Decision Table: A Procedural Approach to Making Decisions in Complex Situations

***Edward E. Ogheneovo**, Dept. of Computer Science, University of Port Harcourt, Nigeria.

Promise A. Nlerum, Dept. of Computer Science and Informatics, Federal University, Otuoke, Nigeria.

*edward.ogheneovo@uniport.edu.ng, nlerumpa@fuotuoke.du.ng

Abstract

Decision tables are used to model complicated logic. Building and maintaining knowledge base is not a trivial task. Knowledge acquisition, verification efforts are major problems in construction process. Decision table is a recommended tool for verification and validation processes in knowledge bases due to the fact that it has the ability to detect incompleteness and inconsistency. Thus a decision table is used to represent conditional logic by creating a list of tasks depicting business. They are usually concise visual representation for specifying which actions to perform depending on given conditions. In this paper, we discussed decision tables as a procedural approach or technique to making decisions in complex situations. The paper also discuss how decision tables can be built and its development in recent times as a tool for making complex decisions in complex situations in artificial intelligence especially in the areas of knowledge base and expert systems.

Keywords: Decision tables, decision making, knowledge bases, verification, validation, incompleteness, inconsistency

1.0 Introduction

One of the major challenges for a Systems Analyst is to effectively communicate the system requirements to a diverse audience [1] [2]. Quite often this entails taking facts gotten from various stakeholders and presenting the in readable form with enough details for facilitate testing and activities involved in the development processes [3]. Supplemental specifications and other techniques such as data definition, prototyping, use case, unified modeling language (UML), etc., can be effective ways to communicate the functional and non-functional requirements to designers and system analysts. Also, structure English can be used. However, all these techniques may have limited use at some points when used to describe complicated business decision logic. This is where decision tables and decision trees come in. Decision tables work equally well with functional and non-functional requirements [4] [5]. They also help to identify redundant logic, which can be simplified by the use of indifferent conditions.

Decision tables are used to model complicated logic [6] [7] [8]. Building and maintaining knowledge base is not a trivial task. Knowledge acquisition, verification efforts are major problems in construction process. Decision table is a recommended tool for verification and validation processes in knowledge bases due to the fact that it has the ability to detect incompleteness and inconsistency. Thus a decision table is used to represent conditional logic by creating a list of tasks depicting business. They are usually concise visual representation for specifying which actions to perform depending on given conditions. Building and maintaining knowledge base is not a trivial task. Knowledge acquisition and verification efforts are two major problems in construction process. Decision table is a recommended tool for verification and validation processes in knowledge bases due to the fact that it has the ability to detect incompleteness and inconsistency [9] [10]. Thus a decision table is used to represent conditional logic by creating a list of tasks depicting business. They are usually concise visual representation for specifying which actions to perform depending on given conditions. They are algorithms whose output is a set of actions. The information expressed in decision tables could also be represented as decision trees or in a programming language in the form of if – then – else and switch – case statements. A decision table is a tabular representation used to describe and analyze procedural decision situations, where the state of a number of conditions determines the execution of a set of actions [11]. Not just any situation but one in which all distinct situations are shown in columns in a table such that every possible case is included in one and only one column, i.e., completeness and exclusivity [12] [13].

Propositional rule–based systems can take various forms incorporating some representation, most important are: 1) decision tables and 2) decision trees. A decision table is a table representing the exhaustive set of mutual exclusive conditional expressions within a predefined problem area [14] [15] [16]. In most cases, decision tables are preferred to other representation techniques such as texts, decision trees, flowcharts, IF – THEN constructs, Horn-clauses, nested IF=THEN-ELSE constructs). The have the advantages in that they are very concise, flexible, easy checking for completeness, consistency and correctness, readable, in two ways (data oriented, goal oriented), correctness, and quick decision-making process, uniform communication medium, automatic conversion into computer programs, simple specification for test data, and easy of translation between languages [18] [19]. Although, the use of decision tables have been in existence for quite some time now, when it was first used to construct the logic of computer programs and other applications involving procedural decision making process but also for representing complex decision situations in a compact way, complex decision situations in a compact way, easy to check for completeness and consistency. However, recent developments in artificial intelligence especially in the arear of knowledge base and expert systems have led to a renewed interest in the technique [20] [21].

