

COMS30026 Design Verification

# Coverage

## Part III: Coverage Analysis

Kerstin Eder

(Acknowledgement: Avi Ziv from the IBM Research Labs in Haifa has kindly permitted the re-use of some of his slides.)

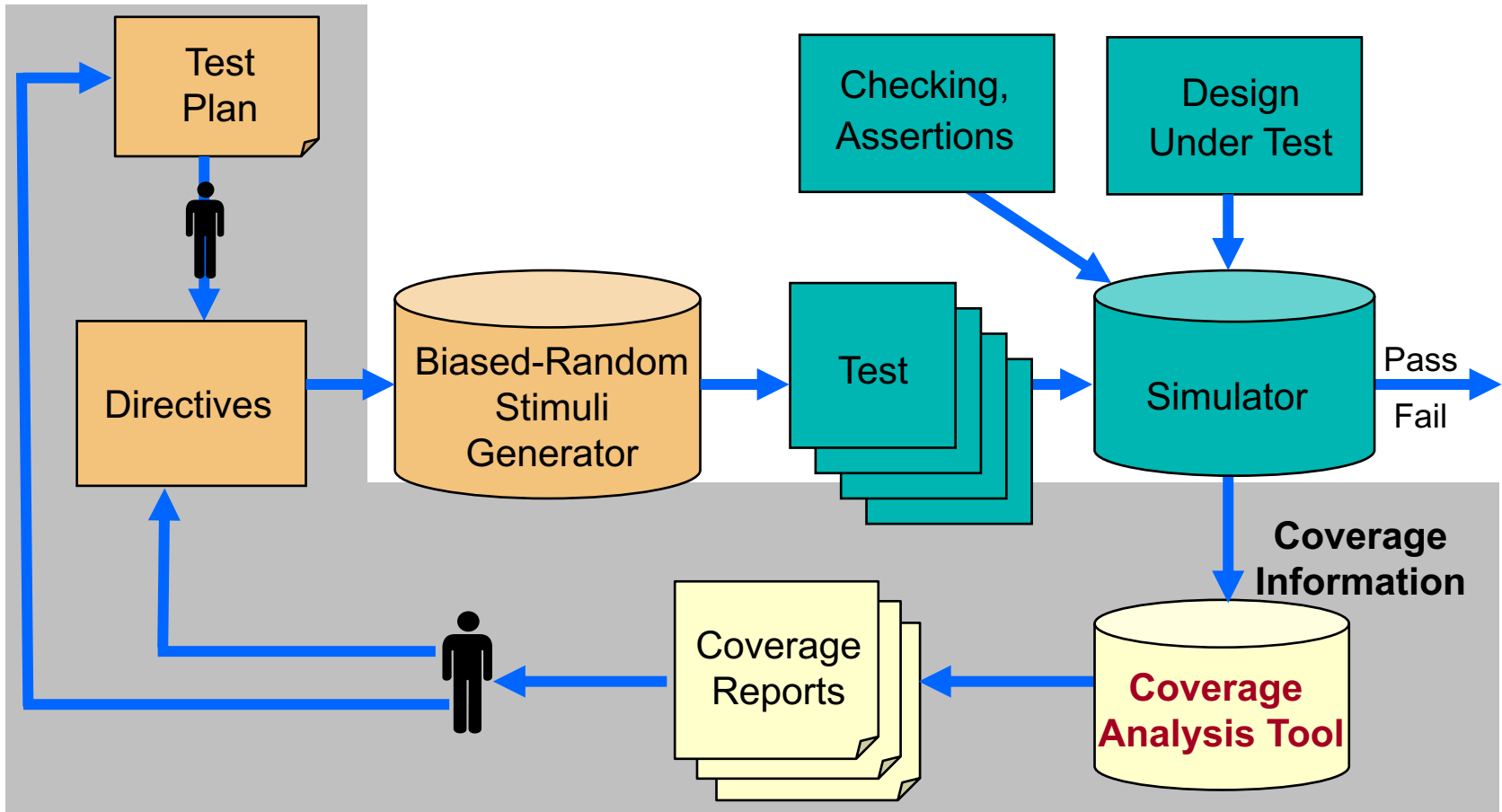
# Outline

---

- Introduction to coverage
- Part I: Coverage Types
  - Code coverage models
  - (Structural coverage models)
- Part II: Coverage Types (continued)
  - Functional coverage models
- **Part III: Coverage Analysis**
- Previously: Verification Tools
  - Coverage is part of the Verification Tools.



# Coverage Analysis



# Why Coverage Analysis

---

- The main goals of the coverage process are
  - Monitor the quality of the verification
  - Identify unverified and lightly verified areas
  - Help us understand verification progress
- **Coverage analysis** helps closing the loop from coverage measurement to the verification plan and test generation



# Coverage Analysis Goals

---

- **Conflicting goals for coverage analysis:**
  - Want to collect as much data as possible
    - Not to miss important events
  - User needs concise and informative reports
    - Not to get drawn into too much detail
- Different types of users require different types of information
- **Goal:** provide concise and informative reports that address the specific needs of the report user



# Types of Coverage Reports

---

- Progress reports
  - Progress of coverage over time (more on this later)



# Types of Coverage Reports

---

- Progress reports
  - Progress of coverage over time (more on this later)
- Status reports
  - Coverage status summary
  - Detailed status reports of covered and uncovered tasks
    - Reports can be adapted to specific user needs
    - Allow interactive navigation between reports to explore coverage state



# Coverage Status Summary

- Provides a short summary of the coverage to date
- Provides the overall state of the coverage model (or models)
- Useful for
  - Status meetings and status reports
  - A quick glance at the coverage state

Size of coverage space:	1539648
Number of tasks:	4200
Number of tasks covered:	1273
Percent tasks covered:	30.39524
Number of holes:	2927
Number of illegal tasks:	9
Number of traces measured:	16254
Number of cycles measured:	94231273





# Detailed Status Report

- Provides details on each task in the coverage model
  - Covered or not
  - How many times covered
  - In how many tests covered
  - First and last time covered
  - Coverage goals, i.e. how often it needs to be covered
  - ...

Ints1	Inst2	Reg	Dep	goal	Tests covered	Times covered
Add	Mul	GPR	RR	3	1	2
Add	Stw	G0	RW	3	13	21
Sub.	Add.	CR	WR	3	2	3
Mul	Div	GPR	WW	3	0	0
Ldw	And	GPR	None	3	3	9
FPdiv	FPsub	FPR	WW	3	1	1
Br	Sub.	CR	RR	3	12	11



# Detailed Status Report

- Provides details on each task in the coverage model
  - Covered or not
  - How many times covered
  - In how many tests covered
  - First and last time covered
  - Coverage goals, i.e. how often it needs to be covered
  - ...

Ints1	Inst2	Reg	Dep	goal	Tests covered	Times covered
Add	Mul	GPR	RR	3	1	2
Add	Stw	G0	RW	3	13	21
Sub.	Add.	CR	WR	3	2	3
Mul	Div	GPR	WW	3	0	0
Ldw	And	GPR	None	3	3	9
FPdiv	FPsub	FPR	WW	3	1	1
Br	Sub.	CR	RR	3	12	11



# Detailed Status Report

- Provides details on each task in the coverage model
  - Covered or not
  - How many times covered
  - In how many tests covered
  - First and last time covered
  - Coverage goals, i.e. how often it needs to be covered

...

