
GAZİ ÜNİVERSİTESİ
Gazi
1926

OTOMOTİV SEKTÖRÜNDE PARÇA LOJİSTİĞİ İÇİN BİR KARAR DESTEK SİSTEMİ ÖNERİSİ

Mevlüt UYSAL

DOKTORA TEZİ
YÖNETİM BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

OCAK 2020

Mevlüt UYSAL tarafından hazırlanan "OTOMOTİV SEKTÖRÜNDE PARÇA LOJİSTİĞİ İÇİN BİR KARAR DESTEK SİSTEMİ ÖNERİSİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Yönetim Bilişim Sistemleri Ana Bilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. Uğur ÖZCAN
Endüstri Mühendisliği, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

[redacted]

Başkan: Prof. Dr. M. Duran TOKSARI
Endüstri Mühendisliği, Erciyes Üniversitesi
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

[redacted]

Üye: Doç. Dr. Selçuk Kürşat İŞLEYEN
Endüstri Mühendisliği, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

[redacted]

Üye: Doç. Dr. Diyar AKAY
Endüstri Mühendisliği, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

[redacted]

Üye: Doç. Dr. Harun ARTUNER
Bilgisayar Mühendisliği, Hacettepe Üniversitesi
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

[redacted]

Tez Savunma Tarihi: 14/01/2020

Jüri tarafından kabul edilen bu tezin Doktora Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

[redacted]

Doç. Dr. Aslıhan TÜFEKCİ
Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Mevlüt UYSAL
14/01/2020

OTOMOTİV SEKTÖRÜNDE PARÇA LOJİSTİĞİ İÇİN BİR KARAR DESTEK SİSTEMİ ÖNERİSİ

(Doktora Tezi)

Mevlüt UYSAL

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Ocak 2020

ÖZET

Modern üretim sistemlerinde, sürekli artan ürün çeşitliliği, karışık modelli montaj hatlarını gerekli parçalarla beslemek için kullanılan iç lojistik sistemleri açısından büyük bir zorluk teşkil etmektedir. Montaj hatlarına parça akışının hiç durmaması için binlerce parça ve tedarikçi, çok sayıda donanım ve yüzlerce montaj işçisi koordine edilmelidir. Son yıllarda, özellikle otomotiv endüstrisinden birçok üretici, tam zamanında üretim felsefesine göre esnek ve güvenilir parça tedariki sağlamak için süpermarket konseptini benimsemiştir. Bu bağlamda süpermarket, yakınındaki montaj hattının parça talebini karşılayabilmek için kurulan, merkezi olmayan ara bir saklama alanıdır. Süpermarketler merkezi depodan montaj hatlarının ihtiyaç duyduğu farklı büyüklükteki montaj bileşenleri ile doldurulur ve küçük kutular halinde çekici trenler aracılığıyla servis edilir. Bu çalışma kapsamında işletme zeminine kurulacak süpermarketlerin yerleşim problemi ele alınmıştır. Yerleşim problemi kurulan her bir süpermarketin getirdiği sabit maliyet ile süpermarketlerden montaj hattına parça taşıma maliyetinin toplamı en aza indirgenecek şekilde, süpermarketlerin sayısının ve yerlerinin belirlenmesi problemidir. Daha önceki çalışmalarda süpermarketlerin kapasiteleri göz ardı edilmiş ya da sabit kapasiteli süpermarketler problem çözümünde kullanılmıştır. Bu çalışmada problem çözümünde farklı kapasiteli süpermarketler dikkate alınmıştır. Diğer çalışmalarda problem çözümünde popülasyon tabanlı algoritmalar kullanılmıştır. Bu çalışmada tek çözüm tabanlı bir metasezgisel algoritma olan tavlama benzetimi algoritması geliştirilmiş ve bir dizi test problemi ile test edilmiştir. Testler sonucunda, algoritmanın başarılı sonuçlar verdiği görülmüştür. Önerilen algoritma kullanılarak bir karar destek sistemi geliştirilmiştir. Geliştirilen karar destek sistemi ile özellikle otomotiv endüstrisinde fabrika zeminine süpermarket kurmak isteyen yöneticilerin karar vermelerine yardımcı olmak amaçlanmıştır.

Bilim Kodu	:	114606
Anahtar Kelimeler	:	Parça lojistiği, süpermarket, tavlama benzetimi, karar destek sistemleri
Sayfa Adedi	:	103
Danışman	:	Doç. Dr. Uğur ÖZCAN

A DECISION SUPPORT SYSTEM PROPOSAL FOR PART LOGISTICS IN THE AUTOMOTIVE SECTOR

(Ph. D. Thesis)

Mevlüt UYSAL

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

January 2020

ABSTRACT

In modern production systems, the ever-increasing product range is a major challenge for internal logistics systems to feed mixed-model assembly lines with the necessary components. Thousands of parts and suppliers, a large number of equipment and hundreds of assembly workers should be coordinated to ensure that parts flow to assembly lines never stop. In recent years, many manufacturers, especially from the automotive industry, have adopted the supermarket concept to provide flexible and reliable part supply according to the philosophy of just-in-time production. In this context, supermarket is a decentralized intermediate storage area established to meet the parts demand of the assembly line. Supermarkets are replenished from the central warehouse with assembly components of different sizes required by assembly lines and served in small bins by tow trains. In this study, the location problem of the supermarkets to be established on the factory floor is discussed. The location problem is the problem of determining the number and location of the supermarkets in such a way that the sum of the fixed cost of each supermarket with the cost of carrying parts from the supermarkets to the assembly line is minimized. In previous studies, the capacity of supermarkets was ignored or fixed capacity supermarkets were used. In this study, different capacity of supermarkets were considered. In other studies, population based algorithms are used for the problem. In this study, a simulated annealing algorithm, which is a single solution based metaheuristic has been developed and tested with a set of test problems. As a result of the tests, the algorithm was found to be successful. A decision support system was developed using the proposed algorithm. With the developed decision support system, especially in the automotive industry, it is aimed to help the managers who want to establish supermarkets on the factory floor.

Science Code	:	114606
Key Words	:	Part logistics, supermarket, simulated annealing, decision support systems
Page Number	:	103
Supervisor	:	Assoc. Prof. Dr. Uğur ÖZCAN

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren, ilgi ve muhabbetini esirgemeyen ve beni hiçbir zaman geri çevirmeyen kıymetli hocam Doç. Dr. Uğur ÖZCAN'a teşekkür ederim. Her zaman olumlu tavır ve anlayışlarıyla bana yol gösteren değerli hocalarım Doç. Dr. Selçuk Kürşat İŞLEYEN ve Prof. Dr. Duran TOKSARI'ya teşekkür ederim. Sabır ve anlayışla hep yanımda olan sevgili eşim Yeşim'e ve hayatımın her aşamasında beni destekleyen tüm ailem ve dostlarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET iv

ABSTRACT v

TEŞEKKÜR vi

İÇİNDEKİLER vii

ÇİZELGELERİN LİSTESİ x

ŞEKİLLERİN LİSTESİ xi

RESİMLERİN LİSTESİ xii

SİMGELER VE KISALTMALAR xiii

1. GİRİŞ 1

2. KAVRAMSAL ÇERÇEVE VE LİTERATÜR ÖZETİ 5

2.1. Tam Zamanında Üretim Sistemi 5

2.2. Tam Zamanında Üretimde Süpermarket Kullanımı 6

2.2.1. Yerleşim problemi 10

2.2.2. Rotalama ve çizelgeleme problemi 11

2.2.3. Yükleme problemi 13

2.3. Literatür Özeti 14

2.4. Karar Destek Sistemleri (KDS) 21

2.4.1. Karar destek sistemi kavramı 21

2.4.2. Karar destek sistemi bileşenleri 22

2.4.3. Karar destek sistemi türleri 24

2.4.4. Model sürümlü KDS 26

2.4.5. Optimizasyona dayalı KDS 27

2.4.6. Optimizasyonda kullanılan metasezgisel yöntemler 28

Sayfa

2.5. KDS ile İlgili Çalışmalar 33

3. KAPASİTESİZ SÜPERMARKET YERLEŞİM PROBLEMİ 37

3.1. Problem Tanımı 37

3.2. Kapasitesiz Süpermarket Yerleşim Problemi İçin Tavlama Benzetimi Yaklaşımı 40

3.2.1. Başlangıç çözümü 43

3.2.2. Komşu çözüm oluşturulması 44

3.2.3. Örnek problem 46

3.3 Deneysel Çalışma 48

4. DEĞİŞKEN KAPASİTELİ SÜPERMARKET YERLEŞİM PROBLEMİ 53

4.1. Problem Tanımı 53

4.2. Değişken Kapasiteli Süpermarket Yerleşim Problemi İçin Rastsal Arama Yaklaşımı 54

4.2.1. Deneysel çalışma 55

4.3. Değişken Kapasiteli Süpermarket Yerleşim Problemi İçin Tavlama Benzetimi Yaklaşımı 58

4.3.1. Başlangıç çözümü 58

4.3.2. Komşu çözüm oluşturulması 59

4.3.3. Örnek problem 61

4.3.4. Deneysel çalışma 63

5. SÜPERMARKET YERLEŞİM PROBLEMİ İÇİN BİR KARAR DESTEK SİSTEMİ 67

5.1. Sistemin Teknik Altyapısı 67

5.2. Sistemin Arayüzü 68

5.3. Örnek Uygulama 72

Sayfa

6. SONUÇ VE ÖNERİLER 81

KAYNAKLAR 85

EKLER..... 91

EK-1. Kapasitesiz SYP parametre optimizasyonuna ilişkin analiz sonuçları..... 93

EK-2. Değişken Kapasiteli SYP parametre optimizasyonuna ilişkin analiz sonuçları... 97

ÖZGEÇMİŞ 101

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge 2.1. İlgili çalışmalar 15

Çizelge 3.1. Matematiksel modelde kullanılan kısaltmaların gösterimi 39

Çizelge 3.2. Kapasitesiz SYP için örnek veri 47

Çizelge 3.3. Kapasitesiz SYP için önerilen TB algoritması uygulamasında kullanılan kısaltmaların gösterimi 49

Çizelge 3.4. Kapasitesiz SYP için önerilen TB algoritması uygulama sonuçları 50

Çizelge 4.1. Değişken kapasiteli SYP için önerilen matematiksel modelde kullanılan kısaltmaların gösterimi 54

Çizelge 4.2. RA uygulamasında kullanılan kısaltmaların gösterimi 56

Çizelge 4.3. RA uygulama sonuçları 57

Çizelge 4.4. Değişken kapasiteli SYP için örnek veri 62

Çizelge 4.5. TB ve RA algoritmaları karşılaştırmasında kullanılan kısaltmaların gösterimi 63

Çizelge 4.6. TB ve RA algoritmaları uygulama sonuçları 64

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Şematik bir süpermarket düzeni 7

Şekil 2.2. Parça lojistiği problemleri..... 9

Şekil 2.3. İki süpermarketi olan bir fabrika zemini 10

Şekil 2.4. Birbirine bağımlı rotalama ve çizelgeleme kararlarının şematik gösterimi.... 13

Şekil 2.5. Temel KDS bileşenleri 24

Şekil 2.6. Optimizasyona dayalı KDS'lerin genel görünümü..... 28

Şekil 2.7. TB algoritmasında komşu çözümlerin değerlendirilmesi..... 32

Şekil 2.8. Optimizasyon araçlarının tarihsel gelişimi 33

Şekil 3.1. TB algoritmasının pseudo kodu..... 41

Şekil 3.2. TB algoritmasının akış diyagramı 42

Şekil 3.3. Çalışmadaki kromozom gösterimi..... 43

Şekil 3.4. Rastgele seçilen bir düğümün değerinin değiştirilmesi 44

Şekil 3.5. Rastgele seçilen bir süpermarketin çıkarılması ve rastgele seçilen noktada yeni bir süpermarket eklenmesi 45

Şekil 3.6. Komşu çözüm algoritmasının pseudo kodu..... 45

Şekil 3.7. Komşu çözüm algoritmasının akış diyagramı 46

Şekil 3.8. Örnek problem için başlangıç çözümü 47

Şekil 4.1. RA algoritmasının pseudo kodu 55

Şekil 4.2. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm üretme yöntemleri..... 59

Şekil 4.3. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm algoritmasının pseudo kodu..... 60

Şekil 4.4. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm algoritmasının akış diyagramı 61

Şekil 5.1 Geliştirilen Model Tabanlı KDS Yapısı 68

RESİMLERİN LİSTESİ

Resim **Sayfa**

Resim 5.1. Sistemin arayüzü 69

Resim 5.2. Sisteme girilen istasyon x koordinatları 69

Resim 5.3. Sisteme girilen istasyon talepleri 70

Resim 5.4 Sisteme girilen süpermarket kapasiteleri 70

Resim 5.5 Sisteme girilen süpermarket maliyetleri 70

Resim 5.6. Sisteme girilen iterasyon parametresi 71

Resim 5.7. Sisteme girilen süpermarket uzaklığı 71

Resim 5.8. Sistemin çözüm ekranı 71

Resim 5.9. Tek tip süpermarket içeren örnek uygulama sonucu 74

Resim 5.10. İki tip süpermarket içeren örnek uygulama sonucu 75

Resim 5.11. Üç tip süpermarket içeren örnek uygulama sonucu 76

Resim 5.12. Kurulum maliyetleri artırılmış örnek uygulama sonucu 77

Resim 5.13. Sistemde bir önceki çözümün gösterilmesi 78

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
ARP	Araç rotalama problemi
DP	Doğrusal programlama
E	Enerji
GA	Genetik algoritma
IT	Markov zinciri uzunluğu
KDS	Karar destek sistemi
RA	Rastsal arama
SYP	Süpermarket yerleşim problemi
T	Sıcaklık
TB	Tavlama benzetimi
TP	Tam sayılı programlama
TZÜ	Tam zamanında üretim
WPF	Windows Presentation Foundation

1. GİRİŞ

Lojistik kavramı, uluslararası lojistik yönetim konseyi tarafından “Müşterilerin ihtiyaçlarını karşılamak üzere her türlü ürün, servis hizmeti ve bilgi akışının, başlangıç noktasından (kaynağından) tüketildiği son noktaya (nihai tüketiciye) kadar olan tedarik zinciri içindeki hareketinin etkili ve verimli bir şekilde planlanması, uygulanması, taşınması, depolanması ve kontrol altında tutulması hizmetidir” şeklinde tanımlanmaktadır [1]. Lojistik temel olarak üç kısımda incelenebilir: Tedarikçiden tesise kadar olan lojistik (in-bound logistics), tesisten müşteriye olan lojistik (Outbound logistics) ve tesis içi lojistik (in-plant logistics, dock to dock logistics) [2]. Lojistik denilince genelde tedarikçiden tesise ve tesisten müşteriye olan lojistik akla gelmektedir. Fakat imalat lojistiği olarak da geçen tesis içi lojistik, önemli lojistik boyutlarından bir tanesidir. Çalışma kapsamında ele alınacak olan lojistik boyutu tesis içi lojistiktir.

Özelleştirme seçeneklerini artırma eğilimi ve artan ürün çeşitliliği nedeniyle, tam zamanında (just-in-time) parça lojistiği gittikçe artan bir şekilde günümüz otomobil üretimindeki büyük zorluklardan biri haline gelmektedir. Montaj hatlarına parça akışının hiç durmaması için binlerce parça ve tedarikçi, çok sayıda donanım ve yüzlerce montaj işçisi koordine edilmelidir [3]. Montaj hattına, iç lojistik faaliyetleri kapsamında yer alan, montaj işlemleri için gerekli malzeme ve parça tedarikinin istenen zamanda istenen miktarda ve istenen yere sağlanması büyük önem taşımaktadır. Özellikle montaj hattının malzeme ve parça yoksunluğundan dolayı durması yüksek maliyetlerin ortaya çıkmasına neden olmaktadır.

Son yıllarda üreticiler tesis içinde hızlı, esnek ve güvenilir parça lojistiği sağlayabilmek için süpermarket konseptini benimsemişlerdir. Tam Zamanında Üretim (TZÜ) süpermarketi yakınındaki montaj hattının parça talebini karşılayabilmek için kurulan, merkezi olmayan ara bir saklama alanıdır. Süpermarketler merkezi depodan montaj hatlarının ihtiyaç duyduğu farklı büyüklükteki montaj bileşenleri ile doldurulur ve küçük kutular halinde çekici trenler aracılığıyla servis edilir. Kısa mesafeler ve küçük çaplı teslimatlar sayesinde parça tedariki daha esnek bir şekilde yönetilebilir ve öngörülemeyen durumlarla daha kolay başa çıkılabilir [4]. Merkezi depolar beraberinde montaj hattına esnek olmayan malzeme ve parça akışını, büyük miktarlarda fakat az sayıda taşımanın yapılmasını ve istasyon önünde yüksek stok birikimlerini getirir. Fakat malzeme ve parçaların sevkiyatının kontrolünde, tedarik için

anlık stok seviyesini belirleyebilmede ve tesis içi alan kazanımında kolaylıkları da yine beraberinde getirir. Süpermarketler ise merkezi depoların aksine esnek malzeme ve parça akışını sağlar, küçük miktarlarda fakat sık malzeme ve parça taşımaya olanak verir ve istasyon önü stok birikimi bu sayede azalır. Bununla birlikte süpermarketler, tesis içerisinde montaj hattına yakın alanlarda yer alır ve küçük de olsa bir alan işgal ederler, eğer mevcut tesis içerisinde yeni süpermarketlerin oluşturulması gerekiyorsa burada tesis içi yerleşim problemi ortaya çıkar.

Otomotiv sektöründe süpermarket konsepti kullanılmak istenirse çözülmesi gereken karar problemleri vardır. Bu problemler aşağıdaki gibi özetlenebilir [5].

Yerleşim problemi, merkezi olmayan süpermarketlerin sayısının ve fabrika zemini üzerinde yerleşimin belirlenmesi ve sonrasında montaj istasyonlarına atanması problemidir.

Rotalama problemi, her bir çekicinin hangi istasyonlara hangi rotada malzeme ve parça taşıyacağının belirlenmesi problemidir. Çekicilerin rotası süpermarketten başlamakta ve rotadaki bütün istasyonlara sırasıyla gidildikten sonra yine başlangıç noktasında sona ermektedir.

Çizelgeleme problemi, bir çekicinin rotası belirlendikten sonra hizmet verdiği her bir istasyona ne zaman uğrayacağının belirlenmesi problemidir. Özellikle istasyonların malzeme ve parça yoksunluğundan dolayı durmaması için kritik önem taşımaktadır.

Yükleme problemi, bir çekicinin rotası ve çizelgesi belirlendikten sonra istasyonlara her turda hangi malzeme ve parçaların taşınacağının belirlenmesi problemidir. Çekicilerin vagon sayısı, vagon kapasiteleri ve istasyon önü stok düzeyi dikkate alınarak bu karar problemi çözülmektedir.

Bu çalışma kapsamında yukarıdaki problemlerden yerleşim problemi ele alınmıştır. Yerleşim problemi fabrika zemini üzerinde kurulacak süpermarket alanlarının, sayısının ve yerleşiminin belirlenmesidir. Fabrika alanı çok değerlidir, bu nedenle çok sayıda süpermarket yaratmak genellikle avantajdan ziyade daha fazla maliyete neden olacaktır. Diğer taraftan, az sayıdaki, kötü yerleştirilmiş süpermarketlere sahip olmak olumlu etkilerini büyük ölçüde azaltmasına rağmen, merkezileştirilmiş depolarla kurulan taşıma

sistemlerinden daha iyi sonuçlar verir. Bu nedenle mümkün olan alanı kullanarak ve mümkün olan en iyi süpermarket sayısını seçerek, süpermarketleri stratejik olarak yerleştirmek kritik bir öneme sahiptir [6].

Bu çalışmada problemin çözümü için bir Tavlama Benzetimi (TB) algoritması önerilmiştir. Literatürde önerilen bir matematiksel model temel alınarak TB algoritması kullanan bir uygulama geliştirilmiş ve algoritmanın başarısı bir dizi test problemi ile test edilmiştir. Birerleme yöntemiyle test problemlerinin en iyi sonuçları hesaplanmış, uygulamanın sonuçlarıyla karşılaştırılmış ve geliştirilen algoritmanın başarılı şekilde çalıştığı görülmüştür.

Çalışmanın ikinci aşamasında, sabit kapasiteli süpermarketler kullanan matematiksel model, değişken kapasiteli süpermarketler kullanılacak şekilde geliştirilmiştir. Yeni geliştirilen modele göre TB algoritması güncellenmiş ve yeni test problemleri ile algoritmanın başarısı ölçülmüştür. Algoritmanın başarılı şekilde çalıştığı anlaşıldıktan sonra bu algoritmayı kullanan bir Karar Destek Sistemi (KDS) geliştirilmiştir. Sistem süpermarketlerin kapasite ve maliyetlerini, montaj hattı düzeni gibi bilgileri aldıktan sonra, çok kısa süre içerisinde kullanılması gereken süpermarketlerin sayısını ve kapasitelerini, aynı zamanda hangi istasyonlara hangi süpermarketlerin hizmet vereceği konusunda çözüm üretebilmektedir.

Çalışmanın ikinci bölümünde, çalışma ile ilgili kavramlar ve yapılan çalışmalar açıklanmıştır. Süpermarket konseptinin bir parçası olduğu TZÜ sistemi anlatılmış, süpermarket konsepti uygulanırken çözülmesi gereken problemler detaylı olarak ele alınmıştır. Karar destek sistemleri açıklanmış, bileşenleri, özellikleri ve türleri anlatılmıştır. Konu ile ilgili literatür çalışmaları özetlenmiştir.

Çalışmanın üçüncü bölümünde Süpermarket Yerleşim Problemi (SYP) açıklanmış ve problemin süpermarket kapasiteleri dikkate alınmadan TB algoritması yaklaşımı ile çözümü açıklanmıştır. Geliştirilen algoritmanın test verileriyle çalıştırılan sonuçları ve farklı bir algoritma ile kıyaslaması gösterilmiştir.

Çalışmanın dördüncü bölümünde SYP, farklı kapasitedeki süpermarketlerin aynı problem içerisinde çözümü ele alınmıştır. Üçüncü bölümde geliştirilen matematiksel model ve TB algoritması, değişken kapasiteli süpermarketler kullanılacak şekilde geliştirilmiştir.

Algoritma örnek bir problem ile açıklanmış, deneysel çalışma ile elde edilen sonuçlar ve farklı bir algoritma ile üretilen sonuçların karşılaştırılması gösterilmiştir.

Çalışmanın beşinci bölümünde, değişken kapasiteli SYP'nin çözümü için geliştirilen TB algoritmasını model olarak kullanarak geliştirilen karar destek sistemi anlatılmıştır. Sistemin teknik altyapısı anlatılmış, sistemin kullanımı ve kullanıcı arayüzü görsellerle açıklanmıştır. Örnek bir problem geliştirilen karar destek sistemi ile çözülmüştür.

Çalışmanın altıncı bölümünde, bu çalışmada elde edilen sonuçlar ve literatüre katkısı anlatılmıştır. SYP çözümü için önerilen karar destek sisteminin fayda ve avantajları açıklanmıştır. Çalışmanın sınırlılıkları ve gelecek çalışma önerilerine yer verilmiştir.

2. KAVRAMSAL ÇERÇEVE VE LİTERATÜR ÖZETİ

Bu bölümde çalışmada geçen kavramlar açıklanmış ve kavramlarla ilgili literatürden örnekler verilmiştir. İlk olarak TZÜ kavramı açıklanmış, sonrasında süpermarket konsepti ile ilgi çözülmesi gereken problemler anlatılmıştır. Problemlerle ilgili daha önceki yapılan çalışmalara yer verilmiştir. Çalışma sonunda bir KDS geliştirildiği için, KDS ele alınmış, özellikleri ve çeşitleri açıklanmıştır. KDS ile ilgili benzer çalışma örnekleri verilmiştir. Geliştirilen KDS altyapısında bir metasezgisel algoritma kullanıldığı için, metasezgisel algoritmalar açıklanmış ve benzer çalışmalara yer verilmiştir.

2.1. Tam Zamanında Üretim Sistemi

TZÜ, tam zamanında satın alma ve tam zamanında üretimi gerektiren bir maliyet ve stok kontrol sistemidir. Tam zamanında üretim ilk olarak Toyota başmühendisi Taiichi Ohno tarafından Japonların II. Dünya Savaşı sonrasında içinde bulundukları ekonomik şartlarda ortaya atılmış bir yöntemdir. Savaşın sonrasında Japonya, sınırlı doğal kaynaklar, işgücü ve sermaye yetersizliği sebebiyle sınırlı olan kaynakları verimli kullanmayı öğrenmek durumunda kalmıştır. Bir felsefe olarak da görülen TZÜ'nün ortaya çıkışında bu tür ihtiyaçların önemli rolü olmuştur. TZÜ felsefesinin önemi, 1971 petrol krizi sonrasında diğer Japon işletmelerince de anlaşılmış ve ülke genelinde uygulama alanı bulmuştur. 1980'lerin başından itibaren bu felsefeye dayalı üretim Amerika, Avrupa ve sonrasında bütün dünyaya yayılmıştır. TZÜ felsefesi, Amerikan imalat sistemindeki temel ilkelerin Japonya ortamında şekillendirilmesi ile geliştirilmiştir. T. Ohno Amerikan süpermarket fikrinden etkilenmiş ve süpermarketlerin işletilmesindeki temel ilkeler TZÜ sisteminin kavramsal alt yapısını oluşturmuştur [7]. Günümüzde Toyota, Mcdonald's, General Motors, Apple Computer ve IBM gibi dünyaca ünlü firmalar bu yöntemi uygulamaktadırlar.

TZÜ sistemine olan ilgi, Japon şirketlerinin başarılı olduğu görüldükten sonra artmıştır. Ancak TZÜ, Japonya dışında genellikle bir stok kontrol sistemi olarak tanınmıştır. Gerçekte bu yöntem; stoksuz, yani sıfır stokla üretime karşılık gelir ve bu yönüyle diğer stok kontrol sistemlerinden ayrılmaktadır. TZÜ satın alma, mühendislik, pazarlama, personel, kalite-kontrol, müşteri ve satıcı arasındaki ilişkiyi de belirleyerek israfın azaltılması, verimliliğin artırılması amacı güden bir üretim sistemi olarak tanımlanabilir. Görüldüğü üzere TZÜ

bazen bir felsefe, bazen bir üretim sistemi, bazen de bir yönetim tarzı olarak ifade edilmektedir. Ancak gerek felsefe, gerekse yönetim tarzı olarak ifade edilsin, TZÜ bir üretim sistemine karşılık gelmektedir [7].

2.2. Tam Zamanında Üretimde Süpermarket Kullanımı

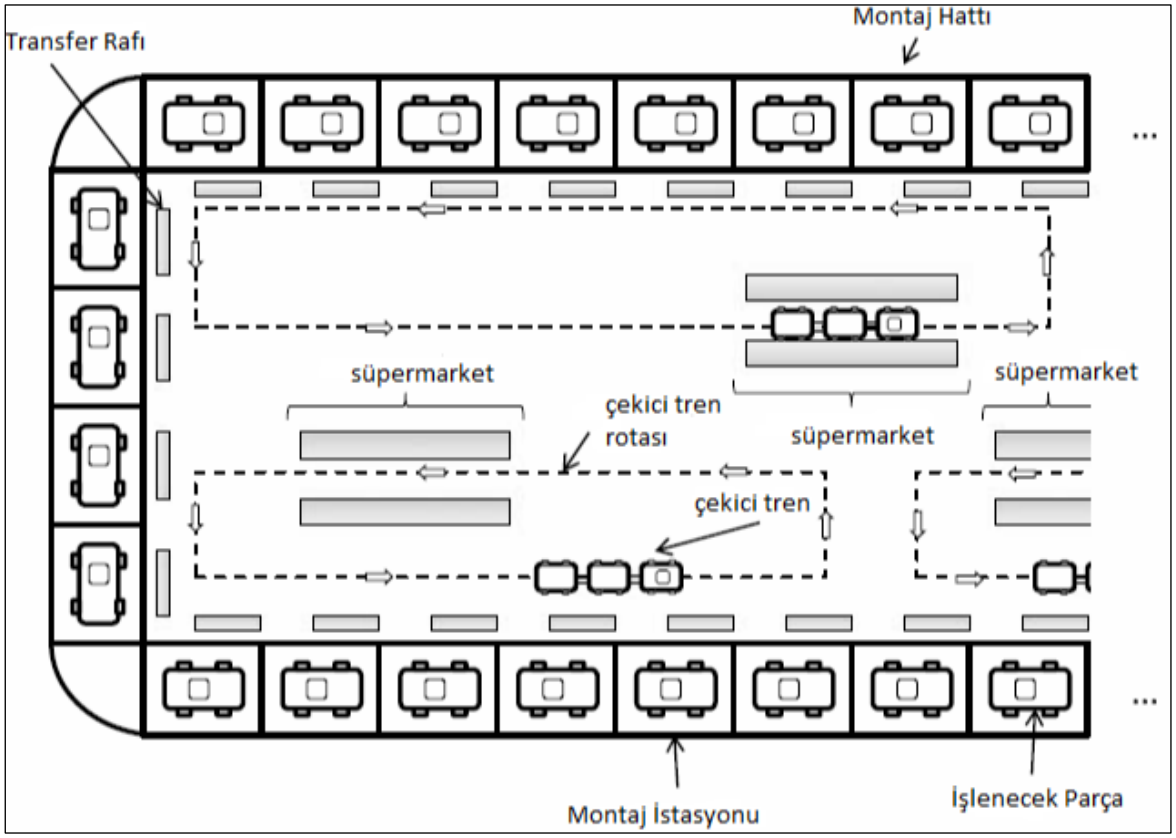
TZÜ süpermarketi, yakındaki montaj istasyonlarına sevk edilecek parçalar için merkezi olmayan, ara bir saklama alanıdır. Süpermarketler, endüstriyel kamyonlar aracılığıyla merkezi alım deposundan doldurulur. Buradan parçalar, montaj istasyonlarına küçük çekici trenler ile servis edilir. Bu yolla, kısa mesafelerdeki küçük parti teslimatlarının hızlı bir şekilde yeniden planlanabilmesi nedeniyle, parça tedarikinde öngörülemeyen olaylara daha esnek bir şekilde cevap verilebilir [5].

TZÜ süpermarketi aynı zamanda bilgisayar teknolojilerinde kullanılan ön bellek (cache) mekanizmasına benzetilebilir. Ön bellek, verilerin geçici olarak saklandığı ve ihtiyaç duyulduğunda hızlı bir şekilde erişim sağlanabilen ara bir saklama alanıdır. Sık kullanılan veriler, gerçek bellekten alınarak ön belleğe yazılır, bu sayede verilere hızlı ve etkili bir şekilde erişilebilir. TZÜ felsefesinin önerdiği şekilde, tüm verileri saklamak yerine, ihtiyaç duyulan veriler saklanarak istenildiğinde verimli bir şekilde erişilebilir.

Süpermarket kullanımının avantajları bugünkü otomobil üretimi açısından çok önemlidir, çünkü montaj hattı istasyonlarındaki alan genel olarak yetersizdir. Ayrıca, küçük parti teslimatlar, montaj hattına yakın raflarda saklanabilecek daha küçük kutuları da gerektirir. Bu sayede montaj işçileri parçalara ergonomik ve verimli bir şekilde erişebilir, işlemler kolaylaşır ve parçanın işleme girme süresini azalır. Süpermarketlerle malzeme taşıma süreci Şekil 2.1'de gösterilmiştir [5].

Sevkiyat kamyonu süpermarkete ulaştığında, lojistik çalışanları gelen parçaları raflara sıralar. Burada, parçalar montaj istasyonundan parça talebi süpermarkete iletilinceye kadar geçici olarak depolanır. Ardından, bir seçim listesi oluşturulur ve bir lojistik işçisi kutuları, parçalarının belirtilen montaj sırasında tanımlandığı şekilde sıralanması gereken seçim listesine göre hazırlar. Doldurulmuş kutular boş vagonlara yüklenir ve vagonlar çekici trenlerin bekleme noktasına taşınır [5].

Çekici tren bir süpermarketin bekleme noktasına geldiğinde, sürücü kendisine tur atanan istasyonlar için vagonları birleştirir ve istasyonları çizelgesinde tanımlandığı gibi ziyarete başlar. Bu çizelge istasyon ziyaretlerinin sırasını ve her durakta durulacak zamanı kesin olarak belirtir. Bazı üreticiler, otobüs ve demiryolu istasyonlarına benzer şekilde her istasyona monte edilmiş, çekicinin bir sonraki gelişine dair geri sayım sayacı gösteren ekran panelleri de uygular. Böylece parça yetersizliği gibi durumları telafi edilebilir bir süre öncesinde görmek, montaj işçileri ve ekip liderleri için daha kolay hale gelir [5].



Şekil 2.1. Şematik bir süpermarket düzeni [5]

Çekici tren bir istasyona geldiğinde, sürücü montaj işlemini engellemeksizin boş malzeme kutularını istasyonun malzeme raflarındaki dolu kutularla değiştirir. Bazı otomobil üreticileri otomatik yönlendirilmiş araçlar ve akış raflarıyla bu süreci tamamen otomatik hale getirmiştir [8]. Bu özel akış rafları, sürüş esnasında tren vagonlarının yükleme-boşaltma yapmasına izin verir. Tren durduğunda, vagonun yanakları ve rafın arkasında bulunan kapılar otomatik olarak açılır ve kutular elastik yaylar vasıtasıyla raflara aktarılır. Bunun tersi de geçerlidir. Bu raflar, bekleme süresinin uzunluğunu yalnızca birkaç saniyeye düşürür, böylece güvenilir çekici tren çizelgeleri oluşturabilir. Bir trenin turundaki tüm

istasyonlara malzeme akışı sağlandıktan sonra, araç süpermarkete geri döner, boş vagonu ayırır ve bir sonraki turunda yola çıkmak için yukarıdaki adımları tekrarlar [5].

