Decision Table Based Testing

Eike Neumann

March 2023

Decision Table Based Testing

- Essentially a special case of equivalence class testing.
  - A special method for identifying equivalence classes.
  - A special mechanism for mitigating combinatorial explosion.
- Idea: try to capture the possible behaviours of a system using a decision table.

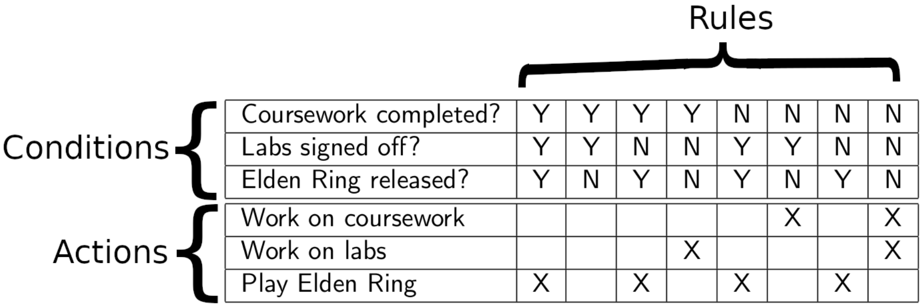
Decision Tables

Decision Tables

Coursework completed?	Y	Y	Y	Y	N	N	N	N
Labs signed off?	Y	Y	N	N	Y	Y	N	N
Elden Ring released?	Y	N	Y	N	Y	N	Y	N
Work on coursework						X		X
Work on labs				X				X
Play Elden Ring	X		X		X		X	

- Structured way of expressing **actions** based on **conditions**.

Decision Tables



		Rules							
Conditions	Coursework completed?	Y	Y	Y	Y	N	N	N	N
	Labs signed off?	Y	Y	N	N	Y	Y	N	N
	Elden Ring released?	Y	N	Y	N	Y	N	Y	N
Actions	Work on coursework						X		X
	Work on labs				X				X
	Play Elden Ring	X		X		X		X	

- Semantics of rules: for a given combination of conditions, take **exactly those actions** that are marked with an “X”.
- This choice of semantics is tailored to the use of decision tables for testing.

Decision Tables

Coursework completed?	**Y**	Y	Y	Y	**N**	N	N	N
Labs signed off?	**Y**	Y	N	N	**Y**	Y	N	N
Elden Ring released?	**Y**	N	Y	N	**Y**	N	Y	N
Work on coursework						X		X
Work on labs				X				X
Play Elden Ring	**X**		X		**X**		X	

- **Observation:** the actions in the first and fifth column are identical.
- The condition “Coursework Completed?” changes its value from “true” to “false” between those columns.
- ⇒ The actions we take do not depend on this condition.

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	Y	Y	N	N	N
Labs signed off?	**Y**	Y	N	N	Y	N	N
Elden Ring released?	**Y**	N	Y	N	N	Y	N
Work on coursework					X		X
Work on labs				X			X
Play Elden Ring	**X**		X			X	

- Table can be simplified by merging the two columns.
- The “-” means that the condition can take any value.

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	**Y**	Y	N	**N**	N
Labs signed off?	Y	Y	**N**	N	Y	**N**	N
Elden Ring released?	Y	N	**Y**	N	N	**Y**	N
Work on coursework					X		X
Work on labs				X			X
Play Elden Ring	X		**X**			**X**	

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	-	Y	N	N
Labs signed off?	Y	Y	N	N	Y	N
Elden Ring released?	Y	N	Y	N	N	N
Work on coursework					X	X
Work on labs				X		X
Play Elden Ring	X		X			

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	-	Y	N	N
Labs signed off?	**Y**	Y	**N**	N	Y	N
Elden Ring released?	**Y**	N	**Y**	N	N	N
Work on coursework					X	X
Work on labs				X		X
Play Elden Ring	**X**		**X**			

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	Y	N	N
Labs signed off?	-	Y	N	Y	N
Elden Ring released?	**Y**	N	N	N	N
Work on coursework				X	X
Work on labs			X		X
Play Elden Ring	**X**				

Decision Tables – “Don’t Care” Entries

Coursework completed?	-	Y	Y	N	N
Labs signed off?	-	Y	N	Y	N
Elden Ring released?	Y	N	N	N	N
Work on coursework				X	X
Work on labs			X		X
Play Elden Ring	X				

- Here, we take the actions “Play Elden Ring”, “Do not work on labs”, “Do not work on coursework” whenever the condition “Elden Ring Released?” is true.
- The actions do not depend on the values of the other conditions.

