AYA: HABERLEŞME UYDUSU FAYDALI YÜK SİSTEMİ AKILLI YEDEKLEME ALGORİTMASI

Şenol GÜLGÖNÜL*, Etem KÖKLÜKAYA, İsmail ERTÜRK*** ve Ahmet Y. TEŞNELİ****

*TÜRKSAT Uydu Haberleşme Kablo TV ve İşletme A.Ş., Ar-Ge ve Uydu Tasarım Direktörü, Ankara
**Sakarya Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği Bölümü, Sakarya
***Turgut Özal Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Ankara
sgulgonul@turksat.com.tr, ekaya@sakarya.edu.tr, ierturk@turgutozal.edu.tr, atesneli@sakarya.edu.tr

(Geliş/Received: 08.07.2013; Kabul/Accepted: 27.12.2013)

ÖZET

Haberleşme uydusu faydalı yük sisteminde, cihazlar birbirine çok uçlu anahtarlar ile bağlanmaktadır. Arıza oluşması durumunda, anahtarların pozisyonları değiştirilerek yedek cihazlara bağlantı sağlanır. Uydu işletmeciliğinde, uygun anahtar pozisyonlarının hesaplanmasını gerektiren bu zor problemin çözümü için ticari yazılımlar kullanılmaktadır. Bu tez kapsamında geliştirilen Akıllı Yedekleme Algoritması (AYA), faydalı yük sisteminin iki matris ile modellendiği, yedek cihazlara giden yolların özyinelemeli olarak bulunduğu yeni bir algoritmadır. AYA'nın açık mimarisi ile her türlü faydalı yük şebekesi tanımlanabilmekte ve karşılaşılabilecek yedekleme problemleri çözülebilmektedir. AYA'nın uygulama sonuçları ve etkinliği, TÜRKSAT-3A uydusunun işletmesinde kullanılan ICAREF ticari yazılımı ile doğrulanmıştır. AYA, yüksek maliyetli ticari yazılımlara bir alternatif sunmaktadır. AYA, yedek cihazlara giden yolları bulurken diğer bağlantılarda oluşacak kesinti sayısı ve sinyalin üzerinden geçebileceği anahtar sayısı kriterlerini de dikkate alarak kısa sürelerde uygun çözümleri üretebilmektedir.

Anahtar Kelimeler: Haberleşme uydusu, faydalı yük, yedekleme

SRRA: SMART REDUNDANCY RECONFIGURATION ALGORITHM FOR COMMUNICATION SATELLITE PAYLOAD

ABSTRACT

Redundancy is provided by using multiport switches which are connecting equipment in communication satellite payload systems. In case of a failure, signal path is re-routed by changing suitable switch positions. Smart Redundancy Reconfiguration Algorithm (SRRA) is a novel algorithm, able to model any payload system and to find paths recursively to redundant equipment. Results of the SRRA are verified by means of comparing them to those of the ICAREF commercial software which is currently being used in TURKSAT-3A satellite operations. SRRA can find all paths connecting given input(s) to any output(s) by taking number of interruptions and number of switches crossed constraints into account.

Key Words: Communication satellite, payload, redundancy

1. GİRİŞ (INTRODUCTION)

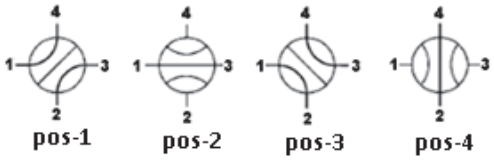
Haberleşme uydularında faydalı yük sistemi, yer istasyonlarından gönderilen sinyalleri alıp frekanslarını değiştirerek güçlendirdikten sonra, tekrar yeryüzüne gönderilmesini sağlar [1]. Faydalı yük sistemi, Şekil 1'de gösterildiği gibi antenler, alıcılar, frekans düşürücüler, giriş çoklayıcıları, güçlendiriciler ve çıkış çoklayıcılarından oluşan karmaşık bir sistemdir [2]. Yeryüzünden uyduya gönderilen sinyaller, alış antenleri ile alınmaktadır. Alıcılar, gürültüsü düşük bir şekilde alınan sinyalin güçlendirilmesini sağlamaktadır. Sinyalin frekansı, frekans düşürücüler ile değiştirilmektedir. Alınan geniş bant sinyaller, giriş çoklayıcıları ile çoğaltılıp filtreler ile dar bantlı (36 - 72 MHz vb) kanallara



Şekil 1. Haberleşme uydusu faydalı yük sistemi blok diyagramı (Communication Satellite Payload Diagram)

bölünmektedir. Her bir kanal güçlendirilmek üzere ilgili güçlendiricilere yönlendirilmektedir. Çıkış çoklayıcıları ise kanallara bölünmüş ve güçlendirilmiş sinyalleri tekrar birleştirilerek, yeryüzüne gönderilmek üzere veriş antenlerine ulaştırmaktadır.

