CS406 Software Engineering II

Term Paper

Coupling and Cohesion Metrics for HOOMT

By Raghavendra Kotikalapudi

Table of Content

Abstract

Software metrics play a very important role in developing high quality software. High order object modeling technique integrates structural analysis with object oriented paradigm to strike a mean between both the extremes. In this paper, we developed metrics to evaluate the quality of high order object design at any given stage of development. In particular, we evaluate quality in terms of coupling and cohesion among various types of objects in a high order object model. Coupling and cohesion metrics are developed by combining several ideas from object oriented modeling domain.

1. Introduction

HOOMT (Markose, Liu, & McMillin, 2008) is a novel way of uncovering structures within a network of objects linked by complex communications and relationships. It works by combining structural analysis with traditional object oriented approaches, thus striking a mean between both the extremes.

In HOOMT modeling, one starts with the context object diagram showing the interested system as a high order object interacting with its surroundings. A high order object (HOO) can be decomposed into primitive objects. Thus, HOO represents an object abstracting its constituent components in addition to attributes and methods contained in traditional object classes. Primitive object (PO) on the other hand is represents an entity that requires no further decomposition in the analysts mind. An example of HOO and PO is shown in figure 1.

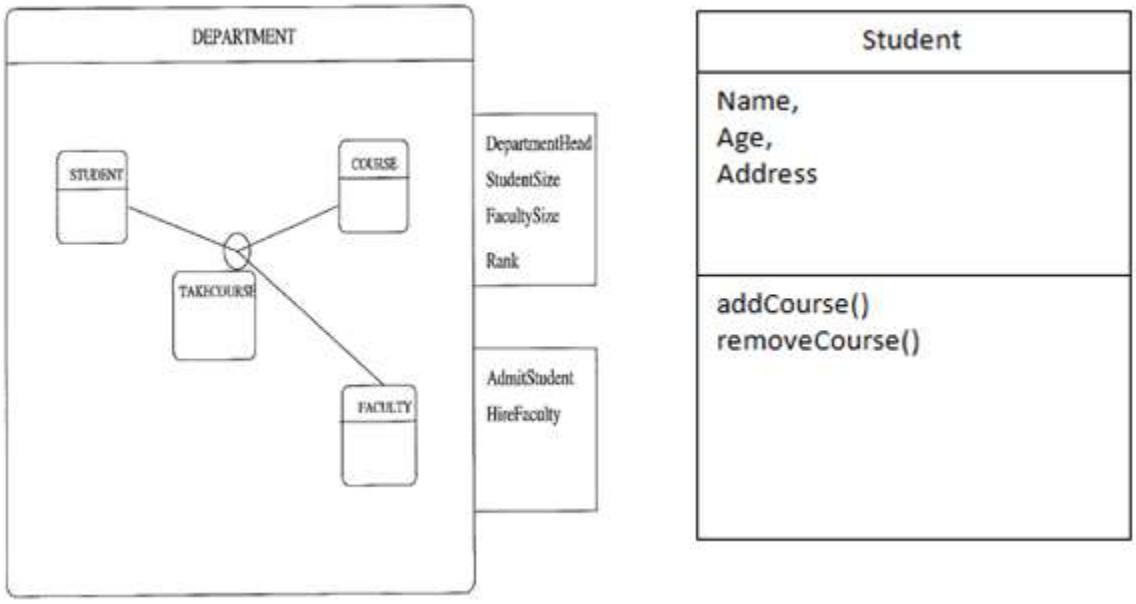

Figure 1: Illustration of HOO and PO

In figure 1, Department is a HOO as it encapsulates student, course, and faculty components within it. Student on the other hand can be considered primitive depending on how the requirements are laid out. It is primitive if Student suffices the requirement. If the requirements require analyst to decompose the object further, then it is considered as a HOO.

HOOMT works by simultaneously decomposing HOO and by performing SA side by side. At every level, functionalities identified in SA are mapped to objects identified by HOOM. The process is illustrated by figure 2. This process is continued until a satisfactory design is generated.

