SOEN 384
Management, Measurement and Quality Control

http://users.encs.concordia.ca/~s384_2/

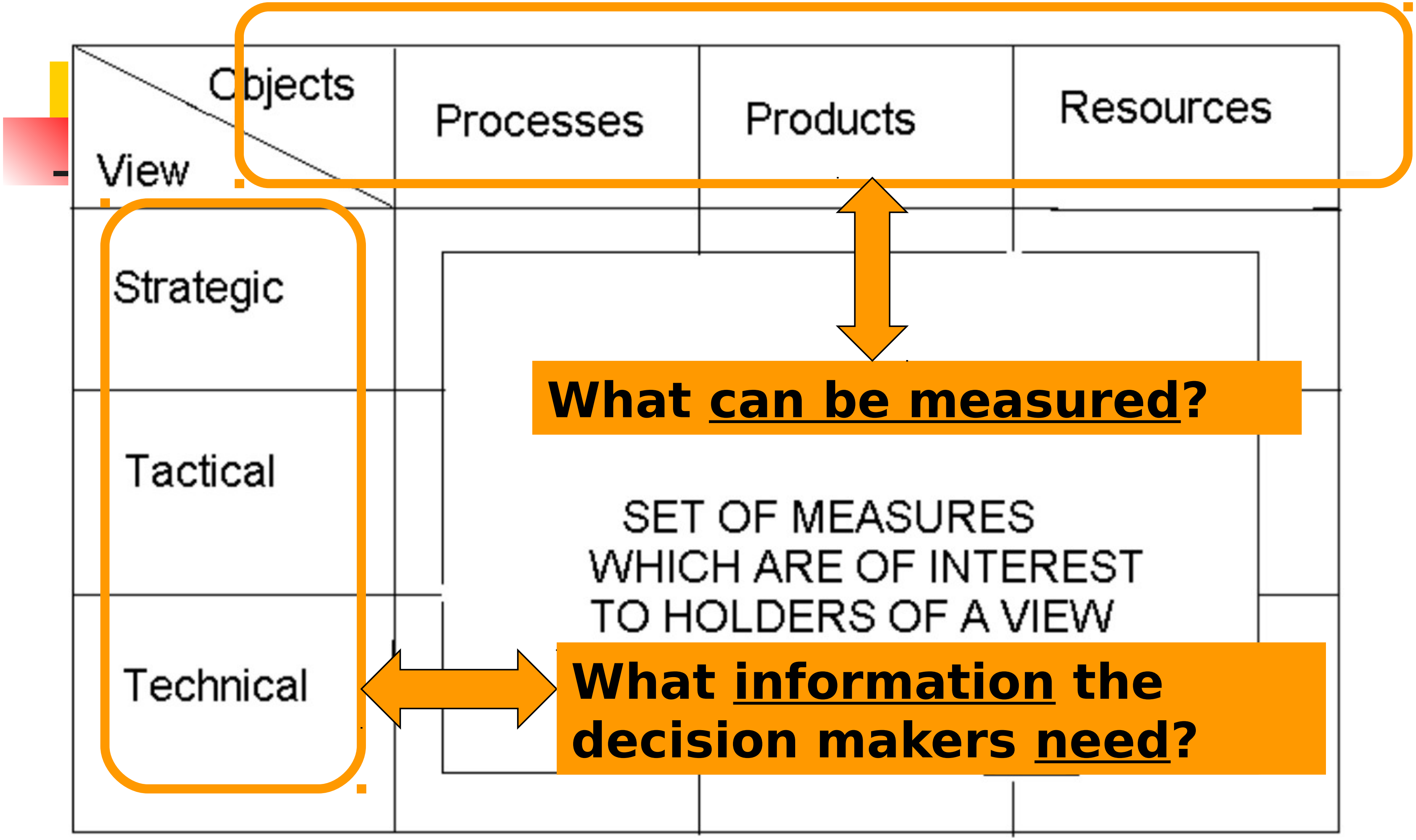
Lecture 9:

McCabe Cyclomatic Complexity, Essential Complexity

Agenda

- **Review**
- Quality modeling – OO code quality
- More classical metrics: McCabe metrics
  - Cyclomatic complexity
  - Essential complexity
- Next?

A goal-based framework for software measurement



Simple measurement plan – tactical view
(source: Sylvie Trudel)

- Small Canadian software development organisation
- “**Not to exceed**” estimate business model, guarantees that fixing all defects found by their customer are free of charge.
- **Motivation for a measurement program**: the inaccuracy of initial estimates (half of the projects ended up exceeding estimates)
- Measurement results were used to improve the accuracy of estimation models
- With more accurate estimates, several sound business decisions were made regarding future projects

Simple measurement plan (source: Sylvie Trudel)

ID	Objectives	Reason
G1	Deliver projects within effort estimates	To reach corporate goal of 30% gross margin.
G2	Deliver defect free versions into production	Ensure product quality and customer satisfaction, minimise rework.

Simple measurement plan: Questions & Indicators (source: Sylvie Trudel)

Q1	Q2	Q3	Q4
For each project, what is the difference between actual effort and planned effort?	What project proportion has an overrun > 5%?	What are the differences between the planned effort and the initial Scrum detailed estimate?	How many defects do we have per year and per release?
(Actual effort - (planned effort + CRs))*100 /(planned effort + CRs)	(Number of projects of overrun>+5%)*100/ total number of projects	Planned effort – Scrum initial effort	Number of defects per release and total
G1	G1	G1	G2
- Verify that the process was applied, especially on CRs. - Verify any encountered issue.	When > 15%, adjust estimation model	- Re-estimate either plan or Scrum. - If appropriate, advise customer of an estimate change prior to beginning project.	When > 1, do a retrospective.

Simple measurement plan:

ID	M1	M2	M3	M4
Measures	Actual effort	Planned effort	Total effort for all CRs	Scrum initial effort
Scope	Per project	Per project	Per project	Per project
U of M	Hours	Hours	Hours	Hours
Precision	1 hr	1 hr	1 hr	1 hr
Measured by	Employees	PM	PM	Employees
Data source	Anatime	Project plan	CR files	Scrum Works
Data collection procedure	Timesheet must be entered every day	Project < 50 hrs = manual only Project > 50 hrs = FSM	As soon as a CR is approved, enter it in the CR Follow-up table in the project plan.	As soon as Scrum initial effort is completed, the PM copies the effort value in the project portfolio file.