In this paper, we discussed decision tables as a procedural approach or technique to making decisions in complex situations. The paper also discuss how decision tables can be built and its development in recent times as a tool for making complex decisions in complex situations in artificial intelligence especially in the areas of knowledge base and expert systems. Various examples of decision tables are also discussed.

2.0 Building Decision Tables

The tabular representation of the decision situation is characterized by the separation between conditions and actions on one hand and between subjects and conditional expressions (States) on the other hand. Thus the decision table has three main components: 1) conditions, 2) actions, and 3) rules. The conditions are the factors one should consider when making certain business decision; the actions are the possible decisions to take when certain decision(s) is / are made; and the rule part combines the condition(s) and action(s) that forms the knowledge base. Thus every table column (decision column) indicates which actions should (or should not) be executed for a specific combination of condition states. Table 1 shows an example of a decision table.

Table 1: An example of a decision table

	Rules					
Condition	1	2	3	4	5	6
C1: Address Proof provided	N		Y	Y	Y	Y
C2: Identity proof provided		N	Y	Y	Y	Y
C3: Loan amount < monthly salary			Y			
C4: Loan amount >= monthly salary					Y	Y
C5: Loan purpose				Home Purchase	Pay Tax	Other
C6: Home owner?						
Action	1	2	3	4	5	6
A1: Approved loan request immediately			X	X	X	
A2: Review loan request manually						X
A3: Reject loan request	X	X				

Table 2.1 shows the decision table for loan request. It contains three components as stated earlier: condition(s), action(s) and rules which combines the condition and action to determine if an action will be taken or not. These signifies "Y" for "Yes" and "N" for "No". Hence, decision making is a way to decision making that involves considering a variety of conditions and their interrelationships particular for complex interrelationships. Managers of industries, organizations and experts use decision tables to represent and new business logic and knowledge representation model and guidelines represented in rules to make decisions. These rules sets are often verified to ensure completeness and consistency. Decision tables are used by computer programmers and systems analysts to confirm completeness and identify ambiguity in rule sets. They are used to display, verify and optimize guideline knowledge as logically cohesive sets of rules. A rule set represented in decision table form can be readily checked to assure comprehensiveness and the absence of redundancy and contradiction.

In medicine, for example, decision variables include patients' symptoms, physical examination findings, and the results of laboratory tests. Actions include initiating a treatment, undertaking a risky or expensive diagnostic evaluation, or concluding a diagnosis. In a decision table, each decision value is represented as a categorical value (e.g., diabetes is present or absent) or as a range of a continuous variable (e.g., cholesterol > 270 mg/d1). The number of values that each decision variable can assume is defined as the modulus of the decision variable. The conventional display of a decision table lists the condition (i.e., decision variables) in the upper left quadrant and the relevant actions on the lower left quadrant. The rules are listed on the right quadrants in each of the columns. That is, each column is a rule whose antecedents are derived from the condition entries and whose consequents are indicated by the action entries below them. Sometimes, the value of a particular decision variable is irrelevant to the satisfaction of a rule. For example, in deciding whether to treat a patient who has sore throat, cervical adenopathy, and a positive throat culture, the presence or absence of adenopathy is immaterial, although it may be an important consideration if the throat culture result is unknown. Such irrelevant decision values are represented by dashes (-) in the table.

