JMP GENOMICS VALIDATION USING COVERING ARRAYS AND EQUIVALENCE PARTITIONING

WENJUN BAO AND JOSEPH MORGAN
JMP DIVISION, SAS INC.

THE SOFTWARE TESTING DILEMMA

Defect density[1] for released software:

- **Industry**: ≈ 1 - 25 defects/1000 lines of code.
- **Microsoft Applications**: ≈ 0.5 defects/1000 lines of code.
- **Space Shuttle**: ≈ 0 defects/1000 lines of code.

[1]S. McConnell, "Code Complete: A Practical Handbook of Software Construction," 2004

THE SOFTWARE TESTING DILEMMA

Downtime hourly cost[1]:

❑ Brokerage operations	$6,450,000
❑ Credit card authorization	$2,600,000
❑ Ebay	$ 225,000
❑ Amazon	$ 180,000
❑ Home shopping channel	$ 113,000
❑ Airline reservation center	$ 89,000
❑ Cellular service activation	$ 41,000

[1]Hartman, A., “Covering Arrays: Mathematical Engineering & Scientific Perspectives,” Fields Institute Workshop on Covering Arrays, 2006

THE SOFTWARE TESTING DILEMMA

- Debugging, testing, and verification activities in typical commercial software development organizations range from **50%** to **75%** of total software development costs.
- A recent National Institute of Standards and Technology (NIST) report[1] investigating the economic impact of inadequate software testing methods and tools found:

  "...THE NATIONAL ANNUAL COSTS OF AN INADEQUATE INFRASTRUCTURE FOR SOFTWARE TESTING IS ESTIMATED TO RANGE FROM $22.2 BILLION TO $59.5 BILLION"

[1] NIST Planning Report 02-3, "The Economic Impacts of Inadequate Infrastructure for Software Testing," 2002.

THE SOFTWARE TESTING DILEMMA

- Testing is necessary.
- Testing is important.
- Testing is expensive.
- Poor testing is even ***more*** expensive!

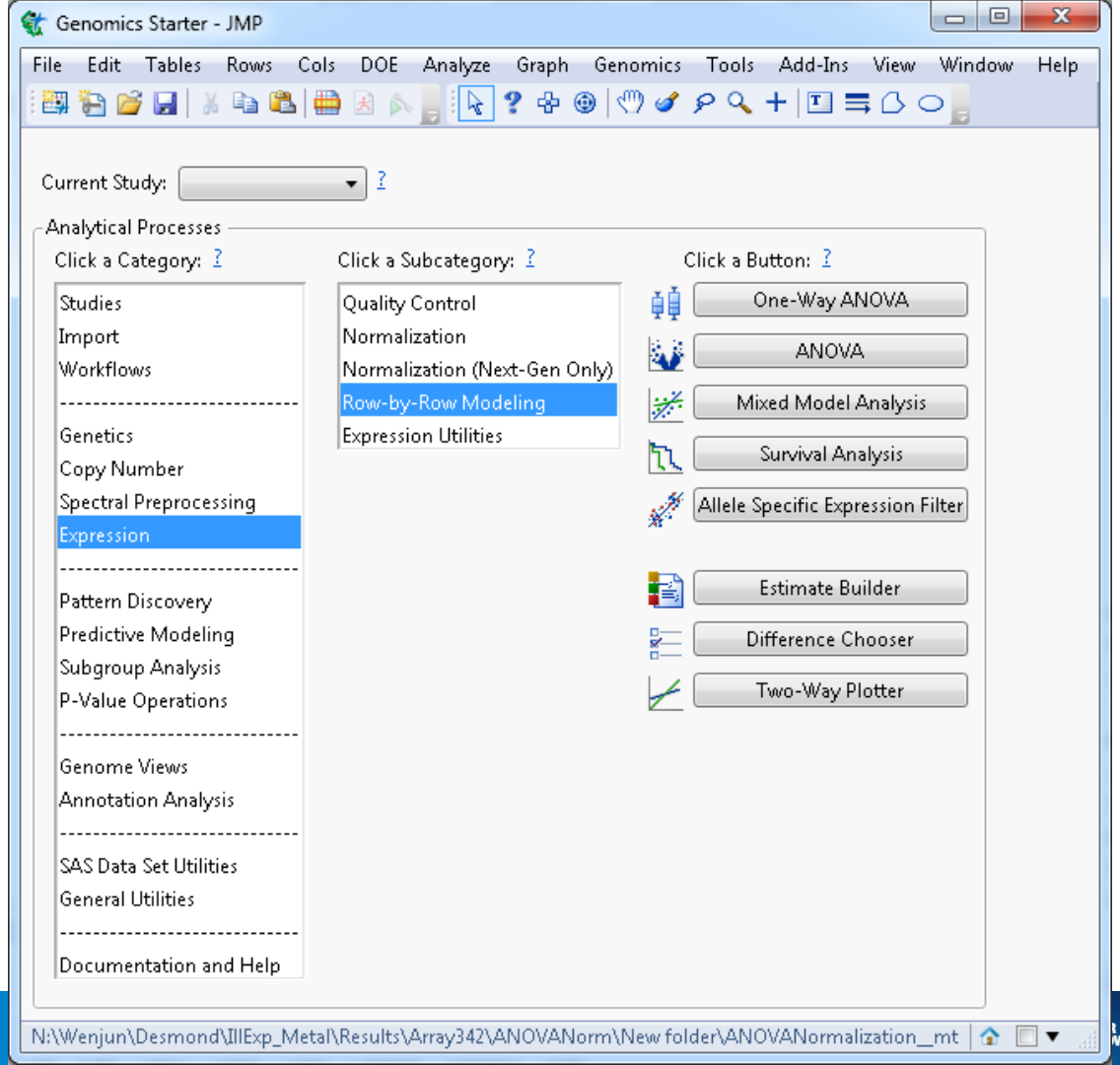
"… TOO LITTLE TESTING MAY BE INITIALLY CHEAPER BUT *NOT* IN THE LONG RUN[1]."

