Journal of Software Engineering and Applications, **2011, 4, 123-128**
doi:10.4236/jsea.2011.42013 Published Online February 2011 (http://www.SciRP.org/journal/jsea)

Scientific Research

Analysis of Inconsistencies in Object Oriented Metrics

Ahmed M. Salem[1], Abrar A. Qureshi[2]

[1]Department of Computer Science, California State University, Sacramento, USA; [2]Department of Mathematics and Computer Science, University of Virginia's College at Wise, Virginia, USA.
Email: salema@ecs.csus.edu, aqureshi@uvawise.edu

Received December 17th, 2010; revised January 25th, 2011; accepted January 28th, 2011.

ABSTRACT

Software Metrics have been proposed for procedural and object oriented paradigms to measure various attributes like complexity, cohesion, software quality, and productivity. Among all of these, "Complexity" and "Cohesion" are considered to be the most important attributes. As object oriented analysis and design appears to be at the forefront of software engineering technologies, many different object-oriented complexity and cohesion metrics have been developed. The aim of the paper is to compare some of the complexity and cohesion metrics and to analyze these metrics and expose their inconsistencies. The paper provides a brief introduction of CK and Morris's metrics for calculating the complexity and cohesion of a software. The inconsistencies in these methods are exposed by providing various examples. The paper concludes by proving inconsistencies in CK's cohesion matrices and Morris's complexity matrices.

Keywords*: *Software Metrics, Object Orient Systems

1. Introduction

Object oriented design and development has been the cornerstone for many development projects. With the increase in the use of object oriented methods there is a growing need to improve current practices in object development and in particular software measurement and metrics. A great deal of research work has been done with respect to object oriented measurement and metrics.

The focus has been on developing various object-oriented metrics to measure software quality, productivity, complexity, coupling, cohesion. More research and attention has been given to complexity and cohesion which constitute two of the most important attributes. Complexity of a class can be defined as the number of methods used in a class whereas cohesion can be defined as the measure of the degree to which elements of a module relates to one another. Cohesion implies the tightness among the components of a software module.

Approaches of how to measure complexity and cohesion of the object oriented programs has become an important research area. Many object oriented complexity and cohesion metrics have been proposed in [1-5]. Various studies have been performed on these metrics but not many object oriented metrics has achieved a wide spread use or standard. Even in [6-8] inconsistencies have been shown in various object oriented metrics. In this research work some of complexity and cohesion metrics proposed for object oriented systems have been compared and their inconsistencies were discussed. The first section of this paper introduces the basic concepts used in object oriented metrics and provides a description of some existing complexity and cohesion measures. The second section addresses the complexity and cohesion metrics proposed by CK and Morris. In the third section, an analysis of the complexity and cohesion metrics is provided. The paper concludes with a conclusion and final remarks.

2. Related Work

Measurement is the process by which numbers or symbols are assigned to the attributes of entities in the real world to describe them according to clearly defined rules. Software metrics are units of measurement and play an important role in the software engineering process.

A number of traditional metrics were proposed for structural techniques, but with the advancement of Object Oriented Methodology (inheritance, polymorphism etc) the traditional metrics were no longer useful to measure Object Oriented systems. In an attempt to measure the attributes of object oriented systems many researchers have proposed various metrics like CK, MOOD, Morris,

Chen and Li etc. Reference [9] presents a survey of object oriented design metrics and identifies a set of metrics that have effect on design quality attributes. In [10] the authors investigate and explain 22 different object oriented metrics proposed by various researchers. However, none of these metrics have been able to acquire an accepted standard. Furthermore, some of these proposed metrics are only useful and applicable to measure only some types of object oriented systems. This has been and remains to be a challenge in software engineering.

Complexity and Cohesion are two important attributes in object oriented systems and brief description of each is outlined below:

2.1. Complexity Measures

The complexity of an object class can be defined as the cardinality of its set of properties and it can be measured in terms of the depth of inheritance. In object-oriented design there is a need to develop new metrics for class of objects and their associated children of the class. This concept relates to the complexity as the depth of inheritance indicates the extent to which the class is influenced by the property of its ancestor and number of children indicates the potential impact on descendant. The depth of inheritance and number of children collectively indicate the genealogy of a class. Various other complexity metrics have been proposed and two complexity metrics, proposed by CK and Morris, are discussed and compared.

