KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

KABLOSUZ ALGILAYICI AĞLARDA KAPASİTEYE DAYALI AKILLI VE ADAPTİF AĞ YAPILANDIRILMASI

YÜKSEK LİSANS TEZİ

Elektrik – Elektronik Müh. Amir NASER

EYLÜL 2016
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

KABLOSUZ ALGILAYICI AĞLARDA KAPASİTEYE DAYALI AKILLI VE ADAPTİF AĞ YAPILANDIRILMASI

Elektrik – Elektronik Müh. Amir NASER

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"ELEKTRONİK YÜKSEK MÜHENDİSİ"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 29.08.2016
Tezin Savunma Tarihi : 20.09.2016

Tez Danışmanı : Prof. Dr. İsmail H. ÇAVDAR

Trabzon 2016

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Elektrik Elektronik Mühendisliği Anabilim Dalında

Amir NASER Tarafından Hazırlanan

KABLOSUZ ALGILAYICI AĞLARDA KAPASİTEYE DAYALI AKILLI VE ADAPTİF AĞ YAPILANDIRILMASI

başlıklı bu çalışma, Enstitü Yönetim Kurulunun 20/09/2016 gün ve 1666 sayılı kararıyla oluşturulan jüri tarafından yapılan sınavda
YÜKSEK LİSANS TEZİ
olarak kabul edilmiştir.

Jüri Üyeleri

Başkan : Prof. Dr. İsmail H. ÇAVDAR .....

Üye : Yrd. Doç. Dr. Yasin OĞUZ .....

Üye :Yrd. Doç. Dr. Ayhan YAZGAN .....

Prof. Dr. Sadettin KORKMAZ
Enstitü Müdürü

ÖNSÖZ

Kablosuz algılayıcı ağların yerlestirildiği ortamlarda herhangi bir altyapı bulunmamaktadır ve bir çok durumda ağda bulunan düğümlerin bakım, onarım ve enerjilerini yenileme imkanları bulunmamaktadır. Bu nedenle optimum küme başı yeri seçimi ve kümelemesinin yanısıra ağ maliyetını düşürerek KAA'lar enerjiyi verimli bir şekilde kullana bilmekte ve böylece ağ ömrünü uzatmaktadır. Bu derece güncel ve önemli bir konuyu seçmemde bana destek olan ve tüm çalışma boyunca desteklerini esirgemeyen saygıdeğer hocam Prof. Dr. İsmail Hakkı ÇAVDAR'a çok teşekkür eder, şükranlarımı sunarım. Bu çalışmada yardımı geçen herkese çok teşekkür ederim. Ayrıca hayatım boyunca her an yanımda olan sevgili anneme ve saygıdeğer babama teşekkür ederim.

Amir NASER
Trabzon, 2016

TEZ ETİK BEYANNAMESİ

Yüksek Lisans Tezi olarak sunduğum “Kablosuz algılayıcı ağlarda kapasiteye dayalı akıllı ve adaptif ağ yapılandırılması” başlıklı bu çalışmayı baştan sona kadar danışman hocam Prof. Dr. İsmail Hakkı ÇAVDAR’ın sorumluluğunda tamamladığımı, verileri kendim topladığımı, analizleri ilgili laboratuarlarda yaptığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 20/09/2016

Amir NASER

İÇİNDEKİLER

Sayfa No

Yüksek Lisans Tezi

ÖZET

KABLOSUZ ALGILAYICI AĞLARDA KAPASİTEYE DAYALI AKILLI VE ADAPTİF AĞ YAPILANDIRILMASI

Amir NASER

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendeiliği Anabilim Dalı
Danışman: Prof. Dr. İsmail H. ÇAVDAR
2016, 91 Sayfa, 4 Sayfa Ek

Bu tez, algılayıcı ağlarda kapasiteli sunucuların konumlandırma probleminin çözümü üzerine bir çalışmadır. Bu problem np-zor kombinatoryel problemlerin grubunda yer almaktadır. Problemin çözümü ile hem adaptif kümeleme hem de optimum küme başı seçimi gerçekleştirilmiştir. Bu çalışmada küme başlarının seçimi yapay zeka parçacık sürü optimizasyon (*PSO: Particle Swarm Optimization*), diferansiyel gelişim algoritması (*DE: Differential Evolution*) ve yayılmacı yarışmacı algoritma (*ICA: Imperalist Competitive Algorithm)* gibi optimizasyon teknikleri ile yapılmış ve aynı anda her iterasyondaki amaç fonksiyonun değerine bağlı kümeleme ve ağın konfigürasyonu değişmektedir. Yapılan denemelerde elde edilen sonuçlara göre ICA algoritması hız açısından daha iyi sonuçlar göstermektedir ama maliyet açısından PSO algoritması en düşük değerlerle daha iyi sonuçlar sunmaktadır. Kablosuz algılayıcı ağlarda kısıtlı işlemciler kullanıldığı için en uygun algoritmanın çalıştırılması gerekmektedir. ICA algoritması iyi performansa sahiptir ve düşük işlem gücü kullandığı için kablosuz algılayıcı ağlarda kullanımı önerilmiştir.

Anahtar Kelimeler: Kablosuz Algılayıcı Ağlar, Optimizasyon, Adaptif Kümeleme, Yapay Zeka, Sürü Zekası, Yayılmacı Yarışmacı Algoritma, Diferansiyel Gelişim Algoritma

Master Thesis

SUMMARY

SELECTION THE OPTIMUM CLUSTER HEAD IN THE WIRELESS SENSOR NETWORKS AND ADAPTIVE CLUSTERING VIA OPTIMIZATION ALGORITHMS

Amir NASER

Karadeniz Technical University
The Graduate School of Science
Electronic Engineering Graduate Program
Supervisor: Assoc. Prof. İsmail Hakkı ÇAVDAR
2016, 91 Pages, 4 Pages Appendix

This thesis is a study on the solution of the problem of capacity servers' location in sensor networks. This problem is part of the group of np hard combinatorial problems. With the solution of the problem, both adaptive clustering and optimal cluster head selection have been achieved.

In this study, cluster head selection is carried out by optimization techniques such as particle swarm intelligence optimization, differential evolution algorithm, and imperialist competitive algorithm. The purpose of each iteration is dependent upon the value of the function and clustering and network configuration change accordingly.

Tests results show that ICA Algorithm has better results in terms of speed, while PSO Algorithm is better in terms of costs. Since processors with limited capacity are used in wireless sensor networks, it is necessary to use the most appropriate algorithm. ICE algorithm has good performance, and as it uses low processing power, it is suggested for use in wireless sensor networks.

Key Words: Wireless Sensor Networks, Optimization, Adaptive Clustering, Artificial Intelligence, Particle Swarm Optimization, And Imperialist Competitive Algorithm, Differential Evolution Algorithm

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

WSN : Wireless Sensor Networks
PSO : Particle Swarm Optimization
DE : Differential Evolution
ICA : Imperalist Competitive Algorithm
KAA : Kablosuz Algılayıcı Ağ
MEMS : Mikro Elektro-Mekanik
UCLA : University of California, Los Angeles
ADCs : Analog/Dijital Dönüştürücüler
KVAA : Kablosuz Vucut Alan Ağları
FNR : Flat Networks Routing
SPIN : Sensor Protocolsfor Information via Negotiation
ID : Identification
DY : Doğrudan Yayılma
RR : Rumor Routing
HNR : Hierarchical Networks Routing
LEACH : Low Energy Adaptive Clustering Hierarchy
LBR : LocationBased Routing
GPS : Global Positioning System
GAF : Geographic Adaptive Fidelity
MANET : Mobile Ad-Hoc Network
RSSi : Received Signal Strength İndicator
TOF : Time Of Flight
RSSi : Received Signal Strength
DYHM : Do you hear me
DGA : Diferansiyel Gelişim Algoritması
NP : Number of Population
CR : Cross Over Rate

1. GENEL BİLGİLER

1.1. Giriş

Kablosuz Algılayıcı Ağları (KAA) kavramı ilk kez 1980'lerin başlarında karşımıza çıkmıştır. Mikro elektro-mekanik (MEMS) sistemlerdeki gelişmeler ve kablosuz haberleşme sistemlerindeki ilerlemelerle birlikte 1990'lı yıllarda önemli bir araştırma alanı haline gelmeye başlamıştır. İlk zamanlarda askeri alanda kullanılan kablosuz algılayıcı ağları; zamanla maliyetlerinin düşmesi ile çok yaygın olarak kullanılmaya başlanmıştır [1].

Kablosuz algılayıcı ağlar halihazırda askeri, sağlık, domestik ve endüstriyel uygulamalardan çevresel bilgilerin izlenmesine örneğin, atmosferik kirlilik, su kalitesi, tarım ve daha birçok alanda kullanılmaktadır [2]. KAA oldukça basit yapıları, her ortama uyumlu olabilmeleri, düşük maliyetli ve gelişen teknolojiye uymak kapasitesine sahiplerdir. Bu türlü ağlar geniş uygulama teyflerine sahip olduğu için araştırmacıların bu alana ilgisini yönlendirmede önemli bir rol oynamışlardır [3].

KAA birçok uygulama alanında kullanılmasına rağmen sınırlı enerji kaynağına, sınırlı işlem kapasiteleri ve sınırlı haberleşme yeteneklerine sahip olduklarından dolayı kullanımlarında çeşitli kısıtlamalar ortaya çıkmaktadır. Dolayısıyla bu alanda çalışma yapılması durumunda bu kısıtlamaların göz önünde bulundurulması gerekmektedir. Büyük alanlara yerleştirilen ve sınırlı kaynaklara sahip algılayıcı elemanların kullanımı ve kaynakları değiştirilememesinden dolayı araştırmacılar KAA'lar üzerinde yönlendirme algoritmaları ve protokoller üzerine odaklanmışlardır [4, 5]. Böylece mevcut kaynakların etkin kullanımını sağlayarak ağ performansı ve ömrünün üzerine olumlu etkiler elde etmek amaçlanmıştır.

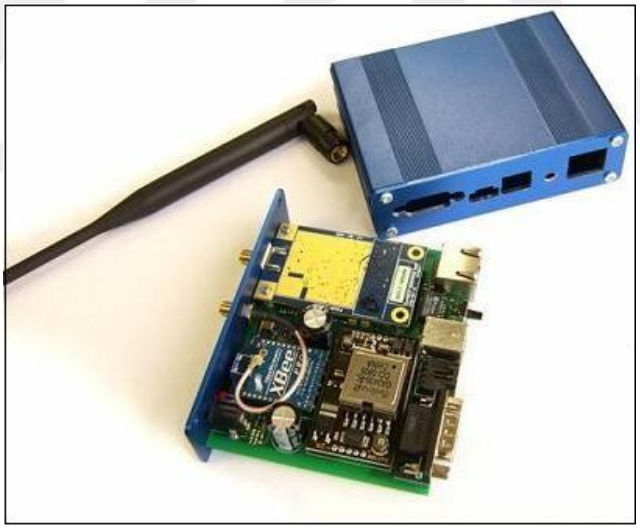
1.2. Kablosuz Algılayıcı Ağlar

KAA genellikle belirli bir coğrafi bölge üzerinde fiziksel olguları (sıcaklık, nem, basınç, hız, ışık, gürültü vb.) gerçek zamanlı olarak gözetlemek üzere bir araya getirilmiş çok sayıda düşük maliyetli algılayıcı düğümden oluşan mikro-elektromekanik sistemler

(MEMS) olarak tanımlanabilir [6]. Bu algılayıcılar etraflarındaki olguları bilgi olarak toplayıp belirli rotalama algoritmalarıyla bir merkeze iletirler [7].

1.3. Algılayıcı Düğüm

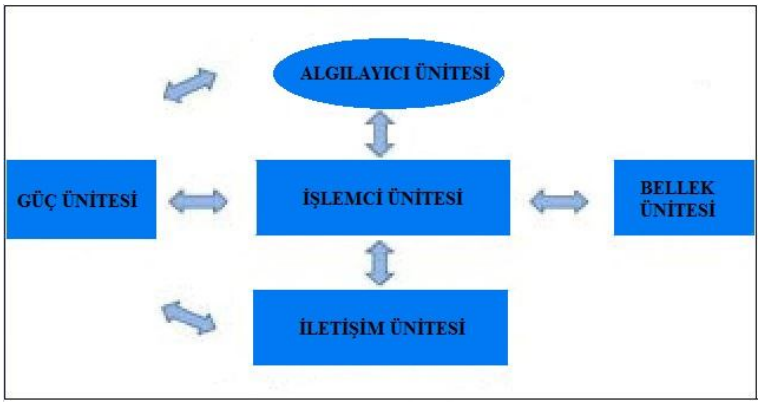
Algılayıcı düğümlerin geliştirmesinin başlangıcı 1988 yılındaki Smartdust projesine dayanır [8], yine ayni yılda UCLA ve Rockwell Bilim Merkezi tarafından ticari algılayıcı düğümler üretilmesi başlanmıştır. Bu algılayıcı düğümler düşük maliyet, düşük güç tüketimi, veri işleme ve kablosuz iletişim kapasitelere sahiptir [9]. Halihazırda dünyada bulunan çeşitli algılayıcı düğüm markaları var örneğin: TelosB [10], MicaZ [10]. Şekil 1' de bir algılayıcı düğümü gösterilmiştir.



Şekil 1. Algılayıcı düğüm

1.4. Algılayıcı Düğümlerin Bileşenleri

Genel olarak bir algılayıcı düğüm dört bileşenden oluşmaktadır: i) Bir mikroişlemci veya mikro denetleyici ünitesi, ii) İletişim ünitesi, iii) Algılayıcı ünitesi, iv) Güç kaynağı ünitesi. Algılayıcı düğümlere bazen uygulamaya bağlı olarak başka ünitelerde eklenir örneğin bir konum bulma ünitesi, bir güç jeneratörü yada bir dışsal bellek. Şekil 2' de algılayıcı düğümün temel bileşenleri gösterilmiştir.



Şekil 2. Bir algılayıcı düğümün temel bileşenleri

1.4.1. Algılama Ünitesi

Algılama ünite genellikle iki alt üniteden oluşur: Algılayıcı düğümler ve analog/dijital dönüştürücüler (ADCs). Algılayıcı düğümler fiziksel olayları (sıcaklık, nem, basınç vs.) gözlemleyip analog sinyale dönüştürürler ve bu analog sinyaller analog/dijital dönüştürücüler ile dijital sinyale çevirirler. Algılayıcı ünitesindeki algılayıcı düğümler tarafından algılanan veriler, işlenmesi için bir sonraki ünite olan işleme ünitesine gönderilir.

1.4.2. İşlem Ünitesi

İşleme ünitesinde yerleşik bir mikroişlemci ve bellek bölümleri bulunmaktadır. Mikroişlemci, algılayıcı ünitesinden gelen verileri işlerken bellek ise uygulamaya göre ihtiyaç duyulan bellek alanını yönetir. Bu ünite algılayıcıların kontrolü, iletişimi protokolü ve diğer düğümlerle atanan algılama görevleri yürütmektedir [11].

1.4.3. İletişim Ünitesi

Bir alıcı-verici ünitesi bir algılayıcı düğümü ağa bağlar, yani algılanan değerleri kendi düğümünden alır ve değer düğümler ve baz istasyonuna ulaştırır.

1.4.4. Güç Kaynağı Ünitesi

Bir algılayıcı düğümünün en önemli bileşenlerinden biri güç birimidir ve algılayıcı düğümün bütün bileşenlerin enerjisini sağlar. Birçok uygulamalarda güç ünitesinin değişme imkanı yoktur ve bu nedenle güç ünitesinin ömrü algılayıcı düğümün ömrünü belirler. Bu yüzden algılayıcı düğümlerin donanım tasarımında az enerji tüketen parçaları ve tasarımlar istenir.

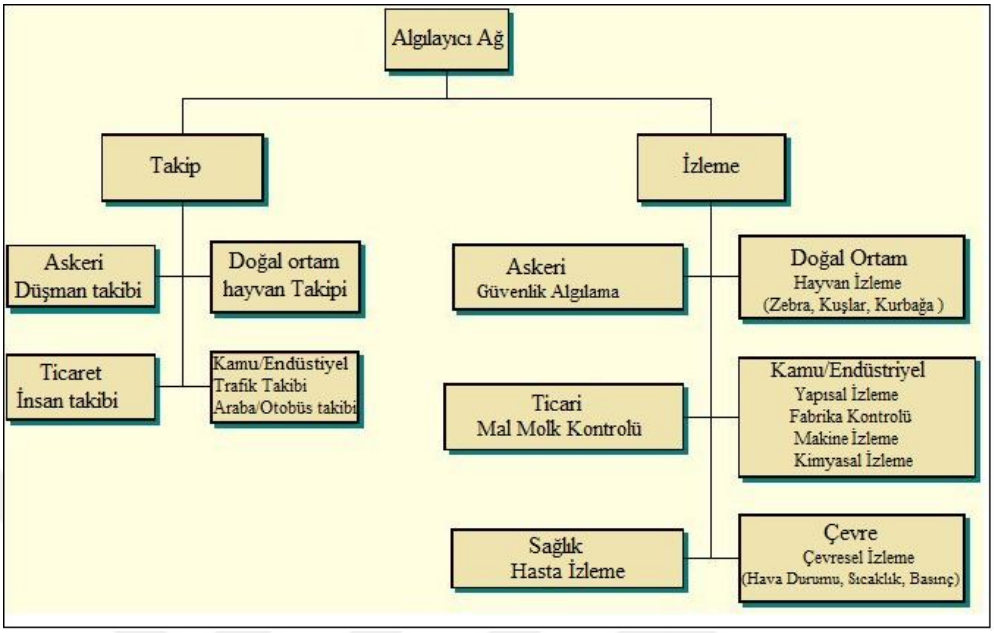
1.5. Kablosuz Algılayıcı Ağların çeşitleri

Güncel kablosuz algılayıcı ağlar karada, yer altında ve su altında yayılmıştır. Ortamına bağlı olarak, bir kablosuz algılayıcı ağı farklı zorluklar ve kısıtlamalar ile karşı karşıyadır. Aslında algılayıcı ağlarda beş türü vardır: karasal KAA, yeraltı KAA, sualtı KAA, Multimedya KAA ve mobil KAA [12].

1.6. Kablosuz Algılayıcı Ağlar Uygulama Alanları

KAA uygulamaları, izleme ve takip olmak üzere iki grubta sınıflandırılabilir.

İzleme uygulamaları, iç / dış mekan çevre izleme, sağlık ve sağlıklı yaşam izleme, enerji izleme, mal ve mülk konumunu izleme, fabrika ve otomasyon işlemini, depremsel ve yapısal izlemeyi içermektedir. Takip uygulamaları ise nesneleri, hayvanları, insanları ve araçları takip etmeyi içermektedir. Bu sınıflandırma Şekil 3'de gösterilmiştir [12].



Şekil 3. Algılayıcı uygulamalarına bir bakış

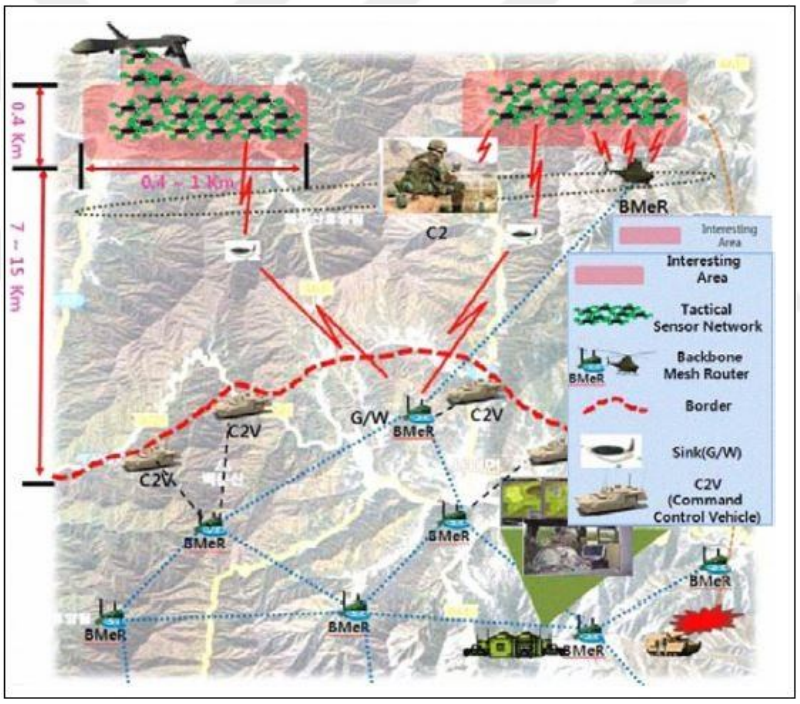
Literatür taramalarına göre kablosuz algılayıcı ağlarının kullanım alanlarının sınıflandırılması yukarıdaki gibi olduğu görülmektedir. Bu uygulama alanları ilerleyen yıllarda daha da gelişeceğini ve insanların yaşamına daha çok gireceği de bir gerçektir.

1.6.1. Askeri Uygulamalar

1978 yılında kablosuz algılayıcı ağlarının ilk kullanımının askeri uygulama alanında olması sebebiyle, askeri uygulamalar kablosuz algılayıcı ağların birincil uygulama alanı olarak kabul edilmektedir [13, 14]. Kablosuz Algılayıcı Ağlar, yoğun dağıtıma dayalı ve düşük maliyetli olmaları, ağ içerisindeki bazı düğümlerin yok edilmesi ya da kaybının askeri operasyonları etkilememesi nedeniyle geleneksel yöntemlere göre çok daha güvenli yaklaşımlar sunmaktadır. Askeri uygulamalarda kullanılan algılayıcı düğümler akustik, deprem, manyetik, kızılötesi, elektro-optik, elektromanyetik algılayıcılar içerebilmekte ve bu ve benzer algılayıcılar ile kontrol, iletişim, keşif, istihbarat, hasar değerlendirme, biyolojik ve kimyasal saldırı tespiti ve hedef tespiti yapılabilmektedir [14,15].

1.6.1.1. Uzaktan Büyük Ölçekli Gözetimsiz Savaş Alanlarda Ağ Tasarımı ve Düşman Hedeflerini Belirlemesi

Bu uygulama ile aynı bölümde binlerce algılayıcı düğümler düşman kuvvetleri tarafında Statik olarak elle veya rastgele insansız hava araçları ve toplar vasıtasıyla dağıtılmaktadır. Bu algılayıcılar kendileri tarafından organize edilmektedir. Algılayıcı düğümleri özerk bağlantıya sahiptir. Bu uygulamada çeşitli algılayıcılar kullanarak düşman kuvvetlerini gözetimi ve belirlemesi yapılıyor [16]. Şekil 4'de uzaktan büyük ölçekli bir ortamda algılayıcı ağın sistemi göstermektedir.



Şekil 4. Uzaktan büyük ölçekli bir ortamda algılayıcı ağı sistemi

1.6.2. Çevresel Uygulamaları

Gerçek zamanlı çevresel felaketlerin izlenmesi donayanın temel ihtiyaçlarından biridir. Farklı teknolojiler bu amaçla geliştirilmiştir, Kablosuz algılayıcı ağlar gerçek zamanlı izleme için kullanılabilir önemli teknolojilerden biridir [17].

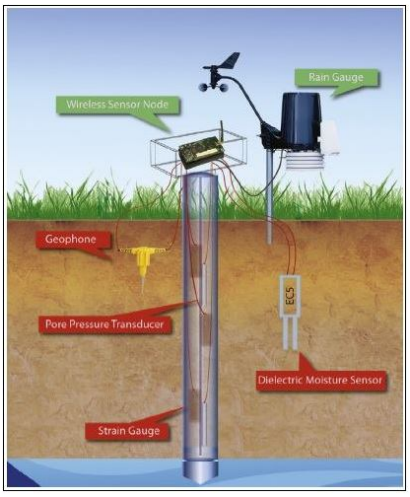
Kablosuz algılayıcı ağlarda kullanılan çevresel algılayıcıların istenilen bölgeye konumlandırılması ile çevre koşullarının izlenmesi (buzul izleme, güneş radyasyonu

haritalama, nehirler, su havzaları ve ekosistem haritalama, çevresel kirlilik izleme gibi), doğal afetlerin algılanması ve tahmin yapıla bilmesi (yangın, sel, tsunami gibi), toprak ve yeraltı gözlemi ile verimli madencilik ve tarım alanlarının tespiti ve kontrolü sağlana bilmektedir [18, 19].

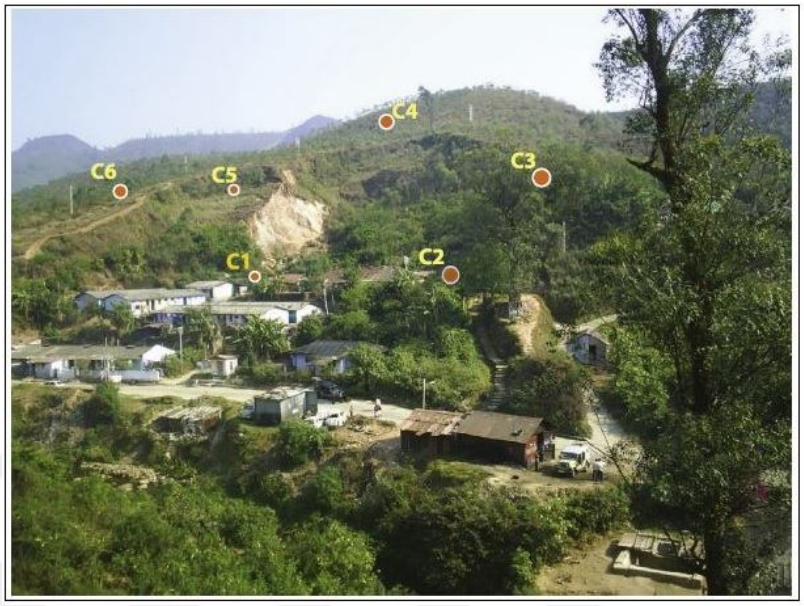
1.6.2.1. Heyelan Tespiti

Bir heyelan bir kısa süreli potansiyel ama yıkıcı bir olaydır. Bu olayın tetikleyici faktörleri yağış veya deprem olabilir. Heyelan sebebi ile Hindistan da her sene aşırı derece mali kaybı oluyor ve bu nedenle insan yaşamı büyük tehlike ile yüzleşiyor. Kablosuz algılayıcı ağları Hindistan da güneybatı bölgesinde yüksek heyelan eğilimli alanın Idukki ilçesinde kullanılarak heyelan tespiti üzerine odaklanılmıştır, bu iş afet zararlarının azaltılmasına yönelik kablosuz ağlar yeteneğini teyit eder. Bu projede heyelan izlemek için bir çok algılayıcı düğüm jeolojik ve hidrolojik özelliklerini ölçmek için yeraltında yerleştirilmiştir. Bu algılayıcı düğümler bir çöklü algılayıcı düğüm derin prob olarak toprağın nem, basınç, yer tabakalarının hareketini, titreme, sıcaklık ve başka olayları algılayıp merkeze göndermektedirler [20].

Şekil 5'te bir çoklu algılayıcı düğüm ve Şekil 6'da çöklü algılayıcı düğümler yerleştiği alan gösterilmektedir [20].



Şekil 5. Çöklü algılayıcı



Şekil 6. Anthoniyar Colony Sitesi, Munnar, Kerala, Hindistan çöklü algılayıcı düğümler dağıtım yerleri

1.6.3. Sağlık Uygulamaları

Dünya üzerindeki nüfusun yaşlanmasıyla birlikte, kronik hastalıkların, ruhsal sorunların, yaralanmaların ve ölümcül bulaşıcı hastalıklara yakalanan kişilerin sayısının artması, sağlık hizmetlerinin kullanımını ve maliyetini arttırmıştır. Bununla birlikte, hastane ortamındaki uzun süreli tedaviler, insanlar üzerinde hastaneye yatış ile ilgili birçok fiziksel, psikolojik ve sosyal sorunlara neden olabilmektedir. Yukarıda belirtilen sorunların azaltılabilmesi için tansiyon, kan şekeri vb. gibi tıbbi ölçümlerin kişinin günlük yaşamını değişikliğe uğratmadan ve kullanıcı konforu gözetilerek yapılması gelişmekte olan kablosuz teknolojilerin kullanımı ile mümkün olmaktadır. Kablosuz teknolojilerin en büyük avantajı sağlık hizmetlerinin sağlık tesisleri dışında da verilebilmesidir. Uzaktan bakım hizmetinin kablosuz teknolojiler kullanarak verilmesi, sağlık hizmeti veren yerlerin iş yükünün azalmasına yardımcı olmaktadır [21].

Kablosuz algılayıcı ağlar, dış ortama ait ışık, sıcaklık, basınç, ses, hareket vb. birçok fiziksel büyüklüğü toplama, toplanan verileri işleme ve haberleşme yapabilme becerisine sahip düğümlerden oluşan bir ağ sistemidir. KAA'ların bir türevi olan kablosuz vücut alan ağları (KVAA) ile bireylere ait fizyolojik veriler toplana bilmektedir. KVAA'lar ile hasta üzerine giydirile bilen veya vücuduna yerleştirile bilen algılayıcılar sayesinde fizyolojik

veriler, istenilen aralıklarda, hastaya acı vermeden ve günlük hayatına devam ederken toplanabilmektedir [21].