Uydu faydalı yük sistemindeki cihazlar birbirlerine çok uçlu anahtarlar ile bağlı olup sinyalin özelliğine ve ihtiyaca göre göre farklı tiplerde anahtarlar kullanılmaktadır. Yaygın olarak kullanılan R-tip anahtarlar, Şekil 2'de gösterilen dört farklı pozisyonda, uçlar arasında bağlantı sağlayabilmektedir.



Şekil 2. R-tip anahtar pozisyonları (R-type switch positions)

Haberleşme uydularının herhangi bir arıza durumunda tamir edilmeleri mümkün olmadığından, uydu üzerindeki cihazların hemen hepsinin yedekleri bulunmaktadır. Uydu üzerinde çalışan kanallardaki kesintiler ticari cezalar doğurabilmektedir. Bu sebeple, herhangi bir arıza durumunda, yedekleme, çalışan kanallar üzerinde kesintiye sebep olmayacak şekilde, bu mümkün değilse kesinti sayısının ve sinyalin üzerinden geçtiği anahtar sayısının en az olması gibi kriterler dikkate alınarak yapılmaktadır.

Yüksek sayıdaki ekipman ve anahtarlardan dolayı, istenilen kriterleri sağlayan yedek yolları bulmanın basit bir yolu yoktur. Haberleşme uydularının faydalı yük sistemlerinde karşılaşılan yedekleme problemlerini çözmek üzere GMV firmasının SmartRings [3] ve Thales Alenia Space firmasının ICAREF [4] gibi çeşitli ticari yazılımlar kullanılmaktadır. SmartRings yazılımı, özyinelemeli bir yol bulma algoritması kullanmakta olup, bulunan çözümleri, anahtar sayısını, kesintileri göstermektedir. ICAREF yazılımında ise faydalı yük sistemi mimarisi, yazılımın içinde gömülüdür. Kesinti ve anahtar sayısı kriterleri değiştirilebilmektedir. Ticari yazılımlarda kullanılan algoritmalar bilinmediğinden, bu ürünler kara-kutu bir problem çözücü olarak çalışmaktadır. Yeni yedekleme kriterlerinin veya farklı cihazların eklenmesi için uygun esnekliğe sahip değildir [5].

Lorenzo Simone ve Ernesto Pensa'nın çalışmasında, yedekleme şebekesi netlist şeklinde tanımlanmıştır [6]. Bu çalışmada, her bir giriş ekipmanının gidebileceği yollardan bir çözüm ağacı oluşturulmuştur. Çözüm ağacı üzerinden özyinelemeli bir tarama algoritması ile giriş cihazının bağlanabileceği tüm yollar çıkartılmaktadır. Sonrasında ise üzerinden geçilen anahtar sayısının ve kesintinin en az olduğu yollar bulunan çözüm kümesinden seçilmektedir. A. Stathakis, G. Danoy, P. Bouvry ve G. Morelli tarafından yapılan çalışmada problemin çözümü için, Tamsayılı Doğrusal Programlama (TDP) yöntemi kullanılmıştır [5]. Verilen giriş ve çıkış cihazlarının birbirine en az anahtar pozisyonu değişimi ile bağlanması için TDP yönteminin kullanılabileceği gösterilmiş ve farklı şebeke mimarileri için hesaplama süreleri karşılaştırılmıştır. Aynı ekibin diğer çalışmasında ise, anahtar pozisyonlarındaki değişim kriteri yanında en kısa yol kriteri de eklenerek, iki kriterli bir optimizasyon sağlanmıştır [7]. Son çalışmalarında, mevcut kanallarda en az kesinti oluşturma kriteri de eklenmiştir [8]. Şebeke tasarımı boyutunda, faydalı yük sistemi yedekleme mimarisinde verilen giriş cihazlarının tümünün, belirli sayıdaki çıkış cihazının arızalanması durumunda yol bulabileceği, en az anahtar sayısına sahip en uygun şebekeyi oluşturmak üzere de çeşitli çalışmalar yapılmıştır [9]. Faydalı yük sistemi, sabit anahtar pozisyonlarında matematiksel graf olarak tanımlanabilir. Ancak anahtar pozisyonları değişken olduğundan graf ve şebeke akışı algoritmalarının direk kullanımı mümkün görünmemektedir [8]. Bununla birlikte graf teori faydalı yük sisteminin optimizasyonu ve yönlendirme problemleri için kullanılabilmektedir [10]. Faydalı yük ön tasarımı yapılan VX-SAT uydusunun yedekleme mimarisi için farklı arıza senaryolarındaki çözümler ve her bir çözüm için üzerinden geçilen anahtar sayısı ile kesinti sayıları sunulmuştur [11].

