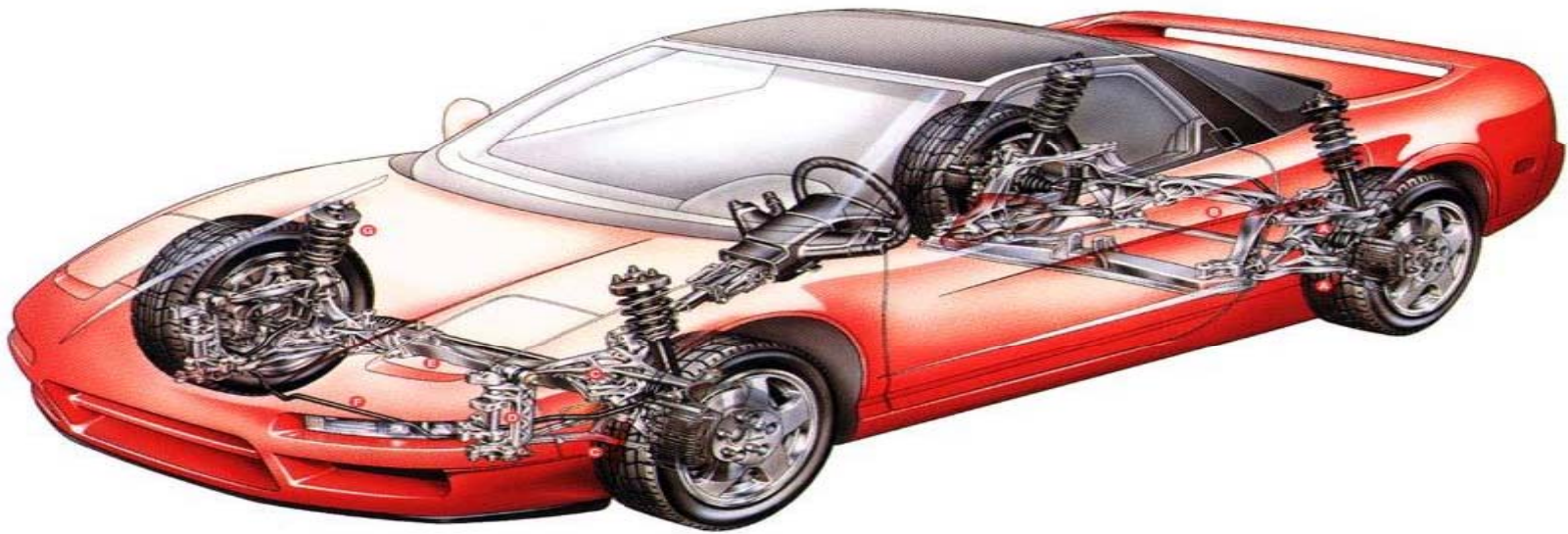


Advanced Software Testing

Testing Complex Logic with Domain Analysis



RBCS
TIME TESTED.
TESTING IMPROVED.
www.RBCS-US.com



Advanced Software Testing

- ❖ A series of webinars, this one excerpted from *Pragmatic Software Testing*, a book for test analysts and test engineers
- ❖ Equivalence partitioning and boundary value analysis are useful for testing input field validation
- ❖ Domain analysis allows us to scrutinize the inner workings of the business logic to uncover interesting test cases



Combinatorial Explosions

- In some cases, the number of possible test cases becomes very large due to the number of variables or factors
 - Factors that must interact properly:
 - E.g., forms or business logic: Different fields and operations, order of completion, etc.
 - Factors that shouldn't interact but might
 - E.g., application compatibility: Platforms, OSes, printers, cohabitating apps, networks, modems...
- Boundary value analysis alone won't suffice
- Domain analysis works for interacting factors



Domain Example: Frequent-Flyer

<i>Status Level</i>	<i>None</i>	<i>Silver</i>	<i>Gold</i>	<i>Platinum</i>
<i>Trip Bonus</i>	0%	25%	50%	100%
<i>Distance Traveled</i>	d	d	d	d
<i>Points Awarded</i>	d	1.25*d	1.5*d	2*d

