A computer aid for the analysis of complex systems

J. V. Hansen* and Lynn J. McKell

Institute of Professional Accountancy, Graduate School of Management, Brigham Young University, Provo, Utah 84602, USA

Analysts from many disciplines are actively involved in the construction of models which will enable better understanding of system structure, improve systems design, and aid in planning and decision making. One important obstacle to achieving these ends is the frequent necessity to involve resource people in the modelling process. Such persons are typically expert concerning a particular segment of system operation, but are not as a rule skilled model builders and may have only sketchy knowledge of the overall system structure. Thus, the analyst's challenge is to provide a means of working systematically to establish relationships among fragmented inputs from a body of experts. This paper reports on a computer aid which can be useful in combining the skills of analysts and resource people in the synthesis and design of complex systems. Illustrative applications to information systems and health care planning are included.

(Received January 1978)

1. Introduction

A significant amount of intellectual activity has been stimulated by people's attempts to construct models of complex systems which are of importance to their welfare. These may be physical systems, interpersonal relationships, biological systems, or public and societal systems. Increasingly efforts by applied scientists, systems analysts and decision analysts are being focused on the synthesis, comprehension and effective use of system models.

One major obstacle to system synthesis is that the model builder may not have sufficient expertise to determine what variables should be included in a particular model and what the relationships are between those variables. The model builder must therefore elicit information from those who are most knowledgeable (experts) about the system being modelled. Notwithstanding the reasonableness of this procedure, it is frequently the case that such experts are knowledgeable about only a specialised segment of the system being analysed, with the net result that the model builder constructs a fragmented system description. This leaves the difficult problem of connecting the fragments in a proper manner, as well as resolving the inevitable omissions and inconsistencies.

The objective of this paper is to develop and demonstrate concepts which can facilitate the analysis of complex systems toward improving model construction. The motivation for this effort stems from the problems noted above. A methodology is proposed which can mitigate these problems through:

(*a*) identifying key system elements and their relationships
(*b*) showing the existence or nonexistence of required elements and relationships
(*c*) defining levels in the system hierarchy
(*d*) specifying cycles—subsystems—within levels
(*e*) facilitating user (decision maker) participation in the system design and evaluation process.

First a brief summary of the nature and use of graph theory is presented. The methodology and computer software required to implement these concepts is then described. This is followed by two short application examples.

2. Graph theory

Over the past few years, graph theory has emerged as one of the most powerful tools for the representation and investigation of processes which are essentially sequential in nature. Many examples of its development and use can be cited. Sargent and Westerberg (1964) developed an algorithm in which a directed graph (digraph) is partitioned by tracing information flow backwards until nodes which had previously been encountered are encountered again. Steward (1965) and several others have used an adjacency matrix (see Appendix) to search for shortest paths in a digraph. An excellent summary of the use of graph theory to identify structure in chemical process simulation programs is presented by Kahat and Sacham (1973), which also contains a number of useful references. The work by Tarjan (1972) provides an excellent summary of computational algorithms for structure determination. Warfield (1974) has applied graph-theoretic concepts to the study of large scale societal systems.

Several graph-theoretic concepts are basic to the approach outlined in this paper. For the interested reader, the Appendix describes in some detail the necessary partitioning techniques and the algorithm for realising a structural model. (The term structural model derives from graph theory, as well, since many of its applications are motivated by a search for structure.)

3. Computer aided structural modelling

A primary advantage of structural modelling lies in the programmability of the approach, i.e. it can be systematised and implemented on a computer which can perform many of the routine functions—thus making structured modelling a practical tool for analysis.

Interpretive Structural Modelling Software (ISMS) is a computer based aid, utilising concepts summarised in the Appendix, which helps analysts think and communicate more effectively about complex design issues. ISMS has been designed in such a way that users are responsible for making all subjective judgements, and the computer is used in an unobtrusive manner for book-keeping and for displaying the results and implications of the judgemental decisions made. The noteworthy contribution of ISMS procedures (**Fig. 1**) is that they operate without the requirement of a prior knowledge of system structure.

ISM procedures

The ISM process is initiated by specification (by the analyst and his group of experts) of the set of elements comprising the system to be modelled. For example, if an organisational decision support system is being constructed, the elements could be the decisions and information sets important to the organisation's operation. Once these elements are input to ISMS, the software systematically interrogates its users with regard to the presence or absence of that relationship between

*Now at Graduate School of Business, Indiana University, Bloomington, Indiana 47401, USA.

pairs of elements which is of interest. For our example, the users might be primarily interested in queries of the form: '(Is) (element s_i) (required for) (element s_j)?'. That is: Is information element s_i required for decision s_j?; is decision element s_i required for updating information element s_j?, etc. Obviously, the number of pairwise queries required in a system of any size would make this task very tedious. To keep the task manageable, the computer is employed to keep track of responses supplied by the users to provide implicit transitive inferences based upon these responses. This allows for an efficient ordering of subsequent queries.

Following the input operation, the element set is partitioned following procedures discussed in the Appendix. The resulting digraph provides an efficient, hierarchically ordered display of both the direct responses and the indirect transitive inferences resulting from the input operation. The ISM algorithm removes all redundant links from the digraph; however, redundant links may be essential to convey the full meaning and pattern of the system. For this reason, ISMS provides for a comparison operation in which the user examines the result of the mathematical operations and heuristics of the process and

