T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NESNEYE YÖNELİK YAKLAŞIMLA GERÇEKLEŞTİRİLEN YAZILIMLARDA SINIF UYUM ÖLÇÜTLERİNİN İNCELENMESİ VE YENİ BİR SINIF UYUM ÖLÇÜTÜ ÖNERİSİ

YÜKSEK LİSANS TEZİ

Bilg. Müh. Ahmet ARSLAN

Enstitü Anabilim Dalı	**:**	**Bilgisayar ve Bilişim Mühendisliği**
Enstitü Bilim Dalı	**:**	**Bilgisayar ve Bilişim Mühendisliği**
Tez Danışmanı	**:**	**Yrd. Doç. Dr Hayrettin EVİRGEN**

Ağustos 2012

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NESNEYE YÖNELİK YAKLAŞIMLA GERÇEKLEŞTİRİLEN YAZILIMLARDA SINIF UYUM ÖLÇÜTLERİNİN İNCELENMESİ VE YENİ BİR SINIF UYUM ÖLÇÜTÜ ÖNERİSİ

YÜKSEK LİSANS TEZİ

Bilg. Müh. Ahmet ARSLAN

Enstitü Anabilim Dalı : **Bilgisayar Ve Bilişim Mühendisliği**

Enstitü Bilim Dalı : **Bilgisayar Ve Bilişim Mühendisliği**

Bu tez 14/08/2012 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Yrd. Doç. Dr Hayrettin EVİRGEN	Prof. Dr. Emin GÜNDOĞAR	Yrd. Doç. Dr. Ali GÜLBAĞ
.....	.....	.....
Jüri Başkanı	**Üye**	**Üye**

ÖNSÖZ

Tez çalışmalarımın gerçekleşmesinde çok büyük desteği olan tez danışmanım Sayın Yrd. Doç. Dr. Hayrettin EVİRGEN'e yüksek lisans eğitimim boyunca bana verdikleri eğitim ve destekleri nedeniyle Sakarya Üniversitesi Bilişim ve Bilgisayar Bilimleri Fakültesi Bilgisayar Mühendisliği bölümünde görevli olan tüm öğretim üyelerine ve araştırma görevlisi arkadaşlara teşekkür ederim.

Yüksek lisans eğitimim boyunca destek ve sevgisini hep hissettiğim bana kuvvet veren eşime en içten dileklerimle teşekkür ederim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

LCOM	: Lack of Cohesion of Methods
Co	: Connectivity
TSU	: Tüm Sınıf Uyumu metriği
Coh	: Cohesion

ŞEKİLLER LİSTESİ

ÖZET

Anahtar kelimeler: Sınıf uyum ölçütü, cohesion metriği, sınıf uyum metriği.

Yazılım sektörünün gelişmesi ile yazılımlarda kalite ihtiyacı kendini hissettirmiştir. Kaliteyi sağlamak için çeşitli çalışmalar yapılmış bu çalışmaların sonucunda kaliteyi ölçme ihtiyacı doğmuştur. Ölçme işlemi yazılım metrikleri aracılığıyla gerçekleştirilmektedir. Bu metrikler arasında; nesneye dayalı programlamanın temel taşı olan sınıfların, sahip olduğu metot ve özniteliklerin birbiri ile uyumunun derecesini ölçen sınıf uyumu(cohesion) metrik kümesi önemli bir yere sahiptir. Sınıf uyumunu ölçmek için bu güne kadar yapılan çalışmalar göreceli olarak bir başarı sağlasa da henüz olgunluk seviyesine ulaşmamıştır.

Bu çalışmada temel sınıf uyum metrikleri teorik olarak incelenmiş, deneysel olarak incelemek için program yazılmıştır. İncelenen metriklerin zayıf noktaları belirtilmiş ve bu zayıflıklara karşı yeni metrik önerilmiştir.

ANALYSIS OF COHESION METRICS IN OBJECT ORIENTED SOFTWARE AND PROPOSE FOR A NEW METRIC

SUMMARY

Key Words: Cohesion Metrics, software quality

With the development of the software industry, high quality software needed. Various efforts were made to ensure that quality. As a result of these studies, to measure software quality became a necessity. The measuring process is carried out by means of software metrics. Cohesion metric is important in these metrics. This metric measures the degree of alignment of classes with its methods and attributes. Studies to measure the cohesion of the class have a relative success, but have not yet reached level of maturity.

This study analyzed the theoretical base of cohesion metrics. To examine metrics experimental, a program has been written. To solve weak points and weaknesses in the metrics a new metric is proposed.

BÖLÜM 1. GİRİŞ

Bilgisayar yazılımlarının her geçen gün insan hayatının daha çok alanında kullanılmasıyla, geliştirilen yazılım sistemleri daha büyük çaplı ve daha karmaşık bir hale gelmiştir. Geliştirilen sistemlerden beklentiler ve talepler artmış ve yazılım sistemleri daha karmaşık ve kontrol edilebilirliği daha zor hale gelmiştir. Bunun sonucu olarak da uygulama geliştirme ve bakım maliyetleri, geliştirme zamanları, test süreçleri ve yazılımın hata oranları da artmaktadır.

Yazılım sistemlerinde oluşabilecek sorunların ekonomik kayıpların yanı sıra, tıbbi cihazlar, nükleer tesisler, ulaştırma gibi alanlarda insan hayatı kaybına yol açabilecek sonuçlara da sebep olması mümkündür. Geliştirilen yazılımların başarısızlığa uğraması, bir süre sonra işlevini yitirerek artık kullanılmaması veya yerine yenisinin yazılması gibi durumlar sık karşılaşılan durumlar olmaya başlamıştır. Yazılım projelerinin maliyetlerinin artması, geliştirilen yazılımların belli bir kalite standardına uygun olması düşüncesini doğurmuştur. Yazılımda kalite kavramı son yıllarda üzerinde en çok çalışılan konulardan biri haline gelmiştir.

Yazılımda kalite konusu, herkes tarafından sezgisel olarak anlaşılabilen fakat fiziksel ürünlere oranla ölçümünün zor olduğu soyut ve öznel kavramlardan oluştuğundan dolayı tam olarak tanımlanamayan bir kavramdır. Bir yazılımın kalitesiyle ilgili son kullanıcıların veya uygulama geliştiricilerin tecrübelerinden alınan geri dönüşlerle genel bir kanıya varılabilir, fakat nicel bir değerlendirme yapılmış olmaz. Bir ürünün kalitesinin tam olarak tanımlanabilmesi için o ürün hakkında gelişmiş ölçme araçlarına ve herkes tarafından kabul edilen karşılaştırma değerlerine ihtiyaç vardır. Bu karşılaştırma değerleri yazılım alanında yazılım metrikleri olarak tanımlanmaktadır. Yazılım metrikleri yazılımın çeşitli özelliklerinden elde edilen ölçüm değerleridir.

Yazılım kalitesinin ölçümünde kullanılan yazılım metrikleri;
- Sistemin uygun bir şekilde tasarımında ve geliştirilmesinde yazılımcılara yol gösterir,
- Yazılım geliştirme sürecinde nesneler arasındaki ilişkiler hakkında ölçülebilir değerler üreterek yazılımın durumu hakkında geliştiriciyi bilgilendirir.
- Yazılım geliştirme süreçlerini tahmin etmede ve geliştirilen yazılımın kalitesini değerlendirmede kullanılır.
- Proje yöneticilerine yazılımın denetlenebilmesinde yardımcı olur.

Yazılım alanında geliştirilen ürünler elle dokunulamayan ve kolay ölçülemeyen sanal ürünler olduklarından fiziksel ürünlere oranla daha fazla ölçüte ihtiyaç duyarlar. Belli bir değer hakkında yapılan ölçüm dolaylı olarak yazılımın farklı alanlarıyla ilgili değerlendirmeler verebilir. Bu değerlendirmeler sistemin hangi biriminin diğer hangi birimini etkileyeceğini bilen yazılımcının ortaya çıkarması gereken değerlendirmelerdir. Bunların haricinde yazılım ölçümleri fiziksel ürünlerin ölçülmesinde kullanılan ölçütlere göre tanımlanması daha zor ve anlaşılması daha karmaşık ölçümlerdir. Tüm bu dezavantajlar, yazılımda kalite ve metrikler hakkında çalışmaların bugüne kadar az olmasına sebep olmuştur.

Günümüzde ise yazılım projelerinin büyümesi, bakım masraflarının artması, geliştirilen yazılımların tıptan endüstriye, enerjiden bankacılık ve finans uygulamalarına kadar son derece kritik ve hayati süreçlerde kullanılması sonucu yazılım kalitesi kavramı, hem akademik hem de endüstriyel alanda en çok çalışılan konular arasına girmiştir. Hata oranı yazılımın hassasiyetinin kaldırabileceği kadar düşük, gelişime ve değişime açık, kaliteli yazılımlar üretmek için çaba sarf edilmektedir. Yazılım maliyetlerini azaltmak, yazılımın hata oranını düşürmek, tekrar kullanılabilir, bitirme süresi ve maliyeti öngörülebilir yazılımlar üretmek; ancak yüksek kalitede yazılımların geliştirilmesiyle sağlanabilir (Eski,2011)

Yazılım kalitesi, farklı kişilerce farklı anlamlar yüklenebilen bir kavramdır. Geliştiricinin gözüyle kaliteli bir yazılım, geliştirme ve bakım maliyetleri düşük ve tekrar kullanılabilir sınıflara sahip bir yazılımı ifade ederken, müşterinin gözüyle

kolay kullanımı olan, müşterinin tüm isteklerini karşılayan, hatasız ve yeterli performansa sahip bir yazılımdır. Bu farklı bakış açıları literatürde içsel bakış açısı ya da geliştirici gözüyle kalite, dışsal bakış açısı ya da müşteri gözüyle kalite olarak sınıflandırılmıştır. Biz bu çalışmada yazılımın iç özelliklerinin kalitesi üzerinde duracağız. (Baldassari, 2004)

Nesneye dayalı programlamada uyumluluk; bağımlılık ve karmaşıklıkla beraber yazılımın en önemli iç özelliklerindendir. Bu tezin kapsamı dışında olan bağımlılık nesneye dayalı programlamada sınıflar arasındaki bağımlılığın derecesini, karmaşıklık yazılımın içyapısının anlaşılmasının zorluğunu ölçer. Uyumluluk ise sınıfın üyelerinin birbiriyle olan ilişkisini ölçer. Kaliteli yazılımdan beklenen uyumluluğun yüksek, karmaşıklığın ve bağımlılığın düşük olmasıdır (Buzluca vd 2008)

Yazılımın iç özellikleri olarak bağımlılık ve karmaşıklık alanında yapılan metrik tanımlama çalışmaları belirli bir noktaya gelmiş olmasına karşın uyum ölçütünün diğerlerine göre daha sanal ve ölçülmesi zor olduğundan bu konuda istenilen noktaya henüz ulaşılamamıştır. Ortaya konulan uyum ölçütleri incelendiğinde hepsinin uyumun bir yanıyla ilgilendiği görülmektedir. Bu ölçütlerden bazıları üye öznitelik kullanımını, bazıları üye metot çağrımlarını, bazıları bunların birleşimi, bazıları aynı özniteliklere ulaşan metrik gurupları üzerinde durmuştur. Tanımlanan metrikler araştırılmaya devam etmektedir (Basili,1999).

Birçok araştırmacı sınıf uyumunu, kendi bakış açılarına göre tanımlamışlardır. Chidamber ve Kemerer tarafından tanımlanan LCOM (Lack Of Cohesion) ölçütü diğer ölçütlere örnek olmuştur(Chidamber,1991). Diğer araştırmacılar LCOM ölçütünü geliştirerek tanımlamışlar Literatürde görüleceği gibi birçok araştırmacı, LCOM ölçütünü tekrar tanımlamıştır. Sınıfın değişkenlerinin çoğu metodları tarafından kullanılıyorsa uyumlu olarak tanımlandığı gibi(Briand 1998a), Metod çiftleri değişkenleri kullanıyorsa uyumlu olarak da tanımlanmıştır(Bieman,1995).

Chidamber ve Kemerer, sınıf uyumunu LCOM (LCOM1 ve LCOM2) ölçütleri ile ölçmeyi önermişlerdir. Ortak özniteliklere ulaşmayan metot çiftleri sayılarak

LCOM1 elde edilir. LCOM2 ise Ortak özniteliklere ulaşmayan metot çiftleri Ortak özniteliklere ulaşmayan metot çiftlerini sayısından ortak özniteliklere ulaşan metot çiftlileri sayısı çıkarılması ile elde edilir. İlk tanımlanan ölçütlerden olması sebebi ile geniş olarak tartışılmıştır (Henderson,1996). LCOM1 de bütün metot çiftleri ortak özniteliğe ulaşıyorken 0 değeri ölçülürken LCOM2 de, eğer ortak özniteliklere ulaşmayan metot çiftlerinin sayısı ortak özniteliklere ulaşan metot çiftlilerinin sayısından küçük veya eşit ise 0 ölçülür.

Li ve Henry, tanımladıkları LCOM3 (Li,1995) ölçütünde özelliklere ulaşan metot çiftlerini ilişkili kabul ederek bu metot çiftlerini kümelemiştir. Bütün metot çiftlerinin bağıntılı olduğu durumda bir küme sayılmakta, bütün metotlar ayrık olduğu durumda metot sayısı kadar ölçüm yapılmaktadır.

LCOM4 u Hitz ve Montazeri LCOM3 ü geliştirerek tanımlamışlardır (Hitz,1995). LCOM 3 e ek olarak metotlar birbirini çağırınca da arasında ilişki oluşmaktadır.

LCOM4 tam uyum değeri olan 1 e ulaştığında metodlar arası ilişki artsa da bunu ölçememektedir. Hitz ve Montazeri LCOM4 1 olduktan sonra artan uyumu ölçmek için Co (Connectivity) ölçütünü tanımlamışlardır.

Henderson-Sellers LCOM5'i önererek metrik geliştirme çalışmalarına katkıda bulunmuşlardır (Henderson,1996). Önceki metrikler metotların ilişkilerini incelerken LCOM5 özelliklerin çağrılma sayısı ile ölçülmektedir.

Briand LCOM5 için tekrar bir tanımlama sunmuştur (Briand 1998a). Yeni tanımlama literatüre Coh olarak geçmiştir. LCOM5'ten farklı olarak bazı özniteliklerin metotlar tarafından erişilmemesi durumunu göz önüne almaktadır(Kurubaş.Ö.).

Tezin birinci bölümündeki girişten sonra, ikinci bölümünde yazılım kalitesinin değerlendirilmesi ve ölçme kavramı üzerinde durulmuş, üçüncü bölümde bugünde kadar önerilen uyum ölçütleri incelenmiş, dördüncü bölümde yeni önerilen uyum ölçütü anlatılmıştır. Beşinci bölümde elde edilen sonuç ve öneriler yer almaktadır.

BÖLÜM 2. YAZILIM KALİTESİ VE ÖLÇME KAVRAMI

2.1. Yazılım Kalitesi

Yazılım kalitesi denildiğinde akla ilk gelen, müşterinin ihtiyaçlarını tam olarak karşılayan, tahmin edilen bütçe ve istenen zamanda tamamlanan, değişim ve bakım istekleri, az maliyetle ve kolaylıkla yerine getirilebilen, tekrar kullanılabilir ve geliştirilebilir yazılımlardır.

Yazılım kalitesi için literatürde birçok farklı tanım yapılmıştır. Bazı tanımlar; kaliteli yazılımları; ihtiyaçları tam olarak karşılayan yazılımlar olarak ifade ederken bazıları ise sınıflar arasında ilişkileri iyi tanımlanmış, bakımı yapılabilen ve genişletilebilen yazılımlar olarak tanımlamıştır. IEEE Yazılım Mühendisleri Terminolojisi Standart Sözlüğü'ndeki tanıma göre yazılım kalitesi; yazılımın geliştirilmesi sırasında kaliteyi sağlamak amacıyla yapılan planlı ve sistematik aktiviteler kümesidir. Buna göre yazılım kalitesi; sistem, bileşen veya sürecin belirtilen isteklere uyum derecesi veya sistem, bileşen veya sürecin müşterinin ihtiyaç ve beklentilerini karşılama derecesi olarak tanımlamıştır. (IEEE Std 610.12). Başka bir tanımda; belirtilmiş isteklere uygunluk, standartlara uygunluk, bunların yanı sıra belirtilmeyen ve varsayılan isteklere uygunluk olarak tanımlanmıştır (KURNAZ vd.).

Yazılımda kalite kavramının birçok farklı tanımı olması; kalite kavramının kişilere göre değişebilen bir terim olmasından ve herkesin konuyu farklı açılardan değerlendirmesinden kaynaklanmaktadır. Yazılım kalitesi müşterilerin; yani yazılımı kullananların, programcıların; yani yazılımı geliştirenlerin, yöneticilerin; yani yazılımın geliştirilmesi için gerekli mali kaynakları sağlayanların bakış açısına göre farklı değerlendirmelere tabi tutulmaktadır. Müşteriler, genel olarak ihtiyaçlarını karşılayan, kullanımı kolay, yönetilen verilerin güvenilirliğini ve bütünlüğünü sağlayan ve yeterli performansa sahip yazılımlara ihtiyaç duyar ve bu faktörlerle ilgilenir. Yazılımı geliştirenler, yazılımın, en az hatayla geliştirilebilmesini, yazılıma

sağlanan teknik destek ve bakımın kolay yapılabilir olmasını ve yazılımın tekrar kullanılabilir olmasını ister. Yazılım yöneticileri ise; yazılımların, daha az zamanda ve daha az maliyetle tamamlanmasını beklerler. Tüm bu kişilerin yazılım kalitesi hakkındaki bireysel eğilimlerini ve kaliteye bakış açısındaki farklılıklarını içerisinde barındıran nesnel yöntemlerle yazılımın kalitesini nicel olarak ölçen, metrik olarak adlandırılan ölçüm kümeleri tanımlanması gerekmektedir.

Yazılımın kalitesinin ölçülmesi ve dolayısıyla kalitenin arttırılması amacıyla tanımlanmış standartlar kümesi olarak da tanımlayabileceğimiz metriklerin büyük çoğunluğu yazılımların müşteriye yansıyan kalite özellikleri ve yazılım geliştirme süreçleri üzerine yoğunlaşmaktadır. İyi tanımlanmış bir metrik kümesinin, yazılımın tasarım aşamasından başlayarak, geliştirme süreci boyunca, kaynak kod içerisindeki tüm sınıfların, sınıflar arasındaki bağımlılıkların, sınıf üyeleri arasındaki ilişkilerin ölçülebilmesi, yazılım kalitesinin tüm bu faktörlere bağlı olarak değerlendirilmesi ve buna bağlı olarak geliştirilebilmesi gerekir.

