

Bedrijfseconomische Verhandeling Nr. 8101

PRODEMO : PROcedural DEcision MOdeling
through the use of decision tables

by

R. MAES and J. VAN THIENEN

Version 2, february 1981

Wettelijk Depot : D/1981/2376/3

"Prodemo", abbreviation of "PROcedural DEcision MOdeling system", is also Greek for "for the people". The authors are convinced that the PRODEMO system is a useful tool for a wide class of users.

CONTENTS

1. Introduction
2. Intended audience
3. Procedural decision modeling and the use of decision tables
4. A rationale for computer introduction
5. The PRODEMO system
6. Directions for future extensions
7. A short note on the choice of the Plato environment

Acknowledgements

1. INTRODUCTION

Traditionally, management sciences paid much more attention to supplying accurate and up-to-date information rather than to increasing the quality of the decision process itself. Undoubtedly, this tendency was prompted by the more quantitative nature of the information delivering process and hence by the potential introduction of the computer.

In this report, we want to emphasize that the decision table technique, originating from the information sciences, can add to the quality of everyday decision making by providing facilities for designing and making procedural decisions in a structured and disciplined way.

However, standardized techniques for translating complex decision descriptions into decision tables should be used. It is demonstrated that the interactive PRODEMO system greatly contributes to this achievement and that every decision maker can benefit, without prior knowledge, from its application.

2. INTENDED AUDIENCE

The application field of decision tables in general and of PRODEMO in particular can be divided into two parts :

- a. the design of new procedures, regulations, laws, ...
- b. the analysis of existing procedures in order to examine their correctness and their completeness.

In particular, the PRODEMO system may prove to be valuable a.o. for the following categories of people :

- Managers who want to design new procedures or to analyse and correct existing ones used in their organizations (or even by themselves).
- Lawyers who are confronted with logically chaotic and even incomplete and/or contradictory laws.

- Legislators who have to design new laws adapted to complex modern life.
- Anyone involved in the design of new regulations and prescriptions in general.
- Information analysts who have to grasp complex organizational and infological problems.
- Teachers looking for a clear and unambiguous tool for representing complex material.

3. PROCEDURAL DECISION MODELING AND THE USE OF DECISION TABLES

The wide class of decision problems indicated in chapter 2 can advantageously be dealt with using decision tables. In this chapter, we first outline some basic definitions of the decision table technique, after which we present a straightforward procedure for obtaining decision tables. Finally, some advantages of the technique are mentioned.

3.1. Definitions

A decision table is a tabular representation of a decision situation showing the actions to be taken depending on the combined values all the relevant conditions can have.

The general structure of a decision table is shown in figure 3.1. Each decision table is identified by a name; further, it is split up into a condition part and an action part. The left part of the table is called the stub, while in the right part the entries, forming the individual decision rules, are filled in.

TABLE NAME	
CONDITION STUB	CONDITION ENTRIES
ACTION STUB	ACTION ENTRIES

Fig. 3.1.

Fig. 3.2. represents a decision table. Looking at the first rule (the first column to the right of the double bar), we immediately see that for a customer requesting a First Class seat and in the case of such a seat being available, a 1st Class ticket will be issued and the appropriate reservation balance will be updated (indicated by X). A don't care condition entry (-) means that this condition is irrelevant for the decision rule under consideration.

It is obvious that any decision rule having one or more don't care entries (a so-called contracted decision rule) can be translated into a set of (simple) decision rules having the same configuration of actions (see figure 3.3 for the first rule of the Airline Reservation table). Obviously the opposite transformation can always be performed; we call this process the "contraction" of a set of decision rules. It is automatically performed by the PRODEMO system on simple request of the user.

Airline Reservation						
Customer requests	First Class			Economy Class		
1st Class seat available ?	Y	N		-		
Economy Class seat available ?	-	Y	N	Y	N	
Is Economy Class also O.K. ?	-	Y	N	-	-	-
Issue 1st Class ticket	X	-	-	-	-	-
Update 1st Class reservation balance	X	-	-	-	-	-
Issue Economy Class ticket	-	X	-	-	X	-
Update Econ. Class reservation balance	-	X	-	-	X	-
Put customer on 1st Class waiting list	-	-	X	X	-	-
Put customer on Econ. Class waiting list	-	-	-	-	-	X

Fig. 3.2.

First Class	First Class			
Y	Y			
-	Y			
-	Y		N	
	Y	N	Y	N
X	X	X	X	X
X	X	X	X	X
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Fig. 3.3.

3.2. Constructing decision tables

In chapter 2, a basic distinction between a priori structured and a priori unstructured situations was made. Analyzing the former class starts from a (mostly narrative) description. A rather straightforward procedure for transforming these descriptions in their decision table equivalent was developed by M. VERHELST and is briefly given in this section. On the other hand, dealing with a priori unstructured situations, where no initial description exists, is a much more cumbersome task. In this case, resorting to the PRODEMO system is an even more advisable way of proceeding.

