İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

PETRİ AĞLARININ İNCELENMESİ
ve
ÖRNEK BİR KONTROL SİSTEMİNE UYGULANMASI

YÜKSEK LİSANS TEZİ

Kontrol ve Bilg. Müh. Cem BAŞKOCAGİL

Anabilim Dalı : Elektrik Mühendisliği

Programı : Kontrol ve Otomasyon Mühendisliği

MAYIS 2004

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

PETRİ AĞLARININ İNCELENMESİ ve ÖRNEK BİR KONTROL SİSTEMİNE UYGULANMASI

YÜKSEK LİSANS TEZİ

Kontrol ve Bilg. Müh. Cem BAŞKOCAGİL
(504011124)

Tezin Enstitüye Verildiği Tarih : 15 Nisan 2004

Tezin Savunulduğu Tarih : 18 Mayıs 2004

Tez Danışmanı : **Doç. Dr. Salman KURTULAN**

Diğer Jüri Üyeleri **Prof. Dr. Leyla GÖREN**

Doç. Dr. Mürvet ÜÇER

MAYIS 2004

ÖNSÖZ

Bu tez çalışması esnasında bana her zaman yol gösterici olan ve yardımlarını esirgemeyen tez danışmanım sayın Doç. Dr. Salman KURTULAN'a, bu tez çalışmasını TÜBİTAK – UEKAE bünyesinde gerçekleştirme imkânını sunan enstitü müdürüm Önder YETİŞ'e ve proje yürütücüm Mert ÜNERİ'ye, tez çalışmam boyunca bana manevi destekte bulunan babam İbrahim BAŞKOCAGİL'e ve annem Beyhan BAŞKOCAGİL'e teşekkürlerimi sunarım.

MAYIS 2004 Cem BAŞKOCAGİL

İÇİNDEKİLER

KISALTMALAR **iv**
TABLO LİSTESİ **v**
ŞEKİL LİSTESİ **vi**
SEMBOL LİSTESİ **viii**
ÖZET **ix**
SUMMARY **x**

1. GİRİŞ **1**
1.1 Giriş ve Çalışmanın Amacı 1

2. PETRİ AĞLARINA GENEL BAKIŞ **4**
2.1 Petri Ağları Tanımları 4
2.2 Petri Ağları Notasyonları 4
2.3 İşaretli Petri Ağları 7
2.4 Petri Ağları Dinamikleri 10
2.4.1 Erişilebilir Durumlar 14
2.4.2 Durum Denklemleri 15

3. PETRİ AĞLARI İLE MODELLEME **18**
3.1 Petri Ağları Yapıları 18
3.1.1 Paralellik 18
3.1.2 Senkronizasyon 19
3.1.3 Çakışma 19
3.1.4 Sıralama 20
3.1.5 Kaynak Sınırlama 21
3.2 Bozucu Yaylar 22
3.3 Petri Ağlarına Eklentiler 23
3.3.1 Modüler Petri Ağları 24
3.3.2 Renkli Petri Ağları 25
3.3.3 Zaman Etiketli Petri Ağları 26
3.4 Modelleme Örnekleri 27
3.4.1 Kuyruk Sisteminin Modellenmesi 27
3.4.2 Trafik Işıklarının Modellenmesi 28
3.4.3 Meşrubat Otomatının Modellenmesi 33

4. PETRİ AĞLARININ ANALİZİ **35**
4.1 Petri Ağlarının Davranış Özellikleri 35
4.1.1 Güvenlik Özelliği 35
4.1.2 Sınırlılık Özelliği 35
4.1.3 Canlılık Özelliği 35
4.1.4 Sakınım Özelliği 37
4.1.5 Kapsanabilirlik Özelliği 38
4.1.6 Süreklilik Özelliği 39
4.2 Petri Ağlarının Analiz Yöntemleri 39
4.2.1 Matematiksel Analiz Yöntemleri 39
4.2.2 Yapısal Analiz Yöntemleri 48
4.3 Petri Ağlarının Sınıflandırılması 50

5. PETRİ AĞLARI İLE SİSTEM KONTROLÜ **52**
5.1 Ağ Değişmezlerine Dayalı Kontrolör Tasarımı 53
5.2 Kontrol Edilemez ve Gözlenemez Geçişler 58

6. PETRİ AĞLARI İLE ÖRNEK BİR UYGULAMA **60**
6.1 Kontrol Edilecek Sistemin Tanıtımı 60
6.2 Kontrol Edilecek Sistemin Çalışma Mantığı 62
6.3 Sistemin Kontrol Altına Alınması 65

7. SONUÇ **71**

KAYNAKLAR **72**

EK A. KONTROLLÜ SİSTEME AİT PLC PROGRAMI **73**

ÖZGEÇMİŞ **80**

KISALTMALAR

PN	**:** Petri Net (Petri Ağı)
DES	**:** Discrete Event System (Ayrık Olaylı Sistem)
PLC	**:** Programmable Logic Controller (Programlanabilir Lojik Kontrolör)

TABLO LİSTESİ

Sayfa No

Tablo 3.1 **:** Modellenecek trafik ışıklarının yanma sırası 28
Tablo 4.1 **:** Koruyucu PN örneğindeki geçişlerin giriş ve çıkış yeri sayısı 38
Tablo 4.2 **:** Örnek PN grafındaki savak ve kapanlar 49
Tablo 6.1 **:** Sisteme ait kısımların işlevleri 64
Tablo 6.2 **:** Sistemin PN grafındaki yerlere ilişkin açıklamalar 64
Tablo 6.3 **:** Kontrolör yerlerine ilişkin bilgiler 69
Tablo 6.4 **:** Kontrolör yerlerine ilişkin açıklamalar 70

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1 : PN grafı bileşenleri 6
Şekil 2.2 : Örnek bir PN grafı 6
Şekil 2.3 : Aynı PN'ye ait iki farklı x_1 ve x_2 vektörü 8
Şekil 2.4 : Aktif ve pasif geçiş örneği 9
Şekil 2.5 : x_0 = [1, 0, 0, 2, 1] ilk durumlu örnek bir PN 11
Şekil 2.6 : x_1= [1, 0, 1, 3, 0] durumlu PN 12
Şekil 2.7 : x_2= [0, 1, 2, 5, 0] durumlu PN 13
Şekil 2.8 : x_3= [0, 1, 2, 3, 1] durumlu PN 14
Şekil 2.9 : Örnek bir PN grafı için erişilebilir durumlar 14
Şekil 3.1 : Paralellik örneği 19
Şekil 3.2 : Senkronizasyon örneği 19
Şekil 3.3 : Çakışma örneği 20
Şekil 3.4 : Sıralama örneği 20
Şekil 3.5 : Kaynak sınırlama örneği 21
Şekil 3.6 : Bozucu yay örneği 22
Şekil 3.7 : Çoklu bozucu yay örneği 22
Şekil 3.8 : Geçişlere öncelik atama 23
Şekil 3.9 : Modüler PN örneği 24
Şekil 3.10 : Renkli PN örneği 25
Şekil 3.11 : Zaman etiketli PN örneği 26
Şekil 3.12 : Örnek bir kuyruk sisteminin PN ile modellenmesi 27
Şekil 3.13 : Modellenecek trafik ışıklarının yerleşimi 28
Şekil 3.14 : Trafik ışıklarının PN ile modellenmesi 29
Şekil 3.15 : Trafik ışıklarının ys_A geçişinin tetiklenmesinden sonraki durumu 30
Şekil 3.16 : Trafik ışıklarının sk_A geçişinin tetiklenmesinden sonraki durumu 31
Şekil 3.17 : Trafik ışıklarının ky_B geçişinin tetiklenmesinden sonraki durumu 31
Şekil 3.18 : Trafik ışıklarının ys_B geçişinin tetiklenmesinden sonraki durumu 32
Şekil 3.19 : Trafik ışıklarının sk_B geçişinin tetiklenmesinden sonraki durumu 32
Şekil 3.20 : Trafik ışıklarının ky_A geçişinin tetiklenmesinden sonraki durumu 33
Şekil 3.21 : Örnek bir meşrubat otomatının PN ile modellenmesi 33
Şekil 4.1 : Canlılık özelliğine ilişkin örnek PN grafı 36
Şekil 4.2 : Koruyucu bir PN grafı örneği 38
Şekil 4.3 : Süreklilik özelliği olmayan örnek bir PN grafı 39
Şekil 4.4 : Kapsanabilirlik ağacı için örnek PN grafı 40
Şekil 4.5 : Örnek PN grafının kapsanabilirlik ağacı 40
Şekil 4.6 : Sonsuz kapsanabilirlik ağacı için örnek PN grafı 43
Şekil 4.7 : Örnek PN grafının sonsuz kapsanabilirlik ağacı 44
Şekil 4.8 : Değişmez yerler ve geçişler için örnek PN grafı 46
Şekil 4.9 : Örnek bir savak ve kapan 48

Şekil 4.10 : Savak ve kapan içeren örnek bir PN grafı 49
Şekil 4.11 : PN sınıfları arasındaki ilişkiler 51
Şekil 5.1 : Kontrol edilmek istenen sistemin PN grafı 56
Şekil 5.2 : Kontrol altına alınmış sistemin PN grafı 58
Şekil 6.1 : Montaj işleminde kullanılan parçalar 61
Şekil 6.2 : Montaj sonunda elde edilen dönel mekanizma 61
Şekil 6.3 : SMC MAP – 205 Montaj Sistemi 62
Şekil 6.4 : SMC MAP – 205 Montaj Sistemi'nin PN grafı 63
Şekil 6.5 : Kontrol altına alınmış montaj sisteminin PN grafı 70
Şekil A.1 : PLC ana programı (Networks 1 – 16) 73
Şekil A.2 : PLC ana programı (Networks 17 – 30) 74
Şekil A.3 : PLC ana programı (Networks 31 – 42) 75
Şekil A.4 : PLC ana programı (Networks 43 – 57) 76
Şekil A.5 : PLC ana programı (Networks 58 – 74) 77
Şekil A.6 : PLC ana programı (Networks 75 – 90) 78
Şekil A.7 : PLC alt programı 79

SEMBOL LİSTESİ

P	**:** Yerler kümesi
T	**:** Geçişler kümesi
A	**:** Yönlü yaylar kümesi
w	**:** Ağırlık fonksiyonu
I	**:** Giriş fonksiyonu
O	**:** Çıkış fonksiyonu
∅	**:** Boş küme
N	**:** Doğal sayılar kümesi
R	**:** Erişilebilir durumlar kümesi
f	**:** Durum geçiş fonksiyonu
u	**:** Tetikleme vektörü
x	**:** Durum vektörü
x′	**:** Bir sonraki durum vektörü
D	**:** Sistem matrisi
ΔT	**:** Gecikme süresi
γ	**:** Koruyuculuğa ilişkin ağırlık vektörü
i	**:** Değişmez yerler vektörü
j	**:** Değişmez geçişler vektörü
L	**:** Kısıtlamalara ilişkin ağırlık vektörü
b	**:** Kısıtlamalara ilişkin skaler değer
$\mathbf{x_p}$	**:** Kontrol edilen sistemin durum vektörü
$\mathbf{x_c}$	**:** Kontrolörün durum vektörü
$\mathbf{x_{p0}}$	**:** Kontrol edilen sistemin başlangıç durum vektörü
$\mathbf{x_{c0}}$	**:** Kontrolörün başlangıç durum vektörü
$\mathbf{x_0}$	**:** Kontrollü sistemin başlangıç durum vektörü
$\mathbf{D_p}$	**:** Kontrol edilen sistemin sistem matrisi
$\mathbf{D_c}$	**:** Kontrolörün sistem matrisi
$\mathbf{p_n}$	**:** Kontrol edilen sisteme ait yer
$\mathbf{c_n}$	**:** Kontrolöre ait yer

PETRİ AĞLARININ İNCELENMESİ
ve
ÖRNEK BİR KONTROL SİSTEMİNE UYGULANMASI

ÖZET

Gerçek hayattaki sistemlerin davranışlarını incelemek ve analiz etmek için bu sistemlerin modellenmesi artık bir ihtiyaç haline gelmiştir. Bu amacın karşılanması için sayısız yöntem ve modelleme araçları mevcuttur. Petri Ağları (Petri Nets) sistemlerin grafiksel olarak modellenmesini sağlamakta ve sunduğu matematiksel ve yapısal analiz teknikleriyle bir sistemin davranışlarının incelenmesini bir hayli kolaylaştırmaktadır. Petri ağları bir graf olarak ifade edilmekte ve bu graf modellenecek sistemin yapısına karşılık düşürülmektedir. Sistemin dinamik davranışı jeton denilen aktif nesnelerle ifade edilmektedir. Böylelikle modellenen sistemin hem statik yapısı hem de dinamik yapısı tek bir graf ile gösterilebilmektedir. Bunun yanı sıra matematiksel olarak ifade edilebilmesi bakımından Petri ağları güçlü analiz teknikleri sunmaktadır. Petri ağları ile modellenen bir sistemin kontrol altına alınması gerektiğinde yine Petri ağlarından yararlanılabilir. Petri ağları kontrolör tasarımı için basit ama güçlü teknikler sunmaktadır. Bu tez çalışmasında Petri ağları ile ilgili temel bilgiler verildikten sonra Petri ağlarının sistemlerin modellenmesinde, analiz edilmesinde ve kontrol altına alınmasına nasıl kullanılabileceği anlatılacak ve bu bilgiler kullanılarak örnek bir sistem Petri ağları ile kontrol altına alınacaktır.

INVESTIGATION OF PETRI NETS
and
ITS APPLICATION ON A SAMPLE CONTROL SYSTEM

SUMMARY

In real life, it is a necessity to model a system in order to analyze and inspect the behavior of that system. For this purpose, there are numerous methods and modeling tools. Among all of these tools, Petri Nets arises to be the best modeling tool. Petri Nets combines both graphical and mathematical aspects of a modeling tool in one. It provides a graphical representation of the system as well as mathematical analysis and inspection opportunities for that system. Graphical aspect of Petri Nets makes it easy to comprehend the system. Petri Nets actually consists of a graph and a token traveling through that graph. The graph structure is used to model the static aspects of the system whereas the token is used to model the dynamic behavior of the same system. Besides that the mathematical properties exposed by the Petri Nets are used to analyze the modeled system in detail. Petri Nets possesses powerful analysis techniques that make it possible to resolve a system failure. When it comes to the control of the modeled system, Petri Nets are very helpful as well. Petri Nets is a handy tool to be used in designing the desired controller for a system. The techniques used to control a system using Petri Nets do not require complex knowledge and too much theory so they are simple but powerful and easy to apply to a system. The objective of this study is to give a bit of know-how about Petri Nets consisting of what Petri Nets are, how Petri Nets can be used to model, analyze and control a system and finally to control a sample real – life system by making use of the aforementioned concepts of Petri Nets.

1. GİRİŞ

1.1 Giriş ve Çalışmanın Amacı

Petri Ağları (Petri Nets) gerçek hayattaki sistemlerin modellenmesinde kullanılan hem grafiksel hem de matematiksel bir araçtır. Petri ağları grafiksel yönüyle sistemlerin davranışlarının daha kolay incelenmesini, matematiksel yönüyle ise sistemlerin daha detaylı analiz edilmesini sağlamaktadır. Petri ağları birbiri ile etkileşim içerisinde olan süreçler barından bir sistemin modellenmesi için idealdir.

Petri ağları teorisi ilk defa 1962 yılında Carl Adam Petri tarafından ortaya atılmıştır. Bu tarihten itibaren Petri ağları ile ilgili birçok makale yayınlanmıştır. Giderek artan ilgi ile Petri ağları teoriden pratiğe geçirilmiş ve gerçek hayattaki birçok sistemin modellenmesinde kullanılmaya başlanmıştır.

Geniş uygulama alanları bulunan Petri ağları genellikle aşağıdaki gibi sistemlerin modellenmesinde kullanılır:

- Ayrık olaylı sistemler
- Üretim kontrol sistemleri
- Endüstriyel kontrol sistemleri
- Çok işlemcili bilgisayar sistemleri

Petri ağları ile bir sisteme ait aşağıdaki davranış şekilleri modellenebilir:

- Senkronizasyon (synchronization): Bir olay gerçekleştirilmeden önce bu olaya bağlı diğer olayların gerçekleşmesinin beklenmesi.
- Sıralama (sequentiality): Olayların gerçekleşme sıralarının belirlenmesi.
- Paralellik (concurrency): Olayların aynı anda gerçekleşmesinin sağlanması.
- Çakışma (conflict): Aynı koşula bağlı iki ya da daha fazla olaydan aynı anda yalnızca birinin gerçekleşmesine izin verilmesi.

Petri ağları sunduğu güçlü analiz teknikleriyle bir sistemin daha kolay incelenebilmesini sağlamaktadır. Bu analiz teknikleri kullanılarak bir sistemin davranışları detaylı olarak incelenebilir ve ortaya çıkan problemler rahat bir şekilde giderilebilir.

Zamanla klasik Petri ağlarında birtakım eksiklikler boy göstermiştir. Örneğin klasik Petri ağlarında zaman kavramı bulunmamaktadır. Ayrıca karmaşık yapıdaki Petri ağlarında gelişen olayların takip edilmesi zor bir işlem haline gelmiştir. Bu amaçla klasik Petri ağlarına bazı eklentiler yapılarak bu eksiklikler giderilmiş ve Petri ağlarının modelleme kabiliyeti arttırılmıştır. Modüler Petri ağları, Renkli Petri ağları ve Zaman Etiketli Petri ağları yapılan bu eklentilere örnektir. Modüler Petri ağları ile karmaşık yapıdaki modellerin incelenmesi kolaylaşmış, Renkli Petri ağları ile olayların takip edilmesi basit bir hale gelmiş ve Zaman Etiketli Petri ağları ile zamana dayalı sistemlerin modellenmesi sağlanmıştır.

Giderek yaygınlaşan Petri ağları sistemlerin kontrol edilmesi amacı ile de kullanılmaya başlanmıştır. Bu konuda sunduğu yöntemler ile Petri ağları kontrolör tasarımını kolay bir hale getirmiştir. Petri ağlarının kontrolör tasarımında kullanılmasının avantajları aşağıdaki gibi verilebilir:

1. Tasarım yöntemi yoğun teorik bilgi ve detaylı analiz gerektirmez.
2. Sonuçta ortaya çıkan kontrolör basit yapıdadır.
3. Tasarlanan kontrolörün hem matematiksel hem de grafiksel olarak test edilebilmesi için birçok araç mevcuttur.

Petri ağları ile kontrol altına alınmak istenen sistem süreç ağı, sonuçta kontrolör eklenerek kontrol altına alınan bütün sistem ise kontrollü sistem ya da kapalı çevrim sistemi olarak anılır. Petri ağları ile kontrolör tasarımı sistemin belirlenen kısıtlamalara uymasını sağlayarak gerçekleştirilir. Bu amaçla sistemin yasaklanmış durumlara erişmesi engellenir.

Bazı durumlarda kontrol altına alınmak istenen sistem kontrol edilemez ve gözlenemez olaylar içerebilir. Bu durumda aynı kontrol yöntemleri bazı değişikliklerle söz konusu sisteme uygulanabilir. Ancak bu tez çalışmasında kontrol edilemez ve gözlenemez olaylar içeren sistemler ve bu sistemler için kontrol tekniklerine değinilmeyecektir.

Bu tez çalışmasında öncelikle Petri ağları ile ilgili bilgi altyapısı oluşturulacak ve ardından bu bilgiler kullanılarak örnek bir sistem Petri ağları ile modellenip kontrol altına alınacaktır. Kontrol altına alınacak sistem olarak SMC MAP – 205 MONTAJ SİSTEMİ (ASSEMBLY CELL) seçilmiştir. 2. bölümde Petri ağları ile ilgili temel bilgiler verilecek, 3. bölümde Petri ağlarının sistem modellemesinde nasıl kullanılabileceği anlatılacak, 4. bölümde Petri ağlarının analizinde kullanılan tekniklerden bahsedilecek, 5. bölümde sistemlerin kontrol altına alınması için Petri ağlarında kullanılan yöntemler anlatılacak ve 6. bölümde örnek olarak seçilen sistemin kontrolör tasarım teknikleri kullanılarak Petri ağları ile nasıl kontrol altına alınabileceği gösterilecektir.

2. PETRİ AĞLARINA GENEL BAKIŞ

2.1 Petri Ağları Tanımları

Petri ağı (Petri Net – PN) iki bölgeli (bipartite) ve yönlü (directed) bir graf olarak tanımlanır. Bir PN (P, T, A, w) dörtlüsüyle ifade edilmektedir. Burada;

P, sonlu yerler (place) kümesini;

T, sonlu geçişler (transition) kümesini;

$\mathbf{A} \subseteq (P \times T) \cup (T \times P)$, yerlerden geçişlere ve geçişlerden yerlere giden yönlü yaylar kümesini;

$\mathbf{w} : A \rightarrow \{1, 2, 3, \ldots\}$ ise yaylara ilişkin ağırlık fonksiyonunu

belirtir[1].

PN'de geçişler olayları temsil eder. Bir geçişin gerçekleşebilmesi için bazı koşulların sağlanmış olması gerekir. PN grafındaki yerler bu koşullara ilişkin bilgileri tutar. Geçişlere ait hem giriş hem de çıkış noktaları mevcuttur. Bazı yerler bir geçişe ait girişler (input) olabileceği gibi bazı yerler de bir geçişe ait çıkışlar (output) olabilir. Giriş olan yerler ilgili geçişin gerçekleşebilmesi için gerekli koşul bilgilerini barındırır. Çıkış olan yerler ise ilgili geçiş gerçekleştikten sonra oluşan koşul bilgilerini saklar[1].

Geçişler, yerler ve bu ikisi arasındaki ilişkiler PN grafının temel bileşenlerini oluşturur. Bir PN grafı iki tip düğüme sahiptir, yerler ve geçişler. PN grafına ait bir diğer önemli bileşen ise yönlü yaylardır. Yayların görevi yerleri ve geçişleri birbirine bağlamaktır. PN grafı iki bölgeli bir graftır, bu nedenle yaylar aynı tip iki düğümü birbirine bağlayamaz. Başka bir deyişle yaylar ya yerleri geçişlere ya da geçişleri yerlere bağlar[1].

2.2 Petri Ağları Notasyonları

PN'de yerler kümesi $P = \{p_1, p_2, \ldots, p_n\}$ ve geçişler kümesi de $T = \{t_1, t_2, \ldots, t_m\}$ olarak gösterilir. Yönlü bir yay ise (p_i, t_j) veya (t_j, p_i) olarak ifade edilir. Yönlü

yaylara ilişkin ağırlık değerleri pozitif tamsayılarla gösterilir. Bu notasyonlarda dikkat edilecek olursa yerler ve geçişler kümesinin sonlu olduğu kabullenilmiştir; çünkü bu özellikte bir PN kullanılarak olayların modellenmesi ve incelenmesi daha kolay olacaktır[1].

Görüldüğü gibi bir PN bir otomattan daha fazla detay içeren bir yapıya sahiptir. Bir otomatta durum geçiş diyagramına ait düğümler tek bir kümeden seçilmektedir. Buna karşılık bir PN'de düğümler ya yerler kümesinden (P) ya da geçişler kümesinden (T) seçilebilir. Bunun yanı sıra bir otomata ilişkin durum geçiş diyagramında her bir durum geçişi için tek bir yay bulunmaktadır. Fakat bir PN'de aynı tip olmayan herhangi iki düğümü birleştiren birden fazla yay bulunabilir. Bu yayların sayısını ağırlık fonksiyonu belirler. Bu nedenlerden dolayı PN grafı bir çoklu graf olarak anılmaktadır[1].

Genelde bir PN tanımlanırken bir t_j geçişine ait olan giriş yerleri kümesi $I(t_j)$ ve benzer bir şekilde bir t_j geçişine ait olan çıkış yerleri kümesi de $O(t_j)$ olarak gösterilir. Böylelikle aşağıdaki notasyonlar elde edilir[1]:

$$I(t_j) = \{p_i \in P \ : \ (p_i, t_j) \in A\} \tag{2.1}$$

$$O(t_j) = \{p_i \in P \ : \ (t_j, p_i) \in A\} \tag{2.2}$$

Bir p_i yeri için de benzer notasyon şu şekilde kullanılabilir: $I(p_i)$, $O(p_i)$.

PN grafı çizilirken iki tip düğüm (yerler ve geçişler) arasında ayrım yapılmalıdır. Yaygın olarak yerleri temsil etmek için içi boş daireler ve geçişleri temsil etmek içinse dikey çizgiler kullanılır. Yerleri ve geçişleri birbirine bağlayan yönlü yaylar, yaylar kümesini (A) oluşturur. Bir p_i yerinden t_j geçişine çizilen yay, $p_i \in I(t_j)$ anlamına gelmektedir. Eğer $w(p_i, t_j) = k$ ise o zaman p_i yerinden t_j geçişine giden k tane yay var demektir. Başka bir deyişle p_i yerinden t_j yerine giden k ağırlıklı tek bir yay mevcuttur. Benzer bir şekilde eğer bir t_j geçişinden p_i yerine giden k tane yay varsa, $p_i \in O(t_j)$ demektir ve $w(t_j, p_i) = k$ değerini alır[1].