Simple measurement plan: Goal 2

- See “Quality-Management-SylvieTrudel.xls” file

Absolutely necessary project measurements (short list)

- **Product size**
  - SLOC. Distinguish between new and reused or automatically generated code.
  - Function points (FPA, COSMIC)
- **Number of defects.**
- **Quality.**
  - OO code quality, Defect density, reliability, performance, security, safety, maintainability, …
- **Effort.**
  - Person-Month. Basic monitoring parameter to assure that you stay within budget
- **Schedule and time.**
  - budget adherence, earned value, etc. Iteration completion must be lined up with defined quality criteria to avoid poor quality being detected too late
- **Project progress.**
  - evaluate how results (such as implemented and tested requirements or closed work packages) relate to the effort spent and elapsed time.

Agenda

- Review
- **Quality modeling – OO code quality**
- More classical metrics: McCabe metrics
  - Cyclomatic complexity
  - Essential complexity
- Next?

SWEBOK
Quality definition

- **"Quality is defined in terms of pertinent attributes of the specific project and any associated product(s), perhaps in both quantitative and qualitative terms. "**
  - quality characteristics will have been determined in the specification of detailed software requirements
  - Thresholds for adherence to quality are set for each indicator

How to Model Quality?

- Choose FACTORS: a set of quality attributes important for a given product from an external point of view
- decide on their decomposition into quantifiable CRITERIA from the internal point of view
- agree on specific MEASURES for the criteria and specific relationships between them

Factor – Criteria –Metrics Software Quality **hierarchical model**

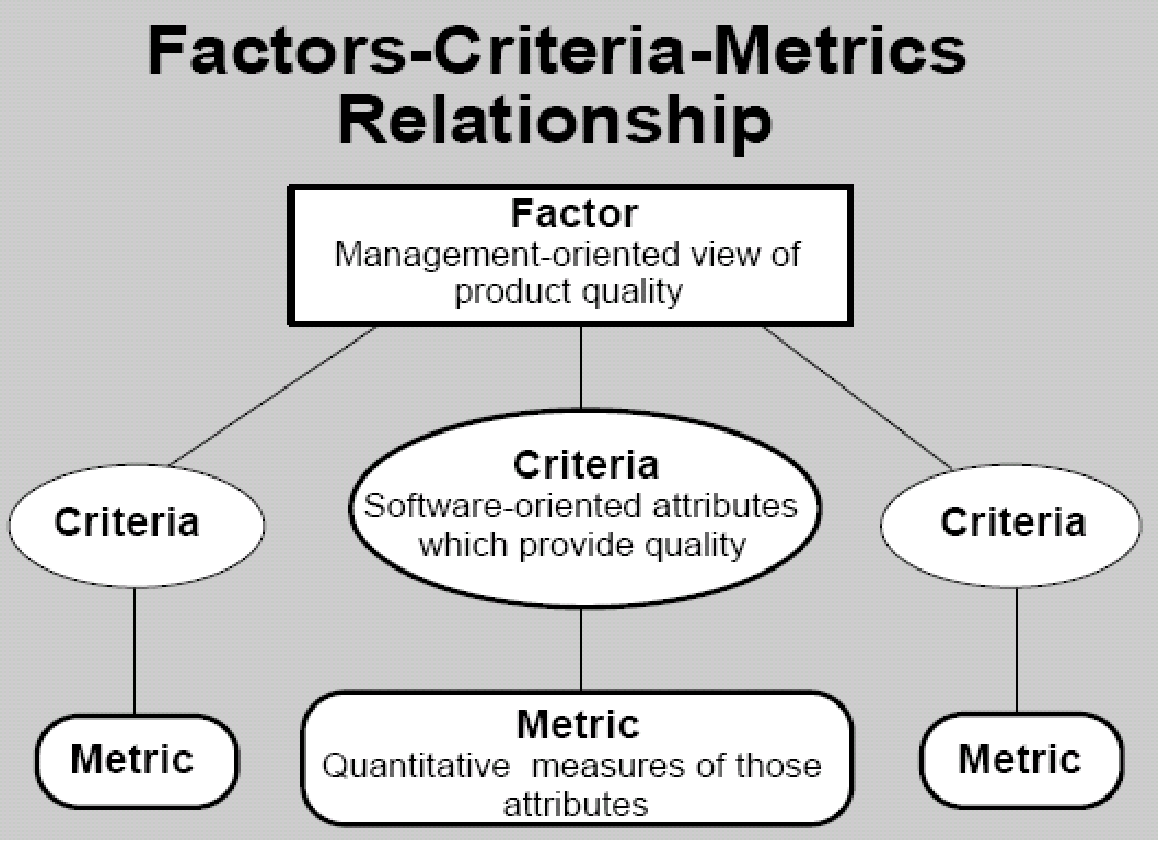
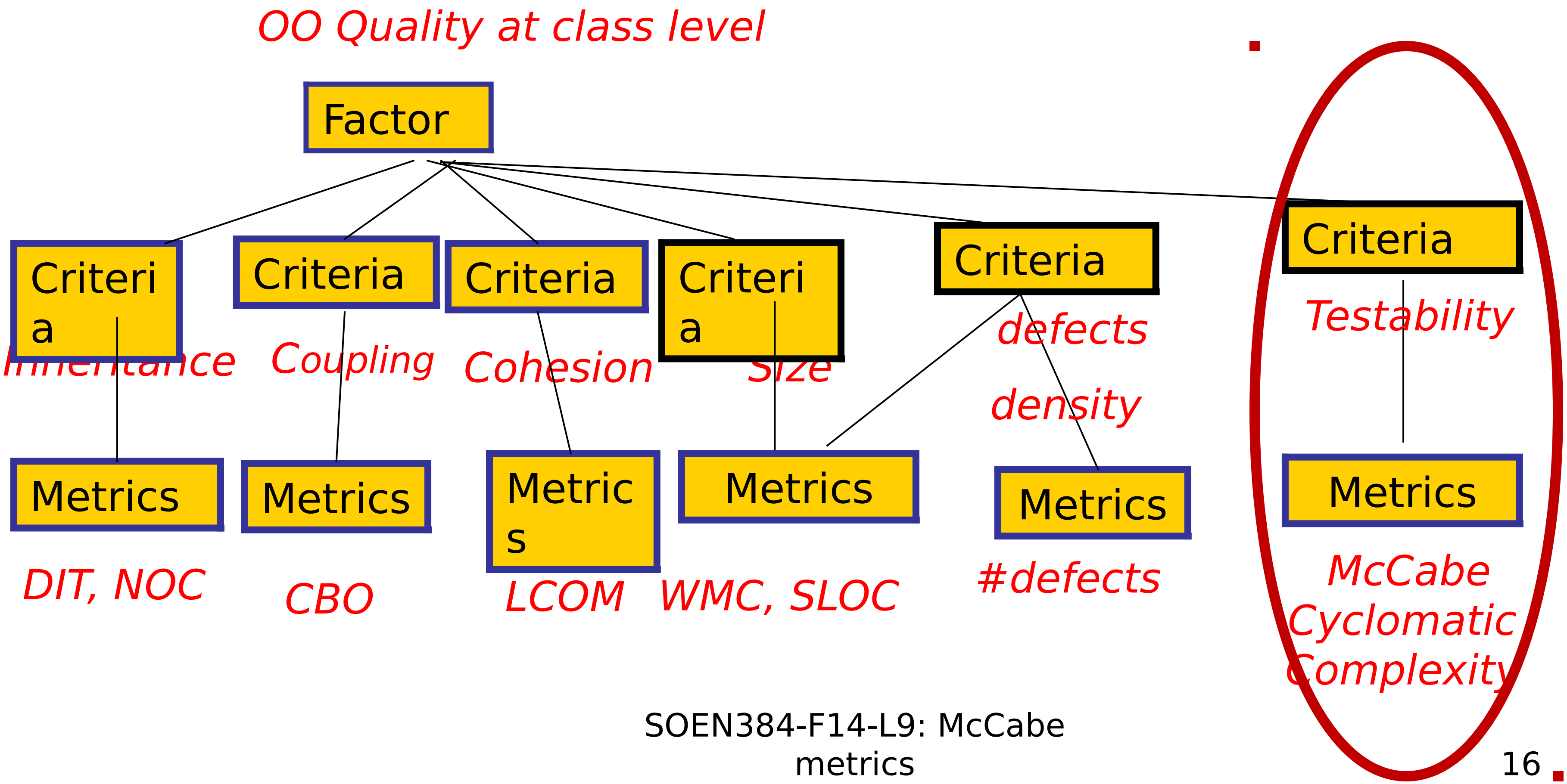