Otomotiv endüstrisinde gözlemlenen tipik bir süpermarket şu özellikleri gösterir [9]: Bir süpermarket, 20 ila 30 istasyon arasında hizmet verir ve kendisine atanan montaj hattı segmentinin hemen yakınında bulunur. Tam bir çekici tren turu genellikle 200-500 metreye denk gelmektedir. Her bir süpermarket için üç ila beş çekici tren atanır, böylece beş istasyon tur başına ziyaret edilen duruş noktası sayısını temsil eder ve istasyon ve saat başına yaklaşık üç ziyaret planlanır.

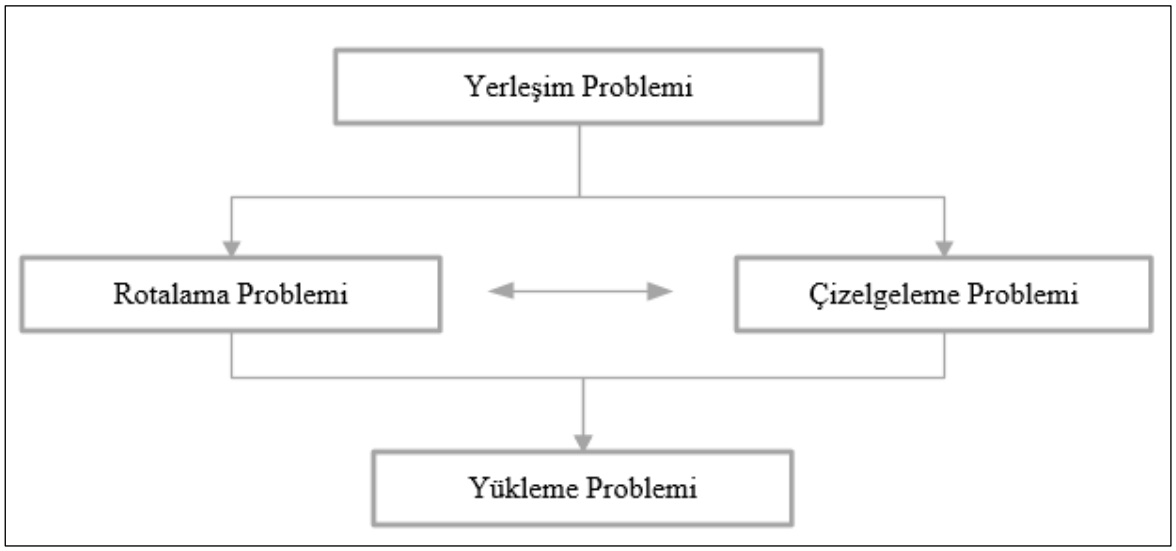
Uzun vadeli stratejik kararlardan kısa vadeli operasyonel süreçlere kadar süpermarket konsepti uygulanırken çözülmesi gereken karar problemleri şu şekilde özetlenebilir [5]:

- Yerleşim Problemi: İşletme zeminine kurulacak süpermarket sayısı ve bunların kesin yerleri belirlenmelidir. Sonrasında, her süpermarket, hizmet ettiği istasyonlara kabaca atanmalıdır. Açıkça görüldüğü gibi, yeni kurulan bir fabrikada ilk yer planlaması ile bir yer değiştirme problemi arasında farklılık olacaktır. Birinci durumda, süpermarket başına uygun bir yer bulma konusunda bir miktar esneklik mevcutken, ikinci durumda, hareketsiz makinelerin neden olduğu, boyut ve konuma ilişkin ciddi kısıtlamaların düşünülmesi gerekir. Amaç fonksiyonunda, ek bir süpermarketin getirdiği sabit maliyetin, süpermarket konseptinin operasyonel kazanımlarına karşı tartılması gerekir. Açıkçası, bu kazançların kesin bir şekilde ölçülmesi, tam planlama hiyerarşisi üzerinde bir tahmin gerektirir. Alternatif olarak, örneğin herhangi bir ilave süpermarket için montaj hattına olan hareket mesafesindeki azalmanın ölçülmesi de problem çözümünde kullanılabilir.
- Rotalama Problemi: Her bir çekicinin hangi istasyonlara hangi rotada malzeme ve parça taşıyacağının belirlenmesi problemidir. Herhangi bir süpermarket ve kendisine tahsis edilen montaj hattı segmenti için, çekici trenlerin araç rotalama probleminin çözülmesi gerekir. Diğer bir deyişle süpermarkete tahsis edilen trenlerin sayısı ve her bir tren tarafından hizmet verilecek montaj istasyonlarına karar verilmelidir. Süpermarkette başlayıp biten bir rota, bir çekici trenle ziyaret edilecek bir istasyon dizisi ile tanımlanır. Montaj hattına eklenen her bir çekici tren bir sabit maliyet artışına sebep olurken, diğer taraftan rota başına ulaşım mesafesi kısalacağından montaj hattındaki stok azaltılabilir.
- Çizelgeleme Problemi: Çekicinin rotası belirlendikten sonra hizmet verdiği her bir istasyona ne zaman uğrayacağının belirlenmesi problemidir. TZÜ felsefesine göre

istasyonlardaki stok birikimi en aza indirgenirken, istasyonlar hiçbir zaman parça yoksunluğu yaşamamalıdır. Aksi halde, en kötü durumda yüzlerce montaj işçisinin boşta kaldığı bir montaj hattı durması yaşanır.

- Yükleme Problemi: Bir çekicinin rotası ve çizelgesi belirlendikten sonra istasyonlara her turda hangi malzeme ve parçaların taşınacağının belirlenmesi problemidir. Çekicilerin vagon sayısı, vagon kapasiteleri ve istasyon önü stok düzeyi dikkate alınarak bu karar problemi çözülmektedir. Tur başına yüklenecek parça kutuları sayısı ve türleri, planlama problemi haline gelebilir. Fabrika zemininin dar koridorlarındaki keskin dönüşler, çekici trendeki vagon sayısını sınırlar. Bu nedenle, çekici tren kapasitesi genellikle yetersizdir. Yükleme problemi istasyonlarda stok birikiminin en aza indirgenmesini sağlarken, istasyonların parça yoksunluğu yaşamamasını garanti altına almalıdır.

Bir montaj hattında öncelikle, her bir istasyondaki iş yüklerinin dengelenmesi anlamına gelen montaj hattı dengeleme problemi çözülmelidir. Sonrasında sırasıyla SYP, rotalama, çizelgeleme ve yükleme problemleri çözülür. Daha önceki çalışmalarda rotalama ve çizelgeleme problemlerinin genelde birlikte çözüldüğü görülmektedir. Yükleme probleminin de bazen bu iki problemle birlikte çözüldüğü görülürken, SYP diğer problemlerden önce ve bağımsız olarak çözülmektedir. Şekil 2.2 bu problemlerin çözüm sırasını göstermektedir.

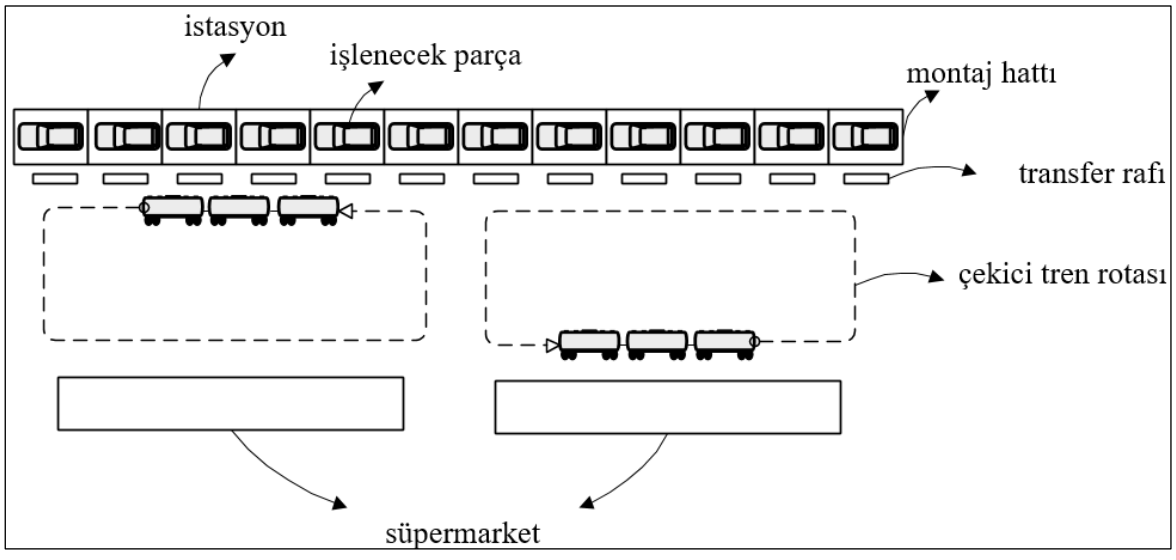


Şekil 2.2. Parça lojistiği problemleri

2.2.1. Yerleşim problemi

Süpermarket konseptine geçiş için önemli optimizasyon problemlerinden bir tanesi, süpermarket alanlarının sayısının ve yerleşiminin belirlenmesidir. Kurulan her süpermarketin kurulum ve fabrika zemininde kapladığı yer bakımından bir maliyeti vardır. Bu nedenle çok sayıda süpermarket yaratmak genellikle avantajdan daha fazla maliyete neden olacaktır. Diğer taraftan, az sayıdaki, kötü yerleştirilmiş süpermarketlere sahip olmak olumlu etkilerini büyük ölçüde azaltmasına rağmen, merkezileştirilmiş depolarla kurulan taşıma sistemlerinden daha iyi sonuçlar verir. Bu nedenle mümkün olan alanı kullanarak en iyi süpermarket sayısını seçmek ve onları stratejik olarak yerleştirmek kritik bir öneme sahiptir [6].

Şekil 2.3'te iki süpermarketi olan bir fabrika ortamı düzenini kavramsal olarak gösterir. Merkezi depo, büyük paket teslimatlarını süpermarketlere dağıtır. Burada parçalar çekici trenlerin uğrayacağı istasyonlara göre yeniden paketlenir ya da sıralanır. Burada unutulmaması gereken bir sonraki adımda, parçaların istasyonlara gerçekte gönderilmesi süreci, süpermarket için atanmış çekici trenlerin rotalama, çizelgeleme ve yükleme işlemlerini gerektirdiğidir. Tüm bu sorunlar birbirine bağlantılı olsa da pratikte hepsinin birlikte çözümünü görmek zordur. Çünkü süpermarket yerleşim problemi öncelikli ve uzun bir karar süreci olduğu halde; rotalama ve çizelgeleme problemi ancak üretim sırası hakkında bilgi sahibi olduktan sonraki birkaç gün içerisinde çözülebilir [6].



Şekil 2.3. İki süpermarketi olan bir fabrika zemini [6]

2.2.2. Rotalama ve çizelgeleme problemi

Çekici trenler için rotalama ve çizelgeleme şüphesiz önemli optimizasyon problemlerinden bir tanesidir. Her bir çekici tren parça tedarik edilecek olan istasyonların bir alt kümesine atanmalı ve istasyon ziyaretlerinin sırası belirtilmelidir (rotalama). Ayrıca belirli bir rota için her bir duruş noktasına varış ve ayrılışları tanımlayan teslimat çizelgesi belirlenmelidir (çizelgeleme). Açıkça görülüyor ki, her iki problem de birbirine çok bağımlıdır ve bu yüzden çok karmaşık bir karar problemi ortaya çıkmaktadır. Bununla birlikte, fabrika zemininin dar koridorlarında hareket eden çekici trenlerin sınırlı manevra yeteneği ve TZÜ felsefesine göre çekicileri yüklemek gibi otomotiv endüstrisindeki bazı gerçek dünya koşulları, bazı basitleştirmelere izin vermektedir. Bu sayede en iyi çözümler birleştirilmiş tek bir optimizasyon yaklaşımıyla belirtilebilir [9].

Literatürde süpermarket bağlamında rotalama ve çizelgeleme konusunda çok az kaynak olmasına rağmen, bu iç lojistik problemleri geleneksel dağıtım ağları problemleriyle benzerlik gösterir. Yine de Araç Rotalama Problemi (ARP) müşterilerin (ya da istasyonların) kısıtlama olmaksızın tam bir kez ziyaret edilmesini gerektirir. Fakat bu durum montaj hattında parça tedarikindeki durumu yansıtmaz. Diğer bir deyişle probleme sadece rotalama açısından bakarken, çekici tren rotalama probleminin bir parçası olan çizelgeleme ile ilgilenmez [9].

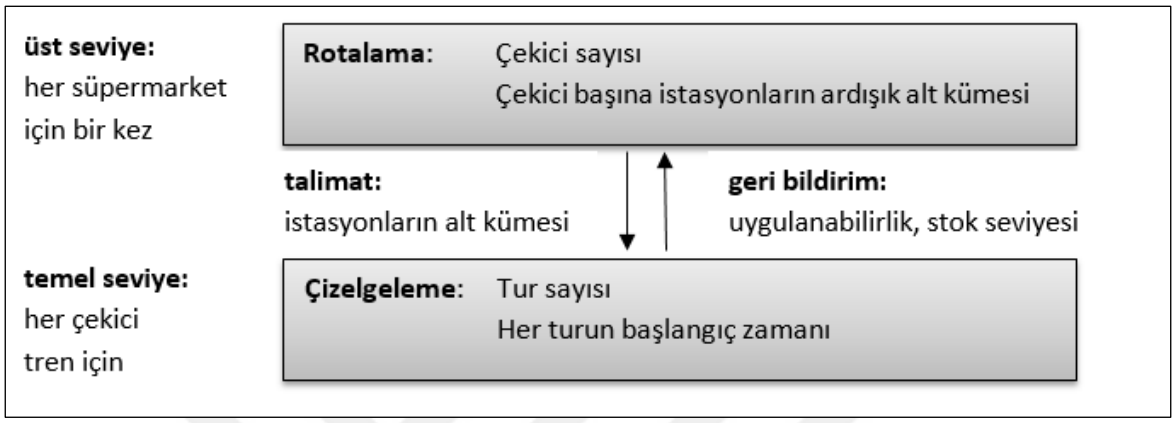
Rotalama problemi parça tedariki için atanacak çekici trenlerin filo büyüklüğünü ve belirli bir istasyon seti üzerinde çekicilerin nasıl taksim edileceğinin belirlenmesidir. Ayrıca tipik bir rotalama probleminde durma noktalarının sırası kararlaştırılmalıdır. Bununla birlikte, otomotiv endüstrisindeki iki özellik bu ilave kararı kolaylaştırmaktadır. Birden fazla vagon takılı haldeyken bir çekici ters yönde hareket edemez ve tipik olarak dönme yarıçapı bir atölyenin dar koridorları için çok büyüktür. Bu sebeple montaj hattının akış yönünde sadece tek yönlü çekici turları mümkündür. Ayrıca dar koridorlarda sıkışıklıktan kaçınmak için, genellikle otomobil üreticileri, çekiciler arasında istasyonların art arda bölünmesini hedeflemektedir. Bu durum üç, beş ve yedi istasyonlarının bir rota üzerindeyken, aradaki istasyonların aynı rotada olma zorunluluğu doğurur. Bu iki özellikle birlikte, rotalama problemi istasyonların çekici trenler üzerinde ardışık alt gruplara bölümlendirilmesine indirgenir. Kapsamlı bir çekici filosu hem satın alma, hem de bakım maliyeti ve personel masrafı gerektirdiğinden bu sayının en düşük düzeyde tutulması arzu edilir. Diğer taraftan,

daha fazla araç parça tedarikini kolaylaştırır, daha sık teslimatlar yapılmasını sağlar ve bu sayede hatlardaki stok birikimi en aza indirgenir [9].

Çizelgeleme problemi, rotalama problemi tarafında belirlenen her bir çekici tren ve ilgili istasyon alt kümesi için çözülmelidir. Bir yandan, her çekici için, süpermarkette yükleme işlemi ve atanmış tüm istasyonlarda eksiksiz bir döngüden oluşan tur sayısı belirlenmelidir. Diğer yandan her duruş noktasının tur boyunca varış zamanları belirtilmelidir. Bununla birlikte, yine otomotiv endüstrisindeki özel koşullar, sonraki karar probleminin basitleştirilmesine izin verir. Süpermarket ve istasyonlar arasındaki kısa mesafelere ve kısa süreli istasyon ziyaretlerine bağlı olarak, turlar, bir çekici üzerine yüklenen kutu kompozisyonu ne olursa olsun, aynı süreye sahip olarak yaklaştırılabilir. Böylece, çizelgeleme problemi tur sayısını ve her turun başlangıç saatini belirlemeye indirgenmiş olur. Bu karar problemi, kapasitesi sınırlı çekicilerin bulunduğu montaj hatlarında stoğu en küçüklerken; aynı zamanda istasyonların malzeme yoksunluğu yaşamamasını hedefler. Bir çekiciye sınırlı sayıda vagonun bağlanabileceğini unutmamak gerekir, aksi takdirde bir çekici tren, atölyenin keskin dönüşleri üzerinde güvenle sürülemez. Teslim edilecek parçalara olan talep, karışık modelli montaj hattında başlatılan ürün modellerinin üretim sırası tarafından belirlenir. Otomotiv endüstrisinde üretim sırası, üretim başlamadan üç-dört gün önce belirlenir ve parça tedarikçilerine bildirilir. Çekici trenlerinin rotalama ve çizelgeleme problemi, her üretim vardiyası için çözülen günlük bir sorundur; üretim sırası kesinlikle bilinir ve her istasyon için parça talepleri ve üretim döngüsü tam olarak tanımlanır. Açıkça görülüyor ki, parça yoksunluğu ve bunun sonucunda oluşan montaj hattı durmaları önlenmelidir. Ancak, sınırlı çekici kapasitesi göz önüne alındığında, rotalama problemini çözerek atanan herhangi bir istasyon alt seti için bu her zaman mümkün değildir. Diğer yandan TZÜ felsefesine göre stok azaltılmalıdır. Bir istasyondaki aşırı stok, montaj işlemlerini engeller ve bu diğer istasyonları etkiler. Bu nedenle, tüm üretim döngüleri ve istasyonlar üzerinde stok toplamını en aza indirmek hedeflenmelidir [9].

Açıkça görülebildiği gibi, her iki sorun birbirine çok bağımlıdır. Rotalama problemi, her çekici tren tarafından hizmet verilecek istasyonların alt kümesini belirler. Rotalama probleminin bu karar değişkeni, her çekicinin çizelgelenmesindeki alt düzeydeki bir parametre olarak işlev görür. Öte yandan, yalnızca ilgili çizelgeleme probleminin çözülmesi, istasyonların belirli bir alt grubunun, sınırlı çekici kapasitesi göz önüne alındığında uygulanabilir bir teslimat çizelgesine izin verip vermediğini belirleyebilir. Aynı şekilde, bir

çekicinin atanmış istasyonlarında parça envanterini tam olarak belirlemek için, çizelgeleme problemi çözülmelidir. Bu nedenle, bu güçlü karşılıklı bağımlılık nedeniyle, her iki sorunu da aynı anda çözmenin daha uygun olduğu söylenebilir. Şekil 2.4 bu iki seviye için karar görevlerini ve aralarındaki ilişkiyi göstermektedir [9].



Şekil 2.4. Birbirine bağımlı rotalama ve çizelgeleme kararlarının şematik gösterimi [9]

2.2.3. Yükleme problemi

TZÜ ilkesine göre, parça talepleri genel olarak bir üretim sisteminin lojistik zinciri boyunca "çekilir" ve son montajda asıl ihtiyaca göre mümkün olduğunca eş zamanlı olarak teslim edilmelidir. Dolayısıyla yükleme problemi basitçe, belirli bir çekici tren çizelgesinin, çekicinin bir sonraki ziyaretine kadar tam olarak her istasyonda işlenecek parça kadar yüklenmesi sayesinde çözülür. Bununla birlikte, büyük karışık modelli montaj sistemlerinde, belirli istasyonlar için bir turda en azından bir noktada çekici tren kapasitesi sıklıkla darboğaz haline gelir. Bu sebeple gerekli tüm kutular o turda sevk edilemeyebilir. Çok sayıdaki parçadan dolayı, üretim dizilerinin çok dikkatli planlanması bile bu durumlardan kaçınamaz; böylece bu kısa vadeli teslimat darboğazlarına birtakım müdahale gerekli hale gelir. Bir çekicinin kapasitesinde geçici bir artış, emniyet kısıtlamaları nedeniyle genellikle imkânsızdır ve kısa süreli ek çekiciler kullanmak, atölye trafiğini artıracağı ve ek kaynakların kullanılmasına sebep olacağı için istenmemektedir. Aynı şekilde, tren çizelgelerinin değiştirilmesi, ek bir planlama çaba harcamasına da neden olabilir; çünkü herhangi bir değişiklik, trenin hizmet verdiği tüm istasyonlarda gelecekteki turlarını etkileyecektir. Bunun yerine darboğazda kalacak trenin daha önceki turlarındaki toplama listeleri uzatılabilir, böylece fazlalık parçalar istasyonlara gerektiğinden önce ulaşmış olur.

Bu politikanın arkasındaki mantık, TZÜ felsefesinden uzaklaşmadan, önceki turlardaki uygun çekici tren kapasitelerini kullanarak dar boğazları mümkün olduğunca azaltmaktır [8].

Karışık modelli bir montaj hattı, bazı mekanik taşıma sistemi boyunca, genellikle bir konveyör bandında düzenlenmiş bir dizi istasyondan oluşur. Sevkiyat sistemi sabit bir hızda çalışır ve iş parçaları montaj hattına sabit bir hızda sevk edilir. Toplam planlama anlayışı bu nedenle sabit bir çevrim süresi ile ilişkili olan iş çevrimleri olarak ayrılabilir. Çevrim süresi, genellikle, herhangi iki iş parçasının bir istasyona girişi arasındaki zaman aralığı ile ilgilidir ve bu nedenle, bir istasyonun belirli görevlerinin ortalama olarak yürütülmesi gereken zamana karşılık gelir (Örneğin otomotiv endüstrisinde her çalışma döngüsü gereken zaman 60-90s civarındadır). Çalışma döngüleri artan sırayla numaralandırılmış ve iş parçalarına atanmıştır, böylece çevrim adı verilen döngü, üretim sırasının belirli bir çevriminde monte edilen iş parçasına karşılık gelir. Hattın sabit hızı ve besleme hızından dolayı, bir iş parçasının hangi istasyonda hangi zamanda olacağını kolayca tespit edilebilir, iş parçalarının parça talepleri de bilineceği için, malzeme talebi ve tüketim süresi herhangi bir istasyonunda kesin olarak bilinir [8].

Parça talebi, yükleme probleminin çözülmesinden saatler önce tüm tedarikçilere iletilen son montajın üretim sırası ile belirlenebilir. Günümüzde otomobiller gibi çok yönlü ürünler müşteriler tarafından kendi ihtiyaçlarına göre belirlenebilir, böylece açılır tavan veya deri kaplama gibi özelleştirilebilir ürün seçenekleri de seçilebilir. Bu nedenle, her çalışma döngüsünde gereken parça sayısını tam olarak belirten, çizgi üzerinde başlatılan özelleştirilmiş ürün modelleri dizisidir. Ek olarak, her bir çekici trenin sabit programı, bir çekici treninin bir istasyona malzeme sağladığı döngüleri tam olarak belirler. Parçaların her biri ayrı ayrı değil, birkaç parça içeren küçük kutular halinde taşınır. Bununla birlikte, kutuların boyutu standartlaştırıldığı için, çekici treninin herhangi iki durağı arasındaki bir istasyondaki kutulara olan talep kolayca hesaplanabilir. Kutu talebi belirlendikten sonra, her bir trenin her turda ihtiyaç duyulan kutu kadar yüklenmesi ile TZÜ prensibi sağlanmış olur [8].

2.3. Literatür Özeti

İç lojistik kavramının planlama ve kontrolü, süpermarket yerleşim problemi, çekici tren rotalama, çizelgeleme ve yükleme gibi birbirine bağlı karar problemleri içerir. Süpermarket

yerleşim problemi ile benzerlik gösteren depo yerleşim problemi literatürde yaygın olarak yer bulmuştur. Fakat bu çalışmalar, bizim problemimiz olan tesis içi montaj hatlarını besleyen merkezi olmayan depoların (süpermarket) yerleşim problemiyle aynı değildir. Aynı şekilde ARP ile benzerlik gösteren süpermarket rotalama probleminin, ARP'den farklı olarak çizelgeleme problemi ile beraber ele alındığı görülmektedir. Yükleme problemi ise bu iki problem çözüldükten sonra ele alınmaktadır. Bununla birlikte her üç probleminde birlikte dikkate alındığı çalışmalar da bulunmaktadır. Konu ile ilgili Boysen ve diğerlerinin literatür araştırması incelenebilir [3].

Merkezi olmayan parça lojistiği ve merkezi olmayan depo konseptleri literatürde geniş yer bulmuştur. Örneğin Yano ve diğerleri merkezi olmayan lojistiğin avantajlarını vurgularken, Wanke ve Zinn merkezi olmayan depo kullanımındaki stratejik karar seçeneklerini incelemiştir [10,11]. Bunlara rağmen literatürün çoğu bu kararı alırken üretim ortamından daha çok tedarik zinciri ortamına yoğunlaşmıştır. Az sayıda çalışmada üretim ortamındaki merkezi olmayan depo (süpermarket) konseptine yer verilmiştir.

Aşağıda otomotiv sektöründe süpermarket konsepti uygulanırken çözülmesi gereken karar problemleri olan yerleşim, rotalama, çizelgeleme ve yükleme ile ilgili çalışmalara yer verilmiştir. Çalışmamızla doğrudan ilgili olduğu için önce süpermarket konsepti ve yerleşim problemi ile ilgili yayınlardan bahsedilmiş, sonrasında diğer problemlerle ilgi çalışmalar kronolojik sırayla anlatılmıştır. Çizelge 2.1'de bu çalışmalarda ele alınan problem ya da problemler ve önerilen çözüm yöntemleri gösterilmiştir.

Çizelge 2.1. İlgili çalışmalar

Yazar(lar), Yıl	Problem	Yöntem	Açıklama
Battini vd. (2010)	Yerleşim Problemi	Karar Verme Prosedürü	Süpermarket konseptine geçiş karar seçeneklerini incelemişlerdir.
Emde ve Boysen (2012)	Yerleşim Problemi	Dinamik Programlama Algoritması	Problem polinom zamanda çözülmüştür.
Alnahhal ve Noche (2015)	Yerleşim Problemi	Genetik Algoritma	Problem NP-Complete sınıfındadır.

Çizelge 2.1. (devam) İlgili çalışmalar

Yazar(lar), Yıl	Problem	Yöntem	Açıklama
Nourmohammadi ve Eskandari (2017)	Yerleşim Problemi	Matematiksel Model	Montaj hattı dengeleme ve SYP'i birlikte çözmüşlerdir. Problem NP sınıfındadır.
Nourmohammadi vd. (2018)	Yerleşim Problemi	Stokastik Matematiksel Model	SYP'i değişken istasyon talepleri ile birlikte incelemişlerdir.
Nourmohammadi vd. (2019)	Yerleşim Problemi	Hiyerarşik Matematiksel Model	Montaj hattı dengeleme ve SYP'i birlikte çözmüşlerdir.
Zhou ve Tan (2019)	Yerleşim Problemi	Dağılım tahmini algoritması	Süpermarket kullanım oranı ve kapasitesi dikkate alınmıştır. Problem NP-Complete sınıfındadır.
Battini vd. (2013)	Süpermarket Konsepti		Detaylı bir literatür araştırması sunmuşlardır.
Vaidyanathan vd (1999)	Rotalama Problemi	En Yakın Komşu Algoritması	
Kozan (2000)	Atama ve Sıralama Problemi	Genetik Algoritma	
Choi ve Lee (2002)	Rotalama ve Çizelgeleme Problemi	Yerel Arama Prosedürü	
Boysen ve Bock (2011)	Çizelgeleme Problemi	Dinamik Programlama ve Tavlama Benzetimi Algoritması	Problem NP-Hard sınıfındadır.
Emde vd. (2012)	Yükleme Problemi	Polinom Zamanlı Çözüm Prosedürü	Problem polinom zamanda çözülmüştür.
Emde ve Boysen (2012)	Rotalama ve Çizelgeleme Problemi	Dinamik Programlama Algoritması	Problem polinom zamanda çözülmüştür.
Golz vd (2012)	Rotalama, Çizelgeleme ve Yükleme Problemi	Sezgisel Çözüm Prosedürü	
Faccio vd. (2013)	Rotalama, Çizelgeleme ve Yükleme Problemi	Simülasyon Modeli	

Çizelge 2.1. (devam) İlgili çalışmalar

Yazar(lar), Yıl	Problem	Yöntem	Açıklama
Faccio vd. (2013)	Kanban Numaraları En Küçüklenmesi		Karar değişkenlerini optimize etmek amacıyla bir prosedür önermişlerdir.
Rao vd. (2013)	Rotalama ve Çizelgeleme Problemi	Genetik Algoritma ve Tavlama Benzetimi Algoritması	
Fathi vd. (2014)	Çizelgeleme ve Yükleme Problemi	Tavlama Benzetimi Algoritması	
Fathi vd. (2014)	Çizelgeleme ve Yükleme Problemi	Karınca Kolonisi Algoritması	Problem NP-Hard sınıfındadır.
Dong vd. (2015)	Rotalama, Çizelgeleme ve Yükleme Problemi	Sezgisel Tabanlı Dinamik Dağıtım Stratejisi	
Boysen vd. (2015)	Parça Lojistiği		Otomotiv sektöründe parça lojistiği ile ilgi detaylı bir literatür araştırması sunmuşlardır.
Dong vd. (2016)	Çizelgeleme Problemi	Dinamik Programlama Algoritması	
Emde ve Gendreau (2017)	Çizelgeleme Problemi	Tabu Arama Algoritması	Problem NP-Complete sınıfındadır.

Battini ve diğerleri siparişe dayalı montaj ortamında merkezi olmayan depo karar seçeneklerini incelemiştir. Montaj hatlarını süpermarketlere bileşen ortaklığına göre atamışlar ve süpermarketleri mümkün olduğunca kendi hatlarına yakın olacak şekilde yerleştirmişlerdir. Her süpermarkette tutulacak güvenlik stokları nedeniyle oluşacak stok maliyetlerini ve montaj hattının süpermarketten yeniden doldurulması nedeniyle doğacak taşıma maliyetlerini ölçmek için bir yöntem önermişlerdir [12].

Emde ve Boysen benzer şekilde, tek bir montaj hattı boyunca istasyonlara parça tedariki sağlayan süpermarketlerin yerleştirilmesi için en iyi çözüm üreten bir algoritma sunmuşlardır. Yeni bir süpermarket eklemenin yaratacağı sabit maliyet ile istasyon talepleri ile ağırlıklandırılmış taşıma mesafelerinin azalması arasındaki farkı hesaplayan ve ortaya

çıkan problemi polinom zamanda çözen bir dinamik programlama algoritması modellemişlerdir [6]. Alnahhal ve Noche ise aynı problemi tam sayılı programlama modeli ve genetik algoritma (GA) kullanarak çözmeye çalışmışlardır. Emde ve Boysen'in çalışmasından farklı olarak, problemi çözerken tesis içinde süpermarket kurulumu için uygun olmayan bölgeler ve süpermarket kapasiteleri göz önüne alınmıştır [13]. Yeni eklenen bu parametrelerle birlikte problem NP-complete hale gelmiştir [14]. Zhou ve Tan, SYP için dağılım tahmini algoritması (Estimation of distribution algorithm) önermişler ve problemi süpermarket kullanım oranları ve kapasiteleri ile birlikte incelemişlerdir. Geliştirdikleri algoritmanın performansını, Alnahhal ve Noche tarafından önerilen GA ile karşılaştırmışlar ve daha iyi sonuçlar verdiğine karar vermişlerdir [14].

Nourmohammadi vd. [15] süpermarket yerleşim problemini değişken istasyon talepleri ile birlikte incelemişlerdir. Değişken istasyon talepleri, envanter ve kurulum maliyetlerini de içeren bir SYP modeli önermişler ve önerdikleri modeli test problemleri ve bir işletme üzerinde denemişlerdir.

Bunların dışında, süpermarket yerleşim problemini farklı bir optimizasyon problemi olan montaj hattı dengeleme problemi ile birlikte çözen çalışmalara da rastlanmaktadır. Nourmohammadi ve Eskandari daha önceki çalışmalardan farklı olarak, iki seviyeli bir matematiksel model ile montaj hattı dengeleme ve SYP'i ilk defa birlikte çözmüşlerdir [16]. Daha sonra Nourmohammadi ve diğerleri bu iki problemin birlikte çözümü için hiyerarşik bir matematiksel model önermişlerdir. SYP'i çözerken aynı zamanda stok ve kurulum maliyetlerini de göz önüne almışlardır [17].

Vaidyanathan ve diğerleri çekicilerin rotalama problemi için bir matematiksel model önermişlerdir. Bununla birlikte bir en yakın komşu algoritması ve bir iyileştirme algoritması da önermişlerdir. Planlama periyodu boyunca herhangi bir çekicinin rotası aynı kalmaktadır. İstasyonların malzeme ve parça talepleri sabittir ve zaman içerisinde değişmemektedir. Geleneksel araç rotalama problemi konseptine bağlı kalınarak rotalar oluşturulurken toplam seyahat zamanı en küçüklenmesi amaç edinilmiştir. Bununla birlikte istasyon ihtiyacı çekicinin bir sonraki gelme zamanına kadar karşılanmaktadır. Böylece istasyon önü stok yığılmalarının en küçüklenmesi da amaçlanmaktadır. Bu nedenle çekiciler kısa aralıklarda tura çıkmaktadır [18].

