BSM462
Yazılım Testi

Hafta - 2

Program Testi Teorisi

Dr. Öğr. Üyesi M. Fatih ADAK

fatihadak@sakarya.edu.tr

İçerik

- Test Teorisi
- Goodenough ve Gerhart'ın Test Teorisi
- Tam Test
- Program Hata Türleri
- Test Verisi & Test Beyanı
- Güvenilirlik Koşulları
- Weyuker ve Ostrand'ın Test Teorisi
- Açıklayıcı Kriter
- GOURLAY Test Teorisi
- Test Metotlarının Gücü
- Testin Yeterliliği
- Testin Sınırları

Test Teorisi

- 1970’lerde yeni bir araştırma alanı boy gösterdi.
  - Test Teorisi
- Aslında programı test etme programlama kadar eski bir fikirdir.
- Bu fikir gün geçtikçe daha sistematik hale getirilmiş ve günümüzdeki halini almıştır.

Test Teorisi Neyi Vadeder?

- Programın yürütülmesi sürecindeki kusurları tespit eder.
- Farklı kaynaklardan test senaryolarını tasarlar.
- Bu test senaryolarından alt senaryo kümeleri seçebilir.
- Test senaryolarını seçme stratejisinin etkinliği belirler.
- Test durumlarının yeterlilik analizini yapar.

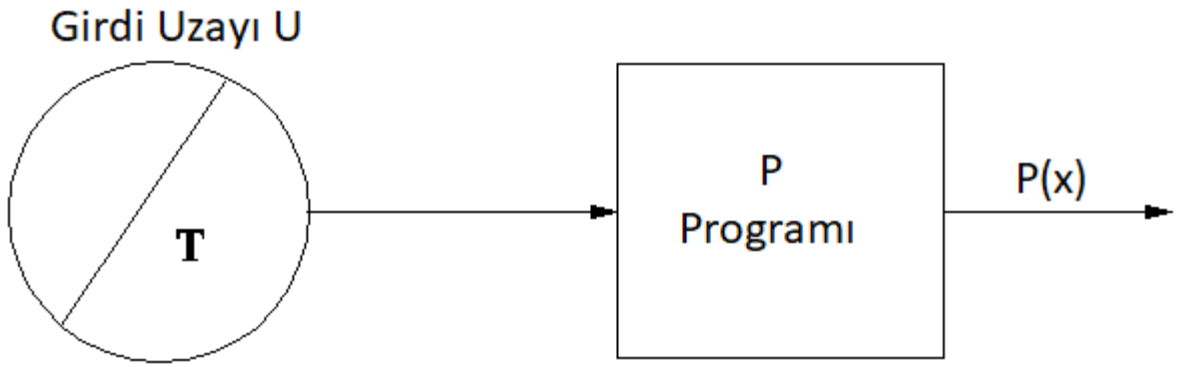
Üç Farklı Test Teorisi

- Goodenough ve Gerhart'ın Test Teorisi - 1975

  «Test Program çökerse başarılıdır.»
- Weyuker ve Ostrand'ın Test Teorisi – 1980
- Gourlay Test Teorisi - 1983

Goodenough ve Gerhart'ın Test Teorisinde Temel Kavramlar

- P bir program
  - U ise bu P programının alabileceği girdiler kümesi
  - $T \subseteq U$
  - P(x), P programının x girdisi verildiğinde oluşacak sonuç.



Goodenough ve Gerhart'ın Test Teorisinde Temel Kavramlar devam

- Eğer P(x) kabul edilebilir ise
  - OK(x) = true
- Eğer $\forall t \in T, OK(t)$ ise
  - T başarılıdır
  - Başarı testini geçmiştir. BAŞARI(T)
- İdeal Test: T İdeal testi geçmiştir, Eğer

$$OK(t), \forall t \in T \Rightarrow OK(u), \forall u \in U$$

Temel Kavramlar Devam

- Güvenilirlik Kriteri
  - Bir test seçme kriteri k'nın güvenilir olabilmesi için bu kriter tarafından seçilen testler **başarılı** olmalıdır.
- Geçerlilik Kriteri
  - Bir test seçme kriteri k'nın geçerliliği olabilmesi için P'nin doğru **olmadığı** durumlarda k kriteri, P'de **başarılı** olmayan en az bir T kümesi seçmelidir.
- K, k(x) testinin doğru olduğu söylenebilmesi için
  - $u \in U$ test kriterini $k \in K$ sağlamalıdır.