Akıllı Yedekleme Algoritmasının (AYA) tek giriş cihazı için uygulanması, makalenin yazarları tarafından bir bildiride sunulmuştur [2]. Bu yeni çalışmada, geliştirilen algoritmaya çoklu giriş cihazları için yedek yolların bulunması ile üzerinden geçilen anahtar sayısı ve kesinti sayısı kriterleri eklenmiştir. AYA'nın 30 anahtarlı TÜRKSAT-3A uydusu üzerinde karşılaşılabilecek gerçek problemleri çözebildiği de yine bu çalışmada gösterilmektedir.

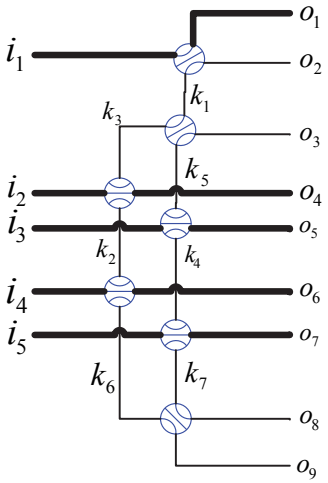
Problemin matematiksel modellemesi Bölüm 2.1'de açıklanmaktadır. Matematiksel model üzerinden AYA'nın tek giriş cihazı için yedek yolları bulması Bölüm 2.2'de, çoklu cihazlar için yolların bulunması Bölüm 2.3'de, kesinti ve anahtar sayısı kriterlerinin uygulanması ise Bölüm 2.4'de yer almaktadır. Uygulama sonuçları Bölüm 3'de sunulmuş olup Bölüm 4'de ise çalışmanın sonuçları değerlendirilmektedir.

2. AKILLI YEDEKLEME ALGORİTMASI (SMART REDUNDANCY RECONFIGURATION ALGORITHM)

Haberleşme uydularında karşılaşılan yedekleme problemlerinin çözümü için bu makalede önerilen AYA'nın matematiksel modeli ve çözüm algoritması alt bölümlerde detaylandırılmaktadır. Problemin çözümüne, öncelikle verilen bir giriş cihazının, bir çıkış cihazına bağlanabileceği tüm yolların hesaplanması ile başlanmıştır. Sonrasında birden fazla giriş cihazının, beraberce ve birbirlerinin geçtiği yolları kullanmadan, çıkış cihazlarına bağlanabileceği yolların nasıl hesaplanabileceği gösterilmiştir. Son aşamada ise anahtar sayısı ve kesinti sayısı kriterlerinin katılımı ile problemin nihai çözümü tamamlanmıştır.

2.1. Matematiksel Model (Mathematical Model)

Problemin matematiksel modellemesi için giriş cihazlarına bağlı anahtar uçları i_x çıkış cihazlarına bağlı olan anahtar uçları o_x ve bir başka anahtar ucuna bağlı olan uçlar ise k_x indisleri ile adlandırılmaktadır. Her bir satırda, şebekedeki anahtarların dört ucunun indisleri C Bağlantı Matrisini oluşturulmaktadır. Bağlantı Matrisi anahtar sayısı kadar satır ve dört adet sütuna sahiptir. 1-4 arasında bir değer alacak olan anahtar pozisyonları ise P Pozisyon Vektörünü oluşturmaktadır. Pozisyon vektörü de anahtar sayısı kadar satırdan oluşmaktadır. Bu modelleme ile her türlü faydalı yük sistemi, satırları anahtar uçları indislerinden oluşan C Bağlantı Matrisi ve anahtarların mevcut pozisyonlarını gösteren P Pozisyon Vektörü ile tanımlanabilmektedir [2].



Şekil 3. Faydalı yük sistemi alış bölümü. (Payload system receiver section)

Şekil 3'te gösterilen örnek faydalı yük sisteminde i_1, i_2, i_3, i_4 ve i_5 girişleri sırasıyla o_1, o_4, o_5, o_6 ve o_7 çıkış cihazlarına bağlıdır. Giriş cihazlarının bir çıkış cihazına bağlı olduğu yollar kalın olarak gösterilmiştir. Diğer çıkış cihazları o_2, o_3, o_8, o_9 ise yedek durumdadır. Örnek faydalı yük şebekesi, satırları anahtar uçlarının indislerinden oluşan C Bağlantı Matrisi ve anahtarların pozisyonlarından oluşan P Pozisyon Vektörü ile (1) eşitliğinde tanımlanmıştır. C Bağlantı matrisinin ilk satırı i_1'in bağlı olduğu anahtar uçları olan i_1, k_1, o_2, o_1 indislerinden oluşmaktadır. Bu anahtarın pozisyonu Şekil 2'de gösterildiği üzere "1" olup, P pozisyon vektörünün ilk satırında gösterilmiştir.

