Grafiksel Kullanıcı Arayüzleri için Düzenli İfade Bazlı Test Kapsama Kriterleri

Onur Kılınççeker[1,3] and Fevzi Belli[2,3]

[1] Mugla Sitki Kocman University, Mugla, Turkey
[2] Izmir Institute of Technology, Izmir, Turkey
[3] Paderborn University, Paderborn, Germany
okilinc@mail.upb.de,belli@upb.de

Özet. Grafiksel Kullanıcı Arayüzleri (GKA), insan-bilgisayar etkileşimi açışından, bilgisayar tabanlı sistemlerin ana bileşenleridir. Bu çalışma, GKA'ların düzenli ifadeler (Dİ; *regular expression*) ile modellenmesi ve dahası test kapsama (*coverage*) kriterleri elde edilmesi için yeni bir yaklaşım öne sürmektedir. Verilen GKA, ya doğrudan bir Dİ ile, ya da (pratikte daha çok yapıldığı şekilde) olay bazlı bir yönlü çizge (YÇ; *digraph*) ile modellenir ve bu YÇ bir Dİ'ye dönüştürülür. Ne var ki bu Dİ semantik bakımından yalnızca olay bazlıdır. Modeli durum açısından zenginleştirmek için Dİ özel bir teknik ile endekslenir. Önerilen yaklaşım, bu endekslenmiş Dİ'yi analiz ederek durum ve olay bazlı test kapsama kriterleri üretir. Ön araştırmalar göstermektedir ki, önerilen yaklaşım, diğerleri ile karşılaştırıldığında, daha özlü (*compact*) test takımı üretimine olanak sağlamaktadır.

Anahtarlar: Grafiksel Kullanıcı Arayüzü, Modelleme, Test Kapsama Kriterleri, Düzenli İfadeler, Sonlu durum makinaları

Coverage Criteria For Testing Graphical User Interfaces Based On Regular Expressions

Onur Kılınççeker[1,3] and Fevzi Belli[2,3]

[1] Mugla Sitki Kocman University, Mugla, Turkey
[2] Izmir Institute of Technology, Izmir, Turkey
[3] Paderborn University, Paderborn, Germany
okilinc@mail.upb.de,belli@upb.de

Abstract. Graphical User Interfaces (GUI) are popular for enabling comfortable user interactions (UI) of computer-based systems with users. This paper introduces a new approach for modeling GUIs with regular expresion

(RE), and moreover, obtaining coverage critaria for GUI testing. The GUI under test is assumed to be modeled by a RE, the symbols of which are semantically interpreted as events. For enriching this RE by information also about states, the RE will be scanned by equivalent FSAs, forwards and backwards. Thereby, the appearing states will be noted as indices of the RE. The resulting coded format of the RE, which is obtained by simultaneous backwards and forwards scanning, contains all information necessary for generating coverage citeria that includes both event and state information. Preliminary experiments show that proposed approach enables to generate more compact test cases at less costs compared with others.

Keywords: Graphical User Interface, Modeling, Test Coverage Criteria, Regular Expression, Finite State Automata, Event Sequence Graphs

1 Giriş

Günümüz karmaşık yazılım sistemleri gözönünde bulundurulduğunda bilgisayar tabanlı sistemlerin ana bileşenlerinden biri olan Grafiksel Kullanıcı Arayüzlerin (GKA) geniş kullanım alanları, gün geçtikçe daha karmaşık GKA'ların inşa edilmesini sağlamaktadır [15]. GKA'lar ile geleneksel yazılımlar arasında bir çok farklılık göze çarpmaktadır. Bu farklılıklar, GKA'ların tasarım ve sınanmasnda farklı yetenekler ve tekniklerin geliştirilmesini gerektirmektedir [9].

GKA'larin modellenmesi ve elde edilen modeller aracılığı ile inşa ve test edilmesi süreci için gereken zaman, geleneksel yazılımlarda olduğu gibi, modelsiz sürece nazaran daha ekonomiktir. Arzu edilen bu zaman ve masraf tasarrufu, şüphesiz analiz ve test işlemleri için uygun modeller seçilmesi ile mümkündür.

Bu bildiri, GKA'ların düzenli İfadeler (Dİ) ile modellenmesini önerir ve bu modellerle ifade edilebilen GKA'ların test edilmesi için gerekli test kapsama kriterleri üretilmesi amacı ile yeni bir yaklaşım öne sürmektedir. Test kapsama kriterleri test sürecini değerlendirme ve sonlandırma açısından çok önemlidir.