Yazılım metrikleri sayesinde yazılım geliştiricileri, sınıf üye ve metotlarının doğru tanımlanıp tanımlanmadığını, sınıfların yaptıkları işin kendi içerisinde uyumlu olup olmadığını, sınıfların birbirleri ile olan bağımlılıklarını derecesini anlayabilirler ve geliştirme esnasında yazılımın daha kaliteli olması için bazı kararlar almada bu ölçümleri kullanabilirler. Bu yüzden yazılım metriklerinin verdiği sonuçların neyi ifade ettiğinin bilinmesi ve yorumlanabilmesi gerekmektedir. Bu da yazılım kalitesini ölçmek için tanımlanan metriklerin anlamlı sonuçlar üretmesine ve tanımlanan metriğin sınıfın kalitesini tüm yönleriyle ölçmesine bağlıdır.

2.2. Ölçüm Teorisi

Ölçüm, gerçek dünyadaki varlıkların özelliklerinin, daha önceden tanımlanmış ve kabul görmüş bazı kurallar yardımı ile sayı ve birimler vasıtası ile gösterilmesidir. Burada varlıktan kasıt belli ölçülebilir özellikleri olan herhangi bir şey olabilir. Ağırlığı, hacmi veya boyutu olan bir cisim olabileceği gibi gözle görülemeyen fakat ölçülebilen bir enerji de olabilir. Bir insanın boyunun uzunluğu veya bir cismin

ağırlığı olduğu gibi bir yazılım için kodun satır sayısı da ölçülebilir değerlerdir. Fakat bir yazılımın satır sayısı yazılımın kalitesi açısından herhangi bir fikir vermeyebilir.

Bu açıdan yazılımı geliştiren veya kullanan kişilere belli yönlerde fikir verebilecek ölçüm kümeleri yazılımlar için de tanımlanmıştır ve bu konudaki çalışmalar devam etmektedir. Bir yazılım ölçütü, diğer disiplinlerdeki ölçümlerde olduğu gibi, genel kabul görmüş ölçüm oluşturma kurallarına uygun olmalıdır (Fenton, N).

Ölçme, doğrudan ya da dolaylı ölçme olmak üzere iki şekilde yapılmaktadır. Doğrudan ölçme bir şeyin herhangi bir özelliğinin diğer başka özellikleri yardımı ile ölçülmediği durumlardaki ölçüm işidir. Dolaylı ölçümde ise bir şeyin ölçümü bazı diğer özelliklerin ölçümü yardımı ile yapılır. Sınıf uyumu için bir kalite ölçütü değerlendirildiğinde veya geliştirilmek istendiğinde sınıfın üyeleri arasındaki ilişkileri ölçen doğrudan ölçüm yöntemleri ve sınıflar arasındaki ilişkileri ölçen dolaylı ölçüm yöntemlerini de içinde barındıran bir sınıf uyumu ölçüm metriği geliştirilmesi gerekmektedir.

Ölçüt geliştirme; ilk olarak ölçütün tanımlanması ve diğer ölçütlerden farkının ortaya konması; yani ne olduğunun ve ne olmadığının belirlenmesi daha sonra da ölçütü kanıtlayacak deneysel çalışmaların yapılması çalışmalarından oluşmaktadır.

Yazılım ölçütü tanımlamayı Fenton şu şekilde açıklamıştır:
- Gerçek dünya varlığı için ölçülecek özelliğin tespiti,
- Bu varlık için deneysel ilişkilerin belirlenmesi,
- Her bir deneysel ilişki için sayısal değerlerin atanması,
- Varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunup korunmadığının kontrol edilmesi(Fenton, N).

2.3. Metrikler

Sürekli kaliteyi sağlamak için kullanılan yöntemlerin başında kalitenin ölçülmesi gelir. Bu durum kalitesi arttırılmak istenen her nesnede olduğu gibi yazılım ürünlerinde de geçerlidir. Yazılımda kalitenin ölçülmesi; yazılımların daha iyi anlaşılması, hakkında daha doğru bilgi sahibi olunması ve geleceğe yönelik bazı çıkarsamaların yapılması açısından önemlidir. Yazılım metrikleri, en basit anlamda, kalitenin ölçülmesi için kullanılan sayısal değerlerdir. Bu sayısal değerler dinamik olarak yazılımların çalışma zamanında elde edilebildiği gibi tersine mühendislik yöntemleri kullanılarak statik olarak kaynak koddan hesaplanarak da elde edilebilir

Yazılım metrikleri; projenin kalitesinin belirlenmesinin yanı sıra, projenin büyüklüğü ve karmaşıklığının da belirlenmesine katkı sağlayarak ileriye dönük doğru kararlar alınması konusunda geliştiricilere ve yöneticilere yardımcı olur. Bu yönüyle; yazılımın bütçe, test ve bakım maliyetlerinin belirlenmesinde, yazılımdaki kritik kısımların bulunmasında, tasarım hatalarının tespit edilmesinde kullanılabilir.

Günümüzde herkes tarafından kabul gören yazılım metrikleri henüz bulunmamaktadır. Bunun altında yukarıda da değindiğimiz gibi yazılımlardaki bazı özelliklerin kişiden kişiye değişen sezgisel anlamlar ifade etmesi, herkesin kendi bakış açısıyla yazılımı değerlendirmesi gibi etkenler bulunmaktadır. Bu durum yazılım metrikleri oluşturmada iyi modeller oluşturulmasının önünde en büyük engellerden birisidir. Çünkü oluşturulacak modelin herkesin bakış açısını içerisinde barındırması zor bir iştir.

Yazılım, özelliklerin ölçme ve değerlendirilmesi açısından karmaşık bir yapıya sahiptir. Aynı zamanda müşterilerin, geliştiricilerin ve yazılım yöneticilerinin bakış açılarının hepsinin birden değerlendirilerek bir metrik kümesi tanımlamak da bu zorluğun katlanmasına sebep olmaktadır. Bu yüzden günümüze dek herkes tarafından kabul gören bir yazılım metrik kümesi henüz tanımlanamamıştır. Bu çalışmada bu alanda yapılan çalışmalara katkı sağlamak amacıyla tüm bu özellikleri

içinde barındıran ve herkes tarafından kabul görmesi amaçlanan bir sınıf uyum metriği geliştirilmeye çalışılmaktadır

2.4. Sınıf Uyumu

Sınıf uyumu, sınıf üyelerinin ilişki derecesine işaret etmektedir. ilişkililik, sınıfın sunduğu metotların işlevsel olarak aynı amaca hizmet etme açısından benzerliği anlamını taşımaktadır. Genelde tüm yazılım mühendisliği çalışmaları yüksek uyumu sınıflar için erişilmesi zaruri özellik olarak tanımlamışlardır. Çünkü oluşturulacak sistemin bakımı ve devamlılığı, sistemin yapıtaşları olan sınıfların ne kadar uyumlu olduğu ile yakından ilgilidir. Uyumsuz işlevselliğin bir arada tutulması neticesinde; sistemin bir kısmı işlerlik kazanırken diğer kısmının sorun oluşturmasına sebep olabilmektedir.

Halbuki aynı amaca yönelik işlevsellik bir kapsam içine yerleştirildiğinde sorun yerel bir problem haline gelmekte ve uygun bir şekilde diğer kısımlara sirayet etmeden düzenlenebilmektedir (Kurubaş,2010).

2.5. Sınıf Uyumunun Kullanım Alanları

Son yıllarda yazılım kalitesi ölçütleri yazılım dünyasında çok önem kazanmaktadır. Ölçütler, yazılım geliştiricilere ve yöneticilere yazılımın birçok yönden değerlendirilmesine imkân tanır. Bu değerlendirmeler, iyi ve kaliteli bir yazılım geliştirme yolunda yardımcı rol oynarlar.

Ölçme eylemi, somut veya soyut bir varlığın sahip olduğu bir özelliğini, sayısal veya derecelendirilmiş bir veri olarak ifade etmektir. Ölçülecek özelliği ölçme yolu ve biçimine ölçüt, bir ölçüt kullanılarak yapılan ölçme eyleminin sonucunda elde edilen veriye ise ölçüm adı verilir. Ölçme eylemi işimize yarayacak, anlamlı sonuçlar elde etmek için yapılır. Amaç yazılımın kalitesini iyileştirmek olarak belirlendiği zaman,

yazılımın kalitesi yazılım yaşam döngüsünün çeşitli aşamalarında ölçülmeli, elde edilen ölçümler karşılaştırılmalı ve değerlendirilmelidir.

Ölçme yapabilmek için önce bir veya birkaç ölçüt belirlenmelidir. Bu aşamada tanımlanması bile zor bir kavram olan yazılım kalitesi ile ilgili üzerinde anlaşmaya varılmış özellikler bulmanın zorluğu ile karşılaşılır. Yazılım yaşam döngüsünün erken aşamalarında ölçüme başlayabilmek için, içsel kalite özelliklerinin kullanılması gerekecektir. Bağlaşım (coupling) (Stevens vd., 1974) ve uyum (cohesion) (Bunge, 1977), bu amaçla kullanılan en temel özelliklerdir. Nesneye Yönelik Programlama (NYP) yaklaşımının doğru kullanımı ile oluşturulan yazılımlarda bağlaşımın düşük ve uyumun yüksek olması beklenir (Lieberherr vd., 1988). Literatürde bağlaşım ve uyum için birçok ölçüt bulunmaktadır ancak yazılım ile ilgili ölçütlerin deneysel açıdan doğrulanması kolay olmadığından bu yönde daha çok çalışma yapılmasına gerek vardır (Briand vd., 1999).

Ölçüm verilerini anlamlı bilgilere dönüştürmek için yapılması gereken yorumlama işlemi zor olabilir. Literatürde “intelligence barrier” (Kriz, 1988) olarak adlandırılan bu zorluk, matematiksel ve istatistiksel yöntemlerle aşılmaya çalışılabilir (Ebert ve Dumke, 2007), ancak bu yöntemlerin de kolay olduğu söylenemez.

Yazılım kalitesinin değerlendirilmesi için yapılan bir çalışmada (Bansiya ve Davis, 2002), Nesneye Yönelik (NY) tasarımlarda yüksek-seviye tasarım kalitesi özelliklerini değerlendirmek için iyileştirilmiş bir hiyerarşik model tanımlanır. Bu modelde sınıfların, nesnelerin ve ilişkilerinin yapısal ve davranışsal tasarım özellikleri NY tasarım ölçütleri kullanılarak değerlendirilmektedir. Diğer bir çalışmada (Briand vd., 2000) amaç literatürdeki NY tasarım ölçütlerinin özellikle hata eğilimliği konusunda deneysel geçerliliğini gerçekleştirmek olmuştur. Elde edilen bir sonuç; literatürdeki çoğu ölçütün karşılaştırılabilir fikirlere ve hipotezlere dayalı olduğu ve bu nedenle gereksiz olabilecekleri görüşüdür. “uyum”, hata eğilimliği üzerinde anlamlı bir etkiye sahip gözükmüyor olarak değerlendirilmiştir.

BÖLÜM 3. SINIF UYUMU METRİKLERİNİN İNCELENMESİ

Bu bölümde sınıf uyum metrikleri teorik ve deneysel olarak incelenecektir. Fenton ’un “varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunup korunmadığının kontrol edilmesi” (Fenton, N) gerektiği tezinden hareketle, temel metrikler incelenmiştir. Metriklerin sınıf uyumu arttıkça sezgisel olarak sınıf uyum değerlerinin de artması beklenmektedir. Bu amaçla metrikler teorik incelenmiş bunun yanı sıra bir program geliştirilerek deneysel olarak da incelenmektedir.

3.1. Artan Sınıf Uyumu Grafiğinin Hazırlanması

Sınıf uyum metriklerini inceleme yöntemlerinden birisinin de sınıfların uyumu arttıkça nasıl değiştiğini incelemek olduğu düşünülmektedir. Bu inceleme için bir program geliştirilmiştir. Bu program bir sınıfın uyumu arttıkça, sınıf uyumunun değerini hesaplamakta ve sınıf uyum metriklerini karşılaştırılabilmek için grafik oluşturmaktadır.

Grafik dört aşama sonucu elde edilmektedir.

- Minimum uyumlu bir sınıf oluşturulması
- Sınıf dizisi oluşturulması
- Dizideki sınıfların uyumu hesaplanarak başka bir diziye aktarılması
- Artan sınıf uyumu grafiği oluşturulması

Minimum uyumlu sınıf istenen sayıda metodu ve özniteliği içerebilir. Metot sayısı “m”, öznitelik sayısı “o” olarak belirtilir. Oluşturulan sınıfın sınıf uyumundaki değişimi gösterebilmesi için minimum uyum değerinde olması seçilmiştir. Minimum

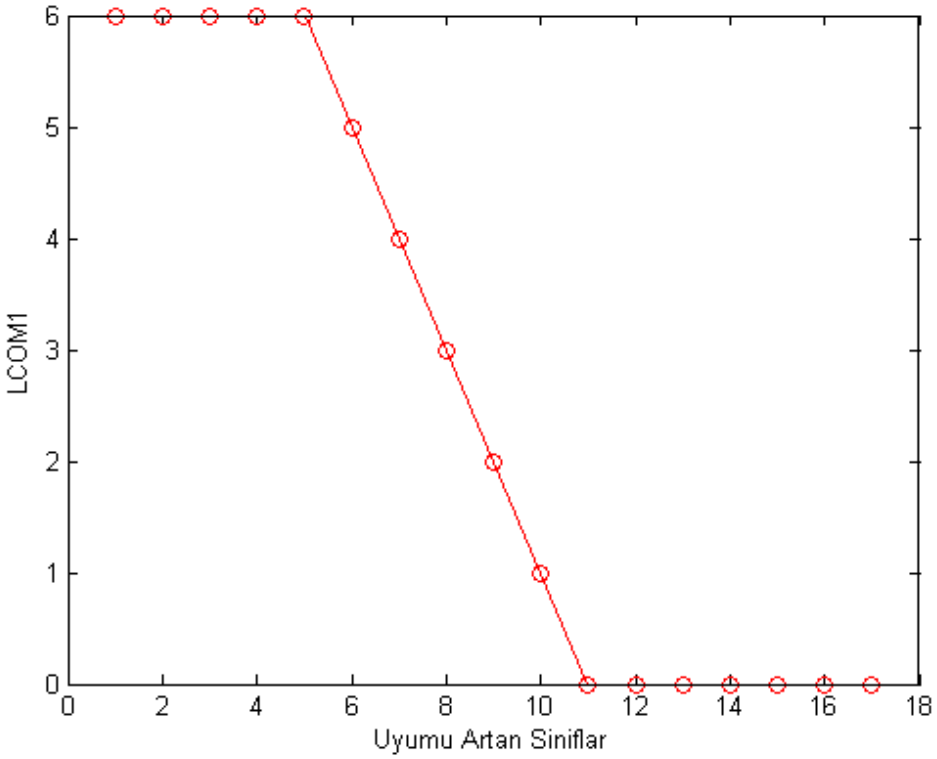
sınıf uyumunun sağlanması için hiçbir metodun özniteliklere ulaşmaması gerekmektedir.

Sınıf dizisi m*o+1 boyutunda oluşturulur. Oluşturulan sınıfın metot ve öznitelik arası maksimum bağıntı miktarını m*o formülü verir. Sınıf dizinin ilk elemanına atanır. Daha sonra sıra ile sınıfın kopyaları, her adımda bir metodun bir özniteliğe daha ulaşması sağlanarak, dizinin bir sonraki elemanına atanır. Bu bütün metotlar bütün özniteliklere ulaşıncaya kadar devam eder.

Oluşturulan dizideki sınıfların sınıf uyum değerini hesaplamak için aynı boyutta dizi oluşturulur ve bu diziye sınıf dizisindeki sınıfların sınıf uyum değerleri atanır. Her bir metrik için ayrı ayrı sınıf uyum değerlerinin tutulduğu dizilerden oluşturulmaktadır.

Sınıf uyum değerlerinin tutulduğu dizi grafik olarak gösterilmektedir. Bu şekilde belirtilen sayıda metot ve özniteliği olan sınıfın, sınıf uyumu arttıkça aldığı değerler elde edilmektedir.

Aşağıda 4 metot 4 özniteliği olan bir sınıfın bu yöntem ile ölçülen LCOM1 değerlerinin grafiği yer almaktadır. Dikey eksen LCOM1 değerini yatay eksen dizideki sınıfların uyum değerini göstermektedir. LCOM1 incelenirken grafik yorumlanacaktır.



Şekil 3.1. 4 Metot 4 özniteliği olan bir sınıfın artan sınıf uyum grafiği

3.2. Sınıfların Gösterim Yaklaşımı

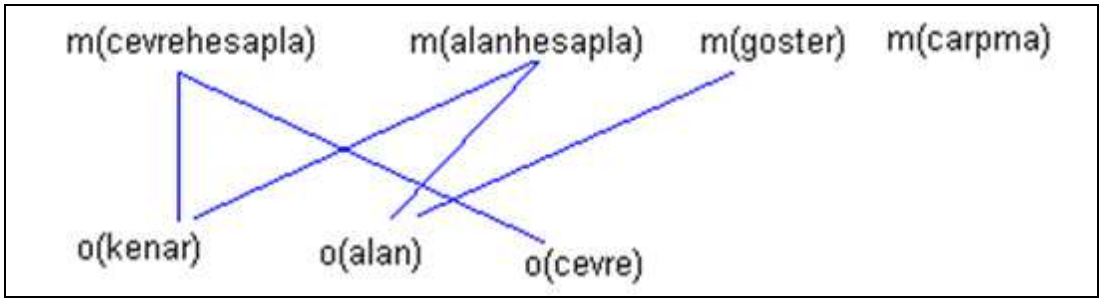
Bu çalışmada sınıflar sembolik olarak ele alınacaktır. Örnek olarak Şekil 3.2. de verilen kod Şekil 3.3. te sembolik olarak gösterilecektir. Daha sonraki örnekler de aynı sembolik gösterim yaklaşımı kullanılacaktır.

```
class Kare
{

  public int kenar;
  public double alan;
  public double cevre;

public void alanhesapla()
{
  alan=kenar*kenar;
}

Public void cevrehesapla()
{
  cevre=kenar*4;
```

```
}

Public void kenarata(int a)
{
  kenar=a;
}

Public void goster()
{
  Console.WriteLine("alan="+alan);
}

Public double carpma(inta,int b)
{
  return a*b;
}
}
```

Şekil 3.2. Örnek kare sınıfı C# kodu

Örnekte kare sınıfında kenar, alan, çevre özniteliği cevrehesapla, alanhesapla, göster, çarpma metotları vardır. Kare sınıfı metot ve öznitelikleri arasındaki ilişkiler Şekil 3.3. te gösterilmiştir.