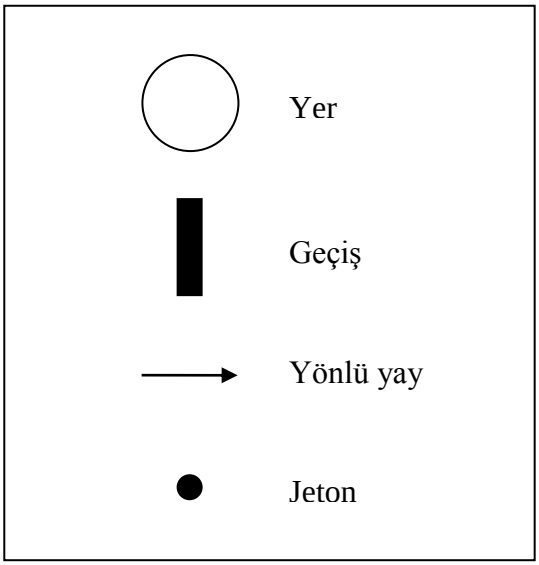
Ağırlıklar genelde ağırlığın değeri kadar yay çizilerek ifade edilir. Ancak eğer çok sayıda yayın çizilmesi gerekiyorsa, tek bir yayın çizilmesi ve yay sayısını belirten ağırlık değerinin bu yay üzerinde gösterilmesi daha etkili bir yoldur. Eğer bir yay

üzerinde ağırlık değeri gösterilmiyorsa bu ağırlık değeri 1 olarak kabul edilir. Son olarak ağırlık fonksiyonuna ilişkin aşağıdaki denklemler yazılabilir[1]:

$$p_i \notin I(t_j)\ ise\ w(p_i, t_j) = 0 \tag{2.3}$$

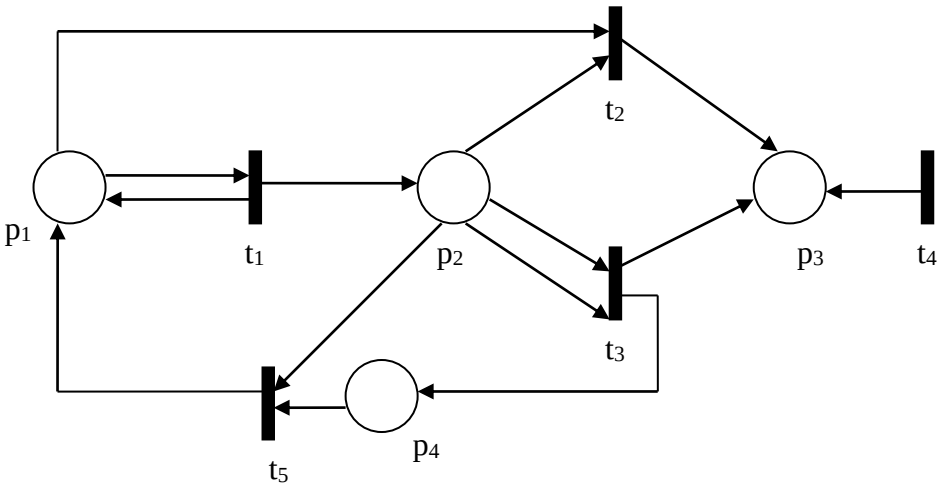
$$p_i \notin O(t_j)\ ise\ w(t_j, p_i) = 0 \tag{2.4}$$

Bir PN grafını oluşturan bileşenler Şekil 2.1'de gösterilmiştir:



Şekil 2.1 PN grafı bileşenleri

Örnek bir PN grafı Şekil 2.2'de verilmiştir[1]:



Şekil 2.2 Örnek bir PN grafı

Şekil 2.2'de gösterilen PN grafı için aşağıdakiler yazılabilir[1]:

$$P = \{p_1, p_2, p_3, p_4\}$$

$$T = \{t_1, t_2, t_3, t_4, t_5\}$$

$$A = \{(p_1, t_1), (p_1, t_2), (p_2, t_2), (p_2, t_3), (p_2, t_5), (p_4, t_5),$$
$$(t_1, p_1), (t_1, p_2), (t_2, p_3), (t_3, p_3), (t_3, p_4), (t_4, p_3), (t_5, p_1)\}$$

$w(p_1, t_1) = 1$ $w(p_1, t_2) = 1$ $w(p_2, t_2) = 1$ $w(p_2, t_3) = 2$

$w(p_2, t_5) = 1$ $w(p_4, t_5) = 1$ $w(t_1, p_1) = 1$ $w(t_1, p_2) = 1$

$w(t_2, p_3) = 1$ $w(t_3, p_3) = 1$ $w(t_3, p_4) = 1$ $w(t_4, p_3) = 1$

$w(t_5, p_1) = 1$

Burada şunu belirtmekte fayda vardır: Dikkat edilirse t_4 geçişi hiçbir girişe sahip değildir. Bu demektir ki t_4 geçişi hiçbir koşula bağlı olmadan gerçekleşebilir. Bunun aksine örneğin t_2 geçişine ilişkin olayın gerçekleşebilmesi için p_1 ve p_2 yerlerindeki mevcut koşulların sağlanmış olması gerekir[1]. Aşağıda Şekil 2.2'deki PN grafında bulunan geçişlere ilişkin giriş ve çıkış fonksiyonları gösterilmiştir:

$I(t_1) = \{p_1\}$ $O(t_1) = \{p_1, p_2\}$

$I(t_2) = \{p_1, p_2\}$ $O(t_2) = \{p_3\}$

$I(t_3) = \{p_2\}$ $O(t_3) = \{p_3, p_4\}$

$I(t_4) = \varnothing$ $O(t_4) = \{p_3\}$

$I(t_5) = \{p_2, p_4\}$ $O(t_5) = \{p_1\}$

2.3 İşaretli Petri Ağları

Bir PN grafındaki geçişlerin bir DES'e ilişkin olayları temsil ettiğini ve yerlerin bu olayların gerçekleşebilmesi için gerekli koşulları tanımladığını biliyoruz. Bunlardan yola çıkarak söz konusu koşulların sağlanıp sağlanmadığını belirlemek için bir mekanizmaya ihtiyaç duyarız. İşte bu mekanizma ise yerlere jetonların (token) konmasıyla gerçeklenir. Jetonların PN grafındaki bir yere konmasıyla bu yere ilişkin koşulların sağlanmış olduğu belirtilebilir. Koşulların sağlanma durumunun bu şekilde jetonlarla belirlendiği PN'lere İşaretli PN (Marked PN) denir[1].

İşaretli PN (P, T, A, w, x) beşlisiyle tanımlanır. Burada (P, T, A, w) dörtlüsü bir PN grafını ve x ise bu PN grafındaki yerlere atanmış işaretleri tanımlar[1]:

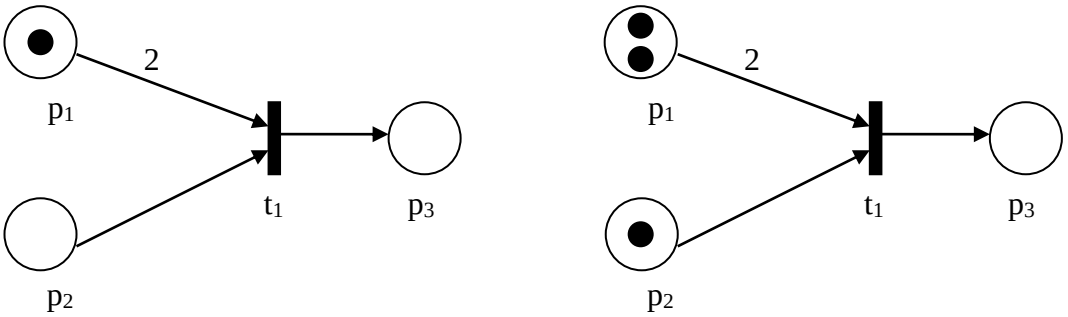
$$(P, T, A, w), \text{ PN grafı}$$

$$x : P \rightarrow N = \{0, 1, 2, \ldots\}$$

Aslında x bir satır vektörünü tanımlar:

$$x = [x(p_1), x(p_2), x(p_3), \ldots, x(p_n)] \in N^n$$

Bu satır vektöründe n, PN'deki yerlerin sayısını belirtir. Bu vektörün i. elemanı, p_i yerindeki jetonların sayısını (pozitif tamsayı) belirtir. Yani $x(p_i) \in N$'dir. İşaretli PN'de jetonlar yerlerin içine konmuş siyah noktalar olarak gösterilir[1]. Şekil 2.3'te İşaretli PN'ye bir örnek gösterilmiştir:



$x_1 = [1, 0, 0]$ $x_2 = [2, 1, 0]$

Şekil 2.3 Aynı PN'ye ait iki farklı x_1 ve x_2 vektörü

Buradan itibaren İşaretli PN, sadece PN olarak anılacaktır. Bir PN aslında iki kısımdan oluşur[4]:

1. Sistemin statik bir modelini temsil eden graf yapısı
2. Sistemin herhangi bir andaki durumu temsil eden işaretlemeler

Jetonların PN'deki yerler arası dağılımına işaretleme denir. Eğer bir yerde bir ya da daha fazla jeton bulunuyorsa bu yer işaretli olarak tanımlanır. Aksi takdirde hiçbir jeton içermeyen bir yer işaretsiz olarak anılır. Bir yerde bulunan jeton sayısı o yere ilişkin durumu, PN grafında bulunan tüm jetonların yerler arasındaki dağılımı ise PN'nin genel durumunu gösterir. Başka bir deyişle bir PN'nin herhangi bir andaki durumu $x = [x(p_1), x(p_2), x(p_3), \ldots, x(p_n)]$ işaretleme vektörü ile tanımlanır. Sistemin dinamik yapısı ise aslında bu jetonların bir yerden başka bir yere akışıyla ve geçişlerin tetiklenmesiyle (firing) modellenir. Bir geçişin tetiklenmesi, jetonların bir

giriş yerinden alınıp başka bir çıkış yerine konması anlamına gelmektedir[1, 4]. Geçişlerin tetiklenmesi konusuna daha sonra değinilecektir.

Bir yere atanmış olan jeton sayısı pozitif bir tamsayı olmakla birlikte sonlu bir sayı olmak zorunda değildir. Buradan çıkan sonuç şudur: Genelde bir PN'nin sahip olabileceği durum sayısı aslında sonsuz olarak nitelendirilir. Sonuç olarak n adet yere sahip bir PN'nin X durum uzayı, pozitif tamsayılardan oluşan n – boyutlu vektörlerle tanımlanır: $X = N^n$[1].

Şu ana kadar PN grafının durum geçiş mekanizmasına değinilmedi. Aslında PN'nin dinamik sistem modellemesinde kullanılmasının asıl amacını durum geçişleri oluşturur. Durum geçiş mekanizmasını PN grafının yapısı belirler. Durum geçiş mekanizmasına değinilmeden önce "aktifleşmiş geçiş (enabled transition)" kavramının tanımı yapılmalıdır. Temelde bir $t \in T$ geçişinin aktif hale gelebilmesi için bu geçişe ait tüm giriş yerlerinde jetonların bulunması gerekir[1].

PN'deki bir $t_j \in T$ geçişinin aktif hale gelmesi için aşağıdaki koşul sağlanmalıdır[1]:

$$\text{Her } p_i \in I(t_j) \text{ için } x(p_i) \geq w(p_i, t_j) \text{ olmalıdır.}$$

Sözlerle ifade etmek gerekirse; PN'deki bir t_j geçişi, p_i giriş yerindeki jeton sayısı en az söz konusu p_i yerini t_j geçişine bağlayan yayın ağırlık değeri kadarsa aktifleşmiş demektir. Bu koşul t_j geçişine ait tüm p_i giriş yerleri için sağlanmalıdır[1, 4].



$x_1 = [1, 0]$ $x_2 = [2, 0]$

Şekil 2.4 Aktif ve pasif geçiş örneği

Şekil 2.4'te x_1 durumlu PN grafında, $x(p_1) = 1 < w(p_1, t_1)$ olduğundan t_1 geçişi aktif halde değildir. Fakat x_2 durumlu PN grafında $x(p_1) = 2 = w(p_1, t_1)$ olduğundan t_1 geçişi aktif hale gelmiştir[1].

Özetlemek gerekirse, PN grafındaki yerler ilgili bir geçişin gerçekleşebilmesi için gerekli koşulları temsil eder. Eğer bir geçişe ilişkin bütün koşullar sağlanıyorsa bu geçiş aktif hale gelir. Söz konusu koşulların sağlanmış olup olmadığı ise jetonlarla belirtilir[1].

2.4 Petri Ağları Dinamikleri

PN'deki durum geçiş mekanizması jetonların bir yerden başka bir yere taşınmasıyla sağlanır. Böylelikle PN'nin durumu değiştirilmiş olur ve PN bulunduğu durumdan başka bir duruma geçer. Bu da ancak aktifleşmiş geçişlerin tetiklenmesiyle (firing) mümkün olabilir. PN'ye ilişkin durum geçiş fonksiyonu, aktifleşmiş geçişlerin tetiklenmesiyle PN'nin bulunduğu durumda meydana gelen değişimler olarak tanımlanır[1].

Bir (P, T, A, w, x) PN'sine ilişkin $f : N^n \times T \rightarrow N^n$ durum geçiş fonksiyonu aşağıdaki koşulu sağlayan $t_j \in T$ geçişi için tanımlanmıştır:

$$\forall p_i \in I(t_j),\ x(p_i) \geq w(p_i, t_j) \tag{2.5}$$

Eğer $f(x, t_j)$ tanımlıysa, $x' = f(x, t_j)$ aşağıdaki gibi tanımlanır:

$$x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i),\ \ i = 1, \text{K}\ , n \tag{2.6}$$

Koşul (2.5), durum geçiş fonksiyonunun sadece aktifleşmiş geçişler için tanımlanmış olmasını garanti eder. Durum geçiş fonksiyonu PN grafının yapısıyla alakalıdır. Denklem (2.6)'da tanımlanmış bir sonraki durum ifadesi ise bir geçişin giriş ve çıkış yerlerine ve bu yerleri ilgili geçişe bağlayan yayların ağırlık değerlerine bağlıdır[1].

Denklem (2.6)'ya göre, eğer bir p_i yeri t_j'nin giriş yeri ise bu yerdeki jeton sayısı p_i'yi t_j'ye bağlayan yayın ağırlık değeri kadar azalır; eğer p_i yeri t_j'nin çıkış yeri ise bu yerdeki jeton sayısı t_j'yi p_i'ye bağlayan yayın ağırlık değeri kadar artar. Bir p_i yerinin bir t_j'ye ait hem giriş hem de çıkış yeri olması mümkündür. Bu durumda p_i yerinden $w(p_i, t_j)$ kadar jeton alınır ve hemen ardından p_i yerine $w(t_j, p_i)$ kadar jeton konur[1].

Bir PN'de herhangi bir geçişin tetiklenmesinin ardından bu geçiş tetiklenmeden önceki toplam jeton sayısının korunması gerekmez. Bu sonuca aşağıdaki olası eşitsizliklerden varabiliriz:

$$\sum_{p_i \in p} w(t_j, p_i) > \sum_{p_i \in p} w(p_i, t_j) \tag{2.7}$$

$$\sum_{p_i \in p} w(t_j, p_i) < \sum_{p_i \in p} w(p_i, t_j) \tag{2.8}$$

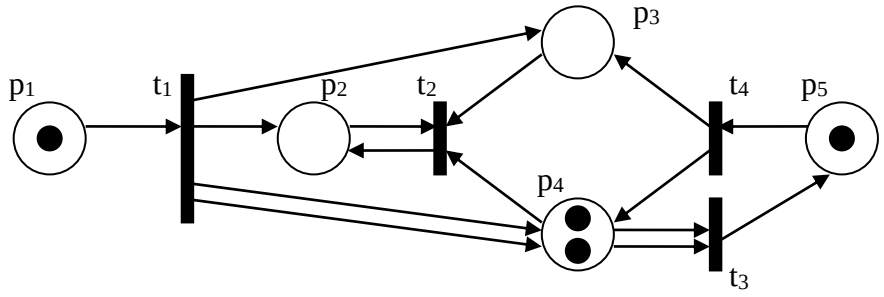
(2.7) eşitsizliğinin geçerli olması durumunda $x' = f(x, t_j)$ x'ten daha fazla sayıda jeton içerecektir. (2.8) eşitsizliğinin geçerli olması durumunda ise $x' = f(x, t_j)$ x'ten daha az sayıda jeton içerecektir. Bazı durumlarda birkaç geçişten sonra $x = [0, \dots, 0]$ durumuna varılması da mümkün olmaktadır. Bir başka ihtimal olarak da aşırı miktardaki geçişin ardından bazı yerlerdeki jeton sayısının büyük miktarlardaki artışı gösterilebilir[1].

Özetlemek gerekirse bir geçişin tetiklenmesi ile ilgili olarak aşağıdakiler söylenebilir:

1. Eğer bir geçişe ilişkin bütün giriş yerleri en az bu yerleri ilgili geçişe bağlayan yayların ağırlık değeri kadar jeton içeriyorsa söz konusu geçiş aktifleşmiş demektir[4].
2. Ancak aktifleşmiş bir geçiş tetiklenebilir. Böylelikle bir PN'nin bir durumdan başka bir duruma geçmesi sağlanabilir[4].
3. Aktifleşmiş bir geçiş tetiklendiğinde her bir giriş yerinden o giriş yerini söz konusu geçişe bağlayan yayın ağırlık değeri kadar jeton alınır[4].
4. Aktifleşmiş bir geçiş tetiklendiğinde her bir çıkış yerine söz konusu geçişi o çıkış yerine bağlayan yayın ağırlık değeri kadar jeton konulur[4].

Şunu belirtmekte fayda vardır: Aktifleşmiş bir geçiş her zaman tetiklenemeyebilir; çünkü bir geçiş olaylarla ilişkilendirilmiştir ve pratikte bazen PN'deki koşullar sağlanmış olsa bile geçişlerin tetiklenebilmesi için olaylara ilişkin bazı harici koşulların da sağlanmış olması gerekebilir[4].

Şekil 2.5'te örnek bir PN grafı verilmiştir. Bu PN grafının durum geçiş dinamikleri adım adım incelenecektir:



Şekil 2.5 $x_0 = [1, 0, 0, 2, 1]$ ilk durumlu örnek bir PN

Şekil 2.5'ten de görülebileceği gibi ancak t_1, t_3 ve t_4 geçişleri mümkündür; çünkü (2.5) koşuluna göre:

$$x(p_1) = 1 = w(p_1, t_1)$$

$$x(p_4) = 2 = w(p_4, t_3)$$

$$x(p_5) = 1 = w(p_5, t_4)$$

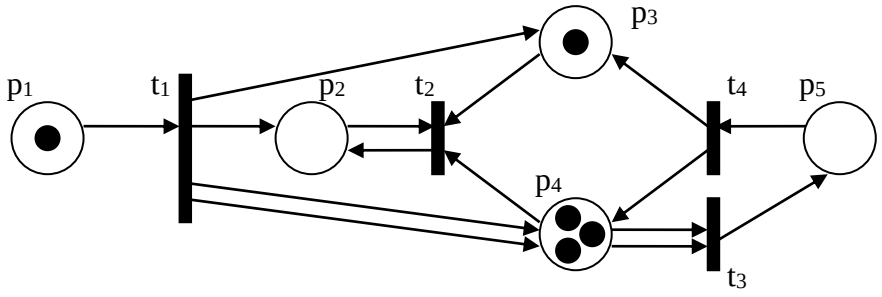
Eğer bir PN'de birden fazla aktifleşmiş geçiş varsa bu geçişler istenen sırada tetiklenebilir. Eğer t_4 geçişi tetiklenirse ve (2.6) denklemi uygulanırsa x_1 durumuna aşağıdaki gibi ulaşılır:

$$x_1(p_3) = x_0(p_3) + w(t_4, p_3) = 0 + 1 = 1$$

$$x_1(p_4) = x_0(p_4) + w(t_4, p_4) = 2 + 1 = 3$$

$$x_1(p_5) = x_0(p_5) - w(p_5, t_4) = 1 - 1 = 0$$

p_1 ve p_2 yerlerindeki jeton sayısı değişmeyeceği için x_1 = [1, 0, 1, 3, 0] olarak elde edilir. Bu vektöre ilişkin PN grafı Şekil 2.6'da gösterilmiştir:



Şekil 2.6 x_1= [1, 0, 1, 3, 0] durumlu PN

Şekil 2.6'dan da görülebileceği gibi ancak t_1 ve t_3 geçişleri mümkündür; çünkü (2.5) koşuluna göre:

$$x(p_1) = 1 = w(p_1, t_1)$$

$$x(p_4) = 3 > w(p_4, t_3) = 2$$

Eğer t_1 geçişi tetiklenirse ve (2.6) denklemi uygulanırsa x_2 durumuna aşağıdaki gibi ulaşılır:

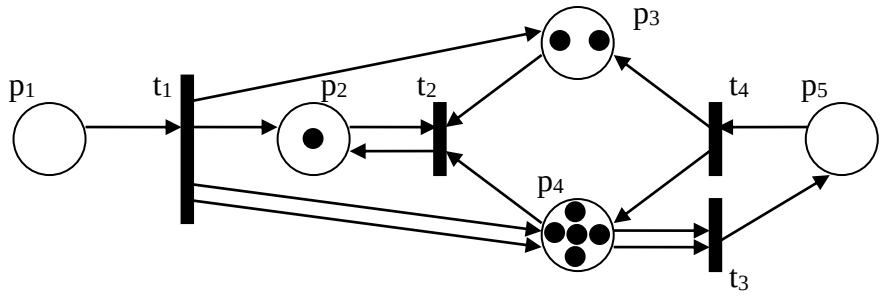
$$x_2(p_1) = x_1(p_1) - w(p_1, t_1) = 1 - 1 = 0$$

$$x_2(p_2) = x_1(p_2) + w(t_1, p_2) = 0 + 1 = 1$$

$$x_2(p_3) = x_1(p_3) + w(t_1, p_3) = 1 + 1 = 2$$

$$x_2(p_4) = x_1(p_4) + w(t_1, p_4) = 3 + 2 = 5$$

p_5 yerindeki jeton sayısı değişmeyeceği için $x_2 = [0, 1, 2, 5, 0]$ olarak elde edilir. Bu vektöre ilişkin PN grafı Şekil 2.7'de gösterilmiştir:



Şekil 2.7 x_2= [0, 1, 2, 5, 0] durumlu PN

Şekil 2.7'den de görülebileceği gibi ancak t_2 ve t_3 geçişleri mümkündür; çünkü (2.5) koşuluna göre:

$x(p_2) = 1 = w(p_2, t_2)$

$x(p_3) = 2 > w(p_3, t_2) = 1$ } t_2 geçişi için,

$x(p_4) = 5 > w(p_4, t_2) = 1$

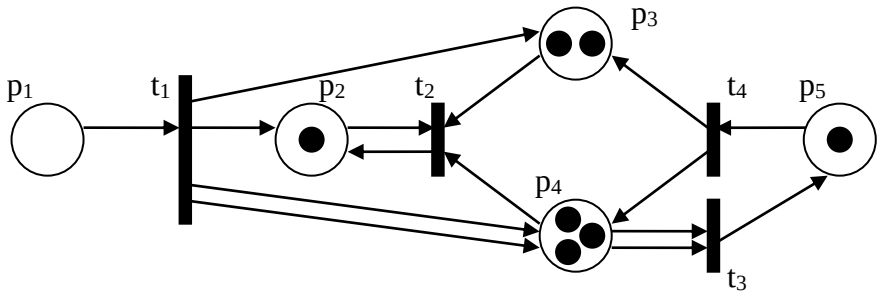
$x(p_4) = 5 > w(p_4, t_3) = 2$ } t_3 geçişi için.

Eğer t_3 geçişi tetiklenirse ve (2.6) denklemi uygulanırsa x_3 durumuna aşağıdaki gibi ulaşılır:

$$x_3(p_4) = x_2(p_4) - w(p_4, t_3) = 5 - 2 = 3$$

$$x_3(p_5) = x_2(p_5) + w(t_3, p_5) = 0 + 1 = 1$$

p_1, p_2 ve p_3 yerlerindeki jeton sayısı değişmeyeceği için x_3 = [0, 1, 2, 3, 1] olarak elde edilir. Bu vektöre ilişkin PN grafı Şekil 2.8'de gösterilmiştir:

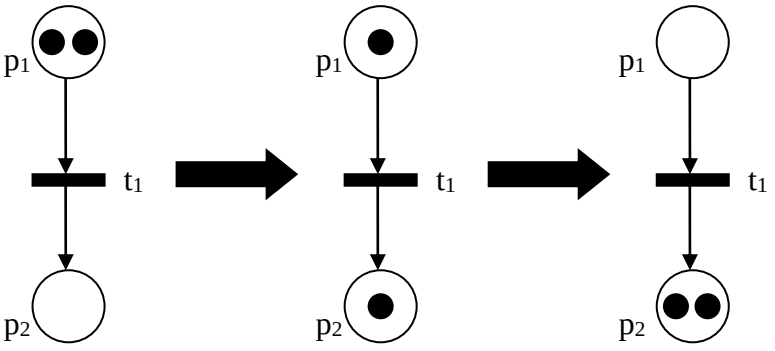


Şekil 2.8 x_3= [0, 1, 2, 3, 1] durumlu PN

Bu örnek daha da uzatılabilir ancak burada kesilmiştir. Örnekte jeton sayısı korunmamaktadır. x_0 durumu 4, x_1 durumu 5, x_2 durumu 8 ve x_3 durumu 7 adet jeton içermektedir.

2.4.1 Erişilebilir Durumlar

PN'de ele alınması gereken bir diğer konu ise mümkün olan bütün erişilebilir (reachable) durumlar kümesinin oluşturulmasıdır. Bir PN grafında her zaman N^n'deki bütün durumlara erişilmesi mümkün olmayabilir[1]. Bu amaçla Şekil 2.9'da verilen PN grafını inceleyelim:



x_0 = [2, 0] x_1 = [1, 1] x_2 = [0, 2]

Şekil 2.9 Örnek bir PN grafı için erişilebilir durumlar

Görüldüğü gibi x_0 = [2, 0] ilk durumlu PN grafında sadece x_1 = [1, 1] ve x_2 = [0, 2] durumlarına erişmek mümkün olmaktadır. O halde Şekil 2.9'daki PN grafına ilişkin R erişilebilir durumlar kümesi R = {[2, 0], [1, 1], [0, 2]} olarak gösterilebilir.

Bir (P, T, A, w, x) beşlisine ilişkin PN'ye ait erişilebilir durumlar kümesi aşağıdaki gibi tanımlanır[1]:

$$R[(P,T,A,w,x)] = \{y \in N^n : \exists s \in T^* \ (f(x,s) = y)\} \tag{2.9}$$

(2.9)'da verilmiş olan tanım şu şekilde yorumlanabilir: Herhangi bir PN grafına ilişkin R erişilebilir durumlar kümesi, söz konusu PN grafında mümkün olan bütün geçiş tetiklemeleri tek tek yapıldıktan sonra elde edilen durumlardan oluşur[1].

2.4.2 Durum Denklemleri

Buraya kadar bir PN grafına ilişkin durumları grafiksel olarak nasıl elde edeceğimizi öğrendik. Ancak grafiksel yöntemin kullanılması karmaşık yapıdaki PN'lerin incelenmesinde pek pratik bir yol değildir. Bu nedenle bir sonraki durum bilgisinin elde edilmesi için matematiksel yollara başvurulur. (2.6)'da verilmiş olan denklem incelenecek olursa, bu denklem sayesinde bir geçiş tetiklendiğinde bu geçişe ilişkin yerin bir sonraki durum değeri matematiksel olarak elde edilebilmektedir. Buradan yola çıkılarak PN'ye ilişkin bir sonraki durum vektörü matematiksel olarak hesaplanabilir. $x = [x(p_1), x(p_2), \ldots, x(p_n)]$ ilk durumuna sahip ve t_j geçişi tetiklenmiş olan bir PN'ye ait $x' = [x'(p_1), x'(p_2), \ldots, x'(p_n)]$ bir sonraki durum vektörü şu şekilde elde edilebilir[1]:

Öncelikle m – boyutlu bir u tetikleme vektörü (firing vector) aşağıdaki gibi tanımlanır[1]:

$$u = [0, \mathrm{K}, 0, 1, 0, \mathrm{K}, 0] \tag{2.10}$$

Burada u vektörünün j. elemanı 1'e eşit olur. Böylelikle j. geçişin tetikleneceği belirtilir. Ayrıca PN'ye ilişkin m × n boyutlu D sistem matrisi (incidence matrix) aşağıdaki gibi tanımlanır[1]:

$$d_{ji} = w(t_j, p_i) - w(p_i, t_j) \tag{2.11}$$

Bu ifadenin, denklem (2.6)'da gösterilen ağırlık farkına eşit olduğu görülebilir. Sonuç olarak D sistem matrisi kullanılarak PN'ye ilişkin bir sonraki durum vektörü aşağıdaki denklem sayesinde kolaylıkla elde edilebilir[1]:

$$x' = x + u.D \tag{2.12}$$

(2.12) denkleminde;

x′ : Bir sonraki durum vektörünü,

x : Başlangıç durum vektörünü,

u : Tetikleme vektörünü,

D : Sistem matrisini

ifade etmektedir.

Bu denklemde $f(x, t_j) = x + u.D$ olduğu görülebilir. Burada $f(x, t_j)$ durum geçiş fonksiyonudur ve t_j ise u tetikleme vektörünün j. elemanının 1'e eşit olduğunu göstermektedir[1].

(2.12)'de verilen denklem, geçişlerin tetiklenmesini ve PN'nin bir sonraki durum bilgisinin elde edilmesini matematiksel olarak gerçekleştirmemizi sağlar[1].