1.6.3.1. Kanser Tespiti

Kansere tanı koymak sağlıktaki önemli sorunlardan sayılmaktadır. Günümüzde, kanser insan yaşamı için büyük bir tehdit olmaktadır. ABD'de her yıl sayısı artan kanser ikinci ölüm nedenidir. Halihazırda 9 milyon kişide kanser tanımlanmıştır [22]. Araştırmalar kanser hücrelerinin nitrik oksit yaymasını göstermiştirki bu gazlar tümörün çevresindeki kanı etkilemektedir. Şüpheli konumlarda kan değişiklikleri tespit etme yeteneğine sahip olan bir algılayıcı düğüm yerleştirilebilir. Araştırmalar aynı zamanda bir iğne üzerinde algılayıcı düğümler yerleştirerek yürütülmektedir**.** Bu yöntem hekimlere biyopsi almadan kanser tanımlama olanağı sağlamıştır. Bu cihazda kullanılan algılayıcı düğümler farklı hücre türleri arasında ayrım yeteneğine sahiptirler ve kanserli hücreleri belirtirler [23].

1.6.4. Endüstriyel Uygulamaları

Kablosuz algılayıcı ağ teknolojisi endüstriyel, ticari ve tüketici uygulamaları için büyük bir potansiyel göstermektedir. Özellikle, işlem izleme ve kontrol, basınç, nem, sıcaklık, akış seviyesi, viskozite gibiveri işlemesi, yoğunluk ve titreşim yoğunluğu ölçümleri algılayıcı birimleri aracılığıyla toplanılır ve kablosuz şekilde işletme ve yönetim için bir denetim sisteme gönderilir [24].

Şekil 7. KAA'lar ile işlem kontrolü için bir örnek

Örnek olarak Şekil 7'de görüldüğü gibi soğutma ceketi içinden akış hızının ayarlanması ile bir reaktörün soğuması gerçekleşmiştir. Bu işlemde istenilen sonuca ulaşmak için: önceden belirlenmiş sıcaklığı zaman içinde sürdürmektir.

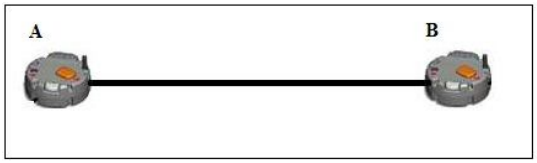
1.7. Kablosuz Algılayıcı Ağ Topolojileri

Kablosuz algılayıcı ağlarda, ağın topolojisi veri transferi sırasında yaşanan gecikme, sınırlı enerji, iletişim kalitesinde kaynak krizi gibi çeşitli kısıtlamaları azaltmak için en önemli role sahip elementtir [25]. Algılayıcı düğümlerin sahip oldukları iletim mesafesi ve komşu düğüm sayısı gibi ağ parametreleri, minimum toplam enerji ihtiyacı ve arızalanan algılayıcı düğümün tolere edilmesi gibi tercihler ağ topolojinin belirlenmesine katkıda bulunur. Bir kablosuz algılayıcı ağda çift, yıldız, karmaşık ve ağaç topoloji olmak üzere kullanıla bilecek dört adet ağ topolojisi mevcuttur [26].

1.7.1. Çift Topoloji

Çift ya da diğer adıyla noktadan noktaya topoloji iki algılayıcı düğüm arasında özel uzun menzilli, yüksek kapasiteli kablosuz bağlantı oluşturur. Anahtarlamalı noktadan

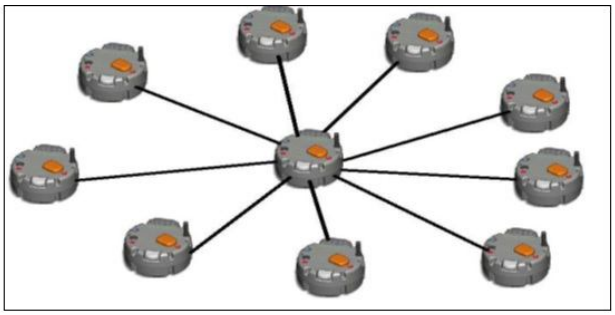
noktaya topoloji geleneksel ağların temelidir. Bu topoloji kablosuz algılayıcı ağlarda kullanılan en yaygın topoloji olmuştur. Bu topolojinin en temel avantajı ve dezavantajı tek bir veri iletişim kanalının bulunmasıdır. İletişimin ancak bu iki düğüm arasında bulunan tek bir veri iletişim kanalı ile sağlanması güvenlik avantajı oluştururken, bu kanalın bozulması iki düğüm arasındaki iletişimin kesilmesine neden olacaktır [26, 27]. Şekil 10 Çift topolojisini göstermektedir.



Şekil 8. Çift topoloji

1.7.2. Yıldız Topoloji

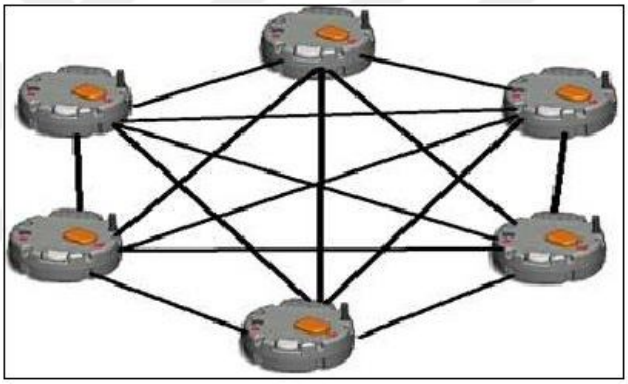
Yıldız topolojinin düzenlenmesi oldukça basittir. Bir merkez düğüm merkezde konumlanır diğer düğümlerse merkezde olan düğüme bağlanır. Uçlarda bulunan bu düğümler ölçtükleri verileri merkez düğüme gönderirler. Uç düğümlerin birbirleriyle iletişimi yoktur. Yıldız topoloji düşük güç tüketimi sağlar ve ağa yeni uç düğümler eklemek kolay olduğundan ölçeklenebilir bir topolojidir. Merkezde bulunan düğümün arızası uç düğümler arasında alternatif bir yol bulunmadığından tüm ağın çalışmamasına sebep olacaktır [26, 27]. Şekil 9 Yıldız topolojisini göstermektedir.



Şekil 9. Yıldız topoloji

1.7.3. Tamamen Bağlı Örgü Topoloji (Mesh)

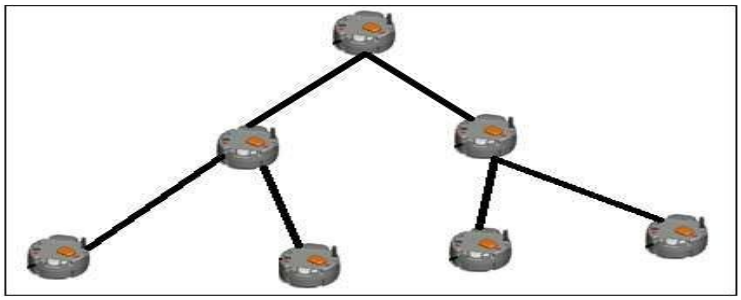
Bu topolojide uç düğüm ve merkez düğüme ek olarak yönlendirici düğümler vardır. Merkez düğüm ağı yönetmek için vardır. Ağda merkez ve yönlendirici düğümlere bağlı olan uç düğümler vardır. Uç düğümler ölçtükleri verileri direkt merkez düğüm ile bağlantıları varsa merkez düğüme, yönlendiricilere bağlantıları varsa yönlendiricilere iletirler. Yönlendiriciler belirlenen güzergahı izleyerek aldıkları veriyi bir sonraki yönlendirici düğüme ve ardından merkez düğüme iletirler. Ağ üzerinde çok fazla gereksiz yol bulunuyor olabilir bu sebeple gecikme ve enerji tüketimi artacaktır [26, 27, 28]. Şekil 10 Tamamen bağlı örgü topolojisini göstermektedir.



Şekil 10. Tamamen bağlı örgü topolojisini (Mesh)

1.7.4. Ağaç Topoloji

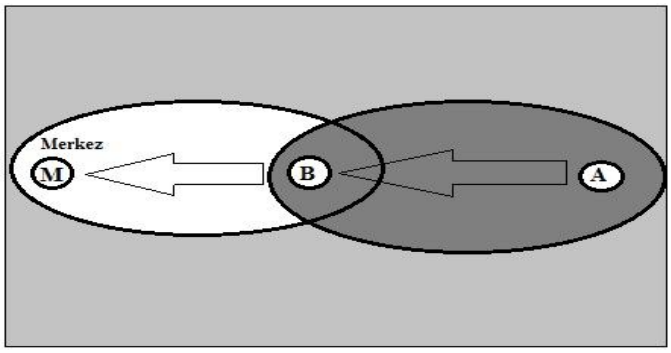
Ağaç topolojisi tamamen bağlı örgü topolojinin (Mesh) değişikliğe uğramış halidir. Ağ üzerinde merkez düğüm, uç düğümler ve yönlendirici düğümler bulunur. Ağda yine merkez ve yönlendirici düğümlere bağlı olan uç düğümler vardır. Uç düğümler ölçtükleri verileri direkt merkez düğüm ile bağlantıları varsa merkeze, yönlendirici düğümlerle bağlantıları varsa onlara iletirler. Yönlendirici düğümler aldıkları verileri direkt bağlı oldukları merkez düğüme iletirler. Yönlendirici düğümlerin arızası uç düğümlerin iletişiminin kopmasına yol açacaktır [26, 27]. Şekil 11 Ağaç Topolojisi göstermektedir.



Şekil 11. Ağaç topolojisi

1.8. Algılayıcı Ağlarda Yönlendirme

Bir kablosuz algılayıcı ağında, algılayıcı düğümleri bazı kısıtlamalara sahiptir, bunlar kapsam alanı, limitli enerji, limitli işlem gücü ve limitli bant genişliğidir. Bu ağlarda verilerin toplanacağı merkeze verilerini ulaştıramayan algılayıcı düğümler komşu algılayıcı düğümler vasıtasıyla verilerini ulaştırmak zorundalar. Şekil 12'de gösterdiği gibi A düğümü merkez düğümle iletişim kurabilmesi için B düğümün yönlendirme yapması gerekmektedir.



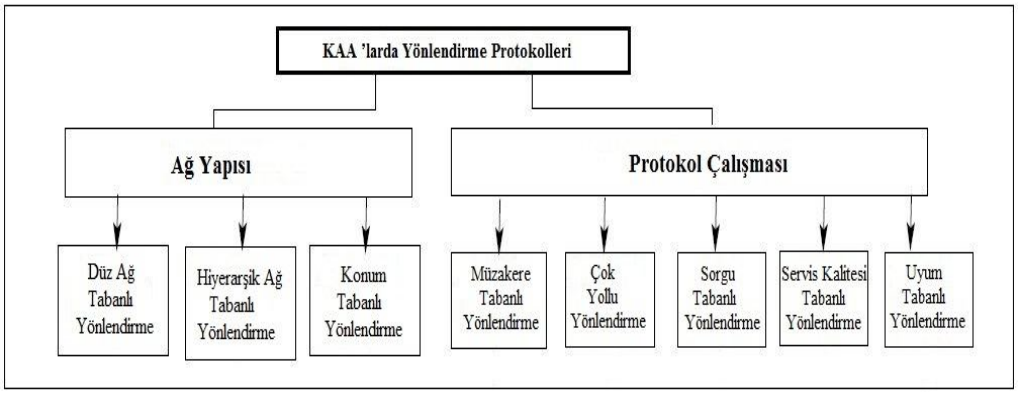
Şekil 12. Algılayıcı düğümlerde yönlendirme

KAA yönlendirmenin temel amaçlarından biri ağ ömrünün artırılması ve yoğun enerji yönetimi tekniklerinin kullanımıyla ortaya çıkan bağlantı hatalarının önlenmesidir. KAA' ın enerji kaynakları sınırlıdır ve değiştirilememektedir. Bununla birlikte, algılayıcı noktaların konumlandırılmaları genelde önceden belirlenmemekte ve zor coğrafya bölgelerine veya afet bölgelerine rastgele yerleştirilmektedirler. Bu ise, algılayıcı noktaların herhangi bir bakım veya düzenleme yapılmaksızın, uzun süre algılama ve

iletişim yapabilmelerini gerektirmektedir. Dolayısıyla, yüksek kalitede algılama ve hatasız çalışma için algılayıcı noktalar arasında önemli ölçüde dayanışma ve işbirliğinin gerçekleştirilmesi gerekmektedir. Bir algılayıcı ağına klasik bir ağ protokolü uygulanamamaktadır. Bu ise literatürde bu alanda yapılmış olan birçok çalışmanın KAA yapılarına uyarlanamamasına neden olmakta ve KAA yapılarına özgü kriterleri dikkate alan ağ protokollerinin geliştirilmesini zorunlu kılmaktadır [29].

1.8.1. Yönlendirme Protokolleri

Genel olarak KAA'larda yönlendirme protokolleri iki kısma bölünebilir, birinci kısım ağ yapısına bağlı, Düz ağ tabanlı Yönlendirme, Hiyerarşik ağ tabanlı yönlendirme ve Konum tabanlı yönlendirme ve ikinci ise protokol çalışmasına bağlı, Sorgu tabanlı yönlendirme, Çok Yollu yönlendirme, Servis Kalitesi tabanlı yönlendirme, Müzakere tabanlı yönlendirme, Uyum tabanlı yönlendirme bölünmektedir [30]. KAA'larda yönlendirme protokollerine bir sınıflandırması Şekil 13'de gösterilmiştir.



Şekil 13. KAA'larda yönlendirme protokollerine bir sınıflandırma

1.8.1.1. Düz Ağ Tabanlı Yönlendirme Protokolleri

Düz ağ tabanlı yönlendirme protokollerinde (Flat Networks Routing Protocols - FNR) ağ içerisindeki tüm algılayıcı düğümleri eşit statüye sahiptir. Her bir düğüm aynı görevleri yapmak ile yükümlüdür. Düz ağ yapısında amaç kaynak düğümden hedef düğüme verinin algılayıcı düğümlerin yardımlaşarak iletilmesini sağlamaktır [30].

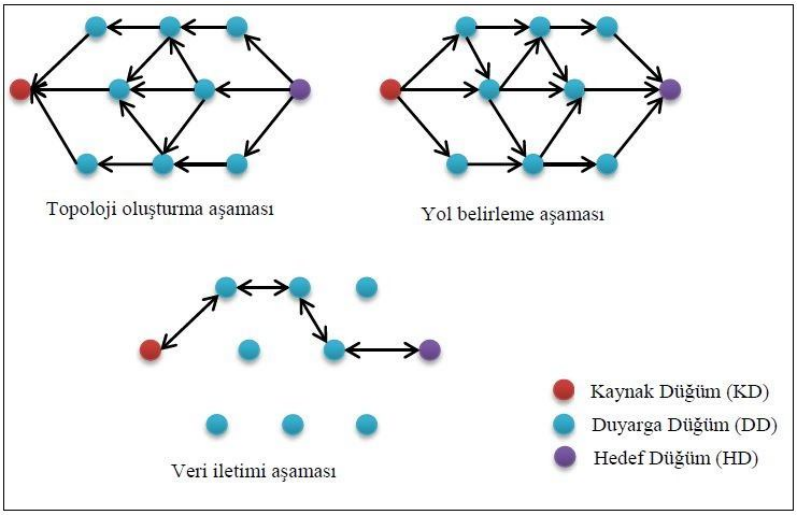
1.8.1.2. Görüşme Tabanlı Yönlendirme Protokolleri

Görüşme Tabanlı Yönlendirme Protokolünün (Sensor Protocolsfor Information via Negotiation - SPIN) çalışması ağ topolojisinin içerisindeki her algılayıcı düğümün hedef düğüm gibi çalışması prensibine dayanmaktadır. Bu protokolde bir algılayıcı düğüm veriyi tüm komşularına iletir. Komşu algılayıcı düğümler verinin geleceği an aktif olarak veriyi alır ve aynı şekilde kendi komşularına iletir. Bu sayede kaynak düğümden hedef düğüme kadar veri iletilmiş olur. Bu protokol algılayıcı düğümlerin kendisine gelen veriyi ileterek daha verimli bir ağ yapısı oluşturulacağı kabulüne dayanmaktadır. Kulaktan kulağa olarak adlandırılan bu yöntemde veri belirli bir algılayıcı düğüme değil de tüm komşulara iletildiği için büyük ağ yapılarında aşırı trafik oluşmasına neden olur. Oluşan trafik enerji sarfiyatını arttıra bilmektedir. SPIN'in bir sakıncası da birden fazla algılayıcı düğümün ortak komşulara aynı veriyi göndermesi durumunda ağ içerisinde tekrarlı verilerin dolaşmasına sebep olmasıdır. Bu da enerjinin gereksiz yere kullanılması anlamına gelmektedir. SPIN'de bu problem veri paketlerine bir ID değeri verilerek kısmen çözülmüştür. SPIN'de algılayıcı düğümlerin anlık enerji seviyelerine erişmek mümkündür. Bu sayede ağın enerji sarfiyatını inceleyerek ağın yaşam ömrü analiz edilebilir. SPIN'in en önemli sorunlarından birisi ise bilgi iletiminin garanti edilememesidir. SPIN'de algılayıcı düğümlerin belirli bir yöne değil de tüm komşulara veriyi yönlendirmesi verinin hedef düğüme ulaşacağı anlamına gelmemektedir. Özellikle büyük ağ yapılarında hedef düğüme çok uzak algılayıcı düğümlerin yönlendirdikleri veriler hedef düğüme komşuluk ilişkilerinde belirli bir yön bilgisi olmadığı için gönderilemeyebilir. Bu da ağın gereksiz yere kullanılmasını ve enerji sarfiyatını önemli derecede arttırır [31, 32].

1.8.1.3. Doğrudan Yayılma Protokolü

Doğrudan yayılma protokolü (DirectedDiffusion - DY) veri merkezli ve uygulama tabanlı bir protokoldür. Veri merkezli protokolün temel amacı farklı kaynaklardan gelen bilgileri birleştirmektir. Bu sayede gereksiz veriler elenerek ağın enerji sarfiyatının önüne geçmek mümkün olmaktadır. Protokol enerjinin etkin kullanımını baz almaktadır. Bu protokolde kaynak düğümlerden hedef düğüme gönderilecek veri için bulunan alternatif yollar bulunarak veri gönderilmektedir. Protokolün yapısında birden çok kaynak düğüm olabilir fakat hedef düğüm tektir [33].

DY protokolünde iletişim 3 aşamada gerçekleşmektedir. Birinci aşama ağın topolojisinin oluşması aşaması, ikinci aşama açı bilgilerine göre yolların belirlenmesi aşaması, son aşamada veri iletiminin gerçekleşmesi aşamasıdır. DY protokolünün veri iletimi ile ilgili model Şekil 14'de gösterilmiştir.



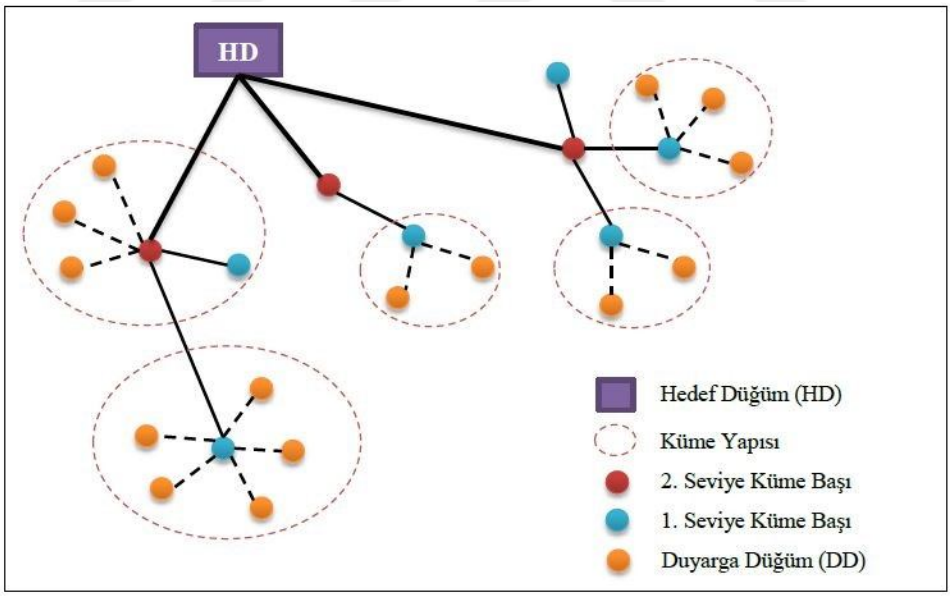
Şekil 14. Doğrudan yayılma protokolü yönlendirme yapısı

1.8.1.4. Söylenti Yönlendirme Protokolü

Söylenti yönlendirme protokolü (Rumor Routing - RR) coğrafi yönlendirmenin uygun olmadığı uygulamalarda tercih edilmektedir. Doğrudan yayılma (DY) protokolünün bir versiyonu olarak kabul edilebilir. RR enerji açısından DY protokolüne göre daha etkindir. Yapılan uygulamada coğrafi bir kriter olmadığı zaman ortama yayılan yayım mesajları ile yollar tespit edilmektedir. RR protokolü, KAA uygulamasında olayların sayısı az ise performanslı sonuçlar vermektedir. RR protokolünde her algılayıcı düğüm kendisine ait bir yönlendirme tablosu tutar. Bu tablodaki bilgileri komşularından aldığı bilgiler ile günceller. Hedef düğümden herhangi bir yönlendirme bilgisi gelmez ise kendi tablosundaki bilgilere göre yönlendirme gerçekleştirilir. Bu da enerji tüketimini arttırmaktadır. RR protokolü, büyük KAA uygulamalarında komşuluk ilişkileri ile ilgili algılayıcı düğüme çok fazla yük bindirdiği için enerji tüketimini arttırmakta bu da ağın toplam yaşam ömrünü olumsuz etkilemektedir. Bu yüzden RR protokolü büyük ağ yapılarında tercih edilmemektedir [34, 35].

1.8.1.5. Hiyerarşik Ağ Tabanlı Yönlendirme Protokolleri

Hiyerarşik yönlendirme (Hierarchical Networks Routing - HNR) veya küme tabanlı yönlendirme olarak adlandırılan bu gruba giren protokollerin temel amacı enerjinin etkin kullanılmasının sağlanmasıdır. HNR yapısında ağ içerisinde bulunan algılayıcı düğümlerin düşük enerji miktarına sahip olanlar sadece algılama yaparken, enerjisi yüksek olan algılayıcı düğümler ise algılama, bilginin işlenmesi ve yönlendirme amaçlı kullanılmaktadır. HNR'dealgılayıcı düğümler kümelere bölünmüştür. Bu kümeler içerisinde algılayıcı düğümler kümenin yaşam ömrünü uzatabilmek için enerji seviyelerine göre ya sadece algılama yaparlar ya da tüm fonksiyonları icra ederler. HNR'lerin bir avantajı da küme içerisinde verilerin işlenip gerekli olanların hedef düğüme iletilmesinin sağlanmasıdır. Bu da ağ içerisinde oluşacak gereksiz trafiğin önüne geçmek açısından önemlidir. Şekil 15'de HNR'lerin genel yapısı görülmektedir [29, 35].



Şekil 15. Hiyerarşik yönlendirme

1.8.1.5.1. LEACH Yönlendirme Protokolü

Kümeleme tabanlı düşük enerjili yönlendirme protokolü (LowEnergyAdaptive Clustering Hierarchy – LEACH) [36], küme tabanlı bir protokoldür. LEACH protokolünde

KAA içerisindeki algılayıcı düğümlerden rastgele birkaç algılayıcı düğüm küme başı olarak seçilmektedir. Yapılan araştırmalar ağ içerisindeki algılayıcı düğümlerin %5'inin küme başı olmasının yeterli olduğunu göstermiştir [28]. Küme başı olan algılayıcı düğümlerin asıl görevi küme içerisindeki algılayıcı düğümlerin enerjisinin eşit olarak kullanıla bilmesini sağlamaktır. Küme başı olan algılayıcı düğümlerin diğer bir görevi ise kendisine gelen bilgileri işleyip sıkıştırarak hedef düğüme iletilmesini sağlamaktır.

LEACH protokolü iki aşamada çalışmaktadır, Bunlardan birincisi ağ topolojisinin oluşturulması aşaması, ikincisi ise kalıcı durum olarak ta adlandırılan veri iletiminin sağlandığı aşamadır. Ağ topolojisinin oluşturulması aşamasında ağ içerisinde bulunan algılayıcı düğümlerden küme başı olacak algılayıcı düğümlerin belirlenmesi ve kümelerin oluşturulması işlemleri gerçekleştirilir. Kalıcı durum aşaması ise hedef düğüme veri iletiminin başlaması ve sürekliliğinin sağlanması işlemleri gerçekleştirilmektedir.

LEACH protokolü, tüm algılayıcı düğümlerin her aşamada eşit enerjiye sahip olduğunu ve algılayıcı düğümlerin tümünün eşit enerji harcadığını varsaymaktadır. Bu protokolün yapısı gereği birçok dezavantajı vardır. Bu dezavantajlardan birkaç tane altta sayılmaktadır.

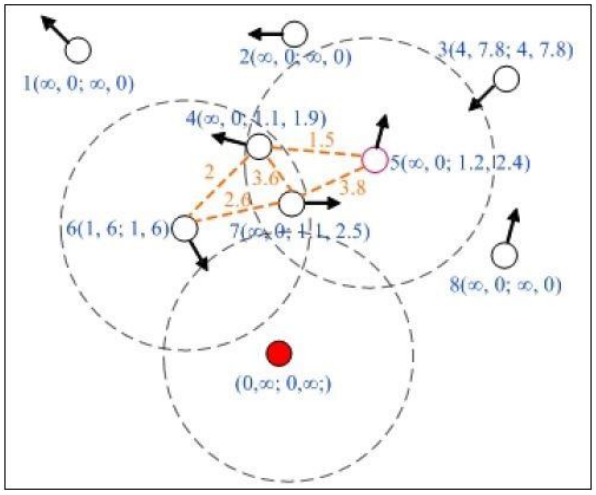
- LEACH tüm algılayıcı düğümlerin hedef düğüme ulaşabilmek için yeterli güçte iletim yapabileceğini varsayar. Rastgele dağıtılmış algılayıcı düğümler kimi zaman birbirinden çok uzak noktalarda olabilir ve veri iletimi yapamaya bilirler.
- Algılayıcı düğümlerin hep gönderilecek bir veriye sahip olduğu ve yakın komşularında da benzer verilerin olduğu varsayılır. Bu da küme başlarının gereksiz analizler yapmasına ve fazla işlem yükü yüzünden gereksiz enerji tüketimi yapmasına neden olur.
- Ağ topoloji oluşturulurken ağ içerisinde rastgele bir şekilde belirlenmiş küme başı algılayıcı düğümlerin ağ içerisinde homojen olarak dağılacağı kesin değildir. Hatta küme başları belirlendikten sonra bazı algılayıcı düğümler ağ içerisinde kendine bir küme başı bulamayabilir.

LEACH protokolünün bu sorunlarından dolayı özellikle ağ yapısı büyüdükçe kullanılabilir olmadığı görülmektedir [36].

1.8.1.6. Konum Tabanlı Yönlendirme Protokolleri

Konum tabanlı yönlendirme protokolleri (LocationBased Routing - LBR) KAA içerisinde yol belirleme ve verinin yönlendirmesi konularında algılayıcı düğümlerin konumlarının tespit edilerek komşuluk ilişkilerinin ortaya çıkmasını sağlayan protokollerdir. Bu protokollerde komşu algılayıcı düğümlerden gelen sinyal gücü incelenerek düğümlerin mesafesi hesaplanmaktadır [37, 38]. LBR'ler yapılan çalışmalarda sinyal gücünün yanı sıra algılayıcı düğüm ile ilgili yön tayini üzerine de çalışmalar yapılmaktadır. Konum tabanlı yönlendirme yapılırken;

- Komşu algılayıcı düğümlerin birbirleri ile yaptıkları veri iletişimi esnasında göreceli olarak koordinatları belirleme
- Algılayıcı düğümlerin konumunu uydular aracılığı ile belirleme
- Düğümler üzerinde bulunan küçük güçlü GPS alıcıları yolu ile konum belirleme yöntemlerinden biri kullanılarak yön tayini yapılmaktadır. Konum tabanlı yönlendirme protokolleri incelendiğinde yön tayini için kullanılan GPS veya uydu bağlantısı gibi yöntemlerden dolayı enerji tüketimlerinin düz ağ tabanlı yönlendirme veya hiyerarşik ağ tabanlı yönlendirme yöntemlerine göre fazla olduğu görülmüştür [39], [40]. Şekil 16'de LBR protokollerinin genel yapısı görülmektedir.



Şekil 16. Lbr protokollerinin yapısı

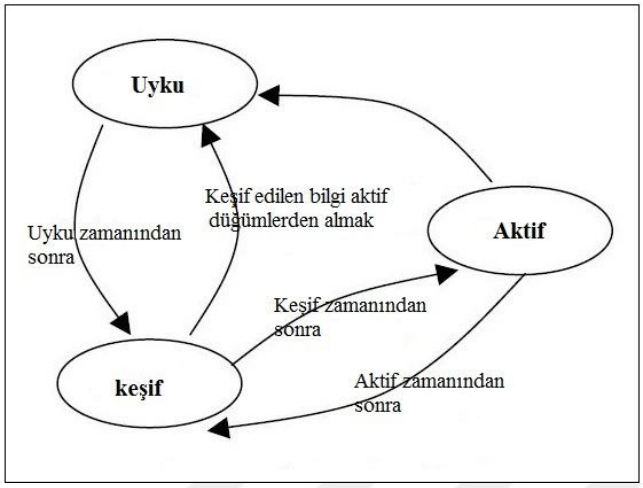
1.8.1.6.1. Coğrafik Uyarlamalı Doğruluk Protokolü

Coğrafik Uyarlamalı Doğruluk Protokolü (Geographic Adaptive Fidelity – GAF) [41], özellikle MANET'ler (Mobile Ad-Hoc Network) için enerji etkin yönlendirme protokolü olarak sunulmuş sonrasında ise KAA'lara uygulanmıştır. GAF protokolünün temeli paketlerin alınması veya gönderilmesinin yanı sıra algılayıcı düğümlerin uyku veya aktif olma durumlarının tespiti için oluşturulan modele dayanmaktadır.