$$C = \begin{bmatrix} i_1 & k_1 & o_2 & o_1 \\ i_2 & k_2 & o_4 & k_3 \\ i_3 & k_4 & o_5 & k_5 \\ i_4 & k_6 & o_6 & k_2 \\ i_5 & k_7 & o_7 & k_4 \\ k_3 & k_5 & o_3 & k_1 \\ k_6 & o_9 & o_8 & k_7 \end{bmatrix} \quad P = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \\ 3 \end{bmatrix} \tag{1}$$

2.2. Tek Giriş İçin Çözüm (Solution for Single Input)

Verilen bir giriş cihazının bir çıkış cihazına bağlanabileceği tüm yollar, şebekede bulunan anahtarların alabileceği pozisyonların kombinasyonları denenerek bulunabilir. Ancak anahtar sayısı arttıkça, örneğin 30 anahtarlı bir şebekede, 4^{30} farklı kombinasyonun hesaplanması gereği, bu metodun uygun olmadığını göstermektedir. Bir giriş cihazının bağlı olduğu anahtar üzerinden gidebileceği üç farklı seçenek vardır. Bunlar;

1. Bir başka giriş cihazı
2. Bir çıkış cihazı
3. Diğer bir anahtar ucu

olabilir. Giriş cihazının bağlı olduğu anahtar üzerinden bir çıkış cihazına bağlanması seçeneği geçerli bir çözümü oluşturmaktadır. Giriş cihazının bir başka giriş cihazına bağlanması, geçerli bir çözüm değildir. Giriş cihazının bağlı olduğu anahtar üzerinden bir başka anahtara yönlendiği durumda ise yol aramaya devam edilmelidir.

Şekil 3'te gösterilen örnek şebekede, i_1 giriş sinyalinin gidebileceği üç farklı yol bulunmaktadır. Bu yollar k_1, o_2 veya o_1'dir. i_1-o_2 ve i_1-o_1 bağlantıları geçerli bir çözüm oluşturmaktadır. Üçüncü bağlantı seçeneği olan i_1-k_1 bağlantısında ise yol aramaya devam edilmelidir. Geçerli bir çözümü sağlayan anahtar pozisyonları Çözüm Matrisini oluşturmaktadır. Pozisyonları çözümü etkilemeyen diğer anahtar pozisyonları "0" olarak

gösterilmiştir. i_1 - o_2 bağlantısı için, i_1'in bağlı olduğu anahtarın pozisyon değeri "2" olmaktadır. Bu çözümde, sinyal sadece ilk anahtar üzerinden geçtiğinden diğer anahtarların durumu çözümü değiştirmemektedir ve "0" olarak gösterilmiştir. Benzer şekilde i_1 - o_1 bağlantısı için anahtar pozisyonu "1" olmaktadır ve diğer anahtarların pozisyonları ise "0" değerini almaktadır. Bu şekilde verilen bir i_x giriş indisi ve C_n Bağlantı Matrisi için Çözüm Matrisini yukarıda açıklandığı şekilde hesaplayan fonksiyon $F_s(i_x, C_x)$ olmak üzere S_0 Çözüm Matrisi (2) eşitliğinde gösterilmektedir.

$$C_o = \begin{bmatrix} \mathbf{i}_1 & \mathbf{k}_1 & \mathbf{o}_2 & \mathbf{o}_1 \\ \mathrm{i}_2 & \mathrm{k}_2 & \mathrm{o}_3 & \mathrm{k}_3 \\ \mathrm{i}_3 & \mathrm{k}_4 & \mathrm{o}_5 & \mathrm{k}_5 \\ \mathrm{i}_4 & \mathrm{k}_6 & \mathrm{o}_6 & \mathrm{k}_2 \\ \mathrm{i}_5 & \mathrm{k}_7 & \mathrm{o}_7 & \mathrm{k}_4 \\ \mathrm{k}_3 & \mathrm{k}_5 & \mathrm{o}_3 & \mathrm{k}_1 \\ \mathrm{k}_6 & \mathrm{o}_9 & \mathrm{o}_8 & \mathrm{k}_7 \end{bmatrix} \quad S_0 = F_s(i_1, C_0) = \begin{bmatrix} 2 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{2}$$

i_1 giriş cihazının gidebileceği üçüncü seçenek olan k_1 bağlantısında ise i_1 giriş cihazı k_1 anahtar ucuna taşınarak başlangıçtaki Bağlantı Matrisi olan C_o üzerinde, k_1 indisi i_1 ile değiştirilir. Üzerinden geçilen yolların tekrar kullanılmaması için ilk satırdaki i_1 ve k_1 indisleri ise silinir. Bu şekilde (3) eşitliğinde gösterilen yeni C_1 Bağlantı Matrisi oluşturulur. Bağlantı Matrisini hesaplayan $F_c(i_1, C_0)$ fonksiyonu (3) eşitliğinde gösterilmiştir.