Şimdi önceden grafiksel olarak durumlarını bulduğumuz Şekil 2.5'de verilmiş olan $x_0 = [1, 0, 0, 2, 1]$ ilk durumlu PN grafının t_4 geçişi tetiklendiği zaman sahip olacağı x_1 durumunu matematiksel olarak hesaplayalım:

Öncelikle bu PN grafına ilişkin D sistem matrisini elde edelim:

$d_{11} = w(t_1, p_1) - w(p_1, t_1) = 0 - 1 = -1$	$d_{31} = w(t_3, p_1) - w(p_1, t_3) = 0 - 0 = 0$
$d_{12} = w(t_1, p_2) - w(p_2, t_1) = 1 - 0 = 1$	$d_{32} = w(t_3, p_2) - w(p_2, t_3) = 0 - 0 = 0$
$d_{13} = w(t_1, p_3) - w(p_3, t_1) = 1 - 0 = 1$	$d_{33} = w(t_3, p_3) - w(p_3, t_3) = 0 - 0 = 0$
$d_{14} = w(t_1, p_4) - w(p_4, t_1) = 2 - 0 = 2$	$d_{34} = w(t_3, p_4) - w(p_4, t_3) = 0 - 2 = -2$
$d_{15} = w(t_1, p_5) - w(p_5, t_1) = 0 - 0 = 0$	$d_{35} = w(t_3, p_5) - w(p_5, t_3) = 1 - 0 = 1$
$d_{21} = w(t_2, p_1) - w(p_1, t_2) = 0 - 0 = 0$	$d_{41} = w(t_4, p_1) - w(p_1, t_4) = 0 - 0 = 0$
$d_{22} = w(t_2, p_2) - w(p_2, t_2) = 1 - 1 = 0$	$d_{42} = w(t_4, p_2) - w(p_2, t_4) = 0 - 0 = 0$
$d_{23} = w(t_2, p_3) - w(p_3, t_2) = 0 - 1 = -1$	$d_{43} = w(t_4, p_3) - w(p_3, t_4) = 1 - 0 = 1$
$d_{24} = w(t_2, p_4) - w(p_4, t_2) = 0 - 1 = -1$	$d_{44} = w(t_4, p_4) - w(p_4, t_4) = 1 - 0 = 1$
$d_{25} = w(t_2, p_5) - w(p_5, t_2) = 0 - 0 = 0$	$d_{45} = w(t_4, p_5) - w(p_5, t_4) = 0 - 1 = -1$

$$D = \begin{bmatrix} -1 & 1 & 1 & 2 & 0 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

t_4 geçişi tetikleneceği için u = [0 0 0 1] olur. Buradan bir sonraki durum vektörü;

$$x' = x + u.D$$

$$x' = [1\ 0\ 0\ 2\ 1] + [0\ 0\ 0\ 1].\begin{bmatrix} -1 & 1 & 1 & 2 & 0 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

$$x' = [1\ 0\ 0\ 2\ 1] + [0\ 0\ 1\ 1\ -1] = [1\ 0\ 1\ 3\ 0]$$

olarak bulunur. Bulunan bu sonucun Şekil 2.5'te verilen örnekte grafiksel olarak bulunan sonuçla aynı olduğu görülür. Aynı örnek için PN grafının t_1 geçişi tetiklendiği zaman sahip olacağı x_2 durumunu benzer bir şekilde hesaplayabiliriz:

$$x_2 = [1\ 0\ 1\ 3\ 0] + [1\ 0\ 0\ 0].\begin{bmatrix} -1 & 1 & 1 & 2 & 0 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

$$x_2 = [1\ 0\ 1\ 3\ 0] + [-1\ 1\ 1\ 2\ 0] = [0\ 1\ 2\ 5\ 0]$$

Sonuç olarak x_0 ilk durumu verilmiş bir PN grafına ilişkin tüm durumlar aşağıdaki denklem sayesinde elde edilebilir[1]:

$$x_{k+1} = f(x_k, t_k) = x_k + u_k.D \tag{2.13}$$

Burada u_k, k. geçişin tetikleneceği bilgisini taşımaktadır. Eğer hiçbir geçişin tetiklenemeyeceği bir duruma gelindiyse, PN için ölümcül kilitlenme (deadlock) gerçekleşir[1]. Bu konuya daha sonra değinilecektir.

3. PETRİ AĞLARI İLE MODELLEME

PN'lerin kullanımındaki asıl amaç gerçek hayatta karşımıza çıkan sistemlerin modellenmesidir. Bu sistemlerin modellenmesinde kullanılan PN'ler, koşullar ve bu koşullara bağlı olarak gerçekleşen olaylardan oluşur. PN grafındaki yerler modellenen sistemin belirli kısımlarına ilişkin koşulları (örneğin bir makinenin boşta olması, çalışıyor olması ya da görevini yerine getiremiyor olması gibi) belirtirken PN grafındaki geçişler ise koşullara bağlı olarak bir olayın gerçekleşmesini (örneğin görevini yerine getiremeyen bir makinenin tamir edilmesi gibi) temsil eder. Bir olayın gerçekleşebilmesi yani PN grafındaki ilgili geçişin tetiklenebilmesi için bu olaya ilişkin tüm koşulların sağlanmış olması gerekir. Başka bir deyişle bu olayla ilgili geçişe ait tüm giriş yerlerinde yeterince jeton bulunması gerekir. Bir olayın gerçekleşmesi PN grafının işaretleme vektörünü yani o anki durumunu değiştirip başka bir duruma geçişine neden olur. PN grafındaki bir yerde bulunan jeton sayısı bu yere ilişkin koşulları temsil edebileceği gibi sistemin yapısına bağlı olarak ilgili olayın gerçekleşebilmesi için gerekli kaynak (resource) sayısını da belirtebilir[3].

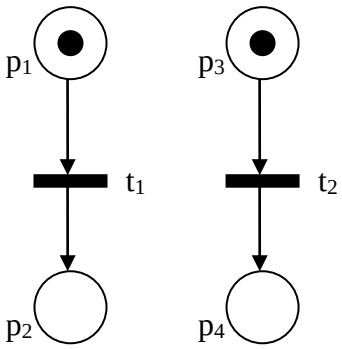
3.1 Petri Ağları Yapıları

Sistem modellemesinde kullanılan PN yapıları aşağıdaki gibi verilebilir[3]:

1. Paralellik (Concurrency)
2. Senkronizasyon (Synchronization)
3. Çakışma (Conflict)
4. Sıralama (Sequentiality)
5. Kaynak Sınırlama (Limited Resources)

3.1.1 Paralellik

Bağımsız olarak gerçekleşebilen iki ya da daha fazla olayın modellenmesinde kullanılan PN yapısıdır. Bu PN yapısında olaylar birbirinden bağımsız olarak aynı anda gerçekleşebilir[3].

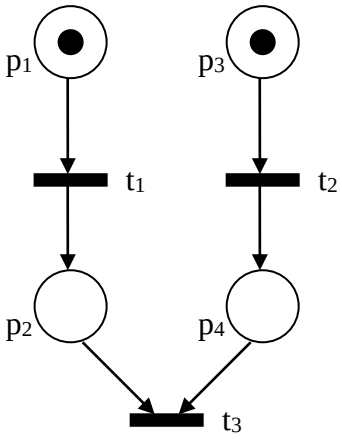


Şekil 3.1 Paralellik örneği

Şekil 3.1'de verilen PN grafında t_1 ve t_2 geçişleri aynı anda tetiklenmektedir[3].

3.1.2 Senkronizasyon

Bir geçişin gerçekleşebilmesi için iki ya da daha fazla koşulun sağlanmış olmasını gerektiren PN yapısıdır. Bu PN yapısında herhangi bir geçiş ancak tüm koşullar sağlandığı zaman tetiklenebilir. Aksi takdirde eğer koşullardan en az biri bile sağlanmıyorsa ilgili geçiş tetiklenmez[3].



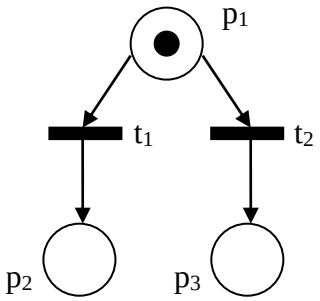
Şekil 3.2 Senkronizasyon örneği

Şekil 3.2'de verilen PN grafında t_3 geçişinin tetiklenebilmesi için hem t_1 hem de t_2 geçişinin tetiklenmiş olması gerekmektedir[3].

3.1.3 Çakışma

Bir sistemde aynı kaynağı paylaşan iki ya da daha fazla olay olabilir. Bu durumda kaynağa aynı anda yalnızca bir olay erişmeli ve diğer olayların söz konusu kaynağa olan erişimleri engellenmelidir[3].

İki ya da daha fazla olay tarafından paylaşılmış olan kaynağa aynı anda yalnızca bir olayın erişmesinin sağlandığı PN yapısı çakışma olarak adlandırılır[3].

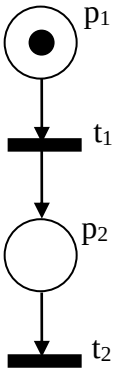


Şekil 3.3 Çakışma örneği

Şekil 3.3'te verilen PN grafında p_1 yeri hem t_1 hem de t_2 geçişleri tarafından paylaşılmakta olan ortak bir kaynağı temsil etmektedir. Bu durumda t_1 ve t_2 geçişleri aynı anda gerçekleşemez. Başka bir deyişle ya t_1 geçişi ya da t_2 geçişi tetiklenecektir. Bunun seçimi sistemle ilgili harici koşullara bağlı olacaktır[3].

3.1.4 Sıralama

Bölüm 2.4'te bir PN grafında olayların gerçekleşme sırasında herhangi bir sınırlama olmadığından ve geçişlerin istenen sırada tetiklenebileceğinden bahsedilmişti. Ancak bunun aksi olan durumlar da söz konusu olabilir. Yani olayların sırasının önemli olduğu sistemlerin modellenmesi gerekebilir. Bu durumda sıralama adı verilen PN yapısı kullanılır. Böylelikle olayların gerçekleşme sırası belirlenip modellenebilir[3].

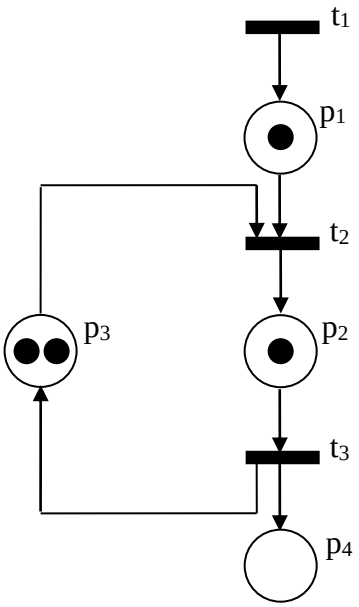


Şekil 3.4 Sıralama örneği

Şekil 3.4'te verilen PN grafında t_1 geçişi her zaman t_2 geçişinden önce gerçekleşir. Böylelikle olayların gerçekleşme sırası modellenmiş olur[3].

3.1.5 Kaynak Sınırlama

Dağıtık (distributed) sistemler ve üretim (manufacturing) sistemleri gibi sistemlerin performansını etkileyen bir faktör mevcut kaynakların sınırlı olmasıdır. Kaynakların tükenmesi işlemlerin yürümesini engelleyip sistemin çalışmasını durdurabilir. Bu tür sistemlerin modellenmesinde kaynak sınırlama adı verilen PN yapısı kullanılır[3].

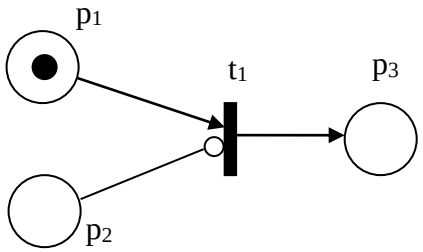


Şekil 3.5 Kaynak sınırlama örneği

Şekil 3.5'te verilen PN grafında kaynakları sınırlı bir kuyruk modellenmiştir. Bu kuyruğun bir alışveriş merkezine ait alışveriş kuyruğu olduğu düşünülebilir. O halde PN grafındaki p_3 yeri bu kuyrukta boşta olan pozisyon sayısını belirtirken p_2 yeri ise dolmuş pozisyon sayısını göstermektedir. Eğer dikkat edilecek olursa p_2 ve p_3 yerlerindeki toplam jeton sayısı sabit olup kuyruktaki toplam boş pozisyon sayısını ifade etmektedir. Bu örnekte kuyruktaki toplam boş pozisyon sayısı üç'tür. p_1 yeri gelen müşterileri ve p_4 yeri ise işini bitirip kuyruktan ayrılan müşterileri temsil etmektedir. t_2 geçişi bir müşterinin kuyruğa girmesini ya da başka bir deyişle bir pozisyonun bir müşteri tarafından doldurulmasını temsil etmektedir. t_2 geçişinin tetiklenebilmesi için p_1 ve p_3 yerlerinde en az bir jeton bulunmalıdır. Başka bir deyişle kuyruğa giren bir müşteri olmalı ve kuyrukta bu müşteri için boşta bir pozisyon bulunmalıdır. t_3 geçişi bir müşterinin işini bitirip kuyruktan ayrılmasını temsil etmektedir. Bunun için p_2 yerinde en az bir jeton bulunmalıdır, yani kuyrukta en az bir müşteri olmalıdır. t_3 geçişi tetiklendiğinde bir müşteri kuyruktan ayrılır ve boş pozisyon sayısı bir artar, yani p_2 yerinden p_3 yerine bir jeton geçmiş olur[3].

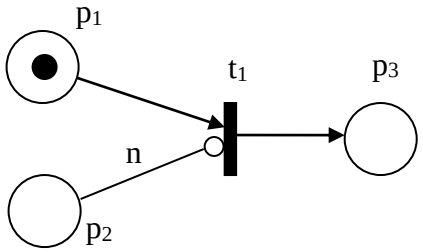
3.2 Bozucu Yaylar

PN grafında p_i yerinden t_j geçişine çizilmiş bir bozucu (inhibitor) yay t_j geçişinin tetiklenme mekanizmasını şu şekilde etkiler: Eğer p_i yerinde en az bir jeton varsa, t_j geçişi aktif hale gelmiş bile olsa tetiklenemez. Bozucu yay, ucu daire şeklinde biten yayla gösterilir[3].



Şekil 3.6 Bozucu yay örneği

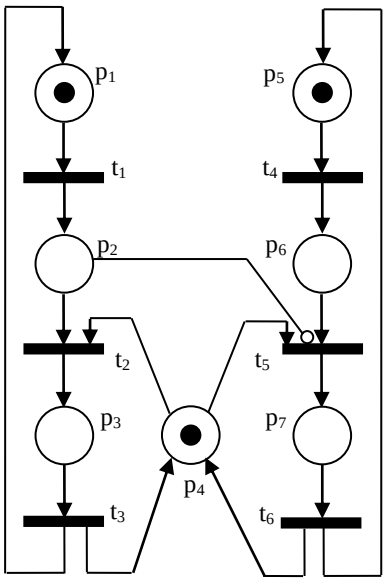
Şekil 3.6'da verilen örnekte p_2 yerinden t_1 geçişine çizilmiş ucu daire şeklinde biten yay bir bozucu yaydır. p_1 geçişinde bir jeton bulunduğu için t_1 geçişi aktif durumdadır ve p_2 yerinde herhangi bir jeton bulunmadığı için t_1 geçişi tetiklenebilecektir. Eğer p_2 yerinde en az bir jeton bulunsaydı t_1 geçişi tetiklenemeyecekti[3].



Şekil 3.7 Çoklu bozucu yay örneği

Şekil 3.7'de verilen örnekte ise p_2 yerinden t_1 geçişine çizilmiş olan n adet bozucu yay mevcuttur. Bu durumda eğer p_2 yerinde en az n adet jeton bulunursa t_1 geçişi tetiklenemez. Diğer durumlarda t_1 geçişi tetiklenebilecektir[3].

Bozucu yayların kullanılmasındaki asıl amaç çakışma durumunda olan iki ya da daha fazla geçişe bir öncelik atamak ve atanan öncelik sırasına göre bu geçişlerin tetiklenmesini sağlamaktır. Şekil 3.8'de bu durumu ele alan bir örnek verilmiştir[3].



Şekil 3.8 Geçişlere öncelik atama

Şekil 3.8'de verilen PN grafında p_2 yerinden t_5 geçişine çizilmiş bir bozucu yay görülmektedir. İlk durumda yalnızca t_1 ve t_4 geçişi aktif durumdadır ve paralel olarak tetiklenebilir. t_1 ve t_4 geçişleri tetiklendikten sonra p_2 ve p_6 yerleri birer jeton kazanacağından ve p_4 yerinde bir jeton bulunduğundan hem t_2 hem de t_5 geçişi aktif durumda olacaktır. Bu durumda t_2 ve t_5 geçişi p_4 yerindeki kaynağı ortak olarak kullandığından çakışma içindedir. Çakışma durumundaki bu geçişlerden birine öncelik vermek için bozucu yay kullanılmaktadır. Örnekteki bozucu yay p_2 yerinde bir jeton bulunmasından dolayı t_5 geçişinin tetiklenmesini engeller ve t_2 geçişinin tetiklenmesine olanak sağlar. Başka bir deyişle bozucu yay t_2 geçişini t_5 geçişinden daha öncelikli konuma getirir ve böylelikle çakışma durumunda öncelikle t_2 geçişi tetiklenir. Bu durumda t_5 geçişi daha sonra tetiklenecektir[3].

3.3 Petri Ağlarına Eklentiler

Şimdiye kadar incelediğimiz PN'lerde modelleme bakımından birtakım eksiklikler mevcuttur. Bu eksiklikler aşağıdaki gibi sıralanabilir[3]:

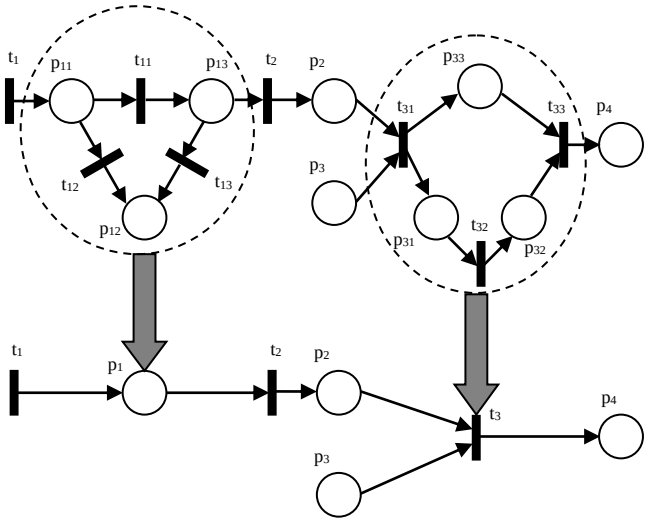
- Karmaşık yapıdaki PN'lerin incelenmesi bir hayli zordur[3].
- Jetonların tek tek ayırt edilebilmesi mümkün değildir[3].
- Zamana dayalı sistemler modellenemez[3].

Bu eksikliklerin giderilmesi ve PN'lerin modelleme kabiliyetinin daha da arttırılması amacıyla PN'lere bir takım eklentiler (extension) yapılmıştır. Bu eklentiler aşağıdaki gibi sıralanabilir[3]:

1. Modüler (hierarchical) PN'ler
2. Renkli (coloured) PN'ler
3. Zaman etiketli (timed) PN'ler

3.3.1 Modüler Petri Ağları

Karmaşık yapıdaki PN'lerin analiz işlemlerinin kolaylaştırılması ve bu amaçla grafiksel gösteriminin sadeleştirilmesi gerekebilir. Bunun için PN'lerin modüler olarak ifade edilmesi yoluna gidilir[5].



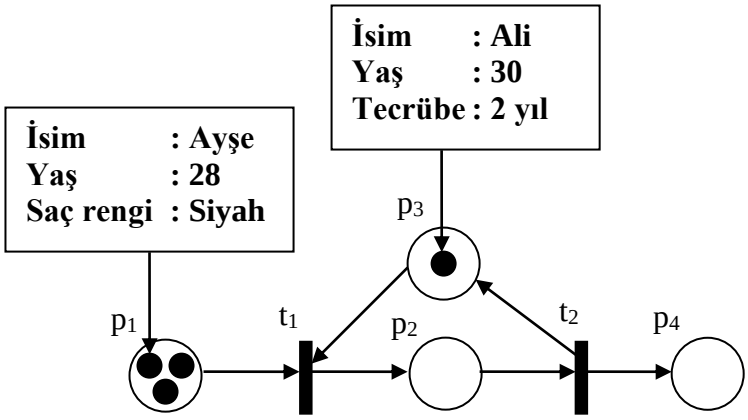
Şekil 3.9 Modüler PN örneği

Şekil 3.9'da verilen örnekte üstte bulunan PN grafı modülerlik kullanılıp sadeleştirilerek altta bulunan PN grafına dönüştürülmüştür. Dikkat edilecek olursa sadeleştirme sonucunda p_1 yeri ve t_3 geçişi elde edilmiştir. Böylelikle analiz edilmesi daha kolay bir PN grafı oluşturulmuştur. Modülerliğin sağladığı avantajlar aşağıdaki gibi özetlenebilir[5]:

- PN grafındaki herhangi bir yere ya da yerler topluluğuna ilişkin kısmi durum bilgisi ve kısmi durum değişiklikleri, tüm PN grafının durum bilgisi ve durum değişikliklerinden bağımsız olarak elde edilebilir[5].
- Modellenen sistemler hiyerarşik yapıda gösterilebilir. Örneğin tüm PN grafı tek bir yer ya da tek bir geçiş olarak gösterilebilir[5].

3.3.2 Renkli Petri Ağları

Bundan önce ele alınan PN'ler, tek bir jetonun diğer jetonlardan ayırt edilmesine ve bu jetonun tetiklemeler sırasında bir yerden başka bir yere akışının takip edilmesine olanak tanımamaktadır. Bu eksikliğin giderilmesi amacıyla Renkli PN kavramı ortaya atılmıştır. Renkli PN'lerde jetonlara değişik renkler/özellikler atanır. Atanan bu renkler/özellikler jetonların birbirinden ayırt edilebilmesini sağlar. Böylelikle spesifik bir jetonun PN grafı içindeki yeri tam olarak takip edilebilir[3, 6].



Şekil 3.10 Renkli PN örneği

Şekil 3.10'da verilen PN grafı bir bayan kuaförünü modellemektedir. Bu modeldeki yerlerin ve geçişlerin karşılığı aşağıda verilmiştir[6]:

p_1 : Kuyrukta bekleyen müşteriler

p_2 : Hizmet alan müşteri

p_3 : Müşteriye hizmet veren kuaför

p_4 : İşi biten müşteriler

} Yerler

t_1 : Kuyrukta bekleyen müşterinin hizmet almaya başlaması

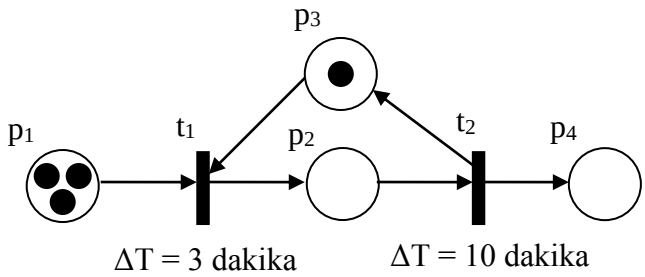
t_2 : Müşterinin işinin bitmesi

} Geçişler

Örnekte müşterileri simgeleyen jetonlardan birine ve kuaförü simgeleyen jetona belirli özellikler atanmıştır. Böylelikle bu jetonların birbirinden ayırt edilebilmesi sağlanmıştır. Şekil 3.10'daki PN'de kuaförün belirli bir zamanda hangi müşteri ile ilgilendiği kolaylıkla takip edilebilir veya hangi müşterilerin işlerinin bitmiş olduğu hemen anlaşılabilir[6].

3.3.3 Zaman Etiketli Petri Ağları

Olayların meydana gelmesinde zamanlamanın önemli olduğu sistemlerin modellenmesi amacıyla Zaman Etiketli PN'ler kullanılır. Zaman Etiketli PN'lerde geçişlerin tetiklenebilmesi için bir zamanlama mekanizması mevcuttur. Bu tip PN'lerde bazı geçişlere bir ΔT gecikme süresi atanmaktadır. Eğer zaman etiketi atanmış bir geçiş aktif hale gelmişse ΔT süresi kadar bekler ve ondan sonra tetiklenir. Sadece tetikleme gerçekleştikten sonra ilgili girişe ait jetonlar giriş yerlerinden alınır ve çıkış yerlerine konulur. Aktif hale gelmiş zaman etiketli bir geçiş tetikleninceye kadar bu geçişe ilişkin hiçbir jeton transferi gerçekleşmez[3, 6].

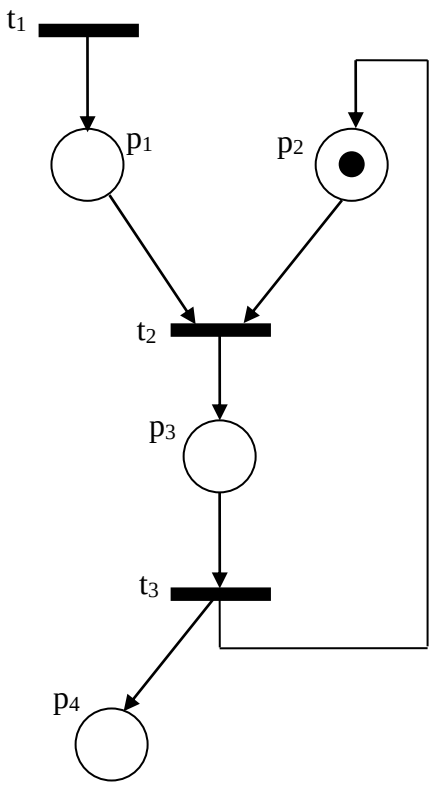


Şekil 3.11 Zaman etiketli PN örneği

Şekil 3.11'de verilen PN grafı ile Şekil 3.10'da verilmiş olan bayan kuaförü örneği Zaman Etiketli PN'ye dönüştürülmüştür. Bu durumda kuaför boşta olduğunda ve kuyrukta en az bir müşteri bulunduğunda t_1 geçişi aktif hale gelecek fakat hemen tetiklenmeyecektir; çünkü t_1 geçişi ΔT = 3 dakikalık bir gecikme süresine sahiptir ve aktif hale geldikten ancak 3 dakika sonra tetiklenebilecektir. Yani kuyrukta bekleyen müşteri, kuaför boşta olduktan tam 3 dakika sonra hizmet almaya başlayacaktır. Aslında bu süre ile kuaförün bir sonraki müşteri için hazırlanma süresi modellenmektedir. Müşteri hizmet almaya başladıktan tam 10 dakika sonra işi bitecek ve kuaförden ayrılacaktır. Ayrıca bu süre sonunda kuaför işini bitirmiş olacak ve bir diğer müşteri için bekliyor durumda olacaktır. Bu süre ile de kuaförün işini ne kadar sürede tamamladığı modellenmektedir[6].

3.4 Modelleme Örnekleri

3.4.1 Kuyruk Sisteminin Modellenmesi



Şekil 3.12 Örnek bir kuyruk sisteminin PN ile modellenmesi

Şekil 3.12'de örnek bir kuyruk sistemi PN ile modellenmiştir. Bu kuyruğun bir bilgisayar sisteminde işlenmeyi bekleyen süreçlere (process) ait olduğu düşünülürse bu kuyruk sistemine ilişkin olaylar ve koşullar kümesi aşağıdaki gibi belirtilebilir[5]:

Olaylar: (Geçişler)

t_1 : Yeni bir sürecin kuyruğa girmesi

t_2 : Sürecin işleme alınması

t_3 : Sürecin işlenmesinin tamamlanması

Koşullar: (Yerler)

p_1 : Kuyrukta bekleyen süreçler

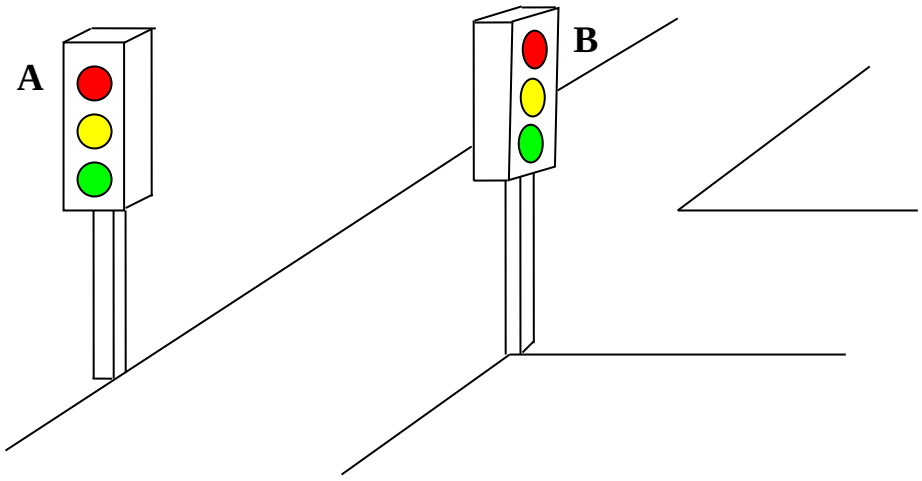
p_2 : İşlemcinin boşta (idle) olması

p_3 : İşleme alınan süreçler

p_4 : İşlemleri tamamlanmış süreçler

Bu kuyruk sistemi incelenecek olursa öncelikle t_1 geçişinin koşulsuz olarak kendiliğinden tetiklendiği göze çarpar. t_1 geçişinin tetiklenmesi ile işlenmek üzere kuyruğa yeni bir süreç eklenir. Eğer işlemci boşta değilse söz konusu süreç hemen işlenmeye alınmaz ve işlemcinin işini bitirmesini bekler. İşlemci işini bitirdikten sonra eğer kuyrukta işlenmeyi bekleyen en az bir süreç mevcut ise bu süreç kuyruktan çekilip işleme alınır. Başka bir deyişle t_2 geçişi tetiklenir. Sürecin işlemi tamamlandıktan sonra t_3 geçişi tetiklenir. Bu tetiklenme sonucu işi tamamlanmış olan süreç işlemciyi terk eder ve işlemci tekrar boşta konumuna geri döner. Başka bir deyişle süreci temsil eden jeton p_4 yerine konulur ve işlemciyi temsil eden jeton ise p_2 yerine geri döner[5].