[1]Hartman, A., "Covering Arrays: Mathematical Engineering & Scientific Perspectives," Fields Institute Workshop on Covering Arrays, 2006

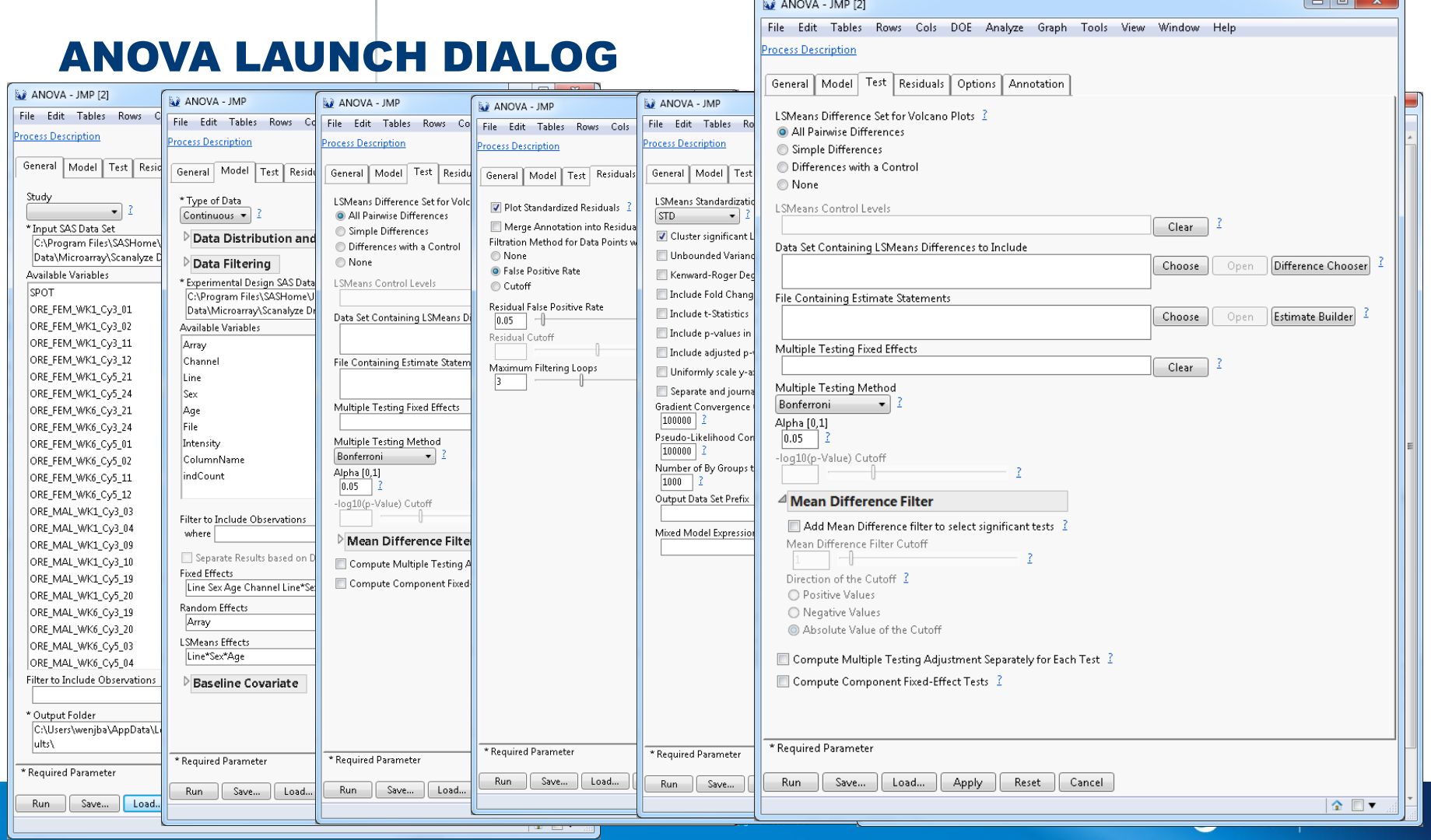
JMP GENOMICS

- A statistical discovery software tool for biological data including different big data (“Omics”).
- A JMP/SAS Integration product.
- More than ***two hundred*** analytical procedures (AP), with a rich set of options for each procedure.

JMP GENOMICS



ANOVA LAUNCH DIALOG



ANOVA LAUNCH DIALOG VALIDATION

- Twenty one distinct GUI controls.
- Each control provides between two and fifteen choices.

1.	**Distribution**	15 choices
2.	**Filter zero/missing data**	2 choices
3.	**Separate results**	2 choices
4.	**LSMeans diff. set**	4 choices
5.	**Multiple testing method**	14 choices
6.	**Alpha**	3 choices
7.	**Multiple testing adj**.	2 choices
8.	**Component fixed-eff. tes**ts	2 choices
9.	**Plot std. residuals**	3 choices
10.	**Filtration method**	9 choices
11.	**LSMeans methods**	15 choices
12.	**Cluster LSMean profiles**	2 choices
13.	**Unbounded variance**	2 choices
14.	**Kenward-Roger df method**	2 choices
15.	**U diffs**.	2 choices
16.	**Include p-values**	2 choices
17.	**Include adj. p-values**	2 choices
18.	**Scale Volcano y-axis**	2 choices
19.	**Journal results by chrom.**	2 choices
20.	**Primary annotation**	2 choices
21.	**Secondary annotation**	2 choices

ANOVA LAUNCH DIALOG VALIDATION

- Input space:

  $15 * 2^2 * 4 * 14 * 3 * 2^2 * 3 * 9 * 15 * 2^{10} = 16{,}721{,}510{,}400$ inputs

- Exhaustive testing:
  - ≈ 17 billion test cases.
  - Assuming 5 seconds would be needed to execute and verify each case, then the ANOVA AP would require:

2,651 YEARS TO COMPLETE!

THE *DESIGN OF EXPERIMENTS* INSPIRATION

One way of addressing the dilemma is to recognize that it is somewhat similar to a statistical **design of experiments** (**DOE**) problem. There have been several key milestones in the application of DOE ideas to software testing.

