# Notes from Instructor

# Exam Topics

Part I: Preliminaries

1. Software Testing
2. Mathematical

Part II: Test Generation

3. Domain Partitioning

4. Predicate Analysis

Peer Reviews

SQA

# Problem Types

Statement, decision, condition coverage

McCabe cyclomatic complexity

Abstract syntax trees

Control flow graphs

Equivalence class partitioning

Boundary value analysis

Cause-and-effect graphs (decision tables)

Homework Review

A01

A02

Statement Domain - give each line a number, cross out braces, else, end

Statement Coverage

Decision Domain

Decision Coverage - if, while statements

Condition Domain

Condition Coverage - 2 conditions for each if decision, 1 condition for while

A03

GFG

V(G) = E – N + 2(1):

Edge - if at the end of a while, gives 2 edges still

Node - start and exit count as nodes

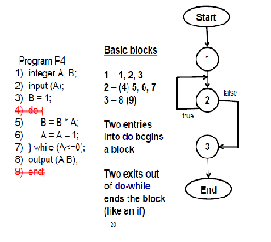
For a structured program, decision count rule:

V(G) = #decisions + 1

A05

AST, precedence starts low and moves up,

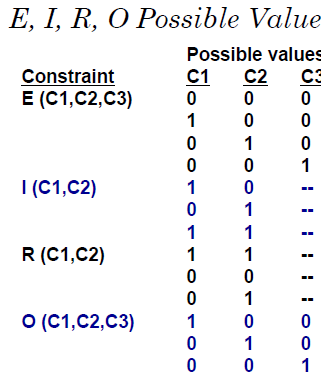
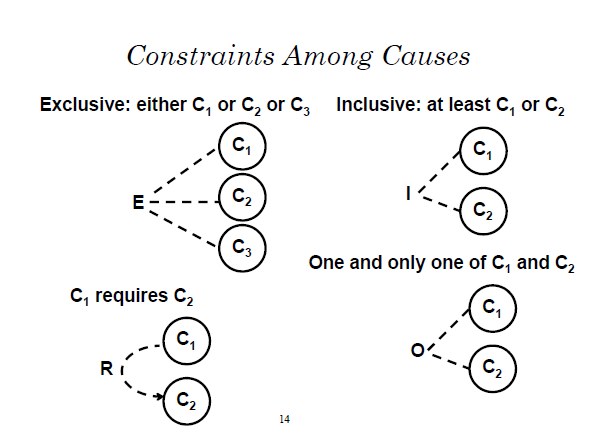
CEG, remove syntactical markers, give each line a number, basic blocks have 1 entry 1 exit, do while have two entries, two exits



A06

A07

Boolean expression doesn’t need to address constraints, R, E, O, I



Decision table, map EF1, strike out 0 effect, strike out constraints, map out EF2, strike out 0 effects, strike out constraints, combine remaining vectors from EF1 and EF2 table. transpose

# PowerPoint Summaries

# Admin

## Summary Slide:

# Prelim

## Summary Slide:

Purpose of testing

Testing vs correctness

Fault tolerance terminology – subtle distinctions

Black box vs white box

Kinds of testing – unit, integration, system,

acceptance, regression

Conflict of interest for developers

# Debug

## Summary Slide:

# Test Generation

## Summary Slide:

Early testing removes defects cheaper

Defect clustering aka Pareto principle – most

defects are in a small part of the code

Saturation region aka pesticide paradox – using

the same testing criterion reaches a point of

diminishing returns

Absence of errors fallacy – if you aren’t building

the right thing, building it right is a waste

# Complexity

## Summary Slide:

## Definitions

LOC - includes comments

SLOC - excludes comments

includes headers, declarations, executable and non-executable statements

Coupling

Cohesion

Directly Connected

Indirectly Connected

Fan-in, fan-out

## Goal Driven Measurement

Two key questions, are we measuring the right thing? Are we measuring it right?

### Goal/ Question / Metric paradigm

Answers are we measuring the right thing

Business objectives <-> data

Back and forth from either side of the EQ

### Operational Definitions

These definitions answer are we measuring it right?

Communication - do others know what is been measured?

Repeatability - would others get the same results from these measurements?

## Defects

### Casual Factors

Difficulty of problem

Complexity

programmer skill

design methods

### Explanatory variables

Size

Complexity

OO Structure

Code Churn

Process changes

## McCabe’s Cyclomatic Complexity.

V(G) = E – N + 2p

# Math

## Summary Slide:

Boolean algebra

DNF and CNF

singular and mutually singular

Drawing ASTs

Drawing CFGs

Complete and feasible paths

Structured program theorem

# Domain Partitioning

## Summary Slide:

Equivalence class partitioning

Boundary value analysis

## Equivalence class partitioning

Subdivide input domain

Small number of domains

Partitions, are disjoint

Subdivisions are the EQ classes

Assume similar behavior for all P

E legal, U illegal

Black box

## Boundary value analysis

Edge of EQ class

# Software Quality Analysis

## Summary Slide:

QA is not testing (even if frequently implemented

that way)

process assurance vs product assurance

3 different ways to resolve a nonconformance

# Peer Reviews

## Summary Slide:

3:1 ROI for walkthroughs, 5:1 ROI for inspections

Peer reviews early in the life cycle maximize ROI

Four roles in inspections

Rules for doing inspections right

26

# Predicates

## Summary Slide:

Singular, mutually singular, DNF

BOR (n+2) and BRO (2n+3) test generation

singular predicates

MI test generation

nonsingular DNF predicates

BOR-MI test generation

smaller test sets, more powerful than MI

# CEG

## Summary Slide:

Cause-effect graphs

notation (implies, not, and, or)

constraints (E, I, R, O)

masking

CEG decision tables