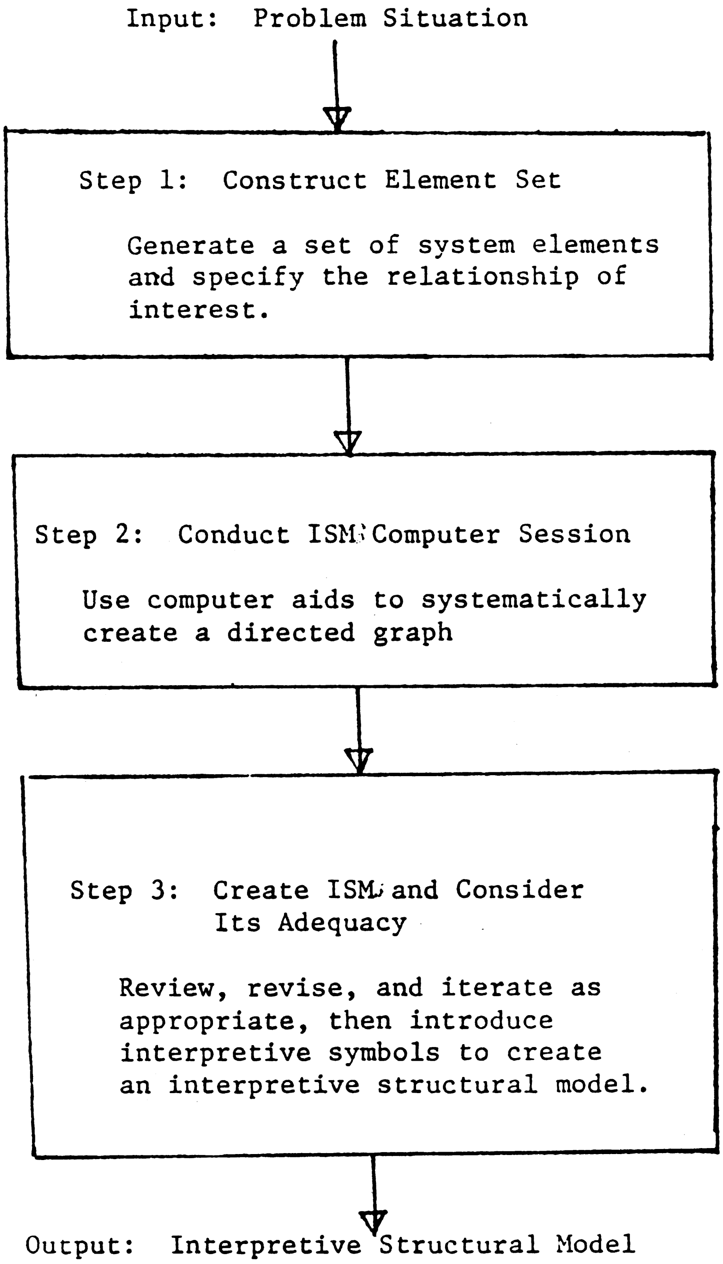

Fig. 1 The basic operational steps for application of the ISM technique

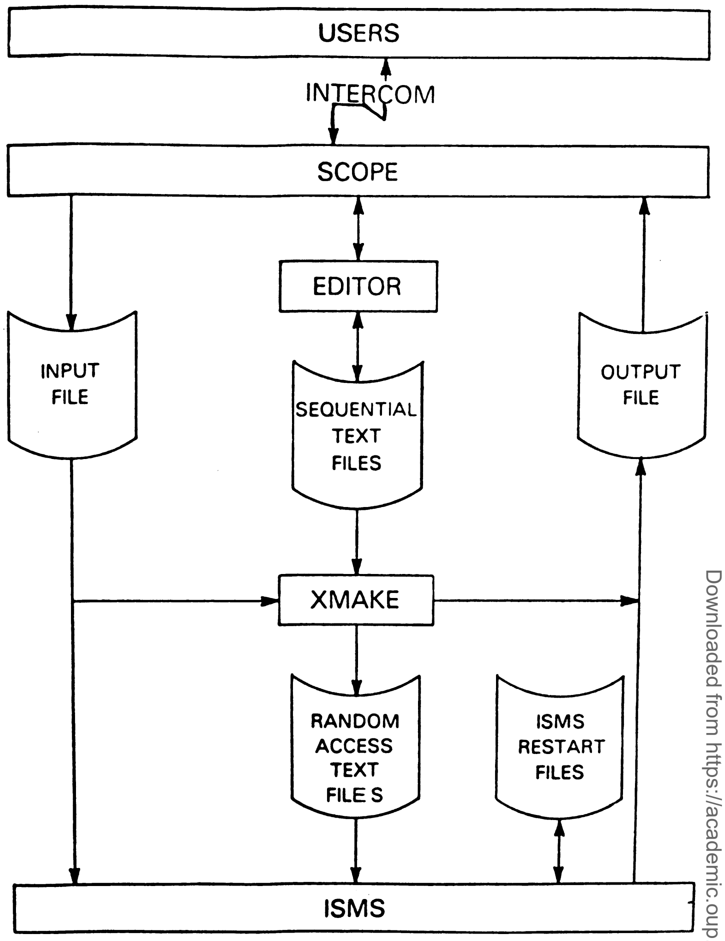

Fig. 2 ISMS files and data flow

introduces modifications or corrections to the digraph. The final operation consists of the introduction of elaborative text, interpretive symbols, or additional graphical embellishments which will make the final structural model comprehensible to a wider audience.

One might question the usefulness of ISMS in situations where a given relationship between two system elements may sometimes be true and sometimes not. Considering again our decision support system example, if decision s_j only sometimes requires information element s_i, the system structure certainly ought to provide for provision of that information element even though the relationship may not hold in a strict sense. There may be, however, other systems where contingency conditions may be more limiting.

ISMS data flow

Fig. 2 depicts the data flow among the various components of ISMS and its host computer. Users may access ISMS from a remote computer terminal and conduct a dialogue by means of responding to ISMS generated queries with either English phrases or symbolic responses to denote directed connectives between element pairs.

If the user desires ISMS to present queries in English text, an element text file must be created prior to using ISMS. These element text files consist of the English text for a relational expression and for those elements being considered. The EDITOR permits the creation and modification of sequential line-oriented files. The EDITOR responds to simple commands which initiate, maintain, correct and/or complete file construction. Although the EDITOR is a comprehensive editing package capable of performing varied tasks, the user's concern with respect to ISMS is minimal.