Figure 1: This hierarchical model has been the process to define program-specific metrics for the past 20 years.

Sample OO Quality Model

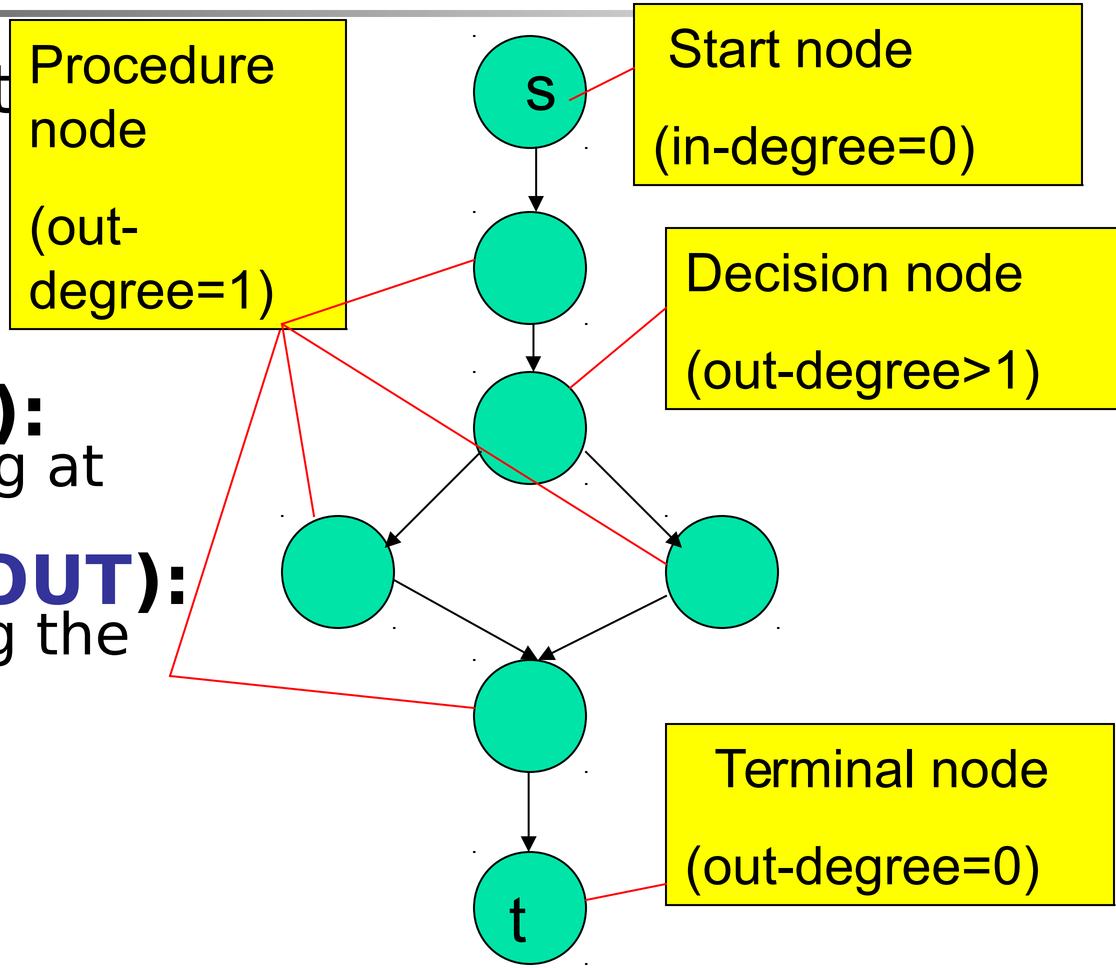


Agenda

- Review
- Quality modeling – OO code quality
- **More classical metrics: McCabe metrics**
  - Cyclomatic complexity
  - Essential complexity
- Next?

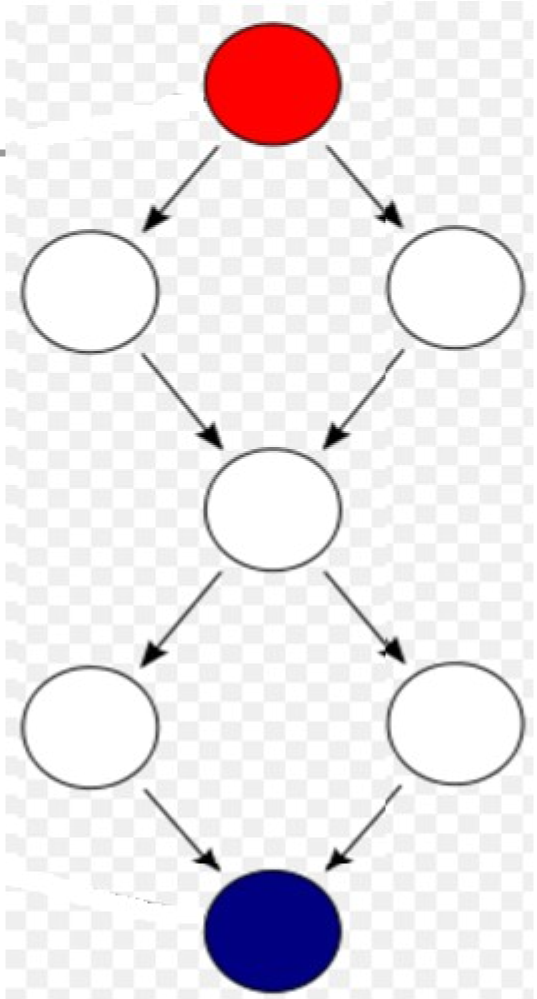
Find a graph: Control-Flow Graph (Flowgraph)

