TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

COĞRAFİ RADAR DAĞITIM OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Furkan ŞAVŞATLI

Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Prof. Dr. Osman ABUL

AĞUSTOS 2021

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Furkan ŞAVŞATLI

İMZA

ÖZET

Yüksek Lisans Tezi

COĞRAFİ RADAR DAĞITIM OPTİMİZASYONU

Furkan ŞAVŞATLI

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Prof.Dr. Osman ABUL

Tarih: AĞUSTOS 2021

Teknolojide yaşanan gelişmelerle birlikte, mobil hastaneler, internet servis sağlayıcıları, hava durumu ölçme balonları, ordunun veya donanmanın kullandığı radarlar gibi hizmet sunan sistemler de ortaya çıkmıştır. Coğrafya büyüdükçe sistemlerin sayısı arttıkça bu servislerin bir bütün içinde hareket edip optimum şekilde dağıtılması gerekir. Ordunun veya donanmanın da büyük coğrafyalara sistemlerini optimum bir şekilde dağıtması hem barış hem de savaş zamanında önemli bir konudur. Bunun temel gerekçesi kullanılan sistemlerin çeşitli sebeplerden (maliyet, ambargo, teknoloji) dolayı kısıtlı olmasıdır. Bu tezde özellikle askerî hava savunma sistemlerinin, kritik bölgelere optimum şekilde dağıtılması problemi adreslenmiştir. Öncelikle üç farklı problem tanımı verilmiştir. Problem 1'de tek bir hava savunma sisteminin optimum şekilde konumlandırılması amaçlanırken, problem 2'de birden fazla hava savunma sisteminin optimum şekilde dağıtılması amaçlanmıştır. İlk iki problem için dörder tane çözüm algoritması öneri olarak sunulmuştur. Problem 3'te tanımlanan varlıkları minimum maliyetle kapsayacak hava savunma sistemleri aranmıştır. Farklı savunma doktrinleri için karşılaştırmalar yapılmıştır. Çözümlerin test edilebilmesi ve görselleştirilmesi için yazılım uygulaması geliştirilmiştir. Deneylerde rastgele ve stratejik bölgelerde varlıklar tanımlanmıştır. Geliştirilen algoritmalar, tanımlanan varlıkları ve hava savunma sistemlerini kullanılarak uygulanmıştır. Sonuçlar uygulama ile görselleştirilerek karşılaştırılmıştır.

Anahtar Kelimeler: Radar kapsama alanı, Coğrafi servis dağıtımı, Komuta kontrol, Optimizasyon.

ABSTRACT

Master of Science

GEOSPATIAL RADAR DEPLOYMENT OPTIMIZATION

Furkan ŞAVŞATLI

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Prof.Dr. Osman ABUL

Date: AUGUST 2021

Along with the developments in technology; Service provider systems such as internet service providers, mobile hospitals, weather measurement balloons, radars used by the military or navy have also emerged. As the target geography grows and the number of systems increases, these services should act as a whole and be optimally distributed. The optimum distribution of the army or navy systems to large geographies is an important issue both in times of peace and war. The main justification of this is that the systems are limited for various reasons (cost, embargo, technology). In this thesis, particularly, the problem of optimal distribution of military air defense systems to critical regions has been addressed. Firstly, three different problem definitions are given. While it is aimed to deploy a single air defense system in the optimum way in the problem 1, it is aimed to distribute multiple air defense systems in the problem 2. For each problem, four solution algorithms are proposed as suggestions. Air defense systems were sought to cover the assets defined in the problem 3 with minimum cost. Comparisons were made for different defense doctrines. A software application has been developed to test and visualize solutions. In the experiments, data sets in random and strategic regions were defined. The developed algorithms have been applied using defined assets and air defense systems. The results were compared by visualizing via the application.

Keywords: Radar coverage area, Geospatial service deployment, Command and control, Optimization.

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Prof.Dr. Osman ABUL'a, kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bölümü öğretim üyelerine ve destekleriyle her zaman yanımda olan aileme ve arkadaşlarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET v
ABSTRACT viii
TEŞEKKÜR x
İÇİNDEKİLER xii
ŞEKİL LİSTESİ xvi
ÇİZELGE LİSTESİ xviii
KISALTMALAR xx
1. GİRİŞ 2
1.1 Durumsal Farkındalığın Önemi 3
1.2 Teknik Taktik Strateji 4
1.3 Yazılım Uygulamalarında Kullanıcı Deneyimi 5
1.4 Kıymetli Bölge Savunması 7
2. KÜMELEME ALGORİTMALARI 10
2.1 Bölümleme (Partitioning) Algoritmaları 10
2.1.1 K-Means Algoritması 11
Avantajları 11
Dezavantajları 12
2.1.2 Pam Algoritması 12
2.1.3 Mini Batch K-Means 13
Avantajları 13
Dezavantajları 13
2.1.4 Clara Algoritması 13
2.1.5 Clarans Algoritması 13
2.2 Hiyerarşik Kümeleme Algoritmaları (Dendrogram ile Kümeleme) 14
Avantajları 14
Dezavantajları 15
2.2.1 SLINK metodu (En yakın komşuluk metodu) 15
Avantajları 15
Dezavantajları 15
2.2.2 CLINK metodu (En uzak komşuluk metodu) 16
Avantajları 16
Dezavantajları 16
2.2.3 Average linkage metodu (Ortalama bağlantı yöntemi) 17
2.2.4 Centroid metodu (Küresel ortalama yöntemi) 17

2.2.5 Median metodu 18
2.2.6 Ward metodu 18
2.2.7 Cure Algoritması 19
Avantajları 19
Dezavantajları 20
2.2.8 CHAMELEON Algoritması 20
2.3 Yoğunluk Tabanlı (Density Based) Kümeleme Algoritmaları 21
2.3.1 Mean Shift(Ortalama Kaydırma) Yöntemi 21
2.3.2 Dbscan Algoritması 22
Avantajlar 23
Dezavantaj 23
3. COĞRAFİ RADAR DAĞITIM OPTİMİZASYONU 24
3.1 Radar Sistemleri 27
3.2 Coğrafi Servis Dağıtımı 28
3.2.1 Tanımlar 28
Tanım 1 (Operasyon bölgesi): 28
Tanım 2 (Coğrafi-referanslı varlık): 29
Tanım 3 (Coğrafi-referanslı servis): 29
Tanım 4 (Kapsanan varlıklar): 29
Tanım 5 (Servis-varlık mesafesi): 30
3.2.2 Problem Tanımları 30
3.2.2.1 Problem 1 - Tek Servis ile Maksimum Değerli Varlık Kapsama . 30
Problem 1 (Tek servis ile maksimum değerli varlık kapsama problemi): 30
3.2.2.2 Problem 1c 30
Problem 1c: 30
3.2.2.3 Problem 2 - Çoklu Servis ile Maksimum Degerli Varlık Kapsama 31
Problem 2 (Maksimum degerli varlık kapsama problemi): 31
3.2.2.4 Problem 2c 31
Problem 2c: 32
Tanım 6 (Kalite skoru): 32
3.2.2.5 Problem 3 - Minimum Maliyetle Tüm Varlıkları Kapsama 32
Tanım 7 (Servis envanteri): 32
Tanım 8 (Servis envanteri maliyeti): 32
Problem 3 (Minimum maliyetle tüm varlıkları kapsama problemi): . 32
3.3 Coğrafi Servis Dağıtım Problemi Çözümü 33
3.3.1 Problem 1 Çözümü 33
3.3.1.1 Optimum Çözüm 33
3.3.1.2 Ağırlık Merkezi ile Kümeleme 37
3.3.1.3 Maksimum Kesişim Bölgesi 38
Teorem 1 (Maksimum ağırlıklı düğüm): 39

3.3.2 Problem 1c Çözümü 40
3.3.2.1 Servis Konumları Aday Listesi 40
3.3.3 Problem 2 Çözümü 40
3.3.3.1 Optimuma Yakın Çözüm 40
3.3.3.2 Ağırlık Merkezi ile Kümeleme 42
3.3.3.3 Maksimum Kesişim Bölgesi 44
3.3.4 Problem 2c Çözümü 44
3.3.4.1 Servis Konumları Aday Listesi 44
3.3.5 Problem 3 Çözümü 45
Tanım 9 (Uygulanabilir servis envanteri): 45
3.3.5.1 Ağırlık merkezi ile kümeleme 45
3.3.5.2 Ağırlık merkezi ile kümeleme - Rekürsif Optimizasyon 46
3.3.5.3 En küçük daire ile hiyerarşik kümeleme 48
En küçük daire problemi 48
4. DENEYSEL DEĞERLENDİRME 50
4.1 Veri Kümeleri 50
4.2 Problem 1 Deneyi 53
4.2.1 Deney 1 Sonucu 53
4.3 Problem 2 Deneyi 62
4.3.1 Deney 2 Sonucu 63
4.4 Problem 3 Deneyi 66
4.4.1 Deney 3 Sonucu 67
5. SONUÇ VE ÖNERİLER 74

Kaynaklar **76**
ÖZGEÇMİŞ 80

ŞEKİL LİSTESİ

Şekil 1.1: Kullanıcıya aynı amaç için sunulan iki farklı ekran tasarımı. . . . 6
Şekil 1.2: Türkiye hidroelektrik santralleri haritası [28]. 7
Şekil 1.3: Türkiye petrol rafinerileri haritası [28]. 8
Şekil 2.1: Dendrogram yapısı 14
Şekil 2.2: 2 küme arasında mesafeye bakılırken kümelerin birbirlerine en yakın olan *a* ve *b* noktaları ele alınır. Hesaplanan uzaklık diğer kümelerle karşılaştırılır. 16
Şekil 2.3: 2 küme arasında birbirlerine en uzak nokta olan *a* ve *b* noktaları seçilir. Aradaki uzaklık diğer kümelerle karşılaştırılır. En küçük uzaklığa sahip olan 2 küme birleştirilir. 17
Şekil 2.4: Yeni ağırlık merkezi sağdaki kümenin elaman sayısı daha çok olduğu için sağ taraftaki kümeye yakın olacaktır. 18
Şekil 2.5: figure 3 18
Şekil 2.6: Yerleştirilen daire kendi içindeki noktaların ağırlık merkezine göre sağa doğru hareket eder. 22
Şekil 2.7: Birinci şekilde *b*, *c* ve *d* noktaları *a* kümesi ile birleştirilirken, alttaki şekilde *d* noktasının *eps* bölgesinde sadece 2 komşusu olduğu için *e* noktası diğer noktalarla birleştirilmez. 23
Şekil 3.1: Radar ufku ve rakım ilişkisi 4/3 dünya modeli 24
Şekil 3.2: Rakıma göre hava radarlarının kapsadığı alan (a) 3 km (b) 5 km . 25
Şekil 3.3: A (karınca algoritması), GA (Genetik Algoritma) ve IGA-BAC 2 algoritmanın kombinasyonu 26
Şekil 3.4: 27
Şekil 3.5: Servis için optimum pozisyonlardan birini gösteren durum. 34
Şekil 3.6: Servisin çemberi üzerinde 2 varlık bulunacak şekilde kaydırılması. 34
Şekil 3.7: İki nokta ile oluşturulabilecek iki çember bulunmaktadır. 35
Şekil 4.1: `DSSPARSE`: 1000 tane coğrafi referanslı varlık içeren veri kümesi. 50
Şekil 4.2: `DSNORMAL`: 2000 tane coğrafi referanslı varlık içeren veri kümesi. 51
Şekil 4.3: `DSDENSE`: 3000 tane coğrafi referanslı varlık içeren veri kümesi. . 51
Şekil 4.4: `STRA`: 40 tane stratejik coğrafi referanslı varlık içeren veri kümesi. 52
Şekil 4.5: "Candidate List of Service Locations" algoritmasında kullanılacak aday noktalar. 53
Şekil 4.6: Yoğunluk `DSSPARSE` olan veri seti seçildiğinde çıkan sonuç. . . . 54
Şekil 4.7: Yoğunluk `DSNORMAL` olan veri seti seçildiğinde çıkan sonuç. . . . 54
Şekil 4.8: Yoğunluk `DSDENSE` olan veri seti seçildiğinde çıkan sonuç. 55
Şekil 4.9: The "Maximum Corner Weight Clique" yöntemi diğer üç algoritmaya göre yavaş çalışır. 56

Şekil 4.10: The "Candidate List of Service Locations" Yöntemi en hızlı çalışan algoritmadır. 56
Şekil 4.11: Coğrafi-referanslı Hisar-O servisinin "Optimal Solution (Algoritma 3.1)" yöntemine göre son konumu. Kalite skoru 1700'dür. . 57
Şekil 4.12: Coğrafi-referanslı Hisar-O servisinin "Optimal Solution (Algoritma 3.1)" yöntemine göre son konumu. Kalite skoru 4900'dur. . 58
Şekil 4.13: Coğrafi-referanslı Hisar-O servisinin "Center of Mass with Clustering (Algoritma 3.2)" yöntemine göre son konumu. Kalite skoru 1700'dür. 58
Şekil 4.14: Coğrafi-referanslı S-400 servisinin "Center of Mass with Clustering (Algoritma 3.2)" yöntemine göre son konumu. Kalite skoru 4800'dür. 59
Şekil 4.15: Coğrafi-referanslı Hisar-O servisinin "Maximum Vertex Weight Clique (Algoritma 3.3)" yöntemine göre son konumu. Kalite skoru 1700'dür. 59
Şekil 4.16: Coğrafi-referanslı Hisar-O servisinin "Candidate List of Service Locations (Algoritma 3.4)" yöntemine göre son konumu. Kalite skoru 1300'dür. 60
Şekil 4.17: Coğrafi-referanslı S-400 servisinin "Candidate List of Service Locations (Algoritma 3.4)" yöntemine göre son konumu. Kalite skoru 4300'dür. 60
Şekil 4.18: Rastgele olarak yerleştirilen S400, Hisar-O ve Hisar-U radarları. 62
Şekil 4.19: Çoklu servis için deney sonuçları. 63
Şekil 4.20: Algoritma 3.5'in (önce büyük servis sıralı) kalite skoru 9600'dür. 64
Şekil 4.21: Algoritma 3.5'in (önce küçük servis sıralı) kalite skoru 8800'dir. 64
Şekil 4.22: Algoritma 3.6'nın kalite skoru 9400'dür. 65
Şekil 4.23: Algoritma 3.7'nin kalite skoru 9500'dür. 65
Şekil 4.24: Algoritma 3.8'in kalite skoru 6300'dür. 65
Şekil 4.25: Algoritma 3.9'un envanter maliyeti 1740 olmuştur. 67
Şekil 4.26: Algoritma 3.10'un envanter maliyeti 1460 olmuştur. 68
Şekil 4.27: Algoritma 3.11'in envanter maliyeti 1560 olmuştur. 68
Şekil 4.28: Envanter maliyeti 4520 olmuştur. Toplam 17 savunma sistemi kullanılmıştır (14 S-400, 2 Hisar-U, 1 Hisar-O). 69
Şekil 4.29: Envanter maliyeti 4120 olmuştur. Toplam 37 savunma sistemi kullanılmıştır (6 S-400, 27 Hisar-U, 4 Hisar-O). 70
Şekil 4.30: Envanter maliyeti 3600 olmuştur. Toplam 9 savunma sistemi kullanılmıştır (9 S-500). 71
Şekil 4.31: Envanter maliyeti 1120 olmuştur. Toplam 19 savunma sistemi kullanılmıştır (2 Hisar-U, 5 Hisar-O, 12 Hisar-A). 72

ÇİZELGE LİSTESİ

Çizelge 4.1: VARLIK TIPLERI VE VARLIK DEĞERLERI. 51
Çizelge 4.2: STRATEJIK VARLIK TIPLERI VE VARLIK DEĞERLERI. 52
Çizelge 4.3: VERI SETLERI VE IÇERDIKLERI VARLIK SAYILARI. 52
Çizelge 4.4: RADARLAR TIPLERI VE KAPSAMA YARIÇAPLARI. 57
Çizelge 4.5: ALGORITMA 3.1, ALGORITMA 3.2, ALGORITMA 3.3 VE ALGORITMA 3.4 SONUÇLARINA GÖRE KALITE SKORLARI. . 61
Çizelge 4.6: RADARLAR TIPLERI VE KAPSAMA YARIÇAPLARI. 62
Çizelge 4.7: ALGORITMA 3.5, 3.6, 3.7 VE 3.8 SONUÇLARINA GÖRE KALITE PUANLARI. 66
Çizelge 4.8: RADARLAR, KAPSAMA YARIÇAPLARI VE MALIYETLERI. . . 67
Çizelge 4.9: MALIYET VE SÜRE KARŞILAŞTIRMALARI. 69
Çizelge 4.10: RADARLAR, KAPSAMA YARIÇAPLAR VE MALIYETLERI. . 71

KISALTMALAR

OR : Operasyon bölgesi(Operation region)
A : Coğrafi-referanslı varlık(Geo-referenced asset)
S : Coğrafi-referanslı servis (Geo-referenced service)
I : Envanter (Inventory)
ZPT : Zırhlı personel taşıyıcı
ZMA : Zırhlı muharebe aracı
İHA : İnsansız hava aracı

1. GİRİŞ

Muharebelerin farklı kısımları vardır. Bu sebeple muharebelerin sonucuna etki eden tek faktörün silah gücü olduğunu düşünmemek gerekir. Ordunun silahlı unsurları dışında, silahlı olmayan ama savaşın sonucuna doğrudan etki eden birimler de önemlidir. İyi teknolojiyle geliştirilmiş silahlara sahip bir ordunun ayrıca lojistik, haberleşme, istihbarat hatta günümüzde elektronik harp gibi birimlere de ihtiyacı vardır. Bütün bu sistemler ayrı ayrı çok iyi olsa da bir arada organize bir şekilde kullanılamıyorlarsa ciddi sıkıntılar yaşanabilir. Komuta kontrol bu yüzden önemlidir. Komuta kontrol yukarıda sayılan bütün birimleri kapsar ve bu birimlerin eş zamanlı, kordineli bir şekilde planlanmasını, yönlendirilmesini ve kontrol edilmesini sağlar.

Geçmiş yüzyıllardaki muharebeler, genellikle küçük bir bölgede olan cephe savaşlarıydı. Komutan yanındaki astlarıyla birlikte tüm orduyu kolay bir şekilde kontrol edebilirdi. Savaş sırasında komutanın emirlerini uygulamak kolaydı; çünkü kendi sesiyle bile ordusunun büyük bir kısmına seslenebiliyordu. Günümüzde ise muharebeler çok daha geniş cephelerde uzun yıllar boyunca sürmektedir. Bu sebeple muharebeden sorumlu komutanın işi zorlaşmaktadır. Komutan, donanma gemilerinin ve uçaklarının nerede ve nasıl şartlara sahip olduğunu ve o sırada düşmanın nasıl bir taktik içinde olduğunu bilmezse; hatta iki gün sonrası için uzaktaki birliklerinin yiyeceğini ve yakıtını hesaplamazsa, çok daha iyi birimlere sahip olsa bile savaş kaybedilebilir. Komuta kontrol sistemleri burada devreye girmektedir. Tüm cephelerden ve sensörlerden gelen veriler füzyon edilip anlamlandırıldıktan sonra komutana sunulur. Bu sayede komutan en hızlı şekilde kararlar alıp uygulatabilir. Komuta kontrol sistemleri bu sebeple çok önemli bir hale gelmiştir. İyi bir komuta kontrol sistemi, dağıtık bir mimaride elektronik saldırılara dirençli ve hızlı bir şekilde çalışıp, veriyi kullanıcının hızlıca anlayacağı şekilde sunmalıdır.

Komuta kontrol sistemlerinin konuları çeşitlilik gösterir. Donanmayı yönetmek ayrı bir uygulamayken, bir uçağın doğru pozisyonda gemiye iniş yapması ayrı bir uygulama olur. Aynı şekilde atış kontrol de komuta kontrolün bir parçasıdır. Gemiye gelmekte olan füzeye hangi şekilde ve sırayla angajmana girileceği atış kontrol sistemleriyle yapılır. Bu sırada karargâhın durumdan haberi olması gerekir. Komuta kontrol, büyük orduların birlikte çalışması için olmazsa olmazdır.

Teknoloji geliştikçe, savunma silahlarındaki çeşitlilik de artmaktadır. Kuvvetlerin ihtiyaçları farklı olduğu gibi aynı kuvvetteki sistemlerin ihtiyaçları da farklılık gösterir. Örneğin bir tank ve zırhlı muharebe aracı (zma) karşılaştırıldığında ihtiyaçları çok farklı olacaktır. Bu sebeple yapılan komuta kontrol sistemleri de farklılaşmaktadır. Kara kuvvetleri için sağlanan çözümlerden bir tanesi de HAVELSAN

tarafından zırhlı araçlarda kullanılmak üzere geliştirilmiş komuta kontrol sistemidir. Bu sistem ile savaş ve barış zamanında zırhlı araçların komuta kontrolü sağlanabilmektedir. Sistem ile karar vericiler için durumsal farkındalık artacak ve birlikler için önemli kazanımlar elde edilecektir.

Sistemlerin çeşitlenmesinden dolayı katmanlı komuta kontrol sistemleri de önemli olacaktır. Örneğin donanmayı yöneten bir karargâh vardır. Karargâh verdiği kararlarla gemileri yönetirken, daha küçük gemilerin yönetimi o bölgede karargâh gibi çalışabilen büyük gemilere bırakılırsa daha verimli bir sonuca ulaşılacaktır. Bu sebeple donanma için komuta kontrol sistemleri yapılırken yazılım uygulamaları katmanlı sistemleri destekleyecek şekilde yapılır.

Komuta kontrol sistemlerinin yakın gelecekte daha kapsayıcı şekilde görülmesi beklenmektedir. Günümüzde komuta kontrol sistemleri farklı kuvvetler için farklı sistemler olarak ayrışmaktadır. Hatta aynı kuvvet farklı sistemleri bile kullanmaktadır. Örneğin kara kuvvetleri tanklar ve zırhlı personel taşıyıcılar (zpt) için farklı komuta kontrol sistemleri kullanabilmektedir. Amerika Savunma Bakanlığı bütün kuvvetleri birbirine bağlayacak bir sistem için çalışmaktadır ve buna Joint All Domain Command and Control (JADC2) ismini vermiştir [22]. JDAC2 tüm kuvvetleri tek çatı altında komuta etmeyi amaçlamaktadır. Bu kadar fazla veriyi yönetmek şu an için zor olsa da gelecekte bu tarz sistemlerin daha da gelişeceği aşikârdır.

Bu tezin çözmeye çalıştığı problem komuta kontrol sistemlerinin bir parçasıdır. Hava savunma sistemlerinin merkezi bir yerden yönlendirilip konumlandırılmalarına çözüm aranacaktır. Kısıtlı sayıdaki hava savunma sistemlerinin optimum şekilde dağıtılması veya ihtiyaç duyulan en az maliyetli sistemlerin bulunması savaş ve barış durumunda kuvvete yarar sağlayacaktır.

1.1 Durumsal Farkındalığın Önemi

Durumsal farkındalık, çevrenin algılanışındaki doğruluk derecesidir[34]. İnsan çevresindekilerle veri alış verişini duyu organlarıyla yapar. İnsanlar her ne kadar aynı şeylere bakıyor veya duyuyor olsalar da nesneleri algılamaları, yani durumsal farkındalıkları farklı olabilir. Örneğin araç sürücüleri için, yola aniden atlayan bir çocuğu fark etme ve tepki verme süreleri farklılık gösterebilir. Bu süre tecrübeye, dikkate, yaşanmışlıklara ve önceliklere göre değişebilir.

İnsanların fiziksel ve zihinsel olarak mükemmel olmaması dolayısıyla askeri yaklaşımda durumsal farkındalık mümkün olduğunca sensörlere bırakılmak istenir. Örneğin bir tankın komutanı üzerine gelebilecek bir füzeyi ne kadar erken tespit

edebilirse durumsal farkındalığı o konuda o kadar fazla olacaktır. Ayrıca yine aynı komutan gözleriyle sürekli gözlem yapamayacağı ve gözlem yapsa bile tepki süresi uzun olacağı için askeri sistemlerde mümkün olan en iyi teknolojiye ihtiyaç duyulmaktadır.

Durumsal farkındalığın en önemli örneklerinden biri de radarlar olagelmiştir. 2. Dünya Savaşı'nda İngiltere kullandığı radarlar ile Alman uçaklarını önceden fark etmiş ve bu durum savaşın sonucuna önemli derecede etki etmiştir. Günümüzde ise radarlar durumsal farkındalığın en önemli bileşeni haline gelmiştir. Durumsal farkındalık komuta kontrol için de çok önemlidir. Bu sebeple iyi bir komuta kontrol sistemi iyi radarlara ihtiyaç duyar.

Tezin amaçları içinde doğrudan durumsal farkındalığı geliştirmek bulunmamaktadır. Bunun yanında hava savunma sistemlerinin pozisyonlarını takip etmek dolaylı yoldan durumsal farkındalığa yarar sağlayacaktır.

1.2 Teknik Taktik Strateji

Teknik, taktik ve strateji harbin sonucunu belirleyen önemli özelliklerdendir [33]. Teknik, bir nevi kullanılan teknolojilerdir. Uçaklarınız düşman uçaklarından üstünse teknik olarak üstünsünüz denilebilir. Taktik, tekniğin nasıl kullanılacağının belirlenmesidir. Örneğin ordu formasyonunun nasıl olacağı ya da hilal taktiğinin yapılması taktiğe örnek olarak verilebilir. Taktik aslında savaş anında ekipmanların nasıl kullanılacağının belirlenmesidir. Daha pratik çözümler içerir ve süre olarak kısa bir zamanda uygulanır. Starateji ise taktiğin üst pencereden yönetilmesidir. Hangi cephede ne yapılacağını, lojistiği, saldırma zamanını ve geri kalan her şeyi içerir. Savaş sırasında yapılan lobi faaliyetleri de strateji için örnektir [33]. Örneğin 2020'de gerçekleşen Azerbeycan-Ermenistan Savaşı'nda Ermeni lobisi sayesinde Azerbeycan ordusunun kullandığı silahlı insansız hava araçlarının (iha) kameralarına Kanada tarafından ambargo uygulanmıştır. Bu sonuç savaşın sonucuna etki etmese de Ermeni lobisinin elde ettiği stratejik bir kazanımdır.

Türk Askeri Kültürü kitabında da söylendiği üzere; "Teknik olarak gücünün yetmediği düşmana taktik perdeden taarruz et, taktik olarak da gücün yetmiyorsa stratejik olarak taaruz et [33]." Bu sebeple teknik ve taktikte geride olan bir ordu stratejik akılla üstünlüğü eline alabilir. Tam tersi ise çok daha zor olacaktır. Sun Tzu'nun da dediği gibi "Stratejideki hatayı taktikle düzeltmek çok zordur [41]." Strateji ve taktik arasındaki farkı anlatan diğer bir güzel sözü de satranç ustası Savielly Tartakower, "Taktik, yapacak bir şey varken ne yapılması gerektiğini bilmek, strateji ise yapacak hiçbir şey yokken ne yapmak gerektiğini bilmektir." söz-

leriyle dile getirmiştir.

Bu tezde teknik ve strateji konularında katkı sunulmaya çalışılacaktır. Geliştirilecek olan teknik bir yazılımdır ama vereceği katkı stratejik boyutlarda olacaktır. Hava savunma sistemlerini optimum şekilde yerleştirmek stratejik kazanım sağlayacaktır.