ISMS methodology requires that elements be accessible in a

SOURCE \ DESTINATION	PLANT MANAGEMENT	SHIPPING & RECEIVING	PURCHASING	PRODUCTION MANAGEMENT	PRODUCTION — DIVISION A	PRODUCTION — DIVISION B	MARKETING	STORES CONTROL	PRODUCTION SCHEDULING	INTERNAL PRICING	PLANT ACCOUNTING	DATA PROCESSING	PRODUCTION ENGINEERING	MAINTENANCE	TOOL ROOM
PLANT MANAGEMENT	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SHIPPING & RECEIVING	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
PURCHASING	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
PRODUCTION MANAGEMENT	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0
PRODUCTION - DIVISION A	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
PRODUCTION - DIVISION B	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
MARKETING	1	1	1	1	1	0	1	1	1	0	0	0	0	0	0
STORES CONTROL	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
PRODUCTION SCHEDULING	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0
INTERNAL PRICING	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0
PLANT ACCOUNTING	1	1	1	1	1	0	1	1	1	0	1	0	0	0	0
DATA PROCESSING	1	1	1	1	1	0	1	1	1	0	1	1	0	0	0
PRODUCTION ENGINEERING	1	1	0	1	1	0	0	0	1	0	0	0	1	0	0
MAINTENANCE	0	1	0	0	1	1	0	0	0	1	0	0	0	1	0
TOOL ROOM	0	1	0	0	1	1	0	0	0	1	0	0	0	0	1

Fig. 3 Reachability matrix of information flows

random order; that is, although the elements are ordered by the user in some desired sequential fashion prior to an ISM session, the introduction of responses to relational queries determines new sequential orders within the element set. To allow for this random ordering, a computer program called XMAKE was developed. XMAKE primarily restructures the EDITOR created sequential text file on to a new random text file. This new random text file contains all of the textual information of the sequential, but has the property that any specific element's text may be accessed independently of the positioning of the file due to a previous access.

ISMS restart files are created and maintained without user interaction. The purpose of these files is to capture the step-by-step development of the digraph data. This permits the user to terminate an ISMS session at selected break points prior to completion of analysis. The interrupted session may then be restarted at a later time. The restarted session is begun by ISMS access of the proper restart file to capture digraph development up to the point of interruption.

4. An application to structuring an information system

An application of ISMS to information systems analysis is suggested by the work of analysts such as Orden (1971). Following his procedure, a list of work centres is designated as shown in **Fig. 3**. The work centres represent functional areas such as production, marketing, and data processing. While the example represents a highly simplified organisation structure for a manufacturing firm, the procedures are readily extended to more complex structures.

Information flow is defined here as the periodic transfer, at regular or irregular intervals, of designated sets of information among work centres and other organisational elements. This information definition and format can be modified to accommodate nearly any information flow environment. Fig 3 represents a partial reachability matrix for a manufacturing organisation resulting from inputs to ISMS. For convenience, **Fig. 4** represents the same matrix with centre titles replaced by numbers. ISMS partitions this reachability matrix into seven blocks, each representing one system level. In particular,

$$\{B_1; B_2; B_3; B_4; B_5; B_6; B_7\}$$
$$= \{\overline{1, 2}; \overline{5, 6}; \overline{4, 9, 10}; \overline{3, 8, 13, 14, 15}; \overline{7}; \overline{11}; \overline{12}\}$$

These results may be readily observed from the canonical matrix shown in **Fig. 5.** Levels are identified by the heavy lines. Strongly connected subsets are indicated within levels by elements which reach at least one other element at the same level. Subsystems are identified by those elements of strongly connected subsets which are reachable from, and antecedent to, one another. For example, the first level has no subsystems. Elements 5 and 6 form a subsystem at the second level, etc. Reachability from one level to the next is indicated by the dashed lines.

Note that subsystems and levels have some similarities and some differences. A subsystem is formed by taking the smallest possible diagonal submatrix such that there are no 1's to the right in the matrix, while a level in a hierarchy is identified by taking the largest possible diagonal submatrix that is filled with zeros (except on the main diagonal) and has no 1's to the right.

*The use of some graph theoretic terms is unavoidable in discussing these applications. However, we have restricted terms to those which are most elementary. The reader is referred to the Appendix for definitions.

The final product of the ISMS applied to the production information system is illustrated in **Fig. 6.** With the new knowledge of each level, its subsystems and the types of information comprising each arrow, it is straightforward to determine the precedence relationships, and where they exist, among different kinds of information. In designing information systems, this knowledge can limit the extensive synchronisation problems which are endemic to information systems development. Further, specification of all successors of an information set can provide a complete knowledge of all operations which use that set as an input. This identification is important to the estimation of the utility of each information set to the system, since it shows where in the system the information will be used.

Additionally, in the analysis of information systems, it is important to be able to track, for any information which is produced, all its precedences of all orders. Langefors (1973) has observed that this knowledge may be required in connection with auditing because in this way it is found at what points in the system there is a possibility to manipulate the information produced.

5. An application to regional health care program planning

As another example of an application of ISMS, we consider a regional health care study outlining recommendations for 19 programs to be implemented. Each program was to have responsibility vested in a planning subcommittee, with opera-

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
4	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0
5	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
6	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
7	1	1	1	1	1	0	1	1	1	0	0	0	0	0	0
8	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
9	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0
10	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0
11	1	1	1	1	1	0	1	1	1	0	1	0	0	0	0
12	1	1	1	1	1	0	1	1	1	0	1	1	0	0	0
13	1	1	0	1	1	0	0	0	1	0	0	0	1	0	0
14	0	1	0	0	1	1	0	0	0	1	0	0	0	1	0
15	0	1	0	0	1	1	0	0	0	1	0	0	0	0	1

Fig. 4 Relabelled matrix of information flows

	2	1	5	6	10	9	4	14	15	13	3	8	7	11	12
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
10	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
9	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0
4	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0
14	1	0	1	1	1	0	0	1	0	0	0	0	0	0	0
15	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0
13	1	1	1	0	0	1	1	0	0	1	0	0	0	0	0
3	1	1	1	0	0	1	1	0	0	0	1	1	0	0	0
8	1	1	1	0	0	1	1	0	0	0	1	1	0	0	0
7	1	1	1	0	0	1	1	0	0	0	1	1	1	0	0
11	1	1	1	0	0	1	1	0	0	0	1	1	1	1	0
12	1	1	1	0	0	1	1	0	0	0	1	1	1	1	1