2.2. Cohesion Measures

Cohesion can be defined as the interamodular functional relatedness of a software module. Cohesion is commonly categorized into seven levels (ranging from low cohesion to high cohesion): coincidental, logical, temporal, procedural, communicational, sequential and functional. Low cohesion modules for example are modules with coincidental cohesion, are indicative of a module that performs two or more basic functions. High cohesion modules have functional cohesion which indicates that modules perform only one basic function. In such modules all module components need to perform the same task which makes functional cohesion is the most desirable among all of them. Many researchers have proposed their own cohesion metrics and some of which are CK's lack of cohesion, Montazeri's connectivity Metrics, Chen and Lu's cohesion measures, Bieman/Kang's class cohesion measures and H. Morris's Degree of cohesion of objects.

Some research has been done on existing object oriented metrics and studies found inconsistencies among them. Some inconsistencies have been found in the most popular object oriented metrics CK which wads proposed by Chidamber and Kemerer. There are also inconsistencies in the cohesion metrics which have been shown in [6,8]. In [6] Bindu Gupta has compared the existing cohesion measures with the properties of cohesion to show the inconsistencies whereas in [7,8] they showed that Lack Of cohesion Metric is inconsistent as it gives the same value of LCOM to both less cohesive and highly cohesive class. In the next section CK'S and Morris's complexity & cohesion metrics are discussed.

3. CK Metrics

CK's metrics have been most readily used. CK's metrics contain fewer flaws than any other metrics. The following is a brief discussion on CK's metrics.

3.1. CK's Depth of Inheritance

CK's metrics have used McCabe's complexity to measure the complexity of an object oriented system. CK's Depth of Inheritance metrics is directly related with the complexity of an object. It relates to the notion of scope of properties, it is a measure of how many ancestor classes can potentially affect this class. The deeper the class is in its hierarchy, the greater the number of methods it is likely to inherit, making it more complex. Deeper trees constitute greater design complexity since more methods and classes are involved. C&K used McCabe's Cyclomatic number to measure the complexity of a class. As detailed in [7,8] McCabe's cyclomatic number can be calculated by subtracting total number of nodes of a tree from total number of edges of a tree and by adding 2 to the result of subtraction.

So McCabe's Cyclomatic Number = E – N + 2 {where E is total no of edges, N total no of nodes}. Then **complexity** = McCabe's cyclomatic number/total no of objects.

Consider the inheritance tree in **Figure 1**. The Cyclomatic Number for the tree would be calculated as follows,

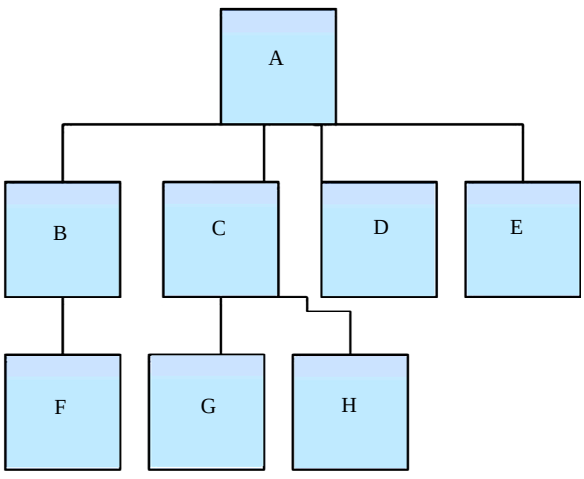


Figure 1. Inheritance tree of objects with complexity 1/8.

Cyclomatic Number = E – N + 2
= 7 – 8 + 2
= 1

Thus the complexity would be,
= Cyclomatic number/total number of objects
= 1/8.