Therefore, decision tables are used to model complicated programming logic (Structure) thus making it easier to ensure that all possible combination of conditions have been considered. Table 2 Shows a structured English process description of a program.

```
IF    Student – Age < 22 THEN
        IF Default – Free = “N” THEN
                    Surcharge = 0.2

            ENDIF
    ELSE
            IF Student – Gender = “F” THEN
                        Surcharge = 0.15
            ENDIF
    ELSE
            IF Student – Year 4 = “N” THEN
                        Surcharge = 0.10
            ENDIF
    ELSE
            IF Accommodated = “N” THEN
                        Surcharge = 0.10
            ENDIF
    ELSE
            IF Bally - behaved THEN
                    Surcharge = 0.15
            ENDIF
    ELSE
            IF Student – CGPA < 4.5 THEN
                        Surcharge = 0.05
            ENDIF
    ELSE
            IF Student – CGPA > = 4.5 THEN
                        Surcharge = 0.001
            ENDIF
    ELSE
            IF Sick – Free = “Y” THEN
                        Surcharge = 0.00
ENDIF
```

Table 2: Students' Decision Table

Student Age	**22 yrs +**	**22 yrs +**	**<22 yrs**	**< 22 yrs**	**<22 yrs**	**< 22 yrs**	**< 22 yrs**	**< 22 yrs**
Defaulter Free	Y	N	N	Y	Y	Y	Y	Y
Student Gender	-	-	-	Female	Male	Male	Male	Male
Student Year 4	-	-	-	-	N	Y	Y	Y
Accommo dated	-	-	-	-	-	N	Y	Y
Badly behaved	-	-	-	-	-	-	N	Y
Student CGPA	-	-	-	-	-	-	24.5	4.5+
20% Surcharge				X				
15% Surcharge								
10% Surcharge					X	X		
5% Surcharge			X				X	
1% Surcharge		X			X			X
No Surcharge	X							

Developing a decision tables require a number of steps. First, one need to determine the maximum size of the table, eliminate any impossible situations, eliminate inconsistencies or redundancies, and then simplify the table as much as possible. These steps are further discussed below.

STEP 1: Determine the number of conditions that may affect the decision. To do this, we need to combine overlapping rows, for example, rows that their conditions are mutually exclusive after which the number of conditions becomes the number of rows in the top half of the decision table.

Conditions	Condition Alternatives
Actions	Action Entries

Such mutually exclusive conditions are: Gender (Male and Female), Education (Secondary School, Bachelor, Master, and Doctor, etc.). To simplify a table, mutually exclusive condition should be combined to form one condition with multiple alternatives.

STEP 2: Determine the number of possible actions that can be taken as shown in the table.

Conditions	Condition Alternatives
Actions	Action Entries

This becomes the number of rows in the lower half of the decision table.

STEP 3: Determine the number of condition alternatives for each condition
In the simplest form of decision table, there would be two alternatives (“Y” for Yes or “N” for No) for each condition. However, in an extended table, there may be many alternatives for each condition. The table below illustrate is used to determine the number of condition alternatives.

Conditions	**Condition Alternatives**
Actions	Action Entries

STEP 4: Calculate the maximum number of columns in the decision table.
This is done by multiplying the number of alternatives for each condition. For example, if there are four conditions and two alternatives for each of them, there will be $2^4 = 16$ possibilities.

Conditions	Condition Alternatives
Actions	**Action Entries**

STEP 5: Fill the condition alternatives.
This is done by starting with the first condition and divide the number of columns by the number of alternatives for that condition. The table below shows an example of how to fill in the condition alternatives.

Table 3: Filling alternative conditions in a decision table

Condition 1	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N
Condition 2	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N
Condition 3	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
Condition 4	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N

STEP 6: Complete the table by inserting an X where rules suggest certain actions.

STEP 7: Combine rules where it is apparent that an alternative does not make a difference in the outcome

STEP 8: Check the table for any impossible situations, contradictions, redundancies.

STEP 9: Rearrange the conditions and actions (or even rules) to make the decision table more understandable.

Suppose a customer decides to buy a product for personal use. The conditions are: 1) the transaction must be above $100, 2) pay by cheque, 3) or pay using credit card. The possible outcomes are: 1) call up sale, 2) check local database for customer's details 3) call the sales manager, and 4) verification of credit card from service company database.