Şekil 3.3. Kare sınıfının sembolik gösterimi

3.3. Sınıf Uyum Metriklerinin İncelenmesi

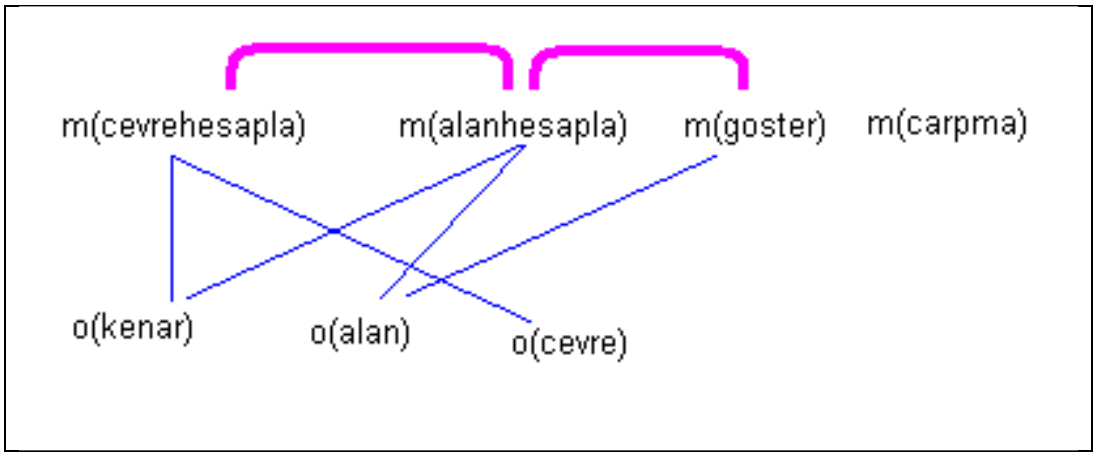
Literatüre geçmiş birçok uyum metriği vardır. Çalışmada eksik yönleri olduğu düşünülen temel sınıf uyum metrikleri incelenecektir.

3.3.1. LCOM1

LCOM1(Lack of Cohesion of Methods) metriği tanımlanan ilk sınıf uyum metrikleri olarak literatüre girmiştir. Chidamber ve Kemerer tarafından önerilmiştir(Chidamber ve Kemerer, 1991).Chidamber ve Kemerer sınıf uyumunu açıklamak için Bunge (Bunge,1977) benzerlik üzerine kuramını kullanmışlardır. Bunge; benzerlik, ortak özellikler kümesidir der. Chidamber ve Kemerer buradan yola çıkarak sınıf uyumu ölçüsünü metotların ortak kullandığı özelliklerin sayısı olarak açıklarlar.(Ertemel,2009) İlk haliyle kullanımı uygun olmasa da literatürün başlangıcı olduğu için önemlidir. Sınıf uyumunun eksikliğini ölçmektedir. Sınıf uyumunu sınıftaki metotların ilişkisini inceleyerek ölçer. Aynı özniteliğe erişen metotları ilişkili kabul eder.(Kurubaş,2011)

LCOM1 de amaç uyum eksikliğinin derecesini bulmaktır. Herhangi bir ortak özniteliğe ulaşmayan başka bir ifade ile atama yada okuma yapmayan metot çiftlerinin toplamı LCOM1 değerini verir. LCOM1 değerinin yüksek olması uyumun düşük olduğunu gösterir. Bu duruma ters uyum ölçümü denir. LCOM1 değeri düştükçe uyum artar. Maksimum uyuma 0 değerinde ulaşılır. 0 değeri bütün metotların birbiri ile ilişkili olması durumunda elde edilir.

Aşağıda verilen örnek sınıf LCOM1 metriğine göre ölçülecektir.



Şekil 3.4. Dört Metotlu 3 öznitelikli örnek sınıf

Cevrehesapla ve alanhesapla metotları aynı özniteliğe ulaştığı için arasında ilişki oluşmuştur. Aynı durum alanhesapla ile goster arasında vardır. Çarpma metodu sınıf içinden hiçbir üyeye ulaşmadığı görülmektedir. LCOM1 metriğinin değerini veren ortak bir özniteliğe ulaşmayan metotlar: çarpma-göster, çarpma-alanhesapla, çarpma-cevrehesapla, cevrehesapla-goster metotlarıdır. Toplam 4 metot ilişkili değildir.

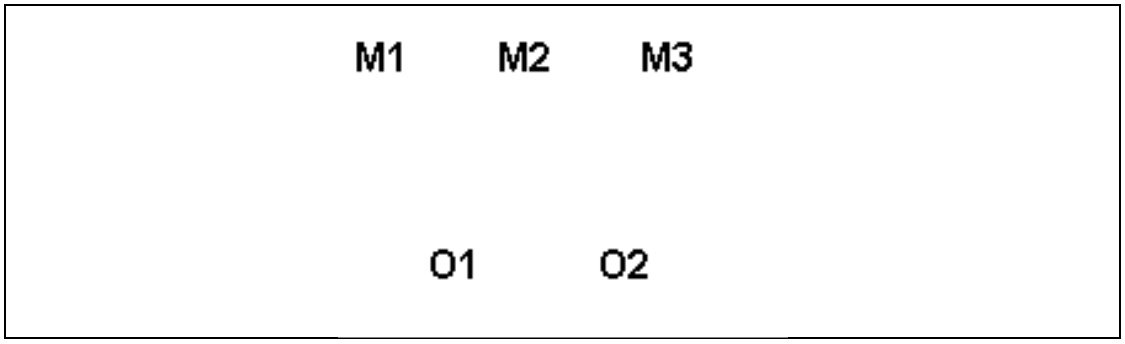
LCOM1=4.

LCOM1 değeri herhangi bir ortak özniteliğe ulaşmayan metotların toplamı olduğu için maksimum LCOM1 değeri, metotların hiçbirisinin bağıntılı olmadığı durumda, metotların arasında oluşturabilecek maksimum metot çifti sayısıdır. Buda metot sayısı m olarak alındığında maksimum LCOM1 aşağıdaki formül ile verilebilir:

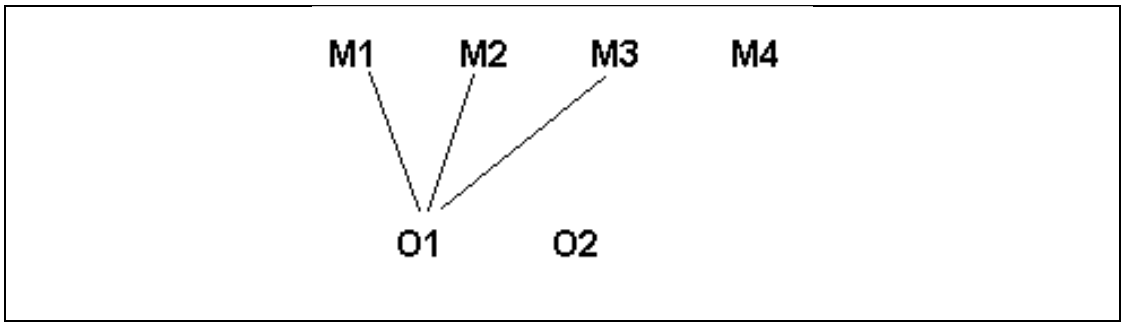
MAX(LCOM1)=m*(m-1)/2 dir.

LCOM1 normalize edilmemiştir. 3 metotlu bir sınıfın LCOM1 değeri 3*2/2=3 ile 0 arasında, 5 metotlu bir sınıfın LCOM1 değeri 5*4/2=10 ile 0 arasında, 10 metotlu bir metotlu bir sınıfın LCOM1 değeri 10*9/2=45 ile 0 arasında ölçülür.

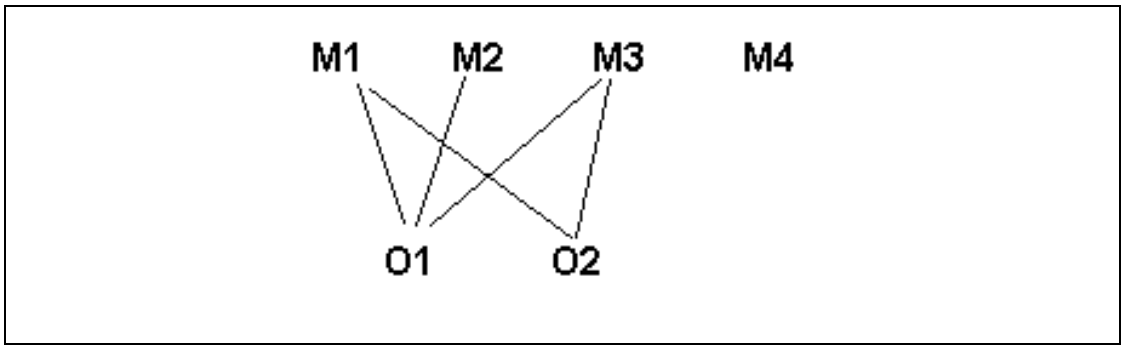
Aşağıdaki şekillerde farklı özellikte olan sınıflar aynı LCOM1 uyum değerine sahiptir. LCOM1 metriği normalize edilmediği için LCOM1 değerleri aynı olsa da sezgisel olarak farklı uyum değerlerine sahip olmaları gerekmektedir.



Şekil 3.5. 3 Metotlu 2 öz nitelikli LCOM1=3 olan sınıf



Şekil 3.6. 4 Metotlu 2 öz nitelikli LCOM1=3 olan sınıf



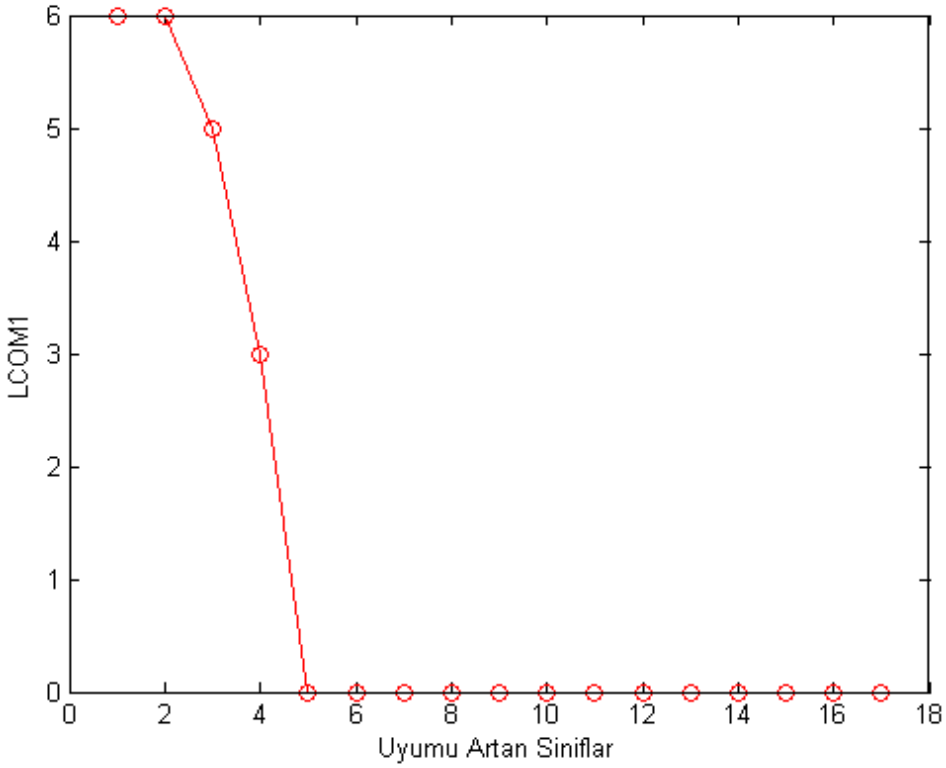
Şekil 3.7. 4 Metotlu 2 öznitelikli LCOM1=3 olan sınıf

Şekil 3.6. ve Şekil 3.7. te Metotların öznitelikleri çağırması daha fazla olmasına rağmen LCOM1=3 değeri artmamaktadır.

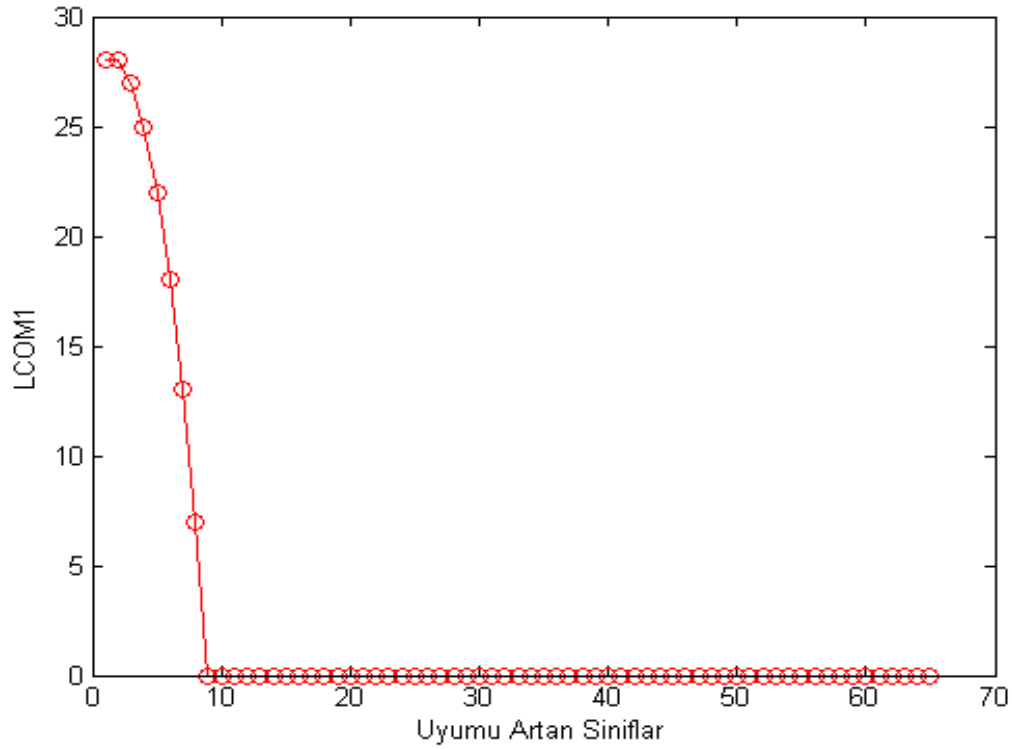
Şekil 3.6. ve Şekil 3.7. te metotların özniteliğe ulaşımı artmış fakat uyum değerindeki beklenen artma görülmemiştir.

Örneklerden anlaşılacağı gibi sadece LCOM1 değerinin bilinmesi, sınıfın uyumunun derecesi hakkında fikir vermemektedir. 10 metotlu bir sınıfta LCOM1=3 nerede ise maksimum uyumu göstermekte ise de 3 metotlu bir sınıfta LCOM1=3 maksimum uyumsuzluğu göstermektedir. Bu durum LCOM1 metriğinin normalize edilmemesinden kaynaklanmaktadır.

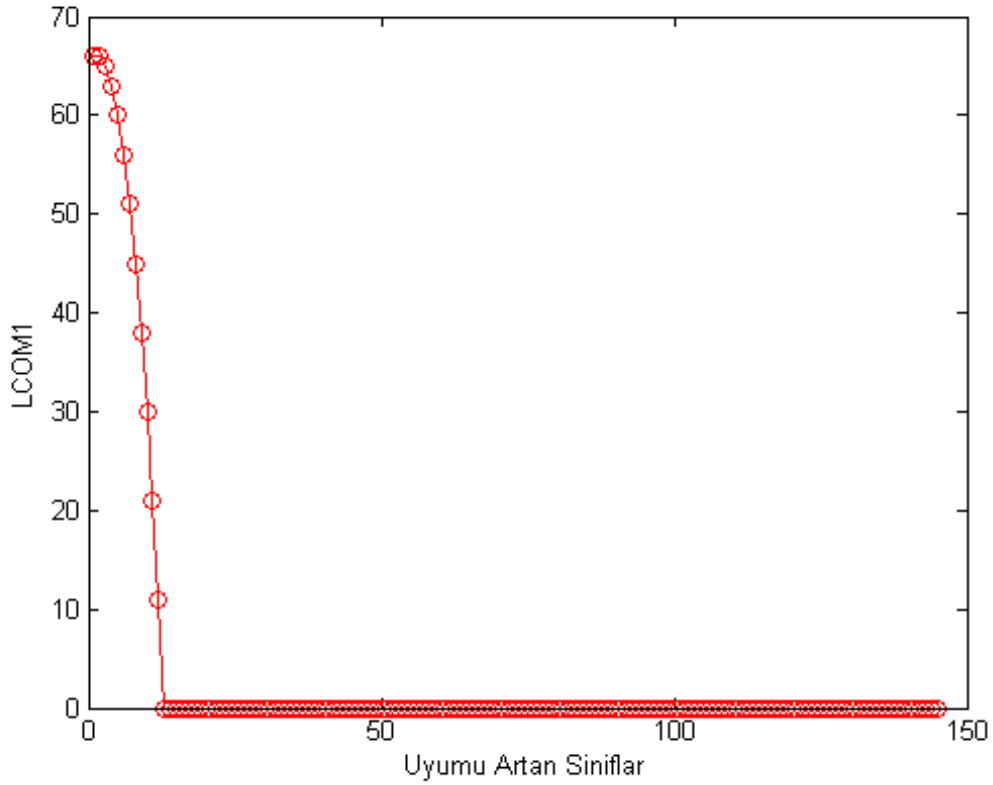
LCOM1 metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.8. 4 Metot 4 öznitelikli sınıfların LCOM1e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.9. 8 Metot 8 öznitelikli sınıfların LCOM1 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.10. 12 Metot 12 öznitelikli sınıfların LCOM1 e göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.8, Şekil 3.9. ve Şekil 3.10. da görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artmasına rağmen belli bir süre sabit kalmıştır. Daha sonra azalmış ve maksimum uyum minimum uyum değeri olan 0 a ulaşınca tekrar sabit kalmıştır. Sezgisel olarak ilk ölçümden son ölçüme kadar sınıfın uyumunun azalması gerekirdi. Bu durumdan yola çıkarak LCOM1 Fenton 'un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

3.3.2. LCOM2

LCOM2, Chidamber ve Kemerer tarafından önerilmiştir (Chidamber ve Kemerer, 1994). LCOM1 metriğinin gelişmiş halidir. LCOM2 de LCOM1 deki gibi amaç uyum eksikliğinin derecesini bulmaktır. LCOM2 de herhangi bir ortak özniteliğe ulaşmayan başka bir ifade ile atama ya da okuma yapmayan metot çiftleri P ve en az bir ortak özniteliği olan metod çiftleri Q bulunur. P ve Q farkı 0 dan büyük ise LCOM2= P-Q dur. Aksi halde LCOM2 =0 dır.

$$LCOM2 = \begin{cases} |P| - |Q|, \text{if } |P| > |Q| \\ 0, \text{if } |P| \leq |Q| \end{cases}$$

LCOM2 nin LCOM1 den farkı LCOM1 de maksimum uyuma, bütün metotlar birbiri ile ilişkili ise ulaşılıyorken LCOM2 de metotların yarısı birbiri ile ilişkili ise maksimum uyuma ulaşılır.

LCOM2 değerinin yüksek olması uyumun düşük olduğunu gösterir. LCOM2 değeri düştükçe uyum artar. Maksimum uyuma 0 değerinde ulaşılır.