Fig. 5 Hierarchically arranged reachability matrix of information flows

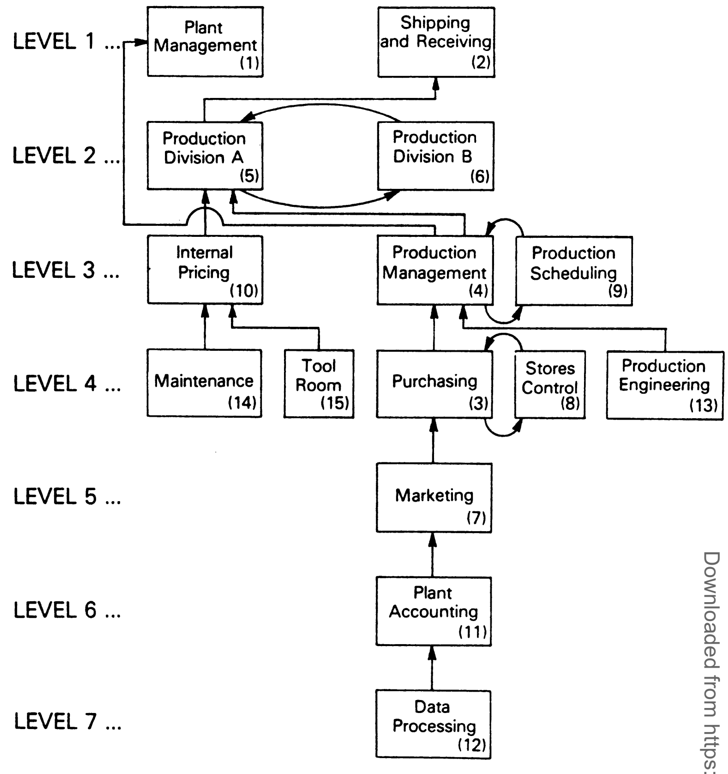

Fig. 6 Structural model of information system

tional action being carried out by agency staff. In all, the number of subcommittee program assignments necessary to implement the recommended programs concurrently was quite infeasible. Therefore, many of the needed programs were postponed.

For purposes of illustration, four of the 19 programs initially designed are summarised in **Fig. 7.** The structure of these programs was developed in an ad hoc manner based upon needs identified by the study.

An alternative approach to the structuring of program elements with inputs provided by regional planners is illustrated by the reachability matrix of **Fig. 8.** The product of this input to ISMS, the four programs developed by regional planners, is illustrated in **Figs. 9 and 10.**

Interestingly, the program design which has resulted reduces four programs to one. We do not propose that this is necessarily the optimal program plan, but that the thought process by which it was developed takes a more rational form and is less subject to omission of important relationships. Inspection of Fig. 8 reveals some apparent program fragmentation. However, it is improbable that the program design of **Fig. 11** would have resulted from closer intuitive scrutiny. This effort was undertaken by several staff members prior to using ISMS, with results varying little from those shown in Fig. 8.

Sackman (1970) has reported that major insight associated with computer assisted problem solving often occurs while engaged in group discussions at the computer terminal. Consistent with this observation, informal evaluations held after the computer assisted program design suggested that important changes resulted from the facilitated process of evaluating relationships and immediately receiving feedback on resulting system structure.

Such an approach may be of particular importance to the evaluation and/or design of complex health care systems. Generally, the number of elements to be considered is large, with the number of interactions comparable to the square of the number of elements. The logistics of dealing with so many interactions in an ad hoc manner is a major inhibitor to

