A TWO-DIMENSIONAL TRIM-LOSS PROBLEM WITH SEQUENCING CONSTRAINTS

A, I. Hinxman
University of Liverpool
Liverpool, England

Abstract

A two-dimensional trim-loss problem is considered in which the cutting is two-stage but there are constraints on the sequencing of the cutting of orders, A method is developed in which problem reduction is used to generate a set of possible cutting patterns at each step and an heuristic choice made of a pattern from that set. The sequences of patterns produced, although slightly sub-optimal as regards trim-loss, fully satisfy the sequencing constraints.

1. Introduction

The trim-loss, or cutting stock, problem arises whenever material manufactured continuously or in large pieces has to be cut into pieces of sizes ordered by customers. The basic problem is to so organize the cutting as to minimize the amount of waste (trim-loss) resulting from the cutting.

In the one-dimensional case order lengths of some material such as steel bars have to be cut from the stock lengths held by the supplier. Gilmore and Gomory (1961, 1963) have formulated a linear programming solution of this problem, which also arises in the slitting of steel rolls, cutting of metal pipe, and slitting of cellophane rolis.

In the two-dimensional case the stock is held as large rectangular sheets from which smaller rectangles must be cut. Work has been done on this problem by Gilmore and Gomory (1965), who use a linear programming method, Adamowicz and Albano (1976), whose method is one of dynamic programming, and Hinxman (1976), using a problem reduction method. Hahn (1968) considers the problem that arises when the stock sheets contain flaws, and gives a dynamic programming algorithm.

Dyson and Gregory (1974) consider a problem arising in the flat glass industry in which the problem is not simply to determine a set of cutting instructions (known as cutting patterns) which minimize trim-loss, but to find a set which whilst leading to low trim-loss are also an acceptable sequence according to other operational criteria. Their method is to produce a minimum loss set of cutting patterns and then to find an acceptable way of sequencing them. This report describes an alternative approach to a similar problem, in which the sequencing and trim-loss criteria are considered together,

2. Statement of the problem

The material under consideration is glass. Each order received from a customer consists of a demand for specified numbers (demands) of pieces of each of one or more shapes, a shape being specified by its length (longer dimension) and breadth (shorter dimension). Each order is designated by a code. If the order is for glass with high quality edges the first character of the code is "F". We refer to these as type F orders and to the remainder as type M orders.

The cutting of the stock sheets is a two stage process, and corresponding to each stage of the cutting there is a stripping process, to produce edges of the required quality, that may be manual or automatic. During the first stage of the cutting the sheet is moving in the direction of its longitudinal axis and may be considered to have a "leading" and a "trailing" edge with respect to this motion. The first set of cuts are perpendicular to the direction of motion and are cross cuts. The sub-sheets resulting from the first stage of cutting will be cut into pieces according to the corresponding sub-patterns.

Operational constraints on the design of cutting patterns due to the design of the cutting machinery impose:

i) an upper bound on the number of shapes in a sub-pattern,
ii) an upper bound on the number of sub-patterns in a pattern,
iii) a lower bound on the distance of the first cross cut from the leading edge of the sheet,
iv) a lower bound on the distance of the last cross cut from the trailing edge of the sheet.

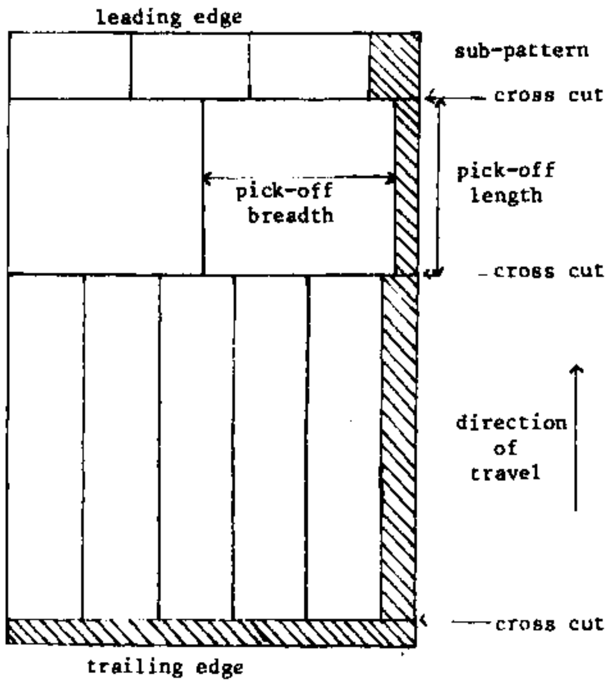
The pick-off length of a sub-sheet or piece is its dimension in the direction of the length of the sheet from which it was cut, its other dimension being its pick-off breadth (see figure 1). The stripping following each stage of cutting may be manual or automatic. This affects the reduction in dimensions of the initially cut pieces by the stripping process.