- **Orthogonal Latin squares:** In 1985, R. Mandl (*Commun. ACM, v28.10*) proposed using orthogonal Latin squares for testing compilers.
- **Orthogonal arrays:** In 1992, R. Brownlie (*AT&T Tech. Journal, v71.3*) proposed orthogonal arrays to generate test suites for application testing.
- **Covering arrays:** In 1998, C. Mallows (*Technometrics, v40.3*) proposed covering arrays to generate test suites for application testing.

COVERING ARRAYS

Definition 1

A *covering array* **CA**(N; t, k, v) is an $N \times k$ array such that the *i*-th column contains v distinct symbols. If a **CA**(N; t, k, v) has the property that for any *t* coordinate projection, all v^t combinations of symbols exist at least once, then it is a ***t***-covering array (or strength t covering array). A ***t***-covering array is optimal if N is minimal for fixed t, k, and v.

COVERING ARRAYS

1	1	1	1	1
2	2	2	2	2
2	2	2	1	1
2	1	1	2	2
1	2	1	2	1
1	1	2	1	2

CA(6; 2, 5, 2)

Are these optimal?

CA(12; 3, 5, 2)

1	1	1	1	1
1	1	2	1	1
1	2	1	1	2
1	2	2	1	2
2	1	1	2	1
2	1	2	2	1
2	2	1	2	2
2	2	2	2	2
1	1	1	2	2
2	2	1	1	1
2	1	2	1	2
1	2	2	2	1

MIXED LEVEL COVERING ARRAYS

Definition 2

A *mixed level covering array* **MCA**(N; t, $(v_1 \cdot v_2 \cdot \ldots \cdot v_k)$) is an $N \times k$ array such that the *i*-th column contains v_i distinct symbols. If for any *t* coordinate projection, all $\prod v_i$ combinations of symbols exist, then it is a ***t***-covering array and is optimal if N is minimal for fixed t, k, and $(v_1 \cdot v_2 \cdot \ldots \cdot v_k)$.

1	1	2	2	2
1	2	2	2	1
1	3	1	1	1
2	1	1	2	1
2	2	1	1	2
2	3	2	1	1
3	1	1	1	2
3	2	2	1	1
3	3	2	2	2

An optimal **MCA**(9; 2, $(3^2 \cdot 2^3)$).

COVERING ARRAYS AND GENOMICS TESTING

- ANOVA - Mixed Level Covering Array:

$$\mathbf{MCA}(\ N;\ 2,\ (15^2 \cdot 14 \cdot 9 \cdot 4 \cdot 3^2 \cdot 2^{14}\ )\ )$$

- What is *N*?

N = 225

- If we need 5 seconds for each case, then the ANOVA AP would require:

18.75 MINUTES TO COMPLETE!

COVERING ARRAYS AND GENOMICS TESTING

Design

Run	Distribution	FilterData	SeparateResults	LSMeansDiffSet	MultipleTestingMethod	Alpha	MultipleTestAdj
1	Lognormal	yes	no	Simple Differences	Hochberg	0	yes
2	Generalized Poisson	yes	yes	Differences with a Control	Hommel	1	no
3	Poisson	no	yes	Simple Differences	None	0	no
4	t	no	yes	Simple Differences	Bonferroni	0.05	yes
5	Gamma	no	yes	Differences with a Control	DependentFDR	0.05	yes
6	Exponential	no	yes	All Pairwise Differences	Holm	1	yes
7	Negative Binomial	no	yes	Differences with a Control	pFDR	0	no
8	Multinomial	no	yes	All Pairwise Differences	Sidak	0.05	yes
9	t	yes	no	Simple Differences	FDR	0	no
10	Beta	no	no	All Pairwise Differences	AdaptiveHochberg	0.05	yes
223	None	yes	no	All Pairwise Differences	Sidak	0.05	no
224	Binary	yes	yes	Differences with a Control	None	1	yes
225	Multinomial	no	yes	Simple Differences	Holm	1	no

Optimize Maximum iterations: 250

Metrics

t	t-Coverage	t-Diversity
2	100.00	7.90
3	95.16	21.06
4	80.72	41.01
5	58.32	62.35
6	35.59	79.68

COVERING ARRAYS AND GENOMICS TESTING

- Can we do better?

YES!

