LETTERS TO THE EDITOR

To the Editor:

In the paper entitled "Decomposition of Systems of Nonlinear Algebraic Equations" (**30**, p. 92, Jan., 1984), M. Shacham demonstrates how considerable savings in computation can be achieved by exploiting linear subsets of systems of nonlinear equations. Furthermore, he describes a procedure for minimizing the number of nonlinear equations by defining new variables to replace repeated nonlinear terms. Automating this procedure presents several problems, and I wish to comment on Shacham's approach to three of them.

Comparing Nonlinear Terms

In order to compare different nonlinear terms and establish whether or not they are the same, Shacham suggests associating a different prime number to each variable in the problem; terms that evaluate to the same numerical result are supposed to be the same.

However, the above is only true for expressions involving products or quotients of the unknowns but not necessarily of their sums or differences. Consider, for example, a problem with five variables; by assigning the first five prime numbers (3, 5, 7, 11 and 13) to X_1, X_2, X_3, X_4 and X_5 respectively, one observes that the nonlinear terms

$$\frac{X_3 - X_2}{X_2 - X_1} \text{ and } \frac{X_1 + X_2}{X_5 - X_2} \text{ and } \frac{X_5 - X_4}{X_3 - X_2} \text{ and } \frac{X_4 - X_3}{X_3 - X_1}$$

all evaluate to unity although they are totally different.

Furthermore, even if the above procedure were valid, it would not be practical, since some terms cannot be evaluated for arbitrary values of the unknowns (e.g., the term $\sqrt{X_1 - X_2}$ using the values $X_1 = 3$, $X_2 = 5$).

Although some form of algebraic manipulation seems inevitable, it may still be a good idea to evaluate the terms and only apply the symbolic comparison to those with the same value. For this purpose, one may use the initial values of the unknowns, or some random value between their lower and upper bounds, if these are available. The latter choice should minimize the probability of two different terms yielding the same value by accident.

Identifying the Nonlinear Terms in a Given Equation

Shacham does not deal with this problem. In fact it is not trivial and its solution certainly requires algebraic manipulation. Defining what exactly constitutes a nonlinear term is an additional problem; for instance, in the equation

$$AX_1X_2 + X_1X_2^2/X_3 = 0$$

with A being a constant and X_3 a design variable (i.e., not an unknown), it is advantageous to identify the nonlinear terms as (X_1X_2) and $(X_1X_2^2)$ rather than (AX_1X_2) and $(X_1X_2^2/X_3)$ in order to allow more scope for matching with terms in other equations.

Locating Profitable Definitions of New Variables

This involves identifying sets of k equations ($k \leq n$) containing fewer than k distinct nonlinear terms (NLT's); these equations can then be rendered linear by defining new variables for the NLT's, thus decreasing the total number of nonlinear equations in the system.

Shacham's algorithm for achieving this objective involves an initial simplification followed by a multistage search; at the kth stage, groups of k NLT's are tested against the above requirements. This procedure, however, may become too expensive for large systems if more than a couple of stages are required. In fact, examples may be constructed, for which the algorithm needs to examine every single combination of NLT's.

In view of this, a more efficient algorithm is proposed here. It is based on Hall's Theorem of Systems of Distinct Representatives (Hall, 1935), which, effectively, states that in a system of n equations, each equation can be assigned a different NLT, if and only if every subset of k equations ($1 \leq k \leq n$) contains at least k distinct NLT's.

The present problem is, therefore, equivalent to searching for cases where no such assignment is possible; it belongs to the family of assignment (or "maximum transversal") problems of network theory.

Our algorithm is best described by representing the equations and the NLT's as two disjoint sets of nodes in a bipartite graph. *Edge* $(i\text{–}j)$ exists if equation i contains the jth NLT. An *assignment* is a set of edges $(i\text{–}j)$ such that no node i or j appears in more than one edge in the set. Edges in the assignment are *matching* edges; a node is *exposed* if it does not appear in any matching edge. An *augmenting path* is a path with exposed nodes at both ends and alternating nonmatching and matching edges (if any) in the middle.

Initially all NLT's are labeled as **uncolored.** The assignment is set to the empty set.

Each equation is dealt with in turn. The algorithm attempts to find a feasible assignment by constructing augmenting paths emanating from this equation. (See Duff, 1981, for a very efficient procedure based on depth first search.)

If a complete augmenting path is found, the nonmatching edges on the path replace the matching edges in the assignment. Since the end nodes of the path are both exposed, this results in an increase of the number of edges in the assignment.

If no augmenting path can be found, no assignment is possible for the equation currently under consideration. The group of equations visited during the search for augmenting path can be profitably linearized by defining new variables for the (fewer) NLT's visited; the latter are now labeled as *colored*.