- Quadruple (E, N, s, t
  - **N :** set of nodes
  - **s:** Start node
  - **t:** Terminal node
  - **E :** set of edges
- **In-degree (FAN-IN):** number of edges arriving at node
- **Out-degree (FAN-OUT):** number of edges leaving the node
- **Path: sequence of consecutive edges**

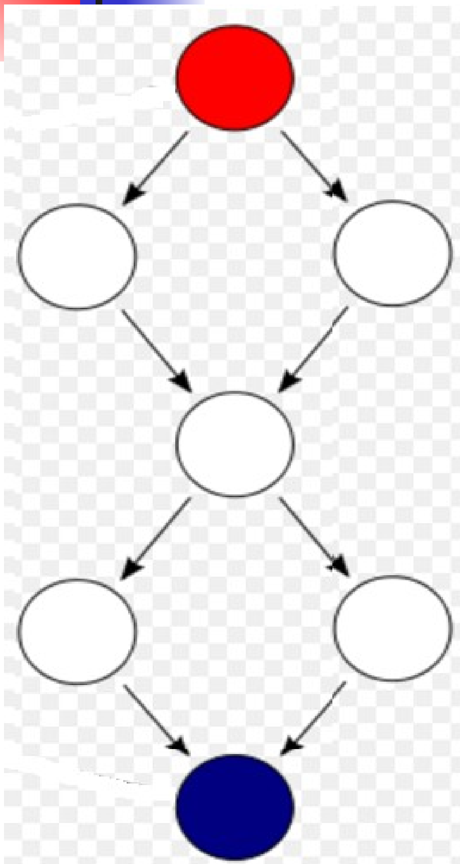


CFG Example

```
if( c1() )
    f1();
else
    f2();

if( c2() )
    f3();
else
    f4();
```