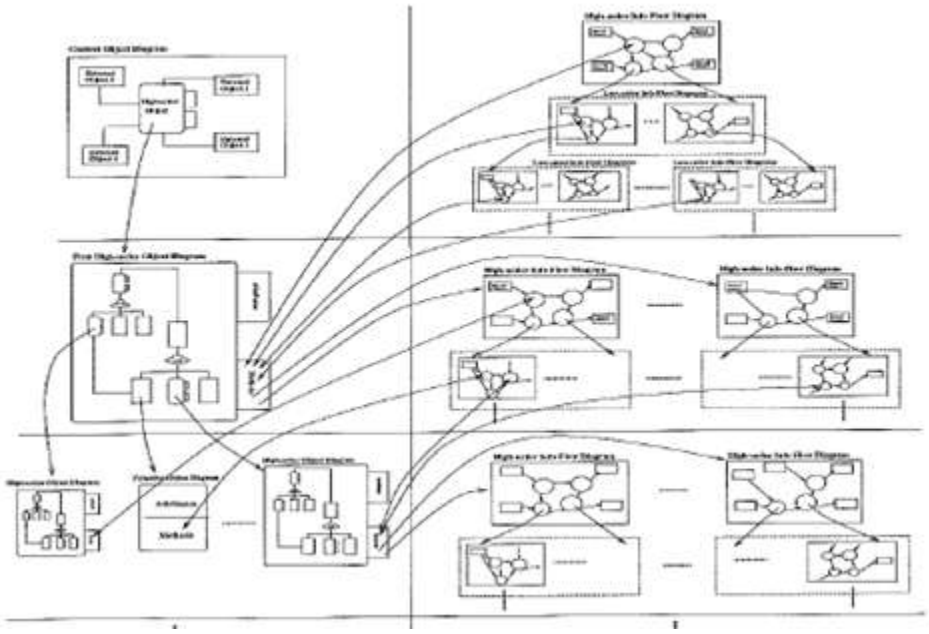

Figure 2: HOOMT phase by phase decomposition

In essence, HOOMT integrates tightly with OOA and SA. The top to bottom approach of HOOMT provides a systematic approach to the realization of a more structured system with highly cohesive structures.

1.1 Practical Difficulties

HOOMT is a novel process model with a unique starting point leading to a full fledged object oriented design. At each level of HOOM, functionalities identified by SA are mapped to HOO and PO identified by hierarchical object decomposition. Typically, analyst has many candidate choices for mapping responsibilities to objects. Among all the possible choices, the analyst needs make a design decision that increases the overall quality of the design.

To mitigate this issue, we propose coupling and cohesion metrics as a measure of design quality. The idea is to quantify coupling and cohesion for all the candidate options, thus providing analyst a better sense of judgment. We consider coupling and cohesion as they're the most popular indicators of design quality in an object oriented system (Mao & Jiang, 2008; Pfleeger & Atlee, 2006; Myers, 1976). Typically, one would choose a design with high coupling and low cohesion. Tradeoff decisions among coupling and cohesion are not discussed in this paper. The scope of this paper is only to provide metrics for coupling and cohesion.

1.2 Literature Survey

Since HOOMT is an object modeling technique, we looked at existing coupling and cohesion metrics for object oriented systems. The idea is to identify candidates that can be modified and applied to HOO and PO. In particular, we are interested in metrics that can be extended and applied to HOO.

1.2.1 Coupling Metrics

Generally, coupling interaction refers to the degree of interdependence between components of a system. High coupling between two components makes it harder to understand and maintain one of them in isolation. In contrast, low coupling leads to self-contained and thus easy to understand and maintain them.

In (Chidamber & Kemerer, 1991; Chidamber, Kemerer, & Mit, 1994), Chidamber and Kemerer introduced a suite of metrics for OO systems. The suite includes a simple coupling metric called coupling between objects (CBO) that defines the coupling of a class to be the number of classes to which it is coupled. Initial definition of CBO excluded coupling due to inheritance (Chidamber & Kemerer, 1991), but was changed in later work (Chidamber, Kemerer, & Mit, 1994).