	Inst1	Inst2	Reg	Dep	goal	Tests covered	Times covered
	Add	Mul	GPR	RR	3	1	2
	Add	Stw	G0	RW	3	13	21
	Add	Mul	GPR	RR	3	1	2
	Add	Stw	G0	RW	3	13	21
	Sub.	Add.	CR	WR	3	2	3
	Mul	Div	GPR	WW	3	0	0
	Ldw	And	GPR	None	3	3	9
	Add	Mul	GPR	RR	3	1	2
	Add	Stw	G0	RW	3	13	21
	Sub.	Add.	CR	WR	3	2	3
	Mul	Div	GPR	WW	3	0	0
	Ldw	And	GPR	None	3	3	9
	FPdiv	FPsub	FPR	WW	3	1	1
	Br	Sub.	CR	RR	3	12	11
	FPdiv	FPsub	FPR	WW	3	1	1
	Br	Sub.	CR	RR	3	12	11
	Sub.	Add.	CR	WR	3	2	3
	Mul	Div	GPR	WW	3	0	0
	Add	Mul	GPR	RR	3	1	2
	Add	Stw	G0	RW	3	13	21
	Sub.	Add.	CR	WR	3	2	3
	Mul	Div	GPR	WW	3	0	0
	Ldw	And	GPR	None	3	3	9
	FPdiv	FPsub	FPR	WW	3	1	1
	Br	Sub.	CR	RR	3	12	11
	Ldw	And	GPR	None	3	3	9
	FPdiv	FPsub	FPR	WW	3	1	1
	Add	Mul	GPR	RR	3	1	2
	Add	Stw	G0	RW	3	13	21
	Sub.	Add.	CR	WR	3	2	3
	Mul	Div	GPR	WW	3	0	0
	Ldw	And	GPR	None	3	3	9
	FPdiv	FPsub	FPR	WW	3	1	1
	Br	Sub.	CR	RR	3	12	11
	Br	Sub.	CR	RR	3	12	11



# Detailed Status Reports

---

- Detailed status reports can provide too much detail even for a moderately sized coverage model
  - Hard to focus on the areas in the coverage model we are currently interested in
  - Hard to understand the meaning of the coverage information
    - Are we missing something important?
- Solution: **Views into the coverage data**
  - Allow the user to focus on the current area of interest and inspect the coverage data using the appropriate level of detail
  - Allow to dynamically re-define the coverage model using different perspectives



# Types of Coverage Views

---

- Views based on **coverage data**
  - Counts
  - Date stamps
- Views based on **coverage definition**
  - Projection
  - Selection
  - Partitioning
- Other filtering mechanisms

All the above options can be combined.



# Projection

- Project the  $n$ -dimensional coverage space onto an  $m$  ( $< n$ ) -dimensional subspace
- Allow users to concentrate on a specific set of attributes
- May help investigate some of the things leading up to the bigger picture, and may inform test generation

Instruction	Count
fadd	12321
fsub	10923
fmul	4232
fsqrt	13288
fabs	9835



# Selection

- Select a subset of the values in the report
- Allows the report to concentrate on a specific area in the coverage model
- Clears the report from data that is not of interest at the time

Instruction	Count	Density
fadd	12321	127/136
fdiv	11729	101/136
fmadd	9725	107/136
fmsub	9328	111/136
fmul	4232	94/136
fres	10373	105/136
frsqрте	9792	23/36
fsqrt	13288	40/56
fsub	10923	122/136



# Selection

- Select a subset of the values in the report
- Allows the report to concentrate on a specific area in the coverage model
- Clears the report from data that is not of interest at the time

Instruction	Count	Density
fmadd	9725	107/136
fmsub	9328	111/136
frsqrt	9792	23/36
fsqrt	13288	40/56





# Partitioning

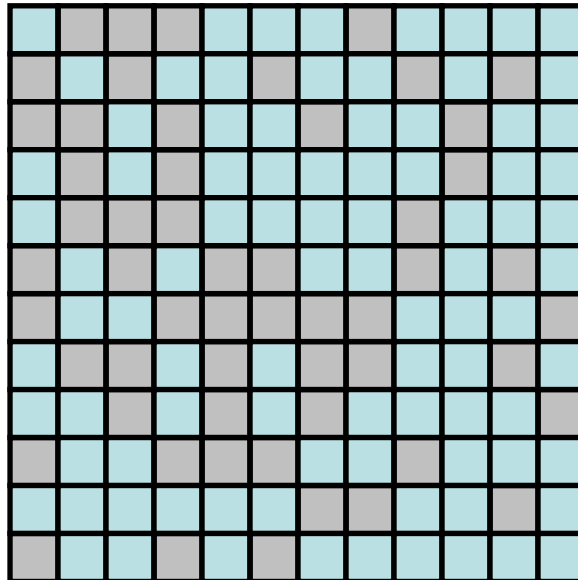
---

- Provides a more **coarse-grained view** of the coverage data
- Partition values of given attributes into non-overlapping sets
  - Example: **Instruction types** -> Arith, Branch, Load, Store, etc



# Partitioning

- Provides a more **coarse-grained view** of the coverage data
- Partition values of given attributes into non-overlapping sets
  - Example: **Instruction types** -> Arith, Branch, Load, Store, etc



# Partitioning

- Provides a more **coarse-grained view** of the coverage data
- Partition values of given attributes into non-overlapping sets
  - Example: **Instruction types** -> Arith, Branch, Load, Store, etc

4/12	9/12	9/12
5/12	10/12	8/12
7/12	3/12	9/12
8/12	7/12	10/12



# Automatic Coverage Analysis

---

- Detailed status reports do not always reveal interesting information hidden in the coverage data
  - You need to know where to look
  - You need to know which questions to ask the coverage tool, and which views to select
- Specifically, we often want to **find large areas of uncovered tasks** in the coverage model, ideally automatically
  - *Why are these important?*



# Large Holes Example

- All combinations of two attributes, X and Y
  - Possible values 0 – 9 for both (10x10 = 100 coverage tasks)
- After a period of testing, 70% coverage is achieved

## Uncovered Tasks

X	Y
0	2
0	3
1	2
1	4
2	1
2	2
2	6
3	2
3	7
4	2

X	Y
4	4
5	2
5	8
6	2
6	6
6	7
6	8
7	2
7	3
7	4

X	Y
7	6
7	7
7	8
8	2
8	6
8	7
8	8
8	9
9	2
9	9

Can you spot any patterns?