McCabe's Cyclomatic Complexity



If G is the control flowgraph of program P and G has e edges and n nodes

$$\boxed{v(P) = e - n + 2}$$

v(P) is the number of linearly independent paths in G

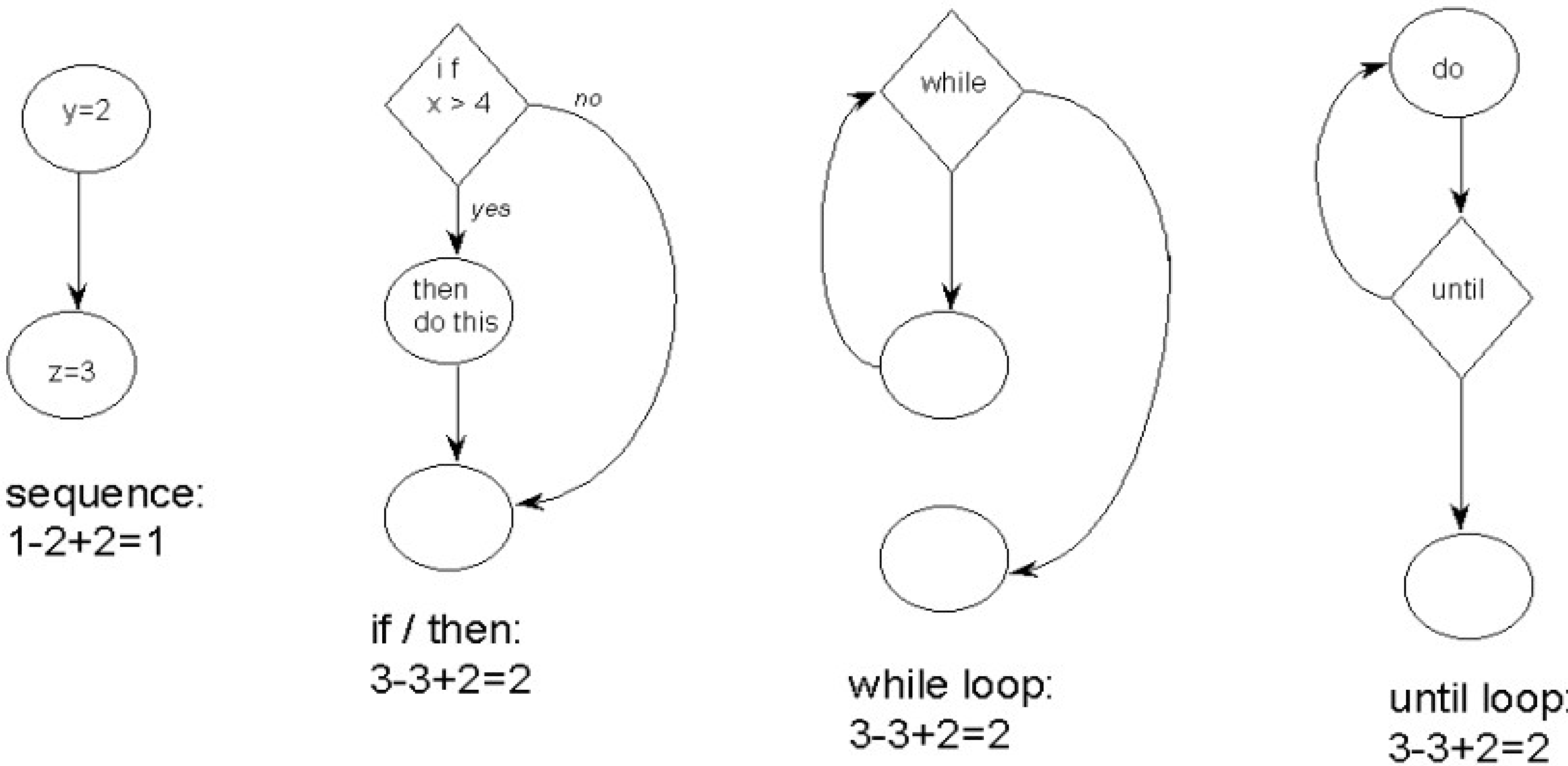
here, e = 16, n=13, and v(P) = 5

More simply, if d is the number of decision nodes in G then

$$\boxed{v(P) = d + 1}$$

McCabe proposed: v(P) < 10 for each module P

McCabe cyclomatic complexity of the basic constructs



Example

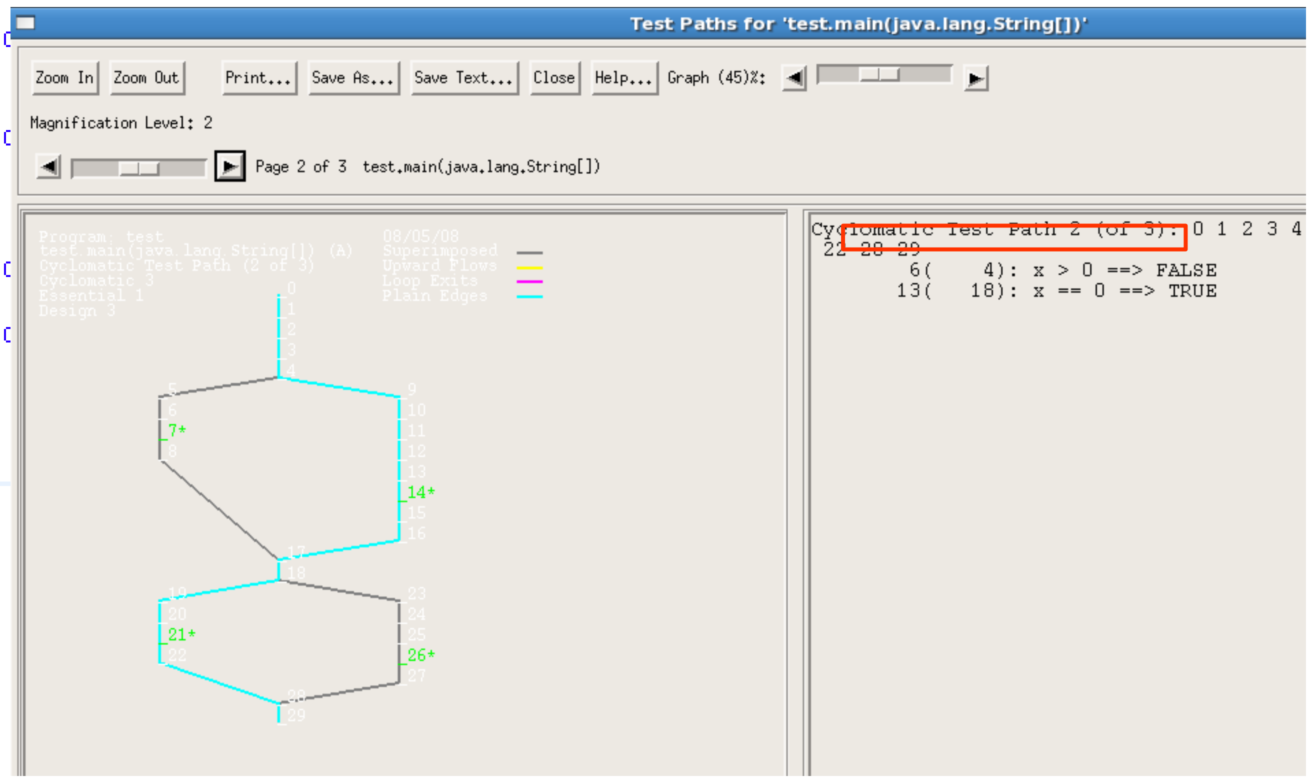
- In class

```
if (x>0)
    System.out.println("x>0 true branch");
else{
    x=2;
    System.out.println("x>0 false branch");
}

if (x==0)
    System.out.println("x=0 true branch");
else
    System.out.println("x=0 false branch");
 }
```

Infeasible Paths: McCabe Example

test.java

```
public class test {

    public static void main(String[] args) {
        int x=0;

        if (x>0)
            System.out.println("x>(
        else{
            x=2;
            System.out.println("x>(
        }

        if (x==0)
            System.out.println("x=(
        else
            System.out.println("x=(

    }

}
```



McCabe interpretation:

Cyclomatic Complexity	
Cyclomatic Complexity	**Risk Evaluation**
1-10	a simple program, without much risk
11-20	more complex, moderate risk
21-50	complex, high risk program
greater than 50	untestable program (very high risk)

Order: P3 “is more complex than”