Table 4: Customer verification in a company's database

Conditions	**Condition rule**							
Above $ 100	Y	Y	Y	Y	N	N	N	N
Pays with Cheque	Y	Y	N	N	Y	Y	N	N
Pays with credit card	Y	N	Y	N	Y	N	Y	N
Action	**Action Entries**							
Call sales department								
Check local database for customer's								
Call sales manager								
Verify credit card database								

Those conditions underlined are invalid because a customer cannot pay by cheque and pay with credit card at the same time or not pay by either cheque or credit card. The customer must pay for the product if s/he intends to go with it. These invalid condition rules must be deleted. This takes us to Table 5. Also, Table 6 is used to indicate the action entries for the customer's table.

Table 5: Customer verification in a company's database

Conditions	**Condition rule**			
Above $ 100	Y	Y	N	N
Pays with Cheque	Y	N	Y	N
Pays with credit card	N	Y	N	Y
Actions	**Action Entries**			
Call sales department				
Check local database				
Call sales manager				
Verify credit card database				

Table 6: Customer verification in a company's database indicating the action entries

Conditions	**Condition rule**			
Above $ 100	Y	Y	N	N
Pays with Cheque	Y	N	Y	N
Pays with credit card	N	Y	N	Y
Actions	**Action Entries**			

Call sales department	X			
Check local database		X		
Call sales manager			X	
Verify credit card database				X

The next step is to check for completeness. Supposing the customer has never shopped in that supermarket before. All we need do is to add a new condition to the original table.

Table 7: Checking for completeness in customer verification in a company's database

Conditions	**Condition rule**							
Above $ 100	Y	Y	Y	Y	N	**N**	**N**	N
Pays with Cheque	Y	Y	N	N	Y	Y	N	N
Pays with credit card	N	N	Y	Y	N	N	Y	Y
Unknown customer	Y	N	Y	N	Y	N	Y	N
Actions	**Action Entries**							

In Table 7, we have added a new condition (i.e., unknown customer) since the customer's details are not found in their database. Thus the number of condition have increases by a multiple of 2 (i.e., the number of alternatives for the new condition). After this we then indicate the actions as shown in Table 8. note that the number of actions remain the same.

Table 8: Indicating the actions

Conditions	**Condition rule**							
Above $ 100	Y	Y	Y	Y	N	N	N	N
Pays with Cheque	Y	Y	N	N	Y	Y	N	N
Pays with credit card	N	N	Y	Y	N	N	Y	Y
Unknown customer	Y	N	Y	N	Y	N	Y	N
Actions	**Action Entries**							
Call sales department		X						
Check local database for customer's				X				
Call sales manager	X				X	X		
Verify credit card database			X				X	X

In Table 9, we delete mutually exclusive condition. Recall that a customer cannot pay for a product using cheque and credit card at the same time, we need to eliminate them since they are mutually exclusive.

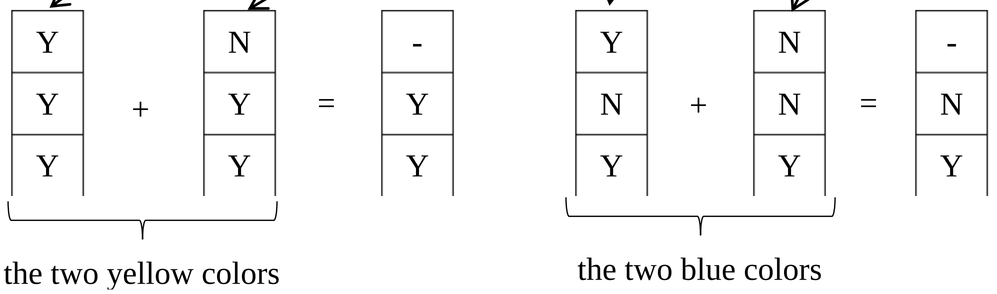
Table 9: Delete mutually exclusive conditions

Conditions	**Condition rule**							
Above $ 100	Y	Y	N	N	Y	Y	N	N
Pays with Cheque	Y	Y	N	N	Y	Y	N	N
Pays with credit card	-	-	-	-	-	-	-	-
Unknown customer	Y	N	Y	N	Y	N	Y	N
Actions	**Action Entries**							
Call sales department		X						
Check local database for customer's				X				
Call sales manager	X				X	X		
Verify credit card database			X				X	X

As see in Table 10, the pay with cheque has dash in the condition entries' all through. What this means is that it does not about the condition rule, pays with credit card because if a customer already pay with cheque s/he can no longer pay with credit card.
Table 10 Combines is used to possible situations according to actions.