Briand et al. defined a framework (Briand, Daly, & Wust, A unified framework for coupling measurement in Object-Oriented systems, 1999) where coupling measures are based on set and graph theory. He captured several types of interactions between classes like class-attribute, class-method, as well as method-method interactions. The measures from the suite also differentiate between import and export coupling as well as other types of relationships like friends, ancestors, descendants etc.

Huan Li introduced a novel coupling metric, called global coupling metric (Li, 2008), to evaluate coupling interactions between classes of object-oriented systems. The metric differs from the majority of existing metrics in two aspects: it takes into account the strength that one class is dependent on another, it reflects indirect coupling.

1.2.1 Cohesion Metrics

Cohesion is an important attribute referring to the quality of the abstraction provided by a class under consideration. In object-oriented software systems, cohesion has normally been calculated on a per-class or per-object basis. Good abstractions typically possess high cohesion. The original cohesion metric was first given by Chidamber and Kemerer (Chidamber & Kemerer, 1991). They define Lack of Cohesion of Methods (LCOM) as opposed to cohesion exhibited by a class.

Briand et al. (Briand, Daly, & Wust, A unified framework for cohesion measurement in object-oriented systems, 1998) extended LCOM to take inheritance into consideration. He also considered attributes in a class might not be accessed by any method in the class, which the original metric did not. This metric was called Coh.

Chae et al. (Chae, Kwon, & Bae, 2000) developed a cohesion metric called Cohesion Based on Member Connectivity (CBMC). Chae et al. believed that the cohesiveness of a class was based not only on the number of interactions between sections of a class, but also on the patterns of the interactions. Chae et al. differentiated between glue methods and non-glue methods. Glue methods are methods without which its reference graph becomes disjoint.

2. Proposed Approach

HOOM, at any given stage consists on HOO and PO with simple or complex relationships. Therefore, we consider coupling between:

1) PO – PO
2) HOO – PO
3) HOO - HOO

Similarly, for cohesion, we consider

1) Cohesion in PO
2) Cohesion in HOO

2.1 Coupling Metrics

Deriving from the description in the paper (Briand, Daly, & Wust, A unified framework for coupling measurement in Object-Oriented systems, 1999), we denote the class that accesses another class as client class, and the class that be accessed as server class. As the server class controls the internal logic of the client class, the direction of coupling interaction is important. For instance, the server class may determine the implementation of the client access. A change of the server class will likely require relevant.

Drawing on the ideas from the paper (Li, 2008), we start by constructing a directed graph G(V, E, F) such that V is a set of all the classes (HOO or PO), E is a set of all coupling relations between classes. Coupling between class i and class j is represented by an arrow from i to j as shown in figure 3.

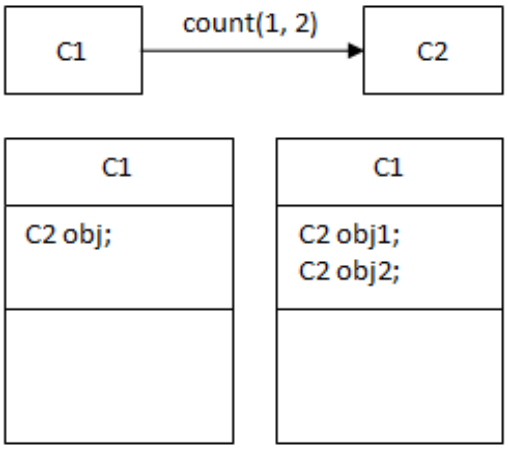

Figure 3: Coupling between class C1 and class C2

We then define $count(i, j)$ as the total amount of coupling of class i accessing class j. The total amount of coupling between can be defined in terms of:

- Data Coupling (communication via scalar parameters)
- Stamp Coupling (dependency induced by the type of structured parameters)
- Control Coupling (parameters are used to control the behavior of a module)
- Common Coupling (communication via shared global data)
- Content Coupling (one module shares and/or changes the definition of another module)