1.3 Yazılım Uygulamalarında Kullanıcı Deneyimi

Teknoloji ilerledikçe komuta kontrol sistemlerinde toplanan veri de büyümektedir. Bir aracın data üretebilecek birçok sensörü vardır. Üretilen bu dataların anlamlandırılması, füzyon edilmesi komuta kontrol sistemlerinin en zor konularından biridir. Bunun sebebi, her ne kadar fazla data üretilse de karar verici bir insan olduğu için algılayabileceği miktar kısıtlıdır. Bu sebeple toplanan datanın hepsinin gösterilmesi zaman kaybına hatta yanlış karar verilmesine sebep olabilir. Bunun için kullanıcıya gösterilecek veri üzerinde çalışma yapılması gerekir. Çok bilgiyi aynı anda mı göstereceğiz yoksa bazı bilgilerden feragat edip hızlı karar almayı mı sağlayacağız? Bu denge çok önemlidir. Bu konuyla ilgili yapılan bir çalışmada kullanıcı dostu ve fazla veri odaklı iki farklı ekran tasarımında kullanıcı göz hareketleri ve karar verme süreleri incelenmiştir [37]. İki farklı yaklaşım için kullanılan ekran tasarımının çok farklı olduğu şekil 1.1'de açıkça gözükmektedir. Komuta kontrol projelerinde ekran tasarımına gereken önemin verilmemesi operasyon anında zaman kayıplarına neden olabilir. Bu sebeple kullanıcı iyi deneyimlenmeli ve ihtiyaca yönelik tasarım yapılmalıdır.

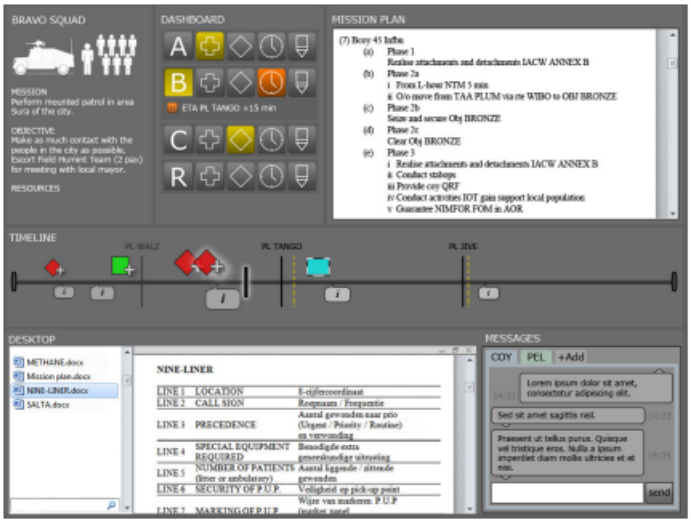


Figure 2. CZ-Man information management screen for the *decision support* variant, showing the Object Status box, Dashboard, Mission Plan, Timeline, Desktop and Message box (shown as right display).

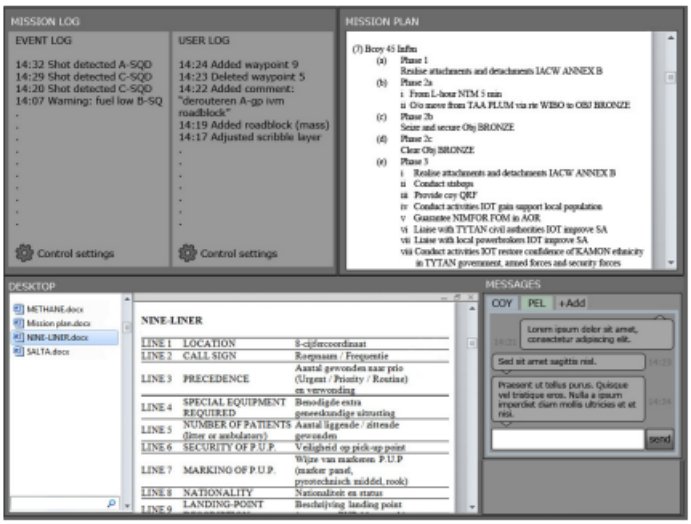


Figure 3. CZ-Man information management screen for the *personalization* variant, showing the Mission Log, Mission Plan, Desktop and Message box (shown as right display).

Şekil 1.1: Kullanıcıya aynı amaç için sunulan iki farklı ekran tasarımı.

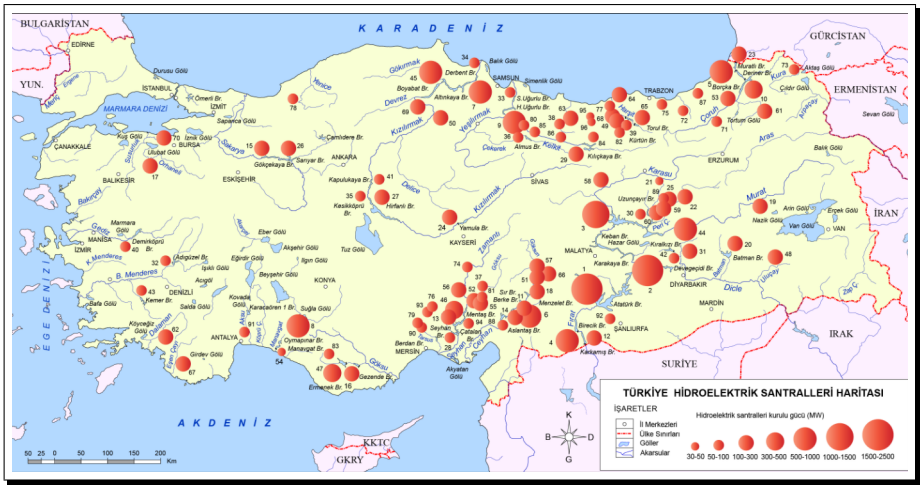
Bunların dışında bir de teknoloji yetersizliği durumu vardır. Sensörlerden toplanan veriler ihtiyaca göre 1-60 sn aralağında yenilenmeye ihtiyaç duyar. Yüzlerce

sistemin farklı sensörlerden tüm verilerinin toplanıp, yine tüm sistemlere iletilmesi fazla memory, cpu ve storage ihtiyacı doğurabilir. Bu ihtiyaçlar projelerin başında göz önüne alınmazsa daha sonra ciddi performans problemlerine sebep olabilir.

Tez için geliştirilen uygulamada kullanıcı arayüzünün basitliğine önem verilmiştir. Çeşitli seneryolar için uygulanması kolay bir uygulama geliştirilmiştir. Uygulamanın kullanılmak istenmesi durumunda kullanacak kişiye öğretilmesi son derece kolaydır.

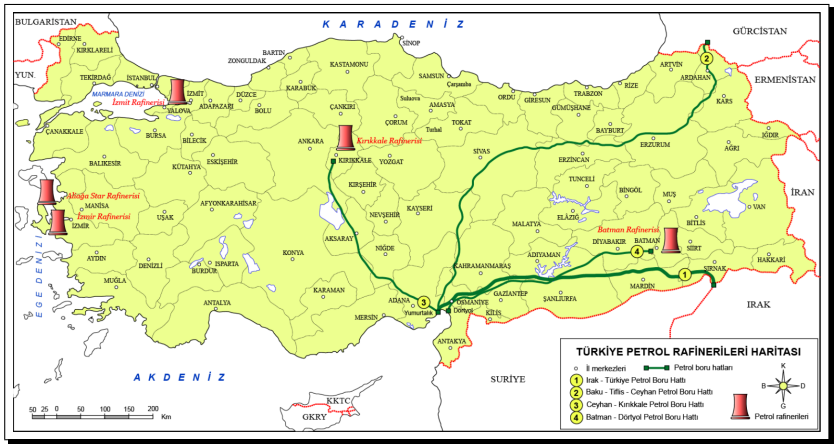
1.4 Kıymetli Bölge Savunması

Ülkeler geliştikçe korunması gereken yapılar da doğru orantılı bir şekilde artmaktadır. Bu sebeple gelişmiş bir ülkenin ekonomisine katkı sağlayan yapıları, sürdülülebilir bir ekonomi için son derece önemlidir. Muharebe cephede kazanılsa bile stratejik öneme sahip enerji santralleri, rafineriler, hava alanları, barajlar ve körpüler gibi ekonomiye doğrudan katkısı olan yapılar zarara uğrarsa ülkenin dinamikleri ciddi manada sarsılır. Türkiye enerji konusunda cari açık veren bir ülkedir [42]. Bu sebeple, enerjiye katkı sağlayan yapılar stratejik olarak değerlidir. Türkiye'de bu yapıların başında elektrik üreten barajlar gelmektedir [2]. Şekil 1.2'de Türkiye'deki elektrik üreten barajlar gösterilmiştir.



Şekil 1.2: Türkiye hidroelektrik santralleri haritası [28].

Türkiye petrol ithal eden bir ülkedir ve petrolü kendi rafinerilerinde işler. Bu sebeple şekil 1.3'te görülen rafinerilerin savunması da stratejik olarak çok önemlidir.



Şekil 1.3: Türkiye petrol rafinerileri haritası [28].

Sinop ve Mersin'e yapılacak nükleer enerji santralleri, İstanbul Boğazı'ndaki köprüler ve havaalanları stratejik olarak korunması gereken diğer önemli yapılardır. Görüldüğü üzere stratejik yapılar ülkenin dört bir yanına dağılmış durumdadır.

Bir komutanın ordusunu kontrol ederken düşünmesi gereken en önemli konulardan biri de ülkedeki önemli noktaların savunmasıdır. Günümüzde cephe savaşları yapılırken savaş sadece cephe hattında olmaz çünkü uzun menzilli balistik ve seyir füzeleri ile cephe arkasına da saldırılar yapılabilir. Örneğin Tomahawk füzesinin menzili 2500 km'ye kadar varabilmektedir. Bu sebeple komutanlar, hava savunma sistemleri ve radarlarının pozisyonlarını düşünürken bu tarz saldırıları hesaba katmalıdır. Örneğin Türkiye'nin gireceği bir savaşta Ankara, cephe hatlarına uzak olacaktır ama birçok önemli merkezi olması sebebiyle düşmanların hedefi haline gelebilecektir. Burada verilmesi gereken karar eldeki kısıtlı savunma ve radar sistemlerinin nasıl bir öncelikle nerelere dağıtılacağıdır. Sınırsız savunma sistemi olmadığı için bazı bölgelere öncelik vermek gerekecektir. Doğru bir yerleştirme yapılmazsa bazı önemli bölgeler açıkta kalabilir veya daha az önemsiz bölgelere fazla öncelik verilebilir. Bunun için yerleştirme stratejisini iyi düşünmek gerekir.

Önemli merkezlerin korunması sadece savaş zamanında düşünülecek bir konu değildir. Örneğin Yunanistan ile yapılacak bir savaş çok kısa sürede engellenebilir ve sonlanabilir. Böyle bir savaşta iki ülke de kritik tesisleri olabildiğince çabuk saf dışı bırakmaya çalışacaktır. Bu sebeple savunma sistemlerimizin konumu her zaman çok önemlidir.

Bu tez ile stratejik olduğu düşünülen noktaların veya bölgelerin mevcut savunma sistemleriyle optimum şekilde nasıl korunacağına cevap aranır. Problem aslında genel bir problemdir. Örneğin internet sağlayıcılarının nasıl dağıtılacağı veya ne-

relere kafe açılacağı da benzer problemler olarak gösterilebilir.

Yapılacak problem tanımında önemli olduğu düşünülen staratejik noktalar veya bölgeler çeşitli hava savunma sistemleri ile optimum şekilde kapsanmaya çalışılacaktır. Sonuç olarak kısıtlı sayıdaki savunma sistemleri savunma doktrinine göre en iyi şekilde dağıtılabilecektir. Bunun dışında tanımlanacak diğer bir problemle ihtiyaca yönelik maliyeti en az olacak şekilde savunma sistemleri araştırılacaktır.

Tanımlara ve çözümlere başlamadan önce bu çalışmaya katkı verebilecek kümeleme algoritmaları incelenecektir.

2. KÜMELEME ALGORİTMALARI

Kümeleme, temel olarak varlıkları, benzerlik ve farklılıklarına göre gruplara ayırmaktır. Ayrışan her gruba küme adı verilir. Kümeleme, literatür açısından herhangi bir veri kümesini, küme içi benzerliğin en üst düzeye çıkarılması ve küme arası benzerliğin en aza indirilmesi için gruplayan bir keşif sürecidir [18] [39].

Kümeleme algoritmaları denetimsiz (unsupervised) öğrenme algoritmalarıdır [14] [15]. Bunun sebebi verilerin herhangi bir çıktısı olmamasıdır. Veriler özelliklerine göre gruplandırılmaya çalışılır. Daha sonra gelecek yeni veri ise oluşan kümelerden kendisine en yakın olana yerleştirilir.

Kümeleme algoritmaları, denetimli (supervised) algoritmalardan farklılık gösterir. Denetimli algoritmalarda eldeki verilerin çıktısı vardır ve sınıfları bellidir. Algoritma mevcut verilere göre eğitilir ve yeni verilerin hangi sınıfa gireceğine karar verilir [10].

Bir çok kümeleme algoritması vardır ancak hiç bir algoritma varlıkları kesin olarak kümelere ayıramaz. Bunun sebebi küme tanımının değişken olması ve kişiden kişiye değişmesidir [13].

Kümeleme algoritmalarında iki nokta veya küme arasındaki uzaklık hesaplanırken çeşitli algoritmalardan yararlanılır. Bunlara örnek olarak Öklid, Mahalanobis ve Minkowski uzaklık algoritmaları verilebilir [26]. Daha sonraki bölümlerde ve tez için tanımlanan problemlerde kullanılacak olan Öklid mesafesi, verilen n boyutlu $P = \{p_1, p_2, \ldots, p_n\}$ ve $Q = \{q_1, q_2, \ldots, q_n\}$ noktaları arasında şu şekilde tanımlanır.

$$Oklid(P,Q) = \|(P-Q)\| = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.1}$$

Kümeleme algoritma yaklaşımlarından bölümleme, hiyerarşik ve yoğunluk tabanlı kümeleme algoritmaları incelenecektir.

2.1 Bölümleme (Partitioning) Algoritmaları

Hedeflenen küme sayısana göre verinin bölümlere ayrıldığı algoritmalardır. Bu algoritmalar, optimal bir bölüm elde edilinceye kadar veri noktalarını, kümeler arasında yinelemeli olarak yeniden konumlandırarak belirli bir kümeleme kriterini

en aza indirene kadar yinelemeli olarak devam eder [5]. Bazı bölümleme algoritmaları sırayla incelenecektir.

2.1.1 K-Means Algoritması

Mac Queen tarafından 1967'de bulunan bölümleme tipinde kümeleme algoritmasıdır [19]. En yaygın kullanılan algoritmalardan birisidir. Verilerin herhangi bir çıktısı olmadığından dolayı denetimsiz öğrenme algoritması olarak tanımlanır. K-Means verilerin birbirlerine olan benzerliklerine göre, verileri parametre olarak aldığı k sayısı kadar kümeye ayırır. Benzerliklerin hesaplamasında kullanılan mesafe fonksiyonu Öklid mesafesidir. Algoritma, seçilen k adet merkez noktayı kümelerin ağırlık merkezine yerleştirmeye çalışan bir algoritma olmasından dolayı *Sum of Squared Error*'u değerini optimize eder [38]. Algoritma kümeleri net bir şekilde ayırır ve bir nokta aynı anda iki kümeye ait olamaz. Algoritma aşağıdaki gibidir.

Algoritma 2.1 K-Means Algorithm

Girdi: $k, \mathbf{A} = \{a_1, a_2, \ldots, a_n\}$ *data set*
Çıktı: $\mathbf{C} = \{c_1, c_2, \ldots, c_k\}$ *cluster centers*

1: $\mathbf{C} \leftarrow RandomPoints()$
2: **while** *until the centers don't move* **do**
3: $\quad \mathbf{S}_j \leftarrow null \; \forall j \in [k]$
4: $\quad$ **for** $a_i \in A$ **do**
5: $\quad\quad j \leftarrow argmin_j \|a_i - c_j\|$
6: $\quad\quad \mathbf{S}_j \leftarrow \mathbf{S}_j \bigcup a_i$
7: $\quad c_j \leftarrow \frac{1}{|S_j|} \sum_{a \in S_j} a, \forall j \in [k]$

Başlangıçta verilen k parametresi kadar rastgele nokta, küme merkezleri $\mathbf{C}$ olarak seçilir. Bütün noktalar $\mathbf{A}$ kendisine en yakın merkez noktasına atanır. Bu atama işleminden sonra her merkez noktası sahip olduğu küme noktalarının ağırlık merkezine hareket eder. Bu işlem merkez noktaları hareket etmeyene kadar devam eder. Hareket bittikten sonra kümeleme işlemi sonlandırılır. Bu çalışmada Algoritma 2.1 yoğun olarak kullanılacaktır. Probleme uygun olarak mesafe ve ağırlık merkezi hesaplamalarında bazı durumlar göz önüne alınacaktır.

Avantajları

- Uygulaması kolaydır.

- Veri kümeleri ayrıksa çok iyi sonuç verir.
- İşlem sayısı az olduğu için hızlı sonuç verir.
- Büyük veri kümelerinde kullanılabilir.

Dezavantajları

- Kaç kümeye ihtiyaç olduğu deneme yanılma yöntemiyle bulunur.
- Sadece merkeze uzaklığı değerlendirdiği için iç bükey olmayan kümeleri iyi ayıramaz.
- Her noktanın anlamı eşit olduğu için ayrık noktalar, sonucu olumsuz etkiler.
- Rastgele seçilen ilk konumlar sonucu etkiler.
- Doğrusal olmayan veri kümelerini iyi ayıramaz.

2.1.2 Pam Algoritması

K-Medoids algoritmasının bir türevidir [3]. Algoritma 2.1'deki gibi kümenin öğeleri dışından nokta seçmek yerine kümeye ait rastgele noktaları seçer. Algoritma 2.1'ye göre daha maliyetlidir fakat ayrık noktalardan daha az etkilenir. K-Medoids her adımda bir maliyet fonksiyonu hesaplar. Fonksiyon, küme noktalarının kendilerine en yakın medoid noktalarına uzaklıkları toplamını verir. Uzaklık hesaplanırken Minkowski, Euclidean veya Manhattan mesafe algoritmaları kullanılabilir.

Algoritma şu şekilde çalışır; verilen parametreye göre küme noktalarından rastgele medoidler seçilir. Medoid olarak seçilmeyen noktalar kendilerine seçilen mesafe algoritmasına göre en yakın medoide atanır. Atama işlemi bitince o an için maliyet fonksiyonu hesaplanır. Bundan sonra medoid olmayan rastgele bir nokta seçilir ve bir medoid ile yer değiştirilir. Yani eski medoidlerden biri artık medoid değildir. Medoid olmayan noktalar kendilerine en yakın medoidlere tekrar atanır. Maliyet fonksiyonu tüm noktalar ile baştan hesaplanır. Maliyet fonksiyonu mevcuttan büyük ise değişim geri alınır ve devam edilir. Maliyet fonksiyonu mevcuttan küçük ise değişiklik korunur. Seçilen yeni medoid sırayla kalan noktalar ile de yer değiştirir. Bu adımlar medoid olmayan noktaların hepsi için yapılır. Maliyet fonksiyonu değişmemeye başlarsa algoritma sonlandırılır.

2.1.3 Mini Batch K-Means

Mini Batch K-Means algoritması temelde Algoritma 2.1 gibi çalışan bir algoritmadır [32]. Algoritmanın, Algoritma 2.1'e göre farkı bütün verilerle çalışmak yerine rastgele verilerin seçilip kullanılmasından ibarettir. Her adımda seçilen veriler değiştirilir.

Avantajları

- K-Means'e göre daha hızlı çalışır.

Dezavantajları

- K-Means'e göre daha kötü bir sonuç verebilir.

2.1.4 Clara Algoritması

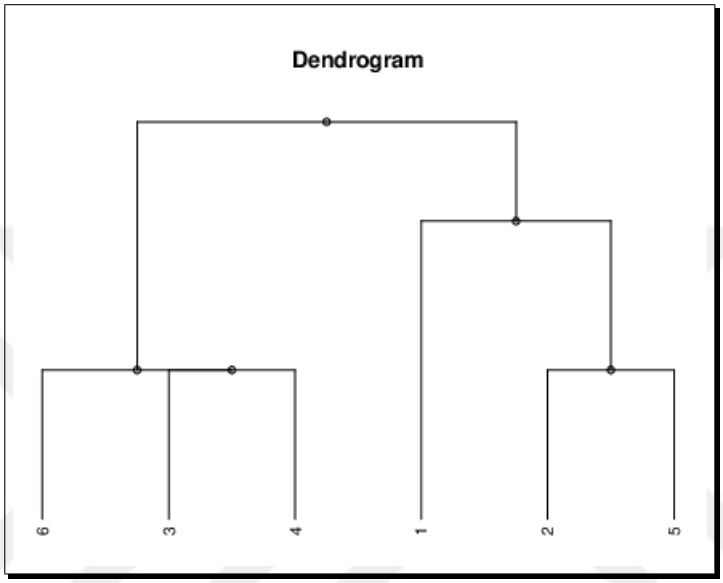
Pam algoritması büyük veritabanlarında performans probleminden dolayı iyi sonuç vermez. Clara Algoritması bu konuda çözüm üreten bir algoritmadır ve verinin tamamını kullanmaz [30]. Veriyi temsil edecek daha küçük bir küme seti alınır ve Pam algoritmasının adımları uygulanır. Bu sayede büyük veri tabanlarında daha hızlı sonuçlar alınır ama seçilen veri iyi bir temsil sunmazsa başarılı sonuçlar alınmayabilir.

2.1.5 Clarans Algoritması

Clara algoritması verinin tamamı yerine onu temsil eden daha az veride çalıştığı için bazen doğru sonuçlar vermeyebilir. Clarans algoritması bu problemi çözen bir algoritmadır [17]. Her adımda aynı verileri kullanmak yerine kullanılan veriler değiştirilir. Bu şekilde verinin belli bir kısmına göre hatalı sonuç bulmanın önüne geçilir.

2.2 Hiyerarşik Kümeleme Algoritmaları (Dendrogram ile Kümeleme)

Dendrogram ile kümeleme, hiyerarşik kümeleme algoritmalarından biridir. Ayrıştırıcı (DIANA) ve birleştirici (AGNES) kümeleme algoritmaları olarak ikiye ayrılır [30].



Şekil 2.1: Dendrogram yapısı

Dendrogramın en alt bölgesi tüm verilerin ayrık olduğu yeri gösterir. En üst bölgedeyse tek bir küme vardır. Birleştirici kümeleme algoritmalarında her adımda seçilen yakınlık algoritmasına göre en yakın iki küme birleştirilir. Ayrıştırıcı kümeleme algoritmaları ise bunun tam tersidir. Her adımda yeni kümeler ortaya çıkacak şekilde kümeler bölünür. Bu yöntem birleştirici algoritmalara göre daha az tercih edilir. Çünkü çok daha fazla işlem yapılması gerekir.

Avantajları

- Giriş parametresine gerek yoktur.
- İstenilen küme sayısı elde edildiği zaman durdurulabilir.
- Hızlıdır.
- Anlaşılması kolaydır.

Dezavantajları

- Hangi noktada durmak gerektiğine karar verilmesi büyük veri tabanlarında zor olabilir.
- Tek yönlü hareket vardır.
- Dendrogramda bir yöne hareket ettikten sonra eski küme bilgilerine erişilemez.

Dendrogramda kullanılabilecek yakınlık meyhodları ve bazı hiyerarşik kümeleme algoritmaları sırayla incelenecektir.

2.2.1 SLINK metodu (En yakın komşuluk metodu)

Slink kümeleme algoritmasında kümelerin birbirlerine en yakın noktaları seçilerek yeni küme oluşturulur [1].

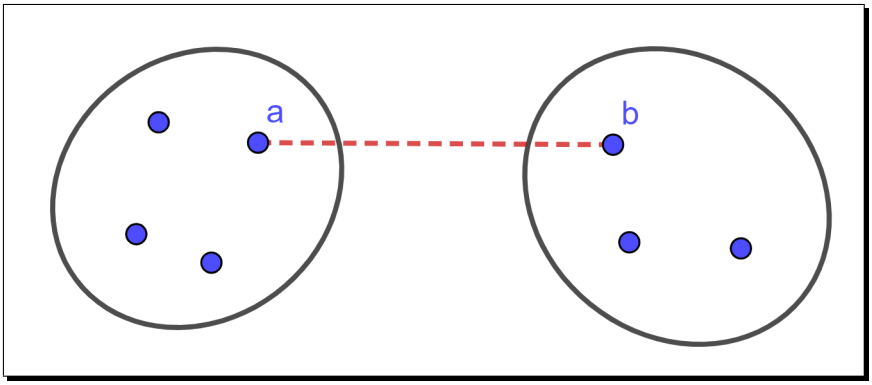
Dendrogramda bulunan en yakın mesafedeki iki küme birleştirilerek bir üst seviyede yeni bir küme elde edilmiş olur. Kümelerin birleştirilme işlemi bir treshold noktasında kesilir. Küme içinde birbirlerinin en yakınında olmayan diğer noktalar sonuca etki etmez.

Avantajları

- Ayrık kümelerde başarılı sonuç verir.

Dezavantajları

- Birbirine yakın iç içe geçmiş kümelerde erken birleştirme yapılacağı için kötü sonuçlar verebilir.
- Ayrık noktalardan olumsuz etkilenir.



Şekil 2.2: 2 küme arasında mesafeye bakılırken kümelerin birbirlerine en yakın olan *a* ve *b* noktaları ele alınır. Hesaplanan uzaklık diğer kümelerle karşılaştırılır.

2.2.2 CLINK metodu (En uzak komşuluk metodu)

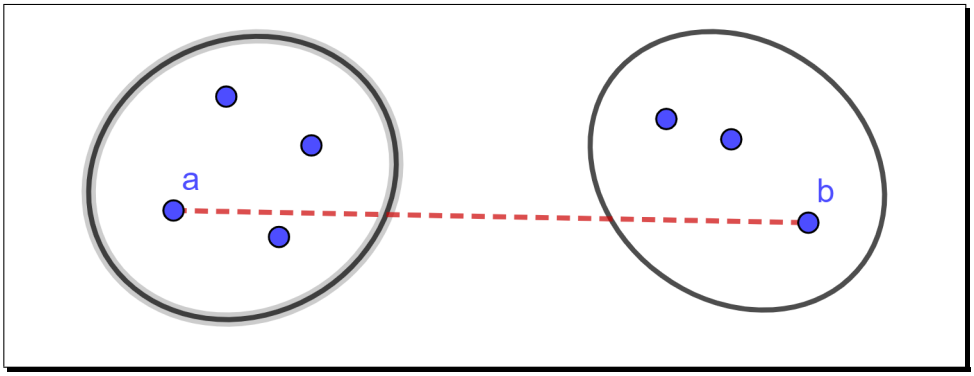
Clink kümeleme algoritması kümeler arasındaki en uzak mesafeyi kullanan yöntemi benimser [8]. Kümelerin içindeki diğer noktalar hesaba katılmaz. En uzak noktalar arasındaki mesafesi en küçük olan iki küme Dendrogramda birleştirilir ve bir sonraki adıma geçilmiş olur.

Avantajları

- Slink'e göre ayrık noktalardan daha az etkilenir. Birbirine yakın kümeleri ayırmada daha iyi sonuç verir.

Dezavantajları

- Ayrık noktalara Slink metodu kadar olmasa da duyarlıdır.
- Çokgen görünümlü kümeleri ayırmada başarısız olabilir.