GAF protokolünde algılayıcı düğümlerin keşif, uyku ve aktif olmak üzere üç konumu bulunmaktadır. Keşif durumu ağ içindeki diğer algılayıcı düğümler hakkında bilgi edinerek ağ topolojisinin oluşturulması için kullanılmaktadır. Bir algılayıcı düğüm uyku moduna girdiğinde radyolarını kapatarak enerji tasarrufu yapmaktadır. Aktif durumdaki algılayıcı düğüm ise veri iletimi için hazır aktif olduğunu komşu algılayıcı düğümlere bildirmek için periyodik olarak yayın yapar.

GAF protokolünde algılayıcı düğümlerin aktif olarak kaldığı süre uygulamanın yapısı, algılayıcı düğümlerin hareketliliği gibi birçok faktöre göre değişebilmektedir. GAF amacına uygun olarak kaynak düğüm ile hedef düğüm arasında gerçekleşecek iletişimde kullanılacak algılayıcı düğümler haricindeki algılayıcı düğümlerin uyku modunda kalmasını sağlayan bir mekanizma sunmaktadır. Bu yüzden GAF protokolünde ağ ortamı ızgaralara bölünerek her algılayıcı düğüm, GPS verilerine göre gruplara bölünerek kaynak düğüm ile hedef düğüm arasında yollar oluşturulmuştur. Böylece GAF protokolü ile kaynak düğüm ile hedef düğüm arasındaki iletişim için atanan yoldaki algılayıcı düğümlerin haricindeki algılayıcı düğümlerin uykuda kalarak enerji verimliliği sağlanmıştır. Şekil 17'de GAF protokolünde durum geçişleri gösterilmektedir[41].

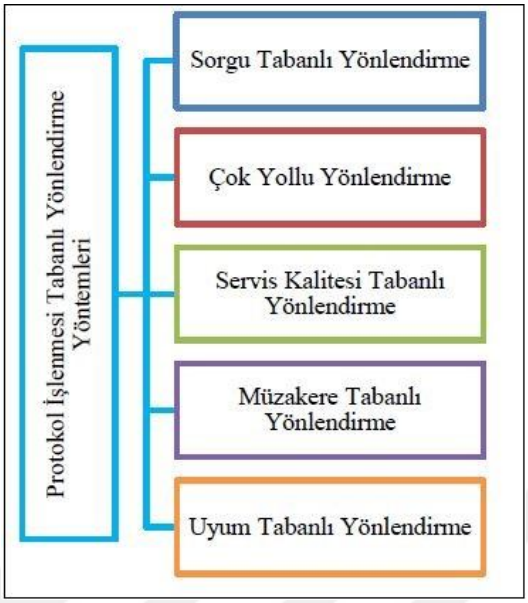
GAF protokolü ağ üzerinde iletişimin gerçekleştiği yol üzerindeki algılayıcı düğümleri aktif diğer tüm algılayıcı düğümleri uyku konumuna alarak ağ yaşam ömrünü maksimize etmektedir. Bu yüzden algılayıcı düğümleri enerji düzeyleri esas alınmaktadır. Böylece enerji seviyesi yüksek olan algılayıcı düğüm üzerinden iletişimin sağlanması mümkün olmaktadır.



Şekil 17. GAF protokolünde durum geçişleri

1.8.2. Protokol İşlenmesi Tabanlı Yönlendirme

Protokol işlenmesi tabanlı yönlendirme (Protocol OperationBased Routing – PO) KAA'larda yapılan çalışmalar incelendiğinde geliştirilen yönlendirme protokollerinin yanı sıra ağ yapısındaki birden çok katman ile iletişimi sağlamaya yönelik yöntemler veya algoritma tabanlı çalışmalar konusunda da yoğunlaşıldığı gözlemlenmektedir. Bu çalışmalar genellikle mevcut protokollerin geliştirilmesi veya birden çok protokol yapısının birlikte kullanıldığı çalışmalar veya yönlendirme ile ilgili tüm iletişim süreçlerinin yapılandırıldığı yazılım tabanlı çalışmalardır. Bu tarz yönlendirme yöntemleri genellikle yönlendirme protokollerinin tek başlarına yeterli olmadığı ve ağ ortamına göre özel düzenlemelerin yapılması gerektiği yapılarda tercih edilmektedir. PO yönlendirme işlemini yöntemlerine göre 5 ayrı başlık altında toplamak mümkündür [42]. Şekil 18'de bu yöntemler gösterilmektedir.



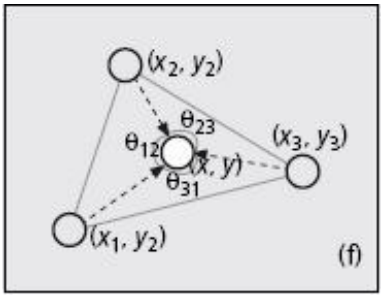
Şekil 18. Protokol işlenmesi tabanlı yönlendirme yöntemleri

1.9. Kablosuz Algılayıcı Ağlarda Konumlandırma

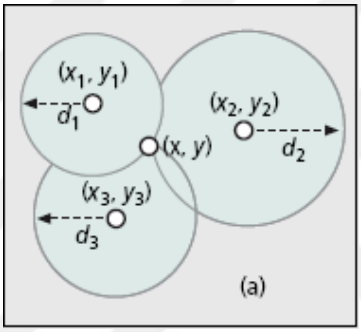
Algılayıcı ağların kullanıldığı bütün uygulamalarda, algılayıcı düğümlerin çalışmalarının doğru bir şekilde idare edilebilmesi için bu ağların yer belirleme yeteneğine sahip olmaları gerekmektedir. Bu durum ise algılayıcı ağlarda yer belirleme problemini ortaya çıkarmıştır. Literatürde bu sorunun çözümü için değişken veya sabit olan ağ düğümlerinin yer belirlemeleri için iki ayrı yaklaşım varsayılmış ve bunlara dayalı yöntemler gerçekleştirilmiştir. Bunlar mesafe bağımlı ve mesafe bağımsız yer belirleme yaklaşımlarıdır [43, 44].

1.9.1. Mesafe Bağımlı Yer Belirleme Yöntemleri

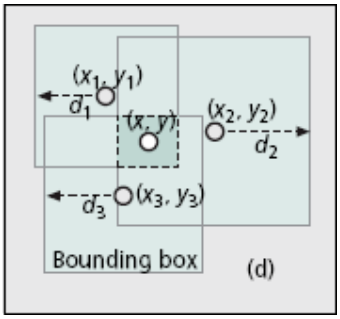
Yöntemin temelinde alıcı ve verici düğümleri arasındaki mesafe ölçümü önemlidir. Ölçümde kullanılan birimler temel geometrik yapılardır. Bunlar Şekil 19, 20 ve 21' de görüldüğü gibi üçgen, daire ve dikdörtgen olarak seçilebilir. Genel olarak kabul görmüş mesafe bağımlı konum tespit yöntemleri: Alınan Sinyal Gücü (RSSi), Uçuş Zamanı (ToF) ve bu iki yöntemin ortak kullanıldığı Calamari'dir [41, 42].



Şekil 19. Üçgen algoritmalı mesafe bağımlı yer belirleme yöntemi



Şekil 20. Dairesel algoritmalı mesafe bağımlı yer belirleme yöntemi



Şekil 21. Dikdörtgen algoritmalı mesafe bağımlı yer belirleme yöntemi

1.9.1.1. Alınan Sinyal Gücü Göstergesi (Received Signal Strength - RSSi) Belirleme Yöntemi

Sinyal gücü ölçümüne dayanan yöntemde amaç; gücü belli verici yayınının, alıcı üstündeki şiddetine göre mesafe ölçmektir. Alıcı üstünde alınan sinyal gücü, sinyalin ne

kadar mesafede ne kadarlık bir zayıflama gösterdiği bilgisi ile mesafe bilgisine dönüştürülür. Belirlenen ölçüm/mesafe şablonuna göre konum bilgisi belirli hata sınırları dâhilinde saptanabilir. Konum saptamak için gerekli en az üç (3) alıcı düğümü gereklidir. Kullanılacak geometrik şekle göre her bir düğümün merkez düğüme olan uzaklığı hesaplanıp, kesişim noktalarının taradığı alan bulunur. Bu taralı alan aynı zamanda hata değerini de içeren konum bilgisini vermektedir [43].

1.9.1.2. Uçuş Zamanı (Time of Flight) Belirleme Yöntemi

Sinyalin vericiden çıkışından alıcıya gelişine kadar geçen sürenin ölçümüne dayanan yöntemde amaç; hızı belli verici çıktısının zaman olarak eşgüdümlü çalıştığı alıcı üniteye ulaşmasına kadar geçen süreyi ölçmektir. Günümüzde kullanılan küresel konumlama sistemi (GPS) de radyo sinyal hızının ölçümüne dayanan time of flight yöntemi ileçalışırlar. Derinlik gibi fazladan bir değere sahip olan küresel konumlama sisteminde konum tespiti için gerekli düğüm sayısı dörde (4) çıkmaktadır. Çalışma yapısında da eşgüdümlü olarak gönderilen sinyaller alıcıya farklı zamanlarda ulaşır. Bu zaman farklarına göre alıcı düğüm kendi konumunu hassas bir şekilde tespit edebilmektedir [43].

1.9.1.3. Calamari Belirleme Yöntemi

Sinyal güç ölçümü ve sinyal ulaşım süresi yöntemlerini birleştiren yapıda amaç; oluşabilecek ölçümleme hatalarını yaşamamak adına yapıdaki bir verici düğümün kendini referans ilan etmesidir. İlan gerekli dönüşüm şablonları içerir ve gerekli zamanlarda tekrar ilanı ile ölçüm hataları minimuma indirgenir [45].

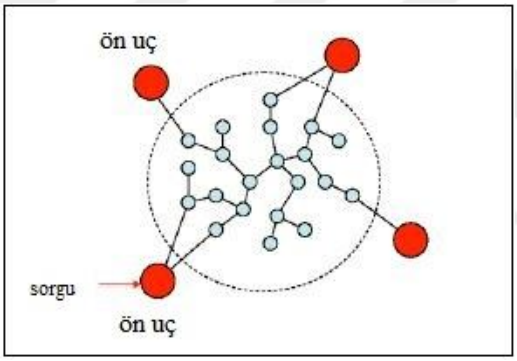
1.9.2. Mesafe Bağımsız Konum Belirleme Yöntemleri

Yöntemin temelindeki mantık; alıcı ve verici düğümleri arasındaki sinyal haberleşmeleri ve bağlantılardır. Düğümlerin kapsam alanlarına ve bu alan içindeki haberleşmelerine dayanan yöntemler ile konum belirlemesi gerçekleştirilir. Belirlemedeki temel geometrik yapı çemberdir. Genel olarak kabul görmüş mesafe bağımsız konum tespit

yöntemleri: Beni Duyuyor musun? (Do youhear me? - DYHM), Çoklu atlama (Multi-hop) ve Centroid'dir [45].

1.9.2.1. Beni Duyuyor musun? ve Çoklu Atlama Konum Belirleme Yöntemi

Beni duyuyor musun? Ve çoklu atlama (Multi-hop) yöntemlerinde amaç; ağ yapısının haritasını çıkartmaktır. Fakat çoklu atlama (Multi-hop) yöntemde sadece komşu ilanı yerine, her bir düğümün iletişim kurabildiği ve kuramadığı komşular diğer düğümlerle karşılaştırılır. Böylece konum mümkün oldukça gerçeğe yakın haritalanır Şekil 22'de Beni duyuyor musun? Yöntemi görülmektedir [46].



Şekil 22. Beni duyuyor musun? Yöntemi

1.9.2.2. Centroid Belirleme Yöntemi

Konumu sabit ve bilinen düğümlerden, konumu bilinmeyen düğümün belirlemesi yöntemidir. En az üç (3) düğümden gelen verilere göre konum değerlendirilir. Diğer yöntemlerinden farklı olarak ölçümü gerçekleştiren düğümlerin sabit ve belirli konumlar içermesidir. Böylece konumu değişken düğümün daha iyi konum tespiti gerçekleştirilir. Bu yöntemde gezici düğüm her mevkii değiştirdiğinde göndereceği bilgiler ile konumu sabit düğümlere olan mesafeleri belirlenir. Böylece konum tespiti tam olarak gerçekleştirilebilir. Yöntemin yapısı gereği gezici düğümün alıcı olma durumu da mümkündür. Böylece sabit düğümlerden gönderilecek bilgiler ile gezici düğümün konumunu kendisinin tespiti de mümkündür [46].

1.9.3. Yöntemlerin Değerlendirmeleri

Belirtilen yöntemler içinde mesafe bağımlı yöntemler konum saptama başarıları ile ön plana çıkmaktadır. Alınan sinyal gücüne göre konum saptama gerçekleştiren RSSi yöntemi hava şartlarına bağlılığın daha az olduğu ortamlara uygunlukları ile ön plana çıkarken; sinyalin ulaşma süresini ölçen uçuş zamanı (ToF) yöntemi zaman sayacındaki programlama zorluğuna rağmen ultrasonik alıcı vericilerin çalıştığı kısa mesafelerde verimli bir şekilde konum tespiti yapabilmektedir. İki yöntemi de kapsayan Calamari yöntemi ise güçlü hata filtreleme formülleri ile orta mesafe ölçümlerinde öne çıkmaktadır.

Düğüm sayısının arttığı ve referans alınabilecek düğüm sayısının azaldığı durumlar için mesafe bağımsız konum saptama teknikleri geliştirilmiştir. Bu tekniklerden Beni Duyuyor musun? Yöntemi referans alınabilecek düğümün olmadığı durumlarda düğüm haritasını çıkarta bilmektedir. Konum saptamadaki hata büyüklüğünün çok önemli olmadığı habitat izleme gibi durumlarda kullanımı uygundur [47].

1.10. Optimizasyon

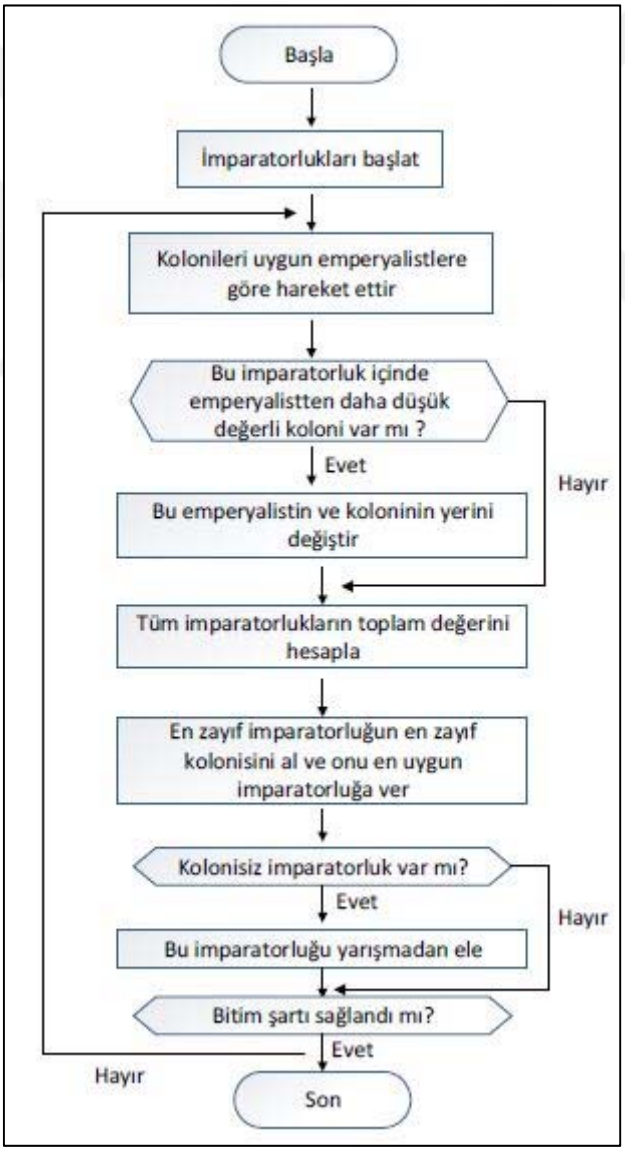
Optimizasyon, en iyileme anlamına gelmektedir. Bir problem için, verilen şartlar altında tüm çözümler arasından en iyi çözümü elde etme işidir. Belirli sınırlamaları sağlayacak şekilde, bilinmeyen parametre değerlerinin bulunmasını içeren herhangi bir problem, optimizasyon problemi olarak adlandırıla bilmektedir [48].

Optimizasyon problemlerinin değişik yapıları ve matematiksel karakteristikleri, nanlineerlik, süreklilik, dış bükeylik gibi ortak özellikleri olan özel problemlerin çözümünde kullanılacak algoritmalar geliştirilmesini gerektirmiştir. Günümüzde, çok çeşitli problem türlerinin çözümü için geliştirilmiş algoritmalar mevcuttur.

1.10.1. Yayılmacı Yarışmacı Algoritma (Imperalist Competitive Algorithm)

Atashpaz-Gagari ve Lucas [49] tarafından önerilen yayılmacı yarışmacı algoritma da diğer evrimsel algoritmalar gibi bir başlangıç popülasyonu ile başlar. Bu algoritmada, popülasyondaki her bir eleman bir ülke olarak adlandırılır. Popülasyondaki ülkeler iki yayılmacı ve koloni bölüme ayrılır. Popülasyondaki birkaç en iyi ülke yayılmacı olmak

için seçilir ve kalanlar da bu yayılmacıların kolonisi olur. Tüm koloni ülkeler bu yayılmacı ülkeler arasında dağıtılır, Her hangi yayılmacı ülke gücüne bağlı koloni ülkelere sahip olur ve bu ülkeleri kontrol eder. Tüm koloniler uygun yayılmacılara doğru hareket etmeye başlar. Sonra imparatorlar arasında yarış başlar. Eğer bu yarışta bir imparatorluk gücünü artırmazsa yarıştan elenir. Bu yarışta güçlü imparatorluklar zayıf imparatorlukları gücünü azaltarak kendi gücüne güç katıp ve zayıf imparatorlukları sonunda yıkılır. Bu yarışın sonunda tek bir imparatorluk kalır ve diğer ülkeler bu imparatorluğun bir kolonisi olur. Bu durum yayılmacı yarışmacı algoritmanın amaç fonksiyonun optimum noktasıdır [49]. Şekil 23'de algoritmanın akış şeması görülmektedir.



Şekil 23.Yayılmacı yarışmacı algoritmaakış şeması

Yayılmacı yarışmacı algoritma yeni bir algoritma olduğu için, yapılan çalışma sayısı da bu yöntem ile oldukça azdır. Analog devre optimizasyonu [50], birliktelik kurallarının keşfi [51], gezici robotun global konumlandırılması [52], kablosuz algılayıcı düğüm ağının yerleştirilmesi [53], çevrimiçi PI kontrolü [54], DC motorun hız kontrolü [55], uyarlanabilir yayılmacı yarışmacı algoritma [56] ve kaotik temelli yapay sinir ağının eğitimi [57] yayılmacı yarışmacı algoritma ile yapılmış bazı çalışmalardır.

1.10.1.1. Başlangıç İmparatorluklarını Üretme

Optimizasyonun amacı problem değişkenlerinden en optimal çözümü bulmaktır. Değer değişkenleri için bir dizi oluşturulur. Genetik algoritmada buna "kromozom" adı verilirken yayılmacı yarışmacı algoritmada "ülke(country)" adı verilir. Bir Nvar boyutlu optimizasyon probleminde bir ülke bir 1x Nvar dizisidir. Bu dizi Denklem 1'deki gibi tanımlanır.

$$contry=[u_1,u_2,\ldots,u_{Nvar}] \tag{1}$$

Ülkedeki değişken değerleri noktalı kayan sayılarla temsil edilir. Bir ülkenin maliyeti(Cost) $u_1,u_2,\ldots,u_{Nvar}$ değişkenlerindeki değer fonksiyonu ile ölçülür.

$$cost = f(contry) = f(u_1, u_2 , \ldots , u_{Nvar}) \tag{2}$$

Optimizasyon algoritmasına başlamak için N_{pop} boyutundaki başlangıç popülasyonu oluşturulur. İmparatorlukları oluşturmak için N_{imp} sayıdaki en güçlü ülkeler seçilir. Geri kalan N_{col} ise bir imparatorluğa ait olan kolonilerdir. Bu şekilde yayılmacı ve koloni olmak üzere 2 tip ülkeye sahip olunur.

Başlangıç imparatorluklarını oluşturmak için koloniler, yayılmacılar arasında yayılmacıların güçlerine göre dağıtılır. Böylece yayılmacıların başlangıç koloni sayısı gücü ile doğru orantılı olur. Kolonileri yayılmacılar arasında doğru orantılı dağıtmak için, bir yayılmacının normalize edilmiş maliyeti Denklem 3'te gösterildiği gibi tanımlanır:

$$C_n=c_n -maks_i\{ci\} \tag{3}$$

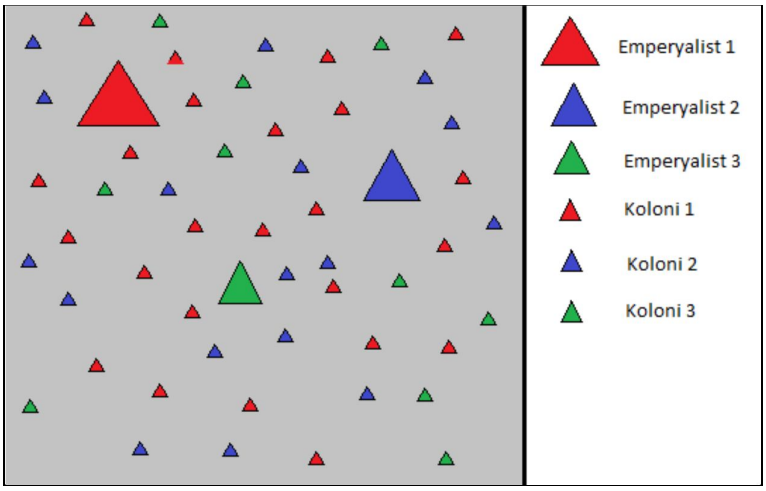
C_n, n. yayılmacının maliyeti ve $c_n$ise normalize edilmiş maliyetidir. Tüm yayılmacıların normalize edilmiş maliyetlerini içeren, her yayılmacının normalize edilmiş gücü ise Denklem 4'te gösterildiği gibi tanımlanır:

$$P_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \tag{4}$$

Bir başka bakışla, yayılmacının sahip olduğu koloniler kısmı, bir yayılmacının normalize edilmiş gücüdür. Böylece bir imparatorluğun kolonilerinin başlangıç sayısı Denklem 5"teki gibi olacaktır.

$$N.C._n = round\{p_n . N_{col}\} \tag{5}$$

$N.C._n$ n.İmparatorluğun başlangıç koloni sayısıdır. N_{col}tüm kolonilerin sayısıdır. Kolonileri yayılmacılara dağıtmak için gelişigüzel $N.C._n$koloni seçilir ve yayılmacıya verilir. Bu koloniler yayılmacı ile birlikte n.imparatorluğu oluşturur. Şekil 24 her imparatorluğun ilk popülasyonunu gösterir ve daha güçlü imparatorlukların daha çok koloniye sahip olduğu görülebilir (Abdechiri vd., 2010:941) [56].



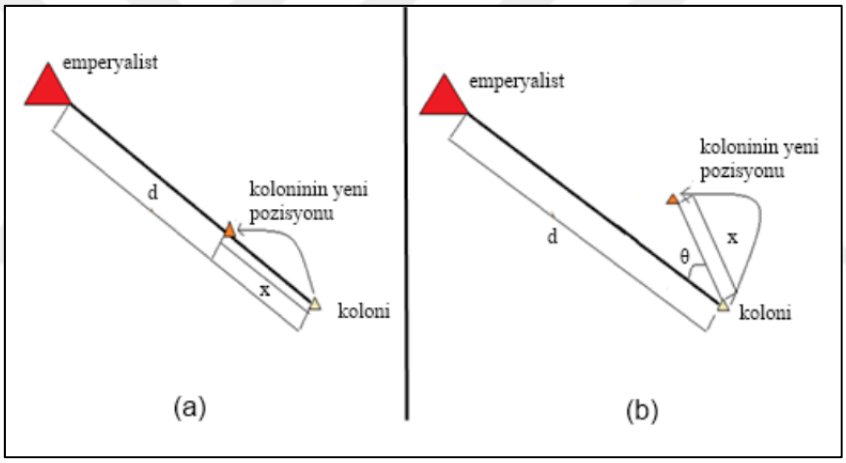
Şekil 24. İmparatorluğun ilk popülasyonu

1.10.1.2. Kolonilerin Hareketi

Zamanla yayılmacı ülkeler kolonilerini arttırmaya başlar. Bu durum kolonilerin yayılmacıya doğru hareketi şeklinde gerçekleşir. Şekil 25 (a)'da koloninin x birim yayılmacıya hareketi gösterilmektedir. Hareketin yönü koloniden yayılmacıya doğru bir vektördür. Şekildeki $x$rastgele bir değerdir.

$$X \sim U(0,\ \beta \times d\ ) \tag{6}$$

β1 den büyük bir sayı ve d ise aradaki uzaklıktır. $B > 1$ Olması koloninin yayılmacıya yaklaşmasına neden olur.



Şekil 25. a) Koloninin yayılmacıya hareketi b) Koloninin yeni pozisyonu

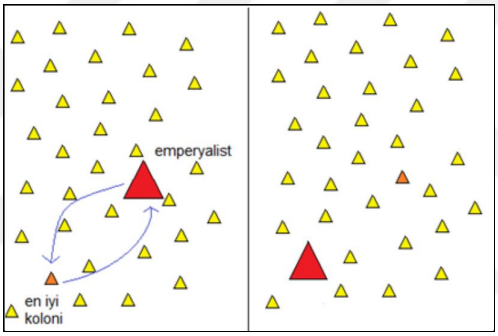
Yayılmacının etrafında farklı noktalar aramak için hareketin yönüne gelişigüzel bir değerde sapma eklenir. Şekil 25(b)'de θ gelişigüzel bir değerdir.

$$\theta \sim U(-\gamma, \gamma) \tag{7}$$

γ Orijinal yönden sapmanın değerini ayarlayan parametredir. Bununla beraber β ve γ değerleri keyfidir [49].

1.10.1.3. Yayılmacı ve Koloninin Yerini Değiştirme

Bir koloni yayılmacıya doğru ilerlerken, yayılmacının maliyetinden daha düşük maliyetli bir konuma erişebilir. Böyle bir durumda yayılmacı koloni ile yer değiştirir. Daha sonra algoritma, yayılmacının yeni pozisyonu ve kolonilerin bu yeni pozisyona hareketi ile devam eder. Şekil 26, koloni ile yayılmacının yer değiştirmesini gösterir. İmparatorluğun en iyi kolonisi koyu renk ile gösterilmiştir. Bu koloni yayılmacıdan daha düşük maliyete sahiptir. Şekil 26'de imparatorluğun, yayılmacı ile koloninin yer değiştirmesinden sonraki durumu da gösterilmiştir [49].



Şekil 26. Koloni ile yayılmacının yer değiştirmesi

1.10.1.4. İmparatorluğun Toplam Gücü

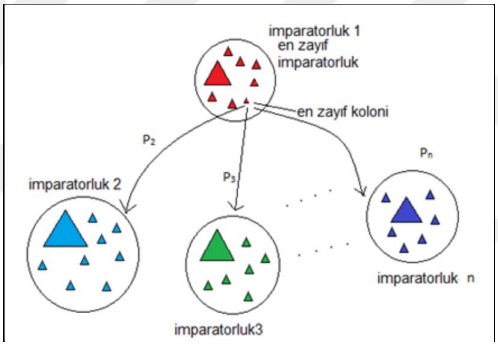
Bir imparatorluğun toplam gücü çoğunlukla yayılmacı ülkenin gücünden etkilenir. Fakat imparatorluktaki kolonilerin gücü göz ardı edilebilir olsa da bu imparatorluğun toplam gücü üstünde bir etkiye sahiptir. Bu durum Denklem 8'de gösterildiği gibi toplam maliyeti tanımlayarak modellenmiştir:

$$T.C._n = Cost(imprialist_n) + \xi\ mean\{Cost(colonies\ of\ empire_n)\} \quad (8)$$

$n.$ imparatorluğun toplam maliyeti $T.C._n$'dir ve ξ 1'den küçük olan pozitif bir sayıdır. ξ için küçük bir değer sadece yayılmacı tarafından belirlenen imparatorluğun gücünü etkiler ve bu değeri arttırmak imparatorluğun toplam gücüne karar vermede kolonilerin rolünü arttırır. 0,1 değeri birçok uygulamada ξ için kullanılmıştır [49].