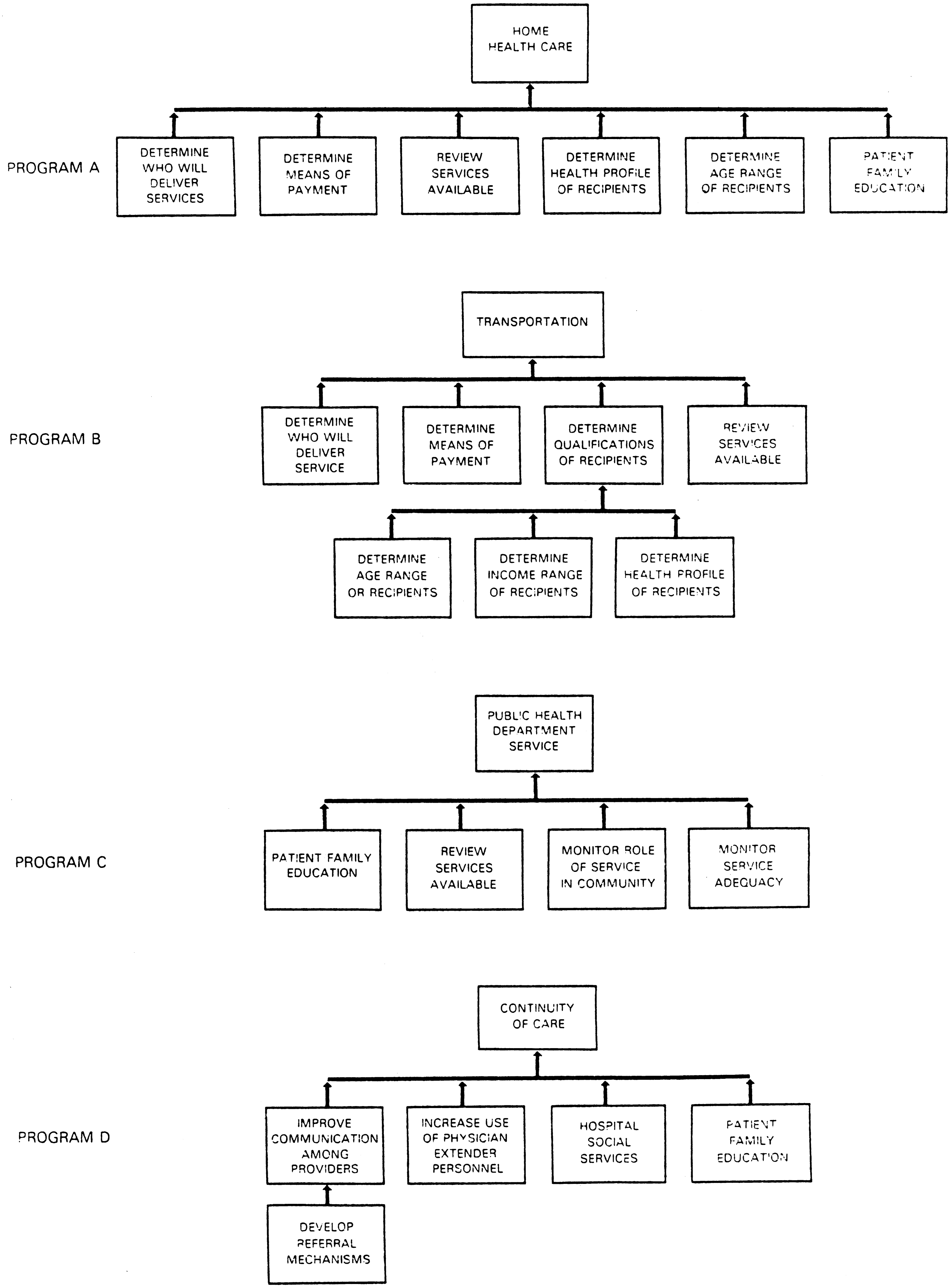

Fig. 7 Four of the initial 19 planned programs

SYSTEM ELEMENTS	1 HOME HEALTH CARE	2 TRANSPORTATION SERVICES	3 RESPONSIVENESS OF PUBLIC HEALTH DEPT. SERVICES	4 CONTINUITY OF CARE	5 DETERMINE AGE RANGE OF RECIPIENTS	6 DETERMINE INCOME OF RECIPIENTS	7 DETERMINE WHO WILL DELIVER SERVICE	8 DETERMINE MEANS OF PAYMENT	9 DETERMINE QUALIFICATIONS OF RECIPIENTS	10 DETERMINE HEALTH PROFILE OF RECIPIENTS	11 REVIEW SERVICES AVAILABLE	12 MONITOR ROLE OF SERVICE IN COMMUNITY	13 MONITOR SERVICE ADEQUACY	14 PATIENT/FAMILY EDUCATION	15 IMPROVE COMMUNICATION AMONG PROVIDERS	16 INCREASE USE OF PHYSICIAN EXTENDER PERSONNEL	17 REFERRAL MECHANISMS	18 HOSPITAL SOCIAL SERVICES
1. HOME HEALTH CARE	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2. TRANSPORTATION SERVICES	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3. RESPONSIVENESS OF PUBLIC HEALTH DEPARTMENT SERVICES	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1
4. CONTINUITY OF CARE	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5. DETERMINE AGE RANGE OF RECIPIENTS	1	1	0	0	1	0	1	0	1	0	0	1	1	0	0	0	1	1
6. DETERMINE INCOME RANGE OF RECIPIENTS	1	1	0	0	0	1	0	0	1	0	1	1	1	0	0	0	1	1
7. DETERMINE WHO WILL DELIVER SERVICE	1	0	0	1	0	0	1	0	1	0	1	1	1	0	0	0	1	0
8. DETERMINE MEANS OF PAYMENT.	1	1	1	1	0	0	1	1	0	0	0	0	0	1	1	0	1	1
9. DETERMINE QUALIFICATIONS OF RECIPIENTS	1	1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1
10. DETERMINE HEALTH PROFILE OF RECIPIENTS	1	1	1	1	0	0	0	0	1	1	0	0	0	1	0	0	1	1
11. REVIEW TRANSPORTATION SERVICES AVAILABLE	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
12. MONITOR ROLE OF SERVICE IN LOCAL COMMUNITY	1	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1
13. MONITOR SERVICE ADEQUACY	1	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	1
14. PATIENT/FAMILY EDUCATION	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1
15. IMPROVE COMMUNICATION AMONG PROVIDERS	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16. INCREASE USE OF PHYSICIANS EXTENDER PERSONNEL	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
17. REFERRAL MECHANISMS	1	0	1	1	0	0	0	0	0	0	1	1	1	1	0	1	1	0
18. HOSPITAL SOCIAL SERVICES	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fig. 8 Reachability matrix of program elements

conducting a thorough analysis of system structure and interrelationships. A tool such as ISMS allows concentration on the substantive rather than the logistical aspects of systems development—making it much easier for system structuring to evolve from group activity, an important consideration in health care where there are numerous areas of expertise.

Conclusion

The problem of systems complexity cannot be ignored; it affects too many aspects of our lives. Yet, even the complexity measurement process is of marginal aid in developing useful models. A more promising approach is through the development of methodologies which directly assist the model builder in designing and developing models of complex environments.

The ISMS can provide valuable assistance to the model builder through: (*a*) partitioning a system into a set of subsystems; (*b*) specifying subsystems sufficiently to permit separate testing before total system tests; (*c*) eliminating redundant elements and connections; and (*d*) facilitating expert and model builder co-ordination in the design process.

	2	4	1	15	18	3	11	12	13	14	16	17	9	5	6	7	10	8
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
11	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
12	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
13	1	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0
14	1	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0
16	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0
17	0	1	1	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
9	1	1	1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0
5	1	0	1	0	1	0	1	1	1	0	0	1	1	1	0	0	0	0
6	1	0	1	0	1	0	1	1	1	0	0	1	1	0	1	0	0	0
7	1	1	1	0	0	0	1	1	1	0	0	1	1	0	0	1	0	0
10	1	1	1	0	1	1	0	0	0	1	0	1	1	0	0	0	1	0
8	1	1	1	1	1	0	0	0	0	1	0	1	0	0	0	1	0	1

Fig. 9 Hierarchically arranged reachability matrix

Appendix

The concepts of utilising interaction matrices as tools for transforming mental models of the interaction between elements into a communicable format are well known (Harary, Norman and Cartwright, 1975; Kahat and Sacham, 1973; Sage, 1977). Only those concepts related to the application described in this paper are presented here.