# Large Holes Example

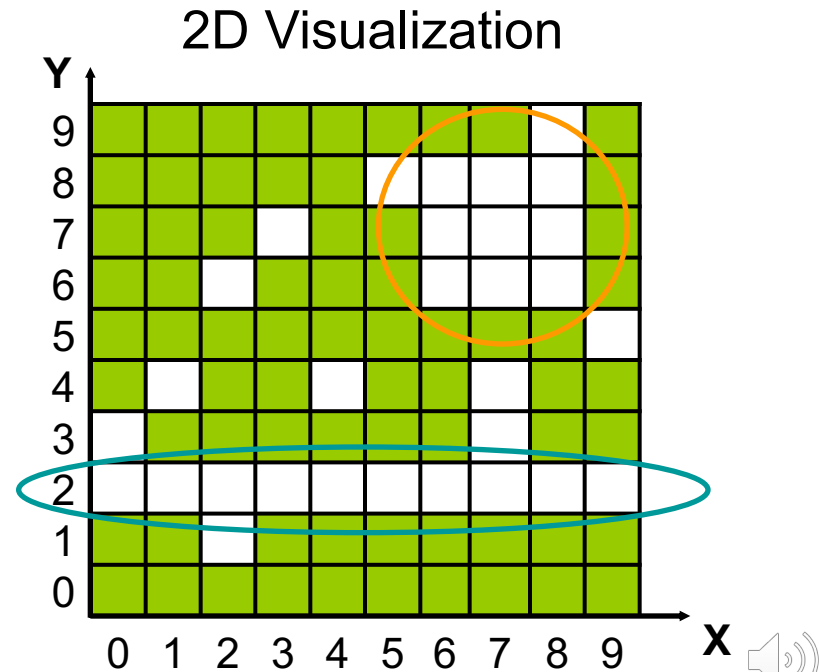
- All combinations of two attributes, X and Y
  - Possible values 0 – 9 for both (100 coverage tasks)
- After a period of testing, 70% coverage is achieved

## Uncovered Tasks

X	Y
0	2
0	3
1	2
1	4
2	1
2	2
2	6
3	2
3	7
4	2

X	Y
4	4
5	2
5	8
6	2
6	6
6	7
6	8
7	2
7	3
7	4

X	Y
7	6
7	7
7	8
8	2
8	6
8	7
8	8
8	9
9	2
9	9



# Large Holes Example

- All combinations of two attributes, X and Y
  - Possible values 0 – 9 for both (100 coverage tasks)
- After a period of testing, 70% coverage is achieved

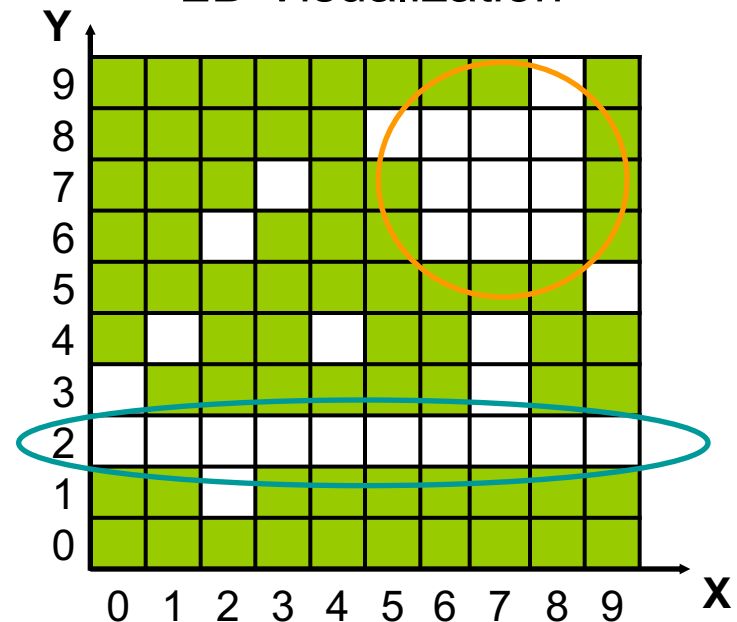
## Uncovered Tasks

X	Y
0	2
0	3
1	2
1	4
2	1
2	2
2	6
3	2
3	7
4	2

X	Y
4	4
5	2
5	8
6	2
6	6
6	7
6	8
7	2
7	3
7	4

X	Y
7	6
7	7
7	8
8	2
8	6
8	7
8	8
8	9
9	2
9	9

## 2D Visualization



# Hole Analysis Algorithms

- Try to find large areas in the coverage space that are not covered
- Use basic techniques to combine sets of uncovered events into large meaningful holes
- Two basic algorithms
  - Aggregation
  - Projected holes



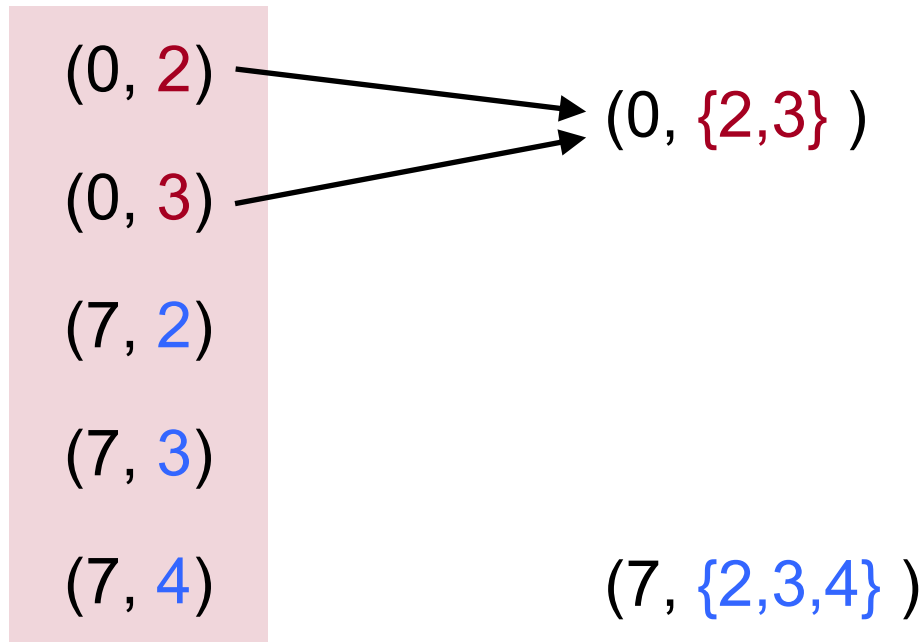
H. Azatchi, L. Fournier, E. Marcus, S. Ur, A. Ziv and K. Zohar, "Advanced Analysis Techniques for Cross-Product Coverage," in *IEEE Transactions on Computers*, vol. 55, no. 11, pp. 1367-1379, Nov. 2006, DOI: [10.1109/TC.2006.173](https://doi.org/10.1109/TC.2006.173).