As illustrated in figure 3, there may be multiple instances of C2 coupled with C1, in which case, summation of coupling over all instances are considered. Therefore, $count(i, j)$ is defined as:

$$count(i,j) = \sum_{i=1}^{num\ \ instances} Num\ of\ method\ calls\ for\ instance\ i$$

$f(i,j)$ is defined as the dependency frequency between class i and class j. It indicates coupling proportionality if a class is coupled with multiple classes. It is defined as:

$$f(i,j) = \frac{count(i,j)}{\sum_{k \in v} count(i,k)}$$

Figure 4 illustrates how $f(i,j)$ is computed from $count(i,j)$. Each edge label in the figure shows $count(i,j)$, $f(i,j)$ is shown in the brackets following it.

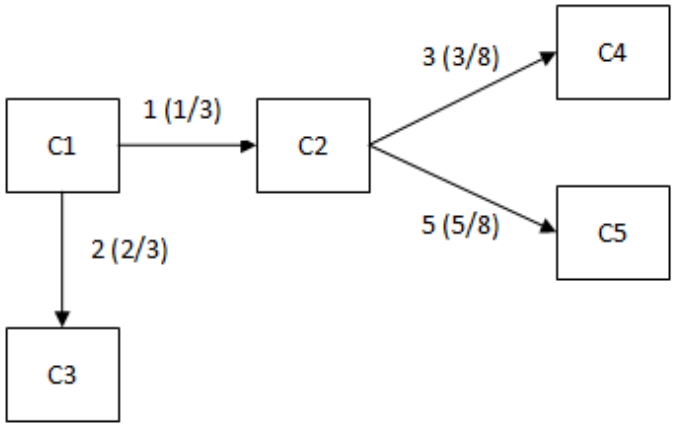

Figure 4: Computing frequency dependencies among various classes.

As defined in paper (Li, 2008), let C_i denote the global coupling value for class i. If class 1 accesses a class 2, which in turn accesses a class 3, then class 1 is indirectly coupled to class 3. To account for this fact, global coupling contributed by class i to the overall system is defined as:

$$C_i = \sum_{k \in v} f(k,i) \times C_k$$

According to this definition, we have the following linear equations:

$$\begin{cases} s_1 = f(1,1)s_1 + \ldots + f(i,1)s_i + \ldots + f(n,1)s_n \\ \vdots \\ s_n = f(1,n)s_1 + \ldots + f(i,n)s_i + \ldots + f(n,n)s_n \end{cases}$$

In what follows, we use matrix A with $A_{ij} = f(j,i)$ to denote a directed coupling graph of an OO software system. The above linear equations can be written as $AX = X$, where vector $X = [C_1, C_2 \ldots C_n]^T$. This transforms the problem of measuring global coupling metric into the problem of finding an eigenvector for a square matrix. We thus seek an eigenvector X with eigenvalue 1 for the matrix A.

The column j of matrix A is corresponding to the dependence frequencies of class j to other classes. Definition of dependency frequency ensures the column of that contains nonzero entries sums to 1. Therefore A is a stochastic column matrix, and this ensures that it has an eigen vector with an eigen value of 1.

Consider the graph shown in the figure 5. It shows all the counts and dependency frequencies.

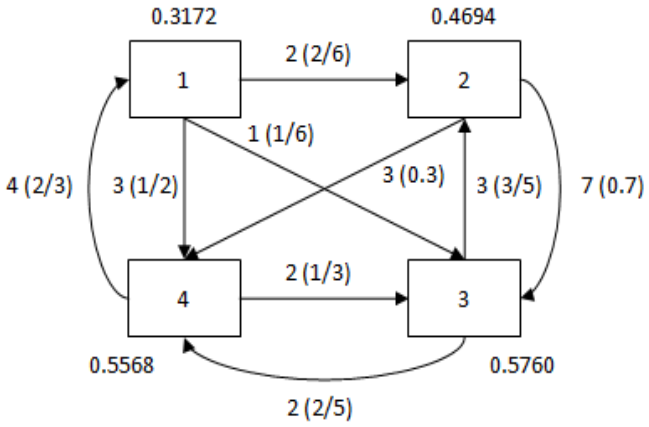

Figure 5: Sample graph with count (i, j) and dependency frequencies

For this graph, Matrix A can be written as:

$$A = \begin{pmatrix} 0 & 0 & 0 & 2/3 \\ 1/3 & 0 & 3/5 & 0 \\ 1/6 & 0.7 & 0 & 1/3 \\ 1/2 & 0.3 & 2/5 & 0 \end{pmatrix}$$