1.10.1.5. Yayılmacılık Yarış

Bütün imparatorluklar diğer imparatorlukların kolonilerini ele geçirmeye çalışır ve bunları kontrol eder. Bu yayılmacılık yarış gitgide zayıf imparatorlukların gücünde azalmayı ve daha güçlü olanların gücünde artışı beraberinde getirir. Bu yarış, zayıf imparatorlukların zayıf kolonilerinden bazılarının alınması ve diğer imparatorlukların bu kolonileri ele geçirmeye çalışmaları şeklinde modellenmiştir. Şekil 27, modellenmiş yayılmacılık yarışın büyük bir resmini gösterir. Bu yarışta imparatorlukların her biri zayıf kolonileri ele geçirme ihtimaline sahiptir. Fakat güçlü imparatorlukların bu kolonileri ele geçirme olasılıkları daha fazla olacaktır.



Şekil 27. Yayılmacılık yarışma

Yarışa başlamak için ilk olarak her imparatorluğun toplam gücüne göre ele geçirme olasılığı bulunmalıdır.

$$N.T.C._N = T.C._n - max_i\{T.C._i\} \quad (9)$$

Denklem 9'daki$T.C._n$n. imparatorluğun toplam maliyeti ve $N.T.C._n$ normalize edilmiş toplam maliyetidir.

$$p_{pn} = \left|\frac{N.T.C._n}{\sum_{i=1}^{N_{imp}} N.T.C._i}\right| \quad (10)$$

Bahsi geçen kolonileri imparatorlukların sahip olduğu ele geçirme olasılıkları arasında dağıtmak için, Denklem 11'de gösterilen P vektörü oluşturulur.

$$P = [p_{p1}, p_{p2}, \dots, p_{pn_{imp}}] \quad (11)$$

Sonra P ile aynı boyutta gelişigüzel sayılardan oluşan R vektörü oluşturulur.

$R=[r_1,r_2,\ldots,r_{Nimp}]$

$r_1, r_2, \dots, r_{Nimp}$ ~U$(0,1)$ (12)

Daha sonra P'den R'yi çıkarma işlemi Denklem 13'de gösterilmiştir:

$$D = P - R = [\,D_1,D_2,\ldots,D_{Nimp}]=[P_{P1} - r_1\ , P_{P2} - r_2\ ,\ldots,\ P_{Pnimp} - r_{nimp}] \quad (13)$$

Vektör D'ye dayanarak, D'nin maksimum indeksi ile alakalı olan imparatorluğun kolonileri elde edilir [49].

1.10.1.6. Bir Noktada Birleşme

Güçsüz imparatorluklar yayılmacılık yarışta geride kalır ve çöker. Onun kolonileri ise diğer imparatorluklar arasında paylaştırılır. En güçlü imparatorluk dışındaki bütün imparatorluklar çöktükten sonra tüm koloniler tek bir imparatorluğun kontrolüne girer. Bu ideal yeni dünyada aynı pozisyon ve maliyette olan koloniler, onlarla aynı pozisyon ve maliyete sahip olan bir yayılmacı tarafından kontrol edilecektir. Böyle bir durumda yayılmacı yarışa son verilmeli ve algoritma durdurulmalıdır [49].

1.10.2. Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)

Parçacık Sürü Optimizasyonu (PSO), Kenedy ve Eberhart tarafından sürü halinde hareket eden hayvanlardan esinlenerek geliştirilmiş bir optimizasyon yöntemidir [58, 59]. Bu algoritma temel olarak sürü zekâsına dayanan bir algoritmadır. Sürü halinde hareket eden balıklar böcekler gibi hayvanların yiyecek ve güvenlik gibi durumlarda, çoğu zaman rastgele sergiledikleri hareketlerin, amaçlarına daha kolay ulaşmalarını sağladığı görülmüştür. PSO bireyler arasındaki sosyal bilgi paylaşımını esas alır. Arama işlemi genetik algoritmalarda olduğu jenerasyon sayısınca yapılır. Her bireye parçacık denir ve

parçacıklardan oluşan popülasyona da sürü (swarm) denir. Her bir parçacık kendi pozisyonunu, bir önceki tecrübesinden yararlanarak sürüdeki en iyi pozisyona doğru ayarlar. PSO, temel olarak sürüde bulunan bireylerin pozisyonunun, sürünün en iyi pozisyona sahip olan bireyine yaklaştırılmasına dayanır. Bu yaklaşma hızı rastgele gelişen durumdur ve çoğu zaman sürü içinde bulunan bireyler yeni hareketlerinde bir önceki konumdan daha iyi konuma gelirler ve bu süreç hedefe ulaşıncaya kadar devam eder[58, 59]. Şekil 28'de PSO'nun akış diyagramı görülmektedir.

Şekil 28. Parçacık sürü optimizasyonu akış diyagramı

Algoritma temel olarak aşağıdaki basamaklardan oluşur;

i. Rastgele üretilen başlangıç pozisyonları ve hızları ile başlangıç sürüsü oluşturulur.

ii. Sürü içerisindeki tüm parçacıkların uygunluk değerleri hesaplanır.

iii. Her bir parçacık için mevcut jenerasyondan yerel en iyi (pbest) bulunur. Sürü içerisinde en iyilerin sayısı parçacık sayısı kadardır.

iv. Mevcut jenerasyondaki yerel eniyiler içerisinden küresel en iyi (gbest) seçilir.

v. Pozisyon ve hızlar aşağıdaki gibi yenilenir.

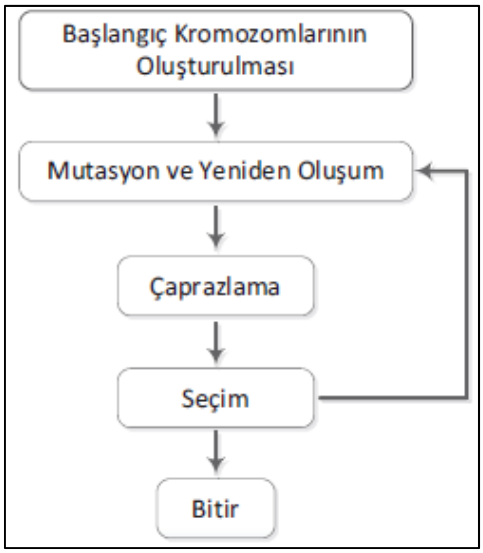
$$V_{id} = W * V_{id} + c_1 * rand_1(P_{id} - X_{id}) + c_2 * rand_2 * (P_{gd} - X_{id}) \quad (14)$$

$$X_{id} = X_{id} + V_{id} \quad (15)$$

Denklem 14 ve 15'te X_{id} pozisyon ve V_{id} hız değerlerini verirken, $rand_1$ve $rand_2$ve değerleri rasgele üretilmiş sayılardır. WAtalet ağırlık değeri ve C_1, C_2, ölçeklendirme faktörleridir.

1.10.3. Diferansiyel Gelişim Algoritması (Differential Evolution)

DGA, Price ve Storn tarafından 1995 yılında geliştirilmiş, işleyiş açısından genetik algoritmaya dayanan popülasyon temelli sezgisel bir optimizasyon tekniğidir. Özellikle sürekli parametreli problemlerin çözümüne yönelik geliştirilmiştir. Popülasyona dayalı çalışmayıp, tek tek kromozomlar operatörlere tabi tutularak yeni bir birey elde edilmektedir. Bu işlem sırasında mutasyon ve çaprazlama operatörleri kullanılmaktadır. Yeni bireyin uygunluğu eskisinden daha iyi ise yeni birey, aksi takdirde eski birey bir sonraki jenerasyona(nesile) aktarılmaktadır. Bu algoritmanın kolay kodlanabilmesi diğer algoritmalara göre üstünlüğü olarak belirtilebilir. Algoritmanın basit bir şeması Şekil 29'de verilmiştir [60, 61].



Şekil 29. DGA'nın akış diyagramı

Algoritma temel olarak aşağıdaki adımlardan oluşur

i. Rastgele üretilen başlangıç kromozom popülasyonu, Popülasyonun büyüklüğü (NP) üçten büyük olmalıdır. Çünkü DGA'da yeni kromozomların üretilmesi için mevcut kromozom dışında üç adet kromozom daha gereklidir.

ii. Bu adımda mutasyon işlemi yapılır, kromozomun genleri üzerinde tesadüfi değişiklikler yapılır. Mutasyon konusunda kromozom dışında ve birbirlerinden farklı olan üç kromozom seçilir. İlk ikisinin farkı alınır ve F ile çarpılır. F genellikle 0-2 arasında değerler almaktadır. Ağırlıklandırılmış fark kromozomu ile üçüncü kromozom toplanır ve yeni bir kromozom oluşur [62].

iii. Çaprazlama adımı, Mutasyon sonucunda elde edilen fark kromozomu ve ilk kromozomu kullanılarak yeni kromozom üretilir. Deneme kromozomu için genler CR olasılıkla fark kromozomundan, (1-CR) olasılıkla mevcut kromozomdan seçilir. Ve sonra en az bir tane genin üretilen yeni kromozomdan alınmasını garanti etmek için, $j = j_{rand}$ koşulu kullanılmaktadır. Rastgele seçilen$j = j_{rand}$ noktasındaki gen CR değerine bakılmaksızın mutasyon'den seçilir [62].

iv. Değer fonksiyonu, Üç farklı kromozom birer tane mutasyon ve çaprazlama operatörleri ve hedef kromozomla birlikte kullanılarak yeni bir kromozom (deneme kromozomu) elde edilmiştir. Yeni nesile $(g = g + 1)$ aktarılacak olan kromozom uygunluk değerine bakılarak belirlenir. Problemin amaç fonksiyonu değeri hesaplanır [62].

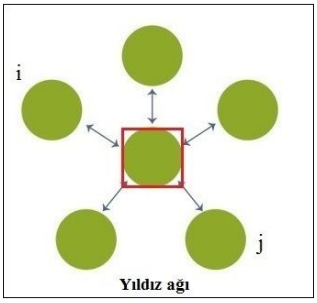
v. Seçim, Kromozomlardan değer yüksek olan kromozom yeni nesile aktarılır. Döngü $g = g_{max}$ olana kadar devam eder, g_{max} olduğunda mevcut en iyi birey çözüm olarak alınır [62].

2. YAPILAN ÇALIŞMALAR

Bu tezde, algılayıcı ağlarda kapasiteli sunucuların konumlandırma probleminin çözümü üzerine bir çalışmadır. Bu problem np-zor kombinatoryel problemlerin grubunda yer almaktadır. Problemin çözümü ile hem adaptif kümeleme hem de optimum küme başı seçimi gerçekleştirilmiştir. Bu çalışmada küme başlarının seçimi yapay zeka PSO, GA ve ICA gibi optimizasyon teknikleri ile yapılmış ve aynı anda her iterasyondaki amaç fonksiyonun değerine bağlı kümeleme ve ağın konfigürasyonu değişmektedir. Burada önerilen yöntem kablosuz algılayıcı ağları için iteratif, akıllı ve adaptif bir çözüm sağlanmaktadır.

2.1. Ağ yapısı

Varsayalım bir alanda algılayıcı düğümler dağıtılmıştır. Bu düğümlerin arasında bağlantı kurmak istiyoruz. Bu yönde kullanılan iki yöntem vardır. Birinci yöntemde bir merkezi baş düğüm seçilir ve düğümlerin tamamına bu merkezden bilgi gönderme ve alma imkânları vardır. Bu yönteme yıldız ağı denir. Şekil 30. bir yıldız ağını i düğümü j düğüme veri göndermesini göstermektedir.



Şekil 30. Yıldız ağında i düğümünden j düğümüne veri gönderme

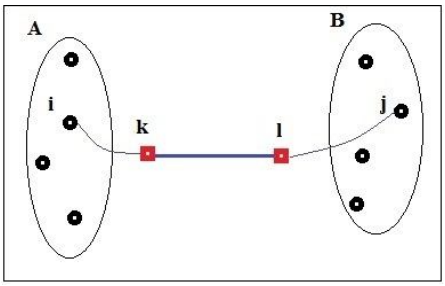
Bu metotta örneğin i' dan j'e ve tersine j den i'e bilgi göndermek için ilk bilgi i'den baş düğüme sonra baş düğümden j düğümüne gönderilmektedir. Eğer j den bir cevap gönderme gereği varsa bu bilgi baş düğüm vasıtasıyla i'e gönderilmektedir. İkinci

yöntemde düğümler birbirine bilgi aktarmaktadır yani düğümlerin tamamı baş düğümdür ve aralarında direk iletişim vardır. Bu yöntem tamamen bağlı örgü yöntemi (Mesh) denilir.

Bu yöntemleri karşılaştırdığımızda birinci yöntemde çok az baş düğüm kullanılmaktadır ama bilgi bir yönde birikiyor ve bu nedenle ağın hızı düşmektedir. İkinci metot ise yüksek hız var ama gereksiz direk iletişim ve baş düğüm sayısı fazla olduğu için ağın maliyeti artmaktadır. Bu iki yöntemin avantajları ve dezavantajlarından çıkarım yapılarak yeni bir yöntemle baş düğümlerin sayısını belirleyerek direk iletişimin azalmasını ve bilgilerin bir yönde birikmemesini sağlanmaktadır.

2.1.1. Önerilen Ağ Yapısı

Örneğin bir alana 8 tane algılayıcı düğüm dağıtılmıştır ve 2 tane baş düğüm seçilmiştir. Baş düğümlerin etki alanları kısıtlı olduğu için baş düğümler sadece kendi etki alanlarında olan düğümlere ve yakın olan baş düğüme hizmet vere bilmektedir. Şekil 31'de yeni yöntemin ağ yapısı gösterilmektedir.



Şekil 31. Yeni yöntem ağ yapısı

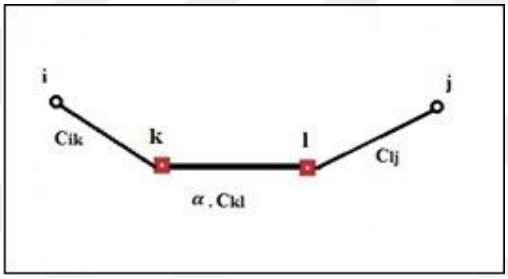
Şekil 31'de **k** baş düğümü A alanında olan düğümlere ve **l** baş düğümü B alanındaki düğümlere hizmet vermektedir. Bu yöntemde i düğümü j düğümüne, **k** ve **l** baş düğümleri aracılığıyla bilgi göndermektedir.

Bu yöntemde düğümler çok hızlı şekilde birbiriyle iletişim kurabilimekteler. Çünkü baş düğümler sadece kendi etki alanlarında olan düğümlere ve nihayet komşu baş düğümüne hizmet vermektedir. Ayrıca baş düğümlerin arasında olan bağlantı çok yoğun hacimde bilgi taşımaktadır. Bu bağlantı yüksek hızlı haberleşme kanalı olabilir ve böylece bilgi taşması hızlandırılmaktadır. Aynı zamanda bilgi taşıma maliyeti düşmektedir.

En uygun dizilimi elde etmek için Yıldız ve Mesh ağırlarını aralarında olan dizilimi bulunmaktadır. Bu işlemi bir ağda tanımlanabilir baş düğüm oluşturma ve bilgi transfer maliyeti bu konuya dâhil ederek toplam maliyeti düşürülmekte. Böylece en uygun dizilim elde edilmktedir.

2.1.1.1. Bilgi Transfer Maliyeti Bir Ağda Tanımlaması

Varsayalım bir alanda 2 tane düğüm ve 2 tane baş düğüm vardır, i düğümünden j düğümüne bilgi transfer maliyeti Şekil 32'de olduğu gibi gösterilmektedir.



Şekil 32. Bilgi transfer maliyeti

Bu şekilde c_{ik} i düğümü ve k baş düğümün arasında, c_{lj} 'de l baş düğümün ve j düğümünün arasında olan bilgi transfer maliyetini göstermektedir. K ve I arasındaki maliyet yoğun olduğu için indirim katsayısına (*disconunt factor*) çarpılmaktadır. Böylece her birim bilgi i düğümünden j düğümüne gönderdiğinde denklem 16'deki maliyeti oluşturmaktadır.

$$c_{ij}^{kl} = c_{ik} + \alpha c_{kl} + c_{lj} \tag{16}$$

$$0 \leq \alpha \leq 1$$

Bu aşamada bulduğumuz bilgi transfer maliyeti optimize edeceğiz, ama bu işlemi yapmadan önce hangi düğüm baş düğüm olmasını ve düğümler hangi baş düğümden servis alacağı belirlenmektedir.

2.1.1.2. Baş Düğüm Seçilmesi

Bu senaryoda varsayımlar ve bilinmeyenlerin tanımlanması gerekmektedir;

1. Farz edelim belirli bir alanda belirli sayıda düğümler vardır. Bu düğümların her bir çiftinin arasında belirli bir yoğunlukta iletişim var. örneğin her ay ortalama 1000 data paketi düğümler arasında taşınmakta ve taşınan verilerin istatistik ortalaması mevcuttur.
2. Kaç algılayıcı düğüm baş düğüme dönüştürmesi belirlenmekte, Örneğin: 3 düğüm baş düğüm olarak ele alındı. Burada maksimum 3 baş düğüm olmaktadır ama bazı istisnalarda bir ağda sadece bir baş düğüm yeterli olabilmektedir. Örneğin bilgisayar yıldız topolojisi ağında olmak üzere bir sunucu kullanımı hız ve masraf bakımından iyi sonuçlar vermektedir. Bu yöntemde yani tek bir baş düğümün seçilmesinde önemli problemler vardır. Örneğin düğüm sayısının fazla olduğu durumda bu yöntemin performansı düşmektedir.

Varsayalım ki bir ağda önceden baş düğümler belli olmuştur. Bu durumda farklı özellikte problemlerin tanıtılması gerekmektedir. Aksi takdirde baş düğümlerin önceden belirlenmediği durumda bütün düğümler baş düğüm olmaktadır. Bu ağ (*fully conected*) meş ağına dönüşmektedir.

3. Bu aşamada hangi düğümün hangi baş düğümden servis aldığının belirlenmesi gerekmektedir. Dolayısıyla burada bir kaç karar değişkeninin tanımlanması gerekmektedir.

x_{ii} = 1 → i baş düğümdür

x_{ii} = 0 → i baş düğüm değildir

x_{ij} = 1 → j, i'den servis alırsa

x_{ij} = 0 → j, i'den servis almazsa

i bir baş düğüm ve j bir normal düğümlerdir. Bu karar değişkenler (*Decision Variable*) x matrisinde bu şekilde dizilmektedir.

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ x_{21} & x_{22\ldots\ldots} & \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

X matrisi bir kare matrisidir ve bu matriste köşegen, baş düğümleri temsil etmektedir. Yani baş düğümler köşegendedir ve onun dışında olan x'ler, hangi normal düğümün hangi baş düğümden servis aldığını göstermektedir. Bu çalışmadaki matris ile değişken yapıyı düzenli matris yapısına dönüştürmektedir.

$$X = \begin{bmatrix} x_{11} & x_{12} \cdots & x_{1n} \\ x_{21} & x_{22\ldots\ldots} & \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

Eğer x_{11} = 1 ve x_{12} = 1 olursa → x_{12} , x_{11} 'den servis alıyor demektir.

Bu tanımlamalara baktığımızda düğümler sadece baş düğümlerden servis almakta yani hiç bir düğüm baş düğüm olmayan düğümlardan servis alamaz.

$x_{ij} \leq x_{ii}$ ∀ i , j → x_{ii} = 0 olursa x_{ij} 'e servis veremez. (17)

(x_{ii} baş düğümler ve x_{ij} düğümlardır.)

2.1.1.3. Baş Düğüm Sayısının Belirlenmesi

Baş düğüm sayısının belirlenmesi denklem 18'ile hesaplanmaktadır.

$$\sum_i x_{ii} \leq P \quad (18)$$

$$\sum_i x_{ii} \geq 1 \quad (19)$$

Denklem 19'da düğümler arasındaki bağlantı sadece baş düğümleri ile sağlanmaktadır ve en az bir tane baş düğümün olması göstermektedir.

Baş düğüm sayısı x matrisinin köşegeninde belirlenmektedir.

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ x_{21} & x_{22\ldots\ldots} & \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

x_{11} , x_{22} ,...., x_{ii} , x_{nn} $x_{ii} \in \{0 , 1\}$

Bu aşamada n tane değişkenimiz var (x_{ii}) ve bunların tamamı 0 veya 1'lerden oluşur (tam sayı). Kullanacağımız optimizasyon algoritması bir sürekli algoritmadır. Sürekli bir algoritma tam sayılar ile çalışmamaktadır ve problem oluşmaktadır. Bu problemi çözmek için çalışmada x_{ii} gerçek sayı olarak kodlanmaktadır. Sonra bunların üzerinden p tane x_{ii} (baş düğüm) seçilmektedir. Bu işlemi yapmak için ilk adımda 0 ve 1 arasında $\hat{x}_{ii}$'lar üretilmiştir daha sonra bunların içinden 0,5'ten küçük olanları 0 ve diğerleri 1 kabul edilmektedir. Örneğin öğrencilerin sınavdan geçme ve kalma notları gibi 50 geçiş notu ise bu notu geçenlere 1 ve geçmeyenler 0, yani sadece öğrencilerin geçme puan almaları önemli ve notun kaç olduğunun bir önemi yoktur. Bu değerlendirme denklem 20'de gösterilmiştir.

$$\widehat{x_{11}}\ ,\ \widehat{x_{22}}\ ,\ \ldots,\ \widehat{x_{ii}}\ ,\ \widehat{x_{nn}} \leftarrow x_{11}\ ,\ x_{22}\ ,\ldots.,\ x_{ii}\ ,\ x_{nn}$$
$$\hat{x}_{ii} \in [0\ ,\ 1] \leftarrow x_{ii} \in \{0\ ,\ 1\}$$
$$x_{ii} = \begin{cases} 1 & \leftarrow \ \widehat{x_{ii}} \geq 0.5 \\ 0 & \leftarrow \text{aksi halde} \end{cases} \qquad (20)$$

Yukarıdaki kodlamada sürekli sayılar ikili sayılara dönüştürülmüştür. Bu senaryoda herhangi bir düğümun değeri $\widehat{x_{ii}} \geq 0{,}5$ olduğu zaman karşısındaki x_{ii} baş düğüm olabilmektedir. Baş düğümün sayısında hiç bir sınırlama yoktur. İkinci senaryoda p tane baş düğümün seçmesi istenilmektedir:

$$\sum_i x_{ii} = \mathrm{P}$$

Ürettiğimiz $\widehat{x_{ii}}$ sayılarını değerlerine göre sıralanmaktadır. Sonra ilk p tane büyük $\widehat{x_{ii}}$ karşısındaki x_{ii} baş düğüm seçiyoruz yani p tane kesin baş düğüm seçilmektedir.

Üçüncü senaryoda baş düğüm sayısının sınırlı olmasını istenmektedir yani x_{ii}'lerin baş düğüm olması için 2 koşulun uygulanması gerekmektedir:

1) $x_{ii} = 1 \leftarrow \ \widehat{x_{ii}} \geq 0{,}5$
2) $\sum_i x_{ii} \leq \mathrm{P}$

Bu senaryoda da önceki senaryoda olduğu gibi ilk x_{ii}'lar her zaman seçilmiyor yani ilk x_{ii}'ların en fazla p tanesi seçilir ve bu ilk p tanenin $\widehat{x_{ii}} \geq 0{,}5$ olması gerekmektedir.

Baş düğüm sayısının seçilmesindeki 3 senaryoyu bir örnekte açıklanmaktadır; varsayalım 8 tane 0 ve 1 arasında sayımız var:

- $\widehat{x_{11}}$= 0.2769, $\widehat{x_{22}}$= 0,0462, $\widehat{x_{33}}$= 0,0971, $\widehat{x_{44}}$= 0,8337, $\widehat{x_{55}}$= 0,6947, $\widehat{x_{66}}$= 0,3171, $\widehat{x_{77}}$= 0,9502, $\widehat{x_{88}}$= 0,0344
- Varsayalım p= 4

Tablo 1. x_{ii}= 1 ← $\widehat{x_{ii}} \geq 0{,}5$ ve baş düğüm sayısı 4

$\widehat{x_{ii}}$		0,2769	0,0462	0,0971	0,8337	0,6947	0,3171	0,9502	0,0344
x_{ii}	$\widehat{x_{ii}} \geq 0.5$ İlk senaryo	0	0	0	1	1	0	1	0
	$\sum_i x_{ii} = 4$ İkinci senaryo	0	0	0	1	1	1	1	0
	$\sum_i x_{ii} \leq 4$ Üçüncü senaryo	0	0	0	1	1	0	1	0

Tablo 1'de İlk senaryoda 3 baş düğüm seçilme potansiyeli var yani 3 tane

$\widehat{x_{ii}} \geq 0{,}5$ bu nedenle x_{44}, x_{55}, x_{77} =1

İkinci senaryoda 4 baş düğüm seçilme potansiyeli var ve 4 en büyük sayıyı sırasıyla baş düğüm olarak seçebilmektedir.

Üçüncü senaryoda x_{ii}'lar içinde en fazla 4 tane baş düğüm seçilmektedir burada x_{44}, x_{55}, x_{77} =1 ve baş düğümlerdir

Bu kodlamada eğer tüm $\widehat{x_{ii}}$'ların değerleri $0{,}5'den$ az olursa bir problem oluşmaktadır. Örneğin:

- $\widehat{x_{11}}$= 0.1385, $\widehat{x_{22}}$= 0,0231, $\widehat{x_{33}}$= 0,0486, $\widehat{x_{44}}$= 0,0411, $\widehat{x_{55}}$= 0,3474, $\widehat{x_{66}}$= 0,1585, $\widehat{x_{77}}$= 0,4751, $\widehat{x_{88}}$= 0,0178
- P = 4

Tablo 2. Tüm düğümlerin değerleri $\widehat{x_{ii}} \leq 0{,}5$ olduğu zaman ve P = 4

$\widehat{x_{ii}}$		0.1385	0.0231	0.0486	0.0411	0.3474	0.1585	0.4751	0.0178
x_{ii}	$\widehat{x_{ii}} \geq 0.5$ İlk senaryo	0	0	0	0	0	0	0	0
	$\sum_i x_{ii} = 4$ İkinci senaryo	0	0	0	1	1	1	1	0
	$\sum_i x_{ii} \leq 4$ Üçüncü senaryo	0	0	0	0	0	0	0	0

Bu durumda hiçbir baş düğüm seçilmemektedir ama denklem 19'a göre ağdaki algılayıcı düğümler arasında bağlantı kurmak için en az bir baş düğüm seçmemiz gerekmektedir. Bu problemi çözebilmek için düğümlerin arasında en fazla değeri olan düğüm seçilmemelidir, Tablo 3'de veriler ile bu durumu açıklanmaktadır.

Tablo 3. Düğümlerin değerleri $\widehat{x_{ii}} \leq 0{,}5$ olduğu zaman çözümünün açıklanması

$\widehat{x_{ii}}$		0.1385	0.0231	0.0486	0.0411	0.3474	0.1585	0.4751	0.0178
x_{ii}	$\widehat{x_{ii}} \geq 0.5$ İlk senaryo	0	0	0	0	0	0	0 1	0
	$\sum_i x_{ii} = 4$ İkinci senariyo	0	0	0	1	1	1	1	0
	$\sum_i x_{ii} \leq 4$ Üçüncü senariyo	0	0	0	0	0	0	0 1	0

Tablo 3'e bakıldığında $\widehat{x_{77}}$ en değerli düğüm olmaktadır. Bu düğümun x_{77} =1 kodlayarak baş düğüm olarak seçilmektedir.

Aslında yapılan işlemde x_{ii}'larımızın 0 veya 1 olmasını istenmektedir ve 1 olan x_{ii}'lar arasında en fazla p tane baş düğümün olması istenmektedir yani uygulamada üçüncü senaryo ele alınacaktır ve bu işlem aşağıdaki gibi kodlanmaktadır:

i. 0 ve 1 arasında $\widehat{x_{ii}}$ 'lar üretilmektedir.
ii. $\widehat{x_{ii}}$ 'ları değerlerine göre sıralanmaktadır.
iii. 0.5'den küçük olan $\widehat{x_{ii}}$ 'ları 0'a eşit tutulmaktadır.

iv. En fazla P tane 0,5'den büyük olan $\widehat{x_{ii}}$ 'ların içinden en büyüklerini 1'e eşit tutulmaktadır.

($\widehat{x_{ii}}$ 'ların hiçbiri $\geq 0{,}5$ olmazsa, en büyük değeri olan $\widehat{x_{ii}}$ 'ı 1'e eşit tutulur.)

Yaptığımız işlemi bir örnekte açıklanmaktadır; varsayalım 9 tane 0 ve 1 arasında sayımız var:

- $\widehat{x_{11}}$= 0,4387, $\widehat{x_{22}}$= 0,3816, $\widehat{x_{33}}$= 0,7655, $\widehat{x_{44}}$= 0,7952, $\widehat{x_{55}}$= 0,1869, $\widehat{x_{66}}$= 0,4898, $\widehat{x_{77}}$= 0,4456, $\widehat{x_{88}}$= 0,6463, $\widehat{x_{99}}$= 0,7094
- Ve en fazla P = 3 seçilmesi istenilmektedir.