The pieces resulting from the cutting and stripping process are picked off for despatch from one of four legs, a leg being a conveyor belt with its associated washing and handling facilities. All the pieces resulting from the cutting of a sub-pattern must be routed to the same leg. The routing of pieces to legs must be consistent with the dimension constraints on the legs and also with the requirement that type F pieces must be picked off from legs with washing facilities. There is a further restriction that pieces for not more than three different orders may be picked off any leg during the cutting of any one pattern.

The restrictions on the design of individual cutting patterns having been described it is now necessary to consider how a set of cutting patterns is used to provide instructions for cutting all the orders in an order list.

Initially a cutting pattern is selected.

Figure 1
Typical cutting pattern



It will specify the size of stock sheet to be used and the order pieces to be cut from this sheet. For each shape, s, in the pattern, there is a number, n , such that if the pattern is repeated n times the demand for that shape will be satisfied, whilst if the pattern is only repeated n -1 times this will not be the case. The number of stock sheets cut to this pattern will be the smallest of the n_s 's,
s

When this has been done the original order list will be converted to a new one reflecting the demands that have not yet been satisfied. A new cutting pattern is selected and the process repeated until all the demands have been satisfied.

Operational costs connected with the picking off and packing of orders will affect the choice of cutting patterns and the sequence in which they are used. Suppose the use of a cutting pattern has just been completed. The next cutting pattern to be used should include all the shapes in the last cutting pattern the demands for which have not yet been satisfied. In addition, if there were any shapes, the demands for which were satisfied at the completion of use of the last cutting pattern, belonging to orders which include other shapes, the demands for which have not been satisfied, then such a shape from each such order should be included in the new cutting pattern. For each failure to make the appropriate inclusion the <u>sequence break cost</u> is incurred.

There is also an absolute constraint on the sequence of cutting patterns. If a shape belongs to a loose load order, which will have a code starting with the characters "FD", then it must not be cut until demands for all larger shapes in that order have been satisfied.

Table 1 shows the dimensions of the available stock sheets.

Table 1

Sizes of stock sheets

Length mm	Breadth mm
4648	2540
4699	2616
4648	3048

3. <u>The top-level state-space</u>

The process of choosing and utilizing cutting patterns described above can be represented in terms of a state-space (Nilsson, 1971; chaps. 2, 3). The states are lists of unsatisfied orders, and the operators are cutting patterns together with the number of times they are used. The start state is the original order list and the goal state the empty order list. We require to find the path between the two for which the combined cost of trim-loss and sequence breaks is minimal.

In order that the resources required by a program to compute a sequence of cutting patterns may be kept within reasonable bounds, backtracking must be restricted. A certain amount of time is allocated for the generation of possible first patterns. At the end of this time the "best" of the generated patterns is selected. The number of times it is to be used is calculated, the demands in the order list are adjusted to take account of the pieces cut by the use of this pattern, and the adjusted order list together with the first pattern defines the starting point for the generation of possible second patterns. The process is repeated until all the demands in the order list have been satisfied.

This method of working corresponds to the development of a path across the state-space graph without backtracking. The generation of a subset of the possible cutting patterns at each step (done in the work presented here by a problem reduction technique) corresponds to partial development of the corresponding node, and the selection of the "best" pattern represents the application of an evaluation function.

The evaluation function used in the present case combines a measure of the possibly avoidable trim-loss (it can be determined that the cutting of some of the pieces included in the pattern will unavoidably involve trim-loss) in the pattern, the "urgency" (see section 6.2) of the pieces included in it, the cost of sequence breaks that would occur at the present point in the sequence if it were used, and the possible costs of sequence breaks that might occur later in the sequence if it were introduced at this point. The evaluation function is not, in Nilsson's terms "admissible", but has been chosen on the basis of experience as one whose use should not result in seriously sub-optimal solutions.

