Fault Tolerant Computing

CS 530

Random Testing

Yashwant K. Malaiya

Colorado State University

Random Testing

- Random testing, in some form, is common for both hardware or software testing.
- It is sometimes assumed that an "average" fault can represent most faults. In reality some faults are easy to find, while some faults are very hard to find.
- For highly reliable, the real challenge is in finding hard-to-test bugs.
- "Detectability-profile" concept is introduced here.

Random Testing: Outline

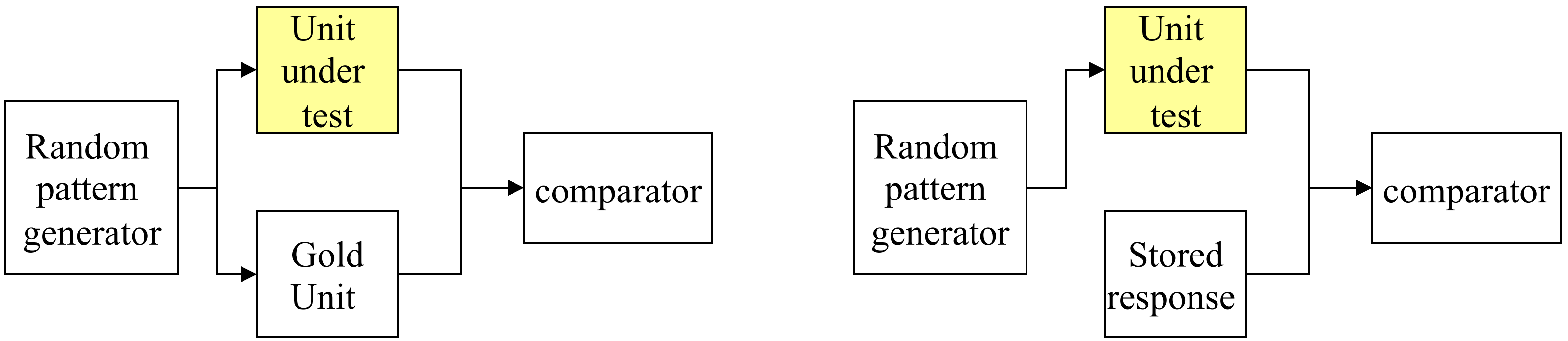
- **Random Testing (RT): advantages and tradeoffs**
- **RT vs pseudorandom testing (PR)**
- **Coverage and detectability profile (DP)**
- **Hardware and software DPs**
- **Connection between coverage and faults found?**
- **High and low testability faults during early & late testing**
- **Implications of an asymmetric DP**

Random Testing

- **Extensively used for both hardware and software**
- **Ideally each input is selected randomly. PR (Pseudorandom) schemes approximate random.**
- **Generally, quite effective for moderate coverage.**
- **Disadvantages:**
  - Coverage hard to determine a priori.
  - Ineffective for random-pattern-resistant faults.
- **Coverage tools: Random (functional) followed by Structural testing.**

Random Testing: Advantage

- **No test generation using structural information needed.**
- **Test set-up using comparison:**