TAM TEST (Complete Test)

$$TAMTEST(\mathrm{T},\mathrm{K}) \equiv (\forall\, \mathrm{k} \in \mathrm{K})(\exists\, \mathrm{t} \in \mathrm{T})\, \mathrm{k}(\mathrm{t}) \wedge (\forall\, \mathrm{t} \in \mathrm{T})(\exists\, \mathrm{k} \in \mathrm{K})\, \mathrm{k}(\mathrm{t})$$

- T: Girdi alt kümesi
- K: Kriterler kümesi
- k(x): Herhangi bir kriteri uygulayan küme

Temel Teorem

$$(\exists T \subseteq U)(\text{TAMTEST}(\text{T}, \text{K}) \wedge \text{GÜVENİLİR}(\text{K}) \wedge \text{GEÇERLİ}(\text{K}) \wedge \text{BAŞARILI}(\text{T})) \Rightarrow (\forall \text{u} \in \text{U})\,\text{OK}(\text{u})$$

- Bir programın testten geçebilmesi için girdi uzayındaki bazı girdi kümeleri için güvenilir, geçerli, başarılı ve Tam testten geçmiş olması gerekir.
- Eğer bir tam test başarısız olursa bütün testler başarısız olur.

Temel Teoremin Uygulanmasının Zorluğu

- Bir programdaki hatalar bilinmediğinden güvenirliği ve bir kriterin geçerliliğini kanıtlamak imkansızdır.
- Bir kriterin güvenilir ve geçerli olmasının koşulu Girdi kümesindeki bütün girdileri seçebilmelidir.
- Bu pratik olmadığı gibi istenen bir durum da değildir.
- Eğer bir P programı doğru ise herhangi bir test başarılı olacaktır ve her seçilen kriter güvenilir ve geçerli olacaktır.
- Eğer bir P programı doğru değil ise P'deki tüm hataları bilmeden bir kriterin ideal olduğunu söyleyecek bir yol yoktur.

Program Hataları

- Yazılım testine her türlü yaklaşım varsayımlara dayanır.
- Hatalar iki ana nedenden dolayı oluşur.
  - Programla ilgili tüm koşulların yetersiz anlaşılmasından kaynaklanan hatalar
    - Örnek: Öğrencilerin genel ortalaması hesaplanması yerine final sınavının ortalamasının hesaplanması
  - Programla ilgili bazı kombinasyonların özel müdahalelere ihtiyaç duyduğununun bilinmemesi
    - Örnek: Bölme işleminde bölene sıfır girilme durumu

Program Hata Türleri

- Mantıksal Hata
  - Gereksinim Hatası
  - Tasarım Hatası
  - Yapı Hatası
- Performans Hatası
- Eksik Kontrol Akışı Hatası
- Uygun Olmayan Akış Seçimi
- Uygun Olmayan ya da Eksik Eylem

Mantıksal Hata

- Bu tür hatalarda program gerekli olan kaynaklardan bağımsız yanlış sonuçlar üretir.
- Kaynak yetersizliği bu hata içerisinde yer almaz.
- Üç kategoride incelenebilir.
  - Gereksinim hatası : Müşterinin gerçek talebine karşılık verememe durumu
  - Tasarım hatası : Talepleri karşılamama durumu
  - Yapı hatası : Tasarımı yanlış yapma durumu
    - Örnek: 100 öğrencinin ortalaması alınacak

```
for(int i = 0; i <= 100; i++)
{

}
```

→ Döngü 101 kez çalışır

Performans Hatası

- Kaynakların limitler dahilinde kullanılmadığı ve bu limitlerin dışında programın sonuç ürettiği
- Veri işlemede her 10 ns bir veri geliyorsa, diğer veri gelmeden elindeki veriyi 10 ns içerisinde işlemek.
- Aşırı bellek kullanımı
- Yüksek işlemci sıcaklığı

Eksik Kontrol Akışı Hatası

- s nesne referansının null gelebilme ihtimali göz ardı edilmiş.

```
Sayi s = EnlyiSayiyiGetir();
if (s.Deger < 100) return true;
```

- Eksik kontrol akışı içermeyen versiyonu

```
Sayi s = EnlyiSayiyiGetir();
if (s != null && s.Deger < 100) return true;
```

Uygun Olmayan Akış Seçimi

Talep Edilen	Gerçekleştirilen
if(x) fonk1() else fonk2();	if(x && y) fonk1() else fonk2();

- y true geldiği sürece talep edilenin karşılandığı sanılacaktır.

