For Updated version, please click on this http://ocw.ump.edu.my

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

BCS3323 – Software Testing and Maintenance

Combinatorial Testing

Editors
Prof. Dr. Kamal Z. Zamli
Dr. AbdulRahman A. Alsewari
Faculty of Computer Systems & Software Engineering
alswari@ump.edu.my
Authors

Chapter 3.3

- Aims is to discover
  - The Combinatorial testing need
  - The efficiency of Combinatorial testing
  - The type of Combinatorial testing
  - Different Combinatorial testing approaches
- Expected Outcomes
  - Students can use Combinatorial testing to design test List
  - Differentiate between all Combinatorial testing approaches

- References
  - ISTQB
  - MSTB/GTB
  - http://www.softwaretestingclass.com/software-testing-tools-list/
  - http://www.softwaretestinggenius.com/articalDetails.php?qry=572#commentsList

http://www.softwaretestinggenius.com/articalDetails.php?qry=572#commentsList

Is Combinatorial Testing Important?

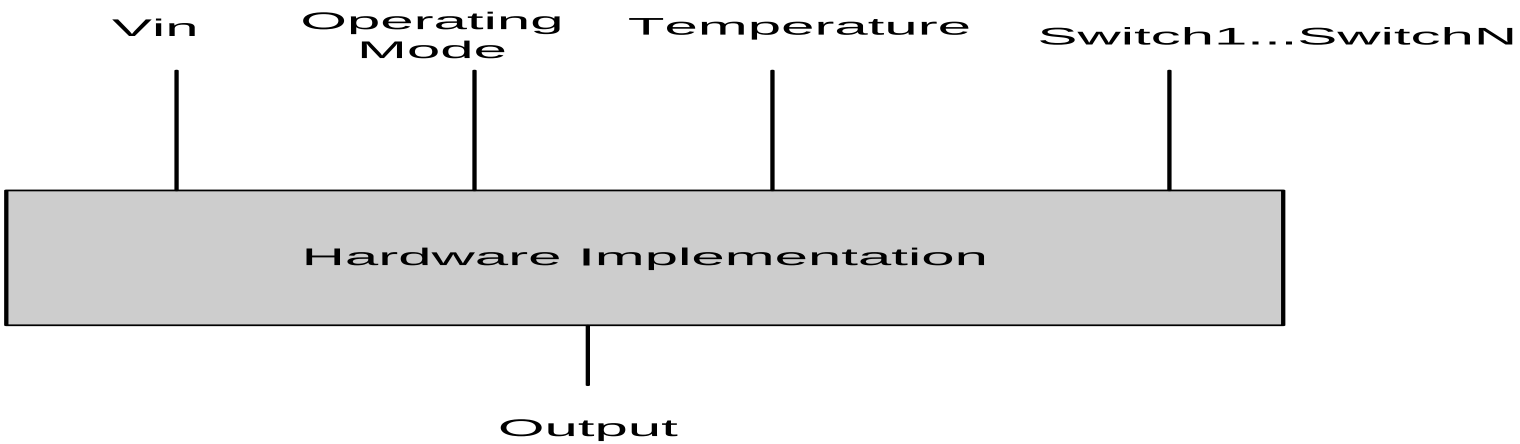
How much to test?

- 34 hardware switches = 2^{34} = 1.7 x 10^{10} possible input configurations = 1.7 x 10^{10} tests.
- Is it possible to test every possible combinations of inputs?

Combinatorial Explosion Problem in Hardware Product Testing

PRODUCT TESTING

Vin = {-5,0,+5} Volts
Operating Mode = {Reverse, Forward}
Temperature = {Low, Medium, High}
Switch1.. SwitchN = {On, Off}



Let us assume N=5
Exhaustive possibilities = $3x2x3x2^5$= 576

What if we need to test at least 100 products a day?

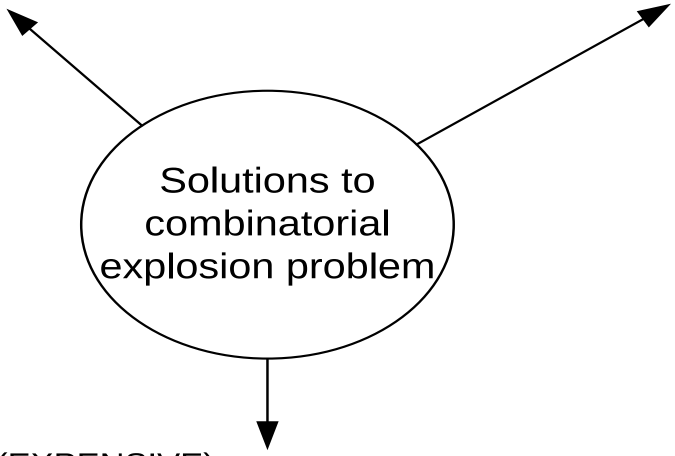
576x100 tests per day!

If we are late, we will lose our business to our competitors......

Approach to address combinatorial explosion problem

Random testing (UNFAIR DISTRIBUTION)
- Non systematic
- Tend to stress on certain configurations

Manual testing process (IMPOSSIBLE)
- -Test engineers need to write write test cases
- -Test engineers need to write test drivers for each test cases
- -Test engineers need to execute the test driver



Parallel Testing (EXPENSIVE)
- -Software applications are getting more complex and larger in terms of line of codes
- -Test engineers need to test more and more codes
- -Many more combinations of input parameters and system conditions need to be tested

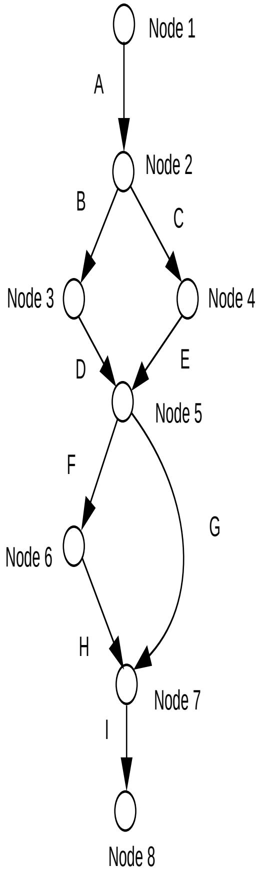
Given limited time and resources, what are the optimum sets of smaller test cases to be used for testing?

On Sampling Strategy

- As highlighted earlier, testing of every possibilities is impractical. There is a need for a sampling strategy.
- Is the idea of sampling strategy for software/hardware testing new?
- No!!!
- Some of existing sampling strategies: Equivalent Partitioning, Boundary Value, Robustness, Cause and Effect etc.
- Existing sampling strategies does not sufficiently cater for fault possibilities due to interaction

```
void ValveControl (int pressure, int temperature)
{
  if (pressure >=100)
   {
     OpenTheValve();
     printf ("Valve opened\n");
   }
  else
   {
    CloseTheValve();
    printf ("Valve closed\n");
   }

  if (temperature > 27)
   {
    EnableCoolingCoil();
    printf ("Cooling coil enabled\n");
   }
}
```



Motivating Example – Interaction Testing

- Interaction testing (a.k.a. t-way testing) complements existing sampling strategies.
- Consider a system with 4 parameters with each with 2 values. At full interaction strength, t =4, correspond to all combinations

Parameters are A, B, C, D

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2

Values

Full Interaction Strength, t = 4

	Input Variables			
	A	B	C	D
Base Values	a1	b1	c1	d1
	a2	b2	c2	d2
Exhaustive Combinations	a1	b1	c1	d1
	a1	b1	c1	d2
	a1	b1	c2	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c1	d2
	a1	b2	c2	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c1	d2
	a2	b1	c2	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c1	d2
	a2	b2	c2	d1
	a2	b2	c2	d2

- What if we relax or compromise the interaction strength?