The quality of solutions generated will be limited by the correctness of the partial development (whether the operator corresponding to the arc belonging to the optimal path from this node is generated) and the accuracy of the evaluation function (whether when generated the operator is correctly identified).

More complicated search strategies than the one adopted could have been followed. A wider ranging search of the top-level state-space might have been performed. This would have required the use of more computer time. Alternatively the problem reduction at a node might have used information about the path on which the node lay, rather than simply information local to the node, to decide on reductions. This would require a great deal more information to be simultaneously present in the store of the machine. The method chosen was one in which it seemed that low computational costs would be paid for the finding of acceptably good solutions.

4. Problem reduction

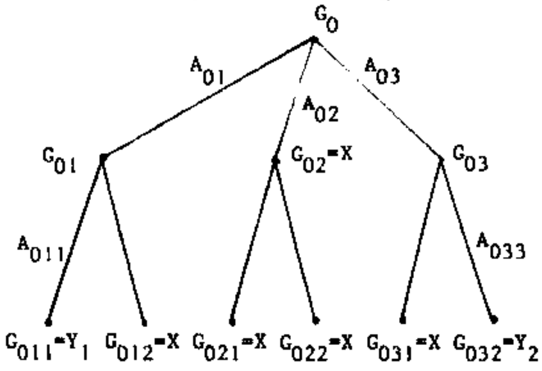
When a problem reduction method is applied to a complex problem, this problem, or goal, is reduced by successive refinement to a number of simpler problems, subgoals, whose solutions, when taken together, can be combined to provide a solution to the original goal. Usually there will be more than one way in which the goal, or a subgoal, can be broken down into subgoals. It is, of course, only necessary to solve the subgoals corresponding to one breakdown in order to solve the goal or subgoal from which they were generated.

A good general description of problem reduction methods is given by Nilsson (1971; chaps.4, 5). Here will be described only the structure of the method used in the present work. Let G_0 be the goal. Then there is a set of actions $(A_{01}, A_{02}, \ldots\}$ which may be taken towards achieving G_0. After action A_{on} has been taken G_o will have been reduced to subgoal G_{On}, A_{On} followed by the actions necessary to achieve G_{on} will achieve G_o. G_{on} may be indivisible. In this case either' it can be achieved by some basic action, in which case a solution to G_o has been found,or no such appropriate basic action exists, in which case there is no solution of G_o including A_{0n}. If G_{on} is not indivisible then it can be reduced to some action A_{Onto} and a further subgoal GQ , this process being repeated until an indivisible subgoal is defined. If no solution to any subgoal of a divisible subgoal can be found then that higher subgoal is itself insoluble.

This process can be represented graphically (see figure 2). Let GQ be the root of a tree* Then A_{01}, A_{02}, ••• are represented as directed arcs connecting G_O with its successor nodes G_{01}., G_{02} , . . a similar representation applying to the connection of these with their successors. A terminal node represents an indivisible subgoal. If it is soluble, its solution action can be associated with it. This together with the labels of the arcs connecting it with G_0 defines a solution of G_0. If it is not soluble then an "insoluble" marker can be attached to it. Similarly an insoluble marker can be attached to a node all of whose successors are insoluble.

Figure 2

Simple tree of subgoals



■ X denotes insoluble

= Y_n denotes solution of indivisible subgoal

Thus G_0 is solved by

A_{01}, A_{011} and Y_1

or A_{03}, A_{032} and Y_2,.

Note that G_{02} is deduced to be insoluble by virtue of the insolubility of its successors.

This form of problem reduction, in which all the nodes of the problem reduction tree are OR nodes, can be regarded as a state-space method, with start node an empty list of actions to be taken to achieve the goal and goal nodes complete lists of such actions. However, two features of the present work make it more desirable to consider it as a problem reduction method. Firstly, when a successor node of the tree is generated it contains a more complete list of actions than its parent; the subgoal it specifies is less complex and the problem is indeed "reduced". Secondly, whereas in state-space methods the available operators usually have some similarity to each other, here there are very distinct classes of action that may be taken, depending on what part of the list of actions remains to be completed.