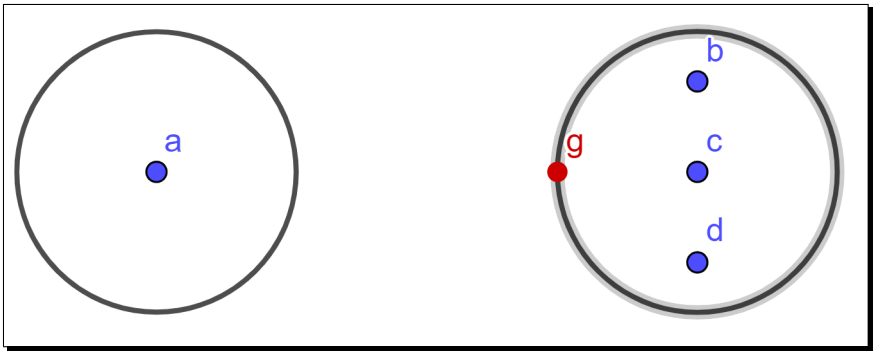
Şekil 2.3: 2 küme arasında birbirlerine en uzak nokta olan *a* ve *b* noktaları seçilir. Aradaki uzaklık diğer kümelerle karşılaştırılır. En küçük uzaklığa sahip olan 2 küme birleştirilir.

2.2.3 Average linkage metodu (Ortalama bağlantı yöntemi)

Average linkage yönteminde 2 kümedeki noktaların diğer kümedeki bütün noktalara olan uzaklıklarının ortalaması esas alınır [23]. 2 küme arasındaki elemanların her birinin diğer kümedeki elemanlara uzaklıkları toplanır ve ortalaması alınır. Sonuç diğer kümelerle kıyaslanır ve ortalama değeri en küçük olan 2 küme birleştirilir.

2.2.4 Centroid metodu (Küresel ortalama yöntemi)

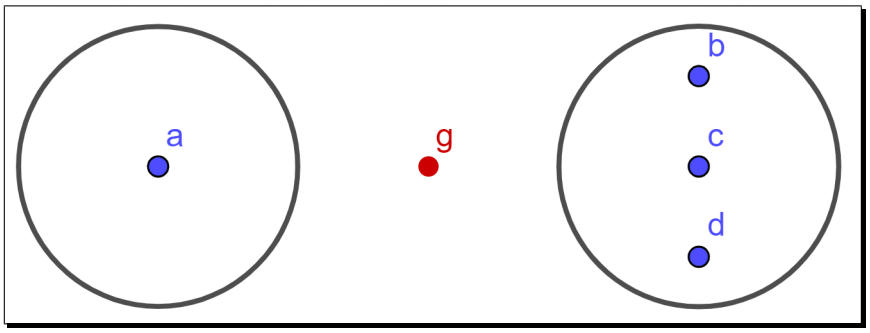
Centroid metodunda kümeler arasındaki mesafe hesaplanırken öncelikle kümelerin ağırlık merkezi hesaplanır [27]. Bu hesaplamalar sonunda ortaya çıkan ağırlık merkezlerine göre Öklid mesafesi kullanılarak en yakın kümeler birleştirilirken yeni küme merkezi ağırlıklar oranına göre belirlenir. Bu yöntemde tüm noktalar sonucu etkiler. Yeni küme merkezi nokta sayısı fazla olan kümeye daha yakın olacaktır.



Şekil 2.4: Yeni ağırlık merkezi sağdaki kümenin elaman sayısı daha çok olduğu için sağ taraftaki kümeye yakın olacaktır.

2.2.5 Median metodu

Median metodunda küme kendi içinde ağırlık merkezini hesapladıktan sonra ağırlığın bir önemi kalmaz. Yani Öklid mesafesine göre en yakın 2 küme ağırlık merkezi bulunduktun sonra birleşen kümenin yeni merkezi tam orta nokta olur [24]. Centroid metodunda olduğu gibi çok noktası olan kümeye yaklaşma olmaz.



Şekil 2.5: Birleşmeden sonra oluşacak yeni ağırlık merkezi, 2 kümenin noktalarından bağımsız şekilde Öklid mesafesine göre tam orta noktada olur.

2.2.6 Ward metodu

Ward metodu küme içi varyansı en küçük olacak merkez noktaları hedefleyen algoritmadır [25]. Küme içinde merkeze olan uzaklıkların karesi toplanır. İki farklı küme için yapılacak işlem şöyle olur:

$$\sum(x_i - x_c)^2 + \sum(x_j - x_c)^2 \tag{2.2}$$

Bulunan sonuç, iki kümenin birleşiminden oluşacak kümenin, yeni merkezine uzaklıklarının karelerinin toplamından çıkarılır.

$$\sum(x_{ij} - x_c)^2 - (\sum(x_i - x_c)^2 + \sum(x_j - x_c)^2) \tag{2.3}$$

Çıkan sonuçların en küçüğüne göre küme birleştirme işlemi gerçekleştirilir.

2.2.7 Cure Algoritması

Cure Algoritması hiyerarşik kümeleme algoritmalarından biridir [16][29]. Algoritma 3 tane parametre alarak başlar bunlar c, k ve α parametreleridir.

- k hedeflenen küme sayısıdır. Algoritmanın başında verilen nokta kadar küme olduğu kabul edilir. Küme sayısı k'ye düştüğü zaman algoritma sona erer.
- c ayrık kümeleri temsil eden nokta sayısıdır. Her küme için kendi sahip olduğu noktalardan iyi dağılmış olacak şekilde c tane seçilir.
- α, seçilen c noktalarının küme merkezine yakınlaştırılmasına yarayan katsayıdır ve α için 0 ve 1 arasında bir değer seçilmelidir. Her bir nokta α ile çarpılarak merkeze yaklaşır. α büyüdükçe aykırı noktaların ektisi azalır ama birleşmesi gereken kümeler birleşmeyebilir.

Başlangıçta her nokta bir clusterdır. Yani c verilen parametreden bağımsız $c = 1$ olur. Algoritma her adımda en yakın kümeleri birleştirir ve daha büyük bir küme elde eder. Yeni küme oluştuğunda küme içindeki noktalardan birbirlerine en uzak olacak şekilde c tane nokta seçilir ve bu noktalar α katsayısına göre merkeze yaklaştırılır. Birleştirme adımı k tane küme kalana kadar devam eder.

Avantajları

- Aykırı noktalardan etkilenmez.
- Küresel olmayan şekilleri iyi ayırabilir.
- Big data problemlerinde hızlı çalışır.

Dezavantajları

- Doğru parametreleri bulmak zor olabilir.

2.2.8 CHAMELEON Algoritması

Mevcut hiyerarşik birleştirme algoritmalarının bazı eksikleri bulunmaktadır. Bunun sebebi noktalar arasında dinamik bir ilişki olmamasıdır. Bu tarz eksiklikleri gideren bir algoritma olan Chameleon, 1999 yılında Karypis, Han ve Kumar tarafından sunulmuştur [18]. Örneğin çember şeklinde kümelenmiş noktalardan bazıları, ait olmadığı kümenin merkezine daha yakın olursa birçok kümeleme algoritmasında hatalar meydana gelebilir. Chameleon şekilsiz kümelerin ayrılmasında da iyi sonuçlar verir. Bunun temel sebebi Chameleon algoritmasının noktaların yakınlığına bakması dışında, ait olduğu küme içindeki noktalarla ilişkilerine de bakmasıdır [35].

Rock ve Cure algoritmasına göre daha iyi sonuçlar verir. Rock algoritması iki kümenin yakınlığını, Cure algoritması ise iki farklı kümenin arasında ne kadar bağ olduğunu göz ardı eder. Chamellon algoritması dinamik modeli sayesinde işlemciyi zorlasa da çok iyi sonuçlar verir.

Chameleon algoritması 2 adımdan oluşur. Birinci aşamada algoritma birbirine yakın küçük kümelere bölünür ve birbirine bağlı küçük kümeler elde edilir. Veriler arası ilişki için graph oluşturulur. Oluşturulan graphta her bir kenarın değeri yakınlık ilişkisini gösterdiği için yakınlık arttıkça kenar değeri de artar. Yani farklı 2 kümedeki i ve j noktaları birbirine ne kadar yakınsa graphta ilişkiye karşılık gelen kenarın değeri o kadar yüksek olur. İkinci aşamada ise küçük kümelerin üzerinde kümeleme işlemi yapılır. Chamellon bu aşamada 2 farklı dinamik değer hesaplar. Bunlar relative closeness $RC(C_i,C_j)$ ve relative interconnectivity $RI(C_i,C_j)$ değerleridir. Bu değerleri yüksek olan kümeler birleştirilir.

relative interconnectivity:

$$RI(C_i,C_j) = \frac{|EC_{CiCj}|}{\frac{|EC_{Ci}|+|EC_{Cj}|}{2}} \tag{2.4}$$

formülüyle ifade edilir. Aslında 2 kümenin birbirine ne kadar bağlı olduğunu gösteren bir değerdir. Payda bulunan $EC_{C_iC_j}$ 2 kümeyi birbirine bağlayabilecek kenarların ağırlıkları toplamını ifade eder. Bunun sayesinde 2 küme arasındaki noktaların sayısı anlamlı hale gelir. Paydadaki Ec_i ve Ec_j değerleri ise kabaca kümeleri

kendi içinde bölebilecek kenar sayılarının toplamıdır. Yani bir küme yoğunlaştıkça bu değerler artar ve $RI(C_i, C_j)$ değeri düşer.

relative closeness:

$$RC(C_i, C_j) = \frac{\overline{S}_{EC_{C_i C_j}}}{\frac{|C_i|}{|C_i+C_j|}\overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i+C_j|}\overline{S}_{EC_{C_j}}} \tag{2.5}$$

formülüyle ifade edilir. 2 küme arasındaki bağıl yakınlık değeridir. Bu değerin yüksek olması 2 kümenin birbirine yakın olduğunu ve birleştirilebileceğini ifade eder. $\overline{S}_{EC_{C_i C_j}}$ iki kümeyi birbirine bağlayan kenarların ortalamasını belirtir. $\overline{S}_{EC_{C_i}}$, $\overline{S}_{EC_{C_j}}$ değerleri kümelerin kendi kenarlarının ortalama ağırlığıdır. C_i, C_j ise kümedeki kenar sayılarıdır. Kümelerin kendi bağının kuvvetli olması $RC(C_i, C_j)$ değerini düşürecektir.

Bu iki değeri fazla olan kümeler birleştirilecektir. İstenilen treshold seviyesine kadar kümeler birleştirilmeye devam edilir.

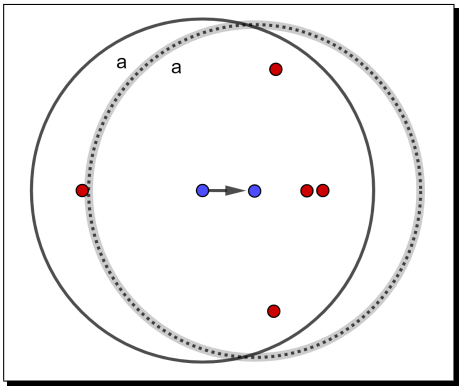
2.3 Yoğunluk Tabanlı (Density Based) Kümeleme Algoritmaları

Yoğunluk tabanlı kümeleme algoritmaları düzensiz ve şekilsiz verilerin kümelenmesinde kullanılan algoritmalardır. Ayrıca hedeflenen küme sayısını talep etmezler [6].

Bu çalışmada yoğunluk tabanlı kümeleme algoritmalarından Mean Shift ve Dbscan algoritmaları incelenecektir

2.3.1 Mean Shift(Ortalama Kaydırma) Yöntemi

Mean Shift algoritması yoğunluk tabanlı bir algoritmadır [7]. Algoritmada rastgele bir bölgeye yerleştirilen daire, içindeki noktaların ağırlık merkezine doğru hareket eder. Her adımda tekrar ağırlık merkezi hesaplanır ve yönelme devam eder. Hareket olmadığı zaman algoritma sonlanır. Dairenin başlangıç noktasına göre sonuç değişebilir.



Şekil 2.6: Yerleştirilen daire kendi içindeki noktaların ağırlık merkezine göre sağa doğru hareket eder.

2.3.2 Dbscan Algoritması

Dbscan algoritması yoğunluk temelli bir algoritmadır. Ester, Sander, Kriegel ve Xu tarafından ortaya çıkarılmıştır [12]. Yoğunluk temelli bir algoritma olduğu için uzayda şekilsiz biçimde dağılmış verileri iyi bir şekilde sınıflandırır. Bunu yaparken aykırı noktalardan kurtulur. Eps ve $MinPts$ olacak şekilde 2 farklı parametre alır ve seçilen parametreler algoritmanın sonucunu etkiler.

Eps parametresi noktanın komşuluk mesafesidir. Bir x noktasından eps uzaklığındaki tüm noktalar x noktasının komşusudur. $MinPts$ ise bir noktanın küme olabilmesi için gerekli olan minumum nokta sayısıdır. Yani x noktasının eps mesafesindeki komşularının sayısı $MinPts$'ye eşit veya büyükse ($komsuSayisi(x) \geq MinPts$) x noktası ve komşuları bir kümedir ve büyüyebilir. x noktası için 4 durum geçerlidir. $komsuSayisi(x) < MinPts$ ise ayrık kümedir ve herhangi bir kümeye dahil olmaz. $komsuSayisi(x) \geq MinPts$ ise ve komşularında daha önceden küme olarak işaretlenmiş bir nokta yoksa yeni bir küme oluşturulur. Komşularında daha önceden küme olmuş bir nokta varsa o kümeye dahil olur. İki farklı küme ile komşuysa kümeler birleştirilir ve tek bir küme elde edilir.

Dbscan herhangi bir kümeye dahil olmamış bütün noktalar için bu işlemi yapar ve noktalar birleşmeye başlar. Sonunda ayrık noktaların kümelere dahil edilmediği kümeler ortaya çıkar.

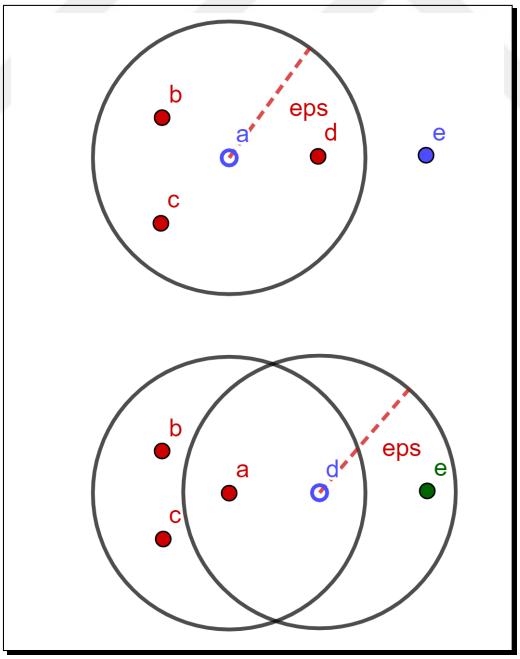
Avantajlar

- Ayrık noktaları eleyebilir.
- Şekilsiz kümelerde başarılıdır.

Dezavantaj

- Aldığı parametrelere karşı çok duyarlıdır.
- Deneme yanılma yapılıp doğru değerler bulunması gerekir.

Aşağıdaki grafikte *minpts* değeri 3 ve *eps* değeri 1 birimdir. Şekildeki *a* noktasının 3 komşusu *eps* kadar mesafe içinde olduğu için küme oluşmuştur. İkinci şekilde ise *eps* mesafesinde sadece 2 nokta olduğu için *e* noktası aykırı nokta olmuştur ve kümelere dahil edilmemiştir.

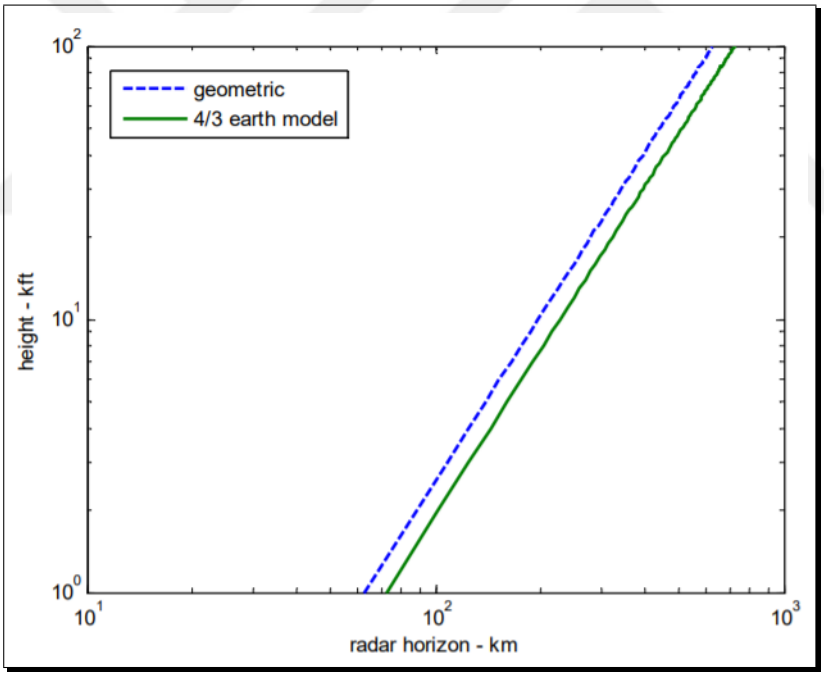


Şekil 2.7: Birinci şekilde *b*, *c* ve *d* noktaları *a* kümesi ile birleştirilirken, alttaki şekilde *d* noktasının *eps* bölgesinde sadece 2 komşusu olduğu için *e* noktası diğer noktalarla birleştirilmez.

3. COĞRAFİ RADAR DAĞITIM OPTİMİZASYONU

Hizmet veren sistemlerin optimum şekilde konumlandırılması önemli bir problemdir. Örneğin hastenelerin veya okulların inşa edilirken nüfus ve ulaşım gibi kriterlere göre konumları belirlenir. İnternet servis sağlayıcıları konumlandırılırken kullanım talepleri ve mesafeler göz önüne alınır.

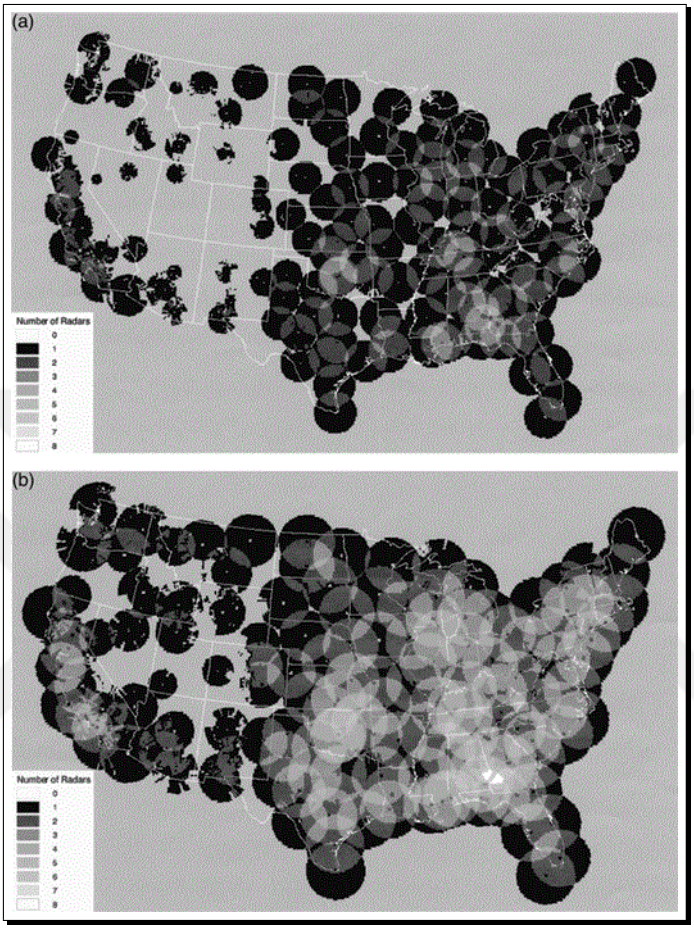
Hizmet veren servisler içinde belki de en önemli konu radar ve hava savunma sistemlerinin konumlandırılmasıdır. Kısıtlı sayıdaki radarların optimum olacak ve kör nokta bırakmayacak şekilde yerleştirilmesi zayiatları engelleyip savaşın kaderini tayin edebilir. Bu problem, esasında radar sistemlerinin sınırları olmasından kaynaklanır. Radarların taradıkları alanlar, bulundukları yere ve coğrafi durumlarına göre değişiklik gösterir. Bunun nedeni dünyanın şeklidir. Radarın görüş alanı sınırsız olsa bile deniz seviyesinde radarın kapsama alanı sınırlıdır [9]. Şekil 3.1'de rakıma göre görüş mesafesinin grafiği gösterilmiştir.



Şekil 3.1: Radar ufku ve rakım ilişkisi 4/3 dünya modeli

Dünyanın şeklinden dolayı uçan sistemlerin görüş açısı, kara sistemlerine göre daha avantajlıdır. Uçan sistemler içinde de daha yüksekte bulunan sistemin görüş mesafesi daha başarılı olacaktır. Örneğin hava ölçüm radarlarının gözlemleyebildiği alan, bulundukları yüksekliğe göre değişiklik gösterir [20]. Şekil 3.2'de

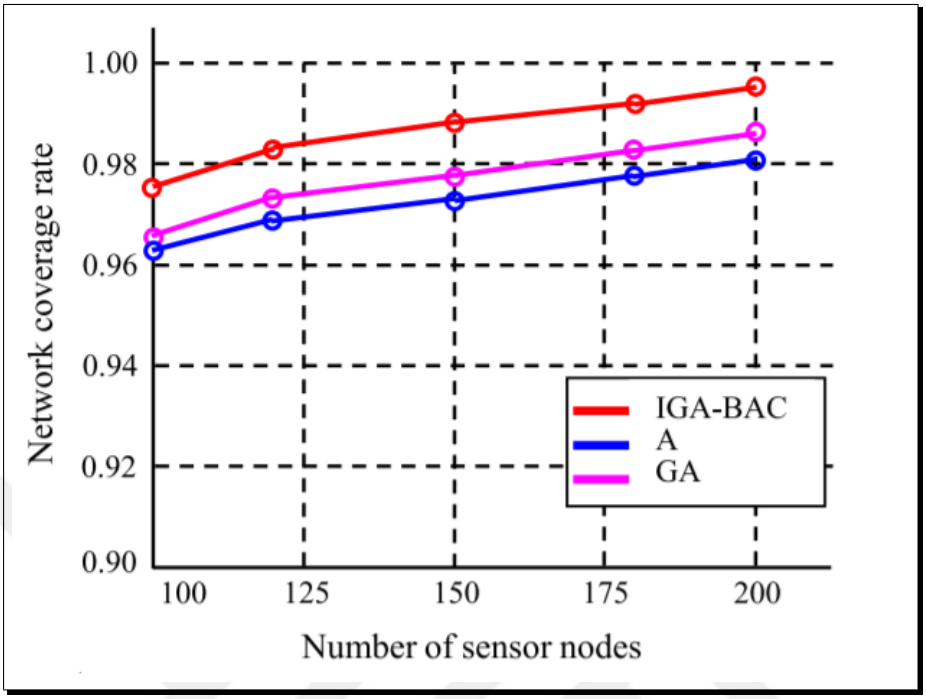
rakıma göre hava radarlarının kapsadığı alanların farkı gösterilmiştir.



:

Şekil 3.2: Rakıma göre hava radarlarının kapsadığı alan (a) 3 km (b) 5 km

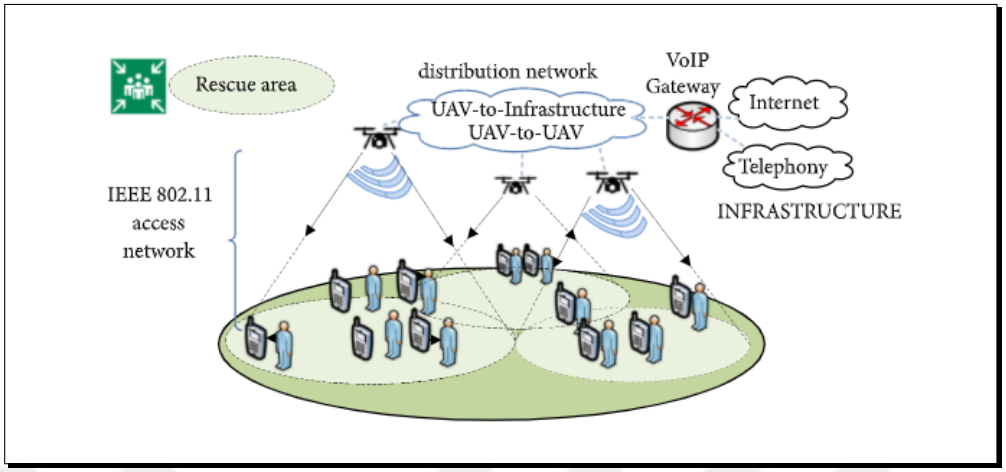
Servis sağlayıcılarının denize göre yüksekliklerinin etkisi yanında nasıl dağıtılacağına karar verilmesi de önemlidir. Bu konuda wireless sensörlerinin nasıl dağıtılması gerektiğine karar veren bir çalışma yapılmıştır. Bu araştırma sonucunda sensörlerin optimum şekilde nasıl dağıtılması gerektiğini bulmak için karınca ve genetik algoritmasının yanı sıra bu iki algoritmanın kombinasyonundan yararlanılmıştır [40]. Şekil 3.3'te sonuçlar gösterilmektedir.



Şekil 3.3: A (karınca algoritması), GA (Genetik Algoritma) ve IGA-BAC 2 algoritmanın kombinasyonu

Kapsayıcı sistemlerin ortak amacı hizmet verilen ortamdaki hedef varlıkları olabildiğince fazla kapsamaktır. Bu tezde coğrafi konumu verilen varlıkları, radarlar ile korumayı amaçlayan coğrafi radar dağıtım optimizasyonu problemi incelenmiştir. Kapsama alanı optimizasyonu için problem tanımları, çözüm yöntemleri ve deneysel sonuçlar sırasıyla verilmiştir.

Tezde tanımlanacak problemlere benzer çalışmalar farklı alanlarda yapılmıştır. Bu araştırmalardan bir tanesi felaket anında afet bölgesine iletişim amaçlı iha dağıtma projesidir [21]. Şekil 3.4 ile amaçlanan proje gösterilmiştir. İhaların dağıtımında K-Means ve Genetik algoritmalardan yararlanılmıştır.



Şekil 3.4

Benzer bir araştırmada ihalar, dağınık halde bulunan wireless sensör ağlarından veri toplamak için kullanılmıştır [4]. İhaların dağıtımında K-Means algoritmasından yararlanılmıştır.

Diğer bir benzer araştırma Kâbe'deki hacı adaylarına internet sağlamak için yapılmıştır [31]. Hac sırasında oluşan kalabalığın yoğunluk bölgelerine göre internet sağlayabilmek için ihalar kullanılmıştır. İhaların dağıtımında kullanılan algoritmalardan biri de K-Means algoritmasıdır.

Bu tezin konusu matematiksel olarak yukarıda tanımlanan problemlere benzer olsa da hava savunma radarları konusunda özelleşmiştir. Bu sebeple radarlar konusunda temel tanımlamalar yapılacaktır.

