectral radius, it would follow that the decomposition with the double-tear was less rapidly convergent. Unfortunately, such an effect of double-tearing does not hold for a general Jacobian matrix. Parlett (1973) has shown that this behavior is necessary only for nonnegative Jacobians. Since Jacobians corresponding to real processes are not necessarily nonnegative, a strong mathematical argument for the undesirability of double-tearing streams cannot be established on this basis.

However, weaker arguments that indicate the undesirability of double-tears can still be made. It has been observed that the number of negative entries in most Jacobians derived from chemical processes is quite small because of the necessarily positive sensitivities arising in material and energy conservation requirements. Since the spectral radii of matrices are continuous functions of matrix elements, an increase in the radii with double-tearing might be expected for matrices with few negative elements.

It can also be shown (Upadhye, 1974) that for a process with essentially no interactions between torn variables, double-tearing a stream must degrade the convergence rate. Interactions among torn variables, however, do exist and may be important. As mentioned earlier the graph of the logarithm of error against iteration number is a straight line for a process where the torn variables do not interact, and as the interactions increase such a graph for a nonlinear process deviates more and more from linearity. The form of such a graph, therefore, may be an indication of the severity of interactions between the torn variables. For most simulations of real chemical processes, these graphs are, at least after the initial iterations, very nearly linear, indicating, in the presence of highly nonlinear units, some form of diagonal dominance. This observation has a rational explanation. Most chemical units are conservative. For such units, a tendency to diagonal dominance is a natural consequence of the laws of conservation of mass and energy. The general approach of the Jacobians of real chemical processes to diagonal or nearly diagonal matrices is therefore another basis for the expectation that double-tearing of streams should degrade the convergence rate of process simulations.

Finally, for all of the simulations based on chemical processes that have been examined in this work, double-tearing of one or more streams has led to degraded convergence behavior. These processes range from very simple to moderately complex and involve most types of interactions among variables and interconnections among units that are encountered in chemical process plants.

From these arguments and observations we have concluded that decompositions with double-tears are computationally inefficient and should be avoided. Acceptance of this criterion also leads us to reject all decompositions in families which contain one or more double-tears or multiple-tears. Thus only nonredundant families are candidates for selection.

The number of nonredundant families is, generally, small for connected graphs with the extent of looping found in chemical processes. For example, out of some 300 chemical processes represented by flow diagrams published in the recent *Hydrocarbon Processing Petrochemical Handbook* issues (1971, 1973), only one process, containing a trivial

* Another interpretation of a double-tear, which introduces an identity unit in the doubly-torn stream, is shown in Figure 2.

unit (heater for circulating Dowtherm) and related insignificant streams, was found to have more than one nonredundant decomposition family. Although these flow sheets are considerably simplified, the same observation applies for a number of detailed process flow sheets that were examined. Even if all significant streams are considered, only those processes whose configurations are very complex and where nearly all possible connections between units are present can be expected to have two or more nonredundant families. Thus, in a vast majority of cases, to choose the best decomposition for DS simulation it is only necessary to find the single nonredundant decomposition family and to select any convenient decomposition in that family.

Based on this criterion a procedure for selection of computationally efficient decompositions has been developed. Starting with the cycle/stream matrix for the process in question, consider any stream j. Define a weighting factor b_j for stream j as

$$b_j \equiv \sum_{i=1}^{a} L_{ij} \qquad (11)$$

where a is the number of cycles in the process and L_{ij} is the entry for the jth stream and the ith cycle in the cycle/stream matrix. Then find a decomposition with minimum weighted sum of streams, the weights being the b_j's so defined. Many methods to accomplish this minimization are available (such as those by Lee and Rudd, 1966; Upadhye and Grens, 1972), and any convenient method may be used. The decomposition found in this way can never belong to a mixed or redundant family (Upadhye, 1974). Also, the decomposition thus found is not unique; other members of the corresponding family can be found (if one wishes to see if they might be more convenient in application) by repeated applications of the Replacement Rule to this decomposition.