$$\mathrm{C_1 = F_c(i_1, C_0) = } \begin{bmatrix} '' & '' & \mathrm{o}_2 & \mathrm{o}_1 \\ \mathrm{i}_2 & \mathrm{k}_2 & \mathrm{o}_4 & \mathrm{k}_3 \\ \mathrm{i}_3 & \mathrm{k}_4 & \mathrm{o}_5 & \mathrm{k}_5 \\ \mathrm{i}_4 & \mathrm{k}_6 & \mathrm{o}_6 & \mathrm{k}_2 \\ \mathrm{i}_5 & \mathrm{k}_7 & \mathrm{o}_7 & \mathrm{k}_4 \\ \mathrm{k}_3 & \mathrm{k}_5 & \mathrm{o}_3 & \mathrm{i}_1 \\ \mathrm{k}_6 & \mathrm{o}_9 & \mathrm{o}_8 & \mathrm{k}_7 \end{bmatrix} \tag{3}$$

$$\mathrm{S_1 = F_s(i_1, C_1) = } \begin{bmatrix} 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 3 \\ 0 \end{bmatrix} \tag{4}$$

Algoritma aynı şekilde özyinelemeli olarak i_1 için bu sefer C_1 Bağlantı Matrisi üzerinde çözüm aramaya devam etmektedir. C_1 matrisinde i_1 'in gidebileceği üç yol k_3 k_5 o_3'dür. Bu üç seçenekten sadece i_1 - o_3 bağlantısı geçerli bir çözüm olup bu bağlantı için birinci ve altıncı satırdaki anahtarların "3" pozisyonunda olması gerekmektedir. Diğer anahtarların pozisyonu izlenen yolu değiştirmediğinden "0" değerini almaktadır. Bulunan S_1 Çözüm Matrisi (4) eşitliğinde gösterilmiştir.

Bu örneklerden sonra, verilen bir i_x giriş indisi ve C_n Bağlantı Matrisi için Çözüm Matrisini yukarıda açıklandığı şekilde hesaplayan $F_s(i_x, C_n)$ fonksiyonu (5), bir sonraki iterasyon için gerekli yeni Bağlantı Matrisini hesaplayan $F_c(i_x, C_n)$ fonksiyonu ise (6) eşitliğinde verilmektedir. Problemin tüm çözümleri ise Çözüm Matrislerinin birleştirilmesi (Concatenation) ile (7) elde edilmektedir.

$$\mathrm{S_n = F_s(i_x, C_n)} \tag{5}$$

$$\mathrm{C_{n+1} = F_c(i_x, C_n)} \tag{6}$$

$$\mathrm{S = [S_o S_1 S_2 ... S_n]} \tag{7}$$

i_1 giriş sinyalinin izleyebileceği yolların oluşturduğu Şekil 5'te gösterilen ağaç yapısı, graf teorisindeki Sığ Öncelikli Arama (Breadth First Search, BFS) algoritmasına benzemektedir. İlk adımda i_1'in gidebileceği o_2 ve o_1 çıkışları geçerli bir çözüm oluştururken, k_1 üzerinden yol aramaya devam edilmektedir. BFS algoritması tüm indisler üzerinden geçildiğinde sonlanırken, AYA, i_x giriş cihazının gidebileceği yol buldukça devam eder. Tüm anahtarlar üzerinden geçildiğinde veya devam edecek bir anahtar ucu olmadığında ise yol arama sonlandırılır.

2.3. Çoklu Giriş İçin Çözüm (Solution for Multiple Input)

AYA, birden fazla giriş cihazının, birer çıkış cihazına bağlanabileceği yoları da aynı yöntem ile bulabilmektedir. Çoklu yolların bulunmasında giriş sinyallerinin izlediği yolların birbirini kesmediğine dikkat edilmelidir. Yol aramanın her adımında giriş cihazlarının beraberce gidebileceği tüm yollar hesaplanır. Giriş cihazlarının hepsinin birer çıkış cihazına ulaştığı anahtar pozisyonları Çözüm Matrisini oluşturur. Giriş cihazlarının başka giriş cihazına bağlandığı durumlar geçerli bir çözüm oluşturmadığından yol arama, bu dal üzerinde devam etmez. Giriş cihazlarının başka anahtar uçlarına

bağlandığı durumlarda ise yeni bağlantı matrisi hesaplanarak algoritma özyinelemeli şekilde yol aramaya devam etmektedir.