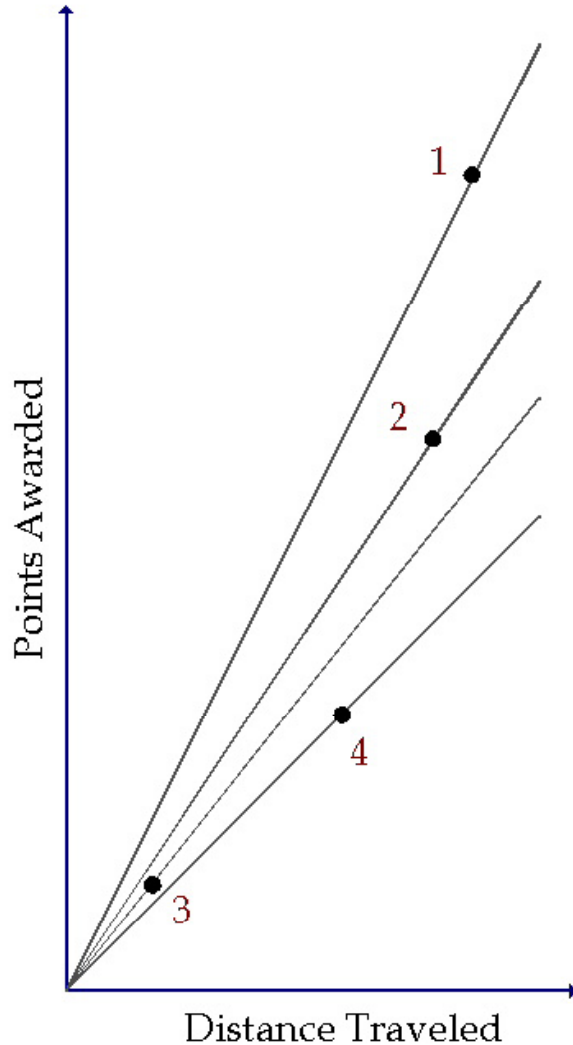
In this example, the points awarded for any trip is a function of the traveler's status level and the distance traveled.

$$\text{points} = (1 + \text{bonus}) * \text{distance}$$

Suppose our bug assumption is that the system will miscalculate the points awarded? We need to test at least one trip for each traveler status.



Frequent-Flyer Calculation Tests



These four points covers the four domains (i.e., the four calculations). Like the decision table technique, domain analysis focuses on central layers of the system architecture that implement the processing, not the user or data interface layers. Good system design principles says that, as much as possible, accepting valid inputs, displaying valid outputs, rejecting invalid inputs, and marking invalid outputs should reside in the user or data interface(s). You can test those interface using boundary values or equivalence partitioning.