3.4.2 Trafik Işıklarının Modellenmesi



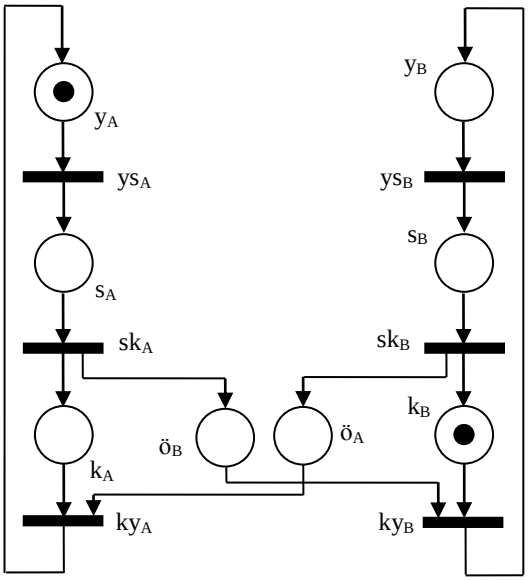
Şekil 3.13 Modellenecek trafik ışıklarının yerleşimi

Şekil 3.13'te modellenecek olan trafik ışıklarının kavşak içindeki yerleşimi gösterilmiştir. İlk durumda A trafik ışığının yeşil ve B trafik ışığının ise kırmızı yandığı varsayılmıştır. Bu durumda A ve B trafik ışıklarına ilişkin yanma sırası Tablo 3.1'de verilmiştir[6]:

Tablo 3.1 Modellenecek trafik ışıklarının yanma sırası

Trafik Işığı – A	Trafik Işığı – B
Yeşil	Kırmızı
Sarı	Kırmızı
Kırmızı	Yeşil
Kırmızı	Sarı

Tablo 3.1'de verilen trafik ışıklarının yanma sırası periyodik olarak aynı şeklide devam edecektir. Yani son olarak A trafik ışığı kırmızı ve B trafik ışığı sarı yandıktan sonra yanma sırası tekrar başa dönüp A trafik ışığı yeşil ve B trafik ışığı ise kırmızı yanacaktır[6].



Şekil 3.14 Trafik ışıklarının PN ile modellenmesi

Şekil 3.14'te trafik ışıklarının PN ile modellenmiş hali gösterilmektedir. Bu PN grafındaki yerler ve geçişlerin açıklaması aşağıdaki gibi verilebilir:

Yerler:

y_A : Trafik ışığı – A'nın yeşil yanması

s_A : Trafik ışığı – A'nın sarı yanması

k_A : Trafik ışığı – A'nın kırmızı yanması

y_B : Trafik ışığı – B'nin yeşil yanması

s_B : Trafik ışığı – B'nin sarı yanması

k_B : Trafik ışığı – B'nin kırmızı yanması

$ö_A$: Yeşil yanmadaki öncelik sırasının Trafik ışığı A'da olması

$ö_B$: Yeşil yanmadaki öncelik sırasının Trafik ışığı B'de olması

Geçişler:

ys_A : Trafik ışığı – A'nın yeşilden sarıya geçmesi

sk_A : Trafik ışığı – A'nın sarıdan kırmızıya geçmesi

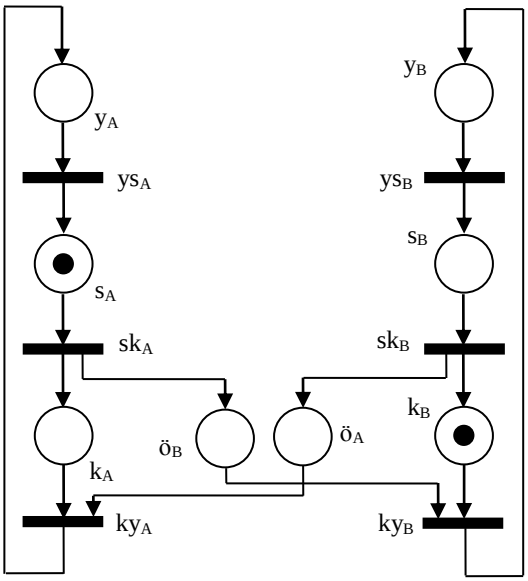
ky_A : Trafik ışığı – A'nın kırmızıdan yeşile geçmesi

ys_B : Trafik ışığı – B'nin yeşilden sarıya geçmesi

sk_B : Trafik ışığı – B'nin sarıdan kırmızıya geçmesi

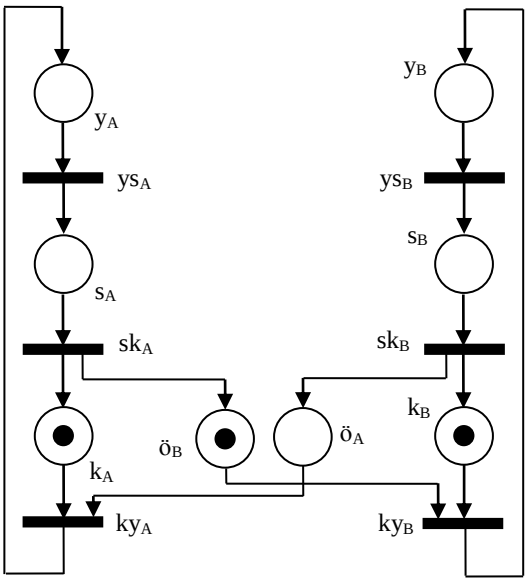
ky_B : Trafik ışığı – B'nin kırmızıdan yeşile geçmesi

Şekil 3.14'teki PN modeli incelenecek olursa, ilk durumda A trafik ışığının yeşil ve B trafik ışığının ise kırmızı yandığı görülebilir. Bu durumda ky_B geçişi tetiklenemeyecektir; çünkü bu geçişin giriş yerlerinden biri olan $ö_B$ yerinde herhangi bir jeton bulunmamaktadır. Tetiklenebilecek durumda olan tek geçiş ys_A geçişidir. Şekil 3.15, PN grafının ys_A geçişi tetiklendikten sonraki durumunu göstermektedir.



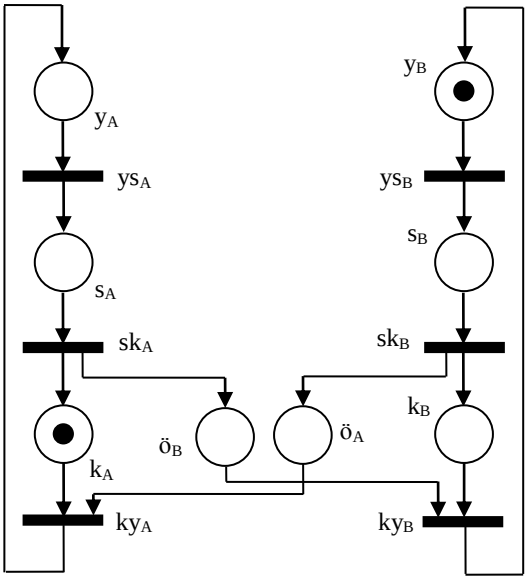
Şekil 3.15 Trafik ışıklarının ys_A geçişinin tetiklenmesinden sonraki durumu

Şekil 3.15'e göre A trafik ışığı sarı ve B trafik ışığı ise hala kırmızı yanmaktadır. Bu durum Tablo 3.1'in ikinci satırındaki yanma sırasına denk gelir. Bu sefer tetiklenebilecek olan tek geçiş sk_A geçişidir. Şekil 3.16, PN grafının sk_A geçişi tetiklendikten sonraki durumunu göstermektedir.



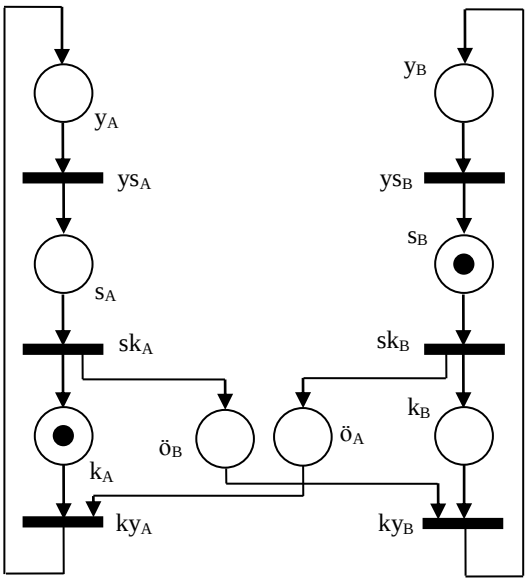
Şekil 3.16 Trafik ışıklarının sk_A geçişinin tetiklenmesinden sonraki durumu

Şekil 3.16'ya göre A trafik ışığı kırmızı ve B trafik ışığı ise hala kırmızı yanmaktadır. Bu sefer ky_B geçişi aktif hale gelmiş durumdadır; çünkü her iki giriş yerinde de birer jeton bulunmaktadır. Sonuç olarak ky_B geçişi tetiklenecektir. Şekil 3.17, PN grafının ky_B geçişi tetiklendikten sonraki durumunu göstermektedir.



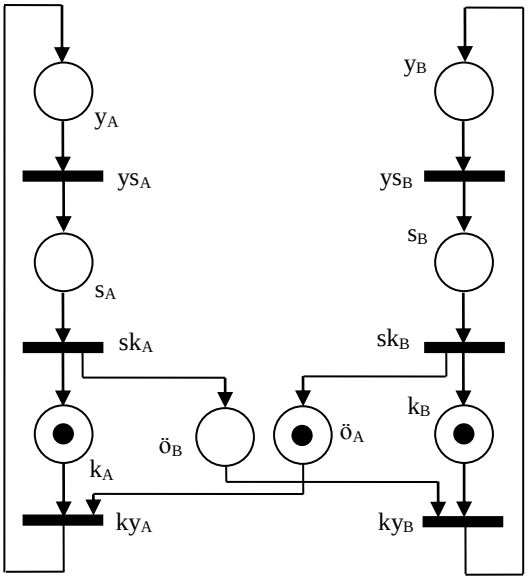
Şekil 3.17 Trafik ışıklarının ky_B geçişinin tetiklenmesinden sonraki durumu

Şekil 3.17'ye göre A trafik ışığı hala kırmızı ve B trafik ışığı ise yeşil yanmaktadır. Bu durum Tablo 3.1'in üçüncü satırındaki yanma sırasına denk gelir. Bu sefer tetiklenebilecek olan tek geçiş ys_B geçişidir. Şekil 3.18, PN grafının ys_B geçişi tetiklendikten sonraki durumunu göstermektedir.



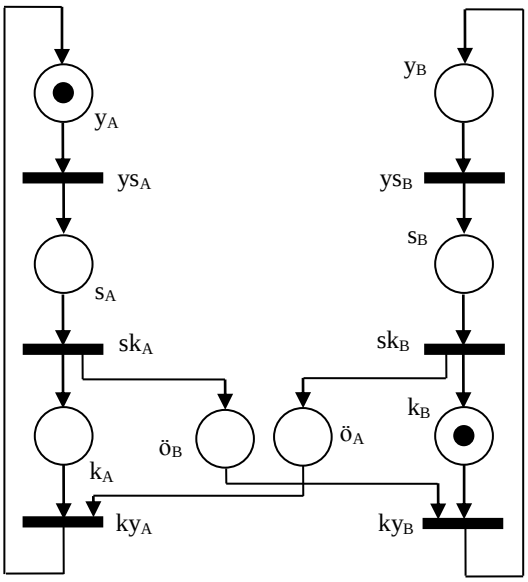
Şekil 3.18 Trafik ışıklarının ys_B geçişinin tetiklenmesinden sonraki durumu

Şekil 3.18'e göre A trafik ışığı hala kırmızı ve B trafik ışığı ise sarı yanmaktadır. Bu durum Tablo 3.1'in dördüncü (son) satırındaki yanma sırasına denk gelir. Bu sefer tetiklenebilecek olan tek geçiş sk_B geçişidir. Şekil 3.19, PN grafının sk_B geçişi tetiklendikten sonraki durumunu göstermektedir.



Şekil 3.19 Trafik ışıklarının sk_B geçişinin tetiklenmesinden sonraki durumu

Şekil 3.19'a göre A trafik ışığı hala kırmızı ve B trafik ışığı ise kırmızı yanmaktadır. Bu sefer ky_A geçişi aktif hale gelmiş durumdadır; çünkü her iki giriş yerinde de birer jeton bulunmaktadır. Sonuç olarak ky_A geçişi hemen tetiklenecektir. Şekil 3.20, PN grafının ky_A geçişi tetiklendikten sonraki durumunu göstermektedir.

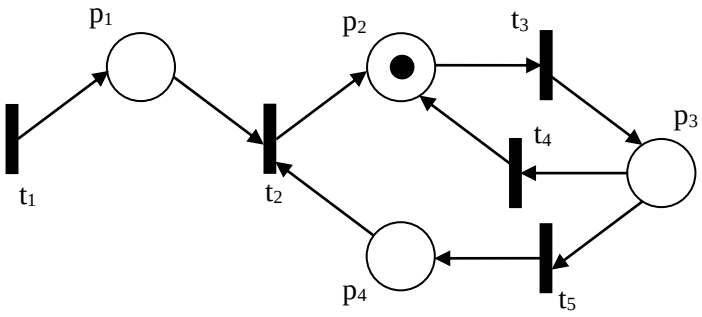


Şekil 3.20 Trafik ışıklarının ky_A geçişinin tetiklenmesinden sonraki durumu

Şekil 3.20'ye göre A trafik ışığı yeşil ve B trafik ışığı ise kırmızı yanmaktadır. Bu durum Tablo 3.1'in ilk satırındaki yanma sırasına denk gelir. Yani ilk duruma geri dönülmüştür ve bu yanma sırası Tablo 3.1'de belirtilmiş sıralamaya uygun olarak aynen devam edecektir.

3.4.3 Meşrubat Otomatının Modellenmesi

Madeni parayla çalışan bir meşrubat otomatı PN grafı ile Şekil 3.21'deki gibi modellenebilir[7]:



Şekil 3.21 Örnek bir meşrubat otomatının PN ile modellenmesi

Şekil 3.21'deki PN grafındaki yerler ve geçişlere ilişkin açıklama aşağıdaki gibi verilmiştir:

Yerler:

p_1 : Otomatın meşrubat haznesi

p_2 : Otomatın hazır durumda olması

p_3 : Otomatın madeni para haznesi

p_4 : Otomatın meşrubat vermek için hazır durumda olması

Geçişler:

t_1 : Otomatın meşrubat haznesinin doldurulması

t_2 : Otomatın kullanıcıya meşrubat vermesi

t_3 : Kullanıcının otomata madeni para atması

t_4 : Atılan madeni paranın otomat tarafından reddedilmesi

t_5 : Atılan madeni paranın otomat tarafından kabul edilmesi

Şekil 3.21'deki PN grafı dikkatli incelenecek olursa t_1 geçişinin koşulsuz olarak tetiklendiği görülebilir. t_1 geçişi tetiklendiği zaman p_1 yerine bir adet jeton konulur. Böylelikle meşrubat otomatının haznesine bir adet meşrubat konmuş olur. İlk başta p_2 yerinde bir jeton bulunduğundan meşrubat otomatı hazır durumdadır ve kullanıcının para atmasını beklemektedir. Kullanıcı otomata para attığı zaman t_3 geçişi tetiklenir ve para otomatın madeni para haznesine düşer. Başka bir deyişle p_2 yerindeki jeton alınır ve p_3 yerine konulur. Burada atılan para otomat tarafından kontrol edilir. Eğer atılan para geçersiz ise otomat tarafından reddedilir ve kullanıcıdan tekrar para atması beklenir. Yani t_4 geçişi tetiklenerek p_3 yerindeki jeton alınır ve tekrar p_2 yerine konulur. Eğer atılan para geçerli ise otomat tarafından kabul edilir ve otomat kullanıcıya meşrubat vermeye hazırlanır. Yani t_5 geçişi tetiklenerek p_3 yerindeki jeton alınır ve p_4 yerine konulur. Son olarak otomatın kullanıcıya meşrubat verebilmesi için hem meşrubat haznesinin dolu olması hem de atılan parayı önceden kabul etmiş olması gerekir. Yani t_2 geçişinin tetiklenebilmesi için hem p_1 hem de p_4 yerinde birer jeton bulunmalıdır. Bu durumda t_2 geçişi tetiklenir ve otomat kullanıcıya meşrubat vermiş olur. Daha sonra tekrar başlangıç durumuna geri dönülür ve meşrubat otomatı hazır durumda beklemeye başlar.

4. PETRİ AĞLARININ ANALİZİ

PN'lerin sistemlerin modellenmesinde kullanılmasının bir önemli nedeni ise sağladığı güçlü analiz imkânlarıdır. PN grafı analiz edilerek sistemlerin doğru bir şekilde modellenmiş olup olmadığı kontrol edilebilir. Bu amaçla modellenen sistemlerin davranışları derinlemesine incelenebilir ve sistemlerde ortaya çıkan problemlerin nedenleri etkin bir şekilde araştırılabilir.

Bu bölümde öncelikle PN'lerin davranış özellikleri ele alınacak ve bu özelliklere dayalı olarak geliştirilmiş analiz yöntemlerine değinilecektir. Son olarak PN'lerin sınıflandırılmasından bahsedilecektir.

4.1 Petri Ağlarının Davranış Özellikleri

4.1.1 Güvenlik Özelliği

Bir PN'de herhangi bir yerdeki jeton sayısı, tüm yeni durumlar için 1'i kesinlikle geçmiyorsa, bu yer güvenlidir (safeness). Eğer PN'deki bütün yerler için bu koşul sağlanıyorsa söz konusu PN de güvenlidir[3].

4.1.2 Sınırlılık Özelliği

Sınırlılık (boundedness) özelliği, güvenlik özelliğinin genelleştirilmiş bir halidir. PN grafındaki bir yerdeki jeton sayısı, tüm yeni durumlar için belirli bir k pozitif tamsayısını ($k > 1$) aşmıyorsa bu yer k – sınırlıdır. Görüldüğü gibi sınırlılık özelliği bir yerdeki jeton sayısının belirli bir değeri aşmaması koşuluna bağlıdır. Eğer PN'deki bütün yerler için bu koşul sağlanıyorsa söz konusu PN de k – sınırlıdır[3].

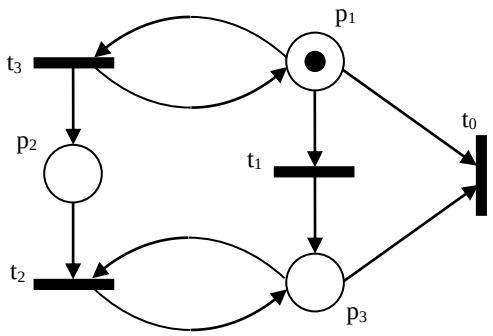
4.1.3 Canlılık Özelliği

Bir PN'de erişilen her bir yeni durumda eğer tetiklenebilir olan en az bir geçiş mevcutsa söz konusu PN canlıdır (liveness). Aksi takdirde eğer hiçbir geçişin tetiklenemeyeceği bir duruma gelinmişse ölümcül kilitlenme (deadlock) gerçekleşir. O halde canlı bir PN'de, ilk durumdan itibaren seçilen tetikleme sırası ne olursa olsun ölümcül kilitlenme hiçbir zaman gerçekleşmez[8].

Canlılık özelliği ideal sistemlerde aranan bir özelliktir. Ancak büyük sistemler için bu özelliğin var olup olmadığının araştırılması bir hayli zordur. Bu amaçla canlılık özelliği kısımlara bölünmüş ve değişik seviyeler için tanımlanmıştır. Bu seviyeler aşağıda belirtilmiştir[8]:

- **L0 – canlı:** PN'deki bir geçiş hiçbir zaman tetiklenemiyorsa bu geçiş ölü olarak tanımlanır ve L0 – canlıdır[8].
- **L1 – canlı:** PN'deki bir geçiş ilk durumdan itibaren en az bir kere tetiklenebiliyorsa bu geçiş L1 – canlıdır[8].
- **L2 – canlı:** PN'deki bir geçiş ilk durumdan itibaren belirli bir k pozitif tamsayısı (k > 1) kadar tetiklenebiliyorsa bu geçiş L2 – canlıdır[8].
- **L3 – canlı:** PN'deki bir geçiş herhangi bir tetikleme sırası için sonsuz sayıda tetiklenebiliyorsa bu geçiş L3 – canlıdır[8].
- **L4 – canlı:** PN'deki bir geçiş erişilen tüm yeni durumlar için L1 – canlı ise bu geçiş L4 – canlıdır[8].

Dikkat edilecek olursa L4 – canlılık özelliği en üst seviyeli canlılık özelliğidir ve L1, L2 ve L3 seviyeli canlılık özelliklerini kapsamaktadır. Şekil 4.1'de canlılık özelliğine ilişkin örnek bir PN grafı verilmiştir[8]:



Şekil 4.1 Canlılık özelliğine ilişkin örnek PN grafı

Şekil 4.1'deki PN grafında t_0 geçişi hiçbir zaman tetiklenemeyeceğinden ölü bir geçiştir, yani L0 – canlıdır. t_1 geçişi sadece bir kere tetiklenebilir ve bundan sonra asla tetiklenemez. Bu durumda t_1 geçişi L1 – canlıdır. t_2 geçişi bir kereden fazla sayıda tetiklenebilir; ancak sonsuz sayıda tetiklenemeyecektir. Bu nedenle t_2 geçişi L2 – canlıdır. Son olarak dikkatli incelenecek olursa t_3 geçişi sonsuz sayıda tetiklenebilmektedir ve bu nedenle L3 – canlıdır[8].

4.1.4 Sakınım Özelliği

Bir PN'de ilk durumdaki toplam jeton sayısı eğer erişilen her yeni durum için sabit kalıyorsa söz konusu PN ağı koruyucudur (conservative). Bu durum sakınım (conservation) özelliği olarak adlandırılır. Eğer jetonların sistemdeki bir kaynağı temsil ettiği düşünülürse bu önemli bir özelliktir; çünkü jeton sayısının korunması kaynakların kullanıldıktan sonra sisteme geri iade edildiğini ve boşa harcanmadığını gösterir[1, 3].

Sakınım özelliğinin genelleştirilmesi için $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_n]$ ağırlık vektörü ($\gamma_i \geq 0$) tanımlanır. Bu ağırlık vektörünün elemanları PN grafındaki yerlere $p_1, p_2, \ldots, p_n$ karşılık düşer. Sakınım özelliği bu vektörde belirtilmiş ağırlık değerlerine göre bağıl olarak tanımlanır. Bu ağırlık değerleri PN'deki yayların ağırlık değeriyle karıştırılmamalıdır. γ_i değeri genellikle pozitif bir tamsayı olarak seçilir ancak bu zorunlu değildir. Bu anlatılanlara dayalı olarak sakınım özelliği aşağıdaki gibi tanımlanabilir[1]:

x_0 ilk durumlu bir PN, erişilen tüm yeni durumlar için aşağıdaki koşul sağlandığı takdirde, $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_n]$ vektöründeki belirtilmiş ağırlık değerlerine göre koruyucudur[1]:

$$\sum_{i=1}^{n} \gamma_i x(p_i) = sabit \tag{4.1}$$

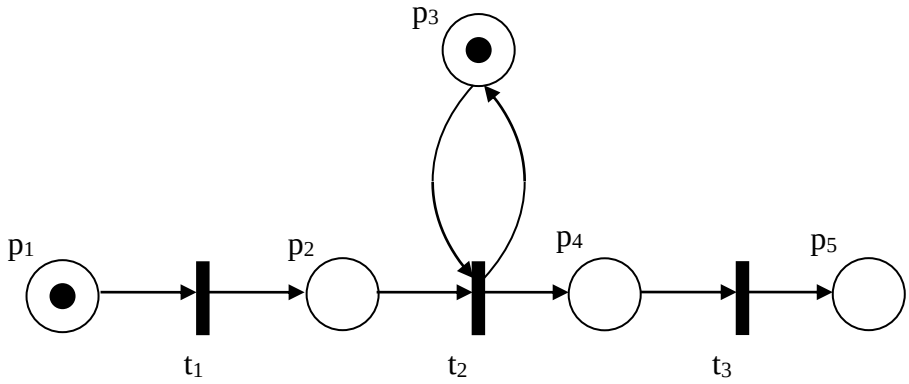
Bunun dışında bir de kesin (strict) koruyuculuk özelliği mevcuttur. Bir PN grafının kesin koruyucu olması için bu PN grafındaki her geçişin koruyucu olması gerekir. Bunun için PN grafındaki her bir geçiş için bu geçişe ilişkin toplam giriş sayısı ve toplam çıkış sayısı birbirine eşit olmalıdır. Bu koşul aşağıda belirtilmiştir[5]:

$$\sum_{p_i \in P} w(t_j, p_i) = \sum_{p_k \in P} w(p_k, t_j),\ p_i \in O(t_j)\ ve\ p_k \in I(t_j) \tag{4.2}$$

Aslında kesin koruyuculuk özelliği koruyuculuk özelliğinin bir alt kümesi niteliğindedir. Yani eğer bir PN grafı $\gamma = [1, 1, \ldots, 1]$ ağırlık vektörüne göre koruyucuysa kesin koruyucu olarak anılır. Bu koşul aşağıda belirtilmiştir[8]:

$$\sum_{i=1}^{n} x(p_i) = sabit \tag{4.3}$$

Bazı uygulamalar için bütün bir PN grafının kesin koruyucu olması pek pratik değildir. Bunun yerine PN grafında sadece belirli yerlerdeki toplam jeton sayısının her yeni durum için sabit kalması istenebilir. Bu durumda söz konusu yerlere değişmez (invariant) yerler denir[7]. Bu konuya daha sonra değinilecektir.