Örneğin, i_1'in gidebileceği k_1, o_2, o_1 ve i_2'nin gidebileceği k_2, o_4, k_3 üç yol olup, i_1 ve i_2'nin beraberce gidebilecekleri yollar ise k_1-k_2, k_1-o_4, k_1-k_3, o_2-k_2, o_2-o_4, o_2-k_3, o_1-k_2, o_1-o_4 ve o_1-k_3 olacaktır. Bu yollardan i_1 ve i_2'nin her ikisini de birer çıkış ekipmanına ulaştıran o_2-o_4 ve o_1-o_4 birer çözüm olup ilgili anahtar pozisyonları Çözüm Matrisini oluşturur. Diğer durumlarda ise yol aramaya devam edilecektir. i_x'in çoklu giriş indislerinden oluştuğu durumda da (5), (6) ve (7) eşitlikleri geçerlidir. Tek giriş cihazı için yol bulmada olduğu gibi çoklu yol bulmada da giriş cihazlarının indisleri ilerledikleri anahtar uçları indislerine taşınır ve üzerinden geçilen yollar Şekil 4'de gösterildiği gibi silinir, tırnak içinde '' boş karakter olarak gösterilmiştir.

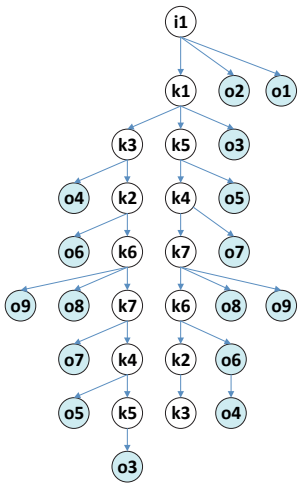
$$C_o = \begin{bmatrix} i_1 & k_1 & o_2 & o_1 \\ i_2 & k_2 & o_3 & k_3 \\ i_3 & k_4 & o_5 & k_5 \\ i_4 & k_6 & o_6 & k_2 \\ i_5 & k_7 & o_7 & k_4 \\ k_3 & k_5 & o_3 & k_1 \\ k_6 & o_9 & o_8 & k_7 \end{bmatrix} \quad C_1 = \begin{bmatrix} '' & '' & o_2 & o_1 \\ '' & '' & o_3 & k_3 \\ i_3 & k_4 & o_5 & k_5 \\ i_4 & k_6 & o_6 & i_2 \\ i_5 & k_7 & o_7 & k_4 \\ k_3 & k_5 & o_3 & i_1 \\ k_6 & o_9 & o_8 & k_7 \end{bmatrix}$$

Şekil 4. i_1 - i_2'nin k_1-k_2 ye taşınması (Move of i_1 - i_2 to k_1-k_2)

2.4. Kısıtlama Kriterleri (Limiting Constraints)

Uydu işletmeciliğinde, yedek cihazlara giden çözümlerin mevcut bağlı kanallarda kesinti oluşturmaması ve yolların üzerinden geçtiği anahtar sayısının en az olması esastır. Bu sebeple yukarıda açıklanan algoritmada kesinti sayısı ve üzerinden geçilen anahtar sayısı kriterlerinin de dikkate alınması gerekmektedir. AYA, giriş cihazlarından başlayan bir ağaç (Şekil 5) yapısına sahip olduğundan, kesinti sayısı veya üzerinden geçilen anahtar sayısı kriterleri aşıldığında yol arama ilgili dalda kesilerek diğer dallarda devam eder.

Kesinti sayısı, başlangıçtaki Pozisyon Vektörü ile Çözüm Vektörü karşılaştırılarak bulunabilir. Bu karşılaştırmada, kendisine yeni yol aranan girişlerin bağlı olduğu anahtarlar hariç tutulmalıdır. Zira bu girişlerin gidebileceği yeni yollar araştırıldığından bunların bağlı olduğu anahtarların durumları da değişecektir. Giriş sinyallerinin üzerinden geçtiği diğer anahtarlardaki değişimler ise kesintiye sebep olacağından bu iki durum matrisi karşılaştırılarak kesinti sayısı bulunmaktadır. Bazı durumlarda mevcut kanallarda kesintiye yol açmayacak bir çözüm bulunamayabilir. Bu durumda algoritmanın kesinti oluşan kanallar için de yeni bağlantı yolları bulması gerekecektir. AYA, başlangıçtaki giriş cihazlarına, kesintiye uğrayan giriş cihazları da eklenerek hepsi için kesintisiz bir çözümleri hesaplamaktadır.