Rule Counts

Coursework completed?	-	Y	Y	N	N
Labs signed off?	-	Y	N	Y	N
Elden Ring released?	Y	N	N	N	N
Rule Count	4	1	1	1	1
Work on coursework				X	X
Work on labs			X		X
Play Elden Ring	X				

- "Rule Count" = number of rules encoded by a column with "don't-care"-entries.
- Summing up the rule counts for all columns must yield the size of the decision table that we obtain by expanding all columns with "don't-care" entries.

Rule Counts

Coursework completed?	-	Y	Y	N	N
Labs signed off?	-	Y	N	Y	N
Elden Ring released?	Y	N	N	N	N
Rule Count	4	1	1	1	1
Work on coursework				X	X
Work on labs			X		X
Play Elden Ring	X				

- Rule counts allow us to check that different columns don't contain the same rule.
- The expanded version of the above decision table has $2^3 = 8$ entries.
- The rule counts sum up to $4 + 1 + 1 + 1 + 1 = 8$.
- Hence, the table has the correct number of rules.

Decision Tables – An Inconsistent Table

Coursework completed?	-	N	-
Labs signed off?	-	-	N
Elden Ring released?	Y	-	-
Rule Count	4	4	4
Work on coursework		X	
Work on labs			X
Play Elden Ring	X		

- The expanded table has $2^3 = 8$ entries.
- The rule counts sum up to $4+4+4 = 12$.
- Hence, there are columns which encode competing rules.

Decision Tables – An Inconsistent Table

Coursework completed?	-	N	-
Labs signed off?	-	-	N
Elden Ring released?	Y	-	-
Rule Count	4	4	4
Work on coursework		X	
Work on labs			X
Play Elden Ring	X		

- Assume the labs are signed off, the coursework is not completed, and Elden Ring is released.
  - The first column says: play Elden ring, do not work on coursework, do not work on labs.
  - The second column says: do not play Elden ring, work on the coursework, do not work on labs.

Extended Decision Tables

Coursework completed?	-	-	-	N	N	N	Y	Y	Y
Which labs signed off?	Neither	One	Both	Neither	One	Both	Neither	One	Both
Elden Ring released?	Y	Y	Y	N	N	N	N	N	N
Rule Count	2	2	2	1	1	1	1	1	1
Work on coursework				X	X	X			
Work on labs	X			X			X	X	
Play Elden Ring	X	X	X						

- Here, the condition “Which labs signed off?” can take the values “Neither”, “One”, or “Both”.
- More generally:
  - In a *limited-entry* decision table every condition can only have two values (“Yes”/“No” or “True”/“False”).
  - In an *extended* decision table every condition can have finitely many values.

Decision Tables for Testing

- Let $f : X \to Y$ be a function.
- A boolean condition on the input domain defines a partition of X into two disjoint subsets.
- An "extended" condition with n values defines a partition of X into n disjoint subsets.
- Outputs or properties of outputs can be expressed as actions of a decision table.
- **Example.** The absolute value function $f : \mathbb{Q} \to \mathbb{Q}, x \mapsto |x|$.

Input non-negative?	Y	N
Return Input	X	
Return Minus Input		X

A Decision Table for the Triangle Problem

Triangle?	N	Y	Y	Y	Y	Y	Y	Y	Y
a = b?	-	Y	Y	Y	Y	N	N	N	N
a = c?	-	Y	Y	N	N	Y	Y	N	N
b = c?	-	Y	N	Y	N	Y	N	Y	N
Rule Count	8	1	1	1	1	1	1	1	1
Not a Triangle	X		?	?		?			
Scalene			?	?		?			X
Isosceles			?	?	X	?	X	X	
Equilateral		X	?	?		?			

- Problem: some rules are infeasible – the set of inputs they define is empty.
- To keep track of this, add a row that indicates that the combination of conditions is impossible.

A Decision Table for the Triangle Problem

Triangle?	N	Y	Y	Y	Y	Y	Y	Y	Y
a = b?	-	Y	Y	Y	Y	N	N	N	N
a = c?	-	Y	Y	N	N	Y	Y	N	N
b = c?	-	Y	N	Y	N	Y	N	Y	N
Rule Count	8	1	1	1	1	1	1	1	1
Not a Triangle	X								
Scalene									X
Isosceles					X		X	X	
Equilateral		X							
Impossible			X	X		X			

- The columns that aren't labelled "impossible" form a partition of the input domain.
- $\rightsquigarrow$ Employ Equivalence Class Testing to obtain a test suite: one test case per feasible column.