Tablo 4. Yeni bir örnek için baş düğüm sayısı P≤ 3

$\widehat{x_{ii}}$		0.4387	0.3816	0.7655	0.7952	0.1869	0.4898	0.4456	0.6463	0.7094
x_{ii}	$\sum_i x_{ii} \leq 3$	0	0	1	1	0	0	0	1 0	1

Yapılan çalışmada basit bir algoritmayla x matrisin köşegeninden baş düğümler seçilmektedir. Sonra x matrisin diğer değişkenlerini tanımlanmaktadır.

$$X = \begin{bmatrix} x_{11} & x_{12} \cdots & x_{1n} \\ x_{21} & x_{22\ldots\ldots} & \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

Yani hangi düğüm hangi baş düğümden servis almasını belirlenmektedir bunun için iki tane koşul tanımlanmatadır. Bu koşullar denklem 21ve 22'de gösterilmektedir.

$$1)\ x_{ij} \leq x_{ii} \rightarrow \text{eğer}\ \ x_{ii} = 0 \Longrightarrow \forall \text{j} :\ x_{ij} = 0 \qquad (21)$$

Birinci koşulda x_{ii} sıfır olduğu zaman baş düğüm olmamaktadır ve bu nedenle hiçbir düğüm o x_{ii}'den servis alamamaktadır. Yansıra x_{ij} lerin yerine sıfır ve bire eşit ve aralarında gerçek sayı kullanımı istenmektedir çünkü problemin sürekli algoritmalarla çözülmesi istenmektedir.

$$2)\ \sum_j x_{ij} = 1 \quad \forall \text{j} \qquad (22)$$

İkinci koşulda j düğümu en az bir tane baş düğümünden servis almasını göstermektedir. Ama hangi baş düğümden servis alacak bu konuda aşağıdaki açıklamayla anlatılacaktır. Açıklama: burada birkaç tane sayımız ($\widehat{xij}$) var ve bu sayılar $0 \leq \widehat{xij} \leq 1$ aralığındadır. Bizim bakışımızdan bu sayıların her hangisi i'ın j'ye sahip olma puanıdır. Aynı bir açık artırma gibi herhangi baş düğüm en yüksek i' ve dolaysıyla $(\widehat{xij})$ önerirse o i'ıncı baş düğüm kazanır yani j'ye sahip olur. Bu işlem denklem 23'te gösterilmektedir.

$0 \leq \widehat{xij} \leq 1$

$$x_{ij} = \begin{cases} 1 \quad \Rightarrow \quad i = argmax_{i'}\ \widehat{xi'j} \\ 0 \qquad \Rightarrow \text{ aksi halde} \end{cases} \tag{23}$$

i': en fazle değeri olan i

Ama yine de baş düğüm olarak seçilen bir düğümun kesinlikle düğümlere servis vereceği garanti edilmemektedir. $(x_{ij} \leq x_{ii})$

Bu sorunu denklem 24 ile çözülmektedir.

$$x_{ij} = \begin{cases} 1 \quad \Rightarrow \quad i = argmax_{i'}\ \widehat{xi'j} \,.\, x\ i'i' \\ 0 \qquad \Rightarrow \text{ aksi halde} \end{cases} \tag{24}$$

Denklem 24 ile köşegen üzerinde olan elemanlar o satırdaki bütün elemanlarla çarpılır.

Böylece en yüksek değeri olan $i'\ x_{ij}$ bir olur aksi halde sıfır olmaktadır.

Bir örnek: varsayalım bir 6*6 matrisimiz var olmaktadır ve 3 tane baş düğüm seçilecektir. Burada nasıl baş düğüm seçilmesi ve hangi düğüm hangi baş düğümden servis alacağı gösterilmektedir.

0.5853	0.8909	0.8047	0.1966	0.5853	0.3804
0.2238	0.9593	0.2543	0.2511	0.5497	0.5674
0.7513	0.5472	0.8143	0.6760	0.9172	0.0759
0.2551	0.1386	0.2436	0.4733	0.2858	0.5401
0.5060	0.1493	0.9293	0.3517	0.7572	0.5308
0.6991	0.2575	0.3500	0.8308	0.7537	0.7792

İlk olarak baş düğümleri matrisin köşegenden seçilmektedir.

0,5853, 0,9593, 0,8143, 0,4733 0,7572, 0,7792

Tablo 5. Üstteki 6*6 boyutlu matrisin baş düğümlerin seçilmesi

$\widehat{x_{ii}}$		0.5853	0.9593	0.8143	0.4733	0.7572	0.7792
x_{ii}	$\sum_i x_{ii} \leq 3$	0	1	1	0	0	1

Tablo 6'da anlaşılıyor ki x_{22} ve x_{33} ve x_{66} baş düğüm olarak seçilmektedir.

$x_{22} = x_{33} = x_{66} = 1$

0	0.8909	0.8047	0.1966	0.5853	0.3804
0.2238	1	0.2543	0.2511	0.5497	0.5674
0.7513	0.5472	1	0.6760	0.9172	0.0759
0.2551	0.1386	0.2436	0	0.2858	0.5401
0.5060	0.1493	0.9293	0.3517	0	0.5308
0.6991	0.2575	0.3500	0.8308	0.7537	1

Köşegende baş düğüm seçilmeyen elemanlar sıfır olur ve hiç bir düğüme servis veremez ve bu nedenle ifade 12'ye bakarak o satırda olan bütün $\widehat{xij}$ sıfır olur yani bizim için önemsiz olurlar.

0	0	0	0	0	0
0.2238	1	0.2543	0.2511	0.5497	0.5674
0.7513	0.5472	1	0.6760	0.9172	0.0759
0	0	0	0	0	0
0	0	0	0	0	0
0.6991	0.2575	0.3500	0.8308	0.7537	1

Daha sonra hangi düğüm hangi baş düğümden servis alacağını veya hangi baş düğüm hangi düğümlara servis vereceği denklem 23'e göre belirlenmektedir ayrıca her baş düğüm kendisinden servis alır yada verir.

Bu matriste ilk sütun birinci düğüme aittir ve birinci düğüm x_{22} ve x_{33} ve x_{66} aralarında en fazla $\widehat{x_{i'j}}$ öneren baş düğümden servis almaktadır. Böylece birinci sütunda x_{31} en yüksek değere sahip olduğu için x_{33} baş düğümünden servis almakta ve bu sütunda diğer elemanlar sıfır olmaktadır.

İkinci düğüm için ikinci sütuna bakılmalı ve burada x_{22} baş düğüm olduğu için kendisinde servis almakta ve bu sütunda diğer elemanların değeri sıfır olacak çünkü zaten x_{22} değeri birdir ve kendisi baş düğüm olmaktadır.

Üçüncü düğüm için aynı ikinci düğüm 'un senaryosu gibi x_{33} kendisi başdüğüm olduğu için diğer elemanlar önemsiz olmaktadır.

Dördüncü düğüm için dördüncü sütuna baktığımızda $\widehat{x_{i'_J}}$'in en büyük miktarı x_{64} görülmektedir ve böylece x_{64} bir olmakta ve x_{66} baş düğümünden servis almakta ve diğer elemanlar sıfır olmaktadırlar.

Beşinci sütunda en fazla olan $\widehat{x_{i'_J}}$' bu sütunda x_{35}' de olmaktadır ve beşinci düğüm x_{35} üçüncü baş düğümden servis almaktadır ve böylece x_{35} bir olur ve değer elemanlar sıfır olmalıdır ve altıncı düğüm kendisi baş düğüm olduğu için kendisinden servis almakta ve değer elemanlar bu sütunda sıfır olmaktadır.

0	0	0	0	0	0
0	1	0	0	0	0
1	0	1	0	1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	1	0	1

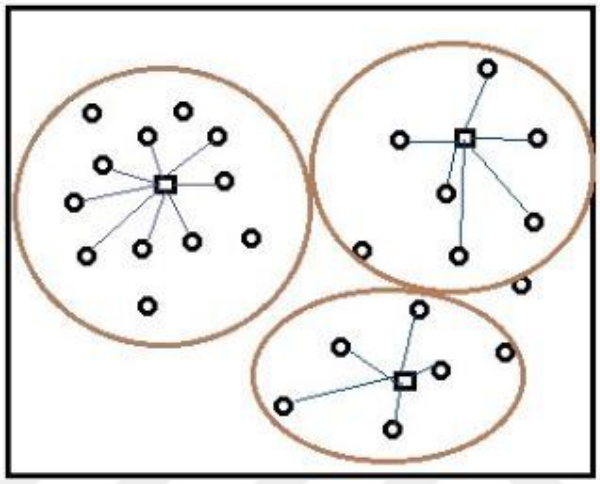
Sonuç olarak bir rastgele matrisle belli (visible) bir yaklaşıma varılmaktadır ve matrisimizin son hali üsteki matriste gösterilmektedir. Bu matriste birinci ve beşinci düğüm x_{33} baş düğümünden servis alır ve dördüncü düğüm ise x_{66} baş düğümünden servis almaktadır ama x_{22} başdüğümü hiç bir düğüma servis verememektedir ve x_{22} baş düğümüne mağlup (loser) baş düğüm denilmektedir.

2.2. Baş Düğüm Sayısının Belirlenmesindeki Hacim Kapasite Problemi

İnternet ağında olduğu gibi algılayıcı ağlarda da baş düğümler düğümlerden aldıkları veri hacmine karşı sınırlı olmakta ve bu nedenle verecekleri hizmetlerde sınırlı olmaktadır. Gönderilen veri baş düğüm kapasitesinden fazla olduğu zaman bazı düğümler baş düğümlerden hizmet alamamaktadır. Bu problemi aşmak için baş düğümlere hacim kapasitesi tanımlanmaktadır. Şekil 33'de bir ağda kapasite problemi göstermektedir. Bu Denklemde baş düğümlerin kapasitesini göstermektedir.

$\sum_j x_{ij}\ dj \leq \Gamma_i\ \forall i$ (d_j : j'in talebi ve Γ_i : hacim kapasitesidir)

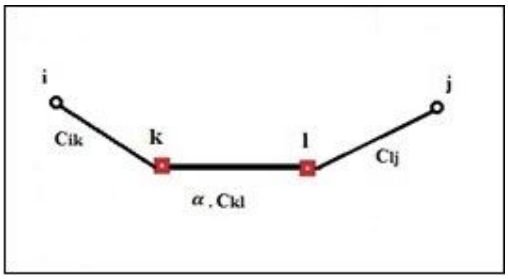
Yani x_{ij}'in toplam hacim talebi Γ_i 'dan (hacim kapasitesinden) az olmalıdır.



Şekil 33. Algılayıcı ağlarda kapasite problemi

2.3. Kablosuz Algılayıcı Ağın Maliyeti

Belli (visible) bir yaklaşımı elde ettikten sonra ağın maliyetini hesaplanmaktadır. Sonra maliyet fonksiyonu minimize edilecektir. Şekil 34 bizim ağımızı sergilemektedir.



Şekil 34. Önerilen ağ yapısı

Şekil 32'de c_{ik} i düğümu ve k baş düğümün arasında, c_{lj} 'de l baş düğümün ve j düğümun arsında olan bilgi transfer maliyetini göstermektedir. k ve l arasındaki veri transferi yoğun olduğu için indirim katsayısına (*disconunt factor*) çarpılmaktadır. Böylece

her birim bilgi i düğümundan j düğümuna gönderdiğinde denklem 16'da maliyeti göstrilmiştir.

$$c_{ij}^{kl} = c_{ik} + \alpha c_{kl} + c_{lj} \quad (16)$$

$0 \leq \alpha \leq 1$ α = İndirim katsayısıdır

Bu aşamada bulduğumuz bilgi transfer maliyetini bütün i ve j'ler için hesaplanılacaktır.

Transfer maliyetinin toplamı alttaki denklem 25'te gösterilmiştir.

$$\sum_i \sum_j \sum_k \sum_l x_{ki} x_{lj} c_{ij}^{kl} r_{ij} \quad (25)$$

Bu formülde x_{ki} i düğümün k baş düğümünden hizmet almasını temsil etmektedir. x_{lj} ise j düğümün l baş düğümünden hizmet almasını temsil etmektedir. i düğümden j düğümüne k ve l baş düğümleri vasıtasıyla gönderilen veri maliyeti c_{ij}^{kl}'le gösterilmektedir. r_{ij} ise i'ler ve j 'ler arasında verinin gidiş dönüş yolu sayısını göstermektedir.

Bir düğümü baş düğüm yapmak için maliyet harcanmaktadır ve bu maliyet transfer maliyetine ekleyerek toplam maliyet elde edilmektedir. Toplam masraf yada maliyet formül 25 gösterilmektedir ve bu maliyet alttaki koşulları sağlayarak minimize edilmektedir. Minimize edilmiş maliyet denklem 26'da gösterilmektedir.

$$\text{Min } \sum_i \sum_j \sum_k \sum_l x_{ki} x_{lj} c_{ij}^{kl} r_{ij} + \sum_i x_{ii} fi \quad (26)$$

Koşullar ise

- $xij \leq xii \qquad \forall i, j$
- $\sum_i x_{ij} = 1$
- $\sum_i x_{ii} \geq 1$
- $\sum x_{ii} \leq p$
- $x_{ij} \epsilon \{0,1\}$

Bu zor problem ve koşulların yerine basit bir matrisle bu problemin amaç fonksiyonunu ve koşullarını anlayan mekanizma yardımıyla bu problem minimize edilecektir.

2.4. Problemin Çözümü ve Matlab Programında Kodlaması

Genelde bütün problemlerde yapılan işler aynıdır. Problemlerin amaç fonksiyonuna göre çözüm işlemlerinin sırası belirlenmektedir. Problemin çözümünde yapılan işlemler altta sıralanmaktadır.

1. İlk adımda bir yapı tanımlanmakta ve bu yapının içinde modelin (problemimizin) bütün verilerini kapsaması sağlanmaktadır.
2. Bu adımda problemimize ilk cevabı üretmek için bir fonksiyon oluşturulmaktadır.
3. Bu aşamada bir başka fonksiyon oluşturulmaktadır, bu fonksiyon ilk cevapları alıp fiziksel değişkenlere dönüştürecek ve diğer hesaplamaları yapmaktadır yani hangi düğümlerin baş düğüm olup olmayacağını yansıra hangi düğüm hangi baş düğümden servis alacağını belirlemektedir.
4. Bu adımda değerlendirme yapılacaktır.
5. Son aşamada elde edilen bulgular 3 tane farklı ve sürekli optimizasyon algoritmasına bağlanmaktadır. Sürekli algoritma kullanımın nedeni de problemin sürekli kodladığından kaynaklanmaktadır.

Son zamanlarda genellikle sürekli algoritmalar kullanılmaktadır. Yani sürekli algoritmalar fazlasıyla sunuluyor ve ayrık algoritmalar çok nadir kullanılmaktadır. Bu nedenle sürekli kodlama çök önem kazanmaktadır. Kullandığımız algoritmalar, Diferansiyel Gelişim, Parçacık Sürü ve Yayılmacı Yarışmacı algoritmalardır.

2.4.1. Bir Yapının Tanımlanması

İlk adımda rastgele bir model oluşturulmaktadır. Sonra bu modelin farklı boyutlarını oluşturup kayd edecilmektedir bunu için bir fonksiyon oluşturulmaktadır. Sonra modelleri seçimi için bir başka fonksiyon oluşturulmaktadır.

2.4.1.1. Rasgele Bir Model

İlk yapılan şey rastgele bir (CreateRandomModel(N)) isimli model oluşturulmaktadır, en önemli şey bu aşamada baş düğümlerin sayısıdır. En ideal baş düğüm

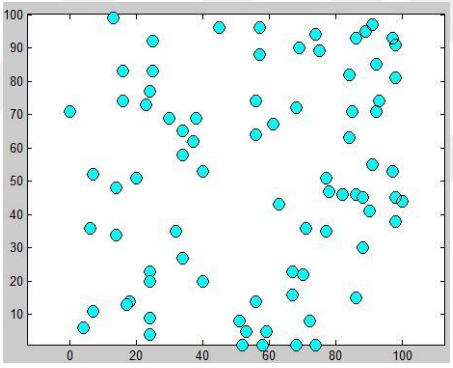
sayısı bu çalışmada deneysel olarak %15 ve %30 arasında bulunmaktadır. Baş düğüm sayısı Matlab üzeninde kodlaması altta gösterilmektedir.

```
Pmin = ceil(0.15 * N);
Pmax = ceil(0.30 * N);
P = randi([Pmin Pmax]);
```

Varsayalım eğer düğüm sayısı 20 olursa baş düğüm sayısı 3, 4, 5 ve 6 sayılarında herhangi biri olabilmektedir.

Rastgele modelde ikinci yapılan şey, düğümlerin konumlarının bulmasıdır. Çünkü düğümlerin konumlarını göz önüne alarak mesafe ve maliyetleri hesaplanabilmektedir.

Bu nedenle bir (0,0) ve (100,100) koordinatlı alanda bütün i ve j düğümleri dağıtılmış farz edilmektedir. Şekil 35'te dağıtılmış düğümler ve alanı gösterilmektedir.



Şekil 35. Dağıtılmış düğümler ve alan

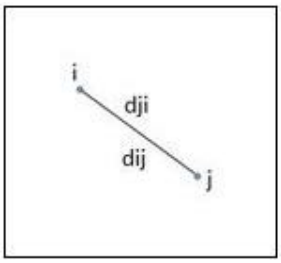
Şekil 35'te düğümlerin dikey ve yatay koordinatları bulunmaktadır. Matlab üzerinde de bütün düğümler için bu koordinatları kodlanacaktır.

```
Xmin = 0;
Xmax = 100;
X = randi([Xmin Xmax],1,N);
Ymin = 0;
Ymax = 100;
Y = randi([Ymin Ymax],1,N);
```

Düğümler aralarındaki mesafenin hesaplanması:

Mesafe hesaplaması i ve j düğümlerinin arasında bir kere i düğümden j düğümüne, bir kerede j düğümünden i düğümüne hesaplanmaktadır. Mesafe, matematikte iki nokta

arasındaki uzunluğu bulma formülünün aynısı kullanılarak hesaplanılmaktadır. Şekil 36'da iki düğüm arasındaki mesafe hesaplanması ve Matlab üzerinde kodlaması gösterilmiştir.



Şekil 36. İki düğüm arasındaki mesafe

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{27}$$

$$d_{ji} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{28}$$

Bu aşamada üsteki formülleri kullanarak her ikili düğümlerin arasındaki mesafe Matlab üzerinde kodlanmaktadır.

$d = zeros(N, N);$

$for\ i = 1{:}N - 1$

$for\ j = i + 1{:}N$

$d(i,j) = sqrt((X(i) - X(j))\text{^}2 + (Y(i) - Y(j))\text{^}2);$

$d(j,i) = d(i,j);.$

Düğümler aralarındaki maliyet hesaplanması:

Bu aşamada her bir mesafe karşılığındaki maliyet bulunmaktadır. Normalde maliyet i ve j aralığında dij'le orantılıdır.

$$cij \propto dij \tag{29}$$

Aslında her bir veri transferi için bir maliyet harcanmaktadır. Bu maliyeti dij çarpılıyor ve nihayet i ve j arasındaki maliyet alttaki formülü oluşturulmaktadır.

Her bir mesafe ünitesini transfer maliyeti bir sabit rakamdır. Bu rakım her ne olabilir bu çalışmada 10 yapılmaktadır.

- Her bir ünite maliyeti = 10
- Her bir ünite transfer maliyeti = Her bir ünite maliyeti × her bir mesafe

Matlab üzerinde kodlaması:

$CostPerDistanceUnit = 10;$

$c = (CostPerDistanceUnit * d);$

İndirim katsayısı:

Baş düğümler arasındaki veri transferi toplu olduğu için bu veri transferlerin maliyetleri bir indirim kat sayısına (Discount Factor) çarpılmaktırlar. İndirim kat sayısı 0 ve 1 aralığındadır. Genelde 0,7 olarak alınmaktadır.

Düğümlerin aralarındaki veri isteği (rij):

Düğümlerin aralarındaki veri isteğin rij'le gösterilmektedir. Burada i ve j aralarındaki veri isteği simetrik olup olmadığı önemli değildir, önemli olan bir N×N boyutlu matris oluşturulması ve bu matrisin köşegeninin sıfır olmasıdır.

1. $rmin = 10;$
2. $rmax = 50;$
3. $r = randi([rmin\ rmax], N, N);$
4. $r = r - diag(diag(r));$

Diag fonksiyonun Matlab programında bir matrisi köşegen ve ters durumda bir köşegenden matris yapmaktadır.

Burada ilk başta veri isteğinin alt ve üst sınırlarını belirlenmektedir. Sonra bir rastgele N×N boyutlu r matrisi oluşturulmaktadır. Rastgele r matrisin köşegenini sıfır yapmaktayız. Bu nedenle dördüncü formülü kullanılmaktadır. Bu formülde parantez içindeki diag(r)'le r matrisin köşegenin bir sütunda ekstraksiyon yapılmaktadır. Sonra parantez dışındaki diag r matrisinin boyutunda bir matris oluşturulmaktadır. Bu matrisin köşegenleri aynı r matrisidir ve diğer elemanları sıfır olmaktadır. Son aşamada diag(diag(r)) matrisi r matrisine dönüştürülüyor ve böylece yeni r matrisinin köşegenleri sıfır ve diğer elemanlar aynı ilk oluşturulmuş r matrisindeki gibi olmaktadır.

Baş düğüm yapmak için harcanan maliyetin hesaplanması:

Baş düğüm yapılan düğümlerin maliyeti $c_{ij}^{kl} \times r_{ij}$ orantılı olmaktadır. Bu nedenle r_{ij} ve c_{ij}^{kl} tüm düğümler için hesaplandıktan sonra birbirine çarpılmaktadır. Sonra bu maliyet baş düğümlerin sayısına bölünmektedir.

$rc = r.* c;$

$SumRC = sum(rc(:));$

$fmean = SumRC / P;$

$fmin = round(0.8 * fmean);$

$fmax = round(1.2 * fmean);$

$f = randi([fmin\ fmax],1,N);$

Bu aşamada gereken her şey bu problemimiz için yapılmış ve Matlab üzerinde kodlanmaktadır. Ama bu yapının sonuçlarını görmek için alttaki kodları Matlab üzerinde yazıp program Matlab üzerinde çalıştırılmaktadır.

$model.N = N;$

$model.P = P;$

$model.X = X;$

$model.Y = Y;$

$model.d = d;$

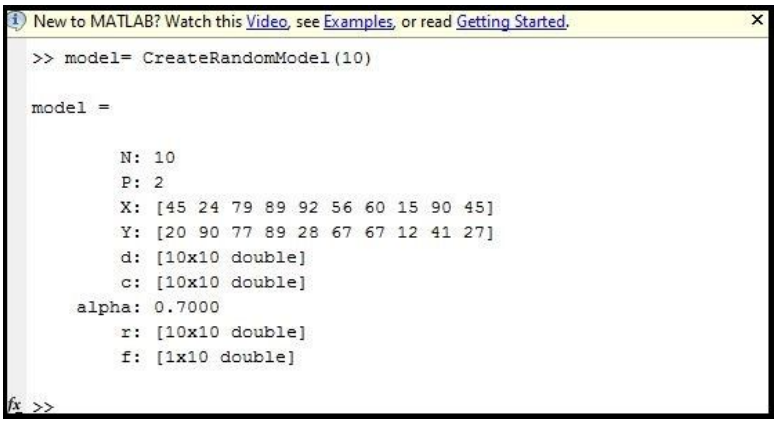
$model.c = c;$

$model.alpha = alpha;$

$model.r = r;$

$model.f = f;$

Eğer düğüm sayısı 10 tane (N=10) olursa programı çalıştırdığımız zaman üretilen model Şekil 37'deki sonuçlar gösterilmektedir.



Şekil 37. Üretilen model

2.4.1.2. Tanımlanan Yapıya Farklı Boyutlarda Model Oluşturulması

Rastgele ürettiğimiz model her çalışmasında farklı sonuçlar vermektedir. Bu sonuçları sabit yapmak için ve farklı boyutlar üretmek için bir başka fonksiyon oluşturulmaktadır. Bu fonksiyon istediğimiz model boyutlarını üretmektedir ve bu farklı boyutlu modelleri bir yerde kaydetmektedir. Bu durumda herhangi çağırdığımız model rastgele halinden bir sabit hale dönüşmektedir.

Şu aşamada rastgele ürettilen modelin farklı boyutları üretilmektedir. Bu nedenle CreateAndSaveModele isminde bir fonksiyon oluşturulmaktadır.

$function\ CreateAndSaveModels()$

$N = [10\ 20\ 30\ 40\ 50\ 60\ 70\ 80\ 90\ 100];$

$nProblem = numel(N);$

$for\ k = 1{:}\ nProblem$

$model = CreateRandomModel(N(k));$

$ModelName = ['chlap\_'\ num2str(model.N)];$

$save(ModelName, 'model');$

end

end

2.4.1.3. Modellerin Seçimi

Oluşturduğumuz modelleri çağırmak ve seçimini yapmak için bir fonksiyon üretilmektedir. Bu fonksiyon Matlab üzerinde grafik arayüz komutlarını kullanarak tüm modelleri bir pencerede göstermektedir. Bu oluşturduğumuz pencerede tüm modelleri çağırılıp seçebilmektedir.

Bu fonksiyon Matlab üzerinde kodlanması ve oluşturulmuş model seçim penceresinde gösterilmektedir. Şekil 37'de model seçimi için oluşan pencere gösterilmektedir.

$function\ model = SelectModel()$

$Filter = \{'*.mat', 'MAT\ Files\ (*.mat)''*.*', 'All\ Files\ (*.*)'\};$ %% pencerenin altında filtreleri göstermek için

$[FileName\ FilePath] = uigetfile(Filter, 'Select\ Model\ \ldots');$

$if\ FileName == 0$

$model = [];$

$return;$

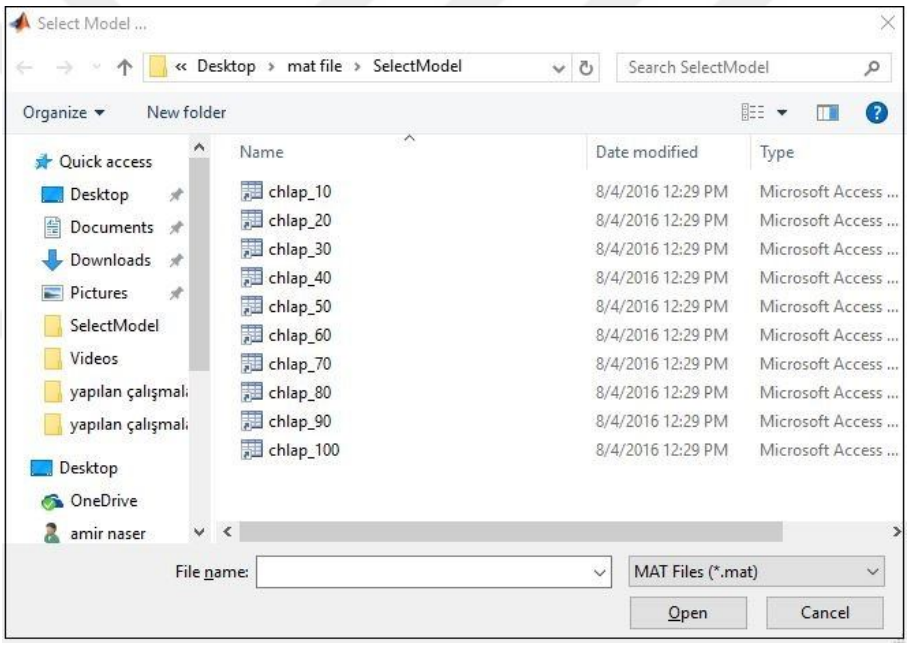
end

$FullFileName = [FilePath\ FileName];$

$data = load(FullFileName);$

$model = data.model;$

end



Şekil 38. Model seçimi için oluşan pencere

Bu aşamada problem için 3 tane mekanizma yapılmaktadır. Bu modelin farklı boyutlarının seçimi için rastgele bir model üretilmektedir. Böylece problemin çözümünün birinci adımı yapılmaktadır.

2.4.2 İkinci Adım Problemin İlk Rastgele Çözümünün Üretilmesi

Bu adımda problemimizin üretilmiş modelleri için rastgele ilk yaklaşımı bulunmaktadır. Bu nedenle önce düğümlerin sayısı belirlenmetedir. Sonra bu modellerin boyutuna göre bir kare matris (N×N) oluşturulacaktır.

$function \;\; xhat = CreateRandomSolution(model)$

$N = model.N;$

$xhat = rand(N, N);$

end

Bu fonksiyon sonucu bir N×N matristir. İçindeki rakamlarının rastgele hepsi 0 ve 1 arasında gerçek sayılardır. Üretilen $xhat$ matrisin içindeki rakamlar bizim ham cevaplarımızdır. Bir sonraki adımda bu matrisin köşegenini ayırarak değerlendireceğiz ve sonra diğer işlemleri yapılmaktadır. Şekil 39'da oluşturulmuş $xhat$ matrisi göstermektedir.

```
>> Xhat=CreateRandomSolution(model)

Xhat =

    0.8147    0.1576    0.6557    0.7060    0.4387    0.2760    0.7513    0.8407    0.3517    0.0759
    0.9058    0.9706    0.0357    0.0318    0.3816    0.6797    0.2551    0.2543    0.8308    0.0540
    0.1270    0.9572    0.8491    0.2769    0.7655    0.6551    0.5060    0.8143    0.5853    0.5308
    0.9134    0.4854    0.9340    0.0462    0.7952    0.1626    0.6991    0.2435    0.5497    0.7792
    0.6324    0.8003    0.6787    0.0971    0.1869    0.1190    0.8909    0.9293    0.9172    0.9340
    0.0975    0.1419    0.7577    0.8235    0.4898    0.4984    0.9593    0.3500    0.2858    0.1299
    0.2785    0.4218    0.7431    0.6948    0.4456    0.9597    0.5472    0.1966    0.7572    0.5688
    0.5469    0.9157    0.3922    0.3171    0.6463    0.3404    0.1386    0.2511    0.7537    0.4694
    0.9575    0.7922    0.6555    0.9502    0.7094    0.5853    0.1493    0.6160    0.3804    0.0119
    0.9649    0.9595    0.1712    0.0344    0.7547    0.2238    0.2575    0.4733    0.5678    0.3371

>>
```

Şekil 39. Oluşturulmuş $xhat$ matrisi

2.4.3. Üçüncü Adım Baş Düğüm Belirlenmesi ve Diğer Hesaplamalar

Bu aşamada bir $\boldsymbol{sol = AmirSolution(xhat, model)}$ isminde fonksiyon oluşturulmaktadır. Bu fonksiyon üretilmiş $xhat$ matrisin içindeki ham değişkenleri alıp fiziki değişkenlere dönüştürmektedir. Bu nedenle ilk $xhat$ matrisin köşegeni alınmaktadır ve değerlendirildikten sonra baş düğüm olan değişkenler 1 ve baş düğüm olmayan değişkenler 0 olmaktadır. Sonra hangi düğüm hangi baş düğümden hizmet alacağı belirlenecektir.