Kozan, istasyonlara malzeme ve parça taşımada kullanılan forkliftler için atama ve sıralama problemlerini beraber ele alan bir matematiksel model ve GA önermiştir [19].

Choi ve Lee, çekicilerin rotalama ve çizelgeleme problemlerini birlikte ele almıştır. Dağıtım zamanı sapması en küçüklenmesi amaç edinerek bir yerel arama prosedürü önermişlerdir [20].

Boysen ve Bock, merkezi bir depodan forkliftler vasıtasıyla istasyonlara taşınan malzeme ve parçalar için forklift çizelgeleme problemini ele almışlardır. Problemin çözümü için dinamik programlama ve tavlama benzetimi (TB) algoritması önermişlerdir. Bütün istasyonlardaki ağırlıklı en büyük stok seviyesinin en küçüklenmesi amaçlanmaktadır [21].

Emde ve diğerleri, karışık modelli montaj hatlarında, çekicilerin yüklenmesi problemini ele almışlardır. Limitli çekici kapasiteleri altında istasyon önü stok seviyesi en küçüklenmesi amaç edinilmiştir. Çekiciler kısa aralıklarda tura çıkmaktadır ve istasyonların malzeme ve parça yoksunluğunun önüne geçilmektedir. Problemin çözümü için bir polinom zamanlı çözüm prosedürü önerilmiş ve test edilmiştir [8].

Emde ve Boysen, karışık modelli montaj hatlarında, rotalama ve çizelgeleme problemlerini ele almışlardır. Merkezi olmayan depolar dikkate alınmıştır ve istasyonlara malzeme taşımaları çekiciler ile yapılmaktadır. Problemlerin çözümü için bir dinamik programlama yaklaşımı önerilmiştir. Zaman boyutunda değişken taleplere sahip istasyonların malzeme ve parça stok seviyelerinin ve çekicilerin vagon sayısının en küçüklenmesi amaç edinilerek periyodik ve periyodik olmayan çizelgeler ve sabit rotalar elde etmişlerdir. Ayrıca bu çizelgeler birbirleri ile kıyaslanmıştır ve periyodik olmayan çizelgelerde daha az istasyon önü stok birikimi olduğu ifade edilmiştir [9].

Golz ve diğerleri karışık modelli montaj hatlarında çekicilerin rotalama, çizelgeleme ve yükleme problemlerini ele almışlardır. Malzeme ve parçalar merkezi bir depodan taşınmaktadır. Araç sayısı en küçüklenmesi amaçlanmıştır ve problemlerin çözümü için bir sezgisel yaklaşım önerilmiştir [22].

Faccio ve diğerlerinin yaptığı çalışmada merkezi olmayan depolardan istasyonlara malzeme ve parça taşımada kullanılan çekicilerin rotalama, çizelgeleme ve yükleme problemi ele

alınmıştır. Karışık modelli montaj hatları dikkate alınmıştır. Problemlerin çözümünde simülasyon modeli kullanılmıştır [23].

Battini ve diğerleri merkezi olmayan depo konseptini ele almışlar ve süpermarket konseptinde ortaya çıkan temel problemleri tanımlamışlardır. Bununla birlikte konu ile ilgili detaylı bir literatür araştırması sunmuşladır [5].

Faccio ve diğerlerinin yaptığı çalışmada merkezi olmayan depolarda malzeme ve parça tedariki ele alınmıştır. Kanban sayısı en küçüklenmesi altında karışık modelli montaj hatları dikkate alınmıştır. Süpermarket konseptindeki tüm karar değişkenlerini optimize etmek amacıyla bir prosedür önermişlerdir [24].

Rao ve diğerleri merkezi bir depodan karışık modelli montaj hattına malzeme ve parça taşıyan tek bir araç için rotalama ve çizelgeleme problemini ele almışlardır. Problemlerin çözümünde GA ve TB yaklaşımını birlikte kullanarak hibrit bir çözüm yaklaşımı önermişlerdir [25].

Fathi ve diğerleri karışık modelli montaj hatlarında merkezi olmayan depolardaki çekiciler için çizelgeleme ve yükleme problemlerini beraber ele almışlardır. Problemlerin çözümü için bir karışık tam sayılı doğrusal model ve bir TB yaklaşımı önerilmiştir. Tur sayısı en küçüklenmesi ve istasyon önü stok seviyesi en küçüklenmesi amaçlanmıştır [26].

Fathi ve diğerleri çift taraflı montaj hatlarında merkezi olmayan depolardaki çekiciler için çizelgeleme ve yükleme problemlerini beraber ele almışlardır. Problemlerin çözümü için bir karışık tam sayılı doğrusal model ve bir karınca kolonisi algoritması önerilmiştir. Tur sayısı en küçüklenmesi ve istasyon önü stok seviyesi en küçüklenmesi, dağıtım zamanı, araç kapasitesi, istasyon kapasitesi kısıtları altında amaçlanmıştır [27].

Dong ve diğerleri karışık modelli montaj hatlarında kullanılan merkezi olmayan depolarda dağıtım faaliyetleri için dinamik bir dağıtım stratejisi önerilmiştir. Beklenmeyen karışıklıklar, diğer bir deyişle çekicilerde ortaya çıkan sorunlar ve üretim sırasında ortaya çıkan problemler de dikkate alınmıştır. Çekicilerin rotalama, çizelgeleme ve yükleme problemleri için sezgisel tabanlı dinamik dağıtım stratejisi önerilmiştir [28].

Boysen ve diğerleri otomotiv endüstrisindeki parça lojistiğinin temel süreçlerini siparişlerin alınmasından boş parça konteynırlarının geri dönüşüne kadar ele almışlar ve konu ile ilgili detaylı bir literatür araştırması sunmuşlardır. Karşılaşılan karar problemlerini açıklamışlar ve süpermarket konseptinin avantajlarından bahsetmişlerdir [3].

Dong ve diğerleri çekici trenlerin sayısını ve tur süresini en küçüklemek amacıyla çizelgeleme problemini ele almışlardır. Problemin çözümü için bir matematiksel model ve polinom zamanda çalışan dinamik programlama algoritması sunmuşlardır [29]. Emde ve Gendreau ise aynı problemi ele almışlar, çekici trenlerin durma zamanları ve istasyon kapasiteleri dikkate alındığında problemin NP-complete olduğu göstermişler ve problemin çözümü için bir tabu arama algoritması önermişlerdir [30].

Yukarıda görüldüğü gibi SYP için literatürde farklı çözümler önerilmiştir. Problem en temel haliyle polinom zamanda çözülebildiği için dinamik programlama algoritması kullanılmıştır. Fakat fabrika zemininde uygun olmayan bölgeler, süpermarket kapasitesi, değişken istasyon talepleri gibi yeni parametreler probleme eklendiğinde, problem polinom zamanda çözülememektedir. Bu nedenle çözüm için modelleme araçları ya da GA ve dağılım tahmini algoritması gibi popülasyon tabanlı meta-sezgisel algoritmalar önerilmiştir. Bu çalışmada problem değişken kapasiteli süpermarketler kullanılarak farklılaştırılmış ve tek çözüm tabanlı algoritmaların da çözümde etkili olabileceği düşünülerek problemin çözümü için TB algoritması önerilmiştir.

2.4. Karar Destek Sistemleri

Bu bölümde KDS ile ilgi genel kavramlara, KDS bileşenleri, özelikleri, türleri ve çalışma kapsamında geliştirilen KDS türü olan model sürümlü ve optimizasyona dayalı KDS hakkında bilgi verilmiştir.

2.4.1. Karar destek sistemi kavramı

KDS'nin temelleri 1960'lı yıllarda atılmaya başlamıştır. Bu konudaki ilk kavramlar 1971'de M. Scott Morton tarafından "Management Decision Systems (Yönetim Karar Sistemleri) başlıklı bir yazıda ele alınmıştır [31]. Daha sonra gerek akademik, gerekse endüstriyel alanlarda araştırma-geliştirme ve uygulama çalışmaları hızla yayılmıştır.

KDS geniş kapsamlı olarak insanların bilgisayar iletişimlerini, verilerini, belgelerini, bilgilerini ve modellerini sorunları çözmek ve karar vermek için kullanmalarına yardımcı olan etkileşimli bilgisayar tabanlı sistemler olarak tanımlanır. KDS bağımlı ya da yardımcı sistemlerdir; karar vericinin yerine geçmek amacıyla tasarlanmazlar. Buna bağlı olarak karar verme sürecini otomatikleştirmezler [32].

Karar vermeyi iyileştirme amacıyla, karar ekiplerine ve bireysel karar vericilere yardım etmek için birçok farklı bilgisayar temelli KDS geliştirilmiştir. Bazı sistemler doğrudan yöneticilere yapısal bilgi sağlar. Diğer sistemler, yöneticilere ve personel uzmanlarına çeşitli kantitatif modeller kullanarak durumları analiz etmeye yardımcı olur. Bazı KDS bilgi depolar ve yöneticilere sunar. Bazı sistemler küçük ve büyük gruplar tarafından karar vermeyi destekler. Şirketler, müşterilerinin ve tedarikçilerinin karar vermelerini desteklemek için bile KDS geliştirmektedir [32].

KDS özeliklerine bakılarak daha iyi anlaşılabilir [33]:

- Üst düzey yöneticilerin karşılaştıkları yapısallığı düşük, belirsiz problemleri hedefleme eğilimindedirler;
- Modellerin veya analitik tekniklerin kullanımını geleneksel veri erişimi ve bilgi çıkarım fonksiyonlarıyla birleştirmeye çalışırlar;
- Özellikle bilgisayar yatkınlığı düşük olan insanlar tarafından kullanımını kolaylaştıran etkileşimli özelliklere odaklanırlar;
- Ortamdaki değişiklikleri ve kullanıcının karar verme yaklaşımını karşılamak için esneklik ve uyarlanabilirlik üzerinde dururlar.

2.4.2. Karar destek sistemi bileşenleri

Karar destek sitemleri genel olarak, veri yönetim sistemi, model yönetim sistemi, bilgi motoru, kullanıcı arayüzü ve kullanıcılar olarak beş bileşene ayrılabilir [34].

Veri yönetim sistemi: Açık verileri özel kararlar için depolar ve organize eder. Ek olarak, KDS kullanımı ile ilgili güvenlik fonksiyonlarını, veri entegre prosedürlerini ve genel veri yönetim görevlerini yerine getirir. Bu görevler veri tabanları, veri tabanı yönetim sistemi,

veri havuzu ve veri sorgulama sistemleri gibi veri yönetim bileşeni içinde alt sistemler ile sağlanır.

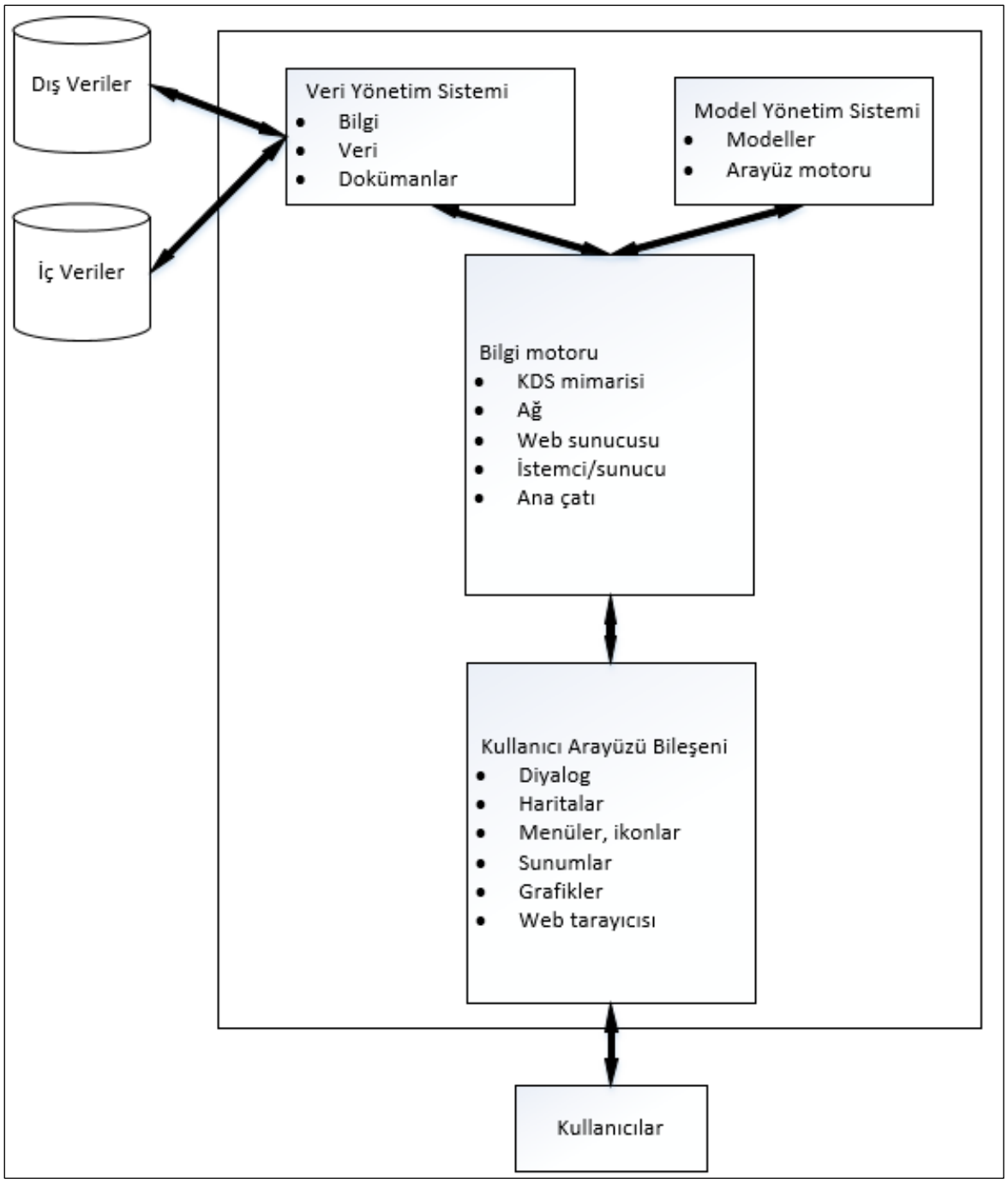
Model yönetim sistemi: KDS için analitik yetenekleri sağlayan niceliksel modeller ile ilişkili erişim, depolama ve örgütsel faaliyetleri yürütür. Model tabanı, model yönetim sistemi, model havuzu, model yürütme işlemcisi ve model sentez işlemcisini kapsar.

Bilgi motoru: Sistemin beyni olarak tanımlanabilir. Veri ve model, karar konusunu destekleyecek şekilde bu bileşende işlenir. Problemi tanıma ve çözme işlemleri ile ilişkili bütün fonksiyonları yürütür.

Kullanıcı arayüzü: Fonksiyonel bir KDS için bu bileşenin tasarımı ve işlevselliği anahtar bileşendir. Bu bileşen sayesinde KDS'nin veri, model ve bilgi motoruna kolayca erişilebilmeli ve düzenlenebilmelidir.

Kullanıcılar: Başarılı bir KDS için sistemin kullanıcılıların beceri ve motivasyonları, bilgi sahası ve organizasyondaki rolleri dikkate alınmalıdır.

Şekil 2.5'te KDS bileşenleri örnek olarak gösterilmiştir.



Şekil 2.5. Temel KDS bileşenleri [32]

2.4.3. Karar destek sistemi türleri

KDS türleri aşağıdaki gibi sınıflandırılabilir [32,35].

Model Sürümlü KDS: Finansal, optimizasyon veya simülasyon modellerine erişim ve düzenlemeyi vurgular. Basit kantitatif modeller, en temel işlevsellik seviyesini sağlar. Model

sürümlü KDS karar vericilerin bir durumu analiz etmelerine yardımcı olmak için karar vericiler tarafından sağlanan sınırlı veri ve parametreleri kullanırlar, ancak genel olarak model sürümlü KDS için büyük veri tabanlarına ihtiyaç duyulmaz.

Veri Sürümlü KDS: Genel olarak, Veri Sürümlü KDS bir şirketin belirli bir dönemine ait dış ya da gerçek zamanlı verilerine erişim ve düzenlemeyi vurgular. Sorgu ve erişim araçlarıyla erişilen basit dosya sistemleri en temel işlevsellik seviyesini sağlar. Verilerin belirli bir göreve göre ayarlanmış bilgisayarlı araçlar veya daha genel araçlar ve operatörler tarafından değiştirilmesine izin veren veri ambarı sistemleri ek işlevsellik sağlar. Çevrimiçi analitik işlemlere (Online analytical processing) sahip veriye dayalı KDS, büyük tarihsel veri koleksiyonlarının analizine bağlı olarak en üst düzeyde işlevsellik ve karar desteği sağlar.

İletişim Sürümlü KDS: Kararla ilgili işbirliği ve iletişimi kolaylaştırmak için ağ ve iletişim teknolojilerini kullanır. Bu sistemlerde, iletişim teknolojileri baskın mimari bileşendir. Kullanılan araçlar arasında grup yazılımı, video konferans ve bilgisayar tabanlı bülten panoları bulunur.

Doküman Sürümlü KDS: Doküman alımı ve analizi sağlamak için bilgisayar depolama ve işleme teknolojilerini kullanır. Büyük doküman veri tabanları, taranmış belgeleri, köprü metni belgelerini, görüntüleri, sesleri ve videoyu içerebilir. Dokümanlara dayalı bir KDS ile erişilebilecek belgelere örnek olarak politikalar ve prosedürler, ürün özellikleri, kataloglar ve toplantı tutanakları ve yazışmalar dâhil olmak üzere kurumsal tarihî belgeler verilebilir.

Bilgi Sürümlü KDS: Yöneticilere eylem önerebilir. Problem çözme uzmanlığına sahip kişisel bilgisayar sistemleridir. Uzmanlık, belirli bir alan hakkında bilgi, o alandaki sorunların anlaşılması ve bu sorunların bazılarını çözme becerisinden oluşur. Bu sistemler Öneri KDS ve Bilgi Tabanlı KDS olarak adlandırılmıştır.

Bu çalışma kapsamında geliştirilen KDS, TB algoritmasını model olarak kullanarak, bir optimizasyon problemini çözdüğü ve yöneticileri karar desteği sağladığı için Model Sürümlü KDS sınıfına girmektedir. Bu sebeple aşağıda Model Sürümlü KDS ve alt türü olan Optimizasyona Dayalı KDS ve bu sistemlerde kullanılan meta-sezgisel algoritmalar ayrı başlıklar olarak ele alınacaktır.

2.4.4. Model sürümlü KDS

Model güdümlü ya da modele dayalı KDS olarak da adlandırılırlar. Model sürümlü KDS (Model-Driven DSS), muhasebe ve finansal modeller, temsil modelleri ve optimizasyon modellerini kullanan sistemleri içerir. Matematiksel ve analitik modeller, modele dayalı bir KDS'nin ana bileşenidir. KDS modelleri doğrudan yöneticiler ve alan uzmanları tarafından kullanılmalı ve değiştirilmelidir. Her model sürümlü KDS'nin belirli bir amacı vardır ve bu nedenle farklı modellere ihtiyaç duyulur ve kullanılır. Uygun modelleri seçmek kilit bir tasarım konusudur. Ayrıca, belirli modeller oluşturmak için kullanılan yazılımın gerekli verileri ve kullanıcı arayüzünü yönetmesi gerekir. Model sürümlü KDS'de, temel değişkenlerin veya parametrelerin değerleri, arz, üretim, ekonomi, satış, pazar, maliyetler veya diğer çevresel ve iç faktörlerdeki potansiyel değişiklikleri yansıtacak şekilde sıklıkla tekrar tekrar değiştirilir. Modellerden gelen bilgiler daha sonra karar verici tarafından analiz edilir ve değerlendirilir [32].

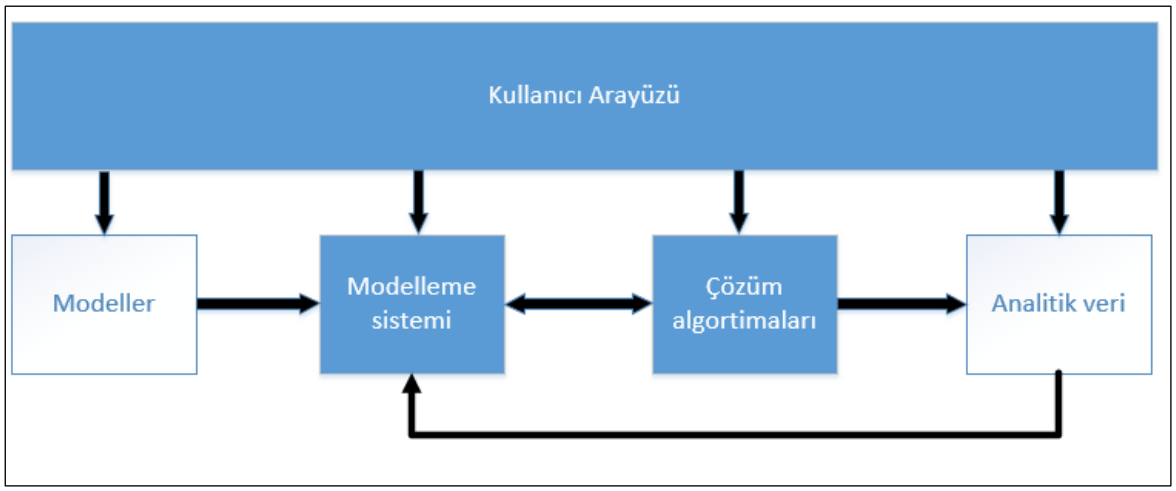
Tanım olarak, bir veya daha fazla niceliksel model, modele dayalı bir karar destek sisteminin birincil işlevselliğini sağlayan baskın bileşenlerdir. Ayrıca, tanım gereği, kullanıcının çıktıların hassasiyetini inceleyebilecek şekilde parametreleri düzenleyebildiği ve daha özel bir "farz edelim" analizi yapabileceği şekilde tasarlanırlar. İki özellik, modele dayalı bir KDS'yi karar analitiği için bilgisayar desteğinden veya yöneylem araştırması özel karar çalışmasından ayırmaktadır. Birincisi modele dayalı bir KDS'deki model, kullanımı kolay bir arabirim aracılığıyla yönetici gibi teknik olmayan bir uzman tarafından erişilebilir duruma getirilir. İkincisi, özel bir modele dayalı KDS aynı veya benzer durumlarda tekrar kullanılabilmek amacıyla tasarlanırlar. Modele dayalı KDS'de kullanılan genel nicel model türleri, cebirsel ve diferansiyel denklem modellerini, analitik hiyerarşi süreci, karar matrisi ve karar ağacı, çok özellikli ve çok kriterli modeller, tahmin modelleri, ağ ve optimizasyon modelleri, ayrık olay simülasyon modelleri ve çok ajanlı simülasyonlar için kantitatif davranış modelleri dahil olmak üzere çeşitli karar analiz araçlarını içerir. Modele dayalı bir KDS'deki modeller, karar vericinin anlayabileceği bir durumun basitleştirilmiş bir sunumunu sağlamalıdır [36].

Model sürümlü KDS'lerin çoğu çeşitli karar problemleri ile ilgili optimizasyon problemlerini çözmek için tasarlanırlar ve optimizasyona dayalı KDS olarak adlandırılırlar [36].

2.4.5. Optimizasyona dayalı KDS

Pratik optimizasyon, mühendislik, bilgisayar bilimi ve birçok endüstri alanındaki uygulama disiplinlerinde giderek daha önemli bir rol oynamaktadır. Optimizasyon modellemesi ve çözüm teknikleri, altmışlı yılların başından bugüne kadar sürekli gelişmeler görmüştür. Ana başarılar, model kavramsallaştırmalarındaki, çözüm algoritmalarındaki ve yazılım tekniklerindeki gelişmelerden gelmiştir. Daha yakın bir zamanda, 1990'lar boyunca, analitik veriler, optimizasyon modelleri ve güçlü çözücü motorlar arasındaki önemli bağlantılar, büyüyen bir araştırma ve geliştirme alanı olarak ortaya çıkmıştır [37]. Bu alandaki gelişmeler, daha güçlü bilgisayarların kullanılabilirliği ile birleştiğinde, finans, üretim ve tedarik zinciri yönetimi, enerji ve kamu hizmetleri de dahil olmak üzere çeşitli sanayi sektörlerinde karar destek sisteminde optimizasyonun bir çıkarım motoru olarak kabul edilmesinde etkili olduğunu kanıtlamıştır. Bu alanlarda, modern optimizasyon araçları sadece mevcut kurumsal bilgi sistemlerine başarılı bir şekilde dahil edilmemiş, aynı zamanda kurumsal iş süreçlerinin ayrılmaz bir parçası olmuştur. Sonuç olarak, karar teknolojisi iş süreçleri yönetiminde artan önemde rol oynamaktadır [38].

Model sürümlü karar desteği üç aşamaya ayrılabilir: formülasyon, çözüm ve analiz. Formülasyon, bir model çözücünün kabul edebileceği bir biçimde bir model üretilmesini ifade eder. Çözüm aşaması, modelin algoritmik çözümünü ifade eder. Analiz aşaması, “farz edelim” analizini ve bir model çözümünün veya bir dizi çözümün yorumunu ifade eder. Bu üç aşamayı destekleyecek KDS araçlarının geliştirilmesi farklı oranlarda gerçekleşmiştir. Geleneksel olarak daha iyi bir çözüm algoritması oluşturmaya odaklanan optimizasyon araştırmaları; teknoloji geliştikçe, KDS’nin formülasyonu ve analiz fonksiyonlarında daha fazla ilerleme kaydedilmiştir [39]. Özel bir model sürümlü KDS olan optimizasyona dayalı KDS’de kullanıcı, arayüz aracılığıyla modelleme sistemine ve çözüm algoritmalarına müdahale edebilir. Şekil 2.6’da optimizasyona dayalı KDS’lerin genel görünümü gösterilmiştir.



Şekil 2.6. Optimizasyona dayalı KDS'lerin genel görünümü [38]

Tarihsel olarak, yöneylem araştırmasında, araştırma çabalarının çoğu sorunları daha hızlı çözmek için yeni algoritmaların geliştirilmesine odaklanmıştır. Karar destek yazılımı geliştiricileri, kullanıcının bu geliştirmelerden faydalanmasını sağlamak için çözüm algoritmalarındaki gelişmeleri oldukça hızlı bir şekilde birleştirmiştir. Doğrusal programlama, tam sayılı programlama gibi yöntemlerin yanında, meta-sezgisel algoritmaların yöneylem araştırmasında kullanımı hızla artmaktadır. Bu teknikler tabu araması, genetik algoritmalar, tavlama benzetimi, yapay sinir ağları vb. yöntemleri içermektedir. Yapay zeka ve yöneylem araştırması tekniklerinin birlikte kullanılması önümüzdeki yıllarda KDS gelişimine büyük fayda sağlayacaktır [39].

2.4.6. Optimizasyonda kullanılan metasezgisel yöntemler

Optimizasyon problemi, istenilen hedefe ulaşmak için uygulanabilir çözümler arasından en iyisini seçmek olarak tanımlanabilir. Günümüzde sayısız optimizasyon uygulamaları vardır. Her işlem optimize etmek için bir potansiyele sahiptir. Optimizasyon problemi ile karşılaşmamış hiçbir firma yoktur. Gerçekten de, bilim ve endüstri alanındaki birçok zorlu uygulama, optimizasyon problemleri olarak formüle edilebilir. Optimizasyon, zaman, maliyet ve riskin en aza indirilmesi veya kar, kalite ve verimin en üst düzeye çıkarılması ile gerçekleşir. Örneğin, maliyet ve hizmet kalitesini optimize etmek için bir ağ tasarlamanın birçok yolu vardır; zamanı optimize etmek için bir üretimi planlamanın birçok yolu vardır; potansiyel enerjiyi optimize etmek için bir proteinin üç boyutlu yapısını tahmin etmenin birçok yolu vardır [40]. Her gün, mühendisler ve karar vericiler, çeşitli teknik alanlarda

ortaya çıkan karmaşıklığı sürekli artan problemlerle karşı karşıya kalmaktadır. Yöneylem araştırması, mekanik sistemlerin tasarımı, görüntü işleme ve özellikle elektronik alanlar bunlara örnektir. Çözülecek problem genellikle bir optimizasyon problemi olarak ifade edilebilir. Burada, ilgili tüm parametreler karşısında en küçüklemek ya da en büyüklemek istenen bir amaç veya maliyet fonksiyonu tanımlanabilir. Optimizasyon probleminin tanımı genellikle kısıtlar hakkındaki bilgi ile desteklenir. Kabul edilen çözümlerin tüm parametreleri bu kısıtları sağlamalıdır, aksi takdirde bu çözümler gerçekleştirilemez [41].

Bazı optimizasyon problemlerinin karmaşıklığı bütün çözümlerin araştırılmasını imkânsız kılar. Metasezgisel yöntemler, deneme yanılma yöntemiyle karmaşık bir probleme, makul sürelerde kabul edilebilir çözümler bulmaya yarar. En iyi çözümün bulunabileceği ya da algoritmanın işe yarayıp yaramayacağının bir garantisi yoktur. Buradaki fikir, çoğu zaman işe yarayacak ve kaliteli çözümler üretebilecek, etkin fakat pratik bir algoritmaya sahip olmaktır. Bulunan çözümler arasında bazılarının neredeyse en iyi olması beklenir, ancak bunun bir garantisi yoktur [42].

Metasezgiseller son yirmi yılda daha fazla popülerlik kazanmıştır. Birçok faklı uygulamada kullanılması, büyük ve karmaşık problemlerin çözümünde verimlilik ve etkinliklerini gösterir. Endüstri ve hizmetlerdeki karmaşık sorunları çözmek için, finanstan üretim yönetimine ve mühendisliğe kadar uzanan alanlarda yaygın olarak kullanılmaktadır. Bazı uygulama alanları; mühendislik tasarımı ve optimizasyonu, elektronik, otomotiv ve robotik, makine öğrenmesi ve veri madenciliği, sistem modellemesi ve simülasyonu, görüntü işleme, planlama ve planlama problemleri, rotalama problemleri, tedarik zinciri yönetimi problemleri ve lojistik ve taşıma şeklinde özetlenebilir [40].

Metasezgisellerin özellikleri aşağıdaki gibi özetlenebilir [43]:

- Metasezgiseller arama sürecine rehberlik eden stratejilerdir.
- Hedefleri arama uzayını etkili bir şekilde araştırarak (neredeyse) en iyi çözümler bulmaktır.
- Metasezgisellerde kullanılan teknikler yerel aramadan karmaşık öğrenme süreçlerine kadar çeşitlilik gösterir.
- Metasezgisel algoritmalar yaklaşıktır ve genellikle deterministik değildir.
- Arama uzayının sınırlı alanlarında sıkışıp kalmamak için mekanizmalar içerebilirler.

- Metasezgiseller problemlere özel değillerdir.
- Metasezgiseller alana özgü bilgiyi, üst düzey strateji ile kontrol edilen sezgisel tarama biçiminde kullanabilir.
- Günümüzde daha gelişmiş metasezgiseller arama sürecine rehberlik ederken arama tecrübesi kullanırlar.