# Aggregated Holes

- Combine uncovered tasks with common values in some attributes
  - Like using Karnaugh maps
- Example coverage space, attributes X and Y each 0..9

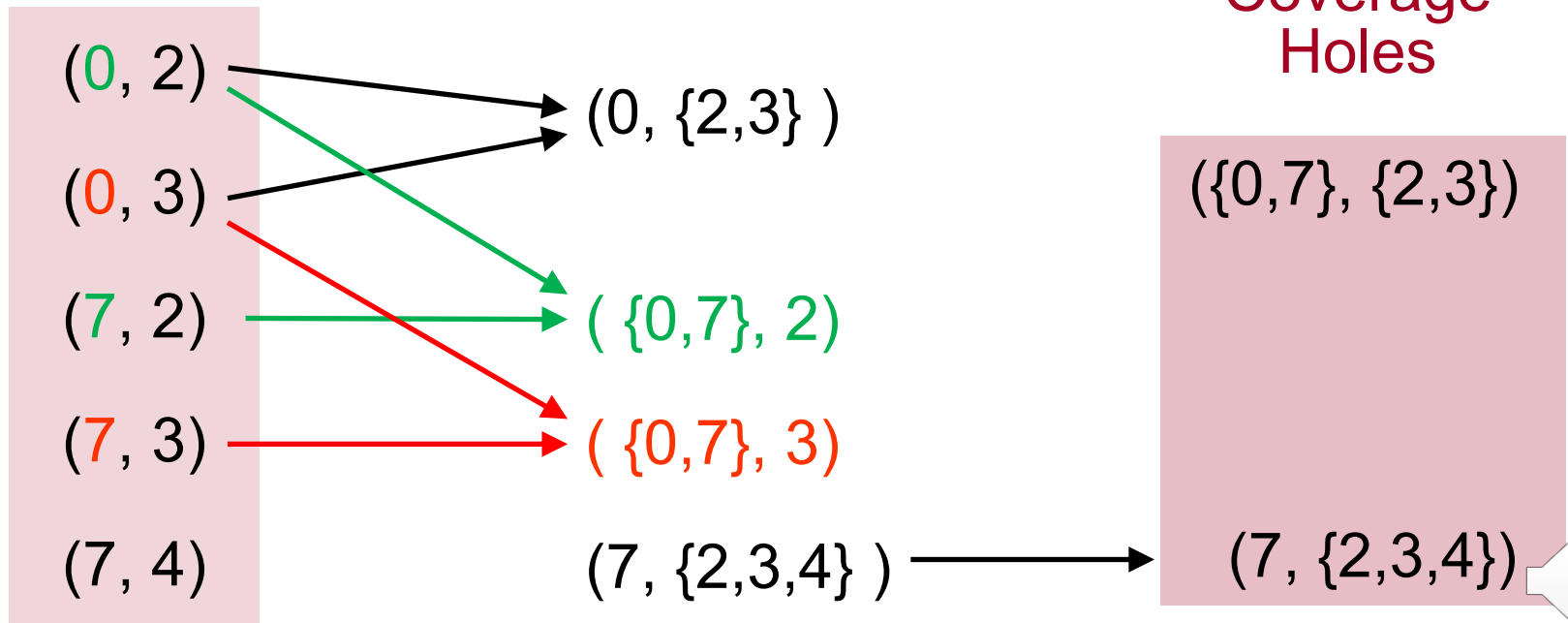
## Uncovered Tasks (X,Y)



# Aggregated Holes

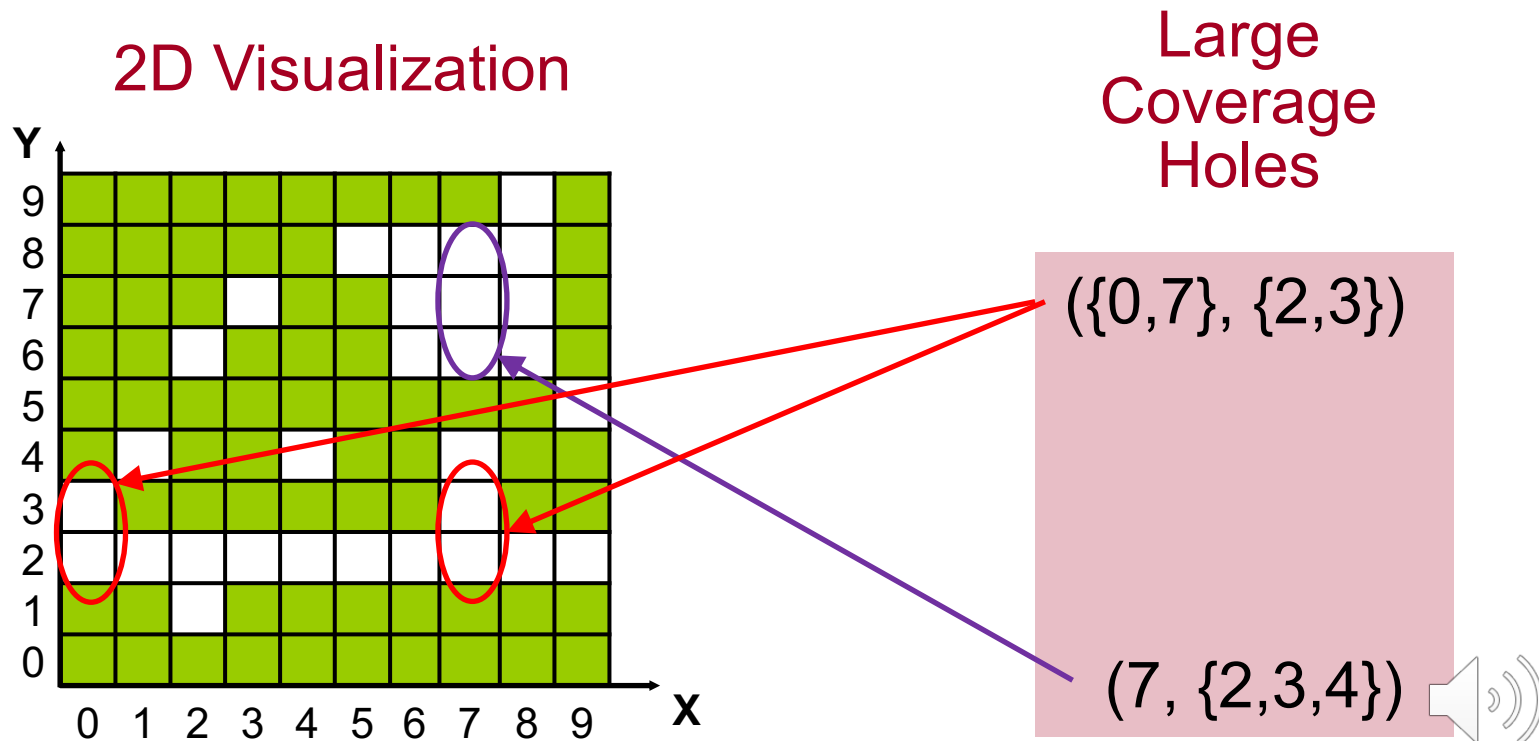
- Combine uncovered tasks with common values in some attributes
  - Like using Karnaugh maps
- Example coverage space, attributes X and Y each 0..9

Uncovered Tasks (X,Y)



# Aggregated Holes

- Combine uncovered tasks with common values in some attributes
  - Like using Karnaugh maps
- Example coverage space, attributes X and Y each 0..9



# Projected Holes

- Find holes that are **complete subspaces** of the functional cross-product coverage space
- Coverage holes are in the form  $(q_1, q_2, \dots, q_n)$ 
  - $q_i$  is either a single value or a wildcard (\*)
- The dimension of a coverage hole is the number of wildcards in the tuple
- **Example:**  $(fadd, add, *, WW)$  has dimension 1

*“There has not been an instruction sequence where fadd is followed by add with a WW dependency for any of the registers.”*



# Projected Holes

## Terminology:

- Coverage hole  $p$  is an **ancestor** of coverage hole  $q$  if all the tasks in  $q$  are also in  $p$ .