Yaklaşım ana hatları ile şu adımlardan oluşmaktadır:

1. İncelenen sistem (İS; *system under consideration*) bir Dİ ile modellenir. Bu modelleme şekli yaygın olmadığından, önce olay bazlı bir yöntem ile, örneğin yönlü çizge (YÇ; *digraph*) ile modellenir ve bu YÇ bilinen kuramsal metodlar ile otomatik olarak bir Dİ'ye dönüştürülür [1],[12], [16].
2. Elde edinilen Dİ, yalnızca olay bazlıdır ve aşağıda belirtilen özel teknik ile endekslenir. Bu endekslenmiş Dİ, İS'in olay ve durum bazlı karakteristik bilgilerini içermektedir.
3. Bu bilgiler Bağlam Kapsama Kriterlerini (*context coverage criteria*) üretmek için kullanılır.

İkinci adımda değinilen teknik, ilk etapta elde edinilen Dİ'ye ve bu Dİ'nin tersten okunmasından (*mirrored term*) oluşan sonlu durum makinalarını (SDM) kurar. Akabinde eldeki Dİ, bu makinalar tarafından önce doğrudan (*forwards scanning*), sonra tersinden (*backwards/reverse scanning*) okunur ve makinaların tarama anlarında aldıkları durumlar Dİ'nin o anki olayına (=simgesine) üst (ön okuma;

forwards indexing) ve alt (*backwards/reverse indexing*) endeks olarak işlenir. Dİ'nin aynı anda üst ve alt endekslenmesi (*coding*) önerilen yaklaşım için gerekli bütün bağımsal (*contextual*) bilgileri içerir.

Bildirinin 2. Bölüm'ünde ilgili çalışmalar gözden geçirilecek, sunulan yaklaşım ile ilgileri kısaca açıklanacaktır. 3. Bölüm'de GKA modellenmesi için önerilen yöntem bir örnek ile anlatılacak, ardından 4. Bölüm'de Dİ'lerin önemli özelliklerinin endeksleme yöntemi ile elde edilmesine konu olacaktır. 5. Bölüm'de yeni kapsama kriterleri tanımlanacaktır. 6. Bölüm'de ise önerilen kapsama kriterlerinin bir uygulaması olarak test üretimi için örnekler verilecektir. Son olarak 7. Bölüm'de elde edilen sonuçlar ve gelecek için planlanan çalışmalar özetlenecektir.

2 İlgili Çalışmalar

Literatürde verilen bir sistemin bir model yardımıyla test edilmesi 'model tabanlı test etme' olarak geçmektedir. Bu bildiride önerilen yaklaşım ile ilişkili modeller: Olay Sıra Çizgesi (OSÇ) (*Event Sequence Graph*) [9] ve Olay Akış Çizgesi (OAÇ) (*Event Flow Graph*) [15]. OSÇ'ler bir çok interaktif sistemin modellenmesinde kullanılmaktadır. Bunlara gerçek zamanlı sistemler, gömülü sistemler ve GKA'lar örnek olarak verilebilir [8]. Bununla birlikte yine bu model aracılığı ile sistemin analiz edilmesi ve test durumlarının üretilmesi [6],[7],[8],[9] ise bir diğer avantajlı yanıdır.

OAÇ'ler ise yapısal olarak OSÇ'ler ile benzerlikleriyle birlikte her iki modelde bir çok ortak uygulama alanı bulmaktadır. Bunlar ise çoğunlukla GKA'ların modellenmesi ve test edilmesi üzerine yoğunlaşmaktadır [14], [15].

Sonlu durum makinaları (SDM) on yıllardan beri sıralı (*sequential*) yazılım ve donanım sistemlerinin modellenmesi ve test edilmesi için kullanılmaktadır. Bu çalışmalara ilk olarak Chow [10] öncülük etmiş, SDM ile modellenen sıralı sistemlerin sınanması için 'w-metodu' olarak adlandırılan bir metot öne sürmüştür. Bu çalışmanın ardından SDM bazlı modellerin test sürecinde kullanılması için bir çok metot ileri sürülmüştür.

Dİ'lerin gerek modelleme gerekse test için kullanımıyla ilgili çalışmalar diğer modellerle kıyaslandığında yok denecek kadar azdır. Örneğin, Web of Science veri tabanında "düzenli ifade" ve "test üretimi" anahtar kelimeleri aratıldığında, konuyla alakalı yalnızca altı çalışma ile karşılaşılmaktadır. Bunlardan bir tanesi [17] donanım için test üretimi ve daha ziyade belirli bir seviyede verilmiş donanımın test edilebilirlik analizi ve optimizasyonu üzerinedir. Diğer beş çalışma ise Dİ'ler ile modellenen yazılımların test edilmesi hakkındadır. Bu veritabanında rastlanılmayan [11]'de Dİ'ler sıralı devrelerin test edilmesi için rastgele test takımları üretiminde kullanımı için bir yaklaşım verilmektedir. Ayrıca Shaw tarafından [18]'de öne sürülen yaklaşım, Dİ'lerin yazılım tasarımı (*design*) ve belirtimi (*specification*) için kullanımı üzerinedir. Liu ve Miao çalışmasında [13] Dİ yardımıyla yazılım davranışları için test modelinin kurulması hakkındadır. Bir diğer çalışmada Belli ve Grosspietsch [3] Dİ ile modellenen karmaşık yazılım sistemlerinin kusur dayanıklılığı (*fault tolerance*)

açısından analiz edilmesi ve bu analiz sonucunda, sistem arzu edilen özellliği içermiyorsa genişletilip kusur dayanıklı hale getirilmesi ile ilgilidir.