In almost every case, the decomposition family corresponding to this decomposition will be the only nonredundant family for the process; if the cycle vector corresponding to this family consists of all unity entries, this family must be the only nonredundant one. If the process should have more than one nonredundant family, this method will yield the family with the smallest sum of elements of the cycle vector. There is no simple way of determining whether there exist other nonredundant decomposition families. One either must assume that there is only one nonredundant family, an assumption that is very good as shown earlier, or must resort to much more time-consuming procedures.

APPLICATION TO EXAMPLE PROCESSES

The characteristics of convergence behavior described here, and the validity of the decomposition selection procedure based thereon, have been confirmed for direct substitution simulations of several processes of varying degrees of complexity. Three of these, derived from cases previously reported, are briefly discussed here.

1. Cavett's process (Cavett, 1963): This process consists of four flash drums connected as shown in Figure 3. Mixing units are here shown in place of simple junctions, where two or more streams are mixed. The design details of the units and feed composition are given by Cavett.

2. Rosen's process (Rosen, 1962): This process consists of two stirred-tank reactors, one flash drum, three separators and two junctions (here shown as mixing units), connected as shown in Figure 4. The output of the second reactor is flashed, the liquid going through the separation system, and part of the vapor is recycled. The design details of the units and the compositions of the feed and make up are given by Rosen.

3. Ramji's dimerization process (Ramji, 1967): This is the simplest of all the processes considered here. It consists of one reactor, one junction, one stream splitter and one separator, connected as shown in Figure 5. The design details of the units and the composition of the feed are given by Ramji.

Computer implemented simulations of these processes were conducted using decompositions from the single nonredundant family in each case, as well as from mixed and redundant families. The number of iterations required for convergence of these simulations using direct substitution (to a fixed tolerance limit of 0.1% relative change in torn variables) are given in Table 4. In all cases, starting from common initial estimates for torn variables, the decompositions in each family converged in the same number of iterations and decompositions from the nonredundant family converged more rapidly than those from mixed or

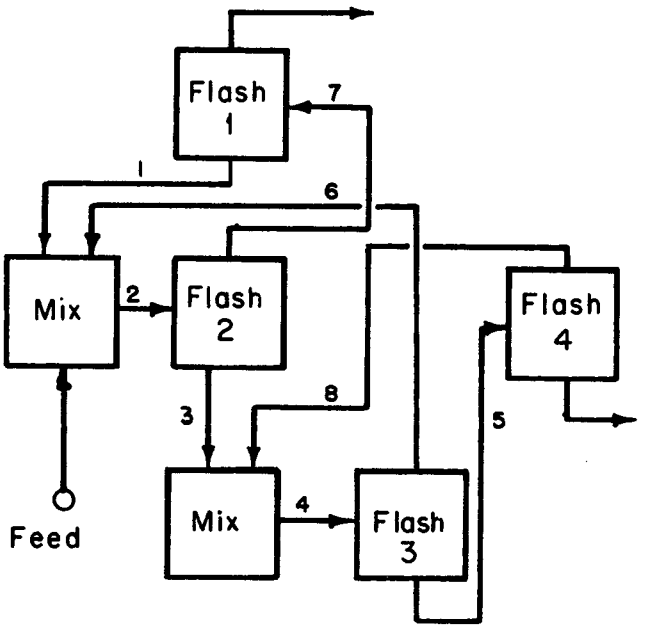

Fig. 3. Cavett's process (1963).