- **Alternative: Is response reasonable?**

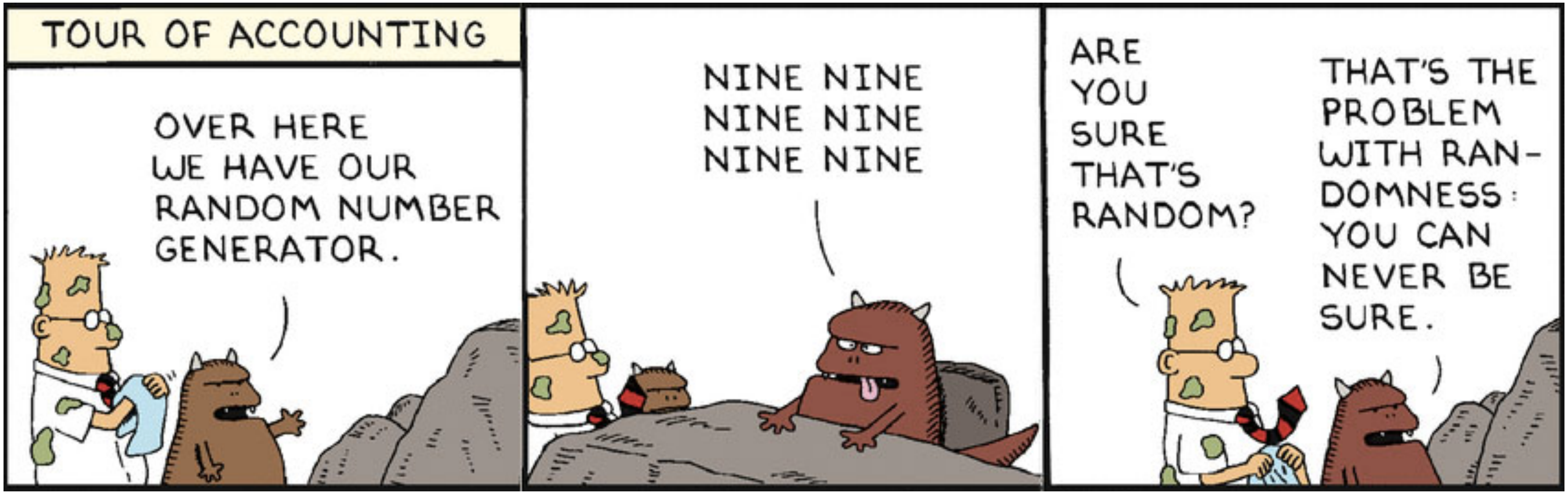
Q: For software testing, how do you know the expected response?

Pseudorandom (PR) Testing

- **Unlike true random, reproducible.**
- **Will not repeat until all combinations applied.**
- **Generation: usually just-in-time (not stored).**
  - Autonomous linear feedback shift register (ALFSR).
  - Cellular automata etc possible.
- **Some *randomness properties* satisfied, but not all.**

Ex: Set of vectors with more 1s than 0s is not quite random.

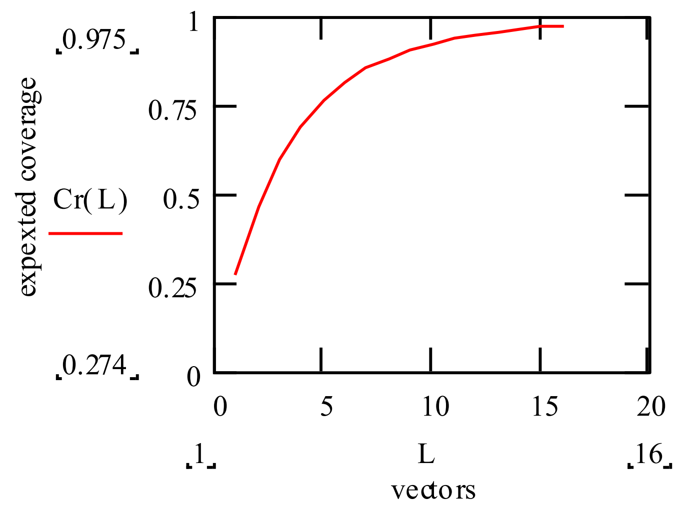
Randomness is hard to achieve



NIST: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications

Coverage Achieved vs tests

- **Coverage grows fast in the beginning, saturates near end.**
- **Is it described by**
  - $C(L) = 1 - e^{-aL}$?
  - No, doesn't fit.
- **It is controlled by distribution of detectability of faults.**
- **Detectability profile (Malaiya &Yang '84):**
- $\boldsymbol{H = \{h_1, h_2, .. h_N\}}$
  - N: total possible vectors
  - h_k: number of faults detected by exactly k vectors.

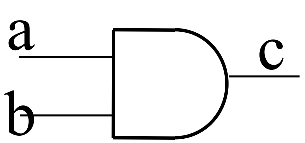


- **Total faults** $\boldsymbol{M = \Sigma h_k}$
- $\mathbf{h_1}$**: number of least testable faults**

Ex: Circuit with higher h_1 would be harder to test.

Ex: Find Detectability Profile

Example:



tests \ faults	a-0	a-1	b-0	b-1	c-0	c-1
00						√
01		√				√
10				√		√
11	√		√		√	

Minimum set

Answer: h_k: number of faults detected by exactly k vectors.
Thus h_1= 5 faults, h_3= 1 fault. Hence H = (h_1, h_3) ={5, 1}

Question: What is the probability that a random test will test for c s-a-1?