If there is a set of system elements $\{s_i\}$ with a relation R^* among some members of the set, the existence of relation R^* can be represented by R and the absence of that relation by $\bar{R}$. Thus, a binary matrix J that represents R may be said to be a full description of the relation for the set. Consider any two elements of a system, s_i and s_j. If it is possible for the analyst to say either that s_i and s_j are related in a certain way ($s_i R s_j$) or

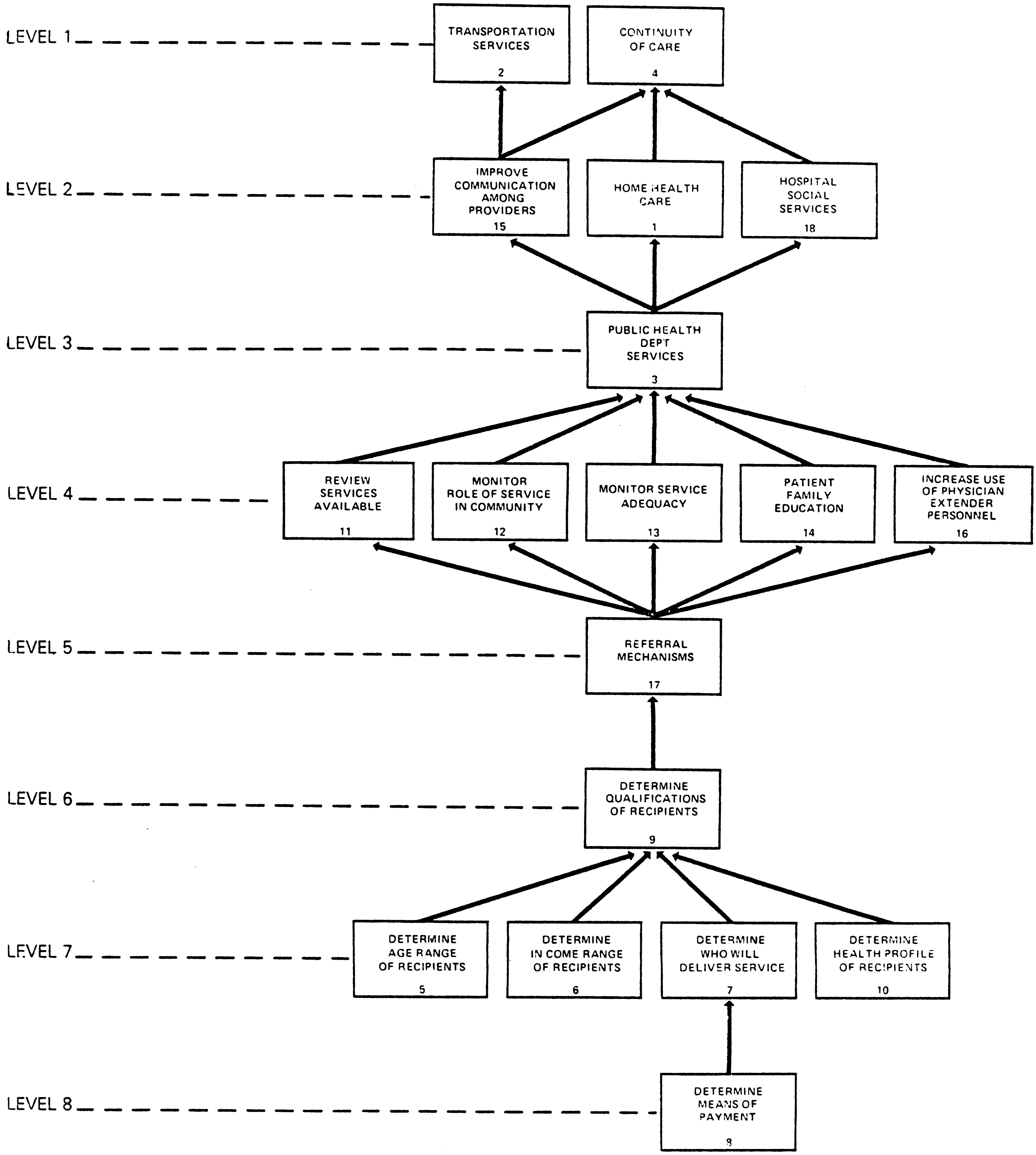

Fig. 10 Alternative program plan utilising structural modelling procedures

they are not ($s_i\bar{R}s_j$), then he can construct a matrix comprised of a set of n elements. This matrix J is termed the adjacency matrix.

An entry in position (i, j) is 1 if s_iRs_j and 0 if $s_i\bar{R}s_j$. A system graph can be constructed by allowing a vertex on the graph to represent a system element, and an edge joining two elements to represent a 1 in the matrix.

Constructing the reachability matrix

Consider the adjacency matrix shown in Equation (1).

$$J = \begin{array}{c|cccc} & s_1 & s_2 & s_3 & s_4 \\ \hline s_1 & 0 & 1 & 0 & 0 \\ s_2 & 1 & 0 & 0 & 1 \\ s_3 & 1 & 1 & 0 & 0 \\ s_4 & 0 & 0 & 0 & 0 \end{array} \qquad (1)$$

If there is a path from s_i to s_j, then s_j is reachable from s_i. The number of lines in the path s_i to s_j is termed the length of the path. The adjacency matrix simply describes reachability for all paths of length 1. By adding the identity matrix I to Equation (1), a matrix is obtained describing reachability for all paths of length 0 and length 1 (Equation (2)).

$$(J+I) = \begin{array}{c|cccc} & s_1 & s_2 & s_3 & s_4 \\ \hline s_1 & 1 & 1 & 0 & 0 \\ s_2 & 1 & 1 & 0 & 1 \\ s_3 & 1 & 1 & 1 & 0 \\ s_4 & 0 & 0 & 0 & 1 \end{array} \quad (2)$$

If Equation (2) is multiplied times itself, one obtains Equation (3), where all operations are Boolean.

$$(J+I)^2 = \begin{array}{c|cccc} & s_1 & s_2 & s_3 & s_4 \\ \hline s_1 & 1 & 1 & 0 & 1 \\ s_2 & 1 & 1 & 0 & 1 \\ s_3 & 1 & 1 & 1 & 1 \\ s_4 & 0 & 0 & 0 & 1 \end{array} \quad (3)$$

Equation (3) identifies reachability for all paths of length 2 or less. To observe this, examine an element of $(J + I)^2$. For $(J+I)^2_{ij} = 1$, we must have $(J + I)_{ik} = 1$ and $(J + I)_{kj} = 1$ for at least one value of k. This tells us that point i reaches point k by a path of length 1 or 0. Thus, there is a path of length 2 or less from point i to point j, and point i reaches point j by a path of length 2 or less.