Eigen vector for A with eigen value = 1 is

$$\begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{pmatrix} = \begin{pmatrix} 0.3172 \\ 0.4694 \\ 0.5760 \\ 0.5568 \end{pmatrix}$$

We can see that C1 has the least global coupling value; this is because only one class 4 is using class 1. Class 2 has a slightly higher value as two classes are using it. Classes 3, 4 have higher values as all the other classes use it. As PO is equivalent to simple object in OO system, the idea described above applies directly among any PO's in the system.

We now consider coupling between PO and HO. Consider a PO coupled with a HOO as shown in figure 6. We know that methods M1() and M2() make use of services provided by the component classes 1, 2 and 3. Therefore, there is a coupling between HO class and classes 1, 2, 3. Hence HOO can be replaced by the graph consisting of HO, classes 1, 2, 3, where HO is coupled with all the other component classes based on the number of services it utilizes from each of the component classes. Thus, given any PO-HO coupling, we can transform it into a simple PO-PO graph and find the coupling metrics. This idea is illustrated in figure 7.

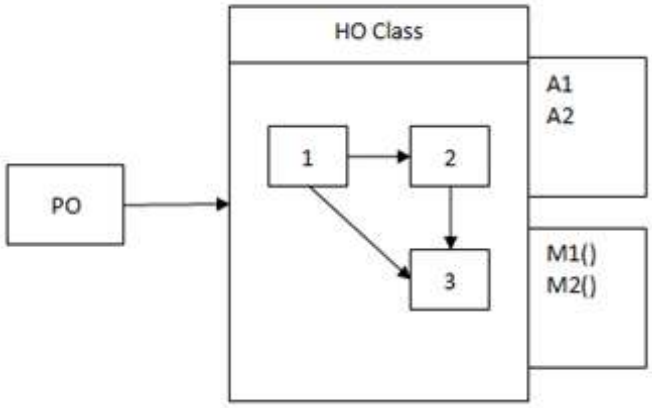

Figure 6: High Order Object

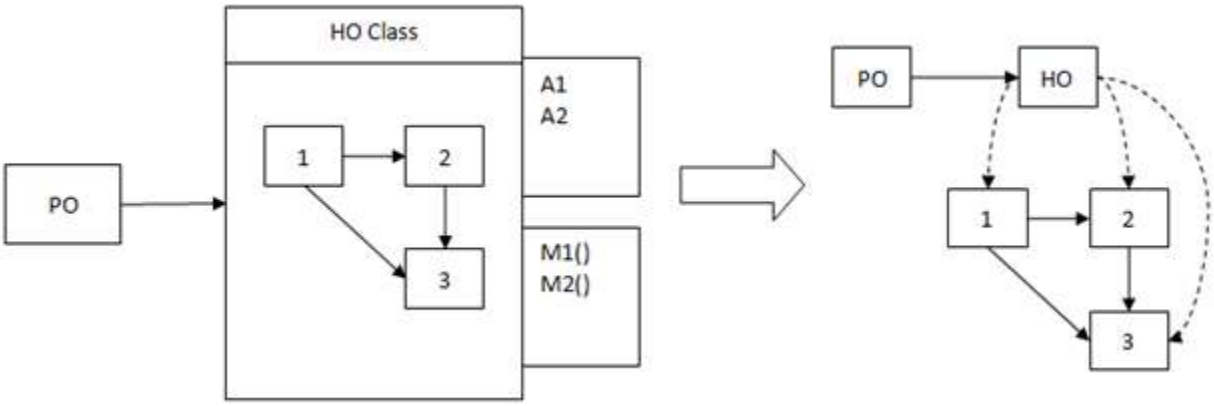

Figure 7: Transforming PO-HO interaction into PO-PO interaction.