Detection Probability

- **Detection probability:** if there are N distinct possible vectors, and if a fault is detected by k of them, then its detection probability is k/N
- A fault with detection probability 1/N would be hardest to test, since it is tested by only one specific test and none other.

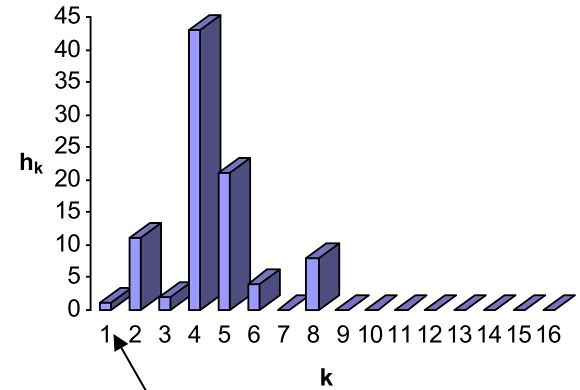
Detectability Profiles: Ex

- **CECL Full adder**

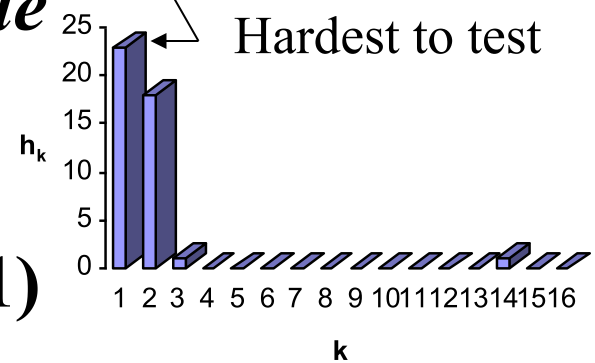
  Inputs=4 (N=16), M=90

  $H=(h_1,h_2,h_3,h_4\ ,h_5,h_6,h_8)$

  $=(1,11,2,43,21,4,8)$



Hardest to test



- ***Schneider's counterexample*** **circuit:**

  Inputs= 4 (N=16), M=44

  $H=(h_1,h_2,h_3,h_{14})=(23,19,1,1)$

Schneider's counterexample circuit has 23 hard to test faults.
A random vector has probability1/16 to detect any one of them.

Coverage with L random vectors

- h_k out of M defects detectable by exactly k vectors: detection probability k/N
- P{a defect with dp k/N not detected by a vector} = $(1-\frac{k}{N})$
- P{a defect with dp k/N not detected by L vectors} = $(1-\frac{k}{N})^L$
- Of h_k faults, expected number not covered is $(1-\frac{k}{N})^L h_k$
- Expected test coverage with L vectors

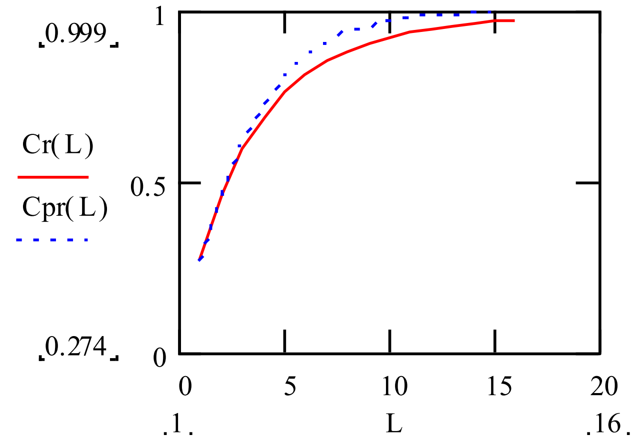
$$\mathrm{C(L)} = 1 - \sum_{k=1}^{N} (1-\frac{k}{N})^L \frac{h_k}{M}$$

Coverage Obtained by L Vectors

- For PR tests (McClusky 87)

$$C(L) = 1 - \sum_{k=1}^{N-L} \frac{{}^{N-L}C_k}{{}^{N}C_k} \frac{h_k}{M}$$

$$\approx 1 - \sum_{k=1}^{N} (1 - \frac{k}{N})^L \frac{h_k}{M} \text{ (for Random)}$$



- For large L, terms with only low k (i.e. faults that are hard to test) have an impact. Thus only lower elements of H need to be estimated.
- For CECL Full Adder,

$$C(15) = 1 - [4.2 + 16.4 + 0.9 + 6.3 + 0.84 + 0.03 + 0 + ...] \cdot 10^{-3}$$

Pseudorandom (PR): a vector cannot repeat, unlike in true Random.