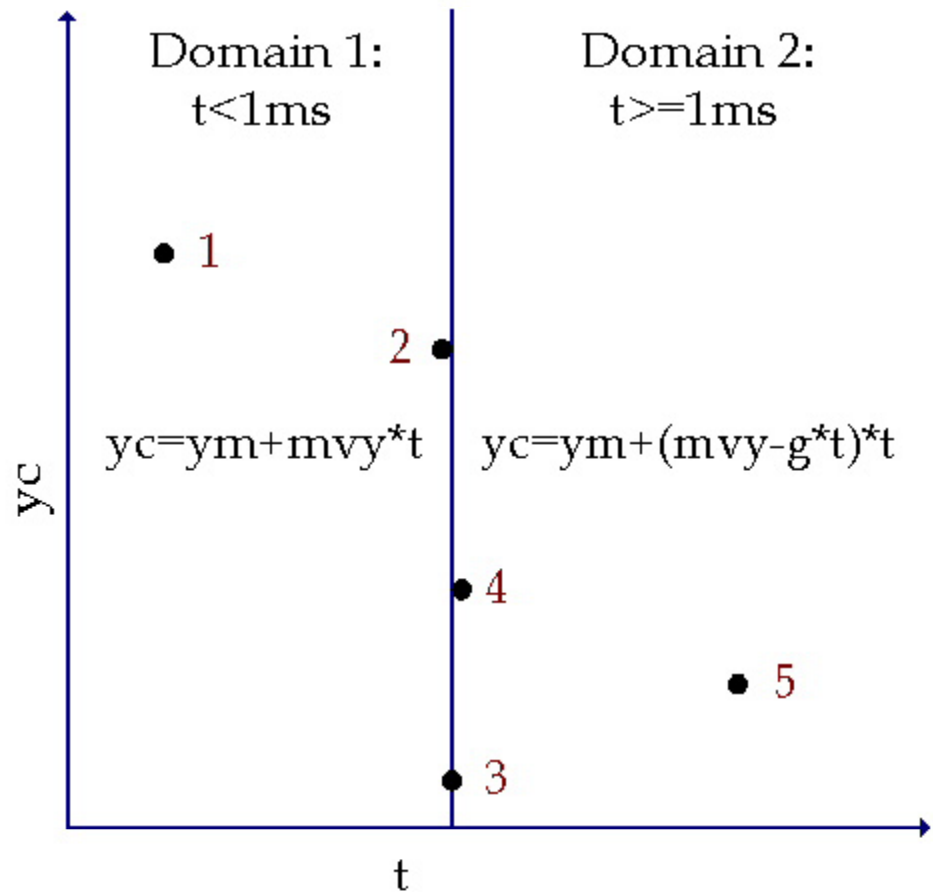


Possible Domain Test Values

<i>Name</i>	<i>Description</i>	<i>Frequent-Flyer Example</i>
<i>On</i>	A value on a domain boundary which may be inside or outside of the domain	1 (on platinum) 2 (on gold) 3 (on silver) 4 (on none)
<i>Off</i>	A value just off a domain boundary by the smallest recognizable amount and outside of the domain	1 (off gold) 2 (off platinum) 2 (off silver) 3 (off gold) 3 (off none) 4 (off silver)
<i>In</i>	A value inside the domain, not on or off	No such value possible
<i>Out</i>	A value outside the domain, not on or off	No such value possible



Aerospace Example



Description

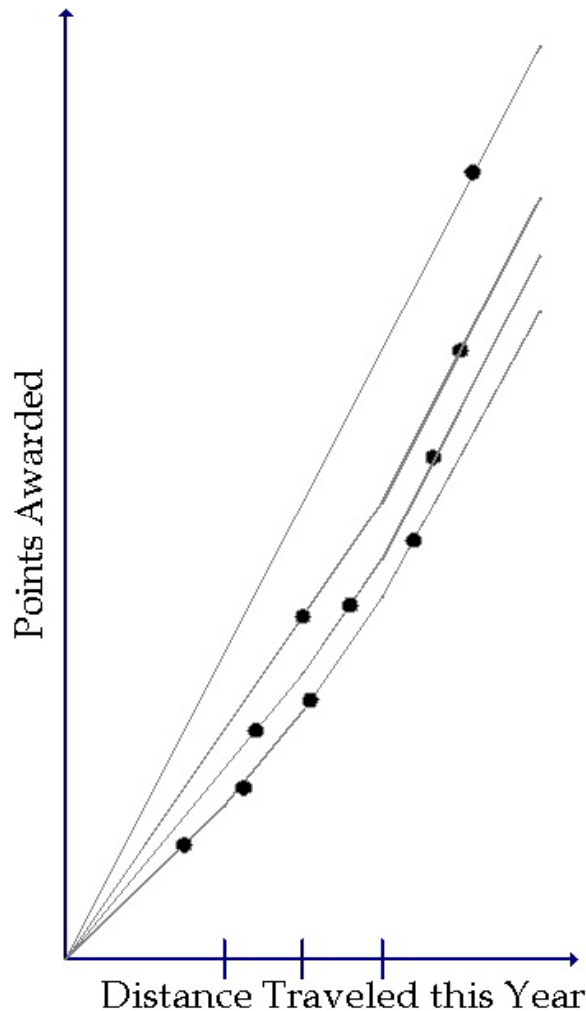
To estimate the current vertical position (y_c) of a large falling object (e.g., incoming missile) some time (t) after taking a velocity and position measurement, add to the measured vertical position (y_m) the product of the measured vertical velocity (m_{vy}) and the time. However, if the time has reached some threshold (here, 1 millisecond), adjust for the increased downward velocity caused by gravity ($g * t$).

Domain Analysis

	On	Off	In	Out
Dom 1	3	4	1	5
Dom 2	3	2	5	1



When Domain Rules Change



Description

Suppose we look at the total number of points awarded to a frequent-flyer for a whole year. A traveler's status is based on the total distance traveled in the current year or the total distance traveled in the past year. At 25,000, the status increases to silver. At 50,000, the status increases to gold. At 100,000, the status increases to platinum. Any distance traveled after the status changes accrues bonus points based on the new status, both in the current year or the next year.

Additional Domain Tests

Cover values in each rule segment. This adds one new test for travelers starting in the gold domain, two for travelers starting in the silver domain, and three for travelers starting with no status.



