İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

AYRIK OLAY SİSTEMLERİ İÇİN OPTİMAL DENETİMSEL GÖZETLEYİCİ TASARIMI

DOKTORA TEZİ

Y. Müh. Özgür Turay KAYMAKÇI

Anabilim Dalı: ELEKTRİK MÜHENDİSLİĞİ

Programı: KONTROL VE OTOMASYON MÜHENDİSLİĞİ

ARALIK 2007

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

AYRIK OLAY SİSTEMLERİ İÇİN OPTİMAL DENETİMSEL GÖZETLEYİCİ TASARIMI

DOKTORA TEZİ
Y. Müh. Özgür Turay Kaymakçı
(504002152)

Tezin Enstitüye Verildiği Tarih : 14 Eylül 2007
Tezin Savunulduğu Tarih : 28 Aralık 2007

Tez Danışmanı :	**Doç.Dr. Salman KURTULAN**	
Diğer Jüri Üyeleri	**Prof. Dr. Galip CANSEVER**	**(Y.T.Ü.)**
	Prof. Dr. A. Çoşkun SÖNMEZ	**(Y.T.Ü.)**
	Doç. Dr. Şeniz ERTUĞRUL	
	Doç. Dr. M. Turan SÖYLEMEZ	

Aralık 2007

ÖNSÖZ

Yorucu bir çalışmanın ardından doktora tezimi tamamlamış bulunuyorum. Bu çalışmanın her aşamasında yanımda olan, bilgi, emek ve desteğini esirgemeyen, kontrol ve otomasyon mühendisliğini gönülden sevmemde çok büyük bir payı olan, Endüstriyel Otomasyon Laboratuarının olanaklarından faydalanmama izin veren sevgili danışman hocam Doç. Dr. Salman KURTULAN'a en içten teşekkürlerimi sunarım. 2 yıl boyunca her hafta aynı gün ve saatte yapılan ayrık olay sistemleri çalışma grubu toplantılarında yorumları ile ufkumu genişleten, bilgisini hiçbir zaman esirgemeyen sevgili hocam Leyla GÖREN'e, hazırlamış olduğum makaleleri inceleme zahmetinden kaçınmayan, fonksiyonel analiz ile ilgili engin bilgisini sonuna kadar paylaşan değerli hocam Yrd. Doç. Dr. Neslihan Serap ŞENGÖR'e, uzun ve yorucu ayrık olay sistemleri toplantılarında benimle beraber her hafta ayrık olay sistemleri üzerine çalışan, konular ile ilgili yaptığı yorumlarla kavramları algılamama katkıda bulunan çalışma arkadaşım Yük. Müh. İbrahim Tolga HASDEMİR'e teşekkürlerimi sunuyorum. Ayrıca ayrık olay sistemleri çalışma grubunun diğer üyelerinden olan Yük. Müh. Yaprak YALÇIN'a ve Yük. Müh. Mustafa KÖSEM'e de buradan teşekkür ederim.

Gösterdikleri anlayış ve destek ile tüm akademik hayatım boyunca yanımda olmuş, bulunduğum noktaya gelmemde büyük emekleri olan sevgili annem Ayşe KAYMAKÇI, rahmetli babam Turan KAYMAKÇI ve sevgili kardeşim Özgün Burak KAYMAKÇI'ya teşekkürlerimi sunuyorum.

Sevgili eşim Zeynep Gülin KAYMAKÇI'ya ondan esirgediğim zaman için özürlerimi sunar, gösterdiği sabır, anlayış, destek ve özveri için teşekkür ederim.

Gönülden sevgisi, desteği ve bana olan inancı olmasaydı karşılaştığım zorlukları aşamazdım. Bulunduğum noktaya gelmemde çok büyük katkısı olan, zor anlarımda hep yanımda olmuş, dert ortağım, sevgili dostum Kâzım ÇAKMAK'a sonsuz teşekkürlerimi sunarım.

Eylül 2007 Özgür Turay KAYMAKÇI

İÇİNDEKİLER

KISALTMALAR **vi**
TABLO LİSTESİ **vii**
ŞEKİL LİSTESİ **viii**
SEMBOL LİSTESİ **x**
ÖZET **xii**
SUMMARY **xvii**

1. GİRİŞ **1**
1.1 Literatür Araştırması 4
2. AYRIK OLAY SİSTEMLERİ **8**
2.1 Giriş 8
2.2 Diller ve Otomatlar ile İlgili Temel Kavramlar 9
2.2.1 Diller 9
2.2.2 Diller Üzerine Tanımlanan İşlemler 9
2.2.2.1 Birbirine Bağlama 10
2.2.2.2 Önek Kapanışı 10
2.2.2.3 Kleene-Kapanışı 10
2.2.2.4 Örnek 10
2.2.3 Otomatlar 10
2.2.3.1 Deterministik Sonlu Durumlu Otomat 11
2.2.3.2 Üretilen Dil 12
2.2.3.3 İşaretli Dil 12
2.2.3.4 Kilitlenme 12
2.2.3.5 Otomatların Eşitliği 13
2.2.4 Otomatlar Üzerine Tanımlanan İşlemler 13
2.2.4.1 Ulaşılabilir Kısım 14
2.2.4.2 İşaretli-Ulaşılabilir Kısım 14
2.2.4.3 Budanmış Otomat 15
2.2.4.4 Tümleyen Otomat 15
2.2.4.5 Çarpım İşlemi 16
2.2.4.6 Paralel Birleşim İşlemi 16
2.2.5 Düzgün Diller 17
2.2.5.1 Düzgün Dillerin Özellikleri 18
2.2.5.2 Düzenli İfadeler 18
2.2.5.3 Kleene Kuramı 18
2.3 Denetimsel Kontrol Kuramı 19
2.3.1 Neden Denetimsel Gözetleyiciye İhtiyaç Vardır 19
2.3.2 Denetimsel Gözetleyici ile Geri Besleme 20
2.3.2.1 Kabul Edilebilirlik Koşulu 21
2.3.2.2 S/G Tarafından Üretilen ve İşaretlenen Diller 21
2.3.3 Kontrol edilebilirlik Kuramı 23
2.3.3.1 Özellik 1 25
2.3.3.2 Özellik 2 25

2.3.3.3 Özellik 3 25
2.3.3.4 Özellik 4 26
2.3.4 En Büyük Kontrol Edilebilir Alt Dil 26
2.3.4.1 Örnek 27
2.3.4.2 ↑C İşleminin Özellikleri 28
2.3.4.3 ↑C İşleminin Sonucunda Elde Edilen Dilin Düzgünlüğü 28
2.3.5 En Küçük Kontrol Edilebilir Önek Kapalı Üst Dil 29
2.3.5.1 ↓C işleminin Özellikleri 30
2.3.5.2 Örnek 30
2.3.5.3 ↓C İşleminin Sonucunda Elde Edilen Dilin Düzgünlüğü 31
2.3.6 Kilitlenmesiz Kontrol Edilebilirlik Kuramı 31
2.4 Denetimsel Kontrol Kuramı ile İlgili Temel Problemler 32
2.4.1 Temel Denetimsel Gözetleyici Problemi 33
2.4.2 Temel Denetimsel Gözetleyici Probleminin Eşleniği 33
2.4.3 Kilitlenmesiz Temel Denetimsel Gözetleyici Problemi 34
2.4.4 Örnek 34
2.4.5 Temel Kilitlenebilir Denetimsel Gözetleyici Problemi 36
2.4.5.1 Önerme 38
2.4.5.2 Kilitlenebilir Denetimsel Gözetleyici Problemi İçin Önerilen Çözümler 39
2.4.5.3 Lemma 40
2.4.5.4 Kilitlenmeyi Azaltmak 40
2.4.5.5 Lemma 41
2.4.5.6 Başarımı Arttırmak 42
2.4.5.7 Örnek 43

3. OPTİMAL KİLİTLENEBİLİR DENETİMSEL GÖZETLEYİCİ TASARIM PROBLEMİ 45
3.1 Giriş 45
3.1.1 Örnek 46
3.1.2 Ön Koşullar 47
3.2 Tanımlar 47
3.2.1 Olay Ederi 47
3.2.2 Kelime Ederi 48
3.2.3 Dilin Ederi 49
3.2.3.1 Özellik 49
3.2.3.2 Özellik 49
3.2.3.3 Özellik 50
3.2.4 Mesafe Fonksiyonu 50
3.2.5 Önerme 50
3.2.6 Metrik Uzay 51
3.2.7 Başarısızlık ve Kilitlenme Ölçüleri 51
3.2.7.1 Örnek 52
3.2.8 Performans Ölçüsü 52
3.2.9 Optimal Kilitlenebilir Denetimsel Gözetleyici Problemi 53
3.2.10 Kayıp Faktörü 54
3.3 Optimizasyon Algoritmasına ait Temel Kavramlar 54
3.3.1 Özellik 54
3.3.2 Özellik. 55
3.3.3 Özellik 56

3.3.4 T İşlemi 57
3.3.5 Önerme 58
3.3.6 Önerme 60
3.3.7 Teorem 62
3.4.1 Belirtme 68
3.5 Uygulama 69

4. AYRIK OLAY SİSTEMLERİ İÇİN MATLAB TABANLI BİR ANALİZ PROGRAMI 76
4.1 Giriş 76
4.2 Ayrık Olay Sistemleri için Geliştirilmiş Paket Programlar 76
4.3 Matlab Tabanlı Analiz Programı 77
4.3.1 Otomatın Oluşturulması 78
4.3.1.1 Örnek 78
4.3.2 Geliştirilen Fonksiyonlar 79
4.3.2.1 Ulaşılabilir Kısmı Fonksiyonu 79
4.3.2.2 İşaretli Ulaşılabilir Fonksiyonu 80
4.3.2.3 Budama Fonksiyonu 81
4.3.2.4 Otomat Tümleyeni Fonksiyonu 81
4.3.2.5 Otomat Birleşimi Fonksiyonu 82
4.3.2.6 Otomat Çarpımı Fonksiyonu 83
4.3.2.7 Kelimenin Önek Kapanışı Fonksiyonu 84
4.3.2.8 Dilin Önek Kapanışı Fonksiyonu 85
4.3.2.9 Dilin Önek Kapanışının Denetimi Fonksiyonu 85
4.3.2.10 Üretilen Kelimeler Fonksiyonu 86
4.3.2.11 İşaretlenen Kelimeler Fonksiyonu 87
4.3.2.12 Üretilen Dil Fonksiyonu 87
4.3.2.13 İşaretlenen Dil Fonksiyonu 88
4.3.2.14 Eşit Otomatlar Fonksiyonu 88
4.3.2.15 Cansız Kilitlenmeler Fonksiyonu 89
4.3.2.16 Otomat İsimlendirme Fonksiyonu 90
4.3.2.17 Kontrol Edilebilirlik Fonksiyonu 90
4.3.2.18 En Büyük Kontrol Edilebilir Alt Dil Fonksiyonu 91
4.3.2.19 En Küçük Kontrol Edilebilir Önek Kapalı Üst Dil Fonksiyonu 92
4.3.3 Örnekler 94
4.3.4 Kilitlenebilir Denetimsel Gözetleyici Problemini Çözmek için Geliştirilmiş Olan Fonksiyonlar 97
4.3.4.1 Eder Fonksiyonu 97
4.3.4.2 Performans Fonksiyonu 98
4.3.4.3 T İşlemi Fonksiyonu 99
4.3.4.4 Çıkartılan Kelimeler Fonksiyonu 100
4.3.4.5 Kayıp Faktörü Fonksiyonu 101
4.3.4.6 Örnek 102

5. SONUÇLAR VE TARTIŞMA 104

KAYNAKLAR 107

ÖZGEÇMİŞ 118

KISALTMALAR

PI : Oransal Entegral Kontrolör
DSDO : Deterministik Sonlu Durumlu Otomat
EAKÇ : En Az Kısıtlamalı Kilitlenmesiz Çözüm
TBÇ : Tam Başarımlı Çözüm
EAKKÇ : En Az Kısıtlamalı Kilitlenmesiz Çözüm
EAKDKÇ : En Az Kısıtlamalı Dış Kilitlenmesiz Çözüm
EAKİKÇ : En Az Kısıtlamalı İç Kilitlenmesiz Çözüm
OKDGP : Optimal Kilitlenebilir Denetimsel Gözetleyici Problemi
KDGP : Kilitlenebilir Denetimsel Gözetleyici Problemi

TABLO LİSTESİ

Sayfa No

Tablo 3.1 Algoritma..... 67
Tablo 3.2 Veritabanı sisteminin ürettiği olay ederleri…….. 72

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1 : G_1 otomatı 11
Şekil 2.2 : G_2 otomatı..... 13
Şekil 2.3 : G_3 otomatı..... 13
Şekil 2.4 : $Ac(G_2)$, $Coac(G_2)$, $Trim(G_2)$ 15
Şekil 2.5 : $G_t = Trim(G_2)$ ve G_t^{comp} 16
Şekil 2.6 : G_4 ve G_5 otomatları..... 17
Şekil 2.7 : $G_4 \times G_5$ ve $G_4 \| G_5$ otomatları..... 17
Şekil 2.8 : Üretim bandının prensip şeması..... 19
Şekil 2.9 : Denetimsel gözetleyici ile sistemin prensip çalışma şeması..... 20
Şekil 2.10 : H_{spec1} hatırlayıcısı otomatı..... 22
Şekil 2.11 : En büyük kontrol edilebilir alt dil..... 27
Şekil 2.12 : Sırasıyla üretim sistemini ve beklentileri ifade eden G_6 otomatı ve H_{spec1} hatırlayıcısı..... 27
Şekil 2.13 : En küçük kontrol edilebilir üst dil..... 29
Şekil 2.14 : $L_m(G)$ kapalı olmayan bir dil örneği..... 32
Şekil 2.15 : $L(S/G) = \overline{L_{am}^{\uparrow C}}$ otomat modeli..... 35
Şekil 2.16 : $L(S/G) = L_{am}^{\downarrow C}$ dilini üreten otomat..... 35
Şekil 2.17 : Temel kilitlenebilir denetimsel gözetleyici problemi..... 37
Şekil 2.18 : $L(S/G) \in L_{cand}$ için iç ve dış kilitlenme..... 38
Şekil 2.19 : $A_{bm}(L)$ işlemi sonunda elde edilen dil..... 41
Şekil 2.20 : $A_{sm}(L) = L \cup K_{\max}$ işleminden sonra elde edilen dil..... 43
Şekil 2.21 : G_7 otomatı..... 43
Şekil 3.1 : G_8 otomatı..... 46
Şekil 3.2 : T_1 ve T_2 geçişlerine ait G_9 ve G_{10} otomatları..... 70
Şekil 3.3 : Veritabanı sistemine ilişkin G_{11} otomatı..... 71
Şekil 3.4 : H otomatı..... 71
Şekil 4.1 : G otomatı..... 78
Şekil 4.2 : Ulaşılabilir fonksiyonuna ilişkin algoritma..... 80
Şekil 4.3 : İşaretli ulaşılabilir fonksiyonuna ilişkin algoritma..... 81
Şekil 4.4 : Budama fonksiyonuna ilişkin algoritma..... 81
Şekil 4.5 : Otomat tümleyeni fonksiyonuna ilişkin algoritma..... 82
Şekil 4.6 : Otomat birlişimi fonksiyonuna ilişkin algoritma..... 83

Şekil 4.7 : Otomat çarpımı fonksiyonuna ilişkin algoritma..... 84
Şekil 4.8 : Kelimenin önek kapanışı fonksiyonuna ilişkin algoritma..... 85
Şekil 4.9 : Dilin önek kapanışı fonksiyonuna ilişkin algoritma……..... 85
Şekil 4.10 : Dilin önek kapanışının denetimi fonksiyonuna ilişkin algoritma. 86
Şekil 4.11 : Üretilen kelimeler fonksiyonuna ilişkin algoritma..... 86
Şekil 4.12 : İşaretlenen kelimeler fonksiyonuna ilişkin algoritma..... 87
Şekil 4.13 : Eşit otomatlar fonksiyonuna ilişkin algoritma..... 89
Şekil 4.14 : Cansız kilitlenmeler fonksiyonuna ilişkin algoritma..... 89
Şekil 4.15 : Kontrol edilebilirlik fonksiyonuna ilişkin algoritma..... 91
Şekil 4.16 : En büyük kontrol edilebilir alt dil fonksiyonuna ilişkin algoritma…… 92
Şekil 4.17 : En küçük kontrol edilebilir önek kapalı üst dil fonksiyonuna ilişkin algoritma……..... 93
Şekil 4.18 : acG, $coacG$ ve tG otomatları..... 95
Şekil 4.19 : Pompa, vana ve denetimsel gözetleyicilere ait otomatlar..... 96
Şekil 4.20 : Eder fonksiyonuna ilişkin algoritma..... 98
Şekil 4.21 : Performans fonksiyonuna ait algoritma..... 99
Şekil 4.22 : T işlemi fonksiyonuna ilişkin algoritma..... 100
Şekil 4.23 : Çıkartılan kelimeler fonksiyonuna ilişkin algoritma..... 101
Şekil 4.24 : Kayıp faktörü fonksiyonuna ait algoritma..... 101
Şekil 4.25 : Sırasıyla G, S ve G_a otomatları..... 102

SEMBOL LİSTESİ

Σ : Olay kümesi
Σ^* : Olay kümesinin kleene kapanışı
Σ_c : Kontrol edilen olaylar kümesi
Σ_{uc} : Kontrol edilemeyen olaylar kümesi
e_j : Olay
ε : Boş olay
L_i : Olay kümesi üzerine tanımlı bir dil
s : Kelime
G : Otomat
X : Otomata ait durum kümesi
f : Otomata ait durum geçiş fonksiyonu
x_0 : Otomatın başlangıç durumu
X_m : Otomata ait işaretli durum kümesi
$L(G)$: G otomatının ürettiği dil
$L_m(G)$: G otomatının işaretlediği dil
S : Denetimsel Gözetleyici
$L(S/G)$: G otomatının S denetimsel gözetleyicisi ile birlikte ürettiği dil
$L_m(S/G)$: G otomatının S denetimsel gözetleyicisi ile birlikte işaretlediği dil
L^* : L dilinin kleene kapanışı
$\overline{L}$: L dilinin önek kapanışı
$Ac(G)$: G otomatının ulaşılabilir kısmı
$CoAc(G)$: G otomatının işaretli-ulaşılabilir kısmı
$Trim(G)$: G otomatının bundamış kısmı
L_a : Kabul edilebilir dil
L_{am} : Kabul edilebilir işaretli dil
L_r : En az beklentileri içeren dil
L_{rm} : En az beklentileri içeren işaretli dil
K : Beklenilen davranışları ifade eden dil
H : Hatırlayıcı
$K^{\uparrow C}$: K dilinin en büyük kontrol edilebilir alt dili
$K^{\downarrow C}$: K dilinin en küçük kontrol edilebilir önek kapalı üst dili
L_{cand} : OKDGP için tüm çözümleri içeren dil ailesi
c_e : Olay ederi
c_s : Kelime ederi

$\beta(L)$ **:** L dilinin ederi

$\widehat{SM^C}$ **:** Başarısızlık ölçüsü

$\widehat{BM}$ **:** Kilitlenme ölçüsü

arg : Dile ait argümanları veren fonksiyon

$\hat{J}$ **:** Performans ölçüsü

F **:** Kayıp faktörü

T **:** T işlemi

$BM(L)\{i\}$ **:** $BM(L)$ kümesinin i. elemanı

TNM **:** "0" karakteri

AYRIK OLAY SİSTEMLERİ İÇİN OPTİMAL DENETİMSEL GÖZETLEYİCİ TASARIMI

ÖZET

Ayrık olaylı sistemler dinamik bir sistem olmasına rağmen, zamanla değişen dinamik sistemlerden farklı olarak adi diferansiyel denklemlerle modellenemez. Bu sistemler zamanda düzensiz aralıklarla oluşan olaylarla zamanda ilerlerler. Genellikle deterministik sonlu durumlu otomatlarla ifade edilen düzenli diller ile modellenirler. Bir ayrık olay sistemin davranışı incelendiğinde, sahip olduğu davranışının genellikle yetersiz olduğu görülür ve bir kontrol hareketi ile "düzenlenmesi" gerekir. Denetimsel gözetleyici bize sistem üzerinde olabildiğince çok değişikliği geliştirme olasılığı sağladığından, Ramadge ve Wonham tarafından temelleri atılan denetimsel gözetleyici kuramı güçlü bir araç olarak karşımıza çıkmaktadır. Fakat kilitlenmesiz ve kontrol edilebilir olma kısıtlarına göre geliştirilmiş olan bir denetimsel gözetleyici istenilen tüm davranışlar her zaman işaretleyemeyebilir. Bu bağlamda kilitlenme önemli bir konu olarak karşımıza çıkmaktadır. Endüstriyel problemlere bakacak olursak, kilitlenmeli denetimsel gözetleyici probleminin kilitlenmesiz denetimsel gözetleyici problemine göre daha genel olduğu açığa çıkmaktadır ve ayrıca bazı uygulamalarda kilitlenmeli denetimsel gözetleyiciler kilitlenmesiz olanlara nazaran tercih edilmektedirler. Eş zamanlı veri yönetimi sistemi bu duruma güzel bir örnek oluşturmaktadır. Birçok popüler protokol kilitlenme bulma düzenleri ile birleştirildiklerinden, kilitlenmeyi engellemek ve kilitlenmeden kaçınmak pratik değildir. Benzer durumlarda kilitlenmesiz çözümler çok tutucu sonuçlara neden olabilmektedir. Bu sebepten sistemin davranışını kısıtlamamak için bazı olası kilitlenmelerin oluşma riskine girilebilir. Açıktır ki sistemin performansında ciddi bir artışa neden olacaksa alınan risk kabul edilebilirdir. Bu yüzden sistemdeki kilitlenmelerin neden olacağı kazançlar ve kayıplar dâhil edilerek sistemin performansı incelenmelidir.

Bir ayrık olay sistemini izin verilen dil ile beraber incelediğimizde, sistem beraberinde çok fazla kilitlenmeye sahip olduğu görülebilir. Bazen sınırlayıcı davranışı yüzünden yetersiz olmasına rağmen genellikle en az kısıtlamalı kilitlenmesiz çözümü denetimsel gözetleyici olarak seçeriz. Kilitlenmeyle sonuçlanan tüm kontrol edilemeyen olayları engellemesi yüzünden en az kısıtlamalı kilitlenmesiz çözüm tutucu bir sonuca neden olabilir. Sonuçta denetimsel gözetleyici istenilen işaretli dilin ufak bir kısmını üretebilir. Bu tarz bir strateji sistemin davranışını ciddi oranda kısıtlayabilir. Diğer bir taraftan tam başarımlı çözüm bütün işaretli kelimeleri üretmektedir, fakat bunun bir karşılığı olarak çok fazla kilitlenmeyi de denetim altındaki dile ekler. Bu yüzden çoğu ayrık olay sistemi probleminde bu iki çözüm de kabul edilebilir değildir. Burada denetimsel gözetleyicinin başarısı üretilen kabul edilebilir işaretli dil ile ilişkilidir. O zaman kilitlenmesiz olma koşulunu gevşeterek, kilitlenebilir denetimsel gözetleyicilerin sentezi göz önüne alınmalıdır. Bize kilitlenebilir denetimsel gözetleyicilerle ilgilenmeye iten asıl neden ise, denetim altında üretilen dilde belli bir oranda kilitlenmeye izin vermenin üretilen istenilen işaretli dilin miktarında artışa neden olmasıdır. Bu noktada "Ne oranda kilitlenmeye izin verilmeli?" sorusu kendini göstermektedir. Bu çalışma bu soruya formal bir yanıt bulmayı hedeflemektedir.

En az kısıtlamalı kilitlenmesiz çözüm ve tam başarımlı çözüm sırasıyla $L(S/G)=\overline{L_{am}^{\uparrow C}}$ ve $L(S/G)=L_{am}^{\downarrow C}$ ile elde edilir. O zaman herhangi bir anlamlı çözüm $\overline{L_{am}^{\uparrow C}} \subseteq L(S/G) \subseteq L_{am}^{\downarrow C}$ koşulunu sağlamalıdır. Bu yüzden tüm kabul edilebilir çözümleri içeren dil kümesi $L_{cand} := \left\{\Gamma : \left(\overline{L_{am}^{\uparrow C}} \subseteq \Gamma \subseteq L_{am}^{\downarrow C}\right) \wedge \left(\Gamma = \overline{\Gamma}\right) \wedge \left(\overline{\Gamma}\Sigma_{uc} \cap L(G) \subseteq \overline{\Gamma}\right)\right\}$ şeklinde tanımlanabilir. Tanımdaki son iki madde herhangi bir çözümün kapalı ve kontrol edilebilir dil olması gerektiğini ifade etmektedir.

İki dil arasındaki farklılık bu iki dil arasında birbirine göre farklı olan kelimelerle ifade edilebilir. O zaman $d(L_1,L_2) := \beta\left(\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\}\right)$ ile ifade edilen mesafe fonksiyonu ayrık olay sistem yapısının doğal bir sonucudur öyle ki iki dil arasındaki mesafe, birbirlerine göre farklı olan kelimelerin ederlerinin toplamı ile ifade edilmiştir. Sonuçta 2^{Σ^*} kümesi ve d mesafe fonksiyonu bir metrik uzay oluştururlar. Aynı zamanda denetim altındaki sistemin performansı $\hat{J}(L(S/G)) := \widehat{SM^C}(L(S/G)) + \widehat{BM}(L(S/G))$ eşitliği ile verilir; burada $\widehat{SM^C}(L(S/G))$ ve $\widehat{BM}(L(S/G))$ sırasıyla başarısızlık ve kilitlenme ölçülerini ifade eder.

Kilitlenebilir denetimsel gözetleyicilerde performans iki genel kavram üzerine dayalıdır: kilitlenme ve başarısızlık. Tanımlanmış olan performans ölçüsü de bu ana fikri bünyesinde barındırmaktadır; öyle ki kilitlenme ölçüsü ve başarısızlık ölçüsünün toplamları üzerinden tanımlanmıştır. Kilitlenebilir denetimsel gözetleyiciyi seçerken açık bir şekilde kilitlenme ve başarısızlık arasındaki değiş tokuş gözükür. Örnek olarak en az kilitlenmeli çözüm ve tam başarımlı çözümü inceleyecek olursak bu değiş tokuş kolaylıkla görülebilir. En az kısıtlamalı kilitlenmesiz çözüm yapısı gereği hiçbir kilitlenme içermediğinden kilitlenme ölçüsü sıfıra eşittir. Fakat kabul edilebilir çözümleri içeren dil kümesi içinde en yüksek başarısızlık ölçüsüne sahiptir. Diğer bir taraftan tam başarımlı çözüm ise bütün istenilen işaretli kelimeleri ürettiğinden başarısızlık ölçüsü sıfırdır fakat diğer kabul edilebilir dillere nazaran en yüksek kilitlenme ölçüsüne sahiptir.

Performans ölçüsü kilitlenme ve başarısızlık üzerine dayandığından en iyi çözüm tüm işaretli kelimeleri hiçbir kilitlenmeye neden olmaksızın üretebilen denetimsel gözetleyicidir. Fakat bu çözüm kontrol edilebilirlik ve kabul edilebilirlik koşulları gereği her zaman makul olmayabilir. O zaman kabul edilebilir çözümler kümesi içinden bir kilitlenebilir denetimsel gözetleyici seçmek gerekir. Aynı zamanda bu çözümün en düşük performans ederine de sahip olmalıdır. Denetimsel gözetleyiciden kabul edilebilir işaretli kelimeleri de olabildiğince çok üretmesini bekleriz. O zaman sonucun sadece optimal olması yeterli değildir, aynı zamanda denetimsel gözetleyicinin istenilen işaretli dili de olabildiğince işaretleyebilir olması gerekir. Sonuç olarak optimal kilitlenebilir denetimsel gözetleyici problemi şu şekilde tanımlanmıştır:

$$\arg\left\{\min_{L(S/G)\in L_{cand}} \hat{J}\left(L\left(S/G\right)\right)\right\}$$

İki girişi olacak şekilde yeni bir işlem tanımlanmıştır; bu iki giriş sırasıyla L_{cand} kümesinin bir elemanı olan kabul edilebilir bir aday dil ve dile ait bir kilitlenmedir. Tanımlanmış bu yeni işlem L_{cand} kümesinden L_{cand} kümesinedir.

$$T\left(L,\alpha_i\right) := \begin{cases} \left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C} & if \;\; \hat{J}\left(\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) < \hat{J}\left(L\right) \\ L & otherwise \end{cases}$$

Ayrık olay sistemlerindeki kilitlenebilir denetimsel gözetleyiciler incelendiğinde bazı olası kilitlenmelere karşılık gelen kelimelerin kesinlikle çıkartılması gerekir ve ayrıca kabul edilebilir işaretli kelimelerin de olabildiğince çok üretilmesi gerektiği görülmektedir. Bu sebepten istenilen işaretli dili de olabildiğince işaretleyebilen

optimal kilitlenebilir denetimsel gözetleyiciyi elde eden bir algoritma sunulmuştur. Elde edilecek sonucun optimallığı ayrıca ispatlanmıştır.

Optimizasyon Algoritması

1. Adım

$L_{\text{IS}} = L_{am}^{\downarrow C}$ hesapla.

$L_{\text{temp 1}} = L_{\text{IS}}$ olarak kabul et.

2. Adım

$BM\left(L_{\text{temp 1}}\right) = \{s_1, s_2, s_3, ..., s_n\}$'yi bul.

$\forall s_i \in BM\left(L_{\text{temp 1}}\right)$ için $F\left(r\left(L_{\text{temp 1}}, s_i\right)\right)$'yi hesapla.

$BM\left(L_{\text{temp 1}}\right)$ kümesini kayıp faktörüne göre azalacak şekilde organize et.

3. Adım

For $i = 1$ *to* $\left\|BM\left(L_{\text{temp 1}}\right)\right\|$

$\quad \text{A} = BM\left(L_{\text{temp 1}}\right)\{i\}$

$\quad L_{\text{x}} = T\left(L_{\text{temp i}},\ \text{A}\right)$

$\quad L_{\text{temp i+1}} = L_{\text{x}}$

end

$L_{rst} = L_{\text{temp i+1}}$

4. Adım

Eğer $L_{\text{rst}} \neq L_{\text{temp 1}}$

$\quad L_{\text{temp 1}} = L_{\text{rst}}$

$\quad 2.adım'a\ git$

diğer

$\quad L_{\text{FS}} = L_{\text{rst}}$

Algoritmanın yapısı karmaşık değildir fakat kendini tekrarlayan bir yapıya sahiptir. Burada başlangıç dilinin performansını optimize etmek için önerilen algoritma belli bir sıra ile bazı kelimeleri dilden çıkartır. Tam başarımlı çözüme ait kilitlenmeler ilk olarak incelenir. 2. adımdaki kurala göre kilitlenmelerin uygulama sırası düzenlenir. 3. adımda düzenlenmiş kelimeler ile T işlemi başlangıç diline uygulanır. Bu noktada bazı kelimeler dilden çıkartılabilir. 4. adımda 3. adımda elde edilen sonuç ile başlangıç çözümü karşılaştırılır. Eğer eşit iseler algoritma sonlandırılır ve çözüm sonuç çözüm olarak kabul edilir. Eğer değil ise sonuç çözüm 2. adımın başlangıç

çözümü olarak kabul edilir ve 2. ve 3. adımlar yeniden işletilir. Sonuç olarak elde edilen çözüm istenilen işaretli dili de olabildiğince işaretleyebilen optimal kilitlenebilir denetimsel gözetleyicidir. Algoritma aynı zamanda eş zamanlı veri yönetim sistemi üzerinde açıklanmıştır.

Bunun yanı sıra ayrık olay sistemleri için Matlab'da bir fonksiyon grubu tasarlanmıştır. Ulaşılabilir kısmı bulma, işaretli-ulaşılabilir kısmı bulma, çarpım işlemi, paralel birleşim işlemi gibi sıkça kullanılan işlemler bu fonksiyon grubuna dâhil edilmiştir. Aynı zamanda geliştirilmiş olan algoritma da programlanmış ve test edilmiştir.

DESIGN OF AN OPTIMAL SUPERVISORY FOR DISCRETE EVENT SYSTEMS

SUMMARY

Although discrete event systems are dynamical systems, they differ from time-varying dynamic systems as they cannot be modeled by ordinary differential equations. These systems evolve in time due to events occurring at possibly irregular time intervals. Discrete event systems are often modeled as regular languages that can be represented by deterministic finite state automata. When the behavior of the system is examined, it is usually seen that the behavior is not satisfactory and must be "modified" by a control action. Thus, supervisory control theory, which was initiated by Ramadge and Wonham, is a powerful tool which gives us the possibility to build in the modifications via a supervisor as much as possible. But all the desired behavior of the system can not be usually marked by the supervisor which is built according to nonblocking and controllability constraints. In this manner, the blocking in discrete event systems is an important issue. When we focus on industrial examples, supervisory control problem with blocking arises more generally than the supervisory control problems of nonblocking and also blocking supervisors are preferred to nonblocking ones in several applications. Database concurrency control is a good example for this case. The most popular protocols are coupled with deadlock detection schemes since deadlock prevention and avoidance is impractical. In similar cases nonblocking solutions may end up being too conservative. So one may be willing to take the risk of some possible blocking occurrences in order not to constrain the system's behavior. It is clear that the taken risk may pay off if there is a serious increase in performance of the system. Therefore the performance of the system has to be investigated by introducing the benefits and losses of the possible blockings to the system.

When we examine a discrete event system within the corresponding admissible language, too many blockings may be realized. We then usually select minimally restrictive nonblocking solution as the supervisor, even though it is inadequate due to its restrictive behavior. Preventing all uncontrollable events that lead to blocking, minimal restrictive nonblocking solution can provide a conservative result. As a result, the supervisor can generate only a small part of admissible marked language. As such this strategy may constrain the behavior of the system considerably. On the other hand, the completely satisfying solution generates all the admissible marked strings, but its price adds too many blockings to the supervised language. Therefore, neither of these two results can be acceptable in many discrete event system problems. Here the success of the supervisor is related to the generated admissible marked language. Then it will be of interest to relax the nonblocking requirement and consider the synthesis of blocking supervisors. What motivates us to consider blocking solutions is that by allowing a certain amount of blocking, we can increase the part of admissible marked language that can be achieved under control. At this point, "how many blockings do arise?" This work aims to find a formal answer to this question.

The minimal restrictive nonblocking solution and the complete satisfying solution are achieved by $L(S/G)=\overline{L_{am}^{\uparrow C}}$ and $L(S/G)=L_{am}^{\downarrow C}$ respectively. Then any meaningful solution should satisfy the following: $\overline{L_{am}^{\uparrow C}} \subseteq L(S/G) \subseteq L_{am}^{\downarrow C}$. condition Therefore we define the class of all admissible solutions is $L_{cand} := \left\{\Gamma : \left(\overline{L_{am}^{\uparrow C}} \subseteq \Gamma \subseteq L_{am}^{\downarrow C}\right) \wedge \left(\Gamma = \overline{\Gamma}\right) \wedge \left(\overline{\Gamma}\Sigma_{uc} \cap L(G) \subseteq \overline{\Gamma}\right)\right\}$. Here the last two clauses refer to the fact that any solution must be a closed and controllable language.

The dissimilarity between two languages can be expressed by the strings that differ from each other. So the distance function expressed with $d(L_1, L_2) := \beta\left(\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\}\right)$ is a natural consequence of discrete event system structure such that the distance between two languages is given by summing the costs of distinct strings between them. Then the set 2^{Σ^*} and the distance function d form a metric space. Also the performance of a supervised system is given as $\hat{J}(L(S/G)) := \widehat{SM^C}(L(S/G)) + \widehat{BM}(L(S/G))$ equation where $\widehat{SM^C}(L(S/G))$ and $\widehat{BM}(L(S/G))$ express the non-satisfying measure and blocking measure respectively.

The performance of blocking supervisors hold on two concepts: blocking and failure. Then the defined performance measure captures the fundamental concept, as it is

given over sum of blocking measure and non-satisfying measure. Here it is clear that a trade-off between blocking and failure usually occurs usually while selecting the blocking supervisor. For example, when we examine the minimal restrictive nonblocking solution and complete satisfying solution, we can easily see the trade-off between blocking and failure. As the minimal restrictive nonblocking solution does not include a blocking string due to its structure, the blocking measure of it is equal to zero. But its non-satisfying measure score is the highest in the admissible solutions set. On the other hand, as the complete satisfying solution generates all the admissible marked strings, its non-satisfying measure equals to zero but it has the highest blocking measure according to all other admissible solutions.

As the performance measure of depends on blockings and failure, the best solution will be the supervisor that generates all the admissible marked strings with having no blockings. But this solution may not be feasible according to controllability and admissibility conditions. Then we have to select a blocking supervisor from the admissible solutions set. Also this solution has the minimum performance measure. At the same time we expect from the supervisor to generate the admissible marked strings as much as possible. Then the optimality of the solution is not enough but also it has to be the maximally permissive solution. As a result the optimal blocking supervisor problem is defined as

$$\arg\left\{\min_{L(S/G)\in L_{cand}} \hat{J}\left(L\left(S/G\right)\right)\right\}$$

We define a new operation over two inputs; an admissible candidate language and a blocking string which is a member of $BM\left(L\right)$. The defined new operation is from from L_{cand} set to L_{cand} set.

$$T\left(L,\alpha_i\right) := \begin{cases} \left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C} & if \quad \hat{J}\left(\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) < \hat{J}\left(L\right) \\ L & otherwise \end{cases}$$

When the blocking supervisors in discrete event systems are examined, it is seen that some of the possible blocking strings have to be removed and also the admissible marked strings have to be generated as much as possible. For this purpose we propose an algorithm to obtain the maximally permissive and also optimal blocking supervisor. The optimality of the obtained solution is also proofed.

The Optimization Algorithm

Step 1

Calculate $L_{\text{IS}} = L_{am}^{\downarrow C}$ and $L_{\text{temp 1}} = L_{\text{IS}}$

Step 2

Find $BM\left(L_{\text{temp 1}}\right) = \{s_1, s_2, s_3, ..., s_n\}$

For $\forall s_i \in BM\left(L_{\text{temp 1}}\right)$ calculate $F\left(r\left(L_{\text{temp 1}}, s_i\right)\right)$ and reorganize the $BM\left(L_{\text{temp 1}}\right)$ set according to its loss factor values in a descending order

Step 3

For $i = 1$ *to* $\left\|BM\left(L_{\text{temp 1}}\right)\right\|$

$\text{A} = BM\left(L_{\text{temp 1}}\right)\{i\}$

$L_{\text{x}} = T\left(L_{\text{temp i}},\ \text{A}\right)$

$L_{\text{temp i+1}} = L_{\text{x}}$

End

$L_{rst} = L_{\text{temp i+1}}$

Step 4

If $L_{\text{rst}} \neq L_{\text{temp 1}}$

$L_{\text{temp 1}} = L_{\text{rst}}$

Goto Step 2

else

$L_{\text{FS}} = L_{\text{rst}}$

The structure of the algorithm is not complex but it has an recursive structure. Here the proposed algorithm removes some strings in a sequence from the initial language in order to optimize the performance of it. The blocking strings of the complete satisfying solution are examined at first. The execution sequence of these blockings is arranged by the rule at step 2. Then T operation is executed with the ordered blocking strings set at step 3. Here some of the blockings can be removed. At step 4, the result obtained at the end of step 3 is compared with the initial one. If they are equal, the algorithm is finalized and the result language is accepted as final language. If not, the result language is accepted as initial language of step 2 and step 2 and step 3 are operated again. As a result the obtained final solution will be maximally permissive and also optimal blocking supervisor. Also the algorithm is explained over a database management system.

Furthermore a toolbox is developed for discrete event systems at Matlab. The frequently used operations are included in this toolbox such that accessibility, co-accessibility, product, parallel composition are one of those. Also the developed algorithm was programmed and tested within this toolbox.

1. GİRİŞ

Mekanik, elektriksel, kimyasal özellikler taşıyan dinamik sistemlerin davranışlarını incelenmek ve kontrol etmek için yer değiştirme, hız, ivme, akım, gerilim, basınç, sıcaklık, akış hızı, viskozite gibi fiziksel büyüklüklere bağlı olarak tanımlanan amaca uygun modeller kullanılır; elde edilen bu modellerden yola çıkılarak kontrol kuralları bulunur. Bu tür sistemleri modellemek için seçilen değişkenlerin ortak özelliği genellikle hepsinin zamana bağlı sürekli değişen büyüklükler olmasıdır. Bu tür sistemleri modellemek, analiz etmek ve kontrol etmek için sistem ve kontrol kuramı kapsamında değişik matematiksel yöntemler ve yaklaşımlar başarıyla kullanılmaktadır.

Diğer yandan son otuz yıl içerisinde bilgisayar, haberleşme ve elektronik teknolojisindeki hızlı gelişimin bir sonucu olarak yeni bir tür dinamik sistem ortaya çıkmıştır. Öyle ki bu tür dinamik sistemler incelendiğinde, büyük bir bölümünün çoğu zaman tamamının ayrık, zamandan bağımsız değişkenlerle kontrol edildiği görülmektedir. Üretim sistemleri, bilgisayar sistemleri, haberleşme sistemleri, hava trafik sistemleri gibi birçok sistem bu tür sistem sınıfına girmektedir. Bu tür sistemlerin davranışı genellikle eş zamansız olarak oluşan ayrık olaylara bağlı olarak değişir. Bu özellikleri nedeniyle bu sistemler Ayrık Olay Sistemleri (Discrete Event Systems) olarak adlandırılmaktadır.

Kapsadığı uygulama alanındaki beklentilerin her geçen gün biraz daha artması ve bu alandaki hızlı gelişmenin sonucunda ayrık olay sistemleri yeni, etkileyici ve ümit vadeden bir araştırma alanı olarak hem akademisyenlerin hem de mühendislerin ilgisini çekmektedir. Bu bağlamda bu güne kadar farklı çalışma grupları tarafından ayrık olay sistemlerini modellemek ve elde edilen modeller üzerinden analiz yapabilmek için değişik yöntemler önerilmiştir. Bu sistemleri kontrol etmek için ise genellikle Ramadge ve Wonham tarafından temelleri atılan denetimsel gözetleyici kuramı bünyesinde elde edilen sonuçlar kullanılmakta ve bu kuram üzerinden denetimsel gözetleyici yapıları önerilmektedir [1, 2]. Bu kuramın akademik ve mühendislik çevreleri tarafından bu kadar kabul görmesindeki en büyük etken, klasik kontrol kuramındaki kavramların bu kuram ile ayrık olay sistemlerine taşınmış olması ve kontrol kuramı bakış açısıyla sistemi analiz etmenin mümkün olmasıdır. Denetimsel gözetleyici kuramının tercih edilmesindeki bir diğer etken de sistem için

istenilen koşulları sağlayan denetimsel gözetleyicinin var olup olmadığını söylemesi ve bu denetimsel gözetleyicinin ne olduğuna ilişkin yanıtı vermesidir. Bu noktada denetimsel gözetleyici kuramı ayrık olay sistemlerinin gerçek hayattaki uygulama alanında sıklıkla tercih edilen başarılı bir kuramsal yaklaşım olarak karşımıza çıkmaktadır.

Klasik kontrol kuramına bakıldığında sistemin dinamiklerini bizim beklentilerimiz doğrultusunda değiştiren birim, kontrolör olarak adlandırılmaktadır. Örneğin, birinci mertebeden bir sistemi PI kontrolörü ile kontrol etmek istediğimizi düşünelim. İlk gözlemlenecek değişim kapalı çevrim sistemin yapısında var olan sürekli hal hatasının ortadan kalktığıdır. Sonuçta P ve I katsayılarına bağlı olarak sistem belli bir aşım ve yerleşme zamanı ile sürekli hal hatası yapmadan istenilen referans değere yerleşecektir. Burada tasarlanan kontrolör ile sistemin çıkışı bizim beklentilerimiz doğrultusunda değiştirilmiş, sisteme yeni bir dinamik kazandırılmıştır.

Ayrık olay sistemlerinde ise klasik kontrol kuramındaki kontrolörden farklı olarak denetimsel gözetleyici; sistemi, yapısında var olan davranış şekillerinin bir alt kümesiyle sınırlandırmaya çalışmaktadır. Öyle ki bu iki farklı kuramdaki iki birimin birbirine göre en temel farkı sistemden beklenilen davranışı elde etmek için izledikleri yöntemdir; kontrolör sistemin dinamiklerini değiştirirken, denetimsel gözetleyici sistemin davranışı üzerine kısıtlar getirmektedir. Burada denetimsel gözetleyici sistem üzerinde hem gözetleme hem de denetim yapmaktadır. Fakat sistemin yapısı gereği yapılan kısıtlama ile kazandırılmak istenilen her davranış gerçeklenebilir değildir.