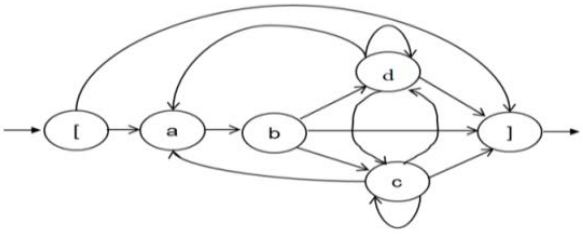
Mevcut literatür incelendiğinde Dİ'lerin GKA'ların test kapsanımları için kullanımına ilişkin bir çalışmaya şu ana kadar yapılan araştırmalarda rastlanılmamıştır.

3 GKA ve Modellenmesi

Günümüzde kullanılan GKA'lar basitten karmaşığa çok çeşitli şekillerde karşımıza çıkmaktadır. Aşağıda **Şekil 1**'de basit bir örnek verilmektedir.

Şekil 1. Örnek bir GKA [7]

Şekil 1'de gösterilen örnek GKA'yı en basit olarak bir yönlü çizge (YÇ; *digraph*) ile modelleyebiliriz (**Şekil 2**'ye bkz.).



[: start (entry menu);]: finish (exit menu); a:pick an object; b: copy an object; c: delete an object; d: paste an object

Şekil 2. Şekil 1.'de gösterilen örnek GKA'nın yönlü çizge (YÇ) modeli

Bu örnekten şu test dizilerini üretebiliriz:

$$[ab],[abc],[abcd] \quad (1)$$

Aynı test dizileri bu YÇ'ye tekabül eden Dİ ile edinmek mümkündür.

Düzenli İfadeler: Aşağıdaki kurallar çerçevesinde, bir *düzenli ifade*, verilen bir alfabe α'nın sıfır veya daha fazla *a, b, c, ...* sembollerinin dizisi ile ifade edilir. Bu diziler aşağıda verilen operatörlerin aracılığı ile kurulur;

- *Bitiştirme* – belirli bir sembol ile ifade edilmeyen bir operatör. Öyle ki *ab* gibidir ve 'a izlenir b tarafından' anlamındadır.
- *Seçim (Birleştirme)*, + ile gösterilir. Öyle ki *a+b* gibidir ve '*a veya b*' anlamındadır.
- *Yineleme (Kleene Yıldız operatörü)*, * ile gösterilir. Öyle ki *a** gibidir ve *'a isteğe bağlı olarak tekrarlanır'* (sıfır tekrarda içerilmektedir ve bu 'boş kelime *λ'yı ifade eder*). Benzer şekilde, a^+ ise a'nın en az bir kez oluşunu belirtir, yani *λ* hariç tutulur.

Şekil 2'deki YÇ'ye tekabül eden Dİ'yi şu şekilde tanımlayabiliriz.

$$[(ab(c+d)^*)^*] \qquad (2)$$

Kolayca görüleceği gibi, (1)'deki test takımları üstte gösterilen Dİ tarafından da üretilebilir. Konunun ayrıntıları için [4],[9]'a bkz.

4 Düzenli İfadelerin Önemli Özelliklerinin Endekslenme Yolu ile Tanımlanması

Giriş bölünde amacı belirtilen ve özetlenen Dİ endekslenmesi, bu bölümde ayrıntılı olarak açıklanacaktır. Bu çerçevede, semboller arasındaki bağlamsal ve uygunsal ilişkilerin çıkarılabilmesi için gerekli bağlamsal (*contextual*) ileri, geri, sağ ve sol endeksleme kavramları tanımlanacaktır. Bu endeksleme işlemlerine dayanarak elde edilecek olan İleri Bağlam, Geri Bağlam ve Uygunluk Tablosu tanımları verilecektir. Bu bölümde tanıtılacak olan kavramlar bir sonraki bölümde GKA test kapsamları üretilmesi aşamasında kullanılacaktır.

4.1 Bağlamsal (contextual) İlişkilerin Görüntülenmesi

Bu bölüm, bir örnek aracılığı ile Dİ'nin karakteristik ilişkisini ortaya çıkaran belirli bir forma dönüştürülmesini göstermektedir. Örnek olarak (2)'de verilen Dİ kullanılacaktır.