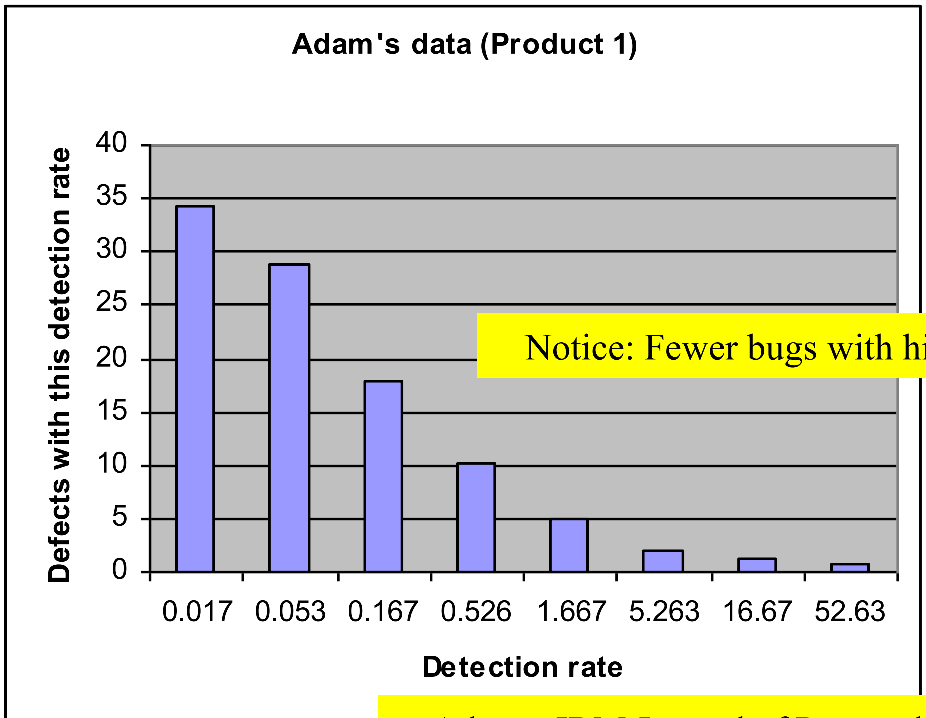
More in Appendix 1

Detectability Profile: software

- **Regardless of initial profile, after some initial testing, the profile will become asymmetric**
- **In the early development phases, inspection and early testing are likely to remove most easy to test bugs, while leaving almost all hardest to test bugs still in.**

Detectability Profile: software

- **Adam's Data for a large IBM software product**

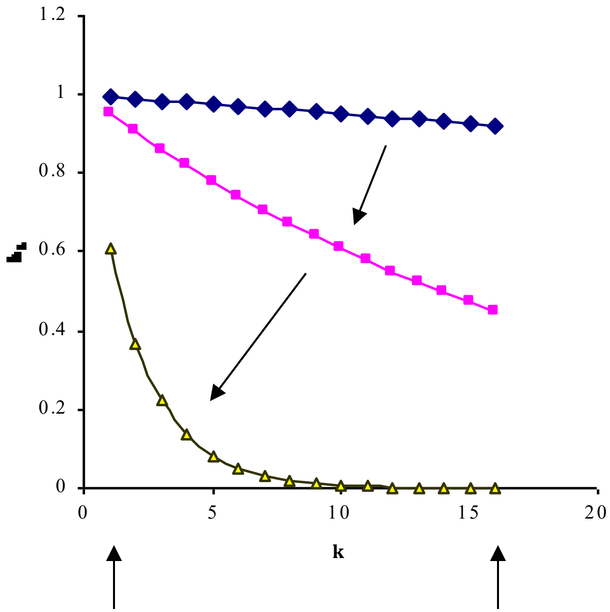


Notice: Fewer bugs with higher detection rates

Adams, IBM Journal of Research and Development, Jan. 1984

Detectability Profile: Software

- **Software detectability profile is exponential**
- **Justification: Early testing will find & remove easy-to-test faults.**
- **Testing methods need to focus on hard-to-find faults.**



Hard to test Low hanging fruit

As testing time progresses, more of the faults are clustered to the left.