- With Equivalence Partitioning, our Mixed Level Covering Array becomes:

$$\mathbf{MCA}(\ 72;\ 2,\ (9 \cdot 8 \cdot 6 \cdot 4 \cdot 3^2 \cdot 2^{14}\ )\ )$$

- If we need 5 seconds for each case, then the ANOVA AP would require:

6 MINUTES TO COMPLETE!

COVERING ARRAYS AND GENOMICS TESTING

Design

Run	DistributionGroup	FilterData	SeparateResults	LSMeansDiffSet	MultipleTestingMethodGroup	Alpha	MultipleTestAdj
1	discrete	no	yes	Simple Differences	None	0.05	no
2	discrete	no	yes	Simple Differences	None	0	yes
3	continuous	no	yes	Simple Differences	FDR	0.05	yes
4	continuous	yes	yes	Differences with a Control	Bonferroni	0	no
5	discrete	no	no	All Pairwise Differences	AdaptiveFDR	0.05	no
6	discrete	no	yes	All Pairwise Differences	pFDR	0	no
7	continuous	yes	yes	Simple Differences	AdaptiveHolm	0	yes
8	discrete	yes	yes	Differences with a Control	None	0	no
9	discrete	yes	yes	None	pFDR	0	no
70	continuous	no	yes	All Pairwise Differences	AdaptiveHolm	1	yes
71	discrete	no	yes	None	DependentFDR	0	yes
72	discrete	yes	yes	Simple Differences	pFDR	0	no

Optimize Maximum iterations: 250

Metrics

t	t-Coverage	t-Diversity
2	100.00	12.22
3	94.44	28.76
4	76.78	51.24
5	50.74	72.22
6	27.46	86.43