Suppose we want to construct a decision table representing the following decision problem :

Filling Orders

When the quantity ordered for a particular item equals or exceeds the minimum discount quantity and the order is from a wholesaler, give the customer a discount and make the shipment. This presumes that there is sufficient quantity on hand to fill the order.

If the quantity ordered is less than the discount quantity, bill at regular rates even if the customer is a wholesaler. Bill at the same rate if the sale is retail.

If there is not a sufficient quantity on hand, bill as above, ship what can be shipped and backorder the remainder of the order. It must be emphasized that, in this situation also, even if the discount quantity is ordered, if the customer is a retailer, the discount is not given.

Carefully analyzing this text results in the following lists of conditions (with their values) and actions :

CONDITIONS	VALUES
1. Quantity ordered \geq minimum discount quantity	Y / N
2. Ordered by	wholesaler / retailer
3. Sufficient quantity on hand to fill order	Y / N

ACTIONS
1. Give discount
2. Bill at regular rates
3. Make shipment of quantity ordered
4. Make shipment of quantity on hand
5. Backorder remainder

The following remarks hold true :

1. Repetitions of the same action or condition name are already omitted from the above lists.
2. The different values a condition can have should
 - be mutually exclusive (e.g. no customer can be wholesaler and retailer at the same time)
 - span the whole range of possible values for this condition (exhaustiveness constraint).

Now, one has to reconsider the starting text in order to establish the relations between conditions and actions, called the decision rules. The following procedure, also applicable when using PRODEMO, can be applied.

Consider the first paragraph of the text. It is said that the customer should be given a discount (i.e. Action 1 of the above list should be executed) and the shipment of the ordered quantity should be made (Action 3) if

- the quantity ordered equals or exceeds the minimum discount quantity (condition 1 takes its first value)
- and
- the order is from a wholesaler (condition 2 takes its first value)
- and
- there is sufficient quantity on hand (condition 3 also takes its first value).

We transcribe this statement using the following conventions :

- the actions, separated by AND's, are put ahead of the equivalent logical expression
- references to conditions are put behind the implication sign (\leftarrow)
- both conditions and actions are referred to using their number in the condition (action) list
- condition values are referred to using letters ('a' for the first value, 'b' for the second one, ...).

This gives :

(1) 1 AND 3 \leftarrow 1a AND 2a AND 3a

In the same way, the following remaining condition/action relations can be derived from the text :

(2) 2 \leftarrow 1b OR 2b

(3) 1 AND 4 AND 5 \leftarrow 3b AND 1a AND 2a

(4) 2 AND 4 AND 5 \leftarrow 3b AND (1b OR 2b)

(5) 2 \leftarrow 3b AND 2b

Notice the following :

- (2) is derived from the second paragraph
- in (3) and (4), both derived from paragraph 3, a certain duplication of (1) resp. (2) is inserted (because of the sentence "bill as above"). One should maintain this duplication in the expressions.
- (as a short exercise : try to discover in the table of figure 3.4 what happens in the second rule if you wrongly replace (3) and (4) by "4 AND 5 \leftarrow 3b")
- (5) represents the last sentence of the starting text where "in this situation" refers to the case of insufficient quantity on hand. It can be seen that it contains no additional information.

At this place, we want to stress that the logical expressions should be a literal translation of the initial text and nothing more ; any further interpretation must be done starting from the resulting decision table. Inserting the logical relations (1) - (5) into the expanded decision table (i.e. the decision table containing no contracted rules), as is done automatically by the PRODEMO system, produces the decision table given in figure 3.4.

FILLING ORDERS								
Quantity ordered (QO) \geq minimum discount quantity ?	Y				N			
	wholesaler		retailer		wholesaler		retailer	
Ordered by								
Sufficient quantity on hand (QOH) to fill order ?	Y	N	Y	N	Y	N	Y	N
Give discount	X	X	-	-	-	-	-	-
Bill at regular rates	-	-	X	X	X	X	X	X
Make shipment of QO	X	-	-	-	-	-	-	-
Make shipment of QOH	-	X	-	X	-	X	-	X
Backorder remainder	-	X	-	X	-	X	-	X
rules	1	2	3	4	5	6	7	8

Fig. 3.4.

Careful looking at this table reveals that

- in a certain number of cases (rules 3,5 and 7) no provisions are made for shipments
- the second condition is irrelevant for the case " $QO < \text{minimum discount quantity}$ " such that rules 5 - 8 can be combined into two rules (this is automatically done by the PRODEMO system).

Now, any correction (e.g. inserting additional provisions for preparing shipments) and/or refinement can be introduced in the decision table of figure 3.4, leading to a more complete and more accurate description of the decision situation dealt with.

3.3. Advantages of using decision tables

When using decision tables, the following advantages are gained :

1. Easy and fast decision making

To make a decision it is only necessary to look for the decision rule which satisfies the desired values of the conditions. There is no need to analyze the text everytime a decision has to be made.