Similarly, one can also transform a HO-HO interaction into PO-PO interaction to compute global coupling metrics.

2.2 Cohesion Metrics

Cohesion can be defined as a measure of the degree to which the elements of a module belong together (Bieman & Kang, 1994). In object-oriented software systems, cohesion has normally been calculated on a per-class or per-object basis.

For HOOMT, we consider cohesion in PO and cohesion in HO. Most of the existing literature talks about *"Lack of cohesion"*, LCOM. In this paper, we introduce metrics to define cohesion in a class. The cohesion metric is normalized to have a range between 0 and 1, with 0 being the least cohesion and 1 being the maximum cohesion.

For cohesion in PO, we consider Method-Method and Method-Attribute interaction. We start by constructing a directed graph G (V, E), V being a set of all the methods and the attributes. Method-Method and Method-Attribute interactions are represented as an edge in this graph.

Let N_m be the number of methods and N_a, the number of attributes. We define cohesion as:

$$Cohesion = Cohesion(Method - Attribute) + Cohesion(Method - Method)$$

$$Cohesion = \frac{\frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Na} Is\_Coupled(i,j)}{N_m * N_a} + \frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Nm} Is\_Coupled(i,j)}{{N_m}^2 - N_m}}{2}$$

$$Is\_coupled(i,j) = \begin{cases} 1; & if\ i\ calls\ or\ uses\ j \\ 0; & if\ i = j \end{cases}$$

In the cohesion metric, Method-Attribute (M-A) cohesion is given by:

$$\frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Na} Is\_Coupled(i,j)}{N_m * N_a}$$

In short, with the double summation, we are finding all the possible pairings among the interacting methods and attributes. This value should be 1 if it equals maximum number of method-attribute pairings, which is given by $N_m * N_a$. Thus we divide the summation by $N_m * N_a$ to normalize range to (0, 1).

Method-Method cohesion is given by:

$$\frac{\sum_{i=1}^{Nm} \sum_{j=1}^{Nm} Is\_Coupled(i,j)}{{N_m}^2 - N_m}$$

The double summation gives all Method-Method (M-M) interactions within an object. There can be a maximum of ${N_m}^2$ such pairings. Excluding (M_i, M_i) type of pairings, we have maximum of ${N_m}^2 - N_m$ pairs. Therefore, we divide the summation by ${N_m}^2 - N_m$ to normalize its value to the range (0, 1).

Since, M-M and M-A cohesion ranges from (0, 1) we divide the entire value by 2 to normalize cohesion value between (0, 1). If number of attributes is zero, then the original formula yields 50% lesser value. Therefore, when there are no attributes, cohesion is defined as:

$$cohesion = \frac{\sum_{i=1}^{Nm} \sum_{j=1}^{Nm} Is\_Coupled(i,j)}{{N_m}^2 - N_m}$$

As an example, consider a class structure given in figure 8. We convert it into a directed graph as shown in figure 8. We have $N_m = 2 \; and \; N_a = 2$.

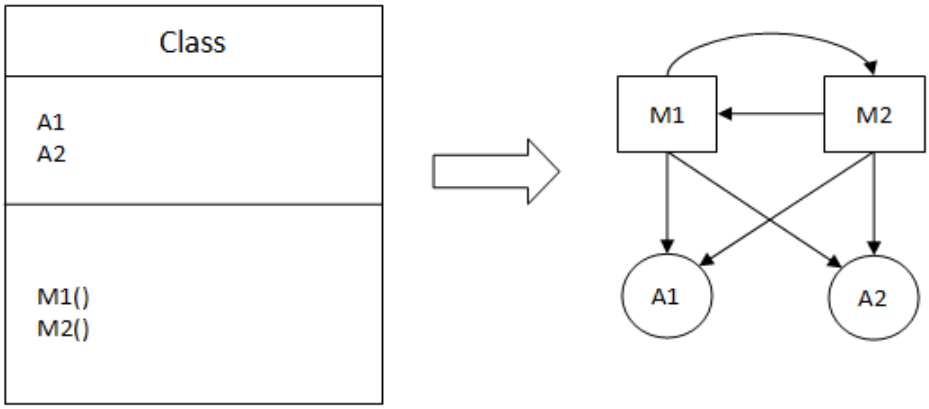

Figure 8: Conversion of class interactions into directed graph for computing cohesion