Metasezgiseller literatürde çeşitli şekilde sınıflandırılmaktadır. En çok kullanılan sınıflandırma şekilleri aşağıdaki gibidir [40,43]:

- Tabiattan esinlenilenler/tabiattan esinlenilmeyenler: Metasezgisellerin çoğu tabiattan esinlenilerek geliştirilmişlerdir. Evrimsel algoritmalar ve yapay bağışıklık sistemleri biyolojiden esinlenmiştir. Karınca kolonisi optimizasyonu karıncalardan, arı kolonileri algoritması arılardan ve kuş sürüsü optimizasyonu kuşlardan ilham alınarak geliştirilmiştir. TB fizik kurallarından yola çıkılarak oluşturulmuştur. Tabu araması ya da yerel arama ise tabiattan esinlenmeyen metasezgisellere örnektir.
- Hafıza mekanizmasına sahip olanlar/hafıza mekanizmasına sahip olmayanlar: Bazı metasezgisellerin geçmiş arama bilgilerini tutan hafıza yapıları bulunmamaktadır. Örneğin, yerel arama, GRASP ve tavlama benzetimi bunlar arasındadır. Çözüm uzayında arama gerçekleştirilirken iterasyonlar boyunca arama geçmişini tutan gelişmiş hafıza mekanizmalarına sahip metasezgiseller de vardır. Örneğin, uzun dönem hafıza ve kısa-dönem hafıza yapılarına sahip tabu araması algoritması bunlar arasındaki en önemlisidir.
- Dinamik amaç fonksiyonlu/statik amaç fonksiyonlu: Bazı metasezgiseller arama sürecinde amaç fonksiyonunu hiç değiştirmeden kullanırken, yönlendirilmiş yerel arama gibi bazıları, yerel en iyilerden kurtulmak için arama sürecinde amaç fonksiyonunu değiştirir.
- Deterministik/stokastik: Deterministik yapıdaki metasezgiseller problemleri çözerlerken deterministik kararlar verirler. Yerel arama ve tabu araması bunlara örnektir. Stokastik metasezgiseller ise arama esnasında rastsal kurallar ile karar veriler. TB ve evrimsel algoritmalar bu sınıfa girer. Deterministik algoritmalarda aynı başlangıç çözümünden yola çıkılarak başlanan optimizasyon süreçlerinin sonunda daima aynı sonuç çözüm elde edilirken, bunun aksine stokastik metasezgisellerde farklı sonuç çözümler elde edilebilir.
- Tek-çözüm tabanlı/popülasyon tabanlı: Yerel arama, TB, tabu araması algoritması gibi tek-çözüm tabanlı metasezgiseller arama uzayını tek bir çözüm üzerinden ve her bir iterasyonda bu çözümün komşu çözümlerine geçerek araştırırlar. Genetik algoritmalar,

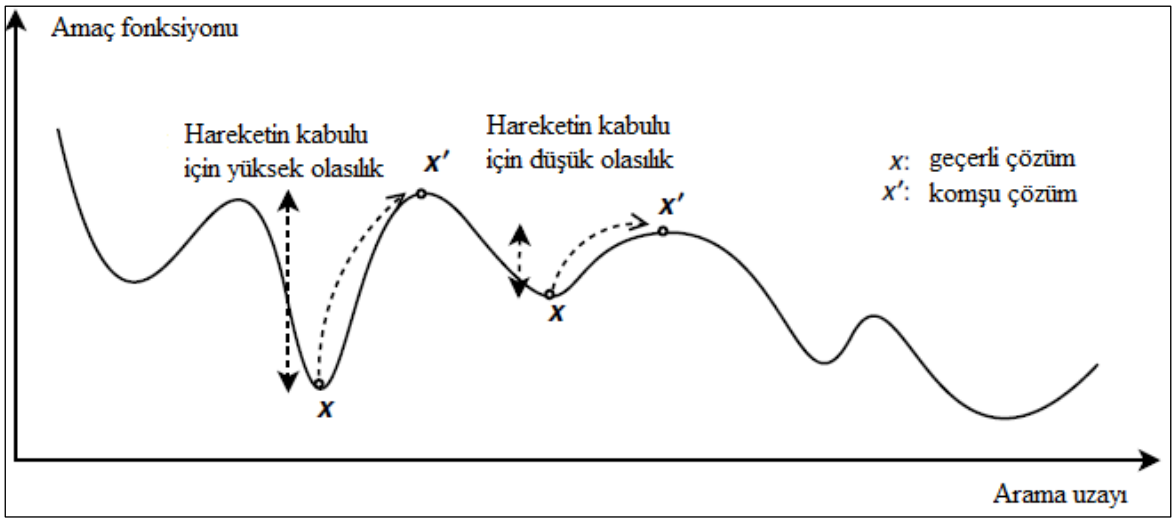
kuş sürüsü optimizasyonu ve evrimsel algoritmalar gibi popülasyon tabanlı metasezgiseller ise çoklu çözümler üzerinden araştırma yaparlar.

- İteratif/açgözlü: İteratif algoritmalar bir başlangıç çözümü ile arama prosedürüne başlarlar. Bu başlangıç çözümü tek bir çözüm veya popülasyon halinde çoklu çözümler de olabilir. Ardından başlangıç çözümü iterasyonlar boyunca iyileştirilerek en iyi çözüm aranır. Açgözlü algoritmalar ise boş çözümle arama prosedürüne başlarlar. Her bir adımda problemin bir karar değişkenine değer atanır ve bu şekilde adım adım gidilerek problemin tam çözümü elde edilir. Metasezgisellerin çoğu iterasyona dayanır.

Tavlama Benzetimi Algoritması

Tavlama benzetimi yöntemi aşırı ısıtılmış metallerin termal soğumasının simülasyonuna dayalı bir optimizasyon yöntemidir. Bir metal erime durumuna gelecek kadar ısıtıldığında, erimiş metaldeki atomlar serbestçe hareket eder. Sıcaklık azaltıldığında ise atomların hareketi kısıtlanır. Sıcaklık düştükçe atomlar düzenli hale girme eğilimindedir ve uygun sıcaklık profili kullanıldığında, son durumda kristaller en az olasılıksal içsel enerjiye sahiptirler. Kristallerin dizilimi soğuma oranına bağlı olarak değişmektedir. Hızlı soğutulduğunda metaller kristal durumuna gelemeyip, metalde kusurlar görülebilir. Metalin yüksek sıcaklıklardan yavaş soğutulması işlemine tavlama denilmektedir [44]. Tavlama benzetimi fikri Metropolis ve diğerleri (1953) tarafından katı metallerin tavlamasını taklit etmek için önerilmiştir ve Kirkpatrick ve diğerleri (1983) tarafından yinelemeli optimizasyon problemlerine uygulanmıştır [45].

Optimizasyon problemlerinde tavlama benzetimi yöntemi katı metallerdekine benzer şekilde uygulanmaktadır. Sıcaklık yüksek iken algoritmada farklı çözümler arasında serbestçe geçiş yapılabilmektedir. Yeni bulunan çözüm kötü sonuç verse dahi kabul edilebilmektedir. Sıcaklık azaldıkça yeni çözüm daha iyiyse kabul edilmekte, kötü çözümlerin kabul edilmesi ise olasılıksal olarak azalmaktadır. Kötü çözümlerin kabulü Boltzmann'ın olasılık dağılımına göre yapılmaktadır ($P(E)=e^{-E/kT}$). Termal dengedeki bir sistemin T sıcaklığındaki enerjisi (E) eşitliğine göre olasılıksal olarak dağıtılır. Burada P(E) enerji seviyesi, k Boltzmann sabiti olarak adlandırılır. Sistem yüksek sıcaklıklarda iken herhangi bir enerji durumunda olabilirken, sıcaklık azaldıkça yüksek enerji durumunda olma olasılığı çok azdır [44]. Şekil 2.7'de sıcaklık azaldıkça kötü çözümlerin kabul edilme olasılığının düştüğü gösterilmektedir.



Şekil 2.7. TB algoritmasında komşu çözümlerin değerlendirilmesi [40]

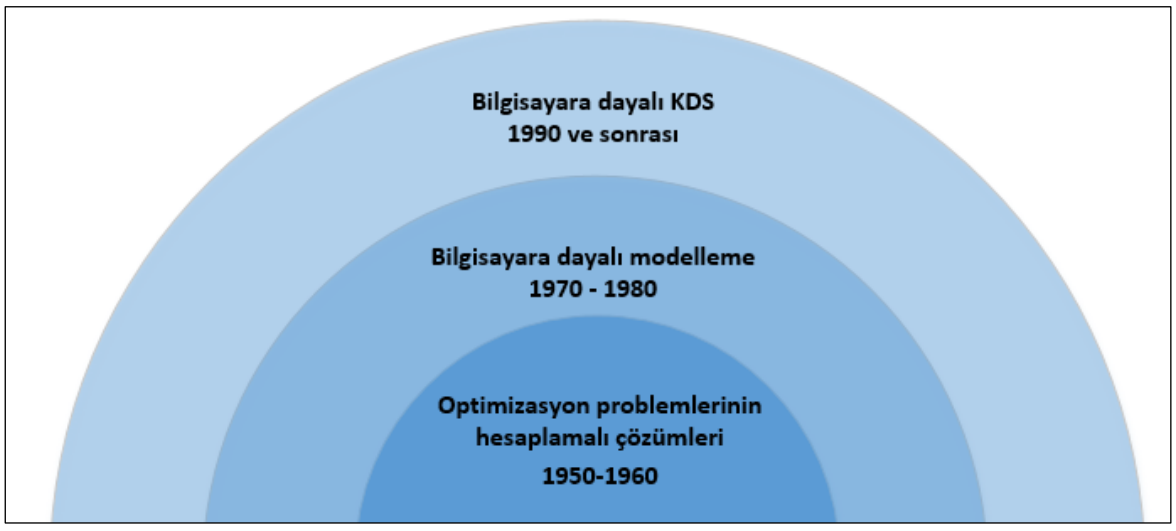
TB algoritması optimizasyon problemlerinin çözümünde sıklıkla kullanılmıştır ve son çalışmalardan bazıları şu şekildedir. Fathollahi-Fard ve diğerleri tarafından yapılan çalışmada, evde sağlık hizmeti lojistiği ele alınmıştır. Hasta bakıcıların eczanelerden hastaların evlerine yönlendirilmesi ve çizelgeleme problemi ile eczanelerin yerleşim problemleri için TB algoritması önermişlerdir [46]. Zaretalab ve Hajipour ise bazı kısıtlamalar altında alt sistemlere yedek bileşenler ekleyerek sistemin güvenilirliğini optimize etme amacı taşıyan yedekliğin kullanıldığı sistem güvenilirliği optimizasyonu problemi üzerine odaklanmıştır. Problemin çözümü için, iyi çözümlerin hakkında bilginin saklanarak tekrar algoritmayı beslemek için kullanıldığı ve bilgi temelli TB olarak adlandırdıkları bir yöntem kullanmışlardır [47]. Di Pace trafik kontrol stratejilerinin geliştirilmesi amacıyla bir çalışma yapmıştır. Trafik akışı gösterimi içi kullanılan iki trafik akış modelinin analizi ve karşılaştırması için TB algoritması kullanmıştır [48]. Xin ve Xiaodong ise arazi kaynakları mekansal optimizasyonu problemi için TB algoritması önermişlerdir. Çatışmalı hedeflerde paydaş tercihleri açısından ideal arazi kullanım modelini belirlemek için etkileşimli bir platform sağlamışlardır [49]. Zhang ve diğerleri ise rüzgâr ve güneş enerjisi, hidrojen ve aküler aracılığıyla depolanırken ortaya çıkan karar değişkenlerini TB algoritması ile optimize etmişlerdir [50]. Wang ve diğerleri araç rotalama probleminin özel bir çeşidi olan, müşterilerin belirli bir zaman aralığında eş zamanlı olarak malların alınıp teslim edilmesi problemi üzerine çalışmışlardır. Problemin çözümü için paralel TB algoritması önermişler ve farklı bir çalışmada GA ile elde edilmiş sonuçlarla karşılaştırmışlardır [51].

Yukarıdaki çalışmalarda görüldüğü gibi TB algoritması, farklı optimizasyon problemlerinin çözümünde sıklıkla kullanılmıştır. SYP için literatürde GA ya da dağılım tahmini algoritması gibi popülasyon temelli metasezgiseller önerilmiştir. Bu çalışmada komşuluk yapısı içeren ve en eski metasezgisellerden olup hala çok farklı problemlerin çözümünde kullanılan TB algoritması SYP için çözüm önerisi olarak sunulmuştur.

2.5. KDS ile İlgili Çalışmalar

KDS ile ilgili literatürde çok sayıda çalışmaya rastlanmaktadır. Bu çalışma kapsamında geliştirilen KDS model sürümlü ve optimizasyona dayalı olduğu için daha çok benzer çalışmalara yer verilmiştir.

Modelleme, problem çözme, veri yönetimi ve optimizasyon problemlerinin ele alınışları yazlım teknolojilerindeki gelişmelerle birlikte değişmiştir. Şekil 2.8 optimizasyon araçlarının tarihsel gelişimini şematik olarak göstermektedir. 1950 ve 1960'larda optimizasyon problemleri çözmek için daha çok hesaplama (bilgisayara dayalı) yoluna gidilmiştir. Yetmiş ve seksenlere gelindiğinde, yazılım teknolojilerindeki gelişmelerle birlikte optimizasyon problemlerinin bilgisayara dayalı modellenmesi mümkün hale gelmiştir. Doksanlarda ise bu problemler KDS ile çözülmeye başlanmıştır. O zamandan beri analitik veriyle olan iletişim önem kazanmıştır [38].



Şekil 2.8. Optimizasyon araçlarının tarihsel gelişimi [38]

Matematiksel programlamanın ilk uygulamalarında, FORTRAN gibi diller kullanılarak yazılmış geçici bilgisayar programları tarafından modeller üretilmiştir. Bu programlar, verilen bir doğrusal programlama (DP) / tam sayılı programlama (TP) modeliyle ilişkili matrisi oluşturmak için kullanılmıştır. Matrisi işlemek ve en uygun çözümleri rapor etmek için ise uygun bir çözüm algoritması kullanılmıştır [38]. Optimizasyon problemlerini çözmek için geliştirilen ilk karar destek sistemleri de çözüm aşamasında bu yöntemleri kullanmıştır. Örneğin Yang ve Mou Çin'de en büyük kimyasal boya tesislerinden birinde, entegre bir KDS geliştirerek uygulamıştır. KDS, DP tabanlı bir modelle kural tabanlı bir yapay zeka modeliyle birleştirmiştir. Operasyonel sonuçlar sistemin 1987 yılında, yıllık karını bir milyon ABD doları artırdığını göstermiştir [52]. Jager ve diğerleri ise Hollanda'da bir ilaç firmasında, tüm planlama süreci için aktif bileşenlerin ve ara maddelerin zaman aşamalı üretim miktarlarını belirlemek amacıyla bir KDS geliştirmişlerdir [53].

Diğer taraftan optimizasyon problemlerinin çözümünde metasezgiseller de büyük ilgi görmektedir. Örneğin Hamamoto ve diğerleri eczacılık sektöründe tesis yerleşim problemini çözmek için genetik algoritmaya dayalı bir simülasyon modeli sunmuşlardır [54]. Diğer bir örnek olarak Evolver (Palisade Yazılım) karmaşık tam sayılı problemleri GA ile çözmek için kullanılan herkese açık bir araçtır [39]. Juan ve diğerleri mevcut ofis binası koşullarını değerlendirmek ve yenileme maliyeti, iyileştirilmiş bina kalitesi ve çevresel etkiler arasındaki dengeleri göz önünde bulundurarak bir sürdürülebilir yenileme çalışmaları seti önermek için entegre bir karar destek sistemi geliştirmişlerdir. Grafik arama algoritması ile GA'yı birleştiren hibrit bir yaklaşım, olası tüm yenileme eylemlerini analiz ederek en iyi sonuca ulaşmak için kullanılmıştır [55].

Belirli optimizasyon problemleriyle ilgili literatür araştırmalarına da rastlanmaktadır. Örneğin Kolisch ve Padman proje çizelgeleme için kullanılan kesin ve sezgisel algoritmaları ve bu algoritmalar kullanılarak geliştirilmiş KDS ile ilgili bir araştırma yapmışlardır [56]. Ernst ve diğerleri ise, personel çizelgeleme ve görevlendirme amacıyla geliştirilen KDS ile detaylı bir literatür araştırması sunmuşlardır. KDS'lerde kullanılan matematiksel model ve kesin ya da sezgisel algoritmaları incelemişlerdir [57].

Metasezgisellerin ve klasik programlama yöntemlerinin birlikte kullanıldığı çalışmalara da rastlanmaktadır. Fikar ve diğerleri özel ve yardım kuruluşları arasında afet yardımı koordinasyonunu kolaylaştırmak için bir simülasyon ve optimizasyona dayalı karar destek

sistemi sunmuşlardır. KDS, felaketleri simüle etmekte ve etkilenen bölgedeki talep noktalarına transfer noktalarından yardım malzemesi gönderilerini planlamaktadır. Problemin çözümü için karışık tam sayılı programlama, sezgisel temelli rotalama ve çizelgeleme prosedürü ve tabu arama algoritması kullanmışlardır [58]. Sama ve diğerleri ise yoğun terminal alanlarında hava trafiğini optimize etmek için bir optimizasyona dayalı KDS önermişlerdir. Havaalanlarında kullanılan ilk giren ilk çıkar kuralını, rotalama ve çizelgeleme için kullandıkları dal sınır algoritması, tabu araması algoritması, karışık tam sayılı programla yöntemleriyle kıyaslamışlar ve optimizasyon algoritmalarının çok daha iyi sonuçlar verdiğini görmüşlerdir [59].

Model tabanlı KDS'lerin optimizasyon amaçlı birçok kullanımı varken, son zamanlarda getiri yönetimi ve tedarik zinciri yönetimi öne çıkmaktadır. Getiri yönetimi tahminleri ve mevcut satış seviyelerini analiz ederek hizmetlerin en iyi fiyat seviyelerini belirlemeyi amaçlar. Havayollarından konaklamaya, üretimden çalışma zamanı çizelgelemeye kadar model tabanlı KDS getiri yönetimi amacıyla kullanılmaktadır. Diğer yandan Kurumsal Kaynak Planlama uygulamaların artmasıyla birlikte, tedarik zinciri yönetimi karar desteği açısından önemli hale gelmiştir. Lojistik planlama, üretim planlama, talep yönetimi, fiyatlandırma kararları gibi tedarik zinciri yönetimi aşamaları için geliştirilmiş karar destek modelleri kullanılmaktadır [36].

3. KAPASİTESİZ SÜPERMARKET YERLEŞİM PROBLEMİ

Literatür incelendiğinde SYP ile ilgili az sayıda çalışma olduğu görülür. Bunlardan Emde ve Boysen probleme dinamik programlama algoritması ile en iyi çözüm önerirken, Alnahhal ve Noche problemi farklılaştırarak, süpermarket kapasitelerini ve işletme zemininde süpermarket kurulumu için uygun olmayan bölgeleri dikkate alarak GA kullanmışlardır [6,13]. Çalışmamızın ilk aşamasında problem Emde ve Boysen'in açıkladığı şekilde ele alınmış ve problemin çözümü için süpermarket kapasiteleri dikkate alınmadan TB algoritması kullanılmıştır [6].

3.1. Problem Tanımı

Karışık modelli montaj hatlarında süpermarket yerleşim problemi çözülürken karar verilmesi gereken değişkenler vardır. Her montaj istasyonunun (s) hangi süpermarketten kaç kutu parça (d_s) alması gerektiği belirlenmelidir. Montaj istasyonlarının ve aynı zamanda çekicilerin durma noktalarının fabrika zemini üzerinde koordinatları (a_s, b_s) hesaplanır. Bu koordinatlar sayesinde istasyon s ve s+1 arasındaki mesafe (e_s) ve aynı zamanda çekici tarafından ziyaret edilecek s istasyonu ile i süpermarketi arasındaki mesafeleri (z_{is}) metrik olarak kolaylıkla hesaplanabilir. Sonrasında SYP şu şekilde tanımlanabilir: En iyi süpermarket sayısı nedir ve hangi süpermarket hangi istasyonlara hizmet vermelidir? Bu sorular cevaplandıktan sonra süpermarket pozisyonları hesaplanır [6].

Süpermarket yerleşim problemi çözülürken dikkat edilmesi gereken hususlar vardır [6]:

- Bir süpermarket tarafından hizmet verilen istasyonlar ardışık olmalıdır. Örneğin bir süpermarket bir, üç ve beş istasyonlarına hizmet verirken başka bir süpermarket iki ve dört istasyonlarına hizmet veremez. Yerleşim yöneticileri çekici trenlerin rotalarının üst üste binmesini istemediği için bu durumdan kaçınırlar.
- Darboğazlar ek rotalar, araçlar ve daha sık turlar aracılığıyla her zaman aşılabileceği için bir süpermarket tarafından hizmet verilen istasyon sayısında ciddi kısıtlamalara gerek yoktur. Eğer bir süpermarketin mesafeleri ve beklenen parça talepleri uygunsa bu durum amaç fonksiyonuna yansıtılarak çözülebilir.
- Transfer raflarının doğurduğu gereklilikten dolayı parçalar eşit büyüklükteki standart kutularla paketlenmelidir.

- Her bir çekici tren için rota, çizelge ve yük ancak daha sonrasında üretim sırası bilindiğinde belirlenebilir. Bu nedenle SYP toplu tahminlere dayanarak çözülmelidir.
- Hızlı teslimat ve kısa rotalar asıl amaçları olduğu için süpermarketler istasyonlara yakın konumlandırılmalıdır. Tam olarak ne kadar yakın oldukları, istasyonların boyutlarına ve sürüş şeritleri için ayrılan alan miktarına bağlıdır.
- Parçalar, süpermarketlerden her bir istasyona noktadan noktaya teslim edilmek yerine, çoklu duruş noktaları içeren kapsamlı turlarla teslim edilir. Bu nedenle süpermarketler tüm istasyonlara değil sadece başlangıç ve bitiş istasyonlarına yakın olması yeterlidir. Böylelikle süpermarketler hizmet verdiği başlangıç ve bitiş istasyonlarının tam ortasına yerleştirilir.

Çekici trenler her bir istasyona ayrı ayrı uğramaktansa birden çok duruş noktasıyla belirli bir rota üzerinde hareket eder. Bu sebeple bir süpermarketten istasyona giden yol düz bir çizgi değildir, bunun yerine çekici trenin toplam dolaşacağı mesafe hesaplanmalıdır. Çekici tren önce rotasındaki ilk istasyona, sonra sırayla her bir istasyona ve sonunda yeniden yüklenmek üzere süpermarkete uğramalıdır. Bu durumda bile mesafeleri azaltmak her zaman iyi çözümleri garanti etmez. Çekicilerin belirli bir kapasitesi olduğundan, bir turda kaç tane ve hangi istasyonlara hizmet edileceği istasyonların parça talebine bağlıdır. Yüksek parça talebi olan istasyonların aynı rotada olması durumunda, çekici tüm istasyonların ihtiyacını karşılayamaz. Bundan dolayı uzaklıklar süpermarketin hizmet verdiği istasyonların toplam talebi ile ağırlıklandırılmalıdır. Son olarak, üçüncü bir bileşen olarak, bir süpermarket oluşturmak ve sürdürmek için sabit maliyet de hesaba katılmalıdır. Bunları göz önüne alarak ve Çizelge 3.1'deki gösterimi kullanarak, SYP aşağıdaki gibi tanımlanabilir [6].

$s = 1, \ldots, S$ süpermarket tarafından hizmet verilecek ardışık istasyonlar olsun. SYP bu istasyonların birbirinden ayrı bir süpermarket tarafından hizmet verilen, değişken alt gruplara bölünmesinden oluşur. Bir çözüm $X(n) = \{1, x_2, x_3, \ldots, x_n, S + 1\} \rightarrow \{2, \ldots, S\}$ şeklinde, x_i'in süpermarketin hizmet verdiği istasyonların en solundakini ve x_{i+1}-1'in en sağdakini gösterdiği bir vektör olarak gösterilebilir. Tüm istasyonlara bir süpermarket tarafından hizmet verilmesi gerektiği için ilk süpermarketin hizmet alanının en solundaki istasyon her zaman 1 ve sonuncunun en sağındaki istasyon da her zaman S'dir. Çözüm vektörün uzunluğu n+1'dir ve elemanları aşağıdaki amaç fonksiyonu ve kısıtlara uymalıdır [6].

En Küçükle $F(X(n))=\sum_{i=1}^{n}\sum_{s=x_i}^{x_{i+1}-1} d_s \cdot (\sum_{\tau=x_i}^{x_{i+1}-2} e_\tau + z_{i,x_i} + z_{i,x_{i+1}-1}) + n \cdot \Gamma$ (3.1)

Koşulları $x_{i+1} \geq x_i+1$ $\forall i= 1, \ldots, n$ (3.2)

Altında

$$z_{is} = |a_{x_i} + (a_{x_{i+1}-1} - a_{x_i})/2 - a_s| + |b_{x_i} + (b_{x_{i+1}-1} - b_{x_i})/2 - b_s| \quad \forall i= 1, \ldots, n;\ s=1, \ldots, S \quad (3.3)$$

Amaç fonksiyonu Eş. 3.1 n sayıdaki Γ ile ağırlıklandırılmış süpermarketleri ve süpermarketin hizmet alanındaki istasyonların toplam talepleri ile çekicinin dolaşacağı toplam mesafelerin çarpımlarının toplamını en küçüklemeyi amaçlamaktadır. Bir çekici trenin bir turunda dolaşacağı mesafe şu şekilde hesaplanır: Önce *i* süpermarketinden kendi hizmet alanındaki ilk istasyon olan x_i'ye kadar mesafe z_{i,x_i}, sonra istasyonlar arası mesafelerin toplamı olan $\sum_{\tau=x_i}^{x_{i+1}-2} e_\tau$ ve son olarak süpermarkete geri dönüş mesafesi olan $z_{i,x_{i+1}-1}$ toplanır. Eş. 3.2 hizmet alanlarının çakışmamasını sağlamaktadır. Eş. 3.3 ise her bir süpermarketten her bir istasyona olan mesafeyi hesaplamaktadır. Fabrika zeminleri genellikle çizgi halinde sürüş yolları ve keskin dönüşlerle tasarlandığı için mesafeleri ölçmek için kullanılacak en iyi yöntem Manhattan metriğidir. Çekiciler bir süpermarketten başlayarak tüm istasyonları uğradıktan sonra tekrar süpermarkete döneceği için, süpermarketler hizmet ettiği istasyonların başlangıç ve bitiş noktalarının ortasına yerleştirilir. Diğer yandan, bir montaj hattıyla çakışma ihtimalinden dolayı süpermarketler her zaman istasyonların tam ortasına yerleştirilemediği durumlar olabilir. Öyle olsa bile, süpermarketler her zaman istasyonlara eşit uzaklıkta ve belirli bir mesafede yerleştirilmesi gerektiği için bu durumun ayrıca modellenmesine gerek yoktur [6].

Çizelge 3.1. Matematiksel modelde kullanılan kısaltmaların gösterimi

S	İstasyon sayısı
n	Süpermarket sayısını gösteren değişken
Γ	Her süpermarket için sabit maliyet
e_s	*s* istasyonun *s*+1 istasyonuna olan mesafe
d_s	*s* istasyonunda her vardiyada tahmin edilen kutu talebi
a_s	*s* istasyonunun *x* koordinatı
b_s	s istasyonunun *y* koordinatı

Çizelge 3.1. (devam) Matematiksel modelde kullanılan kısaltmaların gösterimi

x_i	*i* süpermarketi tarafından hizmet verilen ilk istasyon
z_{is}	*i* süpermarketinden *s* istasyonuna olan mesafe

3.2. Kapasitesiz Süpermarket Yerleşim Problemi İçin Tavlama Benzetimi Yaklaşımı

TB algoritması önceden belirlenen bir başlangıç sıcaklığı ile başlar (T_0). Bir başlangıç çözümü oluşturulur (S_0) ve bu aynı zamanda geçerli çözüm (S_C) ve ilk en iyi çözüm olur (S_B). Başlangıç çözümünün amaç fonksiyonu değeri (E_0), geçerli çözüm (E_C) ve en iyi çözümün amaç fonksiyonu değeri olur (E_B). Geçerli çözümden (S_C) bir değişiklikle komşu çözüm (S_N) oluşturulur ve amaç fonksiyon değerleri hesaplanır (ΔE). Eğer komşu çözümün amaç fonksiyonu değeri (E_N), E_C'den daha iyiyse, yeni geçerli çözüm olarak kabul edilir. Ancak E_N E_C'den daha kötüyse, exp(-ΔE/T) (T geçerli sıcaklık) olasılığı ile S_N yeni S_C olarak kabul edilir, diğer durumda S_N reddedilir. Bu işlem Markov zinciri uzunluğu olarak adlandırılan bir kontrol parametresi (IT) kadar tekrarlanır. T parametresi bir soğutma fonksiyonu (T= T x α) ile yavaşça azaltılır ve T_{min} ulaşılacak son sıcaklıktır. TB algoritmasının kötü çözümleri kabul ederek yerel en iyi noktalardan kurtulma kapasitesi vardır [45]. Çalışmada önerilen TB algoritmasının pseudo kodu Şekil 3.1'de gösterilmiştir ve algoritmanın akış diyagramı Şekil 3.2'deki gibidir.

Giriş parametreleri: T_0,T_{min}, α, $T = T_0$;

Bir S_0 üret ve E_0'ı hesapla;

$S_C = S_0$, $S_B = S_0$, $E_C = E_0$, $E_B = E_0$;

Tekrar

sayaç = 0;

Tekrar

S_C'den bir S_N üret ve E_N'i hesapla;

$\Delta E = E_N - E_C$;

$\Delta E \leq 0$ ise $S_C = S_N$, $E_C = E_N$;

$\Delta E > 0$ ise [0, 1] arasında bir değişken üret (rnd)

$\exp(-\Delta E/T) >$ rnd ise $S_C = S_N$, $E_C = E_N$;

$E_C < E_B$ ise $S_B = S_C$; $E_B = E_C$;

sayaç = sayaç + 1;

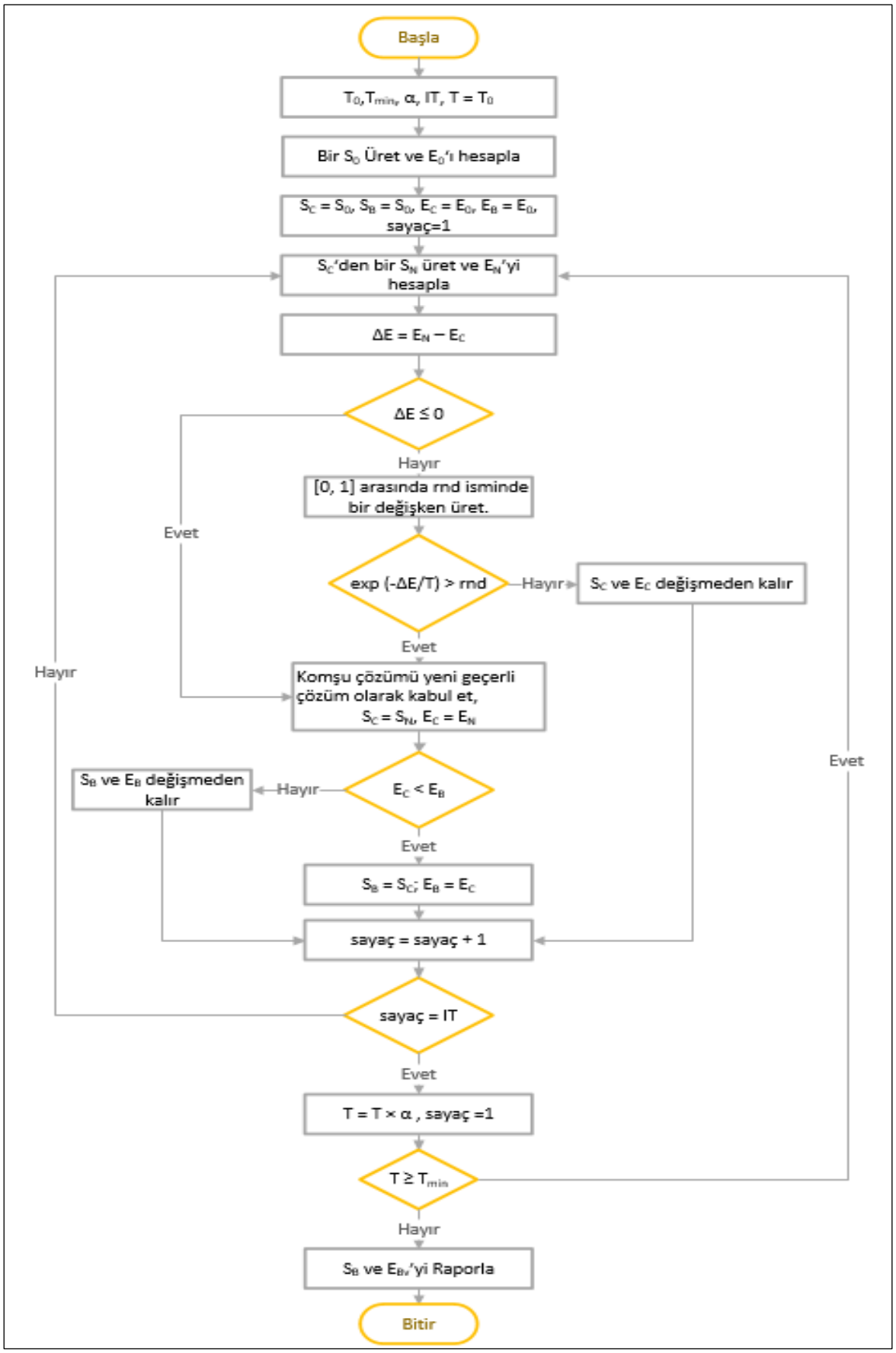
Kadar sayaç > IT

$T = T \times \alpha$;

Kadar $T < T_{min}$

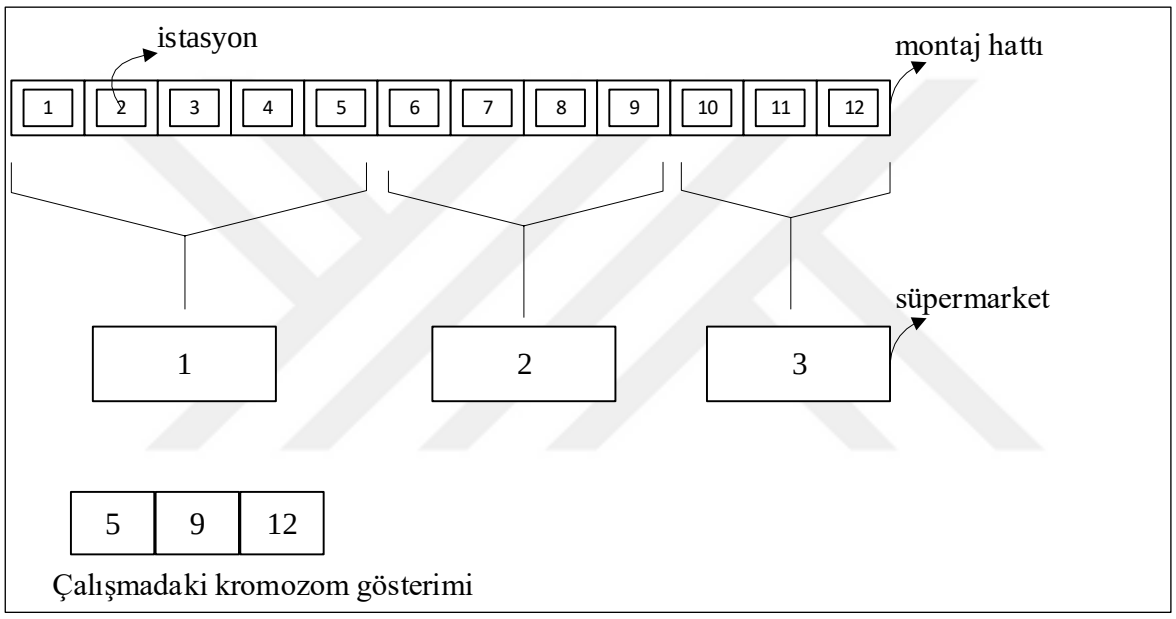
S_B ve E_B'yi raporla.