3.1 Radar Sistemleri

Temel olarak radar, yaydığı elektromanyetik enerjiyi yansıtan cisimleri tespit eden sisteme denir. Yankının dönüş süresine göre cisim ile ilgili tespitler yapılır. Bu sayede nesnelerin konumları, hızları ve hareket yönleri tespit edilir [36].

Elektromanyetik dalgalar için üç temel fizik kuralı geçerlidir [44].

- Elektromanyetik dalgalar iletken bir yüzeye çarptıkları zaman yansır.
- Elektromanyetik dalgalar ışık hızına yakın bir hızda yayılır.

- Elektromanyetik dalgalar doğrusal bir şekilde yayılır.

Radarların menzili zamana ve radarın gücüne bağlıdır. Zamanla ilgili olarak dalgaların yayılması ve yansıması arasında geçecek sürenin garanti edilmesi gerekir. İkinci olarak da elektromanyetik dalgaların yayılıp yansıdıktan sonra algılanabilecek kadar güçlü olması gerekir [44].

Radarlar farklı dalga boylarında çalışır. Frekans arttıkça dalganın taşıdığı enerji de artmaktadır. Bu enerji artışı daha fazla hava sürtünmesine sebep olacağı için menzili kısaltacaktır. Bunun yanında, dalganın frekansı arttıkça bant genişliği de artacağı için taşınan veri artacaktır. Ayrıca verinin iletilme süresi de kısalacaktır. Bu durumda frekans arttıkça menzil azalsa da hedeflenen cismin çözünürlüğü artacaktır. Bu sebeplerden ötürü arama radarları daha düşük frekanslarda olup uzun menzilde tarama yaparken, atış kontrol radarları daha yüksek frekanslarda olup daha düşük menzilde ve yüksek çözünürlükte tarama yapar.

Tezle ilgili deneylerin yapıldığı bölümde bazı radarlar (hava savunma sistemleri) kullanılacaktır. Yapılan deneylerde radar detayları, hesaplamaları basitleştirmek amacıyla ihmal edilecektir. Kullanılan farklı hava savunma sistemlerinin detay özellikleri ele alınmayacaktır. Farklı hava savunma sistemleri sadece menzil ve maliyet bakımından ayrışacaktır. Bu değerler ilgili deney boyunca sabit olacaktır.

3.2 Coğrafi Servis Dağıtımı

Coğrafi servis dağıtımı ifadesi radar dağıtımı ifadesine göre daha genel bir ifade olacağı için tercih edilmiştir. Tezin amacı radarların (hava savunma sistemi radarları) dağıtımı olsa da problem matematiksel olarak genelleştirilebilir. Radarların çeşitli ihtiyaçlara yönelik dağıtılması ve çeşitli seneryolara göre ihtiyaç duyulan radar tiplerinin bulunması problem tanımları ile modellenmiştir. Problemlerde kullanılacak çeşitli tanımlamalar da sırayla verilecektir.

3.2.1 Tanımlar

Tanım 1 (Operasyon bölgesi): Operasyon bölgesi (operational region) *OR*, dünya üzerinde sol-alt ve sağ-üstün koordinatları arasında tanımlanmış belirli bir diktörtgen bölgedir.

Tanım 2 (Coğrafi-referanslı varlık): Coğrafi-referanslı varlık (geo-spatial asset) A, üçlü $A =< id, loc, val >$ formülü ile ifade edilir. id, varlığı temsil eden benzersiz numaradır. loc, enlem x ve boylam y ile konum bilgisini verir($loc =< x, y >\in OR$) ve val, varlığa atanan değerdir. Her varlık OR'de tek bir noktayı belirtir ve val'ın artması varlığın daha kıymetli ve öncelikli olduğunu gösterir. A'nın özniteliklerine atıfta bulunmak için nokta notasyonu kullanılacaktır, yani $A.loc$, varlık A'nın konumunu ifade edecektir.

Coğrafi-referanslı varlık (kısaca varlık), OR'de önem atfedilen bir tesisi temsil eder. Örneğin askeri terminolojide askeri üs, köprü veya fabrika, varlık olarak değerlendirilebilir.

Tanım 3 (Coğrafi-referanslı servis): Coğrafi-referanslı servis (geo-referenced service) S, dörtlü $S =< id, loc, cov, cost >$ ile ifade edilir. Id, varlığı temsil eden benzersiz numaradır. loc, enlem x ve boylam y ile servisin lokasyon bilgisini verir ($loc =< x, y >\in OR$). cov, kapmasa fonksiyonunu ifade eder. $cost$ ise servis yerleştirme maliyetini belirtir. Her servis OR'de tek bir nokta ile belirtilir. S'nin özniteliklerine atıfta bulunmak için nokta notasyonu kullanılacaktır, yani $S.loc$, servis S'nin konumunu ifade edecektir.

Coğrafi-referanslı servis (veya basitçe servis) OR'de bir loc'a yerleştirilir. Askerî terminolojide, radarların özel bir lokasyona yerleştirilmesi servis olarak tanımlanabilir. Yerleştirilen bu servislerin görevi mümkün olduğunca yüksek değerde coğrafi-referanslı varlığı korumaktır.

Servisin kapsama fonkiyonu, OR'de incelenirken kapsam bakımından ikiye ayrılır: (i) kapsadığı alanlar, (ii) kapsamadığı alanlar. Örneğin bir servis dairesel alan ile ifade edilirse (yarı çapı r) daire içinde kalan bölge kapsanan bölge olurken daire dışında kalan bölgeler kapsanmayan bölge olarak ifade edilir. Yani kapsama fonksiyonu daire içinde kalan varlıkları hesaplayacaktır.

Servisler herhangi bir senaryoda coğrafi referanslı varlıkları korumaya çalışacaktır. Korunacak varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ile ifade edilebilir.

Tanım 4 (Kapsanan varlıklar): Bir servis S için mevcut olan varlıklar $\mathbf{A}$ arasındaki kapsanan varlıklar $Cover(S) = \{A \in \mathbf{A} : A.loc \in S.cov\}$ ile ifade edilir. Bu varlıkları bulmak için servisin varlıklara olan mesafelerini ölçebilen bir fonksiyon kullanılacaktır.

Tanım 5 (Servis-varlık mesafesi): Verilen Servis S ve varlık A arasındaki mesafe $Dist(S,A)$ fonksiyonu ile $S.loc$ ve $A.loc$ konumlarına göre hesaplanır.

3.2.2 Problem Tanımları

3.2.2.1 Problem 1 - Tek Servis ile Maksimum Değerli Varlık Kapsama

Askeri hiyerarşide askeri birlikler küçüldükçe birliğin sahip olduğu sistemler de azalmaktadır. Örneğin bir tugayın kontrol ettiği hava savunma sistemi araç sayısının, bir alayın kontrol ettiği hava savunma sistemi araç sayısından fazla olması beklenir. Bu sebeple bazı seneryolarda sadece 1 servisin olduğu durumların düşünülmesi gerekir.

Problem 1 (Tek servis ile maksimum değerli varlık kapsama problemi): Verilen sabit bir varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ve konumu belirtilmemiş servis $S =< id, loc, cov >$ için, kapsanan varlık değerlerinin toplamını maksimize eden servis konumunu $S.loc \in OR$ arayan problemdir. Problem şu formül

$$argmax_{S.loc \in OR} \sum_{A \in Cover(S)} A.val \tag{3.1}$$

ile ifade edilir.

Problem 1 konuşlandırılacak tek bir servis olduğunu varsayar. Servis maksimum değere sahip varlıkları kapsamaya çalışacaktır. Servis stratejik olarak daha önemli olan varlıklara öncelik verecektir.

3.2.2.2 Problem 1c

Servis optimum varlık kapsanacak şekilde konumlandırılmak istense de askeri kısıtlardan dolayı hava savunma sistemlerinin istenilen yere konumlandırılması mümkün olmayabilir. Bu durumda savunma sistemlerinin sadece önceden belirlenmiş yerlere konumlandırılması gerekebilir. Problem 1'in kısıtlanmış hali için özel bir tanım gerekecektir.

Problem 1c: Tanımlanan sabit bir varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ için konumu belirtilmemiş servisin $S =< id, loc, cov >$ yerleştirilebileceği aday noktalar

$\mathbf{C} = \{C_1, C_2, \ldots, C_n\}$ listesi ile verilmiştir. Kapsanan varlık değerlerinin toplamını maksimize eden servis konumunu $S.loc \in \mathbf{C}$ tanımlanmış noktalarda arayan problemdir. Problemin şu şekilde ifade edilir:

$$argmax_{S.loc \in \mathbf{C}} \sum_{A \in Cover(S)} A.val \tag{3.2}$$

Problem 1c (constrained) maksimum değere sahip varlıkları, önceden belirlenmiş konumlarda optimum şekilde kapsamayı amaçlayacaktır.

3.2.2.3 Problem 2 - Çoklu Servis ile Maksimum Degerli Varlık Kapsama

Gerçek bir hava savunma seneryosunda birbirinden farklı özelliklere sahip çok sayıda hava savunma sistemi bulunur. Bu sebeple mevcuttaki hava savunma sistemlerini, stratejik bölgeleri optimum şekilde kapsayacak şekilde dağıtmak komuta kontrol sistemlerinin önemli amaçlarından biridir.

Problem 2 (Maksimum degerli varlık kapsama problemi): Verilen sabit bir varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ve servis kümesi $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ için kapsanan varlık değerlerinin toplamını maksimize eden servislerin konumlarını $S_i.loc \in OR. \forall S_i \in \mathbf{S}$ arayan problemdir. Problemin tanımı aşağıda verilmiştir.

$$argmax_{S_i.loc \in OR. \forall S_i \in \mathbf{S}} \{ \sum_{A \in \bigcup_{\forall S_i \in \mathbf{S}} Cover(S_i)} A.val \} \tag{3.3}$$

Problem 2, birden fazla servis tanımlandığını varsayar. Mevcut servisler ile maksimum değerde varlık kapsanmaya çalışılacaktır.

3.2.2.4 Problem 2c

Askeri veya coğrafi kısıtlardan dolayı servisler Problem 2'deki gibi dağıtılamayabilir. Bu sebeple problemin yerleştirilebilecek servis noktaları için kısıtlanmış bir formunun tanımlanması gerekmektedir.

Problem 2c: Tanımlanan sabit bir varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ve konumları belli olmayan servis kümesi $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ için servislerin yerleştirilebileceği aday noktalar $\mathbf{C} = \{C_1, C_2, \ldots, C_n\}$ kümesi ile verilmiştir. Kapsanan varlık değerlerinin toplamını maksimize eden servislerin konumunu $S_i.loc \in \mathbf{C}.\forall S_i \in \mathbf{S}$ tanımlanmış noktalarda arayan problemdir. Problem 2c (constrained) maksimum değere sahip varlıkları, önceden belirlenmiş konumlarda, birden fazla servisle optimum şekilde kapsamayı amaçlayacaktır.

Problem 1 ve 2'nin çözümünde kullanılacak algoritmaların performansını ölçmek için "kalite skoru" tanımı kullanılacaktır.

Tanım 6 (Kalite skoru): Seçilen algoritmanın servis veya servisler ile kapsadığı varlık değerlerinin toplamı kalite skorunu verecektir.

3.2.2.5 Problem 3 - Minimum Maliyetle Tüm Varlıkları Kapsama

Komutanlıkların sorumlu olduğu bölgeler vardır. Bu sebepten dolayı ihtiyaçları birbirlerinden farklılık gösterir. Bazı seneryolarda ihtiyaca göre hava savunma sistemi temin edilmesi gerekebilir. Kapsanması istenen varlıklar tanımlandıktan sonra tanımlanan varlıkları minimum maliyetle kapsayacak hava savunma sistemlerinin hesaplanması maliyeti düşürecek ve diğer komutanlıklara daha çok fırsat oluşturacaktır.

Tanım 7 (Servis envanteri): Tanımlanan servis tipleri kümesi $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ verildiğinde servis envanteri kümesi **I**, S tiplerinden seçilerek belirlenir. Gösterim olarak $\mathbf{I} = \{I_1, I_2, \ldots, I_k\}$ ile ifade edilir. Her bir I_i'ye karşılık S_i kümesinden servis tipi seçilir. Bu yüzden, $I_i = \{S_{i_1}, S_{i_2}, \ldots, S_{i_{|I_i|}}\}$ ile ifade edilir.

Tanım 8 (Servis envanteri maliyeti): Verilen servis tipleri kümesi $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ ve envanter kümesi $\mathbf{I} = \{I_1, I_2, \ldots, I_k\}$ için servis envanteri maliyeti fonksiyonu $InAcqCost(\mathbf{S}, \mathbf{I}) = \sum_{i=1}^{i=k} |I_i| \times S_i.cost$ ile ifade edilir. Kısaca, $InAcqCost(\mathbf{S}, \mathbf{I})$ bize **I** maliyetini verir.

Problem 3 (Minimum maliyetle tüm varlıkları kapsama problemi): Verilen coğrafi referanslı varlıklar kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ve coğrafi referanslı servis

tipleri $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ için $InAcqCost(\mathbf{S}, \mathbf{I})$ formülünü minimize edecek envanter I'yı arayan problemdir. Matematiksel olarak $S_{i_1}.loc, S_{i_2}.loc, \ldots, S_{i_{|I_i|}}.loc \in OR. \forall S_i \in \mathbf{S}$ için şu şekilde ifade edilir:

$$\forall A \in \mathbf{A}. \exists S_{i_j} \in \mathbf{I} \; s.t. \; A.loc \in Cover(S_{i_j}) \tag{3.4}$$

Problemde hedefin hiçbir varlığı açık bırakmayacak şekilde kapsamak olduğu unutulmamalıdır. Kısaca tüm varlıklar en az maliyetle kapsanmaya çalışılacaktır.

3.3 Coğrafi Servis Dağıtım Problemi Çözümü

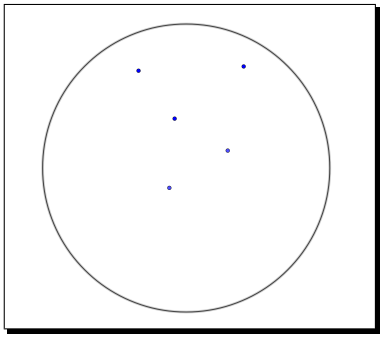
Coğrafi Referanslı Servis Dağıtımı konusundaki üç problemi çözmek için, birkaç somut spesifikasyona ihtiyaç vardır: (i) *OR* operasyon bölgesinin serbest veya kısıtlı arazi olup olmaması, (ii) $Cover(S)$ fonksiyonunu hesaplamak için kullanılanacak $Dist(S,A)$ fonksiyonu hesabı, yükseklik ve sinyal yansımalarının durumu (simetrik, asimetrik mesafa fonksiyonu). Birincisi ile ilgili olarak *OR* dünyanın şekli nedeniyle servisin yerleştirilemediği bölgeleri (su yüzeyi, derin vadi, dik yamaçlar vb.) içerebilir. Bu noktada, çalışma kapsamında serbest arazi modelinde servisin *OR* içinde herhangi bir yere yerleştirilebiceği kabul edilecektir. İkinci konuda ise yükseltilerden ve sinyal yansımalarından dolayı $Dist(S,A)$ fonksiyonu değişebilir. Basitlik için iki boyutta çalışılıp yükseklik ve sinyal yansımaları ihmal edilecektir. Bu sebeple matematiksel formüller için simetrik mesafe fonksiyonu seçilecektir.

3.3.1 Problem 1 Çözümü

Önerilecek çözümlerden önce optimum çözüm bulunacaktır. Geliştirilen algoritmalardaki temel hedef mümkün olduğunca optimum çözüme yakın ve daha performanslı sonuçlar bulabilmektir.

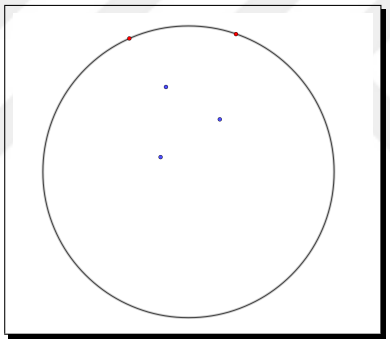
3.3.1.1 Optimum Çözüm

Optimum çözümü sağlayan birden falza servis noktası olabilir. Diğer bir deyişle optimum çözüm, maksimum değerde varlık içeren noktalar kümesidir. Sonuç olarak; servisin kapsadığı varlıklar değişmedikçe sonuç değişmez. Örnek olarak bir servisin kapsadığı varlıkların sayısı bakımından optimum sonuç veren pozisyonlarından bir tanesi aşağıdaki gibi olsun.



Şekil 3.5: Servis için optimum pozisyonlardan birini gösteren durum.

Bu servis öyle varlıklar kapsamıştır ki daha fazla değere sahip varlık kapsayamacağını biliniyor. Bu bilindikten sonra servisin Şekil 3.6'deki gibi yer değiştirmesi sonucu etkilemez. Çünkü kapsadığı varlıklar değişmeden servis kaydırılmıştır.



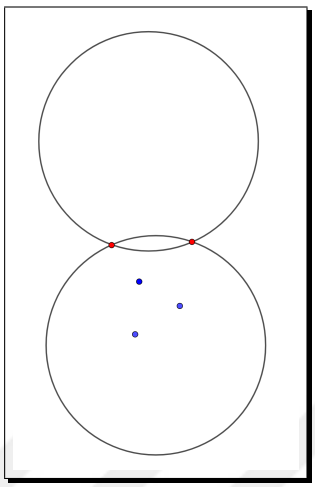
Şekil 3.6: Servisin çemberi üzerinde 2 varlık bulunacak şekilde kaydırılması.

Görüldüğü gibi servis 2 noktanın üzerinden geçerek aynı noktaları kapsamıştır. Bu durumda sonuç değişmeyecektir. Bunun yanında bazen bazı varlıklar diğer varlıklardan uzakta konumlanmış olabilir. Böyle durumlarda 2 nokta ile çember sağlanamayacaktır. Bu sebeple servisin merkezini sırayla tüm varlıkların konumuna da yerleştirmek gerekir.

Tanımdan ötürü optimum sonuç belirtilen durumlardan birinde olmak zorundadır. Servis 2 noktanın üstünden geçerek en optimum sonuca ulaşır veya merkezi bir varlık üstündedir. Algoritma bu durumların hepsine sırayla servisi yerleştirirse optimum sonucu bulmayı garanti edecektir.

Algoritma asimptotik olarak incelenirse, 2 noktanın çember oluşturabileceği tüm durumlar $n(n-1)/2$ kadardır. Yalnız her durumun 2 ile çarpılması gerekir; çünkü

iki nokta ile ifade edilebilecek iki çember oluşmaktadır. Şekil 3.7'de 2 nokta ile oluşturulabilecek 2 çember gösterilmiştir.



Şekil 3.7: İki nokta ile oluşturulabilecek iki çember bulunmaktadır.

Toplamda $n(n-1)$ kadar farklı durum oluşur. Her durum için kapsanan bütün varlıkların kontrol edilmesi gerekecektir. Sonuç olarak algoritmanın karmaşıklığı $O(n^3)$ kadar olacaktır. Servislerin varlık merkezlerine yerleşeceği durumda ise n farklı durum vardır. Bu durum için ise $O(n)$ kadar bir karmaşıklık olacaktır. Toplam algoritma karmaşıklığı optimum durumu bulmak için $O(n^3) = O(n^3) + O(n)$ kadar olacaktır. Geliştirilen algoritma 3.1 aşağıda tanımlanmıştır.

Algoritma tanımlandığı gibi iki aşamadan oluşmaktadır. Önce iki nokta ile oluşturulabilecek çemberler daha sonra da varlıkların pozisyonları kullanılarak oluşturulacak çemberler aranmıştır. $findCircles()$ methodu 2 varlık ile oluşturulabilecek servislerin merkez noktasını bulmaktadır. Bu sebeple z indexi 2 adımdan oluşmaktadır. $centerOfPoint$ şekil 3.7'te de ifade edildiği gibi 2 farklı çemberden oluşmaktadır.

Hava savunma sistemi radarlarının kapsama bölgesi kendi özelliklerinden ve coğrafi özelliklerden dolayı çember olmayabilir. Algoritma 3.1 radarların kapsama alanını çember olarak kabul edildiği zaman optimum sonucu vermektedir.

Geliştirilecek sonraki algoritma ile algoritma 3.1'deki optimum sonuca yakın ve asimptotik olarak daha hızlı bir algoritma hedeflenecektir.

Algoritma 3.1 Optimal Solution

Girdi: $\mathbf{A}, S$
Çıktı: $S.loc$

```
1: centerOfPoint ← [2]
2: selectedService ← null
3: tempService ← null
4: totalValue ← 0
5: maxValue ← 0
6: for i = 1 to |A| − 1 do
7:     for j = i + 1 to |A| do
8:         centerOfPoint ← findCircles(A_i, A_j, S_cov)
9:         for z = 1 to 2 do
10:            tempService.loc ← centerOfPoint[z]
11:            totalValue ← getTotalValue(A, tempService)
12:            if totalValue > maxValue then
13:                selectedService ← tempService
14:                maxValue ← totalValue
15: for i = 1 to |A| do
16:     tempService.loc ← A_i.loc
17:     totalValue ← getTotalValue(A, tempService)
18:     if totalValue > maxValue then
19:         selectedService ← tempService
20:         maxValue ← totalValue
21: S.loc ← selectedService.loc
```

Her bir varlığın sabit bir değeri ve sabit bir konumu olduğu biliniyor. Problem çözülürken varlığın değeri ağırlık olarak kabul edilir ve tüm varlıklar için bir kütle merkezi bulunursa servis o noktaya koyulabilir. Fakat servis konumu için varlıkların kütle merkezi en uygun çözümü vermeyebilir. Örneğin, birbirinden uzakta bulunan iki varlık olduğu düşünülsün. Kütle merkezi ortada olursa iki varlık da kapsanmayan bölgede kalabilir.

3.3.1.2 Ağırlık Merkezi ile Kümeleme

Kütle merkezi çözümü varlıkların konumlarına bağlı olarak başarısız olabileceği gözlemleniyor. Kütle merkezine ek olarak, tüm varlık kümesi iki küme olarak düşünebilir ve ilgili kütle merkezleri servise alternatif konum olarak düşünülebilir. Bu yöntemle az önce verilen iki varlık için olan örnekte servis için iki farklı alternatif nokta daha olacak ve bu noktalar iki varlığın doğrudan kendi merkezleri olacaktır. Bu iki alternatif nokta değerlendirebilir ve servisi konumlandırmak için en iyisi seçilebilir. Genel durum için ise 3 sınıflı kümeleme, 4 sınıflı kümeleme vb. ile yeni adaylar bulunabilir ve alternatif küme merkezleri değerlendirilebilir. Algoritma 3.2 bu prensipte çalışmaktadır.

Algoritma 3.2 Center of Mass with Clustering

Girdi: $\mathbf{A}, S$
Çıktı: $S.loc$

1: $k \leftarrow 1$
2: $bestval \leftarrow -\infty$
3: $bestloc \leftarrow center\ of\ OR$
4: **while** $k \leq |\mathbf{A}|$ and *user has time to run* **do**
5: $\quad ClusProK \leftarrow ClusterK(\mathbf{A}, k)$
6: $\quad bestlocK \leftarrow argmax_{loc \in ClusProK} EvalUtility(\mathbf{A}, loc)$
7: $\quad bestvalK \leftarrow EvalUtility(\mathbf{A}, bestlocK)$
8: $\quad$ **if** $bestvalK > bestval$ **then**
9: $\quad\quad bestval \leftarrow bestvalK$
10: $\quad\quad bestloc \leftarrow bestlocK$
11: $\quad k \leftarrow k + 1$
12: $S.loc \leftarrow bestloc$

Algoritma 3.2'de, *ClusterK* fonksiyonu Algoritma 2.1'ye benzeyen bir K-Prototip kümeleme algoritmasıdır. Parametre olarak coğrafi referanslı varlıkları **A** ve küme sayısı k'yı alır. Dönüş değeri olarak k adet prototip nokta döndürür. Ağırlık merkezi hesaplanırken Algoritma 2.1'den farklı olarak varlıkların değerleri hesaba

katılacaktır. Bu sebeple bulunan ağırlık merkezlerinin değerli varlıklara yakın olması beklenir. Böylece kıymetli varlıklara öncelik verilecektir. $EvalUtility$ (EU) fonksiyonu Denklem 3.5'te gösterilmiştir. Fonksiyon, S servisinin loc konumuna yerleştirilmesi durumu için kapsanan varlıkların değerlerini toplar. Aday noktası için kalite skoru hesabı yapan fonksiyondur. Bu fonksiyon S servisini ilgili loc konumuna konuşlandırdıktan sonra kapsayacağı varlıkların toplam değerini hesaplar.

$$EvalUtility(\mathbf{A}, loc) = \sum_{S.loc=loc \wedge A \in Cover(S)} A.val \tag{3.5}$$

Algoritma 3.2 any-time algoritmasıdır. Bu sebeple kullanıcı istediği bir noktada algoritmayı durdurup sonuç alabilir. Algoritmanın sona kadar çalışması her bir varlık noktasına servisin yerleştirilmesi demek olur ki bu hem iyi sonuç vermeyecektir hem de çok kaynak tüketecektir.

While döngüsü en fazla $|\mathbf{A}|$ kez çalışır. Bu nedenle algoritmanın karmaşıklığı (en kötü durumda) $O(n) = O(|\mathbf{A}| \times n) + O(|\mathbf{A}| \times CC)$ kadar olur. Burada $CC, ClusterK$ işlevinin karmaşıklığıdır. $EvalUtility$ fonsksiyonun karmaşıklığı ise $O(n)$ kadardır. Algoritma (i) serbest arazi modelinine göre çalışır ve (ii) $Dist(S,A)$ çevresel faktörlerden etkilenmeyen simetrik bir mesafe fonksiyonudur.

3.3.1.3 Maksimum Kesişim Bölgesi

Servis kapsama fonksiyonu $Cover(S)$ yarıçapı r olan bir daire kabul edilebilir. Bu şekilde fonksiyon daire içindeki bir varlıkla servis arasında en fazla r kadar mesafe olduğunu belirtir. Bu durumda problem tersine çevrilirse $A \in \mathbf{A}$ içindeki her bir varlık $(A.loc.x, A.loc.y)$ merkezli ve r yarıçaplı bir daire olarak kabul edilebilir. $Dist(S,A)$ fonksiyonu simetrik kullanılacağından dolayı daire içinde yarıçapı r olan herhangi bir servis A varlığını kapsayacaktır ve bu dairenin dışındaki hiçbir servis A varlığını kapsamayacaktır. Bu varlık, daire içindeki her bir noktaya $A.val$ kadar etki eder. Ayrıca, A_i ve A_j varlık çiftlerinin daireleri kesişirse, buradaki servis konumunun hem A_i'yi hem de A_j'yi kapsayacak şekilde yerleştirilebileceği anlamına gelir. Sonuç olarak kesişimlerin arttığı yerlere yerleştirilecek bir servisin daha çok varlık kapsadığı görülmektedir. Algoritma serbest arazi modeline göre çalışacaktır.

Problem çizge ile $G = (V, E)$ ifade edilebilir. Ağırlıklı çizge oluşturulurken her A için $A.val$ düğüm değeri ile düğüm V_A oluşturulur. Herhangi bir A_i ve A_j çiftinin

kendi dairelerinin kesiştiği durumda çizgeye bu ikili için bir kenar eklenir.