What if we consider all possible 3 way interaction: ABC, ABD, ACD, and BCD

Combining each 3-way combinations

	Input Variables			
Base Values	**A**	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for ABC, t=3	a1	b1	c1	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1

\+

	Input Variables			
Base Values	**A**	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for ACD, t=3	a1	b1	c1	d1
	a1	b2	c1	d2
	a1	b1	c2	d1
	a1	b1	c2	d2
	a2	b1	c1	d1
	a2	b1	c1	d2
	a2	b2	c2	d1
	a2	b2	c2	d2

	Input Variables			
Base Values	**A**	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for ABD, t=3	a1	b1	c1	d1
	a1	b1	c1	d2
	a1	b2	c2	d1
	a1	b2	c1	d2
	a2	b1	c1	d1
	a2	b1	c1	d2
	a2	b2	c2	d1
	a2	b2	c2	d2

	Input Variables			
Base Values	**A**	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for BCD, t=3	a1	b1	c1	d1
	a2	b1	c1	d2
	a1	b1	c2	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a1	b2	c1	d2
	a2	b2	c2	d1
	a2	b2	c2	d2

Removing repetitions

=

All 3-way combinations

	Input Variables			
Base Values	**A**	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values with t=3	a1	b1	c1	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1
	a1	b1	c1	d2
	a1	b2	c2	d1
	a1	b2	c1	d2
	a2	b1	c1	d2
	a2	b2	c2	d2

Total test data = 13

The reduction is now from 16 to 13………at t=3

Demonstration of Correctness

All 3 Way Pairs

Interaction	3 Way pairs	Occurrences
ABC	a1, b1,c1	2
	a1,b1,c2	1
	a1,b2,c1	2
	a1,b2,c2	2
	a2, b1,c1	2
	a2,b1,c2	1
	a2,b2,c1	1
	a2,b2,c2	2
ABD	a1, b1,d1	1
	a1,b1,d2	2
	a1,b2,d1	2
	a1,b2,d2	2
	a2, b1,d1	1
	a2,b1,d2	2
	a2,b2,d1	2
	a2,b2,d2	1
BCD	b1,c1,d1	2
	b1,c1,d2	2
	b1,c2,d1	1
	b1,c2,d2	2
	b2, c1,d1	2
	b2,c1,d2	1
	b2,c2,d1	2
	b2,c2,d2	2
ACD	a1,c1,d1	2
	a1,c1,d2	2
	a1,c2,d1	1
	a1,c2,d2	2
	a2, c1,d1	2
	a2,c1,d2	1
	a2,c2,d1	1
	a2,c2,d2	2

All 3-Way Interactions

Test suite based on coverage strength t=3

	Input Variables			
	A	B	C	D
Base Values	a1	b1	c1	d1
	a2	b2	c2	d2
All T-way Combinatorial Values	a1	b1	c1	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1
	a1	b1	c1	d2
	a1	b2	c2	d1
	a1	b2	c1	d2
	a2	b1	c1	d2
	a2	b2	c2	d2

All 4 Way (i.e. t= 4, full strength) => 16

All 3 Way (i.e. t=3) => 13

Hence, the reduction => 3/16 *100 %

= 18.75 %

Notice that all the 3 way pairs have been covered by at least one test

What if we relaxed the interaction to t=2?

Combining each 2-way combinations

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for AB, t=2	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for AC, t=2	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for AD, t=2	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2

+

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for BC, t=2	a1	b1	c1	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d2

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for BD, t=2	a1	b1	c1	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d2

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial values for CD, t=2	a1	b1	c1	d1
	a1	b2	c1	d2
	a2	b1	c2	d1
	a2	b2	c2	d2

Removing repetitions

=

All 2-way combinations

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values with t=2	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2
	a2	b1	c1	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a1	b2	c1	d2
	a2	b1	c2	d1

Total test data = 9

More reduction now from 16 to 9……..at t=2

Why not Pairwise t=2?

- Pairwise => gives the most reduction
  - 4 parameter, 3 values
    - Exhaustive = 3^4 = 3x3x3x3= 81 test cases
    - Pairwise = 9 test cases
  - 13 parameter, 3 values
    - Exhaustive = 3^{13} = 3x3x3x…. = 1,594,323 test cases
    - Pairwise = 15 test cases
  - 20 parameter, 10 values
    - Exhaustive= 10^{20} = 10x10x10x…. = a lot of test cases
    - Pairwise = 180 test cases

Pairwise saves time and costs…………..

Why not Pairwise a.k.a. t=2?

- In some system where there are hardly any interaction of variables, pairwise testing can generates a very efficient test case that are able to locate between 75%-85% of faults.
- While such conclusion may be true for some system, it cannot be generalized to all software/hardware system faults (i.e. especially when there are significant interactions between variables)….

What if we do cumulative t=2 & t=3?

Cummulative t=3 & t=2

All 2-way combinations

Base Values	**Input Variables**			
	A	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values with t=2	~~a1~~	~~b1~~	~~c1~~	~~d1~~
	~~a1~~	~~b2~~	~~c2~~	~~d2~~
	~~a2~~	~~b1~~	~~c1~~	~~d1~~
	~~a2~~	~~b2~~	~~c2~~	~~d2~~
	~~a2~~	~~b1~~	~~c1~~	~~d1~~
	~~a1~~	~~b1~~	~~c2~~	~~d2~~
	~~a2~~	~~b2~~	~~c1~~	~~d1~~
	~~a1~~	~~b2~~	~~c1~~	~~d2~~
	a2	b1	c2	d1

Total test data = 9

+

All 3-way combinations

Base Values	**Input Variables**			
	A	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values with t=3	a1	b1	c1	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1
	a1	b1	c1	d2
	a1	b2	c2	d1
	a1	b2	c1	d2
	a2	b1	c1	d2
	a2	b2	c2	d2

Total test data = 13

Removing repetitions

=

Base Values	**Input Variables**			
	A	**B**	**C**	**D**
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values for Cummulative t=2 & t=3	a1	b1	c1	d1
	a1	b1	c2	d2
	a1	b2	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1
	a1	b1	c1	d2
	a1	b2	c2	d1
	a1	b2	c1	d2
	a2	b1	c1	d2
	a2	b2	c2	d2
	a2	b1	c2	d1

Total test data = 14

Reduction is now from 16 to 14………for cumulative t=2 & t=3…..

How about if we consider variable strength?
In real life, interaction is not always uniform

t=2

t=3

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Combinatorial Values with t=2	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2
	a2	b1	c1	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a1	b2	c1	d2
	a2	b1	c2	d1

Total test data = 9

+

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Variable Strength Combinatorial Values BCD with t=3	~~a1~~	~~b1~~	~~c1~~	~~d1~~
	a1	b1	c1	d2
	~~a2~~	~~b1~~	~~c2~~	~~d1~~
	a2	b1	c2	d2
	a2	b2	c1	d1
	~~a1~~	~~b2~~	~~c1~~	~~d2~~
	a2	b2	c2	d1
	~~a1~~	~~b2~~	~~c2~~	~~d2~~

Total test data = 8

Removing repetitions

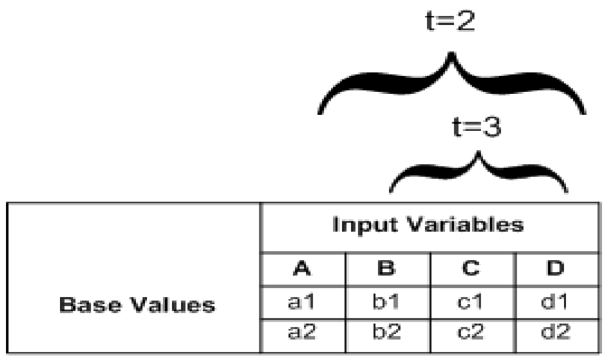
=

Base Values	Input Variables			
	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2
Variable Strength Combinatorial Values	a1	b1	c1	d1
	a1	b2	c2	d2
	a2	b1	c1	d1
	a2	b2	c2	d2
	a2	b1	c1	d1
	a1	b1	c2	d2
	a2	b2	c1	d1
	a1	b2	c1	d2
	a2	b1	c2	d1
	a1	b1	c1	d2
	a2	b1	c2	d2
	a2	b2	c1	d1
	a2	b2	c2	d1

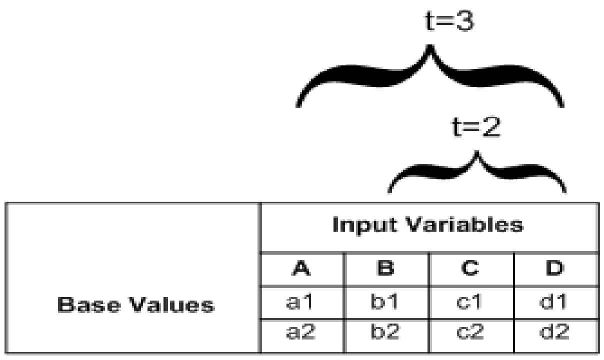
Total test data = 13

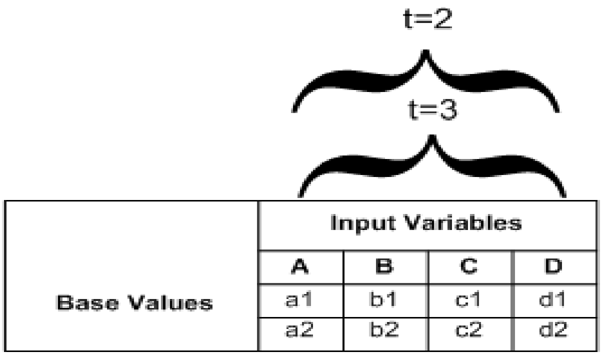
Reduction is now from 16 to 13………

Something to ponder?