İlk aşamada, (2)'nin her bir sembolü oluş sırasına göre ifade edilir. Böylece aşağıdaki *bağlamsal (contextual) endekslenmiş* ifade elde edilir.

$$([^1(((a^1(b^1((c^1+d^1)^*)))^*)]^1) \qquad (3)$$

Burada, "a^1" "a"nın ilk defa var olması demektir. Aynı şekilde diğer sembollerin endeksleri de "1"dir. Yani (3) içerisinde her sembol yalnızca bir defa bulunmaktadır.

4.2 Dİ'ye Doğrudan ve Tersinden Tekabül Eden SDM Kurulması

Ardından, iyi bilinen yönteme dayanarak [3],[5], Dİ'ye tekabül eden durum tablosu E^{forw} elde edilir (**Tablo 1**). Bu tablo, verilen Dİ'ye *doğrudan* (*forward*) tekabül eden SDM'dir. E^{forw}, SDM'nin Dİ'ye dönüşümünde kaybolan, ya da doğrudan Dİ ile çalışıldığında mevcut olmayan durum bilgisini Dİ'ye endeks ile eklemeyi sağlayacaktır.

Sembol	Durum	[	a	]	b	c	d
	0	1					
$[^1$	1		2	3			
a^1	2				4		
$]^1$	3						
b^1	4		2	3		5	6
c^1	5		2	3		5	6
d^1	6		2	3		5	6

Tablo 1. E^{forw} Durum Tablosu

***E^{forw}* kurulması hk.** Örneğin, E^{forw} sıfır (0) başlangıç durumundadır ve "[" sembolünü okur, bu doğru bir şekilde "$[^1$" dir ve ardından durum 1'e geçilir.

Sembol "a" doğru olarak durum 1'de okunur ve bu daima "a^2" olur. Ardından bu durum 2'ye geçileceği anlamına gelir. (2) tarafından oluşturulmayan her bir sembol E^{forw} tarafından da kabul edilmez.

4.3 Olay ve Durumların Kaynaştırılması - Dİ'nin İleri Endekslenmesi

s^i sembolleri ve (2) arasındaki ek ilişkiler ışığında, s^i sembolünün endeksi *i* ile s^i'yi içeren durumların kümesi ile yer değiştirilirse, E^{forw}'un durumları elde edilir.

Böylece (4)'teki T^{forw} oluşturulmuş olur. E^{forw} ile yeniden tanımlanan durum bilgisi T^{forw}'un endekslerini tanımlamaktadır.

$$[^1(\ a^2b^4((c^5+d^6))^*)^*\ ]^3 \tag{4}$$

Görülen endeksler, verilen Dİ'nin E^{forw} tarafından doğrudan, yani soldan sağa (*forward*), "okunması" durumunda sembollerde bırakacağı durum (*state*) izleri olarak kabul edilebilir. Bu işlem (2)'nin *ileri* (*forward*) *endekslemesi* olarak adlandırılır.

4.4 Dİ'nin Ters Endekslenmesi ve Kodlanması

İleri endekslemeye benzer şekilde, (2)'nin ters görüntüsü (*mirror*) T^{mirr} endekslendiğinde T^{mirr} (5) oluşturulur.

$$]^1\ ((((d^1+c^1)^*b^1)a^1))^*[^1) \tag{5}$$

Şimdi, (4)'ü elde etmek icin uygulanan işlemlerin aynısını (5)'e, yani T^{mirr}'a uygularsak T^{mirr_forw} elde edilir (6). T^{mirr_forw} sayesinde (2)'nin tersten okunmasıyla elde edilen Dİ'ye tekabül eden SDM'nin durum bilgisi tekrardan kazanılmış olur.

$$(]^1\ ((((d^4+c^3)^*b^2)a^6))^*[^1) \tag{6}$$

Bu işlem *geri endeksleme* olarak adlandırılır. Burada, ikinci bir ters görüntü işleminden sonra (7) yani $T^{\,mirr\,forw\,mirr}$ veya T_{back} elde edilir.

$$[^1(\ a^6\ \ b^2\ ((\ c^3+d^4\ ))^*\ )^*\ \ ]^1 \tag{7}$$

İleri ve geri endeksleme aynı anda gerçekleştirilirse, (8) yani T_{back}^{forw} oluşturulur.

$$[_5^1\ (a_6^2\ b_2^4\ (c_3^5 + d_4^6\ )^*)^*\ ]_1^3 \tag{8}$$

Bu çift yönlü endeksleme (2)'nin *kodlanması* olarak adlandırılır. Buradaki önemli bir araç kodlanan s_j^i sembollerindeki bir ileri endeks *i* ve bir geri endeks *j*'nin tüm ikililerini içeren uygunluk ilişkisi *C*'dir. Bu *isj*'nin *iCj* notasyonları ile tanımlanır. Yani *i* ve *j* durumları *s* sembolü ile uygundur anlamına gelmektedir. **Şekil 3**.(b) (2) için bu *C* ilişkisini vermektedir.