Sistem ister doğrusal, doğrusal olmayan veya ayrık olay tabanlı olsun; en genel halde sistemin onu denetleyen birim ile birlikteki davranışının her zaman kontrol edilebilir olmasını isteriz. Denetimsel gözetleyici kuramının temelinde de ayrık olay sistemlerindeki bu eksikliği gidermek için sistem kuramındaki kontrol edilebilirlik kavramının ayrık olay sistemlerine taşıma fikri yatar. Sisteme denetimsel gözetleyici ile birlikte kazandırılmak istenilen davranış kontrol edilebilir ise genel anlamda sorun yoktur. Bu davranış gerçeklenebilir bir davranıştır. Eğer kontrol edilebilir değil ise bu noktada istenilen davranışa en yakın kontrol edilebilir davranış var mıdır ve eğer var ise bu davranışı üretebilecek denetimsel gözetleyiciyi nasıl elde ederiz gibi yeni problemler ortaya çıkar. Günümüzde bu tür problemlere denetimsel gözetleyici kuramı kapsamında çözümler aranmaktadır. Bunun yanı sıra zaman içerisinde ayrık olay sistemlerine sistem kuramından aşina olduğumuz gözlenebilirlik kavramı da taşınmıştır. Böylelikle ayrık olay sistemlerini kontrol bakış açısı altında inceleme ve denetleme olanağı sağlanmıştır.

Ayrık olay sistemleri için kontrol edilebilirlik, gözlenebilirlik gibi önemli bir diğer konu da kilitlenmedir. Ayrık olay sistemleri için kilitlenme, tanım olarak sistemin işaretli olmayan bir durum ya da durum kümesinde takılıp kalması ve davranışını sürdürememesidir. Öyle ki bazı çalışmalarda kilitlenmesiz olma canlılık olarak ta ifade edilmektedir. Sistemin yapısı karmaşıklaşmaya başladıkça olası kilitlenme miktarı da artar. Diğer bir taraftan kurulan sistem modeli çok fazla detaya sahip ise model gerektiğinden fazla işlem karmaşıklığına ve analiz zorluklarına neden olur, çok basit bir model de sistem dinamiklerini ifade etmede yetersiz kalabilir. Sonuç olarak kurulan modelin ne kadar detaya sahip olması gerektiği problemin ihtiyaçlarına göre şekillenir.

Denetimsel gözetleyici altında çalışan sistemin davranışının her zaman hem kontrol edilebilir hem de kilitlenmesiz olması istenir. Fakat bu koşulları sağlayan bir denetimsel gözetleyici olmayabilir ya da bu koşulları sağlayan bir denetimsel gözetleyici olsa bile tutucu bir davranışa sahip olabilir. Bu noktada bir çözüm elde edebilmek ya da tutucu davranışa sahip olan bu denetimsel gözetleyiciden daha az tutucu bir denetimsel gözetleyiciyi seçebilmek için kilitlenme ve/veya kontrol edilebilirlik koşullarının yeniden değerlendirilmesi gerekir. Kontrol edilebilirlik koşulu denetimsel gözetleyiciler için vazgeçilmezdir ve kontrol edilemeyen bir davranışa sahip çözüm gerçeklenebilir bir çözüm olmadığından kesinlikle kabul edilebilir değildir. O zaman bu tür ayrık olay sistemlerinde beklenilen davranışa yakınsayabilmek için sistemin bazı durumlarda kilitlenmesine izin verilmelidir.

Diğer bir taraftan ayrık olay sistemlerinde kilitlenme yapısal bir olgudur ve her tür sistem için kilitlenebilir çözümler kabul edilebilir çözümler değildir. Örneğin, trafik sistemleri gibi kilitlenmenin ciddi bir davranış ihlali olarak algılandığı ayrık olay sistemlerinde kilitlenme kabul edilemez. Buna karşılık kilitlenmenin kolaylıkla algılanabildiği ve çözülebildiği veri yönetim sistemleri gibi sistemlerde ise kilitlenmeye genellikle izin verilmektedir. Bu sebepten bu tür sistemlerde kilitlenmesiz çözümden çok istenilen davranışa daha yakın fakat kilitlenebilir çözümler tercih edilmektedir. Fakat kilitlenmesiz olma koşulunu yumuşatmak, tüm kilitlenmelere denetimsel gözetleyici tarafından izin verileceği anlamına gelmez. Sistemdeki kilitlenmeler belli ölçülere göre değerlendirilmelidir. Öyle ki bir kilitlenmeye izin verilmesi durumunda, bu kilitlenmenin sisteme olan etkisi ve kazandırılan yeni davranışın istenilen davranışa olan yakınlığı beraber incelenmelidir. Ayrıca problemin yapısı gereği istenilen davranışa olan yakınlıkla, sistemin sahip olduğu olası kilitlenmeler arasında genellikle bir ilişki vardır. Sisteme denetimsel gözetleyici ile beklenilen davranış kazandırıldığında çok fazla kilitlenmeyi de beraberinde getiriyorsa, kilitlenme miktarını azaltmak çoğu zaman

sistemin istenilen davranıştan uzaklaşmasıyla sonuçlanır. Buna karşın denetim altındaki davranış istenilen davranışa yaklaştıkça kilitlenmelerde bir artış gözlemlenir. Burada kilitlenme ile istenilen davranışın elde edilen kısmı arasında bir değiş tokuş vardır. Bu noktada "Ne oranda bir kilitlenmeye izin verilecek?" sorusunun yanıtı aranmaktadır. Bu çalışmanın temel amacı da kilitlenebilir denetimsel gözetleyici problemini araştırmak ve bu probleme biçimsel bir cevap bulmaktır.

Bu çalışma beş bölümden oluşmaktadır. İkinci bölümde ayrık olay sistemleri için temel kavramlar ve denetimsel kontrol kuramı anlatılmış ve denetimsel kontrol kuramı ile ilgili temel problemlerden bahsedilmiştir. Üçüncü bölümde kilitlenebilir denetimsel gözetleyici problemi ayrıntılı olarak açıklanmış ve istenilen dile en yakın ve en düşük performans ölçüsüne sahip dilin nasıl elde edilebileceğine yönelik bir yöntem önerilmiştir. Dördüncü bölümde ayrık olay sistemleri için Matlab üzerinde geliştirilmiş olan fonksiyonlara değinilmiş, geliştirilen algoritmalar açıklanmıştır. Beşinci bölümde ise sonuçlar aktarılmış ve çalışma sonlandırılmıştır.

1.1 Literatür Araştırması

Ayrık olay sistemlerinde denetimsel gözetleyici kuramına göre kilitlenme kavramı ilk olarak Ramadge ve Wonham tarafından ortaya atılmıştır [1]. Kilitlenebilir denetimsel gözetleyici problemi ise ilk olarak Chen ve Lafortune tarafından ifade edilmiştir [3, 4]. Bu çalışmalarda neden bazı uygulamalarda kilitlenebilir denetimsel gözetleyicilere ihtiyaç olduğu belirtilmiştir. Bu iki çalışmada Chen ve Lafortune tarafından kilitlenebilir denetimsel gözetleyici probleminin dört özel durumu için çözüm önerilmiştir. Ayrıca bu çalışmalarda kapsama anlamında çalışan iki farklı işlem tanıtılmıştır. Bu iki işlemin yardımıyla sistemin ürettiği kilitlenme miktarını azaltmak ve sistemin başarımını arttırmak amaçlanmıştır. Bu çalışmadan esinlenilerek kilitlenebilir denetimsel gözetleyici problemi yönlendirilmiş ağlara taşınmış, benzer çözümler bu yapı için de elde edilmiştir [5].

Ayrık olay sistemlerinde kararlılık farklı gruplar tarafından incelenmiştir [6-9]. Burada elde edilen sonuçlara dayanarak kararlılık kavramı üzerinden ayrık olay sistemlerinde kilitlenme incelenmiştir. Bu çalışmada problem ayrık olay sisteminin ürettiği dil üzerinden değil sistemin durumları üzerinden tanımlanmıştır. [10].

Ayrık olay sistemlerinde kilitlenme ve optimizasyon ise ilk olarak Xi-Ren Cao ve Yu-Chi Ho tarafından incelenmiştir. Bu çalışmada bir ikili üretim sistemi, kilitlenme

içeren döngüsel sıralı ağlar yardımıyla modellenmiş ve ayrık olay sistemleri için perturbasyon yöntemi ile analiz edilmiştir [11].

Denetimsel gözetleyici ile birlikte sistemin davranışının her zaman kilitlenebilir olması istenmeyebilir. Chen ve Lafortune tarafından modüler denetimsel gözetleyicilerde kilitlenmesiz denetimsel gözetleyicinin nasıl elde edileceği araştırılmış ve iki yöntem önerilmiştir [12]. Ayrıca S. E. Bourdon, M. Lawford ve W.M. Wonham tarafından kilitlenmesiz ve dayanıklı denetimsel gözetleyiciler incelenmiş, zamanlı ve zamansız ayrık olay sistemleri için çözümler önerilmiştir [13, 14]. Benzer bir şekilde farklı ayrık olay sistemi problemleri için kilitlenmesiz çözümlerin nasıl elde edileceği ve bunların nasıl uygulanacağı farklı çalışma grupları tarafından incelenmiştir. [15-29]

Ayrık olay sistemlerinde optimum denetimsel kontrol, Ratnesh Kumar ve Vijay K. Garg tarafından da ele alınmıştır. Kumar ve arkadaşları bu çalışmada kontrol eder fonksiyonu ve kontrol ceza fonksiyonu olarak iki tür fonksiyon tanımlamış ve bu fonksiyonlar yardımıyla sisteme ilişkin net ederin minimumunu elde eden dili bulmaya çalışmışlardır. Bu çalışmada optimizasyon yöntemi olarak ağ akış teknikleri kullanılmıştır [30].

Passino ve Antsaklis tarafından sistemdeki olay geçişleri üzerinden yeni bir eder fonksiyonu tanımlanmış, sistemin davranışını kısıtlayacak şekilde bir kontrol amacı ortaya atılmış ve optimizasyon problemi bir yörünge izleme problemine dönüştürülmüştür [31]. Diğer taraftan Yitzhak Brave ve Michael Heymann tarafından ayrık olay sistemlerinde optimizasyon, bir optimal çekim problemi olarak tanımlanmış ve sistemin belli bir durum kümesine yakınsaması ve orada kararlı kılınması olarak ifade edilmiştir [8, 32].

Raja Sengupta ve Stephane Lafortune tarafından Kumar ve Garg'ın çalışmasındakine benzer bir şekilde kontrol edilen sistemin ederini belirlemek üzere kontrol ve eder fonksiyonları tanımlanmıştır. Bu çalışmada olayların tamamının gözlemlenebilir olduğu kabul edilmiş ve sisteme ait davranış kümesi içinde en fazla davranışa izin veren minimum dilin bulunması amaçlanmıştır. Optimizasyon yöntemi olarak dinamik programlama kullanılmıştır [33, 34].

Raja Sengupta ve Stephane Lafortune tarafından geliştirilen yöntem, tek bir amaca göre optimizasyon yapacak şekilde geliştirilmiştir. Herve Marchand, Olivier Boivineau ve Stephane Lafortune tarafından ise bu yöntem birden fazla amacı kapsayacak şekilde genişletilmiştir [35]. Olaylardan bir kısmının gözlemlenebilir olmaması halinde ise problem ayrıca çözülmüştür [36].

Xi Wang ve Asok Ray tarafından düzenli diller için işaretli bir mesafe fonksiyonu önerilmiştir [37, 38]. Öyle ki bu mesafe fonksiyon ile Jinbo Fu, Asok Ray ve Constantino M. Lagoa tarafından düzenli diller için kontrol edilen olayların bir kısmına izin verilmemesine dayanan bir optimal kontrol kuramı ortaya atılmıştır [39-41]. Problem dayanıklı denetimsel gözetleyici problemi olarak ayrıca çözülmüştür [42]. Benzer bir şekilde aynı grup tarafından rastlantısal yapıya sahip ayrık olay sistemleri için genelleştirilmiş bir mesafe fonksiyonu da önerilmiştir [43].

Bunlardan bağımsız olarak Mohanty, Chandra ve Kumar tarafından ayrık olay sistemlerinde optimizasyon Ramadge ve Wongam tarafından önerilen denetimsel gözetleyici yapısı üzerinden değil de, yeni bir denetimsel gözetleyici yapısı ile kontrol edilmesi düşünülmüş ve problem bu şekilde ele alınmıştır. Bu bağlamda ayrık olay sistemleri için belli şartlar altında optimal kontrolörü veren bir algoritma geliştirilmiştir [44-47].

Chung, Lafortune ve Lin tarafından ayrık olay sisteminin çok karmaşık olması ya da çok fazla durum içermesi veya tamamının elde edilmesinin güç olduğu hallerde, sistemin tamamı üzerinden değil de o anki duruma göre belli bir uzunluğa kadar olan kısmıyla ilgilenen bir yöntem önerilmiştir. Literatürde bu yöntem sınırlı görüşlü denetimsel gözetleyici yöntemi olarak anılmaktadır. Chung, Lafortune ve Lin tarafından oluşturulan bu yapıda hangi şartlar altında denetimsel gözetleyici elde edileceği araştırılmıştır [48, 49]. Aynı zamanda Chung ve Lafortune tarafından uygun denetimsel gözetleyiciyi elde eden yeni bir algoritma önerilmiştir [50, 51]. Kısmi gözlemlenebilir ayrık olay sistemleri için sınırlı görüşlü denetimsel gözetleyiciler kullanarak çevrim içi çözümün nasıl elde edileceği ayrıca incelenmiş ve en fazla kontrol edilebilir ve gözlemlenebilir alt dilleri elde eden yeni algoritmalar önerilmiştir [52-55]. Sınırlı görüşlü denetimsel gözetleyiciler için kestirim tabanlı denetimsel gözetleyici tasarımı ayrıca incelenmiştir [56, 57]. Diğer bir taraftan R. R. Gudwin ve F. C. Gomine tarafından bu şekilde modellenen ayrık olay sistemleri için optimal denetimsel gözetleyici tasarımı problemi ortaya atılmış ve bu probleme genetik algoritmalar ile çözüm aranmıştır [58].

L. Grigorov ve K. Rudie sınırlı görüşlü ayrık olay sistemleri için yeni bir dinamik yapı ve bu yeni yapı üzerinden sınırlı görüşlü denetimsel gözetleyiciler için yaklaşık optimal bir çözüm önerilmiştir. Bu önerilen çözümün yalın olması daha hızlı çözüme yakınsanmasını sağlamaktadır [59-61].

F. Lin bulanık yapıya sahip ayrık olay sistemlerini ortaya atmıştır [62-64]. Bulanık yapıyı sınırlı görüşlü ayrık olay sistemlerine de uyarlamış ve yeni bir en iyi karar verme yöntemi önermiştir [65].

Yang gelişimsel programlama yöntemi ile ayrık olay sistemleri için en iyi denetimsel gözetleyici seçim problemi ile ilgilenmiştir. Bu yapılan çalışmaların ilkinde ayrık olay sistemi deterministik sonlu durumlu otomatlar yardımıyla modellenirken, ikincisinde giriş/çıkış otomatları ile modellenmiştir [66, 67].

2. AYRIK OLAY SİSTEMLERİ

2.1 Giriş

Ayrık olay sistemleri dinamik bir yapıya sahip olmasına rağmen kontrol ya da devre kuramı kapsamında incelediğimiz zamanla değişen doğrusal sistemlerden farklı olarak diferansiyel denklemlerle ya da fark denkleri ile ifade edilememektedirler. Ayrık olay sistemleri, zamanla değişen sürekli sistemlerden şu iki özellikle farklılık gösterir:

1. Durum uzayı ayrık bir kümedir.
2. Durum geçişi olay tabanlıdır.

Bu nedenle ayrık olay sistemlerinin davranışını incelemek için farklı modelleme yöntemleri ortaya atılmıştır. Bunlardan en çok kullanılanı sonlu durumlu otomatlar ve denetimsel gözetleyici kuramıdır [1, 2, 68-70]. Bunun yanı sıra petri ağları da akademi ve endüstri tarafından sıklıkla tercih edilmektedir [70-73]. Oluşturulmak istenilen modelde sadece olayların oluşumu değil aynı zamanda olayların etkin olduğu süreler ile de ilgileniliyorsa zamanlı otomatlar ya da zamanlı geçiş modelleri tercih edilir [74-76]. Eğer saat yapısı kesin değil rastlantısal ise rastlantısal zamanlı otomatlar tercih edilir [70]. Zamana bağlı sürekli değişen sistemlere benzer bir şekilde ayrık olay sistemleri cebrik bir modelleme yapısı üzerinden modellenmek istenirse $(max,+)$ cebri ve bunun üzerinden geliştirilmiş otomat yapısı tercih edilir. $(max,+)$ cebri ile sisteme ait öz değer ve öz vektörler elde edilebilir; sistemin matris yapısı ve sistemin sürekli hal davranışı incelenebilir [70, 77].

Bu çalışmada, ayrık olay sistemlerini modellemek için sonlu durumlu otomatlardan faydalanılmıştır. Bu bölümde ise bu modelleme tekniğine ilişkin temel kavramlara ve özelliklere değinilecek, ayrık olay sistemlerine ilişkin temel problemler ve bu problemler için önerilmiş çözümler aktarılacaktır. Bu kapsamda ayrık olay sistemleri için temel problemlerden birisi olarak kabul edilen kilitlenebilir denetimsel gözetleyici problemi de ele alınılacaktır.

2.2 Diller ve Otomatlar ile İlgili Temel Kavramlar

Ayrık olay sistemlerinin davranışını incelemek ve modellemek için geliştirilmiş biçimsel yollardan birisi de diller ve otomatlardır. Bu kuramsal yapının başlangıç noktası, sistemi süren sonlu elemanlı olay kümesidir ve $\sum$ ile gösterilir.

Olay kümesinin elemanları ile oluşturulan diziye kelime denir. Uzunluğu sıfır olan yani hiçbir olaydan oluşmayan kelimeye ise boş kelime denir ve ε ile gösterilir.

2.2.1 Diller

Olay kümesi $\sum$ üzerine tanımlı sonlu uzunluklu kelimelerden oluşan kümeye dil denir [70, 78]. Örneğin $\sum = \{e_1, e_2, e_3\}$ için $L_1 = \{\varepsilon, e_1, e_3, e_3e_2, e_3e_2e_1\}$ ve $L_2 = \{e_1, e_2, e_3, e_1e_2, e_1e_2e_3\}$ bu olay kümesi üzerine tanımlı iki farklı dildir. Burada L_1 ve L_2 dilinin her bir elemanı birer kelimedir.

$s \in L$ olacak şekilde bir kelime olmak üzere, s kelimesinin uzunluğu $\|s\|$ ile ifade edilir ve $\|s\| \to \mathbb{N}$ 'dır. L_1 ve L_2 'ye benzer bir şekilde $L_3 = \{e_1$ *olayı ile başlayan tüm kelimeler*$\}$ 'de bir dili ifade etmektedir.

Herhangi bir dile ait kelimeler olayların birbirine bağlanması sonucu elde edilir. Mesela L_1 diline ait $e_3e_2e_1$ kelimesi e_3e_2 ve e_1 kelimelerinin birbiri bağlanması sonucunda elde edilmiştir.

$\sum$ olay kümesi ile oluşturulacak bütün sonlu kelimelerin kümesine olay kümesinin Kleene-kapanışı denir ve $\sum^*$ ile gösterilir. $\sum = \{e_1, e_2, e_3\}$ olay kümesinin Kleene kapanışı $\sum^* = \{\varepsilon, e_1, e_2, e_3, e_1e_1, e_1e_2, e_1e_3, e_2e_1, e_2e_2, e_2e_3, e_3e_1, e_3e_2, e_3e_3, ...\}$ şeklindedir. Aynı zamanda $\sum$ olay kümesi üzerine tanımlı olan bütün diller her zaman $\sum^*$ 'nin bir alt kümesidir.

$s \in \sum^*$ olacak şekilde herhangi bir kelimenin en başında üretilen olay s kelimesinin öneki ve en sonunda üretilen olay ise s kelimesinin soneki olarak adlandırılır. Mesela $s = e_1e_2e_3$ için e_1 s 'nin öneki ve e_3 ise s 'nin sonekidir.

2.2.2 Diller Üzerine Tanımlanan İşlemler

Diller yapı olarak birer küme olduğundan kümeler üzerine tanımlı olan birleşim, kesişim, fark ve $\sum^*$ göre eşlenik işlemleri diller için de aynen geçerlidir. Bunun yanında diller için şu ek işlemler tanımlanmıştır [68, 70, 79].

2.2.2.1 Birbirine Bağlama

$L_a, L_b \subseteq \Sigma^*$ iki farklı dil olmak üzere, $L_a L_b = \{s \in \Sigma^* : (s = s_a s_b)\ ve\ (s_a \in L_a)\ ve\ (s \in L_b)\}$ bu iki dilin birbirine bağlanmasını ifade etmektedir.

Örnek olarak $L_4 = \{\varepsilon, e_1, e_1 e_2\}$ ve $L_5 = \{e_2, e_3\}$ iki farklı dil olsun. Bu iki dilin birbirine bağlanması sonucunda $L_4 L_5 = \{e_2, e_3, e_1 e_2, e_1 e_3, e_1 e_2 e_2, e_1 e_2 e_3\}$ elde edilir.

2.2.2.2 Önek Kapanışı

$L \subseteq \Sigma^*$ olmak üzere L dilinin önek kapanışı $\overline{L} = \{s \in \Sigma^* : \exists t \in \Sigma^* (st \in L)\}$ olarak tanımlanır.

Sahip olduğu kelimelerin öneklerini de içeren dile önek kapalı olan dil denir. En genel halde $L \subseteq \overline{L}$ dir. $L = \overline{L}$ ise dil önek kapalıdır. Eğer $L_i, L_j \in \Sigma^*$ iki dil olmak üzere $\overline{L_i} \cap \overline{L_j} = \overline{L_i \cap L_j}$ ise L_i ve L_j için çakışmasız diller denir.

2.2.2.3 Kleene-Kapanışı

$L \subseteq \Sigma^*$ olmak üzere L dilinin Kleene Kapanışı $L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \ldots$ ile ifade edilir. Bu işlem, olay kümesinin Kleene-kapanışı işlemi (Σ^*) ile aynıdır. $L^* \subseteq \Sigma^*$ her zaman sağlanır ve L^* önek kapalı olmak zorunda değildir.

2.2.2.4 Örnek

$\Sigma = \{e_1, e_2, e_3\}$ olmak üzere $L_6 = \{\varepsilon, e_1 e_2\}$ ve $L_7 = \{e_2\}$ iki dil olsun. Hem L_6 hem de L_7 dilleri önek kapalı değildir zira $e_1 \notin L_6$ ve $\varepsilon \notin L_7$'dir. Dillere ilişkin bazı işlemlerin sonuçları aşağıdaki gibidir.

$L_6 L_7 = \{e_2, e_1 e_2 e_2\}$
$\overline{L_6} = \{\varepsilon, e_1, e_1 e_2\}$
$\overline{L_6} L_7 = \{e_2, e_1 e_2, e_1 e_2 e_2\}$
$L_6^* = \{\varepsilon, e_1 e_2, e_1 e_2 e_1 e_2, e_1 e_2 e_1 e_2 e_1 e_2, \ldots\}$
$L_7^* = \{\varepsilon, e_2, e_2 e_2, e_2 e_2 e_2, e_2 e_2 e_2 e_2, \ldots\}$

2.2.3 Otomatlar

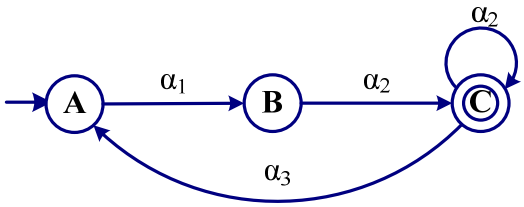
Ayrık olay sistemlerinin davranışını ifade etmede en uygun yollardan birisi de dillerdir. Ayrık olay sisteminin dinamikleri, olay kümesi üzerinden tanımlanmış olan

uygun bir dil ile ifade edilebilirken, sisteme ait dili yazmak her zaman kolay değildir. Örneğin 2.2.1'deki L_1 ve L_2 dillerine ait kelimeler rahatlıkla yazılabilirken L_3 diline ait kelimeleri topluca ifade etmek bu aşamada zordur. Sonuçta diller üzerinden yapılacak olan işlemlerde dillere ait kelimelere ihtiyaç vardır. Ayrık olay sistemlerinde, sistemlerin davranışını tanımlayan dil otomatlar yardımıyla elde edilebilir.

2.2.3.1 Deterministik Sonlu Durumlu Otomat (DSDO)

Deterministik sonlu durumlu bir otomat $G=(X,\Sigma,f,x_0,X_m)$ beşlisi ile tanımlanır. Burada X otomata ait durum kümesini, Σ ise olay kümesini, $f:X\times\Sigma\to X$ durum geçiş fonksiyonunu ifade etmektedir. x_0 deterministik sonlu durumlu otomata ilişkin başlangıç durumudur. $X_m\subseteq X$ ise işaretlenmiş durum kümesidir [68, 70, 80]. Otomatın tanımındaki durum geçiş fonksiyonu her durum ve olay için tanımlı olmak zorunda değildir. Geçiş fonksiyonu $f:X\times\Sigma^*\to X$ olacak şekilde yinelemeli olarak şu şekilde genişletilebilinir: $e_1,e_2\in\Sigma$, $e_1e_2\in L(G)$ ve $x_1,x_2\in X$ olmak üzere $f(x_1,e_1):=x_2$ ise $f(x_1,e_1e_2):=f(f(x_1,e_1),e_2)$'dir. Aynı zamanda işaretli durumlar otomat tarafından kabul edilen durumları göstermektedir.

Deterministik sonlu durumlu otomata örnek olarak Şekil 2.1'deki $G_1=(X_1,\Sigma_1,f_1,x_{1,0},X_{1,m})$ otomatına ilişkin durum geçiş grafı verilebilir.



Şekil 2.1 : G_1 otomatı

G_1 otomatının durum ve işaretli durum kümeleri sırasıyla $X_1=\{A,B,C\}$ ve $X_{1,m}=\{C\}$'dir. G_1 otomatının başlangıç durumu $x_{1,0}=A$, G_1 otomatına ilişkin olay kümesi ise $\Sigma_1=\{\alpha_1,\alpha_2,\alpha_3\}$ şeklindedir. Otomata ait durum geçiş fonksiyonları $f_1(A,\alpha_1)=B$, $f_1(B,\alpha_2)=C$, $f_1(C,\alpha_3)=A$ ve $f_1(C,\alpha_2)=C$ biçiminde ifade edilir. Buna göre $f_1(B,\alpha_2\alpha_2\alpha_3)=f_1(f_1(f_1(B,\alpha_2),\alpha_2),\alpha_3)=A$ ve $f_1(C,\alpha_2\alpha_2...\alpha_2)=f_1(f_1(f_1(C,\alpha_2),\alpha_2)...,\alpha_2)=C$'dir.

2.2.3.2 Üretilen Dil

$G=(X,\Sigma,f,x_0,X_m)$ deterministik sonlu durumlu otomatın ürettiği dil şu şekilde tanımlanır: $L(G)=\{s\in\Sigma^* \text{ öyle ki } f(x_0,s) \text{ tanımlı olsun}\}$ [68].

Burada $L(G)$ dili, durum geçiş grafındaki bütün yönlendirilmiş kolları ifade eder. Tanım gereği her zaman $L(G)=\overline{L(G)}$ yani önek kapalıdır [70]. $L(G)$ dili sistemin tüm davranışlarını kapsar. Aynı zamanda $L(G)$, sistemin kontrol edilmemiş davranışlarını içeren dil olarak ta adlandırılır.

Örneğin Şekil 2.1'de verilen G_1 otomatının ürettiği dil

$$L(G_1)=\{\varepsilon,\alpha_1,\alpha_1\alpha_2,\alpha_1\alpha_2\alpha_2,\alpha_1\alpha_2\alpha_3,\alpha_1\alpha_2\alpha_3\alpha_1,\alpha_1\alpha_2\alpha_2\alpha_2,...\}$$

biçiminde ifade edilir.

2.2.3.3 İşaretli Dil

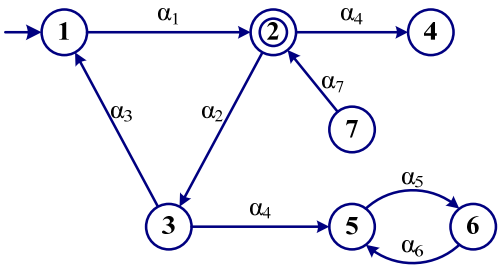
Bir $G=(X,\Sigma,f,x_0,X_m)$ deterministik sonlu durumlu otomata ilişkin işaretli dil şu şekilde tanımlanır: $L_m(G)=\{s\in\Sigma^* : f(x_0,s)\in X_m\}$

Otomata ait işaretli durumlarda sonlanan olası tüm kelimeler işaretli dili oluştururlar. Otomata ilişkin bütün durumlar işaretli durum olmak zorunda olmadığından otomata ait işaretli dil de önek kapalı olmak zorunda değildir [68, 70, 79]. Aynı zamanda işaretli dil her zaman üretilen dilin bir alt kümesidir. Şekil 2.1'de verilen G_1 otomatına ilişkin işaretli dil

$L_m(G_1)=\{\alpha_1\alpha_2,\alpha_1\alpha_2\alpha_2,\alpha_1\alpha_2\alpha_3\alpha_1\alpha_2,\alpha_1\alpha_2\alpha_2\alpha_3\alpha_1\alpha_2,...\}$ biçimde ifade edilir.

2.2.3.4 Kilitlenme

Eğer G otomatı bir $x_k\in X$ durumuna erişmiş, öyle ki $f(x_k)=\varnothing$ ve $x_k\notin X_m$ ise x_k durumunda bir kilitlenme vardır ve x_k durumunda oluşan kilitlenmeye cansız kilitlenme denir. Benzer bir şekilde G otomatı $P=\{x_i,x_j,...\}\in X$ durum kümesine erişmiş, öyle ki $\forall x\in P$ için $x\notin X_m$ olsun. Buna göre P durum kümesine ait herhangi bir durumdan sistemi işaretli bir duruma götürecek bir kelime yok ise G otomatında bir canlı kilitlenme vardır denir [3, 4, 81, 82]. Eğer $\overline{L_m(G)}=L(G)$ ise G otomatı kilitlenmesiz bir otomattır.

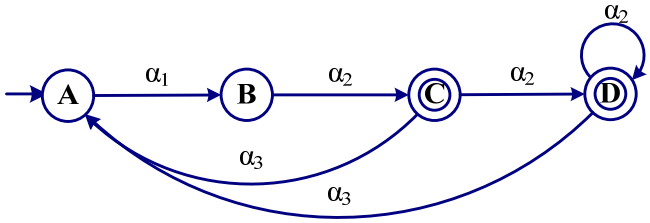


Şekil 2.2 : G_2 otomatı

Şekil 2.2'de verilen G_2 otomatında 4 no.lu durumda cansız kilitlenme, 5 ve 6 no.lu durumlarda ise bir canlı kilitlenme söz konusudur.

2.2.3.5 Otomatların Eşitliği

Ayrık olay sistemlerinde sisteme ait çıkış olaylar üzerinden tanımlanan dildir. Bir dili birbirinden farklı otomatlar ile ifade etmek mümkündür. İki farklı otomatın eşitliği de ürettikleri ve işaretledikleri diller üzerinden tanımlanır. G_i ve G_j iki farklı otomat olmak üzere eğer $L(G_i) = L(G_j)$ ve $L_m(G_i) = L_m(G_j)$ ise G_i ve G_j otomatları eşit otomatlardır denir [68, 70, 78, 79].



Şekil 2.3 : G_3 otomatı

Şekil 2.1'deki G_1 otomatı ile Şekil 2.3'deki G_3 otomatı verilen tanımlara göre eşit otomatlardır.

2.2.4 Otomatlar Üzerine Tanımlanan İşlemler

Otomatlar ile tanımlanan ayrık olay sistemlerini analiz etmek için otomatlar üzerinde bazı işlemler tanımlanmıştır. Böylelikle otomatın sahip olduğu dili incelemek ve düzenlemek mümkün olmaktadır.

2.2.4.1 Ulaşılabilir Kısım

$G=\left(X,\Sigma,f,x_0,X_m\right)$ otomatına ilişkin başlangıç durumundan itibaren ulaşılabilen tüm durumları içeren otomata, G otomatının ulaşılabilir kısmı denir. $Ac(G)$ ile gösterilir.

$$Ac(G):=\left(X_{ac},\Sigma,f_{ac},x_0,X_{ac,m}\right) \text{ öyle ki} \tag{2.1}$$

$$X_{ac}=\left\{x\in X:\exists s\in\Sigma^*\left(f\left(x_0,s\right)=x\right)\right\} \tag{2.2}$$

$$f_{ac}=f\big|_{X_{ac}\times\Sigma\to X_{ac}} \tag{2.3}$$

$$X_{ac,m}=X_m\bigcap X_{ac} \tag{2.4}$$

Bu tanıma göre G otomatının ulaşılabilir kısmını elde etmek için başlangıç durumundan ulaşılamayan bütün durumlar ve bu durumlara ilişkin bütün geçiş fonksiyonları kaldırılır [68, 78, 79].

2.2.4.2 İşaretli-Ulaşılabilir Kısım

İşaretli durumlara ulaşılan durumlar dışındaki bütün durumların silinmesi sonucunda elde edilen otomata G otomatının işaretli-ulaşılabilir kısmı denir ve $CoAc(G)$ ile gösterilir.

$$CoAc(G):=\left(X_{coac},\Sigma,f_{coac},x_0,X_{coac,m}\right) \text{ öyle ki} \tag{2.5}$$

$$X_{coac}=\left\{x\in X:\exists s\in\Sigma^*\left(f\left(x,s\right)\in X_m\right)\right\} \tag{2.6}$$

$$x_0=\begin{cases} x_0 & eğer\ x_0\in X_{coac} \\ tanımsız & diğer\ hallerde \end{cases} \tag{2.7}$$

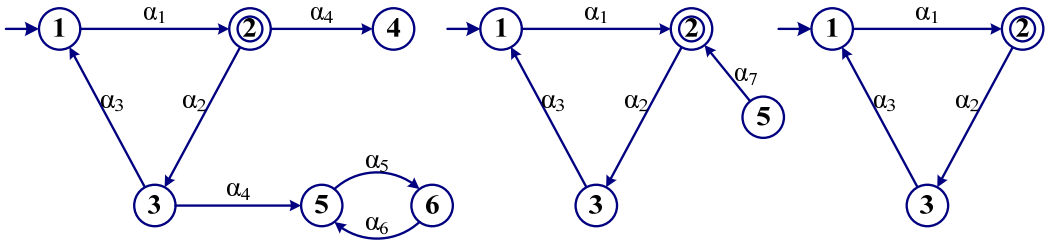
$$f_{coac}=f\big|_{X_{coac}\times\Sigma\to X_{coac}} \tag{2.8}$$

$CoAc$ işlemi Ac işleminden farklı olarak sistemin ürettiği dilde bir kısıtlamaya gidebilir zira bir otomatın işaretli-ulaşılabilir kısmını elde ederken otomata ilişkin işaretli dilde bir değişiklik olmaz. Diğer bir taraftan üretilen dilde bir daralma gözlemlenebilir. Bu özelliğinin bir sonucu olarak işaretli-ulaşılabilirlik kilitlenme ile doğrudan ilişkilidir. Eğer $CoAc(G)\neq G$ ise G otomatı kilitlenmeye sahiptir öyle ki $\overline{L_m\left(G\right)}\neq L\left(G\right)$'dir. Bunun sonucunda G otomatı öyle durumlara sahiptir ki bu durumlar ulaşılabilir iken işaretli-ulaşılabilir değildir [68, 78, 79].

2.2.4.3 Budanmış Otomat

Eğer bir $G=(X,\Sigma,f,x_0,X_m)$ otomatı hem ulaşılabilir yani $Ac(G)=G$ ve hem de işaretli-ulaşılabilir yani $CoAc(G)=G$ ise G otomatına budanmış otomat denir. Bir G otomatının $Trim(G)=CoAc\left[Ac(G)\right]=Ac\left[Coac(G)\right]$ işlemi ile budanmış kısmı elde edilir.

Şekil 2.2'de verilen G_2 otomatının ulaşılabilir kısmı, işaretli-ulaşılabilir kısmı ve budanmış otomatı Şekil 2.4'te verilmiştir.



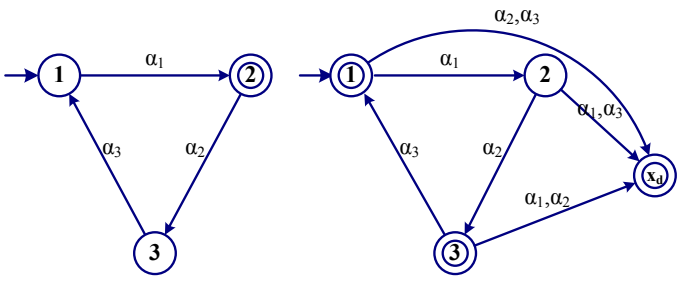
Şekil 2.4 : $Ac(G_2)$, $Coac(G_2)$, $Trim(G_2)$

2.2.4.4 Tümleyen Otomat

İşaretlediği dil $L_m(G)$ olan bir $G=(X,\Sigma,f,x_0,X_m)$ budanmış otomatımız olsun. Bu otomatın ürettiği dil $L(G)=\overline{L_m(G)}$ olacaktır. "\" işlemi küme teorisindeki fark alma işlemi olmak üzere $\Sigma^*\setminus L_m(G)$ dilini işaretleyen otomata tümleyen otomat denir ve $G^{comp}=(X_{comp},\Sigma,f_{tol},x_0,X_{comp,m})$ ile ifade edilir öyle ki $X_{comp}=X\cup\{x_d\}$, $X_{comp,m}=X\cup\{x_d\}\setminus X_m$ 'dir ve durum geçiş fonksiyonu şu şekilde tanımlıdır:

$$f_{tol}=\begin{cases} f(x,e) & e\in f(x) \\ x_d & diğer \end{cases} \qquad \textbf{(2.9)}$$

$L(G^{comp})=\Sigma^*$ ve $L_m(G^{comp})=\Sigma^*\setminus L_m(G)$ 'dir. şekil 2.5'de Şekil 2.2'de verilen G_2 otomatının budanmış kısmının tümleyen otomatı alınmıştır.



Şekil 2.5 : $G_t = Trim(G_2)$ ve G_t^{comp}

2.2.4.5 Çarpım İşlemi

$G_i = (X_i, \Sigma_i, f_i, x_{i,0}, X_{i,m})$ ve $G_j = (X_j, \Sigma_j, f_j, x_{j,0}, X_{j,m})$ iki farklı otomat olmak üzere verilen bu iki otomatın çarpımı şu şekilde tanımlanır:

$$G_i \times G_j := Ac(X_i \times X_j, \Sigma_i \cap \Sigma_j, f, (x_{i,0}, x_{j,0}), X_{i,m} \times X_{j,m}) \text{ öyle ki} \quad \textbf{(2.10)}$$

$$f((x_k, x_l), e) := \begin{cases} (f_i(x_k, e), f_j(x_l, e)) & eğer\ e \in f_i(x_k) \cap f_j(x_l) \\ tanımsız & diğer \end{cases} \quad \textbf{(2.11)}$$

Bu iki otomatın çarpımın ürettiği ve işaretlediği diller sırasıyla şu şekildedir.

$$L(G_i \times G_j) = L(G_i) \cap L(G_j) \quad \textbf{(2.12)}$$

$$L_m(G_i \times G_j) = L_m(G_i) \cap L_m(G_j) \quad \textbf{(2.13)}$$

İki dilin kesişimi ile bu iki dili üreten otomatların çarpımı arasında ilişki vardır. Öyle ki iki otomatın çarpımı ile elde edilen otomata ait üretilen ve işaretlenen diller bu iki otomatın ürettiği ve işaretlediği dillerin kesişimidir. Aynı zamanda eğer $\Sigma_i \cap \Sigma_j = \varnothing$ ise $L(G_i \times G_j) = \{\varepsilon\}$'dir.

2.2.4.6 Paralel Birleşim İşlemi

$G_i = (X_i, \Sigma_i, f_i, x_{i,0}, X_{i,m})$ ve $G_j = (X_j, \Sigma_j, f_j, x_{j,0}, X_{j,m})$ iki farklı otomat olmak üzere verilen iki otomatın paralel birleşimi şu şekilde tanımlanır:

$$G_i \| G_j := Ac(X_i \times X_j, \Sigma_i \cup \Sigma_j, f, (x_{i,0}, x_{j,0}), X_{i,m} \times X_{j,m}) \quad \textbf{(2.14)}$$

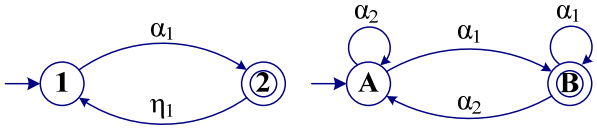
$$f\left(\left(x_k,x_l\right),e\right):=\begin{cases}\left(f_i\left(x_k,e\right),f_j\left(x_l,e\right)\right) & eğer\ e\in f_i\left(x_k\right)\cap f_j\left(x_l\right)\\ \left(f_i\left(x_k,e\right),x_l\right) & eğer\ e\in f_i\left(x_k\right)\setminus\sum_j\\ \left(x_k,f_j\left(x_l,e\right)\right) & eğer\ e\in f_j\left(x_l\right)\setminus\sum_i\\ tanımsız & diğer\end{cases} \tag{2.15}$$

Eğer $\sum_i=\sum_j$ ise paralel birleşim işlemi ile çarpım işlemi aynı sonucu verir. Aynı zamanda G_i, G_j ve G_k üç farklı otomat olmak üzere

$$G_i\left\|G_j=G_j\right\|G_i \tag{2.16}$$

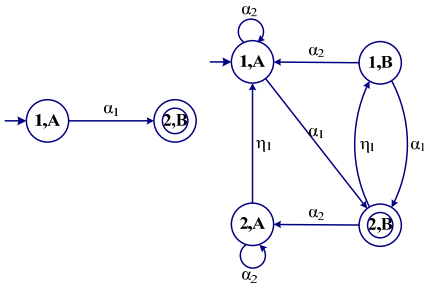
$$\left(G_i\left\|G_j\right.\right)\left\|G_k=G_i\right\|\left(G_j\left\|G_k\right.\right) \tag{2.17}$$

ilişkisi her zaman geçerlidir.



Şekil 2.6 : G_4 ve G_5 otomatları

Şekil 2.6'da verilen G_4 ve G_5 otomatlarının çarpımları ve paralel birleşimleri sonucu elde edilen $G_4\times G_5$ ve $G_4\left\|G_5\right.$ otomatları Şekil 2.7'deki gösterilmiştir.



Şekil 2.7 : $G_4\times G_5$ ve $G_4\left\|G_5\right.$ otomatları

2.2.5 Düzgün Diller

Ayrık olay sistemlerinin davranışını ifade eden diller sistemin yapısı gereği bazen sonlu sayıda kelimeden oluşurken bazen de sonsuz sayıda kelimeden oluşabilir. Bu dillerden, sonlu durumlu otomatlarla işaretlenen dillere düzgün diller denir ve $\mathbb{R}$ ile gösterilir. Sonlu sayıda kelimeden oluşan tüm diller sonlu durumlu otomatlar ile ifade edilebilir. Örneğin n adet kelimeden oluşan bir dil en kötü halde n adet durumla ifade edilebilir. Bunun bir sonucu olarak sonlu sayıda kelime içeren tüm diller düzgün dildir. Fakat sonsuz sayıda kelime içeren dillerin tamamı sonlu

durumlu otomatlar ile ifade edilemez. k^n, k olayının n defa birbiri ardına eklenmesiyle oluşan kelime olmak üzere, $L=\left\{k^n l^n : n \geq 0\right\}$ dilini sonlu sayıda bir otomatla ifade etmek imkânsızdır [68, 78].

2.2.5.1 Düzgün Dillerin Özellikleri

$L_i, L_j \in \mathbb{R}$ iki farklı düzgün dil olmak üzere

$\overline{L_i} \in \mathbb{R}$ $\quad$ $L_i \bigcup L_j \in \mathbb{R}$

$L_i^* \in \mathbb{R}$ $\quad$ $L_i L_j \in \mathbb{R}$

$L_i^c := \Sigma^* \setminus \mathrm{L}_i \in \mathbb{R}$ $\quad$ $L_i \bigcap L_j \in \mathbb{R}$

özellikleri her zaman geçerlidir.

2.2.5.2 Düzenli İfadeler

Düzgün dilleri ifade etmenin bir diğer yolu da düzenli ifadelerdir. Düzenli ifadeler üç işlem üzerine kuruludur. Bunlar sırasıyla birbirine bağlama, Kleene kapanışı ve birleşim işlemleridir. Bu üç işlem ile tüm düzgün diller düzenli ifadeler ile ifade edilebilir. Mesela $\{\varepsilon, s, ss, sss, ssss, ...\}$ dilini s^* düzenli ifadesi ile göstermek mümkündür. Benzer bir şekilde birleşim işlemi düzenli ifadelerde "+" ile ifade edilmektedir. Mesela $\{e_1e_2, e_1e_3e_2\}$ diline $e_1e_2 + e_1e_3e_2$ düzenli ifadesi karşılık gelmektedir.