Önceki kodlanan fonksiyonlar ve AmirSolutin fonksiyonu ve diğer fonksiyonlar birlikte yeni yapılan yöntemin algoritmasını oluşturmaktadır. Bu algoritmanın bazı kodları va akış diagramı şekil 40' da gösterilmektedir.

Şekil 40. Yeni yöntem algoritmasının akış diagramı

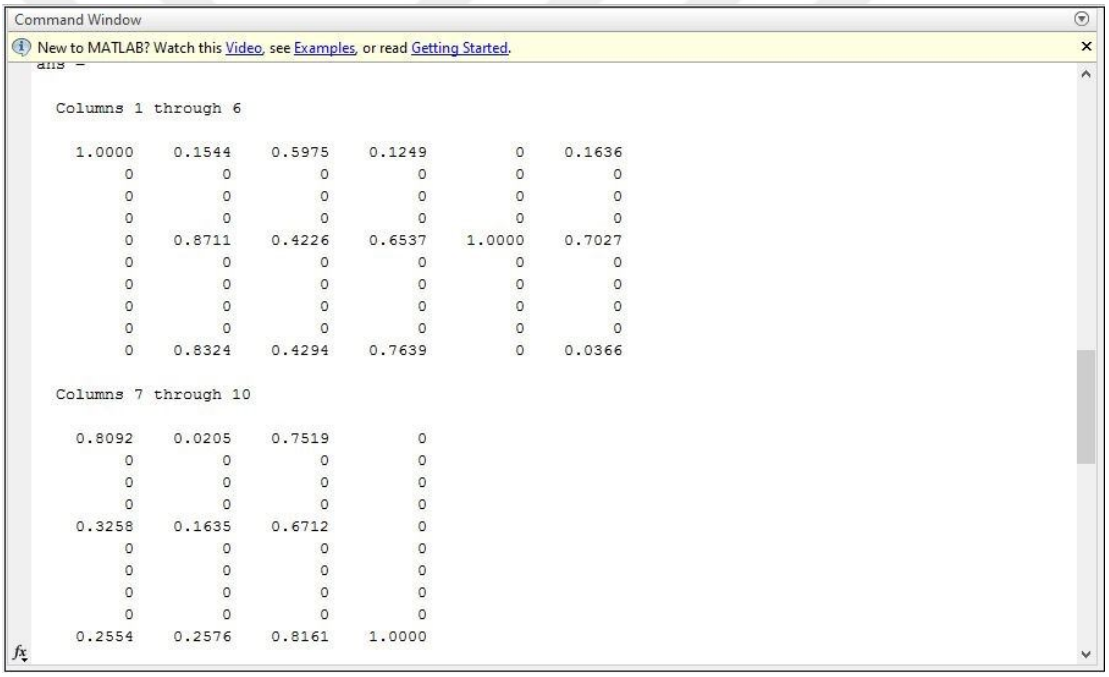
```
function sol=AmirSolution(xhat,model)

    xii=diag(xhat)';

    if any(xii>=0.5)

        [~, so]=sort(xii,'descend');
```

end

$sol.x = x;$

end

Bu aşamada bu fonksiyonu Matlab üzerinde çalıştırdığında Şekil 40'da sonuç gösterilmektedir. Ama önce fonksiyon çalıştırmak için bir model seçilmektedir. Sonra bu model için rastgele xhat matrisi oluşturulmaktadır. Sonra $sol = AmirSolution\,(xhat, model)$ fonksiyonunu çalıştırılmaktadır. Örnek olarak chlap_10 modeli seçilmiştir.



Şekil 40. Sol fonksiyonun Model chlap_10 ilk sonuçları

Üsteki şekilde x_{11} ve x_{55} ve x_{1010} baş düğüm seçilmektedir. Bu aşamadan sonra hangi baş düğümler hangi düğümlere hizmet vereceği belirlenmektedir. Üsteki şekilde her bir sütun bir düğümün indeksleridir, baş düğümler arasında düğümleri kazanması için rekabet vardır yani en fazla değeri olan indeks o satırdaki baş düğümü kazanmaktadır. Üsteki şekilde birinci, beşinci ve onuncu köşegenin elemanları baş düğüm olmaktadır. İlk sütunda birinci indeks kendisi baş düğüm öldüğü için birinci sütunda kendisini kazanmaktadır ve bunun yansıra üçüncü, dördüncü, beşinci, altıncı ve yedinci düğümü kazanmaktadır. Bu düğümler x_{11} baş düğümünden hizmet almaktadır.

x_{55} baş düğümü ikinci düğümü ve kendisini kazanmaktadır. x_{1010} dokuzuncu düğümü ve kendisine hizmet vermektedir. Bu işlemi yapılması için alttaki kodlama yapılmıştır.

$ch = zeros(1, N);$

$for\ i = 1{:}N$

$XI = x(:, i);$

$XI(xii == 0) = -inf;$

$[\sim, c\ h(i)] = max(XI);$

$x(:, i) = 0;$

$x(h(i), i) = 1;$

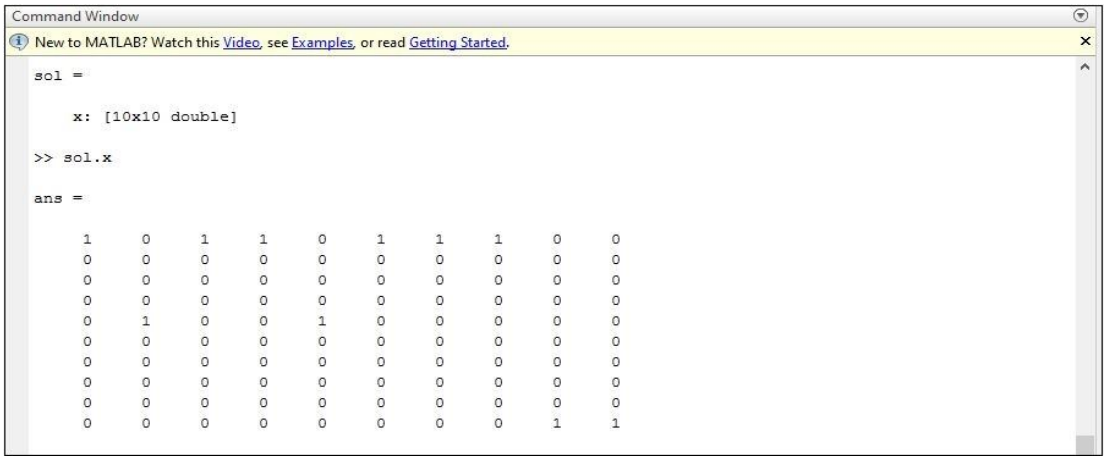
end

end

$sol.x = x;$

end

Bu kodlamadan sonra program çalıştırılmış ve Şekil 42'de sonuçları gösterilmektedir.



Şekil 41. Sol fonksiyonun Model chlap_10 ikinci sonuçları

sol =

ch : [1 5 1 1 5 1 1 1 10 10]

x : [10x10 double]

Chs : [1 5 10]

2.4.4. Amaç Fonksiyonun İçindeki Değerlerin Hesaplanması

Bu çalışmada amaç fonksiyonun içindeki değerler düğümlerin arasındaki veri transferi ve baş düğüm yapılmaktadır düğümlerinin maliyetinden oluşturulmaktadır. Amaç fonksiyonu altta gösterilmektedir.

$$\text{Min } \sum_i \sum_j \sum_k \sum_l x_{ki} x_{lj} c_{ij}^{kl} r_{ij} + \sum_i x_{ii} fi \tag{30}$$

Amaç fonksiyonunun Matlab üzerinde kodlaması için ilk önce c_{ij}^{kl} hesaplanmakta sonra düğümler arasındaki gidiş dünüş veri sayısına çarpılmaktadır. Bu ikisi tüm düğümler için hesaplanır sonra baş düğüm yapılmış düğümlerin toplam maliyet ile toplanılmaktadır. Matlab üzerinde amaç fonksiyon altta kodlanmıştır.

```
oc = zeros(N, N);
for i = 1:N
for j = 1:N
if i == j
oc(i, j) = 0;
else
k = h(i);
l = h(j);
oc(i, j) = c(i, k) + alpha * c(k, l) + c(l, j);
end
end
ocr = oc.* r;
SumOCR = sum(ocr(:));
xiif = xii.* f;
SumXF = sum(xiif);
TotalCost = SumOCR + SumXF;
sol.SumOCR = SumOCR;
sol.SumXF = SumXF;
sol.TotalCost = TotalCost;
end
```

2.4.4.1. Dördüncü Adım Amaç Fonksiyonunun Değerlendirilmesi

Bu adımda bir fonksiyon oluşturulmaktadır. Bu fonksiyon önceki fonksiyonları toplayıp ve problemin belirten koşulların sağlanmaktadır. Bu fonksiyon önceki fonksiyonların cevaplarını alıp ve bir görünebilir cevap vermektedir. Bu fonksiyon Matlab programında kodlaması alta yazılmaktadır.

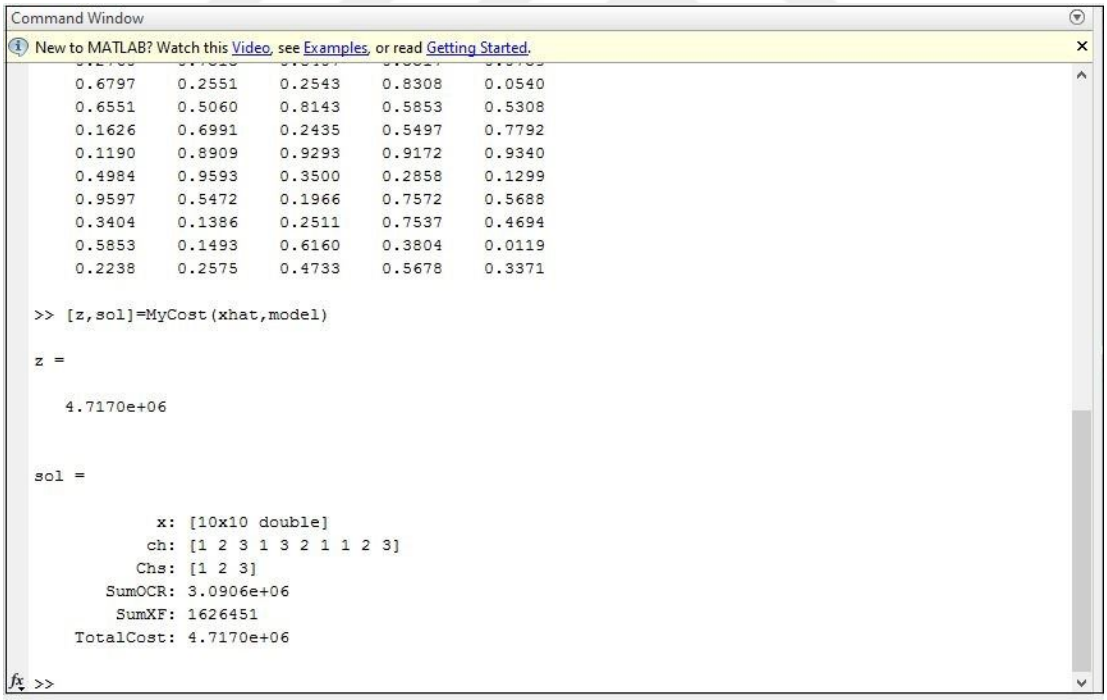
$function\ [z\ sol]\ =\ MyCost(xhat, model)$

$sol\ =\ AmirSolution(xhat, model);$

$z\ =\ sol.TotalCost;$

end

Sonucu göstermek için örnek olarak chlap_10 modeli seçilmektedir. Bu modelin cevabı Şekil 43'de gösterilmektedir.



Şekil 42. Amaç fonksiyonunun chlap_10 modelinin sonuçları

Bu şekilde x matrisi bir 10×10 karar değişken matrisidir. ch hangi düğüm hangi baş düğümden hizmet aldığını göstermektedir. Chs baş düğümleri göstermektedir. Çarpılmış veri transfer ve istek maliyeti SumOCR matrisinde gösterilmektedir. SumXF baş

düğümlerin maliyetlerini göstermektedir. Sonunda TotalCost amaç fonksiyonun maliyetini göstermektedir. TotalCost ve Z aynı olmaktadır.

2.4.4.2. Baş Düğümler İçin Kapasite Tanımlamak

Baş düğümler düğümlerden aldıkları veri hacmine karşı sınırlıdırlar ve bu nedenle verecek hizmetlerinde sınırlı olmaktadır. Gönderilen veri baş düğüm kapasitesinden fazla olduğu zaman bazı düğümler baş düğümlerden hizmet alamamaktadır. Bu problemi aşmak için baş düğümlere hacim kapasitesi tanımlanmaktadır.

$$\sum_i \sum_j x_{ki} r_{ij} \leq G_k \tag{31}$$

Bu denklemde i' ıncı düğümünden k baş düğümüne tüm veri transfer bütün j düğümler için $\sum_j r_{ij}$ la gösterilmektedir. Ve k baş düğümünden hizmet alan düğümleri $\sum_i x_{ki}$'la göstermektedir. Aslında tüm k baş düğümünü seçen i düğümlerin veri transferleri hacmi, k baş düğümünün kapasitesinden(G_k) az ya eşit olması gerekmektedir. Bu koşulla başa çıkmak için bir ihlal (Violation) denklemi k'ıncı baş düğüm için tanıtılmaktadır.

$$\text{Violation} = \begin{cases} 0, & G_k'' \leq G_k \\ \frac{G_k'}{G_k} - 1 & , \ G_k'' > G_k \end{cases} \tag{32}$$

Bir üsteki denklemde ilk taraf G_k' la eşit olmakta ve değişkendir. İkinci taraf ise baş düğümlerin sabit kapasite değerlerini göstermektedir.

Üsteki denklemi k'ıncı baş düğümü için toplanmış hali alttaki denklemde gösterilmektedir.

$$\text{Violation} = \max\left(\frac{G_k'}{G_k} - 1, 0\right) \tag{33}$$

Bu işlemi Matlab üzerinde kodlaması için ilk düğümlerin aralarındaki veri isteği toplanmaktadır. Sonra bulunan rakam baş düğümler arasında bölünür ve ortalama veri kapasite hacmi her baş düğüm için bulunmaktadır. Sonra bu matris değişken olduğu için alt ve üst sınırları belirlenmektedir ve baş düğüm sayısını azaltmak için bulunan kapasite

%35e arttılmaktadır. Bu işlemleri CreateAndSaveModele fonksiyonunda yapılmakta ve altta gösterilmektedir.

$TotalR = sum(r(:));$

$MeanCapacity = TotalR/P;$

$MinCapacity = round(MeanCapacity);$

$MaxCapacity = round(2 * MeanCapacity);$

$G = randi([MinCapacity\ MaxCapacity],1,N);$

$model.G = G;$

Bu işlemler yapıldıktan sonra baş düğümlerin kapasitelerinin güncellenmesi yapılmaktadır. İlk yapılan iş kullanılan ve kullanmayan kapasite baş düğümler için tanımlanmaktadır. Sonra kullanılmayan kapasite azaltıp ve kullanılan kapasite baş düğümler için eklenmektedir. Bu nedenle her bir düğümün baş düğümünün bulduğu yerde o düğümün tüm veri isteğinin miktarı baş düğümünün kapasite miktarına eklenmektedir. Bu işlemler $sol = AmirSolution(xhat, model)$ fonksiyonunda kodlanılmaktadır.

$ch = zeros(1,N);$

$UsedCap = zeros(1,N);$

$for\ i = 1:N$

$XI = x(:,i);$

$XI(xii == 0) = -inf;$

$[\sim, ch(i)] = max(XI);$

$x(:,i) = 0;$

$x(ch(i),i) = 1;$

$UsedCap(ch(i)) = UsedCap(ch(i)) + sum(r(i,:));$

end

$CapV = max(UsedCap./G - 1,0);$

$MeanCapV = mean(CapV);$

$sol.UsedCap = UsedCap;$

$sol.CapV = CapV;$

$sol.MeanCapV = MeanCapV;$

$sol.IsFeasible = (MeanCapV == 0);$

Son aşamada baş düğümlerin kapasitesi fazla olduğu zaman toplam maliyetine eklenecektir. Bu nedenle bu maliyet amaç fonksiyonunda toplam maliyete eklenmektedir.

$function\ [z\ sol] = MyCost(xhat, model)$

$sol = ParseSolution(xhat, model)$;

$w1 = 1$;

$w2 = 5$;

$beta = 100$;

$z = (w1 * sol.SumOCR + w2 * sol.SumXF) * (1 + beta * sol.MeanCapV)$;

2.4.5. Beşinci Adım

Son adımda elde edilen fonksiyon optimizasyon algoritmasına bağlanmaktadır. Optimizasyon algoritması maliyet (Z) miktarı en az olan $xhat$ matrisi bulunmaktadır. Bu $xhat$ matrisinin içinde yaklaşım (sol) bulunmaktadır. Bu yaklaşımın içinde önceki şekilde gösterildiği gibi problemin tüm bilgileri mevcut olmaktadır. Asılında bu kodlama ve optimizasyon algoritmanın arasındaki ortak faktör maliyettir (Z), Az maliyet iyi yaklaşım anlamına gelmektedir.

Bu çalışmada bir 100*100 alanda N sayısında algılayıcı düğümler rastgele dağıtılmıştır ve sonra bir N*N boyutlu matris oluşturulmuş. Bu matrisin içindeki tüm düğümler 2 boyutlu olmaktadır. Baş düğümler bu matrisin köşegeninden seçilip hangi düğüm hangi baş düğümden hizmet alacağı bu matris üzerinden belirlenmektedir. Bu matris aslında karar değişkenidir. Sonra düğümlerin aralarındaki mesafeler hesaplanıp bu mesafelere göre seçilen baş düğümler ve düğümler aralarında veri transfer maliyetleri hesaplanmaktadır. Son aşamada bir amaç maliyet fonksiyonu oluşturulmuştur. Bu maliyet amaç fonksiyonu minimum cevap bulmak için optimizasyon algoritmaları kullanılmaktadır.

2.4.5.1. Tanımlanan Problemin Diferansiyel Gelişim Algoritması (DEA) ile Çözümü

Tanımlanan problemi Diferansiyel Gelişim algoritması ile çözmek için ilk başta problemin parametrelerin tanımlanacaktır.

$model = SelectModel()$; $\%\ Select\ Model$

$CostFunction = @(xhat)\ MyCost(xhat, model)$; $\%\ Cost\ Function$

$VarSize = [model.N\ model.N]$; $\%\ Decision\ Variables\ Matrix\ Size$

$nVar = prod(VarSize)$; $\%\ Number\ of\ Decision\ Variables$

İkinci adımda Diferansiyel Gelişim algoritmasının parametreleri tanımlanmaktadır. Bu kısımda önemli olan parametreler, iterasyon sayısı, popülasyon sayısı, bölme katsayısı ve çaprazlama ihtimali bulunmaktadır.

$MaxIt$ = 100; % $Maximum\ Number\ of\ Iterations$

$nPop$ = 200; % $Population\ Size$

$beta\_min$ = 0.8; % $Lower\ Bound\ of\ Scaling\ Factor$

$beta\_max$ = 1.5; % $Upper\ Bound\ of\ Scaling\ Factor$

Üçüncü adımda ilk popülasyonu oluşturulmaktadır ve tüm popülasyon bireylerine konum, maliyet ve bir sol özelliği tanımlanmaktadır.

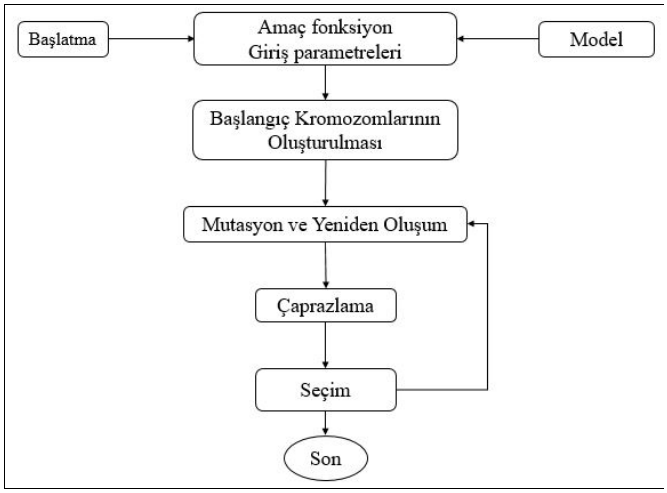
$empty\_individual.Position$ = [];

$empty\_individual.Cost$ = [];

$empty\_individual.Sol$ = [];

Programın bu kısmında tüm bireyler (cevapların) arasında en iyi olan birey seçilmektedir.

Sonra mutasyon işlemi ve çaprazlama işlemi yapılmaktadır. Bu kısımda mutasyon operatörü yeni bir cevap oluşturur. Çaprazlama işlemi bu yeni cevap ve eski cevap üzerine yapılır ve yeni bir cevap oluşturulur. Bu yeni cevap ilk bulunan en iyi cevapla karşılaştırılıyor. Hangi cevap iyiyse o cevap bizim solüsyonumuz oluyor. Problemin yaklaşımının akış diyagramı Şekil 44'te gösterilmektedir.



Şekil 43. Problemin diferansiyel gelişim algoritması ile çözümü için akış diyagramı

2.4.5.2. Tanımlanan Problemin Parçacık Sürü Optimizasyon Algoritması (PSO) ile Çözümü

Tanımlanan problemi Parçacık Sürü algoritması ile çözmek için ilk başta problemin parametreleri tanımlanmaktadır.

$model = SelectModel();$ % $Select\ Model$

$CostFunction = @(xhat)\ MyCost(xhat, model);$ % $Cost\ Function$

$VarSize = [model.N\ model.N];$ % $Decision\ Variables\ Matrix\ Size$

$nVar = prod(VarSize);$ % $Number\ of\ Decision\ Variables$

$VarMin = 0;$ % $Lower\ Bound\ of\ Decision\ Variables$

$VarMax = 1;$ % $Upper\ Bound\ of\ Decision\ Variables$

İkinci adımda Parçacık Sürü algoritmasının parametreleri tanımlanmaktadır. Bu kısımda önemli olan parametreler, iterasyon sayısı, popülasyon sayısı, Atalet ağırlık değeri ve ölçeklendirme faktörleri bulunmaktadır.

$MaxIt = 200;$ % $Maximum\ Number\ of\ Iterations$

$nPop = 200;$ % $Population\ Size\ (Swarm\ Size)$

$w = 0.4;$ % $Inertia\ Weight$

$wdamp = 1;$ % $Inertia\ Weight\ Damping\ Ratio$

$c1 = 0.3;$ % $Personal\ Learning\ Coefficient$

$c2 = 0.9;$ % $Global\ Learning\ Coefficient$

Üçüncü adımda her bir parçacık için konum, maliyet, sol, hız, en iyi konum, en iyi maliyet ve en iyi sol özelliği tanımlanmaktadır.

$empty\_particle.Position = [];$

$empty\_particle.Cost = [];$

$empty\_particle.Sol = [];$

$empty\_particle.Velocity = [];$

$empty\_particle.Best.Position = [];$

$empty\_particle.Best.Cost = [];$

$empty\_particle.Best.Sol = [];$

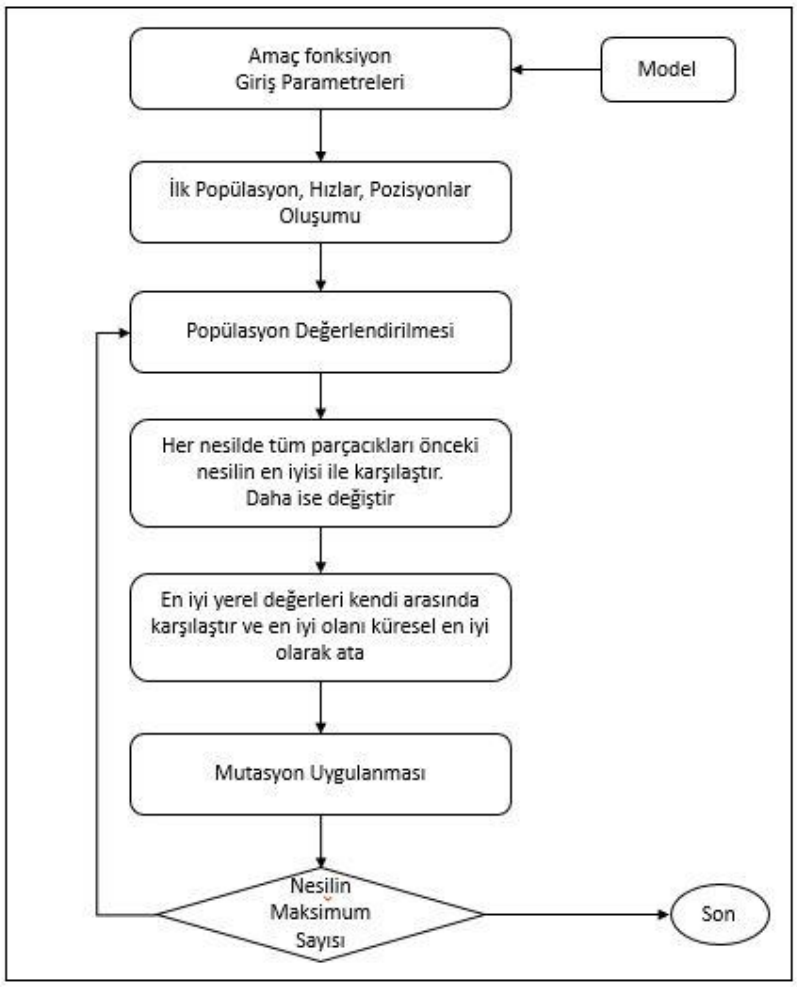
Programın bu kısmında popülasyon sayısında parçacık oluşturulmaktadır (bir $nPop \times 1$ matriste).

Amaç fonksiyonunu değerlendirme kısmında her parçacık için sol ve maliyet hesaplanmaktadır. Bulunan iyi parçacıklara da sol tanımlanmaktadır.

$$[particle(i).Costparticle(i).Sol] = CostFunction(particle(i).Position);$$

$$particle(i).Best.Sol = particle(i).Sol;$$

Algoritmanın üçüncü kısmında her nesilde tüm parçacıkları önceki neslin en iyisiyle karşılaştırılmaktadır. Eğer cevap iyiyse yer değiştirilmektedir. Sonra en iyi yerel değerleri kendi arasında karşılaştırır ve en iyi olanı küresel en iyi olarak atamaktadır. Sonra hız ve pozisyonları hesaplanmaktedır ve mutasyon uygulanır. Daha sonra her paracığın geçirdiği deneyimleri güncellenmektedir. Neslin maksimum sayıya ulaşırsa algoritmanın işi sonlandırılmaktadır. Problemin Parçacık Sürü Algoritması ile çözümünün akış diyagramı Şekil 45'te gösterilmektedir.



Şekil 44. Problemin Parçacık Sürü algoritması ile çözümü için akış diyagramı

2.4.5.3. Tanımlanan Problemin Yayılmacı Yarışmacı Algoritma (ICA) ile Çözümü

Tanımlanan problemi Yayılmacı Yarışmacı algoritması ile çözmek için problemin parametreleri tanımlanmaktadır.

$\%\%\ Problem\ Definition$

$model\ =\ SelectModel();\qquad \%\ Select\ Model$

$CostFunction\ =\ @(xhat)\ MyCost(xhat, model);\quad \%\ Cost\ Function$

$VarSize\ =\ [model.N\ model.N];\quad \%\ Decision\ Variables\ Matrix\ Size$

$nVar\ =\ prod(VarSize);\qquad \%\ Number\ of\ Decision\ Variables$

$VarMin\ =\ 0;\qquad \%\ Lower\ Bound\ of\ Decision\ Variables$

$VarMax\ =\ 1;\qquad \%\ Upper\ Bound\ of\ Decision\ Variables$

İlk başta problemin modellerini ve amaç fonksiyon tanımlanımaktadır. Sonra karar değişkenleri boyutu tanımlanmaktadır. Sonunda karar değişkenlerin sayısı ve miktarları tanımlanmaktadır.