İkinci olarak, daha karmaşık bir araç, *sol bağlam* ve *sağ bağlam* sırasıyla l^{forw} , l_{back} ve r^{forw}, r_{back} ilişkileri kullanılır. Bunlar her bir s^i ve s_j için birinin sonraki ve önceki sembollere karar verir. **Şekil 3**.(a) (2) için bu bağlam ilişkisini vermektedir. Bu konularda detaylı bilgi [5]'de bulunabilir.

Şimdiye kadar önemli olan bir çok kavram açıklandı ve bunlara örnekler verildi. Bir sonraki bölümde, Tanım 1'den itibaren öne sürülen yaklaşımda kullanılacak kavramlara yer verilecektir.

x'L	Symbol	x'R
	1'[	2'a + 3']
1'[+ 4'b + 5'c + 6'd	2'a	4'b
1'[+ 4'b + 5'c + 6'd	3']	
2'a	4'b	2'a + 3'] + 5'c + 6'd
4'b + 5'c + 6'd	5'c	2'a + 3'] + 5'c + 6'd
4'b + 5'c + 6'd	6'd	2'a + 3'] + 5'c + 6'd

(a)

L,x	Symbol	R,x
a,6	b,2	],1 + c,3 + d,4 + a,6
b,2 + c,3 + d,4 + [,5	],1	
b,2 + c,3 + d,4	c,3	],1 + c,3 + d,4 + a,6
b,2 + c,3 + d,4	d,4	],1 + c,3 + d,4 + a,6
	[,5	],1 + a,6
b,2 + c,3 + d,4 + [,5	a,6	b,2

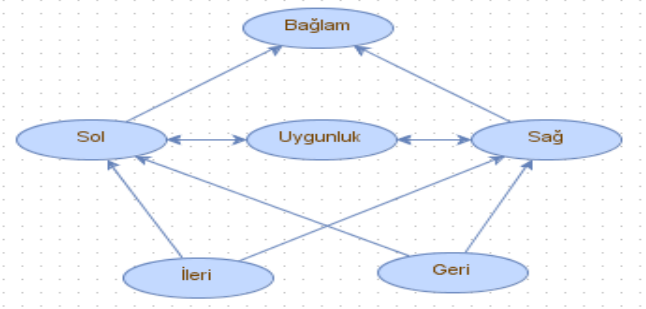
```
i'a,j : 2'a,6
i'b,j : 4'b,2
i'c,j : 5'c,3
i'd,j : 6'd,4
i'[,j : 1'[,5
i'],j : 3'],1
```

(b)

Şekil 3. (a) Bağlam ve (b) Uygunluk Tablosu

5 GKA Kapsama Kriterleri

Bu bölümde kapsama kriterlerinin tanımları verilecektir. Bu kriterler arasındaki ilişkiler (*subsumption relation*) **Şekil 4**'te görülmektedir.



Şekil 4. Kriterler arası kapsama ilişkisi

Dİ'de kullanılan semboller olayları adlandırmaktadır. Dolayısı ile **Şekil 3**'de gösterilen Bağlam ve Uygunluk Tabloları, bu olayların birbirlerine olan bağımlılıklarını ifade etmektedirler. Daha da ötesi, bu bilgileri, SDM'nin durumları ile zenginleştirmektedirler. Bu da, sembollerin semantiğine göre kullanıcı için önemli bilgileri içermektedir. Bu amaçla tablolardan aşağıdaki GKA kapsama kriterlerini üretiyoruz. Bu kriterler, kombine edilerek ya da kullanılan ilişkilerin geçişli kapamaları (transitive closure) alınarak daha da kuvvetlendirilebilirler.

Burada tek yönlü ok, kriterler arasındaki kapsama ilişkisini açıklamaktadır. Örneğin Bağlam Kriteri hem Sol hem de Sağ kriteri kapsamaktadır. Uygunluk kriteri ise Sağ ve Sol arasında uygunluk testi için kullanıldığı için çift yönlü ok ile gösterilmiştir.

Tanım 1: Bir Bağlam Kapsama Kriteri (*Context Coverage Criterium*), üretilen test takımının, tüm durumları Bağlam Tablosu tarafından içerilmesi olarak tanımlanır. Bu formel olarak;

$$\forall t_n \in T \Rightarrow\ t_n = c\ \in\ C_i^j$$

Tanım 2: Bir Sol Bağlam Kapsama Kriteri (*Left Context Coverage Criter*ium), üretilen test takımının, tüm durumlarının Bağlam Tablosu'nun sol endekslenmiş sembolleri tarafından içerilmesi olarak tanımlanır. Bu da formel olarak;

$$\forall t_n \in T \Rightarrow\ t_n = c\ \in\ \underbrace{C}_{left}$$

Sağ Bağlam Kapsama Kriteri (*Right Context Coverage Criterium*), Sol Bağlam Kapsama Kriteri'ne benzer şekilde tanımlanır.