ANOVA VALIDATION

Exhaustive Testing

17 Billion cases
2651 years (5/sec)

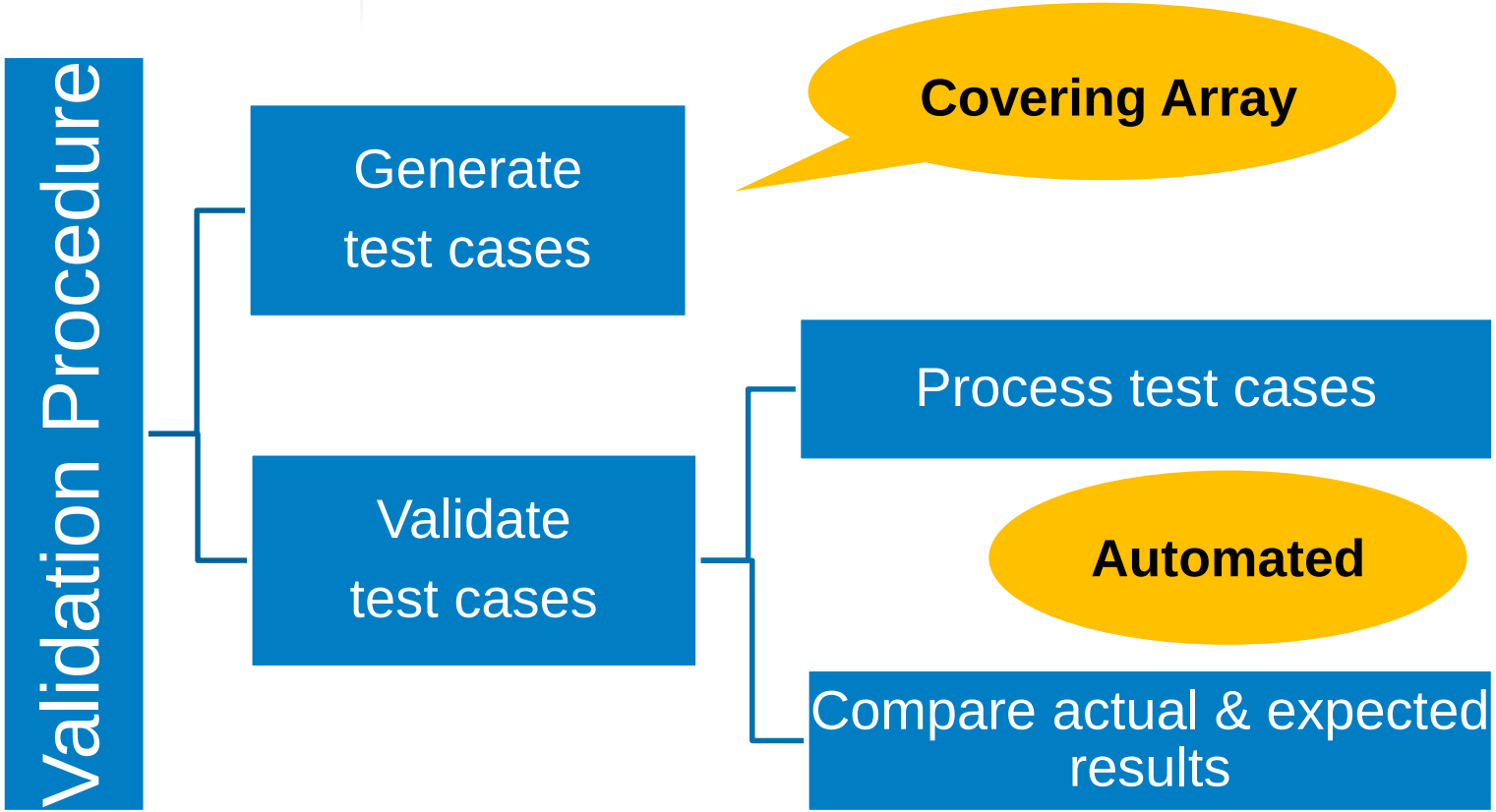
Covering Array Testing

225 cases
19 minutes (5/sec)

Covering Array & Equivalence Partitioning Testing

72 cases
6 minutes (5/sec)

JMP GENOMICS VALIDATION



CONCLUSION

- Test cases derived from covering arrays can exercise all *t*-way combinations of input parameter values at very little cost.
- Compared to exhaustive testing, the cost savings is dramatic with potentially very little loss in quality (i.e. undiscovered faults).
- Increasingly, software development organizations are embracing this technology.

  Note: The JMP development team started using covering arrays in 2006.
- Empirical research suggests that most software failures are due to faults resulting from the interaction of few parameters (typically ≤ 6).

REFERENCES

1. R. Brownlie, J. Prowse, & M. Padke, "*Robust testing of AT&T PMX/StarMAIL using OATS*," AT&T Technical Journal, 71(3), 1992, pg41-47.
2. D. Cohen, S. Dalal, J. Parelius, & G. Patton, "*The combinatorial design approach to automatic test generation*," IEEE Software, 13(5), 1996, pg83-88.