2. Correct decision making

A text can be ambiguous and subject to incorrect interpretations. A decision table however leaves no room for interpretation as it contains no such words as : normally, else, exceptionnally, the previous, in so far, only if, but ...

3. Short and standardized representation format

In most cases there is no need to read and write long, annoying and difficult texts. Decision tables are more compact and easier to understand.

4. Check for completeness

It is very difficult to guarantee that a text treats all possible situations (most texts in facto do not). In a decision table one can immediately check which combinations have not been foreseen (and cause no actions to be taken).

5. Check for contradictions and correctness

If some actions exclude each other (e.g. refuse order, execute order), one can check whether the table contains a decision rule which leads to both actions. In texts it is very difficult, if not impossible, to detect such contradictions.

4. A RATIONALE FOR COMPUTER INTRODUCTION

In chapter 3, a straightforward manual procedure for constructing decision tables was outlined. However, in a high number of cases, the complexity becomes overwhelming; then, introducing the computer into the construction process is the obvious means to enlarge the applicability of the decision table technique. The following reasons can be put forward :

1. Combining different construction methods is hardly possible without the assistance of the computer. Besides, an interactive computer program, like the one described in this report, can give valuable indications about the desirable method.
2. The automatic generation of condition entries guarantees the completeness by enumerating all possible combinations of condition ranges.
3. Lots of administrative and clerical work that inherently go with the use of the decision table technique can be taken over by the computer.
4. Some manipulations of the resulting decision table can very easily be automated.
Ex. : - the contraction of the decision table using various criteria
 - reordering the conditions and actions
5. Some other manipulations lend themselves admirably to the use of interactive problem solving techniques.
Ex. : splitting up a decision table.
6. The resulting decision tables can, at a further stage, form the basis of more or less automated decision making.

5. THE PRODEMO SYSTEM

The PRODEMO (PROcedural DECision MODELing) system is a computer program for the construction of decision tables and hence for the modeling of various procedural decision situations.

Its main purpose is to guide and support the user during the building process by giving suggestions and feedback, by checking for incompleteness and inconsistencies and by executing all of the administrative routine tasks and the cumbersome drawings.

No special knowledge is required in order to use the PRODEMO system. The interactive environment, in which it has been conceived, guarantees an extended and effective user support when this is required and is able to create a highly flexible and well controlled construction process. Moreover, the system is supported with sufficient documentation and comment to provide for the need for any additional information during this process.

In this chapter attention will be paid to the construction process and the PRODEMO methodology will be illustrated with an example.

5.1. General description

PRODEMO is an interactive system for the automated construction of decision tables. Its implementation on the CDC PLATO system enables a very fast and flexible construction process.

According to the nature of the man-machine interaction, the PRODEMO system is able to operate in one of these two modes :

- command mode : the user takes control of the program and determines his path through the modeling process. This is achieved by grouping all important functions on a central index page : the PRODEMO menu. From this page the user can choose which option he wants to take, execute it and then return to menu to choose another option.
- response mode: the computer controls the program and leads the user through the modeling process.
The construction of the decision table is accomplished on suggestions of the PRODEMO software, after deducing underlying rules and constraints.

The user is at any chosen time supplied with the necessary information and explanation for a successful result and is able to switch from one operating mode to the other.

The possible exchange between command mode and response mode guarantees an automated design which is flexible enough for a constructive man-machine interaction.

5.2. PRODEMO Methodology

In order to construct a decision table, PRODEMO needs (fig. 5.1.) :

- a table name
- a list of conditions and condition states
- a list of actions
- a list of decision rules.


The construction of all these elements is grouped around the MENU page (fig. 5.2.). From this page the decision description is gradually built up. The rest of this chapter is an illustration of the construction process and its various stages.

decision table name				
condition name 1				
condition name 2				
condition name 3				
condition name 4				
...				
		decision		
		columns		
action name 1				
action name 2				
action name 3				
action name 4				
...				

Fig. 5.1. : Decision table elements.

12.37.18.

prodemo - menu



Choose one of the following options :

(→ : recommended, + : possible, - : not yet available)

- + a. to start/restart decision modeling
- + b. to change PRODEMO default options
-
- + c. to add/change/delete conditions (or states)
- + d. to reorder conditions or states
-
- + e. to add/change/delete actions
- + f. to reorder actions
-
- + g. to add decision rules
- + h. to inspect/change/delete decision rules
-
- i. to construct decision table
- + j. to make decisions
-
- + r. to reload/delete decision from working storage
- + s. to save decision in working storage
-
- y. to access your own table library
- + z. to access public table library

-stop- to end or restart PRODEMO

-lab- for response mode

-shift help- for general information

-help- available

Fig. 5.2. : MENU of PRODEMO options

5.3. The construction process illustrated with an example

As an example of the construction process, consider the following decision situation, which might represent an existing order processing procedure :

1. When an order comes in, first a check is made whether the customer's credit limit is exceeded.
2. If it is not exceeded, the order is delivered or put on a waiting list, depending on the stock situation.
3. If the credit limit is exceeded, the foregoing holds if the customer is considered important. Else the order is rejected if the amount involved exceed \$ 100.
4. Orders are always delivered if the stock is sufficient.