Table 10: Pay with cheque

Conditions	**Condition rule**							
Above $ 100	Y	Y	Y	Y	N	N	N	N
Pays with Cheque	Y	Y	N	N	Y	Y	N	N
Pays with credit card	-	-	-		-	-	-	-
Unknown customer	Y	N	Y	N	Y	N	Y	N
Actions	**Action Entries**							
Call sales department								
Check local database for customer's								
Call sales manager	X				X	X		
Verify credit card database			X				X	X



the two yellow colors

the two blue colors

In Table 11, we removed the redundant columns having combined all possible situations.

Table 11: Eliminate redundant columns

Conditions	**Condition rule**					
Above $ 100	-	Y	-	Y	N	N
Pays with Cheque	Y	Y	N	N	Y	N
Pays with credit card	-	-	-	-	-	-
Unknown customer	Y	N	Y	N	Y	N
Actions	**Action Entries**					
Call sales department		X				
Check local database for customer's				X		
Call sales manager	X				X	
Verify credit card database			X			X

Table 12: Combining identical actions

Conditions	**Condition rule**					
Above $ 100	-	Y	-	Y	N	N
Pays with Cheque	Y	Y	N	N	Y	N
Pays with credit card	-	-	-	-	-	-
Unknown customer	Y	N	Y	N	N	N
Actions	**Action Entries**					
Call sales department		X				
Check local database for customer's				X		
Call sales manager	X				X	
Verify credit card database			X			X

Table 13: Final Decision Table

Conditions	**Condition rule**					
Above $ 100	-	Y	-	Y	N	N
Pays with Cheque	Y	Y	N	N	Y	N
Unknown customer	Y	N	Y	N	N	N
Actions	**Action Entries**					
Call sales department		X				
Check local database for customer's				X		
Call sales manager	X				X	
Verify credit card database			X			X

Table 12 is used to combine identical actions and finally Table 13 shows the final decision table after the redundant conditional has been eliminated.

Conclusion

Decision tables are used to model complicated logic. Building and maintaining knowledge base is not a trivial task. Knowledge acquisition, verification efforts are major problems in construction process. Decision table is a recommended tool for verification and validation processes in knowledge bases due to the fact that it has the ability to detect incompleteness and inconsistency. Thus a decision table is used to represent conditional logic by creating a list of tasks depicting business. They are usually concise visual representation for specifying which actions to perform depending on given conditions. In this paper, we discussed decision tables as a procedural approach or technique to making decisions in complex situations. The paper also discuss how decision tables can be built and its development in recent times as a tool for making complex decisions in complex situations in artificial intelligence especially in the areas of knowledge base and expert systems. Various examples of decision tables are also discussed.