3. S. Dalal, A. J. Karunanithi, J. Leaton, G. Patton, & B. Horowitz, "*Model-based testing in practice*," Proceedings of the 21st ICSE, New York, 1999, pg285-294.
4. S. Dalal, & C. Mallows, "*Factor covering designs for testing software*," Technometrics, 40(3), 1998, pg234-243.
5. B. Hailpern & P. Santhanam, "*Software Debugging, Testing, and Verification*," IBM Systems Journal, 41(1), 2002.
6. A. Hartman & L. Raskin, "*Problems and algorithms for covering arrays*," Discrete Math, 284(1–3), 2004, pg149–156.
7. R. Mandl, "*Orthogonal latin squares: an application of experiment design to compiler testing*," Communications of the ACM, 28(10), 1985, pg1054-1058.
8. S. McConnell, "*Code Complete: A Practical Handbook of Software Construction*," 2004.
9. J. Morgan, G. Knafl, & W. Wong, "*Predicting Fault Detection Effectiveness*," Proceedings of the 4th ISSM, 1997, pg82-90.
10. NIST Planning Report 02-3, "*The Economic Impacts of Inadequate Infrastructure for Software Testing*," 2002.
11. C. Yilmaz, S. Fouche, M. Cohen, A. Porter, G. Demiroz, & U. Koc, "*Moving Forward with Combinatorial Interaction Testing*," IEEE Computer, 47(2), 2014, pg37 - 45.

Thank You!!