WIKIPEDIA

Decision table

Decision tables are a 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 as a series of if-then-else and switch-case statements.

Play golf dataset

Independent variables				Dep. var
OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
sunny	85	85	FALSE	Don't Play
sunny	80	90	TRUE	Don't Play
overcast	83	78	FALSE	Play
rain	70	96	FALSE	Play
rain	68	80	FALSE	Play
rain	65	70	TRUE	Don't Play
overcast	64	65	TRUE	Play
sunny	72	95	FALSE	Don't Play
sunny	69	70	FALSE	Play
rain	75	80	FALSE	Play
sunny	75	70	TRUE	Play
overcast	72	90	TRUE	Play
overcast	81	75	FALSE	Play
rain	71	80	TRUE	Don't Play

Contents

Overview

Each decision corresponds to a variable, relation or predicate whose possible values are listed among the condition alternatives. Each action is a procedure or operation to perform, and the entries specify whether (or in what order) the action is to be performed for the set of condition alternatives the entry corresponds to.

To make them more concise, many decision tables include in their condition alternatives a don't care symbol. This can be a hyphen[1][2][3] or blank,[4] although using a blank is discouraged as it may merely indicate that the decision table has not been finished. One of the uses of decision tables is to reveal conditions under which certain input factors are irrelevant on the actions to be taken, allowing these input tests to be skipped and thereby streamlining decision-making procedures.[5]

Aside from the basic four quadrant structure, decision tables vary widely in the way the condition alternatives and action entries are represented.[6][7] Some decision tables use simple true/false values to represent the alternatives to a condition (similar to if-then-else), other tables may use numbered alternatives (similar to switch-case), and some tables even use fuzzy logic or probabilistic representations for condition alternatives.[8] In a similar way, action entries can simply

Don't care symbol demonstration

		Rules			
Conditions	Feeling energetic?	Yes	No	Yes	No
	Is raining?	Yes	Yes	No	No
Actions	Stay inside.	✓	✓		
	Go running.			✓	
	Tend the garden.			✓	✓

		Rules		
Conditions	Feeling energetic?	—	Yes	No
	Is raining?	Yes	No	No
Actions	Stay inside.	✓		
	Go running.		✓	
	Tend the garden.		✓	✓

The above tables convey identical information, but the second table uses a hyphen as a don't-care symbol for brevity.

represent whether an action is to be performed (check the actions to perform), or in more advanced decision tables, the sequencing of actions to perform (number the actions to perform).

A decision table is considered *balanced*[4] or *complete*[3] if it includes every possible combination of input variables. In other words, balanced decision tables prescribe an action in every situation where the input variables are provided.[4]

Example

The limited-entry decision table is the simplest to describe. The condition alternatives are simple Boolean values, and the action entries are check-marks, representing which of the actions in a given column are to be performed.

A technical support company writes a decision table to diagnose printer problems based upon symptoms described to them over the phone from their clients.

The following is a **balanced decision table** (created by Systems Made Simple).

Printer troubleshooter

		Rules							
Conditions	Printer prints	No	No	No	No	Yes	Yes	Yes	Yes
	A red light is flashing	Yes	Yes	No	No	Yes	Yes	No	No
	Printer is recognized by computer	No	Yes	No	Yes	No	Yes	No	Yes
Actions	Check the power cable			✓					—
	Check the printer-computer cable	✓		✓					—
	Ensure printer software is installed	✓		✓		✓		✓	—
	Check/replace ink	✓	✓				✓		—
	Check for paper jam		✓		✓				—

Of course, this is just a simple example (and it does not necessarily correspond to the reality of printer troubleshooting), but even so, it demonstrates how decision tables can scale to several conditions with many possibilities.

Software engineering benefits

Decision tables, especially when coupled with the use of a domain-specific language, allow developers and policy experts to work from the same information, the decision tables themselves.

Tools to render nested if statements from traditional programming languages into decision tables can also be used as a debugging tool.[9][10]

Decision tables have proven to be easier to understand and review than code, and have been used extensively and successfully to produce specifications for complex systems.[11]

History

In the 1960s and 1970s a range of "decision table based" languages such as Filetab were popular for business programming.

Program embedded decision tables

Decision tables can be, and often are, embedded within computer programs and used to "drive" the logic of the program. A simple example might be a lookup table containing a range of possible input values and a function pointer to the section of code to process that input.

Static decision table

Input	Function Pointer
"1"	Function 1 (initialize)
"2"	Function 2 (process 2)
"9"	Function 9 (terminate)

Multiple conditions can be coded for in similar manner to encapsulate the entire program logic in the form of an "executable" decision table or control table.

Implementations

- Filetab, originally from the NCC
- DETAB/65, 1965, ACM
- FORTAB from Rand in 1962, designed to be imbedded in FORTRAN[12]
- A Ruby implementation exists using MapReduce to find the correct actions based on specific input values.[13]
- Gandalf (https://gndf.io/), an open-source decision tables engine

See also

- Decision trees
- Case based reasoning
- Cause-effect graph
- Dominance-based rough set approach
- DRAKON
- Semantic decision table

References