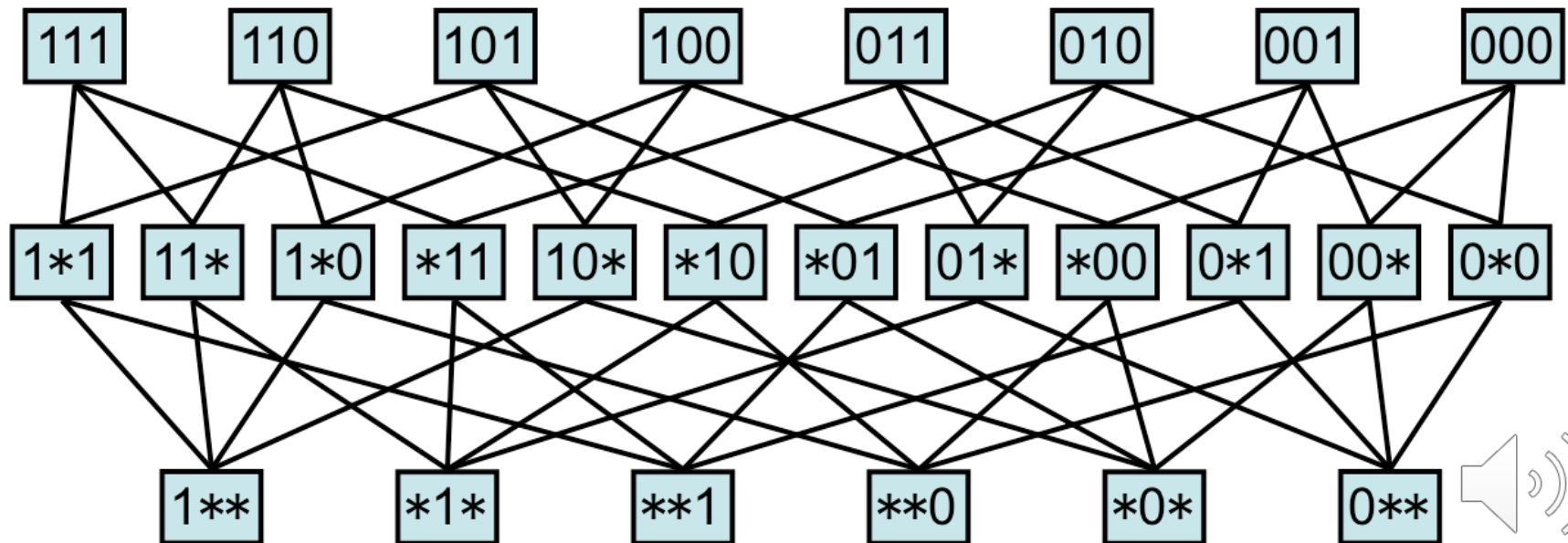
(fadd, \*, \*, WW) is an ancestor of (fadd, add, \*, WW)

- Holes with higher dimensions usually represent larger subspaces.
- The higher the dimension the higher the priority for coverage closure.



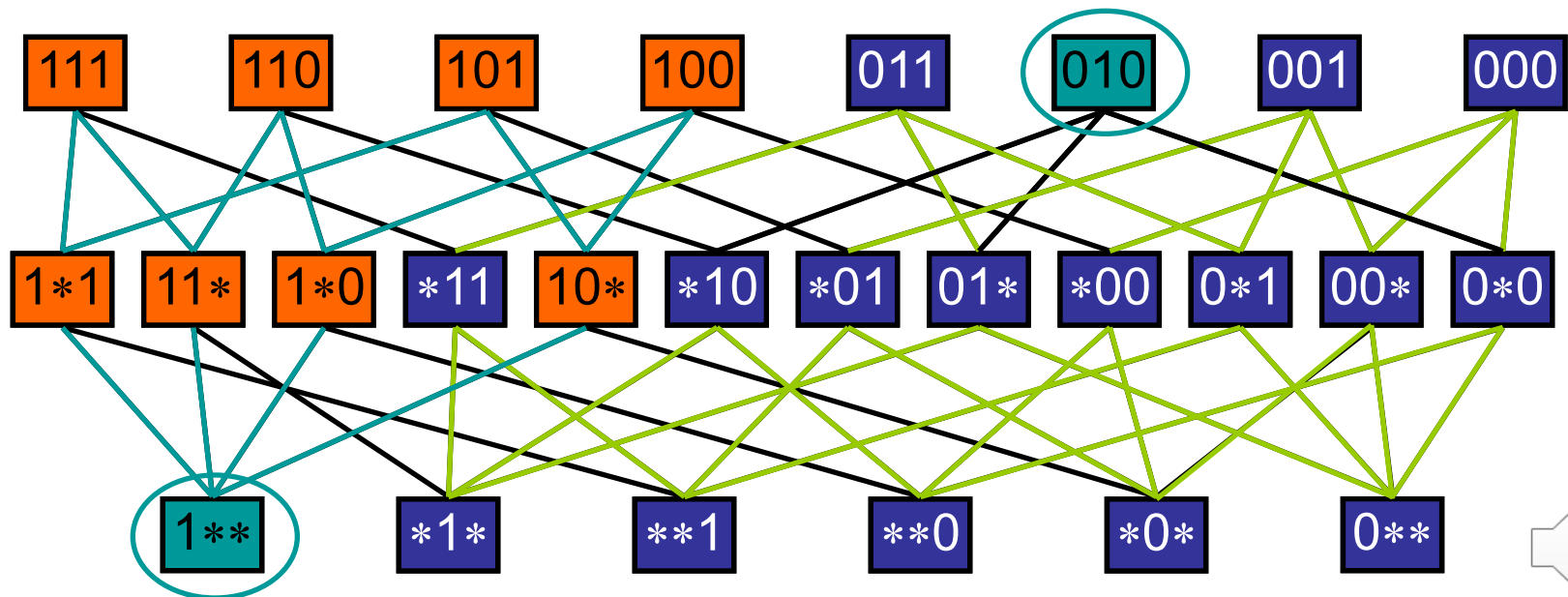
# Projected Holes Algorithm

- Build layered network of all subspaces
  - First layer: All coverage tasks individually listed.
  - Second layer: Projections applied to single elements (medium sized holes if not covered)
  - Third layer: Projections applied to two elements (largest holes if not covered)



# Projected Holes Algorithm

- Build layered network of all subspaces
- Recursively mark the ancestors of **covered tasks**
- Loop from the bottom
  - Report unmarked nodes as holes
  - Recursively mark descendants