3.2. CK's Lack of Cohesion Measures

CK's Lack of Cohesion metrics are used to measure the cohesion of an object. It uses the concept that if an object has more number of pairs of methods who shares their instances than the number of method pairs who don't share each other's instances then the object is classified as cohesive. The following example will provide a more clear view of Lack of Cohesion metrics:

Suppose there are three methods in an object called M1, M2, M3 and I1, I2, I3 are instance variables respectively. Then there is P and Q as

$$P = \{(Ii, Ij) | Ii \wedge Jj = \text{ empty}\}$$

$$Q = \{(Ii, Ij) | Ii \wedge Ji = \text{ nonempty}\}$$

$$LCOM = |P| - |Q| \{|P| > |Q|\} = 0 \text{ otherwise}$$

Let suppose

$I1 = \{a, b, c, d, e\}$

$I2 = \{b, c, d\}$

$I3 = \{f, g\}$

Now $(I1 \wedge I2) = \{ b, c, d,\} = $ non empty

$(I1 \wedge I3) = \{\text{empty}\}$

$(I2 \wedge I3) = \{\text{empty}\}$

$P = \{\text{Total no of empty sets}\} = 2$

$Q = \{\text{Total no. of non empty sets}\} = 1$

$LCOM = 1$

More similarities mean lower value of LCOM. Consider the following example with fewer similarities between the objects.

$I1 = \{a, b, c\}$

$I2 = \{d, e, f\}$

$I3 = \{i, k\}$

$I4 = \{a, g, h\}$

Now

$(I1 \wedge I2)$ = empty

$(I1 \wedge I3)$ = empty

$(I1 \wedge I4) = \{a\}$

$(I2 \wedge I3)$ = empty

$(I2 \wedge I4)$ = empty

$(I3 \wedge I4)$ = empty

$P = \{\text{Total no. of empty sets}\} = 5$

$Q = \{\text{Total no. of non empty sets}\} = 1$

Thus LCOM = 4.

4. Morris's Metrics

Morris proposed a metric suite for object oriented systems in terms of tree structures. And following are Morris's complexity and cohesion metrics:

4.1. Morris's Complexity Metrics

Morris defined complexity of an object oriented systems in terms of depth of a tree. Depth of a tree can be measured in terms of number of sub nodes of a tree. The more the number of sub nodes of tree the more complex the system. So complexity of an object is equal to the depth of tree or total number of sub nodes.

The depth of tree displayed in **Figure 2** is 3. Thus according to Morris's Complexity Metric the complexity of objects of the tree is equal to 3.

The depth of the inheritance tree in **Figure 3** is equal to 1. Thus the complexity of objects on tree is also equal to 1.

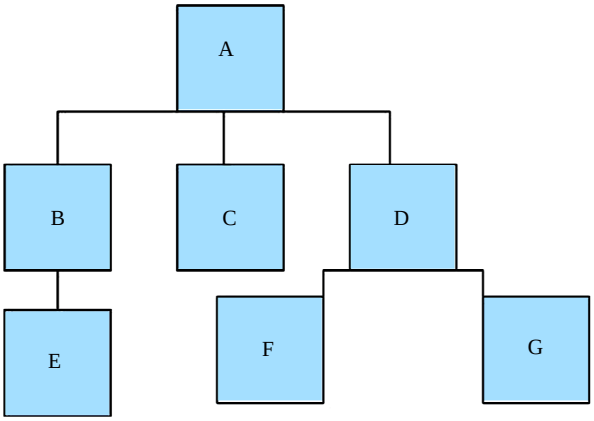


Figure 2. Inheritance tree of objects with depth 3.

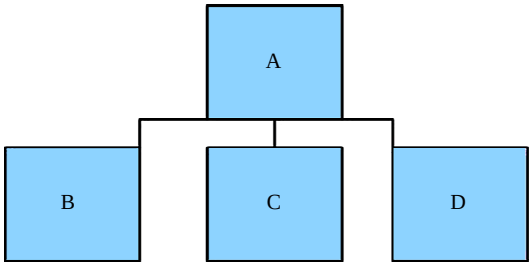


Figure 3. Inheritance tree of objects with depth 1.