5. Structure of the problem reduction

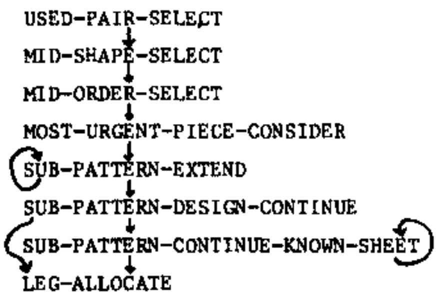
Each node of the problem reduction tree includes in its label a subgoal description and the name of a function that is capable of generating a sequence of alternatives for the reduction of that subgoal. For a primitive subgoal such a reduction is a solution. Let us refer to the occurrence of a function at a node as an instance of that function.

In an iteration of the problem reduction process, a node of the problem reduction tree is selected. The function at that node is invoked to give the reduction for the subgoal that is next in the sequence after those reductions that were generated by previous invocations of that function at that node. It will either create a new node in whose label are included the subgoal resulting from the reduction and the name of the function to be used for reducing this subgoal, or report that the sequence of reductions has been exhausted. Iterations of the process continue until what is regarded as an adequate number of alternative solutions of the goal have been generated.

It has been said that when a function creates a node it places a function name in its label. Figure 3 shows the relationships of the subgoal processing functions in this respect. An arrow directed from one function name to another indicates that the label of the node created by the first may include the name of the second. Where more than one arrow leaves the name of a function, this indicates that a choice of functions is made according to the features of the subgoal set up.

Figure 3

Hierarchy of subgoal processing functions


USED-PAIR-SELECT
MID-SHAPE-SELECT
MID-ORDER-SELECT
MOST-URGENT-PIECE-CONSIDER
SUB-PATTERN-EXTEND
SUB-PATTERN-DESIGN-CONTINUE
SUB-PATTERN-CONTINUE-KNOWN-SHEET
LEG-ALLOCATE

Consider the situation that can occur for the generation of any cutting pattern other than the first. The previous cutting pattern may have included one or more sub-patterns that each included two shapes (pairs). If the demand for a shape belonging to a pair has not yet been satisfied, then there is a choice to be made as to whether it should belong to a pair with the same or a related partner in the proposed new pattern. The choices on the retention of pairings are made in USED-PAIR-SELECT.

There may be a number of shapes included in the previous pattern the demand for which has not yet been satisfied. Some of these may have been included in the proposed pattern by choices made in USED-PAIR-SELECT. The choices as to which of the remainder should be included, and what their orientations should be, in the proposed pattern, are made in MID-SHAPE-SELECT.

There may be one or more shapes completed at the end of the use of the previous cutting pattern that belong to orders containing shapes the demands for which have not yet been satisfied. In MID-ORDER-SELECT the choices are made as to which shapes, if any, from such orders should be included in the proposed pattern, and what their orientations should be.

The choices made in MID-SHAPE-SELECT, MID-ORDER-SELECT and MOST-URGENT-PIECE-CONSIDER, are that shapes shall be included in a pattern in separate sub-patterns. Each instance of SUB-PATTERN-EXTEND is concerned with one such sub-pattern and chooses whether it should include one. or two shapes. If the choice is for two shapes the choice of the second shape is made.

The choice of stock sheet size to be used by the proposed pattern is made in STTB-PATTERN-DESIGN-CONTINUE. A further sub-pattern may also be included in the pattern by this function.

Each instance of SUB-PATTERN-CONTINUE-KNOWN-SHEET chooses whether a further sub-pattern should be included in a partially completed pattern, and if so, what this sub-pattern should be.

In LEG-ALLOCATE an allocation of sub-patterns to legs is made if a possible one exists. If it does not, then the proposed pattern is ruled out of further consideration, otherwise an evaluation is made of the "merit" of the proposed pattern. The allocation of sub-patterns to legs is a primitive subgoal.

6. Heuristics

Space does not permit the description in detail of all the heuristics used. Some of the more significant ones will be outlined here.

6.1 Orientations

For each shape a preferred orientation can be defined. For a shape which the constraints of the problem require to be cut with a certain orientation it is this orientation. For any other shape, all possible sub-patterns that could be formed including this shape are considered. The orientation in that sub-pattern within which there is least trim-loss is the preferred orientation. When the possibility of including a shape in a pattern is considered, it will usually be with this orientation. The principle applied here is that if the trim-loss in each sub-pattern is minimized the overall trim-loss is likely to be minimized.