Domain Analysis Summary

- Combinations of factors or variables often create domains of operation that should be the same
- Domain tests
 - Cover at least on and off values
 - If an on value for one domain is an off value for another, there's no need for an additional value
 - Might also cover in and out values
 - Again, if an in value for one domain is an out value for another, no need for additional value
 - If the rules for a domain's boundary change, consider additional on, off, in, and out values
- Domains can be based on inputs, outputs, or internal data sets
- Domains can involve real numbers, integers, dates, sets, or other intrinsic or programmer-defined data types



Complex Domain Testing Example

- Let's look at an example, using a decision table
- Boundary combinations give us 208 tests
 - 4 yes/no boundaries, 13 speed boundaries
(-1, 0,1,10,11,20,21,25,26,99, 100,max,max+1)
 - $2 \times 2 \times 2 \times 2 \times 13 = 208$
- Perhaps we can use domain analysis
 - Actions are domains
 - Test one violator in each domain, one just outside each domain



Moving Violator Decision Table

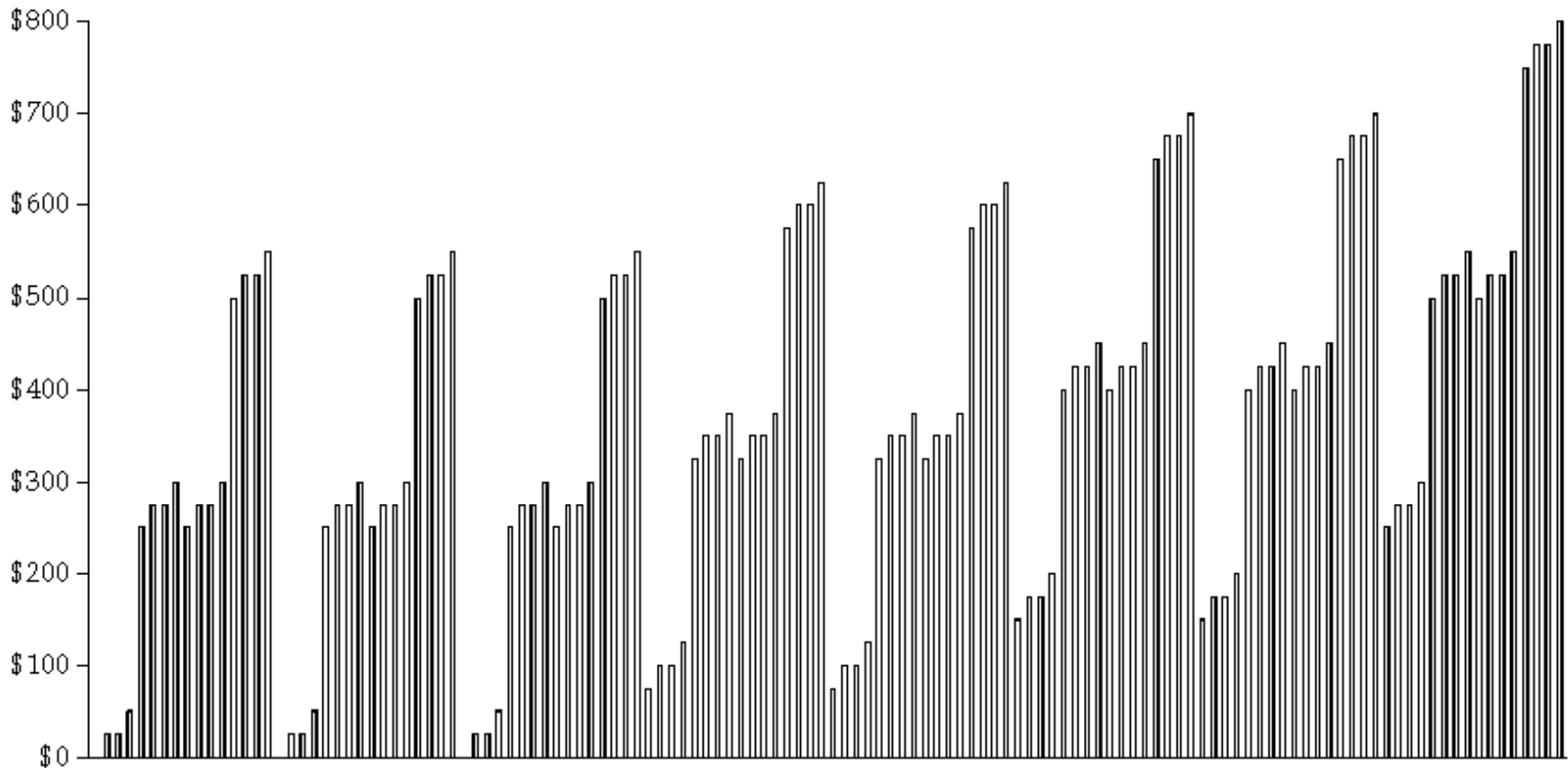
<u>Condition</u>	1	2	3	4	5	6	7	8
License OK	No	-	-	-	-	-	-	-
Warrant	-	Yes	-	-	-	-	-	-
Registration OK	-	-	No	-	-	-	-	-
Vehicle OK	-	-	-	No	-	-	-	-
Excess Speed	-	-	-	-	1-10	11-20	21-25	>25
<u>Action</u>								
Arrest	Yes	Yes	-	-	-	-	-	Yes
Fix-It Ticket	-	-	Yes	Yes	-	-	-	-
Warning	-	-	-	-	Yes	-	-	-
Fine	+250	+250	+25	+25	+0	+75	+150	+250