G_4 otomatının işaretlediği dil $L_m(G_4) = \{\alpha_1, \alpha_1\eta_1\alpha_1, \alpha_1\eta_1\alpha_1\eta_1\alpha_1, ...\}$ ve bu dile ilişkin düzenli ifade ise $(\alpha_1\eta_1)^* \alpha_1$ şeklindedir. Aynı zamanda G_5 otomatının işaretlediği dile ilişkin düzenli ifade ise $\left(\alpha_1\alpha_1^*\alpha_2\alpha_2^*\right)^* \alpha_1\alpha_1^*$ şeklindedir.

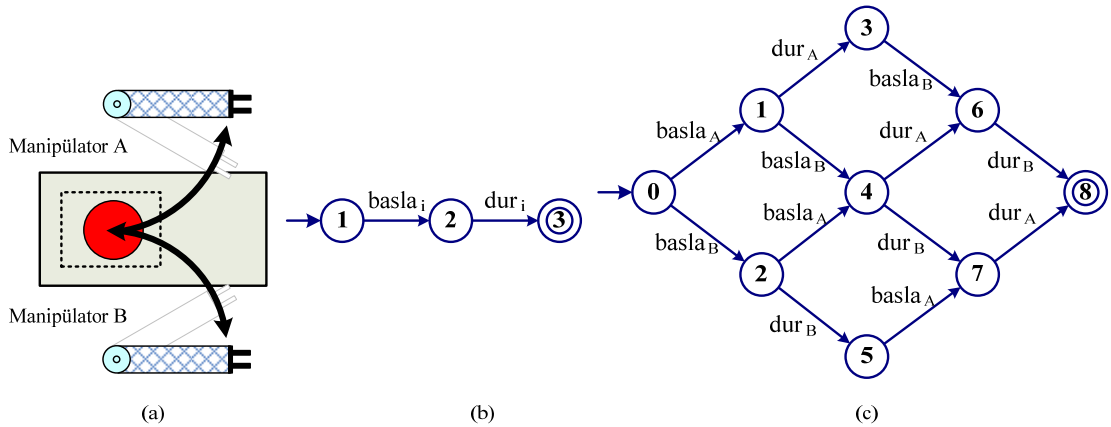
2.2.5.3 Kleene Kuramı

Düzenli ifade ile gösterilebilen her dil düzenli dildir. Her düzenli dil de düzenli ifade ile gösterilebilir. Bunun bir sonucu olarak tüm düzenli diller en az bir sonlu durumlu otomat ile gösterilebildiğinden düzenli ifadeler ile sonlu durumlu otomatlar arasında bir denklik vardır [78].

2.3 Denetimsel Kontrol Kuramı

2.3.1 Neden Denetimsel Gözetleyiciye İhtiyaç Vardır

Pek çok ayrık olay sisteminin sahip olduğu davranış kabul edilebilir değildir. Bu da sistemin davranışının gözlemlenmesini ve kontrol edilmesini gerektirir. Ayrık olay sistemini bir DSDO ile modellendiğimizi düşünelim. Sistemi bir denetimsel gözetleyici ile kontrol etmek, sistemin istenmeyen davranışlarının sınırlandırılması anlamına gelir. Ayrık olay sisteminde kontrol kavramını açıklayabilmek için Şekil 2.8-a'daki iki eş manipülatörden oluşan üretim bandını ele alalım.



Şekil 2.8 : a) Üretim bandının prensip şeması - b) Bir manipülatöre ait otomat Modeli - c) Sistemin otomat modeli

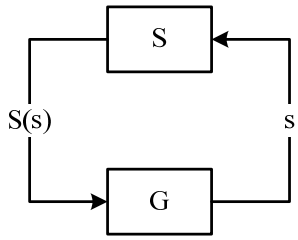
Manipülatörlerden birine ait otomat modeli Şekil 2.8-b de ve tüm sisteme ilişkin otomat modeli ise Şekil 2.8-c de gösterildiği gibi verilebilir. Bu üretim bandından beklenen çalışma şekli ise şu şekilde olsun: Ürün işleme noktasına yerleştirildikten sonra isteğe bağlı olarak $basla_A$ olayı ile A manipülatörü veya $basla_B$ olayı ile B manipülatörü devreye girsin. A manipülatörü devreye girmiş ise dur_A olayı ile A manipülatörü, B manipülatörü devreye girmiş ise dur_B olayı ile B manipülatörü işlemini bitirsin. A veya B manipülatörü işlemini bitirdikten sonra diğer manipülatöre ait başlatma olayı aktif olduğunda ilgili manipülatör devreye girsin. İlgili manipülatöre ait durdurma olayı ile de bu manipülatör işlemini tamamlamış olsun.

Şekil 2.8-c deki modele göre sistem bu beklenilen davranışın yanı sıra başka davranış şekillerini de içermektedir. Örneğin $basla_A$ olayı ile gelinen 1 numaralı durumda $basla_B$ olayı da aktiftir. Bu durumda dur_A olayı üretilmeden sistem tarafından $basla_B$ olayı da üretilebilir. Bu durumda B manipülatörü de işleme noktasına gelir ve her iki manipülatör çarpışır. Sonuçta sisteme ait otomat modeli kabul edilemeyecek kelimeler de üretebilmektedir. Bu kelimelerin üretilmesi engellenmelidir.

Bu örnekten de görüldüğü gibi sistemin tek başına sahip olduğu davranış kabul edilebilir değildir. Sistem beklentiler doğrultusunda kontrol edilmelidir. Bu da denetimsel gözetleyici kuramına göre bir S denetimsel gözetleyicisi ile sağlanır. Denetimsel gözetleyici kuramı ile kontrol edebilmek için sistemden beklenilen davranışa karşılık gelen $L(G)$'nin uygun alt dilleri ile ilgilenilir. Çoğu durumda bu bir alt dil değil bir alt dil grubudur. En az beklentileri içeren dil L_r ve kabul edilebilir dil L_a olmak üzere bu dil aralığı $L_r \subseteq L \subseteq L_a \subseteq L(G)$ şeklindedir. Burada L_r dili denetimsel gözetleyici ile çalışan sistemin en az sağlaması gereken davranışı ifade eder. L_a ise izin verilebilecek en fazla davranışa karşılık gelir.

2.3.2 Denetimsel Gözetleyici ile Geri Besleme

S denetimsel gözetleyicisi dinamik bir yapıya sahiptir, öyle ki G'de oluşan olayları gözlemlerken, bazı olayların olmasına izin verir, bazı olayların oluşmasını engeller ve sistemin ürettiği dili $L(G)$'nin bir alt dili olacak şekilde sınırlandırır. Ayrıca S denetimsel gözetleyicisi, sistemde oluşan tüm olaylar üzerinde denetim sahibi olmayabilir. Buna göre olay kümesi $\sum = \sum_c \cup \sum_{uc}$ olacak şekilde ikiye ayrılır. Burada $\sum_c$ kontrol edilebilen olaylar kümesini ve $\sum_{uc}$ ise kontrol edilemeyen olaylar kümesini ifade eder. Kontrol edilebilen olaylar, oluşmasını sistemin denetleyebildiği olaylardır. Veri iletim sisteminde bir mesajın gönderilmesi olayı ya da bir üretim sisteminde bir motoru devreye alacak çıkışın üretilmesi olayı bu tip olaylara birer örnektir. Kontrol edilemeyen olaylar ise oluşumunu tamamen sistemin kendi dinamiklerinin belirlediği olaylardır. Gerekli koşullar sağlanması halinde oluşması ya da oluşmaması engellenemez. Örneğin bir algılayıcının ürünü fark etmesi olayı ya da sistemin hata mesajı üretmesi olayı bu tür olaylardır. Buna göre denetimsel gözetleyici, sistemi kontrol etmek için sistemin bulunduğu duruma göre kontrol edilebilir olaylardan hangilerinin oluşmasına izin verileceğini belirler ve sistemin davranışını istenilen alt kümeyle sınırlandırmaya çalışır. G otomatının S denetimsel gözetleyicisi ile beraber çalışması Şekil 2.9 gösterildiği gibi ifade edilebilir.



Şekil 2.9 : Denetimsel gözetleyici ile sistemin prensip çalışma şeması

Sistemin s kelimesini üretmesi halinde, denetimsel gözetleyicinin sistem tarafından üretilmesine izin vereceği olay kümesi $S(s)$ ile ifade edilir. Bu şekilde çalışan

sisteme ait kontrol paradigması ise şu şekilde tanımlanmıştır: "G'deki geçiş fonksiyonları S tarafından kontrol edilmektedir öyle ki G'deki kontrol edilebilen olaylar S denetimsel gözetleyicisi tarafından dinamik olarak izin verilir ya da verilmez. Ayrıca kontrol edilemeyen olaylar ise S denetimsel gözetleyicisi tarafından her zaman izinlidirler." S denetimsel gözetleyicisinin G otomatını kontrol etmesi S/G ile gösterilir. Buna göre G otomatının S denetimsel gözetleyicisi ile birlikte ürettiği ve işaretlediği diller sırasıyla $L(S/G)$ ve $L_m(S/G)$'dir [1, 70, 83]. Örneğin Şekil 2.8'deki üretim sistemi için $s = basla_A$ ve $\sum_{uc} = \{dur_A, dur_B\}$ olsun. O zaman üretim sisteminde bir çarpışma olması istenmiyor ise $S(s) = \{dur_A\}$ olmalıdır. Burada denetimsel gözetleyici $basla_B$ olayının oluşmasını engellemiş, sistemin davranışını bir alt dille sınırlandırmıştır.

2.3.2.1 Kabul Edilebilirlik Koşulu

Kontrol paradigmasına göre ortaya konulan denetimsel gözetleyici, kabul edilebilir bir denetimsel gözetleyici olmak zorundadır. Kabul edilebilir bir denetimsel gözetleyici olması için gerek ve yeter koşul

$$\sum_{uc} \cap f(x_0, s) \subseteq S(s) \qquad \textbf{(2.18)}$$

şeklindedir. Bu koşul aslında paradigmadaki ikinci cümlenin matematiksel bir karşılığından başka bir şey değildir. Öyle ki s kelimesi ile gelinen durumda eğer kontrol edilemeyen olaylar var ise tüm bu kontrol edilemeyen olaylar her zaman denetimsel gözetleyici tarafından izin verilmesi gerekir. Diğer bir deyişle denetimsel gözetleyici hiçbir zaman bir kontrol edilemeyen olayın oluşmasını belirleyemez. Örneğin denetimsel gözetleyicinin üretim sisteminde bulunan bir algılayıcının çıkış sinyalini engellemeye çalışması veya haberleşme sisteminde sistemin üretebileceği hata mesajını engellemeye çalışması kabul edilebilirlik koşuluna ters düşmektedir.

2.3.2.2 S/G Tarafından Üretilen ve İşaretlenen Diller

S/G tarafından üretilen dil tekrarlamalı olarak şu şekilde tanımlanır:

- $\varepsilon \in L(S/G)$
- $\left[(s \in L(S/G)) ve (s\sigma \in L(G)) ve (\sigma \in S(s))\right] \Leftrightarrow \left[s\sigma \in L(S/G)\right]$

O zaman S/G'ye ait işaretli dil şu şekildedir:

$$L_m(S/G) := L(S/G) \cap L_m(G) \qquad \textbf{(2.19)}$$

Denetimsel gözetleyicinin tanımı gereği her zaman $L(S/G) \subseteq L(G)$'dir. Aynı zamanda tanım gereği önek kapalı olmak zorundadır: $L(S/G) = \overline{L(S/G)}$. Sisteme

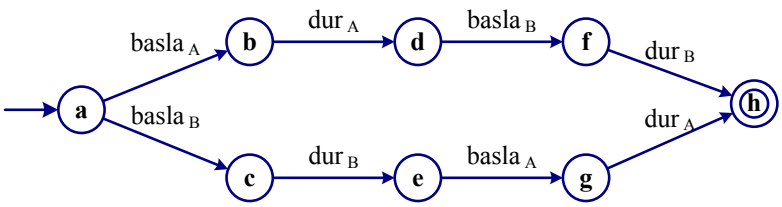
ait davranış denetimsel gözetleyici ile $L(G)$'nin belli bir altkümesi altına sıkıştırılmaya çalışılmaktadır. O zaman

$$L_r \subseteq L(S/G) \subseteq L_a \subseteq L(G) \quad \textbf{(2.20)}$$

$$L_{rm} \subseteq L_m(S/G) \subseteq L_{am} \subseteq L_m(G) \quad \textbf{(2.21)}$$

koşullarının sağlanması istenir. Burada L_a ve L_r daha önce tanımlandığı üzere sırasıyla kabul edilebilir dil ve en az beklentileri içeren dillerdir. Ayrık olay sistemlerinde denetimsel kontrol kuramı sisteme ait otomat modeli olan G'nin sonlu durumlu bir otomat ve L_a, L_{am}, L_r ve L_{rm} dillerinin düzgün birer dil olması durumunda geçerlidir.

Eğer düzgün bir K dili bir H otomatı tarafından işaretleniyor ise H otomatına K dilinin hatırlayıcısı denir. Sistemden beklentiler genellikle düzgün diller ile ifade edildiğinden, sistemden beklenilen davranışlar H_{spec} hatırlayıcısı ile ifade edilir. Şekil 2.8'de verilen üretim sistemine ait beklentileri ifade eden dil $L_{am} = \{basla_A dur_A basla_B dur_B, basla_B dur_B basla_A dur_A\}$ şeklindedir. Bu dili işaretleyen hatırlayıcı H_{spec1} ise Şekil 2.10'daki gibidir.



Şekil 2.10 : H_{spec1} hatırlayıcısı otomatı

Şekil 2.8-c'deki sistemle ilişkili olarak kontrol edilemeyen olaylar kümesi ise $\Sigma_{uc} = \{basla_A, dur_A, dur_B\}$ olsun. O zaman $\overline{L_{am}}$ dili kontrol paradigmasıyla ters düşmektedir. Zira hatırlayıcı c no.lu duruma geldiğinde sistem ise 2 no.lu duruma gelecektir. H_{spec1} c numaralı durumda sadece dur_B olayına izin vermektedir. Fakat sistemin otomat modeline göre gelinen 2 numaralı durumda hem dur_B olayı hem de $basla_A$ olayı izinlidir. Aynı zamanda $basla_A \in \Sigma_{uc}$ olduğundan, kontrol paradigması gereği $basla_A$ olayı da izinli olmalıdır. Fakat H_{spec1} bu olayı engellemeye çalışmaktadır. Bir diğer değişle kontrol paradigmasındaki kontrol edilemeyen olaylar ile ilgili ortaya koyulan kurala ters düşmektedir. Sonuçta H_{spec1}'in ürettiği ve işaretlediği diller kontrol edilebilir şekilde gerçeklenebilir değildir. Bu aslında sistem kuramındaki kontrol edilebilirlik kavramının ayrık olay sistemlerindeki bir

yansımasından başka bir şey değildir. Bununla ilgili olarak bir dilin kontrol edilebilir olup olmadığının anlaşılabilmesi için kontrol edilebilirlik kuramı ortaya atılmıştır.

2.3.3 Kontrol Edilebilirlik Kuramı

$G=(X,\Sigma,f,x_0,X_m)$ sonlu durumlu bir otomat, $\Sigma_{uc}\subseteq\Sigma$ kontrol edilemeyen olayları gösteren küme ve $K\subseteq L(G)$ $(K\neq\varnothing)$ beklenilen davranışları ifade eden dil olsun.

$L(S/G)=\overline{K}$ oluşturacak bir S denetimsel gözetleyicisi vardır $\Leftrightarrow \overline{K}\,\Sigma_{uc}\cap L(G)\subseteq\overline{K}$

K üzerindeki bu koşul kontrol edilebilirlik koşulu olarak adlandırılmaktadır.

İspat – Kontrol edilebilirlik kuramı her iki yönde de ispatlanacaktır.

$\Leftarrow(Yeter\ Koşul)$

$s\in L(G)$ için denetimsel gözetleyici $S(s)$

$S(s)=\left[\Sigma_{uc}\cap f(x_0,s)\right]\cup\left\{\sigma\in\Sigma_c : s\sigma\in\overline{K}\right\}$ olsun.

$L(S/G)=\overline{K}$ olduğunu önerilen $S(s)$ için ispatlayalım.

- Sıfır uzunluklu kelimeler için düşünelim.

Tanım gereği $\varepsilon\in L(S/G)$'dir. $K\neq\varnothing$ olduğundan $\varepsilon\in\overline{K}$'dır. Kelime uzunluğu sıfır için her iki dilin de kelimelerinin uzunluğu aynıdır.

- Bütün s kelimeleri için indüksiyon hipotezimiz şu şekilde olsun:

$\|s\|\leq n$ için $s\in L(S/G)\Leftrightarrow s\in\overline{K}$

$\|s\|=n$ için bunun doğruluğunu kabul edelim. Eğer $\|se\|=n+1$ için bunun doğruluğu ispatlanırsa $\forall s$ için $L(S/G)$ ile $\overline{K}$ dillerinin uzunlukları aynı olup sonuçta $L(S/G)=\overline{K}$ anlamına gelir.

$(İndüksiyon\ hipotezi\ için\Rightarrow)$

$se\in L(S/G)$ olsun. $L(S/G)$'nin tanımından

$\left\{\left[(s\in L(S/G)\text{ ve }(s\sigma\in L(G)\text{ ve }(\sigma\in S(s))\right]\Leftrightarrow\left[s\sigma\in L(S/G)\right]\right\}$

$s\in L(S/G), se\in L(G)$ ve $e\in S(s)$'dir.

$e\in\Sigma_{uc}$ ise $\overline{K}\,\Sigma_{uc}\cap L(G)\subseteq\overline{K}$ gereği $se\in\overline{K}$'dir

$e\in\Sigma_c$ ise $S(s)$'nin tanımı gereği $se\in\overline{K}$

$(\textit{İndüksiyon hipotezi için} \Leftarrow)$

$se \in \overline{K}$ olsun $K \subseteq L(G)$ varsayımından $se \in L(G)$'dir.

$e \in \sum_{uc}$ ise kabul edilebilirlik koşulu gereği $e \in S(s)$'dir.

$e \in \sum_{c}$ ise $S(s)$'nin tanımı gereği $e \in S(s)$'dir.

Sonuçta $s \in L(S/G), se \in L(G)$ ve $e \in S(s) \Leftrightarrow se \in L(S/G)$

Bu da indüksiyon hipotezinin doğruluğu anlamına gelir. $\forall s$ için $L(S/G)$ ile $\overline{K}$ dillerinin uzunlukları aynı sonuçta $L(S/G) = \overline{K}$ dir.

$\Rightarrow (\textit{Gerek Koşul})$

$L(S/G) = \overline{K}$ sağlayan kabul edilebilir bir S vardır.

$s \in \overline{K}$ ve $e \in \sum_{uc}$ ve $se \in L(G)$ olsun.

S denetimsel gözlemleyicisi kabul edilebilir olduğundan $e \in S(s)$'dir.

$L(S/G)$'nin tanımından $\left[(s \in L(S/G) \text{ ve } (se \in L(G) \text{ ve } (e \in S(s))\right]$

$\Leftrightarrow \left[se \in L(S/G)\right]$'dir. Aynı zamanda $L(S/G) = \overline{K}$ olduğundan

$s \in \overline{K}$ ve $se \in L(G)$ ve $e \in S(s) \Leftrightarrow se \in \overline{K}$

$\left[s \in \overline{K}\right]$ ve $\left[e \in \sum_{uc}\right]$ ve $\left[se \in L(G)\right] \Rightarrow se \in \overline{K}$

Sonuçta $\overline{K} \sum_{uc} \cap L(G) \subseteq \overline{K}$ **(2.22)**

Kuram yapıcıdır; verilen kontrol edilebilirlik koşulu sağlandığı takdirde $\overline{K}$ dilini sağlayan denetimsel gözetleyicinin varlığını söylemekle kalmayıp ne olduğunu da vermektedir. Kontrol edilebilirlik koşulu sağlandığı takdirde $\overline{K}$ dilini gerçekleyen denetimsel gözetleyici, kurama göre şu şekildedir:

$$S(s) = \left[\sum_{uc} \cap f(x_0, s)\right] \cup \left\{\sigma \in \sum_{c} : s\sigma \in \overline{K}\right\} \quad \textbf{(2.23)}$$

Şekil 2.10'daki H_{spec1} hatırlayıcısını düşünecek olursak $K = L_{am}$'dir ve $basla_B basla_A \in \overline{K} \sum_{uc} \cap L(G)$ iken $basla_B basla_A \notin \overline{K}$. Bu da $\overline{K} \sum_{uc} \cap L(G) \not\subset \overline{K}$ demektir. Kontrol edilebilirlik koşulundan da görüldüğü gibi Şekil 2.8'deki sisteme H_{spec1} ile kazandırılmak istenilen davranış kontrol edilebilir bir davranış değildir.

Ayrık olay sistemleri için kontrol edilebilirlik en temel kavramlardan birisidir. Kuram, K dilinin sistemin ürettiği $L(G)$ diline ve kontrol edilemeyen olaylar

kümesine göre kontrol edilebilir olup olmadığını söylemektedir. Genelde bir ayrık olay sistemi birden fazla alt sistemin paralel birleşimi ya da çarpımı ile ifade edildiğinden birden fazla dilin kesişimi ya da birleşimi K dilini ifade edebilir. Bu noktada alt dillerin kesişim ya da birleşimlerinin sonucunda elde edilen dilin kontrol edilebilir olup olmadığı önem kazanır.

2.3.3.1 Özellik 1

K_1 ve K_2 dilleri kontroledilebilir ise $K_1 \cup K_2$ kontrol edilebilirdir.

$\overline{K_1}\Sigma_{uc} \cap L(G) \subseteq \overline{K_1}$ ve $\overline{K_2}\Sigma_{uc} \cap L(G) \subseteq \overline{K_2}$

$\overline{(K_1 \cup K_2)}\Sigma_{uc} \cap L(G) = (\overline{K_1} \cup \overline{K_2})\Sigma_{uc} \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cup \overline{K_2}\Sigma_{uc}\right] \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cap L(G)\right] \cup \left[\overline{K_2}\Sigma_{uc} \cap L(G)\right]$

$\subseteq \overline{K_1} \cup \overline{K_2} = \overline{(K_1 \cup K_2)}$

2.3.3.2 Özellik 2

K_1 ve K_2 kontrol edilebilir ise $K_1 \cap K_2$ kontrol edilebilir olmak zorunda değildir.

$\overline{K_1}\Sigma_{uc} \cap L(G) \subseteq \overline{K_1}$ ve $\overline{K_2}\Sigma_{uc} \cap L(G) \subseteq \overline{K_2}$

$\overline{(K_1 \cap K_2)}\Sigma_{uc} \cap L(G) \subseteq (\overline{K_1} \cap \overline{K_2})\Sigma_{uc} \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cap \overline{K_2}\Sigma_{uc}\right] \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cap L(G)\right] \cap \left[\overline{K_2}\Sigma_{uc} \cap L(G)\right]$

$\subseteq \overline{K_1} \cap \overline{K_2} \neq \overline{K_1 \cap K_2}$

2.3.3.3 Özellik 3

Eğer $\overline{K_1} \cap \overline{K_2} = \overline{(K_1 \cap K_2)}$ ve K_1 ve K_2 kontrol edilebilir ise $K_1 \cap K_2$ kontrol edilebilirdir.

$\overline{K_1}\Sigma_{uc} \cap L(G) \subseteq \overline{K_1}$ ve $\overline{K_2}\Sigma_{uc} \cap L(G) \subseteq \overline{K_2}$

$\overline{(K_1 \cap K_2)}\Sigma_{uc} \cap L(G) \subseteq (\overline{K_1} \cap \overline{K_2})\Sigma_{uc} \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cap \overline{K_2}\Sigma_{uc}\right] \cap L(G)$

$= \left[\overline{K_1}\Sigma_{uc} \cap L(G)\right] \cap \left[\overline{K_2}\Sigma_{uc} \cap L(G)\right]$

$$\subseteq \overline{K_1} \cap \overline{K_2} = \overline{(K_1 \cap K_2)}$$
$$\Rightarrow \overline{(K_1 \cap K_2)} \Sigma_{uc} \cap L(G) \subseteq \overline{(K_1 \cap K_2)}$$

2.3.3.4 Özellik 4

Eğer $K_1 = \overline{K_1}$ ve $K_2 = \overline{K_2}$ yani her iki dil de önek kapalı diller iseler $K_1 \cap K_2$ önek kapalı ve kontrol edilebilir bir dildir.

$$\overline{K_1} \Sigma_{uc} \cap L(G) \subseteq \overline{K_1} \text{ ve } \overline{K_2} \Sigma_{uc} \cap L(G) \subseteq \overline{K_2}$$
$$\overline{(K_1 \cap K_2)} \Sigma_{uc} \cap L(G) \subseteq \left(\overline{K_1} \cap \overline{K_2}\right) \Sigma_{uc} \cap L(G)$$
$$= \left[\overline{K_1} \Sigma_{uc} \cap \overline{K_2} \Sigma_{uc}\right] \cap L(G)$$
$$= \left[\overline{K_1} \Sigma_{uc} \cap L(G)\right] \cap \left[\overline{K_2} \Sigma_{uc} \cap L(G)\right]$$
$$\subseteq \overline{K_1} \cap \overline{K_2}$$
$$\left\{\overline{K_1 \cap K_2} = \overline{\overline{K_1} \cap \overline{K_2}} = \overline{K_1} \cap \overline{K_2}\right\}$$
$$= \overline{K_1 \cap K_2}$$
$$\Rightarrow \overline{K_1 \cap K_2} \Sigma_{uc} \cap L(G) \subseteq \overline{K_1 \cap K_2}$$

2.3.4 En Büyük Kontrol Edilebilir Alt Dil

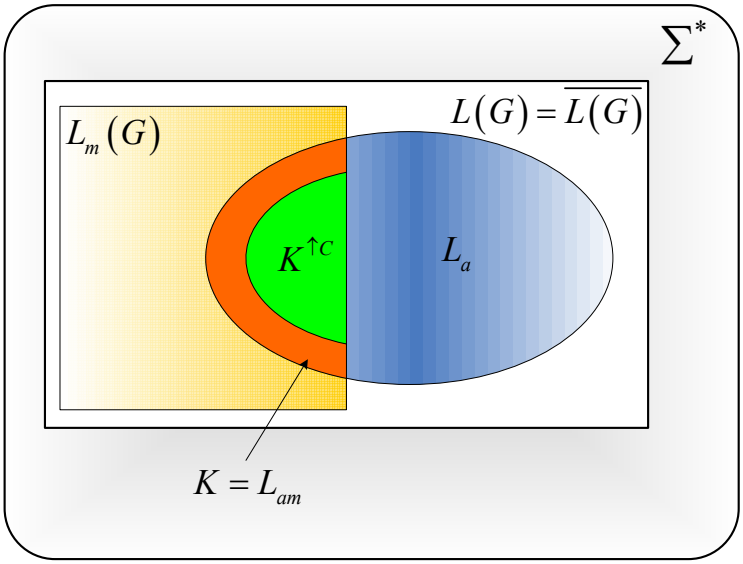
Eğer sistemden beklenenleri ifade eden K dili kontrol edilebilir değil ise, kazandırılmak istenilen davranış gerçeklenemez. Bu noktada beklenen davranışın alt kümeleri içinde en büyük kontrol edilebilir olan küme bulunmak istenilebilir. Bunun için şöyle bir alt dil sınıfı

$$\Theta_{in}(K) := \left\{L \subseteq K \text{ öyle ki } \overline{L} \Sigma_{uc} \cap L(G) \subseteq \overline{L}\right\} \quad \textbf{(2.24)}$$

tanımlayalım. $\Theta_{in}(K)$ birleşime göre kapalıdır. Sonuçta bu birleşim işlemi $\Theta_{in}(K)$ elemanları düşünüldüğünde sonlu birleşmeye genişletilebilir.

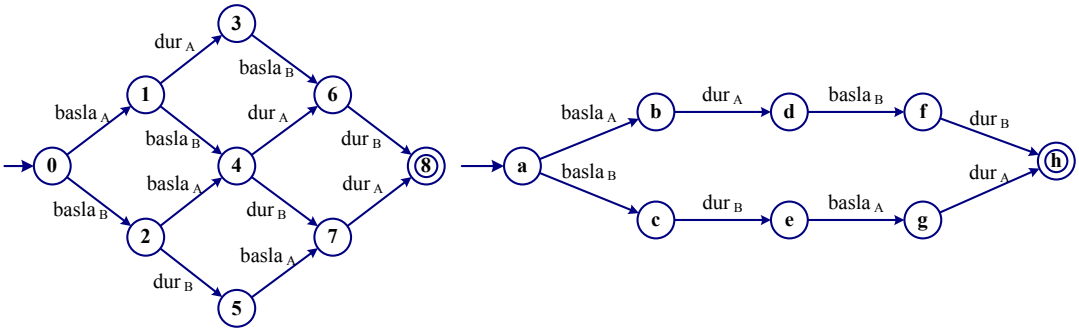
$$K^{\uparrow C} := \bigcup_{L \in \Theta_{in}(K)} L \quad \textbf{(2.25)}$$

Kontrol edilebilirlik 2.3.3.1 no.lu özellik gereği birleşim işlemine göre korunduğundan ve sonlu birleşim için $\Theta_{in}(K)$'nın tanımı gereği $\forall L_1, L_2 \in \Theta_{in}(K)$ için $L_1 \cup L_2 \in \Theta_{in}(K)$ olduğundan $K^{\uparrow C} \in \Theta_{in}(K)$'dır. O zaman bu birleşimlerle elde edilecek olan dil en büyük kontrol edilebilir alt dildir. Sonuçta K dilinin en büyük kontrol edilebilir alt dili $K^{\uparrow C}$ ile ifade edilir ve bu işlem $\uparrow$C ile gösterilir [84, 85].



Şekil 2.11 : En büyük kontrol edilebilir alt dil

2.3.4.1 Örnek



Şekil 2.12 : Sırasıyla üretim sistemini ve beklentileri ifade eden G_6 otomatı ve H_{spec1} hatırlayıcısı

Daha önce yapılan analizlerden Şekil 2.8'de verilen üretim sistemi için istenilen davranış olan $K = L_{am} = L_m\left(H_{spec1}\right)$ dilinin kontrol edilebilir olmadığı ifade edilmişti. Buna göre verilen tanımlar doğrultusunda $K^{\uparrow C}$ elde edilecektir.

$\Sigma_{uc} = \{basla_A, dur_A, dur_B\}$

$K = \{basla_A dur_A basla_B dur_B, basla_B dur_B basla_A dur_A\}$

$\overline{K} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A\}$

$basla_B \in \overline{K}$

$basla_B \Sigma_{uc} = \{basla_B basla_A, basla_B dur_A, basla_B dur_B\}$

$basla_B \Sigma_{uc} \cap L(G) = \{basla_B basla_A, basla_B dur_B\}$

$basla_B basla_A \notin \overline{K}$

$basla_B \in \overline{K}$ olmasına karşın $basla_B basla_A \notin \overline{K}$'dır. O zaman K dili altında herhangi bir kelime $basla_B$'yi önek olarak bulundurmamalıdır.

$K_1 = \{basla_A dur_A basla_B dur_B\}$ elde edilir.

$\overline{K_1} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B\}$
$\forall s \in \overline{K_1}$ için $s \sum_{uc} \bigcap L(G) \in K_1$'dir.
Sonuçta K_1 dili kontrol edilebilir bir dildir.
$\Rightarrow K^{\uparrow C} = K_1$ olduğu söylenebilir.

2.3.4.2 ↑C İşleminin Özellikleri

↑C işlemine ait sıklıkla kullanılan belli başlı bazı özellikler vardır. Bunlar maddeler halinde ispatlanmadan verilecektir.

↑C işlemi monoton bir işlemdir yani $K_1 \subseteq K_2$ ise $K_1^{\uparrow C} \subseteq K_2^{\uparrow C}$'dir. **(2.26)**

Eğer $K = \overline{K}$ ise $\overline{K^{\uparrow C}} = K^{\uparrow C} = \left(\overline{K}\right)^{\uparrow C}$ **(2.27)**

Eğer $K \neq \overline{K}$ ise $\overline{K^{\uparrow C}} \subset \left(\overline{K}\right)^{\uparrow C}$ **(2.28)**

$\left(K_1 \bigcap K_2\right)^{\uparrow C} \subseteq K_1^{\uparrow C} \bigcap K_2^{\uparrow C}$ **(2.29)**

$\left(K_1 \bigcap K_2\right)^{\uparrow C} = \left(K_1^{\uparrow C} \bigcap K_2^{\uparrow C}\right)^{\uparrow C}$ **(2.30)**

Eğer $\overline{K_1^{\uparrow C}} \bigcap \overline{K_2^{\uparrow C}} = \overline{K_1^{\uparrow C} \bigcap K_2^{\uparrow C}}$ ise $\left(K_1 \bigcap K_2\right)^{\uparrow C} = K_1^{\uparrow C} \bigcap K_2^{\uparrow C}$ **(2.31)**

$\left(K_1 \bigcup K_2\right)^{\uparrow C} \supseteq K_1^{\uparrow C} \bigcup K_2^{\uparrow C}$ **(2.32)**

2.3.4.3 ↑C İşleminin Sonucunda Elde Edilen Dilin Düzgünlüğü

Eğer $K = \overline{K}$ dili ve $L(G) = \overline{L(G)}$ dilleri düzgün diller iseler $K^{\uparrow C}$ dili de düzgün bir dildir.

Bu kurama göre eğer K ve $L(G)$ düzgün diller ise elde edilecek olan en büyük kontrol edilebilir alt dil de düzgün bir dil olacaktır. Sonuçta düzgün diller sonlu durumlu otomatlar ile ifade edilebildiklerinden $K^{\uparrow C}$ ile elde ettiğimiz çözüm de her zaman bir sonlu durumlu otomat ile ifade edilebilir.

2.3.5 En Küçük Kontrol Edilebilir Önek Kapalı Üst Dil

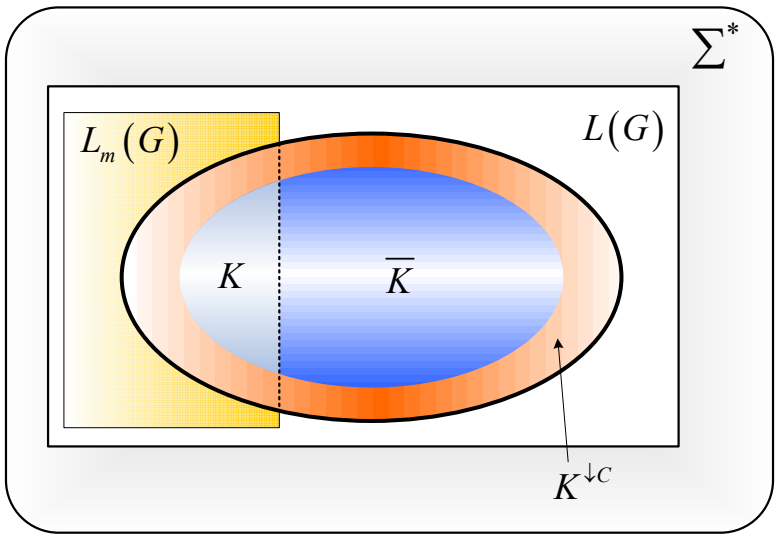
Eğer sisteme kazandırılmak istenilen davranış K kontrol edilebilir değil ise, bu istenilen davranış gerçeklenemez. Bu noktada çözüm olarak K dilinin en büyük kontrol edilebilir alt kümesi tercih edilebilir. Fakat bu çözüm her zaman K dilinin bir alt kümesidir. Bu çözümün yeterli olmaması halinde K dilini kapsayan en küçük kontrol edilebilir önek kapalı üst dili bulmak isteyebiliriz. Böylelikle K diliyle kazandırılmak istenilen davranış da korunmuş olur. Fakat diğer bir taraftan artık sistem beklenilen davranışın yanı sıra başka davranışlar da sergileyecektir. En küçük kontrol edilebilir önek kapalı üst dili bulmak için şu şekilde

$$\Theta_{out}(K) := \left\{ L \subseteq \Sigma^* \text{ öyle ki } \left(\overline{L}\Sigma_{uc} \cap L(G) \subseteq \overline{L}\right) \text{ ve } \left(L = \overline{L}\right) \text{ ve } \left(K \subseteq L \subseteq L(G)\right) \right\}$$

bir dil sınıfı tanımlayalım. $\Theta_{out}(K)$ kesişime göre kapalıdır. $\Theta_{out}(K)$ kümesinin elemanlarının kesişimi ile en küçük kontrol edilebilir önek kapalı üst dil elde edilir.

$$K^{\downarrow C} := \bigcap_{L \in \Theta_{out}(K)} L \qquad \textbf{(2.33)}$$

2.3.2.2 no.lu özellik gereği kesişim işlemine göre kontrol edilebilirlik korunmak zorunda değildir. Fakat $\Theta_{out}(K)$'un tanımı gereği sahip olduğu tüm diller önek kapalıdır yani $\forall L_1, L_2 \in \Theta_{out}(K)$ için $L_1 = \overline{L_1}$ ve $L_2 = \overline{L_2}$'dır. Tüm dillerin ön ek kapalı olması $\forall L_1, L_2 \in \Theta_{out}(K)$ için $\overline{L_1} \cap \overline{L_2} = \overline{L_1 \cap L_2}$ koşulunun sağlanmasını garanti eder; sonuçta 2.3.3.3 no.lu özellik gereği kesişim ile elde edilen sonuç dil $K^{\downarrow C}$ her zaman kontrol edilebilirdir. Bu elde edilen $K^{\downarrow C}$ diline K dilinin en küçük kontrol edilebilir önek kapalı üst dili denir [86, 87].



Şekil 2.13 : En küçük kontrol edilebilir üst dil

En kötü durumda $K^{\downarrow C} = L(G)$'dir öyle ki $L(G) \in \Theta_{out}(K)$'dir. Aynı zamanda eğer K kontrol edilebilir ise $K^{\downarrow C} = \overline{K}$'dir. En küçük kontrol edilebilir önek kapalı üst dil $K^{\downarrow C} = \overline{K} \sum_{uc}^{*} \cap L(G)$ işlemi ile elde edilebilir.

2.3.5.1 ↓C İşleminin Özellikleri

↓C işlemine ait sıklıkla kullanılan belli başlı özellikler vardır. Bunlar maddeler halinde ispatlanmadan verilecektir.

↓C işlemi monoton bir işlemdir yani $K_1 \subseteq K_2$ ise $K_1^{\downarrow C} \subseteq K_2^{\downarrow C}$'dir. **(2.34)**

$$(K_1 \cap K_2)^{\downarrow C} \subseteq K_1^{\downarrow C} \cap K_2^{\downarrow C} \quad \textbf{(2.35)}$$

Eğer $\overline{K_1} \cap \overline{K_2} = \overline{K_1 \cap K_2}$ ise $(K_1 \cap K_2)^{\downarrow C} = K_1^{\downarrow C} \cap K_2^{\downarrow C}$ **(2.36)**

$$(K_1 \cup K_2)^{\downarrow C} = K_1^{\downarrow C} \cup K_2^{\downarrow C} \quad \textbf{(2.37)}$$

2.3.5.2 Örnek

Şekil 2.8'de verilen üretim sistemi için otomat modeli ve istenilen davranışa ilişkin H_{spec1} hatırlayıcısı şekil 2.12'de verilmiştir. Verilen tanımlar doğrultusunda $K^{\downarrow C}$ elde edilecektir.

$\sum_{uc} = \{basla_A, dur_A, dur_B\}$

$K = \{basla_A dur_A basla_B dur_B, basla_B dur_B basla_A dur_A\}$

$\overline{K} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A\}$

$basla_B \in \overline{K}$

$basla_B \sum_{uc} = \{basla_B basla_A, basla_B dur_A, basla_B dur_B\}$

$basla_B \sum_{uc} \cap L(G) = \{basla_B basla_A, basla_B dur_B\}$

$basla_B basla_A \notin \overline{K}$

$basla_B \in \overline{K}$ olmasına karşın $basla_B basla_A \notin \overline{K}$'dır. O zaman $\overline{K}$ diline $basla_B basla_A$ olayı eklenir. K dilini ve $basla_B basla_A$ olayını kapsayan önek kapalı dil

$K_1 = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B basla_A, basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A\}$ şeklindedir.

$\forall s \in K_1$ için $s \sum_{uc} \cap L(G) \notin K_1$ 'dir.

$basla_B basla_A \sum_{uc} L(G) = \{basla_B basla_A dur_A, basla_B basla_A dur_B\}$

$\{basla_B basla_A dur_A, basla_B basla_A dur_B\} \notin K_1$ 'dir. O zaman

$K_2 = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B basla_A, basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A, basla_B basla_A dur_A$
$basla_B basla_A dur_B\}$

Benzer şekilde K_2 dili de kontrol edilebilir değildir. K_2 dilini kontrol edilebilir kılmak için K_2 diline $\{basla_B basla_A dur_A dur_B, basla_B basla_A dur_B dur_A\}$ kelimeleri eklenmelidir. Sonuçta

$K^{\downarrow C} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B basla_A, basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A, basla_B basla_A dur_A$
$basla_B basla_A dur_B, basla_B basla_A dur_A dur_B, basla_B basla_A dur_B dur_A\}$

elde edilir.

2.3.5.3 ↓C İşleminin Sonucunda Elde Edilen Dilin Düzgünlüğü

Eğer K dili ve $L(G)$ dilleri düzgün diller iseler $K^{\downarrow C}$ dili de düzgün bir dildir.

Bu kurama göre K ve $L(G)$ düzgün birer dil olması $K^{\downarrow C}$ dilinin düzgün bir dil olmasını garanti eder. Düzgün diller sonlu durumlu otomatlar ile gösterilebildiğinden $K^{\downarrow C}$ ile elde ettiğimiz çözüm de her zaman bir sonlu durumlu otomat ile ifade edilebilir.

2.3.6 Kilitlenmesiz Kontrol Edilebilirlik Kuramı

$G = (X, \Sigma, f, x_0, X_m)$ sonlu durumlu bir otomat, $\Sigma_{uc} \subseteq \Sigma$ kontrol edilemeyen olayları gösteren küme ve $K \subseteq L(G)$ $(K \neq \varnothing)$ beklediğimiz davranışları ifade eden dil olsun.

$L(S/G) = \overline{K}$ ve $L_m(S/G) = K$ olacak şekilde kilitlenmesiz bir S denetimsel gözetleyicisi vardır $\Leftrightarrow$ $\overline{K} \sum_{uc} \cap L(G) \subseteq \overline{K}$, $\overline{K} \cap L_m(G) = K$

K üzerindeki bu iki koşul kilitlenmesiz kontrol edilebilirlik koşulları olarak adlandırılmaktadır. İkinci koşul aynı zamanda $L_m(G)$ kapalılık olarak ta anılmaktadır.

İspat – Kontrol edilebilirlik kuramı her iki yönde de ispatlanacaktır.

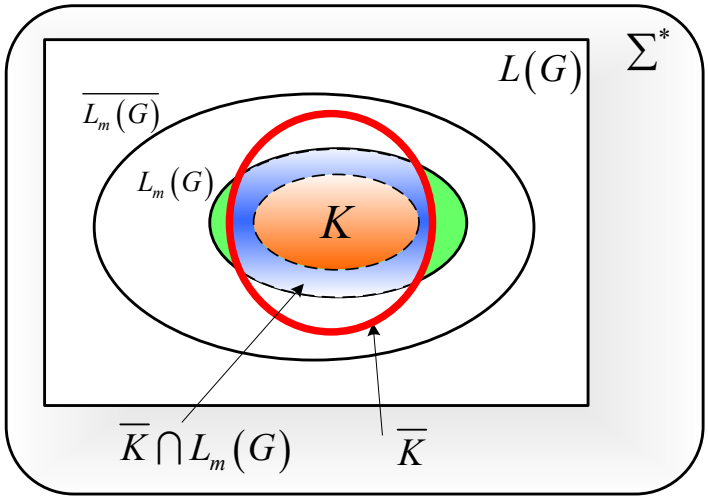
Bunu sağlayan bir $S(s)=\left[\Sigma_{uc}\cap\Gamma\left(f(x_0,s)\right)\right]\cup\left\{\sigma\in\Sigma_c : s\sigma\in\overline{K}\right\}$ denetimsel gözetleyicisi vardır.

Kontrol edilebilirlik teoreminin ispatından $L(S/G)=\overline{K}$

$$L_m(S/G)=L(S/G)\cap L_m(G)=\overline{K}\cap L_m(G)=K$$
$$\Rightarrow(Gerek\ Koşul)$$
$$L_m(S/G)=L(S/G)\cap L_{\ m}(G)$$
$$=\overline{K}\cap L_{\ m}(G)=K$$

Bu kapalılık koşuludur. Kontrol edilebilirlik koşulu 2.3.3'de verilmiştir. Sonuç olarak istenilen davranışın kontrol edilebilir ve kilitlenmesiz olması isteniyorsa kuramda verilen iki koşul da sağlanmalıdır.

Şekil 2.14'de $L_m(G)$ kapalı olmayan bir dil verilmiştir. Sonuçta $L_m(G)$ kapalı olmayan dil aynı zamanda kilitlenmeli bir dildir.



