FIT5171 Applied Session 8–9
Software complexity & metrics

Week 8–9, 2024

Please do try the questions before coming to the tutorial. Your active participation is the most important!

Weyuker's 9 properties have been proposed to evaluate software metrics. Some of the properties (for example, properties 1, 3, 4 and 8) are quite simple and intuitive. However, some other properties are a bit more complicated and needs further analysis.

In this tutorial we will pick two properties (5, 6) and one software complexity metric from each category (structure, testing and object-oriented) and **informally** prove whether the above properties hold or not. If not, give a counter example.

Structure For structure metrics, we choose the morphology metric Tree Impurity: $TIP = \frac{2(\#E-\#V+1)}{(\#V-1)(\#V-2)}$.

Testing For testing metrics, we choose the simple statement coverage metric C_0.

OO For object-oriented metrics we choose the metric Response For a Class: RFC, equal to the number of methods invocable.

We will restrict our discussion to a single language (Java or C#, for example) for simplicity. We also assume that program composition (+) can be either sequence or nesting.

1. Property 5: The complexity of a program segment should be that of the whole program, i.e., $\forall P, Q \bullet M(P) \leq M(P+Q) \wedge M(Q) \leq M(P+Q)$.

(a) Structure metric *TIP*.

(b) Testing metric C_0.

(c) OO metric RFC.

2. Property 6: The complexity of the composition of two programs P and R may not be the same as the composition of programs Q and R, even though P and Q have the same complexity, i.e., $\exists\, P, Q, R \bullet M(P) = M(Q) \wedge M(P + R) \neq M(Q + R)$.

   (a) Structure metric *TIP*.

(b) Testing metric C_0.

(c) OO metric RFC.