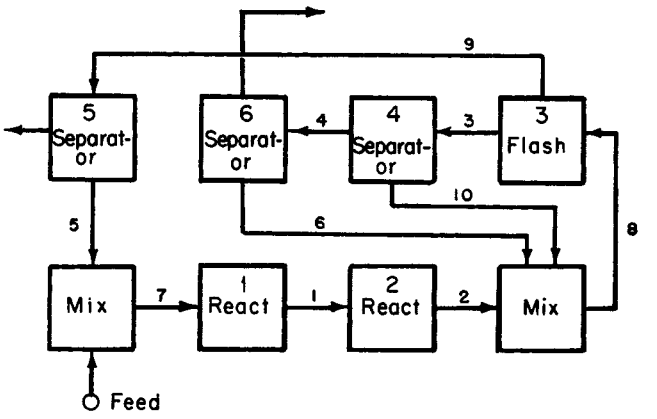

Fig. 4. Rosen's process (1962).

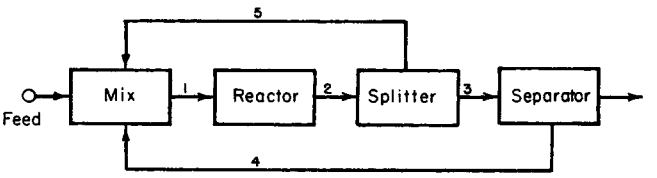

Fig. 5. Ramji's dimerization process (1967).

TABLE 4. CONVERGENCE OF SIMULATIONS FOR PROCESS EXAMPLES

Process	Decomposition	Decomposition family type	Iterations*
Cavett	{2, 5}, {2, 8}, {5, 6, 7}	Nonredundant	63
	{2, 5, 7}, {5, 6, 7, 7}	Redundant	71
	{2, 4}, {5, 6, 6, 7}	Mixed	90
Rosen	{8}, {5, 6, 10}	Nonredundant	12
	{8, 10}, {5, 6, 10, 10}	Redundant	19
	{6, 8}, {5, 6, 6, 10}	Redundant	19
Ramji	{1}, {2}, {3, 5}	Nonredundant	29
	{1, 5}, {3, 5, 5}	Redundant	41
	{1, 3}, {3, 3, 5}	Redundant	>50**

* Number of iterations, from common starting estimates for torn variables, to achieve 0.1% maximum relative change in torn variables.
** No convergence in 50 iterations-computations terminated.

redundant families. In some cases the advantage of a decomposition selected by criteria developed here is modest (12 to 60%); in other cases the advantage may be on the order of a factor of two or more.

ACKNOWLEDGMENT

This work was partially supported by a grant from the National Science Foundation. Computer support was provided by the Computer Center, University of California, Berkeley.

NOTATION

a = number of cycles in a process
A = unit transformation operator
b_j = weighting parameter for stream j as defined in Equation (11)
$\mathbf{c}$, $\mathbf{c}_{\{D\}}$ = cycle vector (a vector parameter defined in Equation (10))
$\{D\}$ = general representation of a decomposition
$\mathbf{F}$ = unit sensitivity matrix
$\mathbf{J}$ = Jacobian matrix for a process simulation
$\mathbf{L}$ = cycle/stream matrix
r = iteration number
u = number of units in a process
$\mathbf{w}_{\{D\}}$ = decomposition vector for the decomposition $\{D\}$
$\mathbf{x}^{(r)}$ = vector of torn variables at iteration r
$\mathbf{y}$, $\mathbf{z}$ = vectors of process input and output variables, respectively

Greek Letters

ϵ_j = error in variable x_j
φ = vector transformation operator representing one iteration of a process simulation