VS

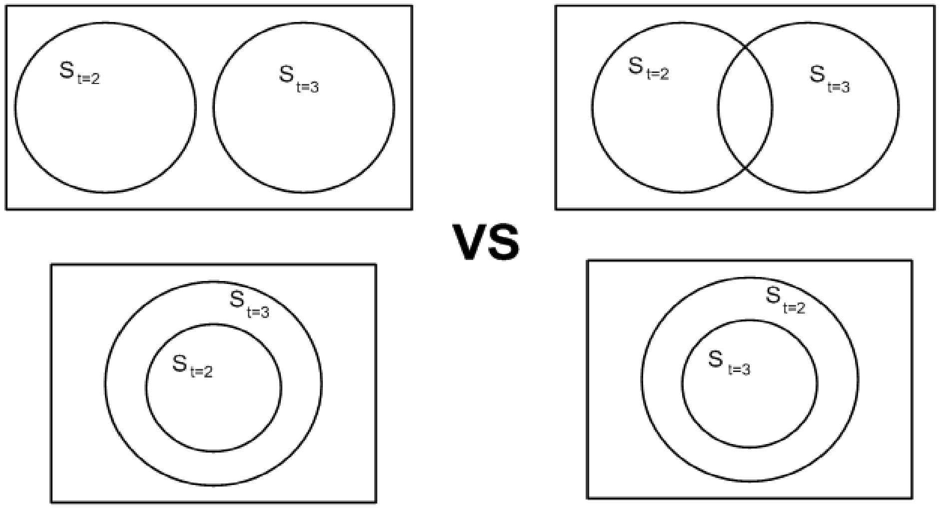




- Which is the good choice for applying variable strength?

Something to ponder?

Venn Diagram



- Le t $S_{t=2}$ = {the set of test cases for t=2} and $S_{t=3}$ = {the set of test cases for t=3}
- Is $S_{t=2}$ is the subset of $S_{t=3}$?

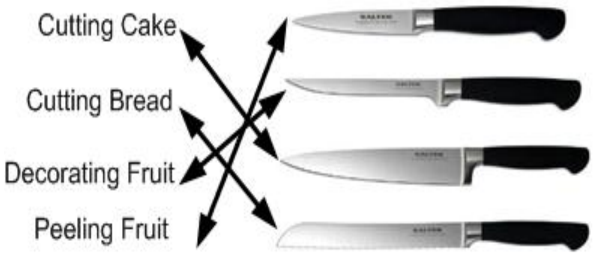
Observation on our application of t-way testing strategies – contd.

- There are four possibilities for applying t-way strategies for test reduction
  - Uniform Strength Interaction
  - Cumulative Strength Interaction
  - Variable Strength Interaction

- Which is the most effective strategy for testing?
  - It depends on circumstances. Refer to the chef’s knife analogy.

Recommendation??

- Which is the most effective strategy for testing?
  - Uniform strength – We have no idea on the system under test…We have no knowledge on the effects of inputs on output(s).
  - Cumulative strength/Variable strength – Based on our experience, we have some knowledge on the contribution of particular parameters. Hence, we want to relax interaction on overall strength but focus on specific interaction. We have no knowledge on the effects of inputs on output(s).
  - The choice of interaction strength (t)???
    - Literature suggests 2<t<6.. See www.pairwise.org
    - We can debate this?



Existing Tools and Strategies

- Some Existing strategies used Optimization Algorithms:
  - ACO
  - PSO
  - HS
  - GA
  - SA
  - HHH
  - Greedy Algorithm
  - FP
  - FF