4.2. Morris's Degree of Cohesion

CK's proposed metrics for cohesion only gives the cohesiveness of a class as either low or high. Morris proposed a cohesion metrics which is not bound to only two values—high and low. Morris defined a new cohesion metrics in his thesis in terms of Fan In. This can be defined as:

"Fan-In" of a module M is the number of local flows that terminate at M, plus the number of data structures from which information is retrieved by M [11]. In simple words Fan In is the total number of inputs received by a method.

Degree of cohesion of objects = total number of Fan-In/total number of objects. For example consider the data structure displayed in **Figure 4**. The arrow heads show the fan-ins to the object.

Thus according to Morris's,

Degree of cohesion = total number of Fan-In/total number of objects.

Degree of cohesion = 11/7.

5. Inconsistencies

As we analyze CK'S and Morris's Metrics, inconsistencies have been found in their metrics. CK's Lack Of cohesion Metrics and Morris's complexity metrics are inconsistent and these inconsistencies are discussed in this section.

5.1. CK's Lack of Cohesion Metrics

In software metric's CK'S metrics are most widely used but these metrics have some inconsistencies, and according to [7,8] LCOM doesn't distinguish two dissimilar entities as its shows same value of LCOM for two different classes with different cohesiveness. LCOM only shows low cohesion or a high cohesion for objects even if some objects have medium cohesiveness.

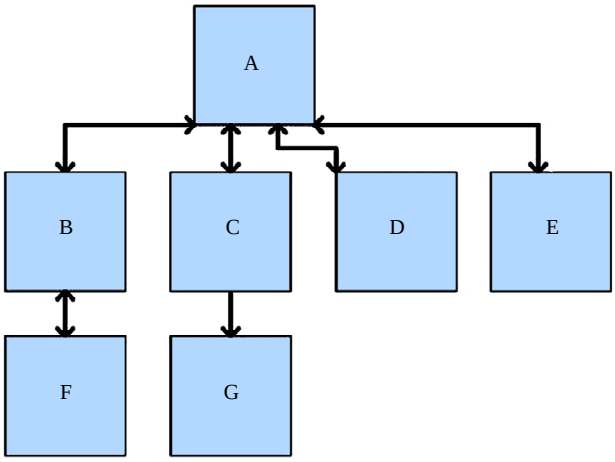


Figure 4. Data structure describing fan-ins to various objects with Degree of Cohesion = 11/7.

In **Figure 5(a)** and **5(b)** two different classes are shown with their methods {Ii, Ji}. In **Figure 5(a)** according to CK, value of LCOM will be 8 as |P| = 9 and |Q| = 1 and |P| – |Q| = 8

Only $Q = \{(I4, J4) \wedge (I5, J5)$ is a non-empty set$\}$ and all other are empty pairs.

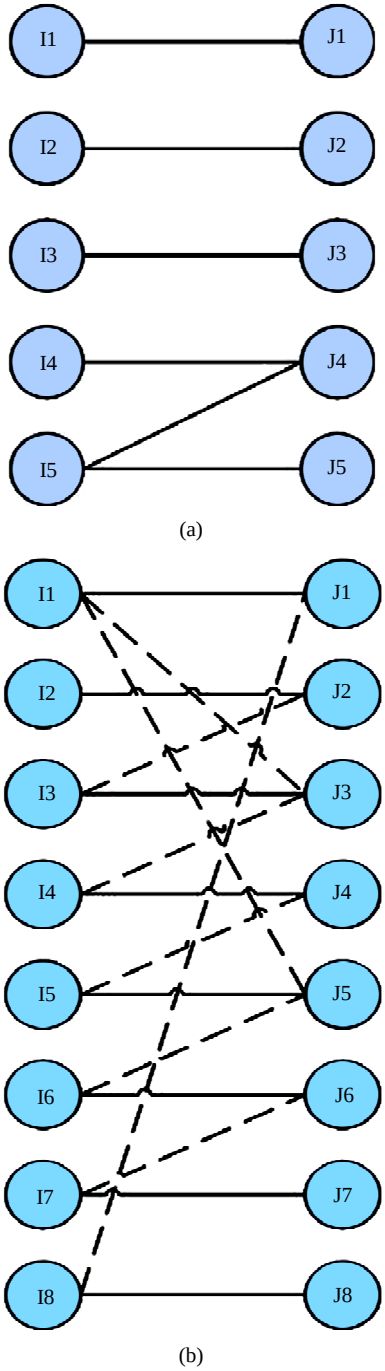


Figure 5. (a) Mapping between methods (Ii) and instance variables (Ji) with LCOM = 8; (b) Mapping between Methods (Ii) and Instance variables (Ji) with LCOM = 8.