Şekil 3.1. TB algoritmasının pseudo kodu



Şekil 3.2. TB algoritmasının akış diyagramı

Çalışmada kolaylık olması açısından genetik algoritmada da kullanılan kromozom gösterimi kullanılacaktır. Benzer gösterim bir başka çalışmada da kullanılmıştır [13]. Şekil 3.3'te 12 istasyonlu bir montaj hattına hizmet veren üç süpermarket gösterilmektedir. İlk süpermarket 1'den 5'e kadar olan istasyonlara hizmet verirken, ikinci süpermarket 6'dan 9'a kadar, üçüncüsü ise 10'da 12'ye kadar hizmet vermektedir. Süpermarketler istasyonların hepsine ardışık olarak hizmet verdiği düşünülürse, her süpermarketin hizmet verdiği son istasyonu, {5, 9, 12} şeklinde ardışık olarak göstermek çalışmada kolaylık sağlayacaktır. Bu gösterimle hangi süpermarketlerin hangi istasyonlara hizmet verdiği kolaylıkla anlaşılacaktır.



Şekil 3.3. Çalışmadaki kromozom gösterimi [13]

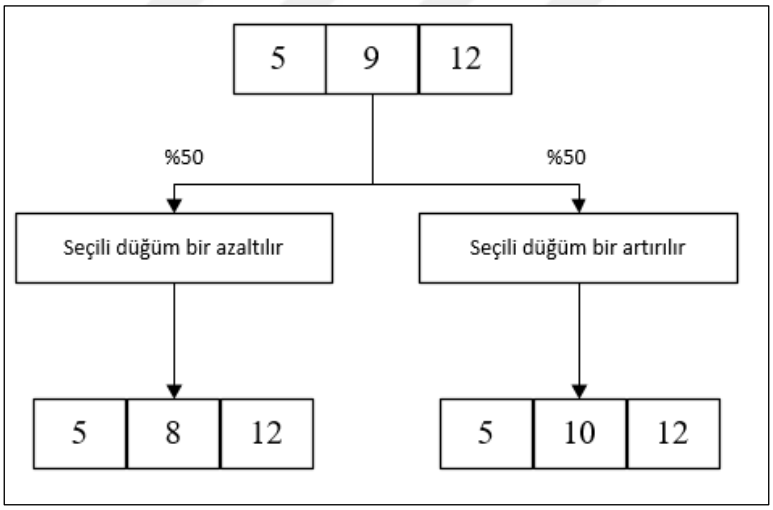
3.2.1. Başlangıç çözümü

SYP'yi TB algoritması ile çözmek için öncelikle bir başlangıç çözümü oluşturmak gerekir. Sonrasında komşu çözümler üretilerek, daha iyi çözümler araştırılır. Başlangıç çözümü oluşturmak için önce rastgele süpermarket sayısı belirlenmiştir. Herhangi bir çözümde en az bir, en fazla istasyon sayısı kadar süpermarket olabilir. Bu sebeple önce süpermarket sayısı bir ile istasyon sayısı arasında rastgele bir sayı olarak belirlenmiştir. Sonrasında her bir süpermarketin hizmet verdiği istasyonları belirlemek için yine bir ile istasyon sayısı arasında rastsal sayılar atanmıştır. Atama yapılırken aynı sayının birden fazla kullanılması engellenmiştir. Örneğin 12 istasyonu olan bir montaj hattı için 3 süpermarketli {5, 9, 12} bir başlangıç çözümü olabilir.

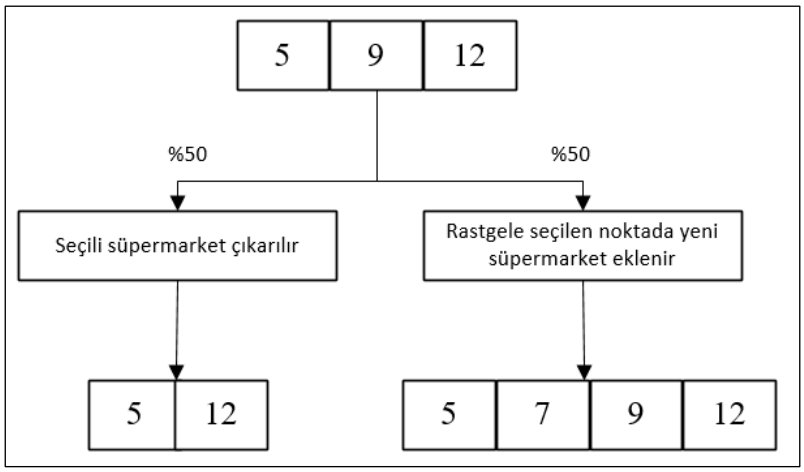
3.2.2. Komşu çözüm oluşturulması

12 istasyonu olan bir montaj hattı için üç süpermarketli {5, 9, 12} bir başlangıç çözümünü ele alalım. Bu çözümde bir değer değiştirilerek {5, 10, 12} şeklinde bir komşu çözüm üretilebilir. Çözüm iyileşmişse {6, 10, 12} şeklinde yeni bir komşu çözüm üretilebilir. Çözümdeki değerleri değiştirmek, süpermarket sayısı aynı kalarak hizmet verilen istasyonları değiştirmek demektir. Fakat farklı sayıdaki süpermarketler de daha iyi sonuç verebileceği için komşu çözümler üretilirken süpermarket sayısının da değiştirilmesi gerekir.

Bu çalışmada TB algoritmasında komşu çözüm üretilirken iki yöntem kullanılmıştır. Birincisi geçerli çözümde rastgele bir değer seçilerek %50 olasılıkla bu değer bir azaltılmış ya da artırılmıştır. İkinci olarak da %50 olasılıkla geçerli çözümden rastgele seçilmiş bir süpermarket çıkarılarak süpermarket sayısı bir azaltılmış ya da rastgele seçilmiş bir noktada çözüme yeni bir süpermarket eklemiştir. Bu iki yöntem de %50 olasılıkla rastsal olarak seçilmiştir. Şekil 3.4 ve Şekil 3.5 bu iki yöntemi {5, 9, 12} örnek çözümü üzerinde göstermektedir.



Şekil 3.4. Rastgele seçilen bir düğümün değerinin değiştirilmesi

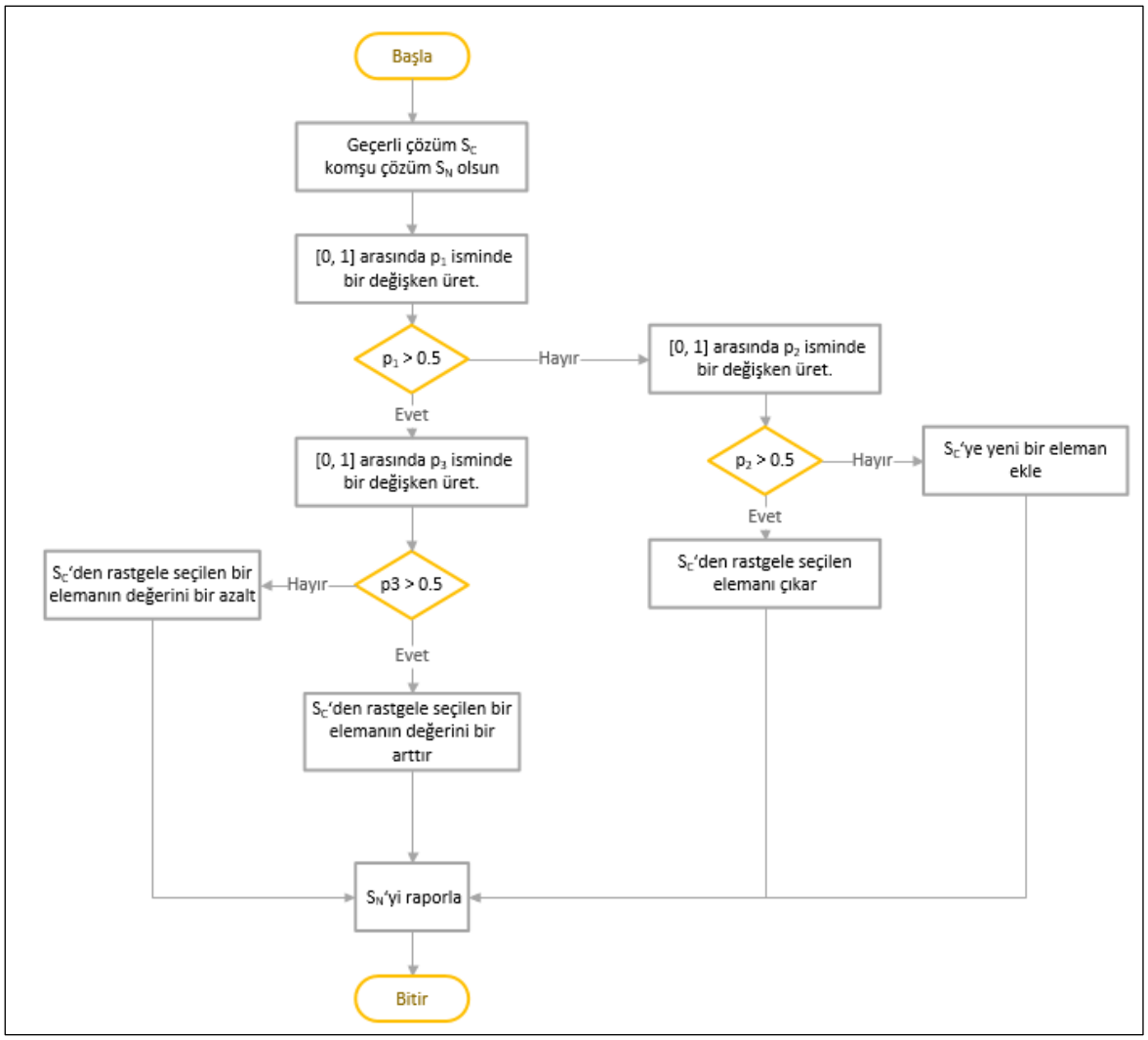


Şekil 3.5. Rastgele seçilen bir süpermarketin çıkarılması ve rastgele seçilen noktada yeni bir süpermarket eklenmesi

Çalışmada önerilen komşu çözüm algoritmasının pseudo kodu Şekil 3.6'da ve akış diyagramı Şekil 3.7'de gösterilmiştir.

Geçerli çözüm S_C ve komşu çözüm S_N olsun;
p1 isminde [0, 1] arasında rastgele bir değer üret;
p1 > 0,5 ise
  p2 isminde [0, 1] arasında rastgele bir değer üret;
  p2 > 0,5 ise S_C'den rastgele seçilmiş bir elemanı çıkararak yeni bir S_N oluştur;
  p2 ≤ 0,5 ise S_C'ye rastgele yeni bir elman ekleyerek yeni bir S_N oluştur;
p1 ≤ 0,5 ise
  p3 isminde [0, 1] arasında rastgele bir değer üret;
  p3 > 0,5 ise S_C'den rastgele seçilmiş bir elemanı bir artırarak yeni bir S_N oluştur;
  p3 ≤ 0,5 ise S_C'den rastgele seçilmiş bir elemanı bir azaltarak yeni bir S_N oluştur;
S_N'yi raporla;

Şekil 3.6. Komşu çözüm algoritmasının pseudo kodu



Şekil 3.7. Komşu çözüm algoritmasının akış diyagramı

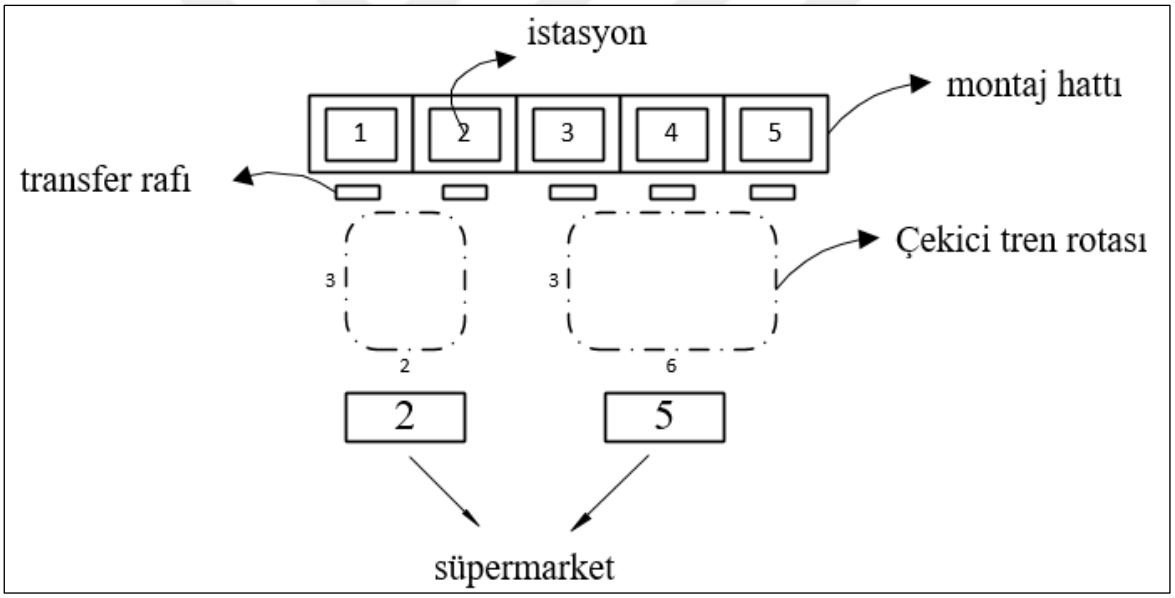
3.2.3. Örnek problem

Çizelge 3.2'deki örnek problem verisini düşünelim. İstasyon talepleri (d_s) 1 ile 10 arasında rastgele belirlenmiş, istasyonlar arası mesafe x ekseninde (a_s) 3 ile 6 arasında ve y koordinatları (b_s) 1 olan ve 5 istasyon içeren çizgi halinde bir montaj hattı olsun. Süpermarketlerin montaj hattından uzaklığı 3 birim ve yeni bir süpermarket açma maliyetini (γ) de 100 birim olarak belirleyelim.

Çizelge 3.2. Kapasitesiz SYP için örnek veri

İstasyon s	1	2	3	4	5
d_s	3	5	7	6	1
a_s	1	3	5	8	11
b_s	1	1	1	1	1

Örnek problemi önerilen algoritma ile çözmek için öncelikle bir başlangıç çözümü oluşturmak gerekir. Şekil 3.3'deki yöntemle gösterilen {2, 5} bir başlangıç çözümü olsun. Bu gösterim çözümde iki süpermarket olduğunu ve birinci süpermarket (2) 1 ve 2 numaralı istasyonlara hizmet verirken, ikinci süpermarketin (5) 3, 4 ve 5 numaralı istasyonlara hizmet verdiğini gösterir. Şekil 3.8 başlangıç çözümünü şematik olarak göstermektedir.



Şekil 3.8. Örnek problem için başlangıç çözümü

Eş 3.1'de gösterildiği şekilde tek bir süpermarketin maliyeti taşıma maliyeti ile süpermarket kurulum maliyeti toplanarak hesaplanır. Taşıma maliyeti ise; hizmet verilen istasyonların talepleri toplamının çekici trenlerin dolaşacağı toplam mesafenin çarpımı ile bulunur. Eş 3.3'e göre {2, 5} başlangıç çözümündeki ilk süpermarket (2), hizmet verdiği 1 ve 2 numaralı istasyonların tam ortasına yerleştirilir. Çekici trenin dolaşacağı toplam mesafe, süpermarketten hizmet verilen ilk istasyona olan mesafe, istasyonlar arası mesafe ve süpermarketten son istasyona kadar olan mesafenin toplamı ile bulunur. Eş 3.3 kullanılarak çekici trenin dolaşacağı toplam mesafe 4 + 2 + 4 = 10 olarak bulunur. 1 ve 2 numaralı

istasyonların talepleri toplam 3 + 5 = 8 olur. Bu durumda ilk süpermarket için taşıma maliyeti 8 x 10 = 80 olarak hesaplanır. Aynı şekilde ikinci süpermarket (5) de 3, 4 ve 5 numaralı istasyonlara hizmet vermektedir. Çizelge 3.2'deki koordinatlar kullanılarak ve Eş 3.3'e göre ikinci süpermarket ile hizmet verilen istasyonlar arasında dolaşacak çekici trenin alacağı mesafe 6 + 6 + 6 = 18 olur. İstasyonların talepleri toplamı ise 7 + 6 + 1 = 14 olarak hesaplanır. Bu durumda taşıma maliyeti 12 x 14 = 252 olur. Toplam taşıma maliyeti, iki süpermarket için taşıma maliyetleri toplamıdır: 80 + 252 = 332. Bu sayıya 2 süpermarket açma maliyeti eklenerek toplam maliyet bulunur: 332 + (100 x 2) = 532.

{2, 5} geçerli çözümünden Şekil 3.6'da gösterildiği gibi yeni komşu çözümler üretilerek maliyetin iyileşme durumuna bakılır. Örneğin çözüme rastgele bir süpermarket eklenerek oluşturulan {2, 4, 5} çözümün maliyeti aynı yöntemle 542 olarak hesaplanır. Maliyet kötüleştiği için belirli bir olasılık dâhilinde kabul edilir. Örneğin {2, 5} geçerli çözümünden elde edilebilecek bir komşu çözüm olan {1, 5} çözümünün maliyeti 636 olarak hesaplanır. {3, 5} çözümünde ise maliyet 494 olarak hesaplandığı ve iyileştiği için geçerli çözüm olarak kabul edilir. TB algoritması bu yöntemle komşu çözümler üreterek en iyi çözümü bulmaya çalışır. Sistem en az sıcaklığa ulaştığında, en az maliyeti veren çözüm algoritmanın sonucu olur. Örnek problem için en iyi çözüm {3, 5} ve maliyeti de 494 birimdir.

3.3 Deneysel Çalışma

Çalışmada önerilen algoritmanın etkinliği değerlendirmek amacıyla, Alnahhal ve Noche'un çalışmasında önerilen yöntemle bir dizi test problemi oluşturulmuştur [13]. İstasyonların talepleri 1 ile 10 arasında rastgele belirlenmiştir. İki istasyon arası mesafe x ekseni koordinatlarına (a_s) 3 ile 6 arasında rastgele değerler eklenerek elde edilmiştir. 20, 30, 40, 60, 100, 150 ve 200 olmak üzere farklı istasyon sayılarıyla (S) ve değişen sabit maliyetlerle (ɼ) 28 adet test problemi oluşturulmuştur. Her bir test problemi için de istasyon koordinatları ve talepleri değişen 4 farklı test problemi oluşturulmuştur.

Daha iyi çözümler elde edilebilmek için önerilen TB algoritmasının parametreleri ön deneylerle belirlenmiştir. T_0, T_{min}, α parametrelerinin etkin değerlerini bulabilmek için rastgele üç test problemi seçilmiştir. T_0 için üç seviye (500, 1000, 1500), T_{min} için üç seviye ("0,1", "1", "10") ve α için üç seviye ("0,9", "0,95", "0,99") belirlenmiş ve her parametre için her seviye üçer kez çalıştırılarak test edilmiştir. Elde edilen sonuçlar üzerinde, toplam

maliyet ve CPU zamanı açısından varyans analizi yapılmıştır. Ayrıca parametrelerin etkilerini açıkça göstermek ve bu parametrelerin etkin seviyelerini belirlemek için Duncan gruplamaları uygulanmıştır. Yapılan analizler sonucunda, T_{min} parametresi CPU zamanı açısından 10 olarak seçilmiş, T_0 ve α parametreleri ise maliyet ve CPU zamanı açısından sırasıyla 1000 ve 0,95 olarak belirlenmiştir. Uygulanan analiz sonuçları EK-1'de gösterilmiştir.

Çalışmada önerilen algoritma kullanılarak MS Visual Studio ortamında C# programlama dili ile bir uygulama geliştirilmiş ve tüm test problemleri 32 GB bellek ve Intel Xeon 3.7 Ghz işlemcili bir kişisel bilgisayar üzerinde çalıştırılmıştır. Her test problemi geliştirilen uygulama aracılığıyla beşer kez test edilmiş, çözümdeki ortalama süpermarket sayısı, ortalama maliyet ve ortalama CPU zamanı hesaplanmıştır.

Önerilen algoritmanın performansını kıyaslayabilmek için, birerleme yöntemiyle en iyi sonuca ulaşan bir uygulama daha geliştirilmiştir. Tüm test problemleri bu uygulama aracılığıyla çalıştırılmış ve 7200 saniyelik çalışma sonrasında sonuç alınamadığı durumda uygulama sonlandırılmıştır. Uygulama küçük ölçekli (20 ve 30 istasyon içeren) test problemlerinde en iyi çözüme ulaşabilmiş, büyük ölçekli (40 ve daha fazla istasyon içeren) test problemlerinde ise belirlenen sürede sonuç üretememiştir.

Çizelge 3.4'te her bir test problemi için, istasyon sayısı ve sabit maliyet aynı kalmak şartıyla geliştirilen 4 farklı veri setinin önerilen algoritma ile beşer kez çalıştırılması sonucu elde edilen ortalama süpermarket sayısı, ortalama maliyet, ortalama CPU zamanı ve birerleme yöntemi ile hesaplanan en iyi süpermarket sayısı ve en iyi maliyetlerin ortalaması gösterilmiştir. Çizelge 3.3'te, Çizelge 3.4'te kullanılan kısaltmalar verilmiştir.

Çizelge 3.3. Kapasitesiz SYP için önerilen TB algoritması uygulamasında kullanılan kısaltmaların gösterimi

S	İstasyon sayısı
r	Her süpermarket için sabit maliyet
Ort. n	Ortalama süpermarket sayısı
Ort. M.	Ortalama maliyet
T	TB algoritması CPU zamanı (saniye)

Çizelge 3.3. (devam) Kapasitesiz SYP için önerilen TB algoritması uygulamasında kullanılan kısaltmaların gösterimi

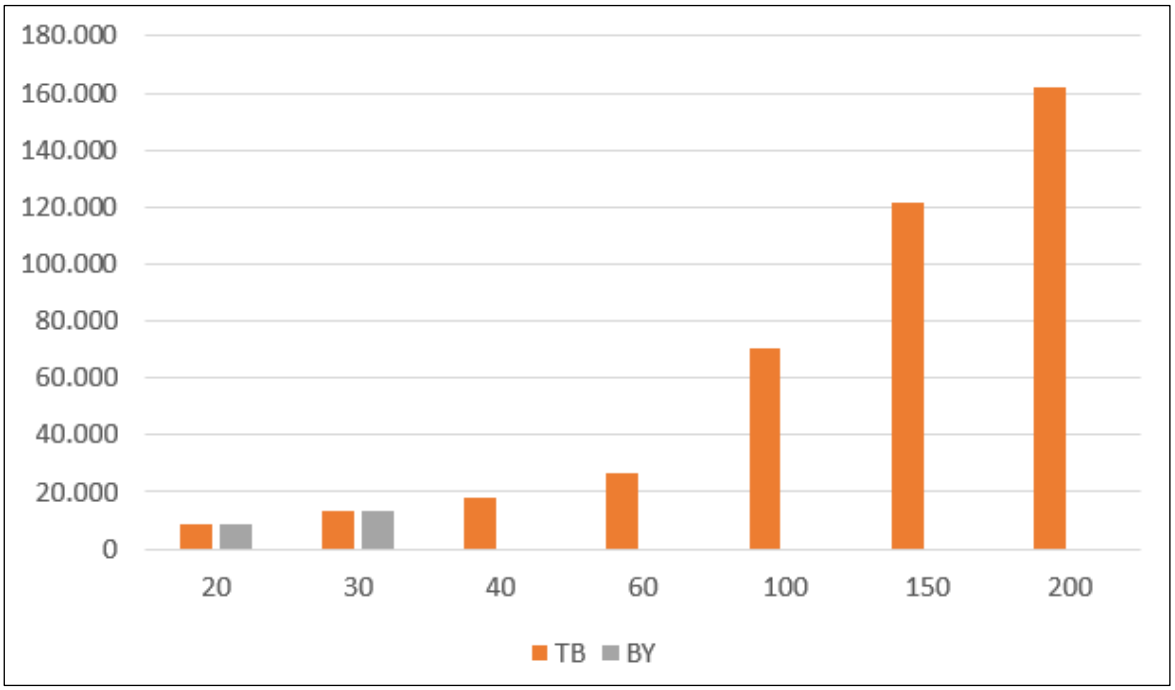
B. Ort. n	Birerleme uygulaması ortalama en iyi süpermarket sayısı
B. Ort. M.	Birerleme uygulaması ortalama en iyi maliyet
B. T	Birerleme uygulaması CPU zamanı (saniye)

Çizelge 3.4. Kapasitesiz SYP için önerilen TB algoritması uygulama sonuçları

No	S	r	Ort. n	Ort. M.	T (s)	B. Ort. n	B. Ort. M.	B. T(s)
1	20	500	5,75	4917,0	0,1	5,75	4917,0	0,4
2		1000	4,00	6955,5	0,1	4,00	6955,5	0,4
3		1500	3,50	9553,5	0,1	3,50	9553,5	0,4
4		3000	2,25	13079,0	0,1	2,25	13079,0	0,4
5	30	500	9,00	7476,0	0,2	9,00	7476,0	534,8
6		1000	6,25	10865,0	0,1	6,25	10865,0	530,8
7		1500	5,25	14293,0	0,1	5,25	14293,0	527,8
8		3000	4,00	20495,0	0,1	4,00	20495,0	527,5
9	40	500	11,50	9661,0	0,2			
10		1000	8,50	14512,5	0,2			
11		1500	7,00	18877,0	0,1			
12		3000	5,00	28538,0	0,1			
13	60	500	18,55	15016,1	0,3			
14		1000	12,60	21939,0	0,2			
15		1500	10,35	27627,8	0,2			
16		3000	7,50	41589,8	0,1			
17	100	500	28,65	24531,5	0,5			
18		1000	21,30	37341,6	0,4			
19		5000	9,40	90738,4	0,2			
20		10000	6,80	128725,6	0,2			
21	150	1000	30,45	55138,7	0,5			
22		3000	18,25	102473,1	0,3			
23		5000	13,85	133079,8	0,3			
24		10000	10,05	195851,5	0,2			
25	200	1000	41,90	74736,3	0,7			
26		3000	24,35	137962,9	0,5			
27		5000	18,80	177663,9	0,4			
28		10000	13,25	258112,7	0,3			

Çizelge 3.4'de görülebileceği gibi geliştirilen uygulama küçük ölçekli tüm test problemleri ve veri setleri ile her çalıştırıldığında en iyi sonuçlara eşit sonuçlar üretmiştir. Büyük ölçekli test problemlerinde ise birerleme yöntemi ile en iyi sonuçlar hesaplanamadığı için kıyaslama yapılamamıştır. Buna rağmen küçük test problemlerinde her çalıştırmada en iyi sonuçlara ulaşıldığı için, büyük test problemlerinde de algoritmanın başarılı olarak çalışacağı görülebilir. Birerleme yönteminde, 20 istasyonlu test problemlerinden 30 istasyon içeren test problemlerine geçildiğinde, çalışma zamanının yaklaşık 1300 kat arttığı gözlenmiştir. Bundan yola çıkarak 40 ve daha fazla istasyon içeren test problemlerinde birerleme yöntemi ile sonuca ulaşmanın günler süreceği söylenebilir. Önerilen TB algoritması büyük ölçekli test problemlerinde de bir saniyenin altında sonuçlar üretebilmiştir.

Hesaplanan maliyetlerin istasyon sayısına göre değişimi ve tavlama benzetimi (TB) ile birerleme yönteminin (BY) kıyaslaması Şekil 3.9'da grafik olarak gösterilmiştir. Grafiğin daha iyi anlaşılması için, her istasyon sayısının farklı sabit maliyetlerine göre elde edilen sonuçlar, ortalaması alınıp gruplanarak gösterilmiştir. Grafikte görülebildiği gibi 20 ve 30 istasyonlu tüm test problemlerinde TB algoritması BY ile hesaplanan en iyi sonuçlara her çalıştırıldığında ulaşabilmiştir. Daha fazla istasyon içeren problemlerde ise BY ile sonuç hesaplanamamış, açılan süpermarket sayısı ve taşıma maliyetleri arttığı için TB algoritmasının ürettiği sonuçlar ise gözle görülür şekilde artmıştır.



Şekil 3.9. Uygulama sonuçlarının grafiksel karşılaştırılması

4. DEĞİŞKEN KAPASİTELİ SÜPERMARKET YERLEŞİM PROBLEMİ

Çalışmanın ilk aşamasında SYP çözülürken süpermarketlerin kapasitesi göz ardı edilmiştir. Fakat gerçekte durum böyle değildir. Süpermarketlerde ancak belirli sayıda montaj bileşenlerini içeren kutu depolanabilir. Süpermarketlerin kurulumu için belirli bir sabit maliyet gereklidir ve süpermarketin kapasitesi değiştiğinde kurulum maliyeti de buna bağlı olarak değişmelidir. Bununla birlikte farklı kapasiteli süpermarketleri problem çözümünde birlikte kullanmak daha iyi çözümler üretebilir. Çalışmanın ikinci aşamasında ilk aşamada ele alınan SYP değişken kapasiteli süpermarketler kullanacak şekilde geliştirilmiştir. Kullanılan matematiksel model ve test verileri buna bağlı olarak değiştirilmiş ve yeni bir deneysel çalışma yapılmıştır.

Çalışmanın ilk aşamasında ele alınan kapasitesiz SYP için önerilen algoritmanın başarısını kıyaslayabilmek için bir birerleme uygulaması geliştirilmişti. İkinci aşamada değişken kapasiteli SYP için bir birerleme uygulaması geliştirilmek istendiğinde, yeni eklenen kapasite parametresiyle birlikte birerleme uygulamasının en küçük boyutlu problemde bile en iyi sonuçlara ulaşmasının günler süreceği hesaplanmıştır. Bu sebeple ikinci aşamada geliştirilen TB algoritmasının başarısını ölçmek için oluşturulan test verileri öncelikle, literatürde metasezgisel algoritmaların başarısını kıyaslamak için de kullanılan rastsal arama (random search) (RA) yöntemiyle çözülmüştür [60]. Daha sonra çalışmanın ilk aşamasında kullanılan TB algoritması yeni modele göre geliştirilmiş ve test problemleri çözülmüştür. Bölüm sonunda RA algoritması ve TB algoritması ile elde edilen sonuçlar kıyaslanmıştır.

4.1. Problem Tanımı

Çalışmanın ikinci aşamasında farklı kapasiteli süpermarketler problem çözümünde kullanılacağı için önerilen matematiksel model de buna bağlı olarak değişmelidir. Süpermarketlerin kapasitelerin değişmesi, amaç fonksiyonunda sadece sabit maliyete etki edeceği için fonksiyonun değişmesine gerek yoktur. Bununla birlikte süpermarketlerin hizmet verdiği istasyonların parça taleplerinin toplamı süpermarketin kapasitesini aşmamalıdır. Önerilen matematiksel modele yeni bir kısıt eklenerek bu koşul sağlanabilir. Değişken kapasiteli SYP için önerilen matematiksel model aşağıdaki gibidir.

"Eş 3.1 – 3.3"

$$\sum_{s=x_i}^{x_{i+1}-1} d_s \leq Kap_i \qquad \forall i = 1, \ldots, n;\ s=1, \ldots, S \qquad (4.1)$$

Yukarıda görülebildiği gibi çalışmanın ilk aşamasında önerilen modele Eş 4.1 eklenmiştir. Eş 4.1 süpermarketlerin kapasitelerinin hizmet verdiği istasyonlarının toplam parça talebini aşmamalarını sağlamaktadır. Modelde kullanılan kısaltmalar Çizelge 4.1'de gösterilmiştir.

Çizelge 4.1. Değişken kapasiteli SYP için önerilen matematiksel modelde kullanılan kısaltmaların gösterimi

S	İstasyon sayısı
n	Süpermarket sayısı gösteren değişken
d_s	s istasyonunda her vardiyada tahmin edilen kutu talebi
x_i	i süpermarketi tarafından hizmet verilen ilk istasyon
Kap_i	i süpermarketinin kapasitesi

4.2. Değişken Kapasiteli Süpermarket Yerleşim Problemi İçin Rastsal Arama Yaklaşımı

Metasezgisel algoritmalar rastsal arama algoritmasının geliştirilmiş hali gibi düşünülebilir. RA arama uzayı içerisinde, her yinelemede rastgele bir çözüm oluşturarak, bunların en iyisinin seçilmesine dayanır. En temel haliyle, metasezgisellerde olduğu gibi komşuluk yapısı ya da iyi çözümlere doğru yönlendirme yoktur [40,41]. Çalışmada kullanılan RA algoritmasının pseudo kodu Şekil 4.1'de gösterilmiştir.

Tekrar

Bir başlangıç çözümü (S_0) üret ve amaç fonksiyonu değerini (E_0) hesapla;

Kadar (kapasite koşulu sağlanana kadar)

En iyi çözüm $S_B = S_0$ ve en iyi amaç fonksiyonu değeri $E_B = E_0$;

Tekrar

Tekrar

Rastgele bir çözüm (S_N) üret ve amaç fonksiyonu değerini (E_N) hesapla;

Kadar (kapasite koşulu sağlanana kadar)

$\Delta E = E_N - E_B$;

$\Delta E < 0$ ise $S_B = S_N$, $E_B = E_N$;

Kadar (durdurma koşulu sağlanana kadar)

S_B ve E_B'yi raporla.