Şekil 2.14 : $L_m(G)$ kapalı olmayan bir dil örneği

2.4 Denetimsel Kontrol Kuramı ile İlgili Temel Problemler

Ayrık olay sistemlerine ait davranışın düzgün diller ile ifade edilmesi halinde bu davranışı modelleyebilmek için genellikle sonlu durumlu otomatlar kullanılır. Sistemden beklenilen davranış K ile ifade edilmesi halinde istenilen davranışın yapısına göre olası çözümler genellikle $\varnothing\subseteq K^{\uparrow C}\subseteq K\subseteq\overline{K}\subseteq K^{\downarrow C}\subseteq L(G)$ ile ifade

edilir. Denetlenmek istenen sistem ve beklenen davranışa bağlı olarak farklı denetimsel gözetleyici problemleri söz konusudur.

2.4.1 Temel Denetimsel Gözetleyici Problemi

$G=\left(X,\Sigma,f,x_0,X_m\right)$ sonlu durumlu bir otomat, $\Sigma_{uc}\subseteq\Sigma$ kontrol edilemeyen olayları gösteren küme ve $L_a=\overline{L_a}\subseteq L\left(G\right)$ kabul edilebilir dil olmak üzere bir S denetimsel gözetleyicisi bulalım öyle ki

- $L\left(S/G\right)\subseteq L_a$
- $\forall L\left(S_{diğer}/G\right)\subseteq L_a$ için $L\left(S_{diğer}/G\right)\subseteq L\left(S/G\right)$

koşulları sağlansın. Burada $S_{diğer}$, S haricinde herhangi bir denetimsel gözetleyici olsun. Bu iki koşulu sağlayan denetimsel gözetleyiciye ait dil şu şekildedir:

$$L\left(S/G\right)=L_a^{\uparrow C} \tag{2.38}$$

Problem tanımındaki ikinci koşul aslında bu problem için optimallik koşulu olarak ta algılanabilir. Bu koşul L_a diline en yakın kontrol edilebilir dilin bulunmasına karşılık gelir. Bu denetimsel gözletleyiciye ait çözüm En Az Kısıtlamalı Çözümü (EAKÇ) olarak adlandırılır. Sonuçta $L_a^{\uparrow C}$ dilini işaretleyecek her zaman bir S hatırlayıcısı vardır. Bu çözümde kilitlenme önemsenmez. Amaç $L_a=\overline{L_a}\subseteq L\left(G\right)$ dilini kontrol edilebilir en büyük kısmını elde etmektir.

2.4.2 Temel Denetimsel Gözetleyici Probleminin Eşleniği

Temel denetimsel gözetleyici probleminin çözümünün $L_r\subseteq L_a^{\uparrow C}$ şeklinde bir sınır değeri vardır. Öyle ki elde edilen çözüm bu sınır değeri kapsamıyor ise uygun bir çözüm olmaktan uzaktır.

O zaman $G=\left(X,\Sigma,f,x_0,X_m\right)$ sonlu durumlu bir otomat, $\Sigma_{uc}\subseteq\Sigma$ kontrol edilemeyen olayları gösteren küme ve en az beklentileri içeren dil $L_r\subseteq L\left(G\right)$ olmak üzere

- $L\left(S/G\right)\supseteq L_r$
- $\forall L\left(S_{diğer}/G\right)\supseteq L_r$ için $L\left(S_{diğer}/G\right)\supseteq L\left(S/G\right)$

koşullarını sağlayan bir S denetimsel gözetleyicisi uygun çözümdür. Bu iki koşulu sağlayan denetimsel gözetleyiciye ait dil

$$L(S/G) = L_r^{\downarrow C} \tag{2.39}$$

şeklindedir. ↓C işlemi gereği $L_r^{\downarrow C}$ dilini işaretleyecek her zaman bir S hatırlayıcısı vardır. Bu problemdeki ikinci koşul optimallik koşuludur zira aranan çözüm L_r dilini kapsamalı ayrıca L_r'yi kapsayan kontrol edilebilir bütün diller tarafından da kapsanmalıdır. Bir diğer değişle L_r dilini kapsayan en küçük kontrol edilebilir dil aranmaktadır. Bu da 2.3.5'de önerilen çözümle örtüşmektedir. Aynı zamanda L_r önek kapalı olmak zorunda değildir ve genellikle $L_m(G)$'nin bir alt kümesi olarak verilir. Eğer $L_r = L_{am}$ olarak alınırsa temel gözetleyici probleminin eşleniğinin çözümü $L(S/G) = L_{am}^{\downarrow C}$ olacaktır. Elde edilen bu dil Tam Başarımlı Çözüm (TBÇ) olarak adlandırılır.

2.4.3 Kilitlenmesiz Temel Denetimsel Gözetleyici Problemi

$G = (X, \Sigma, f, x_0, X_m)$ sonlu durumlu bir otomat, $\Sigma_{uc} \subseteq \Sigma$ kontrol edilemeyen olayları gösteren küme ve kabul edilebilir işaretli dil $L_{am} \subseteq L_m(G)$ olmak üzere bir S denetimsel gözetleyicisi bulalım öyle ki

- $L_m(S/G) \subseteq L_{am}$
- $\forall L_m(S_{diğer}/G) \subseteq L_{am}$ için $L_m(S_{diğer}/G) \subseteq L_m(S/G)$

koşullarını sağlasın. Bu problem L_{am}'nin, $L_m(G)$ kapalı olduğu kabulü altında çözülmüştür. Aynı zamanda ikinci koşul gereğince elde edilecek çözüm kilitlenmesiz ve işaretli dil için en az kısıtlamaya sahip olan çözümdür. Literatürde bu çözüm En Az Kısıtlamalı Kilitlenmesiz Çözüm (EAKKÇ) olarak adlandırılır ve denetimsel gözetleyiciye ait diller

$$L(S/G) = \overline{L_{am}^{\uparrow C}} \tag{2.40}$$

$$L_m(S/G) = L_{am}^{\uparrow C} \tag{2.41}$$

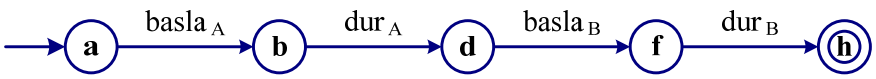
şeklindedir. L_{am}'nin $L_m(G)$ kapalı olması $L_{am}^{\uparrow C}$'nin de $L_m(G)$ kapalı olmasını garanti eder. Sonuçta her zaman $\overline{L_{am}^{\uparrow C}}$ dilini üretecek bir S denetimsel gözetleyicisi elde edilebilir.

2.4.4 Örnek

Şekil 2.8'de verilen üretim sistemi için istenilen davranışa karşılık gelen otomat Şekil 2.10'daki gibidir. Burada istenilen davranış olan $K = L_{am} = L_m(H_{spec1})$,

$\Sigma_{uc} = \{basla_A, dur_A, dur_B\}$ için kontrol edilebilir değildir. Sisteme ait istenilen davranış kümesine göre TBÇ ile EAKKÇ sırasıyla şu şekildedir:

En az kısıtlamalı kilitlenmesiz çözüm $L(S/G) = \overline{L_{am}^{\uparrow C}}$'dir. Buna göre $L_{am} = \{basla_A dur_A basla_B dur_B, basla_B dur_B basla_A dur_A\}$ olduğu düşünülürse $L_{am}^{\uparrow C} = \{basla_A dur_A basla_B dur_B\}$ olur ve çözüm ise $L(S/G) = \overline{L_{am}^{\uparrow C}} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B\}$'dir. Bu dili üreten otomat ise Şekil 2.15'deki gibidir.



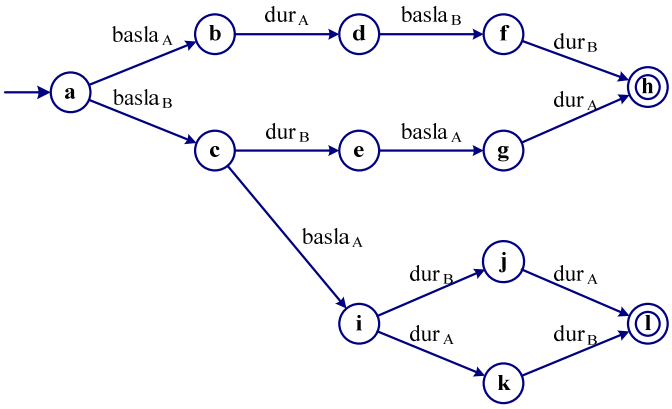
Şekil 2.15 : $L(S/G) = \overline{L_{am}^{\uparrow C}}$ otomat modeli

Şekil 2.15'den de görüldüğü üzere $basla_A$ olayının oluşmasıyla sistemin devreye alınması halinde kilitlenmesiz olarak sistem hareketini sonlandıracaktır.

Benzer bir şekilde tam başarımlı çözüm ise $L(S/G) = L_{am}^{\downarrow C}$'dir. Bu çözüm işaretli dilin en küçük kontrol edilebilir önek kapalı dilini bulmak suretiyle elde edilir. $\Sigma_{uc} = \{basla_A, dur_A, dur_B\}$ ve $L_{am} = \{basla_A dur_A basla_B dur_B, basla_B dur_B basla_A dur_A\}$ olduğu düşünüldüğünde tam başarımlı çözüm aşağıdaki gibidir.

$L_{am}^{\downarrow C} = \{\varepsilon, basla_A, basla_A dur_A, basla_A dur_A basla_B, basla_A dur_A basla_B dur_B, basla_B,$
$basla_B basla_A, basla_B dur_B, basla_B dur_B basla_A, basla_B dur_B basla_A dur_A, basla_B basla_A dur_A$
$basla_B basla_A dur_B, basla_B basla_A dur_A dur_B, basla_B basla_A dur_B dur_A\}$

Bu dili üreten otomat Şekil 2.16'da verilmiştir.



Şekil 2.16 : $L(S/G) = L_{am}^{\downarrow C}$ dilini üreten otomat

Şekil 2.16 incelendiğinde görülecektir ki eğer istenilen işaretli dilin tam olarak üretilmesi istenildiğinde, sistemin kontrol altında ürettiği dile istenmeyen kelimeler de eklenmiştir. Burada $basla_B$ olayının oluşmasının ardından $basla_A$ olayı da oluşabilir öyle ki bu da sisteme ait iki manipülatörün çarpışması anlamına gelir. Ardından dur_A ve dur_B olaylarıyla manipülatörlerin işleri tamamlanmış olur. Bu da modelde $basla_B basla_A dur_B dur_A$ ve $basla_B basla_A dur_A dur_B$ kelimeleri ile ifade edilmiştir. Burada bu iki kelime de işaretli kelimeler olmasına rağmen üretilmesi istenilmeyen kelimelerdir. Fakat kontrol edilebilirlik koşulu gereği istenilen işaretli dili eksiksiz üretebilmek için kontrol altındaki dilin üretebileceği bu kelimeler eklenmiştir. Sonuçta başlangıç durumunda $basla_B$ olayı ile sistemin çalıştırılması halinde, sistemin istenmeyen bir davranış göstermesi mümkündür. Burada bir kilitlenme olmamakla beraber sistem beklenilen davranışın dışında da bir çıkış üretebilmektedir.

2.4.5 Temel Kilitlenebilir Denetimsel Gözetleyici Problemi

Sistemden beklenilen davranış genellikle $L_{am} \subseteq L_m(G)$ işaretli dili ile ifade edilir. Bu işaretli kelimelerin işaretsiz kelimeler ile uygun bir şekilde üretilebilir hale gelmesinden ise L_a elde edilir, öyle ki genellikle $\overline{L_{am}} \subseteq L_a$'dir. Böyle olması halinde $L_a \setminus \overline{L_{am}}$ kümesine ait olan kelimeler hiçbir zaman bir istenilen işaretli kelimeyle sonlanamayacak ve sistemi kilitlenmeye götüreceklerdir. Bu noktada en az kısıtlamalı kilitlenmesiz çözüme ait denetimsel gözetleyici tercih edilebilir. Fakat çözümün ürettiği işaretli dil beklenilen davranıştan çok uzak ise kilitlenme koşulunu belli bir oranda serbest bırakmak gerekir. Bu tür denetimsel gözetleyicilerin tercih edilmesi ile ilgili probleme temel kilitlenebilir denetimsel gözetleyici problemi (KDGP) denir. Özellikle kilitlenmenin rahatlıkla gözlemlenebildiği ve çözülebildiği sistemlerde kilitlenebilir denetimsel gözetleyiciler yukarıda belirtilen sebeplerden ötürü tercih edilmektedir. Temel kilitlenebilir denetimsel gözetleyici problemin tanımı ise şu şekildedir:

$G = (X, \Sigma, f, x_0, X_m)$ sonlu durumlu bir otomat, $\Sigma_{uc} \subseteq \Sigma$ kontrol edilemeyen olayları gösteren küme ve kabul edilebilir işaretli dil $L_{am} \subseteq L_m(G)$ olmak üzere bir S denetimsel gözetleyicisi bulalım, öyle ki

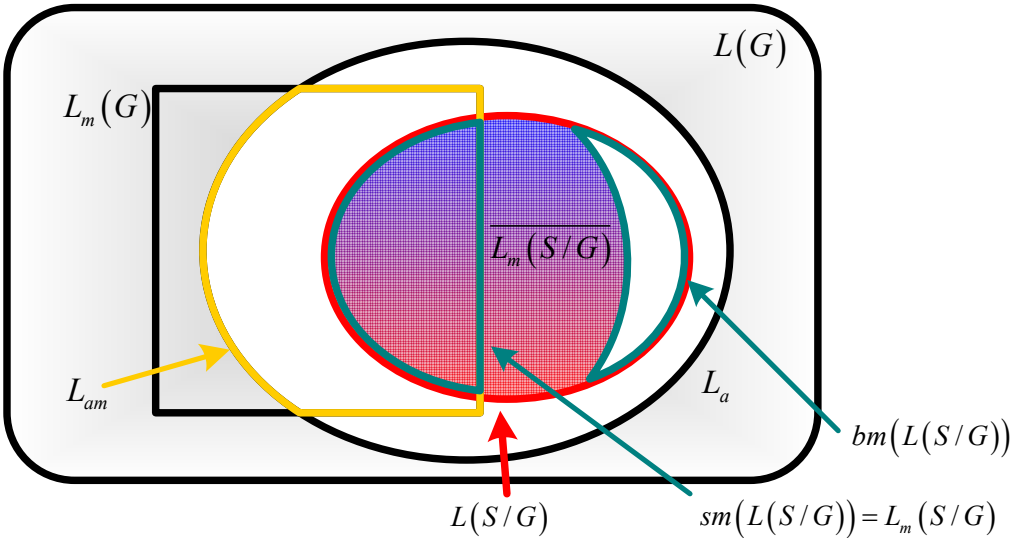
- $L(S/G) \subseteq L_a$ ve $L_m(S/G) \subseteq L_{am}$
- $L(S/G)$ şu iki küme için pareto optimal bir çözüm olsun [88]
    - $bm(S) := L(S/G) \setminus \overline{L_m(S/G)}$

- $sm(S) := L_m(S/G)$

Burada $bm(S)$ kontrol altındaki sistemin sahip olduğu kilitlenmeleri ve $sm(S)$ ise sistemin kontrol altında ürettiği işaretli kelimeleri içeren kümedir. Problem tanımı gereği iki sınır değer çözüme sahiptir. Bunlar sırasıyla en az kısıtlamalı kilitlenmesiz çözüm ve tam başarımlı çözümdür. En az kısıtlamalı kilitlenmesiz çözümde $bm(S) = \varnothing$ iken $sm(S)$ en az işaretli kelimeye sahiptir. Diğer bir taraftan tam başarımlı çözümde ise $sm(S) = L_{am}$'dir fakat $bm(S)$ en yüksek kelime sayısına sahiptir. Problemin çözümünü oluşturabilecek tüm olası çözümler en az kısıtlamalı kilitlenmesiz çözüm ile tam başarımlı çözüm arasındadır. Fakat bir dilin bu iki limit çözüm arasında olması problemin olası çözümlerinden birisi olacağı anlamına gelmez. Aynı zamanda önek kapalı ve kontrol edilebilir olmalıdır. Buna göre temel kilitlenebilir denetimsel gözetleyici problemi için tüm çözümleri içeren dil ailesi kümesi

$$L_{cand} := \left\{ \Gamma : \left(\overline{L_{am}^{\uparrow C}} \subseteq \Gamma \subseteq L_{am}^{\downarrow C} \right) \wedge \left(\Gamma = \overline{\Gamma} \right) \wedge \left(\overline{\Gamma} \Sigma_{uc} \cap L(G) \subseteq \overline{\Gamma} \right) \right\} \qquad \textbf{(2.42)}$$

olarak verilir.

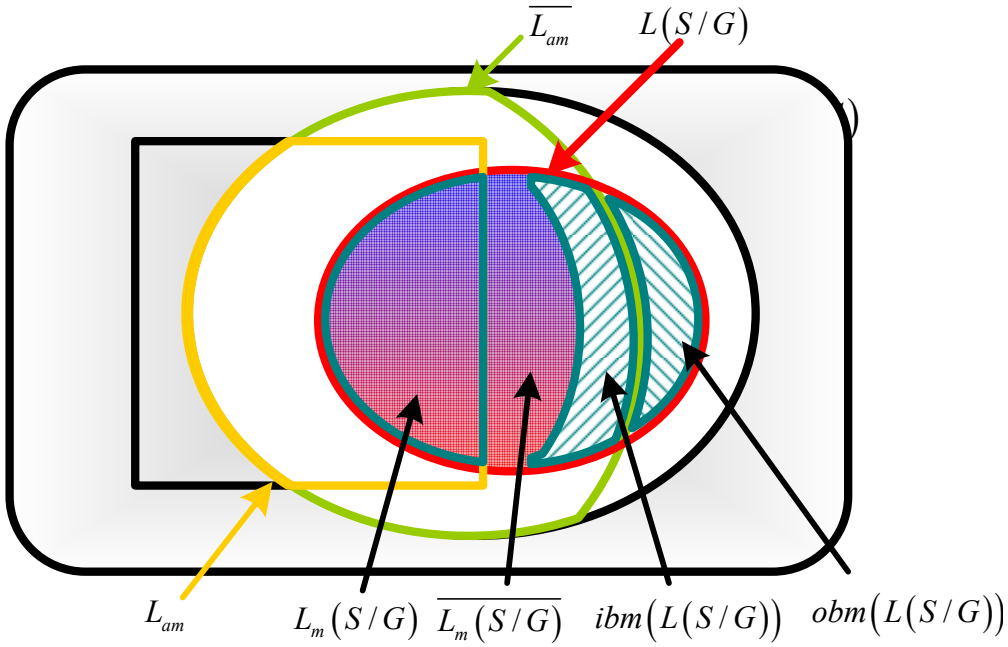


Şekil 2.17 : Temel kilitlenebilir denetimsel gözetleyici problemi

Şekil 2.17'de $L(S/G) \in L_{cand}$ için $bm(S)$ ve $sm(S)$ gösterilmiştir. Burada $L(S/G) \neq \overline{L_{am}^{\uparrow C}}$ ve $L(S/G) \neq L_{am}^{\downarrow C}$'dir.

Aynı zamanda $\forall L(S/G) \in L_{cand}$ için dış kilitlenme ve iç kilitlenme olmak üzere iki farklı kilitlenme söz konusudur, öyle ki $bm(S) = obm(S) \cup ibm(S)$'dir. Bu iki farklı kilitlenme türü için tanımlar sırasıyla şu şekildedir:

- $obm(S) := L(S/G) \setminus \overline{L_{am}}$
- $ibm(S) := \left(L(S/G) \cap \overline{L_{am}}\right) \setminus \overline{L_m(S/G)}$



Şekil 2.18 : $L(S/G) \in L_{cand}$ için iç ve dış kilitlenme

Eğer $\exists L(S/G) \in L_{cand}$ için $bm(S) \neq \varnothing$ ise kontrol altındaki sisteme ait kilitlenmelerin kaynağı iki türlü olabilir. $L(S/G)$ dili $L_a \setminus \overline{L_{am}}$ kümesine ait elemanlar içeriyor ise bunlar dış kilitlenme olarak adlandırılır. Bu kümeye ait elemanlar aslında L_{am} kümesine ait kelimeleri üretebilmek için kontrol edilebilirlik ve nedensellik koşulları gereği $L(S/G)$ diline eklenen kelimelerdir. Diğer bir taraftan $L(S/G)$ dili öyle kilitlenmeye neden olan kelimeler içeriyor olabilir ki bunlar $\overline{L_{am}}$ kümesinin elemanı olmasına rağmen $L(S/G)$ dilindeki herhangi bir istenilen işaretli kelimenin öneki değildir. Bu tarz kilitlenmeler de iç kilitlenme olarak adlandırılır. Şekil 2.18'de $\exists L(S/G) \in L_{cand}$ için hem iç hem de dış kilitlenme grafik üzerinden gösterilmiştir.

2.4.5.1 Önerme

i) Eğer $L(S/G)$ ve L_{am} çakışmasız diller ise $ibm(S) = \varnothing$ olur.

İspat – Çakışmasız diller olduğundan $\overline{L(S/G)} \cap \overline{L_{am}} = \overline{L(S/G) \cap L_{am}}$'dir. Nedensellik koşulu gereği her zaman $L(S/G) = \overline{L(S/G)}$ sağlanır.

$$ibm(S) = \left(L(S/G) \cap \overline{L_{am}}\right) \setminus \overline{L_m(S/G)}$$
$$= \left(L(S/G) \cap \overline{L_{am}}\right) \setminus \overline{L(S/G) \cap L_{am}}$$
$$= \left(L(S/G) \cap \overline{L_{am}}\right) \setminus \left(\overline{L(S/G)} \cap \overline{L_{am}}\right)$$
$$= \left(\overline{L(S/G)} \cap \overline{L_{am}}\right) \setminus \left(\overline{L(S/G)} \cap \overline{L_{am}}\right)$$
$$= \varnothing$$

ii) Eğer $L(S/G) \subseteq \overline{L_{am}}$ ise $obm(S) = \varnothing$ olur.

İspat – $obm(S) = L(S/G) \setminus \overline{L_{am}}$ ve $L(S/G) \subseteq \overline{L_{am}}$ olduğu düşünülürse her zaman ’dir. Sonuçta eğer $L(S/G) = \left(\overline{L_{am}}\right)^{\uparrow C}$ ise her zaman $obm(S) = \varnothing$ ’dır. Diğer bir taraftan $L(S/G) = L_{am}^{\downarrow C}$ ise $ibm(S) = \varnothing$ ’dir.

Buna göre kilitlenebilir denetimsel gözetleyici için dört farklı çözüm önerilmiştir.

2.4.5.2 Kilitlenebilir Denetimsel Gözetleyici Problemi İçin Önerilen Çözümler

- $obm(S) = ibm(S) = \varnothing$ ve $sm(S)$’nin en fazla işaretli kelimeye sahip olduğu çözüm:

Bu koşullar için en az kısıtlamalı kilitlenmesiz çözüm en uygundur. Zira bu çözümde hiçbir kilitlenmenin oluşması istenmemekte ve aynı zamanda istenilen işaretli dile ait olabildiğince çok kelimenin de üretilmesi istenmektedir. Sonuçta bu koşulları sağlayan çözüm

$L(S_{c1}/G) = \overline{L_{am}^{\uparrow C}}$ ve $L_m(S_{c1}/G) = L_{am}^{\uparrow C}$ şeklindedir. En az kısıtlamalı kilitlenmesiz çözüm için $sm(S_{c1}) = L_{am}^{\uparrow C}$ ve $obm(S) = ibm(S) = \varnothing$ ’dir.

- $obm(S) = \varnothing$ ve $sm(S)$’nin en fazla işaretli kelimeye sahip olduğu çözüm:

Bu koşullar için en uygun çözüm 2.4.5.1 no.lu önermenin ii. maddesi gereğince $L(S_{c2}/G) = \left(\overline{L_{am}}\right)^{\uparrow C}$ ’dir. Dikkat edilirse bu çözümde iç kilitlenmenin ne olacağı önemsenmemiştir. Bu çözüm için $sm(S_{c2}) = \left(\overline{L_{am}}\right)^{\uparrow C} \cap L_{am}$ olacaktır. Bu çözüm En Az Kısıtlamalı Dış Kilitlenmesiz Çözüm (EAKDKÇ) olarak adlandırılır.

- $ibm(S) = \varnothing$ ve $sm(S)$’nin en fazla işaretli kelimeye sahip olduğu çözüm:

Bu koşullar için en uygun çözüm 2.4.5.1 no.lu önermenin i. maddesi gereğince $L_a^{\uparrow C} \cap L_{am}^{\downarrow C}$ dilinin L_{am} ’ye göre çakışmasız en büyük kontrol edilebilir önek kapalı alt

dilidir. Bu çözümde dış kilitlenmenin miktarı ne olacağı önemsenmemiştir. Bu çözüm En Az Kısıtlamalı İç Kilitlenmesiz Çözüm (EAKİKÇ) olarak adlandırılır.

- $ibm(S)$ ve $obm(S)$'yi dikkate almadan $sm(S)$'nin en fazla işaretli kelimeye sahip olduğu çözüm:

Bu koşullar için en uygun çözüm $L(S_{c4}/G) = L_a^{\uparrow C} \cap L_{am}^{\downarrow C}$'dir. Bu çözüm en az kısıtlamalı çözüm olarak adlandırılmaktadır. Bu çözüm için $sm(S_{c4}) = L_a^{\uparrow C} \cap L_{am}$'dir.

2.4.5.3 Lemma

Eğer $L \in L_{cand}$ ve $\tilde{L} = (L \cap L_{am})^{\downarrow C} \subset L$ ise $bm(\tilde{L}) \subset bm(L)$ ve $sm(\tilde{L}) = sm(L)$'dir.

İspat – L dili için $sm(L) = L \cap L_{am}$ ve $bm(L) = L \setminus \overline{L \cap L_{am}}$'dir. Aynı zamanda $sm(\tilde{L}) = (L \cap L_{am})^{\downarrow C} \cap L_{am}$. ↓C işleminin özelliği olarak $(L \cap L_{am})^{\downarrow C} \cap L_{am} = L \cap L_{am}$ eşittir. Sonuç olarak $sm(\tilde{L}) = sm(L)$'dir. Diğer bir taraftan

$$bm(\tilde{L}) = (L \cap L_{am})^{\downarrow C} \setminus \overline{(L \cap L_{am})^{\downarrow C} \cap L_{am}}$$
$$= (L \cap L_{am})^{\downarrow C} \setminus \overline{L \cap L_{am}}$$

Aynı zamanda $(L \cap L_{am})^{\downarrow C} \supseteq \overline{L \cap L_{am}}$ dır.

O zaman $L = (L \cap L_{am})^{\downarrow C} \cup \Phi$ şeklinde yazılabilir. Öyle ki $(L \cap L_{am})^{\downarrow C} \cap \Phi = \varnothing$ dir.

$$bm(L) = L \setminus \overline{L \cap L_{am}}$$
$$= \left((L \cap L_{am})^{\downarrow C} \cup \Phi\right) \setminus \overline{L \cap L_{am}}$$
$$= \left((L \cap L_{am})^{\downarrow C} \setminus \overline{L \cap L_{am}}\right) \cup \Phi \supset (L \cap L_{am})^{\downarrow C} \setminus \overline{L \cap L_{am}} = bm(\tilde{L})$$

Sonuçta $bm(L) \supset bm(\tilde{L})$ dir.

$\tilde{L} = (L \cap L_{am})^{\downarrow C}$ işlemi ile L dili üzerinde yapılan iyileştirme sonucunda istenilen işaretli kelimelerde bir değişim olmaksızın kilitlenmelerde bir azalmaya gidilebilir. İşlem sonucunda bir iyileştirmeye gidilmiş ise atılan kilitlenmeler iç kilitlenmelerden de olabilir dış kilitlenmelerden de olabilir.

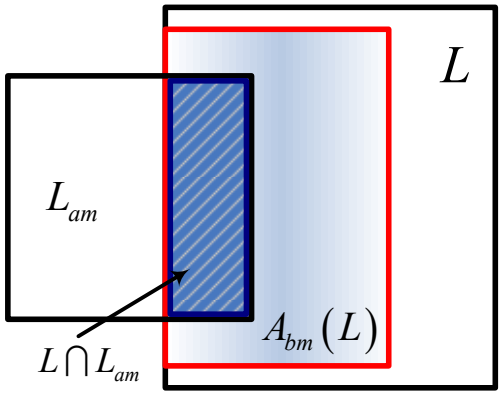
2.4.5.4 Kilitlenmeyi Azaltmak

2.4.5.3 numaralı lemmaya göre $L \in L_{cand}$ için başarımda bir değişime neden olmaksızın kilitlenmeye sebebiyet veren kelimelerde bir azalma elde

edilebilmektedir. Bu lemmanın sonucu $A_{bm}(L): L_{cand} \to L_{cand}$ olacak şekilde $A_{bm}(L)$ operatörü ile ifade edilir öyle ki,

$$A_{bm}(L) := (L \cap L_{am})^{\downarrow C} \tag{2.43}$$

Aynı zamanda ↓C işleminin bir sonucu olarak $A_{bm}(A_{bm}(L)) = A_{bm}(L)$'dir. Şekil 2.19'da $A_{bm}(L)$ işlemi sonucunda elde edilen dil gösterilmiştir.



Şekil 2.19 : $A_{bm}(L)$ işlemi sonunda elde edilen dil

2.4.5.5 Lemma

$L \in L_{cand}$ olmak üzere $\hat{L} = L \cup K_{\max}$ ve $\widehat{L_m} = (L \cap L_{am}) \cup K_{\max}$ olacak şekilde gerçekleyen bir denetimsel gözetleyici vardır öyle ki $K_{\max} := \left\{K : (K \subseteq L_{am} \setminus L) \wedge \left(K^{\downarrow C} \subseteq L \cup L_{am}\right)\right\}^{\uparrow C}$'dır. Eğer $K_{\max} \neq \varnothing$ ise $bm(\hat{L}) = bm(L)$ ve $sm(\hat{L}) \supset sm(L)$'dir

İspat – L dili için $sm(L) = L \cap L_{am}$ ve $bm(L) = L \setminus \overline{L \cap L_{am}}$'dir. Aynı zamanda

$sm(\hat{L}) = (L \cap L_{am}) \cup K_{\max}$ ve $sm(L) = (L \cap L_{am})$'dir.

$K_{\max} \neq \varnothing$ olduğundan her zaman $sm(\hat{L}) \supset sm(L)$ olur.

$$bm(\hat{L}) = \hat{L} \setminus \overline{\hat{L} \cap L_{am}}$$
$$= (L \cup K_{\max}) \setminus \overline{(L \cap L_{am}) \cup K_{\max}}$$
$$= (L \cup K_{\max}) \setminus \left(\overline{L \cap L_{am}} \cup \overline{K_{\max}}\right)$$

$K_{\max}$'dan $\overline{K_{\max}}$ çıkarttığımızda her zaman $K_{\max}$'ın gideceği düşünülgünde

$= L \setminus \left(\overline{L \cap L_{am}} \cup \overline{K_{\max}} \right) \subseteq L \setminus \overline{L \cap L_{am}} = bm(L)$

Sonuçta her zaman $bm\left(\hat{L}\right) \subseteq bm(L)$ olacaktır.

$obm\left(\hat{L}\right) = \left(L \cup K_{\max}\right) \setminus \overline{L_{am}}$

Her zaman $K_{\max} \subseteq L_{am}$ olduğundan

$obm\left(\hat{L}\right) = L \setminus \overline{L_{am}} = obm(L)$'e eşit olur.

$ibm\left(\hat{L}\right) = \left(\hat{L} \cap \overline{L_{am}}\right) \setminus \overline{\widehat{L_m}}$

$= \left(\left(L \cup K_{\max}\right) \cap \overline{L_{am}}\right) \setminus \left(\overline{L \cap L_{am}} \cup \overline{K_{\max}}\right)$

$= \left(L \cap \overline{L_{am}}\right) \setminus \left(\overline{L \cap L_{am}} \cup \overline{K_{\max}}\right)$

$\subseteq \left(L \cap \overline{L_{am}}\right) \setminus \left(\overline{L \cap L_{am}}\right) = ibm(L)$

Sonuç olarak $K_{\max} \neq \varnothing$ ve $\hat{L} = L \cup K_{\max}$ ise $\hat{L}$ dili her zaman $sm\left(\hat{L}\right) \supset sm(L)$ koşulunu sağlamaktadır ve bunun yanı sıra $\hat{L}$ diliyle de iç kilitlenmede bir azalmaya gidilebilmektedir. Diğer bir değişle $\hat{L} = L \cup K_{\max}$ işlemi ile $obm\left(\hat{L}\right) = obm(L)$ iken $ibm\left(\hat{L}\right) \subseteq ibm(L)$ olabilmektedir.

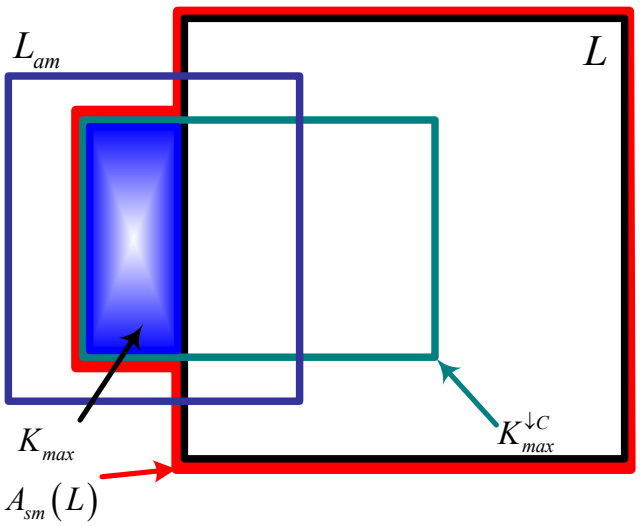
2.4.5.6 Başarımı Arttırmak

2.4.5.5 numaralı lemmaya göre $L \in L_{cand}$ için $K_{\max} \neq \varnothing$ ise L diline ait kilitlenmelerde bir artışa neden olmaksızın kontrol altındaki sistemin ürettiği istenilen işaretli kelimeleri arttırmak mümkündür.

$K_{\max} := \sup\left\{K : \left(K \subseteq L_{am} \setminus L\right) \wedge \left(K^{\uparrow C} \subseteq L \cup L_{am}\right)\right\}$ olmak üzere 2.4.5.5 numaralı lemmanın sonucu $A_{sm}(L) : L_{cand} \to L_{cand}$ işlemi ile gösterilebilir öyle ki,

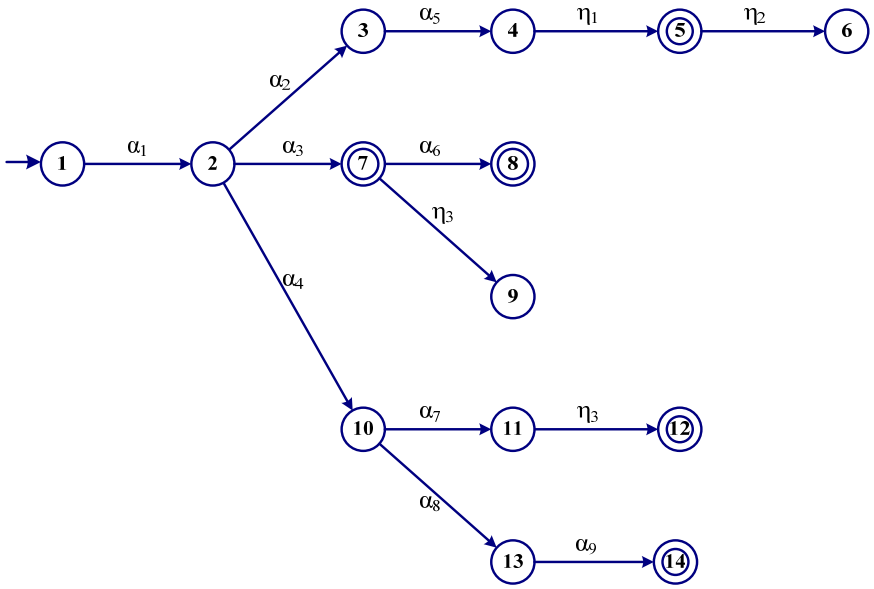
$$A_{sm}(L) := L \cup K_{\max} \tag{2.44}$$

$\uparrow$C işleminin bir sonucu olarak her zaman $A_{sm}\left(A_{sm}(L)\right) = A_{sm}(L)$'dir. Şekil 2.20'de K_{max} kümesi ve $A_{sm}(L)$ işlemi sonucunda elde edilen dil gösterilmiştir. Görüldüğü gibi $K_{max}^{\uparrow C} \setminus L_{am}$ her zaman L nin alt kümesidir. 2.4.5.5 numaralı lemmaya göre ve Şekil 2.20'de gösterildiği gibi K_{max}'ın L diline eklenmesi ile hiçbir zaman L diline yeni bir kilitlenme gelmemiş olur.



Şekil 2.20 : $A_{sm}(L) = L \cup K_{\max}$ işleminden sonra elde edilen dil

2.4.5.7 Örnek



Şekil 2.21 - G_7 otomatı

Şekil 2.21'deki G_7 otomatını düşünelim. Otomata ait kontrol edilemeyen olaylar kümesi $\Sigma_{uc} = \{\eta_1, \eta_2, \eta_3\}$ olsun. Kabul edilebilir işaretli kelimeler kümesi ise $L_{am} = \{\alpha_1\alpha_2\alpha_5\eta_1, \alpha_1\alpha_3, \alpha_1\alpha_3\alpha_6, \alpha_1\alpha_4\alpha_8\alpha_9\}$ olsun. Denetimsel gözetleyici ile birlikte üretilen dil ise $L(S/G_7) = \{\overline{\alpha_1\alpha_2\alpha_5\eta_1\eta_2}, \overline{\alpha_1\alpha_4\alpha_8\alpha_9}, \overline{\alpha_1\alpha_3\eta_3}, \alpha_1\alpha_4\alpha_7\}$ olsun.

Buna göre $bm(L(S/G_7)) = \{\alpha_1\alpha_2\alpha_5\eta_1\eta_2, \alpha_1\alpha_3\eta_3, \alpha_1\alpha_3\alpha_7\}$ ve

$sm\left(L\left(S/G_7\right)\right)=\left\{\alpha_1\alpha_2\alpha_5\eta_1,\alpha_1\alpha_4\alpha_8\alpha_9,\alpha_1\alpha_3\right\}$'dir. G_7 otomatı için en az kısıtlamalı kilitlenmesiz çözüm ve tam başarımlı çözüm ise sırasıyla $L\left(S_{sc1}/G_7\right)=\left\{\overline{\alpha_1\alpha_4\alpha_8\alpha_9}\right\}$ ve $L\left(S_{sc4}/G_7\right)=\left\{\overline{\alpha_1\alpha_2\alpha_5\eta_1\eta_2},\overline{\alpha_1\alpha_4\alpha_8\alpha_9},\overline{\alpha_1\alpha_3\eta_3},\alpha_1\alpha_3\alpha_6\right\}$'dir.

$L\left(S/G_7\right)$ dilene ait kilitlenme miktarını azaltabilmek için $L\left(S/G_7\right)$ dilini A_{bm} işlemine tabii tuttuğumuzda $A_{bm}\left(L\left(S/G_7\right)\right)=\left\{\overline{\alpha_1\alpha_2\alpha_5\eta_1\eta_2},\overline{\alpha_1\alpha_4\alpha_8\alpha_9},\overline{\alpha_1\alpha_3\eta_3}\right\}$ dili elde edilir. Bu elde edilen dil için $bm\left(A_{bm}\left(L\left(S/G_7\right)\right)\right)=\left\{\alpha_1\alpha_2\alpha_5\eta_1\eta_2,\alpha_1\alpha_3\eta_3\right\}$ ve $sm\left(A_{bm}\left(L\left(S/G_7\right)\right)\right)=\left\{\alpha_1\alpha_2\alpha_5\eta_1,\alpha_1\alpha_3,\alpha_1\alpha_4\alpha_8\alpha_9\right\}$'dir. Görüldüğü gibi A_{bm} işlemi ile $L\left(S/G_7\right)$ dilinden $\alpha_1\alpha_4\alpha_7$ kelimesi düşmüştür.

Aynı zamanda L dilene ait başarımı arttırmak için dili A_{sm} işlemine tabii tuttuğumuzda $A_{sm}\left(L\left(S/G_7\right)\right)=\left\{\overline{\alpha_1\alpha_2\alpha_5\eta_1\eta_2},\overline{\alpha_1\alpha_4\alpha_8\alpha_9},\overline{\alpha_1\alpha_3\eta_3},\alpha_1\alpha_4\alpha_7,\alpha_1\alpha_3\alpha_6\right\}$ elde edilir. Bu elde edilen dil için $bm\left(A_{sm}\left(L\left(S/G_7\right)\right)\right)=\left\{\alpha_1\alpha_2\alpha_5\eta_1\eta_2,\alpha_1\alpha_3\eta_3,\alpha_1\alpha_4\alpha_7\right\}$ ve $sm\left(A_{bm}\left(L\left(S/G_7\right)\right)\right)=\left\{\alpha_1\alpha_2\alpha_5\eta_1,\alpha_1\alpha_3,\alpha_1\alpha_3\alpha_6,\alpha_1\alpha_4\alpha_8\alpha_9\right\}$'dir. Fark edildiği üzere A_{sm} işlemi ile $L\left(S/G_7\right)$ diline $\alpha_1\alpha_3\alpha_6$ işaretli kelimesi eklenmiştir.

3. OPTİMAL KİLİTLENEBİLİR DENETİMSEL GÖZETLEYİCİ TASARIM PROBLEMİ

3.1 Giriş

Deterministik sonlu durumlu bir otomat yaklaşımıyla modellenmiş bir ayrık olay sistemi kabul edilebilir dil ile beraber ele alındığında, sistemin istenilen davranışı üretebilmesi için kabul edilebilirlik koşulu gereği ürettiği dil, çok fazla kilitlenme içerebilir. Böyle bir durumda bu kilitlenmelerden kurtulabilmek için denetimsel gözetleyici olarak en az kısıtlamalı kilitlenmesiz çözüm (EAKKÇ) tercih edilir. EAKKÇ kontrol altında üretilen dilden en az sayıda kelime çıkartarak, dilin kilitlenmesiz olması sağlanır. Kilitlenmeleri engelleyebilmek için denetimsel gözetleyicinin çok sayıda istenilen işaretli kelimeyi engelleyebilir ve buna bağlı olarak elde edilen çözüm tutucu bir çözüm olabilir. Beklenen davranışın çok uzağında bir davranış biçimi kabul edilebilir bir çözüm değildir. Bu nedenle istenilen davranışa ilişkin dili üretebilmek için denetimsel gözetleyici olarak tam başarımlı çözüm (TBÇ) seçilir. Bu sefer de kabul edilebilirlik koşulu gereği denetimsel gözetleyici çok fazla kilitlenmeye izin verir. Bu durumda TBÇ ile EAKKÇ'e nazaran istenilen davranış elde edilmiştir fakat beraberinde çok sayıda kilitlenme de üretilen dile eklenmiştir. Bu noktada bu iki çözüm de kabul edilebilir değildir. Birisi kilitlenmeleri engellemek için sistem davranışına çok fazla kısıt getirirken, bir diğeri ise sisteme istenilen davranışı kazandırmak için çok fazla kilitlenmeye izin vermektedir. Sonuç olarak TBÇ ve EAKKÇ yaklaşımları bu tür sistemler için kabul edilebilir değildir.

Sistem çok fazla kilitlenme içeriyor ise sistemin başarımını arttırmak için belli bir oranda kilitlenmeye izin verilmelidir. Bu durumda temel sorun, ne oranda kilitlenmeye izin verilmesi ile ilgilidir. Daha önce de belirtildiği gibi bu çalışmanın esas amacı bu probleme yeni biçimsel bir cevap aramaktır.