Implications of Asymmetrical DP

- **Faults are not alike, and an "average" fault does not represent a hard-to-test fault.**
- **A fault injected artificially typically does not represents a hard-to-test fault.**
- **Faults found early during testing are not a good sample of faults that will be found later during testing.**

Implications of Asymmetrical DP

- **Fault seeding**
- **Fault sampling**
- **Fault exposure ratio**

Implications: Fault Seeding

- **A program has x defects. We want to estimate x.**
- **Seed j new faults.**
- **Do some testing. Let faults found be j_1 seeded faults and x_1 original faults.**
- **Assuming $j_1/j = x_1/x$ we get** $$x = x_1 \frac{j}{j_1}$$
- **However, in reality the x faults include harder faults to test,**

$$\frac{j_1}{j} > \frac{x_1}{x} \quad hence\ x > \frac{x_1 j}{j_1}$$

Implications: Estimation by Inspection Sampling

- Software with x bugs is inspected by two separate teams that finds x_1 and x_2 bugs respectively, of which x_3 are shared.
- **Assuming $x_1/x = x_3/x_2$ we get**

$$x = \frac{x_1 x_2}{x_3}$$

- **However actually since x includes more harder to test faults,**

$$\frac{x_3}{x_2} > \frac{x_1}{x} \; hence \; x > \frac{x_1 x_2}{x_3}$$

Implications: fault exposure ratio

Let N(t) be the number of bugs at time t during testing, then if a is a parameter,

$$\frac{dN(t)}{dt} = -aN(t)$$

If a is constant, then $N(t) = N(0)e^{-at}$ [expo SRGM]

However in random testing a should decline as faults get harder to find.

If testing is intelligent, then a can rise, which can give rise to Logarithmic SRGM.

Don't worry about this here, we will come to it when we will study software reliability.

References

- Y. K. Malaiya, S. Yang, "The Coverage Problem for Random Testing," IEEE International Test Conference 1984, pp. 237-245.
- Y.K. Malaiya, A. von Mayrhauser and P. Srimani, "An Examination of Fault Exposure Ratio," IEEE Trans. Software Engineering, Nov. 1993, pp. 1087-1094.
- S. C. Seth, V. D. Agrawal, H. Farhat, "A Statistical Theory of Digital Circuit Testability," IEEE Trans. Computers, 1990, pp. 582-586.
- K. Wagnor, C. Chin, and E. McCluskey, "Pseudorandom testing". IEEE Trans. Computer, Mar. 1987, pp. 332—343.
- E. N. Adams, "Optimizing Preventive Service of Software Products," in IBM Journal of Research and Development, vol. 28, no. 1, pp. 2-14, Jan. 1984.
- J R Dunham, "Experiments in software reliability: Life-critical applications," IEEE Tran. SE, January 1986, pp. 110 - 123
- H. Hashempour, F.J. Meyer, F. Lombardi,, "Analysis and measurement of fault coverage in a combined ATE and BIST environment," Instrumentation and Measurement, IEEE Transactions on , vol.53, no.2, pp.300,307, April 2004.

Appendix
Ex: Coverage for CECL adder

- 16 vectors (0000 to 1111), 90 potential defects (transistor level)
- A fault with det prob 1/16 has probability 0.0625 to be detected with 1 test, while fault with det prob 8/16 has 0.5 (i.e. 50%) of getting tested by it.
- With 20 vectors, 28% of faults with det prob 1/16 will still be left undetected, while those with det prob 8/16 will almost certainly be found.
- Table next gives the coverage obtained for faults with specific detectability values.