Transforming this procedure into a decision table will not only improve the speed and the simplicity of decision making, but will also check for the correctness and the completeness of the existing regulation.

In order to construct a decision table for this problem, using the PRODEMO system, we successively choose the appropriate options in the menu (fig. 5.2.). The first thing to do is to start decision modeling.

5.3.1. Starting decision modeling

Starting the modeling process merely implies preparing the system to construct a decision table. This is indicated by giving a name to the decision being modeled. After the construction process, the name is used to store the completed decision table in the table library.

Suppose the decision in our example is called 'ORDER TREATMENT'.

5.3.2. Entering conditions and states

The conditions are the criteria on which the decision depends. A condition consists of a name and a number of condition values or states, which affect the decision.

The condition called 'Stock sufficient?' might have two possible states : 'Yes' and 'No'. There are of course also conditions with more than two states, like 'color' could be 'blue', 'green', 'red', 'black',

In order to find the conditions, it is necessary to read the initial text carefully. Because conditions are the factors affecting the decision, they often follow words like : if, unless, else, depending on, in so far,

Analysis of the example results in the following conditions :

1. Is the credit limit exceeded? Yes or no
2. Is the stock sufficient ? Yes or no
3. Are we dealing with an important customer ? Yes or no
4. Amount involved ? ≤ 100 or > 100 .

Entering these conditions and condition states can be done in a really straightforward way. Up to 10 conditions may be supplied each one having a maximum of 6 states. At any time conditions and states can be changed, added or deleted.

Suppose we are satisfied with the conditions as they are listed in figure 5.3., so we can move on to a further stage in the modeling process.

5.3.3. Entering actions

The actions are the decisions which have to be taken, the consequences which result from a certain combination of condition states. In the decision text, they are indicated in combination with words like : then, in that case,...

In order to obtain the actions, we read the text and see what the outcome of the decision might be. We find three possible outcomes: (see figure 5.4.).

1. Execute the order
2. Refuse the order
3. Put the order on a waiting list.

These actions can simply be fed into the computer. Up to 20 actions may be supplied. At any time, actions can be changed, added or deleted when this seems necessary.

One of the purposes of constructing a decision table is to look for inconsistencies. In order to enable the system to detect such errors or shortcomings in the decision description, it is possible to indicate which actions are contradictory, i.e. only one of them should be referenced in the same decision situation. In the above example, all three actions are contradictory. One cannot e.g. execute and refuse an order at the same time. These contradictions can easily be entered.

Note that the input of actions and conditions is not to be done in a specific order. One can at any time add, delete and change either of them. It is also possible to reorder the actions, conditions and condition states (see menu page).

5.3.4. Adding Decision Rules

When conditions and actions have been entered, one can start establishing the relations between them, the decision rules. A decision rule indicates which specific set of condition combinations leads to a certain action.

- action input -

Actions for 'ORDER TREATMENT' :

- [1] Execute order
- [2] Refuse order
- [3] Put order on waiting list

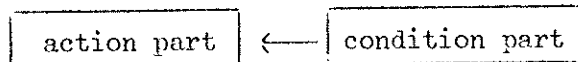
Enter new action or number of action to be changed :

>

-data- to change contradictions -copy- enabled
-help- available -shift data- for menu

Fig. 5.4. : Entering actions

The input of decision rules is based upon the following format (1)



This formulation implies that the actions in the action part have to be undertaken when the condition combination in the condition part holds true.

Actions and conditions are represented by their numbers, condition states by their letters.

The decision rule

$1 \leftarrow 1a$

then means : Action 1 should be executed if condition 1 satisfies state a.

Replacing condition and action names by their numbers largely simplifies the input and error checking effort.

The action part may consist of several actions, if all these actions have to be executed in the specified situation. The action numbers are then connected by 'and' (2).

In the same way, the condition part may contain several conditions, connected by 'and' or 'or'. The use of parentheses is inevitable in some cases to avoid ambiguity.

Some valid decision rules : $1 \leftarrow 1a$

$2 \leftarrow 1a \text{ and } 2a$

$3 \text{ and } 4 \leftarrow 1a \text{ and } (2b \text{ or } 3b).$

(1) A short outline of this method was given in chapter 3.

(2) The PRODEMO software is developed in English. User input, however, is also accepted in Dutch, French or German.

As already mentioned, conditions can take several values, but only one at the time. Condition states are therefore separated by 'or'.

To represent several condition values, one writes e.g. :

1a or 1c,
or shorter : 1ac

Some actions always have to be undertaken, e.g.

