

ECE 322

Formula Sheet

Color-coded by category

1. CYCLOMATIC COMPLEXITY

McCabe Cyclomatic Complexity:

$$v(G) = e - n + 2$$

where:

- e = number of edges in control flow graph
- n = number of nodes in control flow graph
- $v(G)$ = cyclomatic complexity

Meaning: Measures the number of linearly independent paths through the program. Higher values indicate more complex code that is harder to test and maintain.

Alternative Formulas:

$$v(G) = \text{number of binary decisions} + 1$$

$$v(G) = \text{number of regions (planar graph)}$$

Meaning: Can count binary decision points or planar regions to determine complexity.

Combined System Complexity:

For a system with modules G_1, G_2, \dots, G_n :

$$v(G) = v(G_1) + v(G_2) + \dots + v(G_n)$$

where $v(G_i)$ is cyclomatic complexity of module i .

Meaning: Total system complexity is the sum of individual module complexities.

2. MUTATION TESTING

Mutation Score:

$$\text{Mutation Score} = \frac{D}{N - E}$$

where:

- D = number of dead (killed) mutants
- N = total number of mutants
- E = number of equivalent mutants

Meaning: Measures the effectiveness of a test suite. Test suite is mutation adequate if score = 100%. Typical scores: 80-90%.

Test Suite Quality:

$$\text{Quality} = \text{code coverage} + \text{mutation score} + \text{test redundancy}$$

Meaning: Overall quality combines coverage, mutation effectiveness, and redundancy reduction.

3. FAULT SEEDING (Mills)

Estimated Remaining Faults:

$$\frac{s}{S} = \frac{n}{N}$$

Solving for N :

$$N = \frac{n \cdot S}{s}$$

where:

- s = detected seeded faults
- S = total seeded faults
- n = detected non-seeded (real) faults
- N = total non-seeded (real) faults (unknown)

Meaning: Estimates total number of real faults in the system. Assumes seeded faults have same detectability as real faults.

Remaining Faults After Testing:

$$\text{Remaining} = N - n$$

Meaning: Number of faults still present after testing is complete.

4. HALSTEAD METRICS

Basic Measurements:

- n_1 = number of unique operators
- n_2 = number of unique operands
- N_1 = total occurrences of operators
- N_2 = total occurrences of operands

Program Length:

$$N = N_1 + N_2$$

Meaning: Total number of operator and operand occurrences.

Program Vocabulary:

$$n = n_1 + n_2$$

Meaning: Total number of unique operators and operands.

Program Volume:

$$V = N \log_2(n)$$

Meaning: Size of the program in bits. Represents information content.

Program Difficulty:

$$D = \frac{n_1}{2} \times \frac{N_2}{n_2}$$

Meaning: Difficulty to write/understand. Based on unique operators and total operand usage.

Program Effort:

$$E = D \times V$$

Meaning: Mental effort required to develop the program.

Implementation Time:

$$T = \frac{E}{18}$$

Meaning: Estimated time in seconds to implement. Factor 18 is empirically derived.

Estimated Faults:

$$\text{Faults} = \frac{E^{2/3}}{3000}$$

Meaning: Predicted number of faults based on effort. Empirical relationship.

5. COVERAGE CRITERIA

Statement Coverage:

$$\text{Statement Coverage} = \frac{\text{Statements Executed}}{\text{Total Statements}} \times 100\%$$

Meaning: Percentage of code statements executed by test suite.

Branch Coverage:

$$\text{Branch Coverage} = \frac{\text{Branches Taken}}{\text{Total Branches}} \times 100\%$$

Meaning: Percentage of decision branches (true/false) executed. Stronger than statement coverage.

Path Coverage:

$$\text{Path Coverage} = \frac{\text{Paths Executed}}{\text{Total Paths}} \times 100\%$$

Meaning: Percentage of all possible execution paths tested. Often impractical due to exponential growth.

6. BLACK BOX TESTING

Weak nx1 Strategy (Linear Boundaries):

For subdomain with b linear boundaries in n -dimensional space:

$$\text{Number of test points} = (n+1)b + 1$$

where:

- n = number of dimensions
- b = number of linear boundaries

Meaning: Tests boundary tilts. Each boundary needs $n+1$ linearly independent points to define hyperplane, plus 1 interior point.

Extreme Point Combination (EPC):

$$\text{Number of test points} = 4^n + 1$$

where n = number of dimensions.

Meaning: Tests corners of input domain plus center point. Grows exponentially with dimensions.

Weak nx1 for Specific Boundary:

For boundary $|x_1 - a_1| + |x_2 - a_2| + \dots + |x_n - a_n| \leq c$:

$$\text{Test points} = 2^n(n+1) + 1$$

Meaning: Manhattan distance boundary requires exponentially many points.

Total Combinatorial Tests:

For inputs with k_1, k_2, \dots, k_m possible values:

$$\text{Total tests} = k_1 \times k_2 \times \dots \times k_m$$

Meaning: Complete combinatorial explosion. Testing all combinations.

Weak Normal Equivalence Strategy:

$$\text{Number of tests} = \text{number of equivalence classes}$$

Meaning: One test per equivalence class. Single fault assumption.

Strong Normal Equivalence Strategy:

For m parameters with k_i equivalence classes each:

$$\text{Number of tests} = k_1 \times k_2 \times \dots \times k_m$$

Meaning: Tests all combinations of equivalence classes. Multiple fault assumption.

7. FINITE STATE MACHINES

Stationary Probability Testing Order:

For FSM with states and transition probabilities, solve:

$$p_i = \sum_j P_{ji} \cdot p_j$$

with constraint:

$$\sum_i p_i = 1$$

where:

- p_i = stationary probability of state i
- P_{ji} = transition probability from state j to state i

Meaning: Test states in order of decreasing stationary probability. States with higher probability are more likely to be visited.