# Coverage Progress

---

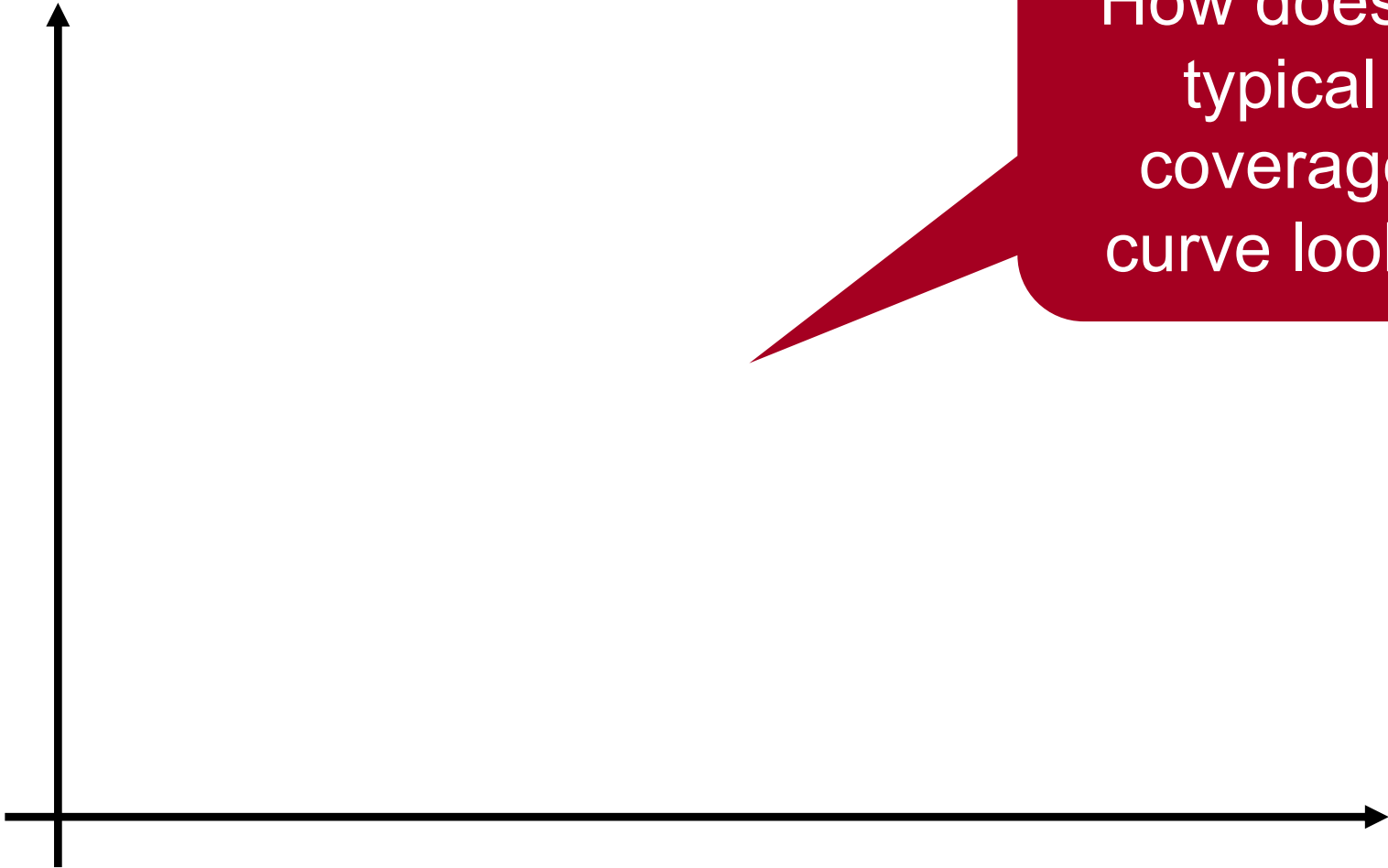
- Shows the progress of coverage over time
- Time can be measured by
  - Wall clock (or calendar) time
  - Number of tests simulated
  - Number of simulation cycles
- Can be used on the entire coverage model or specific views of it





# Coverage Progress

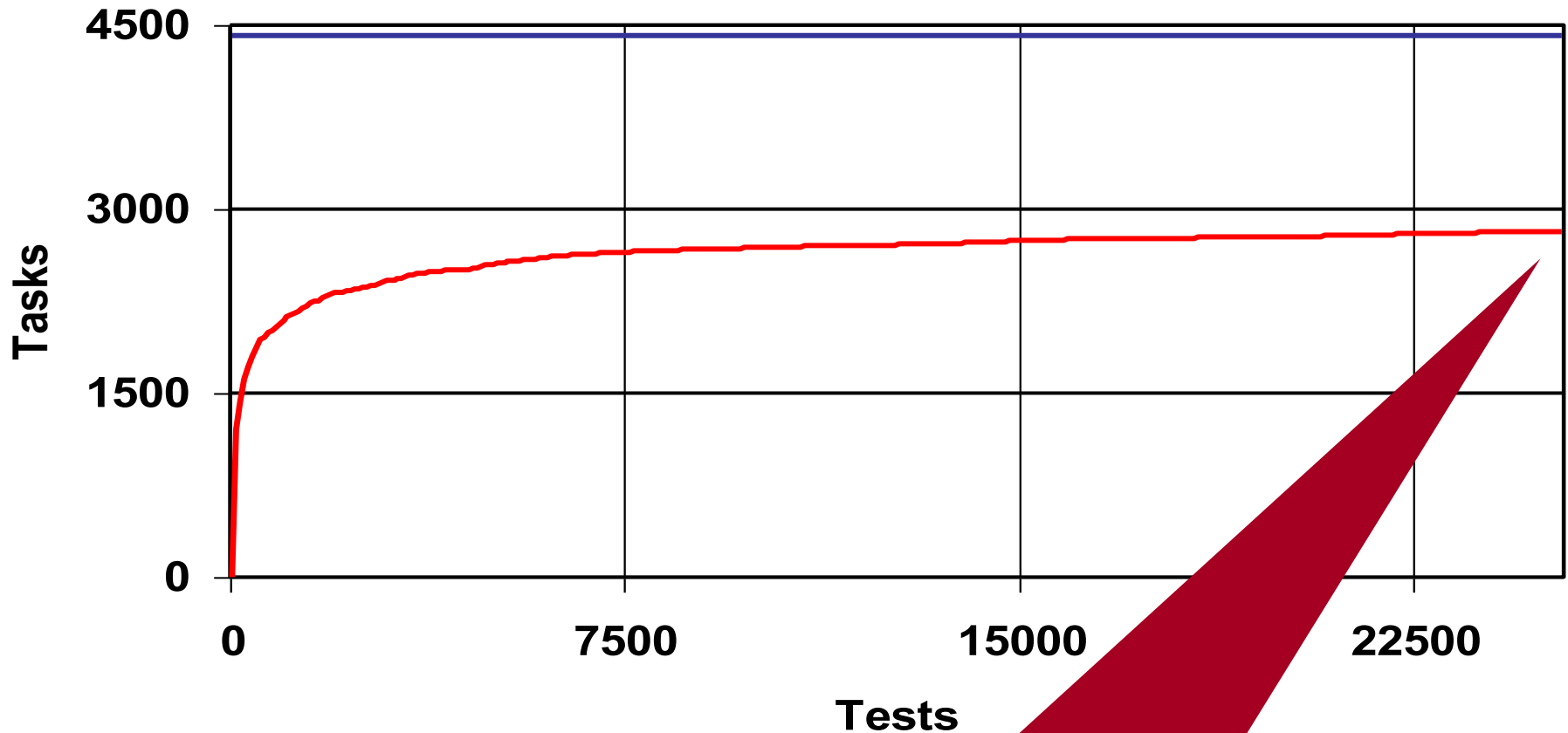
Coverage



How does a typical coverage curve look?