www.rbc-us.com



Lots of Possible Tests

Moving Violator Fine Calculations





Reducing Tests through Domain Analysis

- Assume bugs most likely to result from improper handling of three sets of data
 - Speeds, seven classes
(<0 (invalid), 0, 1-10, 11-20, 21-25, 26-max, $>max$ (invalid))
 - Proper ticket, five classes
(none, warn, fix-it, speed, arrest)
 - Arrest condition, two classes
(don't arrest, do arrest)
- Don't worry about interactions of all possible inputs or outputs at boundaries



A General Rule for Complex Domains

1. Pick “biggest” variable or factor
 - Do the domain analysis
 - Populate the condition(s) needed to cover on/off (and possibly in/out) values
2. Pick “next-biggest” variable
 - Do domain analysis
 - Figure out which on/off (and possibly in/out) values already covered, if any
 - Populate conditions (within existing tests, if possible) to cover those values
 - Add any addition tests needed.
3. Repeat step two until all the variables are handled



Final Domain Tests

	Test Case										
	1	2	3	4	5	6	7	8	9	10	11
Condition											
Lic OK	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No
Warrant	No	No	No	No	No	No	Yes	No	Yes	No	Yes
Req OK	Yes	No	Yes	Yes	Yes	No	Yes	No	No	Yes	No
Veh OK	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	Yes	No
Excss Spd	-1	0	1	10	11	20	21	25	26	Max	Max+1
Expected Result											
Fine	0	25	0	250	75	125	400	200	800	250	0
Ticket	N	F	W	WA	S	SF	SA	SF	SFA	S	N
Arrest	No	No	No	Yes	No	No	Yes	No	Yes	Yes	No
Reject	Yes	No	No	No	No	No	No	No	No	No	Yes
Domain Analysis											
Spd ON	I	0	1-10	1-10	11-20	11-20	21-25	21-25	>25	>25	I
OFF	0	I,1-10	0	11-20	1-10	21-25	11-20	>25	21-25	I	>25
Tck ON	N	F	W	WA	S	SF	SA	SF	SFA	S	N
OFF	WFA	NWSA	NSFA	NSF	NWFA	NWA	NWF	NWA	NW	NWFA	WSFA
Arr ON	No	No	No	Yes	No	No	Yes	No	Yes	Yes	No
OFF	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	No	Yes



Conclusion

- In this webinar, we've seen how to apply domain analysis to the testing of complex logic
- In our previous webinars, we looked at decision tables, state based methods, use cases, and pairwise testing
- Domain analysis allows us to use knowledge of how the system performs its calculations
- With these five techniques, you can perform a range of business logic testing



RBCS

TIME TESTED. TESTING IMPROVED.

...*Contact RBCS*

For over a dozen years, RBCS has delivered services in consulting, outsourcing and training for software and hardware testing. Employing the industry's most experienced and recognized consultants, RBCS conducts product testing, builds and improves testing groups and hires testing staff for hundreds of clients worldwide. Ranging from Fortune 20 companies to start-ups, RBCS clients save time and money through improved product development, decreased tech support calls, improved corporate reputation and more. To learn more about RBCS, visit www.rbc-us.com.

Address: RBCS, Inc.
31520 Beck Road
Bulverde, TX 78163-3911
USA

Phone: +1 (830) 438-4830

Fax: +1 (830) 438-4831

E-mail: info@rbc-us.com

Web: www.rbc-us.com