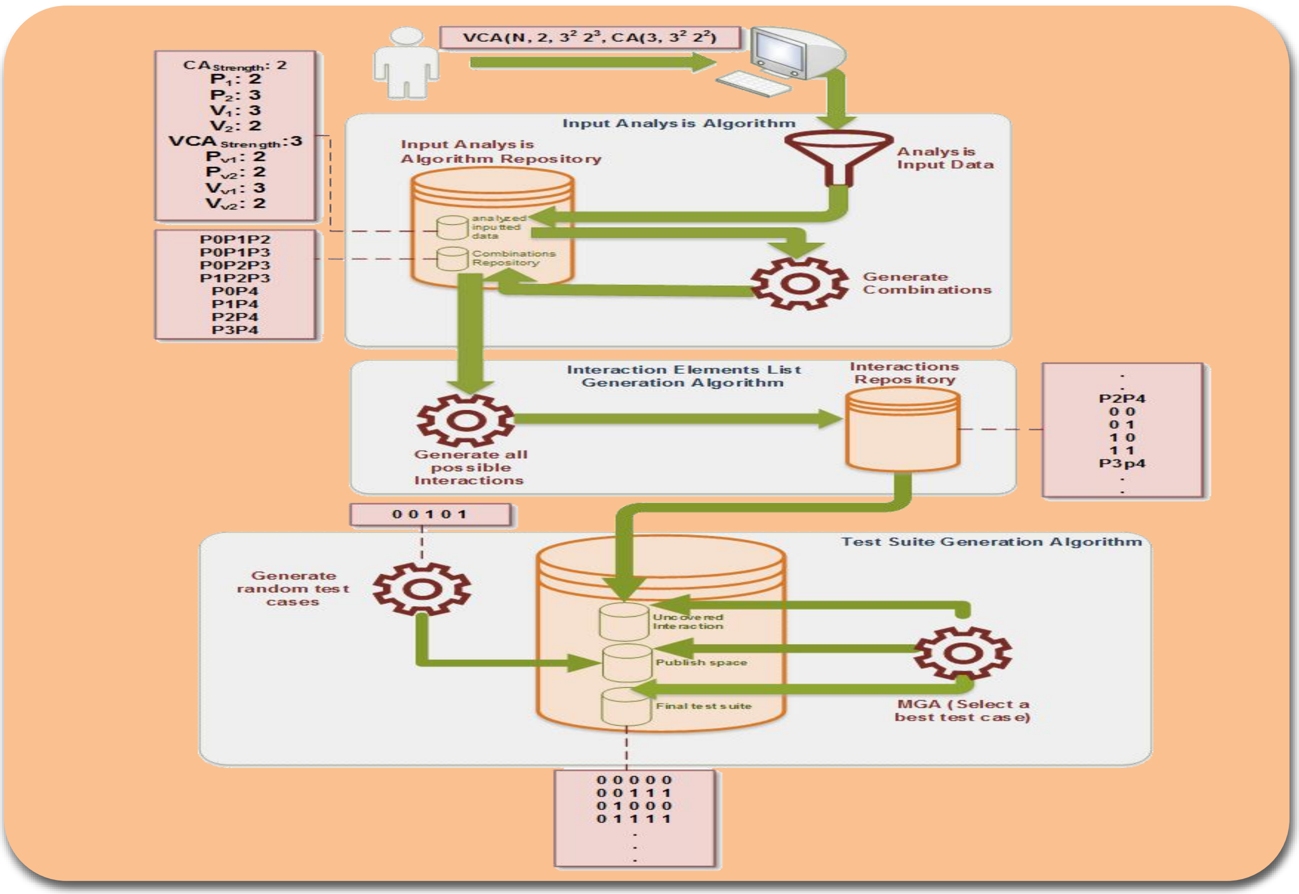
Only published in the research papers

- Source: http://www.pairwise.org/tools.asp

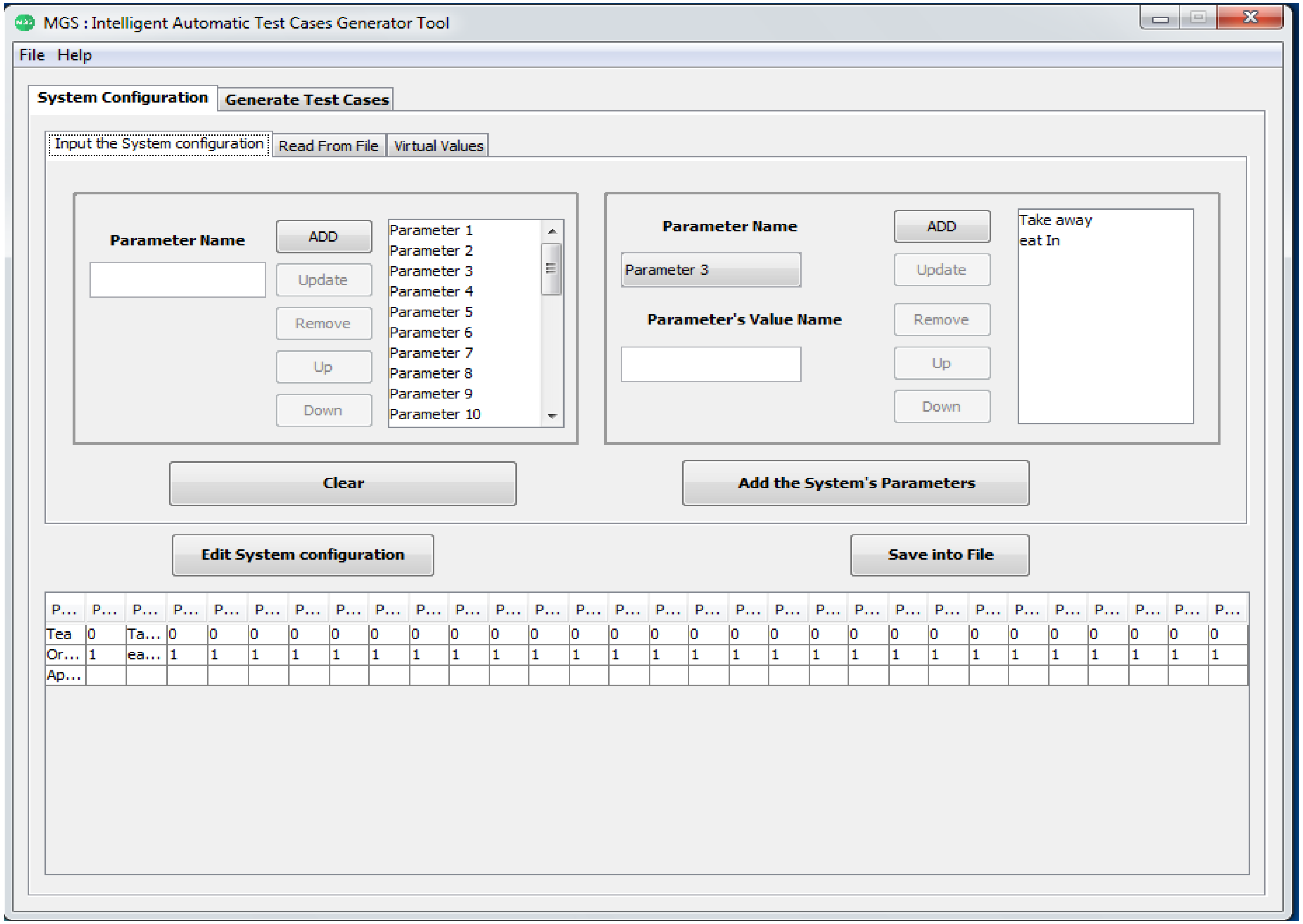
Available Tools

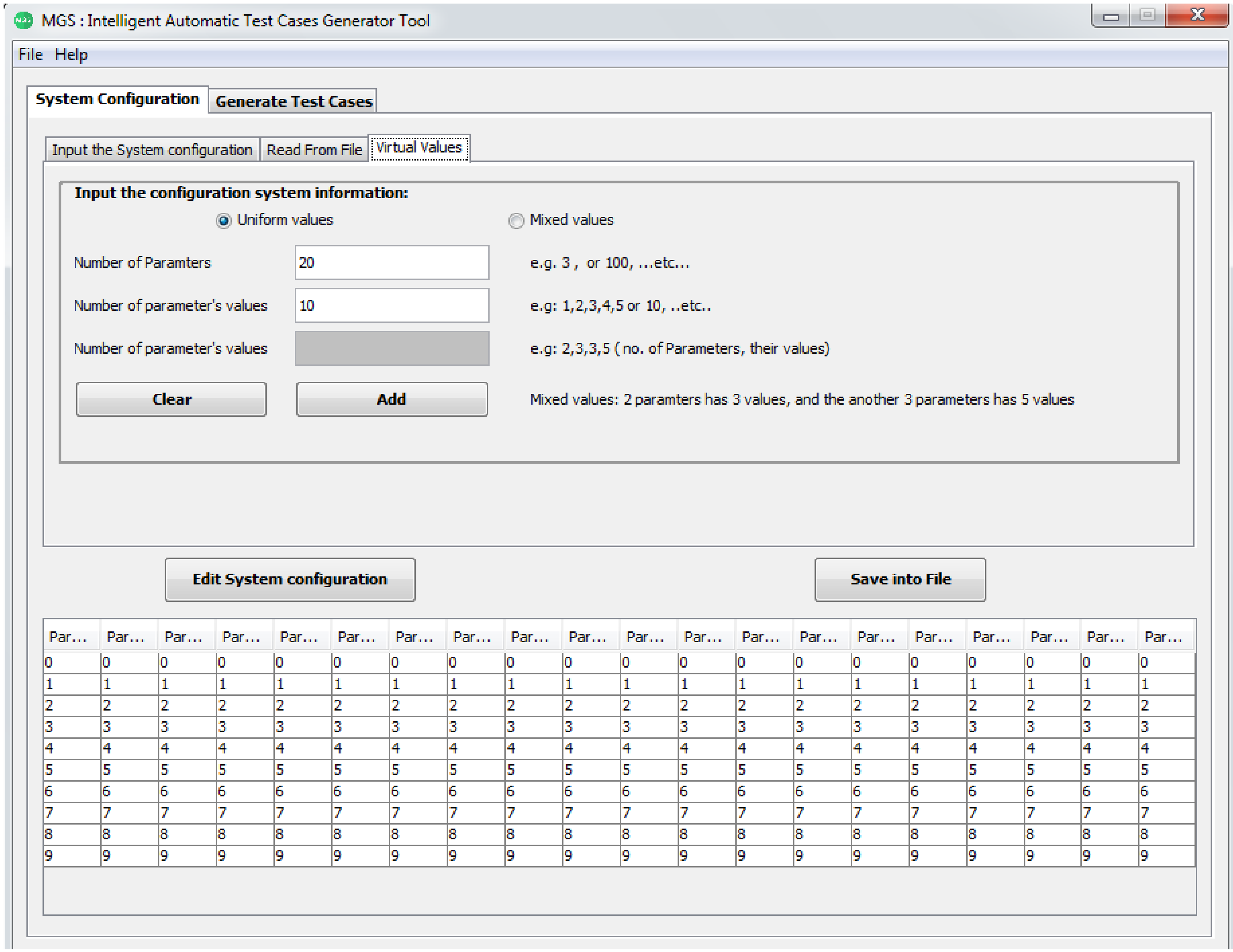
1. CATS (Constrained Array Test System) *)	[Sherwood] Bell Labs.	
2. OATS (Orthogonal Array Test System) *)	[Phadke] ATT	
3. AETG	Telecordia	Web-based, commercial
4. IPO (PairTest) *)	[Tai/Lei]	
5. TConfig	[Williams]	Java-applet
6. TCG (Test Case Generator) *)	NASA	
7. AllPairs	Satisfice	Perl script, free, GPL
8. Pro-Test	SigmaZone	GUI, commercial
9. CTS (Combinatorial Test Services)	IBM	Free for non-commercial use
10. Jenny	[Jenkins]	Command-line, free, public-domain
11. ReduceArray2	STSC, U.S. Air Force	Spreadsheet-based, free
12. TestCover	Testcover.com	Web-based, commercial
13. DDA *)	[Colburn/Cohen/Turban]	
14. Test Vector Generator		GUI, free
15. OA1	k sharp technology	
16. CTE-XL	Berner & Mattner	GUI, free