Tanım 3: Bir İleri Bağlam Kapsama Kriteri (*Forward Context Coverage Criterium*), üretilen test takımının, tüm durumlarının İleri Bağlam Tablosu tarafından içerilmesi olarak tanımlanır. Buda formel olarak;

$$\forall t_n \in T \Rightarrow\ t_n = c\ \in\ C^j$$

Geri Bağlam Kapsama Kriteri (*Backward Context Coverage Criterium*), İleri Bağlam Kapsama Kriteri'ne benzer şekilde tanımlanır.

Tanım 4: Bir Uygunluk Kapsama Kriteri (*Compatibility Coverage Criterium*), üretilen test takımının, tüm durumlarının Uygunluk Tablosu'nda verilen endekslere tam uyumlu olması olarak tanımlanır. Buda formel olarak;

$$\forall t_n \in T \Rightarrow\ t_n = u\ \in\ U$$

Yukarıda tanımı verilen kapsama kriterleri bir Dİ'nin analizi sonucunda elde edilen tablolar ile tespit edilmektedir. Bahsedildiği gibi SDM'den Dİ'ye dönüşümlerde durum bilgisi kaybolmaktadır. Kaybolan durum bilgisi endeksleme işlemi sayesinde tekrardan oluşturulmaktadır. Bu bağlamda öne sürülen kapsama kriterleri ile literatürde sıkça kullanılan durum kapsama (state coverage), geçiş kapsama (transition coverage) vb. kriterleri arasında bir ilişki vardır. Ancak mevcut çalışma bu ilişkiden ziyade öne sürülen kapsama kriterlerinin tanım ve kullanımına yöneliktir.

6 Uygulama

6.1 Test Takımı Üretimi

Test takımları bir önceki bölümde verilen kapsama kriterlerini sağlayacak şekilde bağlam tablosundan elde edilir. **Şekil 3**.(a)'da verilen bağlam tablosu test takımı üretimi için kullanılacaktır. Örnek olarak sol bağlam ve ileri bağlam kapsama kriterleri seçilecek olursa, test takımları t_1 ve t_2 **Şekil 3**.(a)'nın sol tarafında bulunan ileri bağlam ve sürekli sola doğru (x'R) semboller seçilerek elde edilebilir.

$$t_1 = [^1\ a^2\ b^4\ ]^3 \tag{9}$$

$$t_2 = [^1\ a^2\ b^4\ c^5\ d^6\ ]^3 \tag{10}$$

(9) ve (10)'da verilen test takımları sol bağlam ve ileri bağlam kriterleri dikkate alınarak üretildiği için bu kriterleri sağlamaktadır. Elde edilen bu test takımlarının diğer kapsama kriterlerini sağlayıp sağlamadığı şu şekilde test edilir. Öncelikle (9) ve (10)'un **Şekil 3**.(b)'de verilen uygunluk tablosu ile uyumluluğuna bakılır. Yani test takımlarının endeksleri uygunluk tablosunda verilen endeksler ile aynı mıdır diye kontrol edilir. Böylece, t_1 ve t_2 için tüm semboller uygunluk tablosu ile uyumlu olduğu görülebilir. Bu durumda üretilen her iki test takımı da uygunluk kapsama kriterini sağlamaktadır.

t_1 ve t_2'nin alt endeksleri uygunluk tablosu aracılığı ile bulunur. Bu durumda (11) ve (12) elde edilir.

$$t_1 = [^1_5 a^2_6\ b^4_2\ ]^3_1 \tag{11}$$

$$t_2 = [^1_5\ a^2_6\ b^4_2\ c^5_3\ d^6_4\ ]^3_1 \tag{12}$$

Ardından elde edilen alt endekslerin geri kapsama kriterini sağladıkları **Şekil 3**.(a)'daki tablonun sağ tarafındaki tablo ile uyumlu olduğu çıkarılabilir. Böylece, t_1 ve t_2 tüm tablolar ile uyumlu olduğu için tüm kapsama kriterlerini sağlamaktadır ve bu durumda bağlam kapsama kriterini de sağlar.

6.2 Test Takımı Üretimini Durdurma ve Test Masrafları

Test takımı üretimini durdurmak (*Test Termination Criteria*) için sağ bağlam kriterini kapsayacak şekilde üretim yapılıyorsa ve tabloya uygunsa "]" sembolü seçildiğinde, bu üretimin durduğunu ifade etmektedir. Aksine sol bağlam kriterini kapsayacak şekilde üretim yapılıyorsa ve eğer tabloya uygunsa "[" seçildiğinde, bu üretimin durduğunu ifade etmektedir. Sol ve sağ bağlam kriterlerinin başlangıç sembolleri sırasıyla "[" ve "]"dır. Dolayısıyla test takımları bu sembollerle başlayıp yine bu sembollerle bitmektedir.