Example System of Equations:

For state A: $p_A = 0.4p_A + 0.3p_B + 0.7p_F$

Test order: Start with highest p_i .

8. INTEGRATION TESTING

Module Design Complexity:

$$iv(G_i) = \text{cyclomatic complexity of reduced graph for module } i$$

Meaning: Complexity after removing control structures not involved with module calls.

Integration Complexity:

For n modules with design complexities $iv(G_1), \dots, iv(G_n)$:

$$S = \sum_{i=1}^n iv(G_i) - n + 1$$

where:

- S = integration complexity
- $iv(G_i)$ = module design complexity
- n = number of modules

Meaning: Number of independent integration tests required. Accounts for module interactions.

Number of Paths Through Loops:

Sequential loops (Fig a): 5^p paths (where p = repetitions)

Parallel loops (Fig b): 3^r paths (where r = repetitions)

For $5^p < 3^r$:

$$r > p \cdot \frac{\log(5)}{\log(3)}$$

Meaning: Parallel structure has fewer paths for same repetition depth.

9. EQUIVALENCE PARTITIONING

Random Testing Coverage Probability:

For 2D unit square $X = [0, 1] \times [0, 1]$ with:

- $A_1 = \{(x_1, x_2) | x_1 < a, x_2 < a\}$, where $0 < a < 1$
- $A_2 = X - A_1$

$$P(\text{both classes covered}) = 1 - 2(1 - a^2) + (1 - a^2)^2$$

For n -dimensional hypercube with $A_1 = \{x | x_i < a \text{ for all } i\}$:

$$P(\text{coverage}) = 1 - (1 - a^n)^k - k \cdot a^n \cdot (1 - a^n)^{k-1}$$

where k = number of random tests.

Meaning: Probability decreases exponentially with dimensions (curse of dimensionality).

10. COLLINEARITY TEST

Distance Formula:

Distance between points $A(x_1, y_1)$ and $B(x_2, y_2)$:

$$AB = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Collinearity Conditions:

Points A, B, C are collinear if:

$$AB = AC + BC$$

or

$$AC = AB + BC$$

or

$$BC = AB + AC$$

Meaning: One distance equals sum of other two. Used for geometric boundary testing.

11. LINEAR INDEPENDENCE

Path Independence:

Paths are linearly independent if rank of their vector representation matrix equals number of paths:

$$\text{rank}(M) = \text{number of paths}$$

where M is matrix with path vectors as rows.

Meaning: Independent paths provide unique test coverage. Use `numpy.linalg.matrix_rank()` to compute.

12. TRACEABILITY RISK

Risk of Change:

$$\text{Risk} = f(\text{in-degree, out-degree, complexity})$$

where:

- in-degree = number of incoming dependencies
- out-degree = number of outgoing dependencies
- complexity = cyclomatic complexity before/after change

Meaning: Modules with high connectivity and complexity are riskier to change.

13. DECISION TABLES

Partition Testing with Disks:

For equivalence classes A_1, A_2, A_3 (open disks) and A_4 =

$$X = A_1 - A_2 - A_3:$$

Test flags: $F_i : \text{dist}(v_i, v_j) \geq r_i + r_j$ for all pairs

Partition exists when all flags are true.

Meaning: Decision table checks if equivalence classes form valid partition (disjoint, cover space).

14. MODIFIED CONDITION/BRANCH

MC/DC Coverage:

For each condition in compound decision:

- Condition must independently affect outcome
- All conditions take both T and F
- Each condition shown to independently affect result

Minimum tests: $n + 1$ where n = number of conditions

Meaning: Ensures each Boolean condition independently affects decision outcome. Critical for safety-critical systems.

KEY RELATIONSHIPS & HIERARCHIES

Coverage Hierarchy (weakest to strongest):

$$\text{Statement} \subset \text{Branch} \subset \text{MC/DC} \subset \text{Path}$$

Data Flow Coverage:

All-uses subsumes branch coverage

Mutation Testing Effectiveness:

80-90% is typical good score; 100% very difficult

Essential vs. Cyclomatic Complexity:

Essential complexity $ev(G)$ detects structural problems; can increase abruptly with changes

Cyclomatic complexity $v(G)$ changes gradually

Testing Strategy Selection:

Top-down: Unstable environment, interface uncertainty

Bottom-up: Stable utilities first, build upward

2. **Confidence Level (C)** Probability that the software has $\leq N$ faults, given that testing found S seeded faults and n real faults.

Mills Confidence Formula 1 (Basic)

$$C = \begin{cases} \frac{S}{S+N+1} & \text{if } n \leq N \\ 1 & \text{if } n > N \end{cases}$$

Mills Confidence Formula 2 (Hypergeometric)

Used when we want to calculate the probability exactly based on finding all seeded faults:

$$C = \frac{\binom{S}{s-1}}{\binom{S+N+1}{N+s}}$$

C. Capture-Recapture (Two Independent Groups)

Used when two independent test teams (1 and 2) test the same software.

- N_1 : Faults found by Team 1
- N_2 : Faults found by Team 2
- N_{12} : Faults found by **both** teams (overlap)

Total Estimated Faults:

$$N = \frac{N_1 \times N_2}{N_{12}}$$

1. HP / Traditional Formula

$$MI = 171 - 5.2 \ln(V) - 0.23(CC) - 16.2 \ln(LOC) + 50\sqrt{2.46 \times perCOM}$$

- V : Halstead Volume
- CC : Cyclomatic Complexity
- LOC : Lines of Code
- $perCOM$: Percentage of lines that are comments

Scale: > 85 (Highly Maintainable), $65 - 85$ (Moderate), < 65 (Difficult)