One can then proceed to the next equation until all n of them are considered. Then a new variable can be defined for each colored NLT.

Since during the examination of each equation, the algorithm can at most search each of the t edges in the graph once, its worst-case complexity is $O(nt)$. In practice, one does not usually require more than $O(n) + O(t)$ operations, as demonstrated in Duff, 1981. In fact, savings can be achieved by abandoning further search on a path if a colored NLT is reached; since pursuing this path was previously found to be unfruitful (as indicated by the color of the NLT), the algorithm can safely backtrack to the previous

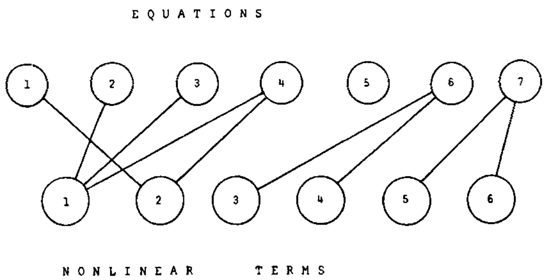

Figure 1. Bipartite graph for example 1.

node on the path and continue the search from there.

The proposed algorithm is demonstrated below by applying it to the two examples used by Shacham in his paper; the same numbering of equations and NLT's is employed.

Example 1: Partial Oxidation of Methane

The bipartite graph representing the system is shown in Figure 1. Initially the assignment is empty and all NLT's are uncolored. Each equation is considered in turn, using the corresponding node as the starting point for the search for an augmenting path.

Eq. 1: Visit NLT2; NLT2 is exposed and, therefore, a trivial augmenting path (E1 . . . NLT2) has been found. Add edge (E1,NLT2) to the assignment.

Eq. 2: Visit NLT1; again, NLT1 is exposed; edge (E2-NLT1) is added to the assignment.

Eq. 3: Visit NLT1. Note that (E3-NLT1) is a nonmatching edge, so the next edge in any augmenting path must be a matching one. The only possibility is (NLT1-E2); visit E2. Since no more progress is possible from E2, backtrack to E3. No more unexplored edges are available, and therefore no augmenting path exists. Color NLT1.

Eq. 4: Visit NLT1; NLT1 is colored; therefore backtrack to E4. Visit NLT2; visit E1 along matching edge (NLT2-E1); backtrack to E4. No further search is possible; color NLT2.

Eq. 5: No edges are available for search.

Eq. 6: Visit NLT3; NLT3 is exposed, thus add (E6-NLT3) to the assignment.

Eq. 7: Visit NLT5; NLT5 is exposed, thus add (E7-NLT5) to the assignment.

All equations have been considered. Define new variables for the colored nonlinear expressions NLT 1 and NLT 2.

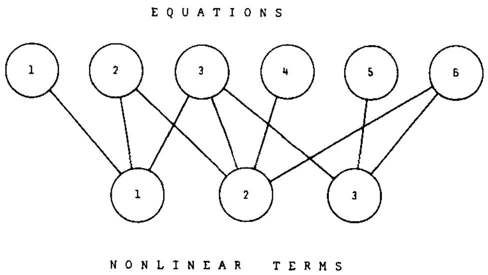

Figure 2. Bipartite graph for example 2.

Example 2: The Williams-Otto Process

The corresponding bipartite graph is shown in Figure 2. The stages of the algorithm follow:

Eq. 1: Visit NLT1, add (E1-NLT1) to the assignment.

Eq. 2: Visit NLT2, add (E2-NLT2) to the assignment.

Eq. 3: Visit NLT3, add (E3-NLT3) to the assignment.

Eq. 4: Visit NLT2; follow matching edge (NLT2-E2) to E2; follow nonmatching edge (E2-NLT1) to NLT1; follow matching edge (NLT1-E1) to E1. No further search is possible from E1; backtrack to E2. No further search is possible from E2; backtrack to E4. No further search is possible from E4; color NLT1 and NLT2.

Eq. 5: Visit NLT3; visit E3; visit NLT1; NLT1 is colored; backtrack to E3. Visit NLT2; NLT2 is colored; backtrack to E3. No further search is possible from E3; backtrack to E5. Color NLT3.

Eq. 6: Visit NLT2; NLT2 is colored; backtrack to E6. Visit NLT3, NLT3 is colored; backtrack to Eq. No further search is possible.

All equations have been considered. Define new variables for colored expressions NLT1, NLT2, and NLT3.

Literature Cited

Hall, P., "On Representatives of Subsets," *J. London Math. Soc.*, **10**, 37 (1935).

Duff, I. S., "On Algorithms for Obtaining a Maximum Transversal," *ACM Trans. Math. Software*, **7**, 3 (1981).

CONSTANTINOS C. PANTELIDES
Dept. of Chemical Engineering
Imperial College
London SW7, England