In **Figure 5(b)** there are 18 empty pairs (|P| = 18) and 10 non empty pairs (|Q| = 10) of (Ii, Ji) so |P| – |Q| = 8. Hence both classes have same cohesion value. But from the image it can be seen that class (b) is more cohesive than class (a). Hence LCOM doesn't distinguish between two dissimilar entities or we can say LCOM does not represent a precise value of cohesiveness of a class.

In [6], LCOM measure is compared with various basic properties of cohesion measure, LCOM satisfies many properties but not all, as minimum value of LCOM is zero but maximum value is not defined.

5.2. Morris Complexity Metrics

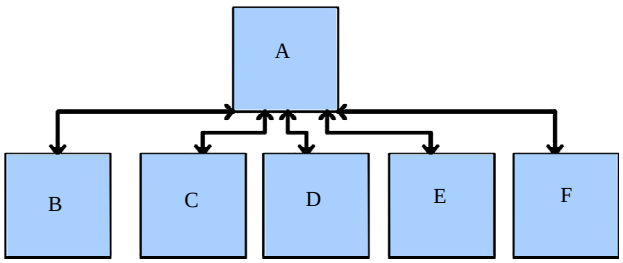
Morris's complexity metrics measure the complexity of an object as the depth of its tree structure. To show inconsistencies in this metrics we have taken the measurement of two classes by using CK's complexity metrics and Morris's Cohesion metrics. Basically a class's cohesiveness and complexity are interrelated. If a class is highly cohesive then it is obvious it will have more complexity. But the Morris's complexity metrics doesn't obey this basic law. Let's have two modules as shown in **Figure 6(a)** and **Figure 6(b)**.

By measuring the complexity of modules shown in **Figure 6(a)** and **Figure 6(b)** using Morris's Complexity Metrics, it can be seen that module given in **Figure 6(b)** is more complex than module given in **Figure 6(a)**.

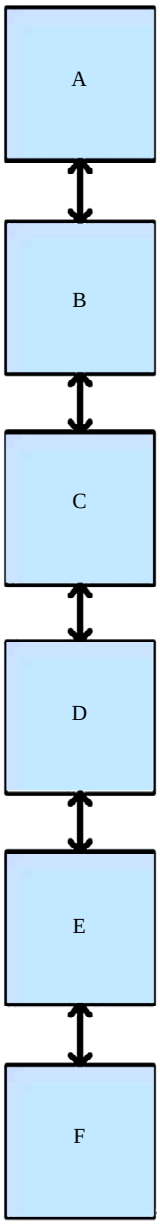
For example, Number of sub nodes in **Figure 6(a)** is 1 and in **Figure 6(b)** the number of sub nodes is 5. Hence module in **Figure 6(b)** has more complexity than module given in **Figure 6(a)**.

Let us suppose that these results are true. The cohesiveness of module given in **Figure 6(a)** using Morris's Cohesion Metrics, is measured as the degree of cohesion = 10/6, where total number of fan-in for module is 10 and total number of objects are 6 {fan-in for A = 5 and for B, C, D, E, F fan-in = 1, total fan-in = 5 + 1 + 1 + 1 + 1 + 1 = 10}. Now if we measure the cohesiveness for second module given in **Figure 6(b)**, results are same and for **Figure 6(b)** Degree of cohesion of objects= 10/6.

Now both modules have same cohesiveness but different complexities. So either Morris's cohesion metrics is inconsistent or complexity metrics is inconsistent.