Test masrafları ise üretilen test takımlarının uzunluğu (*Test Length*) ve sayısı ile orantılıdır. Kapsama kriterleri ile maksimum uzunlukta üretilebilecek test takım uzunlukları arasındaki ilişki aşağıda **Tablo 2**'de verilmektedir.

Kapsama Kriterleri	Maksimum Test Takımı Uzunluğu
Sol, İleri	$\left\lvert \underbrace{C^j}_{left} \right\rvert$
Sağ, İleri	$\left\lvert \underbrace{C^j}_{right} \right\rvert$
Sol, Geri	$\left\lvert \underbrace{C_i}_{left} \right\rvert$
Sağ, Geri	$\left\lvert \underbrace{C_i}_{right} \right\rvert$
Uygunluk	$\lvert U \rvert$
Bağlam	$\max\{\left\lvert \underbrace{C^j}_{left} \right\rvert, \left\lvert \underbrace{C^j}_{right} \right\rvert, \left\lvert \underbrace{C_i}_{left} \right\rvert, \left\lvert \underbrace{C_i}_{right} \right\rvert\}$

Tablo 2. Kapsama Kriterleri ve Test Takımları Uzunluğu

Üretilebilecek test takımları sayısı ise yine **Tablo 2**'te verilen kardinalitelere bağlıdır. Kardinalite hesabı yaparken Dİ'yi oluşturan semboller ve daha da belirleyici olarak bu semboller arasındaki operatörler gözönünde bulundurulur. Bu operatörler Bölüm 3'de verildiği üzere birleşim "+", bitiştirme "" ve yineleme "*"'dir. Bağlam tablosu girdisi olarak operatörler ile sembol sayısı ilişkisi aşağıdaki gibi özetlenir;

- $[(a_1a_2a_3 \dots a_n)] \rightarrow \max\left\{\left\lvert \underbrace{C^j}_{left} \right\rvert, \left\lvert \underbrace{C^j}_{right} \right\rvert, \left\lvert \underbrace{C_i}_{left} \right\rvert, \left\lvert \underbrace{C_i}_{right} \right\rvert\right\} = 2n - 1$
- $[(a_1a_2a_3 \dots a_n)^*] \rightarrow \max\left\{\left\lvert \underbrace{C^j}_{left} \right\rvert, \left\lvert \underbrace{C^j}_{right} \right\rvert, \left\lvert \underbrace{C_i}_{left} \right\rvert, \left\lvert \underbrace{C_i}_{right} \right\rvert\right\} = 2n - 1$
- $[(a_1 + a_2 + a_3 + \dots + a_n)] \rightarrow \max\left\{\left\lvert \underbrace{C^j}_{left} \right\rvert, \left\lvert \underbrace{C^j}_{right} \right\rvert, \left\lvert \underbrace{C_i}_{left} \right\rvert, \left\lvert \underbrace{C_i}_{right} \right\rvert\right\} = 2n$
- $[(a_1 + a_2 + a_3 + \dots + a_n)^*] \rightarrow \max\left\{\left\lvert \underbrace{C^j}_{left} \right\rvert, \left\lvert \underbrace{C^j}_{right} \right\rvert, \left\lvert \underbrace{C_i}_{left} \right\rvert, \left\lvert \underbrace{C_i}_{right} \right\rvert\right\} = n^2 - n + 1$

Üstte verildiği gibi test takımlarının üretildiği tablo eleman sayısı birleşim "+" ve kapama "*" operatörlerinin birlikte verildiği son durumda n sembolden oluşan bir Dİ için $n^2 - n + 1$ yani karesel bir artış göstermektedir. Diğer durumlarda ise artış doğrusaldır. Yani en kötü durumda tablo sembol sayısı $n^2 - n + 1$ olmaktadır. Böylece üretilebilecek maksimum test takımı uzunluğu da $n^2 - n + 1$'dir.

Unutulmaması gerekir ki mevcut test kapsama kriterleri, çoğunlukla yalnızca olay bazlı ya da yalnızca durum bazlıdır. Buna rağmen test kapsama kriterleri kübiksel bir artış gösteren sayıda eleman üretmektedir [2].

7 Sonuçlar

Bu çalışmada GKA'lar için bir model olarak Dİ'ler verilmiştir ve bu model ile mevcut modeller arasında dönüşümler açıklanmıştır. Ayrıca bu model aracılığı ile test takımları üretimine olanak sağlayan ve literatürde mevcut olmayan kapsama kriterleri tanımlanmıştır. Açıklanan kavramlar ve yöntemler bir örnek üzerinde gerçekleştirilmiş ve elde edilen sonuçlar irdelenmiştir.