1 and 2 \leftarrow always.

Some condition combinations are impossible. They cannot occur and are removed from the resulting table. These combinations are indicated as follows, e.g.

impossible \leftarrow 1a and 3b

The decision logic is entered in this form of logical expressions. The decision rules are first checked on syntax and then implemented in the table under construction.

As far as possible, semantic errors are also detected, e.g.

impossible \leftarrow always : would mean that all condition combinations are impossible
 impossible \leftarrow 1a : would mean that 1a is not necessary, so it can be deleted
 1 \leftarrow 1abcdef : enumerating all states of a condition makes the condition irrelevant in the given rule.

At any moment, the user can take a look at the list of already entered decision rules (Fig. 5.6.). In order to correct or complete the existing decision table, the decision rules can simply be adjusted.

Because of their high information value, the decision rules are automatically adapted to all changed circumstances.

E.g. the rule 1 \leftarrow 1ab and (2a or 3a) becomes :

- after interchanging condition 1 and 2 :

1 \leftarrow 2ab and (1a or 3a)

- after deleting condition 2 :

1 \leftarrow 1ab and 3a

- after deleting states a and b of condition 1 :

1 --- 2a or 3a

The decision rules are entered in the above format (fig. 5.5.). It is always possible to request more information on how decision rules have to be entered, changed or deleted.

5.3.5. Inspecting decision rules

This option allows the PRODEMO user to inspect all decision rules he has already entered, but now in a more readable format, i.e. condition, action and state symbols are replaced by the original names.

The decision rules which describe the above example, are given in fig.5.6. As an illustration, the verification of the second rule is displayed in fig. 5.7.

It can be seen here that the decision rules, with their unambiguous interpretation, are the transition between the action-oriented decision text (an action should be executed if...), and the condition-oriented decision table (a condition combination leads to ...).

5.3.6. Default options

The user can choose a number of options concerning the construction and the tabular representation of the decision. Unless he wants to change any of them, the normal default options are taken.

The most important among them are :

- Automatic table diagnosis (ON/OFF), default = ON

The diagnosis leads to the detection of manifest shortcomings in the constructed table, like contradictory actions, unreferenced actions, empty decision situations, etc..

last rule → 1 + 2a

decision input

ACTIONS	CONDITIONS
<p>1° Execute order 2° Refuse order 3° Put order on waiting list</p>	<p>1° Credit limit exceeded ? <input type="checkbox"/> Yes <input type="checkbox"/> No 2° Sufficient stock ? <input type="checkbox"/> Yes <input type="checkbox"/> No 3° Important customer ? <input type="checkbox"/> Yes <input type="checkbox"/> No 4° Amount involved ? <input type="checkbox"/> ≤ 100 <input type="checkbox"/> > 100</p>
<p>Enter new decision rule (-data- to change decision) :</p>	
<p>></p>	

-data- for list of decisions
 -help- available

-copy- enabled
 -shift data- for menu

Fig. 5.5. : Decision input

list of decision rules

Decision rules for 'ORDER TREATMENT' :