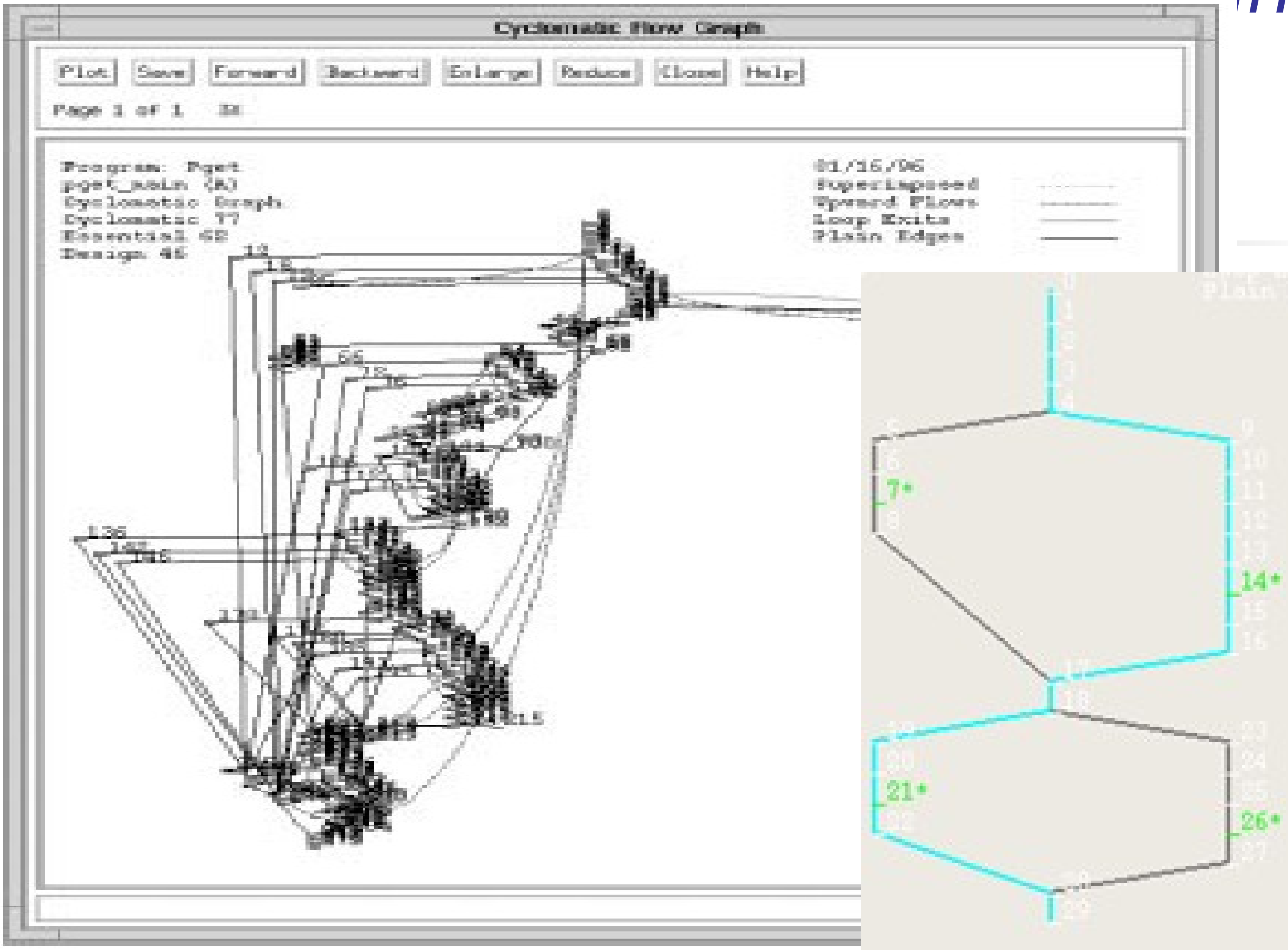


Figure 4-9. Module with complexity 77.

Agenda

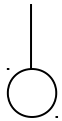
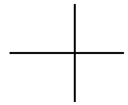
- McCabe cyclomatic complexity metric
- Does McCabe cyclomatic complexity measure **COMPLEXITY**?
- McCabe essential complexity metric

McCabe measurement validation as a measure of the attribute complexity

- Complexity elementary changes (P1, P2 are flowgraphs):
- **e1:** if P2 results from P1 by inserting an edge, then P2 is more complex than P1.
- **e2:** if P2 results from P1 by inserting an edge and a node, then P1 and P2 are of equivalent "complexity"
- **e3:** if P2 results from P1 by transferring an edge from one arbitrary location to another location, then P1 and P2 are of equivalent "complexity"

McCabe Example (Cont.)

The McCabe measure is internally validated on the ordinal scale as a complexity attribute measure if the following holds:

P1 -> P2	P1 -> P2	P1 -> P2
		
Adding an Edge	*Adding an Edge and a Node*	*Transferring an Edge*

McCabe:

v(P2)>v(P1)	=	=

McCabe Cyclomatic Complexity: is it measuring complexity?

- Complexity's Elementary Changes : **e1, e2, e3** were validated successfully
  - **Enough?**
- **e4:** if P2 results from P1 by nesting P3, and P4 results from P1 by sequencing P3, then P2 is more complex than P4.
  - **Validate e4 (in class)**
- **Conclusions?**
  - **McCabe is NOT a valid measure of complexity attribute.**
  - **McCabe = # of linearly independent paths**
  - **indicates the min# of test cases to cover linearly independent paths**

Agenda

- McCabe cyclomatic complexity metric
- Does McCabe cyclomatic complexity measure COMPLEXITY?
- McCabe essential complexity metric

Control-Flow Graph (CFG) Structure

Bohm and Jacobini, 1966:

every algorithm may be implemented using just the constructs sequence, selection, iteration.

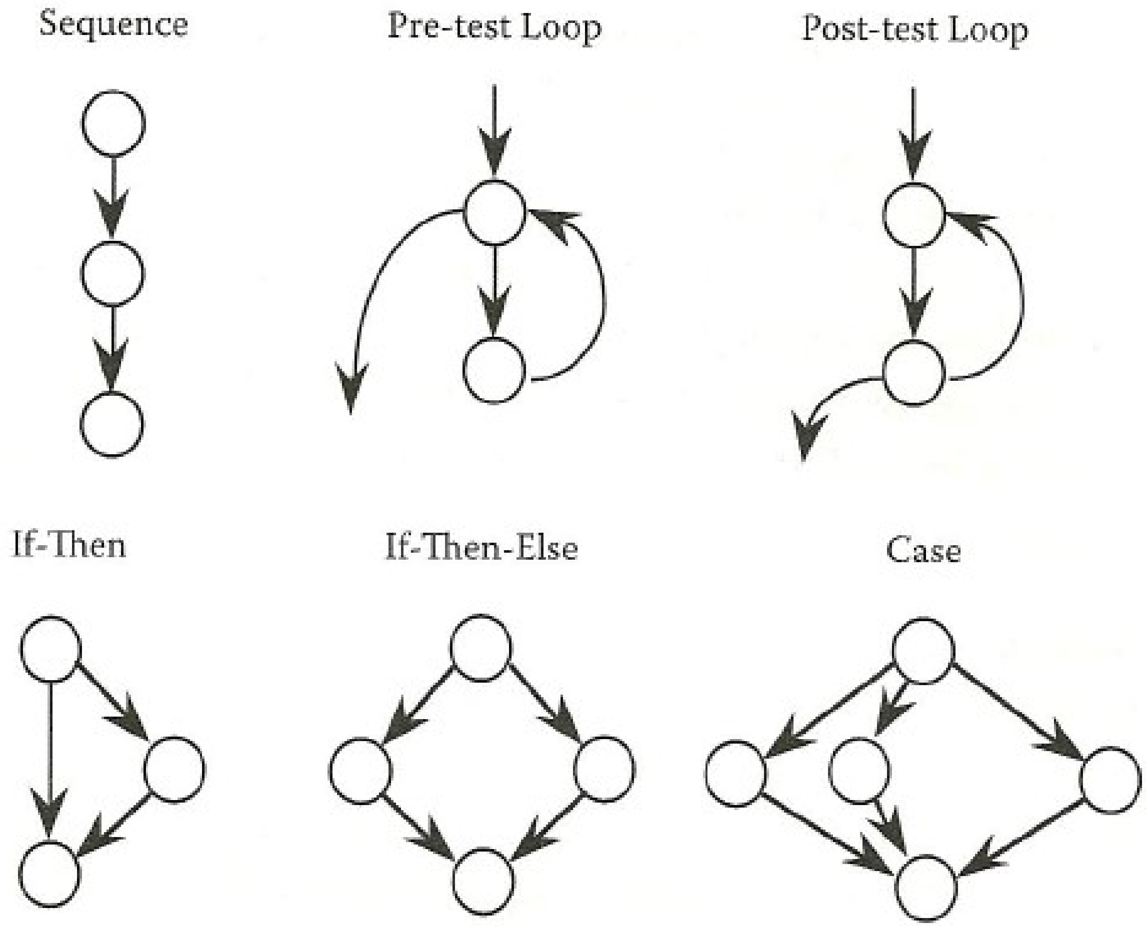


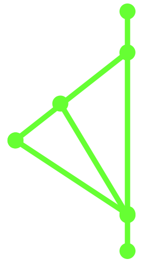
Figure 9.8 Structured programming constructs.