Önerilen yaklaşımın özelliği, test kapsama kriterleri üretirken, mevcut kuramsal test yöntemlerinin ötesinde, incelenen sistemin yalnızca olaysal değil, aynı zamanda durumsal özelliklerini de göz önünde tutmasıdır. Test süreci masrafında önemli rol oynayan test uzunluğu ise doğrusal, en olumsuz durumda karesel artmaktadır ki bu da mevcut yöntemlere göre büyük bir avantajdır.

Bundan sonra yapılacak çalışmalar, ileri sürülen kapsama kriterlerinin GKA'ların testi için öne sürülen diğer çalışmalardaki kapsama kriterleri ile ilişkisinin yanı sıra diğer avantajları ve dezavantajlarını ortaya koymak olarak planlanmaktadır.Örneğin hata bulma kabiliyeti (bulunan hataların sayısı, bulunan hataların arasındaki zaman mesafeleri v.b.) ve masrafları (hata bulma zamanı, tüm test sayısı ve uzunlugu v.b.) gelmektedir.

Ayrıca modelleme ve test işleminin bütünsel (*holistic*) olarak yapılması, yani pozitif (hatalı olmayan modeller ile) ve negatif test (hatalı olan modeller ile) uygulanması diğer bir ileriki çalışmadır.

Test üretim sürecinin otomatik hale getirilmesini mümkün kılacak bir aracın geliştirilmesine başlanmıştır.

Teşekkür
Yazarlar çok değerli tavsiyelerinden ve yardımlarından dolayı Yard.Doç.Dr.Mutlu Beyazıt'a , Yard.Doç.Dr.Nida Gökçe'ye ve sempozyumun anonim hakemlerine teşekkür ederler.

Kaynakça

1. Aho, Alfred V., and Jeffrey D. Ullman. *Foundations of computer science*. Computer Science Press, 1992.
2. Belli, Fevzi, and Christof J. Budnik, "Test minimization for human-computer interaction", Appl. Intell. 26(2), (2007) 161-174.
3. Belli, Fevzi, and K-E. Grosspietsch. "Specification of fault-tolerant system issues by predicate/transition nets and regular expressions-approach and case study." *IEEE Transactions on software engineering* 17.6 (1991): 513-526.
4. Belli, Fevzi, Christof J. Budnik, Lee White, Event-based modelling, analysis and testing of user interactions: approach and case study. Softw. Test., Verif. Reliab. 16(1), (2006) 3-32.
5. Belli, Fevzi, Extending Regular Languages for Self-Detection and Self-Correction of Syntactical Faults (PhD Thesis in German; Technical Univ. Berlin), Bericht 119 der Gesellschaft für Mathematik und Datenverarbeitung, Oldenburg Verlag, 1978.

6. Belli, Fevzi, Mutlu Beyazit, and Atif Memon. "Testing is an event-centric activity." *Software Security and Reliability Companion (SERE-C), 2012 IEEE Sixth International Conference on*. IEEE, 2012.
7. Belli, Fevzi, Mutlu Beyazit, and Nevin Güler. "Event-Oriented, Model-Based GUI Testing and Reliability Assessment—Approach and Case Study." *Advances in Computers* 85 (2012): 277-326.
8. Belli, Fevzi, N. Nissanke, Ch. J. Budnik, A. Mathur, "Test Generation Using Event Sequence Graphs", Technical Report, University of Paderborn, 2005.
9. Belli, Fevzi. "Finite state testing and analysis of graphical user interfaces." *Software Reliability Engineering, 2001. ISSRE 2001. Proceedings. 12th International Symposium on*. IEEE, 2001.
10. Chow, Tsun S. "Testing software design modeled by finite-state machines." *IEEE transactions on software engineering* 3 (1978): 178-187.
11. David, Rene, and Pascale Thevenod-Fosse. "Minimal detecting transition sequences: application to random testing." *IEEE Transactions on Computers*29.6 (1980): 514-518.
12. Hopcroft, John E, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Harlow: Pearson Addison-Wesley, 2014.
13. Liu, Pan, and Huaikou Miao. "Theory of test modeling based on regular expressions." *International Workshop on Structured Object-Oriented Formal Language and Method*.", Springer International Publishing, 2013.
14. Memon, Atif M. "An event-flow model of GUI-based applications for testing." *Software Testing Verification and Reliability* 17.3 (2007): 137-158.
15. Memon, Atif M., Mary Lou Soffa, and Martha E. Pollack. "Coverage criteria for GUI testing." *ACM SIGSOFT Software Engineering Notes* 26.5 (2001): 256-267.
16. Myhill, J., “Finite Automata and the Representation of Events”, Wright Air Devel. Command, TR 57-624, pp. 112-137 (1957).
17. Ravi, Srivaths, Ganesh Lakshminarayana, and Niraj K. Jha. "TAO: Regular expression-based register-transfer level testability analysis and optimization." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 9.6 (2001): 824-832.
18. Shaw, Alan C. "Software specification languages based on regular expressions." *Software Development Tools*. Springer Berlin Heidelberg, 1980. 148-175.