References

1. Garcia, A. M. M., M. Verhells and J. Vanthieman (2003). An Overview of Decision Table Literature 1982-2000. In Proceedings of the 5th Int'l Conference on Artificial Intelligence and Emerging Technologies in Accounting, Finance and Tax. Research Group on Artificial Intelligence in Accounting (GIACA), 2-3 November, 2003, Huewa, Spain.
2. Von Halle, B. and L. Goldberg (2010). The Decision Model: A Business Logic Framework Linking Business and Technology CRC Press, Auerbach Publications, New York.
3. Tumang, K. (1996). Business Process Simulation. In Proceedings of the 28th Conference on Winter Simulation, IEEE Computer Society, Washington DC, Washington, USA.
4. Rajaraman, V. and N. R. Gerud (1996). Non-deterministic Decision Tables in Process Control. Sādhanā, Vol. 21, Part 3, pp. 381-293.
5. Sharma, M. and S. Chandra (2010). Automatic Generation of Test Suites from Decision Table - Theory and Implementation. In Proceedings of the 5th Int'l Conference on Software Engineering Advances, Nice, France, August 2010, ACM, pp' 459-464.
6. Linzi, Y., D. Jiafeng, X. Xu and K. Sun (2017). A Tree-Way Decision Model for Decision Tables. In Proceedings of the IEEE 29th Chinese Control and Decision Conference (CCDC), pp. 258-263.
7. Deng, X. F. and Y. Y. Yeo (2014). Decision-Theoretical Three-Way Approximations of Fuzzy Sets, Information Sciences, Vol. 279, pp. 703-705.
8. Varthienen, J. (1991). A Longest Path Algorithm to Display Decision Tables. The Computer Journal, Vol. 34, Issue 4, pp. 358-380.
9. Varthienen, J. (1991). Knowledge Acquisition and Validation Using a Decision Table Engineering Workbench. In Proceedings of the World Congress on Expert Systems, Pergamon Press, Orlando, Florida, Dec. 16-19, 1991, pp. 1861-1866.
10. Linetzki, A. (2012). Analyzing Decision Tables: A simple Process for Learning, Analyzing and Producing Test Cases from Decision Tables. Testing Experience Magazine.
11. Kalmegh, S. R. (2018). Comparative Analysis of WEKA Classifiers Rules Conjunctive Rule & Decision Table on Indian News Dataset by Using Different Test Mode. Int'l Journal of Engineering Invention (IJES), Vol. 7, Issue 2, Ver. 111, pp. 1-9.

12. Witten, I. H. E. Frank and M. A. Hall (2011). Data Mining Practical Machine Learning Tools and Techniques, Morgan Kaufmann Publishers, Burlington, MA, USA.
13. Connelly, S. and T. Richardson (2004). Exclusion: The Necessary Difference between Ideal and Practical Consensus. Journal of Environmental Planning and Management, Vol. 47, No. 1, pp. 3-17.
14. Bohanes, M. (2009). Decision Making: A Computer Science and Information Technology Viewpoint. Interdisciplinary Description of Computer Systems, Vol. 7, No. 2, pp. 22-37.
15. Bouyssou, D., T. Merchant, N. Pirlot, A. Tsoukies ans P. Vincke (206). Evaluation and Decision-Making Models with Multiple Criteria: Stepping Stones for the Analyst. Int’l Series Operations Research and Management Science 86, Springer, Boston, USA.
16. Varthienen, J. and G. Wets (1995). Integration of the Decision Table Formalism with a Relational Database Environment. Information Systems, Vol. 20, No. 7, 595-616.
17. Santos-Gomez, L. and M. Darnell (1992). Empirical Evaluation of Decision Tables for Constructing and Comprehending Expert System Rules. Knowledge Acquisition, Vol. 4, pp. 427-444.
18. Vanthienen, J. (1991). Knowledge Acquisition and Validation Using a Decision Table Engineering Workbench. In the Proceedings of the 4th Int’l Conference on AI and Law, Amsterdam, pp. 282-291.
19. Vanthienen, J. and E. Dries (1993). Illustration of a Decision Table Tools for Specifying and Implementing Knowledge Based Systems. In Proceedings of the 5th Int’l Conference on Tools with Artificial Intelligence (TAI), Boston (Mass.), pp. 98-205.
20. Vanthienen, J. and G. Wets (1993). Building Intelligent Systems for Management Applications Using Decision Tables. In Proceedings of the 5th Annual Conference on Intelligent Systems in Accounting, Finance and Management, Stanford.
21. VAnthienen, J. and M. Snoeck (1993). Knowledge Factoring Using Normalization Theory. Int’l Symposium on the Management of Industrial and Corporate Knowledge (ISMICK’93), October 27-28, Compiegne, pp. 97-106.