Şekil 4.2 Koruyucu bir PN grafı örneği

Şekil 4.2'de kesin koruyucu olan bir PN grafı verilmiştir. Dikkat edilecek olursa bu PN grafındaki her bir geçiş için o geçişe ait toplam giriş ve çıkış yerleri sayısı birbirine eşittir. Bu durum Tablo 4.1'de belirtilmiştir:

Tablo 4.1 Koruyucu PN örneğindeki geçişlerin giriş ve çıkış yeri sayısı

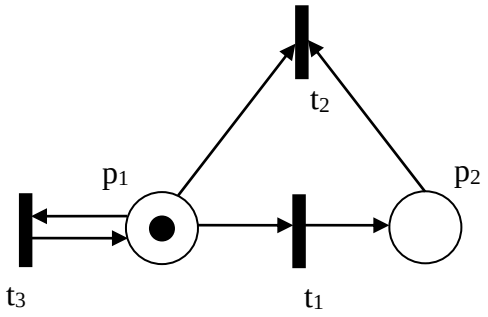
Geçiş Adı	Giriş Yeri Sayısı	Çıkış Yeri Sayısı
t_1	1 $\{p_1\}$	1 $\{p_2\}$
t_2	2 $\{p_2, p_3\}$	2 $\{p_3, p_4\}$
t_3	1 $\{p_4\}$	1 $\{p_5\}$

4.1.5 Kapsanabilirlik Özelliği

Bu özellik aslında erişilebilir durumlar kavramının genelleştirilmiş bir halidir. Kapsanabilirlik (coverability) özelliği bir PN grafındaki herhangi bir geçişin ileriki durumlarda tetiklenip tetiklenemeyeceğiyle ilgilenir. Hatırlanabileceği üzere bir geçişin tetiklenebilmesi için o geçişe ilişkin giriş yerlerinde en azından birer jeton bulunmalıdır. Şimdi herhangi bir $y = [y(p_1), y(p_2), \dots, y(p_n)]$ durumunun bir t_j geçişinin aktif hale gelmesi için yeter sayıda jeton içerdiğini ve PN grafının ilk başta x_0 durumunda olduğunu varsayalım. Eğer PN grafı x_0 durumundan, her $i = 1, \dots, n$ için $x(p_i) \geq y(p_i)$ olacak şekilde bir x durumuna erişebiliyorsa x durumu y durumunu kapsıyor ya da y durumu kapsanabilirdir (coverable) denir[1].

4.1.6 Süreklilik Özelliği

Bir PN'de eğer herhangi iki geçişten birinin tetiklenmesi diğer geçişin tetiklenmesini engellemiyor ise bu PN süreklilik (persistence) özelliğine sahiptir. Süreklilik özelliğine sahip bir PN'de eğer herhangi bir geçiş aktif hale gelmişse bu geçiş tetikleninceye kadar aktif halde kalır. Başka bir deyişle herhangi bir geçişin tetiklenmesi aktif hale gelmiş başka bir geçişin pasif hale gelmesine neden olmaz[1].



Şekil 4.3 Süreklilik özelliği olmayan örnek bir PN grafı

Şekil 4.3'te süreklilik özelliği olmayan bir PN grafı verilmiştir. Bu PN grafında ilk durumda t_1 ve t_3 geçişi aktif durumdadır. Bu durumda t_1 geçişinin tetiklenmesi aktif hale gelmiş olan t_3 geçişinin pasif hale gelmesine neden olur ve süreklilik özelliğini ihlal eder[1].

4.2 Petri Ağlarının Analiz Yöntemleri

4.2.1 Matematiksel Analiz Yöntemleri

Bu analiz yöntemlerinde matematiksel yaklaşımlar kullanılır. Bu amaçla PN grafı matematiksel olarak ifade edilir ve elde edilen bu matematiksel ifadeler yorumlanarak PN grafının analizi yapılır.

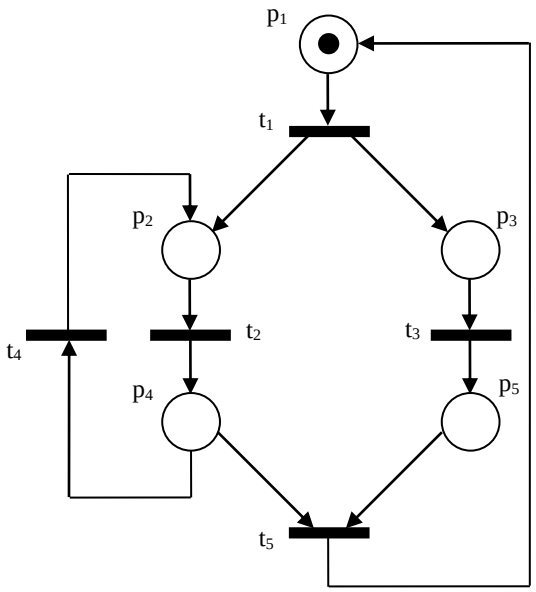
4.2.1.1 Durum Denklemleri

Durum denklemleri yaklaşımından daha önce Bölüm 2.4.2'de bahsedilmişti. Bu yüzden burada bu yaklaşımın üzerinde daha fazla durulmayacak ve sadece bütünlüğü sağlamak bakımından bu yaklaşımdan kısaca bahsedilecektir.

Bu yaklaşımda PN grafı bir durum vektörü ve sistem matrisiyle modellenir. Tetiklenecek olan geçişler ise bir tetikleme vektörü yardımıyla modellenir. Bu matematiksel yaklaşım yardımıyla PN grafının dinamik davranışı kolaylıkla analiz edilebilir[8].

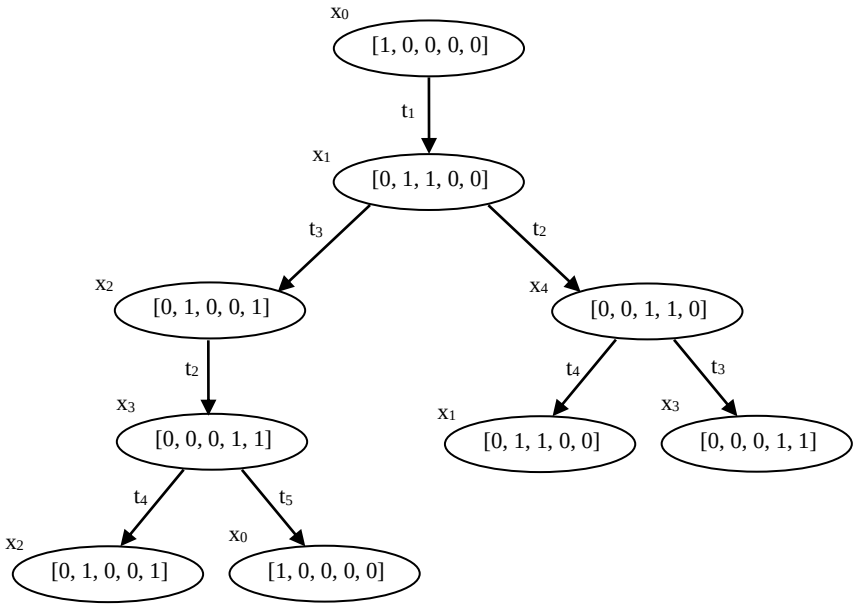
4.2.1.2 Kapsanabilirlik Ağacı

Kapsanabilirlik (coverability) ağacı yaklaşımıyla bir PN grafında x_0 ilk durumundan erişilebilen tüm durumlar elde edilebilir. Bu yaklaşımda elde edilen tüm durumlar bir ağaç yapısıyla gösterilir. Bu ağaç yapısında, düğümler erişilen yeni durumları ve bu düğümleri birbirine bağlayan yaylar ise PN grafının geçişlerini temsil eder[1, 8].



Şekil 4.4 Kapsanabilirlik ağacı için örnek PN grafı

Şekil 4.4'te kapsanabilirlik ağacı oluşturulacak x_0 = [1, 0, 0, 0, 0] ilk durumlu bir PN grafı verilmiştir. Bu PN grafının kapsanabilirlik ağacı Şekil 4.5'te verilmiştir:



Şekil 4.5 Örnek PN grafının kapsanabilirlik ağacı

Şekil 4.5'teki kapsanabilirlik ağacının oluşturulması için aşağıdaki maddeler, madde sonlarında parantezlerle belirtilmiş sıra numaralarına göre takip edilmiştir[3]:

1. Öncelikle $\mathbf{x_0 = [1, 0, 0, 0, 0]}$ ilk durumundan başlanır. Bu durumda sadece t_1 geçişi tetiklenebilir ve p_1 yerindeki jeton alınarak hem p_2 hem de p_3 yerine birer jeton konulur. Böylelikle $\mathbf{x_1 = [0, 1, 1, 0, 0]}$ durumuna erişilir. **(1)**
2. x_1 durumunda hem t_2 hem de t_3 geçişi tetiklenebilir:
    a. Öncelikle t_3 geçişi tetiklenecek olursa p_3 yerindeki jeton alınır ve p_5 yerine konulur. Böylelikle $\mathbf{x_2 = [0, 1, 0, 0, 1]}$ durumuna erişilir. **(2)**
    b. t_2 geçişi tetiklendiği takdirde ise p_2 yerindeki jeton alınır ve p_4 yerine konulur. Böylelikle $\mathbf{x_4 = [0, 0, 1, 1, 0]}$ durumuna erişilir. **(6)**
3. x_2 durumunda sadece t_2 geçişi tetiklenebilir ve p_2 yerindeki jeton alınarak p_4 yerine konulur. Böylelikle $\mathbf{x_3 = [0, 0, 0, 1, 1]}$ durumuna erişilir. **(3)**
4. x_3 durumunda hem t_4 hem de t_5 geçişi tetiklenebilir:
    a. Öncelikle t_4 geçişi tetiklenecek olursa p_4 yerindeki jeton alınır ve p_2 yerine konulur. Böylelikle x_2 durumuna geri dönülmüş olur. **(4)**
    b. t_5 geçişi tetiklendiği takdirde ise hem p_4 hem de p_5 yerinden birer jeton alınarak p_1 yerine konulur. Böylelikle x_0 başlangıç durumuna geri dönülmüş olur. **(5)**
5. x_4 durumunda hem t_3 hem de t_4 geçişi tetiklenebilir:
    a. Öncelikle t_4 geçişi tetiklenecek olursa p_4 yerindeki jeton alınır ve p_2 yerine konulur. Böylelikle x_1 durumuna geri dönülmüş olur. **(7)**
    b. t_3 geçişi tetiklendiği takdirde ise p_3 yerindeki jeton alınarak p_5 yerine konulur. Böylelikle x_3 durumuna geri dönülmüş olur. **(8)**

Bu yöntem kullanılarak olası tüm tetiklemeler gerçekleştirilir ve erişilebilecek bütün durumlar elde edilir. Böylelikle kapsanabilirlik ağacı adım adım oluşturulur[3].

Burada şunu belirtmekte fayda vardır: Tetiklemelerde istenen sıra seçilebilir. Sonuçta ortaya çıkacak kapsanabilirlik ağacı seçilen her tetikleme sırası için görünüş bakımından farklı olacaktır; fakat nitelik bakımından aynı kalacaktır.

Kapsanabilirlik ağacı, bir PN ile ilgili olarak aşağıdakilerin incelenmesini sağlar[8]:

- Eğer bir PN'nin kapsanabilirlik ağacı sonsuz bir şekilde büyümüyorsa bu PN sınırlıdır[8].
- Eğer bir PN'nin kapsanabilirlik ağacının düğümleri sadece 0 ve/veya 1 sayılarından oluşuyorsa bu PN güvenlidir[8].
- Eğer bir PN'nin kapsanabilirlik ağacının yaylarında bulunmayan bir geçiş mevcutsa bu geçiş ölüdür[8].

Bazı durumlarda kapsanabilirlik ağacı sonsuz bir şekilde büyüyebilir. Bu nedenle, kapsanabilirlik ağacının sınırlı tutulabilmesi için bir algoritma geliştirilmiştir. Bu algoritmada kapsanabilirlik ağacının sınırlı tutulabilmesi için bir **ω** sembolü kullanılır. Bu ω sembolü sonsuzluğu ifade eder. Algoritma aşağıda verilmiştir[8]:

Adım 1. İlk durum x_0 olarak etiketlenir ve yeni bir durum olarak kabul edilir[8].

Adım 2. Yeni durum(lar) elde edildiği sürece aşağıdaki işlemler tekrarlanır[8]:

Adım 2.1. Elde edilen yeni durum seçilir ve x olarak etiketlenir[8].

Adım 2.2. Eğer seçilen x durumu zaten önceden elde edilmiş bir durum ise başka bir yeni durum seçilir[8].

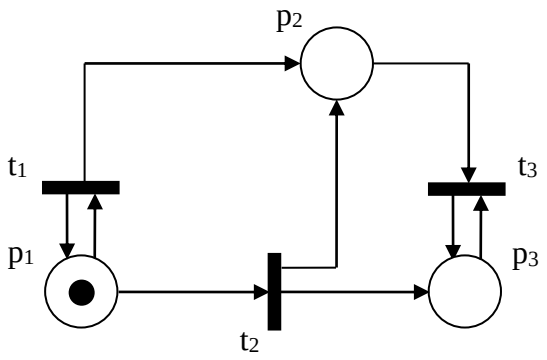
Adım 2.3. Eğer seçilen x durumunda hiçbir geçiş tetiklenemiyorsa bu durum ölü olarak kabul edilir ve başka bir yeni durum seçilir[8].

Adım 2.4. Eğer seçilen x durumunda tetiklenebilecek olan geçiş(ler) mevcutsa bu geçiş(ler) için aşağıdaki işlemler tekrarlanır[8]:

Adım 2.4.1. x durumunda seçilen geçiş tetiklenerek x' yeni durumu elde edilir[8].

Adım 2.4.2. Eğer kapsanabilirlik ağacında herhangi bir x'' durumu, x' durumu tarafından kapsanıyorsa ve bu x'' durumundan x' durumuna giden bir yol mevcutsa, PN grafındaki her p_i yeri için $x'(p_i) > x''(p_i)$ koşulunu sağlayan her $x'(p_i)$ yerine ω sembolü kullanılır[8].

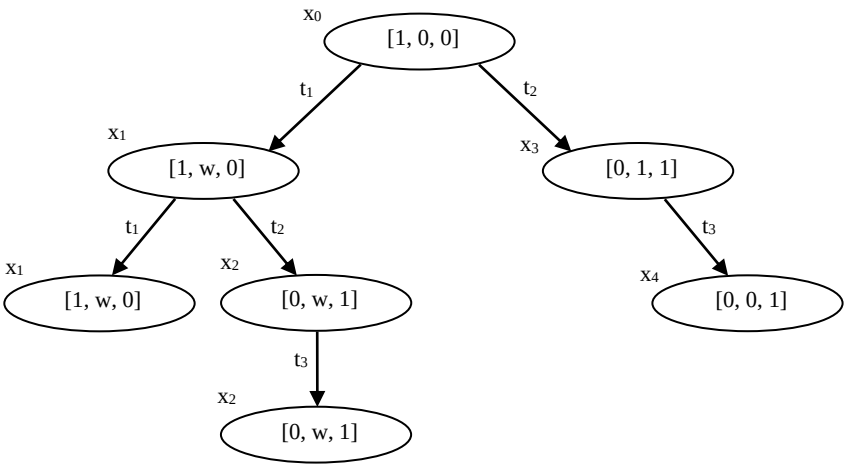
Adım 2.4.3. x' durumu kapsanabilirlik ağacına yeni bir düğüm olarak eklenir ve yeni bir durum olarak kabul edilir[8].



Şekil 4.6 Sonsuz kapsanabilirlik ağacı için örnek PN grafı

Verilen algoritma ışığında Şekil 4.6'da verilmiş olan örnek PN grafının sonsuz kapsanabilirlik ağacı, aşağıdaki maddeler, madde sonlarında parantezlerle belirtilmiş sıra numaralarına göre takip edilerek oluşturulabilir:

1. Öncelikle $\mathbf{x_0 = [1, 0, 0]}$ ilk durumundan başlanır. Bu durumda hem t_1 hem de t_2 geçişi tetiklenebilir:
    a. Öncelikle t_1 geçişi tetiklenecek olursa $\mathbf{x_1 = [1, 1, 0]}$ durumuna erişilir. Dikkat edilecek olursa x_1 durumu x_0 durumunu kapsamaktadır ve algoritmanın **2.4.2** adımı uygulanarak yeni durum $\mathbf{x_1 = [1, w, 0]}$ olarak elde edilir. **(1)**
    b. t_2 geçişi tetiklendiği takdirde ise $\mathbf{x_3 = [0, 1, 1]}$ durumuna erişilir. **(5)**
2. x_1 durumunda hem t_1 hem de t_2 geçişi tetiklenebilir:
    a. Öncelikle t_1 geçişi tetiklenecek olursa **[1, w+1, 0]** durumuna erişilir. Ancak w sembolü sonsuzluğu ifade ettiğinden **w + 1 = w** olacaktır. Sonuç olarak erişilen durum aslında $x_1 = [1, w, 0]$ durumudur. **(2)**
    b. t_2 geçişi tetiklendiği takdirde ise [0, w+1, 1] durumuna erişilir. Ancak w sembolü sonsuzluğu ifade ettiğinden **w + 1 = w** olacaktır. Sonuç olarak erişilen durum aslında $\mathbf{x_2 = [0, w, 1]}$ durumudur. **(3)**
3. x_2 durumunda sadece t_3 geçişi tetiklenebilir ve tetiklenme sonucu **[0, w–1, 1]** durumuna erişilir. Ancak w sembolü sonsuzluğu ifade ettiğinden **w – 1 = w** olacaktır. Sonuç olarak erişilen durum aslında $x_2 = [0, w, 1]$ durumudur. **(4)**
4. x_3 durumunda sadece t_3 geçişi tetiklenebilir ve $\mathbf{x_4 = [0, 0, 1]}$ durumuna erişilir. Dikkat edilecek olursa x_4 durumu ölü bir durumdur. **(6)**



Şekil 4.7 Örnek PN grafının sonsuz kapsanabilirlik ağacı

4.2.1.3 Değişmez Yerler ve Değişmez Geçişler

Bir PN'nin başlangıç durumundan tamamen bağımsız olup sadece yapısına bağlı olan özelliklerinden biri de ağ değişmezleridir (net invariant). Bir PN, ağ değişmezlerinden yararlanılarak dinamik özelliklerinden bağımsız bir şekilde analiz edilebilir. Değişmez yerler ve değişmez geçişler olmak üzere iki tip ağ değişmezi mevcuttur[2].

Değişmez yerler kümesi, PN'nin içinde bulunduğu her durum için içerdiği toplam jeton sayısı sabit kalan yerlerden oluşur. Değişmez yerler, n – boyutlu bir **i** vektörü ile ifade edilir. Buradaki n değeri PN grafındaki yerlerin sayısını belirtir. Bu i vektörünün sıfırdan farklı olan elemanları değişmez yerler kümesine ait olan yerleri temsil eder. Değişmez yerler aşağıdaki gibi tanımlanmıştır[2]:

$$x'.i^T = x_0.i^T \tag{4.4}$$

Burada x_0, PN'nin ilk durumunu ve x' ise PN'nin içinde bulunduğu herhangi bir durumu göstermektedir. Denklem (4.4)'te değişmez yerlerdeki toplam jeton sayısının her bir durum için aynı kaldığı ve bu toplam jeton sayısının PN'nin ilk durumuna bağlı bir değer olduğu belirtilmektedir. Aşağıda verilen denklem çözülerek değişmez yerlere ilişkin i vektörü elde edilebilir[2]:

$$D.i^T = 0 \tag{4.5}$$

Burada D matrisi, m × n boyutlu sistem matrisidir. (4.5) denklemi çözülerek elde edilecek i vektörünün sıfırdan farklı olan elemanları değişmez yerleri verir[2].

(4.5) denklemi çözüldüğü takdirde elde edilecek i vektörünün gerçekten değişmez yerlere ilişkin vektör olacağı, durum denklemlerinden yola çıkılarak aşağıdaki gibi ispatlanabilir[2]:

$$x' = x + u.D$$

$$x'.i^T = (x + u.D).i^T$$

$$x'.i^T = x.i^T + u.\underbrace{D.i^T}_{0}$$

$$x'.i^T = x.i^T \tag{4.6}$$

Denklem (4.6)'da elde edilen koşulu sağlayan i vektörünün değişmez yerlere ilişkin vektör olduğu denklem (4.4)'te belirtilmişti. Değişmez yerler kontrol sistemlerinin sentez ve analizinde kullanılır[2].

Değişmez geçişler kümesi ise tetiklendiği takdirde PN grafını tetiklenmeden önceki durumuna tekrar geri getiren geçişlerden oluşur. Değişmez geçişler, m – boyutlu bir **j** vektörü ile ifade edilir. Buradaki m değeri PN grafındaki geçişlerin sayısını belirtir. Bu j vektörünün sıfırdan farklı olan elemanları değişmez geçişler kümesine ait olan geçişleri temsil eder. Aşağıda verilen denklem çözülerek değişmez geçişlere ilişkin j vektörü elde edilebilir[2]:

$$j.D = 0 \tag{4.7}$$

(4.7) denklemi çözüldüğü takdirde elde edilecek j vektörünün gerçekten değişmez geçişlere ilişkin vektör olacağı, durum denklemlerinden yola çıkılarak aşağıdaki gibi ispatlanabilir[2]:

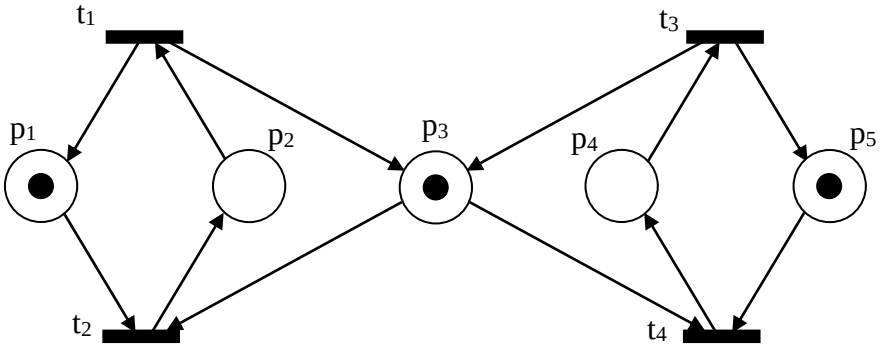
$$\begin{aligned} x_1 &= x_0 + u_0.D \\ x_2 &= x_1 + u_1.D = x_0 + (u_0 + u_1).D \\ &\text{M} \\ x_k &= x_0 + (u_0 + u_1 + \Lambda + u_{k-1}).D = x_0 + U.D \end{aligned} \tag{4.8}$$

Buradaki U vektörü, k adet tetikleme vektörünün toplamına eşittir. Denklem (4.8), PN'nin erişilen k. durumunun aslında seçilen tetikleme sırasına bağlı olmadığını göstermektedir. Böylelikle eğer U = j olursa, yani U vektörü değişmez geçiş vektörüne eşit olursa denklem (4.9)'daki durum ile karşılaşılır[2].

$$x_k = x_0 + \underbrace{j.D}_{0} = x_0 \tag{4.9}$$

Denklem (4.9)'a göre j vektörünün temsil ettiği geçişler başlangıç durumundan itibaren istenen sırada tetiklendiği takdirde yeniden başlangıç durumuna geri dönülmektedir. Yani j vektörü değişmez geçişlere ilişkin vektördür[2].

Örnek olarak Şekil 4.8'te verilen PN grafına ilişkin değişmez yerler ve değişmez geçişleri bulalım[7]:



Şekil 4.8 Değişmez yerler ve geçişler için örnek PN grafı

Şekil 4.8'deki PN grafına ait D sistem matrisi aşağıdaki gibidir:

$$D = \begin{bmatrix} 1 & -1 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 \\ 0 & 0 & -1 & 1 & -1 \end{bmatrix}$$

Değişmez yerlerin bulunabilmesi için aşağıdaki denklem çözülmelidir:

$$\begin{bmatrix} 1 & -1 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 \\ 0 & 0 & -1 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \end{bmatrix} = 0$$

Buradan aşağıdaki eşitlikler elde edilir:

$$i_1 - i_2 + i_3 = 0 \Rightarrow i_2 = i_1 + i_3$$
$$i_3 - i_4 + i_5 = 0 \Rightarrow i_4 = i_3 + i_5$$

Sonuçta;

$$i = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.i_1 + \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}.i_3 + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.i_5$$

olarak elde edilir. Buradan değişmez yerler kümesi aşağıdaki gibi belirlenebilir:

$$i' = (1, 1, 0, 0, 0) \rightarrow \{p_1, p_2\}$$

$$i'' = (0, 1, 1, 1, 0) \rightarrow \{p_2, p_3, p_4\}$$

$$i''' = (0, 0, 0, 1, 1) \rightarrow \{p_4, p_5\}$$

Değişmez yerler vektörü i; **i′**, **i″** ve **i‴** vektörlerinin lineer birleşimi olacaktır.

Değişmez geçişlerin bulunabilmesi içinse aşağıdaki denklem çözülmelidir:

$$\begin{bmatrix} j_1 & j_2 & j_3 & j_4 \end{bmatrix}.\begin{bmatrix} 1 & -1 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 \\ 0 & 0 & -1 & 1 & -1 \end{bmatrix} = 0$$

Buradan aşağıdaki eşitlikler elde edilir:

$$j_1 - j_2 = 0 \Rightarrow j_1 = j_2$$
$$j_3 - j_4 = 0 \Rightarrow j_3 = j_4$$

Sonuçta;

$$j = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}.j_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.j_4$$

olarak elde edilir. Buradan değişmez geçişler kümesi aşağıdaki gibi belirlenebilir:

$$j' = (1, 1, 0, 0) \rightarrow \{t_1, t_2\}$$

$$j'' = (0, 0, 1, 1) \rightarrow \{t_3, t_4\}$$

Değişmez geçişler vektörü j; **j′** ve **j″** vektörlerinin lineer birleşimi olacaktır.

4.2.2 Yapısal Analiz Yöntemleri

Bu analiz yöntemlerinde PN grafının yapısı incelenerek belirli ilişkiler kurulur. Kurulan bu ilişkilerden yararlanılarak PN grafının davranışları incelenir ve analizi yapılır.

4.2.2.1 Savaklar ve Kapanlar

Bir PN grafındaki yerler kümesi eğer bir kere işaretsiz duruma geldiği zaman daima işaretsiz olarak kalıyorsa savak (siphon), eğer bir kere işaretli duruma geldiği zaman daima işaretli olarak kalıyorsa kapan (trap) olarak adlandırılır. Bu yapılar PN'lerin canlılık analizinin yapılmasında ve ölümcül kilitlenme durumunun önlenmesinde kullanılır[2].

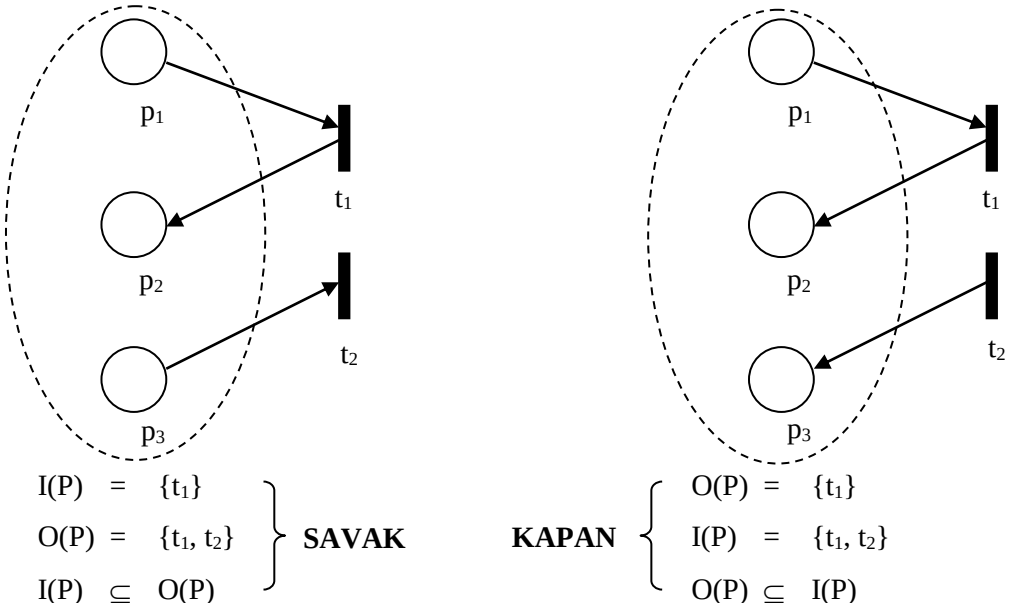
Bir P yerler kümesi aşağıdaki koşul sağlandığı takdirde savak olarak adlandırılır[2]:

$$I(P) \subseteq O(P) \tag{4.10}$$

Bir P yerler kümesi aşağıdaki koşul sağlandığı takdirde kapan olarak adlandırılır[2]:

$$O(P) \subseteq I(P) \tag{4.11}$$

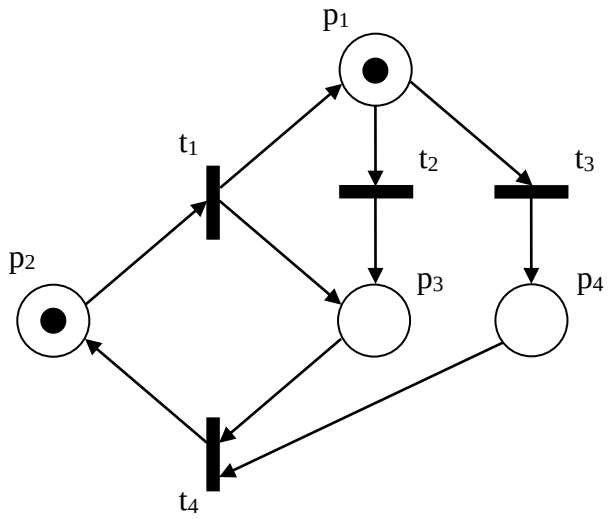
(4.10) ve (4.11) denklemlerinde I sembolü P yerler kümesine giren geçişleri, O sembolü ise P yerler kümesinden çıkan geçişleri temsil etmektedir[2].