$$\sum_{i=1}^{Nm}\sum_{j=1}^{Na} Is\_Coupled(i,j) = Is\_coupled(m_1,a_1) + Is\_coupled(m_1,a_2)$$

$$+ Is\_coupled(m_2,a_1) + Is\_coupled(m_2,a_2) = 1+1+1+1 = 4$$

$$\sum_{i=1}^{Nm}\sum_{j=1}^{Nm} Is\_Coupled(i,j) = Is\_coupled(m_1,m_1) + Is\_coupled(m_1,m_2)$$

$$+ Is\_coupled(m_2,m_1) + Is\_coupled(m_2,m_2) = 0+1+1+0 = 2$$

$$Cohesion = \frac{\frac{4}{2 \times 2} + \frac{2}{2^2 - 2}}{2} = 1$$

This is expected as all methods interact with all the other methods and attributes. It is easy to show that this value decreases as we cut the number of links amongst method-method and method-attribute pairs.

For a HOO, other than cohesion from M-A and M-M interactions, we also have M-C cohesion. Since a method in HOO utilizes methods in the component classes, there exists cohesion between methods in HOO and methods in the component classes. This cohesion is given by:

$$Cohesion\ (M-C) = \frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Nc}\sum_{k=1}^{N_{j,m}} Is\_Coupled(i,k)}{N_m \times \sum_{j=1}^{N_c}\sum_{k=1}^{N_{j,m}} K}$$

Where, N_m is the number of methods in HOO, N_c is the number of component classes and $N_{j,m}$ represents the number of methods for the component class j.

With this analogy, the summation yields all possible pairings of methods in HOO and methods in component classes. There can be a maximum of $N_m \times \sum_{j=1}^{N_c}\sum_{k=1}^{N_{j,m}} K$ such pairs. Therefore we divide the summation with $N_m \times \sum_{j=1}^{N_c}\sum_{k=1}^{N_{j,m}} K$ to normalize this value between (0, 1).

As an example, consider the situation in figure 9.

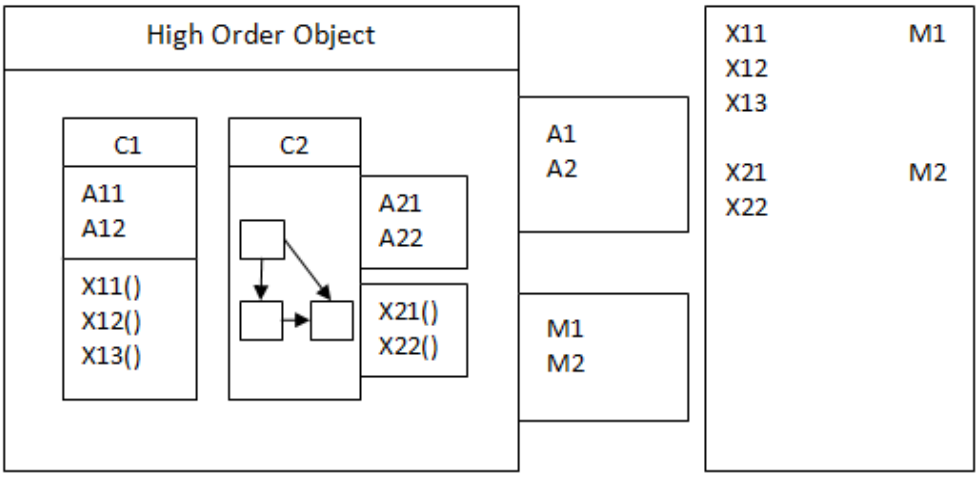

Figure 8: A simple HOO illustrating all the possible pairings.