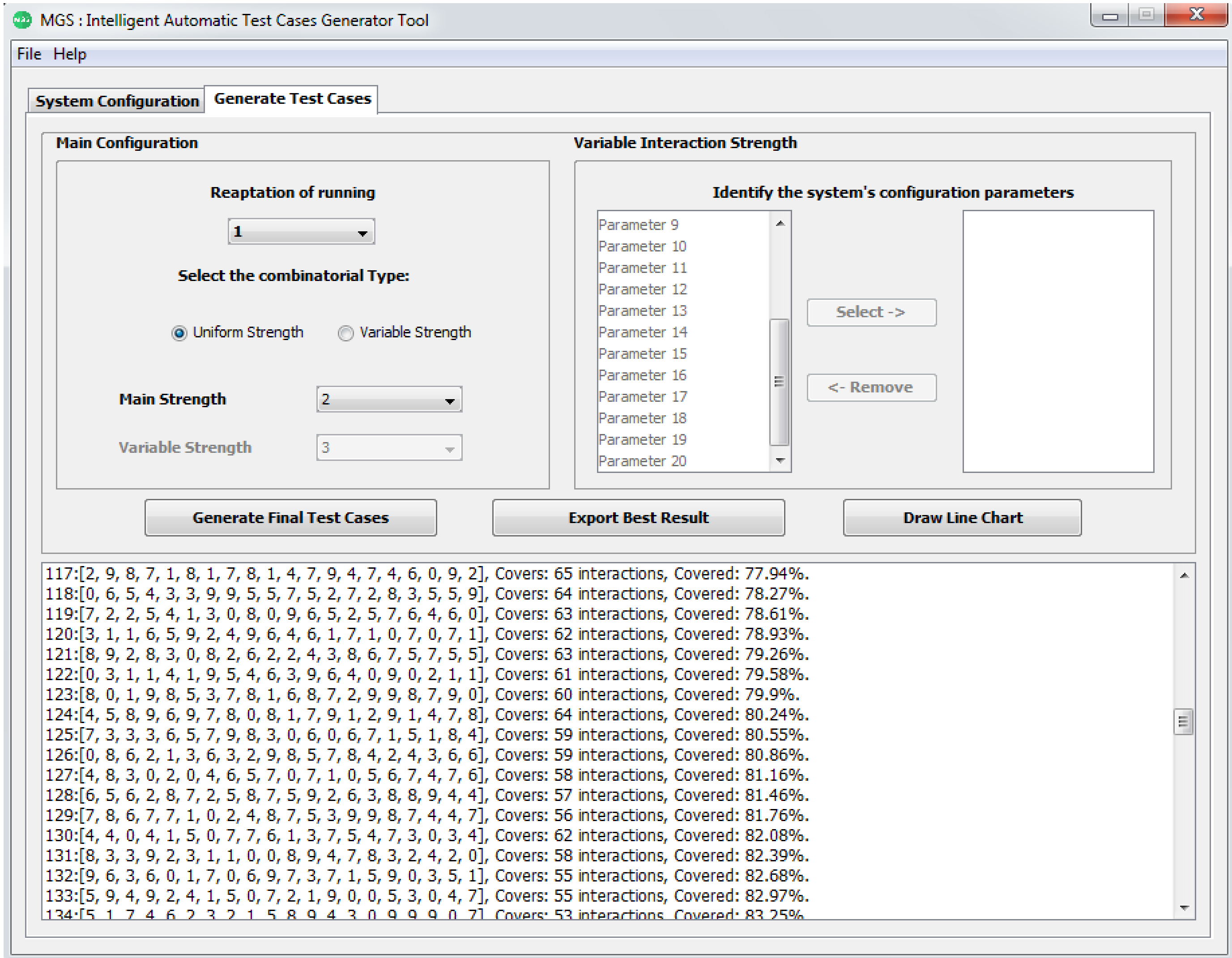
Test list generation based on Optimization Algorithms

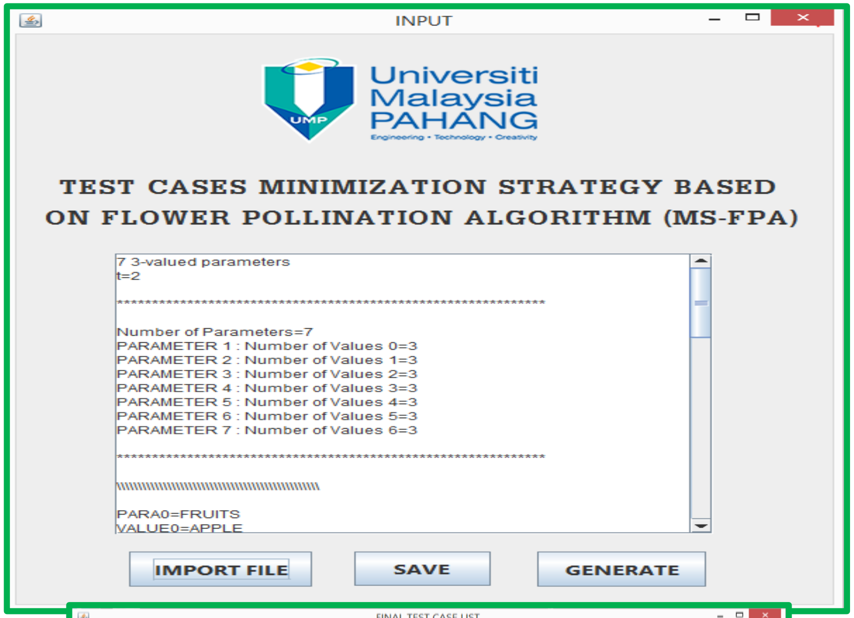


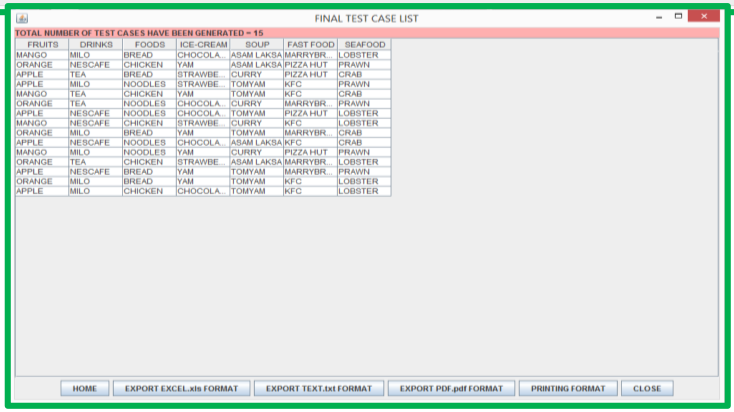
MGS: Intelligent Automatic Test Cases Generator




MGS : Intelligent Automatic Test Cases Generator Tool
File Help
System Configuration
Generate Test Cases
Input the System configuration
Read From File
Virtual Values
Input the configuration system information:
Uniform values
Mixed values
Number of Paramters
20
e.g. 3 , or 100, ...etc...
Number of parameter's values
10
e.g: 1,2,3,4,5 or 10, ..etc..
Number of parameter's values
e.g: 2,3,3,5 (no. of Parameters, their values)
Clear
Add
Mixed values: 2 paramters has 3 values, and the another 3 parameters has 5 values
Edit System configuration
Save into File
Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par... Par...
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9