Conditional Logic and CFG

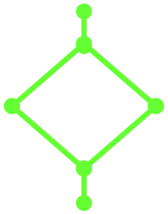
Expanded Form

Compressed Form

```
if ((a == 1) && (b == 1))
      printf("true\n");
```

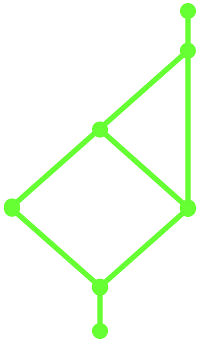


v(G) = 3
ev(G) = 3

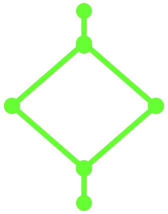


v(G) = 2
ev(G) = 1

```
if ((a == 1) && (b == 1))
      printf("true\n");
else
      printf("false\n");
```

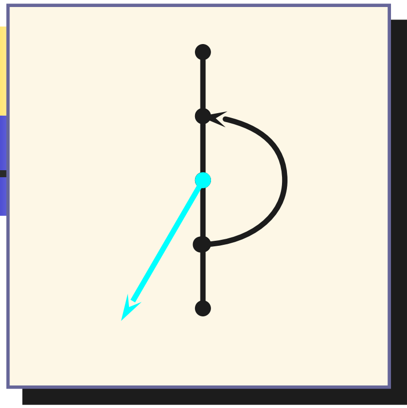


v(G) = 3
ev(G) = 3

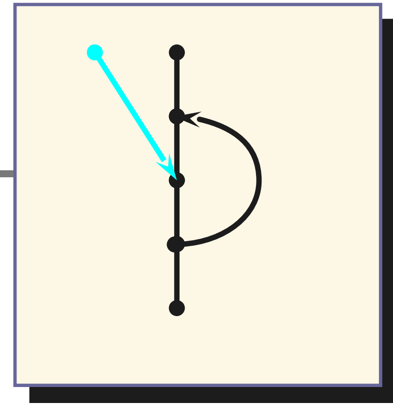


v(G) = 2
ev(G) = 1

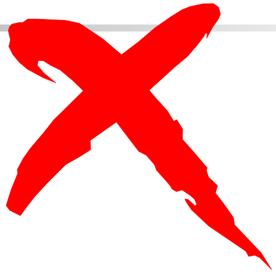
Unstructured Logic

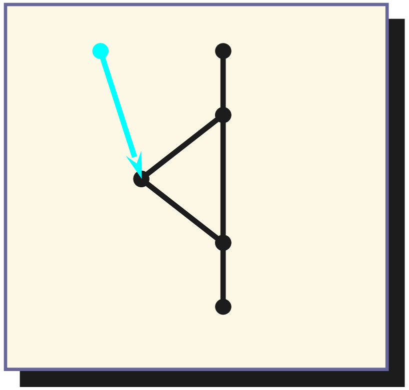


Branching out of a loop

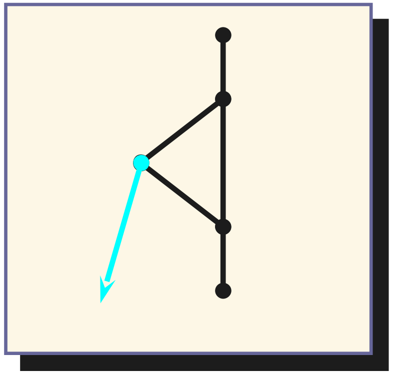


Branching in to a loop





Branching into a decision



Branching out of a decision

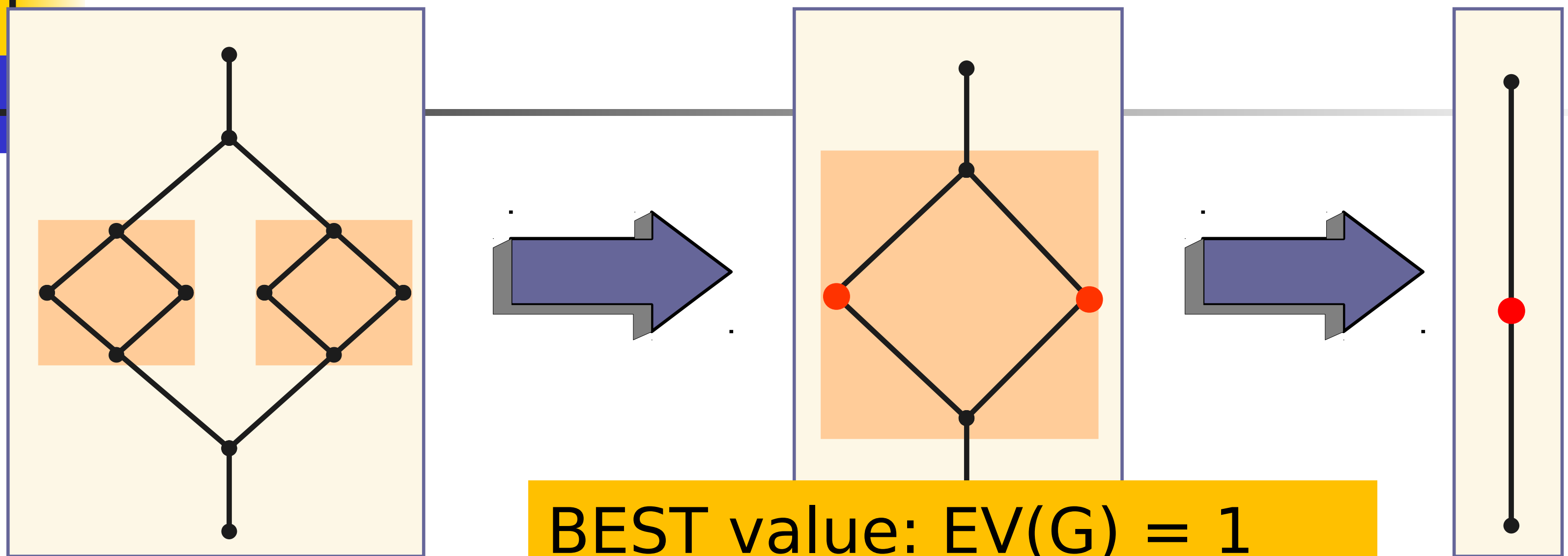
Essential Complexity, EV(G)