İkinci adımda Yayılmacı Yarışmacı algoritmasının parametreleri tanımlanmaktadır. Bu kısımda önemli olan parametreler, iterasyon sayısı, popülasyon ve imparatorluk sayısı, asimilasyon katsayısı(koloninin yayılmacıya yaklaşım katsayısı), alpha (seçim baskısı), devrim Olasılığı, devrim Oranı ve koloniler ortalama maliyet katsayısı (sonuca etki eden) faktörleri bulunmaktadır.

$MaxIt\ =\ 400;\qquad \%\ Maximum\ Number\ of\ Iterations$

$nPop\ =\ 100;\qquad \%\ Population\ Size$

$nEmp\ =\ 10;\qquad \%\ Number\ of\ Empires/Imperialists$

$alpha\ =\ 1;\qquad \%\ Selection\ Pressure$

$beta\ =\ 2;\qquad \%\ Assimilation\ Coefficient$

$pRevolution\ =\ 0.3;\quad \%\ Revolution\ Probability$

$mu\ =\ 0.05;\qquad \%\ Revolution\ Rate$

$zeta\ =\ 0.1;\qquad \%\ Colonies\ Mean\ Cost\ Coefficient$

Programın bu kısmında her ülke için konum, maliyet ve sol özelliği bir ham yapıda tanımlanmaktadır.

$empty\_country.Position\ =\ [];$

$empty\_country.Cost\ =\ [];$

$empty\_country.Sol\ =\ [];$

Her ülkeye üstteki özellikler tanımladıktan sonra popülasyon sayısında ülke oluşturulmaktadır. Oluşturulmuş her ülke için yeni konumlar oluşturulmaktadır. Sonra ülkerlerin konumların amaç fonksiyonunda değerlendirilir. Çıkışta her ülke için bir maliyet ve sol hesaplanmaktadır.

$for\ i = 1{:}nPop$

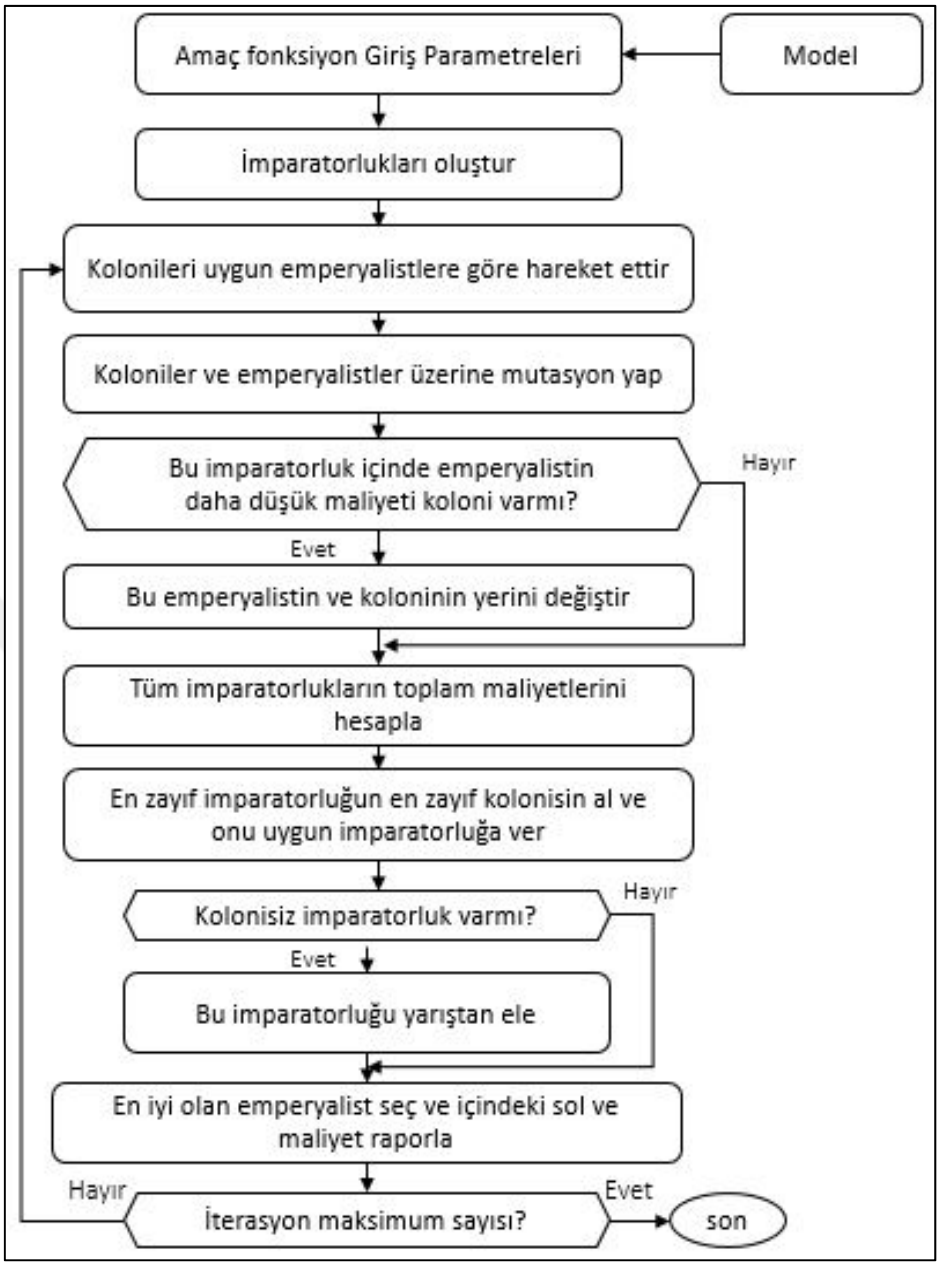
$country(i).Position = unifrnd(VarMin, VarMax, VarSize);$

$[country(i).Cost\ country(i).Sol] = CostFunction(country(i).Position);$

end

En iyi olan ülkelerden birkaçı yayılmacı seçilmektedir. Değer ülkeler koloni olmaktadır. Sonra yayılmacı sayısında imparatorluk oluşturulmaktadır. Sonra koloniler imparatorlukların arasında bölünmetedir. En Son aşamada imparatorluğun toplam maliyeti hesaplanmaktadır. Böylece imparatorluğun ilk popülasyonu oluşturumaktadır.

İmparatorluğun ilk popülasyonu oluşturulduktan sonra koloniler uygun yayılmacılara göre hareket etmektedir. Üçüncü adımda yayılmacıların ve koloniler üzerinde mutasyon yapılmaktadır. Mutasyon yapılan koloni amaç fonksiyonunda değerlendirilmektedir ve çıkışında bir maliyet ve sol hesaplanmaktadır. Sonra her imparatorluk içinde yayılmacı ve koloniler arasında yarış başlanmaktadır. Bu imparatorluk içinde yayılmacıdan daha düşük maliyeti olan koloni varsa yayılmacı ve kolonin yeri değiştirilmektedir. Sonra tüm imparatorlukların toplam değeri hesaplanmaktadır. Sonra en zayıf imparatorluğun en zayıf kolonisi alınmaktadır. Alınan koloni en uygun imparatorluğa verilmektedir. Sonra her nesilde en iyi buluna yayılmacı güncellenmekte ve en iyi yaklaşım olarak kayd edilmektedir. Bu işlemler iterasyon sayısı kadar devam edilmektedir. Problemin Yayılmacı Yarışmacı algoritması ile çözümünün akış diyagramı Şekil 46'te gösterilmektedir.

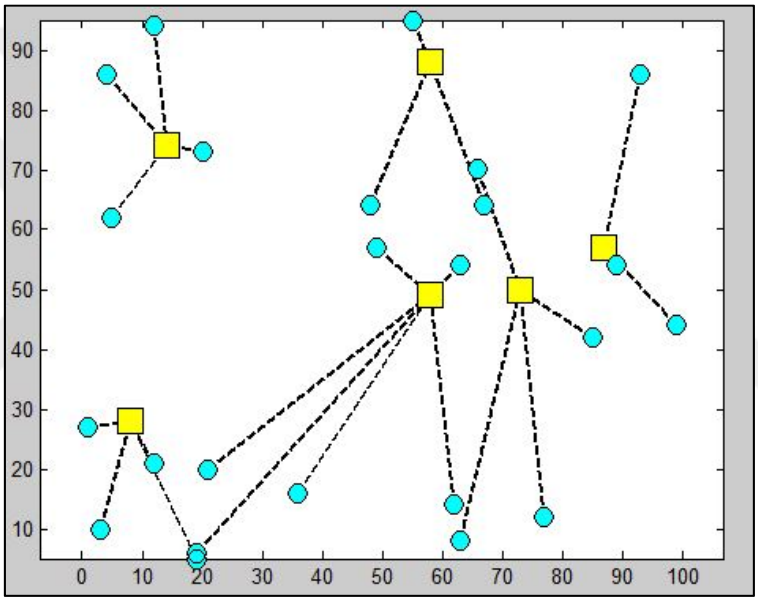


Şekil 45. Problemin yayılmacı yarışmacı algoritması ile çözümü için akış diyagramı

3. BULGULAR

3.1. Problemin Diferansiyel Gelişim Algoritması ile Çözümü ve Bulguları

$W_1 = 1$, $W_2 = 2$ ve iterasyon sayısı 500 parametrelere göre programın problemin 30*30 boyutuna koşturulduğunda elde edilen sonucunun grafiği aşağıdaki Şekil47'da gösterildiği gibi elde edilmektedir.



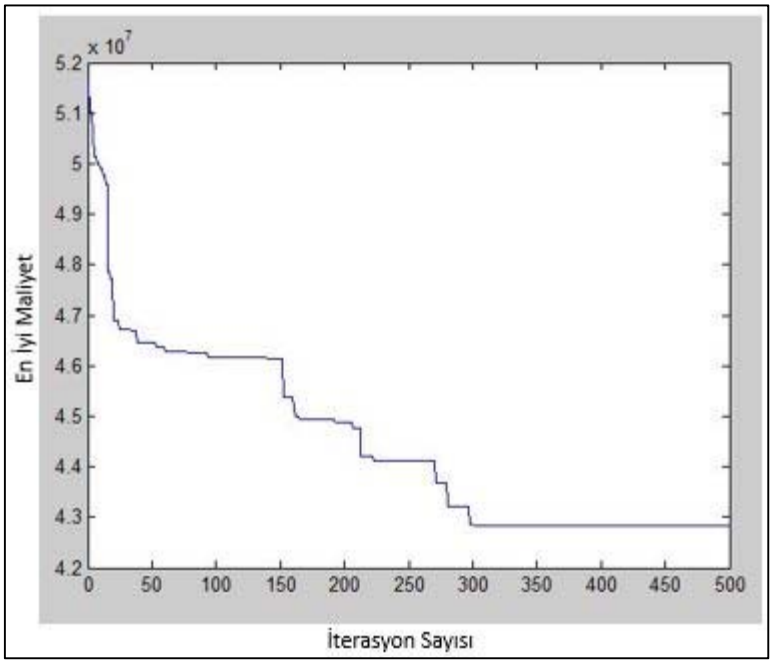
Şekil 46. Problemin diferansiyel gelişim algoritması ile çözümünün grafik sonucu

Baş düğümler sarı kareler ve diğerleri düğümleri göstermektedir. Oluşan kümeler ($cluster$) şeklinde görülmektedir. Her düğüm en yakın kümesine bağlanmıştır ama her baş düğümün belli kapasitesi bulunmktadır ve bu kapasiteyi aştıktan sonra baş düğüme bağlanmış olmayan düğümler kapasitesini aşmayan baş düğümlere bağlanmaktadır. Amaç fonksiyondaki W_1 ve W_2 sırası ile dürümlerin veri transfer ve baş düğümlerin kuruluş maliyetlerini katsayısını temsil etmektedir. Baş düğüm kuruluş maliyeti artarsa baş düğüm sayısı azalmaktadır. Bu bulgunun detaylı raporu Şekil 48'de gösterilmektedir.

```
BestSol =

    Position: [30x30 double]
        Cost: 4.2837e+07
         Sol: [1x1 struct]

>> BestSol.Sol

ans =

            x: [30x30 double]
           ch: [11 2 9 5 5 14 14 5 9 14 11 2 2 14 11 5 9 11 5 2 9 9 28 9 2 28 28 28 9 28]
          Chs: [2 5 9 11 14 28]
       SumOCR: 1.7044e+07
        SumXF: 12896525
```

Şekil 47. Problemin diferansiyel gelişim algoritması ile çözümünün raporu

Bu şekilde en iyi bulunan sonucun maliyeti, konum ve yaklaşımı gössterilmektedir. Yaklaşımın (sol) içinde problemin tüm detaylı mevcut bulunmaktadır. En iyi bulunan karar değişkeni hangi düğüm hangi baş düğümden servis aldığı ve hangi düğümler baş düğüm seçilmesi gösterilmektedir. En sonda sırasıyla veri transfer ve baş düğümlerin kuruluş maliyetlerini gösterilmektedir. Bu problemin her iterasyonun da bulunan maliyet Şekil 49'de gösterilmektedir.

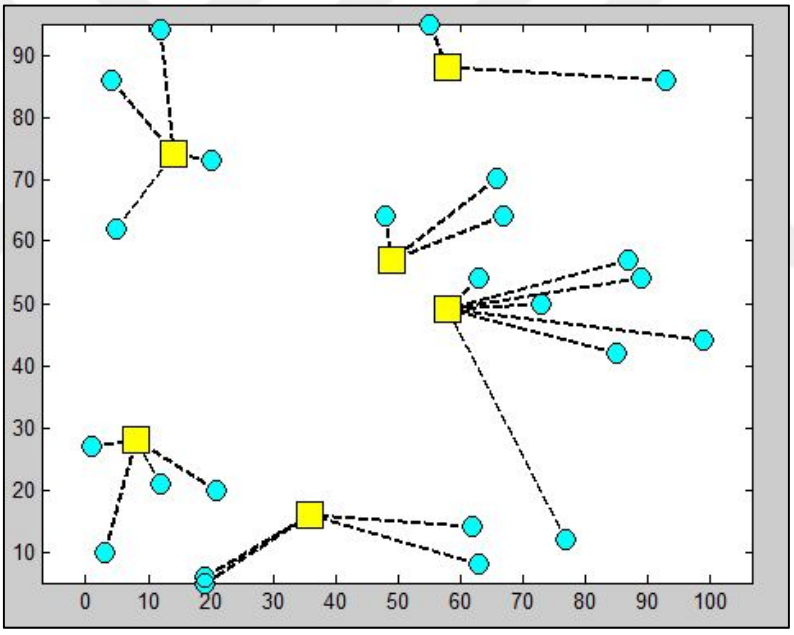


Şekil 48. Problemin diferansiyel gelişim algoritması ile bulunan maliyet raporu

Şekil 48'e göre program çalıştığında maliyet azalmıştır ve 300'üncü iterasyondan minimum maliyet elde edilmektedir. 300'üncü iterasyondan sonra maliyet sabit kalmaktadır bu nedenle algoritma bu iterasyondan sonra çalışmasını hiç önemi bulunmamaktadır.

3.2. Problemin Parçacık Sürü Optimizasyon Algoritması ile Çözümü ve Bulguları

$W_1 = 1$, $W_2 = 2$ ve iterasyon sayısı 500 parametrelere göre programın problemin 30*30 boyutuna koşturulduğunda elde edilen sonucunun grafiği aşağıdaki Şekil 50'de gösterildiği gibi elde edilmektedir.



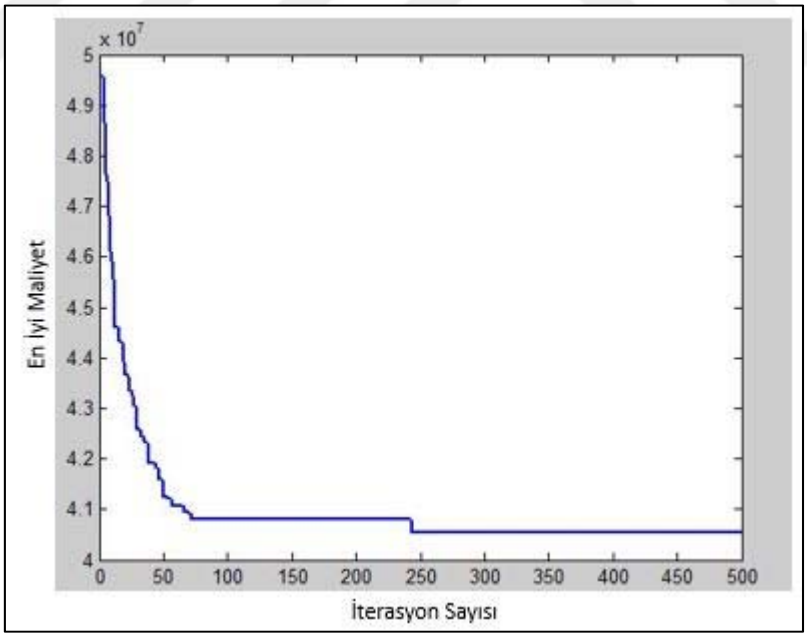
Şekil 49. Problemin parçacık sürü optimizasyon algoritması ile çözümünün grafik sonucu

Baş düğümlerin seçimi her iterasyondaki amaç fonksiyonunun değerine bağlı değişmektedir ve son hali Şekil 50'de gösterilmektedir. Bulunan bulgunun detaylı raporu Şekil 51'de gösterilmektedir.

```
BestSol =

    Position: [30x30 double]
        Cost: 4.0530e+07
         Sol: [1x1 struct]

>> BestSol.Sol

ans =

            x: [30x30 double]
           ch: [24 2 9 24 9 9 9 9 9 11 11 2 2 9 24 9 2 11 22 2 22 22 28 24 22 28 28 28 22 28]
          Chs: [2 9 11 22 24 28]
       SumOCR: 1.5307e+07
        SumXF: 12611172
```

Şekil 50. Problemin parçacık sürü optimizasyon algoritması ile çözümünün raporu

Sekil 49'a göre en iyi yaklaşımın maliyeti 4,0530*10^7 olmaktadır. Bu yaklaşımda 2, 9, 11, 22, 24, 28'inci düğümler baş düğüm seçilmektedir. Ve hangi düğümün hangi baş düğümden servis aldığı belli edilmektedir. Bu problemin her iterasyonun da bulunan maliyet raporu Şekil 52'de gösterilmektedir.



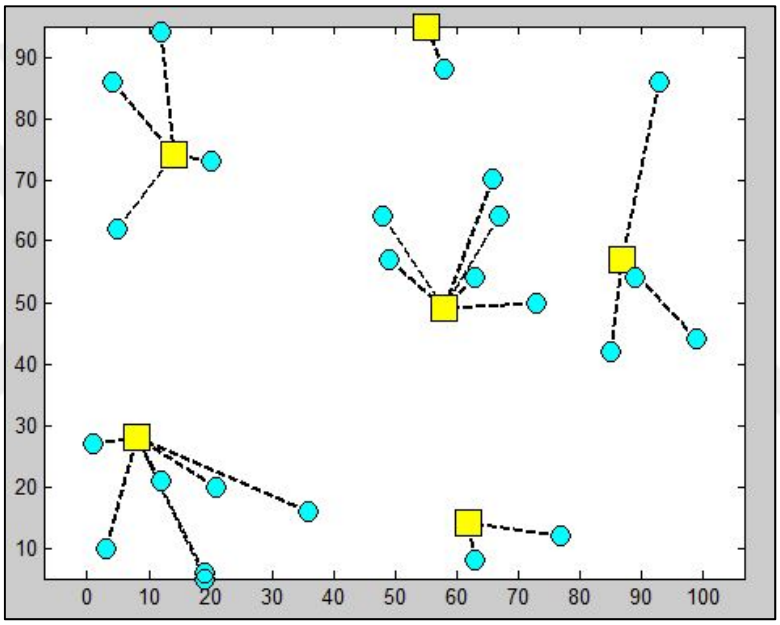
Şekil 51. Problemin parçacık sürü optimizasyon algoritması ile bulunan maliyet raporu

Şekil 52'ye göre problemin maliyeti azalmakta ve yaklaşık olarak 250'inci iterasyondan minimum maliyet elde edilmiştir. 250'inci iterasyondan sonra maliyet sabit

kalmaktadır bu nedenle algoritma bu iterasyondan sonra çalışmasını hiç önemi bulunmamaktadır.

3.3. Problemin Yayılmacı Yarışmacı Algoritma ile Çözümü ve Bulguları

$W_1 = 1$, $W_2 = 2$ ve iterasyon sayısı 500 parametrelere göre programın problemin 30*30 boyutuna koşturulduğunda elde edilen sonucunun grafiği aşağıdaki Şekil 53'de gösterildiği gibi elde edilmektedir.



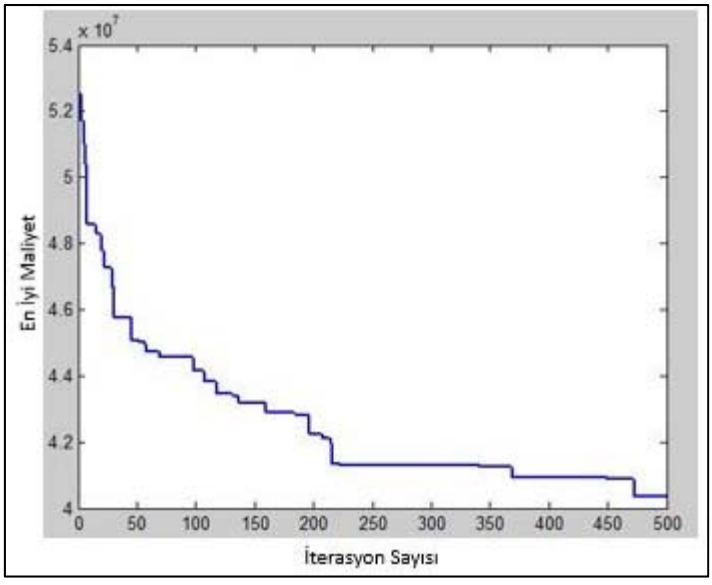
Şekil 52. Problemin yayılmacı yarışmacı algoritma ile çözümünün grafik sonucu

Aynı diğer algoritmalar gibi burada da 6 tane baş düğümü seçilmektedir. Bulunan bulgunun detaylı raporu Şekil 54'te gösterilmektedir.

```
>> BestSol

BestSol =

    Position: [30x30 double]
        Cost: 4.0358e+07
         Sol: [1x1 struct]

>> BestSol.Sol

ans =

            x: [30x30 double]
           ch: [9 2 9 9 9 14 14 21 9 14 18 2 2 14 9 14 2 18 21 2 21 2 28 9 2 28 28 28 2 28]
          Chs: [2 9 14 18 21 28]
        SumOCR: 1.4980e+07
         SumXF: 12688957
     TotalCost: 2.7669e+07
       UsedCap: [0 6909 0 0 0 0 0 0 6148 0 0 0 0 4197 0 0 0 1560 0 0 2662 0 0 0 0 0 0 4404 0 0]
          CapV: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
      MeanCapV: 0
    IsFeasible: 1
```

Şekil 53. Problemin yayılmacı yarışmacı algoritma ile çözümünün raporu

Sekil 54'e göre en iyi yaklaşımın maliyeti 4,0358*10^7 olmaktadır. Bu yaklaşımda 2, 9, 11, 18, 21, 28'inci düğümler baş düğüm seçilmektedir. Ve hangi düğümün hangi baş düğümden servis aldığı ch matrisinde belli edilmektedir. Ayrıca baş düğümlerin kullanılmış kapasiteleri $UsedCap$ matrisin içinde gösterilmektedir. Bu problemin her iterasyonun da bulunan maliyet Şekil 55'de gösterilmektedir.



Şekil 54. Problemin yayılmacı yarışmacı algoritma ile bulunan maliyet raporu

Şekil 55'e göre program çalıştığında maliyet azalmaktadır ve 476'ıncı iterasyondan minimum maliyet elde edilmektedir.

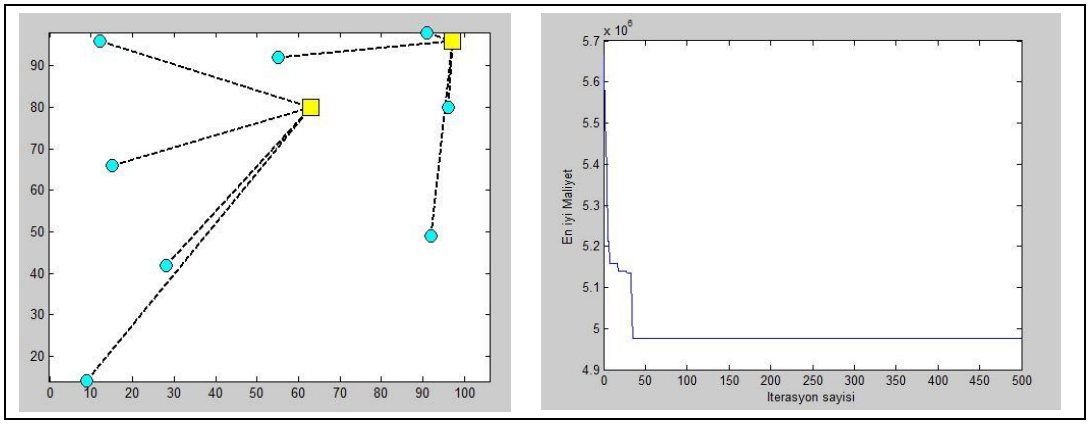
3.4. Algoritmaların Karşılaştırılması

Bu çalışmada algoritmalar tanımlanan problemin 10*10, 50*50 ve 100*100 boyutlarında ve en zor koşullarda koşturulmuştur. Giriş parametreler tüm algoritmalar için aynı olmaktadır. Elde edilen sonuçları Tablo 6'da gösterilmektedir. Bu çalışmada tanımlanan problem Asus İntel Core İ5 ve 1,6 GHz özellilkli olan bilgisayarda çalıştırılmıştır.

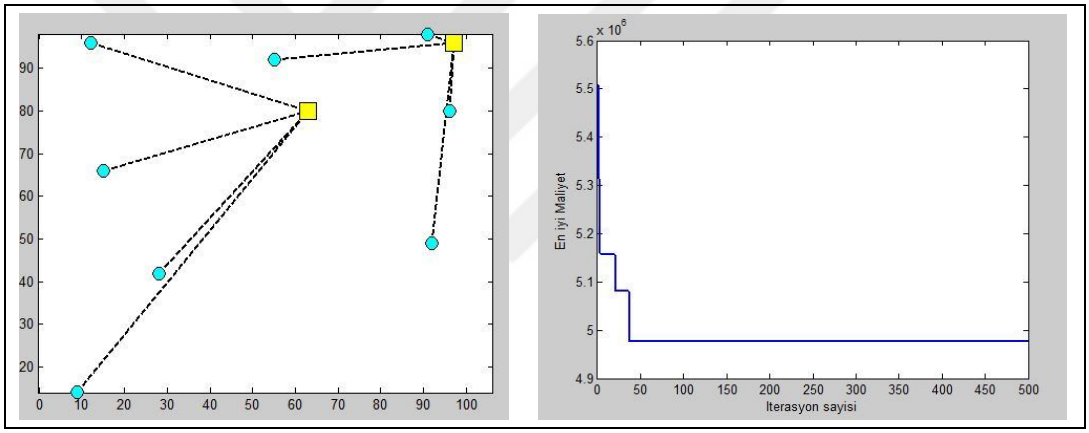
Tablo 6. Problemin DG, PSO, YYA ile çözülen cevapları

Problemin boyutu ve İterasyon sayısı	**Algoritma**	**Minimum maliyet**	**Maliyet normalizesi**	**Zaman**	**Baş düğüm sayısı**
10*10 500 İterasyon	DG	4976163	0.14	85.577732 Saniye	2
	PSO	4976163	0.15	207.317267 Saniye	2
	YY	4976163	0.16	53.85675 Saniye	2
50*50 1000 İterasyon	DG	1.1292e+08	0.24	447.856900 Saniye	12
	PSO	1.0719e+08	0.27	955.395774 Saniye	12
	YY	1.0861e+08	0.26	255.311346 Saniye	12
100*100 1000 İterasyon	DG	6.2311e+08	0.21	907.302383 Saniye	18
	PSO	5.6621e+08	0.29	1995.89954 Saniye	18
	YY	5.8031e+08	0.27	538.562477 Saniye	18

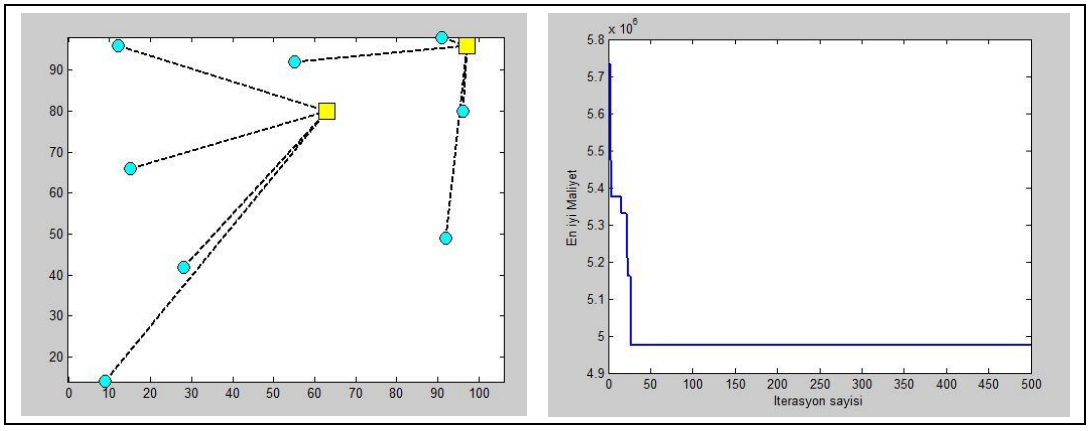
Tablo 6'da elde edilen sonuçların grafikleri ve maliyeti Şekil 56-64'de gösterilmektedir.