This process of multiplying $(J + I)$ times itself continues until successive powers produce identical matrices; that is, until

$$(J + I)^{r-2} \neq (J + I)^{r-1} = (J + I)^r = S \quad (4)$$

where S is defined as the reachability matrix. The power r for which Equation (4) is defined will be less than or equal to the number of elements in the set. This must be true since the longest possible path for n elements is $n - 1$, and the higher power would only indicate reachability through paths of $n - 1$ or less.

Partitioning

Once S has been constructed, it can be partitioned to identify a number of model properties.

Returning to the example and applying Equation (4) reveals that

$$(J + I)^1 \neq (J + I)^2 = (J + I)^3 = S$$

That is,

$$S = \begin{array}{c|cccc} & s_1 & s_2 & s_3 & s_4 \\ \hline s_1 & 1 & 1 & 0 & 1 \\ s_2 & 1 & 1 & 0 & 1 \\ s_3 & 1 & 1 & 1 & 1 \\ s_4 & 0 & 0 & 0 & 1 \end{array} \quad (5)$$

The first partition may be written as $P_1(S) = \{V; \bar{V}\}$, where an ordered pair is contained in V if s_i reaches s_j, or is contained in $\bar{V}$ if s_i does not reach s_j. Thus, in the example (for notational convenience we use only the subscript indices)

$$V = [(1, 1), (1, 2), (1, 4), (2, 1), (2, 2), (2, 4)$$
$$(3, 1), (3, 2), (3, 3), (3, 4), (4, 4)]$$
$$\text{and } \bar{V} = [(1, 3), (2, 3), (4, 1), (4, 2), (4, 3)]$$

The second partition separates the element set into levels and may be expressed as $P_2(S) = \{L_1; L_2 \ldots; L_r\}$, where r is the number of levels. If the 0th level is defined as the empty set, $L_0 = 0$, the algorithm for determining levels may be expressed as $L_j = \{s_i \in S - L_0 - L_i \ldots - L_{j-1} \mid R(s_i) = R(s_i) \cap A(s_i)\}$ where $R(s_i)$ and $A(s_i)$ are the reachability and antecedent sets* determined for the subgraph consisting of the elements in $S - L_0 - L_i - \ldots - L_{j-1}$.

Table 1 **Reachability and antecedent sets for $S - L_0$**

	$R(s_i)$	$A(s_i)$	$R(s_i) \cap A(s_i)$
1	1, 2, 4	1, 2, 3	1, 2
2	1, 2, 4	1, 2, 3	1, 2
3	1, 2, 3, 4	3	3
4	4	1, 2, 3, 4	4

Table 2 **Reachability and antecedent sets for $S - L_0 - L_1$**

s_i	$R(s)$	$A(s)$	$R(s_i) \cap A(s_i)$
1	1, 2	1, 2, 3	1, 2
2	1, 2	1, 2, 3	1, 2
3	1, 2, 3	3	3

Table 3 **Reachability and antecedent sets for $S - L_0 - L_1 - L_2$**

S_i	$R(s_i)$	$A(s_i)$	$R(s_i) \cap A(s_i)$
3	3	3	3

Operating on our example, **Table 1** illustrates the sets $R(s_i)$, $A(s_i)$, and $R(s_i) \cap A(s_i)$ for $S - L_0$. Examination of this table indicates that $R(s_i) = R(s_i) \cap A(s_i)$ for element 4. Thus, $L_1 = 4$. We next delete L_1 from Table 1 and find the top level elements of $S - L_0 - L_1$ which will constitute L_2. This is illustrated by **Table 2.** Inspection reveals that $R(s_i) = R(s_i) \cap A(s_i)$ for elements 1 and 2; therefore $L_2 = [1, 2]$. We now delete L_2 from Table 2 and determine the top level elements for level three. **Table 3** lists $S - L_0 - L_1 - L_2$ and shows that $R(s_i) = R(s_i) \cap A(s_i)$ for the remaining element, 3; thus $L_3 = 3$.

Deletion of L_3 exhausts S and the partition is completed. Three levels have been identified, and $P_2(S)$ can be written as

$$P_2(S) = \{[4]; [1, 2]; [3]\}$$

The third partition $P_3(L_j)$ identifies strongly connected subsets within levels. If an element s_i is not part of a strongly connected subset, then $R_{Lj}(s_i) = s_i$, where $R_{Lj}(s_i)$ indicates reachability with respect to the elements of level L_j. The reachability matrix thus induces a two-block partition $P_3(L_j)$ on the elements of each level $P_3(Lj) = \{W; \bar{W}\}$. An element is contained in W if it is not part of a strongly connected subset. Otherwise, the element is contained in $\bar{W}$. For our example

$$P_3(L_1) = \{[4]; [\emptyset]\}$$
$$P_3(L_2) = \{[\emptyset]; [1, 2]\}$$
$$P_3(L_3) = \{[3]; [\emptyset]\}$$

In this case each $P_3(L_j)$ shows either W or $\bar{W}$ as being empty. This is a special case and cannot be generalised. In general, W or $\bar{W}$ may be empty but not both.

The fourth partition, $P_4(\bar{W})$ identifies cycles. A subset of elements may be identified by $P_4(L_j)$ and placed in the $\bar{W}$ block; but they do not all necessarily belong to the same strongly connected subset. The reachability matrix induces a partition $P_4(\bar{W})$ on the strongly connected subsets such that each group of elements represents a cycle if and only if every element in the group is reachable from, and antecedent to, every other element in the group. In the example $P(L_2)$ identified $\bar{W} = [1, 2]$. Inspection of the reachability matrix shows that both elements 1 and 2 are antecedent to and reachable from each other. Therefore, $P_4(\bar{W}) = \{1, 2\}$.

Determining the structural model

After computing these partitions, the elements of the reachability matrix may be rearranged to obtain canonical form.