Şekil 4.9 Örnek bir savak ve kapan

Şekil 4.9'da örnek bir savak ve kapan verilmiştir. Eğer dikkat edilecek olursa savaktaki jeton sayısı t_1 geçişi tetiklendiğinde sabit kalmakta; ancak t_2 geçişinin tetiklenmesiyle bir azalmaktadır. Yani savaktan çıkan jeton savağa bir daha geri dönememektedir. Kapandaki jeton sayısı ise yine t_1 geçişi tetiklendiğinde sabit kalmakta; ancak t_2 geçişinin tetiklenmesiyle bir artmaktadır. Yani kapana gelen jeton kapandan bir daha geri çıkamamaktadır[8].

Eğer bir savak, başka savakların birleşimi olarak ifade edilemiyorsa temel (basic) savak olarak adlandırılır. Eğer bir savak, içinde başka bir savak barındırmıyorsa minimal savak olarak adlandırılır. Aynı şekilde eğer bir kapan, başka kapanların birleşimi olarak ifade edilemiyorsa temel kapan olarak adlandırılır. Eğer bir kapan, içinde başka bir kapan barındırmıyorsa minimal kapan olarak adlandırılır[8].



Şekil 4.10 Savak ve kapan içeren örnek bir PN grafı

Şekil 4.10'da verilen PN grafındaki savak ve kapanlar Tablo 4.2'de verilmiştir[8]:

Tablo 4.2 Örnek PN grafındaki savak ve kapanlar

Yerler Kümesi	**Geçişler**		**Savak**	**Kapan**
	Giriş	**Çıkış**		
$P_1 = \{p_1, p_2, p_3\}$	$\{t_1, t_2, t_4\}$	$\{t_1, t_2, t_3, t_4\}$	✓	
$P_2 = \{p_1, p_2, p_4\}$	$\{t_1, t_3, t_4\}$	$\{t_1, t_2, t_3, t_4\}$	✓	
$P_3 = \{p_1, p_2, p_3, p_4\}$	$\{t_1, t_2, t_3, t_4\}$	$\{t_1, t_2, t_3, t_4\}$	✓	✓
$P_4 = \{p_2, p_3\}$	$\{t_1, t_2, t_4\}$	$\{t_1, t_4\}$		✓
$P_5 = \{p_2, p_3, p_4\}$	$\{t_1, t_2, t_3, t_4\}$	$\{t_1, t_4\}$		✓

Tablo 4.2'de dikkat edilirse P_1 ve P_2'nin hem temel hem de minimal savak olduğu; P_3, P_4 ve P_5'in de temel kapan olduğu fakat P_3 ve P_5'in minimal kapan olmadığı anlaşılabilir[8].

4.3 Petri Ağlarının Sınıflandırılması

PN'ler yapısal özelliklerine göre birtakım sınıflara ayrılmıştır. Böylelikle PN'lerin analizi, ait oldukları sınıflara özel yaklaşımlar kullanılarak daha etkin bir şekilde yapılabilir. Bu sınıflardan en önemlileri aşağıda verilmiştir[2]:

- **Durum Otomatı (State Machine – SM) :** Bu PN'lerde tüm geçişlerin sadece tek bir giriş ve çıkış yeri mevcuttur.

  Durum otomatı sınıfında bulunan PN'lerde yerler önem kazanmaktadır. Bu PN'lerde bir yerde bulunan tek bir jeton birden fazla geçişi aktif duruma getirip çakışmaya neden olabilir[2].

- **İşaretli Graf (Marked Graph – MG) :** Bu PN'lerde tüm yerlerin sadece tek bir giriş ve çıkış geçişi mevcuttur.

  İşaretli graf sınıfında bulunan PN'lerde geçişler önem kazanmaktadır. Bu PN'lerde tek bir geçişin birden fazla giriş ve çıkış yeri olabilir. Bu sınıfa ait PN'ler senkronizasyona izin verir[2].

- **Serbest Seçim Ağları (Free Choice – FC) :** Bu PN'lerde bir p yerini t geçişine bağlayan her yay için aşağıdaki koşullardan biri sağlanır:

  - t geçişi, p yerine ait tek çıkış geçişidir (çakışma yok), veya
  - p yeri, t geçişine ait tek giriş yeridir (senkronizasyon yok).

  Durum otomatı ve işaretli graf sınıflarına giren bütün PN'ler aynı zamanda serbest seçim ağları sınıfına da girer. Durum otomatı sınıfına giren PN'lerde senkronizasyon söz konusu değilken işaretli graf sınıfına giren PN'lerde ise çakışma durumu söz konusu değildir. Serbest seçim ağları sınıfına giren PN'lerde hem senkronizasyon hem de çakışma durumları söz konusu olabilir. Serbest seçim ağları sınıfında bulunan PN'lerde eğer bir p yerine ait bir çıkış geçişi aktif hale gelmişse söz konusu p yerine ait tüm çıkış geçişleri aktif hale gelmiş olur[2].

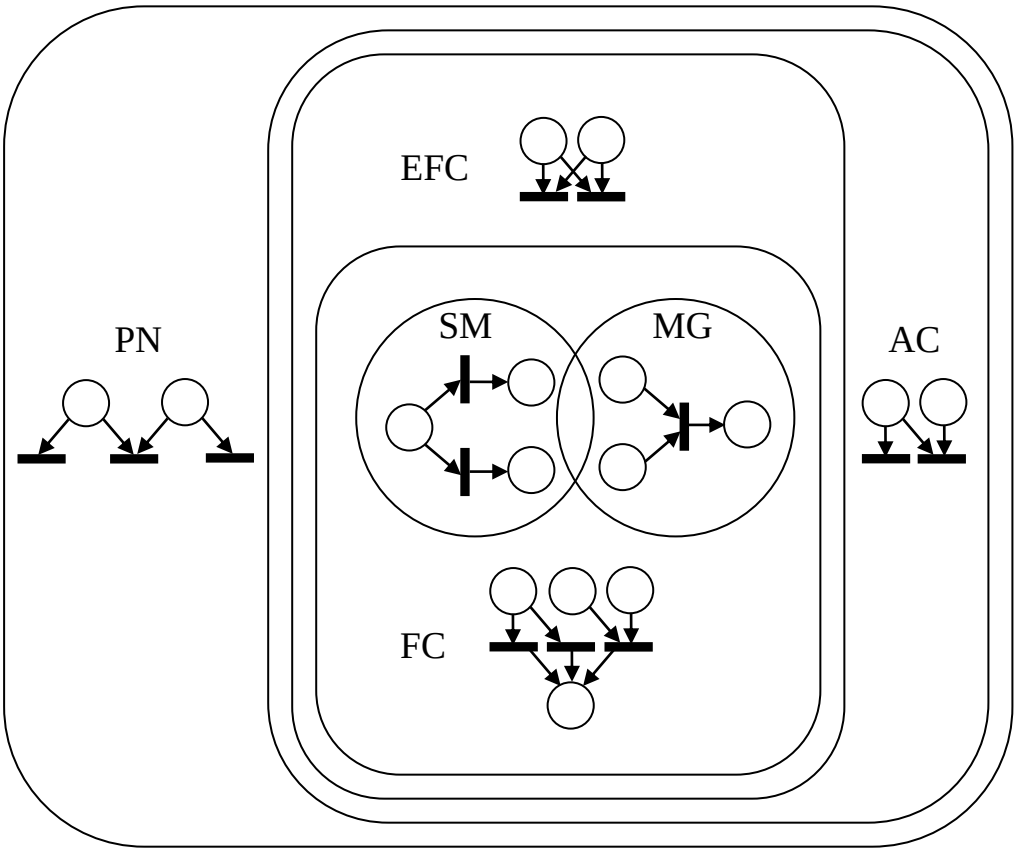
- **Genişletilmiş Serbest Seçim Ağları (Extended Free Choice – EFC) :** Bu PN'lerde bir p yerini t geçişine bağlayan her yay için, t geçişine ait bütün giriş yerlerinden p yerine ait bütün çıkış geçişlerine çizilmiş bir yay mevcuttur[2].

$$\text{Eğer } O(p_1) \cap O(p_2) \neq 0 \text{ ise } O(p_1) = O(p_2) \text{ olur.}$$

- **Asimetrik Seçim Ağları (Asymmetric Choice – AC) :** Bu PN'lerde bütün p_1 ve p_2 yer çiftleri için aşağıdaki koşul sağlanır[2]:

$$\text{Eğer } O(p_1) \cap O(p_2) \neq 0 \text{ ise } O(p_1) \subseteq O(p_2) \text{ veya } O(p_1) \supseteq O(p_2) \text{ olur.}$$

Şekil 4.11'de PN sınıflarının hiyerarşisi ve bu sınıflara ilişkin örnek PN'ler gösterilmiştir[2]:



Şekil 4.11 PN sınıfları arasındaki ilişkiler

5. PETRİ AĞLARI İLE SİSTEM KONTROLÜ

Bazen tasarlanan sistemler istenen davranışı gösteremez. Yani istenen bazı ölçütler (criteria) sistem tarafından karşılanamayabilir. Bu durumda söz konusu sistemin istenen davranışı gösterebilmesi için bir kontrolöre ihtiyaç duyulur. Bu kontrolör, sistemin önceden belirlenmiş kısıtlamaları (constraint) ihlal etmeyecek bir şekilde çalışmasını sağlar. Böylelikle sistemin istenen şekilde davranması sağlanmış olur.

Bir sistemin kontrol edilmesi, aslında bu sisteme ait PN'nin yasaklanmış durumlara erişmesinin engellenmesi şeklinde gerçeklenir. PN grafına sonradan yerler ve yaylar eklenerek bu PN'nin, önceden belirlenmiş bütün yasaklanmış durumlara olan erişimi engellenebilir. PN grafına sonradan eklenen bu yerler ve yaylar, sistemin istenen davranışı göstermesini sağlayacak kontrolöre karşılık gelir. Bu kontrolörün eklenmesi ile elde edilen PN, kontrollü PN (controlled PN) olarak anılır. Sonuçta elde edilen kontrollü PN, istenen davranışı gösteren sistemin modellenmiş haline karşılık gelir[1].

Bu bölümde anlatılan yöntemde kontrolör tasarımı, kontrol edilmek istenen PN'nin değişmez yerlerinden yola çıkılarak gerçeklenir. Bu kontrolör, kontrol edilmek istenen PN'nin geçişlerine bağlanmış yerlerden oluşur. Sonuçta elde edilen kontrollü PN, önceden belirlenmiş bir dizi kısıtlamayı sağlayan değişmez yerlere sahip olur. Bu yöntemde kontrolör, sadece matris işlemleri kullanılarak kolaylıkla elde edilebilir. Elde edilen kontrolörün büyüklüğü, sistemin sağlaması gereken kısıtlamaların sayısı ile doğru orantılı olur. Söz konusu kontrolör, sistemin belirlenmiş kısıtlamalara uymasını gerektirirken, doğrudan veya dolaylı olarak bu kısıtlamalar tarafından yasaklanmamış olan diğer durumlara erişmesine izin vermesi bakımından azami derecede izin verici (maximally permissive) olarak anılır. Bazen kontrol altına alınmak istenen sistemde kontrol edilemez (uncontrollable) ve gözlenemez (unobservable) geçişler mevcut olabilir. Bu gibi sistemler için yapılan kontrolör tasarımında bahsedilen yöntem temel alınarak daha farklı yollar izlenir[2].

5.1 Ağ Değişmezlerine Dayalı Kontrolör Tasarımı

Kontrol edilmek istenen sistemin sağlaması gereken kısıtlamalar genellikle lineer denklemler şeklinde ifade edilir[2]:

$$l.x_p^{\ T} \leq b \tag{5.1}$$

(5.1) denkleminde $\mathbf{x_p}$ PN grafının durum vektörünü, **l** bir ağırlık vektörünü ve **b** ise skaler bir değeri temsil etmektedir. Kısıtlamaların lineer denklemler şeklinde olması kontrolör tasarımını kolaylaştırmaktadır. Bu kısıtlamalar aslında PN'nin erişmemesi gereken yasaklanmış durumları belirtmektedir. Söz konusu kısıtlamalar dikkate alınarak gerçekleştirilen bir kontrolör, sistemin yasaklanmış durumlara olan erişimini engelleyerek istenen davranışı göstermesini sağlar. Kontrolör tasarımında böyle bir yaklaşımın kullanılmasının yararları aşağıdaki gibi sıralanabilir[2]:

1. Tasarım yöntemi sadece değişmez yerler kavramına dayalı olduğu için yoğun teorik bilgi ve detaylı analiz gerektirmez[2].
2. Tasarlanan kontrolör ve sonuçta elde edilen kontrollü sistem, karmaşık yapıda olmayan bir PN grafı ile temsil edilebilir[2]:
    a. Tasarlanan kontrolörün hem matematiksel hem de grafiksel olarak test edilebilmesi için birçok araç mevcuttur[2].
    b. Sisteme eklenen kontrolör, sistemin genel yapısını bozmaz[2].
3. Tasarım yönteminde çok basit nümerik özellikler kullanıldığından karmaşık sistemlerde ortaya çıkabilecek tasarım hataları kolaylıkla giderilebilir[2].

Kontrol edilmek istenen sistemin, n adet yere ve m adet geçişe sahip bir PN grafı ile modellendiğini varsayalım. Bu PN'ye süreç ağı (process net) adı verilir. Süreç ağına ilişkin sistem matrisi $\mathbf{D_p}$ ile verilsin. Bu sistemin istenen davranışı göstermediği düşünülürse bir kontrolöre ihtiyacımız var demektir. Kontrolör ağı (controller net), süreç ağının geçişlerinden ve birtakım yerlerden oluşan $\mathbf{D_c}$ sistem matrisine sahip olsun. Bu durumda kontrollü ağ (controlled net), orijinal süreç ağı ile kontrolör ağının birleşiminden oluşan **D** sistem matrisine sahip olur. Kontrollü ağ aynı zamanda kontrollü sistem (controlled system) veya kapalı çevrim sistemi (closed loop system) olarak da adlandırılır[2].

Sistemi kontrol etmedeki amacımız, bu sistemin (5.1) denkleminde belirtilmiş olan kısıtlamalara uymasını sağlamaktır. Örneğin $\mathbf{x_p(p_1) + x_p(p_2) \leq 1}$ şeklinde bir kısıtlamayı sistem üzerinde uygulamaya çalışalım. Böylelikle sistemin PN grafında, erişilen herhangi bir durum için p_1 ve p_2 yerlerinin sadece birinde en fazla tek bir jeton bulunmasını zorlamış oluruz. Bu aynı zamanda hem p_1 hem de p_2 yeri aynı anda işaretli olamaz anlamına da gelmektedir[2].

Bu eşitsizlik, $\mathbf{x_c(c)}$ **ilave** (slack) değişkeni eklenerek eşitlik haline dönüştürülebilir. Bu durumda kısıtlama $\mathbf{x_p(p_1) + x_p(p_2) + x_c(c) = 1}$ halini alır. Bu denklem aşağıdaki gibi genelleştirilebilir[2]:

$$l.x_p^{\ T} + x_c^{\ T} = b \tag{5.2}$$

İlave değişken negatif bir değer olamaz. Bu ilave değişken, aslında eşitliğin sağlanması için sisteme eklenen yeni bir yere karşılık gelir. PN grafına eklenen bu yeni yer kontrolör ağına ait bir yerdir[2].

Dikkat edilecek olursa, (5.2) denkleminde eklenen yeni yer ile birlikte PN grafındaki bazı yerlerin içerdiği toplam jeton sayısının her durumda belirli bir b değerini aşmaması gerektiği vurgulanmaktadır. Bunun sağlanabilmesi için de söz konusu yerlerin değişmez yerler olması gerekir. O halde kontrollü PN grafına ait değişmez yerler bulunarak istenen kontrolör tasarımı gerçeklenebilir[2].

(5.1) denkleminde belirtilen her kısıtlamaya bir ilave değişken karşılık gelir ve her bir ilave değişken sisteme eklenen yeni bir yer demektir. Bu nedenle kontrolör ağının içereceği yer sayısı, öngörülen kısıtlamaların sayısı ile doğru orantılıdır. Sistemi kontrol etmek için eklenen her yeni yer, kontrollü sistemin D sistem matrisine yeni bir sütun eklenmesine neden olur. O halde D matrisi, iki matrisin birleşimi olarak yazılabilir. Bu matrisler, kontrol edilmek istenen sistemin D_p matrisi ve kontrolörün D_c matrisidir. Kontrolörün yerlerini kontrol edilmek istenen sisteme bağlayan yaylar, (4.5)'te verilen değişmez yerler denkleminden yararlanılarak hesaplanabilir. Burada değişmez yerlere ilişkin vektör $i = [l_1\ l_2\ \ldots\ l_n\ 1]$ olarak tanımlanmıştır[2].

(5.1) denkleminde belirtilen şekilde ifade edilen kısıtlamalar tek bir kısıtlama halinde matrissel olarak aşağıdaki gibi ifade edilebilir[2]:

$$L.x_p^{\ T} \le b \tag{5.3}$$

Burada $\mathbf{x_p}$ sistemin durum vektörünü, **L** ağırlık değerlerinden oluşan ($n_c \times n$) boyutlu bir matrisi ve **b** ise ($n_c \times 1$) boyutlu bir vektörü temsil eder. Buradaki n_c kısıtlamaların sayısını belirtmektedir. İlave değişkenlerin eklenmesiyle (5.3)'teki eşitsizlik aşağıdaki eşitliğe dönüştürülebilir[2]:

$$L.x_p^{\ T} + x_c^{\ T} = b \tag{5.4}$$

Burada $\mathbf{x_c}$ kontrolöre ilişkin durum vektörünü ifade etmektedir[2].

$\mathbf{D_c}$ matrisi kontrolöre ait yerleri sistemin geçişlerine bağlayan yayları temsil eder. Kapalı çevrim sistemine ilişkin **D** sistem matrisi aşağıdaki gibi ifade edilir[2]:

$$D = \begin{bmatrix} D_p & D_c \end{bmatrix} \tag{5.5}$$

Kontrollü sisteme ilişkin **x** durum vektörü ve $\mathbf{x_0}$ ilk durum vektörü ise aşağıdaki gibi ifade edilebilir[2]:

$$x = \begin{bmatrix} x_p & x_c \end{bmatrix} \qquad x_0 = \begin{bmatrix} x_{p0} & x_{c0} \end{bmatrix} \tag{5.6}$$

(5.4)'te verilen denklem (4.5)'teki değişmez yerler denklemini sağlamalıdır[2]:

$$D.i^T = \begin{bmatrix} D_p & D_c \end{bmatrix} \begin{bmatrix} L^T \\ I \end{bmatrix} = 0$$

$$D_p.L^T + D_c = 0 \tag{5.7}$$

Burada I ($n_c \times n_c$) boyutlu bir birim matristir. Buradan $\mathbf{D_c}$ aşağıdaki gibi hesaplanabilir[2]:

$$D_c = -D_p.L^T \tag{5.8}$$

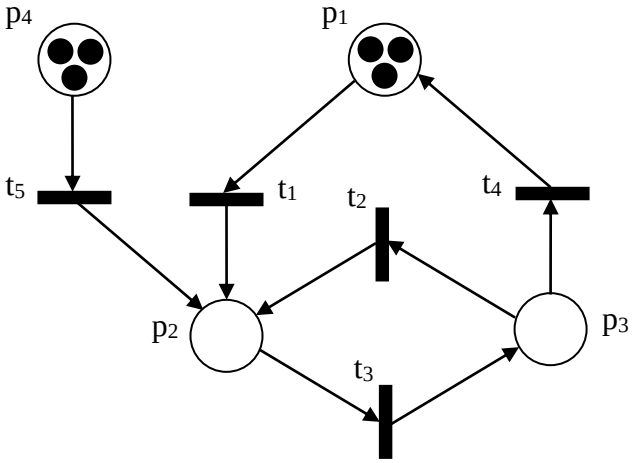
i = [L I] değişmez yerlere ilişkin vektör olduğundan (4.4)'te verilen eşitliği sağlar. Buradan $\mathbf{x_{c0}}$ aşağıdaki gibi hesaplanabilir[2]:

$$L.x_{p0}{}^{T} + x_{c0}{}^{T} = L.x_{p}{}^{T} + x_{c}{}^{T}$$

$$L.x_{p0}{}^{T} + x_{c0}{}^{T} = b$$

$$x_{c0} = \left(b - L.x_{p0}{}^{T}\right)^{T} \tag{5.9}$$

Şekil 5.1'de örnek bir PN grafı verilmiştir. Bu PN grafı kontrol edilmek istenen bir sisteme aittir[2].



Şekil 5.1 Kontrol edilmek istenen sistemin PN grafı

Bu PN grafının sistem matrisi $\mathbf{D_p}$ aşağıda verilmiştir:

$$D_p = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

Kontrol edilmek istenen sistemin başlangıç durumuna ilişkin durum vektörü aşağıdaki gibidir:

$$x_{p0} = \begin{bmatrix} 3 & 0 & 0 & 3 \end{bmatrix}$$

Bu sistemde uygulanmak istenen kısıtlama aşağıdaki gibi verilsin:

$$x_p(p_2) + x_p(p_3) \le 1 \tag{5.10}$$

Eğer (5.3)'te verilen matris notasyonu kullanılacak olursa;

$$L = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}$$

$$b = 1$$

olarak elde edilir. Verilen sistem istenen kısıtlamayı bu haliyle sağlayamayacaktır. Bu nedenle (5.10)'daki eşitsizliğe bir ilave değişken eklenir:

$$x_p(p_2) + x_p(p_3) + x_c(c) = 1 \tag{5.11}$$

Bu yeni eklenen $x_c(c)$ değişkeni kontrolöre ait bir c yerine denk gelmektedir. (5.11) denkleminden de görülebileceği gibi değişmez yerler vektörü i = [0 1 1 0 1] olarak belirlenebilir. Buradan $\mathbf{D_c}$ kontrolör sistem matrisi aşağıdaki gibi hesaplanabilir:

$$D_c = -D_p.L^T = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ -1 \end{bmatrix}$$

Kontrolörün $\mathbf{x_{c0}}$ ilk durum vektörü aşağıdaki gibi bulunur:

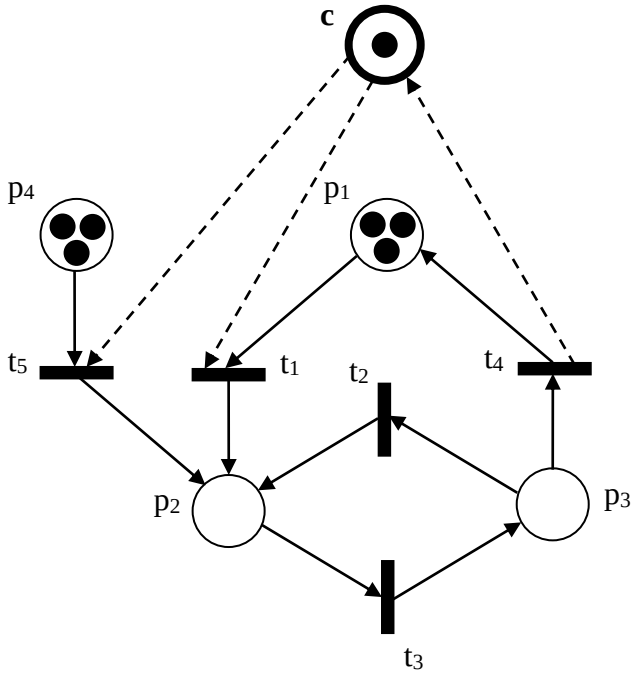
$$x_{c0} = \left(b - L.x_{p0}{}^T\right)^T = \left(1 - \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \\ 0 \\ 3 \end{bmatrix}\right)^T = 1 - 0 = 1$$

Buna göre kontrolöre ilişkin yer ilk durumda bir adet jeton içerecektir.

Elde edilen veriler ışığında kontrollü sistemin D sistem matrisi aşağıdaki gibi belirlenebilir:

$$D = \begin{bmatrix} D_p & D_c \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & -1 \end{bmatrix}$$

D sistem matrisindeki pozitif sayılar geçişlerden yerlere çizilen yayları, negatif sayılar ise yerlerden geçişlere çizilen yayları temsil etmektedir. Buradan yola çıkılarak kontrollü sistemin PN grafı Şekil 5.2'deki gibi çizilebilir[2]:



Şekil 5.2 Kontrol altına alınmış sistemin PN grafı

Şekil 5.2'de kontrolöre ilişkin yaylar kesikli çizgilerle ve kontrolöre ait yer ise kalın olarak belirtilmiştir. Sonuçta elde edilen kontrollü sistem (5.10)'da belirtilen kısıtlamaya uygun olarak davranacaktır[2].

5.2 Kontrol Edilemez ve Gözlenemez Geçişler

Bazı durumlarda kontrolörün belirli bir geçişe müdahale etmesine izin verilmez. Örneğin bir kimyasal reaksiyonu düşünelim. Bir kimyasal reaksiyon başladığı zaman bir daha durdurulmaması gerekir. Bu durumda kontrolör, kimyasal reaksiyonun gerçekleşmesine ilişkin geçişlere müdahale etmemelidir. İşte kontrolörün müdahale edemediği bu tür geçişlere kontrol edilemez (uncontrollable) geçiş denir. Bir PN'de kontrolör ağındaki yerlerden kontrol edilemez geçişlere yaylar çizilemez; fakat kontrol edilemez geçişlerden kontrolör ağındaki yerlere yaylar çizilebilir[2].

Bazen ise bir olayın gözlenmesi çok pahalı veya imkânsız olabilir. Örneğin bir sistemde bulunan algılayıcılarda (sensor) hata oluşması bu algılayıcılara ilişkin geçişlerin gözlenmesini imkânsız hale getirebilir. Bu şekilde eğer bir geçişin tetiklenmesi doğrudan takip edilemiyorsa veya izlenemiyorsa bu geçişe

gözlenemez (unobservable) geçiş adı verilir. Gözlenemez bir geçiş aynı zamanda kontrol edilemez bir geçiştir. Bu nedenle bir PN'de ne kontrolör ağındaki yerlerden gözlenemez geçişlere ne de gözlenemez geçişlerden kontrolör ağındaki yerlere yaylar çizilebilir[2].