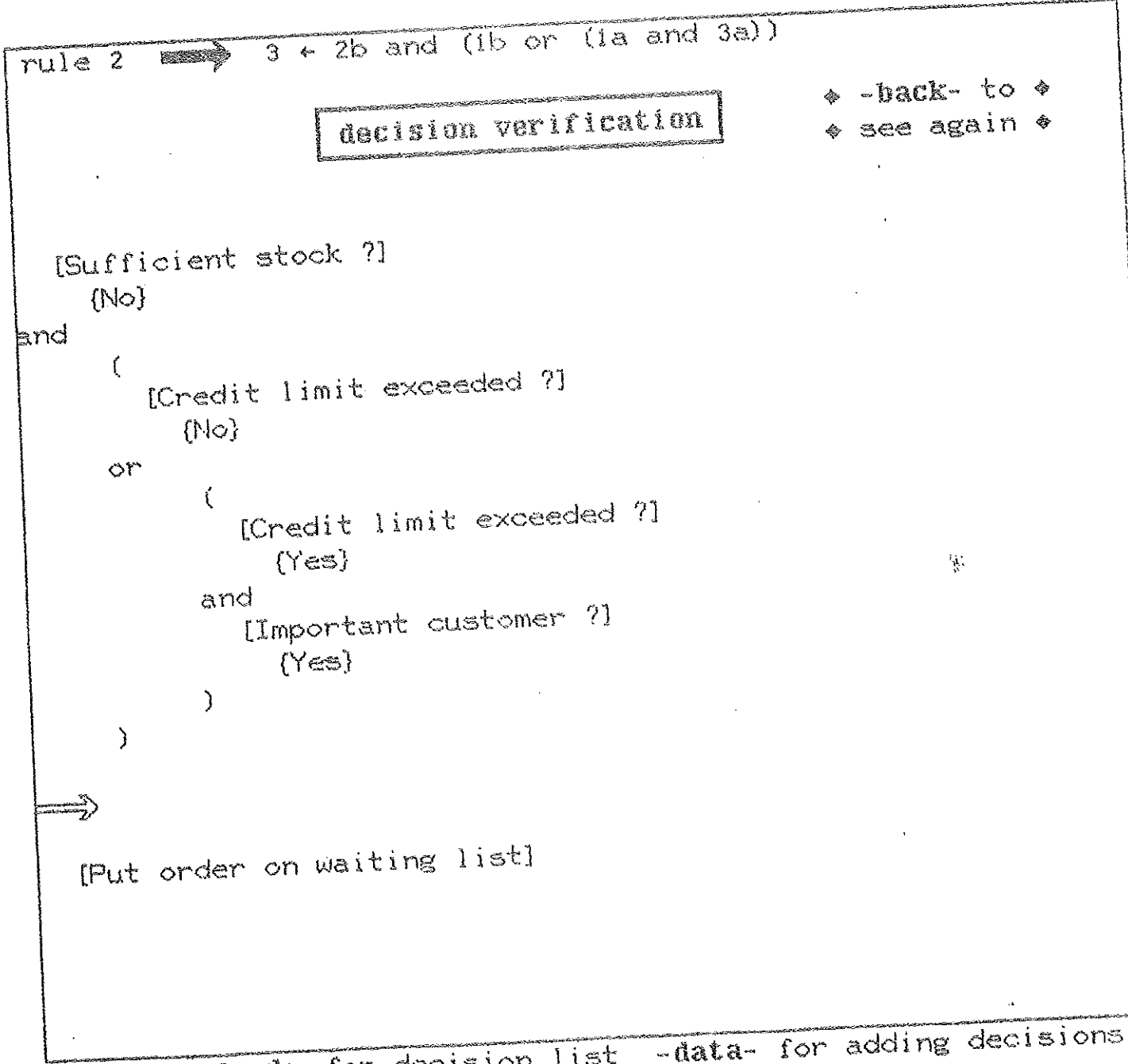
- [1] 1 ← 2a and (1b or (1a and 3a))
- [2] 3 ← 2b and (1b or (1a and 3a))
- [3] 2 ← 1a and 3b and 4b
- [4] 1 ← 2a

Enter no. of rule to be changed (shift help to delete) :

>

-data- for adding decisions ♦ *number for verification ♦
-help- available -shift data- for menu

Fig. 3.6. : Changing, deleting and inspecting decision rules.



-shift back- for decision list -data- for adding decisions
-shift data- for menu -help- available

Fig. 5.7. : Decision verification.

- Decision table constructed (CONTRACTED/EXPANDED), default = CONTRACTED
With this option, the table can be contracted before being plotted upon the screen, i.e. decision columns leading to the same decisions are reduced to one simple situation (for a definition : see section 3.1.).
- Decision table stubwidth, default = 20 characters.
The stubwidth of a decision table is the place where the condition and action names have to be put.
Due to screen limitations, the user must choose between a large stubwidth and a large number of decision columns displayed on one single screen.
If the stubwidth is too small the condition and action names have to be split up, so that the length of the screen may not be sufficient. If the stubwidth is too large there is not much space left for the decision columns and the number of screens would become unacceptably high.

5.3.7. Table construction

At each stage, the user can ask the PRODEMO system to show the decision table. Constructing the table implies matching the decision rules to an expanded table, contracting the table, checking for errors and displaying the table on the screen. If the table contains too many columns to fit into one screen, the display is continued on another screen.

Fig. 5.8. shows the table which was constructed for the earlier given example . When constructing this table, the system reported two shortcomings :

- the third column contains two contradictory actions :
'Execute order' and 'Refuse order'
- the fifth column contains no actions at all.

It is evident that these errors originate from the decision text, but discovering them there will not be as easy. The specific decision table format enables both the user and the system, to find such contradictions or shortcomings in a standard way.

ORDER TREATMENT

01/23/81

Credit limit exceeded ?	Yes						No	
	Yes			No			Yes	No
Sufficient stock ?	Yes	No		Yes	No		-	-
Important customer ?	Yes	No		Yes	No		-	-
Amount involved ?	-	\$ 100 >	100	-	\$ 100 >	100	-	-
Execute order	x	x	x	-	-	-	x	-
Refuse order	-	-	x	-	-	x	-	-
Put order on waiting list	-	-	-	x	-	-	-	x

-back- to replot
-shift data- for menu

-help- available
-data- to touch

END OF TABLE
-shift back- to expand

Fig. 5.8. : Resulting table

5.3.8. Changing and completing the table

Once the errors in the decision text are found, we can update the decision table accordingly in order to get an unambiguous decision description. PRODEMO not only detects the errors in the decision table, but also offers an elegant way to perform and control the necessary changes.