*The antecedent set is simply the set of elements which are subordinate to s_i.

That is, the horizontal index set is arranged in left-to-right order, and the vertical set is arranged in the order, $W_1, \overline{W}_1, W_2, \overline{W}_2, \ldots U_k, W_k$, where the subscripts indicate the level for which the W and $\overline{W}$ partitions are defined. The arrangement by levels allows a property identification. Sets of submatrices may be defined as follows: group elements of the same level in the index sets, and use these level groups to define the submatrices. For a structural model with levels, the matrix or submatrices would be:

$$S \text{ (partitioned)} = \begin{array}{c|ccccc} & L_1 & L_2 & L_3 & & L_r \\ \hline L_1 & N_{11} & 0 & 0 & \ldots & 0 \\ L_2 & N_{21} & N_{22} & 0 & & 0 \\ & \cdot & & & & \\ & \cdot & & & & \\ & \cdot & & & & \\ & \cdot & & & & \\ L_r & N_{r1} & N_{r2} & N_{r3} & \ldots & N_{rr} \end{array} \quad (5)$$

The diagonal submatrices N_{11}, N_{22}, etc. specify the reachability among the elements of levels $L_1, L_2, \ldots$ respectively. These diagonal submatrices become the identity matrices when there are no cycles. Submatrix N_{21} contains information concerning reachability from elements in level L_2 to elements in level L_1. Similar relationships are applicable to N_{31} and N_{32}.

This process operates to rearrange and partition the original element set into hierarchical components, $S_H = [s_4; s_1; s_2; s_3]$. The hierarchically reordered directed graph (digraph), D_H, then becomes:

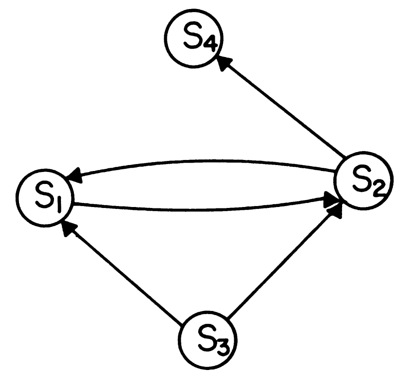

Fig. 11

References

HARARY, F., NORMAN, R. and CARTWRIGHT, D. (1975). *Structural Models: An Introduction to the Theory of Directed Graphs*, John Wiley, New York.

KAHAT, E. and SACHAM, M. (1973). Chemical Process Simulation Programs, *Process Technology International*, Vol. 18 No. 3, pp. 35-49.

LANGEFORS, B. (1973). *Theoretical Analysis of Information Systems*, Auerbach Publishers, Princeton, NJ.

ORDEN, A. (1971). Structuring of Organizational Information Systems, *IFIP Proceedings*, 1971.

SACKMAN, H. (1970). *A Man-Computer Problem Solving*, Auerbach Publishers, Princeton, NJ.

SAGE, A. P. (1977). *Methodology for Large Scale Systems*, McGraw-Hill Book Co.

SARGENT, R. W. H. and WESTERBERG, A. W. (1964). Speed-up in Chemical Engineering Design, *Trans. Inst. of Chemical Engineers*, Vol. 42, pp. 89-101.

STEWARD, D. V. (1965). Partitioning and Tearing Systems of Equations, *SIAM Journal on Numerical Analysis*, Vol. 2 No. 2.

TARJAN, J. (1972). Depth First Search and Linear Graph Algorithms, *SIAM Journal of Computing*, Vol. 1 No. 2, pp. 146-160.

WARFIELD, J. N. (1974). *Structuring Complex Systems*, Battelle Monograph No. 4, April 1974, Battelle Memorial Institute, Columbus, Ohio.

Book review

Artificial Intelligence, Volume 1 by P. H. Winston and R. H. Brown, 1979; 492 pages. (*MIT Press*, £16·25)

This is a collection of 14 articles describing recent work at MIT. Most articles are abridged; a few originated as published papers (1977 or later), the rest as internal reports or overviews of Ph.D. theses. They are grouped into three sections: 'expert problem solving', 'natural language understanding' and 'representation and learning'. (A second volume will have sections on vision, mechanical manipulation, and AI programming concepts.) For each section the editors have provided a few pages of background material.

One associates the MIT approach with the following tenets: 'working programs rather than theories' and 'programs for particular tasks rather than general problem solvers'. As one would expect from the first tenet, most of these articles present working systems—one significant exception is Minsky's reflections on communication between interconnected 'agents'. The second tenet is only an expression of dissent from the old 'general problem solver' paradigm, and in fact only three systems here concern specific application domains: respectively classical mechanics, the Stock Exchange and electrical circuits—the last of which evokes from me the complaint that the article is overbalanced by the electrical stuff. Certain other complaints I have about the book are reactions to all AI literature, although especially applicable perhaps to MIT work (which is not to take issue with their view that work on the pragmatic and particular should precede consideration of the abstract and general).

The first complaint is that for me there is sometimes too much on the behaviour of a system and too little on its internal structure. Given limited space I would rather get to understand completely a small part of a program's behaviour. The editors take the opposite view: they have accentuated matters by abridging in favour of the 'what' rather than the 'how'.

Secondly, comparability. Each paper is a new beginning for the reader: there is no common framework for answering such questions as 'is the problem addressed by paper A a subproblem of that addressed by paper B?' or 'which are the shared and which the incompatible assumptions made by the papers A and B?'. Examples in this book: the AMORD system and Doyle's 'truth maintenance' ought surely to relate; similarly the two papers on the 'Programmer's apprentice/coach' theme.

Lastly, comprehensibility. The worst in AI writing is when it is both informal and obscure. If anything these articles are better than average in this respect, but there are blackspots.

This is not a beginner's book. Furthermore, even with abridgements and background notes, it doesn't constitute a straightthrough read. As a work of reference its value is slightly reduced by the fact that some of the articles appear in full in easily accessible places (*IJCAI*, *AI Journal*). On the other hand, it does provide a broad up-to-date sample of work at MIT, the (too brief?) background notes are helpful and most of the articles are vivid and inviting.

C. T. BURTON (London)