Kontrol edilemez ve gözlenemez geçişlerin bulunduğu bir PN'nin kontrol altına alınabilmesi için belirlenmiş kısıtlamalar değiştirilerek başka kısıtlamalara dönüştürülür. Daha sonra kontrolör tasarımı, Bölüm 5.1'de anlatılan ağ değişmezlerine dayalı kontrol yöntemi kullanılarak bu değiştirilmiş kısıtlamalara uygun bir şekilde gerçekleştirilir. Burada kısıtlamaların dönüştürülmesi işleminin ayrıntılarına girilmeyecektir[2].

6. PETRİ AĞLARI İLE ÖRNEK BİR UYGULAMA

Bir önceki bölümde PN'lerin kontrol amaçlı olarak nasıl kullanılabileceğini öğrendik. Bu bölümde ise gerçek bir sistemin PN ile nasıl kontrol altına alınabileceğini göreceğiz. Bunun için ağ değişmezlerine dayalı kontrol yöntemini kullanacağız. Bunu yaparken kontrol altına alınacak sistemin kontrol edilemez ve gözlenemez geçişler bulundurmadığını kabul edeceğiz.

Kontrol edilecek örnek sistem SMC MAP – 205 MONTAJ SİSTEMİ (ASSEMBLY CELL) olarak belirlenmiştir. Öncelikle bu sistemin özellikleri tanıtılacaktır. Daha sonra bu sistemin çalışma mantığı ele alınacak ve sisteme ilişkin PN grafı elde edilecektir. Son olarak sistemde uygulanmak istenen kontrol amaçları (control objectives) belirlenip bu kontrol amaçlarına uygun bir kontrolör tasarımı PN ile gerçekleştirilecektir.

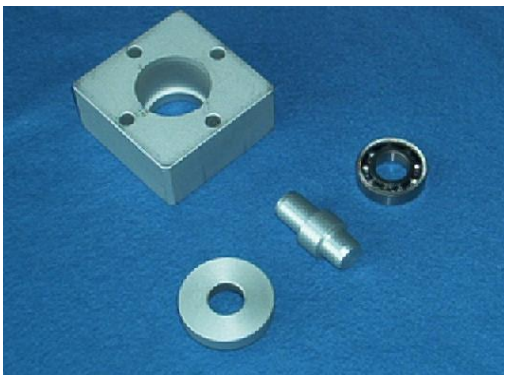
6.1 Kontrol Edilecek Sistemin Tanıtımı

Kontrol edilmek istenen örnek sistem SMC MAP – 205 MONTAJ SİSTEMİ'dir. Bu sistem dört adet parçayı birleştirerek dönel (rotary) bir mekanizmanın montaj işlemini gerçekleştirmektedir. Bu parçalar aşağıda belirtilmiştir[9]:

- Taban (Base)
- Bilye (Bearing)
- Şaft (Shaft)
- Kapak (Cover)

Söz konusu sistem PLC kontrolü altında montaj işlemini yerine getirmektedir. Parçaların yerlerinden alınıp monte edilmesi için sistemde dört adet hareketli mekanizma bulunmaktadır. Taban bir piston yardımıyla, bilye ve şaft birer robot kolu aracılığıyla ve kapak ise vakumlu bir düzenek yardımıyla alınmaktadır. Parçaların alınmasını sağlayan bu hareketli mekanizmalar pnömatik (pneumatic) olarak çalışmaktadır[9].

Şekil 6.1‘de sistemin montaj için kullandığı parçalar, Şekil 6.2’de ise bu parçalar birleştirilerek oluşturulan dönel mekanizma gösterilmektedir.



Şekil 6.1 Montaj işleminde kullanılan parçalar



Şekil 6.2 Montaj sonunda elde edilen dönel mekanizma

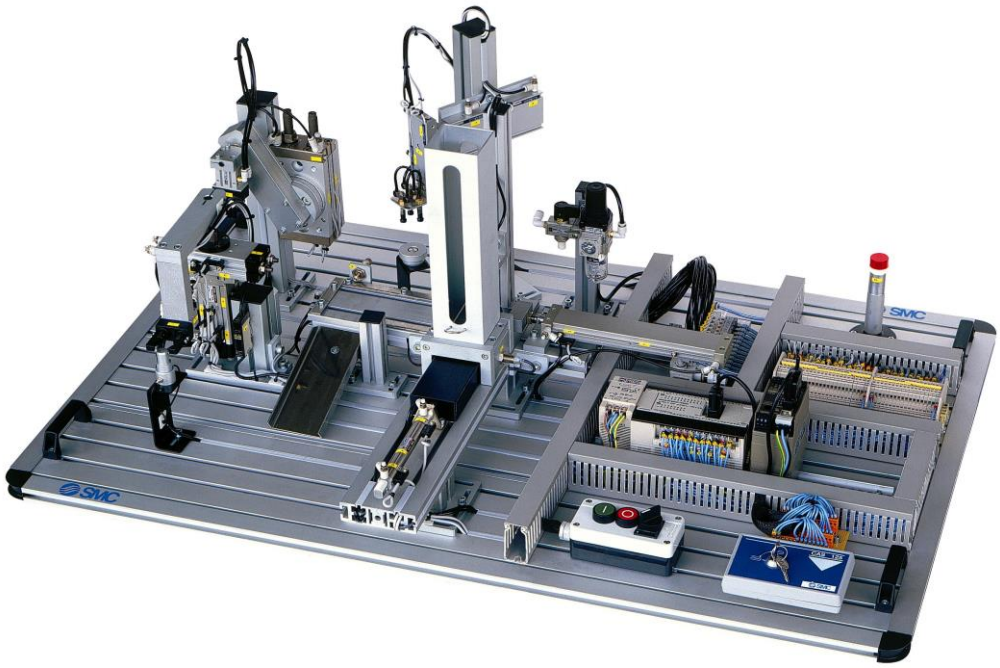
Sistemde parçaların varlığının algılanması için birtakım algılayıcılar kullanılmaktadır. Bu algılayıcıların tipi ve kullanılma nedenleri aşağıda belirtilmiştir[9]:

- Endüktif (Inductive) Algılayıcı: Tabanın varlığını algılamak için kullanılmaktadır[9].
- Fotoelektrik Hücre – Bariyer Tipi (Fotoelectric Cell – Barrier Type): Bilyenin varlığını algılamak için kullanılmaktadır[9].
- Fotoelektrik Hücre – Fiber Optik (Fotoelectric Cell - Fiber Optical): Şaftın varlığını algılamak için kullanılmaktadır[9].

6.2 Kontrol Edilecek Sistemin Çalışma Mantığı

Montaj sisteminin çalışma adımları kabaca aşağıdaki gibi özetlenebilir[9]:

- Tabanın algılanması
- Tabanın piston yardımıyla kontrol alanına itilmesi
- Tabanın kontrol edilmesi
    - Eğer taban hatalı konumlandırılmış ise tabanın diğer piston yardımıyla montaj alanına itilmesi ve montaj işleminin iptal edilmesi
    - Eğer taban doğru konumlandırılmış ise tabanın diğer piston yardımıyla montaj alanına itilmesi ve montaj işleminin başlaması
- Bilyenin algılanması, alınması ve tabana monte edilmesi
- Şaftın algılanması, alınması ve tabana monte edilmesi
- Kapağın algılanması, alınması ve tabana monte edilmesi
- Montaj işleminin tamamlanması ve yeni bir montaj işlemi için hazır duruma dönülmesi



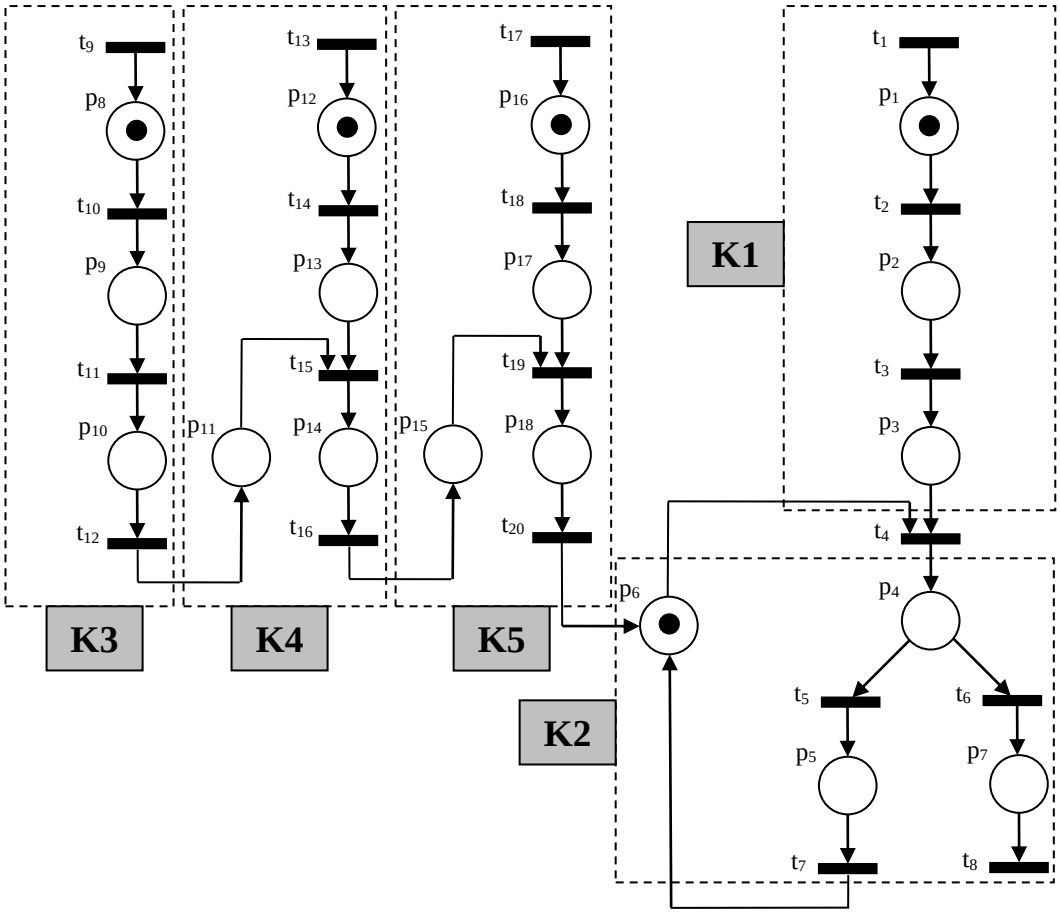
Şekil 6.3 SMC MAP – 205 Montaj Sistemi

Şekil 6.3'te SMC MAP – 205 MONTAJ SİSTEMİ gösterilmiştir. Bu montaj sisteminde kontrol alanı ve montaj alanı olmak üzere iki alan mevcuttur. Kontrol alanında tabanın doğru konumlandırılmış olup olmadığı kontrol edilir. Montaj alanında ise asıl montaj işlemi yapılır. Montaj işlemi şu sırayla yapılır[9]:

1. Bilye
2. Şaft
3. Kapak

Montaj alanında montaj işlemi devam ederken aynı anda yeni bir taban kontrol alanında kontrol edilebilir, fakat bu taban ancak montaj alanı boşaldıktan sonra montaj alanına itilebilir. Başka bir deyişle sistemde bir önceki montaj işlemi tamamlanmadan ya da iptal edilmeden yeni bir montaj işlemine başlanmamaktadır. Ayrıca montaj parçaları (Bilye, Şaft, Kapak) aynı anda alınabilmektedir[9].

Bu bilgiler ışığında sisteme ilişkin PN grafı Şekil 6.4'teki gibi oluşturulabilir:



Şekil 6.4 SMC MAP – 205 Montaj Sistemi'nin PN grafı

Sistemin daha iyi anlaşılabilmesi için Şekil 6.4'teki PN grafı **K1**, **K2**, **K3**, **K4** ve **K5** olmak üzere beş kısma ayrılmıştır. Bu kısımların sistemdeki işlevi Tablo 6.1'de verilmiştir. Sistemin PN grafındaki yerlerin açıklaması ise Tablo 6.2'de verilmiştir:

Tablo 6.1 Sisteme ait kısımların işlevleri

Kısım Adı	İşlevi
K1	Tabanın kontrol alanına alınması ve kontrol edilmesi
K2	Taban hatalı konumlandırıldıysa montaj işleminin iptal edilmesi Taban doğru konumlandırıldıysa montaj işleminin başlaması
K3	Bilyenin algılanması, alınması ve tabana monte edilmesi
K4	Şaftın algılanması, alınması ve tabana monte edilmesi
K5	Kapağın algılanması, alınması ve tabana monte edilmesi

Tablo 6.2 Sistemin PN grafındaki yerlere ilişkin açıklamalar

	Yer Adı	Açıklama
K1	p_1	Taban alınmak için hazır durumda.
	p_2	Taban kontrol alanına alındı.
	p_3	Taban kontrol alanında kontrol ediliyor.
K2	p_4	Taban montaj alanına alındı.
	p_5	Taban hatalı konumlandırılmış.
	p_6	Montaj alanı boş durumda.
	p_7	Taban doğru konumlandırılmış ve montaj için hazır.
K3	p_8	Bilye alınmak için hazır durumda.
	p_9	Bilye alındı.
	p_{10}	Bilye monte ediliyor.
K4	p_{11}	Bilye monte edildi.
	p_{12}	Şaft alınmak için hazır durumda.
	p_{13}	Şaft alındı.
	p_{14}	Şaft monte ediliyor.
K5	p_{15}	Şaft monte edildi.
	p_{16}	Kapak alınmak için hazır durumda.
	p_{17}	Kapak alındı.
	p_{18}	Kapak monte ediliyor ve montaj tamamlanıyor.

6.3 Sistemin Kontrol Altına Alınması

Montaj sisteminin kontrol altına alınabilmesi için öncelikle kontrol amaçlarının belirlenmesi gerekmektedir. Kontrol amaçlarının belirlenmesinden sonra sistemin sağlaması gereken kısıtlamalar tespit edilebilir. Daha sonra ağ değişmezlerine dayalı kontrol yöntemi kullanılarak sistemin bu kısıtlamalara uygun olarak davranması sağlanmış olur.

Sistemin aşağıdaki kontrol amaçlarını yerine getirmesi beklenmektedir:

1. Taban kontrol edilip montaj alanına girmeden montaj başlamayacaktır.
2. Montajı yapan robot kollarından aynı anda sadece biri montaj alanında çalışıyor olacaktır.
3. Montajı yapan robot kolları ancak kendilerine ait montaj işlemini tamamladıktan sonra yeni bir parçayı (Bilye, Şaft, Kapak) alabilecektir.
4. Yeni bir taban ancak bir önceki kontrol işlemi tamamlandıktan sonra kontrol alanına girebilecektir.

Sistem bu haliyle istenen kontrol amaçlarına uymamaktadır ve bu nedenle bir kontrolöre ihtiyaç duymaktadır. Uygun kontrolörün tasarlanabilmesi için öncelikle bu kontrol amaçlarının kısıtlamalar şeklinde ifade edilmesi gerekmektedir.

1 ve 2 no'lu kontrol amacı tek bir kısıtlama haline getirilebilir. Bu kontrol amaçlarının sağlanabilmesi için tabanın montaj işlemine başlanmasından itibaren montaj işlemi tamamlanana (dâhil) kadar geçtiği yerlerdeki toplam jeton sayısı biri aşmamalıdır. Bunun için aşağıdaki kısıtlama sağlanmalıdır:

$$x(p_4) + x(p_5) + x(p_6) + x(p_7) + x(p_{10}) + x(p_{11}) + x(p_{14}) + x(p_{15}) + x(p_{18}) \leq 1 \quad (6.1)$$

3 no'lu kontrol amacı parçaları monte eden üç farklı robot kolu olduğu için üç adet kısıtlamaya karşılık getirilebilir:

Bilyeyi alan robot kolu $\mathbf{p_9}$ yerinde bilyeyi almakta ve $\mathbf{p_{10}}$ yerinde montaj işlemini gerçekleştirmektedir. Bu nedenle bilyeyi alan robot kolu için aşağıdaki kısıtlama sağlanmalıdır:

$$x(p_9) + x(p_{10}) \leq 1 \quad (6.2)$$

Şaftı alan robot kolu $\mathbf{p_{13}}$ yerinde şaftı almakta ve $\mathbf{p_{14}}$ yerinde montaj işlemini gerçekleştirmektedir. Bu nedenle şaftı alan robot kolu için aşağıdaki kısıtlama sağlanmalıdır:

$$x(p_{13}) + x(p_{14}) \leq 1 \tag{6.3}$$

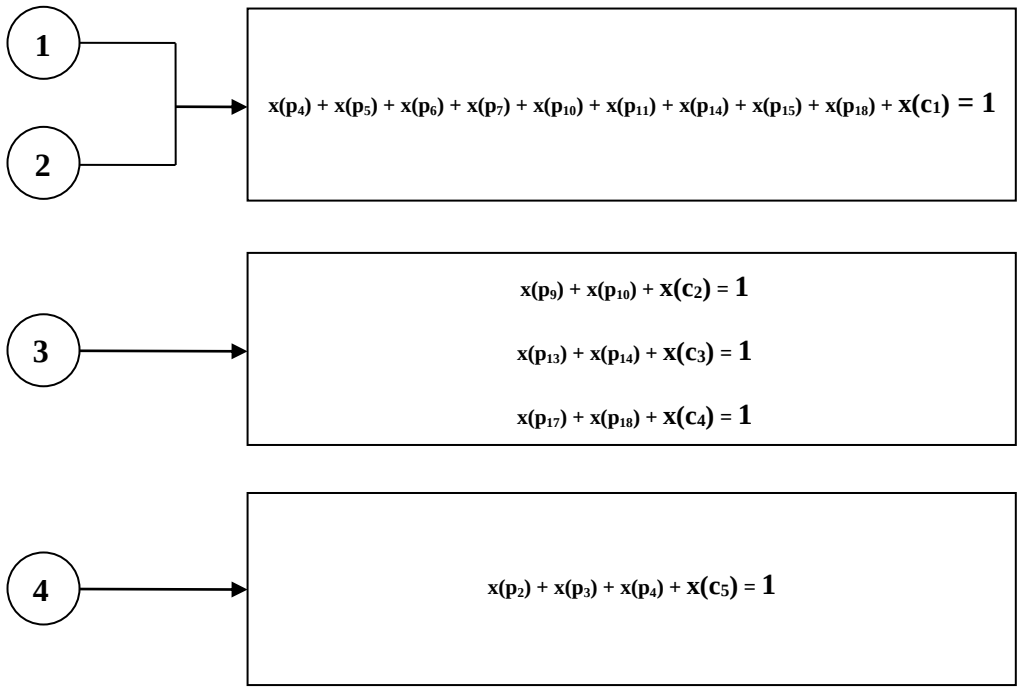
Kapağı alan robot kolu $\mathbf{p_{17}}$ yerinde kapağı almakta ve $\mathbf{p_{18}}$ yerinde montaj işlemini gerçekleştirmektedir. Bu nedenle kapağı alan robot kolu için aşağıdaki kısıtlama sağlanmalıdır:

$$x(p_{17}) + x(p_{18}) \leq 1 \tag{6.4}$$

4 no'lu kontrol amacı ise tek bir kısıtlama haline getirilebilir. Bunun için tabanın kontrol alanına girmesinden itibaren kontrol edilip montaj alanına alınmasına (dâhil) kadar olan yerlerdeki toplam jeton sayısı biri aşmamalıdır. Bunun için aşağıdaki kısıtlama sağlanmalıdır:

$$x(p_2) + x(p_3) + x(p_4) \leq 1 \tag{6.5}$$

Görüldüğü gibi sistemin uyması gereken toplam beş adet kısıtlama mevcuttur. Bu nedenle tasarlanacak olan kontrolör beş adet yere sahip olacaktır. Aşağıda her bir kontrol amacına karşılık düşen kısıtlama(lar) kontrolör yerleri eklenmiş şekilde genel olarak verilmiştir:



Kısıtlamalar aşağıdaki gibi matrissel şekilde ifade edilebilir:

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{L}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \\ p_{10} \\ p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{16} \\ p_{17} \\ p_{18} \end{bmatrix} \leq \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}}_{\mathbf{b}}$$

Montaj sistemine ilişkin $\mathbf{D_p}$ sistem matrisi aşağıdaki gibidir:

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}
t_1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_2	−1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_3	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_4	0	0	−1	1	0	−1	0	0	0	0	0	0	0	0	0	0	0	0
t_5	0	0	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
t_6	0	0	0	−1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
t_7	0	0	0	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	0
t_8	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0
t_9	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
t_{10}	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	0	0	0	0
t_{11}	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	0	0	0
t_{12}	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	0	0
t_{13}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
t_{14}	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0
t_{15}	0	0	0	0	0	0	0	0	0	0	−1	0	−1	1	0	0	0	0
t_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0
t_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
t_{18}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	1	0
t_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	−1	1
t_{20}	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	−1

Bu durumda kontrolöre ilişkin **D**$_c$ sistem matrisi aşağıdaki gibi bulunabilir:

$$D_c = -D_p.L^T$$

$$D_c = -\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0
\end{bmatrix}$$

Buradan **D**$_c$ sistem matrisi aşağıdaki gibi elde edilebilir:

$$D_c = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \\ t_7 \\ t_8 \\ t_9 \\ t_{10} \\ t_{11} \\ t_{12} \\ t_{13} \\ t_{14} \\ t_{15} \\ t_{16} \\ t_{17} \\ t_{18} \\ t_{19} \\ t_{20} \end{array}
\begin{array}{c}
\begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix}
\end{array}$$

Montaj sistemine ilişkin $\mathbf{x_{p0}}$ ilk durum vektörü aşağıdaki gibidir:

$$x_{p0} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Bu durumda kontrolöre ilişkin $\mathbf{x_{c0}}$ ilk durum vektörü aşağıdaki gibi bulunabilir:

$$x_{c0} = \left(b - L.{x_{p0}}^{T} \right)^{T}$$

$$x_{c0} = \left(\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}^{T}$$

Buradan $\mathbf{x_{c0}}$ ilk durum vektörü aşağıdaki gibi elde edilebilir:

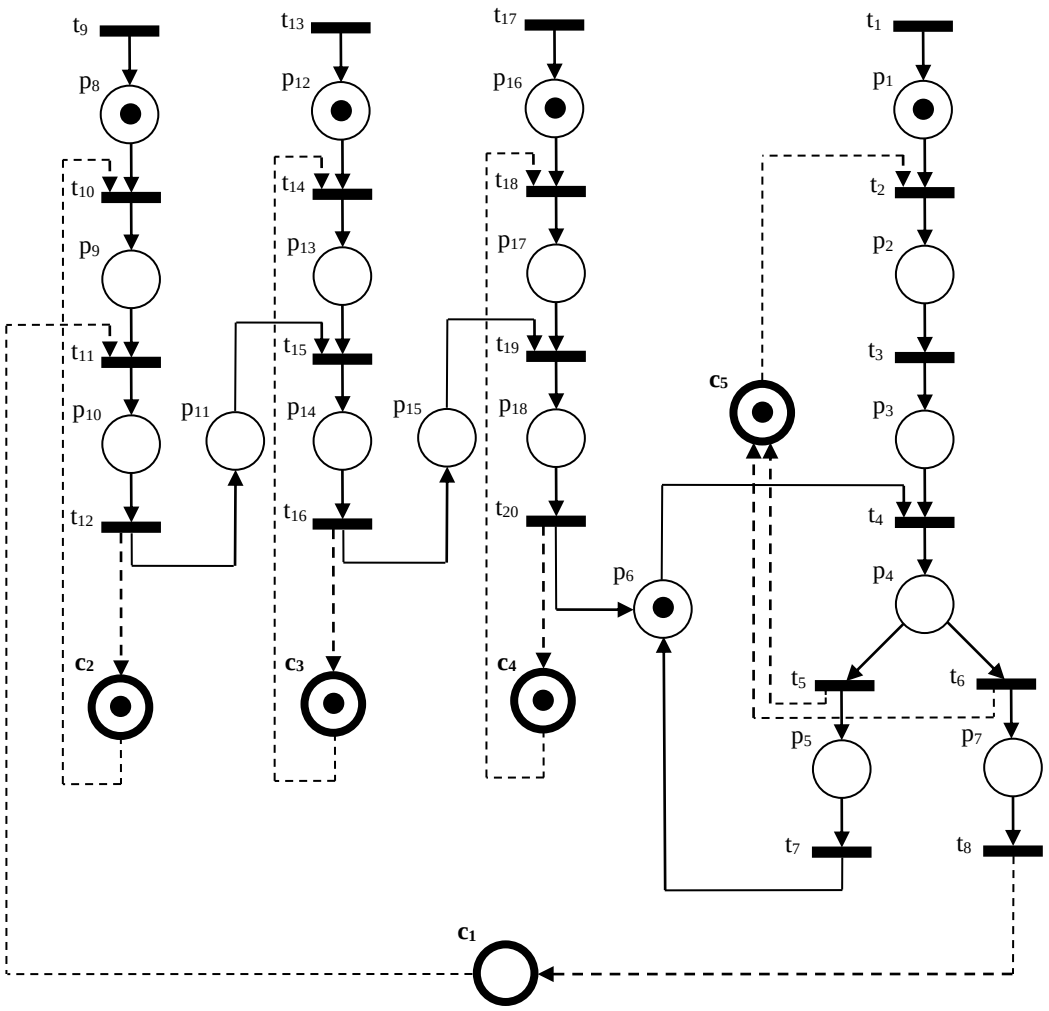
$$x_{c0} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Artık kontrolör ağına ait $\mathbf{D_c}$ sistem matrisinden yararlanarak her bir kontrolör yerine giren ve çıkan geçişleri, kontrolör ağına ait $\mathbf{x_{c0}}$ ilk durum vektöründen yararlanarak başlangıç durumunda her bir kontrolör yerine ilişkin jeton sayısını belirleyebiliriz. Bu bilgiler Tablo 6.3'te verilmiştir:

Tablo 6.3 Kontrolör yerlerine ilişkin bilgiler

Kontrolör Yeri	Giriş Geçişleri	Çıkış Geçişleri	Başlangıçtaki Jeton Sayısı
c_1	t_8	t_{11}	0
c_2	t_{12}	t_{10}	1
c_3	t_{16}	t_{14}	1
c_4	t_{20}	t_{18}	1
c_5	t_5, t_6	t_2	1

Tablo 6.3'te verilen bilgilerden yararlanılarak **sistem** ve **kontrolörden** oluşan **kontrollü sistem** (kontrol altına alınmış sistem) Şekil 6.5'teki gibi çizilebilir:



Şekil 6.5 Kontrol altına alınmış montaj sisteminin PN grafı

Şekil 6.5'te kontrolöre ilişkin yaylar kesikli çizgilerle ve kontrolöre ait yerler ise kalın olarak belirtilmiştir. Tablo 6.4'te kontrolör yerlerine ilişkin açıklamalar verilmiştir:

Tablo 6.4 Kontrolör yerlerine ilişkin açıklamalar

Yer Adı	Açıklama
c_1	Taban montaj alanına geldi. Montaj işlemine başlanabilir.
c_2	Bilyeyi alan robot kolu boş durumda. Bir sonraki bilye alınabilir.
c_3	Şaftı alan robot kolu boş durumda. Bir sonraki şaft alınabilir.
c_4	Kapağı alan robot kolu boş durumda. Bir sonraki kapak alınabilir.
c_5	Kontrol alanı boş durumda. Yeni bir taban kontrol edilebilir.

7. SONUÇ

Petri ağları sistemlerin modellenmesinde sıkça kullanılan bir araçtır. Sahip olduğu grafiksel özellikler matematiksel olarak ifade edilebildiği için Petri ağları ile modellenmiş sistemlerin analiz edilmesi de kolaydır. Ayrıca Petri ağları için birçok benzetim (simulation) paket programları da mevcuttur. Böylelikle kâğıt üzerinde olan tasarım sayısal ortama aktarılarak üzerinde çalışılması daha kolay hale getirilebilir.