LCOM2 değeri herhangi bir ortak özniteliğe ulaşmayan metotların toplamından en az bir ortak özniteliğe ulaşan metotların toplamının farkı olduğu için maksimum LCOM2değeri;en az bir ortak özniteliğe ulaşan metotların toplamı 0 olması durumunda metotların arasında oluşturabilecek maksimum metot çifti sayısıdır. Buda metot sayısına m dersek maksimum LCOM2 aşağıdaki formül ile verilebilir:

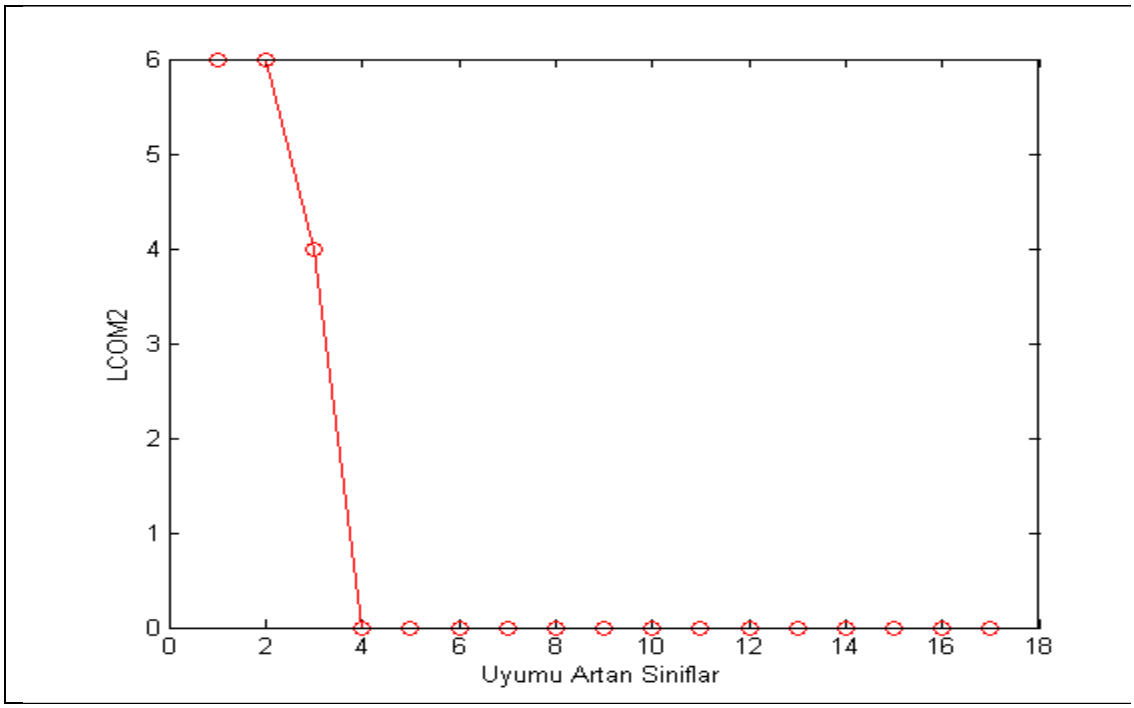
MAX(LCOM2)=m*(m-1)/2 dir.
Örnek verecek olursak 3 metotlu bir sınıfın LCOM2 değeri 3*2/2=3 ile 0 arasında, 5 metotlu bir sınıfın LCOM2 değeri 5*4/2=10 ile 0 arasında, 10 metotlu bir metotlu bir sınıfın LCOM2 değeri 10*9/2=45 ile 0 arasında ölçülür.

Örneklerden anlaşılacağı gibi sadece LCOM2 değerinin bilinmesi ile sınıfın uyumunun derecesi anlaşılamamaktadır. 10 metotlu bir sınıfta LCOM2=3 nerede ise maksimum uyumu göstermesine karşılık 3 metotlu bir sınıfta LCOM2=3 uyumsuzluğu göstermektedir. Bu duruma LCOM2 metriğinin normalize edilmemesi sebep olmaktadır.

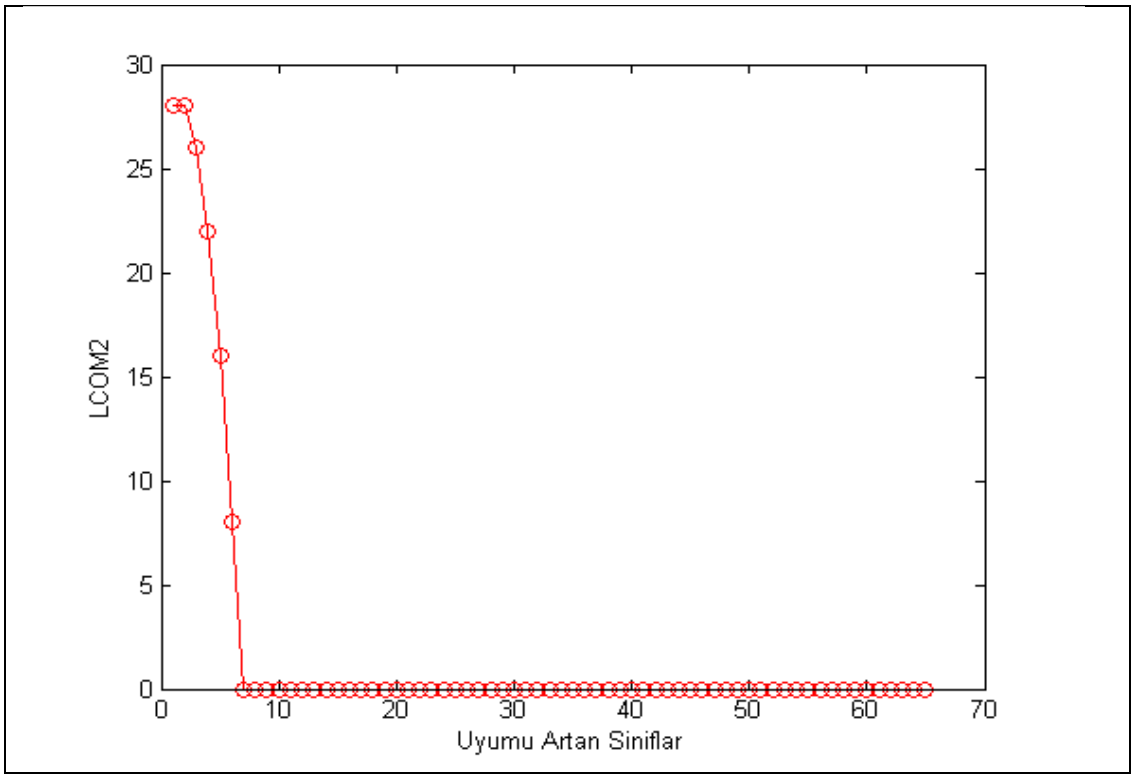
LCOM2 de LCOM1 den farklı olarak eğer ilişkili metot çifti sayısı ilişkisiz metot çiftinden düşük ise LCOM2 değeri 0 olmaktadır. Bu durum sınıf uyum değerinin ölçülmesinde hassasiyeti azaltmaktadır.

Mesela 4 metoda sahip bir sınıfın uyum değeri maksimum 6 uyum ile değerine sahipken LCOM1 in 3,2,1,0 olduğu durumlarda LCOM2=0 değerine sahiptir. Buradan anlaşılacağı gibi LCOM2 metriğinin ölçüm hassasiyeti azalmıştır.

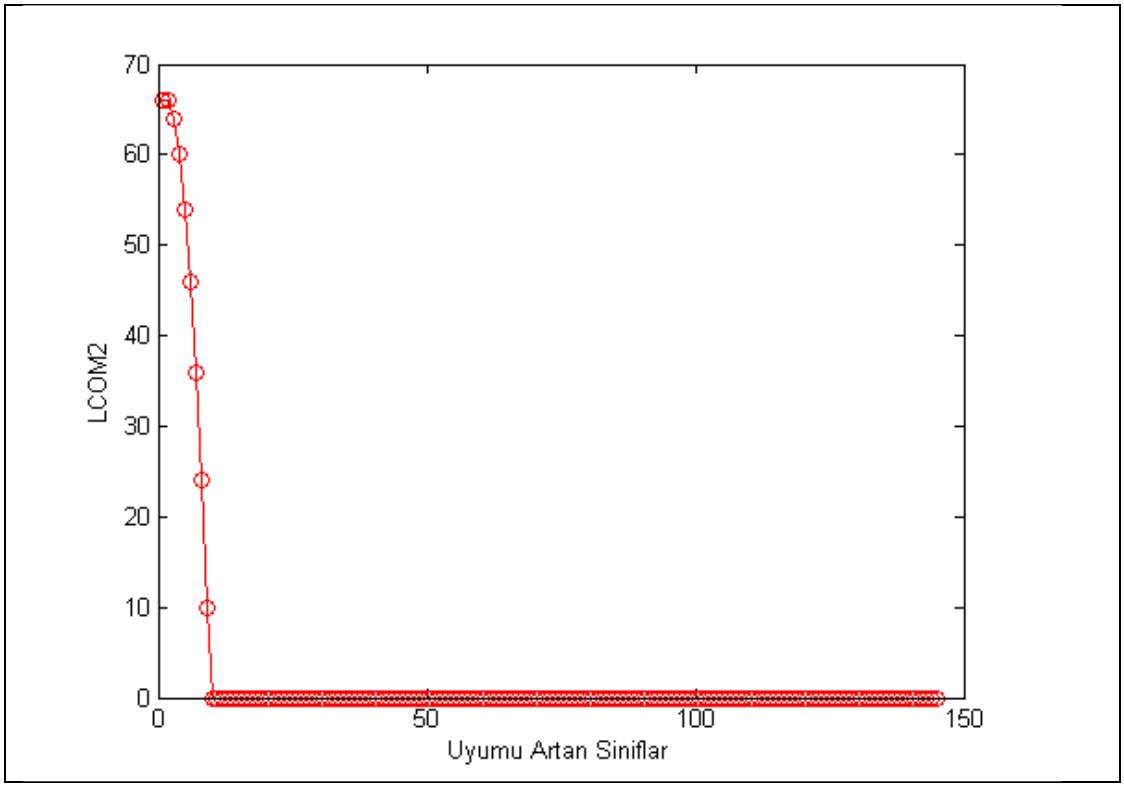
LCOM2 metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.11. 4 Metot 4 öznitelikli sınıfların LCOM2 ye göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.12. 8 Metot 8 öznitelikli sınıfların LCOM2 ye göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.13. 12 Metot 12 öznitelikli sınıfların LCOM2 ye göre sınıf uyumunun artırılarak ölçülmesi.

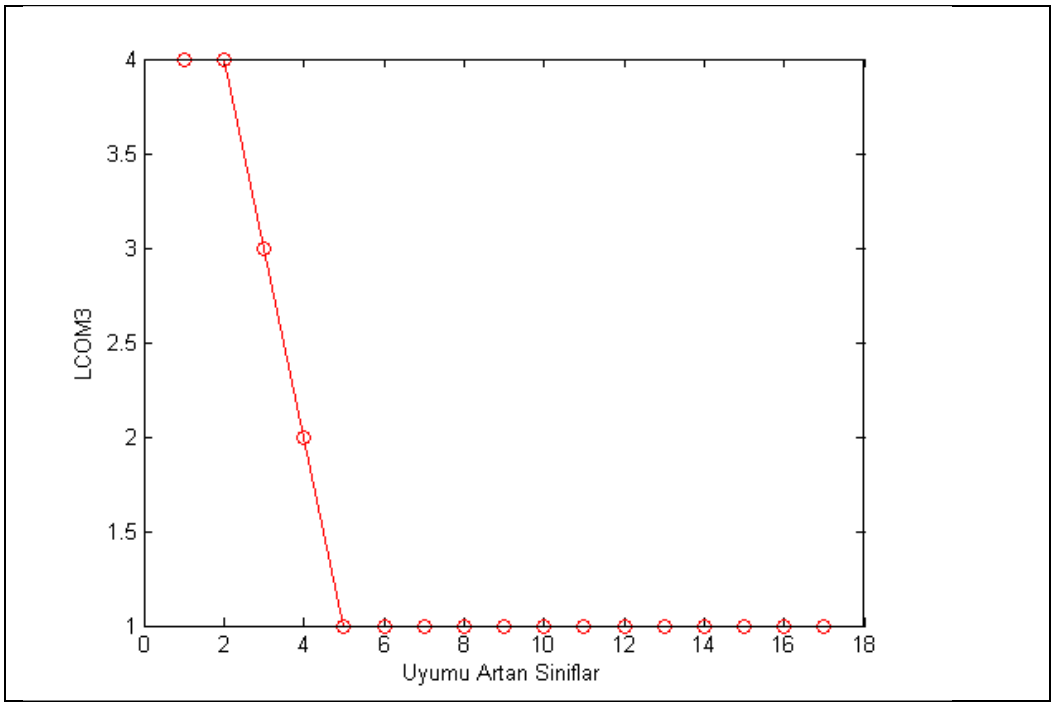
Şekil 3.11.,
Şekil 3.12. ve
Şekil 3.13. te görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artmasına rağmen belli bir süre sabit kalmıştır. Daha sonra azalmış ve maksimum uyum minimum uyum değeri olan 0 a ulaşınca tekrar sabit kalmıştır. LCOM1 den farklı olarak minimum değere daha hızlı ulaşılmıştır. Sezgisel olarak ilk ölçümden son ölçüme kadar sınıfın uyum değerinin azalması gerekirdi. Bu durumdan yola çıkarak LCOM2 Fenton 'un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

3.3.3. LCOM3

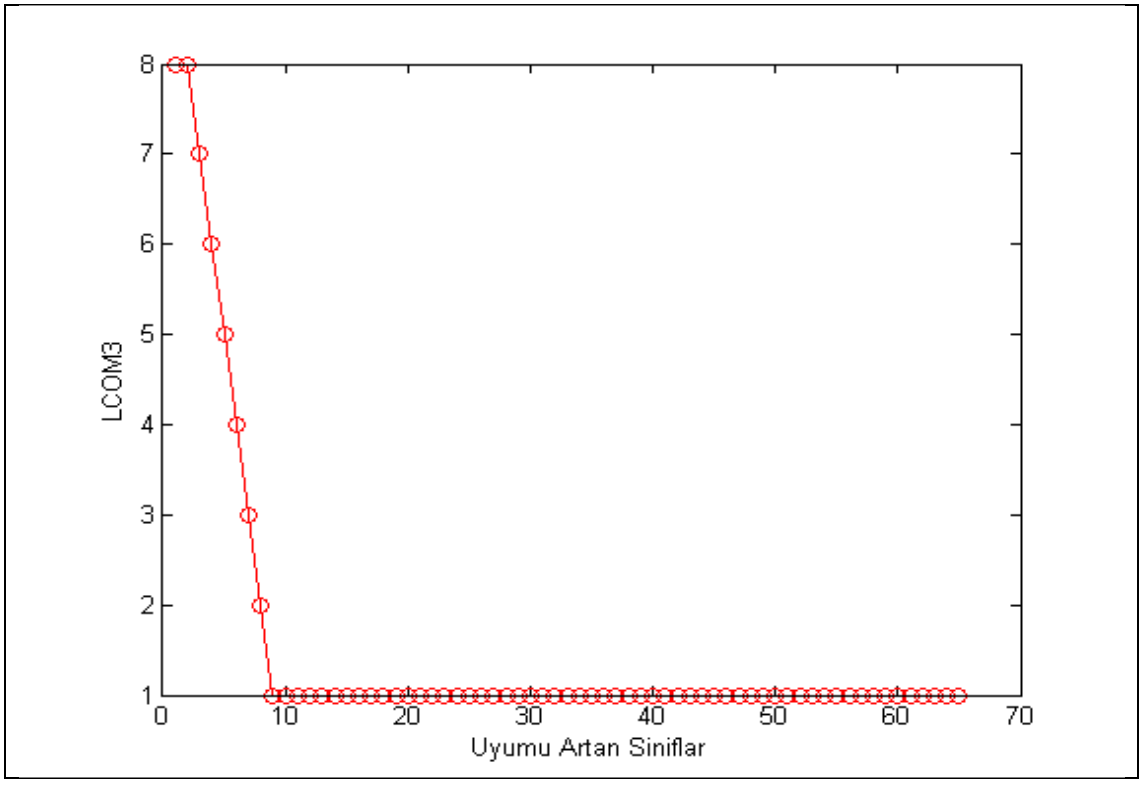
LCOM3 metriği Hitz ve Montazeri tarafından önermiştir (Hitz ve Montazeri, 1995).

LCOM3 te amaç ilişkili metot kümelerinin sayısını ölçmektir. İlişkili metotlar aynı kümede kabul edilerek metotlar kümelenir. Eğer bütün metotlar ilişkili ise LCOM3=1 ölçülür. Bu durum minimum LCOM3 değeri maksimum uyum demektir. Metotların birbiri ile hiçbir şekilde ilişkisi yok ise metot sayısı kadar metot kümesi olacağı için LCOM3=metot sayısı olur. Bu durum maksimum LCOM3 minimum uyum demektir.