On one hand, the errors can be corrected by adopting the decision rules and constructing the table again then. On the other hand, the table can be changed directly, i.e. by changing some action entries '-' into 'X' or vice versa. It is sufficient to touch the screen at the desired locations, because the PLATO system is equipped with a touch-sensible screen. After changing some action entries, the table is again constructed, the decision rules are automatically updated and the revised table is displayed again.

When the necessary corrections are made, the final table could look like the one in figure 5.9. This table is a much more reliable and faster tool of management than the existing regulation we started from.

5.3.9. Save and reload options

Once the table is constructed and satisfies the user, it can be saved in memory for later use. The user can at any moment reload one of his tables to make a decision or to adapt it to the changed circumstances. The tables in memory can be protected against unauthorized access depending on the user's preference.

There are 2 kinds of table storing

- a. A working storage which is designed to contain decision situations under construction and which can be accessed very easily. These tables can be inspected by anyone, but not changed nor deleted.

ORDER TREATMENT .

01/23/81

Credit limit exceeded ?	Yes				No	
	Yes		No		Yes	No
Sufficient stock ?	Yes		No		-	-
Important customer ?	Yes	No	Yes	No	-	-
Amount involved ?	-	≤ 100	> 100	-	-	-
Execute order	x	x	-	-	-	x
Refuse order	-	-	x	-	x	-
Put order on waiting list	-	-	-	x	-	x

-back- to replot

-help- available

END OF TABLE

-shift data- for menu

-data- to touch

-shift back- to expand

Fig. 5.9. : Final table.

- b. A public library containing completed decision descriptions, subdivided in 10 sets according to the subject of the table. At present, this library can handle up to 200 decision tables.

The library tables can take three access levels :

- any user can change and inspect the table
- anyone can inspect the table, but not change it
- no other user can change or inspect the table.

Not only separate decision tables are saved in the public library. Interrelated tables can be connected in table structures, which is the subject of the next topic.

5.3.10. Decision table structures

A table structure is a collection of interrelated decision tables, concerning the same decision situation. The relations are formed by the fact that some tables are a further specification of a condition or an action of another table. In the example outlined above, we found an action, called 'Execute order'. This action could lead to the construction of a new, more detailed, table, containing delivery or shipment details. These two tables would form a structure (also called system).

In most cases, all tables from a system could be combined into one single table with the same logic. Such table, however, would be so large that it would be completely useless. Moreover, the construction of small and related tables, containing coherent decision information, offers some important advantages. It enables the designer to focus on only the relevant aspects of the decision situation part by part and to keep the decision structure in mind. It therefore adds to the modularity and the top-down approach of the problem description.

The relation between tables in a system can take two forms, because of the distinction between action and condition subtables.

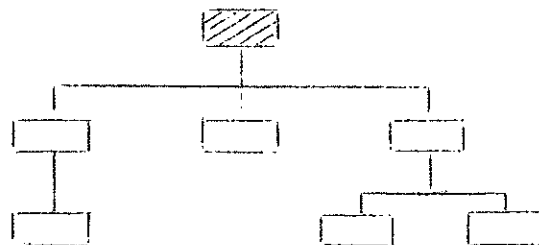
An action subtable is a table which refines an action of another table. If e.g. the table 'Execute Order' is constructed in the above example, it would be an action table as it details the original action 'Execute order'.

A condition subtable, however, refines no action, but a condition of another table. One might e.g. construct a table 'Important customer' for the above example, in order to find out when a customer is considered important. The actions of this table indicate which state of the original condition is satisfied, e.g. customer is important : yes or no.

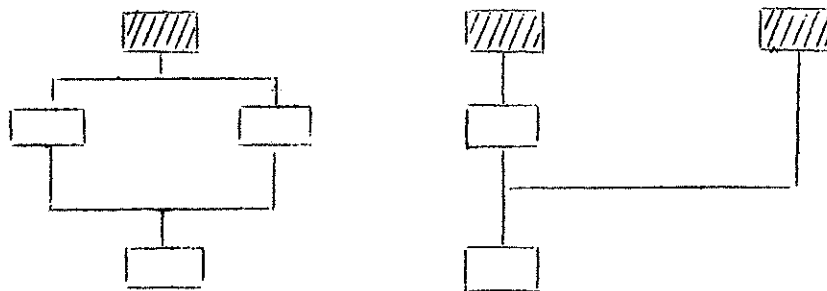
The PRODEMO implementation of table systems shows the following conventions:

- A table system can only contain decision tables of the same public library set (i.e. maximum 20 tables)
- When an action (or condition) refers to a subtable, both the action (or condition) name and the subtable name should be identical and start with A : (C : for conditions), in order to establish the relation between them.
- Only tree structures are allowed. Tree structures consist of 1 head table with underlying levels of subtables, so that each subtable has one and only one higher (parent) table. A table can refer to various (lower level) subtables, but it can only be referred itself by one (higher level) table.