Petri ağları bir graf olarak ifade edilmektedir. Bu graf sistemin statik yapısını modellerken grafta meydana gelen durum değişiklikleri sistemin dinamik davranışını modeller. Böylelikle sistemin statik ve dinamik yapısı ayrı ayrı incelenebilir. Petri ağları ile grafiksel olarak modellenen sistemlerin yapısı daha kolay anlaşılabilir.

Bu tez çalışmasında Petri ağlarının temel özelliklerinden bahsedilmiş, Petri ağlarının sistemlerin modellenmesinde nasıl kullanılabileceği gösterilmiş, Petri ağlarının analizinde kullanılan teknikler anlatılmış, Petri ağlarının sistem kontrolü alanında nasıl kullanılabileceğinden bahsedilmiş ve nihayetinde Petri ağlarının sunduğu kontrol teknikleri kullanılarak örnek bir sistemin nasıl kontrol altına alınabileceği adım adım gösterilmiştir.

Petri ağları kontrol alanında çok basit ama güçlü teknikler sunmaktadır. Bu teknikler fazla teorik bilgi gerektirmediğinden herhangi bir sisteme kolaylıkla uygulanabilmekte ve herhangi bir hata durumda bile geriye dönüş rahatlıkla yapılabilmektedir. Burada kontrol altına alınan SMC MAP – 205 MONTAJ SİSTEMİ için ağ değişmezlerine dayalı kontrolör tasarımı yöntemi kullanılmıştır. Petri ağındaki değişmez yerler sistemin sağlamasını istediğimiz kısıtlamalara denk gelmektedir. Ancak bu yöntem kontrol edilemez ve gözlenemez geçişler bulunduran sistemler için doğrudan uygulanamaz. Bu nedenle kontrol altına alınmak istenen örnek sistemin kontrol edilemez ve gözlenemez geçişler bulundurmadığı kabul edilmiştir. Bu tez ileriki çalışmalarda kontrol edilemez ve gözlenemez geçişler içeren sistemlerin Petri ağları ile nasıl kontrol altına alınabileceğini göstermek üzere genişletilebilir.

KAYNAKLAR

[1] **Cassandras, C.G. and Lafortune, S.,** 1999. Introduction to Discrete Event Systems, Kluwer Academic Publishers, MA.

[2] **Moody, J. O. and Antsaklis, P. J.,** 1998. Supervisory Control of Discrete Event Systems Using Petri Nets, Kluwer Academic Publishers, MA.

[3] **http://www.mfn.unipmn.it/~bobbio/biblio/papers/anno90/kluwerpetrinet.pdf**

[4] **http://www.ite.his.se/ite/utbildning/kurser/auc111/pdf/pn_informal.pdf**

[5] **http://students.cec.wustl.edu/~cs576/Notes/Petrinets.pdf**

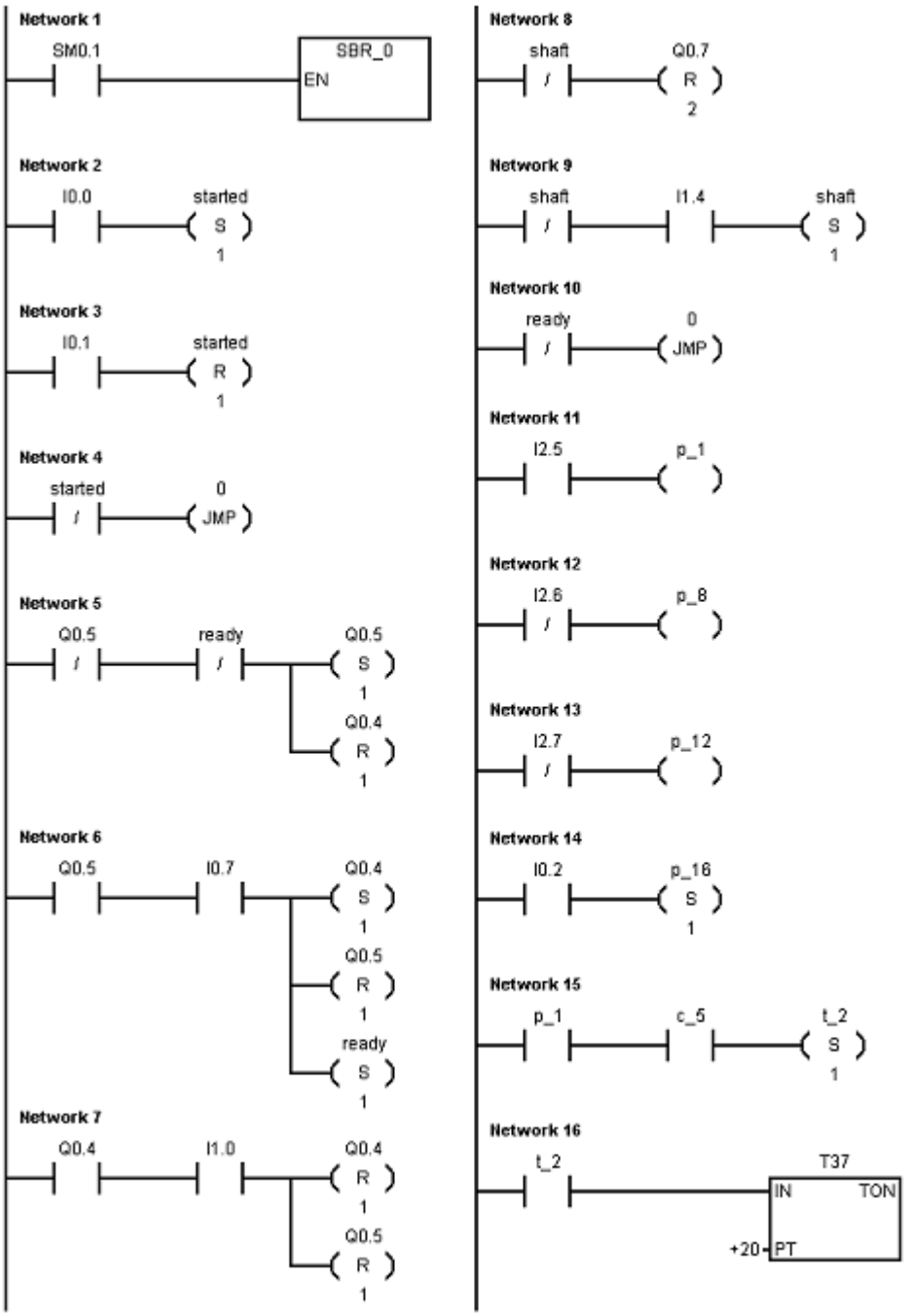
[6] **http://tmitwww.tm.tue.nl/staff/wvdaalst/ courses/wfm/mit2pn.pdf**

[7] **http://www.daimi.au.dk/PetriNets/introductions/pn2000_introtut.pdf**

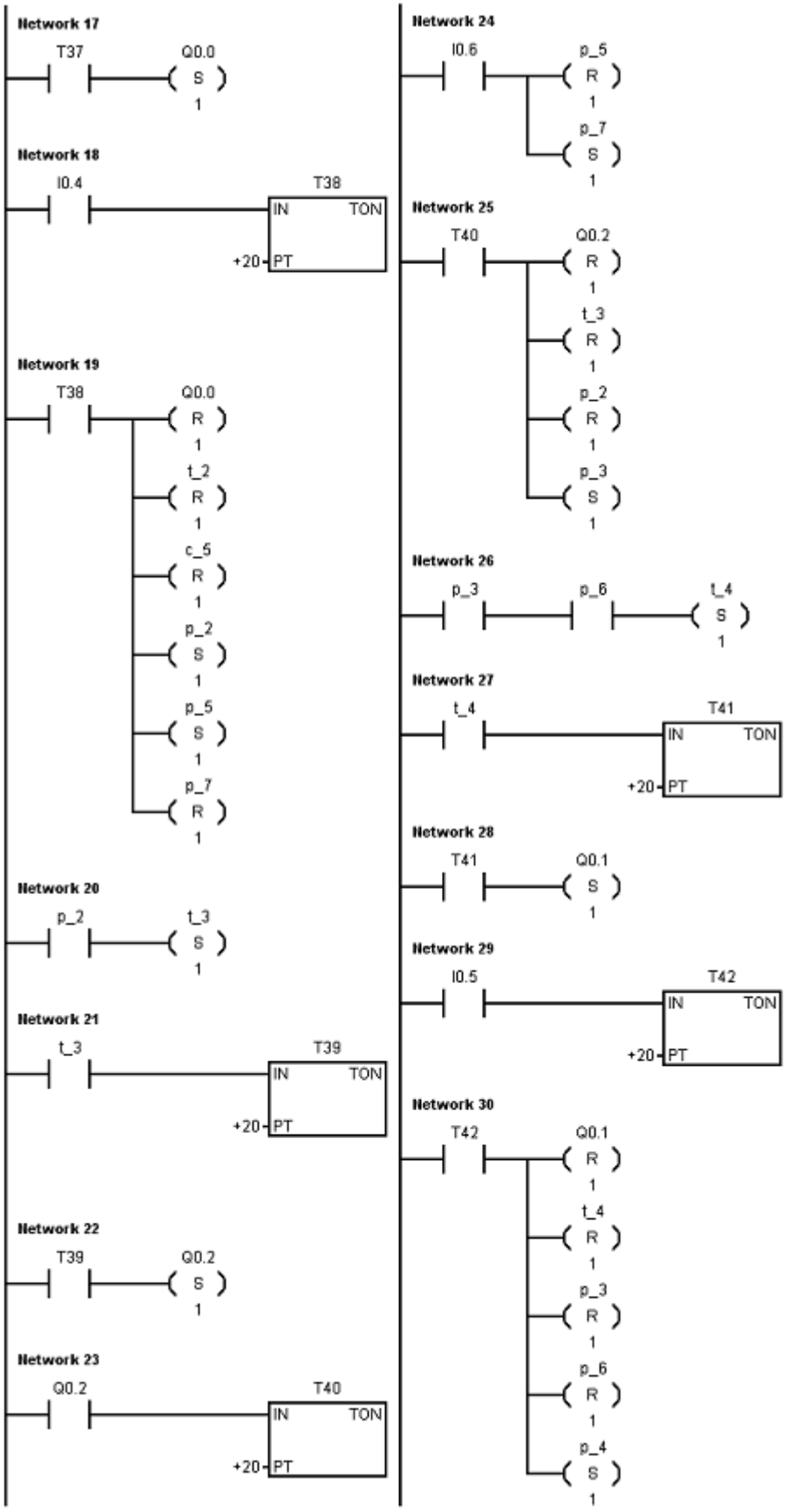
[8] **http://www.cs.unc.edu/~montek/teaching/spring-03/petrinets.pdf**

[9] **SMC International Training,** MAP 205: Assembly Mini – Cell User's Manual

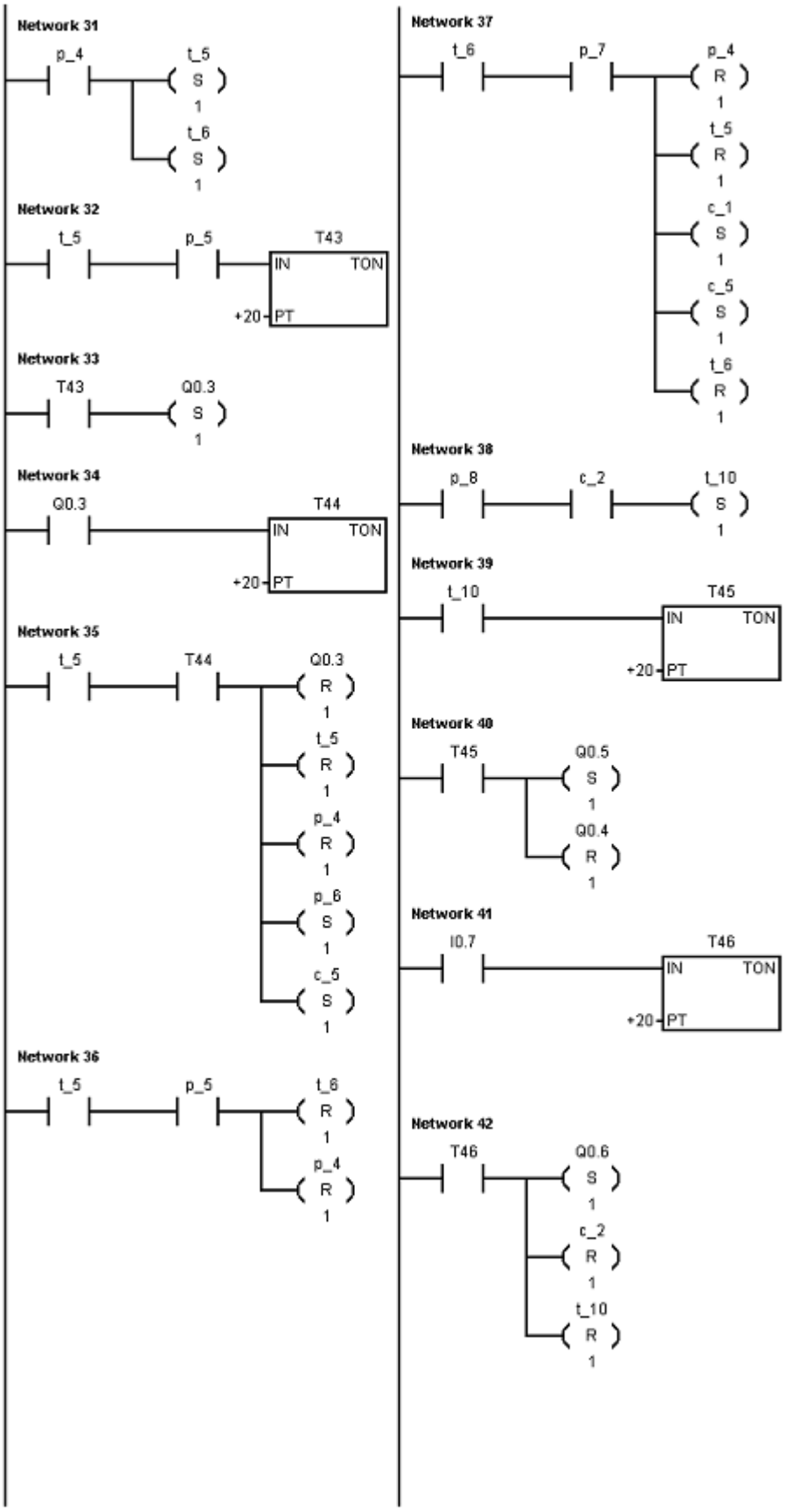
EK A. KONTROLLÜ SİSTEME AİT PLC PROGRAMI



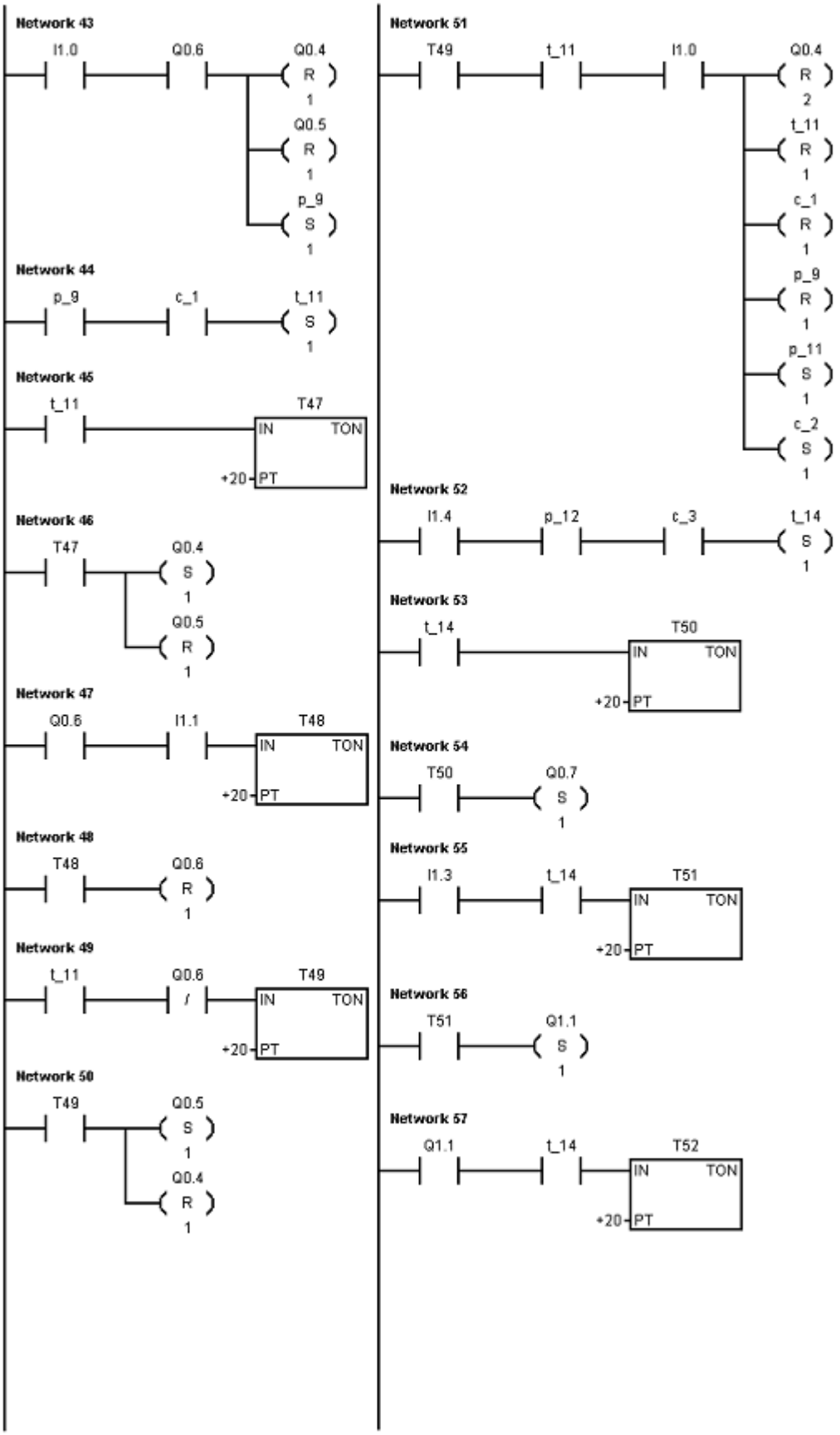
Şekil A.1 PLC ana programı (Networks 1 – 16)



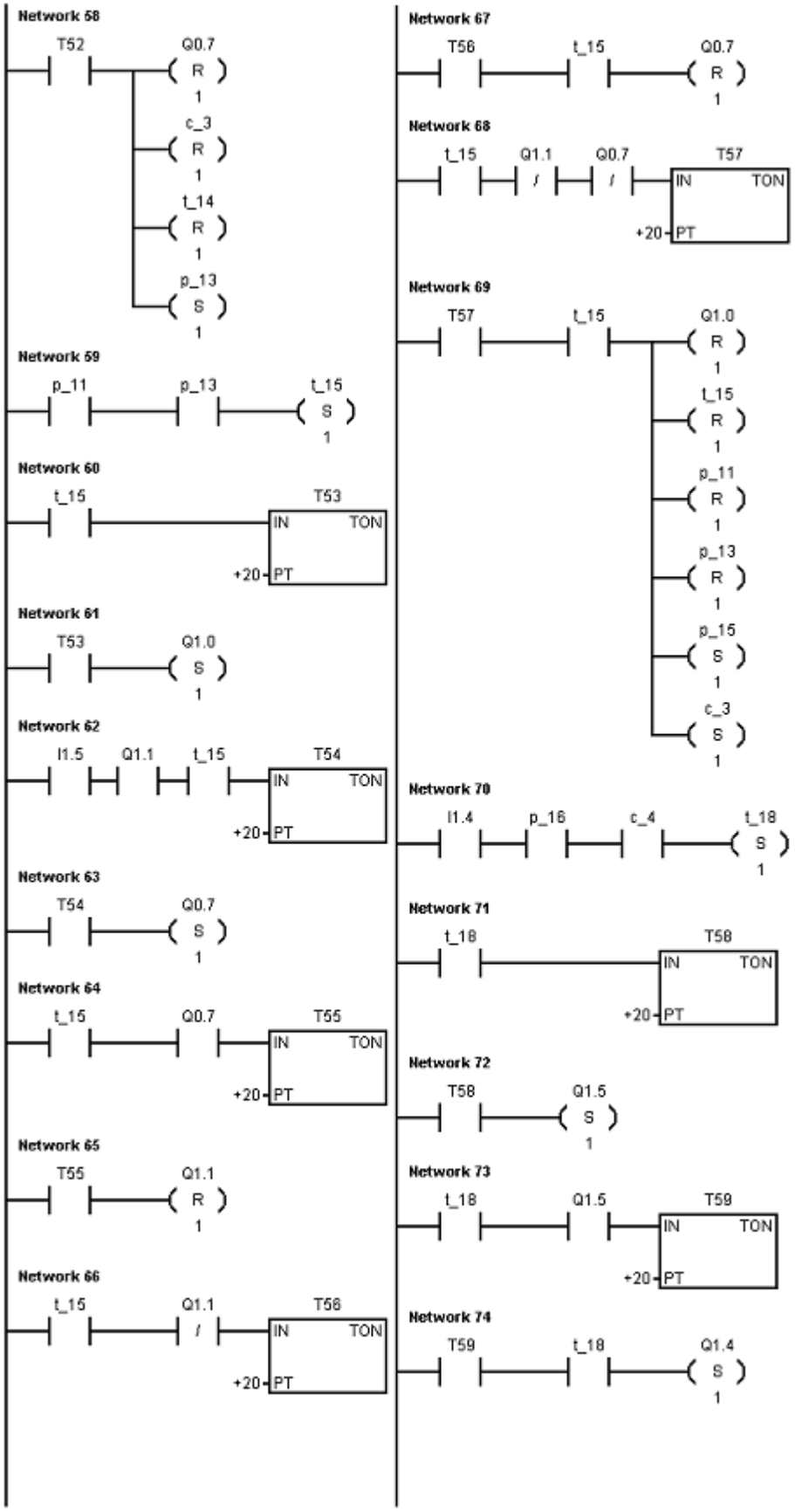
Şekil A.2 PLC ana programı (Networks 17 – 30)



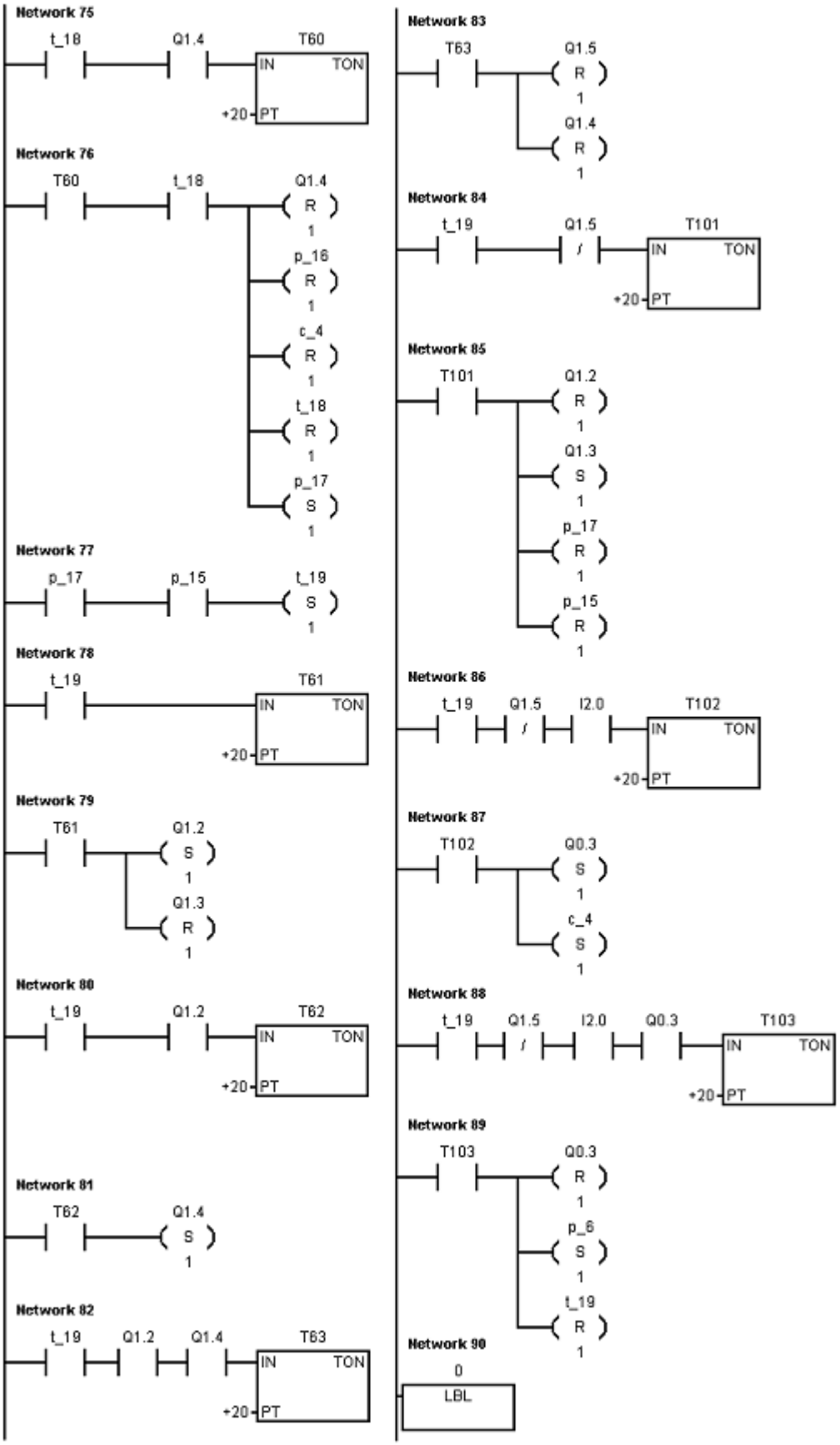
Şekil A.3 PLC ana programı (Networks 31 – 42)



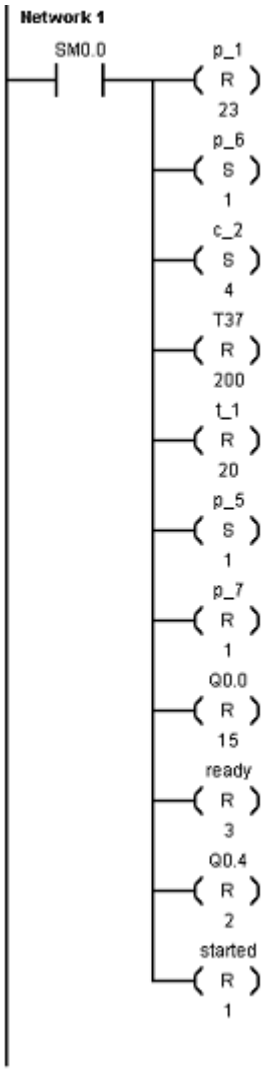
Şekil A.4 PLC ana programı (Networks 43 – 57)



Şekil A.5 PLC ana programı (Networks 58 – 74)



Şekil A.6 PLC ana programı (Networks 75 – 90)



Şekil A.7 PLC alt programı

ÖZGEÇMİŞ

BAŞKOCAGİL Cem

1979 yılında Konya'da doğdu. Liseyi Özel Seymen Fen Lisesi'nde bitirdikten sonra 1997 yılında İ.T.Ü Kontrol ve Bilgisayar Mühendisliği Bölümü'nü okumaya hak kazandı. 2001 yılında bu bölümden mezun olduktan sonra aynı yıl içinde İ.T.Ü Kontrol ve Otomasyon Mühendisliği Yüksek Lisans eğitimi almaya hak kazandı. Yine aynı yıl içinde TÜBİTAK – UEKAE'de Ağ Güvenliği konusu üzerinde çalışmak üzere işe girdi ve halen aynı şekilde çalışmalarına devam etmektedir.