MGS : Intelligent Automatic Test Cases Generator Tool
File Help
System Configuration
Generate Test Cases
Main Configuration
Reaptation of running
1
Select the combinatorial Type:
Uniform Strength
Variable Strength
Main Strength
2
Variable Strength
3
Variable Interaction Strength
Identify the system's configuration parameters
Parameter 9
Parameter 10
Parameter 11
Parameter 12
Parameter 13
Parameter 14
Parameter 15
Parameter 16
Parameter 17
Parameter 18
Parameter 19
Parameter 20
Select ->
<- Remove
Generate Final Test Cases
Export Best Result
Draw Line Chart
117:[2, 9, 8, 7, 1, 8, 1, 7, 8, 1, 4, 7, 9, 4, 7, 4, 6, 0, 9, 2], Covers: 65 interactions, Covered: 77.94%.
118:[0, 6, 5, 4, 3, 3, 9, 9, 5, 5, 7, 5, 2, 7, 2, 8, 3, 5, 5, 9], Covers: 64 interactions, Covered: 78.27%.
119:[7, 2, 2, 5, 4, 1, 3, 0, 8, 0, 9, 6, 5, 2, 5, 7, 6, 4, 6, 0], Covers: 63 interactions, Covered: 78.61%.
120:[3, 1, 1, 6, 5, 9, 2, 4, 9, 6, 4, 6, 1, 7, 1, 0, 7, 0, 7, 1], Covers: 62 interactions, Covered: 78.93%.
121:[8, 9, 2, 8, 3, 0, 8, 2, 6, 2, 2, 4, 3, 8, 6, 7, 5, 7, 5, 5], Covers: 63 interactions, Covered: 79.26%.
122:[0, 3, 1, 1, 4, 1, 9, 5, 4, 6, 3, 9, 6, 4, 0, 9, 0, 2, 1, 1], Covers: 61 interactions, Covered: 79.58%.
123:[8, 0, 1, 9, 8, 5, 3, 7, 8, 1, 6, 8, 7, 2, 9, 9, 8, 7, 9, 0], Covers: 60 interactions, Covered: 79.9%.
124:[4, 5, 8, 9, 6, 9, 7, 8, 0, 8, 1, 7, 9, 1, 2, 9, 1, 4, 7, 8], Covers: 64 interactions, Covered: 80.24%.
125:[7, 3, 3, 3, 6, 5, 7, 9, 8, 3, 0, 6, 0, 6, 7, 1, 5, 1, 8, 4], Covers: 59 interactions, Covered: 80.55%.
126:[0, 8, 6, 2, 1, 3, 6, 3, 2, 9, 8, 5, 7, 8, 4, 2, 4, 3, 6, 6], Covers: 59 interactions, Covered: 80.86%.
127:[4, 8, 3, 0, 2, 0, 4, 6, 5, 7, 0, 7, 1, 0, 5, 6, 7, 4, 7, 6], Covers: 58 interactions, Covered: 81.16%.
128:[6, 5, 6, 2, 8, 7, 2, 5, 8, 7, 5, 9, 2, 6, 3, 8, 8, 9, 4, 4], Covers: 57 interactions, Covered: 81.46%.
129:[7, 8, 6, 7, 7, 1, 0, 2, 4, 8, 7, 5, 3, 9, 9, 8, 7, 4, 4, 7], Covers: 56 interactions, Covered: 81.76%.
130:[4, 4, 0, 4, 1, 5, 0, 7, 7, 6, 1, 3, 7, 5, 4, 7, 3, 0, 3, 4], Covers: 62 interactions, Covered: 82.08%.
131:[8, 3, 3, 9, 2, 3, 1, 1, 0, 0, 8, 9, 4, 7, 8, 3, 2, 4, 2, 0], Covers: 58 interactions, Covered: 82.39%.
132:[9, 6, 3, 6, 0, 1, 7, 0, 6, 9, 7, 3, 7, 1, 5, 9, 0, 3, 5, 1], Covers: 55 interactions, Covered: 82.68%.
133:[5, 9, 4, 9, 2, 4, 1, 5, 0, 7, 2, 1, 9, 0, 0, 5, 3, 0, 4, 7], Covers: 55 interactions, Covered: 82.97%.