Uygun Olmayan ya da Eksik Eylem

- Üç farklı şekilde bu istenmeyen durum ile karşılaşılabilir.
  - Yanlış denklem kullanımı

    Örnek: a = a x b yerine a = a + b yazılmış.

    Bu bile bazı değerlerde istenen sonucu üretebilir a=1,5 b=3 gibi
  - Bir değişkene atanması gereken değerin atlanması eksik eyleme örnek verilebilir.
  - Bir fonksiyonu yanlış argümanlar ile çağırma uygun olmayan eyleme örnek verilebilir.
- Bu tür için en tehlikeli durum sadece bazı durumlarda uygun olmayan ya da eksik eyleme düşmesi
  - Böyle bir durumun test ile yakalanması için bütün girdi kombinasyonları denenmelidir.

Test Verisi & Test Beyanı

- Test Verisi
  - Test verisi girdi uzayındaki asıl verilerdir.
  - Test seçme kriterlerini toplu olarak yerine getirir.
- Test Beyanı (Test Predicate)
  - Programın doğru çalışması için koşulların açıklanmasıdır.
  - Test edilecek programın test edilmesi gereken yönlerini açıklar.
  - Test verisi seçiminin motive edici gücüdür.

Güvenilirlik Koşulları

- Test Beyanlarının kümesi güvenilir olması için aşağıdaki koşulları sağlamalıdır.
  - Bir programdaki her dallanma koşulu K koşul kümesinde eşdeğer bir durum tarafından temsil edilmelidir.
  - Programdaki her olası sonlandırma durumu (örnek: dizi taşması) K koşul kümesindeki bir koşul tarafından temsil edilmelidir.
  - Programın doğru çalışmasını sağlayan her veri yapısı özellikleri ve bilgisi K koşul kümesindeki bir koşul tarafından temsil edilmelidir.

Goodenough ve Gerhart'ın Test Teorisinin Dezavantajları

- Güvenirlilik ve geçerlilik kavramları bir programın girdi uzayının tamamı için tanımlanmıştır.
- Girdi uzayının tamamının test edilmesi mantıklı ve pratik değildir.
- Güvenirlilik ve geçerlilik kavramları programdan programa değişir. Fakat bir test kümesi programlardan bağımsız olmalıdır.
- Güvenirlilik veya geçerlilik debug süresi boyunca korunamaz.
- Test seçme kriterleri monoton bile değildir.

Weyuker ve Ostrand'ın Test Teorisi

- Goodenough ve Gerhart'ın test teorisindeki ana problem güvenirlilik ve geçerlilik kriterleri bir programdaki hataların varlığına ve türlerine bağlıdır.
- Weyuker ve Ostrand ise Goodenough ve Gerhart'ın Test Teorisi üzerinde düzenleme yapılmış test teorisi önermişlerdir.
- Test seçme kriteri, programdan ziyade sadece program özelliklerine bağlıdır.
- Verilen bir çıktı için düzgün ideal test seçme kriteri öneriliyor.

Düzgün İdeal Test Seçme Kriteri

- K Kriteri düzgün geçerli olabilmesi için

$$(\forall P)[(\exists u \in U)(\neg \mathrm{OK}(\mathrm{P},\mathrm{u})) => (\exists \mathrm{T} \subseteq \mathrm{U})(\mathrm{K}(\mathrm{T}) \,\&\, \neg \mathrm{BAŞARILI}(\mathrm{P},\mathrm{T}))]$$

- K Kriteri düzgün güvenli olabilmesi için

$$(\forall P)(\forall \mathrm{T}_1, \forall \mathrm{T}_2 \subseteq U)[(\mathrm{K}(\mathrm{T}_1) \,\&\, \&K(T_2)) => (BAŞARILI(P,T_1) \Leftrightarrow BAŞARILI(P,T_2))]$$

- Düzgün İdeal Test Seçme Kriterinin hem düzgün geçerli hem de düzgün güvenli olması gerekir.

Weyuker ve Ostrand'ın Test Teorisi Üzerine Tartışma

- Yukarıda bahsedilen düzgün ideal geçerlilik ve güvenirlilik aslında kapsamlı (exhaustive) testlere yol açacaktır.
- Kapsamlı testlerin pratik olmadığı bilinir.
- Testleri başarısız kılacak (ya da yenecek) bir program her zaman yazılabilir.
- Bir P programı T testini geçip başka bazı girdilerde başarısız oluyorsa bu P programı T testini başarısız kılmıştır (ya da yenmiştir).