For this object, we try to find all possible pairings of M1 and M2 with X_{ij}. We then divide this the actual value by the maximum to determine cohesion.

Therefore, total cohesion in a HOO is given by:

$$\frac{\frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Na} Is_{Coupled\ (i,j)}}{N_m * N_a} + \frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Nm} Is_{Coupled\ (i,j)}}{{N_m}^2 - N_m} + \frac{\sum_{i=1}^{Nm}\sum_{j=1}^{Nc}\sum_{k=1}^{N_{j,m}} Is_{Coupled\ (i,k)}}{N_m \times \sum_{j=1}^{N_c}\sum_{k=1}^{N_{j,m}} K}}{3}$$

Expression 1 and 2 determine coupling of HOO due to M-M and M-A interactions. We divide the entire summation by 3 as M-M, M-A and M-C are normalized between (0, 1).

3. Limitations and future work

In coupling metric, while transforming PO-HO graph into PO-PO graph, we couple PO with all the methods of HOO. Although highly probable, this may not always be the case.

For the cohesion metric, M-C cohesion doesn't consider cohesion between methods in HOO and attributes in the component classes. Furthermore, M-C cohesion may also be represented as coupling between HOO and component classes. This direction can be explored in the future. The advantage with this approach is that it considers indirect effect of cohesion induced due the interactions of component classes.

4. Conclusions

In this paper, we provided a way to estimate coupling and cohesion of a HOOM at any given phase of the design. Our coupling metrics consider coupling due to inheritance and coupling due to indirect interactions. For the coupling metric, we considered HO-HO, PO-HO and PO-PO coupling. Even though the final system only consists of PO's, measuring HO-HO and HO-PO coupling is important as it enables an analyst to make design decisions at any phase of HOOMT.

We developed a normalized cohesion metric to explain the cohesion in a PO and HOO. These metrics enable an analyst to make unbiased design decisions while developing a system using HOOMT. Although there are some limitations of this work, we tried to future directions and possible improvements wherever possible.

5. References

1) Bieman, J., & Kang, B. (1994). Cohesion and reuse in an object-oriented system. *ACM Symposium on Software Reusability*, (pp. 259–262).

2) Briand, L., Daly, J., & Wust, J. (1998). A unified framework for cohesion measurement in object-oriented systems. *Empirical Software Engineering 3* , 65–117.

3) Briand, L., Daly, J., & Wust, J. (1999). A unified framework for coupling measurement in Object-Oriented systems. *IEEE Transactions on Software Engineering* , 91-121.

4) Chae, H., Kwon, Y., & Bae, D. (2000). A cohesion measure for object-oriented classes. *Software: Practice and Experience 30* , 1405-1431.

5) Chidamber, S. R., Kemerer, C. F., & Mit, C. (1994). A metrics suite for object oriented design. *IEEE Transaction Software Engineering* , 476-493.

6) Chidamber, S., & Kemerer, C. (1991). Towards a Metrics Suite for Object. *Sixth ACM Conference on Object Oriented Programming, Systems, Languages and Applications*, (pp. 197-211).

7) Li, H. (2008). A Novel Coupling Metric for Object-Oriented Software Systems. *IEEE Transactions on Software* , 609-612.

8) Mao, M., & Jiang, Y. (2008). Coherent Object-Oriented (OO) Software Metric Framework. *Software Engineering proceedings of the IEEE.*

9) Markose, S., Liu, X. F., & McMillin, B. (2008). A Systematic Framework for Structured Object-Oriented Security Requirements Analysis in Embedded Systems. *Embedded and Ubiquitous Computing* , 75-81.

10) Myers, G. (1976). *Software Reliability: Principles and Practices.* Wiley.

11) Pfleeger, S., & Atlee, J. (2006). *Software Engineering: Theory and Practice, 3rd ed.* Pearson Prentice Hall.