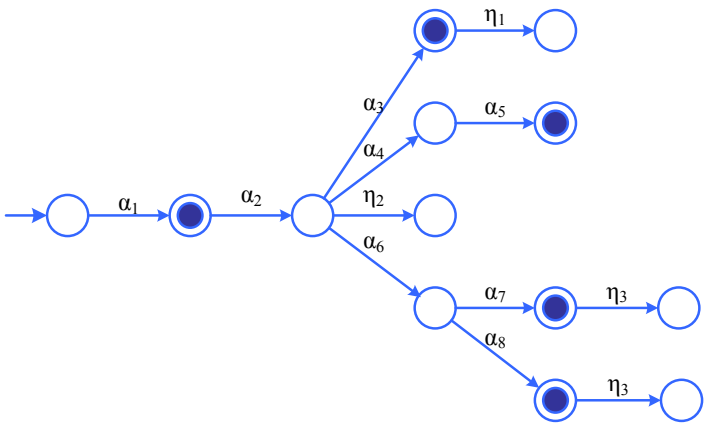
Literatürde kilitlenebilir denetimsel gözetleyiciler ilk olarak Chen ve Lafortune tarafından incelenmiştir [3, 4]. Yapılan çalışmada kilitlenebilir denetimsel gözetleyicinin performansını kilitlenme miktarını arttırmadan başarımı arttırmaya çalışan veya başarımı değiştirmeksizin kilitlenmeleri azaltmaya çalışan bir çözüm

önerilmiştir. Burada önerilen çözüm yöntemi kapsama anlamında başlangıç dilinin performansını iyileştirmeye çalışmaktadır. 2.4.5 numaralı bölümde temel kilitlenebilir denetimsel gözetleyici problemi başlığı altında bu çözüm yöntemi ayrıntılı olarak incelenmiştir. Kuram kendi içinde tutarlı sonuçlar vermesine rağmen eksik olduğu noktalar mevcuttur. İlk olarak başlangıç koşullarına çok bağımlıdır. 2.4.5.3 ve 2.4.5.5 numaralı kuramlar gereği ulaşılan son dil, kapsama anlamında en iyi çözüm olduğundan farklı bir başlangıç dili ile ulaşılan son dil farklı olacaktır. Burada elde edilen çözümler küme teorisine göre kapsama anlamında elde edildiğinden birbirleriyle karşılaştırılması olanaksızdır. Bunun yanı sıra Chen ve Lafortune tarafından önerilen çözümde, bütün kelimeler sistem için aynı anlamı taşıyormuşçasına hareket edilip çözüme ulaşılmıştır. Fakat gerçekte her bir olayın ederi aynı olmadığı gibi her bir kilitlenmenin veya başarımın ederi de aynı değildir. Sonuçta üretilen kelimelerin sistem üzerine etkisi farklı farklıdır. Bunu dikkate alıp belirtilen performans ölçüleri içerisinde çözüm bulmaya çalışan yeni bir algoritma vermek bizi daha anlamlı çözümlere götürecektir [89-92].

3.1.1 Örnek

Şekil 3.1'deki G_8 otomatını düşünelim. İstenilen işaretli dil ve kontrol edilemeyen olaylar kümesi sırasıyla $L_{am} = \{\alpha_1, \alpha_1\alpha_2\alpha_4\alpha_5, \alpha_1\alpha_2\alpha_6\alpha_7, \alpha_1\alpha_2\alpha_6\alpha_8\}$, $\sum_{uc} = \{\eta_1, \eta_2, \eta_3\}$ olsun. Sistemin denetim altında ürettiği dil ise şu şekilde olsun: $L(S/G_8) = \{\overline{\alpha_1\alpha_2\eta_2}, \overline{\alpha_1\alpha_2\alpha_4\alpha_5}, \overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}\}$

$L(S/G_8)$ dili incelenirse $\alpha_1\alpha_2\eta_2$ ve $\alpha_1\alpha_2\alpha_6\alpha_7\eta_3$ gibi iki farklı kilitlenmenin oluştuğu görülür. Bu noktada sadece modelin değil aynı zamanda sistemin ürettiği kelimelerin hakkında da ayrıntılı bilgiye sahip olduğumuzu düşünelim, öyle ki $\alpha_1\alpha_2\alpha_6\alpha_7\eta_3$ kelimesi kritik bir kilitlenmeye karşılık gelmiş olsun.



Şekil 3.1 : G_8 otomatı

$L(S/G_8)$ dili sezgisel olarak incelenirse, $\alpha_1\alpha_2\alpha_6\alpha_7\eta_3$ kelimesini sistemin üretmemesi gerekir. Fakat Chen ve Lafortune tarafından geliştirilen yapı bu kelimeyi $L(S/G_8)$ dilinden çıkartmaya izin vermez. Zira bu kelimeyi çıkartabilmek için kabul edilebilirlik koşulu gereği beraberinde $\alpha_1\alpha_2\alpha_6\alpha_7$ başarımını da üretilen dilden çıkartmak gerekir. Bu beklenen bir sonuçtur, zira Chen ve Lafortune tarafından geliştirilen operatörler kapsama anlamında çalışmaktadır. Üretilen kelimelerin hepsi aynı öneme sahipmişçesine değerlendirilmekte, onların sistem üzerindeki etkilerini göz ardı edilmektedir. Bu noktada sistemin denetim altında ürettiği dili yeniden belirleyecek biçimsel bir yapıya ihtiyaç vardır.

3.1.2 Ön Koşullar

Yukarıda genel olarak ifade edilen kilitlenebilir denetimsel gözetleyici tasarımı problemi için aşağıda verilen ön koşullar dâhilinde çözüm aranacaktır.

$$L_m(S/G) := L(S/G) \cap L_m(G) \quad \textbf{(3.1)}$$

$$L_m(S/G) \subseteq L_{am} \quad \textbf{(3.2)}$$

$$L(S/G) \subseteq L_a = \overline{L_a} \subseteq L(G) \text{ ve } L_{am}^{\downarrow C} \subseteq L_a \quad \textbf{(3.3)}$$

$$L_{am} = L_a \cap L_m(G) = L_a \cap L_{am}. \quad \textbf{(3.4)}$$

Bunun doğal bir sonucu olarak L_{am}, $L_m(G)$ kapalıdır.

Kilitlenebilir denetimsel gözetleyici problemi için tüm çözümleri içeren dil kümesi

$$L_{cand} := \left\{ \Gamma : \left(\overline{L_{am}^{\uparrow C}} \subseteq \Gamma \subseteq L_{am}^{\downarrow C} \right) \wedge \left(\Gamma = \overline{\Gamma} \right) \wedge \left(\overline{\Gamma} \Sigma_{uc} \cap L(G) \subseteq \overline{\Gamma} \right) \right\}. \quad \textbf{(3.5)}$$

şeklinde verilebilir. Aynı zamanda bu çalışmada sisteme ait tüm çözümlerin sonlu sayıda kelimeden oluştuğu varsayılmıştır.

3.2 Tanımlar

Burada optimal kilitlenebilir denetimsel gözetleyici tasarım problemi için yapılmış tanımlar verilecektir.

3.2.1 Olay Ederi

G deterministik sonlu durumlu bir otomat, $\sum$ bu otomata ilişkin olay kümesi ve $e_i \in \sum$ herhangi bir olay olmak üzere olmak üzere bu olayın ederi c_e olarak verilir öyle ki $c_e(s, e_i) : \sum^* \times \sum \to \Re^+$

Ayrıca $c_e(s,\varepsilon) := \varepsilon$ olarak tanımlanmıştır. Bir olay bir otomatta birden fazla durumda aktif olabilir ve her aktif olduğu durumda aynı edere sahip olmak zorunda değildir. c_e olay ederi ile sadece olayın oluşmasının ederi ortaya konulmamış aynı zamanda olayın hangi kelimeden sonra oluştuğunun bilgisi de tanıma eklenmiştir. Bu da ayrık olay sistemlerindeki olay kavramının anlamını değiştirmiştir. Bu durumda örneğin bir sistem için "arıza" olayının oluşmasının ederi olayın gerçekleştiği duruma göre farklılık gösterir. Sistem bekleme halindeyken arıza olayının oluşmasının sistem için ederi ile çalışıyorken oluşmasının ederi farklıdır. Diğer bir şekilde $s_1, s_2 \in \Sigma^*$ iki farklı kelime ve $e_i \in \Sigma$ olmak üzere $c_e(s_1, e_i)$ ve $c_e(s_2, e_i)$ bu iki farklı kelimenin ardından e_i olayının üretilmesinin ederini ifade eder ve bunların birbirine eşit olması gerekmez [91]. Olayların ederlerinin nasıl seçileceği bu çalışmanın dışındadır. Bu konu ile ilgili çeşitli çalışmalar literatürde mevcuttur [93].

3.2.2 Kelime Ederi

G deterministik sonlu durumlu bir otomat, Σ bu otomata ilişkin olay kümesi ve $s_k, s_l \in L(G)$ bu otomatın ürettiği iki farklı kelime olmak üzere bir kelimenin ederi $c_s(s_k, s_l) : \Sigma^* \times \Sigma^* \to \Re^+$ olarak verilir öyle ki

$$c_s(s_k, s_l) := \frac{\sum_{i=1}^{\|s_l\|} c_e\left[s_k . p_{i-1}(s_l), q_i(s_l)\right]}{\|s_l\|}$$

Burada $c_s(s_k, s_l)$ ile s_l kelimesinin s_k kelimesinden sonra üretilmesinin sisteme olan ederi ifade edilmiştir. Ayrıca olay ederinin tanımı gereği her zaman $c_e(\varepsilon, \varepsilon) := \varepsilon$'dir. Bunu elde etmek için s_l kelimesini oluşturan her bir olayın ederlerinin toplamı alınır ve s_l kelimesinin uzunluğuna bölünür. Burada uzunluğa bölmekteki amaç kelimenin ederini normalize etmektir. Eğer bir kelime önemli bir işin bitirilmesini ifade ediyorsa ya da kritik bir kilitlenmeye karşılık geliyorsa bu kelimelerin ederi de diğer kelimelere nazaran yüksek olmalıdır. Tanımda verilen $p_j(s)$ fonksiyonu s kelimesinin j uzunluklu önekini ve $q_i(s)$ fonksiyonu ise s kelimesinin $i.$ olayını verir. Mesela $s = e_1e_2e_1e_3$ için $p_3(s) = e_1e_2e_1$ ve $q_3(s) = e_1$'dir.

Yukarıda verilen tanımı göre $e_4e_3e_2$ kelimesinin $e_1e_2e_3$ kelimesinden sonra üretilmesinin ederi

$$c_s(e_1e_2e_3, e_4e_3e_2) = \frac{c_e(e_1e_2e_3, e_4) + c_e(e_1e_2e_3e_4, e_3) + c_e(e_1e_2e_3e_4e_3, e_2)}{\|e_4e_3e_2\|}$$

şeklinde hesaplanır. O zaman $s_k, s_l \in \sum^*$ gibi iki farklı kelimenin birbirine bağlanması ile oluşan $s_k s_l$ kelimesinin ederi verilen kelime eder tanımı kullanılarak elde edilir.

$$c_s(\varepsilon, s_k s_l) = \frac{c_s(\varepsilon, s_k)\|s_k\| + c_s(s_k, s_l)\|s_l\|}{\|s_k\| + \|s_l\|}$$

Burada $c_s(\varepsilon, s_k)$ ve $c_s(s_k, s_l)$'nin bilindiği kabul edilmiştir. Bundan sonra kolaylık olması için $c_s(\varepsilon, s_k s_l)$ yerine $c_s(s_k s_l)$ kullanılacaktır.

3.2.3 Dilin Ederi

$L = \{s_1, s_2, s_3, ..., s_n\} \subset (\sum^*)^n$ döngüsel olmayan bir dil olmak üzere dilin ederi $\beta(L) : (\sum^*)^n \to \Re^+ + \{0\}$ şeklindedir, öyle ki

$$\beta(L) := \begin{cases} \sum_{i=1}^{n} c_s(s_i) & L \neq \emptyset \\ 0 & \text{diğer} \end{cases}$$

Bir dilin ederi kendisini oluşturan kelimelerin ederlerinin toplamı olacak şeklinde verilmiştir. Her bir kelimenin ederi pozitif olduğundan dilin ederi de her zaman pozitif olacaktır. Ancak elde edilen pozitif sayısal değerin herhangi bir belirleyici özelliği yoktur, öyle ki iki farklı dili bu ederler üzerinden karşılaştırmak mümkün değildir. İki farklı dili karşılaştırabilmek için metrik uzay gibi biçimsel bir yapı kurmak gerekir.

3.2.3.1 Özellik: Eğer $\forall L_1, L_2 \in \sum^*$ ve $L_1 \subseteq L_2$ ise $\beta(L_1) \leq \beta(L_2)$

Bu özellik β fonksiyonun tanımı gereği aşikârdır.

3.2.3.2 Özellik: Eğer $L \in \sum^*$, $L = \bigcup_{i=1}^{n} L_i$ ve $L_k \cap L_l = \emptyset$ öyle ki $k, l = 1, 2, 3, ...n$ ve $k \neq l$ ise $\beta(L) = \beta(L_1) + \beta(L_2) + \beta(L_3) + ... + \beta(L_n)$

İspat – $L = \{s_1, s_2, s_3, ..., s_z\}$ şeklinde ifade ediliyor olsun. Dili $L_1 = \{s_1, s_2, s_3, ...s_k\}$, $L_2 = \{s_{k+1}, s_{k+2}, ..., s_t\}$, ... , $L_n = \{s_{v+1,} s_{v+2}, ..., s_z\}$ olacak şekilde ayrık dillerin bileşimi ile ifade edelim. Aynı zamanda L dilinin ederi kendini oluşturan alt dillerin ederlerinin toplamı ile ifade edilebilir. O zaman

$$\beta(L)=\sum_{i=1}^{z}c_s(s_i)=\sum_{i=1}^{k}c_s(s_i)+\sum_{i=k+1}^{t}c_s(s_i)+...+\sum_{i=v+1}^{z}c_s(s_i) \quad \textbf{(3.6)}$$

$$\beta(L)=\beta(L_1)+\beta(L_2)+\beta(L_3)+...+\beta(L_n) \quad \textbf{(3.7)}$$

3.2.3.3 Özellik: Eğer $\forall L_1, L_2 \in \Sigma^*$ ise $\beta(L_1 \cup L_2) \le \beta(L_1)+\beta(L_2)$

İspat – L_1, L_2 ve $L_1 \cup L_2$'yi şu şekilde yazalım $L_1=\{L_1 \setminus L_2\}\cup\{L_1 \cap L_2\}$, $L_2=\{L_2 \setminus L_1\}\cup\{L_1 \cap L_2\}$ ve $L_1 \cup L_2=\{L_1 \setminus L_2\}\cup\{L_2 \setminus L_1\}\cup\{L_1 \cap L_2\}$. O zaman 3.2.3.2 özelliği gereği

$$\beta(L_1 \cup L_2)=\beta(\{L_1 \setminus L_2\})+\beta(\{L_2 \setminus L_1\})+\beta(\{L_1 \cap L_2\}) \quad \textbf{(3.8)}$$

$$\beta(L_1)=\beta(\{L_1 \setminus L_2\}\cup\{L_1 \cap L_2\})=\beta(\{L_1 \setminus L_2\})+\beta(\{L_1 \cap L_2\}) \quad \textbf{(3.9)}$$

$$\beta(L_2)=\beta(\{L_2 \setminus L_1\}\cup\{L_1 \cap L_2\})=\beta(\{L_2 \setminus L_1\})+\beta(\{L_1 \cap L_2\}) \quad \textbf{(3.10)}$$

Aynı zamanda β fonksiyonunun tanımı gereği $\beta(L_1), \beta(L_2), \beta(L_1 \cup L_2)$ her zaman pozitiftir. Sonuçta

$$\beta(\{L_1 \setminus L_2\})+\beta(\{L_2 \setminus L_1\})+\beta(\{L_1 \cap L_2\}) \le \beta(\{L_1 \setminus L_2\}) + \beta(\{L_2 \setminus L_1\})+2\beta(\{L_1 \cap L_2\})$$

$$\beta(L_1 \cup L_2) \le \beta(L_1)+\beta(L_2)$$

3.2.4 Mesafe Fonksiyonu

$\forall L_1, L_2 \in \Sigma^*$ olmak üzere $d: \Sigma^* \times \Sigma^* \to \Re^+ + \{0\}$ olacak şekilde bir mesafe fonksiyonu tanımlanabilir öyle ki $d(L_1, L_2) := \beta(\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\})$'dir.

Ayrık olay sistemlerinde iki dil arasındaki mesafe birbirlerine göre farklı olan kelimelerle ilişkilidir. Fakat bu farklı olan kelimelerin sadece adedi ya da benzer bir özelliği üzerinden işlem yapmak yukarıda anlatılanlar dikkate alındığında yetersizdir. Bu noktada metrik uzay iki farklı dili karşılaştırmak için uygun matematiksel ortamı sağlamaktadır. Farklı kelimelerin toplam ederleri aslında iki farklı dilin arasındaki gerçek mesafeye karşılık düşmektedir, öyle ki bu mesafe fonksiyonu ile bir metrik uzay oluşturmak ve farklı diller arasındaki mesafeyi ifade etmek mümkündür.

3.2.5 Önerme

2^{Σ^*} kümesi ve d mesafe fonksiyonu $\left(2^{\Sigma^*}, d\right)$ şeklinde metrik uzay aksiyomlarını sağlar.

İspat – Metrik uzaya ilişkin aksiyomların sağlandığı gösterilecektir.

- β'nın tanımı gereği $d(L_1, L_2) \to \Re^+ + \{0\}$, $\forall L_1, L_2 \in \Sigma^*$
- $d(L_1, L_2) = \beta(\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\})$, $d(L_2, L_1) = \beta(\{L_2 \setminus L_1\} \cup \{L_1 \setminus L_2\})$ O zaman $\forall L_1, L_2 \in \Sigma^*$ için $d(L_1, L_2) = d(L_2, L_1)$ dır.
- Eğer $d(L_1, L_2) = 0$ ise $\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\} = \emptyset \Rightarrow L_1 = L_2$ **(3.11)**

Eğer $L_1 = L_2$ ise $d(L_1, L_2) = \beta(\emptyset) = 0$ **(3.12)**

(3.11) ve (3.12) no.lu denklemler gereği $d(L_1, L_2) = 0 \Leftrightarrow L_1 = L_2$, $\forall L_1, L_2 \in \Sigma^*$

- $\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\} \supset (\{L_1 \setminus L_3\} \cup \{L_3 \setminus L_1\} \cup \{L_3 \setminus L_2\} \cup \{L_2 \setminus L_3\})$ ve $s \in \{L_3 \setminus L_1\} \cup \{L_3 \setminus L_2\}$ olduğunu kabul edelim. O zaman kabul gereği $s \in (\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\})$ olmalıdır. Fakat s'nin tanımı gereği $s \notin (\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\})$'dir. Bu bir çelişkidir o zaman kabul doğru değildir.

Sonuçta $\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\} \subseteq (\{L_1 \setminus L_3\} \cup \{L_3 \setminus L_1\} \cup \{L_3 \setminus L_2\} \cup \{L_2 \setminus L_3\})$ **(3.13)**

(3.13) no.lu denklem 3.2.3.1 özelliği gereği (3.14) şeklini alır.

$$\beta(\{L_1 \setminus L_2\} \cup \{L_2 \setminus L_1\}) \le \beta(\{L_1 \setminus L_3\} \cup \{L_3 \setminus L_1\} \cup \{L_3 \setminus L_2\} \cup \{L_2 \setminus L_3\}) \quad \textbf{(3.14)}$$

(3.14) no.lu denkleme 3.2.3.2 özelliği uygulanırsa (3.15)'e dönüşür.

$$\beta(\{L_1 \setminus L_3\} \cup \{L_3 \setminus L_1\} \cup \{L_3 \setminus L_2\} \cup \{L_2 \setminus L_3\}) \le \beta(\{L_1 \setminus L_3\} \cup \{L_3 \setminus L_1\}) + \beta(\{L_3 \setminus L_2\} \cup \{L_2 \setminus L_3\}) \quad \textbf{(3.15)}$$

O zaman $\forall L_1, L_2, L_3 \in \Sigma^*$ için $d(L_1, L_2) \le d(L_1, L_3) + d(L_3, L_2)$'dir.

3.2.6 Metrik Uzay

2^{Σ^*} kümesi ve d mesafe fonksiyonu 3.2.5 önermesi gereği metrik uzay aksiyomlarını sağladığından, $(2^{\Sigma^*}, d)$ bir metrik uzay oluşturur [94].

İki farklı dil arasındaki mesafe, iki dilin birbirlerine göre ne kadar farklı olduğunun bir ölçüsüdür. Dikkat edilecek olursa çok fazla farklı kelimeye sahip olan iki dil gerçekte birbirlerinden o kadar da uzak olmayabilir; diğer taraftan çok az farklılık gösteren iki dil gerçek manada aralarında çok büyük bir mesafeye sahip olabilir. Verilen metrik ifadesi yardımıyla bunu anlamak mümkündür.

3.2.7 Başarısızlık ve Kilitlenme Ölçüleri

S denetimsel gözetleyicisinin kontrolü altında bir G otomatının ürettiği dil $L(S/G)$ olsun ve aynı zamanda $L(S/G)$, L_{cand} kümesinin bir elemanı olsun. O

zaman $L(S/G)$ diline ait başarısızlık ölçüsü ve kilitlenme ölçüsü sırasıyla şu şekilde tanımlanır:

$$\widehat{SM^C}\left(L(S/G)\right) := d\left(L_{am}, L_m(S/G)\right)$$
$$\widehat{BM}\left(L(S/G)\right) := d\left(L(S/G), \overline{L_m(S/G)}\right)$$

Kilitlenme ölçüsü G otomatının S denetimsel gözetleyicisi ile beraber çalışması halinde, G'nin sahip olduğu olası tüm kilitlenmelerin toplam ederini vermektedir. Benzer bir şekilde başarısızlık ölçüsü ise S denetimsel gözetleyicisinin kontrolü altında G otomatının üretemediği kabul edilebilir işaretli kelimelerin toplam ederini verir. Bu noktada elimizde sadece kelimeler değil aynı zamanda bir metrik üzerinden tanımlanmış bu kelimelerin sisteme olan etkilerini ifade eden sayısal değerler mevcuttur.

3.2.7.1 Örnek: 3.1.1 numaralı örnekte verilen G_7 otomatının için iki farklı denetimsel gözetleyici ile birlikte çalışması ile üretilen diller sırasıyla $L(S_1/G_7) = \left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5}, \alpha_1\alpha_2\eta_2\right\}$ ve $L(S_2/G_7) = \left\{\overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}, \alpha_1\alpha_2\eta_2\right\}$ olsun. Sistemin ürettiği bazı kelimelerin ederleri ise şu şekilde olsun: $c_s(\alpha_1\alpha_2\eta_2) = 2$, $c_s(\alpha_1\alpha_2\alpha_4\alpha_5) = 5$, $c_s(\alpha_1\alpha_2\alpha_6\alpha_7) = 7$, $c_s(\alpha_1\alpha_2\alpha_6\alpha_8) = 4$ ve $c_s(\alpha_1\alpha_2\alpha_6\alpha_7\eta_3) = 2$. O zaman $L(S_1/G_7)$ ve $L(S_2/G_7)$ dillerine ait kilitlenme ölçüsü şu şekildedir:

$$\widehat{BM}\left(L(S_1/G_7)\right) = d\left(L(S_1/G_7), \overline{L(S_1/G_7)\cap L_{am}}\right) = \beta\left(\{\alpha_1\alpha_2\eta_2\}\right) = 2,$$
$$\widehat{BM}\left(L(S_2/G_7)\right) = d\left(L(S_2/G_7), \overline{L(S_2/G_7)\cap L_{am}}\right) = \beta\left(\{\alpha_1\alpha_2\eta_2, \alpha_1\alpha_2\alpha_6\alpha_7\eta_3\}\right) = 4$$

Benzer bir şekilde $L(S_1/G_7)$ ve $L(S_2/G_7)$ dillerine ait başarısızlık ölçüleri ise

$$\widehat{SM^C}\left(L(S_1/G_7)\right) = d\left(L_{am}, L(S_1/G_7)\cap L_{am}\right) = \beta\left(\{\alpha_1\alpha_2\alpha_6\alpha_7, \alpha_1\alpha_2\alpha_6\alpha_8\}\right) = 11$$
$$\widehat{SM^C}\left(L(S_2/G_7)\right) = d\left(L_{am}, L(S_2/G_7)\cap L_{am}\right) = \beta\left(\{\alpha_1\alpha_2\alpha_4\alpha_5, \alpha_1\alpha_2\alpha_6\alpha_8\}\right) = 12$$

şeklindedir.

3.2.8 Performans Ölçüsü

S denetimsel gözetleyicisinin kontrolü altında bir G otomatının ürettiği dil $L(S/G)$ olsun. Ayrıca $L(S/G)$, L_{cand} kümesinin bir elemanı olsun. O zaman $L(S/G)$ diline ait performans ölçütü şu şekilde tanımlanır:

$$\hat{J}\left[L(S/G)\right] := \widehat{SM^C}\left(L(S/G)\right) + \widehat{BM}\left(L(S/G)\right)$$

Kilitlenebilir denetimsel gözetleyici tasarım probleminde genellikle kilitlenmelerle başarısızlık arasında bir zıtlık vardır öyle ki kilitlenmeleri azaltmak sistemin başarısızlığında artışa neden olurken, başarısızlığını azaltmak sistemin girebileceği olası kilitlenmelerde artışa neden olur. Örnek olarak EAKKÇ ile TBÇ incelenirse kilitlenme ile başarısızlık arasındaki bu zıtlık hemen kendini gösterir. EAKKÇ'ün yapısı gereği dilde hiçbir kilitlenme yoktur fakat bütün olası kilitlenmelere neden olan kelimelerden kurtulabilmek için çok fazla istenilen işaretli kelime dilden çıkartılmıştır. Sonuçta EAKKÇ için kilitlenme ölçüsü sıfır iken L_{cand} içinde başarısızlık ölçüsü en yüksek çözümdür. Diğer bir taraftan TBÇ bütün kabul edilebilir işaretli kelimeleri üretebilmek için kontrol edilebilirlik koşulu gereği çok fazla kilitlenme içerir. TBÇ için başarısızlık ölçüsü sıfıra eşit olmasına rağmen L_{cand} içinde TBÇ en yüksek kilitlenme ölçüsüne sahip olan dildir. Kilitlenebilir denetimsel gözetleyici tasarım probleminde sistemin performansı iki ölçüye bağlıdır: başarısızlık ve kilitlenme. Bu açıdan bakıldığında verilen performans ölçüsü tanımı bu iki kavramı da içermektedir. Açıktır ki kilitlenebilir denetimsel gözetleyici problemi için bu performans ölçütüne göre sayısal olarak en düşük performans ederine sahip olan dil en iyi dildir. O zaman optimal kilitlenebilir denetimsel gözetleyici tasarım problemi şu şekilde tanımlanabilir.

3.2.9 Optimal Kilitlenebilir Denetimsel Gözetleyici Problemi

Sisteme ait tüm davranışları içeren dil $L(G)$ ve sisteme ait tüm olası çözümleri içeren küme $L_{cand} \neq \varnothing$ olmak üzere S denetimsel gözetleyicisinin kontrolü altında G otomatının ürettiği dil ise $L(S/G)$ olsun. $L(S/G) \in L_{cand}$ için performans ölçüsü $\hat{J}\left[L(S/G)\right]$ olsun. Bu koşullar altında optimal kilitlenebilir denetimsel gözetleyici problemi (OKDGP) şu şekilde tanımlanır:

$$\arg\left\{\min_{L \in L_{cand}} \hat{J}\left[L(S/G)\right]\right\}$$

Burada arg fonksiyonu verilen koşulu sağlayan dile ait kelimeleri ifade eden fonksiyondur. Performans ölçüsü kilitleme ve başarısızlık üzerine tanımlandığından, idealde en iyi çözüm bütün istenilen işaretli kelimeleri herhangi bir kilitlenmeye sebebiyet vermeden üreten çözümdür. Lakin kontrol edilebilirlik ve kabul edilebilirlik koşulu gereği bu çözüm her zaman olası değildir. O zaman olası çözümleri içeren L_{cand} kümesinden bir kilitlenebilir denetimsel gözetleyici seçmek gerekir. Bu noktada seçilen denetimsel gözetleyicinin performansının en düşük olması beklenir. Fakat uygulamada bu yeterli değildir. Zira denetimsel gözetleyiciden amaç elimizdeki sistemden mümkün olan en çok sayıda beklenilen

davranışı elde etmektir. O zaman denetimsel gözetleyicinin ürettiği kabul edilebilir işaretli kelimelerin de olabildiğince fazla olması istenir. Sonuç olarak dil en düşük performans ölçüsüne sahip olmasının yanı sıra istenilen işaretli dili de olabildiğince işaretleyebilir olmalıdır.

3.2.10 Kayıp Faktörü

S denetimsel gözetleyicisinin kontrolü altında bir G otomatının ürettiği dil $L(S/G) \in L_{cand}$ olmak üzere $L(S/G)$ diline ait kayıp faktörü şu şekilde tanımlanır:

$$F(L) := \beta\left(L \setminus \overline{L_m}\right) - \beta(L_m)$$

Kayıp faktörü L diline ait olan kilitlenmelerin ederlerinin toplamından dilin ürettiği işaretlediği kelimelerin ederini toplamını çıkartmak suretiyle elde edilir, öyle ki tanımlanan kayıp faktörü ile bir dile ait kilitlenmeler ile başarımlar arasındaki denge sayısal olarak ifade edilir.

3.3 Optimizasyon Algoritmasına ait Temel Kavramlar

S denetimsel gözetleyicisinin kontrolü altında bir G otomatının ürettiği dil $L(S/G) \in L_{cand}$ olmak üzere kabul edilebilir işaretli kelimelerin oluşturduğu dil $L_{am} \subseteq L_m(G)$ olsun. $L(S/G)$ için kilitlenmeye sebebiyet veren kelimeler ve başarılamayan kelimeler sırasıyla $BM\left(L(S/G)\right)$ ve $SM^C\left(L(S/G)\right)$dir, öyle ki $BM\left(L(S/G)\right) = \{s_1, s_2, s_3, ..., s_m\}$ ve $SM^C\left(L(S/G)\right) = \{t_1, t_2, t_3, ..., t_n\}$'dir. $L(S/G)$ olası tüm çözüm kümesinin bir elemanı olduğundan düzenli bir dildir ve sonuçta $BM\left(L(S/G)\right)$ ve $SM^C\left(L(S/G)\right)$ kümeleri sonlu elemanlı iki kümedir.

3.3.1 Özellik: Eğer $L = \overline{L}$ ve $s_i \in BM(L)$ ise $\overline{(L \setminus s_i)^{\uparrow C}} = (L \setminus s_i)^{\uparrow C}$.

İspat – $K = \overline{K} \in \Sigma^*$ olacak şekilde bir dil olsun. ↑C işlemi gereği her zaman $\left(\overline{K}\right)^{\uparrow C} \supseteq \overline{K^{\uparrow C}}$ koşulu sağlanır. Önek kapanış işleminin özelliği gereği de $\overline{K^{\uparrow C}} \supseteq K^{\uparrow C}$'dir.

$K = \overline{K}$ olduğundan $\overline{K^{\uparrow C}} \supseteq \left(\overline{K}\right)^{\uparrow C}$ dir. Sonuçta $\overline{K^{\uparrow C}} = \left(\overline{K}\right)^{\uparrow C}$ **(3.16)**

Diğer bir taraftan $s_i \in BM(L)$ ve $L = \overline{L}$ olduğundan $L \setminus s_i = \overline{L \setminus s_i}$ dir. **(3.17)**

(3.16) ve (3.17) no.lu eşitliklerin dolayı $\overline{\left(L \setminus s_i\right)^{\uparrow C}} = \left(L \setminus s_i\right)^{\uparrow C}$ dir.

3.3.1 numaralı özellik, önek kapalı herhangi bir dilden kendisine ait bir kilitlenme çıkartılması halinde elde edilen sonuç dilin en büyük kontrol edilebilir kısmının her zaman önek kapalı olduğunu ifade etmektedir.

3.3.2 Özellik: Eğer $L \in L_{cand}$, $L = \overline{L}$ ve $s_i \in BM\left(L\right)$ ise $s_i \notin \left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C}$.

İspat – $\alpha_i \in L$ ve ↑C işleminin monoton olmasının sonucunda (3.18) no.lu eşitsizlik her zaman sağlanır.

$$\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \subseteq L \setminus s_{\mathrm{i}}. \tag{3.18}$$

$$\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am} \subseteq \left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C}$$

$$\left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C} \subseteq \left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C}\right]^{\downarrow C} = \overline{\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C}} \tag{3.19}$$

3.3.1 özelliği gereği (3.19) no.lu eşitsizlik (3.20) no.lu eşitsizliğe dönüşür.

$$\left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C} \subseteq \left(L \setminus s_i\right)^{\uparrow C} \tag{3.20}$$

(3.18) ve (3.20) no.lu eşitsizlikleri birleştirirsek,

$$\left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C} \subseteq L \setminus s_i \tag{3.21}$$

Her zaman $s_i \notin L \setminus s_i$ olduğundan, (3.21) gereği $s_i \notin \left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C}$

3.3.2 numaralı özellik gereği, $L \in L_{cand}$ kümesine ait herhangi bir dilden kendisine ait bir kilitlenme çıkartılır, kalan dilin en büyük kontrol edilebilir alt dili elde edilir ve sonucun kabul edilebilir işaretli dil ile ortak elemanlarının en küçük kontrol edilebilir önek kapalı üst dili elde edilir ise bu elde edilen dil her zaman başlangıç dilinden atılan kilitlenmeyi içermez. Kontrol edilebilirlik ve kabul edilebilirlik koşulu gereği atılan kelime sonuç dile geri gelmez. Aynı zamanda $\left[\left(L \setminus s_{\mathrm{i}}\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C}$ her zaman L dilinin bir alt kümesidir.

3.3.3 Özellik: $s_i \in BM(L)$ ve $L \in L_{cand}$ için $\left[(L \setminus s_i)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$ her zaman L_{cand} kümesinin bir elemanıdır.

İspat – $L \in L_{cand}$ olduğundan her zaman $\overline{L_{am}^{\uparrow C}} \subseteq L \subseteq L_{am}^{\downarrow C}$

$$\left[\overline{L_{am}^{\uparrow C}} \setminus s_{\mathrm{i}} \right]^{\uparrow C} \subseteq (L \setminus s_{\mathrm{i}})^{\uparrow C} \subseteq \left[L_{am}^{\downarrow C} \setminus s_{\mathrm{i}} \right]^{\uparrow C} \quad \textbf{(3.22)}$$

$\forall s_i \in BM(L)$ için $\overline{L_{am}^{\uparrow C}}$ EAKKÇ olduğundan her zaman $s_i \notin \overline{L_{am}^{\uparrow C}}$ sağlanır. O zaman $\left[\overline{L_{am}^{\uparrow C}} \setminus s_{\mathrm{i}} \right]^{\uparrow C} = \overline{L_{am}^{\uparrow C}}$'dır. Diğer bir taraftan ↑C işleminin monotonluk özelliği gereği $\left[L_{am}^{\downarrow C} \setminus s_{\mathrm{i}} \right]^{\uparrow C} \subseteq L_{am}^{\downarrow C}$ her zaman sağlanır.

O zaman (3.22) no.lu eşitsizlik düzenlenirse $\overline{L_{am}^{\uparrow C}} \subseteq (L \setminus s_{\mathrm{i}})^{\uparrow C} \subseteq L_{am}^{\downarrow C}$ elde edilir.

$$\left[\overline{L_{am}^{\uparrow C}} \cap L_{am} \right]^{\downarrow C} \subseteq \left[(L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am} \right]^{\downarrow C} \subseteq \left[L_{am}^{\downarrow C} \cap L_{am} \right]^{\downarrow C} \quad \textbf{(3.23)}$$

Aynı zamanda $\left[\overline{L_{am}^{\uparrow C}} \cap L_{am} \right]^{\downarrow C} = \overline{L_{am}^{\uparrow C}}$ ve $\left[L_{am}^{\downarrow C} \cap L_{am} \right]^{\downarrow C} = L_{am}^{\downarrow C}$ dir. Bu iki eşitliği (3.23) no.lu eşitsizliğe yerleştirirsek,

$$\overline{L_{am}^{\uparrow C}} \subseteq \left[(L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am} \right]^{\downarrow C} \subseteq L_{am}^{\downarrow C} \quad \textbf{(3.24)}$$

Aynı zamanda ↓C işlemi gereği $\left[(L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$ her zaman kontrol edilebilir ve önek kapalıdır.

Sonuç olarak (3.24) eşitsizliği gereği, $\left[(L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am} \right]^{\downarrow C} \in L_{cand}$.

3.3.3 numaralı özellik $\left[(L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$ ile elde edilen dilin her zaman L_{cand} kümesinin bir elemanı olduğu ifade etmektedir. L_{cand} kümesinin kabul edilebilir tüm çözümleri içeren küme olduğu düşünülürse, elde edilen sonuç dilde kabul edilebilir dillerden birisidir. Sonuçta bu yapılan işlem sonucunda her zaman kabul edilebilir dil kümesi içinde hareket edilmiş olunur.

3.3.4 T İşlemi

$L \in L_{cand}$ ve $s_i \in BM\left(L\left(S/G\right)\right)$ olmak üzere, $T : L_{cand} \to L_{cand}$ olacak şekilde bir işlem olsun öyle ki

$$T\left(L, s_i\right) := \begin{cases} \left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C} & eğer \;\; \hat{J}\left(\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) < \hat{J}\left(L\right), \\ L & diğer \end{cases} \text{dir.}$$

T işlemi iki giriş üzerinden tanımlanmıştır; olası kabul edilebilir çözüm kümesine ait bir dil ve bu dile ait bir kilitlenme. İşlemin sonucunu elde edebilmek için ilk olarak $\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ elde edilir ve bu dile ait performans ölçüsü hesaplanır. Eğer $\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ ait performans ölçüsü L diline ait performans ölçüsünden küçük ise işlem sonucunda $\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ elde edilir. Eğer büyük veya eşit ise L dili işlemin sonucu olarak kabul edilir. Burada işlem kesin bir performans değişimine bağlanmıştır. T işlemi 3.3.3 numaralı özellik gereği her zaman L_{cand} kümesine göre kapalılık gösterir. Bu da her zaman L_{cand} kümesinden L_{cand} kümesine olmasını garanti altına alır.

Eğer performans ile ilgili koşul sağlanır ise L dilinden çıkartılan kelimeler $L \setminus \left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ şeklindedir. Gösterim kolaylığı sağlamak adına bundan sonra $L \setminus \left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ yerine $r\left(L, s_i\right)$ kullanılacaktır.

Ayrık olay sistemlerinde kilitlenme incelendiğinde fark edilecektir ki her bir kilitlenmenin sistem üzerine olan etkisi farklı farklıdır. Aynı zamanda kilitlenmelerin bir kısmı yukarda açıklanan sebeplerden ötürü kabul edilebilir iken bir kısmı kesinlikle engellenmelidir. Bu engellenmesi gereken kritik kilitlenmelere ait kelimelerin çıkartılması sistem performansını olumlu yönde etkiliyor ise denetimsel gözetleyiciye ait dilden uygun bir şekilde çıkartılmalıdır. Bunu yaparken sonuç dilin kontrol edilebilirlik ve kabul edilebilirlik koşullarını sağladığından emin olunmalıdır. Herhangi bir kritik kilitlenme denetimsel gözetleyicinin ürettiği dilden çıkartıldığında kontrol edilebilirlik ve kabul edilebilirlik koşulunun gereği olarak çoğu zaman beraberinde fazladan kelimeleri de dilden çıkartılır. Bu çıkarılan kelimeler içerisinde üretilen dilin performansını etkileyen kelimeler de olabilir.

$L \in L_{cand}$ diline ait farklı kilitlenmelerin T işlemi ile dilden çıkartılması halinde, kilitlenmeler ile beraber çıkartılan fazladan kelimelerin ya hiçbir ortak noktası yoktur ya da bir kilitlenmeye ait çıkartılan kelime grubu diğer bir kilitlenmeye ait çıkartılan kelime grubunu kapsar. Bu ↑C işleminin doğal bir sonucudur. Bu sonucu $s_i, s_j \in BM(L)$ olmak üzere şu iki ifade ile göstermek mümkündür: $r(L,s_i) \cap r(L,s_j) = \varnothing$, $r(L,s_i) \subseteq r(L,s_j)$ veya $r(L,s_j) \subseteq r(L,s_i)$

3.3.5 Önerme

$L_1, L_2 \in L_{cand}$ iki dil olsun öyle ki $L_1 \subset L_2$ ve $\left(L_1 \cap \overline{L_{2m}}\right) \setminus \overline{L_{1m}} = \varnothing$ koşulunu sağlasın. O zaman $\hat{J}(L_1) = \hat{J}(L_2) - F(L_2 \setminus L_1)$ her zaman sağlanır.

İspat – $L_1 \subset L_2$ olduğundan $L_{1m} = L_{2m} \setminus \left[(L_2 \setminus L_1) \cap L_{am}\right]$ şeklinde yazılabilir.

O zaman $L_{am} \setminus L_{1m} = L_{am} \setminus \left\{L_{2m} \setminus \left[(L_2 \setminus L_1) \cap L_{am}\right]\right\}$ **(3.25)**

$= \{L_{am} \setminus L_{2m}\} \cup \left\{L_{am} \cap \left[(L_2 \setminus L_1) \cap L_{am}\right]\right\}$

$= \{L_{am} \setminus L_{2m}\} \cup \left\{(L_2 \setminus L_1) \cap L_{am}\right\}$

$\Rightarrow \beta\{L_{am} \setminus L_{1m}\} = \beta\{L_{am} \setminus L_{2m}\} + \beta\left\{(L_2 \setminus L_1) \cap L_{am}\right\}$ **(3.26)**

$L_2 \setminus \overline{L_{2m}} = \left(L_1 \cup (L_2 \setminus L_1)\right) \setminus \overline{\left[L_1 \cup (L_2 \setminus L_1)\right] \cap L_{am}}$ **(3.27)**

$= \left[L_1 \cup (L_2 \setminus L_1)\right] \setminus \overline{\left\{L_{1m} \cup \left[(L_2 \setminus L_1) \cap L_{am}\right]\right\}}$

$= \left\{L_1 \setminus \overline{L_{1m} \cup \left[(L_2 \setminus L_1) \cap L_{am}\right]}\right\} \cup \left\{(L_2 \setminus L_1) \setminus \overline{L_{1m} \cup \left[(L_2 \setminus L_1) \cap L_{am}\right]}\right\}$

$= \left\{L_1 \setminus \left(\overline{L_{1m}} \cup \overline{\left[(L_2 \setminus L_1) \cap L_{am}\right]}\right)\right\} \cup \left\{(L_2 \setminus L_1) \setminus \left(\overline{L_{1m}} \cup \overline{\left[(L_2 \setminus L_1) \cap L_{am}\right]}\right)\right\}$

$= \left\{\left(L_1 \setminus \overline{L_{1m}}\right) \cap \left(L_1 \setminus \overline{(L_2 \setminus L_1) \cap L_{am}}\right)\right\} \cup \left\{\left[(L_2 \setminus L_1) \setminus \overline{L_{1m}}\right]\right.$

$\left. \cap \left[(L_2 \setminus L_1) \setminus \overline{(L_2 \setminus L_1) \cap L_{am}}\right]\right\}$ **(3.28)**

$L_1 \subset L_2$ olmasından dolayı her zaman $L_2 \supseteq \overline{L_{1m}}$ dir. Bu yüzden $(L_2 \setminus L_1) \cap \overline{L_{1m}} = \varnothing$ dir. O zaman $(L_2 \setminus L_1) \setminus \overline{L_{1m}} = (L_2 \setminus L_1)$'dir. (3.28) no.lu eşitliğe bu sonuç uygulanırsa (3.29)'a dönüşür.

$$= \left\{ \left(L_1 \setminus \overline{L_{1m}} \right) \cap \left(L_1 \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right) \right\} \cup \left\{ \left(L_2 \setminus L_1 \right) \cap \left[\left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right] \right\} \quad \textbf{(3.29)}$$

$$= \left\{ \left(L_1 \setminus \overline{L_{1m}} \right) \cap \left(L_1 \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right) \right\} \cup \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\}$$

Diğer bir taraftan $\left(L_1 \cap \overline{L_{2m}} \right) \setminus \overline{L_{1m}} = \varnothing$ olduğundan $L_1 \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} = L_1$. **(3.30)**

(3.29) no.lu eşitliğe (3.30)'u uygulandığında,

$= \left\{ L_1 \setminus \overline{L_{1m}} \right\} \cup \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\}$ 'a dönüşür. (3.27) no.lu eşitlik (3.31) halini alır.

$$L_2 \setminus \overline{L_{2m}} = \left\{ L_1 \setminus \overline{L_{1m}} \right\} \cup \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\} \quad \textbf{(3.31)}$$

$$\beta \left\{ L_2 \setminus \overline{L_{2m}} \right\} = \beta \left\{ \left\{ L_1 \setminus \overline{L_{1m}} \right\} \cup \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\} \right\} \quad \textbf{(3.32)}$$

Aynı zamanda $\left\{ L_1 \setminus \overline{L_{1m}} \right\} \cap \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\} = \varnothing$ olduğundan 3.2.3.2 numaralı özellik gereği (3.32) no.lu denklem (3.33) no.lu denkleme dönüşür,

$$\beta \left\{ L_2 \setminus \overline{L_{2m}} \right\} = \beta \left\{ L_1 \setminus \overline{L_{1m}} \right\} + \beta \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\} \quad \textbf{(3.33)}$$

$$\beta \left\{ L_1 \setminus \overline{L_{1m}} \right\} = \beta \left\{ L_2 \setminus \overline{L_{2m}} \right\} - \beta \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\} \quad \textbf{(3.34)}$$

(3.26) ve (3.34) no.lu denklemleri taraf tarafa toplarsak (3.35) elde edilir.

$$\beta \left\{ L_1 \setminus \overline{L_{1m}} \right\} + \beta \left\{ L_{am} \setminus L_{1m} \right\} = \beta \left\{ L_2 \setminus \overline{L_{2m}} \right\} + \beta \left\{ L_{am} \setminus L_{2m} \right\} - \beta \left\{ \left(L_2 \setminus L_1 \right) \setminus \overline{\left(L_2 \setminus L_1 \right) \cap L_{am}} \right\}$$

$$+ \beta \left\{ \left(L_2 \setminus L_1 \right) \cap L_{am} \right\} \quad \textbf{(3.35)}$$

Sonuç olarak $\hat{J}\left(L_1 \right) = \hat{J}\left(L_2 \right) - F\left(L_2 \setminus L_1 \right)$

Bu önerme iki farklı dilin verilen koşulları sağlaması halinde performans ölçüleri arasındaki ilişkinin ilişkiyi ifade etmektedir. Burada $\hat{J}\left(L_1 \right)$ ve $\hat{J}\left(L_2 \right)$, L_1 ve L_2 dillerine ait performans ölçülerini ifade ederken $F\left(L_2 \setminus L_1 \right)$ ise $L_2 \setminus L_1$ diline ait kayıp faktörünü ifade eder. 3.3.5 numaralı önerme $\left(L_1 \cap \overline{L_{2m}} \right) \setminus \overline{L_{1m}} = \varnothing$ koşulu $BM\left(L_1 \right) \subseteq BM\left(L_2 \right)$ ilişkisinin her zaman sağlanmasını garanti eder.