(a)



(b)

Figure 6. (a) Fan-ins between modules with degree of cohesion 10/6 and complexity 1; (b) Fan-ins between modules with degree of cohesion 10/6 and complexity 5.

In the next step we will now measure the complexity of those modules using CK's complexity metrics. CK's complexity measures complexity in terms of McCabe's Cyclomatic number and therefore complexity of a module given in **Figure 6(a)** = (McCabe's Cyclomatic No/Total number of Methods) = (5 – 6 + 2)/6 = 1/6

Where McCabe's cyclomatic number = e – n + 2 {where e total no of edges, n total no of nodes}

Now if we measure the CK's complexity for module given in **Figure 6(b)**, then the Complexity = (5 – 6 + 2)/6

= 1/6.

So both classes have the same complexity values, which in turn show that Morris's complexity metrics is inconsistent.

6. Conclusions

CK's cohesion metric doesn't distinguish between two classes which have different cohesiveness, it shows both classes have LCOM value =8. The problem is that LCOM value can be either a non-zero value if |P| > |Q| or 0 otherwise, which means low cohesion or high cohesion. This clearly shows that CK's is not able to distinguish a value of cohesion between 0 and any other non-zero value or between a class with high cohesion and a class with medium cohesion, where as Morris's cohesion metrics doesn't bound to only two values of cohesiveness. They can provide a precise value of class cohesiveness.

Morris's complexity metrics shows that the class in **Figure 6(b)** is more complex than the class given by **Figure 6(b)**. However, both classes have the same cohesiveness and so should be equally complex.

This paper documented that cohesion metrics proposed by CK and complexity metrics proposed by Morris both have some inconsistencies. We feel that more it will be more beneficial to use CK'S complexity metrics and Morris's cohesion metrics. The paper also demonstrated the need for refining these metrics and developing new object oriented metrics.

REFERENCES

[1] S. R. Chidamber and C. F. Kemerer, "Towards a Metric Suite for Object Oriented Design," *Proceedings of OOPSLA Conference on Object-Oriented Programming Systems, Language and Applications*, 6-11 October 1991, pp, 1-18. doi:10.1145/117954.117970

[2] S. R. Chidamber and C. F. Kemerer, "Towards a Metric Suite for Object Oriented Design," *IEEE Transactions on Software Engineering*, Vol. 20, No. 6, 1994, pp. 476-493. doi:10.1109/32.295895

[3] J. Bieman and L. Ott, "Measuring: Functional Cohesion," *IEEE Transactions on Software Engineering*, Vol. 20, No. 8, 1994, pp. 644-657. doi:10.1109/32.310673

[4] J. Bieman and B.-K. Kang, "Cohesion and Reuse in an Object Oriented System," *Software Engineering Notes*, 1995, pp. 259-262.

[5] L. C. Briand, J. Daly and J. Wust, "A Unified Framework for Cohesion Measurement in Object-Oriented Systems," *Empirical Software Engineering*, Vol. 3, No. 1, 1998, pp. 65-117.

[6] Bindu Gupta, "A Critique of Cohesion Measures in Object Oriented Paradigm," Master's Thesis, Michigan Technological University, Houghton, 1997.

[7] S. Sunnit and R. S. Salaria, "Critical Analysis of Inconsistencies in CK and MOOD Metrics," 2*nd ACIS International Conference on Software Engineering Research, Management and Applications*, Los Angeles, 5-7 May 2004.

[8] S. Sunnit and R. S. Salaria, "Analysis of CK and Mood Metrics for Inconsistencies and Prediction of Quality Attributes," November 2007.

[9] S. R. Ragab and H. H. Ammar, "Object Oriented Design Metrics and Tools a Survey," *The 7th International Conference on Informatics and Systems*, Cairo, 28-30 March 2010, pp. 1-7.

[10] K. K. Aggarwal, Y. Singh, A. Kaur and R. Malhotra, "Empirical Study of Object-Oriented Metrics," *Journal of Object Technology*, Vol. 5, No. 8, 2006, pp. 149-173. doi:10.5381/jot.2006.5.8.a5

[11] N. Fenton, "Software Metrics: A Rigorous and Practical Approach," 1998.