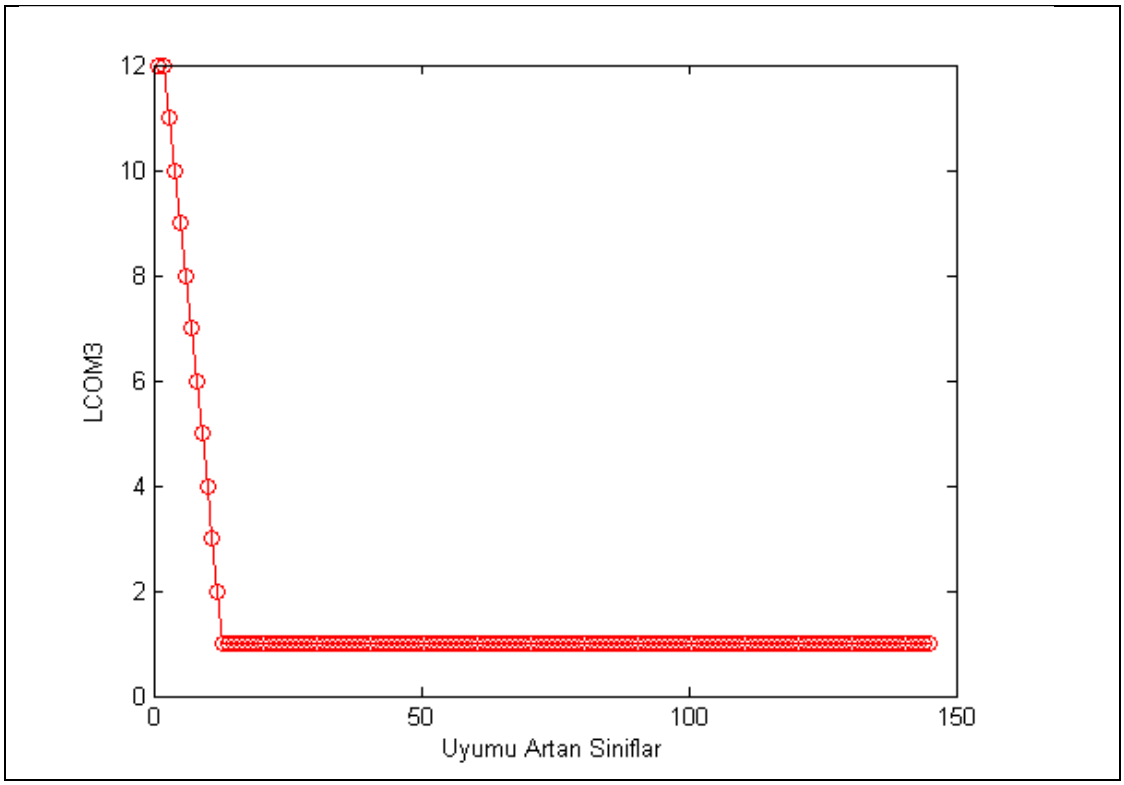
LCOM3 metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.14. 4 Metot 4 öznitelikli sınıfların LCOM3 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.15. 8 Metot 8 öznitelikli sınıfların LCOM3 e göre sınıf uyumunun artırılarak ölçülmesi



Şekil 3.16. 12 Metot 12 öznitelikli sınıfların LCOM3 e göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.14., Şekil 3.15. ve Şekil 3.16. da görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artmasına rağmen belli bir süre sabit kalmıştır. Daha sonra azalmış ve maksimum uyum minimum uyum değeri olan 1 e ulaşınca tekrar sabit kalmıştır. LCOM3 te ilişkili kümeler en az bağlantıyla da oluşsa en fazla bağlantıyla da oluşsa aynı değeri vermektedir. Sezgisel olarak ilk ölçümden son ölçüme kadar sınıf uyumunun değerinin azalması gerekirdi. Bu durumdan yola çıkarak LCOM3 Fenton 'un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

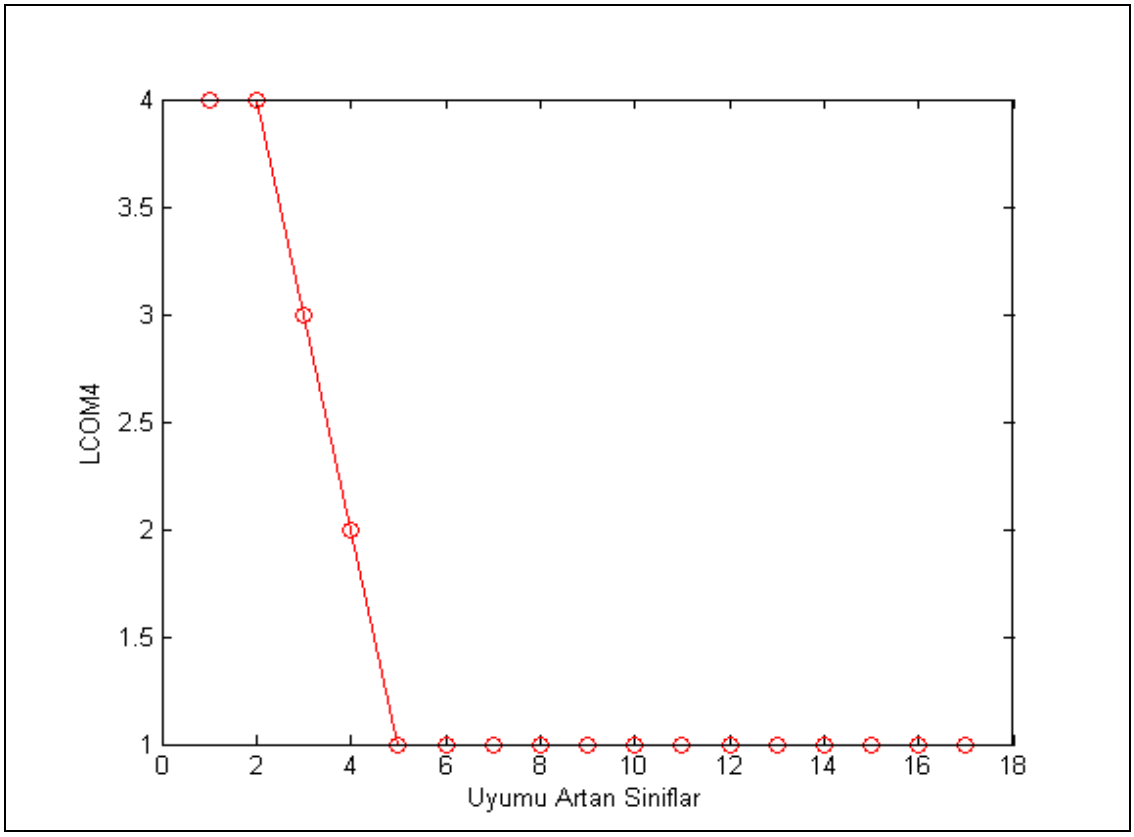
3.3.4. LCOM4

LCOM4 metriği Hitz ve Montazeri tarafından önermiştir (Hitz ve Montazeri, 1995 LCOM3 ün gelişmiş versiyonudur.

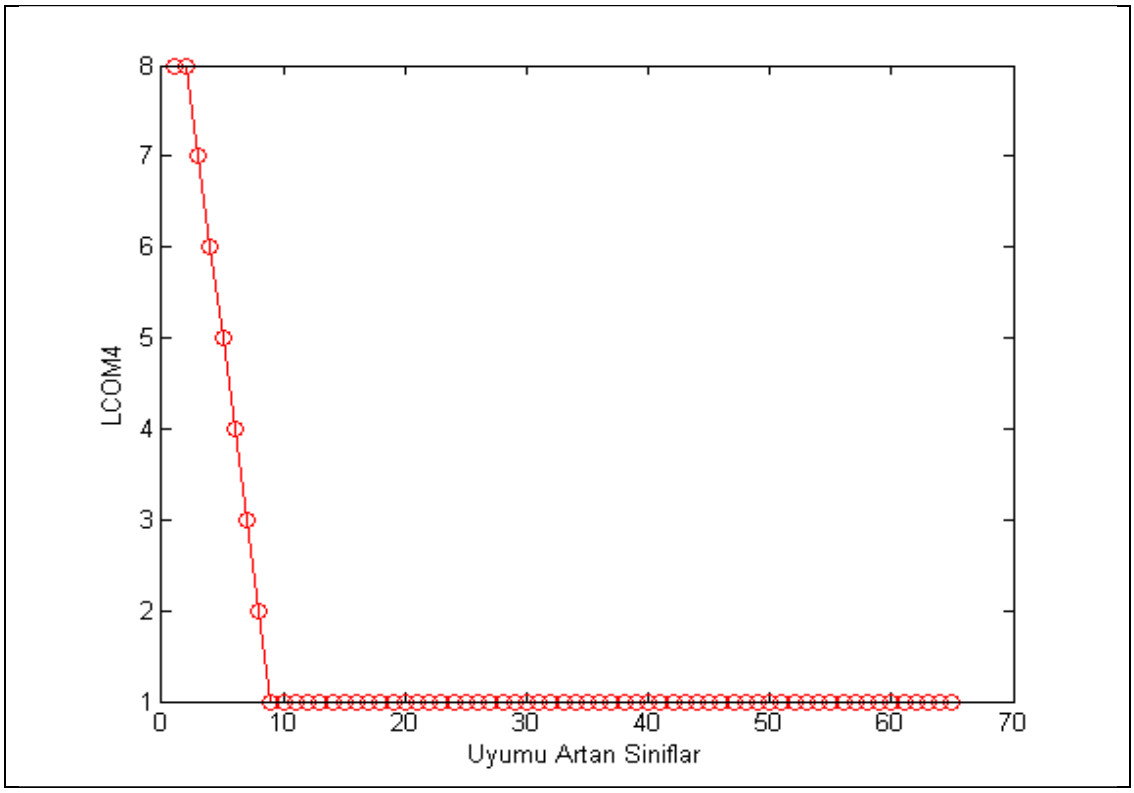
LCOM3 te metotların birbiri ile ilişkiyi sadece en az bir ortak özniteliğe ulaşması ile gerçekleştirebiliyordu. LCOM4 te ise metotların birbirini çağırması ile de ilişki gerçekleşebilmektedir.

LCOM4 te amaç ilişkili metot kümelerinin sayısını ölçmektir. İlişkili metotlar aynı kümede kabul edilerek metotlar kümelenir. Eğer bütün metotlar ilişkili ise LCOM4=1 ölçülür. Bu durum minimum LCOM4 değeri maksimum uyum demektir. Metotların birbiri ile hiçbir şekilde ilişkisi yok ise metot sayısı kadar metot kümesi olacağı için LCOM4=metot sayısı olur. Bu durum maksimum LCOM4 minimum uyum demektir.

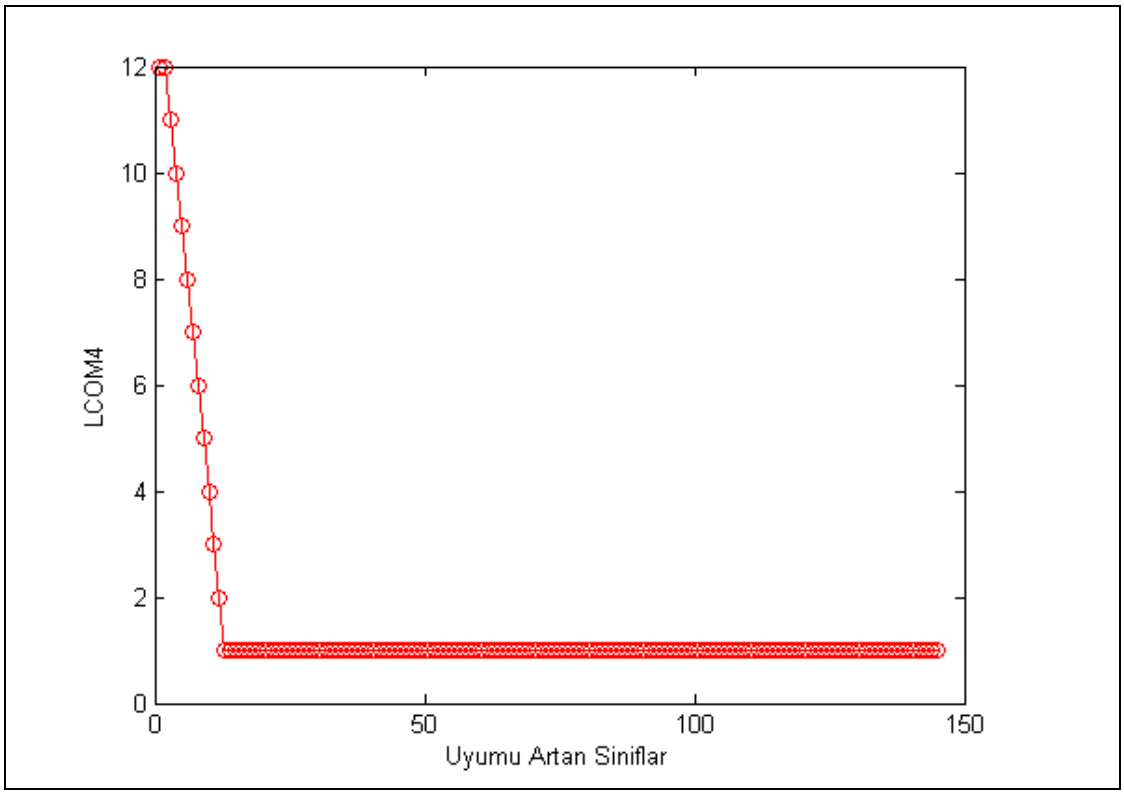
LCOM4 metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.17. 4 Metot 4 öznitelikli sınıfların LCOM4 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.18. 8 Metot 8 öznitelikli sınıfların LCOM4 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.19. 12 Metot 12 öznitelikli sınıfların LCOM4 e göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.17.,
Şekil 3.18. ve Şekil 3.19. da görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artmasına rağmen belli bir süre sabit kalmıştır. Daha sonra azalmış ve maksimum uyum minimum uyum değeri olan 1 e ulaşınca tekrar sabit kalmıştır. LCOM4 te ilişkili kümeler en az bağlantıyla da oluşsa en fazla bağlantıyla da oluşsa aynı değeri vermektedir. Sezgisel olarak ilk ölçümden son ölçüme kadar sınıf uyumunun değerinin azalması gerekirdi. Bu durumdan yola çıkarak LCOM4 Fenton ’un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

3.3.5. Co

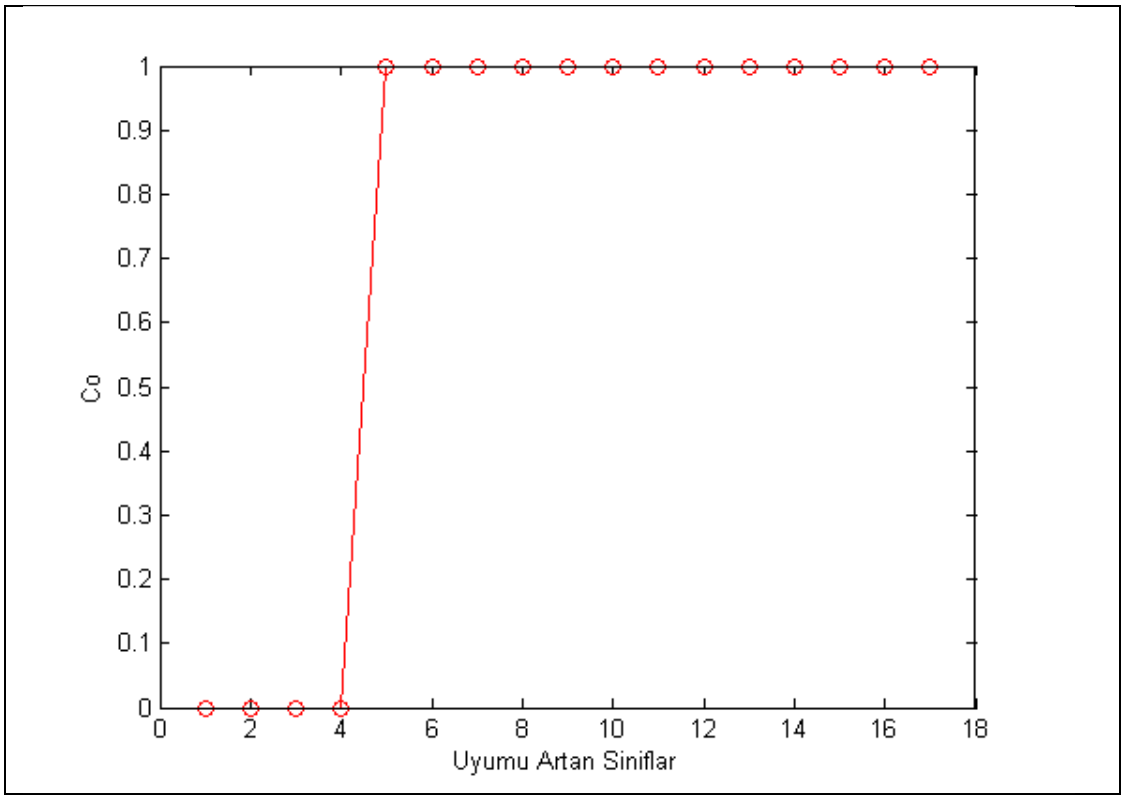
Co metriği Hitz ve Montazeri tarafından önermiştir.

Co metriği bütün metodların birbiri ile ilişkili olması durumundaki sınıfların uyumunu ölçmek için kullanılır. Bunun için LCOM4 değeri 1 olması gerekmektedir.

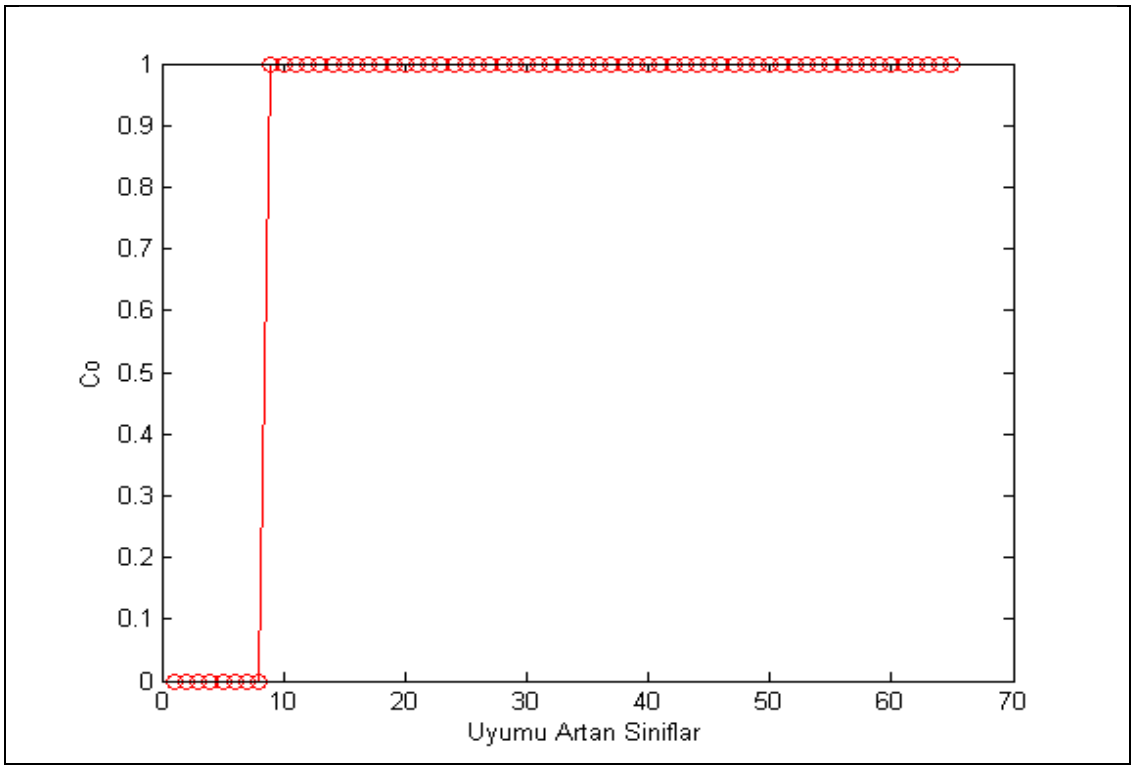
E ilişkili metot sayısı V metod sayısı olmak üzere

$$\text{Co} = 2.\frac{|E| - (|V| - 1)}{(|V| - 1).(|V| - 2)}$$

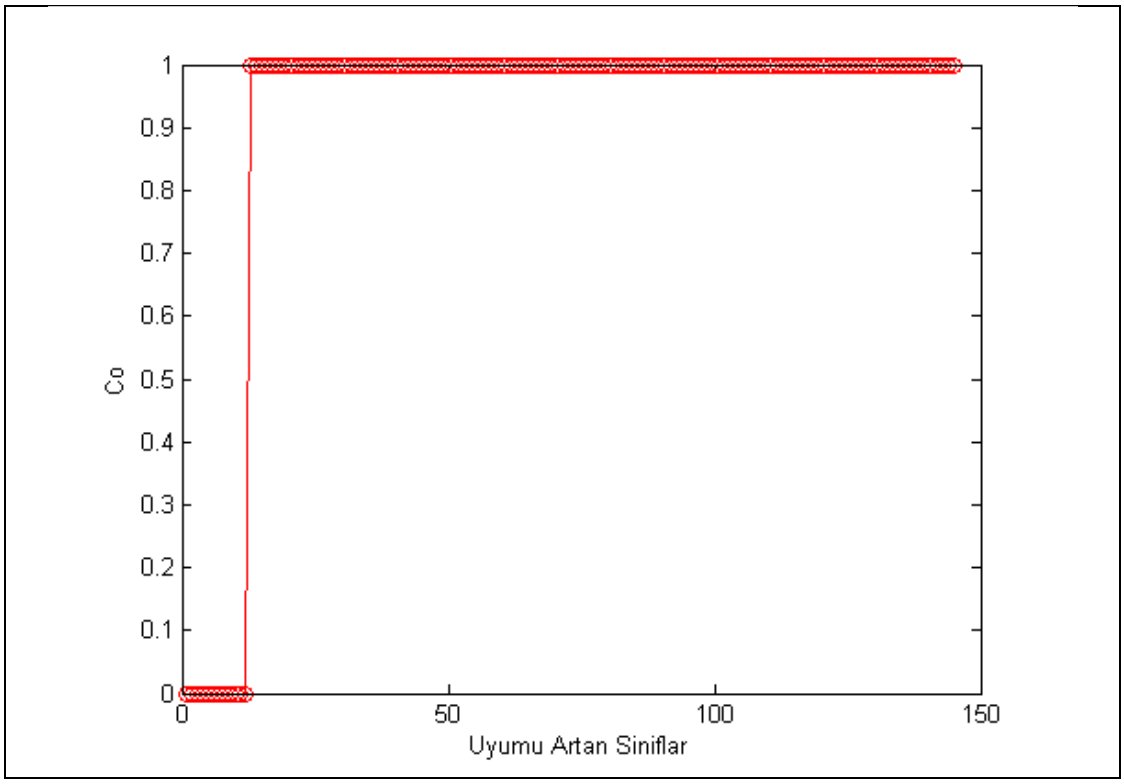
Co metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.20. 4 Metot 4 öznitelikli sınıfların Co ya göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.21. 8 Metot 8 öznitelikli sınıfların Co ya göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.22. 12 Metot 12 öznitelikli sınıfların Co ya göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.21., Şekil 3.22. ve Şekil 3.23.' te görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artmasına rağmen belli bir süre sabit kalmıştır. Daha sonra artmış ve maksimum uyum değeri olan 1 e ulaşınca tekrar sabit kalmıştır. Sezgisel olarak ilk ölçümden son ölçüme kadar sınıf uyumunun değerinin surekli artması gerekirdi. Bu durumdan yola çıkarak Co Fenton 'un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

3.3.6. LCOM5

LCOM5 metriği Henderson ve Sellors tarafından önerilmiştir(1996).

