Hacettepe Üniversitesi Sosyal Bilimler Enstitüsü

İşletme Anabilim Dalı

Sayısal Yöntemler Bilim Dalı

ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİNİN TAMSAYILI DOĞRUSAL PROGRAMLAMA VE SEZGİSEL YAKLAŞIMLAR İLE ÇÖZÜMÜ

Akın ÖZKAN

Doktora Tezi

Ankara, 2019

ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİNİN TAMSAYILI DOĞRUSAL PROGRAMLAMA VE SEZGİSEL YAKLAŞIMLAR İLE ÇÖZÜMÜ

Akın ÖZKAN

Hacettepe Üniversitesi Sosyal Bilimler Enstitüsü

İşletme Anabilim Dalı

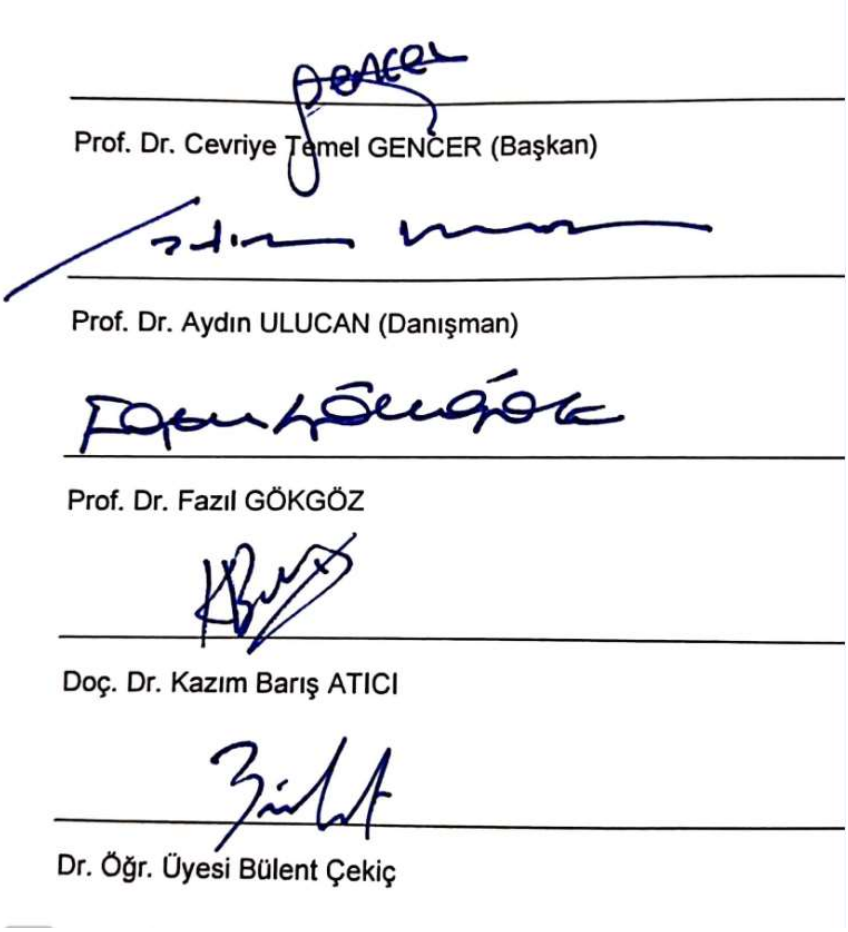
Sayısal Yöntemler Bilim Dalı

Doktora Tezi

Ankara, 2019

KABUL VE ONAY

Akın ÖZKAN tarafından hazırlanan "Üniversite Ders Çizelgeleme Probleminin Tamsayılı Doğrusal Programlama ve Sezgisel Yaklaşımlar ile Çözümü" başlıklı bu çalışma, 01/11/2019 tarihinde yapılan savunma sınavı sonucunda başarılı bulunarak jürimiz tarafından Doktora Tezi olarak kabul edilmiştir.



Yukarıdaki imzaların adı geçen öğretim üyelerine ait olduğunu onaylarım.

Prof. Dr. Musa Yaşar SAĞLAM

Enstitü Müdürü

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI BEYANI

Enstitü tarafından onaylanan lisansüstü tezimin tamamını veya herhangi bir kısmını, basılı (kağıt) ve elektronik formatta arşivleme ve aşağıda verilen koşullarla kullanıma açma iznini Hacettepe Üniversitesine verdiğimi bildiririm. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanım hakları bana ait olacaktır.