Şekil 4.1. RA algoritmasının pseudo kodu

RA algoritmasında başlangıç çözümü oluşturulurken rastgele süpermarket sayısı belirlenmiş ve her süpermarkete rastgele istasyonlar atanmış, rastgele kapasiteler belirlenmiştir. Eğer süpermarket kapasitesi hizmet verdiği istasyonların talep toplamından düşük ise, koşul sağlanana kadar yeni bir çözüm oluşturulmuştur. Bulunan çözüm en iyi çözüm olarak kabul edilmiş ve aynı yöntemle yeni çözümler oluşturulmuştur. Yeni çözümde maliyet daha düşükse en iyi çözüm olarak belirlenmiştir. Durdurma kriteri sağlandığında algoritmanın bulduğu en iyi çözüm raporlanmıştır.

4.2.1. Deneysel çalışma

Geliştirilen RA algoritmasının etkinliğini değerlendirmek amacıyla Nourmohammadi ve diğerlerinin çalışmasına benzer şekilde test problemleri oluşturulmuştur [15]. İstasyonların talepleri 1 ile 10 arasında rastsal olarak belirlenmiştir. İstasyonlar arası mesafe, x ekseni koordinatlarına (a_s) sıfırdan başlayarak birer birim eklenerek elde edilmiştir. İstasyonların tümü x ekseni üzerinde doğrusal olarak yer aldığı için çalışmada kolaylık olması açısından istasyonların y ekseni koordinatları (b_s) 0 olarak alınmış ve süpermarketlerin y ekseni koordinatları 5 birim olarak belirlenmiştir. Kapasiteleri 10, 25 ve 50 olmak üzere 3 farklı tipte süpermarket kullanılmış ve sabit maliyetleri sırasıyla 1000, 2000 ve 3000 olarak

belirlenmiştir. 20, 40, 60, 100, 150, 200 ve 300 olmak üzere farklı istasyon sayılarıyla (S) test problemleri oluşturulmuştur. Farklı kapasitedeki süpermarketlerin çözümde birlikte ve ayrı ayrı kullanılmasının çözüme etkilerini görebilmek için, belirlenen 3 farklı kapasitede süpermarketlerin birli, ikili ve üçlü kombinasyonları kullanılarak her bir problemin 7 farklı versiyonu hazırlanmış ve 49 farklı problem elde edilmiştir. Problemdeki istasyon sayısı ve çözümde kullanılan süpermarket tipi arttığında, buna bağlı olarak çözüm uzayı da büyüyeceği için, RA algoritmasın döngü sayısı istasyon sayısın 1000 katı ve kullanılan süpermarket tip sayısının çarpımıyla elde edilmiştir (RA döngü sayısı = istasyon sayısı x 1000 x çözümde kullanılan süpermarket tip sayısı).

Geliştirilen RA algoritması kullanılarak MS Visual Studio ortamında C# programlama dili ile bir uygulama geliştirilmiş ve tüm test problemleri 32 GB bellek ve Intel Xeon 3.7 Ghz işlemcili bir kişisel bilgisayar üzerinde çalıştırılmıştır. Her test problemi geliştirilen uygulama aracılığıyla beşer kez test edilmiş, çözümdeki ortalama süpermarket sayısı, ortalama maliyet ve ortalama CPU zamanı hesaplanmıştır.

Çizelge 4.3'de test problemlerinin önerilen RA algoritması ile beşer kez çalıştırılması sonucu elde edilen ortalama süpermarket sayısı, ortalama maliyet ve ortalama CPU zamanları gösterilmiştir. Çizelge 4.2'de, Çizelge 4.3'te kullanılan kısaltmalar verilmiştir. Çizelgede görüldüğü gibi istasyon sayısı düşük olan problemlerde, farklı kapasitede süpermarketler kullanıldığında maliyet bir miktar düşmektedir. Fakat istasyon sayısı arttığında, çözüm uzayı büyüdüğü ve düşük kapasiteli süpermarket kullanma eğilimi azaldığı için bu durum her zaman geçerli değildir. Aynı zamanda istasyon sayısı arttığında uygulamanın çalışma zamanı gözle görülür şekilde artmaktadır.

Çizelge 4.2. RA uygulamasında kullanılan kısaltmaların gösterimi

S	İstasyon sayısı
Kap	Çözümde kullanılan süpermarketlerin kapasiteleri
Ort. n	Ortalama süpermarket sayısı
Ort. M.	Ortalama maliyet
T	Ortalama CPU zamanı (saniye)

Çizelge 4.3. RA uygulama sonuçları

No	S	Kap	Ort. n	T (s)	Ort. M.
1	20	10	9,3	0,3	10315
2		25	4,0	0,1	9427
3		50	2,0	0,1	8212
4		10, 25	5,3	0,3	8978
5		10, 50	2,0	0,2	8212
6		25, 50	2,0	0,1	8212
7		10, 25, 50	3,0	0,3	7954
8	40	10	29,3	6,0	31568
9		25	12,0	0,5	27113
10		50	5,0	0,3	19988
11		10, 25	16,3	4,0	28569
12		10, 50	5,0	3,7	19996
13		25, 50	5,3	0,7	20742
14		10, 25, 50	6,3	3,6	19714
15	60	10	44,0	9,7	46945
16		25	17,3	1,1	38925
17		50	6,7	0,7	28040
18		10, 25	24,3	8,6	43509
19		10, 50	8,7	8,4	30013
20		25, 50	8,7	1,8	30555
21		10, 25, 50	9,0	8,4	27986
22	100	10	84,3	91,1	89393
23		25	36,7	4,4	80473
24		50	15,3	2,5	57159
25		10, 25	58,7	67,5	96397
26		10, 50	24,0	67,4	76773
27		25, 50	20,0	7,2	63035
28		10, 25, 50	29,0	62,3	72030
29	150	10	134,0	422,6	141838
30		25	67,3	15,2	144967
31		50	28,7	7,0	102227
32		10, 25	113,3	299,6	171199
33		10, 50	102,0	297,5	209217
34		25, 50	41,7	23,6	121341
35		10, 25, 50	80,3	275,2	163442

Çizelge 4.3. (devam) RA uygulama sonuçları

No	S	Kap	Ort. n	T (s)	Ort. M.
36	200	10	182,0	1142,0	192255
37		25	93,7	30,7	200931
38		50	40,7	14,9	142894
39		10, 25	153,3	813,3	237470
40		10, 50	139,7	810,4	298201
41		25, 50	62,7	49,8	179972
42		10, 25, 50	116,0	745,7	248040
43	300	10	279,0	4977,2	294550
44		25	157,7	101,7	334943
45		50	72,7	44,0	247376
46		10, 25	254,3	3564,5	391375
47		10, 50	266,3	3566,2	509687
48		25, 50	110,0	164,6	310504
49		10, 25, 50	220,0	3269,8	452945

4.3. Değişken Kapasiteli Süpermarket Yerleşim Problemi İçin Tavlama Benzetimi Yaklaşımı

Çalışmanın ikinci aşamasında, ilk aşamada önerilen TB algoritması değişken kapasiteli süpermarketler kullanacak şekilde geliştirilmiştir. Algoritmanın arama yaptığı çözüm uzayı, probleme eklenen süpermarket çeşidine bağlı olarak büyüyecektir. Örneğin 3 farklı kapasiteli süpermarket içeren problemin çözüm uzayı tek tip süpermarket kullanılan problemin çözüm uzayından büyük olacaktır. Algoritmanın daha iyi sonuçlar verebilmesi için, aynı sıcaklıkta daha iyi çözüm arama işlemini belirli sayıda tekrarlanması anlamına gelen markov zinciri uzunluğu kullanılan süpermarket tip sayısına göre artırılmıştır.

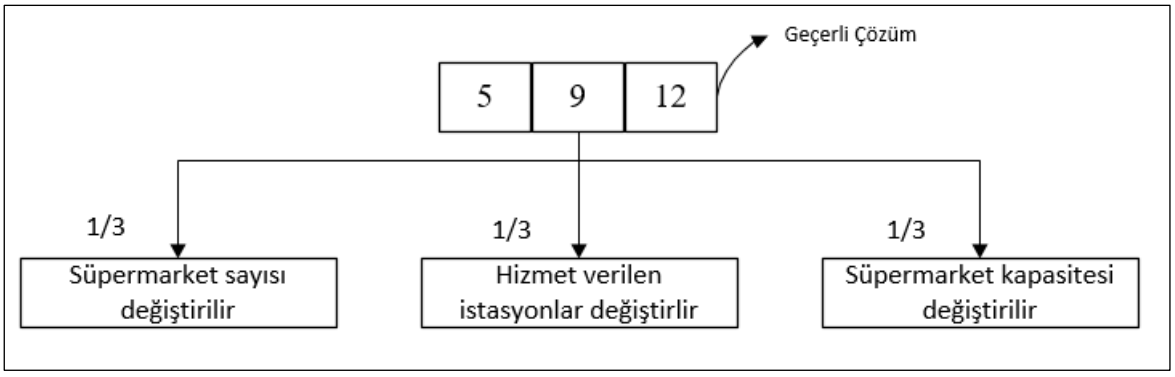
4.3.1. Başlangıç çözümü

Çalışmanın ilk aşamasında başlangıç çözümü oluşturulurken rastgele süpermarket sayısı belirlenip, her istasyona rastgele istasyonlar atanmıştı. İkinci aşamada problem çözümünde farklı kapasitedeki (tipteki) süpermarketler kullanılacağı için başlangıç çözümü oluştururken her süpermarket için kapasite belirlemek gerekir. Hizmet verilen istasyonların kapasite toplamı süpermarket kapasitelerini aşmamalıdır. Atamada kolaylık olması açısından

başlangıç çözümü oluştururken en düşük kapasiteli süpermarket tipi seçilmiş ve kapasitesini aşmayacak şekilde atanabileceği en fazla istasyon sayısına atanmıştır. Örneğin kullanılan en düşük süpermarket kapasitesi 10 birim ve toplam beş istasyonun parça talepleri sırasıyla {3, 6, 8, 4, 5} olsun. 10 birim olan süpermarket kapasitesi aşılmadan en fazla istasyona hizmet verilebilmesi için ilk iki istasyona bir süpermarket atanmalıdır (6+3<10). Üçüncü istasyon eklendiğinde süpermarket kapasitesi aşıldığı için yeni bir süpermarket eklemek gerekir. Bu şekilde üçüncü istasyona bir süpermarket, dördüncü ve beşinci istasyonlara bir süpermarket atanmış olur. Çalışmanın birinci bölümünde anlatılan gösterimle başlangıç çözümü bu örnek için {2, 3, 5} olur.

4.3.2. Komşu çözüm oluşturulması

Çalışmanın ilk aşamasında komşu çözüm üretilirken, süpermarketlerin hizmet verdiği istasyonlar ya da çözümdeki süpermarket sayısı değiştirilmişti. Daha iyi çözümler elde edebilmek için komşu çözümler üretilirken belirlenen kapasiteler de değiştirilmelidir. İkinci aşamada süpermarket kapasitelerinin değiştirilmesi de yeni bir komşu çözüm üretme yöntemi olarak algoritmaya eklenmiştir. Şekil 4.2'de görülebildiği gibi, komşu çözüm üretilirken bahsedilen üç yöntemden bir tanesi 1/3 olasılıkla seçilmiştir.



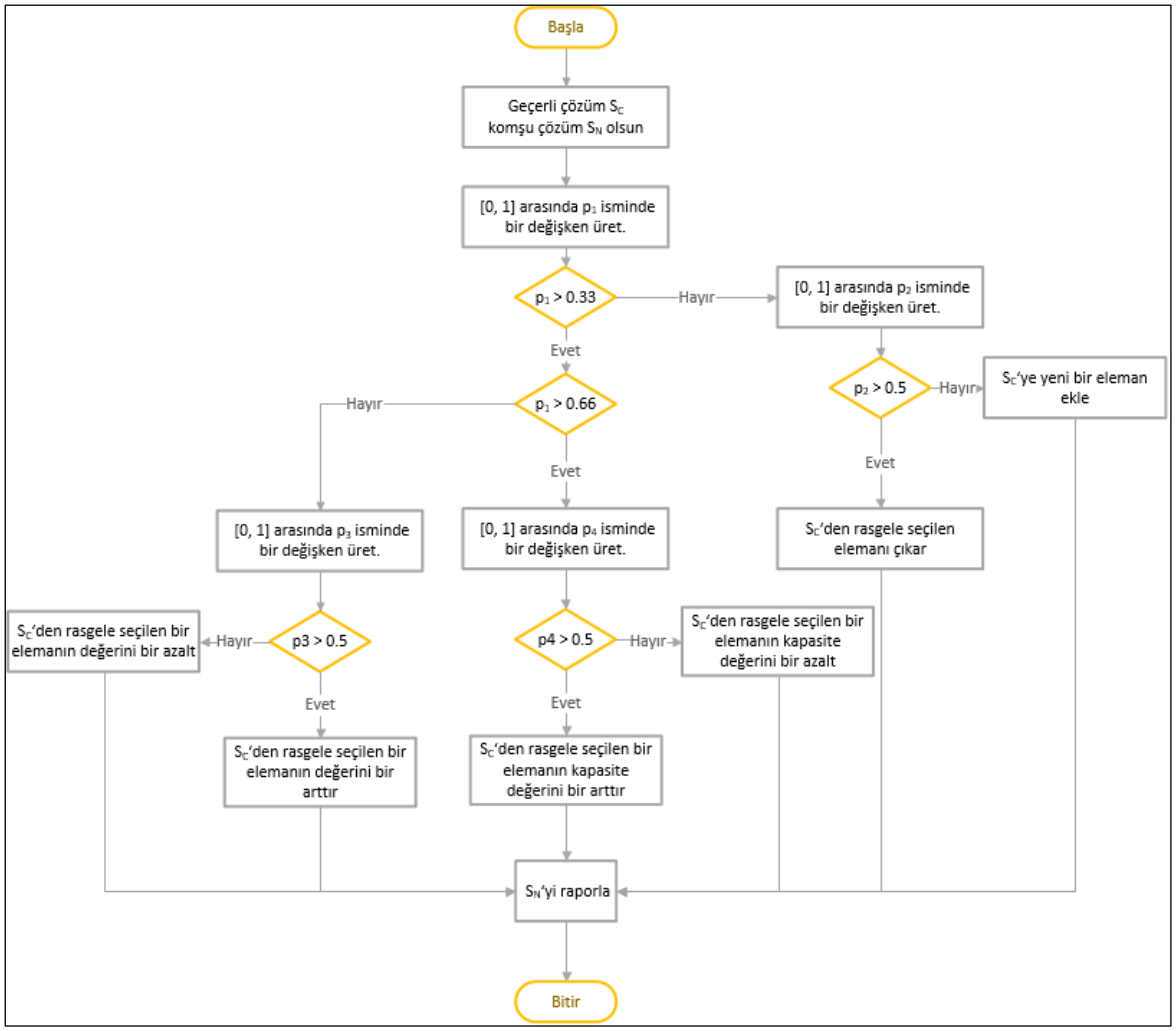
Şekil 4.2. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm üretme yöntemleri

Rastgele seçilen süpermarketin kapasitesinin değiştirilmesi gerekirse, problemde kullanılan süpermarketlerden %50 olasılıkla daha düşük kapasiteli bir süpermarket ve %50 olasılıkla daha yüksek kapasiteli bir süpermarket seçilir. Örneğin problem çözümünde 10, 25 ve 50 olmak üzere üç farklı kapasiteli süpermarket kullanılıyor olsun. Dört süpermarket içeren geçerli çözümünün kapasiteleri sırasıyla 10, 10, 25 ve 50 olabilir. Rastsal olarak üçüncü

sıradaki 25 kapasiteli süpermarket seçilmiş ve kapasitesinin artırılması isteniyorsa, yeni komşu çözümün kapasiteleri sırasıyla 10, 10, 50 ve 50 olacaktır. Çalışmanın ikinci aşamasında önerilen komşu çözüm algoritması Şekil 4.3'te gösterildiği gibidir ve algoritmanın akış diyagramı Şekil 4.4'te gösterilmiştir.

Geçerli çözüm S_C olsun ve komşu çözüm S_N olsun;
p1 isminde [0, 1] arasında rastgele bir değer üret;
p1 < 0,33 ise
  p2 isminde [0, 1] arasında rastgele bir değer üret;
  p2 > 0,5 ise S_C'den rastgele seçilmiş bir elemanı çıkararak yeni bir S_N oluştur;
  p2 ≤ 0,5 ise S_C'ye rastgele yeni bir elman ekleyerek yeni bir S_N oluştur;
p1 < 0,66 ise
  p3 isminde [0, 1] arasında rastgele bir değer üret;
  p3 > 0,5 ise S_C'den rastgele seçilmiş bir elemanı bir artırarak yeni bir S_N oluştur;
  p3 ≤ 0,5 ise S_C'den rastgele seçilmiş bir elemanı bir azaltarak yeni bir S_N oluştur;
p1 ≥ 0.66 ise
  p4 isminde [0, 1] arasında rastgele bir değer üret;
  p4 > 0,5 ise S_C'den rastgele seçilmiş bir elemanının kapasite değerini bir artırarak yeni bir S_N oluştur;
  p4 ≤ 0,5 ise S_C'den rastgele seçilmiş bir elemanının kapasite değerini bir azaltarak yeni
  bir S_N oluştur;

Şekil 4.3. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm algoritmasının pseudo kodu



Şekil 4.4. Değişken kapasiteli SYP için önerilen TB algoritmasında kullanılan komşu çözüm algoritmasının akış diyagramı

4.3.3. Örnek problem

Değişken kapasiteli süpermarket yerleşim problem çözümünün daha iyi anlaşılması için çalışmanın birinci bölümünde geliştirilen örnek problem, süpermarket kapasitesi eklenerek çözülmüştür. Çizelge 4.4'deki örnek problem verisini düşünelim. İstasyon talepleri (d_s) 1 ile 10 arasında rastgele belirlenmiş, istasyonlar arası mesafe x ekseninde (a_s) 1 birim, y koordinatları (b_s) 1 ve süpermarketlerin montaj hattından uzaklığı 5 birim olan 5 istasyon içeren çizgi halinde bir montaj hattı olsun. Problem çözümünde kapasiteleri 10, 15 ve 20 birim olmak üzere, maliyetleri sırasıyla 100, 150 ve 200 birim olan üç farklı süpermarket kullanılsın.

Çizelge 4.4. Değişken kapasiteli SYP için örnek veri

İstasyon s	1	2	3	4	5
d_s	3	5	7	6	1
a_s	1	2	3	4	5
b_s	1	1	1	1	1

Örnek problemi önerilen algoritma ile çözmek için öncelikle bir başlangıç çözümü oluşturmak gerekir. Çalışmanın ilk aşamasından farklı olarak çözümde kullanılacak süpermarketlerin her birinin kapasitesi hizmet verdiği istasyonların kapasite toplamını aşmamalıdır. Ayrıca çözümde farklı tipte süpermarketler kullanılabilir. Örneğin {3, 5} çözümündeki ilk süpermarkete olan toplam talep, hizmet verdiği 1, 2 ve 3 numaralı istasyonların toplam talebi 5+3+7=15 olur. Bu durumda ilk süpermarketin kapasitesi en az 15 olmalıdır. İkinci süpermarketin (5) hizmet verdiği istasyonların toplam talebi ise 6+1=7 olduğu için üç farklı süpermarket de kullanılabilir. Sonuç olarak {3, 5} çözümündeki süpermarketlerin kapasiteleri, kapasite koşulunu sağlamak kaydıyla ve en düşük maliyetle sırasıyla 15 ve 10 olmalıdır. Çalışmanın ilk aşamasında anlatılan yöntemle çözümün maliyeti hesaplanırsa 544 bulunur.

Daha iyi çözümler araştırmak için geçerli çözümden Şekil 4.3'te gösterildiği gibi komşu çözümler üretilir. Çalışmanın ilk aşamasından farklı olarak, süpermarket sayısı ve hizmet verilen istasyonların yanında, süpermarket kapasitesi de değiştirilir. Bu sayede farklı kapasitedeki süpermarketlerin çözüme katkısı da görülmüş olur.

Örnek problem önerilen algoritma ile çözüldüğünde, çözüm {2, 5}, süpermarket kapasiteleri sırasıyla 10, 15 ve toplam maliyet de 542 bulunur. Bu sonucu tek tip süpermarket kullanılan çözümlerle karşılaştırmak istersek; sadece 10, sadece 15 ve sadece 20 birim kapasiteli süpermarketler kullanılan çözümlerin maliyetleri sırasıyla 550, 592 ve 692 olduğu görülür. Bu da bize farklı tipteki süpermarketlerin çözümde birlikte kullanılmasının daha iyi sonuçlar verebileceğini gösterir.

4.3.4. Deneysel çalışma

RA algoritması için oluşturulan test problemleri, geliştirilen TB algoritması ile de çözülmüştür. Daha iyi çözümler elde edilebilmek için önerilen TB algoritmasının parametreleri ön deneylerle belirlenmiştir. T_0, T_{min}, α parametrelerinin etkin değerlerini bulabilmek için rastgele üç test problemi seçilmiştir. T_0 için üç seviye (2500, 5000, 7500), T_{min} için üç seviye ("0,1", "1", "10") ve α için üç seviye ("0,9", "0,95", "0,99") belirlenmiş ve her parametre için her seviye üçer kez çalıştırılarak test edilmiştir. Elde edilen sonuçlar üzerinde, toplam maliyet ve CPU zamanı açısından varyans analizi yapılmıştır. Ayrıca parametrelerin etkilerini açıkça göstermek ve bu parametrelerin etkin seviyelerini belirlemek için Duncan gruplamaları uygulanmıştır. Yapılan analizler sonucunda maliyet açısından T_0 parametresi 7500, T_{min} ve α parametreleri ise CPU zamanı açısından sırasıyla 10 ve 0,9 olarak belirlenmiştir. Uygulanan analiz sonuçları EK-2'de gösterilmiştir. Kullanılan farklı sayıdaki süpermarket tipine göre çözüm uzayı büyüyeceği için; IT değeri tek tip süpermarket kullanıldığında 1000, 2 tip süpermarket kullanıldığında 3000 ve 3 tip kullanıldığında 5000 olarak belirlenmiştir.

Çalışmanın ilk aşamasında olduğu gibi ikinci aşamada da önerilen algoritma kullanılarak MS Visual Studio ortamında C# programlama dili ile bir uygulama geliştirilmiş ve tüm test problemleri 32 GB bellek ve Intel Xeon 3.7 Ghz işlemcili bir kişisel bilgisayar üzerinde çalıştırılmıştır. Her test problemi geliştirilen uygulama aracılığıyla beşer kez test edilmiş, çözümdeki ortalama süpermarket sayısı, ortalama maliyet ve ortalama CPU zamanı hesaplanmıştır.

Çizelge 4.6'da test problemlerinin önerilen TB algoritması ile beşer kez çalıştırılması sonucu elde edilen ortalama süpermarket sayısı, ortalama maliyet ve ortalama CPU zamanları ve RA algoritmasıyla geliştirilen uygulamanın sonuçları karşılaştırılmıştır. Çizelge 4.5'te, Çizelge 4.6'da kullanılan kısaltmalar verilmiştir.

Çizelge 4.5. TB ve RA algoritmaları karşılaştırmasında kullanılan kısaltmaların gösterimi

S	İstasyon sayısı
Kap	Çözümde kullanılan süpermarketlerin kapasiteleri
Ort. n	Ortalama süpermarket sayısı

Çizelge 4.5. (devam) TB ve RA algoritmaları karşılaştırmasında kullanılan kısaltmaların gösterimi

Ort. M.	Ortalama maliyet
T	Ortalama CPU zamanı (saniye)
RA. n	RA algoritması ortalama süpermarket sayısı
RA. M.	RA algoritması ortalama maliyet
RA. T	RA algoritması ortalama CPU zamanı (saniye)
Fark	İki uygulama arasındaki yüzdesel maliyet artışı

Çizelge 4.6. TB ve RA algoritmaları uygulama sonuçları

No	S	Kap	Ort. n	Ort. M.	T (s)	RA. n	RA T.(s)	RA. M.	Fark (%)
1	20	10	9,0	9988	0,1	9,3	0,3	9997	0,1
2		25	4,0	9426	0,1	4,0	0,1	9427	0,0
3		50	2,0	8212	0,0	2,0	0,1	8212	0,0
4		10, 25	4,0	8508	0,2	5,3	0,3	8729	2,6
5		10, 50	2,0	8212	0,1	2,0	0,2	8212	0,0
6		25, 50	2,0	8212	0,1	2,0	0,1	8212	0,0
7		10, 25, 50	3,0	7954	0,2	3,0	0,3	7954	0,0
8	40	10	27,0	29272	0,3	29,3	6,0	31568	7,8
9		25	10,0	23284	0,1	12,0	0,5	27113	16,4
10		50	5,0	19940	0,1	5,0	0,3	19988	0,2
11		10, 25	11,6	22220	0,4	16,3	4,0	28569	28,6
12		10, 50	6,0	19182	0,2	5,0	3,7	19996	4,2
13		25, 50	5,0	19064	0,2	5,3	0,7	20742	8,8
14		10, 25, 50	5,0	19064	0,4	6,3	3,6	19714	3,4
15	60	10	33,0	36150	0,4	44,0	9,7	46945	29,9
16		25	12,0	28806	0,2	17,3	1,1	38925	35,1
17		50	6,0	25494	0,1	6,7	0,7	28040	10,0
18		10, 25	13,4	28693	0,5	24,3	8,6	43509	51,6
19		10, 50	6,4	25477	0,3	8,7	8,4	30013	17,8
20		25, 50	6,0	24840	0,3	8,7	1,8	30555	23,0
21		10, 25, 50	6,0	24840	0,5	9,0	8,4	27986	12,7
22	100	10	65,0	70385	0,7	84,3	91,1	89393	27,0
23		25	22,0	52213	0,3	36,7	4,4	80473	54,1
24		50	11,0	45602	0,2	15,3	2,5	57159	25,3

Çizelge 4.6. (devam) TB ve RA algoritmaları uygulama sonuçları

No	S	Kap	Ort. n	Ort. M.	T (s)	RA. n	RA T.(s)	RA. M.	Fark (%)
25		10, 25	22,2	50570	0,8	58,7	67,5	96397	90,6
26		10, 50	11,2	44374	0,5	24,0	67,4	76773	73,0
27		25, 50	11,0	44688	0,5	20,0	7,2	63035	41,1
28		10, 25, 50	11,2	44176	0,8	29,0	62,3	72030	63,1
29	150	10	100,0	108411	1,1	134,0	422,6	141838	30,8
30		25	34,8	82235	0,5	67,3	15,2	144967	76,3
31		50	17,8	72202	0,3	28,7	7,0	102227	41,6
32		10, 25	35,8	80090	1,2	113,3	299,6	171199	113,8
33		10, 50	17,4	70440	0,7	102,0	297,5	209217	197,0
34		25, 50	17,6	69596	0,7	41,7	23,6	121341	74,4
35		10, 25, 50	17,6	69372	1,1	80,3	275,2	163442	135,6
36	200	10	132,0	143068	1,4	182,0	1142,0	192255	34,4
37		25	44,4	105788	0,6	93,7	30,7	200931	89,9
38		50	23,0	94244	0,4	40,7	14,9	142894	51,6
39		10, 25	46,4	104556	1,5	153,3	813,3	237470	127,1
40		10, 50	22,4	92611	0,9	139,7	810,4	298201	222,0
41		25, 50	23,4	91778	0,9	62,7	49,8	179972	96,1
42		10, 25, 50	23,0	91471	1,4	116,0	745,7	248040	171,2
43	300	10	198,0	214962	2,1	279,0	4977,2	294550	37,0
44		25	70,4	166038	0,9	157,7	101,7	334943	101,7
45		50	35,0	143364	0,6	72,7	44,0	247376	72,6
46		10, 25	77,0	162830	2,5	254,3	3564,5	391375	140,4
47		10, 50	34,8	142329	1,3	266,3	3566,2	509687	258,1
48		25, 50	35,6	140230	1,4	110,0	164,6	310504	121,4
49		10, 25, 50	35,4	139190	2,2	220,0	3269,8	452945	225,4

Çizelge 4.6'da görüldüğü gibi, çözümde farklı tipte (kapasitede) süpermarketler birlikte kullanıldığında, aynı problemde eşit ya da daha düşük maliyette sonuçlar elde edilmiştir. Örneğin tablodaki 20 istasyonlu montaj hattı problemi için tek tip 10 birim kapasiteli süpermarket kullanıldığında ortalama maliyet 9988 birim iken, 10 ve 25 birim olmak üzere iki faklı kapasiteli süpermarket kullanıldığında ortalama maliyet 8508 birim olmuştur. En iyi sonuç ise üç farklı kapasitedeki süpermarket (10, 25, 50) çözümde birlikte kullanıldığında elde edilmiştir (7954). İstasyon sayısı arttığında da bu durumun benzer olduğu görülür.

Çizelge 4.6'da elde edilen sonuçlar Eşlenik T Testi ile analiz edilmiştir. Test sonuçları Ek-2'de verilmiştir. Tavlama benzetimi ve rastsal arama yöntemlerinin eşit olduğuna diğer bir ifadeyle aralarında istatistiksel olarak %95 güven düzeyinde anlamlı bir fark olmadığına ilişkin kurulan sıfır hipotezi reddedilmiştir (P-Değeri=0.000). Buna göre TB bu test problemlerinde rastsal arama yönteminden %95 güven düzeyinde istatistiksel olarak daha düşük sonuçlar üretmektedir. Diğer bir ifadeyle TB algoritması, RA algoritmasına göre çok daha başarılı çalışmaktadır. RA algoritması, küçük ölçekli problemlerinde TB algoritmasına eşit ya da yakın sonuçlar üretebildiği görülürken, istasyon sayısı arttığında aradaki farkın açıldığı görülebilir. Çizelgedeki fark sütünü RA algoritmasının ürettiği maliyetlerin, TB algoritmasında göre yüzdesel olarak ne kadar arttığını göstermektedir. İstasyon sayısının 300'e çıktığında bu artışın %250'leri geçtiği görülebilir.

Aynı zamanda RA algoritması küçük ölçekli problemlerde, makul sürelerde çözüm üretebilirken, problem büyüdükçe bu sürenin katlanarak arttığını, hatta bazı durumlarda bir saati aştığı görülmektedir. Geliştirilen TB algoritması ise, büyük ölçekli problemlerde bile en fazla 2 saniye içerisinde sonuçlar üretebilmiştir.

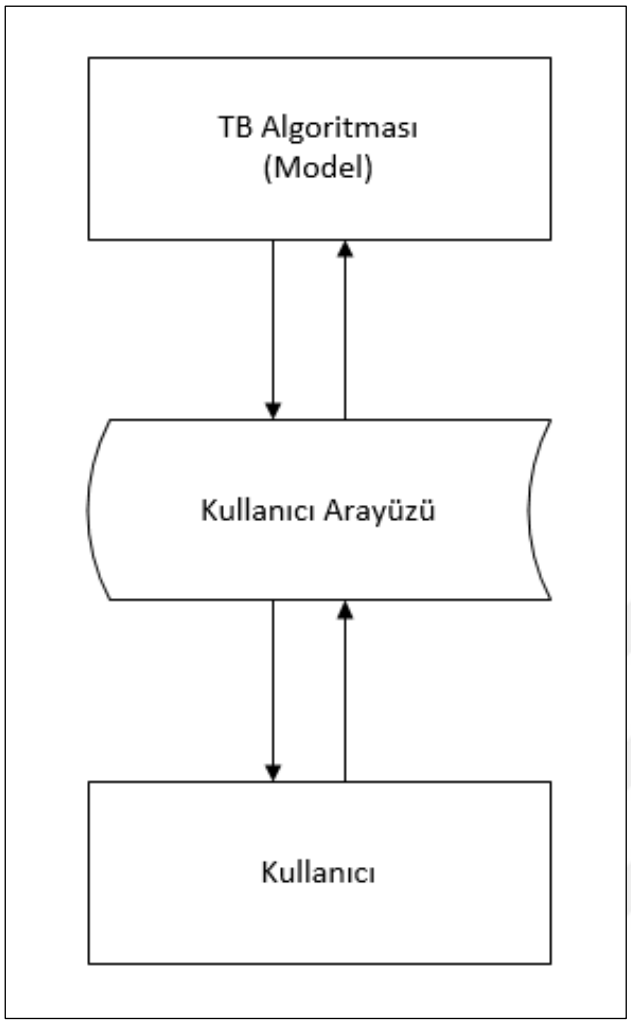
5. SÜPERMARKET YERLEŞİM PROBLEMİ İÇİN BİR KARAR DESTEK SİSTEMİ

Geliştirilen TB algoritmasın testler sonucunda başarılı bir şekilde çalıştığının anlaşılmasından sonra, algoritma kullanılarak bir KDS geliştirilmiştir. Geliştirilen KDS ile özellikle otomotiv sektöründe fabrika zeminine süpermarket yerleştirmek isteyen yöneticilerin karar vermelerine yardımcı olmak amaçlanmıştır. Aşağıda geliştirilen sistemin yapısı ve sistemin nasıl kullanıldığının anlaşılması için örnek bir uygulama anlatılmıştır.

5.1. Sistemin Teknik Altyapısı

Sistem, yazılım geliştirmede yaygın olarak kullanılan C# programlama dili ile geliştirilmiştir. Sistemin arayüzü Microsoft firması tarafından geliştirilen Windows Presentation Foundation (WPF) teknolojisi ile hazırlanmıştır. Sistem kurulum gerektirmeden, dosya uzantısı “exe” olan dosyaya tıklanarak Windows işletim sistemi üzerinde çalıştırılabilir.