"Test hataların sadece varlığını ortaya çıkarabilir.

Yokluğunu asla.." Dijkstra

Açıklayıcı Kriter (*Revealing Criterion)*

- Weyuker ve Ostrand'ın Alt alan uzayını kullanarak açıklayıcı kriteri tanımlamışlardır.
  - Girdi Uzayı U'nun alt alan uzayı S
  - Test seçme Kriteri K, açıklayıcı kriter olabilmesi için

$$AÇIKLAYICI\_KRİTER(K,S) \Leftrightarrow (\exists u \in S)(\neg OK(\mathrm{u})) \Rightarrow (\forall \mathrm{T} \subseteq \mathrm{S})(\mathrm{K}(\mathrm{T}) \Rightarrow \neg \mathrm{BAŞARILI}(\mathrm{T}))$$

- Açıklayıcı kriterin en büyük avantajı bütün bir girdi uzayını almak yerine alt alan uzayı ile ilgilenmektedir.

GOURLAY Test Teorisi

- Bu test teorisi üç varlık kümesi arasında ilişki kurar.
  - Özellik
  - Program
  - Test
- Test seçmede farklı yöntemleri karşılaştırmak için bir temel sağlar.

GOURLAY Test Teorisi

- $\mathcal{P}$: Bütün programların kümesi

$$p \in P \subseteq \mathcal{P}$$

- $\mathcal{S}$: Bütün özelliklerin kümesi

$$s \in S \subseteq \mathcal{S}$$

- $\mathcal{T}$: Bütün testlerin kümesi

$$t \in T \subseteq \mathcal{T}$$

GOURLAY Test Teorisi

- **p OK(t) s** => t tarafından test edilen p programı s özelliği tarafından kabul edilmiştir.
- **p OK(T) s** => p programı $\forall t \in T$ için s özelliği tarafından kabul edilmiştir.
- **p corr s** => s özelliğinde p doğrudur.

GOURLAY'a Göre Test Sistemi

- < $\mathcal{P}$, $\mathcal{S}$, $\mathcal{T}$, corr, ok> bir araya gelerek test sistemini oluşturur.
  - corr $\subseteq \mathcal{P} \times \mathcal{S}$
  - ok $\subseteq \mathcal{T} \times \mathcal{P} \times \mathcal{S}$

$$\forall p \forall s \forall t (p \, corr \, s \Rightarrow p \; ok(t) \, s)$$

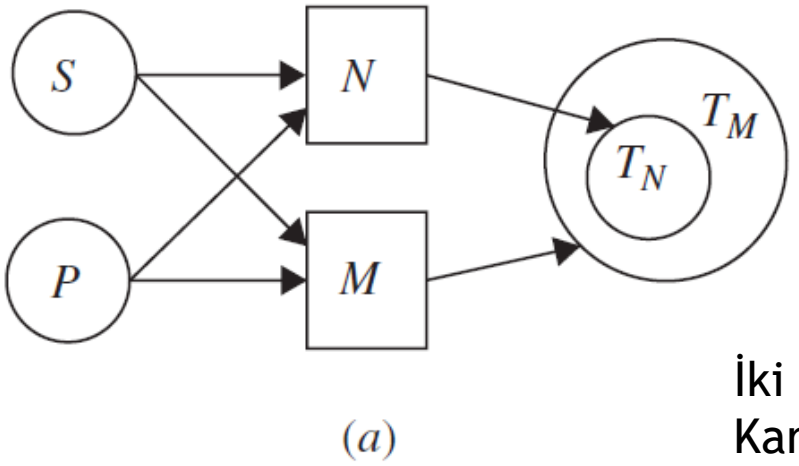
GOURLAY'a Göre Test Metodu

- Bir test metodu $M : P \times S \rightarrow T$ şeklinde tanımlanır.
  - Program Bağımlı
    - $T = M(P)$
  - Özellik Bağımlı
    - $T = M(S)$
  - Beklenti Bağımlı
    - Müşteriler tarafından kendi beklentileri doğrultusunda yazılan testlerdir.
    - Devamlı operasyon testleri
    - Kullanılabilirlik testleri