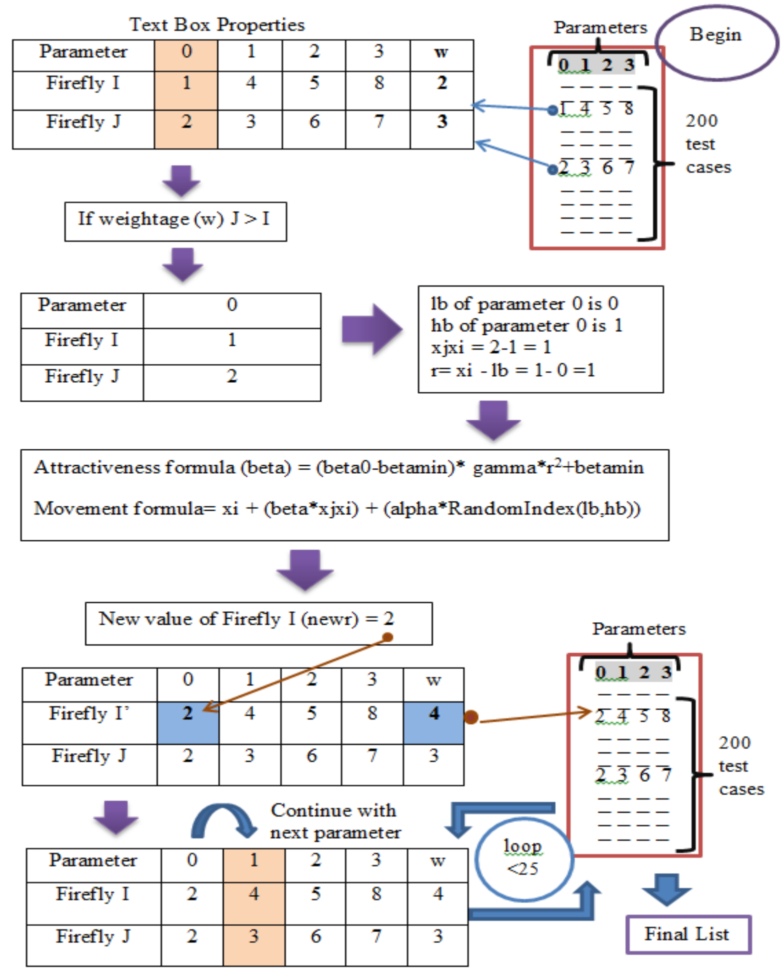




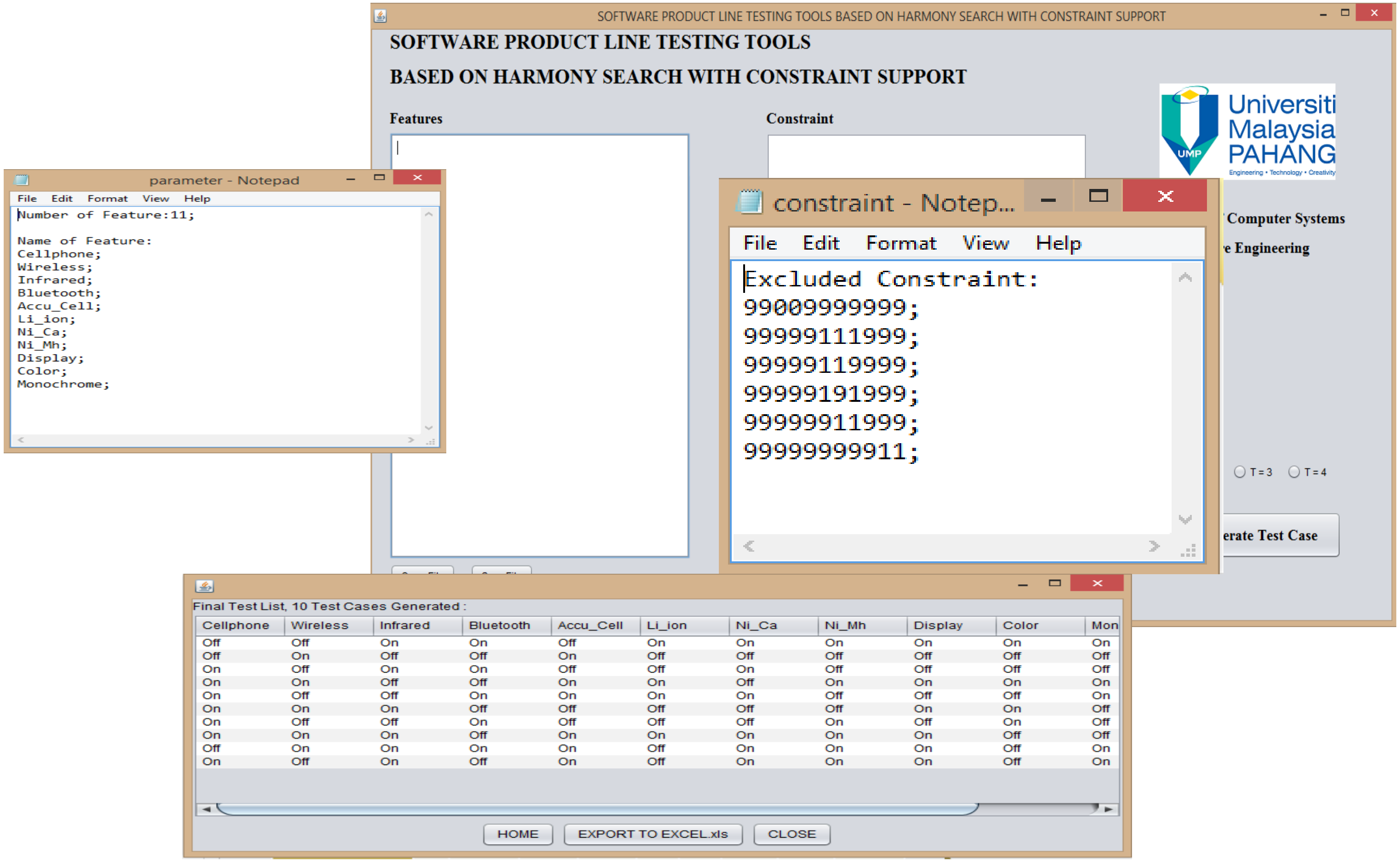
FRUITS	DRINKS	FOODS	ICE-CREAM	SOUP	FAST FOOD	SEAFOOD
MANGO	MILO	BREAD	CHOCOLA...	ASAM LAKSA	MARRYBR...	LOBSTER
ORANGE	NESCAFE	CHICKEN	YAM	ASAM LAKSA	PIZZA HUT	PRAWN
APPLE	TEA	BREAD	STRAWBE...	CURRY	PIZZA HUT	CRAB
APPLE	MILO	NOODLES	STRAWBE...	TOMYAM	KFC	PRAWN
MANGO	TEA	CHICKEN	YAM	TOMYAM	KFC	CRAB
ORANGE	TEA	NOODLES	CHOCOLA...	CURRY	MARRYBR...	PRAWN
APPLE	NESCAFE	NOODLES	CHOCOLA...	TOMYAM	PIZZA HUT	LOBSTER
MANGO	NESCAFE	CHICKEN	STRAWBE...	CURRY	KFC	LOBSTER
ORANGE	MILO	BREAD	YAM	TOMYAM	MARRYBR...	CRAB
APPLE	NESCAFE	NOODLES	CHOCOLA...	ASAM LAKSA	KFC	CRAB
MANGO	MILO	NOODLES	YAM	CURRY	PIZZA HUT	PRAWN
ORANGE	TEA	CHICKEN	STRAWBE...	ASAM LAKSA	MARRYBR...	LOBSTER
APPLE	NESCAFE	BREAD	YAM	TOMYAM	MARRYBR...	PRAWN
ORANGE	MILO	BREAD	YAM	TOMYAM	KFC	LOBSTER
APPLE	MILO	CHICKEN	CHOCOLA...	TOMYAM	KFC	LOBSTER

Input: P: Parameter number n, and
V: set of values for each feature V = [v_0 ..v_j];
Output: Final test suite List *FTS*;
Let *FTS* be a set of candidate tests;
Generate all possible interactions elements IEL based on P and V
Generate initial population of pollens randomly
while IEL is not empty **do**
while t <MaxGeneration **or** stop criterion **do**
for i = 1 : n (all n pollens in the population)
if (rand < *pa*)
Global pollination via $x_t^{i+1} = x_t^i + L(\text{gbest} - x_t^i)$
Else
Do local pollination via $x_t^{i+1} = x_t^i + \varrho(x_t^j - x_t^k)$
End if
Evaluate new solutions
If new solutions are better, update them in the population
End for
Find the current best solution gbest
End while
Add the best test case, gbest, into FTS.
Remove covered interactions elements from IEL.
End while
End-Procedure


INPUT
Universiti Malaysia PAHANG
Engineering • Technology • Creativity
FIREFLY COMBINATORIAL TESTING TOOL
IMPORT FILE
SAVE
GENERATE
FINAL TEST CASE LIST
TOTAL NUMBER OF TEST CASES HAVE BEEN GENERATED = 11
HOME
EXPORT TO
EXCEL.xls
TEXT.txt
PDF.pdf
PRINTING FORMAT
CLOSE
FRUITS DRINKS FOODS ICE-CREAM
MANGO TEA BREAD STRAWBE...
APPLE MILO NOODLES YAM
ORANGE TEA NOODLES CHOCOLA...
ORANGE NESCAFE CHICKEN STRAWBE...
APPLE NESCAFE BREAD CHOCOLA...
MANGO NESCAFE CHICKEN YAM
ORANGE MILO BREAD YAM
MANGO MILO CHICKEN CHOCOLA...
APPLE TEA CHICKEN YAM
MANGO MILO NOODLES STRAWBE...
APPLE NESCAFE NOODLES STRAWBE...


Text Box Properties
Parameter 0 1 2 3 w
Firefly I 1 4 5 8 2
Firefly J 2 3 6 7 3
Parameters
Begin
0 1 2 3
1 4 5 8
2 3 6 7
200 test cases
If weightage (w) J > I
Parameter 0
Firefly I 1
Firefly J 2
lb of parameter 0 is 0
hb of parameter 0 is 1
xjxi = 2-1 = 1
r= xi - lb = 1- 0 =1
Attractiveness formula (beta) = (beta0-betamin)* gamma*r2+betamin
Movement formula= xi + (beta*xjxi) + (alpha*RandomIndex(lb,hb))
New value of Firefly I (newr) = 2
Parameter 0 1 2 3 w
Firefly I' 2 4 5 8 4
Firefly J 2 3 6 7 3
Parameters
0 1 2 3
2 4 5 8
2 3 6 7
200 test cases
Continue with next parameter
loop <25
Parameter 0 1 2 3 w
Firefly I 2 4 5 8 4
Firefly J 2 3 6 7 3
Final List

Universiti Malaysia PAHANG
Engineering • Technology • Creativity



Cellphone	Wireless	Infrared	Bluetooth	Accu_Cell	Li_ion	Ni_Ca	Ni_Mh	Display	Color	Mon
Off	Off	On	On	Off	On	On	On	On	On	On
Off	On	Off	Off	On	Off	Off	Off	Off	Off	Off
On	Off	On	On	Off	Off	On	Off	Off	Off	Off
On	On	Off	Off	On	On	Off	On	On	On	On
On	Off	Off	On	On	On	On	Off	Off	Off	On
On	On	On	Off	Off	Off	Off	Off	On	On	Off
On	Off	Off	On	Off	Off	Off	On	Off	On	Off
On	On	On	Off	On	On	On	On	On	Off	Off
Off	On	On	On	On	Off	On	On	On	Off	On
On	Off	On	Off	On	Off	On	On	On	Off	On

Constraints and Seeding

What about Seedings versus Constraints?

- Constraints – impossible combinations
  - {Crust Thin, Vegetarian, Beef},
  - {Classic Hand Hand Tossed, Vegetarian, Beef}
- Seedings – wanted combinations
  - {Crust Thin, Vegetarian, Pineapple}

What about Seedings versus Constraints?