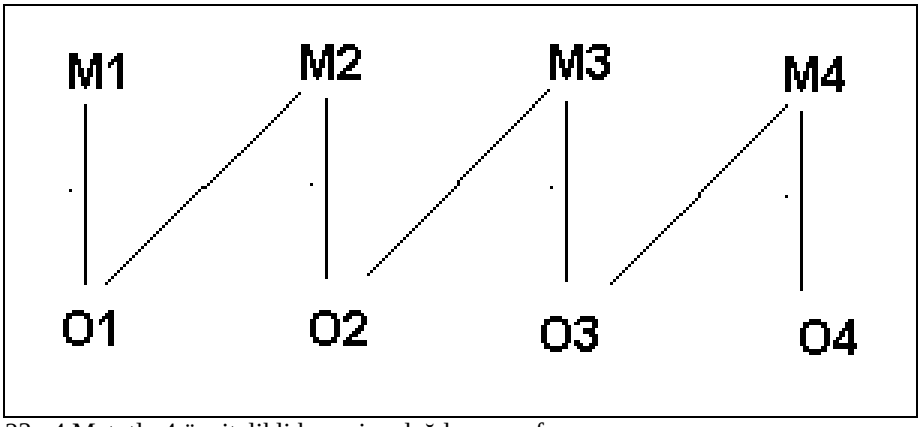
Özniteliklere ulaşan metot sayıları göz önüne alınarak hesaplanır. Her öznitelik için kendisine ulaşan öznitelik sayısı hesaplanır ve toplanır, öznitelik sayısına bölünür. Sonuçtan metot sayısı çıkarılarak 1- metod sayısına bölünür

$$LCOM5 = \frac{\frac{1}{a}\sum_{j-1}^{a}\mu(Aj) - m}{1-m}$$

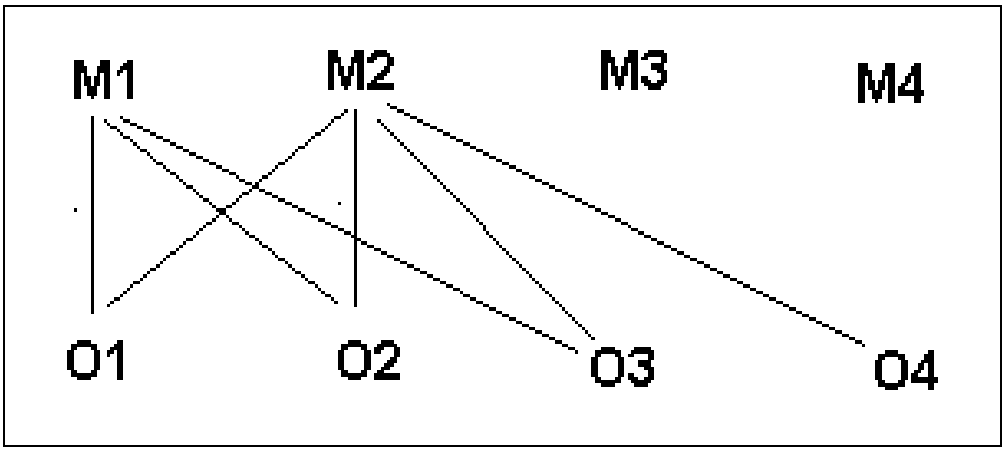
Her bir metodun özniteliğe ulaşması uyumu arttırdığı tezi ile uyum için bir fikir verse de aşağıdaki Şekil 3.24.,
Şekil 3.25. durumunda da aynı LCOM5 uyum değerini vermektedir.

LCOM5=((2+2+2+1)/4-4)/(1-4)=0,75 değerini vermektedir.

Şekilde görüldüğü gibi iki sınıfın LCOM5 değeri aynı çıksa da sezgisel olarak uyum değerinin aynı olmadığı görülmektedir. İkinci sınıfta iki tane metot kesinlikle sınıfla ilişkili olmasa da LCOM5 değeri aynıdır. Bu durumdan yola çıkarak LCOM5 Fenton ’un varlık ile sayıların eşleştirilmesi ve verilen sayılar ile deneysel ilişki arasındaki bağın korunması tezini gerçekleyemediği (Fenton, N) sonucuna varılmıştır.

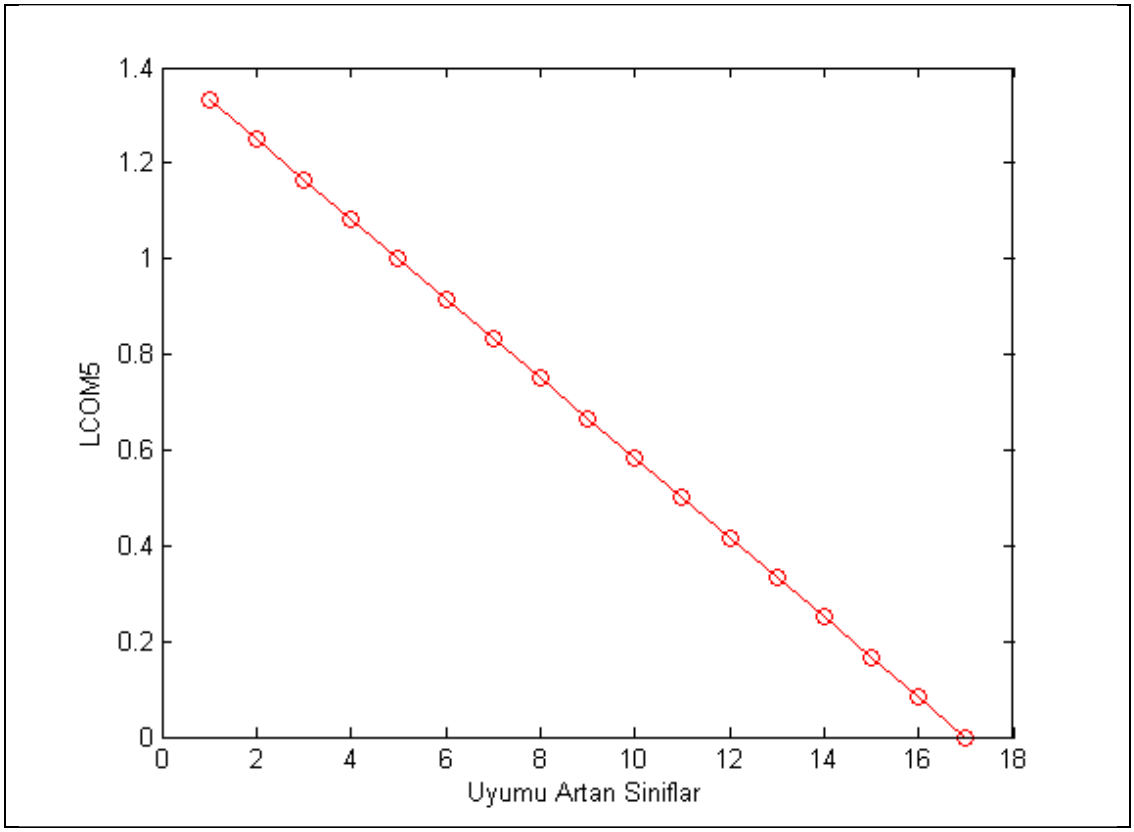


Şekil 3.23. 4 Metotlu 4 öznitelikli homojen dağılmış sınıf

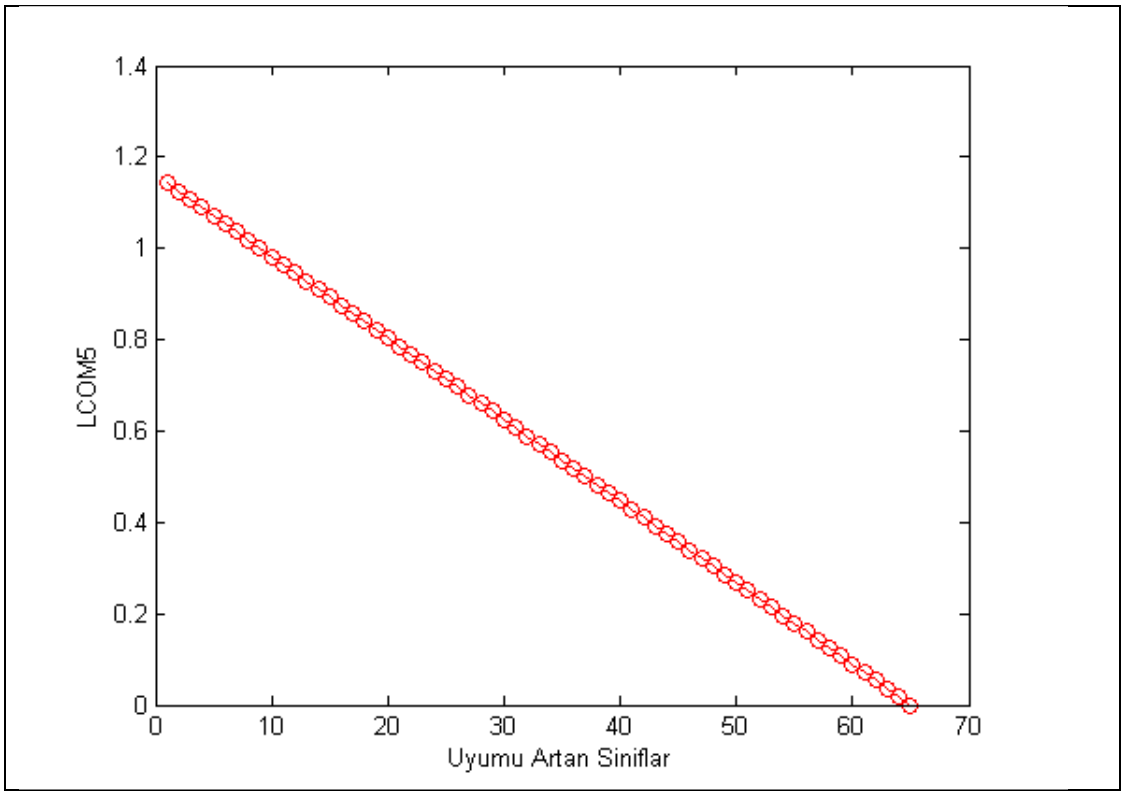


Şekil 3.24. 4 Metotlu 4 öznitelikli heterojen dağılmış sınıf

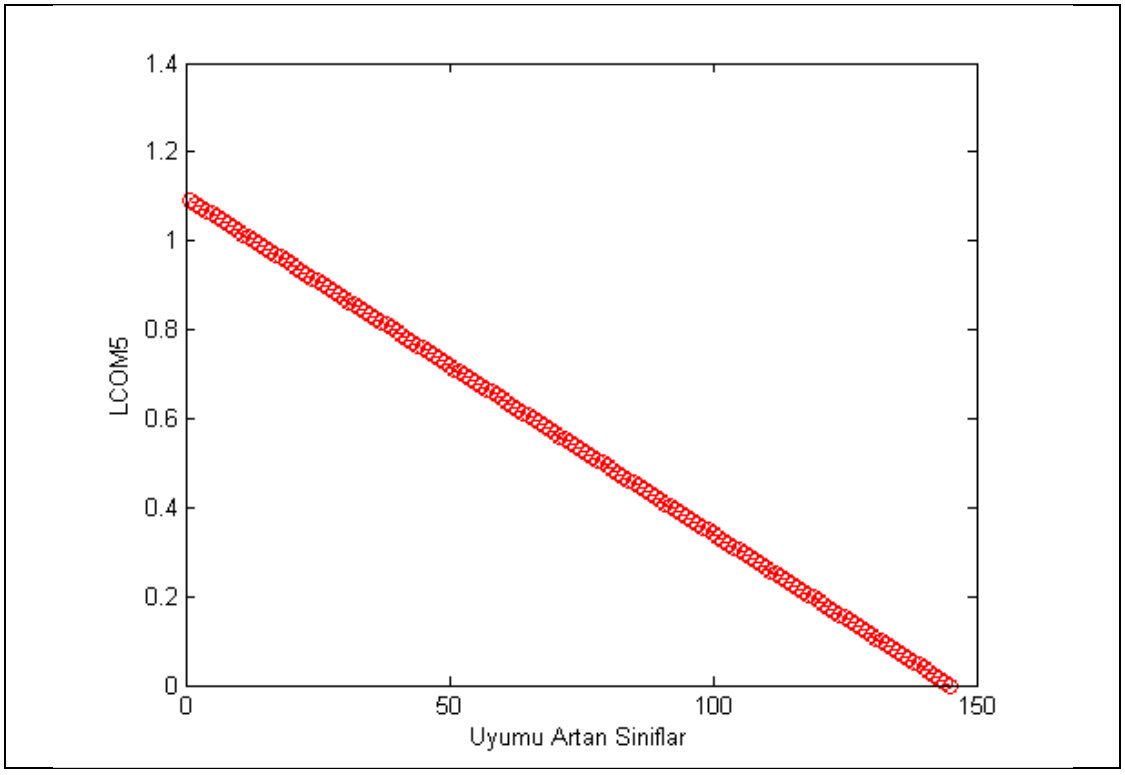
LCOM5 metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.25. 4 Metot 4 öznitelikli sınıfların LCOM5 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.26. 8 Metot 8 öznitelikli sınıfların LCOM5 e göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.27. 12 Metot 12 öznitelikli sınıfların LCOM5 e göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.25., Şekil 3.26. ve Şekil 3.27. de görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artması ile düzenli olarak azalmıştır. LCOM5 metriği artan sınıf uyum grafiğine göre sezgisel beklentileri karşılamaktadır.

3.3.7. Coh

Coh ölçütü Briand tarafından önerilmiştir(Briand vd., 1998).

Özniteliklere ulaşan metot sayıları göz önüne alınarak hesaplanır. Her öznitelik için kendisine ulaşan öznitelik sayısı hesaplanır ve toplanır. Öznitelik sayısına bölünür.

Coh ölçütü, yüksek değerler için yüksek uyumu düşük değerler için düşük uyumu gösterir. Coh formül olarak şu şekilde verilmektedir:

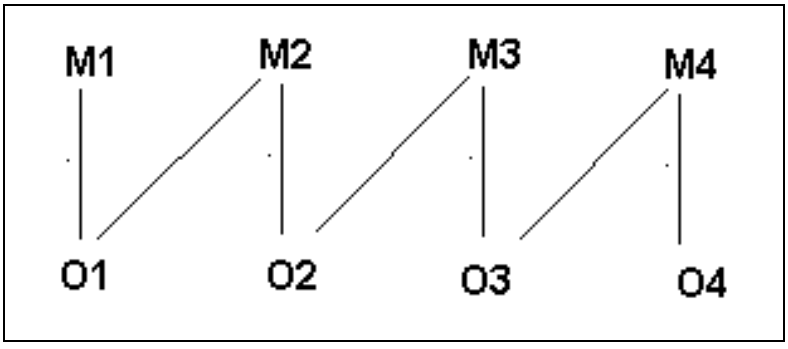
$$Coh = \frac{\sum_{j=1}^{a} \mu(Aj)}{m.a}$$

Coh= Özniteliklere ulaşan metotların toplamı/metot sayısı

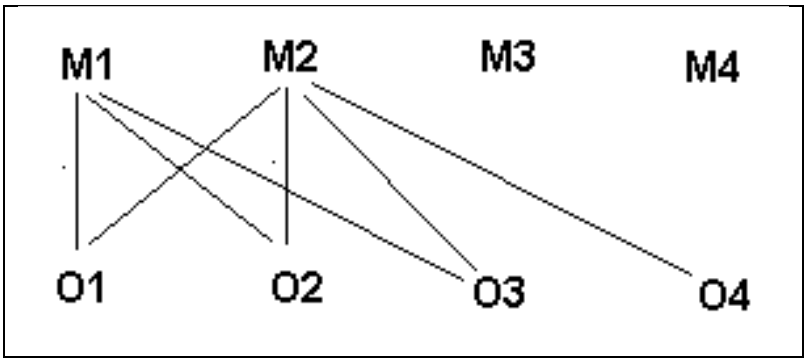
Coh uyum için bir fikir verse de aşağıdaki iki durumda da

Coh=(2+2+2+1)/(4*4)=0,4375

Şekil 3.28. ve Şekil 3.29. görüldüğü gibi iki sınıfın LCOM5 değeri aynı çıksa da uyumunun aynı olmadığı görülmektedir. İkinci sınıfta iki tane metot kesinlikle sınıfla ilişkili olmasa da bu durum fark etmektedir.