LITERATURE CITED

Cavett, R. H., "Applications of Numerical Methods to the Convergence of Simulated Processes Involving Recycle Loops," A.P.I. Div. Ref., **43**, 57 (1963).
Christensen, J. H., and D. F. Rudd, "Structuring Design Computations," *AIChE J.*, **15**, 94 (1969).
Gulf Publishing Company, "Petrochemical Handbook Issue," *Hydrocarbon Processing*, **50**(11), 113 (1972); **52**(11), 89 (1973).
Lee, W., and D. F. Rudd, "Ordering of Recycle Calculations," *AIChE J.*, **12**, 1184 (1966).
Norman, R. L., "A Matrix Method for Location of Cycles of a Directed Graph," *ibid.*, **11**, 450 (1965).
Parlett, B. N., Personal communication (1973).
Ramji, R., "Computational Aspects of Process Simulation," M.S. Thesis, Univ. California, Berkeley (1967).
Rosen, E. M., "A Machine Computation Method for Performing Material Balances," *Chem. Eng. Progr.*, **58**, 69 (1962).
Sargent, R. W. H., and A. W. Westerberg, "SPEED-UP in Chemical Engineering Design," *Trans. Inst. Chem. Engrs.*, **42**, T190 (1964).
Tiernan, J. C., "An Efficient Search Algorithm to Find the Elementary Circuits of a Graph," *Comm. ACM*, **13**, 722 (1970).
Upadhye, R. S., "Selection of Decompositions for Simulation of Chemical Processes," Ph.D. thesis, Univ. California, Berkeley (1974).
———., and E. A. Grens, "An Efficient Algorithm for Decomposition of Recycle Processes," *AIChE J.*, **18**, 533 (1972).

APPENDIX A: PROOF FOR REPLACEMENT RULE

The Replacement Rule (Theorem 1) links members of a decomposition family by replacement in a decomposition of input streams to a unit by the output streams of the same unit. Its validity can be established as follows:

Denote the set of streams into unit A by $\{A'\}$ and the set of streams out of unit A by $\{A''\}$.

Let one iteration be completed using the torn values in $\{D_1\}$, with unit A being computed first. This is quite general, since A can be defined as the unit that is calculated first. This gives the values of the streams in the set $\{A''\}$. Assume values for the streams in $\{D_2\}$ such that

1. all the streams common to $\{D_1\}$ and $\{D_2\}$ have the same initial values in $\{D_2\}$ and $\{D_1\}$
2. all the values for $\{A''\}$ are assumed as returned by the one iteration performed for the decomposition $\{D_1\}$.

If one now performs one iteration for $\{D_2\}$, the results, (that is, the values of the variables in the process streams) would be the same in both cases. This is so because the state of the system after the computation of unit A in scheme $\{D_1\}$ is identical to the state of the system at the beginning of computations in scheme $\{D_2\}$. For the computation of any subsequent units, their input values are identical in both cases. This will be repeated iteration after iteration, and thus the values of process streams will be identical for both the decompositions. Therefore, $\{D_1\}$ and $\{D_2\}$ must converge identically.

APPENDIX B: PROOFS FOR THEOREMS 2-4

Important properties of decomposition families are embodied in the three basic theorems that are presented here.

Theorem 2: If a decomposition family contains at least one redundant decomposition, it also contains at least one double-tear decomposition.

Proof: Let $\{\{D_i\}, S_k\}$ be a valid decomposition. Let $\{D_i\}$ also be a valid decomposition. Also, let $S_k \notin \{D_i\}$. Thus, $\{\{D_i\}, S_k\}$ is a general redundant decomposition. Further, let S_k be an output of unit A.

Using the arguments presented in the proof of Theorem 1, one can show that by repeated applications of the Replacement Rule to the decomposition $\{D_i\}$ one must eventually reach a decomposition where all the inputs of unit A are torn. One more application of the Replacement Rule would then yield a decomposition where S_k is doubly torn.

Theorem 3: For any process flow diagram, there exists at least one nonredundant decomposition family.

Proof: Consider any valid decomposition $\{S_1, S_2, \ldots, S_n\} = \{D\}$. Generate all the members of the family to which this decomposition belongs. Then either

1. no member of the family is a double-tear decomposition or
2. at least one member of the family is a double-tear decomposition.

In case (1), the theorem is proved, as Theorem 2 establishes that all mixed and redundant families contain double-tear decompositions.

In case (2), denote the decomposition with a doubly-torn stream by $\{\ldots, S_i, S_i, \ldots\}$. Then the second S_i can be removed from this decomposition to get another valid decomposition, which will be in some other family.