$L \in L_{cand}$ uygun bir dil ve $s_i \in BM(L)$ bu dile ait bir kilitlenme olsun. Her zaman $\left((L \setminus s_i)^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \subset L$ olduğundan 3.3.5 önermesi bu iki dile uygulanabilir. Eğer $F(r(L,s_i)) \leq 0$ ise 3.3.5 önermesine göre $\hat{J}\left(\left((L \setminus s_i)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) \geq \hat{J}(L)$ olur. Diğer bir taraftan eğer $F(r(L,s_i)) > 0$ ise $\hat{J}\left(\left((L \setminus s_i)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) < \hat{J}(L)$'dir.

3.3.6 Önerme

Eğer $L \in L_{cand}$, $(L \cap L_{am})^{\downarrow C} = L$ ve $s_i, s_k \in BM(L)$ ise

$$\left\{\left[\left((L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \setminus s_k\right]^{\uparrow C} \cap L_{am}\right\}^{\downarrow C} = \left\{\left[\left((L \setminus s_k)^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \setminus s_i\right]^{\uparrow C} \cap L_{am}\right\}^{\downarrow C}$$

İspat – L diline ait herhangi iki kilitlenme $s_i, s_k \in BM(L)$ olsun.

O zaman her zaman $L \setminus s_{\mathrm{i}} \subseteq L$ sağlanacaktır.

$$(L \setminus s_{\mathrm{i}})^{\uparrow C} \subseteq L^{\uparrow C} \qquad \textbf{(3.36)}$$

$L \in L_{cand}$ olduğundan $L = L^{\uparrow C}$ dir. (3.36) no.lu denklem sonuçta (3.37)'e dönüşür.

$$(L \setminus s_{\mathrm{i}})^{\uparrow C} \subseteq L \qquad \textbf{(3.37)}$$

$$\left((L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \subseteq (L \cap L_{am})^{\downarrow C} = L$$

$$\left\{\left(\left((L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \setminus s_{\mathrm{k}}\right)^{\uparrow C} \cap L_{am}\right\}^{\downarrow C} \subseteq \left((L \setminus s_{\mathrm{k}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$$

$$\left[\left\{\left(\left((L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \setminus s_{\mathrm{k}}\right)^{\uparrow C} \cap L_{am}\right\}^{\downarrow C} \setminus s_{\mathrm{i}}\right]^{\uparrow C} \subseteq \left[\left((L \setminus s_{\mathrm{k}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C} \setminus s_{\mathrm{i}}\right]^{\uparrow C} \qquad \textbf{(3.38)}$$

$s_i \in BM(L)$ ve $(L \cap L_{am})^{\downarrow C} = L$ gereği $s_i \notin \left((L \setminus s_{\mathrm{i}})^{\uparrow C} \cap L_{am}\right)^{\downarrow C}$ olduğu gözükür. O zaman (3.38) no.lu eşitsizliğin sol tarafı (3.39)'a eşit olur.

$$\left[\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}\setminus s_{\mathrm{i}}\right]^{\uparrow C}$$

$$=\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C} \qquad \textbf{(3.39)}$$

(3.39) no.lu eşitlik (3.38) no.lu eşitliğe yerleştirilirse (3.40) no.lu eşitsizlik elde edilir.

$$\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}\subseteq\left[\left(\left(L\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{i}}\right]^{\uparrow C} \qquad \textbf{(3.40)}$$

$$\left[\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}\cap L_{am}\right]^{\downarrow C}\subseteq\left\{\left[\left(\left(L\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{i}}\right]^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}$$

$$\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}\subseteq\left\{\left[\left(\left(L\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{i}}\right]^{\uparrow C}\cap L_{am}\right\}^{\downarrow C} \qquad \textbf{(3.41)}$$

s_i ve s_k rasgele iki kilitlenme olduğundan (3.41) no.lu eşitsizlik eşitliğe dönüşür.

$$\left\{\left(\left(\left(L\setminus s_{\mathrm{i}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}=\left\{\left[\left(\left(L\setminus s_{\mathrm{k}}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_{\mathrm{i}}\right]^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}$$

3.3.6 önermesi $L\in L_{cand}$ olacak şekilde bir dile kendisine ait herhangi iki kilitlenme üzerinden gösterildiği gibi bir işlem yapılması halinde, bu iki kilitlenmenin uygulama sırasının ulaşılan sonuç dilde bir değişikliği neden olmayacağını garanti eder. Bu önerme dile ait tüm kilitlenmelere genişletilebilinir. $L\in L_{cand}$ ve $s_1,s_2,...,s_m\in BM\left(L\right)$ olmak üzere önermenin genel hali şu şekildedir:

$$\left\{\left[\left(\left(L\setminus s_1\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}...\setminus s_m\right]^{\uparrow C}\cap L_{am}\right\}^{\downarrow C}$$

$$=\left\{\left[\left(\left(L\setminus s_m\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}...\setminus s_1\right]^{\uparrow C}\cap L_{am}\right\}^{\downarrow C} \qquad \textbf{(3.42)}$$

(3.42) numaralı denkleme göre $BM\left(L\right)$'nin elemanları herhangi bir sırada L diline uygulanması halinde ulaşılan son dil her zaman aynıdır. T işlemindeki koşul

sağlanması halinde dile uygulanan işlem ile 3.3.6 önermesinde dile uygulanan işlem aynıdır. Fakat T işlemi bir ön koşula bağlı olarak işlem yaptığından $BM(L)$'nin elemanlarının uygulama sırasını değiştirmek işlemin sonucunun her zaman aynı dile gideceğini göstermek için yeterli değildir. Öyle ki $L \in L_{cand}$ olacak şekilde herhangi bir dil için dile ait kilitlenmeler iki farklı sırada ardı ardına T işlemi üzerinden dile uygulanması halinde sonuç aynı olmayabilir. Diğer bir değişle $T\left[T\left(T\left(L,s_1\right),s_2\right),...s_j\right]$, $T\left[T\left(T\left(L,s_j\right),s_{j-1}\right),...s_1\right]$ eşit olmayabilir.

3.3.7 Teorem

$L \in L_{cand}$ uygun bir dil ve $s_i, s_j \in BM(L)$ dile ait iki farklı kilitlenme olsun öyle ki $F\left(r\left(L,s_i\right)\right) > F\left(r\left(L,s_j\right)\right)$ ve $\left(L \cap L_{am}\right)^{\downarrow C} = L$ koşulları sağlanıyor ise $\hat{J}\left(T\left(T\left(L,s_i\right),s_j\right)\right) \le \hat{J}\left(T\left(T\left(L,s_j\right),s_i\right)\right)$ her zaman sağlanır.

İspat –

$F\left(r\left(L,s_i\right)\right) > F\left(r\left(L,s_j\right)\right)$ üç şekilde sağlanır. Bunlar sırasıyla

$$0 > F\left(r\left(L,s_i\right)\right) > F\left(r\left(L,s_j\right)\right) \quad \textbf{(3.43)}$$

$$F\left(r\left(L,s_i\right)\right) > 0 > F\left(r\left(L,s_j\right)\right) \quad \textbf{(3.44)}$$

$$F\left(r\left(L,s_i\right)\right) > F\left(r\left(L,s_j\right)\right) > 0 \quad \textbf{(3.45)}$$

$T\left(T\left(L,s_i\right),s_j\right)$ ve $T\left(T\left(L,s_j\right),s_i\right)$ arasındaki ilişki bu üç koşul için ayrı ayrı incelenecektir.

- $0 > F\left(r\left(L,s_i\right)\right) > F\left(r\left(L,s_j\right)\right)$ olsun.

3.3.5 no.lu önerme gereği $\hat{J}\left(\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) = \hat{J}(L) - F\left(r\left(L,s_i\right)\right)$ koşulu her zaman sağlanır ve $0 > F\left(r\left(L,s_i\right)\right)$ olduğundan $\hat{J}\left(\left(\left(L \setminus s_i\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) > \hat{J}(L)$ olur. O zaman $T\left(L,s_i\right) = L$.

Benzer bir şekilde $0 > F\left(r\left(L,s_j\right)\right)$ olduğundan 3.3.5 no.lu önerme gereği $\hat{J}\left(\left(\left(L \setminus s_j\right)^{\uparrow C} \cap L_{am}\right)^{\downarrow C}\right) > \hat{J}(L)$ olur. Buna göre $s_j \in BM(L)$ için işlem $T\left(L,s_j\right) = L$ olacak şekilde sonuçlanır.

Sonuç olarak $T\left(T\left(L,s_i\right),s_j\right)=L$ ve $T\left(T\left(L,s_j\right),s_i\right)=L$'dir.

$\Rightarrow \hat{J}\left[T\left(T\left(L,s_i\right),s_j\right)\right]=\hat{J}\left[T\left(T\left(L,s_j\right),s_i\right)\right]$. **(3.46)**

- $F\left(r\left(L,s_i\right)\right)>0>F\left(r\left(L,s_j\right)\right)$ olsun.

$0<F\left(r\left(L,s_i\right)\right)$ ve 3.3.5 no.lu önerme gereği bu koşul altında $T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$. Aynı zamanda çıkartılan kelimeleri düşünecek olursak $r\left(L,s_j\right)$ ve $r\left(L,s_i\right)$ için üç farklı durum oluşabilir. Bunlar sırasıyla $r\left(L,s_i\right)\cap r\left(L,s_j\right)=\varnothing$, $r\left(L,s_i\right)\supseteq r\left(L,s_j\right)$ ve $r\left(L,s_i\right)\subseteq r\left(L,s_j\right)$'dir. İşlemin sonuçları bu üç koşul için ayrı ayrı incelenecektir.

$\Rightarrow$ $r\left(L,s_i\right)\cap r\left(L,s_j\right)=\varnothing$ olsun. O zaman $0>F\left(r\left(L,s_j\right)\right)$ olduğundan her zaman $\hat{J}\left(\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)<\hat{J}\left(\left[\left(\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\setminus s_j\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)$ olacaktır.

Sonuçta $T\left(T\left(L,s_i\right),s_j\right)=T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ **(3.47)**

$0>F\left(r\left(L,s_j\right)\right)$ gereğince $T\left(L,s_j\right)=L$ olur. Aynı zamanda $0<F\left(r\left(L,s_i\right)\right)$ olduğundan $T\left(T\left(L,s_j\right),s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ **(3.48)**

Sonuç olarak $r\left(L,s_i\right)\cap r\left(L,s_j\right)=\varnothing$ için (3.47) ve (3.48) no.lu denklemlere göre $T\left(T\left(L,s_i\right),s_j\right)=T\left(T\left(L,s_j\right),s_i\right)$dir.

O zaman $\hat{J}\left[T\left(T\left(L,s_i\right),s_j\right)\right]=\hat{J}\left[T\left(T\left(L,s_j\right),s_i\right)\right]$. **(3.49)**

$\Rightarrow$ $r\left(L,s_i\right)\supseteq r\left(L,s_j\right)$ olsun. $s_j\notin T\left(L,s_i\right)$ olduğundan

$T\left(T\left(L,s_i\right),s_j\right)=T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$. **(3.50)**

Diğer bir taraftan $0>F\left(r\left(L,s_j\right)\right)$ olduğundan $T\left(L,s_j\right)=L$ ve bunun sonucunda $T\left(T\left(L,s_j\right),s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ **(3.51)**

Sonuç olarak $r\left(L,s_i\right)\supseteq r\left(L,s_j\right)$ için (3.50) ve (3.51) no.lu denklemlere göre $T\left(T\left(L,s_i\right),s_j\right)=T\left(T\left(L,s_j\right),s_i\right)$dir.

$$\hat{J}\left[T\left(T\left(L,s_i\right),s_j\right)\right]=\hat{J}\left[T\left(T\left(L,s_j\right),s_i\right)\right] \quad \textbf{(3.52)}$$

$\Rightarrow \quad r\left(L,s_i\right)\subseteq r\left(L,s_j\right)$ olsun.

$\left(\left(\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\cap\overline{L_m}\right)\setminus\overline{\left(\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\cap L_{am}}=\varnothing$ olacağından

$F\left(r\left(T\left(L,s_i\right),s_j\right)\right)=F\left(r\left(L,s_j\right)\right)-F\left(r\left(L,s_i\right)\right)$ olur. O zaman

$F\left(r\left(T\left(L,s_i\right),s_j\right)\right)<0$ olduğundan

$$T\left(T\left(L,s_i\right),s_j\right)=T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}. \quad \textbf{(3.53)}$$

Diğer bir taraftan $0>F\left(r\left(L,s_j\right)\right)$ olduğundan $T\left(L,s_j\right)=L$ 'dir. O zaman

$$T\left(T\left(L,s_j\right),s_i\right)=T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C} \quad \textbf{(3.54)}$$

Sonuç olarak $r\left(L,s_i\right)\subseteq r\left(L,s_j\right)$ için (3.53) ve (3.54) no.lu denklemlere göre

$$T\left(T\left(L,s_i\right),s_j\right)=T\left(T\left(L,s_j\right),s_i\right)$$

$$\hat{J}\left[T\left(T\left(L,s_i\right),s_j\right)\right]=\hat{J}\left[T\left(T\left(L,s_j\right),s_i\right)\right] \quad \textbf{(3.55)}$$

- $F\left(r\left(L,s_i\right)\right)>F\left(r\left(L,s_j\right)\right)>0$ olsun.

Burada $0<F\left(r\left(L,s_i\right)\right)$ sağladığından ötürü $T\left(L,s_i\right)=\left[\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ her zaman sağlanır.

$\Rightarrow \quad r\left(L,s_i\right)\cap r\left(L,s_j\right)=\varnothing$ olsun.

3.3.5 no.lu önerme ve $F\left(r\left(L,s_i\right)\right)>F\left(r\left(L,s_j\right)\right)>0$ gereği

$$T\left(T\left(L,s_i\right),s_j\right)=\left(\left[\left(\left(L\setminus s_i\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_j\right]^{\uparrow C}\cap L_{am}\right)^{\downarrow C} \quad \textbf{(3.56)}$$

$$T\left(T\left(L,s_j\right),s_i\right)=\left(\left[\left(\left(L\setminus s_j\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\setminus s_i\right]^{\uparrow C}\cap L_{am}\right)^{\downarrow C} \quad \textbf{(3.57)}$$

3.3.6 no.lu önerme gereği (3.56) ve (3.57) birbirine eşittir. O zaman

$$\hat{J}\left[T\left(T\left(L,s_i\right),s_j\right)\right]=\hat{J}\left[T\left(T\left(L,s_j\right),s_i\right)\right] \quad \textbf{(3.58)}$$

$\Rightarrow \quad r(L,s_i) \supseteq r(L,s_j)$ olsun. Bu sefer $s_j \notin T(L,s_i)$ olduğundan

$$T(T(L,s_i),s_j) = T(L,s_i) = \left[(L \setminus s_i)^{\uparrow C} \cap L_{am} \right]^{\downarrow C} \qquad \textbf{(3.59)}$$

Diğer bir taraftan $F(r(L,s_j)) > 0$ olduğundan $T(L,s_j) = \left[(L \setminus s_j)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$ olur. O zaman $F(r(T(L,s_j),s_i)) = F(r(L,s_i)) - F(r(L,s_j))$ ve $F(r(L,s_i)) > F(r(L,s_j)) > 0$ gereğince $F(r(T(L,s_j),s_i)) > 0$. Sonuç olarak $T(T(L,s_j),s_i) = \left[\left(\left((L \setminus s_j)^{\uparrow C} \cap L_{am} \right)^{\downarrow C} \setminus s_i \right)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$. Aynı zamanda $r(L,s_i) \supseteq r(L,s_j)$ ve 3.3.6 no.lu önerme gereğince $T(T(L,s_i),s_j) = T(T(L,s_j),s_i)$

Sonuçta $\hat{J}\left[T(T(L,s_i),s_j) \right] = \hat{J}\left[T(T(L,s_j),s_i) \right]$ **(3.60)**

$\Rightarrow \quad r(L,s_i) \subseteq r(L,s_j)$ olsun. Bu sefer $T(L,s_i) = \left[(L \setminus s_i)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$.

Aynı zamanda, $F(r(L,s_i)) > F(r(L,s_j)) > 0$ gereğince

$$F(r(T(L,s_i),s_j)) = F(r(L,s_j)) - F(r(L,s_i)) < 0. \qquad \textbf{(3.61)}$$

O zaman (3.61) gereğince $T(T(L,s_i),s_j) = T(L,s_i) = \left[(L \setminus s_i)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$.

Diğer bir taraftan $T(L,s_j) = \left[(L \setminus s_j)^{\uparrow C} \cap L_{am} \right]^{\downarrow C}$ ve $r(L,s_i) \subseteq r(L,s_j)$ olmasından dolayı $s_i \notin T(L,s_j)$.

Sonuçta $T(T(L,s_j),s_i) = T(L,s_j) = \left((L \setminus s_j)^{\uparrow C} \cap L_{am} \right)^{\downarrow C}$

$F(r(L,s_i)) > F(r(L,s_j)) > 0$ olmasından dolayı $\hat{J}\left[T(L,s_i) \right] < \hat{J}\left[T(L,s_j) \right]$

Sonuç olarak $\hat{J}\left[T(T(L,s_i),s_j) \right] < \hat{J}\left[T(T(L,s_j),s_i) \right]$ **(3.62)**

(3.46), (3.49), (3.52), (3.55), (3.58), (3.60) ve (3.62) no.lu denklemlerin sonuçları birleştirirse $\hat{J}\left[T(T(L,s_i),s_j) \right] \leq \hat{J}\left[T(T(L,s_j),s_i) \right]$ elde edilir.

3.3.7 numaralı kuram, $F\left(r\left(L,s_i\right)\right)>F\left(r\left(L,s_j\right)\right)$ koşulunun sağlanması halinde $T\left(T\left(L,s_i\right),s_j\right)$ diline ait performansın her zaman $T\left(T\left(L,s_j\right),s_i\right)$ diline ait performanstan küçük eşit olduğunu ifade etmektedir. Burada $s_i,s_j\in BM\left(L\right)$ kilitlenmeleri rasgele seçilmiş $L\in L_{cand}$ diline ait kilitlenmelerdir. Bu yüzden bu iki kilitlenme için elde edilmiş sonuç çok rahatlıkla diğer kilitlenmelere de genişletilebilinir.

3.3.7.1 Örnek: Şekil 3.1'deki G_8 otomatını inceleyelim. S_3 denetimsel gözetleyicisinin kontrolü altında sistemin ürettiği dil $L=L\left(S_3/G_8\right)=\left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5},\ \alpha_1\alpha_2\eta_2,\overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3},\overline{\alpha_1\alpha_2\alpha_6\alpha_8\eta_3}\right\}$ olsun. Aynı zamanda kontrol edilemeyen olaylar kümesi ve kabul edilebilir dil ise sırasıyla $\sum_{uc}=\left\{\eta_1,\eta_2,\eta_3\right\}$ ve $L_{am}=\left\{\alpha_1,\alpha_1\alpha_2\alpha_4\alpha_5,\alpha_1\alpha_2\alpha_6\alpha_7,\alpha_1\alpha_2\alpha_6\alpha_8\right\}$ olsun. Bazı kelimelere ilişkin ederler ise şu şekilde $c_s\left(\alpha_1\alpha_2\eta_2\right)=2$, $c_s\left(\alpha_1\alpha_2\alpha_4\alpha_5\right)=5$, $c_s\left(\alpha_1\alpha_2\alpha_6\alpha_7\right)=7$, $c_s\left(\alpha_1\alpha_2\alpha_6\alpha_8\right)=4$, $c_s\left(\alpha_1\alpha_2\alpha_6\alpha_7\eta_3\right)=2$ ve $c_s\left(\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)=16$ verilmiş olsun.

Sırasıyla $T\left(T\left(L,\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right),\alpha_1\alpha_2\eta_2\right)$ ve $T\left(T\left(L,\alpha_1\alpha_2\eta_2\right),\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)$ elde edilecek ve iki dil karşılaştırılacaktır.

Verilen kelimelerin ederlerine göre $\hat{J}\left(L\right)=20$'dir. $T\left(T\left(L,\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right),\alpha_1\alpha_2\eta_2\right)$ ulaşabilmek için ilk olarak $\left[\left(L\setminus\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ elde edilir ve $\hat{J}\left(\left[\left(L\setminus\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)=8$ olarak hesaplanır. Bu elde edilen performans L diline ait performanstan küçük olduğundan T işlemindeki koşul sağlanır ve $L_1=T\left(L,\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)=\left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5},\alpha_1\alpha_2\eta_2,\overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}\right\}$ olarak elde edilir. Ardından diğer kilitlenme dikkate alınır. $\left[\left(L_1\setminus\alpha_1\alpha_2\eta_2\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ dili elde edilir ve $\hat{J}\left(\left[\left(L_1\setminus\alpha_1\alpha_2\eta_2\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)=16$ hesaplanır. $\hat{J}\left(\left[\left(L_1\setminus\alpha_1\alpha_2\eta_2\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)>\hat{J}\left(L\right)$ olduğundan $T\left(L_1,\alpha_1\alpha_2\eta_2\right)=L_1$ olur.

Sonuç olarak $T\left(T\left(L,\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right),\alpha_1\alpha_2\eta_2\right)=\left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5},\alpha_1\alpha_2\eta_2,\overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}\right\}$ dir ve performansı 8'e eşittir.

Benzer bir şekilde $T\left(T\left(L,\alpha_1\alpha_2\eta_2\right),\alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)$ işleminin sonucuna ulaşabilmek için $\left[\left(L\setminus\alpha_1\alpha_2\eta_2\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}$ elde edilir ve $\hat{J}\left(\left[\left(L\setminus\alpha_1\alpha_2\eta_2\right)^{\uparrow C}\cap L_{am}\right]^{\downarrow C}\right)=16$

hesaplanır. $\hat{J}\left(\left[\left(L \setminus \alpha_1\alpha_2\eta_2\right)^{\uparrow C} \cap L_{am}\right]^{\downarrow C}\right) < J\left(L\right)$ olduğundan $T\left(L, \alpha_1\alpha_2\eta_2\right) = \left\{\overline{\alpha_1\alpha_2}\right\}$ olarak elde edilir. Aynı zamanda $\alpha_1\alpha_2\alpha_6\alpha_8\eta_3 \notin T\left(L, \alpha_1\alpha_2\eta_2\right)$ olduğu için $T\left(T\left(L, \alpha_1\alpha_2\eta_2\right), \alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right) = \left\{\overline{\alpha_1\alpha_2}\right\}$'e eşittir ve performansı 16'dır. Elde edilen iki sonuç dilin performansları arasıdaki ilişki şu şekildedir:

$$\hat{J}\left(\left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5}, \alpha_1\alpha_2\eta_2, \overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}\right\}\right) < \hat{J}\left(\left\{\overline{\alpha_1\alpha_2}\right\}\right) \quad \textbf{(3.63)}$$

L dilinin $\alpha_1\alpha_2\eta_2$ ve $\alpha_1\alpha_2\alpha_6\alpha_8\eta_3$ kilitlenmeleri için kayıp faktörlerini hesaplanırsa $F\left(r\left(L, \alpha_1\alpha_2\eta_2\right)\right) = 4$ ve $F\left(r\left(L, \alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)\right) = 12$ olarak elde edilir. O zaman 3.3.7 numaralı kurama göre her zaman $\hat{J}\left(T\left(T\left(L, \alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right), \alpha_1\alpha_2\eta_2\right)\right)$ $< \hat{J}\left(T\left(T\left(L, \alpha_1\alpha_2\eta_2\right), \alpha_1\alpha_2\alpha_6\alpha_8\eta_3\right)\right)$ koşulu sağlanır. Bu da (3.63) ifadesiyle örtüşmektedir. Hesaplanan kayıp faktörlerine göre sonuç iki dilin performansları arasındaki ilişki $\hat{J}\left(\left\{\overline{\alpha_1\alpha_2\alpha_4\alpha_5}, \alpha_1\alpha_2\eta_2, \overline{\alpha_1\alpha_2\alpha_6\alpha_7\eta_3}\right\}\right) < \hat{J}\left(\left\{\overline{\alpha_1\alpha_2}\right\}\right)$ olur.

3.4 Optimizasyon Algoritması

Kilitlenebilir denetimsel gözetleyici probleminde çözüm olarak seçilecek denetimsel gözetleyiciyle ilgili iki özellik ön plana çıkar: Denetimsel gözetleyicinin istenilen işaretli dilin ne kadarını işaretleyebildiği ve ürettiği dilin performansı. İstenilen işaretli dile en fazla izin veren ve en düşük performans ederine sahip kilitlenebilir denetimsel gözetleyici bu problem için en uygun çözümdür. Tablo 3.1'de istenilen bu iki koşula göre çözüm üreten yeni bir algoritma verilmiştir. Algoritmada $BM\left(L_{\text{temp 1}}\right)\{i\}$, $BM\left(L_{\text{temp 1}}\right)$ kümesinin i. elemanını ifade etmektedir. Algoritmaya giriş olarak sisteme ait otomat ve sistemden beklentileri içeren dil girilmelidir.

Tablo 3.1 : Optimizasyon algoritması

1. Adım $L_{\text{IS}} = L_{am}^{\downarrow C}$ hesapla. $L_{\text{temp 1}} = L_{\text{IS}}$ olarak kabul et.
2. Adım $BM\left(L_{\text{temp 1}}\right) = \{s_1, s_2, s_3, ..., s_n\}$'yi bul. $\forall s_i \in BM\left(L_{\text{temp 1}}\right)$ için $F\left(r\left(L_{\text{temp 1}}, s_i\right)\right)$'yi hesapla.

$BM\left(L_{\text{temp 1}}\right)$ kümesini kayıp faktörüne göre azalacak şekilde sıralı hale getir öyle ki $F\left(r\left(L_{temp1}, s_i\right)\right) \geq F\left(r\left(L_{temp1}, s_{i+1}\right)\right)$ olsun.

3. Adım

For $i = 1$ *to* $\left\|BM\left(L_{\text{temp 1}}\right)\right\|$

$\quad \text{A} = BM\left(L_{\text{temp 1}}\right)\{i\}$

$\quad L_{\text{x}} = T\left(L_{\text{temp i}},\ \text{A}\right)$

$\quad L_{\text{temp i+1}} = L_{\text{x}}$

end

$L_{rst} = L_{\text{temp i+1}}$

4. Adım

Eğer $L_{\text{rst}} \neq L_{\text{temp 1}}$ ise

$\quad L_{\text{temp 1}} = L_{\text{rst}}$ 2.adım'a git.

değilse

$\quad L_{\text{FS}} = L_{\text{rst}}$

Tablo 3.1'de önerilen algoritmanın yapısı karmaşık değildir. 1. adımda tam başarımlı çözüm hesaplanmakta ve bu başlangıç dili olarak kabul edilmektedir. 2. adımda tam başarımlı çözüme ait kilitlenmeler bulunmaktadır. Modele ait dil düzenli bir dil olduğundan tam başarımlı çözüme ait kilitlenmeler de sonlu sayıda olacaktır. Bu durumda $BM\left(L_{\text{temp 1}}\right) = \left\{s_1, s_2, s_3, ..., s_n\right\}$ gibi bir küme ile gösterilebilir. Aynı zamanda ikinci adımda tam başarımlı çözüme ait her bir kilitlenme için kayıp faktörleri hesaplanmaktadır. Bu hesaplanan kayıp faktörleri üzerinden büyükten küçüğe doğru $BM\left(L_{\text{temp 1}}\right)$ kümesi yeniden sıralanır. 3. adımda ise bu sıralanmış kilitlenmelere göre T işlemi sırayla uygulanır. Son olarak 4. adımda ise 3.adımın sonunda elde edilen L_{rst} ile 3. adımın başında girilen $L_{\text{temp 1}}$ karşılaştırılır. Eğer bu iki dil eşit ise L_{rst} sonuç dil olarak kabul edilir ve algoritma sonlandırılır. Eğer bu iki dil eşit değil ise $L_{\text{temp 1}}$ olarak L_{rst} alınır ve 2. adımdan başlayarak yeniden aynı işlemler yinelenir.

3.4.1 Belirtme

Algoritmanın temelini 3.3.4'de verilen T işlemi ve onunla ilgili elde edilen sonuçlar oluşturmaktadır. T işlemi, $L \in L_{cand}$ olacak şekilde bir dil ve dile ait herhangi bir

kilitlenme için tanımlanmıştır. Kilitlenmeyi dilden tanımlandığı gibi çıkarttıktan sonra elde edilen dilin performansı, L diline ait olan performans ile karşılaştırılır; daha iyi ise elde edilen dil sonuç dil olarak kabul edilir, iyi değil ise L dili değiştirilmez. Algoritmanın 3. adımında T işlemi, başlangıç diline kendisine ait tüm kilitlenmeler üzerinden uygulanır. T işlemi performans değişimine bağlı olarak tanımlandığından dile ait kilitlenmeler farklı sıralamalar ile uygulanması halinde ulaşılan sonuç dil farklılık gösterebilir. Ulaşılan dilin farklı olması dillerin performanslarının da farklı olmasına neden olur. Amaçlardan birisi en düşük performans ederine sahip olan dili elde etmek olduğundan, kilitlenmelerin dile uygulama sırası önemlidir. 3.3.7 numaralı teorem gereği eğer dile ait iki kilitlenmeden kayıp faktörü büyük olan önce T işlemine tabii tutulursa elde edilen sonuç dil diğer şekilde elde edilecek dilden daha düşük bir performans ederine sahip olur. Buna göre algoritmadaki 2. adımda dile ait kilitlenmeler kayıp faktörlerine göre büyükten küçüğe doğru sıralanır. Bu da algoritmanın 3. adımında tüm kilitlenmeler için yapılan T işleminin sonucuna ait performans ederinin diğer sıralamalar sonucunda elde edilen dillerin performans ederlerinden küçük olmasını garanti eder.

Optimal kilitlenebilir denetimsel gözetleyici probleminde amaç, sadece en düşük performans ederine sahip olan denetimsel gözetleyiciyi elde etmek değil aynı zamanda istenilen işaretli kelimenin de olabildiğince çok üretilmesini sağlamaktır. Bu noktada başlangıç dili olarak L_{cand} kümesine ait herhangi bir dili almaktansa, TBÇ'yi başlangıç dili olarak almak bizi en fazla işaretli kelimeyi üreten en düşük performans ederine sahip olan dile götürecektir. Zira tam başarımlı çözüm tüm istenilen kelimeleri içermektedir ve tüm istenilen kelimeler algoritma içerisinde değerlendirilmiş olunur. Böylelikle sonuç dilin en fazla istenilen işaretli kelimeyi üretmesi garanti altına alınır. 2, 3 ve 4 no.lu adımlar gereğince belli bir düzen içerisinde bazı kilitlenmeler düşer ve en fazla işaretli kelimeyi üreten en düşük performans ederine sahip olan dile ulaşılır.

3.5 Uygulama

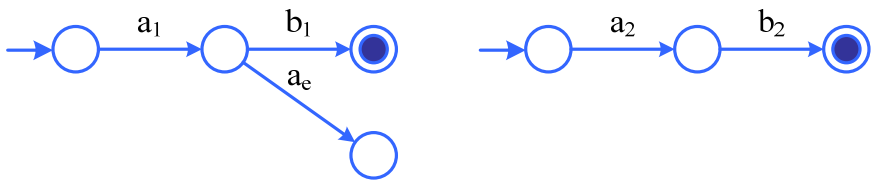
Tablo 3.1'de verilen algoritma bir veri yönetim sistemi üzerinde uygulanacaktır. Uygulama olarak veri yönetim sisteminin seçilmesinin en büyük sebebi problemin yapısı gereği kilitlenmeye sahip olması ve uygulamada kilitlenebilir denetimsel gözetleyicilerin veri yönetim sistemlerinde denetimsel gözetleyici olarak tercih edilmesidir. Veri yönetim sistemleri kilitlenme bulma yapılarını da içermektedir. Böylelikle performansı arttırmak için belli bir orada kilitlenmeye izin verilmektedir.

Kilitlenme bulma algoritmaları ve bunların nasıl uygulanacağı bu çalışmanın konusu dışındadır.

Bir veri yönetim sistemi, birden fazla kullanıcının rastlantısal olarak veri tabanı üzerinde düzenleme ve güncelleme yaparken, veri tabanının tutarlı kalmasını garanti altına alan sistemlerdir. Veri yönetim sistemlerinde ardışık olarak her bir kullanıcının okuma-yazma yapması geçiş olarak adlandırılır. Rastlantısal olarak birden fazla geçişin oluşması halinde veritabanının tutarlılığını koruyabilmek için veritabanı yönetim sistemi geçiş işlemlerinin birbirlerine göre ardı sıra olan geçiş sıralarını tasarlar. Bu açıdan bakıldığında doğru çözüme ulaşabilmek için birden fazla geçişin birbiri içindeki hareketi tamamen engellemeye gerek yoktur. Bu da veri yönetim sisteminin performansını artmasına olanak sağlar. Bunun yanı sıra birçok eşzamanlı kontrol protokolleri kilitlenme bulma yapıları ile beraber çalıştıklarından tamamen kilitlenmeyi engellemek pratik bir çözüm değildir.

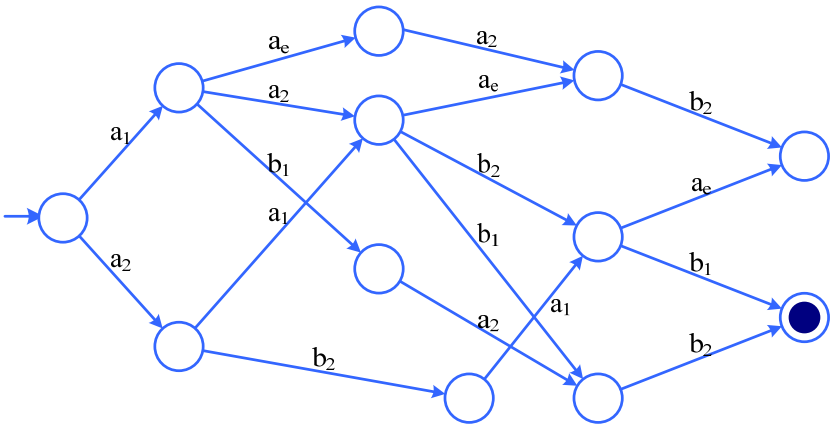
$T_1 = a_1 b_1$ ve $T_2 = a_2 b_2$ gibi iki geçişimizin olduğunu düşünelim öyle ki bu iki geçişin paralel çalışması ile oluşacak olan veri sistemine ilişkin olay kümesi şu şekilde olsun $\sum = \{a_1, b_1, a_2, b_2, a_e\}$. Aynı zamanda kontrol edilemeyen olay kümesi ise şu şekildedir: $\sum_{uc} = \{a_e\}$.

T_1 ve T_2 geçişlerine ilişkin olarak otomat modelleri Şekil 3.2'deki gibi verilebilir. T_1 geçişinde a_1 olayının olmasının ardından bir hatanın oluşabildiğini düşünelim öyle ki bu hatayı a_e ile gösterelim. Burada gösterimde kolaylık olması açısından yapılan işlemin okuma mı yazma mı olduğu önemsiz sayılmıştır.



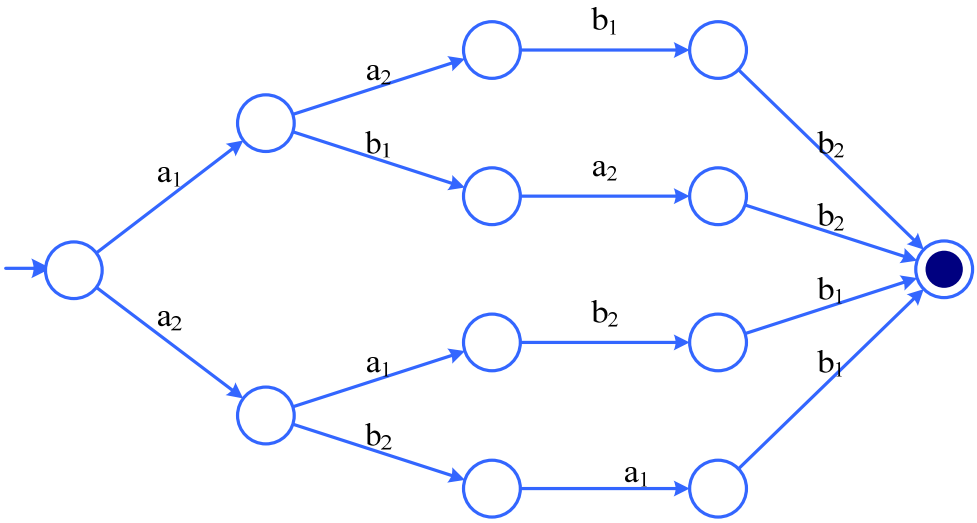
Şekil 3.2 : T_1 ve T_2 geçişlerine ait G_9 ve G_{10} otomatları

Bu iki otomat paralel çalıştığında elde edilen veri tabanı sistemine ait olan otomat modeli ise Şekil 3.3'deki gibidir.



Şekil 3.3 : Veritabanı sistemine ilişkin G_{11} otomatı

Eşzamanlı veritabanı kontrol teorisi gereğince kabul edilebilir işaretli dilde a_1'den sonra a_2 ancak b_1'den sonra b_2'nin üretilmesi halinde kabul edilebilir. O zaman Şekil 3.3'de verilen veri sistem için kabul edilebilir işaretli dil kümesi $L_{am} = \{a_1b_1a_2b_2, a_2b_2a_1b_1, a_1a_2b_1b_2, a_2a_1b_2b_1\}$ şeklindedir. Kabul edilebilir işaretli dili ifade eden H otomatı Şekil 3.4'de, sistem olaylarına ilişkin olay ederleri Tablo 3.2'de verilmiştir. Bu verilen olay ederleri üzerinden en fazla işaretli kelimeyi üreten en düşük performans ederine sahip olan dil elde edilecektir.



Şekil 3.4 : H otomatı

Tablo 3.2 : Veritabanı sisteminin ürettiği olay ederleri

$c_s(\varepsilon, a_1)$	2	$c_s(a_2, a_1)$	10
$c_s(a_1, a_2)$	10	$c_s(a_2, b_2)$	12
$c_s(a_1, a_e)$	2	$c_s(a_2a_1, a_e)$	20
$c_s(a_1, b_1)$	5	$c_s(a_2a_1, b_2)$	7
$c_s(a_1a_2, b_1)$	4	$c_s(a_2b_2, a_1)$	15
$c_s(a_1b_1, a_2)$	10	$c_s(a_2a_1b_2, b_1)$	7
$c_s(a_1a_2, a_e)$	15	$c_s(a_2a_1b_2, a_e)$	5
$c_s(a_1a_2b_1, b_2)$	4	$c_s(a_2b_2a_1, b_1)$	25
$c_s(a_1b_1a_2, b_2)$	12	$c_s(a_2b_2a_1, a_e)$	2
$c_s(\varepsilon, a_2)$	10		

Bu verilen olay ederlerine göre veritabanı sisteminin ürettiği kelimelerin ederleri bölüm 3.2'de verilen tanımlar doğrultusunda aşağıdaki gibi elde edilir.

$$c_s(a_1a_2b_1b_2) = \frac{c_s(\varepsilon, a_1) + c_s(a_1, a_2) + c_s(a_1a_2, b_1) + c_s(a_1a_2b_1, b_2)}{\|a_1a_2b_1b_2\|} = 5$$

$$c_s(a_1b_1a_2b_2) = \frac{c_s(\varepsilon, a_1) + c_s(a_1, b_1) + c_s(a_1b_1, a_2) + c_s(a_1b_1a_2, b_2)}{\|a_1b_1a_2b_2\|} = 7.25$$

$$c_s(a_1a_e) = \frac{c_s(\varepsilon, a_1) + c_s(a_1, a_e)}{\|a_1a_e\|} = 2$$

$$c_s(a_1a_2a_e) = \frac{c_s(\varepsilon, a_1) + c_s(a_1, a_2) + c_s(a_1a_2, a_e)}{\|a_1a_2a_e\|} = 9$$

$$c_s(a_2a_1b_2b_1) = \frac{c_s(\varepsilon, a_2) + c_s(a_2, a_1) + c_s(a_2a_1, b_2) + c_s(a_2a_1b_2, b_1)}{\|a_2a_1b_2b_1\|} = 8.5$$

$$c_s(a_2b_2a_1b_1) = \frac{c_s(\varepsilon, a_2) + c_s(a_2, b_2) + c_s(a_2b_2, a_1) + c_s(a_2b_2a_1, b_1)}{\|a_2b_2a_1b_1\|} = 15.5$$

$$c_s\left(a_2a_1a_e\right)=\frac{c_s\left(\varepsilon,a_2\right)+c_s\left(a_2,a_1\right)+c_s\left(a_2a_1,a_e\right)}{\left\|a_2a_1a_e\right\|}=13.33$$

$$c_s\left(a_2a_1b_2a_e\right)=\frac{c_s\left(\varepsilon,a_2\right)+c_s\left(a_2,a_1\right)+c_s\left(a_2a_1,b_2\right)+c_s\left(a_2a_1b_2,a_e\right)}{\left\|a_2a_1b_2a_e\right\|}=8$$

$$c_s\left(a_2b_2a_1a_e\right)=\frac{c_s\left(\varepsilon,a_2\right)+c_s\left(a_2,b_2\right)+c_s\left(a_2b_2,a_1\right)+c_s\left(a_2b_2a_1,a_e\right)}{\left\|a_2b_2a_1a_e\right\|}=9.75\,.$$

1. Adım:

$$L_{IS}=L_{am}^{\downarrow C}=\left\{\overline{a_1b_1a_2b_2},\overline{a_1a_2b_1b_2},\overline{a_2a_1b_2b_1},\overline{a_2b_2a_1b_1},a_1a_2a_e,a_1a_e,a_2a_1a_e,a_2a_1b_2a_e,a_2b_2a_1a_e\right\}$$

$$L_{\text{temp }1}=L_{\text{IS}}$$

2. Adım

$$BM\left(L_{\text{temp }1}\right)=\left\{a_1a_e,a_1a_2a_e,a_2a_1a_e,a_2a_1b_2a_e,a_2b_2a_1a_e\right\}$$

$$F\left(r\left(L_{\text{temp }1},a_1a_e\right)\right)=-1.25\,,\ F\left(r\left(L_{\text{temp }1},a_1a_2a_e\right)\right)=4\,,\ F\left(r\left(L_{\text{temp }1},a_2a_1a_e\right)\right)=12.58\,,$$

$$F\left(r\left(L_{\text{temp }1},a_2a_1b_2a_e\right)\right)=12.58\,,\ F\left(r\left(L_{\text{temp }1},a_2b_2a_1a_e\right)\right)=-5.75$$

Elde edilen kayıp faktörleri değerlerine göre dile ait kilitlenmeleri içeren küme yeniden düzenlenirse $BM\left(L_{\text{temp }1}\right)=\left\{a_2a_1b_2a_e,a_2a_1a_e,a_1a_2a_e,a_1a_e,a_2b_2a_1a_e\right\}$

3. Adım

$i=1$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{1\}=a_2a_1b_2a_e\,.\quad \hat{J}\left[\left(\left(L_{\text{temp }1}\setminus\mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]<\hat{J}\left[L_{\text{temp }1}\right]$

olduğundan $L_{\text{temp }2}=T\left(L_{\text{temp }1},\ \mathrm{A}\right)=\left\{\overline{a_1b_1a_2b_2},\overline{a_1a_2b_1b_2},\overline{a_2b_2a_1b_1},a_1a_2a_e,a_1a_e,a_2b_2a_1a_e\right\}$

$i=2$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{2\}=a_2a_1a_e$'dir. $\mathrm{A}=a_2a_1a_e\notin L_{\text{temp }2}$ olduğundan $\left(\left(L_{\text{temp }2}\setminus\mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}=L_{\text{temp }2}$ sonuç ta $L_{\text{temp }3}=L_{\text{temp }2}$.