Crust = { Classic Hand Tossed, Crunchy Thin }	→	A = {0,1}
Flavor = { Vegetarian, Pepperoni Delight }	→	B = {0,1}
Toppings = { Pineapples, Beef }	→	C = {0,1}

- Different perspective …
- Now we use numbers

Equivalent base test cases

	Input Variables		
	A	B	C
Base Values	0	0	0
	1	1	1

Exhaustive Test Suite

	Input Variables		
Base Values	A	B	C
	0	0	0
	1	1	1
Exhaustive Combinations	0	0	0
	0	0	1
	0	1	0
	0	1	1
	1	0	0
	1	0	1
	1	1	0
	1	1	1

	Input Variables		
	A	B	C
Base Values	0	0	0
	1	1	1
2-way Combinations for AB	0	0	0
	0	1	0
	1	0	1
	1	1	1

	Input Variables		
	A	B	C
Base Values	0	0	0
	1	1	1
2-way Combinations for BC	~~0~~	~~0~~	~~0~~
	~~1~~	~~0~~	~~1~~
	~~0~~	~~1~~	~~0~~
	~~1~~	~~1~~	~~1~~

	Input Variables		
	A	B	C
Base Values	0	0	0
	1	1	1
2-way Combinations for AC	~~0~~	~~0~~	~~0~~
	0	0	1
	1	0	0
	~~1~~	~~1~~	~~1~~

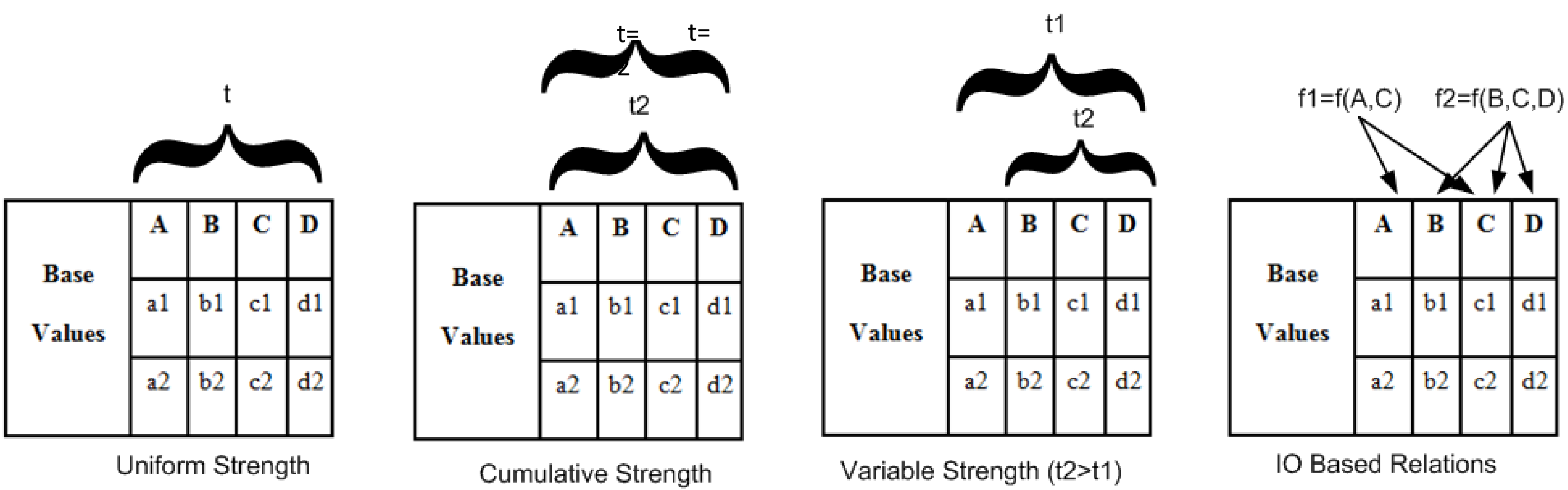
Pairwise, t=2

	Input Variables		
	A	B	C
Base Values	0	0	0
	1	1	1
All 2-way Combinatorial Values	0	0	0
	0	0	1
	0	1	0
	1	0	0
	1	0	1
	1	1	1

Mathematical Notations

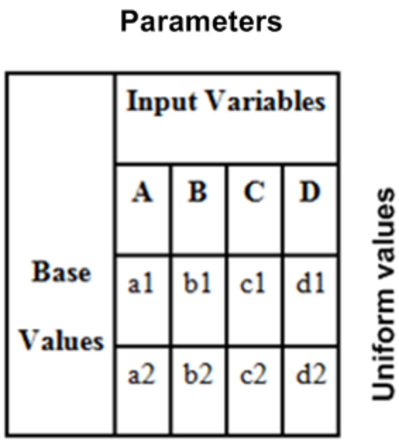
Covering Array vs Mixed Covering Array Notations

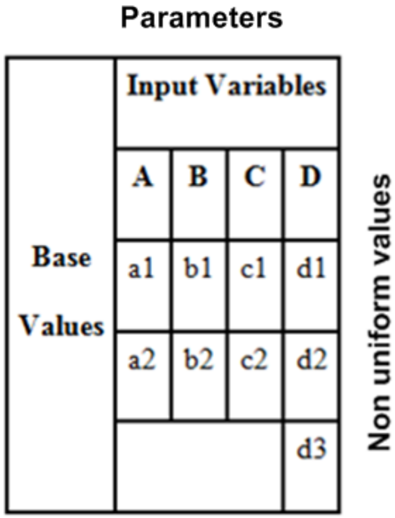
- The problem of t-way test generation can be abstracted as covering array or mixed covering array problem
- Recall, there are four possibilities for applying t-way testing for test reduction
  - Uniform Interaction
  - Cumulative Interaction
  - Variable Strength Interaction



Covering Array vs Mixed Covering Array Notations

- We use:
  - p, v, and t are used to refer to number of parameters (or factor), values (or levels) and interaction strength for the covering array respectively.
  - N is used to refer to test size
  - Covering Array (CA) => values are uniform
  - Mixed Covering array (MCA) => values are non uniform
- Example:
  - CA (N,t,v^{p})
  - CAN (N,t,v^{p}) (i.e. CAN refers to the most minimum test size)
  - MCA $(N,t, v_1^{p1} v_2^{p2} v_3^{p3} .....v_j^{pj})$





Using the CA & MCA notations for Uniform Strength Interaction

- From earlier slides:
  - CA (N, t, v^{p})
  - MCA $(N, t, v_1^{\,p1} v_2^{\,p2} v_3^{\,p3} \ldots v_j^{\,pj})$
- Example Use:
  - CA $(9, 2, 3^{4})$
    - test size = 9, t=2, 4 parameters each with 3 values
  - MCA $(1265, 4, 10^{2}4^{1}3^{2}2^{7})$
    - Test size =1265, t=4, 10 2 valued parameter, 4 1 valued parameter, 3 2 valued parameter, 7 2 valued parameter
- Our motivating example: CA $(13, 3, 2^{4})$ but the most optimum is CAN $(8, 3, 2^{4})$

Parameters are A, B, C

	Input Variables			
Base Values	A	B	C	D
	a1	b1	c1	d1
	a2	b2	c2	d2

Values

Variable Strength Interaction

- Using Covering array notation
  - Variable strength covering array (VCA)
    - VCA (N, t, $v_1^{p1} v_2^{p2} v_3^{p3} \ldots v_j^{pj}$, { $S_1 \ldots S_k$})
    - S consists of a multi-set of disjoint (mixed) covering array with strength larger t.
  - For example, VCA (12, 2, $3^2 2^2$, {CA (3,$3^1 2^2$)}) , indicates the test size of 12 for pairwise interaction (with 2 3 valued parameter and 2 2 valued parameter) and 3-way interaction (with 1 3 valued parameter and 2 2 valued parameter).

Notations for Cumulative Strength Interaction

- Cumulative strength can be viewed as a special case of VCA
- Hence, we can use the same notation for cumulative strength interaction
  - For example, for a 4 parameters system with 2 values, with cumulative strength t=2, and t=3.
  - The notation: VCA $(N, 2, 2^4, \{CA(3,2^4)\})$

Summary - Mathematical Notations for T-Way Strategies

- Uniform Strength Interaction
  - CA (N,t,v^{p}) => uniform parameter values
  - MCA $(N, t, v_1^{p1} v_2^{p2} v_3^{p3} \ldots v_j^{pj})$ => non uniform parameter values
- Variable Strength Interaction (VCA) & Cumulative Strength Interaction
  - VCA $(N, t, v_1^{p1} v_2^{p2} v_3^{p3} \ldots v_j^{pj}, \{ S_1 \ldots S_k \})$
  - S consists of a multi-set of disjoint (mixed) covering array with strength larger t.