Then the same argument can be repeated for the new family, and any double-tear decompositions can be similarly dealt with.

Since all the cycles in the process must be opened, and since the number of streams in a decomposition is finite, such a procedure must terminate and lead to a nonredundant family.

Theorem 4: A family of decompositions contains at least u decompositions, where u is the number of units in the looped process system.

Proof: A process flow graph for a recycle system is a connected graph, that is, there exists a directed path from any vertex to any other vertex. Consider a decomposition $\{D_1\}$ where unit 1 has all its inputs torn. One application of the Replacement Rule would replace these inputs by the outputs of unit 1. Call this decomposition $\{D_2\}$. Once again, there exists a unit with all inputs torn. Call it unit 2. This process of replacement and renumbering of units can be continued for at least u units. Therefore, there must be at least u decompositions in any decomposition family.

Manuscript received August 2 and accepted September 13, 1974.

Some Fundamental Aspects of Spray Drying

Equations are proposed to predict the three-dimensional motion of droplets in a spray dryer, based on a knowledge of the characteristics of the atomizing device and of the gas flow patterns in the drying chamber. If the droplet size distribution produced by the atomizing device is known or can be assumed, the trajectories of the droplets can be calculated throughout the drying process and hence the evaporative capacity and thermal efficiency of the spray dryer can be predicted.

Experimental verification of this theoretical approach was obtained from a study of the drying of calcium lignosulfonate solutions of various concentrations in a 122-cm diam. × 183-cm high laboratory circular concurrent chamber with a conical bottom where the drying air was introduced tangentially near the top.

An experimental study of the effects of a number of operating variables on the capacity and the efficiency of the spray dryer was also carried out. These effects were interpreted in terms of the droplet trajectories obtained in each case.

S. KATTA and W. H. GAUVIN

Department of Chemical Engineering
McGill University
Montreal, Quebec, Canada

SCOPE

Among the many drying methods available, spray drying has gained a unique and important position in industrial applications where low particle temperatures and short residence times are specially advantageous, such as in the drying of foods, drugs, and temperature-sensitive materials in general. Prediction of droplet trajectories and residence times is essential for the sound design and efficient operation of spray dryers. Such knowledge can also be applied in the design of a number of other equipment involving the contacting of a dispersed phase of droplets or particles with a conveying gas, such as spray coolers, gas scrubbers and absorbers, cyclone evaporators, pneumatic transport reactors, and combustion devices involving fuel sprays. Many attempts were made in the past to predict droplet trajectories in spray dryers, but these were based on unrealistic or oversimplified models, which failed to take into account the many complex interdependencies between the transport phenomena occurring during the drying process and thus generally resulted in poor agreement between theoretical predictions and the actual performance of the system.

The objective of the present study was to predict the trajectories of droplets both in the nozzle zone (the region traversed by the rapidly-decelerating droplets between the exit of the atomizing device and the point at which they begin to be freely entrained by the drying gas) and in the free-entrainment zone (where the droplets are freely conveyed by the drying gas) of an experimental spray dryer, using solutions of calcium lignosulfonate (trade-mark LIGNOSOL) as the model material. Predictions of drop size distribution (DSD) and of the size of the largest droplet generated by the nozzle were also developed, as this knowledge is vital for the calculation of the trajectories. The three-dimensional equations of motion of the droplets, along with the equations for particulate heat transfer, for the mass transfer of water vapor and for the changes in the properties of the drying gas were solved numerically and simultaneously on a computer.

To test these theoretical predictions, the effect of a number of operating variables such as the temperature and flow rate of the drying gas, the position of the atomizing nozzle in the chamber, and the concentration of the feed on the evaporative capacity and the thermal efficiency of the spray-drying chamber have been studied experimentally. These experimental results were interpreted in terms of the droplet trajectories obtained under the various operating conditions used.

In an earlier paper, Gauvin et al. (1974) predicted droplet trajectories for the evaporation of water sprays in the same equipment as was used in the present study.