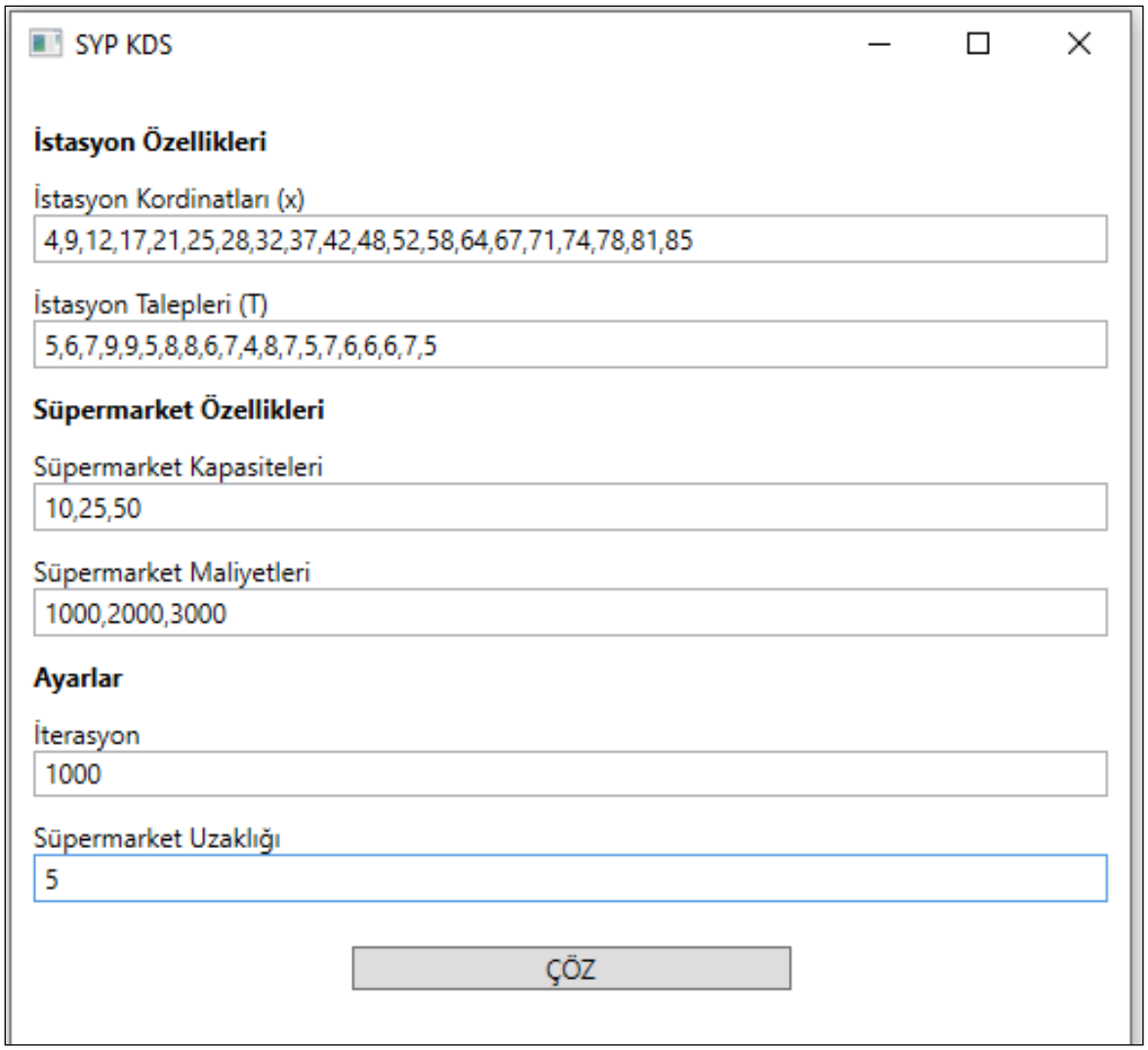
Bir Model Tabanlı KDS olan yazılımın mimarisi Şekil 5.1’de gösterildiği gibidir. Kullanıcı WPF teknolojisi ile geliştirilen arayüz aracılığıyla çözmek istediği SYP’ye ait verileri girer. Arayüz katmanı verileri TB algoritmasının bulunduğu katmana gönderir. Algoritma girilen verilerle çalıştırılarak üretilen çözüm arayüz aracılığıyla kullanıcıya gösterilir.



Şekil 5.1. Geliştirilen Model Tabanlı KDS Yapısı

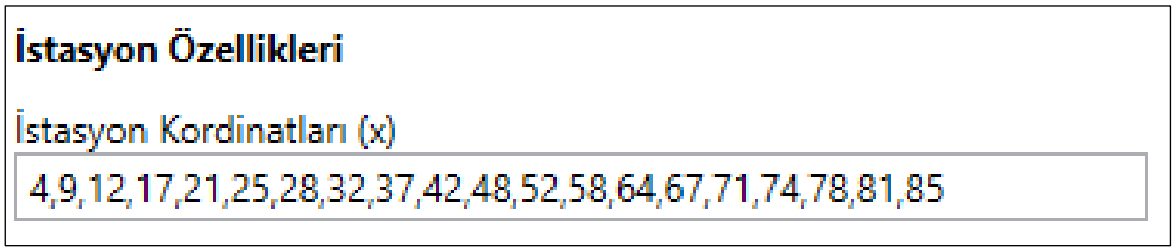
5.2. Sistemin Arayüzü

Yazılım çalıştırıldığında Resim 5.1'de gösterilen arayüz kullanıcıyı karşılar. Sistemin daha iyi anlaşılması için arayüz örnek veriler girilmiş halde başlar. Kullanıcı sırasıyla "İstasyon Kordinatları", "İstasyon Talepleri", "Süpermarket Kapasiteleri", "Süpermarket Maliyetleri", "İterasyon" ve "Süpermarket Uzaklığı" bilgilerini girdikten sonra "ÇÖZ" butonuna tıklayarak çizgi halinde tek bir montaj hattı için kurulması gereken süpermarketlerin sayısı, hizmet vereceği istasyonlar ve fabrika zemini üzerinde yerleri hakkında çözüm önerisi alabilir.



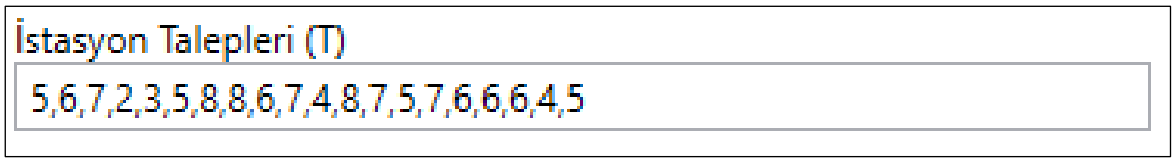
Resim 5.1. Sistemin arayüzü

İstasyon koordinatları Resim 5.2'de gösterildiği gibi, x ekseni üzerinde, istasyon sırasına göre virgülle ayrılarak uzaklık birimi cinsinden girilebilir. Örneğin şekilde, ilk istasyonun x ekseni koordinatı 4, ikinci istasyonun x koordinatı 9 birim olan toplam 20 istasyonlu çizgi halindeki bir montaj hattı için girilmiş koordinatlar gösterilmiştir.



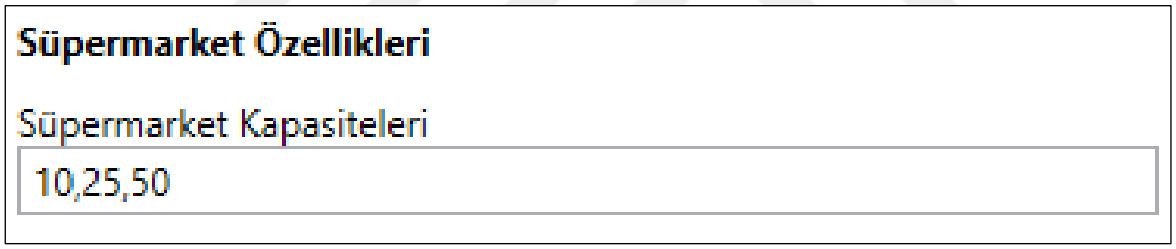
Resim 5.2. Sisteme girilen istasyon x koordinatları

Her bir istasyonun talebi Resim 5.3'te gösterildiği gibi istasyon sırasına göre virgülle ayırarak kutu cinsinden girilebilir. Örneğin şekilde ilk istasyonun talebi 5, ikinci istasyonun talebi 6 kutu olan 20 istasyonun talepleri gösterilmiştir.



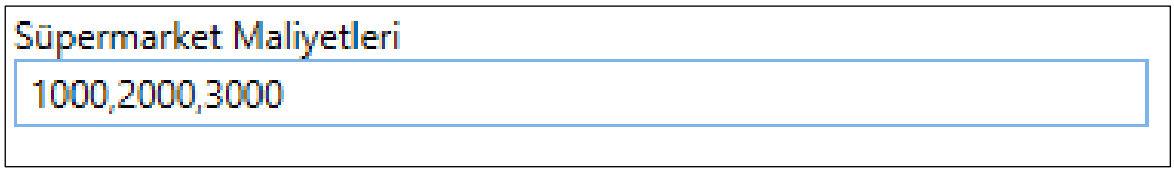
Resim 5.3. Sisteme girilen istasyon talepleri

Fabrika zeminine yerleştirilmek istenen farklı kapasitedeki süpermarketler Resim 5.4'te gösterildiği gibi sisteme girilebilir. Örneğin resimde kapasiteleri 10, 25 ve 50 kutu olan üç farklı tipte süpermarket girilmiştir. İstenilen kadar süpermarket girilebilmekle birlikte, farklı tipte süpermarketler kullanılmayacak ise; tek bir süpermarket kapasitesi girilebilir.



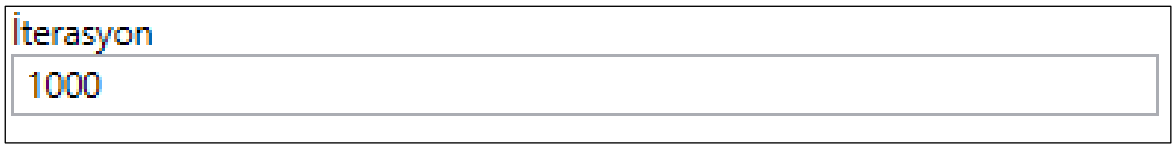
Resim 5.4 Sisteme girilen süpermarket kapasiteleri

Kullanılmak istenen farklı tipteki süpermarketlerin maliyetleri de farklı olacaktır. Her bir süpermarketin maliyeti, kapasite alanına giriş sırasına göre birim cinsinden girilir. Resim 5.5'te, Resim 5.4'te gösterilen ve kapasiteleri 10, 25 ve 50 kutu olan süpermarketlerin kurulum maliyetleri sırasıyla 1000, 2000 ve 3000 birim olarak girildiği gösterilmiştir.



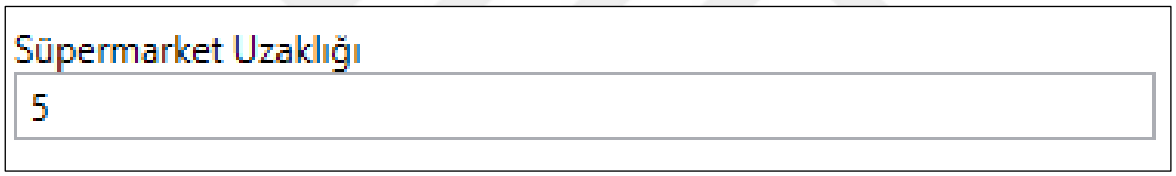
Resim 5.5 Sisteme girilen süpermarket maliyetleri

TB algoritmasında Markov zinciri uzunluğu olarak kullanılan iterasyon (IT) parametresi, yazılımın arayüzünden değiştirilebilir. Parametrenin değeri arttıkça programın çalışma süresi uzarken, daha iyi sonuçlara ulaşılabilir. Resim 5.6'da gösterildiği gibi iterasyon değeri varsayılan olarak 1000 girilmiştir.



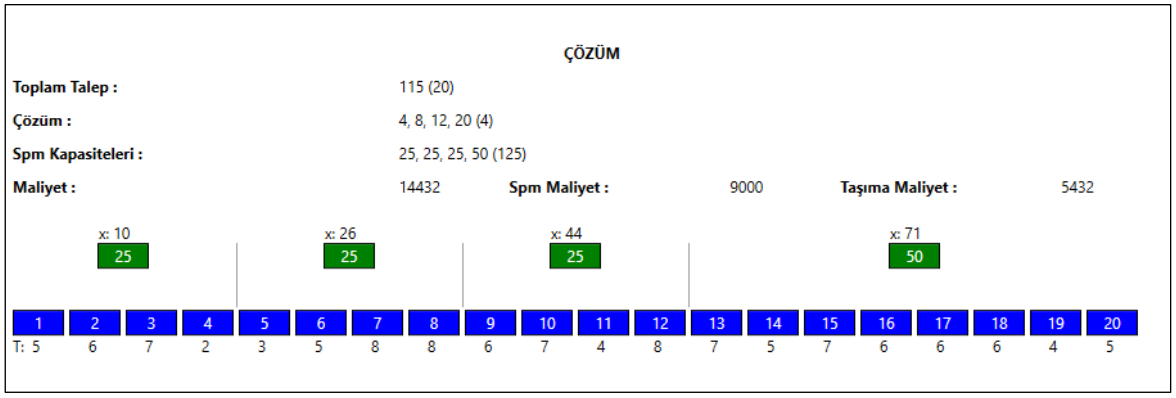
Resim 5.6. Sisteme girilen iterasyon parametresi

Montaj hattı çizgi halinde olduğu için, tüm süpermarketler montaj hattına belirli bir mesafede yerleştirilir. Resim 5.7'de gösterildiği gibi, yazılım süpermarketlerin y ekseni üzerinde montaj hattından uzaklığının birim cinsinden girilmesine olanak tanır.



Resim 5.7. Sisteme girilen süpermarket uzaklığı

Tüm değerler girildikten sonra "ÇÖZ" butonuna tıklandığında, verilerde hata yoksa sistem arka planda TB algoritmasını girilen değerlerle çalıştırır. Girilen değerlere göre, belirli bir süre çalıştıktan sonra sistem Resim 5.8'de gösterildiği gibi sonuç ekranını gösterir.



Resim 5.8. Sistemin çözüm ekranı

Çözümde gösterilen “Toplam Talep” istasyonlarının taleplerinin toplamıdır. Daha önceki bölümlerde anlatıldığı gibi, sistemin ürettiği süpermarket düzeni metinsel olarak 4, 8, 12, 20 şeklinde gösterilmiştir. Bu gösterim çözümde toplam 4 süpermarket olduğunu, ilk süpermarketin 1 ile 4, ikinci süpermarketin 5 ile 8, üçüncü süpermarketin 9 ile 12 ve son olarak dördüncü süpermarketin 13 ile 20 numaralı istasyonlar arasında hizmet vereceğini anlatmaktadır. “Spm Kapasiteleri” başlığında 25, 25, 25, 50 sırasıyla çözümde kullanılan süpermarketlerin kapasitelerini gösterir. Çözümde birinci süpermarketin kapasitesi 25 kutu iken son süpermarketin kapasitesi 50’dir. Bulunan süpermarket düzeninin maliyeti “Spm Maliyet”, “Taşıma Maliyet” ve “Maliyet” olarak üç başlıkta gösterilmiştir. “Taşıma Maliyeti” çekici trenlerin dolaşacağı toplam mesafenin istasyon talepleri ile çarpılmasıyla bulunur. “Spm Maliyet” ise çözümde kullanılan süpermarketlerin maliyetlerinin toplamıdır. “Maliyet” ise toplam maliyet olup, taşıma maliyeti ve süpermarketlerin sabit maliyetlerinin toplamıdır.

Çözümün daha iyi anlaşılması için sistem sonucu görsel olarak da gösterir. Yeşil dikdörtgenlerle gösterilen süpermarketlerin kapasiteleri üzerinde ve x ekseni üzerinde bulundukları nokta “x:” şeklide dikdörtgenlerin üzerinde gösterilir. Süpermarketlerin altına mavi dikdörtgenlerle istasyonlar gösterilir ve hangi süpermarketin hangi istasyonlara hizmet verdiğinin anlaşılması için montaj hattına dik çizgiler çizilmiştir. İstasyonların üzerinde numarası ve altında talepleri gösterilmiştir.

5.3. Örnek Uygulama

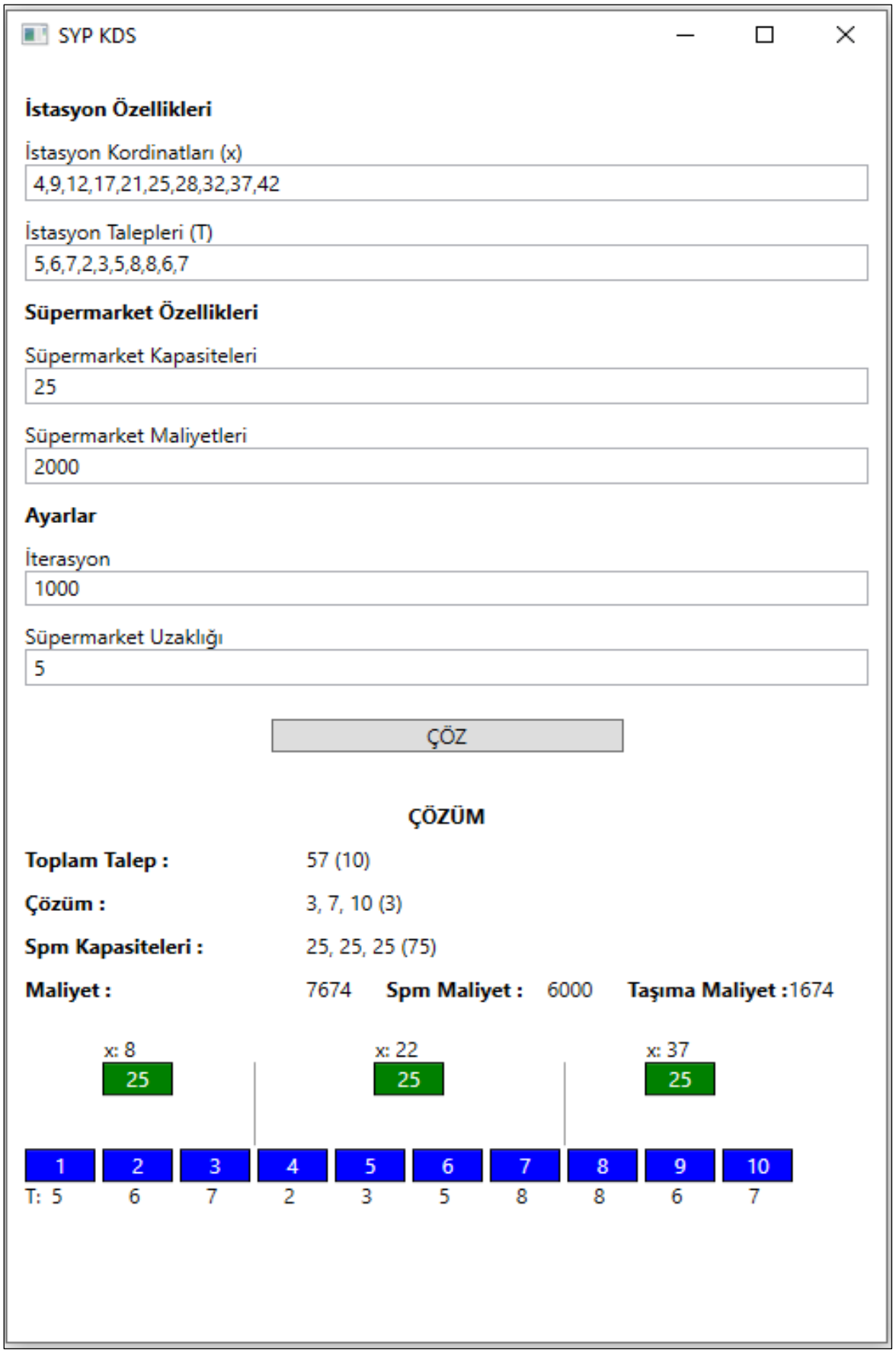
Sistemin daha iyi anlaşılması için bir örnek problem düşünelim. İstasyon x koordinatları sırasıyla 4, 9, 12, 17, 21, 25, 28, 32, 37 ve 42 olan 10 istasyonlu bir montaj hattı düşünelim. İstasyon talepleri sırasıyla 5, 6, 7, 2, 3, 5, 8, 8, 6 ve 7 olsun. İlk aşamada çözümde kapasitesi 25 kutu, maliyeti 2000 birim olan tek bir süpermarket kullanıldığını düşünelim. IT değeri 1000 ve istasyonların y koordinatı 0 iken süpermarketlerin y koordinatları 5 birim olsun. Veriler girilip “ÇÖZ” butonuna tıklandıktan sonra sistem ürettiği çözümü Resim 5.9’da olduğu gibi gösterir. Çözümde görüldüğü gibi sistem tek seçeneği olan 25 kapasiteli süpermarketlerden toplam 3 tane kullanarak, {3, 7, 10} çözümü için toplam maliyeti 7674 şeklinde hesaplamıştır.

Girdilere kapasitesi 50 kutu, kurulum maliyeti 3000 birim olan yeni bir süpermarket tipi eklersek çözüm Resim 5.10'da gösterildiği gibi olur. Çözümde ilki 25 ve ikincisi 50 kapasiteli iki adet süpermarket kullanılmış, {5,10} çözümü için maliyet 7508 olarak hesaplanmıştır. Görüldüğü gibi aynı probleme yeni bir süpermarket tipi eklendiğinde maliyet 7674 birimden 7508 birime düşmüştür.

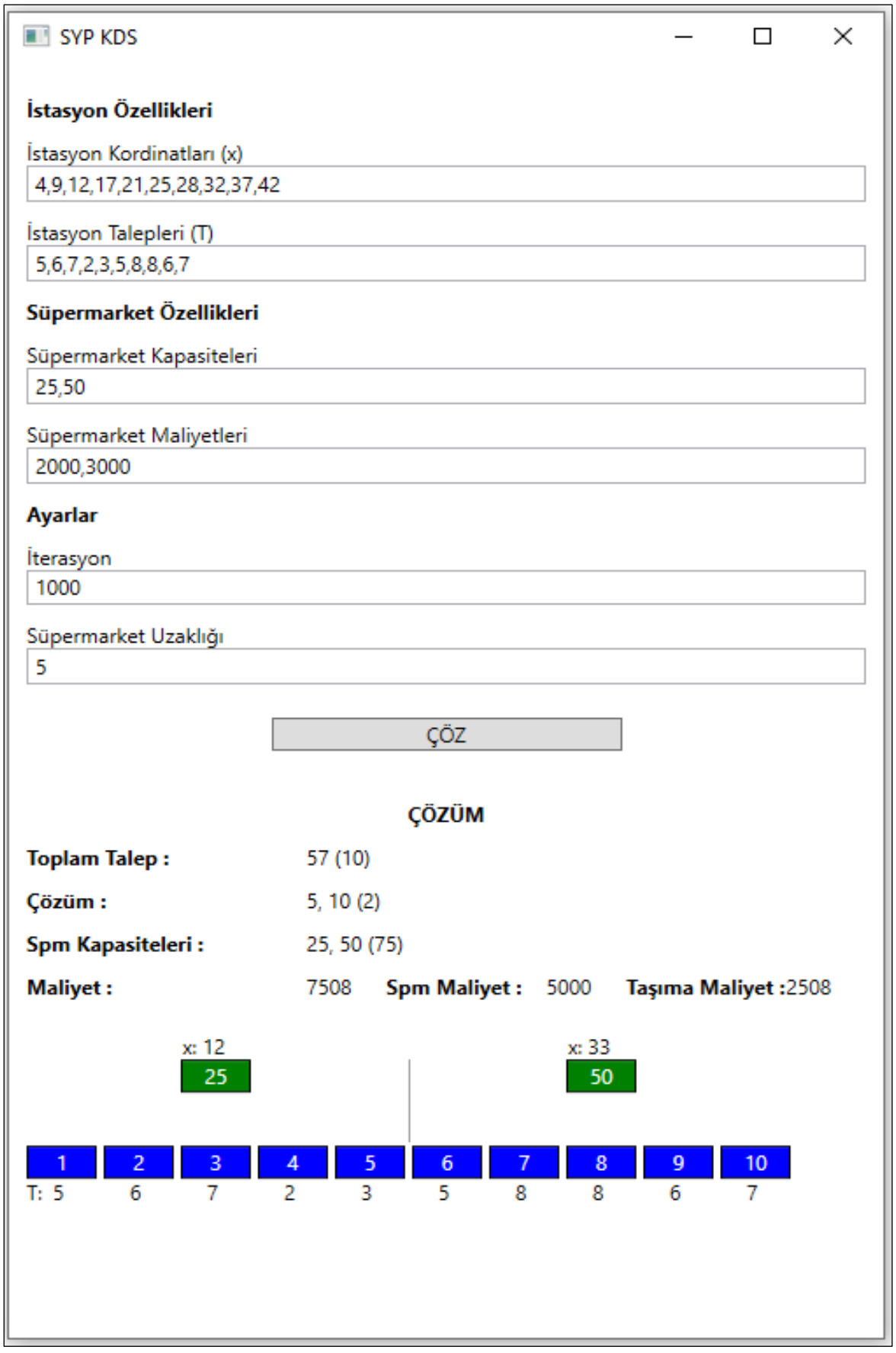
Şimdi de sisteme 10 kutu kapasiteli maliyeti 1000 birim olan yeni bir süpermarket ekleyerek, kullanılan süpermarket tip sayısını üçe çıkaralım. Resim 5.11'de gösterilen çözümde görüldüğü gibi sistem iki adet 25 kutu kapasiteli, iki adet 10 kutu kapasiteli toplam 4 süpermarket kullanılırken, 50 kutu kapasiteli süpermarketlerden kullanılmadığı görülür. Maliyet de 7414 birime gerilemiştir.

Son olarak 3 tip süpermarketin maliyetlerini 1000, 2000 ve 3000 birimden sırasıyla 5000, 10000 ve 15000 birime çıkaralım. Resim 5.12'deki çözümde görüldüğü gibi, aynı problemde kurulum maliyetleri artırıldığında, süpermarket açma eğilimi azalmakta ve bunun sonucunda daha yüksek kapasiteli süpermarketler kullanma eğilimi doğmaktadır. Resimde görüldüğü gibi kapasitesi 25 ve 50 kutu olan iki adet süpermarket kullanılmış ve çözüm {5, 10} olmuştur.

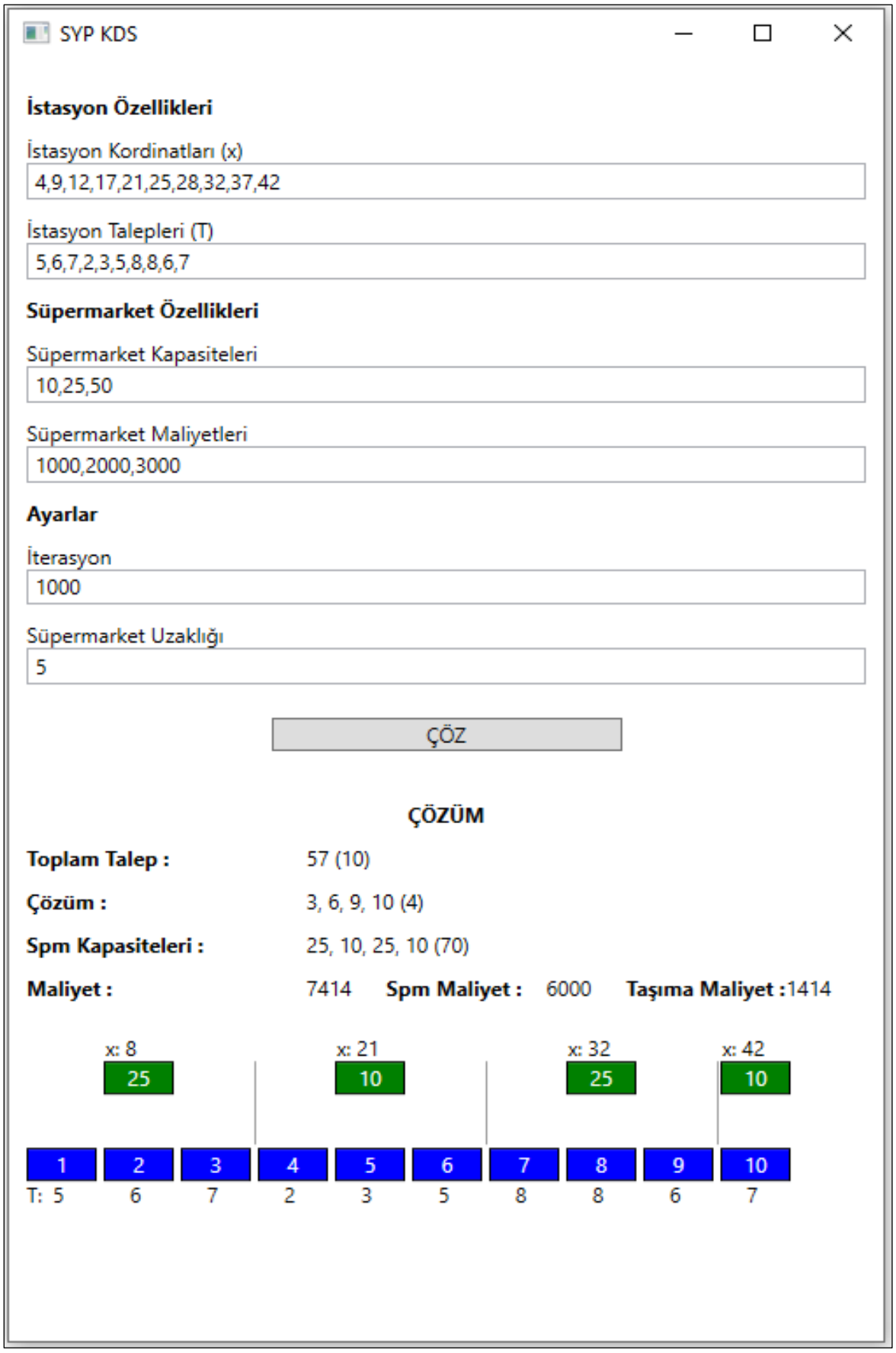
Kullanıcıların uygulama sonuçlarını karşılaştırılabilmesi için, sistem aynı zamanda bir önceki sonuçları göstermektedir. Resim 5.13'te görüldüğü gibi, eğer sistem ilk kez çalıştırılmıyorsa, her seferinde bir önceki çözümü göstermektedir.