A Test Suite Derived from the Decision Table

Case	a	b	c	Expected Output
1	50	10	10	`Not a Triangle`
2	50	50	50	`Equilateral`
3	50	50	40	`Isosceles`
4	50	40	50	`Isosceles`
5	40	50	50	`Isosceles`
6	40	30	50	`Scalene`

- The expanded decision table has 16 columns, 3 of which are infeasible.
- We only have 6 test cases, because 8 columns of the expanded table have been combined into one.

A Decision Table for the Triangle Problem – Feasibility of Columns

Triangle?	N	Y	Y	Y	Y	Y	Y	Y	Y
$a = b$?	-	Y	Y	Y	Y	N	N	N	N
$a = c$?	-	Y	Y	N	N	Y	Y	N	N
$b = c$?	-	Y	N	Y	N	Y	N	Y	N
Rule Count	8	1	1	1	1	1	1	1	1
Not a Triangle	X								
Scalene									X
Isosceles					X		X	X	
Equilateral		X							
Impossible			X	X		X			

- Observe that the first column is feasible, but it contains feasible as well as infeasible rules.

A Decision Table for the Triangle Problem – The First Column Expanded

Triangle?	N	N	N	N	N	N	N	N
a = b?	N	N	N	N	Y	Y	Y	Y
a = c?	N	N	Y	Y	N	N	Y	Y
b = c?	N	Y	N	Y	N	Y	N	Y
Rule Count	1	1	1	1	1	1	1	Y
Not a Triangle	X	X	X		X			~~X~~
Scalene								
Isosceles								
Equilateral								
Impossible				X		X	X	X

- We can combine these columns into a single column, because all *feasible* columns have the same actions.

Decision Table Based Testing

1. Identify relevant characteristics of the problem's input – make them into conditions of a decision table.
2. Identify relevant characteristics of the problem's output – make them into actions of a decision table.
3. Describe the relation between inputs and outputs by a decision table.
4. Reduce the number of columns by using "don't care"-entries.
5. Create a test suite:
    - Add one test case per feasible rule/column.
    - Pick an input that satisfies the conditions of the rule.
    - The expected output is described by the actions.

A Refined Decision Table for the Triangle Problem

$a \geq b + c$?	Y	N	N	N	N	N	N	N	N	N	N
$b \geq a + c$?	-	Y	N	N	N	N	N	N	N	N	N
$c \geq a + b$?	-	-	Y	N	N	N	N	N	N	N	N
a = b?	-	-	-	Y	Y	Y	Y	N	N	N	N
a = c?	-	-	-	Y	Y	N	N	Y	Y	N	N
b = c?	-	-	-	Y	N	Y	N	Y	N	Y	N
Rule Count	32	16	8	1	1	1	1	1	1	1	1
Not a Triangle	X	X	X								
Scalene											X
Isosceles							X		X	X	
Equilateral				X							
Impossible					X	X		X			

A Test Suite for the Refined Decision Table

Case	a	b	c	Expected Output
1	20	13	5	Not a Triangle
2	13	20	5	Not a Triangle
3	13	5	20	Not a Triangle
4	23	23	23	Equilateral
5	42	42	23	Isosceles
6	42	23	42	Isosceles
7	42	42	23	Isosceles
8	2	3	5	Scalene

Discussion

- Decision tables yield a specific method for choosing equivalence classes.
- Rationale: Feasible columns in the decision table correspond to different (high-level) behaviours of the system.
- Adding a new condition at least doubles the size of the table.
- “don’t care”-entries constitute a mechanism for reducing the number of test cases.
- Columns can only be combined using “don’t care”-entries if their set of actions is completely independent from one condition.
- In general, the size of the decision table can grow quite fast in the number of conditions.

The Next Date Problem

The Next Date Problem

- **The Next Date Problem.**
- **Input:** Three integers `day`, `month`, `year`.
- **Output:**
  - `Out of Range` if any of the inequalities $1842 \le \texttt{year} \le 2042$, $1 \le \texttt{month} \le 12$, $1 \le \texttt{day} \le 31$ is violated.
  - `Invalid Date` if the given date is not a valid calendar date.
  - Otherwise: The calendar date of the day after the given date.
- **Examples:**
  - On input $\texttt{day} = 29$, $\texttt{month} = 2$, $\texttt{year} = 2022$ the output is `Invalid Date`.
  - On input $\texttt{day} = 28$, $\texttt{month} = 2$, $\texttt{year} = 2022$ the output is $\texttt{day} = 1$, $\texttt{month} = 3$, $\texttt{year} = 2022$.
  - On input $\texttt{day} = 28$, $\texttt{month} = 2$, $\texttt{year} = 2024$ the output is $\texttt{day} = 29$, $\texttt{month} = 2$, $\texttt{year} = 2024$.