The structure of a system might e.g. look like this



Illegal structures would be :



In both cases a subtable is present which has more than one parent table.

The implementation of table systems in PRODEMO is restricted to the public library, i.e. when a table is saved, the computer asks whether the table should be part of a system, and, if yes, which is the parent table. Safety levels, which were allowed for separate tables, are automatically valid for the complete system as a whole.

The structure of the table system can be displayed under tree format (if the screen dimensions allow) or by indentation.

Tables of the same system can easily be displayed by reloading the complete system from the library, which allows the user to focus on one table, but to keep the rest of the system within reach.

5.3.11. Decision making

PRODEMO and the decision table technique serve a double objective : first the construction and analysis of tables which represent a decision situation; second the possibility of easy and correct decision making with the resulting table(s).

This last option allows the user to make decisions from a table he constructed or reloaded from table memory.

The procedure for making decisions is quite simple. The decision maker just has to enter the range for each condition (Fig. 5.10), so that the relevant decision table column can be traced down. Subtables in systems are automatically inserted in the decision process. At the end of the subtable, decision making returns to the higher level table and continues there with the results obtained in the subtable. A complete system can be travelled down this way.

If one is working with a contracted table, PRODEMO will look up irrelevant conditions (indicated by a '-' in the decision table). As the condition state plays no role here, it need not be entered.

The actions which result from the decision situation are automatically generated and displayed (see fig. 5.11.). If no actions are present, this is indicated with 'NO ACTIONS'. Impossible rules are also detected and marked with 'IMPOSSIBLE RULE'.

6. DIRECTIONS FOR FUTURE EXTENSIONS

A first extension in the immediate future will be the creation of private decision table data bases. A distinction has to be made between the public library and the private library, the former being accessible by all users while the latter is reserved for each specific user. These libraries will be used to store completed decision descriptions.

Attention will be paid to the implementation of a larger decision intelligence, concerning a.o. the automatic separating of complex decision tables, the development of efficient contraction procedures, leading to optimal condition order and minimal table length, and the extension of the response mode.

decision making

Table : ORDER TREATMENT

[1] Credit limit exceeded ?

Yes

No

Enter value (letter) for condition 1 >

-help- available

-back- for another decision

-shift data- for menu

Fig. 5.10 : Decision making : procedure.

decision making

.Table : ORDER TREATMENT

- [1] Credit limit exceeded ?
 B No
- [2] Sufficient stock ?
 A Yes
- [3] Important customer ?
 -irrelevant-
- [4] Amount involved ?
 -irrelevant-



- [1] Execute order
- END -

-help- available

-back- for another decision

-shift data- for menu

Fig. 5.11. : Decision making : result.

The response mode which is now only partly available will enable the user to transfer control to the PRODEMO system. The system analyzes the current decision situation, gives suggestions for the table construction and enables the user to adapt the decision description accordingly.

7. A SHORT NOTE ON THE CHOICE OF THE PLATO ENVIRONMENT

The Control Data Plato System is an interactive computer-based education system originally developed at the University of Illinois under the direction of Dr. D.L. Bitzer. It assists in all phases of the teaching process and has extensive application to business and government as well as to academic institutions. The heart of any Plato system is a large (64-bit) CDC CYBER 73 computer; up to 1024 terminals can be connected to this central device.

Our choice of the Plato System as the host system of PRODEMO, was not only influenced by its incidental presence in our testing environment. We daily appreciate its

- wide facilities for creating visually attractive displays
- touch-sensitive display screen
- quick response time
- challenging environment offered by the Plato community.

These and other capabilities obviously compensate for some annoying inconveniences (e.g. communication errors).

We further believe that Computer Assisted Instruction as embodied by the Plato approach cannot be confined to material traditionally dealt with by classical instructional techniques. Only open-ended lessons aiming at increasing skilfulness and insight rather than pure knowledge can shove away the often felt and even more discussed economic and mental barrier. In this light, one can speak of the PRODEMO "lesson".

ACKNOWLEDGEMENTS

Many fundamentals of the PRODEMO System go back to earlier ideas expressed by Prof. Dr. M. VERHELST, who further stimulated our research by his clear judgment and his abundant advices. The G. BANK and in particular Mr. J.ROMBOUTS, head of the Management and Development Training procured both the computing facilities and our first testing environment (and we fully know how frustrating the latter role may be !). Their continuing effort is highly appreciated. Besides, the Louvain head branch of the same G. BANK and especially Mr. E. HEYVAERT were (and are) our daily hosts. Finally, we want to acknowledge that the name 'PRODEMO' was prompted by an advice 'pro Deo' by S. LEURS, the wife of one of the authors.

The contribution by R. MAES goes back to research sponsored by the Onderzoeksfonds of the K.U. Leuven under Grants No. OT/II/8 and OT/IV/12, while the ongoing research by J. VANTHIENEN is made possible by the Research Program in Management of the C.I.M. (Project No. 10).