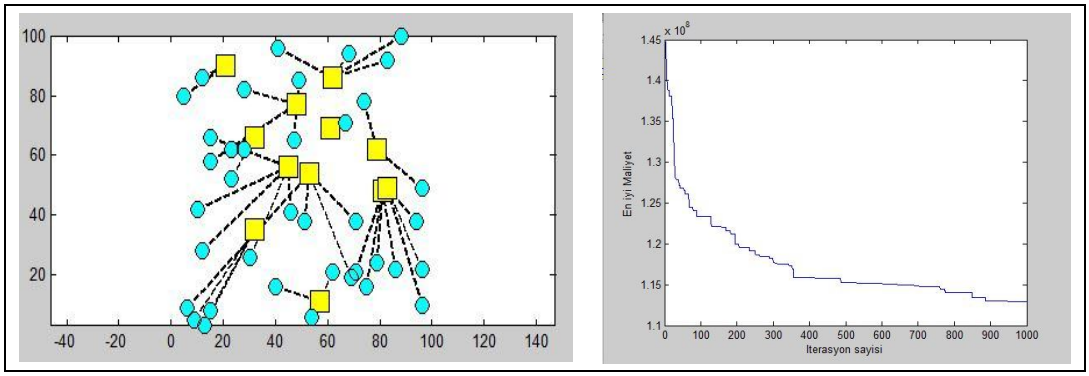
Şekil 55. Problemin 10*10 boyutuna DGA ile elde edilen sonucunun grafiği ve maliyeti



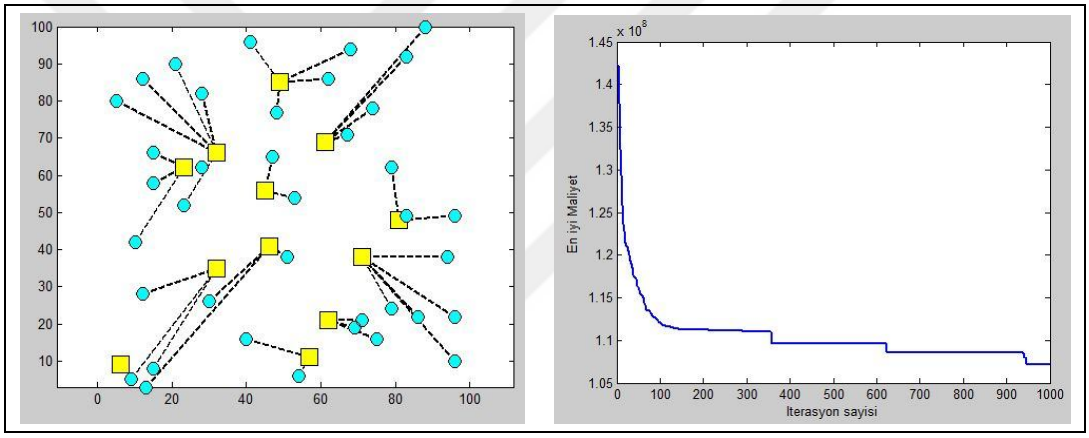
Şekil 56. Problemin 10*10 boyutuna PSOA ile elde edilen sonucunun grafiği ve maliyeti



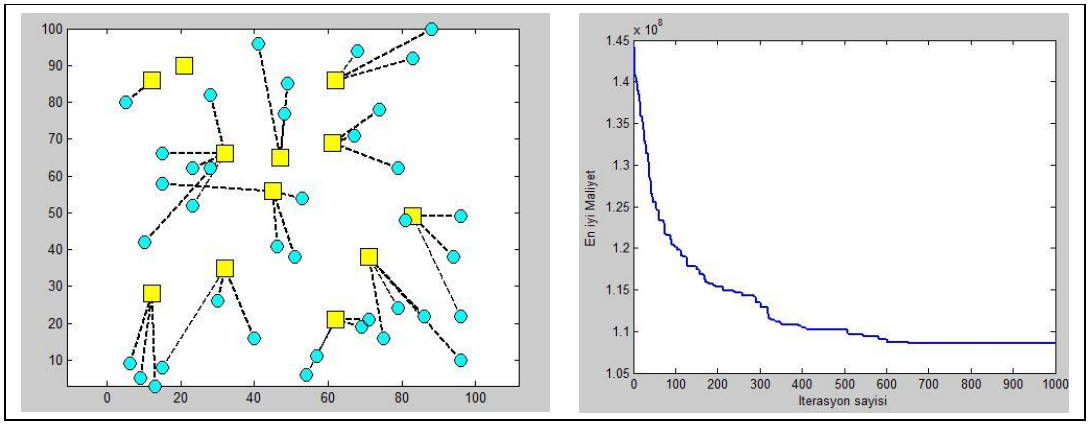
Şekil 57. Problemin 10*10 boyutuna YYA ile elde edilen sonucunun grafiği ve maliyeti



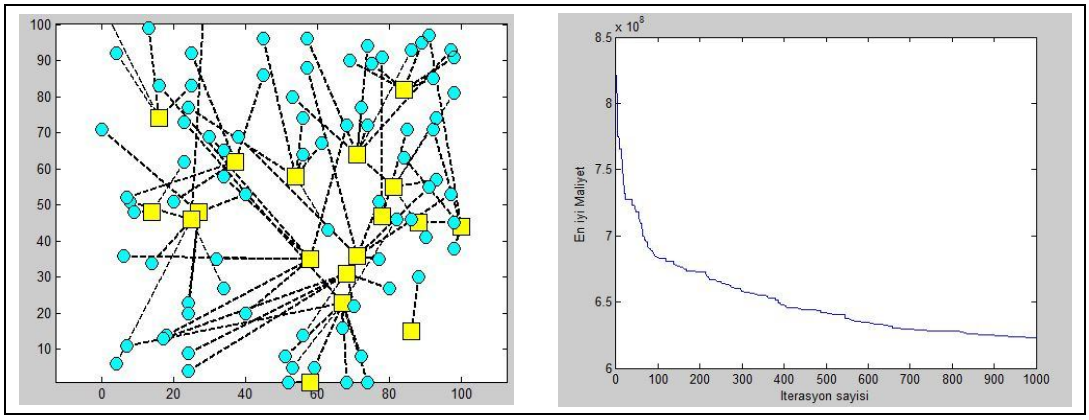
Şekil 58. Problemin 50*50 boyutuna DGA ile elde edilen sonucunun grafiği ve maliyeti



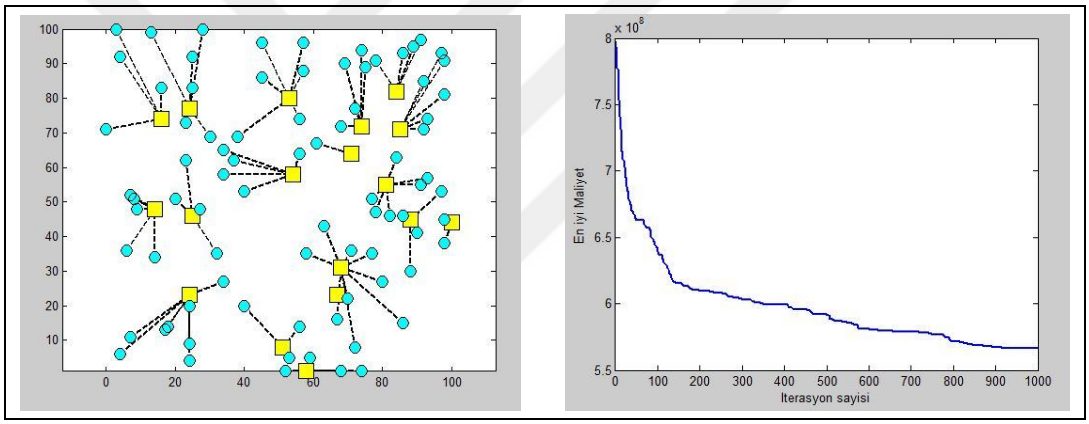
Şekil 59. Problemin 50*50 boyutuna PSOA ile elde edilen sonucunun grafiği ve maliyeti



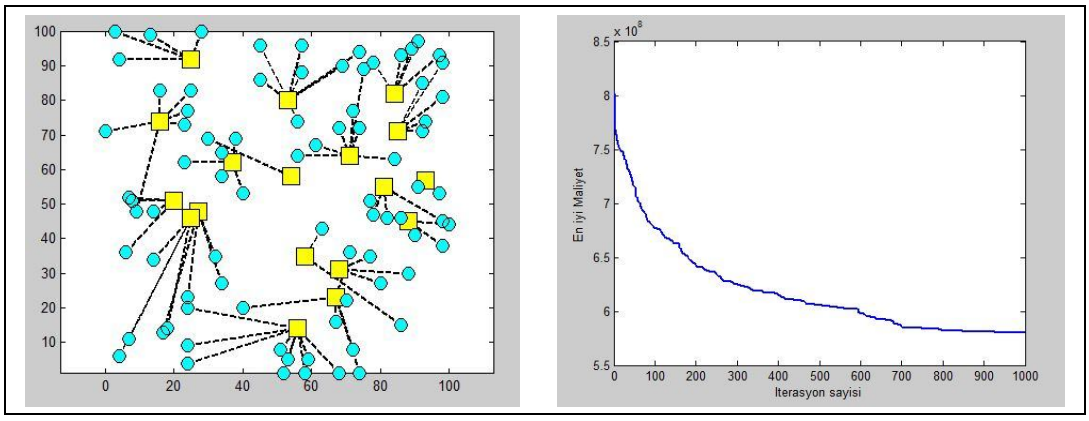
Şekil 60. Problemin 50*50 boyutuna YYA ile elde edilen sonucunun grafiği ve maliyeti



Şekil 61. Problemin 100*100 boyutuna DGA ile elde edilen sonucunun grafiği ve maliyeti



Şekil 62. Problemin 100*100 boyutuna PSOA ile elde edilen sonucunun grafiği ve maliyeti



Şekil 63. Problemin 100*100 boyutuna YYA ile elde edilen sonucunun grafiği ve maliyeti

4. TARTIŞMA

Bu çalışmada Yapılandırılmış ağ yapısı bir matematiksel ve ideal yapılmaktadır. Gerçek hayatta bu yapının kullanımı için ne yapılabilimesi hususu tartışmaya açıktır. Örnek olarak baş düğüm kuruluş maliyetinin yerine baş düğüml pillerinin ömrü ve bunun gibi bir çok çalışma yapılmaktadır.

Aslında tasarlanan yapı için bir akılı model uygulanmaktadır, bu model eski LEACH gibi modeller göre daha üstündür zira tasarlanan yeni modelde tanımlanan problemi optimizasyon algoritmaların çözümü ile hem adaptif kümeleme hem de optimum küme başı seçimi gerçekleştirilmektedir.

5. SONUÇLAR

Tanımlanan problem üç tane farklı algoritma ile koşturulmaktadır. Problemin küçük boyutlarında DG ve YY algoritması PSO algoritmasından daha hızlı performansa sahip olmaktadır. Problemin küçük boyutlarında DG ve YY algoritmaların arasında performans açısından çök az fark görülmektedir. Problemin büyük boyutlarında durum değişmektedir PSO ve YY algoritması DG algoritmasından, ağ maliyetini daha düşük yapmaktadır. Ağ maliyetini minimize etme açısından PSO ve YY algoritmaların arasında çök az fark bulunmaktadır. Zaman açısından YY algoritması PSO algoritmasına göre daha hızlı performansa sahip olmaktadır.

Tanımlanan model tek amaçlı olan problemin çözümü için YY algoritması ile daha hızlı performans ve iyi sonuçlar göstermektedir.

6. ÖNERİLER

Önerilen modelde düğümlerin konumları sabit varsayılmaktadır. Düğümlerin konumları dinamik olduğu zaman ek özellikleri tanımlayarak dinamik düğümler için önerilen modeli dinamik kablosuz algılayıcı ağlarına sunula bilmektedir. Dinamik yaklaşım için baş düğümlerine belirli bir yarıçap parametresi eklene bilmektedir ve komşular ile yeni baş düğüm seçim için müzakere girile bilmektedir.

7. KAYNAKLAR

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y. ve Cayirci, E., Wireless Sensor Networks-A Survey, Computer Networks, 38,4 (2002) 393-422.

2. Saleem, M., Din Caro, G. ve Farooq, M., Swarm Intelligence Based Routing Protocol for Wireless Sensor Networks: Survey and Future Directions, Inforation Scinces , 181,20 (2011) 4597-4624.

3. Yaacoub, E. ve Abu-Dayya, A., Multi-Hop Routing for Routing for Energy Efficiency in Wireless Sensor Networks, INTECH Open Access Publisher, (2012) 165-186.

4. Ganz, A., Ganz, Z. ve Wongthavarawat, K., Multimedia Wireless Networks: Technologies, Standards and QoS, Upple Saddle River, New Jersey, 2003.

5. Xia, F., Wireless Sensor Technologies and Applications, Moleculer Diversity Preservation International, Published by MDPI AG, 9,11 (2009) 8824-8830.

6. Arampatzis, T., Lygeros, J. ve Manesis, S., A Survey of Applications of Wireless Sensors and Wireless Sensor Networks. 13th Mediterrean Conference on Control and Automation Intelligent Control, June 2005, Limassol, IEEE, 719-724.

7. Gangal, V., Kablosuz Algılayıcı Ağlarda Karınca Koloni Algoritmasıyla Rotalama ile Enerji Etkin Rotalamanın İncelemesi, Yüksek Lisans Tezi, K.T.Ü., Fen Bilimler Enstitüsü, Trabzon, 2015.

8. Tahir Emre, K., Kablosuz Algılayıcı düğüm Ağlar ve Uygulamaları, XI. Akademik Bilişim Konferansı, Şubat 2009, Şanlıurfa, Akademik Bilişim, 37-46.

9. Yang, Y., Information Processing and Routing in Wireless Sensor Networks, World Scientific Publishing Co., River Edge, 2007.

10. YÜKSEL, M. E., YÜKSEL, A. S. ve İBİLOĞLU, R., Kablosuz Algılayıcı Ağlar için Güç Tasarruflu Ağ Geçidi Tasarımı. Mehmet Akif Ersoy Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 6,1 (2015) 24-30.

11. http://www.ee.oulu.fi/~carlos/WSNPapers/AK02.pdf. Computer Networks. 15 March 2002.

12. Yick, J., Mukherjee, B. ve Ghosal, D., Wireless Sensor Network Survey, Computer Networks, 52,12 (2008) 2292-2330.

13. Dargie, W. ve Poellabauer, C., Fundamentals of Wireless Sensor Networks: Theory and Practice. John Wiley & Sons, 2010.

14. Winkler, M., Tuchs, K. D., Hughes, K. ve Barclay, G., Theoretical and Practical Aspects of Military Wireless Sensor Networks. Journal of Telecommunications and Information Technology, (2008) 37-45.

15. Akyildiz, I. F. ve Vuran, M. C., Wireless Sensor Networks, 4, 520, John Wiley & Sons, 2010.

16. Lee, S. H., Lee, S., Song, H. ve Lee, H. S., Wireless Sensor Network Design for Tactical Military Applications: Remote Large-Scale Environments. In MILCOM 2009 IEEE Military Communications Conference, October 2009, Bildiri Kitabı, 1-7.

17. Ramesh, M. V., Real-Time Wireless Sensor Network for Landslide Detection. In Sensor Technologies and Applications, SENSORCOMM'09. Third International Conference on, IEEE, (2009) 405-409.

18. Alemdar, H. ve Ersoy, C., Wireless Sensor Networks for Healthcare: A Survey. Computer Networks, 54,15 (2010) 2688-2710.

19. Nallusamy, R. ve Duraiswamy, K., Solar Powered Wireless Sensor Networks for Environmental Applications with Energy Efficient Routing Concepts: A review, Information technology journal, 10, 1 (2011) 1-10.

20. Ramesh, M. V., Design, Development, and Deployment pof a Wireless Sensor Network for Detection of Landslides, Ad Hoc Networks, 13 (2014) 2-18.

21. Kırbaş İ., Online Kablosuz İnkübatör İzlemeve Kontrol Sistemi Tasarımı ve Uygulaması, DoktoraTezi, S.A.Ü., Fen BilimleriEnstitüsü, Sakarya, 2013.

22. www.cdc.gov/nchs/Default.htm. National Center for Health Statistics. Accessed in July 2008.

23. Schwiebert, L., Gupta, S. K. ve Weinmann, J., Research Challenges in Wireless Networks of Biomedical Sensors, In Proceedings of the 7th annual international conference on Mobile computing and networking, July 2001, New York, ACM, 151-165.

24. Zhao, G., Wireless Sensor Networks for Industrial Process Monitoring and Control: A Survey, Network Protocols and Algorithms, 3,1 (2011) 46-63.

25. Stojmenovic, I. ed., Handbook of sensor networks: Algorithms and Architectures, 49, John Wiley & Sons, Ottawa, 2005.

26. Faludi, R., Building Wireless Sensor Networks: with Zigbee, Xbee, Arduino, and Processing, O'Reilly Media, Inc., Farnham, 2010.

27. Wang, X. S., Zhan, Y. Z. ve Wang, L. M., Stcp: Secure Topology Control Protocol for Wireless Sensor Networks Based on Hexagonal Mesh, In 2008 4th International

Conference on Wireless Communications, Networking and Mobile Computing, October 2008, 1-4.

28. Cao, L., Jiang, W., & Zhang, Z., Automatic Meter Reading System Based on Wireless Mesh Networks and SOPC Technology. In Intelligent Networks and Intelligent Systems, ICINIS'09. Second International Conference, November 2009, 142-145.

29. Ökdem, S. ve Derviş, K., Kablosuz Algılayıcı Ağlarında Yönlendirme Teknikleri, XI. Akademik Bilişim Konferansı Bildirileri, Şubat 2007, Kütahya, Bildiriler kitabı, 435-441.

30. Al-Karaki, J. N. ve Kamal, A. E., Routing Techniques in Wireless Sensor Networks: A Survey, IEEE wireless communications, 11,6 (2004) 6-28.

31. Heinzelman, W. R., Kulik, J. ve Balakrishnan, H., Adaptive Protocols for Information Dissemination in Wireless Sensor Networks, in Proceedings of The 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, August 1999, New York, Bildiriler Kitabı, 174-185.

32. Kulik, J., Heinzelman, W. R. ve Balakrishnan, H., Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks. Wireless networks, 8,2/3 (2003), 169-185.

33. Intanagonwiwat, C., Govindan, R. ve Estrin, D., Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In Proceedings of the 6th annual International Conference on Mobile Computing and Networking, August 2000, New York, Bildiriler Kitabı, 56-67.

34. Braginsky, D. ve Estrin, D., Rumor Routing Algorthim for Sensor Networks. in Proceedings of The 1st ACM İnternational Workshop on Wireless Sensor Networks and Applications, September 2002, New York, Bildiriler Kitabı, 22-31.

35. Akkaya, K. ve Younis, M., A Survey on Routing Protocols for Wireless Sensor Networks, Ad Hoc Networks, 3,3 (2005) 325-349.

36. Heinzelman, W. R., Chandrakasan, A. ve Balakrishnan, H., Energy-efficient Communication Protocol for Wireless Microsensor Networks. In System Sciences, Proceedings of the 33rd annual Hawaii International Conference on, Janury 2000, Bildiriler Kitabı, 1-10.

37. Verma, K. ve Gupta, M., A Comparitive Study on Location Based Multicast Routing Protocols of WSN: HGMR, HRPM, GMR. Global Journal of Computer Science and Technology, 15,8 (2016) 27-33.

38. Savvides, A., Han, C. C. ve Strivastava, M. B., Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In Proceedings Of The 7th Annual International Conference on Mobile Computing and Networking, Julay 2001, New York, Bildiri Kitabı, 166-179.

39. Xu, Y., Heidemann, J. Ve Estrin, D., Geography-Informed Energy Conservation for Ad Hoc Routing, In Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, Julay 2001, New York, Bildiri Kitabı, 70-84.

40. Yılmaz, M., Duty Cycle Control in Wireless Sensor Networks, Yüksek Lisans Tezi , O.D.T.U., Fen Bilimleri Ensitüsü, Ankara, 2007.

41. Akkaya, K. ve Younis, M., A Survey on Routing Protocols for Wireless Sensor Networks, Ad hoc networks, 3,3 (2005) 325-349.

42. Boyacı, A., Ata, F. ve Balık, H. H., Telsiz Duyarga Ağlarda Kullanılan Yönlendirme Tekniklerinin Enerji Verimliliği Açısından Karşılaştırması, Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, 2,1 (2013) 68-87.

43. Chraibi, Youssef., Localization in Wireless Sensor Networks, Master's Degree Project , K.T.H., School of Electrical Engineering, Sweden, 2005. Stockholm, 2005

44. Pal, A., Localization Algorithms İn Wireless Sensor Networks: Current Approaches and future Challenges, Network protocols and algorithms, 2,1 (2010) 45-73.

45. Whitehouse, K. ve Culler D., Macro-Calibration in Sensor/Actuator Networks, Mobile Networks and Applications, 8, 4 (2003) 63-472.

46. Bekçibaşı, U. ve Tenruh, M., Kablosuz Algılayıcı Ağlarda Konum Saptama Teknikleri ve Mesafe Bağımlı Tekniklerde Dördüncü Çapa Yaklaşımı, XIV. Akademik Bilişim Konferansı, Şubat 2012, Uşak, Akademik Bilişim, 139-146.

47. Murty, K. G., Optimization Models For Decision Making: Volume, University of Michigan, Ann Arbor, 2003.

48. Karaboğa, D., Yapay Zeka Optimizasyon Algoritmalari, 3. Basım, Nobel Akademik Yayıncılık Eğitim Danışmanlık TİC. LTD. ŞTİ., Ankara, 2014.

49. Atashpaz-Gargari, E. ve Lucas, C., Imperialist Competitive Algorithm: an Algorithm for Optimization İnspired by İmperialistic Competition, 2007 IEEE Congress on Evolutionary Computation, September 2007, Singapore, Bildiri Kitabı, 4661-4667.

50. Razzaghpour, M. ve Rusu, A., Analog Circuit Optimization via a Modified Imperialist Competitive Algorithm, In 2011 IEEE International Symposium of Circuits and Systems, May 2012, Seoul, Bildiri kitabı, 2273-2276.

51. Khademolghorani, F., An Effective Algorithm For Mining Association Rules Based on İmperialist Competitive Algorithm, Sixth International Conference on Digital Information Management, September 2011, Melbourne, Bildiri kitabı, 6-11.

52. Tamimi, A., Sadjadian, H. ve Omranpour, H., Mobile Robot Global Localization using İmperialist Competitive Algorithm, In 2010 3rd International Conference on

Advanced Computer Theory and Engineering, August 2010, Chengdu, Bildiri Kitabı, 524-529.

53. Sayadnavard, M. H., Haghighat, A. T. ve Abdechiri, M., Wireless Sensor Network Localization using İmperialist Competitive Algorithm, In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, Julay 2010, Chengdu, Bildiri Kitabı, 818-822.

54. Movahed, M. A. ve Yazdani, A. M., Application of İmperialist Competitive Algorithm in Online PI Controller. In 2011 Second International Conference on Intelligent Systems, Modelling and Simulation, January 2011, Phnom Penh, Bildiri Kitabı, 83-87.

55. Ghalehpardaz, S. L. ve Shafiee, M., Speed Control of DC Motor Using Imperialist Competitive Algorithm Based on PI-Like FLC, In 2011 Third International Conference on Computational Intelligence, Modelling & Simulation, September 2011, Langkawi, Bildiri Kitabı, 28-33.

56. Abdechiri, M., Faez, K. ve Bahrami, H., Adaptive imperialist competitive algorithm, in Cognitive informatics (ICCI), 2010 9th IEEE İnternational Conference, Julay 2010, Beijing, Bildiri Kitabı, 940-945.

57. Abdechiri, M., Faez, K. ve Bahrami, H., Neural network learning Based on chaotic imperialist competitive algorithm, In Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on, May 2010, 1-5.

58. Eberhart, R. C. ve Kennedy, J., A New Optimizer Using Particle Swarm Theory, In Proceedings Of The Sixth İnternational Symposium on Micro Machine and Human Science, October 1995, Nagoya, Bildiri Kitabı, 39-43.

59. Wilke D. N., Analysis Of The Particle Swarm Optimization Algorithm, Master Thesis, UP., Mechanical and Aeronautical Engineering, Pretoria, 2005.

60. Karaboğa D., Yapay Zeka Optimizasyon Algoritmaları, Atlas Yayın Dağıtım, İstanbul, 2004.

61. Keskintürk, T., Diferansiyel Gelişim Algoritması. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 5,9 (2006) 85-99.

62. Yaşar, C., Temurtaş, H. ve Özyön, S., Diferansiyel Gelişim Algoritmasının Termik Birimlerden Oluşan Çevresel Ekonomik Güç Dağıtım Problemlerine Uygulanması, Elektrik-Elektronik ve Bilgisayar Mühendisliği Sempozyumu, Mayıs 2010, Bursa, Bildiri Kitabı, 108-112.

8. EKLER

Ek 1. Probleme Ait Ana Program

```
function sol=AmirSolution(xhat,model)


    xii=diag(xhat)';

    if any(xii>=0.5)

        [~, so]=sort(xii,'descend');

        nCh=0;
        for i=so
            if xii(i)<0.5 | nCh>=P
                break;
            end

            xii(i)=1;

            nCh=nCh+1;
        end

        xii(xii<1)=0;

    else

        [~, imax]=max(xii);

        xii(:)=0;
        xii(imax)=1;

    end

    Chs=find(xii==1);

    x=xhat;
    for i=1:N
        if xii(i)==0
            x(i,:)=0;
        else
            x(:,i)=0;
            x(i,i)=1;
        end
    end
```

```
    ch=zeros(1,N);
    UsedCap=zeros(1,N);
    for i=1:N

        XI=x(:,i);
        XI(xii==0)=-inf;

        [~, ch(i)]=max(XI);

        x(:,i)=0;
        x(ch(i),i)=1;


        UsedCap(ch(i))=UsedCap(ch(i))+sum(r(i,:));
    end

    CapV=max(UsedCap./G-1,0);
    MeanCapV=mean(CapV);

    oc=zeros(N,N);
    for i=1:N
        for j=1:N
            if i==j
                oc(i,j)=0;
            else
                k=ch(i);
                l=ch(j);
                oc(i,j)=c(i,k)+alpha*c(k,l)+c(l,j);
            end
        end
    end

    ocr=oc.*r;
    SumOCR=sum(ocr(:));

    xiif=xii.*f;
    SumXF=sum(xiif);

    TotalCost=SumOCR+SumXF;


End



function model=CreateRandomModel(N)

    Pmin=ceil(0.15*N);
    Pmax=ceil(0.30*N);
```

```
    P=randi([Pmin Pmax]);

    Xmin=0;
    Xmax=100;
    X=randi([Xmin Xmax],1,N);

    Ymin=0;
    Ymax=100;
    Y=randi([Ymin Ymax],1,N);

    d=zeros(N,N);
    for i=1:N-1
        for j=i+1:N

            d(i,j)=sqrt((X(i)-X(j))^2+(Y(i)-Y(j))^2);

            d(j,i)=d(i,j);

        end
    end

    CostPerDistanceUnit=10;

    c=round(CostPerDistanceUnit*d);

    alpha=0.7;

    rmin=10;
    rmax=50;
    r=randi([rmin rmax],N,N);
    r=r-diag(diag(r));

    TotalR=sum(r(:));
    MeanCapacity=TotalR/P;
    MinCapacity=round(MeanCapacity);
    MaxCapacity=round(2*MeanCapacity);
    G=randi([MinCapacity MaxCapacity],1,N);

    rc=r.*c;
    SumRC=sum(rc(:));

    fmean=SumRC/P;
    fmin=round(0.8*fmean);
    fmax=round(1.2*fmean);
    f=randi([fmin fmax],1,N);


end
```

Ek 2. Problemin Sonuçlarının Grafik Çıkışını Göstermek İçin Program

```
function PlotSolution(sol,model)

    N=model.N;
    X=model.X;
    Y=model.Y;

    x=sol.x;
    ch=sol.ch;

    xii=diag(x)';

    Chs=find(xii==1);
    Clients=find(xii==0);

    for i=1:N
        plot([X(i)     X(ch(i))],[Y(i)     Y(ch(i))],'k--
','LineWidth',2);
        hold on;
   end

    plot(X(Chs),Y(Chs),'ks',...
        'MarkerFaceColor','yellow',...
        'MarkerSize',16);

    plot(X(Clients),Y(Clients),'ko',...
        'MarkerFaceColor','Cyan',...
        'MarkerSize',10);

    hold off;
    axis equal;

end
```

ÖZGEÇMİŞ

Amir NASER, 1985 yılında Khoy'da doğdu. Liseyi Rejayi Lisesi'nde okudu. 2010 yılında Khoy Üniversitesi Elektronik Teknisiyenlilği ve 2012 yılında Tebriz Üniversitesi Elektronik Teknolojisi Mühendisliği Bölümü'nü bitirdikten sonra, 2012 yılında Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim dalında yüksek lisans yapmaya başladı. Anadili Azerbaycan Türkçesi olmakla beraber, Türkçe, İngilizce ve Farsça dillerini çok iyi derecede bilmektedir.