Next Date – Conditions

- Interesting properties of months:
  - Even or odd.
  - In the range January to July or in the range August to December.
  - Equal to February.
  - Equal to December.
- Interesting days: 28, 29, 30, 31.
- Interesting years:
  - Leap years.
  - Non-leap-years.

Next Date – Conditions

- The above suggests to use an extended decision table with the following conditions:
  - Day $\in \{1-27, 28, 29, 30, 31\}$.
  - Month $\in$ {February, December, Rest}.
  - Month even? $\in$ {Yes, No}.
  - Month ≤ 7? $\in$ {Yes, No}.
  - Leap year? $\in$ {Yes, No}.

Next Date – Actions

- We can either output "`Invalid Date`"...
- ...or compute the next date.
- The latter involves a combination of the following:
  - Increment day.
  - Increment month.
  - Increment year.
  - Reset day to 1.
  - Reset month to 1.
- Thus, we have six possible actions: `invalid`, `day++`, `month++`, `year++`, `reset day`, `reset month`.
- Since the combination of our conditions contains infeasible combinations of conditions, we may need a seventh action, `impossible`.

Next Date – “Ordinary” days

Column	1	2	3	4	5
Day	1–27	28	29	28	29
Month	-	Rest	Rest	Dec	Dec
Month even?	-	-	-	-	-
Month $\leq$ 7?	-	-	-	-	-
Leap year?	-	-	-	-	-
Rule Count	24	8	8	8	8
`invalid`					
`day++`	X	X	X	X	X
`month++`					
`year++`					
`reset day`					
`reset month`					
`impossible`					

- Again, we have combined infeasible rules with feasible ones using “don’t care”-entries.
- Total number of rules encoded so far: 56.

Next Date – February

Column	6	7	8	9	10	11
Day	28	28	29	29	30	31
Month	Feb	Feb	Feb	Feb	Feb	Feb
Month even?	-	-	-	-	-	-
Month $\leq$ 7?	-	-	-	-	-	-
Leap year?	Y	N	Y	N	-	-
Rule Count	4	4	4	4	8	8
`invalid`				X	X	X
`day++`	X					
`month++`		X	X			
`year++`						
`reset day`		X	X			
`reset month`						
`impossible`						

- Total number of rules encoded so far: $56 + 32 = 88$.

Next Date – 30th and 31st

Column	12	13	14	15	16	17	18	19	20	21
Day	30	30	30	30	31	31	31	31	30	31
Month	Rst	Rst	Rst	Rst	Rst	Rst	Rst	Rst	Dec	Dec
Month even?	Y	Y	N	N	Y	Y	N	N	-	-
Month ≤ 7?	Y	N	Y	N	Y	N	Y	N	-	-
Leap year?	-	-	-	-	-	-	-	-	-	-
Rule Count	2	2	2	2	2	2	2	2	8	8
`invalid`					X			X		
`day++`		X	X						X	
`month++`	X			X		X	X			
`year++`										X
`reset day`	X			X		X	X			X
`reset month`										X
`impossible`										

- Total number of rules encoded so far: $56 + 32 + 32 = 120$.
- The fully expanded decision table has $5 \times 3 \times 2 \times 2 \times 2 = 120$ entries.
- Actual rule count matches the expected rule count.

A Test Suite Derived from the Decision Table

Case	day	month	year	Expected Output
1	23	06	1912	24/06/1912
2	28	04	1906	29/04/1906
3	29	03	1873	30/03/1873
4	28	12	1903	29/12/1903
5	29	12	1861	30/12/1861
6	28	02	2022	01/03/2022
7	28	02	2024	29/02/2024
8	29	02	2022	Invalid Date
9	29	02	2024	01/03/2024
10	30	02	2022	Invalid Date
11	31	02	2022	Invalid Date
12	30	06	2022	01/07/2022
13	30	08	2022	31/08/2022
14	30	07	2022	31/07/2022
15	30	09	2022	01/10/2022
16	31	06	2022	Invalid Date
17	31	08	2022	01/09/2022
18	31	07	2022	01/08/2022
19	31	09	2022	Invalid Date
20	30	12	2022	31/12/2022
21	31	12	2022	01/01/2023

Decision Table Based Testing – Summary

- Structured way of constructing Equivalence Class Testing test suites.
- Good for problems with prominent branching.
- Offer a (limited) mechanism for reducing complexity.
- Despite this, they do not (easily) scale up.