Test Metotlarının Gücü

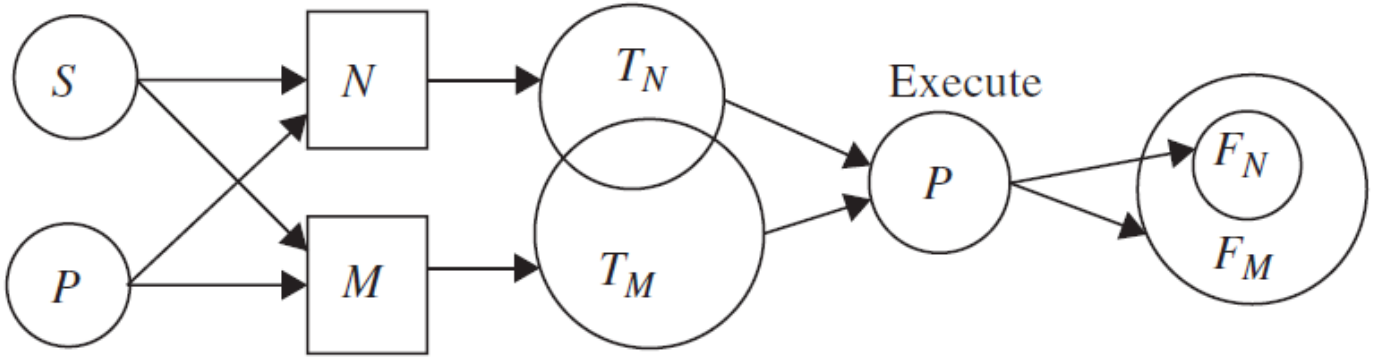
- M ve N iki ayrı test metodu
- M metodunun en az N kadar iyi olmasını istiyorsak…
  - N bir hata bulduğunda M'de bulmalıdır.
  - F_M ve F_N , M ve N test metotlarının ortaya çıkardıkları hatalar kümeleridir.
  - T_M ve T_N , M ve N test metotlarının ürettiği kümeleridir.

Test Metotlarının Gücü



(a)

İki farklı şekilde test metotlarının gücü Karşılaştırılabilir.



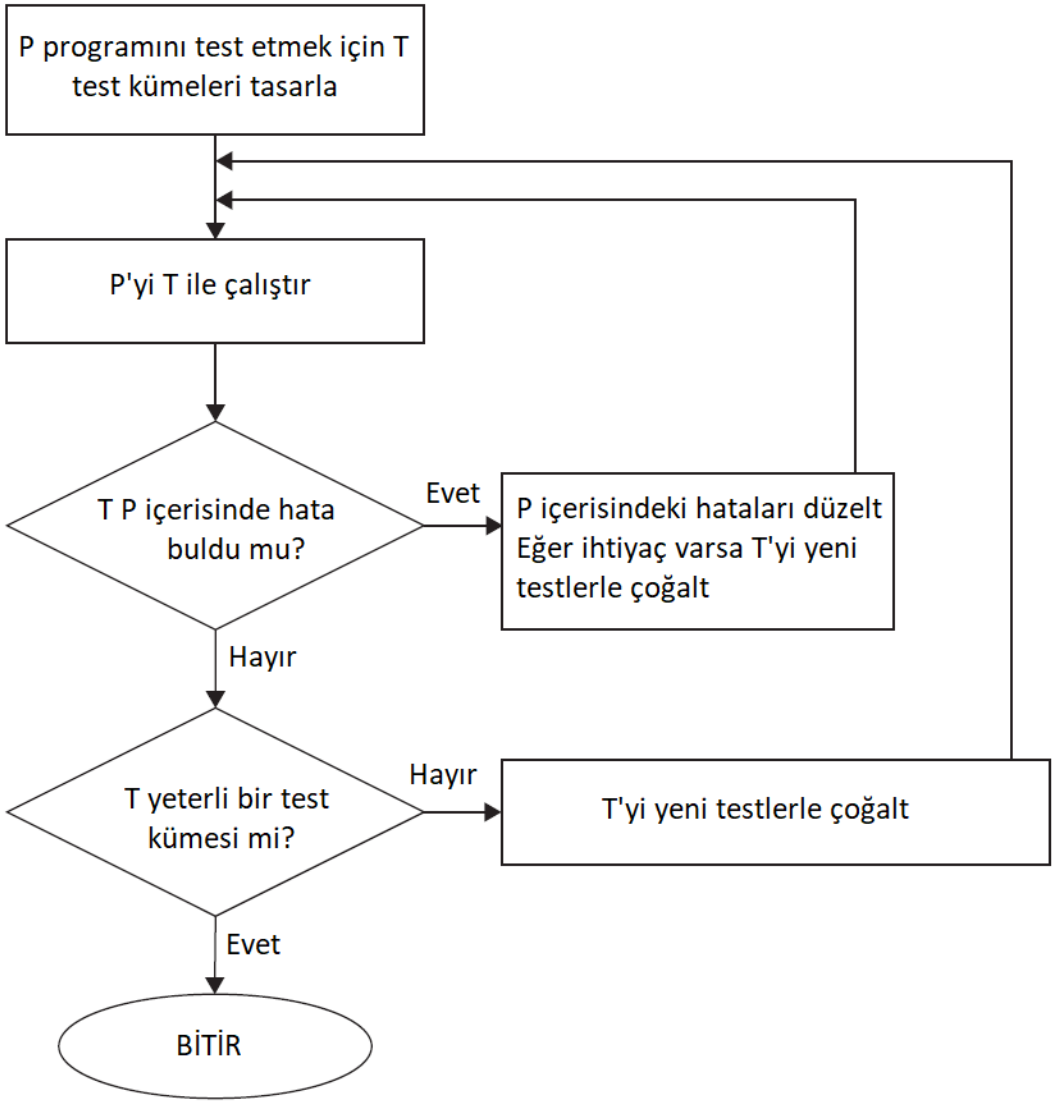
(b)