Şekil 5. i_1 girişi için çözüm ağacı (Solution tree for i_1 input)

3. UYGULAMA SONUÇLARI (APPLICATION RESULTS)

Kaba kodu Şekil 6'da gösterilen AYA'nın uygulaması MATLAB yazılımı ile gerçekleştirilmiştir. i_1 giriş sinyalinin bir çıkış cihazına ulaşabileceği Çözüm Matrisi verilen 16 farklı yol (8) eşitliğinde gösterilmiştir. Yedi anahtarın alabileceği pozisyonların oluşturduğu 16.384 farklı kombinasyona kıyasla AYA'nın hesapladığı 16 adet çözüm, algoritmanın etkinliğini göstermektedir.

$$S = \begin{bmatrix} 2 & 1 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 3 & 4 & 4 & 4 & 4 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 3 & 4 & 4 & 4 & 4 & 4 \\ 0 & 0 & 0 & 0 & 3 & 4 & 4 & 4 & 4 & 4 & 0 & 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 4 & 0 & 3 & 4 & 4 & 4 & 4 \\ 0 & 0 & 3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 4 & 4 & 4 & 4 & 4 & 4 \\ 0 & 0 & 0 & 0 & 0 & 3 & 2 & 1 & 1 & 1 & 0 & 0 & 4 & 3 & 1 & 1 \end{bmatrix} \quad (8)$$

AYA, çoklu giriş sinyalleri için de test edilmiştir. i_1 ve i_2 'nin birer çıkış cihazına ulaşabileceği 66 yol bulunurken i_1, i_2 ve i_3 için 137, i_1, i_2, i_3 ve i_4 için 187, i_1, i_2, i_3, i_4 ve i_5 giriş cihazlarının birer çıkış cihazına bağlanabileceği 200 farklı yol hesaplanmaktadır. Bu sonuçlardan görüldüğü üzere çözüm sayısı giriş, çıkış cihazları ve anahtar sayısına bağlı olarak artmaktadır.

1. Verilen C_n bağlantı matrisi ve pozisyon vektörü için i_x 'in gidebileceği indisleri bul
2. o_x indisleri için
   a. Anahtar sayısını bir arttır
   b. i_x - o_x bağlantılarını sağlayan pozisyon vektörlerini hesapla
   c. Tüm pozisyon vektörleri için
      i. Kesinti sayısını hesapla
      ii. Anahtar ve kesinti sayısı aşılmadı ise pozisyon vektörünü S çözüm matrisine ekle
3. k_x indisleri için
   a. Anahtar sayısını bir arttır
   b. i_x - k_x bağlantılarını sağlayan pozisyon vektörlerini hesapla
   c. Tüm pozisyon vektörleri için
      i. Kesinti sayısını hesapla
      ii. Anahtar ve kesinti sayısı aşılmadı ise bağlantı matrisi üzerinde i_x 'i k_x 'e taşıyarak yeni bağlantı matrisi C_{n+1} 'i hesapla
      iii. Yeni bağlantı matrisi ve pozisyon vektörünü kullanarak 1. adıma git
4. Kesinti sayısı kriteri >0 ise S çözüm matrisini tüm sütunları için
   a. Kesintiye uğrayan i_y girişlerini hesapla
   b. i_x ve i_y leri birleştir ve 1. adıma giderek kesintisiz çözüm bul

Şekil 6. Algoritma kaba kodu (Pseudo code of algorithm)

Windows 7 Enterprise yüklü, Intel Core Duo CPU L9400 (32 bit 1.86 GHz işlemcili ve 2GB RAM) bir dizüstü bilgisayarda MATLAB kullanılarak AYA test edilmiştir. Çözüm sayısı ve süreleri şebeke topolojisine bağlı olmakla birlikte giriş cihazı sayısı arttıkça çözüm sayısı ve çözüm sürelerinin arttığı örnek alış şebekesi için Tablo 1'de gösterilmektedir.

Tablo 1. Çözüm sayısı ve süreleri (Number of solutions and time)

Giriş	Çözüm Sayısı	Süre (saniye)
i_1	16	0,18
i_1, i_2	66	0,88
i_1, i_2, i_3	137	2,16
i_1, i_2, i_3, i_4	187	4,71
i_1, i_2, i_3, i_4, i_5	200	7,75

AYA, 30 anahtar, 24 giriş, 6'sı yedek toplam 30 çıkış cihazından oluşan, TÜRKSAT-3A uydusu faydalı yük sistemi üzerinde de test edilmiş ve halihazırda kullanılan ICAREF ticari yazılımı ile aynı sonuçlar elde edilmiştir. TÜRKSAT-3A uydusunun 30 anahtarlı şebekesinde, tek giriş için 208 çözüm sayısı ve 11,32 saniye çözüm süresinin, anahtar ve kesinti sayısı kriterlerinin kullanılması ile azaldığı Tablo 2'de gösterilmektedir.