Teorem 1 (Maksimum ağırlıklı düğüm): Tanımlanmış sabit bir varlık kümesi $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ ve bir servis S verildiği durumda ağırlıklı çizgenin $G = (V, E)$ olduğu varsayılsın. Varlıklar $V_c \subseteq V$ maksimum düğüm ağırlına sahip bir düğüm olsun. Servis S, $\sum_{c \in V_c} weight(c)$ değeri ile *OR* içinde optimum bir yere yerleştirilebilir.

İspat: V_c bir düğüm olduğuna göre o bölgedeki varlıkların hepsini kapsayacak şekilde bir nokta vardır ve servis S bu noktaya konulabilir.

Teorem 1 sayesinde maksimum kesişim bölgesi aranabilir. Algoritma 3.3'te öncelikle düğüm ağırlıklı G oluşturulur. Daha sonra, maksimum kesişim bölgesini bulmak için $MaxVertexWeightClique(G)$ fonksiyonu çağrılır. Bu problem NP-Hard'dır. Geliştirilen sezgisel yöntem sonuca yaklaşabilir. Ayrıca memory problemlerinden dolayı maksimum düğümü bulmak yerine mümkün olduğunca yüksek sayıda kesişim noktaları bulunmaya çalışılacaktır. Son olarak, kümeye dahil olan A'ların kesişim bölgesi hesaplanıp, bu kesişme içindeki herhangi bir nokta, S'nin konulacağı yer olarak seçilebilecektir. Algoritma kesişim bölgesinin ağırlık merkezine servisi yerleştirir. Bu noktalar içinde en iyi kalite skorunu veren konum aranacaktır.

Algoritma 3.3 Maximum Intersection Set

Girdi: $\mathbf{A}, S$
Çıktı: $S.loc$

1: Construct $G = (V, E)$ as described in the text
2: $V_c \leftarrow MaxVertexWeightClique(G)$
3: Let A_{clique} to be assets corresponding to vertices in V_c
4: $intersectregion \leftarrow Intersect(A_{clique})$
5: $S.loc \leftarrow$ any point from $intersectregion$

Algoritma düğümleri ararken her kesişim sayısı için ayrı ayrı çalışacaktır. Örnek olarak öncelikle tekli düğüm durumu aranacaktır ki bu varlıkların kendi pozisyonu olacaktır. Daha sonra sırayla iki, üç ve fazlası olarak devam edecektir. Algoritma farklı parametrelerde çalışırken aynı sonuçları bulabilir ama bu sonucu etkilemeyecektir.

3.3.2 Problem 1c Çözümü

Askeri sistemler simüle edildiği zaman radarların yerleştirilebileceği bölgeler kısıtlı olabilir. Bu sebeple problemin çözümü kısıtlanarak optimum pozisyon aranacaktır.

3.3.2.1 Servis Konumları Aday Listesi

Servislerin yerleştirileceği $\mathbf{C} = \{C_1, C_2, \ldots, C_n\}$ aday listesi olduğu varsayılsın. Listedeki C_i enlem, boylam çiftini ifade etmektedir. Pozisyon aday listesi önceden belirlenmiştir. Servis bu aday konumlara sırayla yerleştirilip, her bir durum için kapsanan varlıklar hesaplanacaktır. Sonunda aday konumlar arasında en iyi kalite skoru veren nokta bulunmaya çalışılacaktır. Aday konumlar arasında en iyi konumu veren Algoritma 3.4 aşağıdadır.

Algoritma 3.4 Candidate List of Service Locations

Girdi: $\mathbf{A}, S, \mathbf{C}$
Çıktı: $S.loc$
1: $bestloc \leftarrow argmax_{loc \in \mathbf{C}} EvalUtility(\mathbf{A}, loc)$
2: $S.loc \leftarrow bestloc$

Algoritma, tanımından dolayı kısıtlı arazi modeli için hazırlanmıştır ve simetrik $Dist(S,A)$ fonksiyonu kullanır. Algoritmanın karmaşıklığı $O(|\mathbf{A}| \times |\mathbf{C}|)$ şeklindedir.

3.3.3 Problem 2 Çözümü

Problem 2, tanımından ötürü birden fazla servis içerir. Bu sebeple birinci çözümde yazılan algoritmalara bazı eklemeler olacaktır. Problem 2 çözümünde de 4 farklı yöntem kullanılacaktır. Servis envanteri kullanırken coğrafi referanslı varlıklar olabildiğince kapsanmaya çalışılacaktır.

3.3.3.1 Optimuma Yakın Çözüm

Çözüm 1'de kullanılan ve optimum sonucu veren algoritma ikinci problemde de kullanılacaktır; yalnız bu sefer tam optimum sonucu veremeyecek çünkü servis

seçim sırası optimum sonucu etkileyecektir. Bundan dolayı servis seçimi için en uygun senaryo denenmeye çalışılacaktır. Yapılacak farklı deneylerde sıralama algoritması servisleri kapsama alanına göre artan ve azalan sırada olacak şekilde sıralayacaktır. Yöntem algoritma 3.5'te detaylandırılmıştır.

Algoritma 3.5 Near Optimal Solution

Girdi: $\mathbf{A}, \mathbf{S}$
Çıktı: $\mathbf{S}.loc$

$centerOfPoint \leftarrow [2]$
$selectedService \leftarrow null$
$serviceList \leftarrow []$
$maxValue, totalValue \leftarrow 0$
$\mathbf{S} \leftarrow SortService(\mathbf{S})$
for $k = 1$ *to* $|\mathbf{S}|$ **do**
    for $i = 1$ *to* $|\mathbf{A}| - 1$ **do**
        for $j = i + 1$ *to* $|\mathbf{A}|$ **do**
            $centerOfPoint \leftarrow findCircles(A_i, A_j, \mathbf{S}_k.cov)$
            for $z = 1$ *to* 2 **do**
                $\mathbf{S}_k.loc \leftarrow centerOfPoint[z]$
                $totalValue \leftarrow getTotalValue(\mathbf{A}, \mathbf{S}_k)$
                if $totalValue > maxValue$ **then**
                    $selectedService \leftarrow \mathbf{S}_k$
                    $maxValue \leftarrow totalValue$
    for $i = 1$ *to* $|\mathbf{A}|$ **do**
        $\mathbf{S}_k.loc \leftarrow A_i.loc$
        $totalValue \leftarrow getTotalValue(\mathbf{A}, \mathbf{S}_k)$
        if $totalValue > maxValue$ **then**
            $selectedService \leftarrow \mathbf{S}_k$
            $maxValue \leftarrow totalValue$
    $serviceList \leftarrow serviceList \cup selectedService$
    $\mathbf{A} \leftarrow deleteAsset(\mathbf{A}, selectedService)$
    if $\mathbf{A} \leq null$ **then**
        break
$\mathbf{S} \leftarrow serviceList$

Problem 1 çözümünde olduğu gibi tanımlanan noktalar içinde optimum konum aranacaktır. En dış for döngüsü servis kümesini iterate etmektedir. Servislerin sırası sonucu etkileyeceği için servisler algoritmanın başında sıralanacaktır. Algoritma seçilen ilk servisi en optimum yere yerleştirecektir. Bir sonraki servise geçilmeden önce algoritma seçilen servisin kapsadığı varlıkları $deleteAsset()$ methodu ile silecektir. Böylece sıradaki servis en optimum noktayı ararken kapsanan varlık-

ları hesaba katmamış olacaktır. Algoritma, kapsanacak varlık veya yerleştirilecek servis kalmayana kadar devam edecektir.

3.3.3.2 Ağırlık Merkezi ile Kümeleme

Problem 1'in çözümünde olduğu gibi, Problem 2'nin çözümünde de kütle merkezi ile kümeleme çözümü kullanılabilir. Problem 1'den farklı olarak, bu kez çok sayıda coğrafi referanslı servis **S** vardır. İlk yöntemdeki gibi yine aday küme merkezleri bulunacaktır. Yalnız, k değeri bu sefer minimum 1 yerine $|S|$ olacaktır. Coğrafi referanslı birçok servis olduğundan, servislerin cluster merkezlerine doğru bir şekilde dağıtılması gerekmektedir. Bu sorunu çözmek için açgözlü bir yaklaşım izlenecektir. Artan k değerleri ile alternatif kümelemeler denenip, her bir kümeleme için $|S|$ farklı küme merkezi açgözlü bir şekilde seçilecektir. Bu nedenle, en büyük yarıçaplı coğrafi referanslı servisle başlanacak ve servisler sırayla en yüksek değerli coğrafi varlığın kapsandığı merkezlere yerleştirilecektir. Tüm coğrafi varlıkları kapsayana kadar mevcut coğrafi servisler kullanılacaktır. Algoritma, servislerin çakışmaması için bir servis yerleştirildikten sonra o bölgedeki varlıkları silecektir. Son servisi yerleştirdikten sonra toplam puanı hesaplayacaktır. Servisler bitmeden tüm varlıklar kapsanırsa, algoritma erken sona erecektir. Yöntem Algoritma 3.6'da detaylandırılmıştır.

S yinelenerek, servisler açgözlü bir yaklaşımla, en büyük servisten başlayarak en küçük servislere doğru optimum şekilde konumlandırılacaktır. Algoritma döngünün sonunda kullanılan servisleri optimum noktalara yerleştirecektir. Bunun dışında, her iterasyonda servisin yerleştirilmesiyle kapsanan varlıklar $tempA$'dan silinir. Bu sayede servislerin üst üste gelmesi engellenmiş olur.

Algoritma 3.6 Center of Mass with Clustering

Girdi: $\mathbf{A}, \mathbf{S}$
Çıktı: $\mathbf{S}.loc$

$k \leftarrow |\mathbf{S}|$
$bestval \leftarrow -\infty$
$bestlocs \leftarrow center\ of\ OR$
while $k \leq |\mathbf{A}|$ and *user has time to run* **do**
    $bestvalK \leftarrow 0$
    $bestlocsK \leftarrow \emptyset$
    $\mathbf{tempA} \leftarrow \mathbf{A}$
    $ClusProK \leftarrow ClusterK(\mathbf{tempA}, k)$
    for $i = 1$ *to* $|\mathbf{S}|$ **do**
        $S_i.loc \leftarrow argmax_{loc \in ClusProK} EvalUtility(\mathbf{tempA}, loc)$
        $bestvalK \leftarrow bestvalK + EvalUtility(\mathbf{tempA}, S_i.loc)$
        $bestlocsK \leftarrow bestlocsK \bigcup \{S_i.loc\}$
        $\mathbf{tempA} \leftarrow \mathbf{tempA} \setminus \{A \in \mathbf{A} : A.loc \in S_i.cov\}$
    if $bestvalK > bestval$ **then**
        $bestval \leftarrow bestvalK$
        $bestlocs \leftarrow bestlocsK$
    $k \leftarrow k + 1$
$\mathbf{S}.loc \leftarrow bestlocs$

3.3.3.3 Maksimum Kesişim Bölgesi

Bu yöntemde, Problem 1 algoritma 3.2'deki gibi maksimum ağırlıklı düğüm yöntemini kullanacaktır. Farklı olarak, tek bir servis yerine, birden çok servis seçilecektir. Algoritma belirli bir düğüm sayı aralığı için çalıştırılacaktır. Daha sonra servisler açgözlü bir yaklaşımla, bulunan kümelere yerleştirilecektir. Her yinelemede kapsanan varlıklar elenip, $MaxVertexWeightClique(G)$ fonksiyonu tekrar çalıştırılacaktır. Algoritma 3.7 tüm varlıkları kapsadığında ya da bütün servisleri kullandığında duracaktır ve kapsanan varlık değeri toplamını döndürecektir. Detaylar Algoritma 3.7'de verilmiştir.

Algoritma 3.7 Maximum Vertex Weight Clique

Girdi: $\mathbf{A}, \mathbf{S}, \mathbf{C}$
Çıktı: $\mathbf{S}.loc$

Construct $G = (V, E)$ as described in the text
$V_c \leftarrow MaxVertexWeightClique(G)$
Let $\mathbf{A_{clique}}$ to be assets corresponding to vertices in V_c
$INTS \leftarrow Intersect(\mathbf{A_{clique}})$
$bestlocs \leftarrow \emptyset$
tempA $\leftarrow$ **A**
for $i = 1$ *to* $|\mathbf{S}|$ **do**
    $S_i.loc \leftarrow argmax_{loc \in \mathbf{INTS}} EvalUtility(\mathbf{tempA}, loc)$
    $bestlocs \leftarrow bestlocs \bigcup \{S_i.loc\}$
    tempA $\leftarrow$ **tempA** $\setminus \{A \in \mathbf{A} : A.loc \in S_i.cov\}$
$\mathbf{S}.loc \leftarrow bestlocs$

3.3.4 Problem 2c Çözümü

Problem 2c çözümü, servislerin yerleştirilebileceği pozisyonların kısıtlanmış olduğu durumu çözecektir. Böylece gerçek bir askeri seneryo durumu için çözüm üretilmiş olacaktır.

3.3.4.1 Servis Konumları Aday Listesi

Bu çözüm yönteminde, $\mathbf{C} = \{C_1, C_2, \ldots, C_n\}$ aday çiftleri tanımlanmıştır. Problem 1'in ikinci çözümünde olduğu gibi coğrafi referanslı servislerin yerleştirilebileceği noktalar önceden belirlenmiştir. Servislerin aday noktalara optimum şekilde

yerleştirilmesi gerekiymektedir. Algoritma 3.8 coğrafi referanslı servislerin seçimi için $EvalUtility$ fonksiyonunu kullanacaktır. Servisler açgözlü bir yaklaşımla en büyük servisten en küçük servise doğru sırayla seçilecektir. Tüm varlıklar kapsanırsa veya mevcut servislerin hepsi kullanılırsa algoritma sonlanacaktır.

Algoritma 3.8 Candidate List of Service Locations

Girdi: $\mathbf{A}, \mathbf{S}, \mathbf{C}$
Çıktı: $\mathbf{S}.loc$

1: $bestlocs \leftarrow \emptyset$
2: **tempA** $\leftarrow$ **A**
3: **for** $i = 1$ *to* $|\mathbf{S}|$ **do**
4: $\quad S_i.loc \leftarrow argmax_{loc \in \mathbf{C}} EvalUtility(\mathbf{tempA}, loc)$
5: $\quad bestlocs \leftarrow bestlocs \bigcup \{S_i.loc\}$
6: $\quad \mathbf{tempA} \leftarrow \mathbf{tempA} \setminus \{A \in \mathbf{A} : A.loc \in S_i.cov\}$
7: $\mathbf{S}.loc \leftarrow bestlocs$

3.3.5 Problem 3 Çözümü

Askeri hava savunma sistemlerinin koruyabildikleri alanlar birbirinden farklıdır. Aynı şekilde savunma sistemlerinin yerleştirilme maliyetleri birbirinden farklılık gösterir. Bölgelerin veya stratejik noktaların en az maliyetle korunması istenen bir simulasyonda bazı savunma sistemleri maliyet etkin olmayabilir. Bu sebeple Problem 3'te tanımlanan servis tiplerinin sıralanması daha optimum bir çözüm için gereklidir.

Tanım 9 (Uygulanabilir servis envanteri): Verilen servis tipleri $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ kümesinin farklı maliyetleri olduğundan dolayı minimum maliyetli envanter $\mathbf{I} = \{I_1, I_2, \ldots, I_k\}$ kümesini bulmak için öncelikle servis tipleri maliyetlerine göre küçükten büyüye doğru sıralanacaktır. Böylece algoritma sırasında maliyetli servisler az kullanılmış veya kullanılmamış olacaktır. Bu durumda $S_i.cost < S_j.cost$ ve $|S_i.cov| < |S_j.cov|.\forall i, j \neq i$ formülü geçerli olacaktır.

3.3.5.1 Ağırlık merkezi ile kümeleme

Varlıkları minimum maliyetli servislerle kapsayabilmek için ağırlık merkezi ile kümeleme çözümleri Problem 3'te de kullanılmaya devam edilecektir. Algoritma

3.9, en dış döngüde tüm varlıklar kapsanana kadar yinelenir. Her yinelemede, kalan varlıklar k tane kümeye ayrılır ve her bir küme en uygun servis tipi ile kapsanıp kapsanmadığı kontrol edilir. Kapsanması durumunda küme içindeki kapsanan varlıklar, varlık kümesinden çıkarılır ve envantere seçilen servis eklenir. Algoritma ilerledikçe varlık kümeleri kapsanamadığından dolayı daha küçük kümelerde merkezler aranmaktadır. Bu maliyeti arttıran bir durumdur. Bu sebeple algoritma belli bir varlık kümesini kapsadıktan sonra k sayısını $k \leftarrow k-1$ adımı ile azaltır. Böylece varlıkların eksilmesiyle ortaya çıkan daha az kümeye bölerek ve daha merkezi noktalar ile kapsama ihtimalini de hesaplamış olur. Bunun nedeni, en dış döngünün sonraki yinelemesinde, varlıkları daha az sayıda kümeye bölerek kapsama fırsatı vermektir. Kümeler herhangi bir servisle kapsanamadığı durumda ise $k+1$ ile oluşturulacak küme sayısı arttırılacaktır. Küme sayısı arttıkça daha küçük kümeler oluşabileceği için kapsanma fırsatı oluşacaktır.

Algoritma 3.9 Center of Mass with Clustering (Greedy)

Girdi: $\mathbf{A}, \mathbf{S}$
Çıktı: $\mathbf{I}$

1: $k \leftarrow 1$
2: $\mathbf{I} = \{\emptyset, \emptyset, \ldots, \emptyset\}$
3: **while** $\mathbf{A} \neq \emptyset$ **do**
4: $\quad ClusProK \leftarrow ClusterK(\mathbf{A}, k)$
5: $\quad K \leftarrow k$
6: $\quad$ **for** $i = 1$ *to* K **do**
7: $\quad\quad C_i.loc \leftarrow$ *centroid of* $ClusProK(i)$
8: $\quad\quad S_j \leftarrow argmin_{j \in \{1,2,\ldots,k\}} \{A.loc \in S_j.cov \text{ with } S_j.loc = C_i.loc \ : A \in ClusProK(i)\}$
9: $\quad\quad$ **if** S_j exists **then**
10: $\quad\quad\quad \mathbf{A} \leftarrow \mathbf{A} \setminus ClusProK(i)$
11: $\quad\quad\quad I_j \leftarrow I_j \cup S_j$
12: $\quad\quad\quad k \leftarrow k-1$
13: $\quad k \leftarrow k+1$
14: **return I**

3.3.5.2 Ağırlık merkezi ile kümeleme - Rekürsif Optimizasyon

Algoritma 3.9 tanımlanan $InAcqCost(\mathbf{subS}, \mathbf{subI})$ fonksiyonunu optimize etmeyen açgözlü bir yaklaşımdır. Algoritma 3.10, Algoritma 3.9 ile aynı adımları izler, ancak herhangi bir servis S_j tarafından tamamen kapsanan her bir küme için daha fazla arama yapar. Kapsanan kümedeki varlıkların daha düşük toplam maliyetli

servislerle kapsanıp kapsanamayacağını kontrol eder. Bunun için, seçilen küme ve daha az maliyetli servis tipleri ile özyinelemeli çağrılar yapılır. Özyinelemeli çağrı dönüşleri $InAcqCost(\mathbf{subS}, \mathbf{subI})$ fonksiyonu ile hesaplanır ve normal akıştan bulunan sonuç ile karşılaştırılır. Maliyeti daha düşük olan servis veya servis grubu envantere eklenir. Bu yaklaşım ile Algoritma 3.9'da daha büyük yarıçaplı servislerle kapsanan varlık kümeleri maliyeti az olmak koşulu ile daha küçük ve sayıca fazla servisler ile kapsanır.

Algoritma 3.10 Center of Mass with Clustering (Recursive Optimization)

Girdi: $\mathbf{A}, \mathbf{S}$
Çıktı: $\mathbf{I}$

$k \leftarrow 1$
$\mathbf{I} = \{\emptyset, \emptyset, \ldots, \emptyset\}$
while $\mathbf{A} \neq \emptyset$ **do**
    $ClusProK \leftarrow ClusterK(\mathbf{A}, k)$
    $K \leftarrow k$
    for $i = 1$ *to* K **do**
        $C_i.loc \leftarrow$ *centroid of* $ClusProK(i)$
        $S_j \leftarrow argmin_{j \in \{1,2,\ldots,k\}} \{A.loc \in S_j.cov \text{ with } S_j.loc = C_i.loc \ : A \in ClusProK(i)\}$
        if S_j exists **then**
            $\mathbf{A} \leftarrow \mathbf{A} \setminus ClusProK(i)$
            if $j \neq 1$ **then**
                $\mathbf{subA} \leftarrow ClusProK(i)$
                $\mathbf{subS} \leftarrow \{S_1, S_2, \ldots, S_{j-1}\}$
                $\mathbf{subI} \leftarrow$ Make a recursive call with **subA** and **subS**
                if $InAcqCost(\mathbf{subS}, \mathbf{subI}) < S_j.cost$ **then**
                    $\mathbf{I} \leftarrow \mathbf{I} \cup \mathbf{subI}$
                else
                    $I_j \leftarrow I_j \cup S_j$
            else
                $I_j \leftarrow I_j \cup S_j$
            $k \leftarrow k - 1$
    $k \leftarrow k + 1$
return $\mathbf{I}$

3.3.5.3 En küçük daire ile hiyerarşik kümeleme

Problemin tanımından dolayı farklı kapsama alanına ve maliyete sahip servisler bulunmaktadır. Daha büyük bir servis seçilmesinin ana nedeni, o bölgede kümelenmiş varlıklar olmasındandır. Yani bütün varlıklar birbirinden çok uzakta olsaydı, sadece en küçük servislerin seçilmesi gerekecekti. Aksi durumda maliyet artacaktır. Bu bakış açısından yola çıkılarak hiyerarşik kümeleme algoritmaları [30] kullanılabilir. Başlangıçta her varlık maliyeti en düşük servis ile küme olarak temsil edilir ve birleştirici (AGNES) kümeleme algoritmaları kullanılarak oluşturulan her yeni küme için başta seçilen maliyeti en düşük servis yerine uygun servis kullanılabilir. Küme birleştirme sırasında varlıkların maliyetine öncelik verildiği için centroid yöntemi kullanılacaktır [27]. Burada karar verilmesi gereken ne zaman daha fazla maliyetli ve büyük bir servise ihtiyaç olacağıdır. Birleşim kümesindeki varlıkları kapsayacak en küçük daire bulunabilirse, ihtiyaç duyulan servis seçilebilir.

En küçük daire problemi En küçük daire problemi, düzlemdeki tüm noktalar kümesini içeren en küçük daireyi hesaplayan problemdir [11]. Düzlemdeki en küçük daire problemi, en kötü durumda doğrusal zamanda çözülebilir.

Algoritma 3.11'de en küçük daire probleminin çözümü için Welzl'in algoritması kullanılacaktır [43].

Algoritma yeni servis seçerken, birleştirilecek 2 kümenin varlıklarını kapsayacak en küçük daireden daha büyük olan en küçük yarıçapa sahip servisi seçecektir. Böyle bir servis yoksa veya bulunan servisin maliyeti birleştirilecek servislerin maliyetleri toplamından büyükse bu kümeler birleştirilmeyecektir. Algoritma, yerleştirilecek daha büyük bir servis bulamadığında sona erecektir.

Algoritmada $createServiceList()$ ile bütün varlıklar için en düşük maliyetli servisler seçilerek envanter oluşturulur. Bu envanter dendrogramda üstlere çıkıldıkça değişecektir. Kümeler arasındaki en küçük mesafeyi bulmak için bütük uzaklıkların hesaplanması gerekir. $computeDistanceMatrix()$ metodu ile kümeler arasındaki $Dist(S,A)$ ölçülür. $getSmallBall()$ metodu ile birleştirilecek kümelerdeki servislerin kapsadığı varlıkları kapsayacak en küçük daire bulunur. $getProperService()$ metodu bulunan daireden büyük en küçük servis tipini bulan metoddur. Dendrogramda daha üst seviyeye gidilemediği zaman algoritma sonlanır.

Algoritma 3.11 Hierarchical Clustering With The Smallest Circle

Girdi: $\mathbf{A}, \mathbf{S}$
Çıktı: $\mathbf{I}$

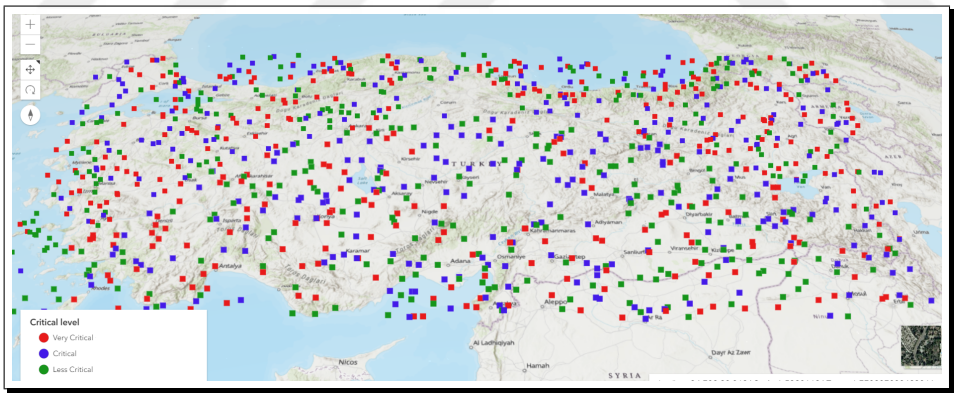
$\mathbf{I} \leftarrow createServiceList(\mathbf{A})$
$miniball = null$
while $true$ **do**
    $distanceMatrix[][] = computeDistanceMatrix(\mathbf{I}, \mathbf{A})$
    $distanceMatrix[cluster_x][cluster_y] \leftarrow getMinDistance(distanceMatrix)$
    if $distanceMatrix[cluster_x][cluster_y] == null$ **then**
        $break$
    $miniball \leftarrow getSmallBall(\mathbf{I}_{cluster_x}, \mathbf{I}_{cluster_y})$
    $S_i \leftarrow getProperService(\mathbf{S}, miniball)$
    if $S_i == null$ **then**
        $break$
    if $S_i.cost > (\mathbf{I}_{cluster_x} + \mathbf{I}_{cluster_y})$ **then**
        $continue$
    $\mathbf{I} \leftarrow \mathbf{I} \setminus (\mathbf{I}_{cluster_x} \cup \mathbf{I}_{cluster_y})$
    $\mathbf{I} \leftarrow \mathbf{I} \cup S_i$
    if $\mathbf{I}.size == 1$ **then**
        $break$
return $\mathbf{I}$

4. DENEYSEL DEĞERLENDİRME

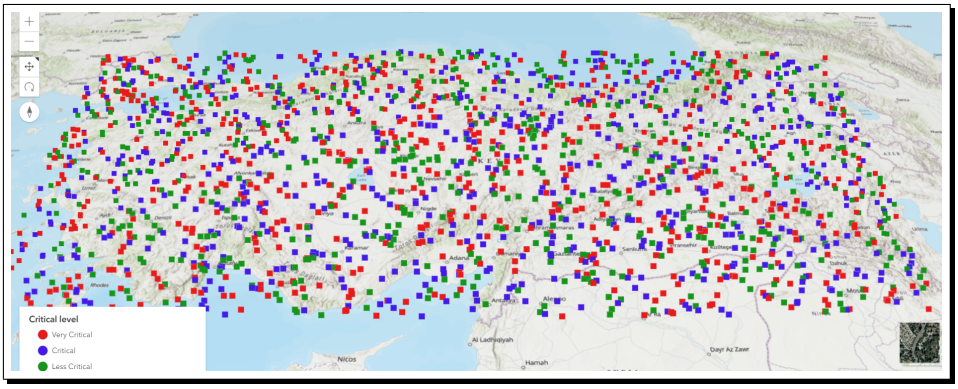
Algoritmalar java programlama dili kullanılarak gerçekleştirildi. Deneylerde kişisel bilgisayar olan 16 GB RAM'e sahip sekiz çekirdekli Intel i7 5700 CPU ile donatılmış donanım ve 64 bit Windows 10 işletim sistemi kullanıldı.