6.2 Urgency

The urgency of a shape reflects the desirability that the shape should feature in the early cutting patterns of a solution sequence and is a function of its dimensions and demand. Two cases are identified in which particularly high urgencies are assigned.

The highest urgencies are given to shapes with lengths above a certain limit, since it will be difficult to introduce these later in the sequence when sequencing desiderata lead to inclusion in a pattern of several other shapes.

The next most urgent set of shapes are those which can only occur in leading sub-patterns. If such shapes were left until late in the pattern sequence, this might be forced to finish with patterns consisting only of leading sub-patterns, with consequent extreme trim-loss.

Adjustments to urgencies are made in the light of sequencing considerations. For example, the largest shape in a loose load order will have its urgency increased to exceed the highest urgency, as determined by the initial calculations, of any shape in the order. This takes account of the fact that the smaller cannot be cut before the larger and therefore its own higher urgency would still not lead it to be introduced into an early pattern.

6.3 Forcing shapes in

The four subgoal processing functions at the top of the hierarchy are all concerned with giving direct consideration to the possible inclusion in the current pattern of shapes that are deemed at this point to be particularly desirable.

Experience shows that once shapes from two different orders have appeared in the same sub-pattern they should continue to do so. USED-PAJR-SELECT introduces such pairs into the proposed pattern.

Sequencing desiderata mean that other shapes for which demand has not been satisfied should be introduced into the proposed pattern. MID-SHAPE-SELECT will introduce these shapes, and its first choice will be to give them their preferred orientations, provided a look-ahead indicates that this will not cause difficulties in MID-ORDER-SELECT.

It is worth giving attention to several alternative developments of the partial proposals generated by these first two functions, so the search is organized in such a way that they are re-entered relatively less frequently.

Choices made in MID-ORDER-SELECT complete consideration of the inclusions desirable for reasons of sequencing. The reasoning that led to the definition of urgency suggests that the most urgent shape remaining unsatisfied in the order list should be specifically considered for inclusion. , This is done by M0ST-URGENT-PIECE-CONSIDER.

6.4 SUB-PATTERN-DESIGN-CONTINUE

When SUB-PATTERN-DESIGN-CONTINUE considers which size of stock sheet should be used, it does so in the light of the specifications that have already been made that sub-patterns containing certain shapes shall be present. For each stock sheet size the manner in which each sub-pattern can be constructed so as to include the stated shapes with minimum trim-loss is determined. The sheets are ranked on the aggregates of these minimum trim-losses, the lowest aggregate indicating the most preferred size. It is likely that a partial cutting pattern for which the trim-loss already unavoidable is low can be completed in such a way that the total trim-loss is low.

Within a choice of stock sheet size a number of possible alternative additions of sub-patterns to the proposal may be generated. The evaluation of these alternatives takes account of the urgency of the piece it has been decided to include in the sub-pattern, the trim-loss within the sub-pattern and, by means of a look-ahead, the likely amount of trim-loss in the pattern as a whole outside of the trim-loss in the individual sub-patterns. The evaluation also gives some preference to additions which keep the number of different shapes in the final pattern small as these will lead to fewer complications in the pattern sequencing.

7. Results

A program using these techniques has been written in Algol 68-R (Woodward and Bond, 1972) and tested on an ICL1906S against 16 sets of data. A typical data set is shown in table 2. The results from these test runs are shown in table 3.

During the development of the program a considerable number of heuristics were, experimented with. These involved different methods of problem reduction, different orderings for the sequences of problem reductions, resulting in different subsets of the possible reductions being used, different ways of calculating the "urgency" of pieces, and different methods of evaluating the "best" pattern given a set of alternatives. Some of these heuristics produced very good results with some sets of data whilst performing unacceptably badly on others. These results have been collated to produce column 4 of table 3. Column 5 thus shows the known deficiencies of the program.

The execution time of the program depends on the size of the data set, the format of the output and whether the object code incorporates run-time checking. However, 4 minutes may reasonably be regarded as an average execution time. No sequence breaks occur in any of the pattern sequences.