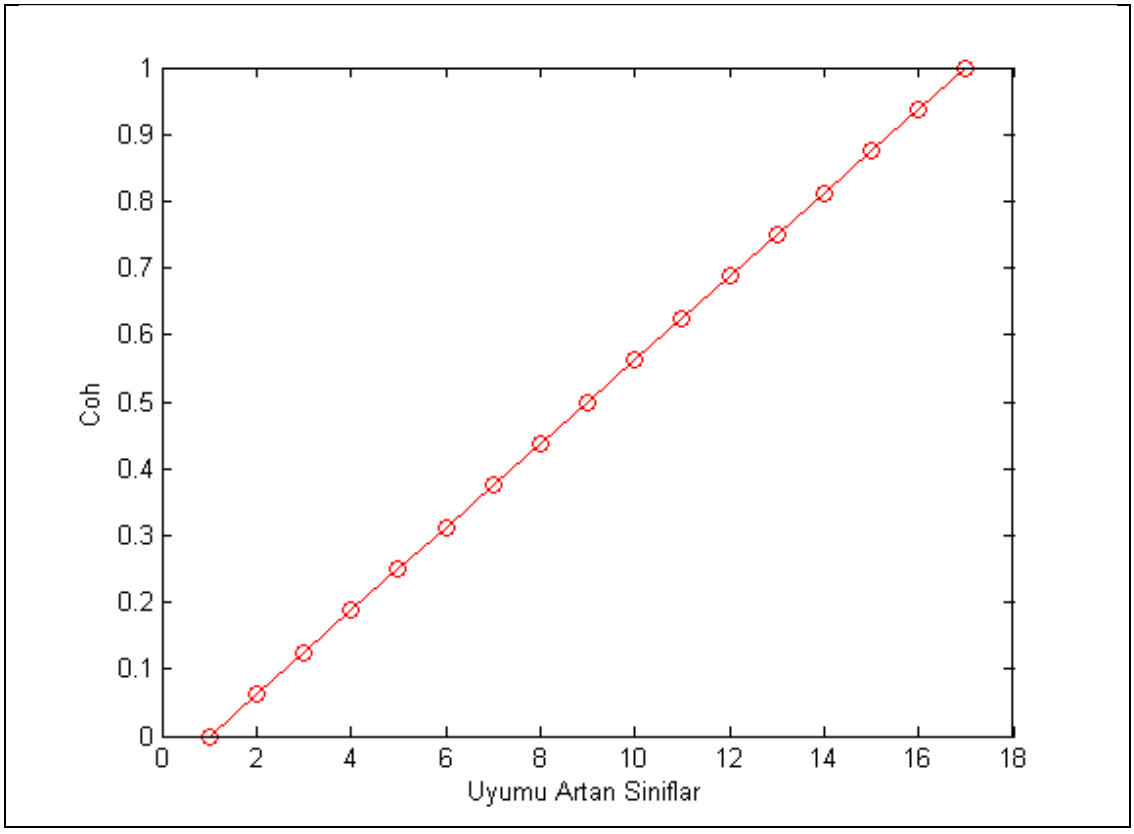


Şekil 3.28. 4 Metotlu 4 öznitelikli homojen dağılmış sınıf

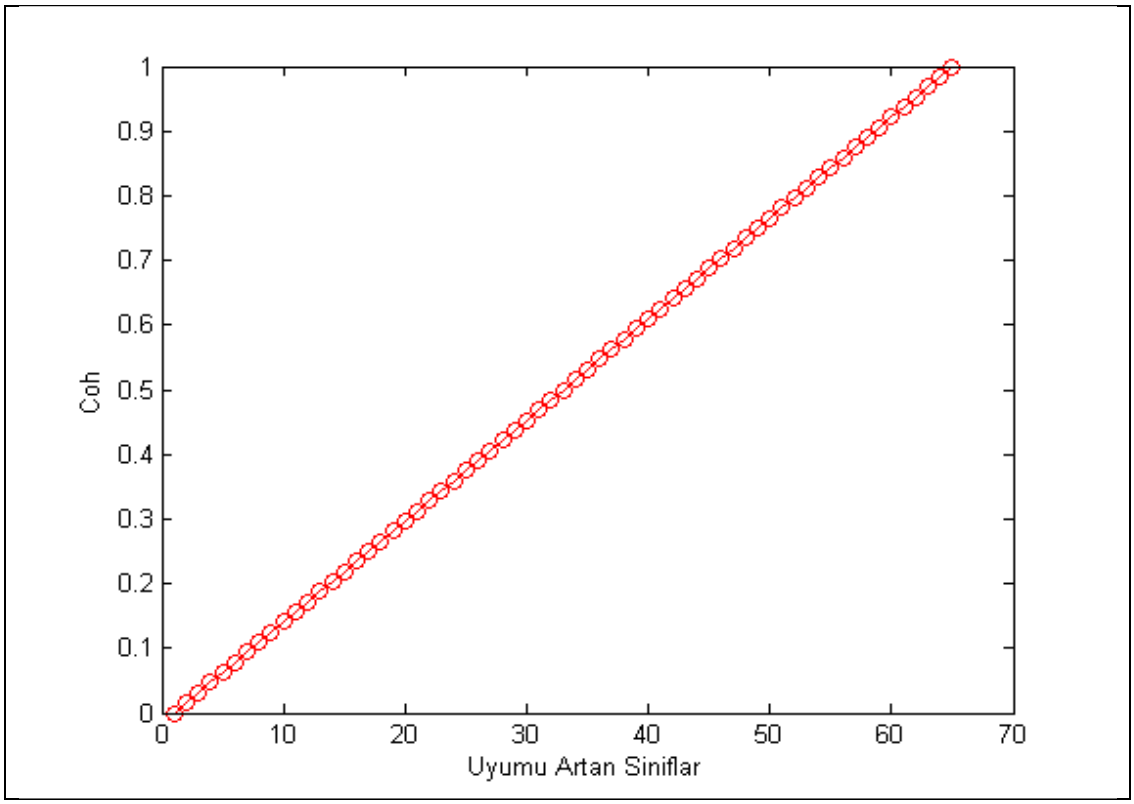


Şekil 3.29. 4 Metotlu 4 öznitelikli heterojen dağılmış sınıf

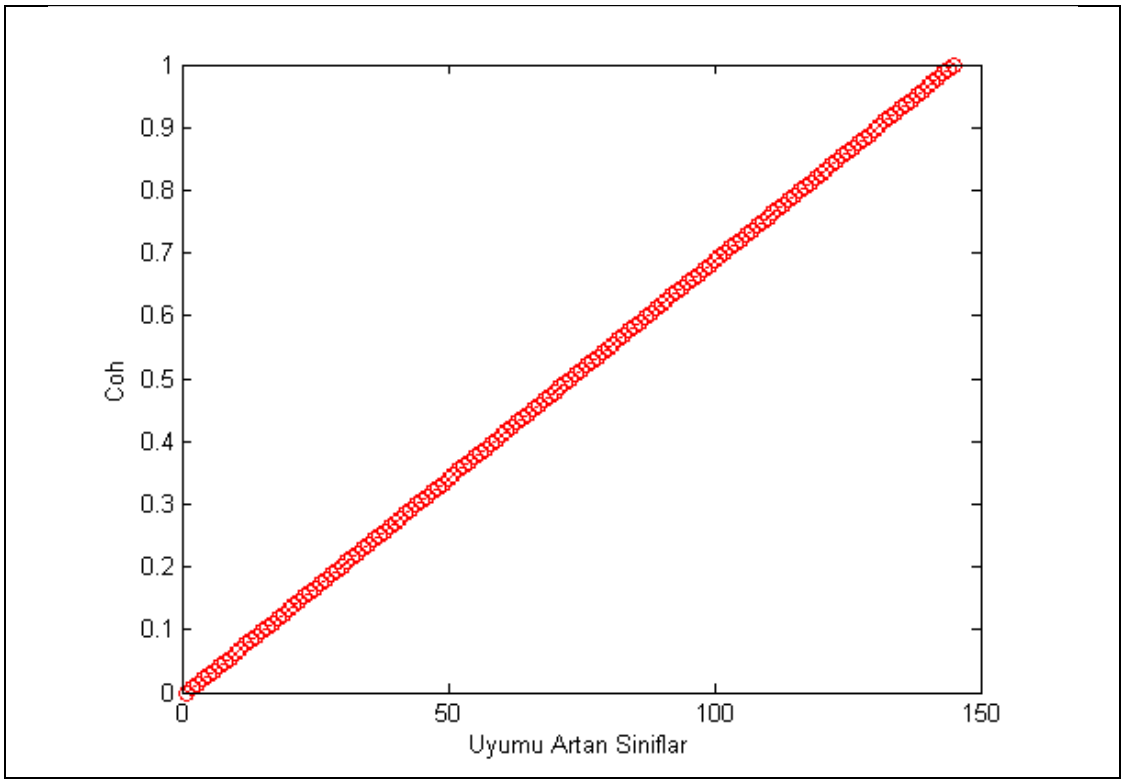
Coh metriğinin sınıf uyumu arttıkça ölçtüğü değeri incelemek için 4 metot 4 öznitelikli, 8 metot 8 öznitelikli ve 12 metot 12 öznitelikli üç sınıfın artan sınıf uyum grafiği gösterilmiştir.



Şekil 3.30. 4 Metot 4 öznitelikli sınıfların Coh a göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.31. 8 Metot 8 öznitelikli sınıfların Coha göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 3.32. 12 Metot 12 öznitelikli sınıfların Coh a göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 3.30.,
Şekil 3.31. ve Şekil 3.32.'de görüldüğü gibi sınıfların uyum değerleri sınıfların metot ve öznitelikleri arasında ilişki artması ile düzenli olarak artmıştır. Coh metriği artan sınıf uyum grafiğine göre sezgisel beklentileri karşılamaktadır.

3.4. Metriklerin Genel Değerlendirmesi

Sonuçlar özetlendiğinde aşağıdaki eksiklikler çıkmaktadır:

LCOM1, LCOM2, LCOM3, LCOM4 metrikleri normalize edilmemiştir. Bu durumda sadece ölçüm sonucu uyumun derecesini anlamada yeterli gelmemektedir.
Coh ve LCOM5 metrikleri sadece metot öznitelik bağıntı sayısını ölçmektedir. Metotlar arası ilişkiyi ölçmemektedir.

Uyum ölçütlerinin sadece uyumun bir yönünü ölçmektedir. Araştırmacılar uyumu kendi referans noktarına göre değerlendirmiş ve sadece o referansa göre ölçmüşlerdir. Bu durumda ölçülen durum gerçekleşinceye kadar sınıf içinde etkileşim artsa da ölçülen metrik değeri artmamaktadır. Tanımlanan maksimum uyum sağlandığı bazı durumlarda ise sınıf içi etkileşim artabilmektedir.

Sadece metot öznitelik ilişkileri ölçülen metriklerde metotların ilişkilerinin göz önüne alınmamıştır. Sınıf uyumunda önemli bir ölçü olan metodların uyumu göz ardı edildiğinde uyum eksik ölçülmektedir.

Bir sonraki bölümde gözlenen bu eksiklikleri gidermek için yeni bir metrik tanımlanacaktır.

BÖLÜM 4. YENİ SINIF UYUM METRİĞİ: TÜM SINIF UYUMU

İncelenen metriklerin her biri sınıf uyumunun bir yönünü ölçmektedir fakat tek başlarına eksik kalmaktadırlar. Örneğin LCOM1 ve LCOM2 sadece sınıftaki metotların uyumunu ölçmektedir. LCOM3 ve LCOM4 ilişkili metotları gruplayarak sınıfta kaç farklı metot grubu olduğunu ölçer. LCOM5 ve Coh ise özniteliklere kaç metot ulaştığını göz önüne alarak uyumu hesaplar.

Önceki bölümde anlatılan, uyum ölçütlerinin sadece uyumun bir yönünü ölçmesi sorununa, çözüm üretilmek için üç uyum metriğini bileştirerek yeni bir metrik tanımlanacaktır. Tanımlanan bu metriğe Tüm Sınıf Uyumu (TSU) metriği denecektir.

Tasarlanan metrik LCOM1 ve LCOM4 in versiyonları ile Coh metriğinin birleşiminden oluşacaktır. Üç metriği birleştirerek metotlar arası uyum, metotların kaç metot grubu oluşturduğu ve özniteliklerin metotlar tarafından kullanımı dolayısı ile sınıf üyeleri tarafından kullanım yoğunluğu ölçülmüş olacaktır.

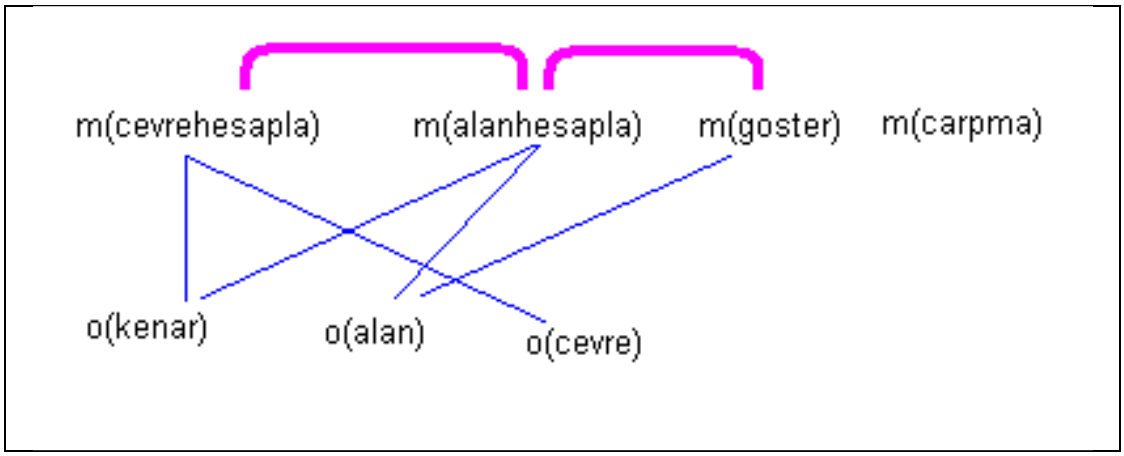
Metrikleri birleştirmesi sırasında normalizasyon yapılmadığı durumlar karşılaşılan problemleri ortadan kaldırmak için normalizasyon yapılacaktır.

Sınıf uyumu üç alt ölçüm ile ölçülerek asıl uyum bu ölçümlerin ortalamaları göz önüne alınarak ölçülecektir. Bu alt ölçümler:

Sınıf Metot İlişkisi(SMI): Sınıftaki metotların uyumunu ölçen LCOM1 versiyonu ölçümdür. LCOM1 den farkı ters uyum ölçümü olmaması ve normalize edilmesidir.

m metod sayısı olursa m*(m-1)/2 olabilecek en fazla metot-metot ilişki sayısıdır. i ye en az bir ortak özniteliğe ulaşan metot sayısı dersek

SMI=i/((m*(m-1))/2)=2*i/(m*(m-1)) dir.



Şekil 4.1. 4 Metotlu 3öznitelikli sınıf

Şekil 4.1. deki sınıf için SMI=2*2/(4*3)=0,33

Metot Ortak Çalışma Ölçümü(MOCO): Sınıftaki metotların ortak öznitelikler üzerinde çalışıp çalışmadığını ölçen LCOM4 versiyonu ölçümdür.

Ortak özniteliğe ulaşan metotları ilişkili kabul etme durumunda birbiriyle ilişkili metotlar kümelenir. Mümkün olan en fazla küme sayısı metot sayısı m e eşittir. Kümelenen metotların sayısına k dersek;

MOCO=(m-k)/(m-1) dir.

Şekil 4.1. örnek sınıfı için MOCO=(4+1-2)/4=0,75 tir.

Üçüncü olarak özniteliklerin yoğunluğu ölçülecektir. Bu ölçüm için Coh yeterlidir.

Coh: Özniteliklerin sınıf Sınıftaki özniteliklerin kullanım yoğunluğunu ölçen LCOM5 versiyonu ölçümdür. 1998 de Briand tarafından önerilmiştir.

Metotların özniteliklere ulaşma sayısının olası en fazla ulaşma sayına oranıdır. Özniteliklere ulaşma sayısı u, metot sayısı m öznitelik sayısı o ve maksimum ulaşma sayısı o*m dir.

Coh=u/(o*m) dir.

Şekil 4.1. örnek sınıfı için Coh=5/12=0,42

Üç ölçüm hesaplandıktan sonra ölçümlerin ortalaması alınır. Aşağıdaki gibi gösterilebilir.

(MOCO+SMI+Coh)/3

Sonuç 0-1 arasında bir değer çıkacaktır. Üç farklı ölçümün birleşmesinden oluşan bu metriği değerlendirirken metriğin düşük olması durumunda metriğin düşüklüğünün üç ölçümden mi kaynaklandığı yoksa bir ölçümden mi kaynaklandığını bulmak için ölçümleri tekrar yapmak gerekmektedir.

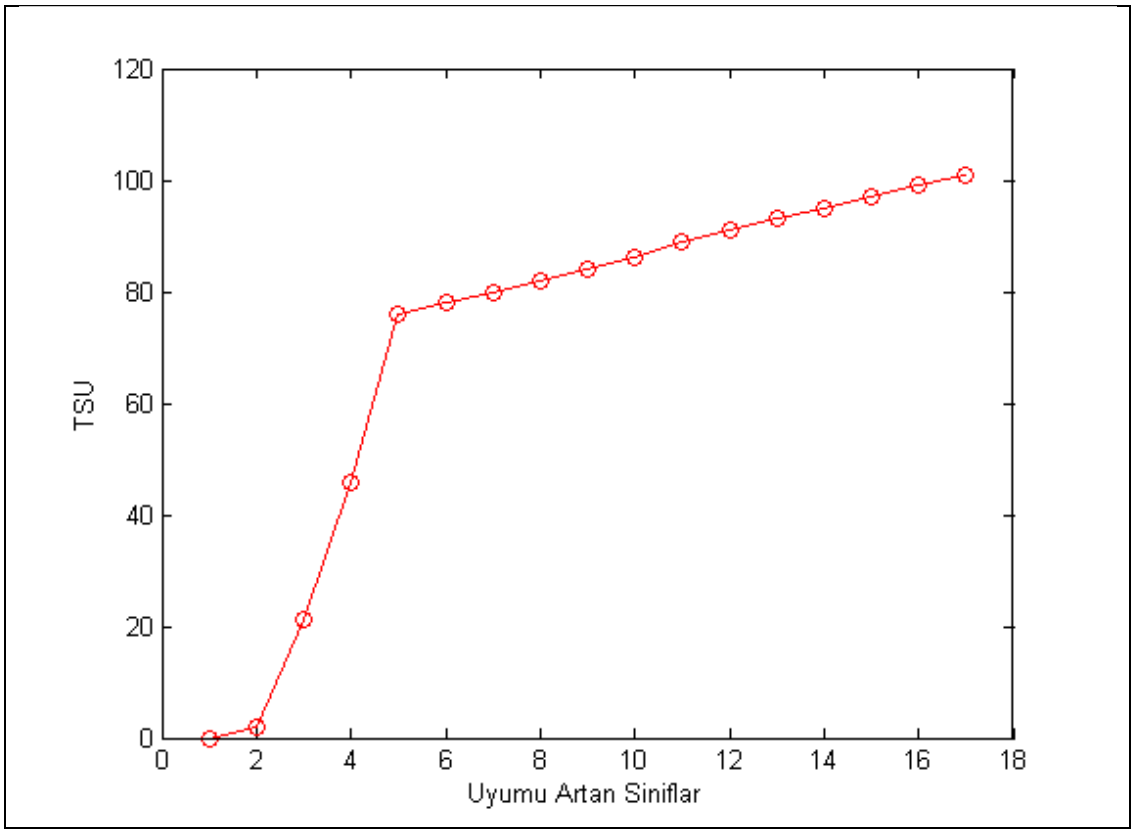
Hızlı bir şekilde üç ölçümü öğrenebilmek için TSU değeri içine bu üç ölçüm gizlenecektir. TSU değerinin değişiklikten etkilenmemesi üç sınıfın ortalaması 100 ile çarpılır. üç ölçüm sıra ile virgülden sonra yazılır. Bu durumda virgülden sonraki küsurat asıl ölçütümüzü etkilemiyecektir.