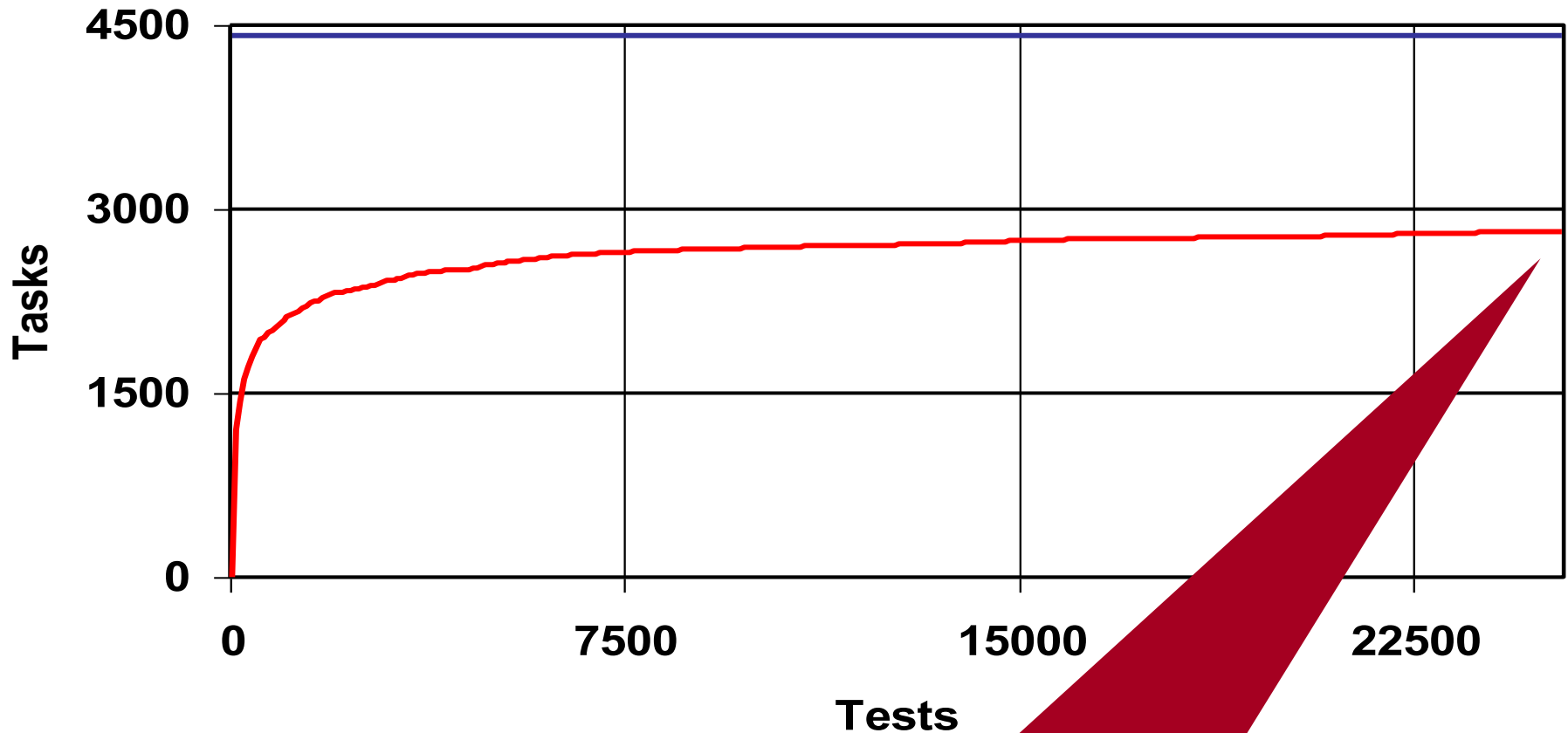
# Coverage Progress Example



After 25,000 tests 2810 / 4418 tasks were covered, a total of 64%.



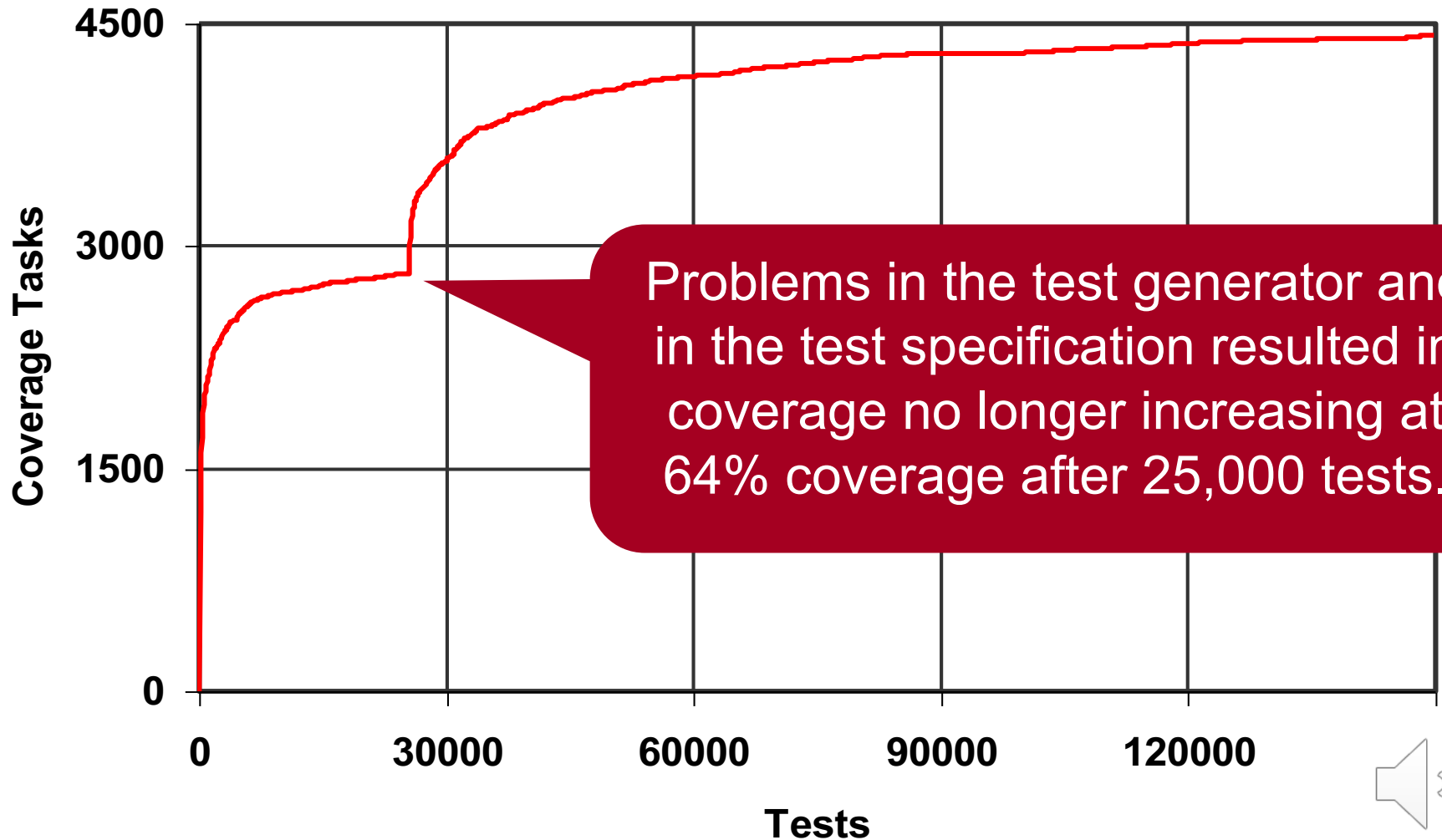
# Coverage Progress Example



After 25,000 tests 2810 / 4418 tasks were covered, a total of 64%.