8. Conclusions

The program succeeds in producing sequences of cutting patterns with no sequence breaks, which was one of the design aims. In this respect it would appear to be superior to that of Dyson and Gregory (1976). Its trim-loss behaviour is known not to be optimal. However, on the test data used the percentage of the material lost as scrap never rose to 10% and the average discrepancy between the solutions produced and the known best solutions was 1% of the material used. Whether the benefits of absence of sequence breaks outweigh the sub-optimability of the trim-loss behaviour would depend on the details of the industrial environment.

Heuristic information is embodied in the program in the urgencies, the evaluation of

Table 2

Data Set 7

Code	Length	Width	Demand
FA9113	2660	1200	225
PS1224	838	635	40
PS1224	1046	711	38
ME2438	1600	840	70
FD6458	2540	1500	25
FD6458	2540	1600	25
FD6458	2540	1700	25
FD6458	2540	1800	175
FN4008	2438	610	184
FN4008	2438	762	150
FN4008	2438	914	116
FN4008	2438	1219	95
N3281	1830	610	33
N3281	1830	762	27
N3281	1830	915	23
N3281	1830	1220	37
FN3922	1829	1219	85
FN3584	2490	1500	25
FN3584	3048	1500	20
MF3473	960	720	45
MF3473	1520	920	100
MH4200	1120	680	39
FN3905	3048	1829	16
N3276	1220	915	28
N3252	1220	610	20
MF2694	1360	560	565
FD9671	2540	1400	140
FD8854	2440	1830	250
FC7792	760	600	140
FC7792	1200	900	86
FN4045	3048	2000	90
FN4035	1524	914	241
STOCK	2490	1800	112
FD9288	2540	1800	300
FN4002	1829	1219	250
FA9143	2240	1000	30
FA9143	2240	1200	30
FA9143	2240	1600	30
FA9143	2240	1800	30
MH0886	1340	920	60
MH0875	740	406	344
MH0875	1440	580	1012

patterns, the chosen problem reduction structure and the extent to which the problem reduction tree is developed. The greatest gains in performance would be likely to result from the improvement of the first two of these.

Acknowledgements

This research was supported by SRC research grant B/RG/83176. The grant holder is Dr. T. B. Boffey.

Table 3

Results of test runs

1	2	3	4	5
Data set	Actual trim-loss (sq.cm $\times 10^7$)	Trim-loss as % of material used	Best known trim-loss (% age)	Δ (difference between 3 & 4)
1	1.4	8.2	5.6	2.6
2	1.7	8.9	7.7	1.2
3	0.7	7.7	7.2	0.5
4	1.2	7.1	6.6	0.5
5	1.6	8.9	6.4	2.5
6	1.4	6.5	6.0	0.5
7	1.1	7.9	7.1	0.8
8	0.8	7.9	6.3	1.6
9	2.0	9.7	7.2	2.5
A	1.2	8.0	7.6	0.4
B	1.5	9.2	8.4	0.8
C	0.9	8.2	6.9	1.3
D	0.8	7.5	7.2	0.3
E	0.8	5.5	5.5	0.0
F	1.6	9.8	9.6	0.2
G	1.3	6.5	6.5	0.0
	Mean	8.0		1.0

References

Adamowicz, M. and Albano, A., A Solution of the Rectangular Cutting-Stock Problem, IEEE Trans. Systems, Man,Cybernetics SMC-6,302-310 (1976).

Dyson, R.C. and Gregory, A.S., The Cutting Stock Problem in the Flat Glass Industry, Opl, Res. Q.25, pp.41-53 (1974).

Gilmore, P.C. and Gomory, R.E., A Linear Programming Approach to the Cutting Stock Problem, Opns. Res. 9, pp.849-859 (1961).

Gilmore, P.C. and Gomory, R.E., A Linear Programming Approach to the Cutting Stock Problem - Part II, Opns. Res. 11, pp.863-888 (1963).

Gilmore, P.C. and Gomory, R.E., Multistage Cutting Stock Problems of Two and More Dimensions, Opns. Res. 13, pp.94-120 (1965).

Hahn, S.G., On the Optimal Cutting of Defective Sheets, Opns. Res. 16, pp.1100-1113 (1968).

Hinxman, A.I., Problem Reduction and the Two-Dimensional Trim-Loss Problem, AISB Summer Conference on Artificial Intelligence and Simulation of Behaviour, pp.158-165, University of Edinburgh, 1976.

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

Woodward, P.M. and Bond, S.G., Algol 68-R Users Guide, H.M.S.0. London, 1972.