4.1 Veri Kümeleri

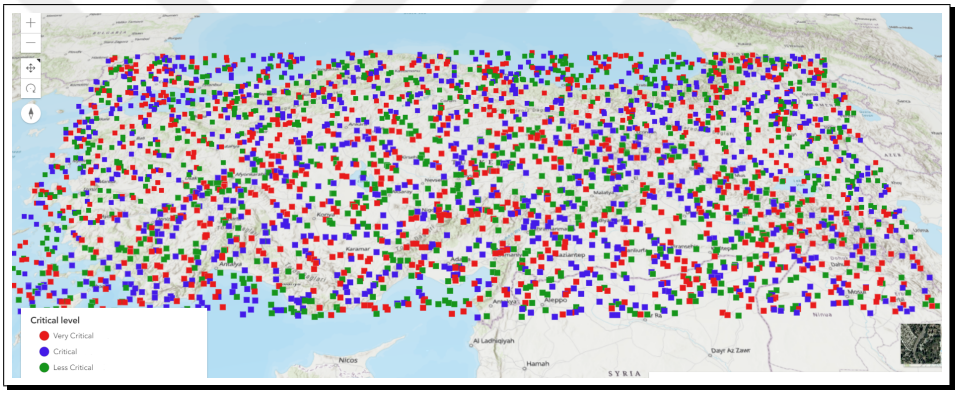
Deneylerde sırasıyla 1000, 2000 ve 3000 coğrafi referanslı varlık içeren `DSSPARSE`, `DSNORMAL` ve `DSDENSE` adlı üç farklı varlık veri kümesi oluşturulmuştur. Tüm veri kümelerindeki varlıkların konumu Türkiye'yi kapsayan dikdörtgen bölge içinden rastgele seçilmiştir (36.0K-42.0K enlem, 26.0E-45.0D boylam). Her varlığın değeri $\{1,2,3\}$ kümesinden rastgele seçilir. 4.1, 4.2 ve 4.3 şekillerindeki grafikler üç veri setini görselleştirir. Parsellerdeki varlıklar, kırmızı (değer 3, yüksek önem), mavi (değer 2, orta önem) ve yeşil (değer 1, düşük önem) ile renklendirilmiştir. Varlık tipleri tablo 4.1'de gösterilmiştir. Algoritmalarda enlem ve boylam değerlerine göre $Dist(S,A)$ fonksiyonu kullanılırken dünyanın yapısından dolayı mesafeye etki eden durumlar hesaba katılacaktır. Böylece farklı enlem aralıklarındaki aynı boylam uzaklıkları farklı olacaktır.



Şekil 4.1: `DSSPARSE`: 1000 tane coğrafi referanslı varlık içeren veri kümesi.



Şekil 4.2: DSNORMAL: 2000 tane coğrafi referanslı varlık içeren veri kümesi.



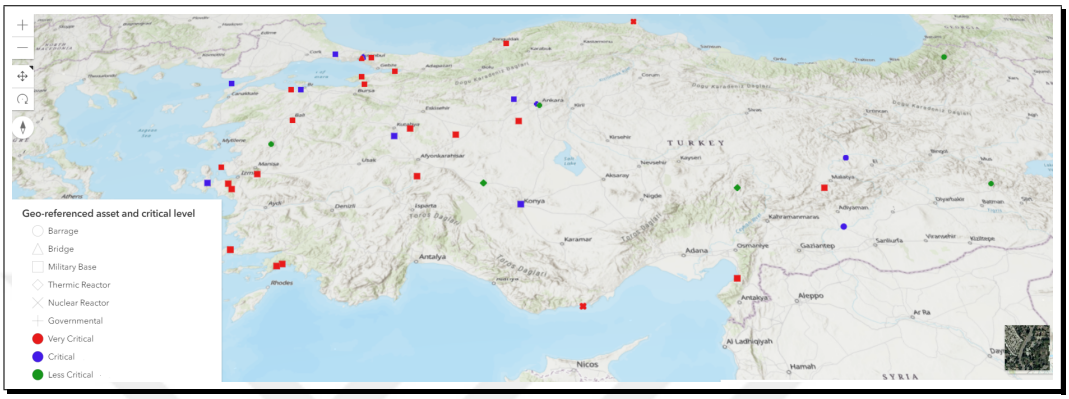
Şekil 4.3: DSDENSE: 3000 tane coğrafi referanslı varlık içeren veri kümesi.

Çizelge 4.1: VARLIK TIPLERI VE VARLIK DEĞERLERI.

VARLIK TIPI	VARLIK DEĞERI
Kırmızı	3
Mavi	2
Yeşil	1

Veri kümelerindeki varlıklar okullar, hastaneler ve kamu binaları gibi göreli olarak düşük öneme sahip varlıklar olarak düşünülmüştür. Stratejik olarak az öneme sahip bu varlıklar dışında, stratejik öneme sahip 40 varlık daha tanımlanmıştır. Bu varlıklar "askeri üs", "köprü", "baraj", "termik santral" ve "nükleer enerji santrali" gibi tesislerden seçilmiştir. Her stratejik varlığın değeri $\{100, 200, 300\}$ kümesinden seçilmiştir. Bu veri kümesi STRA olarak adlandırılmıştır. Stratejik noktaların

renklendirilmesi normal varlıklar gibi gösterilmiştir. Değeri 300 olan varlıklar kırmızı, değeri 200 olan varlıklar mavi ve değeri 100 olan varlıklar yeşil ile ifade edilmiştir. Şekil 4.4 stratejik coğrafi referanslı varlıkları, tablo 4.2 stratejik veri tiplerini, tablo 4.3 tanımlanmış 4 farklı veri setini gösterir.



Şekil 4.4: STRA: 40 tane stratejik coğrafi referanslı varlık içeren veri kümesi.

Çizelge 4.2: STRATEJIK VARLIK TIPLERI VE VARLIK DEĞERLERI.

Varlık Tipi	Varlık Değeri
Kırmızı	300
Mavi	200
Yeşil	100

Çizelge 4.3: VERI SETLERI VE IÇERDIKLERI VARLIK SAYILARI.

Veri Seti	Varlık Sayısı
DSSPARSE	1000
DSNORMAL	2000
DSDENSE	3000
STRA	40

Deneylerde temsili olarak farklı hava savunma sistemleri kullanılacaktır. Normalde hava savunma sistemleri çok katmanlı olmaktadır. Aynı bölgenin farklı özelliklere sahip hava savunma sistemleriyle desteklenmesi savunma için çok önemlidir (alçak, orta ve yüksek katmalı hava savunma sistemleri örnek verilebilir). Farklı hedefler için farklı sistemler ve füzeler kullanılır. Deneylerde bu farklılıklar ihmal edilecektir ve bir bölgeyi veya noktayı tek bir hava savunma sisteminin koruması amaçlanacaktır.

4.2 Problem 1 Deneyi

Deneyler temel olarak ikiye bölünecektir. İlk olarak, bölgesel savunma doktrini için stratejik olarak daha az önemli olan varlıklar (`DSSPARSE`, `DSNORMAL` ve `DSDENSE`) ile deneyler yapılacaktır. Daha sonra stratejik varlıklar (`STRA`) ile stratajik savunma doktrinine göre deney yapılacaktır. Sonunda algoritmalar iki savunma doktrini için karşılaştırılacaktır.

Kapsama fonksiyonunu tanımlamak için servisin (radar) kapsama alanı, seçilen konum merkezli bir daire olarak kabul edilmiştir.

Algoritma 3.2, maksimum 40 kümeyi hedefleyecek şekilde çalıştırılacaktır ($|\mathbf{A}|$ yerine 40 seçilecek). Bunun sebebi yaklaşık olarak küme sayısı 12 seçildikten sonra en iyi sonucun bulunmuş olmasıdır. Her ihtimale karşı işlemci ve bellek çok zorlanmadığı için 40 kümeye kadar çıkılacaktır.

Algoritma 3.3 NP-Hard olduğundan dolayı maksimum düğüm sayısı 20 ile sınırlandırılmıştır.

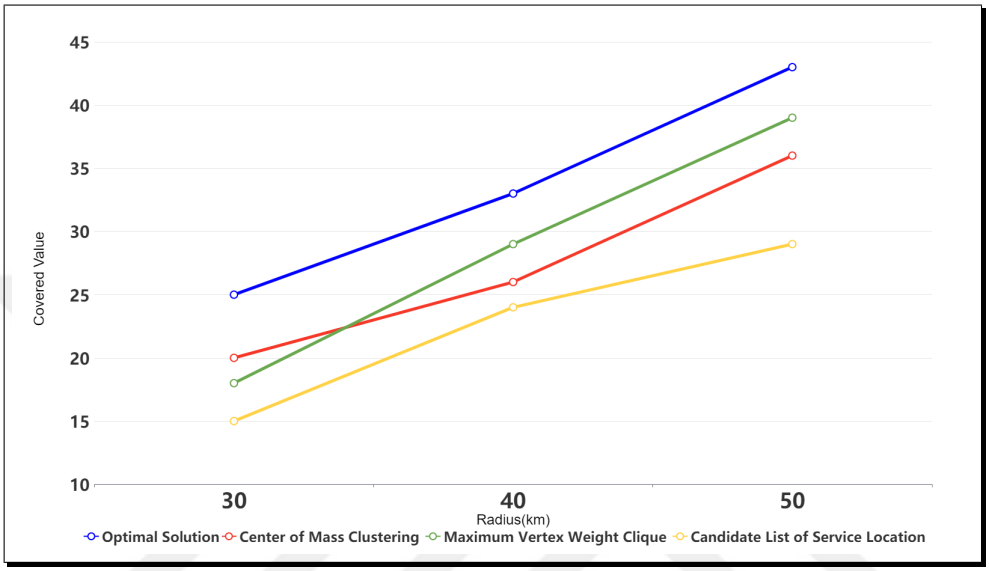
Algoritma 3.4 için Şekil 4.5'de gösterildiği gibi Türkiye bölgesi içinde rastgele seçilen 30 aday hizmet noktası oluşturulmuştur.

Şekil 4.5: "Candidate List of Service Locations" algoritmasında kullanılacak aday noktalar.

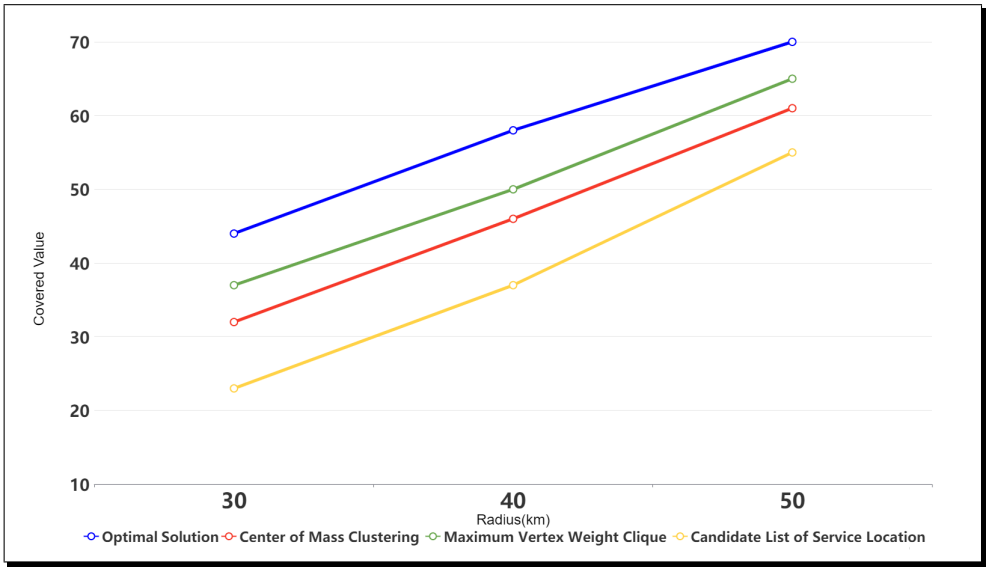
4.2.1 Deney 1 Sonucu

Yapılan ilk deneyde 30, 40 ve 50 kilometrelik yarıçaplara sahip 3 farklı radar tipi kullanılmıştır. Deneyde tanımlanan servisin kapsayabileceği varlıklara göre

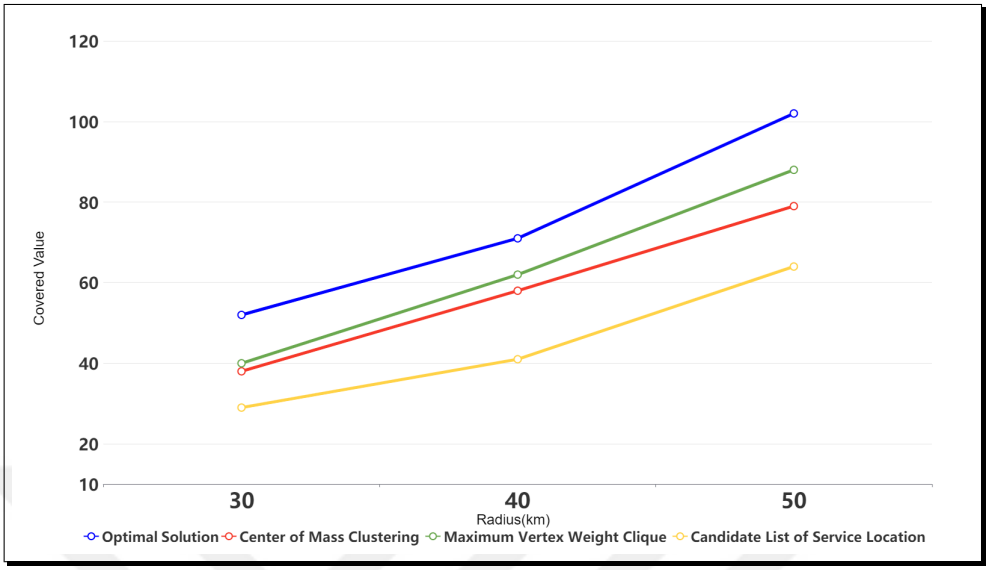
alacağı kalite skorlarının ölçülmesi amaçlanmıştır. Şekil 4.6, 4.7 ve 4.8'de verilen grafikler, sırasıyla DSSPARSE, DSNORMAL ve DSDENSE'deki üç çözümle elde edilen kalite skorlarını ölçer ve karşılaştırır. Bu deneyde stratejik bölge savunması amaçlanmıştır.



Şekil 4.6: Yoğunluk `DSSPARSE` olan veri seti seçildiğinde çıkan sonuç.



Şekil 4.7: Yoğunluk `DSNORMAL` olan veri seti seçildiğinde çıkan sonuç.



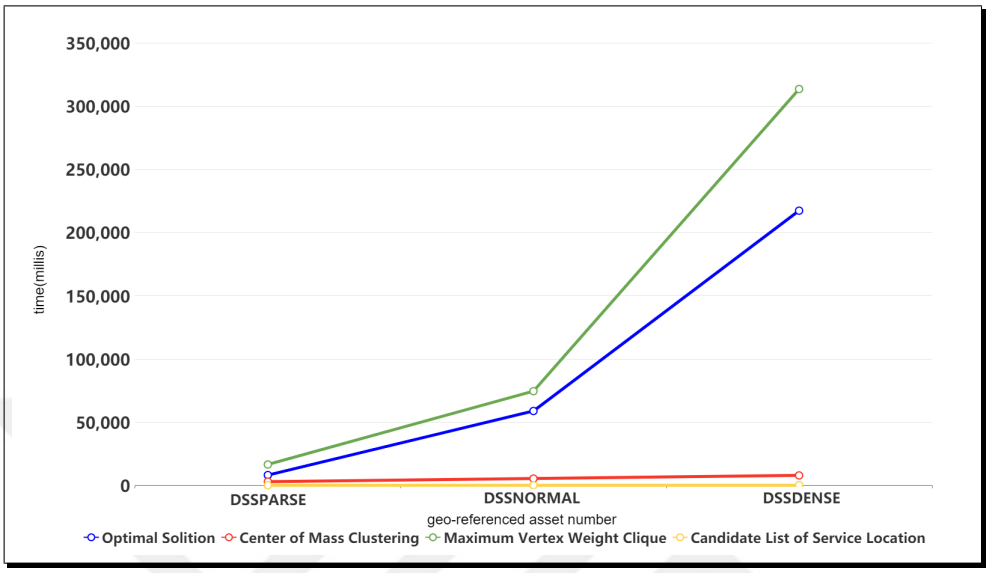
Şekil 4.8: Yoğunluk DSDENSE olan veri seti seçildiğinde çıkan sonuç.

Sonuçlara göre yarıçap büyüdükçe kalite skoru (yani, kapsanan varlık değerlerinin toplamı) monoton bir şekilde artmaktadır. Bu beklenen bir sonuçtur çünkü daha yüksek yarıçap daha yüksek kapsama anlamına gelir. Sonuç olarak yarı çarp arttıkça kapsama artar. Aynı şekilde veri kümesi ne kadar yoğun olursa kalite skoru da o kadar yüksek olmaktadır.

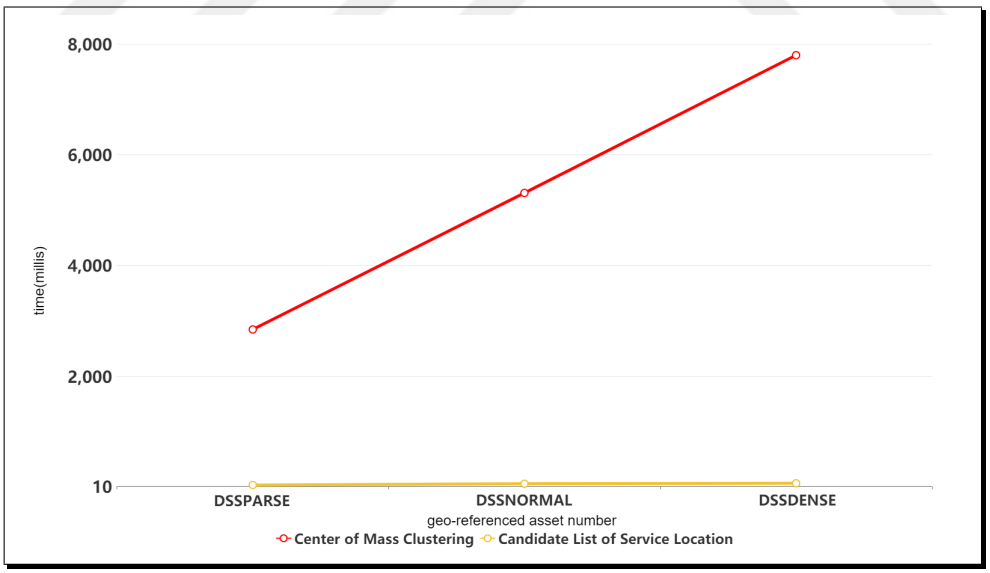
Sonuçlar değerlendirildiğinde algoritma 3.3 çözümü optimum sonuca (algoritma 3.1) en yakın sonucu vermektedir. Bunun yanında algoritma 3.3, algoritma 3.2'ye göre belleği ve işlemciyi daha çok zorlamıştır. Bu sebeple sonuçları almak zaman aldı. Algoritma 3.2'nin sonuçları da optimum sonuca yakın bir görüntü verdi. Farklı yoğunluklu veri kümelerlerinde yapılan deney sonuçlarına göre: Coğrafi referanslı varlıkların sayısındaki artışın, $EvalUtility$ fonksiyonunda doğrusal bir artışa neden olduğu gözlemlendi. Algoritma 3.4 diğer algoritmalara göre kötü bir sonuç vermiştir. Bu beklenen bir sonuçtur çünkü aday noktaları yetersiz kalmıştır.

Sonraki deneyde algoritmalar harcadıkları zamana göre kıyaslanmıştır. Optimum çözümün harcadığı zaman formülünden dolayı veri sayısı arttıkça artmaktadır. Önerilen algoritmalardan beklenilen optimum çözüme göre çok daha iyi sonuçlar vermeleridir. Yapılan deneylerde farklı sayıdaki varlık kümelerinde yarı çapı 50km olan bir servis kullanılmıştır. Beklenildiği gibi algoritma 3.4 en iyi sonucu vermiştir. Algoritma 3.2'den biraz daha iyi sonuç veren algoritm 3 çok yavaş çalışmıştır. Bunun yanında algoritma 3.2 optimum algoritmaya yakın bir sonuç verdiği gibi hızlı da çalışmaktadır. Şekil 4.9 ve 4.10 algoritmaları zamana göre karşılaş-

tırmaktadır.



Şekil 4.9: The "Maximum Corner Weight Clique" yöntemi diğer üç algoritmaya göre yavaş çalışır.



Şekil 4.10: The "Candidate List of Service Locations" Yöntemi en hızlı çalışan algoritmadır.

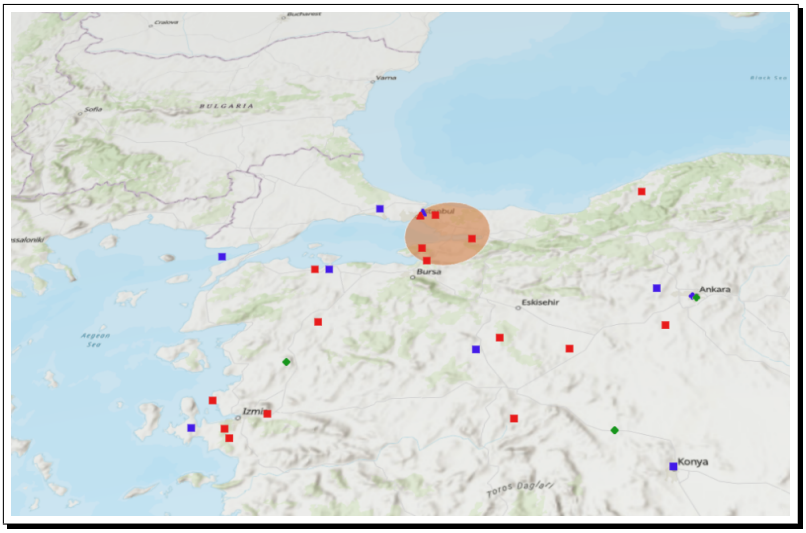
Şu ana kadar rastgele bir veri seti üzerinde bölgesel savunma amaçlı deneyler yapıldı. Deneylerde kullanılan coğrafi referanslı rastgele varlıklar: Okullar, hastaneler, kamu binaları ve alışveriş merkezleri gibi düşük öneme sahip noktalar olarak düşünüldü. Sıradaki deney stratejik olarak önemli noktalar ile yapılacaktır. Böylece stratejik savunma amaçlı doktrine göre deneyler yapılacaktır. Deneyde daha önceden tanımlanan `STRA` veri kümesi kullanılacaktır.

Savunma amaçlı simülasyon amaçlandığı için kapsama alanına karşılık gelen servisler Türk Silahlı Kuvvetleri'nin 50 km menzilli Hisar-O ve 200 km menzilli S-400 hava savunma sistemleri olarak belirlenmiştir. Tablo 4.4 deneyde kullanılacak radar tiplerini ve kapsama yarıçaplarını göstermektedir.

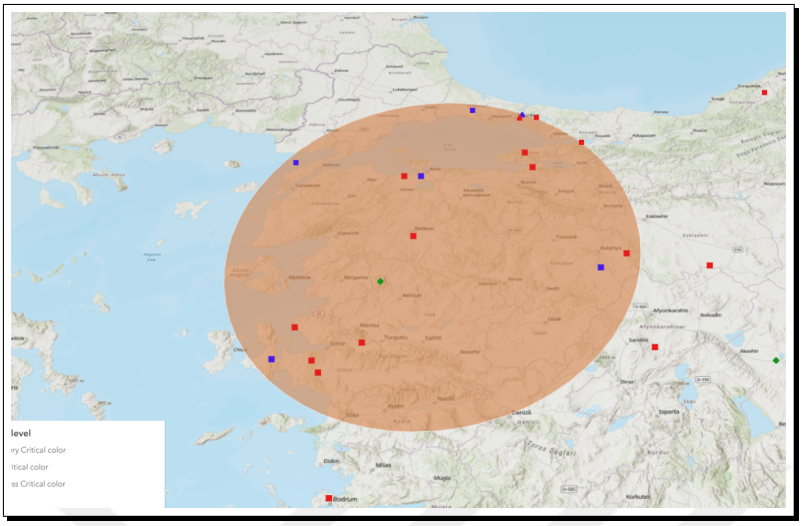
Çizelge 4.4: RADARLAR TIPLERI VE KAPSAMA YARIÇAPLARI.

RADAR TIPI	KAPSAMA YARI ÇAPI(KM)
Hisar-O	50
S-400	200

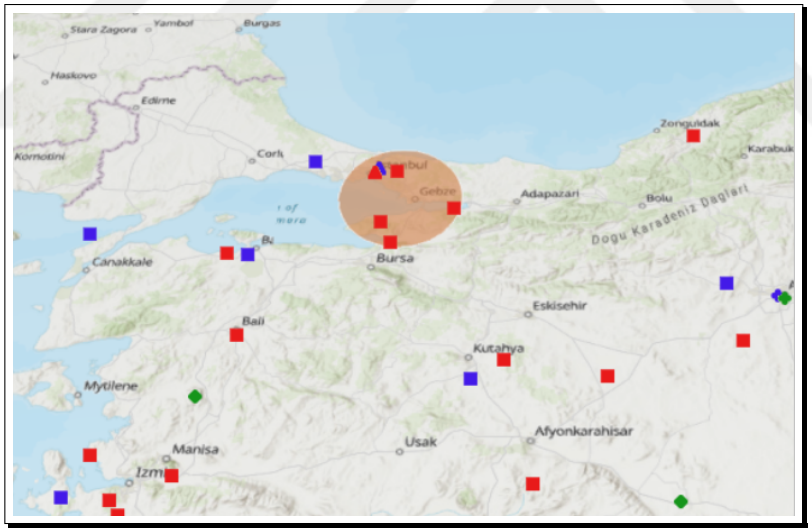
Şekil 4.11, 4.12, 4.13, 4.14, 4.15, 4.16 ve 4.17'de seçilen servislerin algoritmalar sonunda yerleştiği konumları gösterilmiştir. Servislerin son konumlarına dikkat edilirse İzmit-İstanbul gibi Türkiye'nin stratejik varlıklarının daha kıymetli ve yoğun olduğu bölgedir. Algoritmalar servisleri bu bölgeye yerleştirmiştir.