# Coverage Progress Example

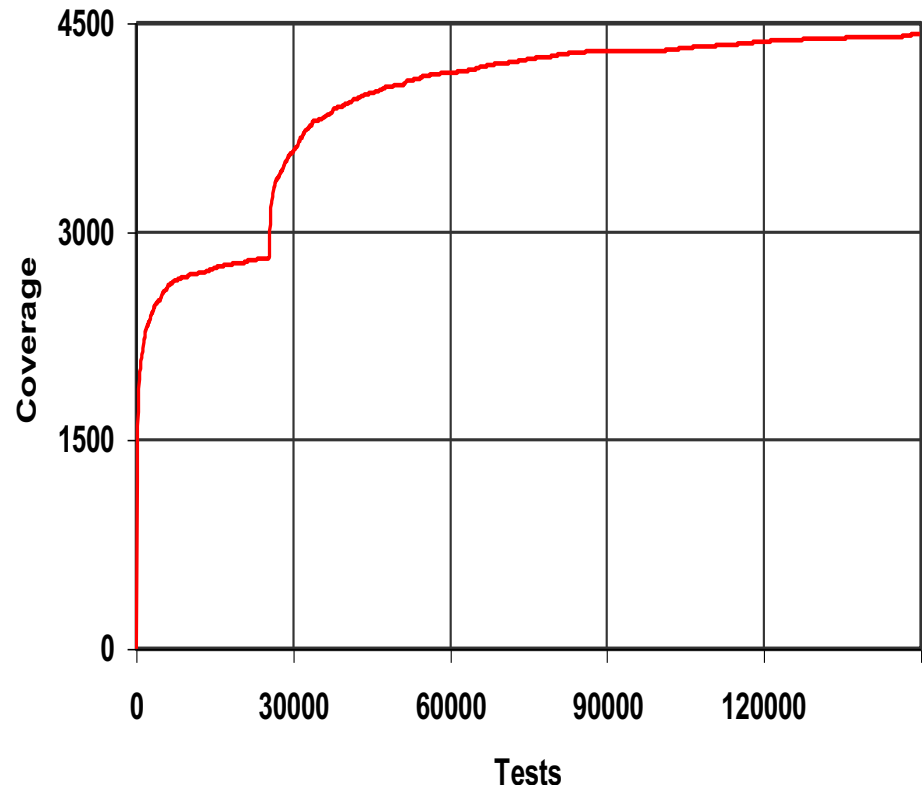


Problems in the test generator and in the test specification resulted in coverage no longer increasing at 64% coverage after 25,000 tests.



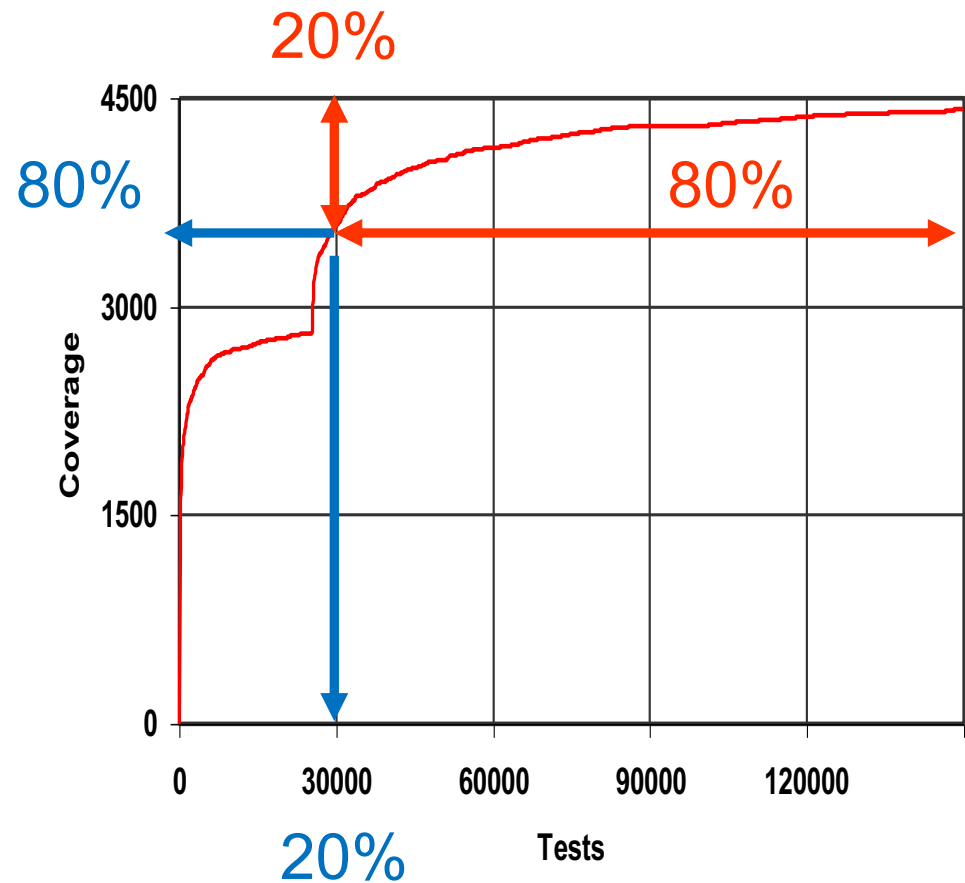
# Progress Report Usage

- Progress report can provide a lot of information
  - How well we are progressing overall
  - What is the current progress rate?
  - Are there any changes in the coverage progress rate?
  - What is the expected maximal coverage?



# Progress Report Usage

- Progress report can provide a lot of information
  - How well we are progressing overall
  - What is the current progress rate?
  - Are there any changes in the coverage progress rate?
  - What is the expected maximal coverage?
  - **When can we expect to reach our coverage target, i.e. maximal coverage?**



# Using Coverage – What can go wrong?

---

- Low coverage goals
- Collecting coverage without analyzing and interpreting the results
- Some coverage models are ill-suited to deal with common problems
  - For example, missing code won't be possible to identify using code coverage
    - Need a requirements-based methodology to overcome this!
- Generating simple tests just to cover specific uncovered tasks
  - There is merit in generating tests outside the coverage!

“Coverage is a  
measure of effort,  
not achievement.”

\*\*\* *Discuss* \*\*\*





# Summary: Coverage

- Coverage is an important verification tool.
  - **Code** coverage: statement, path, expression
    - **Structural** coverage: FSM
  - **Functional** coverage models
  - (**Assertion** coverage as discussed during the lecture on Assertion-based Verification.)
- **Coverage analysis** techniques
- In practice, several coverage models are typically used **in combination**.
  - Code coverage alone does not mean very much!
- For a verification methodology to be effective and efficient, it should be **coverage driven**.