Ex: C(L) and components for CECL Full Adder

CECL full adder	**N =**	**16**	**M =**	**90**

Hk	**1**	**11**	**2**	**43**	**21**	**4**	**8**	
k =>	1	2	3	4	5	6	8	**Coverage**
Test Length L								
0	0	0	0	0	0	0	0	0
1	0.0625	0.1250	0.1875	0.2500	0.3125	0.3750	0.5000	0.2736
5	0.2758	0.4871	0.6459	0.7627	0.8464	0.9046	0.9688	0.7652
10	0.4755	0.7369	0.8746	0.9437	0.9764	0.9909	0.9990	0.9263
15	0.6202	0.8651	0.9556	0.9866	0.9964	0.9991	1.0000	0.9710
20	0.7249	0.9308	0.9843	0.9968	0.9994	0.9999	1.0000	0.9865

After 20 vectors:

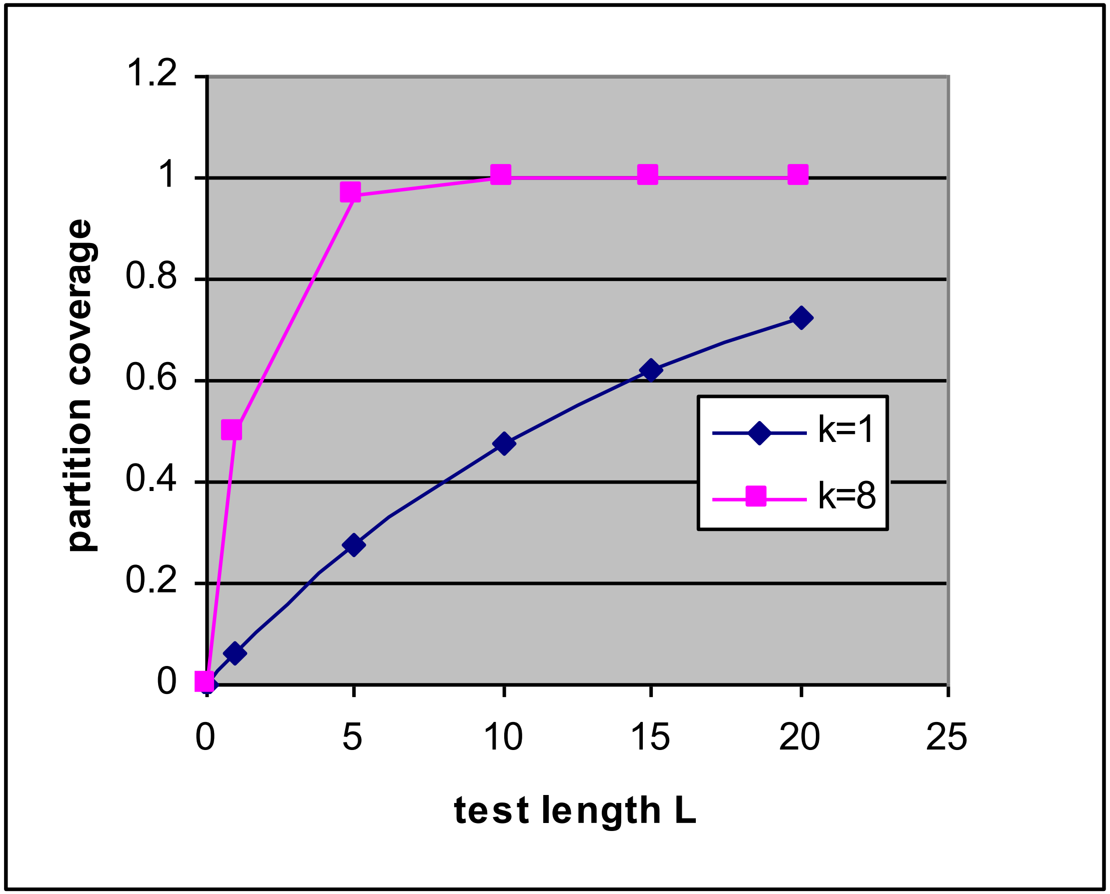
covered	0.72	10.24	1.97	42.86	20.99	4.00	8.00
remaining	0.28	0.76	0.03	0.14	0.01	0.00	0.00

Coverage of faults with different detectability values

The plot in the next slide for CECL Full Adder shows that

- Faults with high detection probability get covered soon,
- while those with low detection probability are resistant to random testing.

Coverage of partitions



Shift in profile with progress in testing

Next slide for CECL Full Adder

•Assume that a fault is removed from consideration when found

•X-axis is k (k=1 hardest to find)

•Plot shows that at the beginning there are nearly 50% of the faults with det prob k/N = 4/16.

• After 20 vectors, more than 60% of the remaining faults have det prob 2/16.

•"Low hanging fruit" get picked quicker.

Shift in profile with progress in testing