Bu durumda formül
TSU=100*(MOCO+SMI+Coh)/3,[10*MOCO][10*SMI][10*Coh]
Şeklinde yazılabilir.

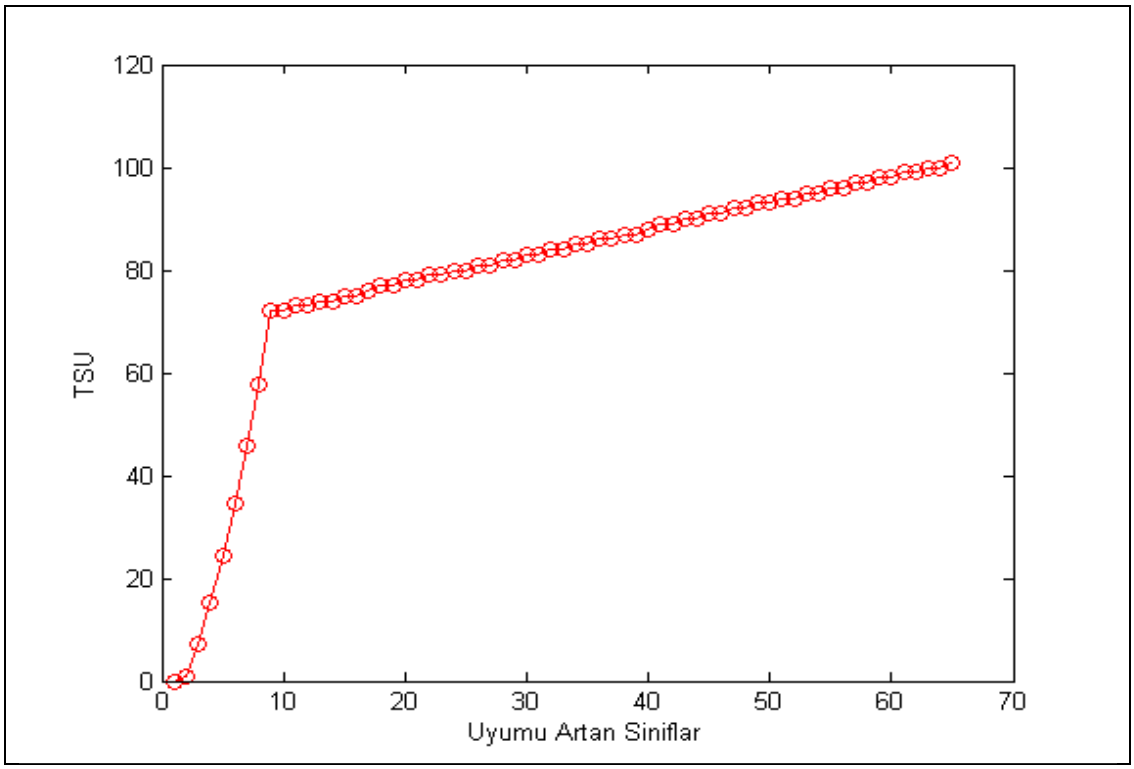
Şekil 4.1. ornek sınıfı için TSU hesaplandığında

TSU=100*(0.75+0.33+0.42)/3,[0.75*10][0.3*10][0.42*10]
TSU=50,834 sonucu bulunur.

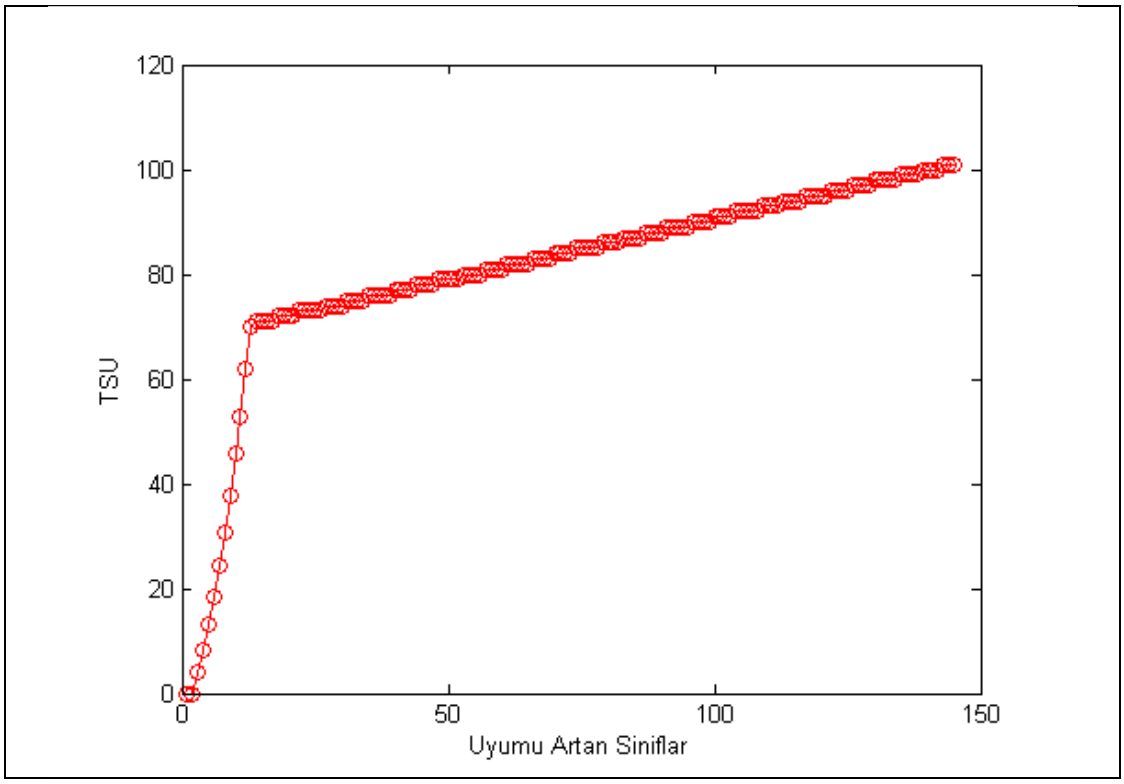
TSÜ için artan sınıf uyum grafiği aşağıda çizdirilmiştir.



Şekil 4.2. 4 Metot 4 öznitelikli sınıfların TSU ya göre sınıf uyumunun artırılarak ölçülmesi.



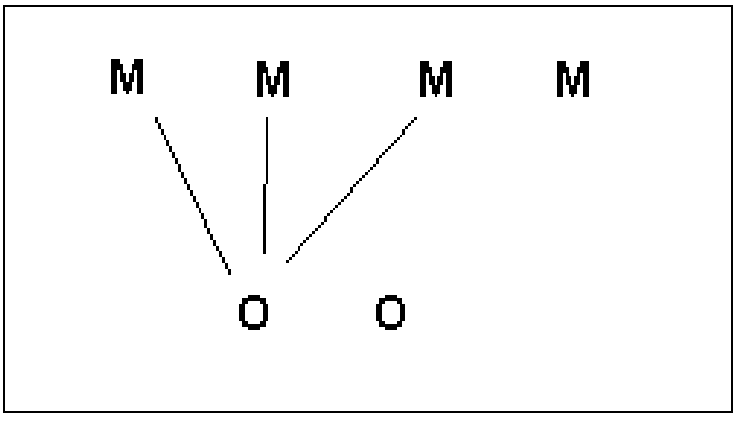
Şekil 4.3. 8 Metot 8 öznitelikli sınıfların TSU ya göre sınıf uyumunun artırılarak ölçülmesi.



Şekil 4.4. 12 Metot 12 öznitelikli sınıfların TSU ya göre sınıf uyumunun artırılarak ölçülmesi.

Şekil 4.2., Şekil 4.3. ve Şekil 4.4. görüldüğü gibi sınıf içi her etkileşim uyumun değerini arttırmıştır. Normalizasyon vardır. Örnekler 4,8 ve 10 metodlu olmalarına rağmen uyum değerleri sezgisel değerlerle uyumludur.

Diğer metrikler ile ölçülen sınıflar TSU ile de ölçüleceklerdir.



Şekil 4.5. 4 Metotlu 2 öznitelikli az 3 etkileşimli sınıf

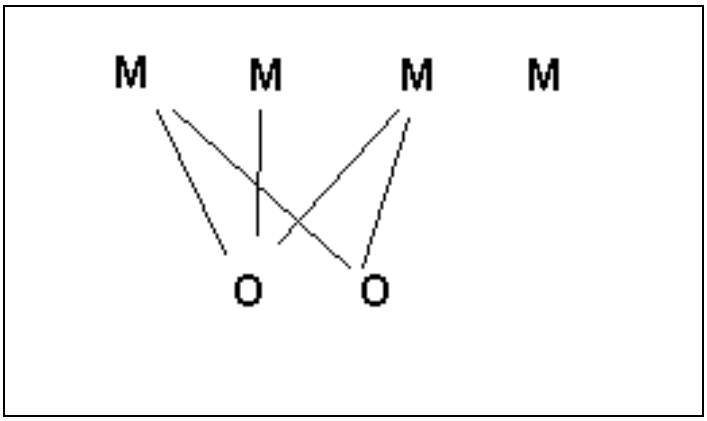
Şekil 4.5 için TSU;

MOCO=(4-2)/(4-1)=0,66

SMI=2*3/(4*3)=0,5

Coh=3/(4*2)=0,38

TSU=(66+50+38)/3,[7][5][4]=51,754



Şekil 4.6. 4 Metotlu 2 öznitelikli az 5 etkileşimli sınıf

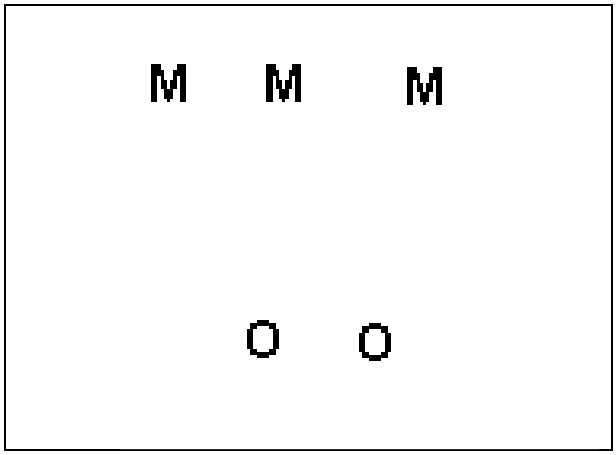
Şekil 4.6. için TSU;

MOCO=(4-2)/3=0,66
SMI=2*3/(4*3)=0,5
Coh=5/(4*2)=0,63

TSU=(66+50+63)/3,[7][5][6]=60,756

Şekil 4.5. ve

Şekil 4.6. de gösterilen sınıflarda çok az bir fark vardır. MOCO ve SCI olçumu aynı çıkmasına rağmen Coh ölçümü farklı sonuç vermiştir. Bunu sonucu olarak TSU ölçümü farklı çıkmış, ön görülen sonuç elde edilmiştir.



Şekil 4.7. 3 Metotlu 2 öznitelikli sınıf

MOCO=(3-3)/(3-1)=0

SMI=2*0/(3*2)=0

Coh=0/(3*2)=0

TSU=(0+0+0)/3,[0][0][0]=0,000

Şekil 4.5. ,

Şekil 4.6. ve Şekil 4.7. de gösterilen sınıfların LCOM1 değeri aynıdır. TSU ölçümünde ise farklı değerler çıkmıştır. Beklenen sonuç elde edilmiştir.

BÖLÜM 5. SONUÇ

Bu çalışmada temel sınıf uyum metrikleri teorik olarak incelenmiş, deneysel olarak incelemek için program yazılmıştır. İncelenen metriklerin zayıf noktaları belirtilmiş ve bu zayıflıklara karşı TSU metriği önerilmiştir.

Sınıf uyum metriği geliştirme çabalarına TSU metriğinin, sınıf uyumunu üç farklı yaklaşım ile ölçmesi, normalizasyon yapması ve sınıf içi her etkileşimin uyumu arttırması ile önemli katkı sağlaması beklenmektedir.

TSU metriği teorik olarak incelendiğinde sınıf uyumunu artırmada etkili olduğu görülse de teori ve deneysel çalışmalarla incelenmelidir.

KAYNAKLAR

BİEMAN, J. ve KANG, B-K., "CohesionandReuse in An Object-OrientedSystem", in Proceedings of ACM Symposium on Software Reusability (SSR'95), April 1995, pp. 259-262.

BRIAND, L.C., DALY, J.C. ve Wüst J. "A unified Framework for Cohesion Measurement in Object-Oriented Systems", Technical Report ISERN-97-05.Fraunhofer Institute for Experimental Software Engineering, 1997.

BRIAND, L.C., DALY, J.C. ve Wüst J. "A unified Framework for Cohesion Measurement in Object-Oriented Systems", Empirical Software Eng.; An Int'l J., vol.3, no. 1, pp. 65-117, 1998a.

BRİAND, L. C., DALY, J. W., PORTER, V., veWüst, J., ″A Comprehensive Empirical Validation of Design Measures for Object-Oriented Systems″, in Proc. of International Software Metrics Symposium, Bethesda, MD, Nov. 21 1998b pp.43-53

BRİAND, L. C., DALY, J. W., ve WÜST, J. K., ″A Unified Framework for Coupling Measurement in Object-Oriented Systems.″ IEEE Transactions on Software Engineering, vol. 25, no. 1, January/February 1999.

BALDASSARİ, B., ROBACH, C. ve du BOSQUET, L., "Early Metrics for Object Oriented Designs", IEEE Testability Assessment, 2004. IWoTA 2004. Proceedings. First International Workshop p.62-69, 2004.

BASILI V., MORRASCA S., and BRIAND L., "Defining and Validating Measures for Object-Based High-Level Design" IEEE Transactions On Software Engineering, vol. 25, no. 5, September/October 1999.

BRIAND, L. C., WÜST, J., DALY, J. W., ve PORTER, V. D., "Exploring the Relationship between Design Measures and Software Quality in Object-Oriented Systems", Journal of Systems and Software, vol. 51, no. 3, May 2000, pp. 245-273.

BUNGE, M. Treatise on Basic Philosophy: Ontology I: The Furniture of the World. Boston: Riedel.1977.

CHIDAMBER, S. R. ve KEMERER, C. F., "Towards a Metrics Suite for Object Oriented Design", in Proceedings of OOPSLA'91, 1991, pp. 197-211.

CHIDAMBER, S. R. ve KEMERER, C. F., "A Metrics Suite for Object Oriented Design", IEEE Transactions on Software Engineering, vol. 20, no. 6, 1994, pp. 476-493.

IEEE Std.1061-1998 “Standard for a Software Quality Metrics Methodology, revision.” Piscataway, NJ,: IEEE Standards Dept., 1998.

IEEE Std 610.12-1990, “IEEE Standard Glossary of Software Engineering Terminology”.

ERDEMIR, U., TEKİN, U., BUZLUCA, F., "Nesneye Dayalı Yazılım Metrikleri ve Yazılım Kalitesi", Yazılım Kalitesi ve Yazılım Geliştirme Araçları Sempozyumu (YKGS’08), Ekim 2008.

ERTEMEL H.Ö Nesneye Yönelik Yazilim Geliştirmede Kalite Ölçütlerinin Incelenmesi, Y.Lisans Yildiz Teknik Üniversitesi Elektrik-Elektronik Fakültesi Bilgisayar Mühendisliği Bölümü 2009

ESKİ S.“Nesneye Dayalı Yazılımlarda Sınama Ve Bakım Öncelikli Sınıfların Belirlenmesi” İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği 2011

FENTON, N.. “Software Measurement : A Necessary Scientific Basis” , IEEE Transactions on Software Engineering, 20(3), 199-206, (1994).

HENDERSON S., B., “Object-Oriented Metrics Measures of Complexity”, Prentice-Hall, (1996).

HITZ, M. ve Montazeri, B., “Measuring Coupling and Cohesion in Object-Oriented Systems”, Proceedings of International Symposium on Applied Corporate Computing. (Monterrey, Mexico, 1995).

KRIZ, J, FACTS ve Artifacts in Social Science: An Epistemological and Methodological Analysis of Empirical Social Science Techniques. McGraw Hill, 1988.

KURUBAŞ Ö., DURU N. “Sınıf Uyumunu Etkileyen Faktörler ve bu Faktörlere Uygun Bir Uyum. Ölçütünün Geliştirilmesi” Yazılım Kalitesi ve Yazılım Geliştirme Araçları Sempozyumu 2010

KURUBAŞ Ö. Nesneye Yönelik Sistemlerde Uyum Ve Sınıf İçi Bağımlılığı Ölçülmesinde Yeni Bir Yaklaşim, DoktoraTezi ,Kocaali üniversitesi, Elektronik ve Haberleşme Mühendisliği , 2011

KURNAZ S. , ÇETİN Ö. , İnce F. “Yazilim Mühendisliğinde Kalite Ve Uml” Havacilik Ve Uzay Teknolojileri Dergisi Temmuz 2003 Cilt 1 Sayi 2 (1-12)

LI, W., HENRY, S., “Object-Oriented Metrics That Predict Maintainability”, Journal of Systems and Software, 23, 111-122, (1993).

POSHYVANYK, D. ve MARCUS, A., “The Conceptual Cohesion of Classes”. 21st IEEE International Conference on Software Maintenance (ICSM'05) 2005 IEEE.

ÖZGEÇMİŞ

Ahmet Arslan, 1978 de doğdu. İlk, orta ve lise eğitimini Sakarya'da tamamladı. 2003 yılında Yıldız Teknik Üniversitesi Bilgisayar Programcılığı'nı bitirdi. Bir süre özel sektörde çalıştıktan sonra dikey geçiş ile girdiği Sakarya Üniversitesi Bilgisayar Mühendisliğini 2009 yılında bitirdi. 2009 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünde araştırma görevlisi olarak göreve başladı.