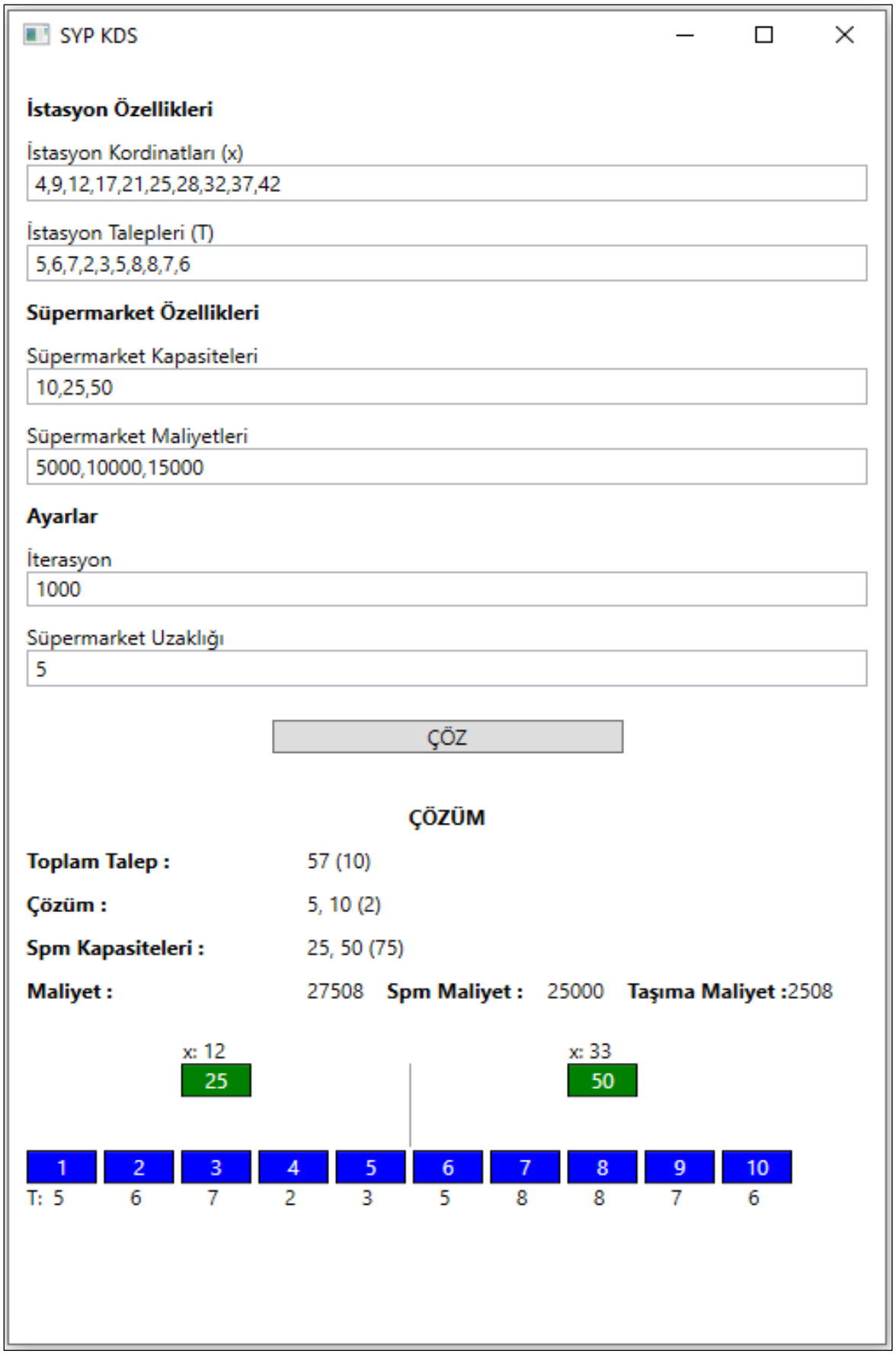
Resim 5.9. Tek tip süpermarket içeren örnek uygulama sonucu



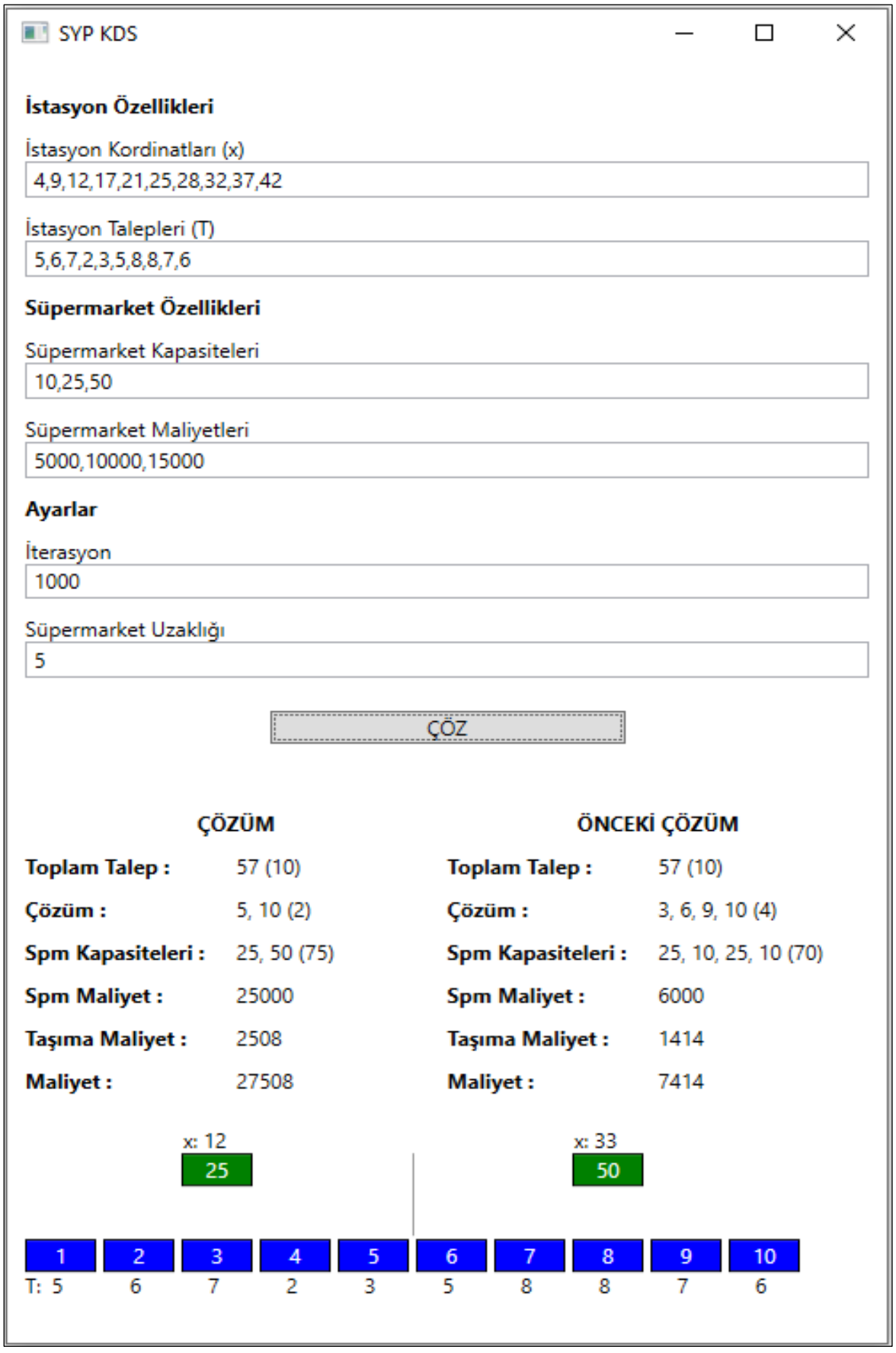
Resim 5.10. İki tip süpermarket içeren örnek uygulama sonucu



Resim 5.11. Üç tip süpermarket içeren örnek uygulama sonucu



Resim 5.12. Kurulum maliyetleri artırılmış örnek uygulama sonucu



Resim 5.13. Sistemde bir önceki çözümün gösterilmesi

Yukarıdaki örneklerde görülebildiği gibi farklı kapasitede süpermarketler aynı çözümde birlikte kullanıldığında, tek tip süpermarket kullanımına göre daha iyi sonuçlar verebilmektedir. Bu durum her zaman geçerli olmayabilir. Görüldüğü gibi sistem taşıma maliyeti ve süpermarket maliyet toplamını en küçüklemeyi amaçlamaktadır. Her yeni süpermarket açıldığında taşıma maliyeti azalırken, yeni bir süpermarket kurulum maliyeti toplam maliyete eklenmektedir. Sistem bu iki maliyet arasındaki dengenin en iyi olduğu noktayı yakalamaya çalışmaktadır. Montaj hattında istasyonların malzeme talebi ve istasyonlar arası mesafeler farklılık gösterebilir. Bazı bölgelerde parça talebi daha yoğunken, bazı bölgelerde istasyonlar arası mesafeler fazla olabilir. Bu gibi durumlarda farklı tipteki süpermarketler çözümü iyileştirebilir. Diğer taraftan sadece yüksek kapasiteli süpermarketler kullanıldığında, kullanılmayan kapasiteler doğabilir. Örneğin parça talebi toplamı 110 kutu olan bir montaj hattını düşünelim. Kapasitesi 100 kutu olan iki tane süpermarket açmak yerine, bir adet 100 kapasiteli, bir adet 10 kapasiteli süpermarket açmak az maliyetli olabilir. Bu durum sistemi kullanacak kullanıcıların problemlerine özeldir. Böyle durumlar oluşmadığında, sistem tek tip süpermarket içeren çözümü en iyi çözüm olarak önerecektir.

6. SONUÇ VE ÖNERİLER

Son yıllarda otomotiv sektöründen birçok üretici, merkezi depolardan karışık modelli montaj hatlarına parça taşımak yerine, esnek ve güvenilir parça tedariki sağlamak için süpermarket konseptini benimsemektedir. Süpermarket konseptine geçilirken çözülmesi gereken dört karar problemi vardır. Bunlar yerleşim, rotalama, çizelgeleme ve yükleme problemleridir. Yerleşim problemi fabrika zemine kurulacak süpermarketlerin yerlerinin ve sayısının belirlenmesi ve sonrasında montaj istasyonlarına atanması problemidir. Bu çalışma kapsamında yerleşim problemi ele alınmıştır.

SYP literatürde geçen, tesislerin müşterilere atanması problemi olan tesis yerleşim problemi ile benzerlik gösterir. Fakat bazı farklılıkları vardır. Örneğin bir tesis tarafından hizmet verilen müşteriler, SYP'de ardışıktır. Bir diğer farklılık ise, akış her zaman süpermarket ve montaj hattı arasındandır ve süpermarketler arasında malzeme akışı yoktur. SYP, Emde ve Boysen'in ortaya koyduğu şekilde, süpermarket sayısının hesaplanarak, hizmet verilen istasyonların ortasına yerleştirilmesi olarak bakıldığında polinom zamanda çözülebilmektedir [6]. Bununla birlikte, gerçekte bazı durumları da göz önünde bulundurmak gerekebilir. Örneğin fabrika zemininde süpermarket kurulumu için uygun olmayan bölgeler olabilir, süpermarketlerin belirli bir kapasitesi vardır ya da süpermarkette güvenlik stoğu bulunması istenebilir. Bu gibi durumlar göz önüne alınan diğer çalışmalarda, problemin karmaşıklığı arttığı için, GA ve dağılım tahmini algoritması gibi metasezgiseller kullanılmış ya da matematiksel modeller, modelleme araçlarıyla çözülmüştür [13–15].

Bu çalışmada SYP, Emde ve Boysen'in ortaya attığı şekilde ele alınmış ve sonrasında farklı kapasiteli süpermarketler aynı çözümde kullanılacak şekilde geliştirilmiştir [6]. Daha önceki çalışmalarda GA ve dağılım tahmini algoritması gibi popülasyon tabanlı metasezgiseller SYP çözümünde kullanıldığı için bu çalışmada tek çözüm tabanlı metasezgisellerden olan ve literatürde optimizasyon problemlerin çözümünde sıklıkla kullanılan TB algoritması kullanılmıştır.

Çalışmanın ilk aşamasında SYP, süpermarketlerin kapasitesi dikkate alınmadan ele alınmıştır. Emde ve Boysen'in önerdiği matematiksel modelden faydalanarak TB algoritması gelişilmiştir [6]. Algoritma kullanılarak C# programlama dili ile bir uygulama

geliştirilmiştir. Alnahhal ve Noche'un çalışmasında kullanılan yöntemle bir dizi test problemi oluşturulmuş ve geliştirilen uygulama aracılığıyla problemler çözülmüştür [13]. Algoritmanın başarısını ölçebilmek için, birerleme yöntemiyle en iyi sonuca ulaşabilen farklı bir uygulama geliştirilmiş ve test problemleri bu uygulama aracılığıyla da çözülmüştür. Birerleme uygulaması küçük ölçekli problemlerde en iyi sonuçlara ulaşabilmiş, fakat problemler büyüdüğünde uygulamanın çalışması günler süreceği hesaplanmış ve sonuç elde edilememiştir. Küçük ölçekli problemlerde, geliştirilen TB algoritması ve birerleme uygulamasının sonuçları karşılaştırıldığında, TB algoritmasının bir saniyenin altındaki zaman dilimlerinde, her zaman en iyi sonuçlara ulaştığı görülmüştür. Bu sayede geliştirilen algoritmanın büyük ölçekli problemlerde de en iyi sonuçlara eşit ya da yakın sonuçlar elde edeceği anlaşılmıştır.

Çalışmanın ikinci aşamasında, daha önceki çalışmalardan farklı olarak, farklı kapasitedeki süpermarketlerin kurulum maliyetleri değişebileceği ve aynı çözümde birlikte kullanılmalarının toplam maliyeti düşürebileceği düşüncesiyle algoritma geliştirilmiştir. Kullanılan matematiksel model, değişken kapasiteli süpermarketler aynı çözümde kullanılabilecek şekilde geliştirilmiştir. Algoritmanın performansını ölçebilmek için Nourmohammadi ve diğerlerinin çalışmasına benzer şekilde test problemleri oluşturulmuş ve algoritma kullanılarak geliştirilen bir uygulama aracılığıyla test problemleri çözülmüştür [15]. Uygulama sonucunda aynı problemde, farklı kapasitede süpermarketler birlikte kullanıldığında eşit ya da daha iyi çözümler elde edildiği görülmüştür. Çalışmanın ilk aşamasında olduğu gibi, algoritmanın performansını kıyaslayabilmek için bir birerleme uygulaması geliştirilmiş; fakat problem karmaşıklığının artmasından dolayı birerleme uygulamasının en küçük ölçekli test probleminde bile sonuca ulaşmasının günler süreceği anlaşılmıştır. Bu sebeple istasyon sayısının 1000 katı kadar rastgele çözüm içerisinden en iyisini seçen RA algoritması geliştirilmiş ve TB algoritmasının performansı bu uygulamanın sonuçları ile karşılaştırılmıştır. Karşılaştırma sonucunda RA algoritmasının küçük ölçekli test problemlerinde, TB algoritmasına nispeten yakın sonuçlara ulaştığı görülürken, istasyon sayısı arttığında RA algoritmasının daha kötü sonuçlar elde ettiği ve iki algoritma arasındaki yüzdesel farkın %250'yi geçtiği belirlenmiştir. Aynı zamanda, TB algoritması en fazla 2 saniye içinde çözüm üretebilirken, RA algoritmasının bazı durumlarda bir saatten fazla çalıştığı görülmüştür.

Geliştirilen TB algoritmasının başarılı şekilde çalıştığı anlaşıldıktan sonra, bu algoritmayı model olarak kullanan, bir model tabanlı KDS geliştirilmiştir. KDS'nin altyapısında C# programla dili ve Windows Presentation Foundation (WPF) teknolojisi kullanılmıştır. Sistem herhangi bir kurulum gerektirmeden, Windows işletim sistemi üzerinde çalışabilmektedir.

Geliştirilen KDS ile özellikle otomotiv sektöründe fabrika zemine süpermarket konseptine geçmek isteyen yöneticilere destek olmak amaçlanmıştır. Kullanıcı istasyon koordinatları ve talepleri ile süpermarket kapasite ve maliyetlerini girdikten sonra, kısa süre içerisinde fabrika zemine kurulacak süpermarket yerleri ve sayıları hakkında görsel ve anlaşılır bir çözüm önerisi alabilmektedir. Sistem tek tip ya da farklı kapasitedeki süpermarketler kullanarak çözüm üretebilmektedir. Bu sayede yönetici, kendi durumuna uygun olarak, süpermarketlerin kapasitelerine göre gerçek maliyetlerini kıyaslayarak, hangi kapasite ya da kapasitelerde süpermarket kullanmak istediğine karar verebilir. Aynı zamanda kullanıcı, TB algoritmasındaki iterasyon parametresine arayüzden müdahale ederek, programın çalışma süresini uzatarak daha hassas sonuçlar elde edebilir.

Bu çalışmada fabrika zemininde süpermarket kurulumu için uygun olmayan bölgeler göz ardı edilmiştir. Bundan sonraki çalışmalarda algoritma sadece fabrika zemininde süpermarket kurulumu için uygun olan bölgeler arasından seçim yapacak şekilde geliştirilebilir. Algoritmanın başarısı diğer metasezgisel algoritmalarla karşılaştırılabilir ve gerçek verilerle test edilebilir.

KAYNAKLAR

1. Langevin, A. ve Riopel, D., Ed. (2005). *Logistics Systems: Design and Optimization*. US: Springer.

2. Baudin, M. (2005). *Lean Logistics: The Nuts and Bolts of Delivering Materials and Goods*. CRC Press.

3. Boysen, N., Emde, S., Hoeck, M., ve Kauderer, M. (2015). Part logistics in the automotive industry: Decision problems, literature review and research agenda. *European Journal of Operational Research*, 242 (1), 107-120.

4. Battini, D., Gamberi, M., Persona, A., ve Sgarbossa, F. (2015). Part-feeding with supermarket in assembly systems: transportation mode selection model and multi-scenario analysis. *Assembly Automation*, 35 (1), 149-159.

5. Battini, D., Boysen, N., ve Emde, S. (2013). Just-in-Time supermarkets for part supply in the automobile industry. *Journal of Management Control*, 24 (2), 209-217.

6. Emde, S. ve Boysen, N. (2012). Optimally locating in-house logistics areas to facilitate JIT-supply of mixed-model assembly lines. *International Journal of Production Economics*, 135 (1), 393-402.

7. Özkan, A. ve Esmeray, M. (2002). Bir Maliyet Kontrol Sistemi Olarak JIT Üretim Sistemi ve Muhasebe Uygulamaları. *Cumhuriyet Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 3 (1), 129-146.

8. Emde, S., Fliedner, M., ve Boysen, N. (2012). Optimally loading tow trains for just-in-time supply of mixed-model assembly lines. *IIE Transactions*, 44 (2), 121-135.

9. Emde, S. ve Boysen, N. (2012). Optimally routing and scheduling tow trains for JIT-supply of mixed-model assembly lines. *European Journal of Operational Research*, 217 (2), 287-299.

10. Yano, C.A., Bozer, Y., ve Kamoun, M. (1998). Optimizing dock configuration and staffing in decentralized receiving. *IIE Transactions*, 30 (7), 657-668.

11. Wanke, P.F. ve Zinn, W. (2004). Strategic logistics decision making. *International Journal of Physical Distribution & Logistics Management*, 34 (6), 466-478.

12. Battini, D., Faccio, M., Persona, A., ve Sgarbossa, F. (2010). "Supermarket warehouses": stocking policies optimization in an assembly-to-order environment. *The International Journal of Advanced Manufacturing Technology*, 50 (5-8), 775-788.

13. Alnahhal, M. ve Noche, B. (2015). A genetic algorithm for supermarket location problem. *Assembly Automation*, 35 (1), 122-127.

14. Zhou, B.-H. ve Tan, F. (2019). A self-adaptive estimation of distribution algorithm with differential evolution strategy for supermarket location problem. *Neural Computing and Applications*, 1-14.

15. Nourmohammadi, A., Eskandari, H., Fathi, M., ve Aghdasi, M. (2018). A mathematical model for supermarket location problem with stochastic station demands. *Procedia CIRP*, 72, 444-449.

16. Nourmohammadi, A. ve Eskandari, H. (2017). Assembly line design considering line balancing and part feeding. *Assembly Automation*, 37 (1), 135-143.

17. Nourmohammadi, A., Eskandari, H., ve Fathi, M. (2019). Design of stochastic assembly lines considering line balancing and part feeding with supermarkets. *Engineering Optimization*, 51 (1), 63-83.

18. Vaidyanathan, B.S., Matson, J.O., Miller, D.M., ve Matson, J.E. (1999). A capacitated vehicle routing problem for just-in-time delivery. *IIE Transactions*, 31 (11), 1083-1092.

19. Kozan, E. (2000). An integrated material handling system for a truck assembly plant. *Journal of the Operational Research Society*, 51 (3), 263-271.

20. Choi, W. ve Lee, Y. (2002). A dynamic part-feeding system for an automotive assembly line. *Computers & industrial engineering*, 43 (1), 123–134.

21. Boysen, N. ve Bock, S. (2011). Scheduling just-in-time part supply for mixed-model assembly lines. *European Journal of Operational Research*, 211 (1), 15-25.

22. Golz, J., Gujjula, R., Günther, H.O., Rinderer, S., ve Ziegler, M. (2012). Part feeding at high-variant mixed-model assembly lines. *Flexible Services and Manufacturing Journal*, 24 (2), 119-141.

23. Faccio, M., Gamberi, M., Persona, A., Regattieri, A., ve Sgarbossa, F. (2013). Design and simulation of assembly line feeding systems in the automotive sector using supermarket, kanbans and tow trains: a general framework. *Journal of Management Control*, 24 (2), 187-208.

24. Faccio, M., Gamberi, M., ve Persona, A. (2013). Kanban number optimisation in a supermarket warehouse feeding a mixed-model assembly system. *International Journal of Production Research*, 51 (10), 2997-3017.

25. Rao, Y.-Q., Wang, M.-C., Wang, K.-P., ve Wu, T.-M. (2013). Scheduling a single vehicle in the just-in-time part supply for a mixed-model assembly line. *Computers & Operations Research*, 40 (11), 2599-2610.

26. Fathi, M., Alvarez, M.J., Hassani Mehraban, F., Rodriguez, V. (2014). A Multiobjective Optimization Algorithm to Solve the Part Feeding Problem in Mixed-Model Assembly Lines. *Mathematical Problems in Engineering*, 2014 (1), 1-12.

27. Fathi, M., Rodriguez, V., ve Alvarez, M.J. (2014). A novel memetic ant colony optimization-based heuristic algorithm for solving the assembly line part feeding problem. *The International Journal of Advanced Manufacturing Technology*, 75 (1-4), 629-643.

28. Dong, J., Zhang, L., Xiao, T., ve Li, H. (2015). A dynamic delivery strategy for material handling in mixed-model assembly lines using decentralized supermarkets. *International Journal of Modeling, Simulation, and Scientific Computing*, 06 (04), 1-19.

29. Dong, J., Zhang, L., ve Xiao, T. (2016). Part supply method for mixed-model assembly lines with decentralized supermarkets. *Tsinghua Science and Technology*, 21 (4), 426-434.

30. Emde, S. ve Gendreau, M. (2017). Scheduling in-house transport vehicles to feed parts to automotive assembly lines. *European Journal of Operational Research*, 260 (1), 255-267.

31. Morton, M.S.S. (1971). *Management Decision Systems: Computer-Based Support of Decision Making*. Boston, US: Harvard University Press.

32. Nof, S.Y. (Editör). (2009). *Springer handbook of automation*. New York, US: Springer.

33. Sprague, R.H. (1980). A Framework for the Development of Decision Support Systems. *MIS Quarterly*, 4 (4), 1.

34. Marakas, G.M. (2003). *Decision Support Systems in the 21st Century*. US: Prentice Hall.

35. Power, D.J. (2008). Decision Support Systems: A Historical Overview, Handbook on Decision Support Systems 1. Berlin, Germany: Springer, 121-140.

36. Power, D.J. ve Sharda, R. (2007). Model-driven decision support systems: Concepts and research directions. *Decision Support Systems*, 43 (3), 1044-1061.

37. Koutsoukis, N.-S., Mitra, G., ve Lucas, C. (1999). Adapting on-line analytical processing for decision modelling: the interaction of information and decision technologies. *Decision Support Systems*, 26 (1), 1-30.

38. Valente, P. ve Mitra, G. (2007). The evolution of web-based optimisation: From ASP to e-Services. *Decision Support Systems*, 43 (4), 1096-1116.

39. Shim, J.P., Warkentin, M., Courtney, J.F., Power, D.J., Sharda, R., ve Carlsson, C. (2002). Past, present, and future of decision support technology. *Decision Support Systems*, 33 (2), 111-126.

40. Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*. Hoboken, US: John Wiley & Sons.

41. Dréo, J., Ed. (2006). *Metaheuristics for hard optimization: methods and case studies*. Berlin, Germany: Springer.

42. Yang, X.-S. (2010). *Nature-inspired metaheuristic algorithms*. UK: Luniver Press.

43. Blum, C. ve Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 35 (3), 268–308.

44. Rao, S.S. (2009). *Engineering optimization: theory and practice*. Hoboken, US: John Wiley & Sons.

45. Özcan, U. ve Toklu, B. (2009). Balancing of mixed-model two-sided assembly lines. *Computers & Industrial Engineering*, 57 (1), 217-227.

46. Fathollahi-Fard, A.M., Govindan, K., Hajiaghaei-Keshteli, M., ve Ahmadi, A. (2019). A green home health care supply chain: New modified simulated annealing algorithms. *Journal of Cleaner Production*, 240.

47. Zaretalab, A. ve Hajipour, V. (2019). An Extended Simulated Annealing Based on the Memory Structure to Solve Redundancy Allocation Problem. *Journal of Advanced Manufacturing Systems*, 18 (4), 527-548.

48. Pace, R.D. (2019). A traffic control framework for urban networks based on within-day dynamic traffic flow models. *Transportmetrica A: Transport Science*, 16 (2), 234-269.

49. Xin, L. ve Xiaodong, M. (2018). An improved simulated annealing algorithm for interactive multi-objective land resource spatial allocation. *Ecological Complexity*, 36, 184-195.

50. Zhang, W., Maleki, A., Rosen, M.A., ve Liu, J. (2018). Optimization with a simulated annealing algorithm of a hybrid system for renewable energy including battery and hydrogen storage. *Energy*, 163, 191-207.

51. Wang, C., Mu, D., Zhao, F., ve Sutherland, J.W. (2015). A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup-delivery and time windows. *Computers & Industrial Engineering*, 83, 111-122.

52. Yang, D.-L. ve Mou, W. (1993). An Integrated Decision Support System in a Chinese Chemical Plant. *INFORMS Journal on Applied Analytics*, 23 (6), 93-100.

53. Jäger, K., Peemöller, W., ve Rohde, M. (1989). A decision support system for planning chemical production of active ingredients in a pharmaceutical company. *Engineering Costs and Production Economics*, 17 (1), 377-387.

54. Hamamoto, S. (1999). Development and validation of genetic algorithm-based facility layout a case study in the pharmaceutical industry. *International Journal of Production Research*, 37 (4), 749-768.

55. Juan, Y.-K., Gao, P., ve Wang, J. (2010). A hybrid decision support system for sustainable office building renovation and energy performance improvement. *Energy and Buildings*, 42 (3), 290-297.

56. Kolisch, R. ve Padman, R. (2001). An integrated survey of deterministic project scheduling. *Omega-International Journal of Management Science*, 29 (3), 249-272.

57. Ernst, A.T., Jiang, H., Krishnamoorthy, M., ve Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153 (1), 3-27.

58. Fikar, C., Gronalt, M., ve Hirsch, P. (2016). A decision support system for coordinated disaster relief distribution. *Expert Systems with Applications*, 57, 104-116.

59. Sama, M., D'Ariano, A., D'Ariano, P., ve Pacciarelli, D. (2014). Comparing Centralized and Rolling Horizon Approaches for Optimal Aircraft Traffic Control in Terminal Areas. *Transportation Research Record*, 2449 (1), 45-52.

60. Hopper, E. ve Turton, B.C.H. (2001). An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. *European Journal of Operational Research*, 128 (1), 34-57.

EKLER

EK-1. Kapasitesiz SYP parametre optimizasyonuna ilişkin analiz sonuçları

Çizelge 1.1. Kısaltmaların gösterimi

KT	Kareler toplamı
KO	Karaler ortalaması
Sd	Serbestlik derecesi
F	Ortalama karesel regresyonu
p	p değeri

Çizelge 1.2. Kapasitesiz SYP TB parametreleri için varyans analizi sonuçları - maliyete göre

Kaynak	**KT**	**Sd**	**KO**	**F**	**p**
T0	5680558878,914	2	2840279439,457	0,085	0,918
Tmin	121641450817,185	2	60820725408,593	1,830	0,163
α	401941826069,431	2	200970913034,716	6,046	0,003
T0 x Tmin	8258903766,716	4	2064725941,679	0,062	0,993
T0 x α	27087992478,024	4	6771998119,506	0,204	0,936
Tmin x α	42832050879,605	4	10708012719,901	0,322	0,863
T0 x Tmin x α	54837449408,197	8	6854681176,025	0,206	0,990

Çizelge 1.3. Kapasitesiz SYP TB parametreleri için varyans analizi sonuçları – CPU zamanına göre

Kaynak	**KT**	**Sd**	**KO**	**F**	**p**
T0	30,280	2	15,140	1,409	0,247
Tmin	123,317	2	61,658	5,740	0,004
α	1151,959	2	575,979	53,615	0,000
T0 x Tmin	40,041	4	10,010	0,932	0,446
T0 x α	58,510	4	14,628	1,362	0,248
Tmin x α	90,658	4	22,665	2,110	0,081
T0 x Tmin x α	51,539	8	6,442	0,600	0,778

EK-1.(devam) Kapasitesiz SYP parametre optimizasyonuna ilişkin analiz sonuçları

Çizelge 1.4. Kapasitesiz SYP TB parametreleri için Duncan gruplama sonuçları – maliyete göre

N	T0	T0 için alt kümeler	Tmin	Tmin için alt kümeler	α	α için alt kümeler	
		1		1		1	2
81	1000	135071,1852	0,1	116982,2222	0,99	91139,5309	
81	1500	143102,7160	1	136692,8889	0,95	142924,1481	142924,1481
81	500	146624,6173	10	171123,4074	0,90		190734,8395
	p	0,707		0,075		0,072	0,097

Çizelge 1.5. Kapasitesiz SYP TB parametreleri için Duncan gruplama sonuçları – CPU zamanına göre

N	T0	T0 için alt kümeler	Tmin	Tmin için alt kümeler		α	α için alt kümeler	
		1		1	2		1	2
81	1000	2,0741	10	1,5926		0,90	0,6543	
81	1500	2,4815	1	2,5679	2,5679	0,95	1,2840	
81	500	2,9383	0,1		3,3333	0,99		5,5556
	p.	0,115		0,060	0,139		0,223	1,000

EK-2. Değişken Kapasiteli SYP parametre optimizasyonuna ilişkin analiz sonuçları

Çizelge 2.1. Değişken Kapasiteli SYP TB parametreleri için varyans analizi sonuçları - maliyete göre

Kaynak	KT	Sd	KO	F	p
T0	36030869,608	2	18015434,804	0,004	0,996
Tmin	7590580,093	2	3795290,047	0,001	0,999
α	304852836,620	2	152426418,310	0,034	0,966
T0 x Tmin	1890626,853	4	472656,713	0,000	1,000
T0 x α	35515418,820	4	8878854,705	0,002	1,000
Tmin x α	41703470,112	4	10425867,528	0,002	1,000
T0 x Tmin x α	34307753,163	8	4288469,145	0,001	1,000

Çizelge 2.2. Değişken Kapasiteli SYP TB parametreleri için varyans analizi sonuçları – CPU zamanına göre

Kaynak	KT	Sd	KO	F	p
T0	7072733,377	2	3536366,689	1,809	0,165
Tmin	75959941,698	2	37979970,849	19,434	0,000
α	1384236397,155	2	692118198,578	354,143	0,000
T0 x Tmin	11882,796	4	2970,699	0,002	1,000
T0 x α	6970656,005	4	1742664,001	0,892	0,468
Tmin x α	65295423,191	4	16323855,798	8,353	0,000
T0 x Tmin x α	14608,908	8	1826,114	0,001	1,000

Çizelge 2.3. Değişken Kapasiteli SYP TB parametreleri için Duncan gruplama sonuçları – maliyete göre

N	T0	T0 için alt kümeler	Tmin	Tmin için alt kümeler	α	α için alt kümeler
		1		1		1
243	7500	118061,8189	10	118141,6872	0,99	117410,4691
243	5000	118197,0617	1	118321,5309	0,95	118476,5844
243	2500	118586,2716	0,1	118381,9342	0,90	118958,0988
	p	0,936		0,971		0,812

EK-2. (devam) Değişken Kapasiteli SYP parametre optimizasyonuna ilişkin analiz sonuçları

Çizelge 2.4. Değişken Kapasiteli SYP TB parametreleri için Duncan gruplama sonuçları – CPU zamanına göre

N	T0	T0 için alt kümeler	Tmin	Tmin için alt kümeler			α	α için alt kümeler		
		1		1	2	3		1	2	3
243	2500	1359,2634	10	1088,0658			0,90	342,9177		
243	5000	1498,2510	1		1490,2881		0,95		690,0412	
243	7500	1599,5514	0,1			1878,7119	0,99			3424,1070
	p	0,073		1,000	1,000	1,000		1,000	1,000	1,000

Çizelge 2.5. Eşlenik T Testi; TB – RA

	N	Ortalama	Standart Sapma	Standart Hata
TB	49	64156	51578	7368
RA	49	124593	126894	18128
Fark	49	-60436	83984	11998
95% Güven Aralığı: (-84559; -36313)				
T-Testi, Ortalama Farkları= 0 (vs ≠ 0): T-Hesap = -5.04 P-Değeri = 0.000				

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : Uysal, Mevlüt

Uyruğu : T.C.

Doğum tarihi ve yeri : [redacted]

Medeni hali : [redacted]

Telefon : [redacted]

Faks : [redacted]

e-mail : [redacted]



Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Doktora	Gazi Üniversitesi Yönetim Bilişim Sistemleri	Devam ediyor
Yüksek lisans	Gazi Üniversitesi Yönetim Bilişim Sistemleri	2014
Lisans	Gazi Üniversitesi Bilgisayar ve Öğr. Tek. Öğretmenliği	2006
Lise	Uşak Fen Lisesi	2002

İş Deneyimi

Yıl	Yer	Görev
2006-Halen	Gazi Üniversitesi	Öğretim Görevlisi

Yabancı Dil

İngilizce

Yayınlar

A. Ulusal hakemli dergilerde yayımlanan makaleler

1. Uysal, M. ve Özcan, U. (2019). Süpermarket Yerleşim Problemi İçin Tavlama Benzetimi Algoritması Yaklaşımı. *Karadeniz Fen Bilimleri Dergisi*, 9(1), 58-69.

ÖZGEÇMİŞ (devam)

2. Tanrıverdi, M., Uysal, M. ve Üstündağ M. T. (2019). Blokzinciri Teknolojisi Nedir? Ne Değildir?: Alanyazın İncelemesi. *Bilişim Teknolojileri Dergisi*, 12(3), 203-217.

3. Uysal, M. ve Güyer, T. (2014). İstatistiksel Veri Analizine İlişkin Genişleyebilir Bir Karar Ağacı Tasarımı. *Bilişim Teknolojileri Dergisi*, 7 (3), 33-43.

B. Ulusal bilimsel toplantılarda sunulan ve bildiri kitaplarında basılan bildiriler

1. Tanrıverdi, M., Uysal, M. ve Üstündağ M. T. (2019). *Öğrenciler için eğitim cüzdanı eğitim kurumları arasında öğrenci etkinliklerinin blokzincir üzerinde paylaşılması*. 13. Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumu, Kırşehir.

2. Tanrıverdi, M., Uysal, M. ve Üstündağ M. T. (2018). *Blokzincir teknolojisi ve eğitim alanında kullanılma potansiyeli.* 1. Uluslararası Çağdaş Eğitim ve Sosyal Bilimler Sempozyumu (ISCESS), Antalya.

Hobiler

Kitap okumak, müzik aletleri çalmak, spor yapmak



GAZİLİ OLMAK AYRICALIKTIR…