Her iki şekilde de M en az N kadar iyidir.

Testin Yeterliliği

- Planlanan test senaryolarına yeni test senaryoları eklenebilir.
  - Bu eklenen yeni test kümesi yeni hatalar ortaya çıkarmıyorsa ikilemden söz etmek mümkündür.
    - P programı hatasız bir programdır ya da
    - Yeni test kümesi yeni hatalar ortaya çıkaracak kadar iyi değildir.
    - Dijkstra'ya göre P programının hatasız olduğunu söyleyemeyiz.
    - Burada yeni test kümesinin en az diğer test kümeleri kadar iyi olduğunu göstermeliyiz.

Testin Yeterliliği



Test Yeterliliği

- Test yeterliliğini değerlendirmek için iki pratik metot
  - **1- Hata Ekme (Fault Seeding)**

    - Bu yöntemde belli sayıda hata programa dahil edilir. T test kümesi ile P programı çalıştırılır.
    - Eğer T testleri dahil edilen hataların % a kadarını bulursa gerçek hatalarında % a kadarını bulacaktır.
    - T testlerinin %100'nü bulması durumunda test yeterliliği hakkında kendimizi daha güvenli hissedebiliriz.

Test Yeterliliği

- Test yeterliliğini değerlendirmek için iki pratik metot
  - **2- Program Mutasyonu**

    - P programında ufak değişiklikler yapılır. Bu ufak değişiklikler hata içerebilir.
    - Eğer değişikliğe uğrayan her hata beklenmedik bir sonuç üretiyorsa T test kümesi yeterlidir sonucuna varılır.

Testin Sınırları

- Astronomik büyüklükte bir girdi kümesi seçilmesi çok maliyetli bir işlemdir.
- Bundan dolayı her zaman küçük boyutlarda bir küme seçilir. Bazı önemli girdilerin gözden kaçması demektir.
- Test kümesi seçildiğinde ikinci bir problemle karşılaşılacaktır.
- Testin doğruluğunun teyidi. Sonuç beklenen gibi çıkmış olabilir ama doğru bir şekilde mi üretildi?

Referanslar

- Naik, Kshirasagar, and Priyadarshi Tripathy. *Software testing and quality assurance: theory and practice*. John Wiley & Sons, 2011.
- Ammann, Paul, and Jeff Offutt. *Introduction to software testing*. Cambridge University Press, 2016.
- Padmini, C. "Beginners Guide To Software Testing." (2004).
- Archer, Clark, and Michael Stinson. *Object-Oriented Software Measures*. No. CMU/SEI-95-TR-002. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1995.
- Pandey, Ajeet Kumar, and Neeraj Kumar Goyal. *Early Software Reliability Prediction*. Springer, India, 2015.
- F.R.D. Velasco, A method for test data selection, Journal of Systems and Software, Volume 7, Issue 2, 1987, Pages 89-97.
- Gourlay, J. S. (1983). A mathematical framework for the investigation of testing. *IEEE Transactions on software engineering*, (6), 686-709.
- Weyuker, E. J., & Ostrand, T. J. (1980). Theories of program testing and the application of revealing subdomains. *IEEE Transactions on Software Engineering*, *6*(3), 236.
- Goodenough, J. B., & Gerhart, S. L. (1975). Toward a theory of test data selection. *IEEE Transactions on software Engineering*, (2), 156-173.