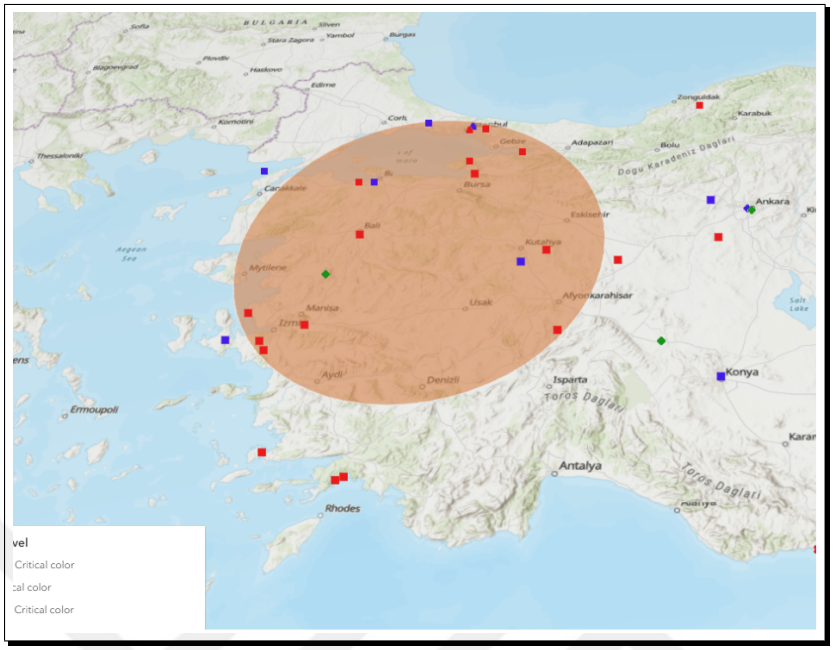
Şekil 4.11: Coğrafi-referanslı Hisar-O servisinin "Optimal Solution (Algoritma 3.1)" yöntemine göre son konumu. Kalite skoru 1700'dür.



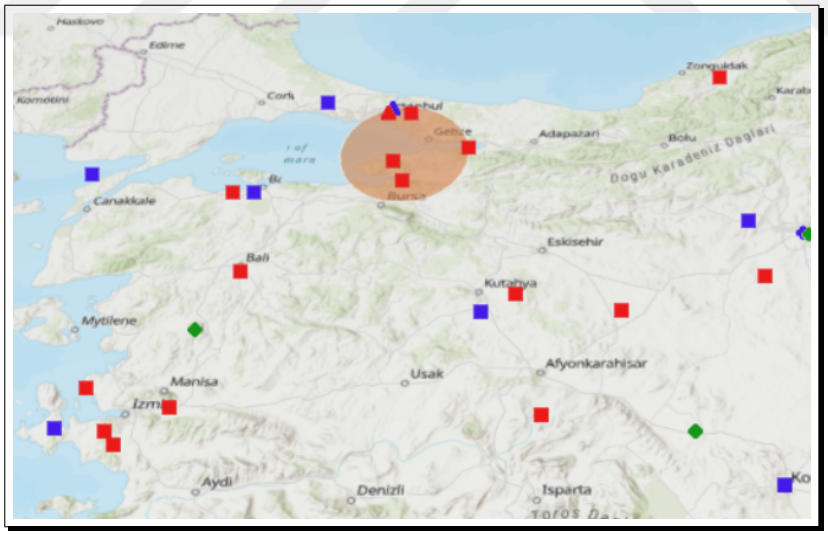
Şekil 4.12: Coğrafi-referanslı Hisar-O servisinin "Optimal Solution (Algoritma 3.1)" yöntemine göre son konumu. Kalite skoru 4900'dur.



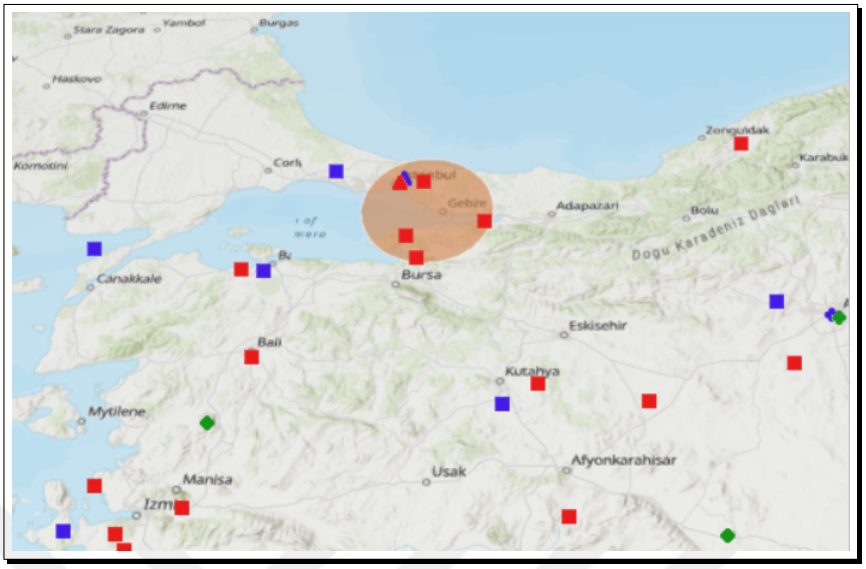
Şekil 4.13: Coğrafi-referanslı Hisar-O servisinin "Center of Mass with Clustering (Algoritma 3.2)" yöntemine göre son konumu. Kalite skoru 1700'dür.



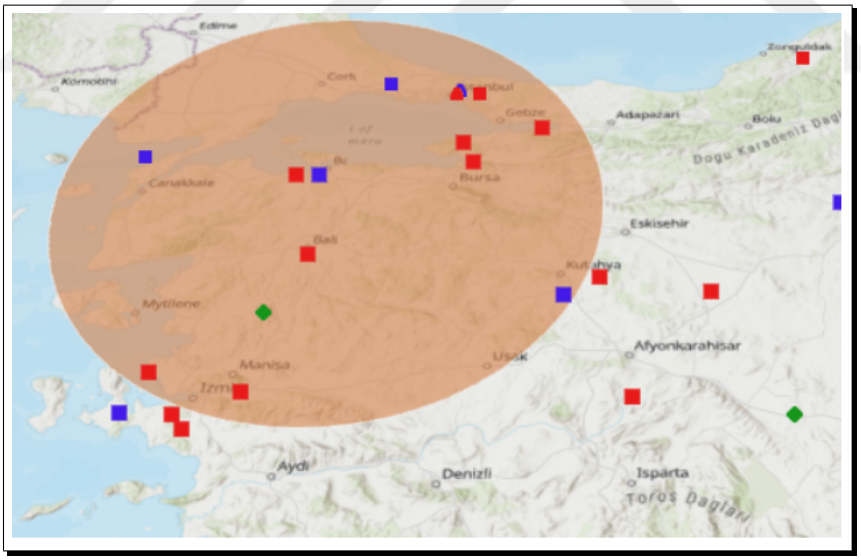
Şekil 4.14: Coğrafi-referanslı S-400 servisinin "Center of Mass with Clustering (Algoritma 3.2)" yöntemine göre son konumu. Kalite skoru 4800'dür.



Şekil 4.15: Coğrafi-referanslı Hisar-O servisinin "Maximum Vertex Weight Clique (Algoritma 3.3)" yöntemine göre son konumu. Kalite skoru 1700'dür.



Şekil 4.16: Coğrafi-referanslı Hisar-O servisinin "Candidate List of Service Locations (Algoritma 3.4)" yöntemine göre son konumu. Kalite skoru 1300'dür.



Şekil 4.17: Coğrafi-referanslı S-400 servisinin "Candidate List of Service Locations (Algoritma 3.4)" yöntemine göre son konumu. Kalite skoru 4300'dür.

Algoritma 3.2, stratejik savunma deneyinde optimum sonuca çok daha yakın sonuçlar verdi. Algoritmaların hızları arasında ciddi bir fark varken alınan sonuç algoritmanın etkin olduğunu göstermektedir.

Algoritma 3.3, Hisar-O deneyinde başarılı sonuç verirken S-400 sistemi için yapılan deneyde bellek yetersizliğinden dolayı sonuçlanamamıştır.

Algoritma 3.4, çok hızlı sonuçlar verse de aday noktalar iyi bir sonuç için yeterli gelmemiştir.

Deney 1'de yapılan çalışmaların tamamı (zaman ölçüm deneyi hariç) tablo 4.5'te gösterilmiştir.

Çizelge 4.5: ALGORITMA 3.1, ALGORITMA 3.2, ALGORITMA 3.3 VE ALGORITMA 3.4 SONUÇLARINA GÖRE KALITE SKORLARI.

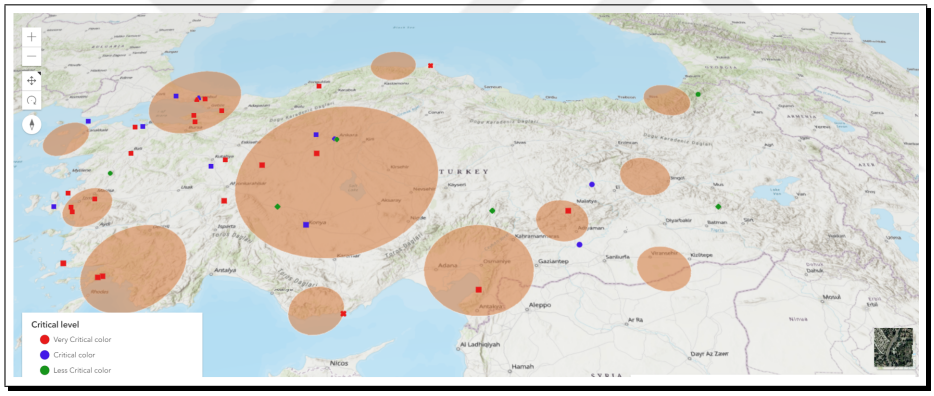
	VERI KÜMESI			
	DSSPARSE	DSSNORMAL	DSDENSE	STRA
Alg 3.1(r=30)	25	33	43	
Alg 3.1(r=40)	44	58	70	
Alg 3.1(r=50)	52	71	102	
Alg 3.1(Hisar-O)				1700
Alg 3.1(S-400)				4900
Alg 3.2(r=30)	18	29	36	
Alg 3.2(r=40)	26	46	58	
Alg 3.2(r=50)	36	62	76	
Alg 3.2(Hisar-O)				1700
Alg 3.2(S-400)				4800
Alg 3.3(r=30)	18	37	40	
Alg 3.3(r=40)	29	50	62	
Alg 3.3(r=50)	39	65	88	
Alg 3.3(Hisar-O)				1700
Alg 3.4(r=30)	15	23	29	
Alg 3.4(r=40)	24	37	39	
Alg 3.4(r=50)	29	55	64	
Alg 3.4(Hisar-O)				1300
Alg 3.4(S-400)				4300

4.3 Problem 2 Deneyi

Deneyde birden çok servis ile kapsanan varlık değerlerinin toplamına göre ölçülen kalite skorları karşılaştırılacaktır. Bu sebeple birden fazla savunma sistemi tanımlanacaktır. Kullanılacak savunma sistemleri S-400 (1 tane), Siper (3 tane) ve Hisar-O (8 tane) sistemleridir. Bu sistemlerin kapsadığı dairesel bölgelerin yarıçapı sırasıyla 200 km, 100 km ve 50 km'dir (tablo 4.6). Radarlar deney başında Türkiye Cumhuriyeti bölgesine rastgele olacak biçimde şekil 4.18'de konumlandırılmıştır .

Çizelge 4.6: RADARLAR TIPLERI VE KAPSAMA YARIÇAPLARI.

RADAR TIPI	KAPSAMA YARI ÇAPI(KM)
Hisar-O	50
Hisar-U	100
S-400	200



Şekil 4.18: Rastgele olarak yerleştirilen S400, Hisar-O ve Hisar-U radarları.

Deneylerde bölgesel savunma doktrini için `DSSPARSE`, `DSNORMAL` ve `DSDENSE` varlık kümesi ve stratejik savunma doktrini için `STRA` veri kümesi kullanılacaktır.

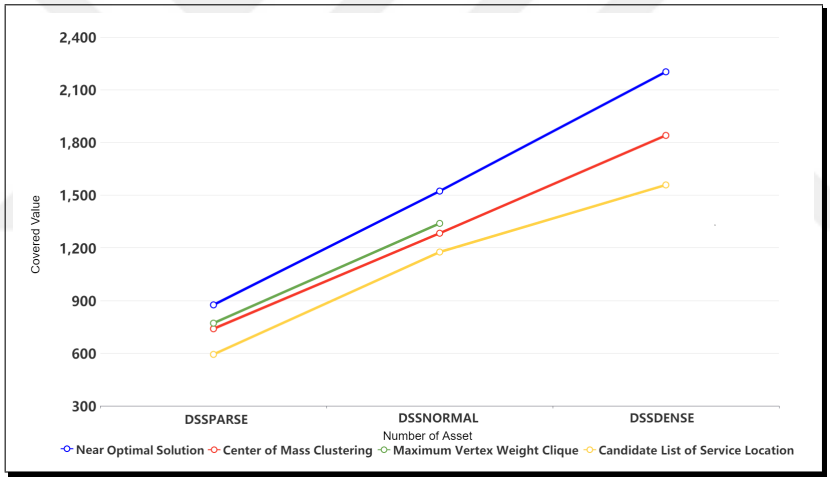
Problem 1'de kullanılan optimum çözüm bu deneyde servis sırasının seçiminden dolayı optimum sonuç veremeyeceği problem çözümünde belirtilmişti. Bu sebeple optimum çözüme yaklaşabilmek için iki farklı yöntem denenmiştir. Öncelikle büyük servisten başlayıp küçük servise olacak şekilde servisler seçilmiştir (azalan sırada). İkinci yöntemde ise öncelikle en küçük servisten başlanıp büyük servislere doğru gidilecektir (artan sırada). Deneylerde öncelikle bölgesel savunma doktrini daha sonra da stratejik bölge savunma doktrini çalışılacaktır.

Problem 1 deneyinin hazırlanışı kısmındaki diğer kabuller Problem 2 deneyinde de geçerlidir.

4.3.1 Deney 2 Sonucu

Deney sonuçlarına göre Optimuma Yakın Çözüm (algoritma 3.5) için servis seçiminde kapsama alanı büyük servislerden başlanması daha iyi sonuç vermiştir. Bunun yanında Problem 1 deney sonuçlarında olduğu gibi algoritma 3.5 yavaş çalışmaktadır. Diğer algoritmalardan beklenen algoritma 3.5'in (azalan sırada) sonuçlarına yakın ama daha hızlı cevaplardır.

Şekil 4.19'da verilen grafik, DSSPARSE, DSNORMAL ve DSDENSE olmak üzere üç farklı yoğunluktaki veri kümesinde bölgesel savunma doktrini için edilen kalite skorlarını karşılaştırır.



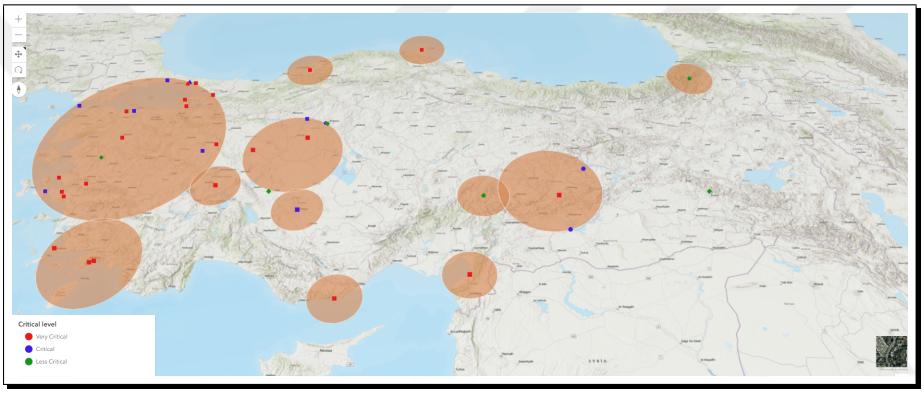
Şekil 4.19: Çoklu servis için deney sonuçları.

Sonuçlardan elde edilen ilk gözlem, deney 1'deki gibi veri kümesi ne kadar yoğunsa kalite skoru (yani, kapsanan varlık değerlerinin toplamı) doğrusal bir şekilde artmaktadır.

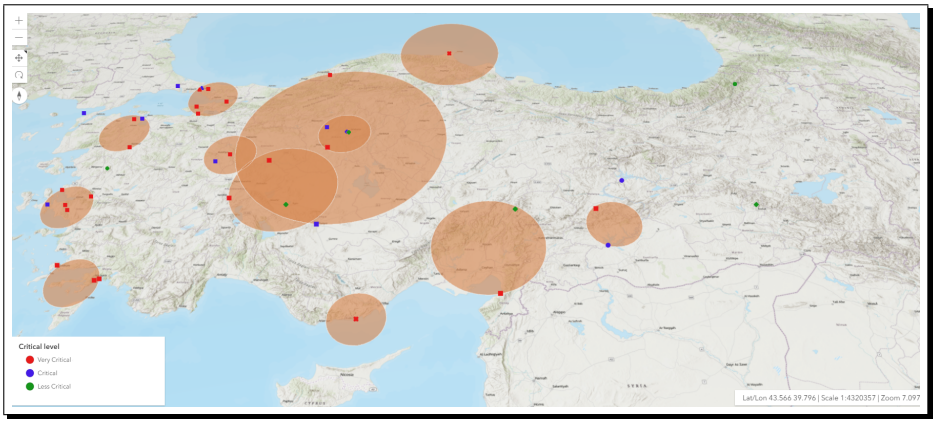
Algoritma 3.6, Algoritma 3.5'e yakın sonuçlar vermiştir. Algoritma 3.7, Algoritma 3.6'dan biraz daha iyi sonuç versede S-400 sisteminden dolayı `DSDENSE` veri kümesi deneyinde sonuç vermemiştir. Algoritma 3.8 ise beklendiği gibi en kötü sonucu vermiştir.

Deneylere stratejik bölge savunması için `STRA` ile devam edilmiştir. Deneyde kul-

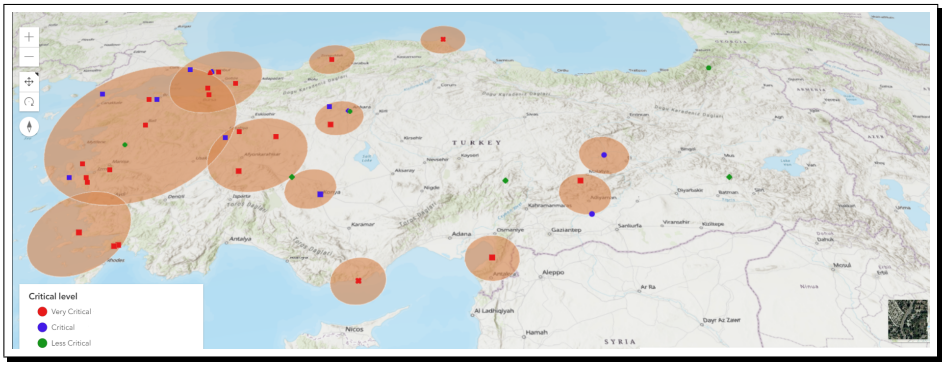
lanılan stratejik değerli varlıkların (40 tane) toplam maliyet değeri 9700'dür. Deneyler sonunda Algoritma 3.5 (azalan sırada) 9600 kalite skoru elde etmiştir. Algoritma 3.7, 9500 puanla algoritma 3.5'e en yakın sonuca ulaşmıştır. Bu sonucu 9400 kalite skoru puanıyla Algoritma 3.6 takip etmiştir. Algoritma 3.7 zaman ve bellek tüketimi konusunda Algoritma 3.6'ya göre daha kötü sonuç vermiştir. Algoritma 3.8 diğer iki algoritmaya göre çok hızlı çalışmasına rağmen kötü sonuçlar vermiş ve 6300 kalite skoru elde etmiştir. Daha öncede belirtildiği gibi aday noktalarının kısıtlı olmasında dolayı daha düşük kalite skoru elde edilmiştir. Aday noktaların artırılması bu durumu düzeltebilir, ancak optimum sonuç için önerilecek aday kümesini oluşturmak zor olacaktır. Hava savunma sistemlerinin aldığı son pozisyonlar şekil 4.20, 4.21, 4.22, 4.23 ve 4.24 ile gösterilmiştir.



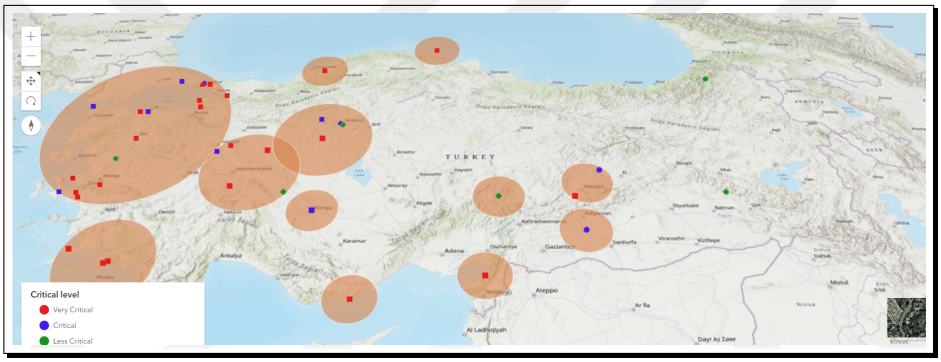
Şekil 4.20: Algoritma 3.5'in (önce büyük servis sıralı) kalite skoru 9600'dür.



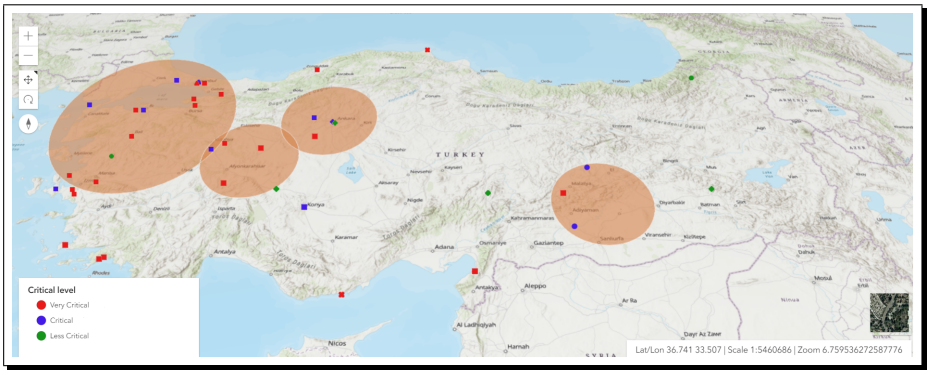
Şekil 4.21: Algoritma 3.5'in (önce küçük servis sıralı) kalite skoru 8800'dir.



Şekil 4.22: Algoritma 3.6'nın kalite skoru 9400'dür.



Şekil 4.23: Algoritma 3.7'nin kalite skoru 9500'dür.



Şekil 4.24: Algoritma 3.8'in kalite skoru 6300'dür.

Algoritmalar tarafından seçilen konumlara dikkat edilirse Türkiye'nin stratejik varlıklarının daha kıymetli ve yoğun olduğu bölgeler kapsanmıştır. Batıda stra-

tajik bölgeler daha çok olduğudan dolayı radar sistemleri ağırlıklı olarak batı bölgelerine dağılmıştır.

Algoritma 3.6 özellikle stratejik savunma doktrini deneylerinde hızlı ve başarılı sonuçlar elde etmiştir. Algoritma 3.5'e yakın kalite puanları elde ederken çok hızlı sonuçlar vermiştir.

Deney 2'de yapılan çalışmaların tamamı tablo 4.7'de gösterilmiştir.

Çizelge 4.7: ALGORITMA 3.5, 3.6, 3.7 VE 3.8 SONUÇLARINA GÖRE KALITE PUANLARI.

	VERI KÜMESI			
	DSS	DSN	DSD	STRA
Algoritma 3.5(Azalan sırada)	875	1523	2202	9600
Algoritma 3.5(Artan sırada)	818	1461	2122	8800
Algoritma 3.6	739	1283	1840	9400
Algoritma 3.7	771	1339	X	9500
Algoritma 3.8	593	1176	1558	6300

4.4 Problem 3 Deneyi

Bu deney ile matematiksel olarak optimizasyon yapılmaya çalışıldığı gibi askeri alanda ihtiyaç duyulabilecek kapsama için gerekli minimum maliyetli savunma sistemleri bulunmaya çalışılacaktır. Öncelikle algoritma 3.9, algoritma 3.10 ve algoritma 3.11 yöntemleri karşılaştırılacaktır. Daha sonra algoritma 3.10 ile farklı durumlar için çözümler araştırılacaktır.

Yapılacak deneylerde 2 farklı savunma doktrini karşılaştırılacaktır. Görece az değerli ama her yere yayılmış varlıklar için bölgesel savunma doktrini araştırması ve görece çok değerli ama sayıca az olan stratejik noktalar için stratejik savunma doktrini araştırması yapılacaktır. Bu doktrinleri karşılayacak 2 farklı varlık seti kullanılacaktır (`STRA`,`DSSPARSE`). `DSSPARSE` ile bölgesel savunma doktrinine göre envanter maliyeti araştırılırken, `STRA` ile stratejik nokta savunma doktrinine göre envanter maliyeti araştırılacaktır.

Deneylerde farklı servis tipi kümeleri denenip algoritmanın tercihleri gösterilecektir. Bu şekilde farklı durumlar için ihtiyaç duyulabilecek sistemler bulunmaya çalışılacaktır. Deneyler toplam envanter maliyetine göre karşılaştırılacaktır.

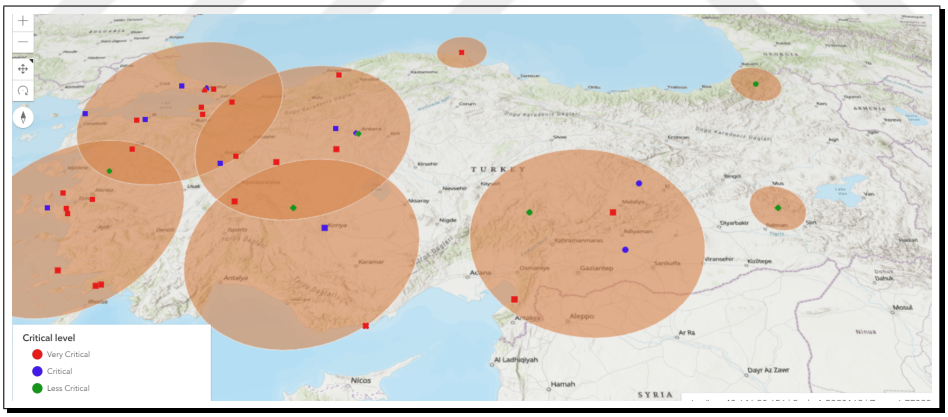
4.4.1 Deney 3 Sonucu

Bu deneyde tanımlanan algoritma 3.9, 3.10 ve 3.11 karşılaştırılacaktır. Veri kümesi olarak stratejik varlıkları içeren STRA kümesi kullanılacaktır. Envantere seçilebilecek **S** tipi kümesi 3 farklı radar servisinden oluşacaktır. Bu servisler Hisar-O, Hisar-U ve S-400 hava savunma sistemleridir. Servislerin kapsama yarıçapları sırasıyla 50, 100 ve 200 km'dir. Servislerin yerleştirilme maliyetleri sırasıyla 80, 120 ve 300 birim olarak seçilmiştir. Tablo 4.8'de servis tipleri ve özellikleri gösterilmiştir.