$i=3$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{3\}=a_1a_2a_e\,.\quad \hat{J}\left[\left(\left(L_{\text{temp }3}\setminus\mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]<\hat{J}\left[L_{\text{temp }3}\right]$

olduğundan $L_{\text{temp }4}=T\left(L_{\text{temp }3},\ \mathrm{A}\right)=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

$i=4$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{4\}=a_1a_e$'dir. $\hat{J}\left[\left(\left(L_{\text{temp }4}\setminus \mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]>\hat{J}\left[L_{\text{temp }4}\right]$

olduğundan $L_{\text{temp }5}=T\left(L_{\text{temp }4},\ \mathrm{A}\right)=L_{\text{temp }4}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

$i=5$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{5\}=a_2b_2a_1a_e$ dir. $\hat{J}\left[\left(\left(L_{\text{temp }5}\setminus \mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]>\hat{J}\left[L_{\text{temp }5}\right]$

olduğundan $L_{\text{temp }6}=T\left(L_{\text{temp }5},\ \mathrm{A}\right)=L_{\text{temp }5}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

$L_{rst}=L_{\text{temp }6}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

4. Adım

$L_{rst}\neq L_{\text{temp }1}$ olduğundan L_{rst}, $L_{\text{temp }1}$ olarak alınır ve ikinci ve üçüncü adımlar yeniden uygulanır.

2. Adım

$L_{\text{temp }1}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

$BM\left(L_{\text{temp }1}\right)=\left\{a_2b_2a_1a_e,a_1a_e\right\}$

$F\left(r\left(L_{\text{temp }1},a_1a_e\right)\right)=-5.25$, $F\left(r\left(L_{\text{temp }1},a_2b_2a_1a_e\right)\right)=-5.75$

Elde edilen kayıp faktörlerine göre $L_{\text{temp }1}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$ diline ait olan kilitlenmeler yeniden düzenlenirse $BM\left(L_{\text{temp }1}\right)=\left\{a_1a_e,a_2b_2a_1a_e\right\}$ şeklini alır.

3. Adım

$i=1$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{1\}=a_1a_e$'dir. $\hat{J}\left[\left(\left(L_{\text{temp }1}\setminus \mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]>\hat{J}\left[L_{\text{temp }1}\right]$

olduğundan $L_{\text{temp }2}=T\left(L_{\text{temp }1},\ \mathrm{A}\right)=L_{\text{temp }1}$

$i=2$ için $\mathrm{A}=BM\left(L_{\text{temp }1}\right)\{2\}=a_2b_2a_1a_e$'dir.

$\hat{J}\left[\left(\left(L_{\text{temp }2}\setminus \mathrm{A}\right)^{\uparrow C}\cap L_{am}\right)^{\downarrow C}\right]>\hat{J}\left[L_{\text{temp }2}\right]$ olduğundan $L_{\text{temp }3}=T\left(L_{\text{temp }2},\ \mathrm{A}\right)=L_{\text{temp }2}$

$L_{rst}=L_{\text{temp }3}=\left\{\overline{a_1b_1a_2b_2},\overline{a_2b_2a_1b_1},a_1a_e,a_2b_2a_1a_e\right\}$

4. Adım

$L_{rst} = L_{\text{temp 1}}$ olduğundan $L_{\text{FS}} = L_{\text{rst}} = \left\{\overline{a_1 b_1 a_2 b_2}, \overline{a_2 b_2 a_1 b_1}, a_1 a_e, a_2 b_2 a_1 a_e\right\}$

Sonuç olarak veritabanı sistemi için verilen olay ederleri üzerinden en fazla işaretli kelimeyi üreten en düşük performans ederine sahip olan dil $L_{\text{FS}} = \left\{\overline{a_1 b_1 a_2 b_2}, \overline{a_2 b_2 a_1 b_1}, a_1 a_e, a_2 b_2 a_1 a_e\right\}$ olarak bulunmuştur. Bu dile ilişkin kilitlenme ölçüsü ve başarısızlık ölçüsü ise şu şekildedir.

$\widehat{SM^C}\left(L_{FS}\right) = \beta\left\{a_1 a_2 b_1 b_2, a_2 a_1 b_2 b_1\right\} = 13.5$, $\widehat{BM}\left(L_{FS}\right) = \beta\left\{a_1 a_e, a_2 b_2 a_1 a_e\right\} = 11.75$ sonuçta dile ait performans ölçüsü $\hat{J}\left(L_{FS}\right) = \widehat{SM^C}\left(L_{FS}\right) + \widehat{BM}\left(L_{FS}\right) = 25.25$ olacaktır.

Başlangıç diline ait başarısızlık ve kilitlenme ölçüleri ise $\widehat{SM^C}\left(L_{IS}\right) = 0$ ve $\widehat{BM}\left(L_{IS}\right) = \left\{a_1 a_2 a_e, a_1 a_e, a_2 a_1 a_e, a_2 a_1 b_2 a_e, \quad a_2 b_2 a_1 a_e\right\} = 42.08$ şeklindedir. O zaman başlangıç diline ait performans ölçüsü $\hat{J}\left(L_{IS}\right) = 42.08$ 'dir.

$\hat{J}\left(L_{IS}\right) < \hat{J}\left(L_{FS}\right)$ 'dir. Sonuç dilin performansı başlangıç dilene göre daha iyidir. Aynı zamanda tam başarımlı çözümü başlangıç dili olarak seçmekle algoritma, bu performans ederi için en fazla işaretli kelimeyi üreten dili elde etmiştir.

4. AYRIK OLAY SİSTEMLERİ İÇİN ANALİZ AMAÇLI MATLAB TABANLI BİR PAKET PROGRAMI

4.1 Giriş

Ayrık olay sistemleri günümüzde önemli bir araştırma alanı olarak hem akademisyenlerin hem de mühendislerin ilgisini çekmektedir. Ayrık olay sistemlerinin genellikle otomatlar yardımıyla modellendiği düşünüldüğünde ortaya koyulan sistem modeli genellikle karmaşık olmakta ve çok fazla durum içermektedir. Bu nedenle kompleks modelleri el ile analiz ve sentez etmek neredeyse imkânsızdır. Bu noktada analiz ve sentez için paket programlara ihtiyaç duyulmaktadır. Özellikle ayrık olay sistemlerinin karmaşık yapısı göz önüne alındığında bilgisayar ortamında çalışan geliştirmeye açık paket programlar önem kazanmaktadır.

4.2 Ayrık Olay Sistemleri için Geliştirilmiş Paket Programlar

Bugüne kadar ayrık olay sistemlerini analiz ve sentez yapmak için geliştirilen başlıca paket programlar şu şekildedir:

2000 yılında Stephane Lafortune yönetiminde Michigan Üniversitesi DES Grubu tarafından UMDES-LIB geliştirilmiştir [95]. Tamamen C'de yazılmış, DOS ortamında çalışan birden fazla koşturulabilir program parçasından oluşmaktadır. Geliştirilen program parçaları tamamen kapalıdır. DOS işletim sistemi üzerinden program koşturulduğundan, bu şekliyle kullanıcı dostu bir program değildir. Bu eksikliği gidermek için Mount Allison University'de Laurie Ricker ve Ian Fraser önderliğindeki bir çalışma grubu Windows ve Linux işletim sistemlerinde koşturulabilen DESUMA programı geliştirilmiştir [96]. DESUMA Java üzerinden geliştirildiğinden rahatlıkla Windows ya da Linux işletim sistemlerinde kullanılabilme olanağına sahiptir. Program arka planda ayrık olay sistemlerini oluşturmak ve analiz etmek için UMDES-LIB'in program parçalarını kullanır. İnternet üzerinden indirilebilir. Hala geliştirme aşamasındadır. Programda çıkan hatalar düzeltilmeye çalışılmaktadır.

Diğer bir program ise W.M. Wonham kendisi tarafından geliştirilmiş olan XPTCT'dir. Bu program da UMDES-LIB gibi tamamen kapalı bir programdır. DOS ortamında çalıştırılan tek bir koşulabilir dosyadan oluşmaktadır. Son sürümü Temmuz 2006'da yapılmıştır. Bu program da internet üzerinden erişilebilir durumdadır. Programa W.M. Wonham'ın kişisel internet sitesinden ulaşılabilir [97].

Bir diğer program da Knut Åkesson liderliğinde Chalmers Teknoloji Üniversitesinde geliştirilmiş olan SUPREMICA programıdır. SUPREMICA diğer iki program gibi kod açısından kapalı bir programdır. Diğer iki programdan farklı olarak sanayinin kullanım amaçlarına daha fazla paralellik göstermektedir öyle ki sisteme ait model oluşturulmakta ve sistemden beklenilen davranışlar ayrıca ifade edilmektedir. Bu istenilen davranışların gerçeklenebilir olup olmadığı kontrol edilmekte ve davranışlar denetimsel gözetleyici kuramına göre analiz edilebilmektedir. [98, 99].

Zamanlı otomatları gerçeklemek ve analiz etmek için Alexandre David ve Tobias Amnell tarafından Mart 2001'de ilk versiyonu yapılmış olan UPPAAL2k vardır. Bu program da Windows işletim sistemi üzerinde koşturulan kodu kapalı bir paket programdır [100-102].

Bu programlar ayrık olay sistemlerindeki problemlerin çözümü için geliştiriliş olan kuramsal gelişmeyi kısmen program boyutuna taşımış olsalar da kodların kapalı olması yeni geliştirilen algoritma yapılarını test etmeye ve gerçeklemeye olanak tanımamaktadır. Kod geliştirme aşamasında programlar sadece belli yapılar için düşünülmüş, kod geliştirmeye açık platformlar üzerinden sunulmamıştır. Bu da bu programların en büyük eksikliğini oluşturmaktadır.

4.3 Matlab Tabanlı Analiz Programı

Bu çalışmada endüstriyel önemi olan ayrık olay sistemlerini analiz etmek için Matlab üzerinden koşturulabilen bir paket program geliştirilmiştir. Bu kapsamda geliştirilen fonksiyonlar ayrık olay sistemleri için literatürde tanımlı olan işlemleri ve tasarım tekniklerini kapsamaktadır. Aynı zamanda kilitlenebilir denetimsel gözetleyicileri elde etmek için 3. bölümde tanıtılan algoritma da bu tasarlanan fonksiyonlar üzerinden gerçeklenmiş ve test edilmiştir.

Ayrık olay sistemleri yapı itibariyle olay, kelime, dil gibi sembolik ifadeler üzerine kurulmuştur; diferansiyel denklemler veya sayısal temelli denklem takımları ile ifade edilememektedir. Bu bağlamda bilgisayar ortamında tasarım ve analizi kolaylıkla yapabilmek için sembolik tarafı güçlü olan bir paket programın kullanılmasının daha uygun olacaktır. Aynı zamanda bu paket programın ileride yapılabilecek

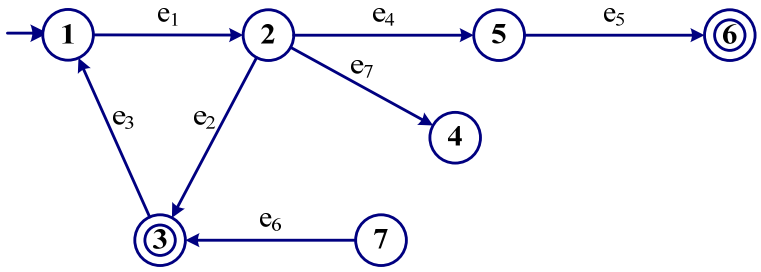
geliştirmelere açık olması, diğer programlarla kolaylıkla haberleşebiliyor olması ve güncel olarak kullanılan ve tercih edilen bir program olması önemlidir. Bu aşamada Mathworks firmasının geliştirmiş olduğu Matlab programı öne çıkmaktadır. Bu programın tercih edilmesinin bir diğer sebebi de güçlü, esnek ve kullanılması kolay bir arayüze sahip olmasıdır.

Ayrık olay sistemleri modellemek ve analiz etmek için Matlab ortamında geliştirilecek olan program parçaları deterministik sonlu durumlu otomatları kapsayacak şekilde hazırlanmıştır. Analiz yöntemi olarak Ramadge ve Wonham'ın ortaya attığı denetimsel kontrol kuramı referans alınmıştır.

4.3.1 Otomatın Oluşturulması

$G=(X,\Sigma,f,x_0,X_m)$ beşlisi üzerinden modellenmiş olan deterministik sonlu durumlu otomatlar Matlab ortamında yapısal(struct) biçimde ifade edilmiştir. $G=(X,\Sigma,f,x_0,X_m)$ şeklinde ifade edilen bir otomatın program içerisinde nasıl ifade edileceği Şekil 4.1'deki G otomatı üzerinden bir örnekle gösterilecektir.

4.3.1.1 Örnek



Şekil 4.1 : G otomatı

G otomatını Matlab'da ifade eden program parçası şu şekildedir.

```
TNM = "0";
G.st = {"1","2","3","4","5","6","7"};
G.ev = {"e1","e2","e3","e4","e5","e6","e7"};
G.tf = {"2"   TNM   TNM   TNM   TNM   TNM   TNM;
        TNM   "3"   TNM   "5"   TNM   TNM   "4";
        TNM   TNM   "1"   TNM   TNM   TNM   TNM;
        TNM   TNM   TNM   TNM   TNM   TNM   TNM;
        TNM   TNM   TNM   TNM   "6"   TNM   TNM;
        TNM   TNM   TNM   TNM   TNM   TNM   TNM;
```

$TNM \quad TNM \quad TNM \quad TNM \quad TNM \quad \text{"3"} \quad TNM\};$

$G.ini = \text{"1"};$

$G.mrk = \{\text{"3"},\text{"6"}\};$

Otomata ait durum, olay ve işaretli durum kümeleri program içerisinde birer liste değişkeni olarak tanımlanmaktadır. Başlangıç durumu ise durum kümesine ait bir durum olacak şeklinde girilmelidir. Bunlardan farklı olarak geçiş fonksiyonu bir matris yapısını andırmasına rağmen sadece geçiş fonksiyonlarının topluca gösteriminden ibarettir. Her bir satır sırasıyla durum kümesinin elemanlarına ve her bir sütün da sırasıyla olay kümesinin elemanlarına karşılık gelmektedir. Mesela 2 no.lu duruma ilişkin geçiş fonksiyonları şu şekildedir: $f(2,e_2)=3$, $f(2,e_4)=5$ ve $f(2,e_7)=4$. O zaman program içerisinde geçiş fonksiyonu ile ilgili 2. satırın ikinci sütuna karşılık düşen eleman "3", dördüncü sütuna karşılık düşen eleman "5" ve son olarak da yedinci sütuna karşılık düşen eleman ise "4" olacaktır.

Aynı zamanda otomatın yapısı gereği her durumda bir olay tanımlı olmayabilir. Bunu ifade etmek için o durumda tanımlı olmayan olayların hizasına TNM yani "0" karakteri yazılır. Bu program içerisinde tanımlanmış özel bir karakterdir ve bu koşulu ifade etmek için kullanılmıştır. G otomatında 2 no.lu durumda e_1, e_3, e_5 ve e_6 olayları tanımlı değildir. Bunu ifade etmek geçiş fonksiyonunda 2 no.lu satırın birinci, üçüncü, beşinci ve altıncı sütunlarına karşı gelen yerlere TNM yani "0" yazılır. "0" karakterine bu şekilde özel bir mana verildiğinden program içerisinde her hangi bir olay ya da durum bu karakter ile kodlanamaz fakat bunun haricinde program içerisinde olaylar ve durumlar istenilen karakter ya da karakter dizisi ile ifade edilebilir. Bunun için bir kısıtlama yoktur.

4.3.2 Geliştirilen Fonksiyonlar

Matlab ortamında deterministik sonlu durumlu otomatlar için şu fonksiyonlara değinilecektir. Bu fonksiyonlar oluşturulurken otomata ait bütün olayların gözlemlenebildiği kabul edilmiştir. Gözlemlenemeyen olayları içeren otomat modeli bu çalışmaya dâhil edilmemiştir.

4.3.2.1 Ulaşılabilir Kısmı Fonksiyonu

Matlab ortamında 4.3.1.1 no.lu örnekteki gibi tanımlanmış bir G otomatının ulaşılabilir kısmını elde etmek için ulaşılabilir kısmı fonksiyonu geliştirilmiştir. Fonksiyon ait algoritma Şekil 4.2'deki gibidir. Algoritma giriş olarak $G=\{X,\Sigma,f,x_0,X_m\}$ otomatını kabul etmekte ve sonuç olarak ta $G_{rst}=\{X_{rst},\Sigma_{rst},f_{rst},x_{rst,0},X_{rst,m}\}$ otomatını üretmektedir.

```
function G_rst = accessible(G)
Σ_rst := Σ , x_rst,0 := x_0 , templist := {x_0}
while templist ≠ ∅
    x := templist{1}
    X_rst := X_rst ∪ {x}
    if x ∈ X_m
        X_rst,m := X_rst,m ∪ {x}
    end
    for each e ∈ Σ
        f_rst(x,e) := f(x,e)
        if f(x,e) ∉ X_rst ∧ f(x,e) ∉ templist
            templist := templist ∪ {f(x,e)}
        end
    end
    templist =: templist \ x
end
X_rst,m = X_rst ∩ X_m
```

Şekil 4.2 : Ulaşılabilir kısmı fonksiyonuna ilişkin algoritma

4.3.2.2 İşaretli Ulaşılabilir Fonksiyonu

Bir G deterministik sonlu durumlu otomatının işaretli-ulaşılabilir parçasını elde etmek için Matlab'da işaretli ulaşılabilir fonksiyonu geliştirilmiştir. Fonksiyonun koşturulması sonucunda bir otomat elde edilir. G otomatının işaretli-ulaşılabilir kısmını elde etmek için Matlab komut satırında *coaccessible*(G) yazılır. Geliştirilen fonksiyonunun algoritmik yapısı Şekil 4.3'teki gibidir.

```
function G_rst = coaccessible(G)
TNM := '0' , Σ_rst := Σ , X_rst,m := X_m , X_rst := X_m
templist := X_m
while templist ≠ ∅
    x := templist{1}
    X_rst := X_rst ∪ {x}
    for each x ∈ X
        for each e ∈ Σ_rst
            y := f(x,e)
```

```
                if y = x  ∧ y ∉ X_rst  ∧ y ∉ templist
                    templist := templist ∪ {x}
                end
            end
        end
        templist := templist \ x
    end
    for each x ∈ X_rst
        for each e ∈ ∑
            if f(x,e) ∉ X_rst
                f_rst(x,e) = TNM
            end
        end
    end
    if x_0 ∈ X_rst
        x_rst,0 = x_0
    else
        x_rst,0 = ' '
    end
```

Şekil 4.3 : İşaretli ulaşılabilir fonksiyonuna ilişkin algoritma

4.3.2.3 Budama Fonksiyonu

Fonksiyon deterministik sonlu durumlu bir otomata budama işlemini uygulamak için geliştirilmiştir. Bu fonksiyonun sonucunda yine bir otomat elde edilir. Matlab komut satırında $trim(G)$ yazmak suretiyle G otomatının budanmış kısmı elde edilir.

```
function G_rst = trim(G)
G_rst = accessible(coaccessible(G))
```

Şekil 4.4 : Budama fonksiyonuna ilişkin algoritma

4.3.2.4 Otomat Tümleyeni Fonksiyonu

Otomat tümleyeni fonksiyonu deterministik sonlu durumlu otomatın tümleyenini almak için geliştirilmiştir. 4.3.1'de ifade edildiği gibi tanımlanan G otomatının tümleyenini almak için Matlab komut satırında $complement(G)$ yazılır.

$TNM := '0'$
$X_{comp} := X \cup \{x_d\}$
$\sum_{comp} := \sum$
$X_{comp,m} := X \cup \{x_d\} \setminus X_m$
$x_{comp,0} := x_0$
for each $x \in X$
    for each $e \in \sum$
        if $f(x,e) = TNM$
            $f_{comp}(x,e) := x_d$
        else
            $f_{comp}(x,e) := f(x,e)$
        end
    end
end

Şekil 4.5 : Otomat tümleyeni fonksiyonuna ilişkin algoritma

4.3.2.5 Otomat Birleşim Fonksiyonu

G_1 ve G_2 gibi iki ayrı deterministik sonlu durumlu otomatın paralel birleşimini elde etmek için otomat birleşim fonksiyonu geliştirilmiştir. Fonksiyonun koşturulması sonucunda elde edilen otomata ait durumların isimleri, birleşim işlemi uygulanan otomatların durum isimlerini birbiri ardına eklenmesiyle elde edilir. G_1 ve G_2 gibi iki otomatın paralel birleşimini almak için Matlab komut satırında $automata\_prl(G_1, G_2)$ komutu yazılır.

function $G_{rst} = automata\_prl(G_1, G_2)$
$TNM := '0'$
$\sum_{rst} := \sum_1 \cup \sum_2$
$x_{0,rst} := 'x_{0,1}, x_{0,2}'$
$templist := \{x_{0,rst}\}$
while $templist \neq \varnothing$
    $x_1 := templist\{1,1\}$
    $x_2 := templist\{1,2\}$
    $newstate := 'x_1, x_2'$
    $X_{rst} := X_{rst} \cup newstate$
    if $x_1 \in X_{1,m} \ \wedge x_2 \in X_{2,m}$
        $X_{rst,m} := X_{rst,m} \cup newstate$
    end

```
for each e ∈ Σ
    y1 := f(x1, e)
    y2 := f(x2, e)
    if e ∈ Σ1 ∧ e ∈ Σ2 ∧ y1 ≠ TNM ∧ y2 ≠ TNM
        frst(newstate, e) := 'y1, y2'
        if 'y1, y2' ∉ Xrst ∧ 'y1, y2' ∉ templist
            templist := templist ⋃ 'y1, y2'
        end
    else if e ∈ Σ1 ∧ e ∉ Σ2 ∧ y1 ≠ TNM
        frst(newstate, e) := 'y1, x2'
        if 'y1, x2' ∉ Xrst ∧ 'y1, x2' ∉ templist
            templist := templist ⋃ 'y1, x2'
        end
    else if e ∉ Σ1 ∧ e ∈ Σ2 ∧ y2 ≠ TNM
        frst(newstate, e) := 'x1, y2'
        if 'x1, y2' ∉ Xrst ∧ 'x1, y2' ∉ templist
            templist := templist ⋃ 'x1, y2'
        end
    else
        frst(newstate, e) := TNM
    end
end
templist := templist \ 'x1, x2'
end
```

Şekil 4.6 : Otomat birleşim fonksiyonuna ilişkin algoritma

4.3.2.6 Otomat Çarpım Fonksiyonu

G_1 ve G_2 iki ayrı deterministik sonlu durumlu otomat olmak üzere bu iki otomatın çarpımını elde etmek için otomat çarpım fonksiyonu geliştirilmiştir. Yeni otomatın durumları, çarpım işlemi uygulanan otomatların durum isimlerinin birbiri ardına eklenmesi suretiyle elde edilir. G_1 ve G_2 gibi iki ayrı deterministik sonlu durumlu otomatın çarpımını elde etmek için Matlab komut satırında $automata\_prd(G_1, G_2)$ yazılır.

```
function G_rst = automata _ prd(G_1, G_2)
x_{0,rst} := 'x_{0,1}, x_{0,2}'
templist := {x_{0,rst}}
while templist ≠ ∅
    x_1 := templist{1,1}
    x_2 := templist{1,2}
    newstate := 'x_1, x_2'
    X_rst := X_rst ∪ newstate
    if x_1 ∈ X_{1,m} ∧ x_2 ∈ X_{2,m}
        X_{rst,m} := X_{rst,m} ∪ newstate
    end
    for each e ∈ Σ
        y_1 := f(x_1, e)
        y_2 := f(x_2, e)
        if y_1 ≠ TNM ∧ y_2 ≠ TNM
            f_rst(newstate, e) := 'y_1, y_2'
            if 'y_1, y_2' ∉ X_rst ∧ 'y_1, y_2' ∉ templist
                templist := templist ∪ 'y_1, y_2'
            end
        else
            f_rst(newstate, e) := TNM
        end
    end
    templist := templist \ 'x_1, x_2'
end
```

Şekil 4.7 : Otomat çarpım fonksiyonuna ilişkin algoritma

4.3.2.7 Kelimenin Önek Kapanışı Fonksiyonu

Bir G deterministik sonlu durumlu otomatının ürettiği herhangi bir kelimenin önek kapanışını elde etmek için kelimenin önek kapanışı fonksiyonu geliştirilmiştir. Matlab'de G otomatının ürettiği s kelimesinin önek kapanışını elde etmek için $prefixclosureofstring(G,s)$ komutu yazılır. Fonksiyona ait algoritma Şekil 4.8'de verilmiştir.

```
// strcon(str1, str2) : str1 ve str2 gibi iki farklı kelimeyi ardıardına ekler.
// strcompare(str1, str2, n) : str1'in baştan n uzunluklu kısmının str2'ye eşit olup
//                             olmamasına bakar
function rst = prefixclosureofstring(G, s)
prestr := ' ' , ls := s , rst := { }
while ||ls|| > 0
    for each e ∈ Σ
        n := ||e||
        if strcompare(ls, e, n) = true
            prestr := strcon(prestr, e)
            rst := rst ∪ {prestr}
            ls := ls(n + 1, ||ls||)
        end
end
```

Şekil 4.8 : Kelimenin önek kapanışı fonksiyonuna ilişkin algoritma

4.3.2.8 Dilin Önek Kapanışı Fonksiyonu

Bir G deterministik sonlu durumlu otomatın ürettiği herhangi bir dilin önek kapanışını elde etmek için dilin önek kapanışı fonksiyonu geliştirilmiştir. Matlab'de G otomatının ürettiği L dilinin önek kapanışını elde etmek için $prefixclosureoflanguage(G, L)$ komutu yazılır. Fonksiyona ait algoritma Şekil 4.9'dadır.

```
function L = prefixclosureoflanguage(G, L)
L := { }
for each s ∈ L
    s̄ := prefixclosureofstring(G, s)
    L := L ∪ s̄
end
```

Şekil 4.9 : Dilin önek kapanışı fonksiyonuna ilişkin algoritma

4.3.2.9 Dilin Önek Kapanışının Denetimi Fonksiyonu

Bir G deterministik sonlu durumlu otomatının ürettiği herhangi bir dilin önek kapalı olup olmadığını kontrol etmek için dilin önek kapanışının denetimi fonksiyonu geliştirilmiştir. Foksiyonu Matlab ortamında koşturabilmek için $checkprefixclosureoflanguage(G, L)$ komutu yazılır. Burada L, G otomatının

ürettiği herhangi bir dildir. Eğer dil önek kapalı ise fonksiyon sonuç olarak lojik 1 ve değil ise lojik 0 sonucunu üretir. Fonksiyona ait algoritma Şekil 4.10'daki gibidir.

```
function rst = checkprefixclosureoflanguage(G, L)
L̄ := prefixclosureoflanguage(G, L)
if L̄ \ L = ∅
    rst := true
else
    rst := false
end
```

Şekil 4.10 : Dilin önek kapanışının denetimi fonksiyonuna ilişkin algoritma

4.3.2.10 Üretilen Kelimeler Fonksiyonu

G deterministik sonlu durumlu otomatının istenilen kelime uzunluğuna kadar ürettiği tüm kelimeler işaretlenen kelimeler fonksiyonu yardımıyla elde edilir**.** G otomatı düzenli fakat sonsuz sayıda kelime üretiyor olabilir. Bu yüzden otomatın ürettiği bütün kelimeleri elde etmek imkânsızdır. Bunun yerine otomatın belli bir uzunluğa kadar ürettiği kelimeleri elde edecek şekilde fonksiyon tasarlanmıştır.

```
function L = generatedstrings(G, k)
templist := {{x0, ' '}}
while templist ≠ ∅
    x := templist{1,1}, s := templist{1,2}
    if ||s|| < k
        for each e ∈ Σ
            y := f(x, e)
            if y ≠ TNM
                L := L ∪ strcon(s, e)
                templist := templist ∪ {y, strcon(s, e)}
            end
        end
    end
    templist := templist \ {x, s}
end
```

Şekil 4.11 : Üretilen kelimeler fonksiyonuna ilişkin algoritma

4.3.2.11 İşaretlenen Kelimeler Fonksiyonu

G deterministik sonlu durumlu otomatının belirlenen kelime uzunluğa kadar ürettiği tüm işaretli kelimeleri elde etmek için işaretlenen kelimeler fonksiyonu tasarlanmıştır**.** Benzer bir şekilde G otomatı düzenli fakat sonsuz sayıda kelime üretiyor olabilir. Bu yüzden otomatın ürettiği tüm işaretli kelimeler gösterilemez. Bunun yerine otomatın belli bir uzunluğa kadar ürettiği işaretli kelimeleri elde edecek şekilde fonksiyon tasarlanmıştır. G otomatının k uzunluğuna kadar ürettiği işaretli kelimeleri elde etmek için komut satırına $markedstrings(G,k)$ komutu yazılır.

```
function L = markedstrings(G,k)
templist := {{x0,' '}}
while templist ≠ ∅
    x := templist{1,1}
    s := templist{1,2}
    if ||s|| < k
        for each e ∈ Σ
            y := f(x,e)
            if y ≠ TNM
                templist := templist ∪ {y, strcon(s,e)}
                if y ∈ Xm
                    L := L ∪ strcon(s,e)
                end
            end
        end
    end
    templist := templist \ {x,s}
end
```

Şekil 4.12 : İşaretlenen kelimeler fonksiyonuna ilişkin algoritma

4.3.2.12 Üretilen Dil Fonksiyonu

G deterministik sonlu durumlu otomatının ürettiği tüm kelimeleri üretilen dil fonksiyonu yardımıyla elde edilir**.** Burada dikkat edilmesi gereken G otomatının yapısında bir döngüsel işlemin olmaması gerektiğidir. Aksi halde otomat sonsuz sayıda kelime ürettiğinden bu fonksiyon kullanılabilir değildir. G otomatının ürettiği kelimeleri elde etmek için Matlab komut satırından $generatedlanguage(G)$ yazılır.

4.3.2.13 İşaretlenen Dil Fonksiyonu

G deterministik sonlu durumlu otomatının ürettiği tüm işaretli kelimeleri işaretlenen dil fonksiyonu yardımıyla elde edilir**.** Generatedlanguage fonksiyonuna benzer şekilde G otomatının yapısında bir döngüsel işlemin olmaması gerekir. G otomatına ait tüm işaretli kelimeleri elde edebilmek için Matlab komut satırında $markedlanguage(G)$ yazılır.

4.3.2.14 Eşit Otomatlar Fonksiyonu

G_1 ve G_2 gibi iki farklı deterministik sonlu durumlu otomatın aynı dilleri üretip üretmediğini kontrol etmek için eşit otomatlar fonksiyonu tasarlanmıştır. Eğer iki otomat eşit ise fonksiyon çıkış olarak mantıksal *"true"* sonucunu üretir ve eğer otomatlar eşit değil ise mantıksal *"false"* sonucunu üretir. Matlab komut satırında $automataequal(G_1, G_2)$ yazmak suretiyle G_1 ve G_2 otomatlarının birbirine eşit olup olmadıkları kontrol edilir. Geliştirilen fonksiyona ilişkin algoritmik yapı Şekil 4.13'de verilmiştir.

$function\ rst = automataequal(G_1, G_2)$
$TNM := '0'$
$if\ \left((\Sigma_1 \setminus \Sigma_2) \cup (\Sigma_2 \setminus \Sigma_1)\right) \neq \varnothing$
    $rst := false$
    $return$
end
$templist := \{\{x_{0,1}, x_{0,2}\}\}$
$while\ templist \neq \varnothing$
    $x_1 := templist\{1,1\}$
    $x_2 := templist\{1,2\}$
    $checked := checked \cup \{x_1, x_2\}$
    for each e $\in \Sigma_1$
        $y_1 := f(x_1, e)$
        $y_2 := f(x_2, e)$
        $if\ (y_1 \neq TNM \wedge y_2 = TNM) \vee (y_1 = TNM \wedge y_2 \neq TNM)$
            $rst := false$
        end
        $if\ (y_1 \neq TNM \wedge y_2 \neq TNM)$
            $if\ (y_1 \in X_m \wedge y_2 \notin X_m) \vee (y_1 \notin X_m \wedge y_2 \in X_m)$

```
                rst := false
                return
            end
            if {y1, y2} ∉ templist ∧ {y1, y2} ∉ checked
                templist := templist ∪ {y1, y2}
            end
        end
    end
    templist := templist \ {x1, x2}
end
rst := true
```

Şekil 4.13 : Eşit otomatlar fonksiyonuna ilişkin algoritma

4.3.2.15 Cansız Kilitlenmeler Fonksiyonu

G deterministik sonlu durumlu otomatta oluşan cansız kilitlenmelerin hangi durumlarda oluştuğunu bulmak için cansız kilitlenmeler fonksiyonu geliştirilmiştir. Matlab komut satırında $blockingstates(G)$ yazmak suretiyle G otomatına ilişkin cansız kilitlenmelerin oluştuğu durumlar bir liste olarak elde edilir.

```
function rst = blockingstates(G)
TNM = '0'
for each x ∈ X
    for each x ∈ X
        nblck := false
        if f(x, e) ≠ TNM
            nblck := true
            break
        end
    end
    if nblck = false ∧ x ∉ Xm
        rst := rst ∪ x
    end
end
```

Şekil 4.14 : Cansız kilitlenmeler fonksiyonuna ilişkin algoritma

4.3.2.16 Otomat İsimlendirme Fonksiyonu

Bir G deterministik sonlu durumlu otomatının durum isimlerinin kullanıcı tarafından belirlenen bir kelimeden başlayacak şekilde yeniden isimlendirilmesi için otomat isimlendirme fonksiyonu geliştirilmiştir. Matlab komut satırında $renameautomata(G,'statename')$ yazarak G otomatına ait durumlar, statename değişkenine yazılan kelimeyi önek alacak şekilde numaralandırılarak yeniden isimlendirilir.

4.3.2.17 Kontrol Edilebilirlik Fonksiyonu

Bir G deterministik sonlu durumlu otomatına, S denetimsel gözetleyicisinin kontrolü altında kazandırılmak istenilen davranışın kontrol edilebilir bir davranış olup olmadığının analizini yapmak için kontrol edilebilirlik fonksiyonu geliştirilmiştir. Matlab komut satırında $controllability(G,S,\Sigma_{uc})$ yazmak suretiyle koşturulur. Burada G deterministik sonlu durumlu otomatına ait kontrol edilemeyen olaylar fonksiyona Σ_{uc} olay listesi ile girilir. Fonksiyon koşturulduktan sonra kazandırılmak istenilen davranış kontrol edilebilir bir davranış ise fonksiyonun çıkışında mantıksal "1" sonucu elde edilir. Eğer kazandırılmak istenilen davranış kontrol edilebilir bir davranış değil ise fonksiyon çıkış olarak mantıksal "0" cevabını üretir. Ayrıca hangi durum çiftlerinde kontrol edilebilirlik koşulunun sağlanmadığı liste şeklinde verilir. "Controllability" fonksiyonuna ilişkin algoritma Şekil 4.15'deki gibidir.

```
function [rst, resultlist] = controllability(G, S, Σuc)
TNM := '0'
if ((ΣG \ ΣS) ∪ (ΣS \ ΣG)) ≠ ∅ ∨ Σuc ⊄ ΣG
    disp('error!')
    return
end
templist := {{x0,G, x0,S}}
while templist ≠ ∅
    xG := templist{1,1}
    xS := templist{1,2}
    checkedlist := checkedlist ∪ {xG, xS}
    for each e ∈ Σ
        yG := f(xG, e)
        yS := f(xS, e)
```

```
        if y_G = TNM ∧ y_S ≠ TNM
            disp('The supervisor is not feasible!')
            return
        end
        if y_G ≠ TNM ∧ y_S = TNM ∧ e ∈ Σ_uc
            rst := false
            resultlist := resultlist ∪ {x_G, x_S}
        end
        if y_G ≠ TNM ∧ y_S ≠ TNM ∧ {y_G, y_S} ∉ templist ∧ {y_G, y_S} ∉ checkedlist
            templist := templist ∪ {x_G, x_S}
        end
    templist := templist \ {x_G, x_S}
end
```

Şekil 4.15 : Kontrol edilebilirlik fonksiyonuna ilişkin algoritma

4.3.2.18 En Büyük Kontrol Edilebilir Alt Dil Fonksiyonu

Bir G deterministik sonlu durumlu otomatının, S denetimsel gözetleyicinin kontrolü altındaki davranışı kontrol edilebilir değil ise en büyük kontrol edilebilir alt dilini bulmak için supcont fonksiyonu geliştirilmiştir. G otomatının S denetimsel gözetleyicisine göre en büyük kontrol edilebilir alt dilini elde etmek için Matlab komut satırında $supcont(G, S, \Sigma_{uc})$ yazılır. G deterministik sonlu durumlu otomatına ait kontrol edilemeyen olaylar fonksiyona Σ_{uc} olay listesi ile girilir.

```
function G_rst = supcont(G, S, Σ_uc)
TNM := '0'
result := controllability(G, S, Σ_uc)
if result.logic := true
    rst := S
    return
end
statelist := result.list(:, 2)'
while statelist ≠ ∅
    x_S := statelist{1}
    dellist := dellist ∪ x_S
    for each e ∈ Σ_uc
```

```
        for each x ∈ X_S
            y_S := f(x,e)
            if y_S = x_S ∧ x ≠ dellist ∧ x ≠ statelist
                statelist := statelist ⋃ x
            end
        end
    end
    statelist := statelist \ x_S
end
Σ_rst = Σ
X_rst := X \ dellist
X_rst,m := X_m ⋂ X_rst
x_rst,0 := x_0
for each x ∈ X_rst
    for each e ∈ Σ_rst
        if f(x,e) ∉ X_rst
            f_rst(x,e) := TNM
        else
            f_rst(x,e) := f(x,e)
    end
end
```

Şekil 4.16 : En büyük kontrol edilebilir alt dil fonksiyonuna ilişkin algoritma

4.3.2.19 En Küçük Kontrol Edilebilir Önek Kapalı Üst Dil Fonksiyonu

Bir G deterministik sonlu durumlu otomatı, S denetimsel gözetleyicisinin kontrolü altında ürettiği dil kontrol edilebilir değil ise en küçük kontrol edilebilir önek kapalı üst dili bulmak için infcont fonksiyonu geliştirilmiştir. G otomatının S denetimsel gözetleyicisine göre infimal kontrol edilebilir üst dilini elde etmek için Matlab komut satırına $infcont(G,S,\Sigma_{uc})$ yazılır. G deterministik sonlu durumlu otomatına ait kontrol edilemeyen olaylar fonksiyona Σ_{uc} ile girilir.

```
function G_rst = infcont(G,S,Σ_uc)
TNM := '0'
result := controllability(G,S,Σ_uc)
```

```
if result.logic = true
    rst := S
    return
end
X_rst := X_S
Σ_rst := Σ
x_rst,0 := x_S,0
X_rst,m := X_S,m
k := 1
statelist := result.list
while statelist ≠ ∅
    x_G := statelist{1,1}
    x_S := statelist{1,2}
    for each e ∈ Σ_uc
        y_G := f(x_G, e)
        y_S := f(x_S, e)
        if y_G ≠ TNM ∧ y_S = TNM
            sname := strcon(_S, int2str(k))
            X_rst := X_rst ∪ {sname}
            f(x_S, e) := sname
            statelist := statelist ∪ {y_G, sname}
            if y_G ∈ X_G,m
                X_rst,m := X_rst,m ∪ {sname}
            end
            k := k + 1;
            for each e ∈ Σ_rst
                f(sname, e) := TNM
            end
        end
    end
    statelist := statelist \ {x_G, x_S}
end
```

Şekil 4.17 : En küçük kontrol edilebilir önek kapalı üst dil fonksiyonuna ilişkin algoritma

4.3.3 Örnekler

- Şekil 4.1'deki G otomatı üzerinde aşağıdaki gibi işlemler yapılırsa;

$$acG = accessible(G)$$
$$coacG = coaccessible(G)$$
$$tG = trim(G)$$
$$accoacG = accessible(coacG)$$
$$result = automataequal(tG, accoacG)$$

Matlab'de şu sonuçlar elde edilir.

$$acG.st = \{'1','2','3','5','4','6'\}$$
$$acG.ev = \{'e1','e2','e3','e4','e5','e6','e7'\}$$
$$acG.tf = \begin{Bmatrix} '2' & '0' & '0' & '0' & '0' & '0' & '0' \\ '0' & '3' & '0' & '5' & '0' & '0' & '4' \\ '0' & '0' & '1' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '6' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '0' & '0' \end{Bmatrix}$$
$$acG.ini = '1'$$
$$acG.mrk = \{'3','6'\}$$

$$coacG.st = \{'3','6','2','7','5','1'\}$$
$$coacG.ev = \{'e1','e2','e3','e4','e5','e6','e7'\}$$
$$coacG.tf = \begin{Bmatrix} '0' & '0' & '1' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '0' & '4' \\ '0' & '3' & '0' & '5' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '3' & '0' \\ '0' & '0' & '0' & '0' & '6' & '0' & '0' \\ '2' & '0' & '0' & '0' & '0' & '0' & '0' \end{Bmatrix}$$
$$coacG.ini = '1'$$
$$acG.mrk = \{'3','6'\}$$

$$tG.st = \{'3','6','2','5','1'\}$$

$$tG.ev = \{'e1','e2','e3','e4','e5','e6','e7'\}$$

$$tG.tf = \begin{Bmatrix} '0' & '0' & '1' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '0' & '0' \\ '0' & '3' & '0' & '5' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '6' & '0' & '0' \\ '2' & '0' & '0' & '0' & '0' & '0' & '0' \end{Bmatrix}$$

$$tG.ini = '1'$$

$$tG.mrk = \{'3','6'\}$$

$$accoacG.st = \{'1','2','3','5','6'\}$$

$$accoacG.ev = \{'e1','e2','e3','e4','e5','e6','e7'\}$$

$$accoacG.tf = \begin{Bmatrix} '2' & '0' & '0' & '0' & '0' & '0' & '0' \\ '0' & '3' & '0' & '5' & '0' & '0' & '0' \\ '0' & '0' & '1' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '6' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' & '0' & '0' \end{Bmatrix}$$

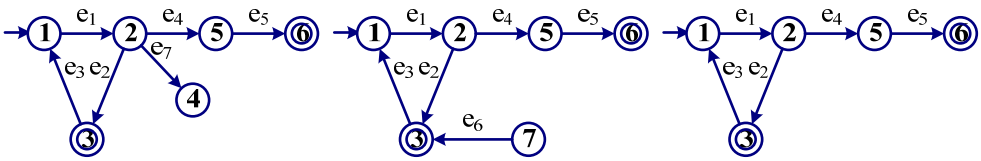
$$accoacG.ini = '1'$$

$$accoacG.mrk = \{'3','6'\}$$

These two automatas are equal!

$$result = 1$$

Elde edilen otomatlar Şekil 4.18'deki gibidir.



Şekil 4.18 : *acG*, *coacG* ve *tG* otomatları

- Benzer bir şekilde, Şekil 4.19'daki pompa, vana ve denetimsel gözetleyici için Matlab'da otomatlar oluşturulduktan sonra aşağıdaki işlemler yapılacak olursa aşağıdaki sonuçlar elde edilir.