Tablo 2. Kısıtlama kriterleri ve çözüm süreleri (Limiting constraints and time)

	Tüm Çözümler	Kesintisiz Çözümler	3 Anahtar Kısıtlı
Çözüm Sayısı	208	1	6
Süre (saniye)	11,32	0,44	0,47

Uydu işletmeciliğinde mevcut kanallar üzerinde kesintiye sebep olmayan ve en fazla üç anahtar üzerinden geçen çözümler tercih edilmektedir. Bu özelliklere sahip bir çözüm bulunamaz ise en az kesintiye sebep olacak çözüm mecburen uygulanmaktadır. Bu durum AYA'nın işletmecilik uygulamaları doğrultusunda kısa sürelerde çözüm bulmasını kolaylaştırmaktadır.

4. SONUÇLAR (CONCLUSIONS)

Bu makalede, haberleşme uydularının faydalı yük sisteminde karşılaşılan yedek yol bulma problemlerini çözmek üzere geliştirilen Akıllı Yedekleme Algoritması (AYA) sunulmuştur. AYA ile her türlü faydalı yük şebekesi, Bağlantı ve Pozisyon Matrisi adı verilen iki matris ile tanımlanabilmektedir. Bağlantı ve Pozisyon matrislerinin kullanan AYA, mevcut kanallarda oluşabilecek kesinti sayısı ve sinyalin üzerinden geçtiği anahtar sayısı kriterlerini de dikkate alarak, tek veya birden fazla giriş cihazının çıkış cihazlarına ulaşabileceği tüm yolları hesaplayabilmektedir.

AYA, TÜRKSAT uyduları işletmesinde kullanılan ICAREF ticari yazılımı ile TÜRKSAT-3A uydusu faydalı yük sisteminde farklı arıza senaryoları için karşılaştırılmış ve aynı sonuçları kısa sürelerde elde ederek doğrulanmıştır.

Ticari yazılımlar, farklı faydalı yük sistemleri cihazlarının tanımlanmasında ve farklı kriterlerin uygulanmasında gerekli esnekliğe sahip değildir. Kendi özel algoritmalarını kullandıklarından, algoritma üzerinde bir geliştirme yapılması mümkün olmamaktadır. AYA'nın açık mimarisi ile her türlü faydalı yük şebekesi, matrisler yardımı ile kolayca tanımlanabilmekte ve karşılaşılabilecek farklı arıza senaryolarında yedek cihazlara bağlantı sağlayacak yolları, kesinti sayısı ve anahtar sayısı kriterlerini de dikkate alınarak kısa sürede hesaplanabilmektedir. AYA'nın uygulaması MATLAB ile gerçekleştirilerek, TÜRKSAT uydularının işletmesinde kullanılabilecek bir program haline getirilmiştir.

KAYNAKLAR (REFERENCES)

1. Maral, G. ve Bousquet, M., **Satellite Communications Systems**, John Wiley & Sons, New York, A.B.D., 1998.
2. Gulgonul, S., Koklukaya, E., Erturk, I. ve Tesneli, A.Y., "Communication Satellite Payload Redundancy Reconfiguration," **Satellite Telecommunications (ESTEL), 2012 IEEE First AESS European Conference on**, 1-4, 2012.
3. Chaumon, J.P., Gil, J.C., Beech, T.W. ve Garcia, G., "SmartRings: Advanced Tool for Communications Satellite Payload Reconfiguration", **IEEE Aerospace Conference,** 2006.
4. TRECS: Transponder Reconfiguration System, http://www.integ.com/TRECS.html
5. Stathakis, A., Danoy, G., Bouvry P. ve Morelli, G., "Satellite Payload Reconfiguration Optimisation: an ILP Model", **Intelligent Information and Database Systems**, Cilt 7197, Springer, Lecture Notes in Computer Science, 311-320, 2012.
6. Simone, L. ve Pensa, E., "Analysis and Design of Redundant Networks for Satellite Payloads", **APPLIED MICROWAVE WIRELESS**, 42-56, 2001.
7. Stathakis, A., Danoy, G., Veneziano T., Bouvry P. ve Morelli G., "Bi-objective Optimisation of Satellite Payload Configuration", **Proceedings of the 13e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF)**, 2012.
8. Stathakis, A., Danoy, G., Veneziano, T., Schleich, J., Bouvry P. ve Morelli G., "Optimising Satellite Payload Reconfiguration: An ILP Approach for Minimising Channel Interruptions", **2nd ESA Workshop on Advanced Flexible Telecom Payloads**, 1-8, 2012.
9. Bermond, J.C., Darrot, E., ve Delmas O., "Design of Fault Tolerant On-Board Networks in Satellites", **Networks,** Cilt 40, 202–207, 2002.
10. Gamvros, I., **Satellite Network Design, Optimization and Nanagement,** Doktora tezi, University of Maryland, 2006.
11. Guerrero, E., Alvarez, J. ve Rivero, L., "Redundancy Ring Design for Transponder Subsystem in the VX-SAT Communication Satellite ", **CADDM** , Cilt 20, No 1, 2010.