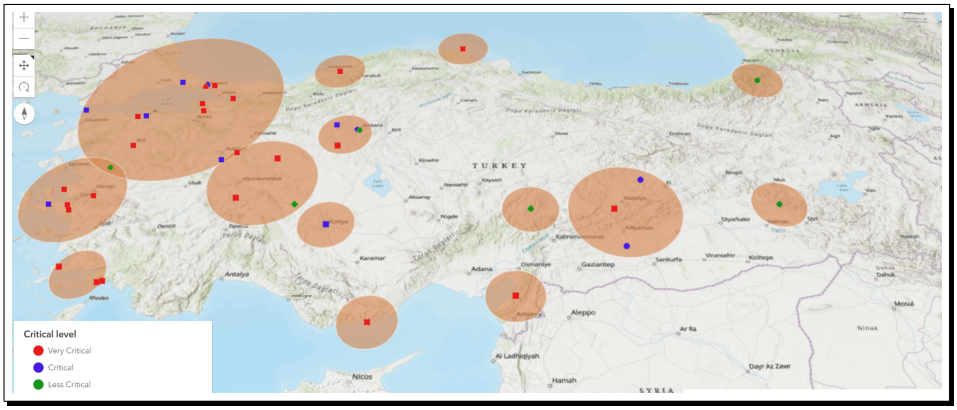
Çizelge 4.8: RADARLAR, KAPSAMA YARIÇAPLARI VE MALIYETLERI.

RADAR TIPI	KAPSAMA YARI ÇAPI(KM)	YERLEŞTIRME MALIYETI
Hisar-O	50	80
Hisar-U	100	120
S-400	200	300

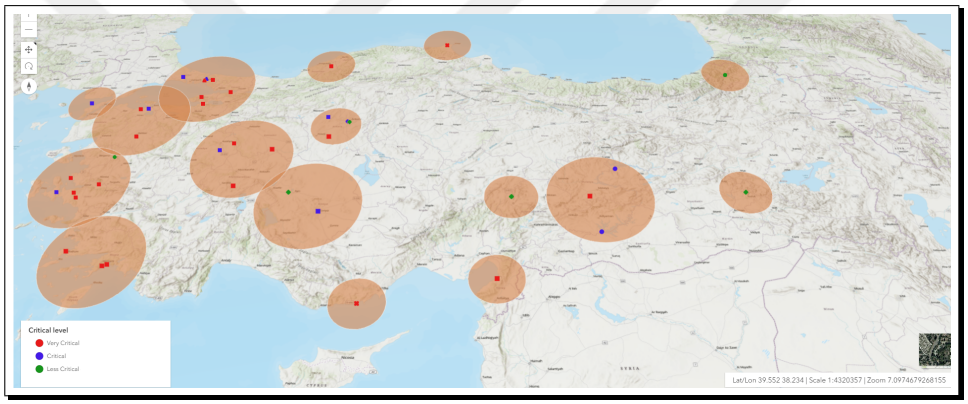
Problem çözümünde tanımlanan algoritmalar ayrı ayrı çalıştırıldıktan sonra algoritmaların seçtiği hava savunma sistemleri ve sistemlerin konumları şekil 4.25, 4.26 ve 4.27'de gösterilmiştir.



Şekil 4.25: Algoritma 3.9'un envanter maliyeti 1740 olmuştur.



Şekil 4.26: Algoritma 3.10'un envanter maliyeti 1460 olmuştur.



Şekil 4.27: Algoritma 3.11'in envanter maliyeti 1560 olmuştur.

Algoritma 3.10, algoritma 3.9'ye göre çoğu bölgede özyinelemeli yaklaşım sayesinde daha az maliyetli servisleri seçebilmiştir ve toplamda daha az maliyetle stratejik varlıkları kapsayabilmiştir. Algoritma 3.10'da S-400 sistemine sadece Marmama Bölgesi'nde ihtiyaç duyulmuştur. Bunun sebebi stratejik olarak yoğun bir bölge olmasından dolayı S-400 maliyetinin, matematiksel olarak optimum sonuç vermesidir. Algoritma 3.11 toplam maliyet olarak, algoritma 3.10'den biraz daha kötü bir sonuç vermiştir. Ayrıca 2 adet Hisar-O servisinin toplam maliyetinin S-400 sisteminden düşük olmasından dolayı envantere S-400 sistemi eklenememiştir. Bu durum algoritmanın birbirinden çok farklı maliyete sahip servisler için kullanıldığında optimum sonuç veremeyeceğinin bir göstergesidir.

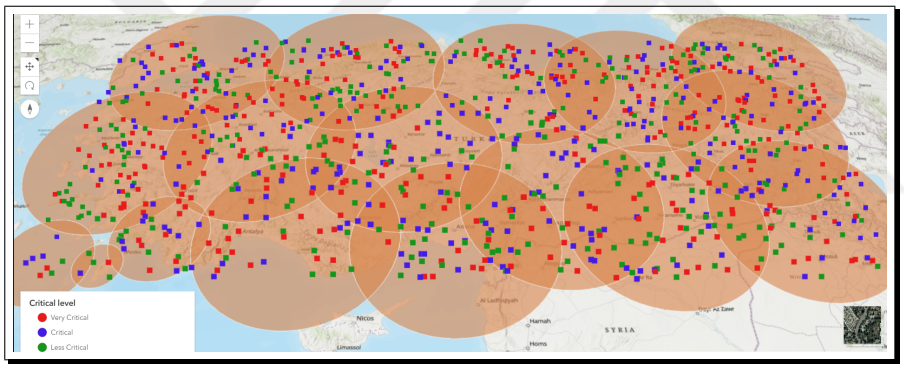
Algoritma 3.9 ve 3.10 harcadıkları zamana göre karşılaştırıldıkları zaman yakın sonuçlar verdikleri görülmüştür. Algoritma 3.9, Algoritma 3.10'a göre biraz daha

hızlı çalışır. Algoritma 3.11 ise diğer iki algoritmaya göre daha yavaş çalışmaktadır. Algoritmaların envanteri oluştururken kullandıkları hava savunma sistemleri ve harcadıkları ortalama süreler tablo 4.9'da gösterilmiştir.

Çizelge 4.9: MALIYET VE SÜRE KARŞILAŞTIRMALARI.

ALGORITMA ADI	HISAR-O	HISAR-U	S-400	MALIYET	SÜRE(MS)
Algoritma 3.9	3	0	5	1740	57
Algoritma 3.10	10	3	1	1460	67
Algoritma 3.11	9	7	0	1560	290

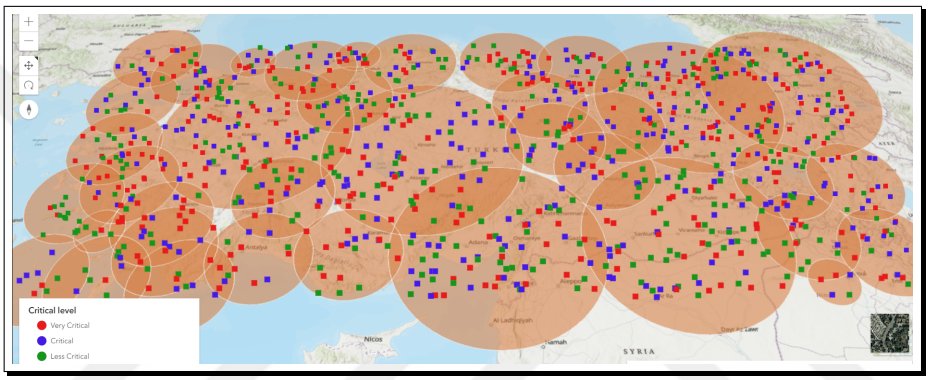
Bir sonraki deneyde `DSSPARSE` veri seti ile bölgesel savunma amaçlı deney yapılacaktır. Bu veri kümesi deney alanına homojen yayılmış varlıklardan oluşmaktadır. Böylece tüm bölge kapsanmak istenirse, algoritma 3.10'un boşluk bırakmadan vereceği çözüm bulunacaktır. Şekil 4.28'de sonuç gözükmektedir. Bundan sonraki deneylerde algoritma 3.10 kullanılacaktır.



Şekil 4.28: Envanter maliyeti 4520 olmuştur. Toplam 17 savunma sistemi kullanılmıştır (14 S-400, 2 Hisar-U, 1 Hisar-O).

Algoritma 3.10, bölgesel savunma yapıldığı zaman daha maliyetli ve kapsama alanı büyük servis seçiminin, optimum sonuç vereceğini göstermiştir. Bu beklenen bir sonuçtur. Bölgesel bir kapsama istendiği için pahalı ama büyük alanları kapsayacak servis seçimi toplam maliyeti düşürmektedir.

Problem 3 deneylerinde şu ana kadar 3 (S-400, Hisar-U, Hisar-O) farklı hava savunma sistemi simüle edilmiştir. Tüm Türkiye bölgesi kapsanmak istendiği zaman ağırlıklı olarak yabancı menşeili S-400 sisteminin tercih edildiği görülmüştür. Bir sonraki deneyde yerli sistemlerin (Hisar-U, Hisar-O) maliyetleri (80, 40) yarı yarıya düşürülmüştür. Şekil 4.29'da sonuçlar gösterilmiştir.



Şekil 4.29: Envanter maliyeti 4120 olmuştur. Toplam 37 savunma sistemi kullanılmıştır (6 S-400, 27 Hisar-U, 4 Hisar-O).

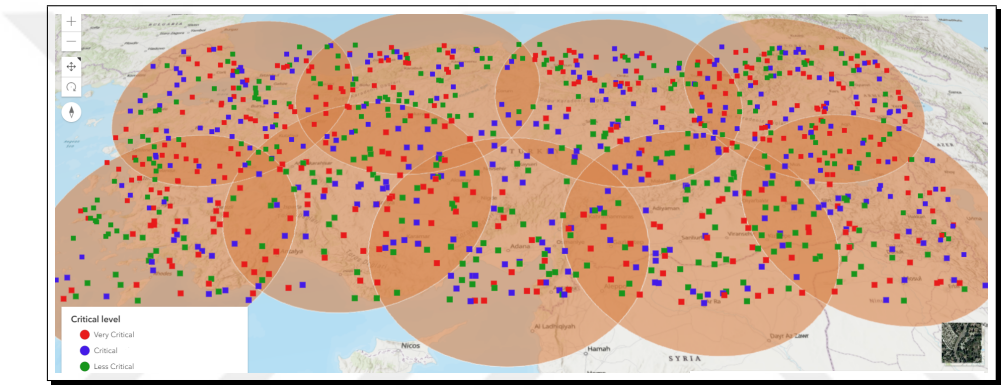
Algoritma 3.10, maliyetlerin düşmesiyle yerli savunma sistemlerini tercih etmeye başlamıştır. Bu deney bölgesel savunmada servis maliyetinin etkisini göstermektedir. Küçük yarıçapa sahip servisler maliyetlerinin düşmesi sonucunda bölgesel savunmada tercih sebebi olmaktadır. Sonuç olarak toplam maliyet önceki deneye göre azalmıştır.

Bir sonraki deneyde servis tipi kümesi genişletilecektir. Deney 3'te kullanılan savunma sistemlerine 2 yeni savunma sistemi (S-400, Hisar-A) daha eklenerek bölgesel ve stratejik savunmadaki sonuçlar değerlendirilecektir. Tablo 4.10'da radarların yarı çapları ve yerleştirme maliyetleri gösterilmiştir

Çizelge 4.10: RADARLAR, KAPSAMA YARIÇAPLAR VE MALIYETLERI.

RADAR TIPI	KAPSAMA YARI ÇAPI(KM)	YERLEŞTIRME MALIYETI
Hisar-A	25	40
Hisar-O	50	80
Hisar-U	100	120
S-400	200	300
S-500	250	400

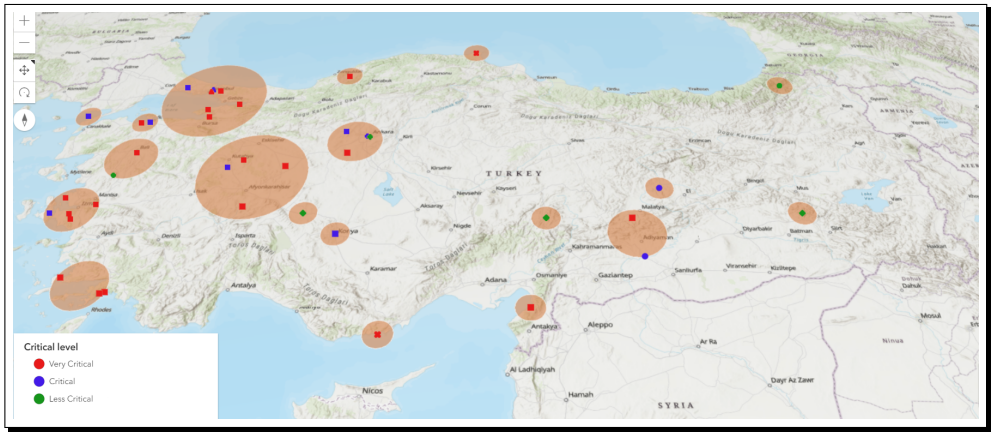
Eklenen yeni servis tipleri ile Algoritma 3.10 tekrar çalıştırılmıştır. Şekil 4.30'da sonuçlar gösterilmiştir.



Şekil 4.30: Envanter maliyeti 3600 olmuştur. Toplam 9 savunma sistemi kullanılmıştır (9 S-500).

S-500 sistemi kapsadığı bölgeye göre oransal olarak S-400 sisteminden daha maliyetli olsa da matematiksel olarak optimum sonuç vermesinden dolayı bölgesel savunmada tercih edilmiştir. Bu değerlere sahip bir servisin gelmesi yeni eklenen Hisar-A sistemi dahil diğer servislerin kullanımını engellemiştir. S-500 ile şekil 4.28'deki deneye göre daha düşük bir maliyet elde edilmiştir.

Bölgesel savunma yerine, stratejik nokta savunması yapıldığı zaman algoritma 3.10'un servis tercihi ağırlıklı olarak Hisar-A olmuştur. Bu beklenen bir sonuçtur, çünkü stratejik bölgelerin birçoğu birbirinden uzaktır. Bu sebeple büyük servis kullanılması matematiksel olarak optimum sonuç vermemiştir. Şekil 4.31 stratejik bölgelerdeki servis dağılımını ve toplam maliyeti göstermektedir.



Şekil 4.31: Envanter maliyeti 1120 olmuştur. Toplam 19 savunma sistemi kullanılmıştır (2 Hisar-U, 5 Hisar-O, 12 Hisar-A).

Savunma doktrinine göre ihtiyaç duyulan servis tipinin değiştiği gözlemlenmiştir. Bölgesel savunmada büyük ama maliyetli savunma sistemleri optimum sonuç verirken stratejik savunmada daha küçük sistemler etkili sonuç vermiştir. Bu deney, savunma doktrinine göre hava savunma sistemi ihtiyaçlarının farklılaştığını göstermektedir. Burada tabii ki istenilen bölgesel savunma olacaktır ama maliyetlerden dolayı stratejik bölge savunmasına ihtiyaç duyulabilir. Algoritma 3.10, ihtiyaca yönelik doğru sonuçlar vermektedir.

5. SONUÇ VE ÖNERİLER

Bu çalışmada rastgele bir şekilde dağılmış birbirinden farklı değerlere sahip noktaların maliyet etkin bir şekilde dairelerle kapsanması problemi özelleştirilmiştir. Bu bağlamda coğrafi referanslı varlıkların, coğrafi referanslı servislerle optimum şekilde kapsanmasına çalışılmıştır. Coğrafi-referanslı varlıkları kapsamak üzerine üç farklı problem tanımlanmış ve çözüm algoritmaları önerilmiştir. Yapılan çözümlerle komuta kontrol sistemlerine katkı verilmesi amaçlanmıştır.

Bu kapsamında Türkiye coğrafyası için çeşitli tiplerde varlık kümeleri oluşturulmuştur. Farklı yoğunlukta tüm coğrafyaya dağılmış rastgele verilerin yanında, coğrafi olarak stratejik noktalar tanımlanmış ve ayrı bir veri kümesi daha oluşturulmuştur. Varlıklar tanımlandıktan sonra çeşitli senaryolar için hava savunma sistemlerini temsil edecek maliyetleri ve kapsama alanları birbirlerinden farklı servisler tanımlanmıştır.

Deneysel değerlendirmeler ile her bir problem için algoritmaların etkinliği ölçülmüş ve karşılaştırılmıştır. Algoritmaların farklı yoğunluklardaki varlık kümelerinde aldığı sonuçların karşılaştırılması yapılmıştır. Mevcut hava savunma sistemlerinin optimum şekilde dağıtılması için gerçekleştirilen çözümler kalitatif ve kuantitatif olarak incelendiğinde servislerin ağırlıklı olarak ülkemizin stratejik bölgelerine dağıtıldığı görülmüştür.

Öncelikle tek bir servisin optimum şekilde konumlandırılması için çalışma yapılmıştır. Algoritmaların rastgele dağıtılmış farklı yoğunluklardaki veri kümelerinde, farklı servis tipleriyle aldıkları sonuçlar karşılaştırılmıştır. Stratejik varlıklar ile yapılan deneylerde marmara bölgesine öncelik verildiği görülmüştür. Bunların yanında algoritmalar harcadıkları sürelere göre de karşılaştırılmıştır. Bu algoritma ile görece olarak küçük birliklerin tek bir sisteme sahip oldukları durumlar düşünülmüş ve bu birliklerin komuta kontrol sistemlerine katkı verilmeye çalışılmıştır.

Daha sonra mevcut savunma sistemleriyle varlıkların optimum şekilde kapsanması için çalışma yapılmıştır. Bunun için servis kümesi tanımlanmış ve bu servisler ile algoritmalar farklı yoğunluktaki varlık kümelerinde karşılaştırılmıştır. Stratejik varlıklar ile yapılan deneylerde simule edilen hava savunma sistemlerinin dağılımları gösterilmiştir. Bu çalışma ile mevcut hava savunma sistemlerinin komuta kontrol sistemleriyle optimum şekilde nasıl dağıtılacağına karar verilmeye çalışılmıştır.

Son olarak rastgele dağıtılmış varlıklar ile bölgesel savunma doktrini ve stratejik noktalar ile stratejik bölge savunması doktrini tanımlanmış ve iki farklı savunma doktrini için gerekli olacak savunma sistemleri bulunmaya çalışılmıştır. Deney-

lerde farklı algoritmalar ile maliyet ve süre karşılaştırmaları yapılmıştır. Daha iyi sonuç veren algoritma ile servis maliyetlerinin hava savunma sistemi seçimine etkisi gösterilmiştir. Bölgesel savunma veya stratejik bölge önceliğine göre farklı savunma sistemlerinin seçilmesi gerektiği görülmüştür. Bu çalışma ile komuta kontrol sistemleri savunma doktrinine göre ihtiyaçları belirleyebilecektir. Böylece maliyetten kazanç sağlanabilecektir.

Yapılan çalışmaların tamamı ile komuta kontrol yazılımlarına servis konumlandırma ve gerekli duyulan servisleri hesaplama konularında katkı sağlanabilir. Böylece komuta kontrol yazılımları, hava savunma sistemlerini savunma doktrinine göre maliyet etkin bir şekilde konumlandırabilir. Örnek bir senaryoda, ihtiyaç duyulan hava savunma sistemleri bulunabilir ve temin edilen sistemler optimum şekilde yerleştirilebilir. Burada gerekli tüm sistemler temin edilemeyebilir. Bu durumda da servisler en optimum şekilde dağıtılacaktır.

Deneylerde kullanılan uygulama ile hava savunma sistemlerinin ihtiyaca göre konumlandırılması görselleştirilmiştir. Hızlı çalışan kullanıcı dostu bir uygulama gerçekleştirilmiştir. Uygulama geliştirmeye açık bir uygulamadır. İstendiği takdirde farklı problem kümeleri için de kullanılabilir.

Bu tezde katmanlı hava savunma mimarisi göz önüne alınmamıştır. Gelecekte yapılacak çalışmalarda farklı senaryolar için katmanlı hava savunma sistemi ihtiyaçları göz önüne alınabilir. Böylece askeri ihtiyaçlara daha iyi cevap verebilecek bir sistem ortaya konabilir.

Kaynaklar

[1] **SIBSON, ROBIN**. Slink: an optimally efficient algorithm for the single-link cluster method. *The computer journal 16*, 1 (1973), 30–34.

[2] **AHMET YILMAZ**. https://www.enerjiatlasi.com/elektrik-uretimi/.

[3] **AL ABID, FAISAL BIN**. A novel approach for pam clustering method. *International Journal of Computer Applications 86*, 17 (2014).

[4] **ALFATTANI, SAFWAN AND JAAFAR, WAEL AND YANIKOMEROGLU, HALIM AND YONGACOGLU, ABBAS**. Multi-uav data collection framework for wireless sensor networks. In *2019 IEEE Global Communications Conference (GLOBECOM)* (2019), IEEE, pp. 1–6.

[5] **ÄYRÄMÖ, SAMI AND KÄRKKÄINEN, TOMMI**. Introduction to partitioning-based clustering methods with a robust example. *Reports of the Department of Mathematical Information Technology. Series C, Software engineering and computational intelligence*, 1/2006 (2006).

[6] **CAMPELLO, RICARDO JGB AND MOULAVI, DAVOUD AND SANDER, JÖRG**. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining* (2013), Springer, pp. 160–172.

[7] **CHENG, YIZONG**. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence 17*, 8 (1995), 790–799.

[8] **DAWYNDT, PETER AND DE MEYER, HANS AND DE BAETS, BERNARD**. The complete linkage clustering algorithm revisited. *Soft Computing 9*, 5 (2005), 385–392.

[9] **DOERRY, ARMIN W**. Earth curvature and atmospheric refraction effects on radar signal propagation. *Sandia Report SAND2012-10690* (2013).

[10] **DOUGHERTY, JAMES AND KOHAVI, RON AND SAHAMI, MEHRAN**. Supervised and unsupervised discretization of continuous features. In *Machine learning proceedings 1995*. Elsevier, 1995, pp. 194–202.

[11] **ELZINGA, D JACK AND HEARN, DONALD W**. The minimum covering sphere problem. *Management science 19*, 1 (1972), 96–104.

[12] **Ester, Martin and Kriegel, Hans-Peter and Sander, Jörg and Xu, Xiaowei and others**. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231.

[13] **Estivill-Castro, Vladimir**. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter 4*, 1 (2002), 65–75.

[14] **Ghahramani, Zoubin**. Unsupervised learning. In *Summer School on Machine Learning* (2003), Springer, pp. 72–112.

[15] **Grira, Nizar and Crucianu, Michel and Boujemaa, Nozha**. Unsupervised and semi-supervised clustering: a brief survey. *A review of machine learning techniques for processing multimedia content 1* (2004), 9–16.

[16] **Guha, Sudipto and Rastogi, Rajeev and Shim, Kyuseok**. Cure: an efficient clustering algorithm for large databases. *ACM Sigmod record 27*, 2 (1998), 73–84.

[17] **Han, Jiawei and Kamber, Micheline and Tung, Anthony KH**. Spatial clustering methods in data mining. *Geographic data mining and knowledge discovery* (2001), 188–217.

[18] **Karypis, George and Han, Eui-Hong and Kumar, Vipin**. Chameleon: Hierarchical clustering using dynamic modeling. *Computer 32*, 8 (1999), 68–75.

[19] **MacQueen, James and others**. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (1967), vol. 1, Oakland, CA, USA, pp. 281–297.

[20] **Maddox, Robert A and Zhang, Jian and Gourley, Jonathan J and Howard, Kenneth W**. Weather radar coverage over the contiguous united states. *Weather and forecasting 17*, 4 (2002), 927–934.

[21] **Mayor, Vicente and Estepa, Rafael and Estepa, Antonio and Madinabeitia, German**. Deploying a reliable uav-aided communication service in disaster areas. *Wireless Communications and Mobile Computing 2019* (2019).

[22] **Mehmet Arda Mevlütoğlu**. “silahların İnterneti”: "jadc2 ve abms".

[23] **MOSELEY, BENJAMIN AND WANG, JOSHUA**. Approximation bounds for hierarchical clustering: Average linkage, bisecting k-means, and local search. In *Advances in Neural Information Processing Systems* (2017), pp. 3094–3103.

[24] **MURTAGH, FIONN**. A survey of recent advances in hierarchical clustering algorithms. *The computer journal 26*, 4 (1983), 354–359.

[25] **MURTAGH, FIONN AND LEGENDRE, PIERRE**. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification 31*, 3 (2014), 274–295.

[26] **NISHOM, M**. Perbandingan akurasi euclidean distance, minkowski distance, dan manhattan distance pada algoritma k-means clustering berbasis chi-square. *J. Inform 4*, 01 (2019).

[27] **RAJAN, ISSAAC AND ARAVAMUTHAN, SARANG AND MANDE, SHARMILA S**. Identification of compositionally distinct regions in genomes using the centroid method. *Bioinformatics 23*, 20 (2007), 2672–2677.

[28] **RAMAZAN SAYGILI**. http://cografyaharita.com/.

[29] **RANI[1], YOGITA AND ROHIL, HARISH**. A study of hierarchical clustering algorithm. *ter S & on Te SIT 2* (2013), 113.

[30] **ROUSSEEUW, PETER J AND KAUFMAN, L**. Finding groups in data. *Hoboken: Wiley Online Library* (1990).

[31] **SAWALMEH, AHMAD AND OTHMAN, NOOR SHAMSIAH AND SHAKHATREH, HAZIM AND KHREISHAH, ABDALLAH**. Providing wireless coverage in massively crowded events using uavs. In *2017 IEEE 13th Malaysia International Conference on Communications (MICC)* (2017), IEEE, pp. 158–163.

[32] **SCULLEY, DAVID**. Web-scale k-means clustering.

[33] **SEFA ÖZKAYA**. *Türk Askeri Kültürü*. Kronik Kitap, 2019.

[34] **SHIMANSKI, CHARLEY**. Situational awareness in search and rescue operations. In *International Technical Rescue Symposium* (2005).

[35] **SILAHTAROĞLU, GÖKHAN**. Veri madenciliği. *Papatya Yayınları, İstanbul* (2008).

[36] **SKOLNIK, MERRILL I**. Introduction to radar. *Radar handbook 2* (1962), 21.

[37] **STREEFKERK, JAN WILLEM AND SMETS, NANJA AND VARKEVISSER, MICHEL AND MASTRIGT, SUZANNE HIEMSTRA-VAN**. Future command and control systems should combine decision support and personalization interface features. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational* (2014), pp. 266–275.

[38] **SU, TING AND DY, JENNIFER G**. In search of deterministic methods for initializing k-means and gaussian mixture clustering. *Intelligent Data Analysis 11*, 4 (2007), 319–338.

[39] **TAN, PANG-NING AND STEINBACH, MICHAEL AND KUMAR, VIPIN**. Classification: basic concepts, decision trees, and model evaluation. *Introduction to data mining 1* (2006), 145–205.

[40] **TIAN, JINGWEN AND GAO, MEIJUAN AND GE, GUANGSHUANG**. Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm. *EURASIP Journal on Wireless Communications and Networking 2016*, 1 (2016), 1–11.

[41] **TZU, SUN**. *Sun Tzu Art of War*. Vij Books India Pvt Ltd, 2012.

[42] **UYSAL, DOĞAN AND YILMAZ, KUBILAY AND TANER, TAŞ**. Enerji ithalatı ve cari açık ilişkisi: Türkiye örneği. *Anemon Muş Alparslan Üniversitesi Sosyal Bilimler Dergisi 3*, 1 (2015), 63–78.

[43] **WELZL, EMO**. Smallest enclosing disks (balls and ellipsoids). 359–370.

[44] **WOLFF, CHRISTIAN**. Radartutorial. eu. *Retrieved May 3* (2012), 2013.