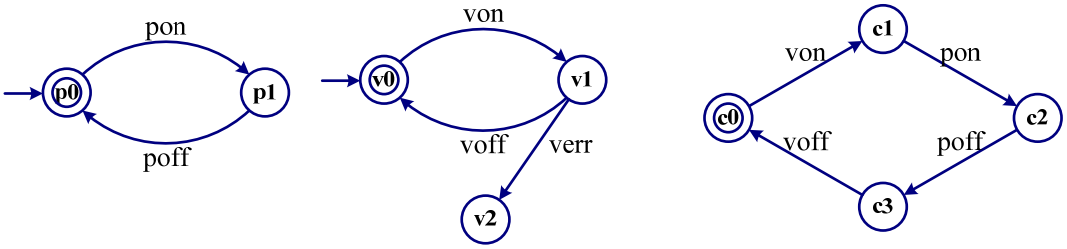
$$Sistem = automata\_prl(Vana, Pompa)$$

$$gstrings = generatedstrings(Sistem, 2)$$

$$mgstrings = markedstrings(Sistem, 2)$$

$$rst = controllability(Sistem, Kont, \{verr\})$$

$$Supden = supcont\left(Sistem, Kont, \{verr\}\right)$$
$$Infden = Infcont\left(Sistem, Kont, \{verr\}\right)$$
$$blkstates = blockingstates\left(Infden\right)$$
$$mrkinfden = markedstrings\left(Infden, 4\right)$$
$$mrksupden = markedstrings\left(Supden, 4\right)$$



Şekil 4.19 : Pompa, vana ve denetimsel gözetleyicilere ait otomatlar

$$sistem.st = \{'p0v0','p1v0','p0v1','p1v1','p0v2','p1v2'\}$$
$$sistem.ev = \{'poff','pon','verr','voff','von'\}$$
$$sistem.tf = \begin{Bmatrix} '0' & 'p1v0' & '0' & '0' & 'p0v1' \\ 'p0v0' & '0' & '0' & '0' & 'p1v1' \\ '0' & 'p1v1' & 'p0v2' & 'p0v0' & '0' \\ 'p0v1' & '0' & 'p1v2' & 'p1v0' & '0' \\ '0' & 'p1v2' & '0' & '0' & '0' \\ 'p0v2' & '0' & '0' & '0' & '0' \end{Bmatrix}$$
$$sistem.ini = 'p0v0'$$
$$sistem.mrk = \{'p0v0'\}$$

$$gstrings = \{pon, von, ponpoff, ponvon, vonpon, vonverr, vonvoff\}$$
$$mgstrings = \{ponpoff, vonvoff\}$$

$$rst.logic = 0$$
$$rst.list = \begin{Bmatrix} 'p0v1' & 'c1' \\ 'p1v1' & 'c2' \\ 'p0v1' & 'c3' \end{Bmatrix}$$

$$Supden.st = \{'c0'\}$$
$$Supden.ev = \{'poff','pon','verr','voff','von'\}$$
$$Supden.tf = \{'0' \quad '0' \quad '0' \quad '0' \quad '0'\}$$
$$Supden.ini = 'c0'$$
$$Supden.mrk = \{'c0'\}$$

$$Infden.st = \{'c0','c1','c2','c3','\_S1','\_S2','\_S3'\}$$

$$Infden.ev = \{'von','voff ',' pon',' poff ','verr '\}$$

$$Infden.tf = \begin{Bmatrix} 'c1' & '0' & '0' & '0' & '0' \\ '0' & '0' & 'c2' & '0' & '\_S1' \\ '0' & '0' & '0' & 'c3' & '\_S2' \\ '0' & 'c0' & '0' & '0' & '\_S3' \\ '0' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' \\ '0' & '0' & '0' & '0' & '0' \end{Bmatrix}$$

$$Infden.ini = 'c0'$$

$$Infden.mrk = \{'c0'\}$$

$$blkstates = \{'\_S1' \ , \ '\_S2' \ , \ '\_S3'\}$$

$$mrksupden = \{\ \}$$

$$mrkinfden = \{'vonponpoffvoff '\}$$

4.3.4 Kilitlenebilir Denetimsel Gözetleyici Problemini Çözmek için Geliştirilmiş Olan Fonksiyonlar

Tanımlanmış olan kilitlenebilir denetimsel gözetleyici problemini çözmek için ayrık olay sistemleri için tanımlanmış standart fonksiyonların yanı sıra kayıp faktörü, performans ölçüsü gibi fonksiyonlar da tanımlanmıştır. Bu bölümde bu fonksiyonlar anlatılacak ve fonksiyonlara ait algoritmalar verilecek ve örnekler üzerinden çalışması açıklanacaktır.

4.3.4.1 Eder Fonksiyonu

Bir G deterministik sonlu durumlu otomatının ürettiği her bir kelimenin ederini kelime eder tanımına göre hesaplayıp bunu bir liste olarak oluşturan fonksiyondur. Bunun için Matlab komut satırına $costfunction(G)$ yazılır. Komutun koşturulması ile birlikte her bir kelimeden sonra oluşan olayların ederinin kullanıcı tarafından girilmesi istenir. Bu olay ederlerine göre yeni oluşan her bir kelimenin ederi fonksiyon tarafından elde edilir. Fonksiyon döngüsel olmayan otomatlar için geliştirilmiştir. Fonksiyona ait algoritma Şekil 4.20'deki gibidir.

```
function [strlist, costlist] := costfuntion(G)
TNM := '0'
templist := {{x0, ' ', 0, 0}}
while templist ≠ ∅
    x := templist{1,1}
    s := templist{1,2}
    cost := templist{1,3}
    length := templist{1,4}
    strlist := strlist ∪ {s}
    costlist := costlist ∪ {cost}
    for each e ∈ Σ
        y := f(x, e)
        if y ≠ TNM
            newstring := strcon(s, e)
            eventcost := input('Enter the event cost')
            strcost := ((cost × length) + eventcost) / (length + 1)
            templist := templist ∪ {y, newstring, strcost, length + 1}
        end
    end
end
```

Şekil 4.20 : Eder fonksiyonuna ilişkin algoritma

4.3.4.2 Performans Fonksiyonu

G deterministik sonlu durumlu otomatının S denetimsel gözetleyicisinin kontrolü altında çalışması halinde performans ölçüsünü değerini tespit etmek için performans fonksiyonu tasarlanmıştır**.** Burada G sisteme ait otomatı ifade etmektedir G otomatının S denetimsel gözetleyicisi ile beraber çalışırkenki performans ölçüsünü hesaplayabilmek için Matlab komut satırına $pmeasure(G, S, G_a)$ komutu yazılır. G_a ise sisteme ait kabul edilebilir dili üreten otomattır.

$function\ pm = pmeasure(G, S, G_a)$
$[\Omega, \Psi] = costfunction(G)$
$L_{S,m} = markedlanguage(S)$
$L_S = generatedlanguage(S)$
$L_{am} = markedlanguage(G_a)$
$\overline{L_{S,m}} = prefixclosureoflanguage(L_{S,m})$
$bstrings = L_S \setminus \overline{L_{S,m}}$
$smcstrings = L_{am} \setminus L_{S,m}$
$for\ each\ s \in bstrings$
$pos = strfind(\Omega, s)$
$bm = bm + \Psi\{pos\}$
end
$for\ each\ s \in smcstrings$
$pos = strfind(\Omega, s)$
$smc = smc + \Psi\{pos\}$
end
$pm = bm + smc$

Şekil 4.21 : Pmeasure fonksiyonuna ait algoritma

4.3.4.3 T İşlemi Fonksiyonu

Bir G deterministik sonlu durumlu otomatının S denetimsel gözetleyicisi ile beraber çalışması halinde üretilen dil ile dile ait bir kilitlenmenin T işlemine tabii tutmak için T işlemi fonksiyonu geliştirilmiştir. Bunun için Matlab komut satırına $T\_operation(S, G, G_a, s)$ yazılır. Burada G ve S sırasıyla sisteme ve denetimsel gözetleyiciye ait otomatlardır. Aynı zamanda G_a kabul edilebilir dili üreten otomattır ve $s \in BM(L(S/G))$'dir. Fonksiyona ait algoritma Şekil 4.22'de verilmiştir.

$function\ G_{rst} := T\_operation = pmeasure(S, G, G_a, s)$
$//\ findst(G, s) : s\ kelimesinin\ üretilmesi\ ile\ ulaşılan\ durum$
$TNM := '0'$
$x^* := findst(S, s)$
$bstates := blockingstates(S)$

```
if x* ∉ bstates
    disp('s is not a blocking string')
    return
end
X1 := XS \ x*
X1,m := XS,m \ x*
Σ1 := ΣS
x1,0 := x0,S
for each x ∈ X1
    for each e ∈ Σ1
        if f(x,e) = x*
            f1(x,e) := TNM
        else
            f1(x,e) := f(x,e)
    end
end
G2 := supcont(G, G1)
G3 := infcont(G, automata_prd(G2, Ga))
pmG := pmeasure(G, S, Ga)
pmG3 := pmeasure(G, G3, Ga)
if pmG3 < pmG
    Grst := G3
else
    Grst := S
end
```

Şekil 4.22 : T işlemi fonksiyonuna ilişkin algoritma

4.3.4.4 Çıkartılan Kelimeler Fonksiyonu

Bir G deterministik sonlu durumlu otomatının S denetimsel gözetleyicisinin kontrolü altında iken üretilen dilden kilitlenmeye sebebiyet veren bir kelimenin T işlemi ile atılması halinde atılan tüm kelimeleri bulmak için çıkartılan kelimeler fonksiyonu geliştirilmiştir. Bunun için Matlab komut satırına $removedstrings(S, G, G_a, s)$ komutu yazılır. Bu fonksiyona ilişkin algoritma Şekil 4.23'de verilmiştir.

$function\ rs = removedstrings\left(S, G, G_a, s\right)$
$G_T := T\_transformation\left(G, G_a, s\right)$
$L_T := generatedlanguage\left(G_T\right)$
$L := generatedlanguage\left(G\right)$
$rs := L \setminus L_T$

Şekil 4.23 : Removedstrings fonksiyonuna ilişkin algoritma

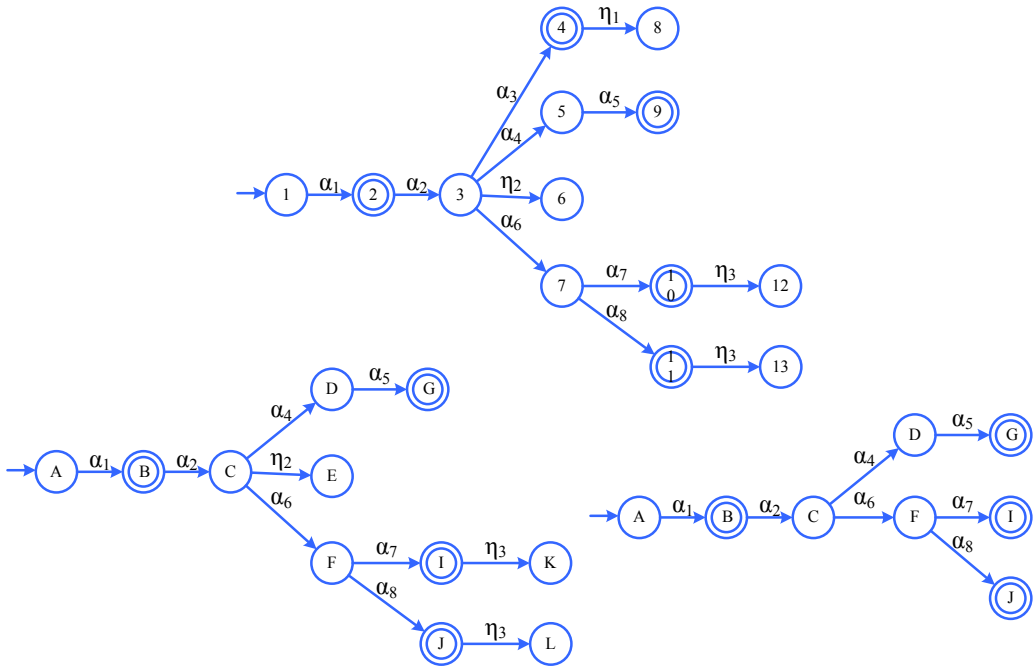
4.3.4.5 Kayıp Faktörü Fonksiyonu

Bir S denetimsel gözetleyicisinin kontrolü altında çalışan G otomatına ait kayıp faktörü kayıp faktörü fonksiyonu ile hesaplanır. Kayıp faktörünü hesaplayabilmek için Matlab komut satırına $lossfactor\left(S, G, G_a\right)$ komutu yazılır. Burada G, G_a ve S sırasıyla sisteme ait otomat, kabul edilebilir dili üreten otomat ve denetimsel gözetleyiciye ait olan otomattır. Lossfactor fonksiyonuna ait olan algoritma Şekil 4.24'te verilmiştir.

$function\ lossf = lossfactor\left(S, G, G_a\right)$
$\left[\Omega, \Psi\right] := costfunction\left(G\right)$
$L_{S,m} := markedlanguage\left(S\right)$
$L_S := generatedlanguage\left(S\right)$
$L_{am} := markedlanguage\left(G_a\right)$
$\overline{L_{S,m}} := prefixclosureoflanguage\left(L_{S,m}\right)$
$bstrings := L_S \setminus \overline{L_{S,m}}$
$smstrings := L_{am} \cap L_{S,m}$
$for\ each\ s \in bstrings$
    $pos := strfind\left(\Omega, s\right)$
    $bm := bm + \Psi\left\{pos\right\}$
end
$for\ each\ s \in smcstrings$
    $pos := strfind\left(\Omega, s\right)$
    $sm := smc + \Psi\left\{pos\right\}$
end
$lossf = bm - sm$

Şekil 4.24 : Kayıp faktörü fonksiyonuna ait algoritma

4.3.4.6 Örnek



Şekil 4.25 : Sırasıyla G, S ve G_a otomatları

Şekil 4.25'deki G otomatı ve S denetimsel gözetleyici için

$$[costlist, strlist] = costfunction(G)$$
$$lf_S = lossfactor(S, G, G_a)$$

işlemleri yapılırsa aşağıdaki sonuçlar elde edilir.

$Cs(,\alpha_1):3$
$Cs(\alpha_1,\alpha_2):2$
$Cs(\alpha_1\alpha_2,\alpha_3):15$
$Cs(\alpha_1\alpha_2,\alpha_4):7$
$Cs(\alpha_1\alpha_2,\alpha_6):3$
$Cs(\alpha_1\alpha_2,\eta_2):7$

$Cs(\alpha_1\alpha_2\alpha_3,\eta_1):3$
$Cs(\alpha_1\alpha_2\alpha_4,\alpha_5):5$
$Cs(\alpha_1\alpha_2\alpha_6,\alpha_7):5$
$Cs(\alpha_1\alpha_2\alpha_6,\alpha_8):8$
$Cs(\alpha_1\alpha_2\alpha_6\alpha_7,\eta_3):20$
$Cs(\alpha_1\alpha_2\alpha_6\alpha_8,\eta_3):4$

$costlist = \{0, 3, 2.5, 6.66, 4, 2.66, 4, 5.75, 4.25, 3.25, 4, 6.6, 4\}$

$strlist = \{\varepsilon, \alpha_1, \alpha_1\alpha_2, \alpha_1\alpha_2\alpha_3, \alpha_1\alpha_2\alpha_4, \alpha_1\alpha_2\alpha_6, \alpha_1\alpha_2\eta_2, \alpha_1\alpha_2\alpha_3\eta_1, \alpha_1\alpha_2\alpha_4\alpha_5, \alpha_1\alpha_2\alpha_6\alpha_7$
$\alpha_1\alpha_2\alpha_6\alpha_8, \alpha_1\alpha_2\alpha_6\alpha_7\eta_3, \alpha_1\alpha_2\alpha_6\alpha_8\eta_3\}$

$lf_S = -1.14$

5. SONUÇLAR VE TARTIŞMA

Ayrık olay sistemleri için kontrol edilebilirlik gibi bir diğer önemli ölçütte kilitlenme koşuludur. Genelde sisteme denetimsel gözetleyici ile birlikte kazandırılmak istenilen davranışın kontrol edilebilir ve kilitlenmesiz olması istenir. Fakat bu her zaman mümkün değildir. İstenilen davranış kontrol edilebilir ya da kilitlenmesiz değil ise çözüm olarak genellikle EAKKÇ tercih edilir. Eğer sistemin yapısında çok fazla kilitlenme var ise bu çözüm çok tutucu bir davranışa neden olacaktır. Bu noktada tutucu davranıştan uzaklaşmak ve istenilen işaretli davranışı üretebilmek için TBÇ tercih edilir. TBÇ sisteme tüm istenilen davranışı kazandırırken çok sayıda kilitlenmeye neden olan kelimeleri de üretilen dile ekliyebilir. Bu noktada bu iki çözüm de kabul edilebilir bir çözüm değildir. Öyle ki EAKKÇ sisteme çok fazla kısıt getirirken, TBÇ ise beklenen davranışı kazandırmak için çok sayıda kilitlenmeye neden olan kelimeyi üretilen dile eklemektedir. O zaman bu iki çözüm arasında istenilen koşulları sağlayan uygun bir kilitlenebilir denetimsel gözetleyiciyi tercih edilmelidir. Problem bu şekliyle kilitlenebilir denetimsel gözetleyici problemi olarak adlandırılmaktadır. Denetimsel gözetleyici ile sisteme kazandırılan davranışta kilitlenmeye izin vermekle, problem sistem için bir çözüm kümesinden uygun denetimsel gözetleyiciyi seçmeye karşılık düşer.

Kilitlenebilir denetimsel gözetleyici probleminde kilitlenmeler ve başarılamayan kabul edilebilir işaretli kelimeler arasında bir değiş tokuş vardır. Öyle ki sistemin sahip olduğu kilitlenmeler azaldıkça başarılamayan işaretli kelimelerin arttığı ve başarılamayan kelimeler azaldıkça kilitlenmelerin arttığı gözlemlenir. Bu çalışmada bu probleme farklı bir bakış açısı altından çözüm bulunmaya çalışılmıştır. Bu amaçla birbirinden farklı başarılamayan işaretli kelimeler ve kilitlenmeler içeren dilleri karşılaştırabilmek için yeni bir mesafe fonksiyonu tanımlanmış ve 2^{Σ^*} uzayı ile bu mesafe fonksiyonunun bir metrik uzay oluşturduğu gösterilmiştir. Aynı zamanda problemin doğasında bulunan bu değiş tokuşu en iyi şekilde yansıtan yeni bir performans ölçüsü ortaya atılmış, herhangi bir denetimsel gözetleyicinin performansı pozitif reel bir sayıya karşılık düşürülmüştür. Bu performans ölçüsüne göre de optimal kilitlenebilir denetimsel gözetleyici problemi tanımlanmıştır. Bu problem için denetimsel gözetleyicinin sağlaması gereken birinci koşul en düşük performans ölçüsüne sahip olması, ikinci koşul ise istenilen işaretli kelimeleri olabildiğince çok

kapsamasıdır. Sonuç olarak optimal kilitlenebilir denetimsel gözetleyici problemi için en iyi çözüm, istenilen işaretli kelimelere en fazla izin veren en düşük performans ölçüsüne sahip çözümdür.

Bu çalışmada istenilen işaretli dile de en fazla izin veren ve en düşük performansa sahip kilitlenebilir denetimsel gözetleyiciyi bulmak amaçlanmıştır. Bunun için yeni bir algoritma önerilmiş ve algoritmanın belirtilen ölçütlerde çözüme ulaştığı ispatlanmıştır. Gerekli ispatlar ve algoritmanın detayları 3. bölümde tartışılmış ve sonuçlar aktarılmıştır. Ayrıca geliştirilen algoritma bir veri yönetim sistemi üzerinde uygulanmıştır. Bu çalışma kapsamında Matlab ortamında ayrık olay sistemleri için yaygın kullanılan işlemleri kapsayan bir program yazılmıştır. Bu program 4. bölümde geliştirilen program parçalarına ilişkin algoritmaları ve 3. bölümde kilitlenebilir denetimsel gözetleyiciler için geliştirilmiş olan algoritmaları içermektedir. Bu programın Ramadge ve Wonham yaklaşımına göre ayrık olay sistemleri için farklı akademik gruplar tarafından geliştirilen paket programlarına göre en önemli farkı açık koda sahip olmasıdır. Bu yapılan çalışmalar 2007 yılında iki farklı dergide yayınlanmıştır. Ayrıca bu çalışmalar IFAC World Congress, Mediterranean Conference on Control and Automation gibi değerli konferanslarda da sunulmuştur.

Bu çalışmada anlatılan optimal kilitlenebilir denetimsel gözetleyici problemi düzenli fakat sayılabilir sonlu sayıda kelime içeren diller için çözülmüştür. Sonsuz sayıda kelime üreten, yapısında periodik geçişler bulunan deterministik sonlu durumlu otomatlar bu çalışmanın kapsamı dışındadır. Özellikle periodik çalışan sistemlerin modellenmesinde tercih edilen bu tür otomatlar değerlendirilmemiştir. Problem sonsuz sayıda kelime üreten deterministik sonlu durumlu otomatları kapsayacak şekilde genişletilebilir.

3.2.1 numaralı bölümde olay ederi kavramı tanıtılmış, hangi aralıkta ve ne tür bir sayısal değer alabileceği ifade edilmiştir. Ancak bir olay ederlerinin nasıl elde edileceği konusu değerlendirilmemiştir. Anlamlı bir çözüm elde edebilmek için olay ederlerinin değerleri algoritma kadar önemlidir ve bu konuda da araştırma yapılmalıdır. Bu araştırmada ederlerin alması gereken değerlerin seçimine ilişkin çözüm yöntemleri geliştirilebilir.

Ayrık olay sistemlerinin modellenmesi ve kontrolünde karşılaşılan sorunlardan birisi de sistemin yapısı karmaşıklaşmaya başladıkça sisteme ait modelin çok fazla duruma sahip olmasıdır. Bu konu literatürde durum patlaması olarak adlandırılmaktadır. Bu problemi çözebilmek için modüler kontrol, dağıtılmış kontrol, hiyerarşik kontrol gibi farklı denetimsel gözetleyici yapıları önerilmiş ve belli başarılar elde edilmiştir [103–

110]. Bu yöntemler denetimsel gözetleyici kuramı neticesinde elde edilen sonuçları temel almaktadır, ancak sistemi tek bir denetimsel gözetleyiciyle kontrol etmeye çalışılmamakta, özellikleri daha önceden belirlenmiş birden fazla denetimsel gözetleyici yapısı ile kontrol edilmesi yoluna gidilmektedir. Tasarım ve uygulamadaki üstünlüklerinden dolayı modüler kontrol, günümüzde en yaygın kullanılan yaklaşımdır. Bu çalışmada optimal kilitlenebilir denetimsel gözetleyici probleminin çözümü araştırılırken tek bir denetimsel gözetleyici ile sistemin kontrol edileceği varsayılmış, durum patlaması oluşması halinde kullanılabilecek diğer kontrol yöntemleri göz önünde alınmamıştır. Diğer kontrol yöntemleri için optimal kilitlenebilir denetimsel gözetleyici probleminin çözümü ayrı bir araştırma konusu olarak değerlendirilebilir.

KAYNAKLAR

[1] **Ramadge, P.J. and Wonham, W.M.**, 1987. Supervisory Control of a Class of Discrete Event Systems, *SIAM Journal Control and Optimization*, **25**, 206-230.

[2] **Ramadge, P.J. and Wonham, W.M.**, 1989. The Control of Discrete Event Systems, *Proceedings of IEEE*, **77**, 81-88.

[3] **Chen, E. and Lafortune, S.**, 1989. Dealing with Blocking in Supervisory Control of Discrete Event Systems, *28th Conference on Decion and Control* Tempa, Florida, 13-15 December, 117-122.

[4] **Chen, E. and Lafortune, S.**, 1991. Dealing with Blocking in Supervisory Control of Discrete-Event Systems, *IEEE Transactions on Automatic Control*, **36**, 724-735.

[5] **Yang, X. and Zheng, Y.**, 1992. A New Graph-based Method Dealing with Blocking in Supervisory Control of Discrete Event Systems, *IEEE International Conference on Robotics and Automation*, Nice, France, 12-14 May, 920 - 924.

[6] **Ozveren, C.M., Willsky, A.S., and Antsaklis, P.J.**, 1991. Stability and Stabilizability of Discrete Event Dynamic Systems, *Journal of the ACM*, **38**, 729 - 751.

[7] **Kumar, R., Garg, V., and Marcus, S.I.**, 1993. Language Stability and Stabilizability of Discrete Event Dynamical Systems, *SIAM Journal on Control and Optimization*, **31**, 1294-1320.

[8] **Brave, Y. and Heymann, M.**, 1990. Stabilization of discrete-event processes, *International Journal of Control*, **51**, 1101-1117.

[9] **Passino, K.M. and Antsaklis, P.J.**, 1990. Optimal Stabilization Of Discrete Event Systems, *29th Conference on Decision and Control*, Honolulu, Hawaii, December 1990, 670-671.

[10] **Takai, S., Ushio, T., and Kodama, S.**, 1995. Stabilization and Blocking in State Feedback Control of Discrete Event Systems, *Discrete Event Dynamic Systems: Theory and Applications*, **5**, 33-57.

[11] **Cao, X.-R. and Ho, Y.-C.**, 1987. Sensitivity Analysis and Optimization of Throughput in a Production Line with Blocking, *IEEE Transactions on Automatic Control*, **32**, 959-967.

[12] **Chen, Y.-L. and Lafortune, S.**, 1996. Priority Assingment Algorithms for Resolving Blocking in Modular Control of Discrete Event Systems, *35th Conference on Decision and Control*, Kobe, Japan, December, 2743-2748.

[13] **Bourdon, S.E., Lawford, M., and Wonham, W.M.**, 2002. Robust Nonblocking Supervisory Control of Discrete-Event Systems, *American Control Conference*, Anchorage, Alaska, USA, 8-10 May, 730 - 735.

[14] **Bourdon, S.E., Lawford, M., and Wonham, W.M.**, 2005. Robust Nonblocking Supervisory Control of Discrete-Event Systems, *IEEE Transactions on Automatic Control*, **50**, 2015 - 2021.

[15] **Yalcin, A. and Boucher, T.**, 2000. Deadlock avoidance in Flexible Manufacturing Systems, *IEEE Robotics & Automation*, **16**.

[16] **Roszkowska, E.**, 2004. Supervisory Control for Deadlock Avoidance in Compound Processes, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems And Humans*, **34**, 52-64.

[17] **Park, S.-J. and Lim, J.-T.**, 2005. Non-blocking Supervision for Uncertain Discrete Event Systems with Internal Unobservable Transitions, *IEE Procedings of Control Theory Applications*, **152**, 165-170.

[18] **Lewis, F.L., Gürel, A., Bogdan, S., Doganalp, A., and Pastranavu, O.C.**, 1998. Analysis of Deadlock and Circular Waits Using a Matrix Model for Flexible Manufacturing Systems, *Automatica*, **34**, 1083-1100.

[19] **Lewis, F.L., Bogdan, S., Gurel, A., and Pastravanu, O.**, 1997. Analysis of Deadlocks and Circular Waits Using a Matrix Model for Discrete Event Systems, *36th Conference on Decision & Control*, San Diego, California, USA, December, 4080-4085.

[20] **Lawley, M.A.**, 1999. Deadlock Avoidance for Production Systems with Flexible Routing, *IEEE Transactions on Robotics and Automation*, **15**, 497 - 509.

[21] **Ezpeleta, J. and Recalde, L.**, 2004. A Deadlock Avoidance Approach for Nonsequential Resource Allocation Systems, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems And Humans*, **34**, 93-101.

[22] **Technical Report of the ISIS Group at the University of Notre Dame,** Method for Deadlock Prevention in Discrete Event Systems Using Petri Nets, 1999.

[23] **Fabian, M. and Kumar, R.**, 2000. Mutually nonblocking supervisory control of discrete event systems, *Automatica*, **36**, 1863-1869.

[24] **Fanti, M.P. and Zhou, M.**, 2004. Deadlock Control Methods in Automated Manufacturing Systems, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems And Humans*, **34**, 5-22.

[25] **Huang, Y.S., Jeng, M.D., Xie, X., and Chung, S.L.**, 2001. A Deadlock Prevention Policy for Flexible Manufacturing Systems Using Siphons, *IEEE International Conference on Robotics & Automation*, Seoul, Korea, 21-26 May 541-546.

[26] **Abdelwahed, S. and Wonham, W.M.**, 2003. Blocking Detection in Discrete Event Systems, *American Control Conference*, Denver, Colorado, USA, 4-6 June, 1673-1678.

[27] **Gaudin, B. and Marchand, H.**, 2005. Supervisory Control and Deadlock Avoidance Control Problem for Concurrent Discrete Event Systems, *44th IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, 12-15 December.

[28] **Kumar, R., Takai, S., Fabian, M., and Ushio, T.**, 2005. Maximally Permissive Mutually and Globally Nonblocking Supervision with Application to Switching Control, *Automatica*, **41**, 1299-1312.

[29] **Saboori, A. and Zad, S.H.**, 2005. Robust nonblocking supervisory control of discrete-event systems under partial observation, *Congress on*

Computational Intelligence Methods and Applications (ICSC), 15-17 December.

[30] **Kumar, R. and Garg, V.K.**, 1995. Optimal Supervisory Control of Discrete Event Dynamical Systems, *SIAM Journal of Control and Optimization*, **33**, 419-439.

[31] **Passino, K. and Antsaklis, P.J.**, 1989. On Optimal Control of Discrete Event Systems, *IEEE 28th. Decision and Control Conference*, Tampa, Florida, 2713-2718.

[32] **Brave, Y. and Heymann, M.**, 1993. On Optimal Attraction in Discrete-Event Processes, *Information Sciences*, **67**, 245 - 267.

[33] **Sengupta, R. and Lafortune, S.**, 1998. An Optimal Control Theory for Discrete Event Systems, *SIAM Journal of Control and Optimization*, **26**, 448-541.

[34] **Sengupta, R. and Lafortune, S.**, 1998. Deterministic Optimal Control Theory for Discrete Event Systems, *32th. Conference on Decision and Control*, San Antonio, Texas, USA, 1182-1187.

[35] **Marchand, H., Boivineau, O., and Lafortune, S.**, 2000. On The Synthesis of Optimal Schedulers in Discrete Event Control Problems with Multiple Goals, *SIAM Journal of Optimization*, **2**, 512-532.

[36] **Marchand, H., Boivineau, O., and Lafortune, S.**, 2002. On Optimal Control of a Class of partially observed Discrete Event Systems, *Automatica*, **38**, 1935-1943.

[37] **Ray, A. and Phoha, S.**, 2003. Signed Real Measure of Regular Languages for Discrete-Event Automata, *International Journal of Control*, **6**, 1800-1808.

[38] **Ray, A., Surana, A., and Phoha, S.**, 2003. A Language Measure for Supervisory Control, *Applied Mathematics Letters*, **16**, 985-991.

[39] **Ray, A., Fu, J., and Lagoa, C.M.**, 2004. Optimal Supervisory Control of Finite State Automata, *International Journal of Control*, **77**, 1083-1100.

[40] **Fu, J., Ray, A., and Lagoa, C.M.**, 2003. Optimal Control of Regular Languages with Event Disabling Cost, *American Control Conference (ACC'03)*, Denver, Colorado, USA 4-6 June.

[41] **Fu, J., Ray, A., and Lagoa, C.M.**, 2004. Unconstrained optimal control of regular languages, *Automatica*, **40**, 639 – 646.

[42] **Lagoa, C.M., Fu, J., and Ray, A.**, 2005. Robust Optimal Control of Regular Languages, *Automatica*, **41**, 1439-1445.

[43] **Chattopadhyay, I. and Ray, A.**, 2007. Generalized Language Measure Families of Probabilistic Finite State Systems, *International Journal of Control*, **80**, 789-799.

[44] **Mohanty, S.R., Chandra, V., and Kumar, R.**, 1999. A Framework for Optimal Control of Discrete Event Systems, *International Conference on Robotics & Automation*.

[45] **Mohanty, S.R., Chandra, V., and Kumar, R.**, 1999. A Computer Implementable Algorithm for the Synthesis of an Optimal Controller for Acyclic Discrete Event Processes, *IEEE International Conference on Robotics and Automation*, Detroit, MI, USA, 10-15 May, 126 - 130.

[46] **Huang, J. and Kumar, R.**, 2005. Nonblocking Directed Control of Discrete Event Systems, *44th IEEE Decision and Control and European Control Conference (CDC-ECC '05)* Spain, 12-15 December, 7627 - 7632.

[47] **Huang, J. and Kumar, R.**, 2006. Directed Control of Discrete Event Systems: Optimization based Approach, *American Control Conference*, Minneapolis, Minnesota, USA, 14-16 June.

[48] **Chung, S.-L., Lafortune, S., and Lin, F.**, 1992. Limited Lookahead Policies in Supervisory Control of Discrete Event Systems, *IEEE Transactions on Automatic Control*, **37**, 1921 - 1935.

[49] **Chung, S.-L., Lafortune, S., and Lin, F.**, 1994. Supervisory Control with Variable Lookahead Policies, *Discrete Event Dynamic Systems - Theory and Applications*, **4**, 237-268.

[50] **Chung, S.-L., Lafortune, S., and Lin, F.**, 1992. Recursive Computation of Limited Lookahead Supervisory Controls for Discrete Event Systems, *31st IEEE Conference on Decision and Control*, Tucson, USA, 16-18 December, 3764-3769.

[51] **Chung, S.-L., Lafortune, S., and Lin, F.**, 1993. Recursive Computation of Limited Lookahead Controls for Discrete Event Systems, *Discrete Event Dynamic Systems: Theory and Applications*, **3**, 71-100.

[52] **Hadj-Alouane, N.B., Lafortune, S., and Lin, F.**, 1993. Control of Partially Observed Discrete Event Systems with Maximal Variable Lookahead Policies, *Thirty-First Allerton Conference on Communications, Control and Computing*, 898-907.

[53] **Hadj-Alouane, N.B., Lafortune, S., and Lin, F.**, 1994. A Distributed On-line Algorithm for Supervisory Control under Partial Observation, *Conference on Information Sciences and Systems*, Princeton, NJ, USA, March.

[54] **Hadj-Alouane, N.B., Lafortune, S., and Lin, F.**, 1994. Variable Lookahead Supervisory Control with State Information, *IEEE Transactions on Automatic Control*, **39**, 2398-2410.

[55] **Hadj-Alouane, N.B., Lafortune, S., and Lin, F.**, 1996. Centralized and Distributed Algorithms for On-Line Synthesis of Maximal Control Policies under Partial Observation, *Discrete Event Dynamic Systems - Theory and Applications*, **6**, 379-427.

[56] **Takai, S.**, 1997. Estimate Based Limited Lookahead Supervisory Control for Closed Language Specifications, *Automatica*, **33**, 1739-1743.

[57] **Kumar, R., Cheung, H.M., and Marcus, S.I.**, 1998. Extension based Limited Lookahead Supervision of Discrete Event Systems, *Automatica*, **34**, 1327-1344.

[58] **Gudwin, R. and Gomide, F.**, 1994. Genetic Algorithms and Discrete Event Systems: An Application, *IEEE World Congress on Computational Intelligence*, Orlando, FL, USA, 27-29 June, 742 - 745.

[59] **Grigorov, L. and Rudie, K.**, 2004. Applicability of Optimizing Control for Discrete Event Systems, *43rd IEEE Conference on Decision and*

Control (CDC'04), Atlantis, Paradise Island, Bahamas, 14-17 December.

[60] **Grigorov, L. and Rudie, K.**, 2005. Issues in Optimal Control of Dynamic Discrete Event Systems, *16th IFAC World Congress*, Prague, Czech Republic, 4-8 July.

[61] **Grigorov, L. and Rudie, K.**, 2006. Near-optimal online control of dynamic discrete-event systems, *Discrete Event Dynamic Systems: Theory and Applications*, **16**, 419-449.

[62] **Lin, F. and Ying, H.**, 2002. Modeling and Control of Fuzzy Discrete Event Systems, *IEEE Transactions on Man, Systems and Cybernetics - Part B*, **32**, 408-415.

[63] **Lin, F., Ying, H., Luan, X., MacArthur, R.D., Cohn, J.A., Barth-Jones, D., and Crane, L.R.**, 2005. Theory for a Control Architecture of Fuzzy Discrete Event Systems for Decision Making, *IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, 12-15 December, 2769-2774.

[64] **Lin, F., Ying, H., Luan, X., MacArthur, R.D., Cohn, J.A., Barth-Jones, D.C., and Crane, L.R.**, *Control of Fuzzy Discrete Event Systems and Its Applications to Clinical Treatment Planning*, in *43rd IEEE Conference on Decision and Control*. 2004. p. 519-524.

[65] **Lin, F., Ying, H., MacArthur, R.D., Cohn, J.A., Barth-Jones, D., and Crane, L.R.**, 2007. Decision Making in Fuzzy Discrete Event Systems, *Information Sciences*, **Accepted**.

[66] **Yang, J.-M. and Kim, J.-H.**, 1996. Optimization of Discrete Event Systems using Evolutionary Programming, *IEEE International Conference on Evolutionary Computation*, Nagoya, Japan, 20-22 May, 131-134.

[67] **Yang, J.-M. and Kim, J.-H.**, 1996. Control and Optimization of Cost-Evaluated Discrete Event Systems, *35th IEEE Decision and Control*, Kobe, Japan, 11-13 December, 1215-1216.

[68] **Hopcroft, J.E. and Ullman, J.D.**, 1979. Introduction to Automata Theory, Languages and Computation, Addison Wesley Publishing Company, Massachusetts, California.

[69] **Savage, J.E.**, 1998. Models of Computation, Addison Wesley Publishing Company, Massachusetts, California.

[70] **Cassandras, C. and Lafortune, S.**, 1999. Introduction to Discrete Event Systems, Kluwer Academic Publishers, Massachusetts, USA.

[71] **Murata, T.**, 1989. Properties, Analysis and Applications, *Proceedings of IEEE*, **77**, 541-580.

[72] **Holloway, L.E., Krogh, B.H., and Guina, A.**, 1997. A Survey of Petri Net Methods for Controlled Discrete Event Systems, *Journal of Discrete Event Systems: Theory and Applications*, **7**, 151-190.

[73] **David, R. and Alla, H.**, 1994. Petri nets for modeling of dynamic systems—a survey, *Automatica*, **30**, 175-202.

[74] **Alur, R. and Dill, D.L.**, 1994. A Theory of Timed Automata, *Theoretical Computer Science*, **126**, 183-235.

[75] **Brandin, B. and Wonham, W.M.**, 1994. Supervisory Control of Timed Discrete Event Systems, *IEEE Transactions on Automatic Control*, **39**, 329-342.

[76] **Ostroff, J.S.**, 1989. Temporal Logic for Real-Time Systems, Research Studies Press and John Wiley & Sons, New York, USA.

[77] **Baccelli, F., Cohen, G., Olsder, G.J., and Quadrat, J.P.**, 1992. Synchronization and Linearity, Wiley.

[78] **Martin, J.C.**, 1991. Introduction to Languages and the Theory of Computation, McGraw-Hill Inc., Singapore.

[79] **Kumar, R. and Garg, V.**, 1995. Modeling and Control of Logical Discrete Event Systems, Kluwer Academic Publishers.

[80] **Caillaud, B., Darondeau, P., Lavagno, L., and Xie, X.**, 2002. Synthesis and Control of Discrete Event Systems, Kluwer Academic Publishers, Dordrecht.

[81] **Kaymakci, O. and Kurtulan, S.**, 2004. A Metric Space Approach for a Class of Discrete Event Systems, *12th Mediterranean Conference on Control and Automation*, Kusadasi, Aydin, Turkey, 6-9 June.

[82] **Kaymakci, O., Kurtulan, S., and Goren, L.**, 2005. Improving the behaviour of supervisor under blocking, *16th IFAC World Congress*, Prague, Czech Republic, 4-8 July.

[83] **Thistle, J.G.**, 1996. Supervisory Control of Discrete Event Systems, *Mathematical and Computer Modeling*, **25**, 25-53.

[84] **Brandt, R.D., Garg, V., Kumar, R., Lin, F., Marcus, S.I., and Wonham, W.M.**, 1990. Formulas for Calculating Supremal Controllable and Normal Sublanguages, *Systems & Control Letters*, **15**, 111-117.

[85] **Wonham, W.M. and Ramadge, P.J.**, 1987. On the Supremal Controllable Sublanguage of a Given Language, *SIAM Journal of Control and Optimization*, **25**, 637-659.

[86] **Lafortune, S. and Chen, E.**, 1990. The Infimal Closed Controllable Superlanguage and Its Application in Supervisory Control, *IEEE Transactions on Automatic Control*, **35**.

[87] **Rudie, K. and Wonham, W.M.**, 1990. The Infimal Prefix-Closed and Observable Superlanguage of a Given Language, *Systems & Control Letters*, **15**, 361-371.

[88] **Osborne, M.J. and Rubenstein, A.**, 1994. A Course in Game Theory, MIT Press.

[89] **Kaymakci, O. and Kurtulan, S.**, 2006. Selecting an Optimal Blocking Supervisor for a Class of Discrete Event Systems, *National Conference On Automatic Control (TOK'07)*, Ankara, Turkey, 06-08 November.

[90] **Kaymakci, O. and Kurtulan, S.**, 2007. A Performance Evaluation Algorithm for Discrete Event Systems under Blocking, *The 15th Mediterranean Conference On Control And Automation (MED'07)*, Athens, Greece, 27-29 June.

[91] **Kaymakci, O. and Kurtulan, S.**, 2007. A Novel Performance Evaluation Method for Discrete Event Systems, *Journal Of Information Science And Engineering*, **Kabul Edildi**.

[92] **Kaymakci, O. and Kurtulan, S.**, 2007. A Performance Evaluation Algorithm for Discrete Event Systems Under Blocking, *WSEAS Transactions on Systems*, **Kabul Edildi**.

[93] **Wang, X., Ray, A., and Khatkhate, A.**, 2005. On-line Identification of Language Measure Parameters for Discrete Event Supervisory Control, *Applied Mathematical Modelling*, **29**, 597-613.

[94] **Kreyszig, E.**, 1989. Introductory Functional Analysis with Applications, John Wiley & Sons.

[95] **Discrete Event Systems Group At University of Michigan Under the Coordination of Stephane Lafortune and Demosthenis Teneletzis,** UMDES-LIB, Library of Commands for Discrete Event Systems Modeled By Finite State Machines, *International Workshop on Discrete Event Systems,* 2000, Ghent, Belgium.

[96] **Ricker, L., Lafortune, S., and Genc, S.**, 2006. DESUMA: A Tool Integrating GIDDES and UMDES, *8th International Workshop on Discrete-Event Systems*, Ann Arbor, MI, USA, 10-12 July.

[97] **Wonham, W.M.**, *Supervisory control of discrete event systems and XPTCT software*, Department of Electrical and Computer Engineering, University of Toronto, http://www.control.utoronto.ca/people/profs/wonham/wonham.html.

[98] **Åkesson, K., Fabian, M., Flordal, H., and Malik, R.**, 2006. Supremica – An integrated environment for verification, synthesis and simulation of discrete event systems, *Workshop of Discrete Event Systems (WODES)*, Ann Arbor, Michigan, USA, 10-12 July.

[99] **Åkesson, K., Fabian, M., Flordal, H., and Vahidi, A.**, 2003. Supremica – A Tool for Verification and Synthesis of Discrete Event Supervisors, *11th Mediterranean Conference on Control and Automation*, Rhodos, Greece, 18-20 June.

[100] **Pettersson, P. and Larsen, K.G.**, 2000. Uppaal2k, *Bulletin of the European Association for Theoretical Computer Science*, **70**, 40-44.

[101] **Behrmann, G., David, A., Larsen, K.G., Mller, O., Pettersson, P., and Yi, W.**, 2001. Uppaal - present and future, *40th IEEE Conference on Decision and Control*, Orlando, Florida, USA, 4-7 December.

[102] **Behrmann, G., David, A., and Larsen, K.G.**, 2004. A Tutorial on Uppaal, *4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, Bertinoro, Italy, 13-18 September.

[103] **Queiroz, M.H.d. and Cury, J.E.R.**, 2000. Modular control of composed systems, *American Control Conference (ACC)*, Chicago, USA, 28-30 June.

[104] **Queiroz, M.H.d. and Cury, J.E.R.**, 2000. Modular supervisory control of large scale discrete-event systems, *International Workshop on Discrete Event Systems (WODES)*, Ghent, Belgium, 21-23 August.

[105] **Queiroz, M.H.d. and Cury, J.E.R.**, 2002. Synthesis and implementation of local modular supervisory control for a manufacturing cell, *International Workshop on Discrete Event Systems (WODES)*, Zaragoza, Spain, 2-4 October.

[106] **Wonham, W.M. and Ramadge, P.J.**, 1998. Modular supervisory control of DESs, *Mathematics of control of discrete event systems*, **1**, 13-30.

[107] **Marchand, H. and Gaudin, B.**, 2002. Supervisory Control Problems of Hierarchical Finite State Machines *41th IEEE Conference on Decision and Control*, Las Vegas, Nevada, December 2002.

[108] **Cho, K.-H. and Lim, J.T.**, 1999. Mixed Centralized/Decentralized Supervisory Control of Discrete Event Systems *Automatica*, **35**, 121-128.

[109] **Wong, K.C. and Wonham, W.M.**, 1996. Hierarchical control of discrete-event systems *Discrete Event Dynamic Systems - Theory and Applications*, **6**, 241-273.

[110] **Wong, K.C., Murray, W., and Wonham, W.M.**, 1998. Modular Control and Coordination of Discrete-Event Systems, *Discrete Event Dynamic Systems - Theory and Applications*, **8**, 247 - 297.

ÖZGEÇMİŞ

Özgür Turay Kaymakçı 11 Eylül 1976'da İstanbul'da dünyaya geldi. Ortaokul ve lise eğitimini Üsküdar Hüseyin Avni Sözen Anadolu Lisesi'nde, lisans eğitimini İ.T.Ü. Elektrik - Elektronik Fakültesi Elektrik Mühendisliği Bölümünde tamamladı. 1998 yılında Elektrik Mühendisliğinden mühendislik diplomasını aldı ve İ.T.Ü. Fen Bilimleri Enstitüsü Kontrol ve Bilgisayar Mühendisliği programında yüksek lisans eğitimine başladı. Aynı yıl içerisinde de Kontrol ve Kumanda Ana Bilim Dalında araştırma görevlisi olarak çalışmaya başladı. 2000 yılında Doç. Dr. Salman Kurtulan'ın danışmanlığında yüksek lisans tezini tamamladı ve İ.T.Ü. Fen Bilimleri Enstitüsüne bağlı Kontrol ve Otomasyon Mühendisliği Programında doktoraya başladı. Hâla Kontrol ve Kumanda Ana Bilim Dalında araştırma görevlisi olarak çalışmaktadır.