- Definition: Essential Complexity, ev, is a measure of the degree to which a module, G, contains unstructured constructs.
- Calculation: ev(G) is equal to the Cyclomatic Complexity of a reduced flowgraph, v(G'), where reduction is completed by removing all structured constructs. The remaining flowgraph is a view of the impact of unstructured code.
- Advantages
  - Quantifies the degree of structuredness
  - Reveals the quality of the code
  - Predicts the maintenance effort
  - Helps the modularization process

Calculating Essential Complexity: Flow Graph Reduction

Original Graph

Reduced Graph



Cyclomatic Complexity = 4

BEST value: EV(G) = 1

<Completely structured>

Essential Complexity = 1

McCabe's Essential Complexity *EV(G)*

Remove structured code constructs, then calculate the Cyclomatic Complexity of reduced graph

BEST value: EV(G) = 1

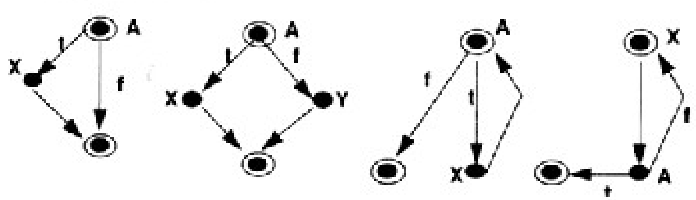
<Completely structured>

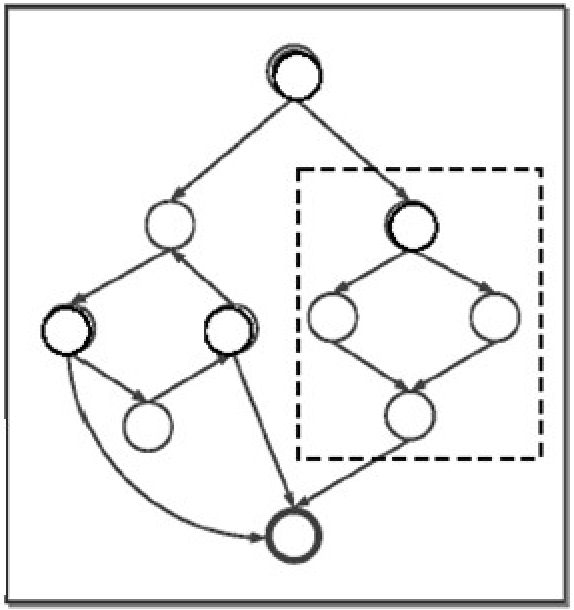
McCabe essential complexity

Essential complexity of a program with flow graph G is given by:

$$ev(G) = v(G) - m$$

m is the number of subflowgraphs of G :

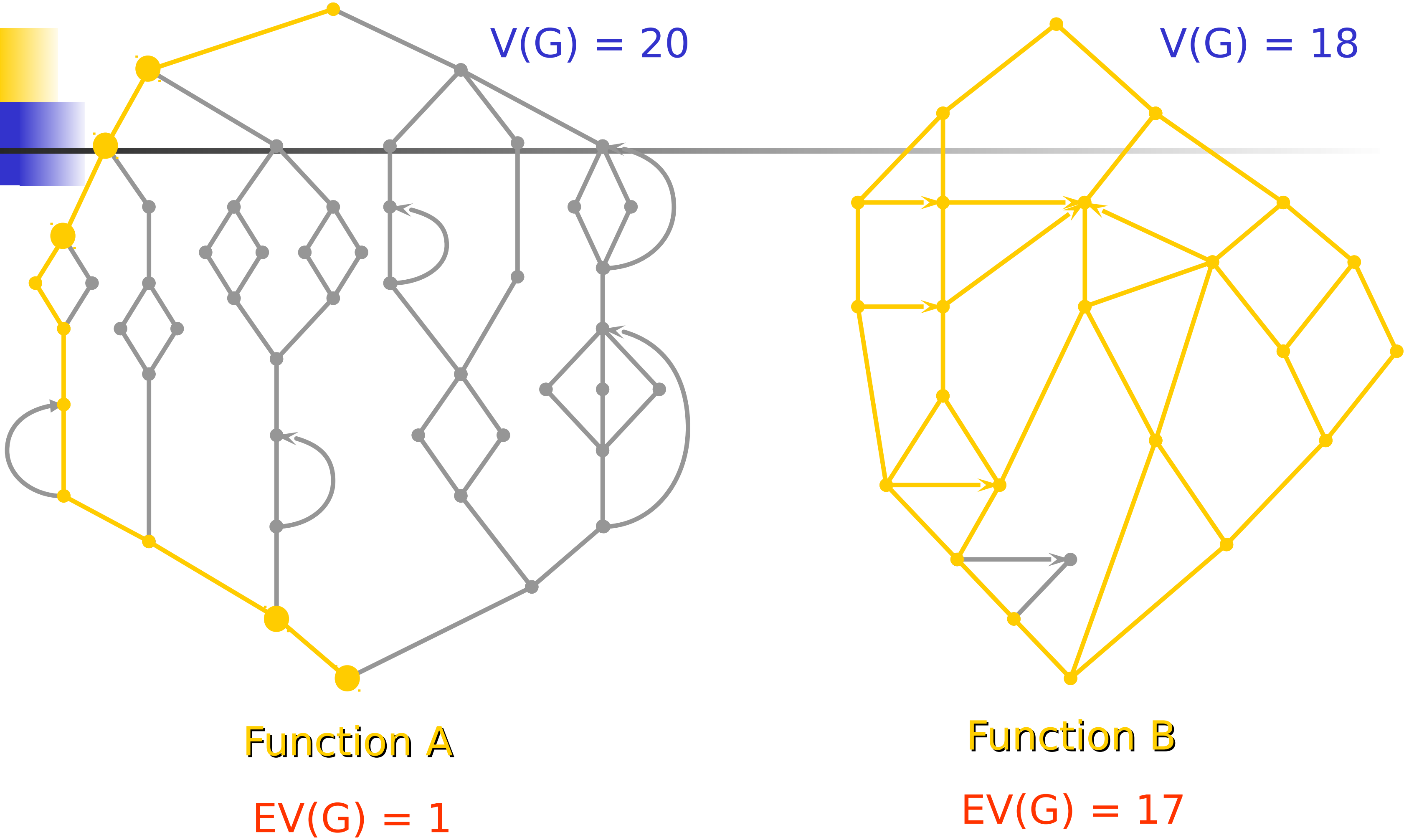




Example:

$$v(G) = 5$$
$$ev(G) = 5 - 1 = 4$$

Essential Complexity



Reduced graphs superimposed on original graphs)

Questions?

- ...

Next?

- Assignment 1 due on **October 8**
- Next lecture: functional size measurement of requirement