Tezin kendi orijinal çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinleri yazılı izin alınarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim Kurulu tarafından yayınlanan ***"Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge"*** kapsamında tezim aşağıda belirtilen koşullar haricince YÖK Ulusal Tez Merkezi / H.Ü. Kütüphaneleri Açık Erişim Sisteminde erişime açılır.

- Enstitü / Fakülte yönetim kurulu kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren 2 yıl ertelenmiştir. (1)
- Enstitü / Fakülte yönetim kurulunun gerekçeli kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren ….. ay ertelenmiştir. (2)
- Tezimle ilgili gizlilik kararı verilmiştir. (3)

01/11/2019

Akın ÖZKAN

"Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge"

*(1) Madde 6. 1. Lisansüstü tezle ilgili patent başvurusu yapılması veya patent alma sürecinin devam etmesi durumunda, tez **danışmanının** önerisi ve **enstitü anabilim dalının** uygun görüşü üzerine **enstitü** veya **fakülte yönetim kurulu** iki yıl süre ile tezin erişime açılmasının ertelenmesine karar verebilir.*

*(2) Madde 6. 2. Yeni teknik, materyal ve metotların kullanıldığı, henüz makaleye dönüşmemiş veya patent gibi yöntemlerle korunmamış ve internetten paylaşılması durumunda 3. şahıslara veya kurumlara haksız kazanç imkanı oluşturabilecek bilgi ve bulguları içeren tezler hakkında tez **danışmanının** önerisi ve **enstitü anabilim dalının** uygun görüşü üzerine **enstitü** veya **fakülte yönetim kurulunun** gerekçeli kararı ile altı ayı aşmamak üzere tezin erişime açılması engellenebilir.*

*(3) Madde 7. 1. Ulusal çıkarları veya güvenliği ilgilendiren, emniyet, istihbarat, savunma ve güvenlik, sağlık vb. konulara ilişkin lisansüstü tezlerle ilgili gizlilik kararı, **tezin yapıldığı kurum** tarafından verilir *. Kurum ve kuruluşlarla yapılan işbirliği protokolü çerçevesinde hazırlanan lisansüstü tezlere ilişkin gizlilik kararı ise, **ilgili kurum ve kuruluşun önerisi** ile **enstitü** veya **fakültenin** uygun görüşü üzerine **üniversite yönetim kurulu** tarafından verilir. Gizlilik kararı verilen tezler Yükseköğretim Kuruluna bildirilir.*
Madde 7.2. Gizlilik kararı verilen tezler gizlilik süresince enstitü veya fakülte tarafından gizlilik kuralları çerçevesinde muhafaza edilir, gizlilik kararının kaldırılması halinde Tez Otomasyon Sistemine yüklenir.

****** Tez **danışmanının** önerisi ve **enstitü anabilim dalının** uygun görüşü üzerine **enstitü** veya **fakülte yönetim kurulu tarafından karar verilir.***

ETİK BEYAN

Bu çalışmadaki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi, görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu, kullandığım verilerde herhangi bir tahrifat yapmadığımı, yararlandığım kaynaklara bilimsel normlara uygun olarak atıfta bulunduğumu, tezimin kaynak gösterilen durumlar dışında özgün olduğunu, **Prof. Dr. Aydın ULUCAN** danışmanlığında tarafımdan üretildiğini ve Hacettepe Üniversitesi Sosyal Bilimler Enstitüsü Tez Yazım Yönergesine göre yazıldığını beyan ederim.

Akın ÖZKAN

ÖZET

Özkan Akın. *Üniversite Ders Çizelgeleme Probleminin Tamsayılı Doğrusal Programlama ve Sezgisel Yaklaşımlar ile Çözümü*, Doktora Tezi, Ankara, 2019.

Üniversite ders çizelgeleme NP-Tam problem sınıfında olan ve her üniversitenin kendine özel gereksinimlerinden dolayı daha da karmaşıklaşan bir problem türüdür. Bu çalışmada, şubeli ve seçmeli derslere sahip olmanın yanında çoğu kaynağı ortak kullanan ve birçok bölüme sahip fakülte düzeyindeki ders çizelgeleme problemlerini tamsayılı doğrusal programlama (TDP) veya sezgisel yaklaşımlar ile çözebilen bir çözücü tasarlamak ve benzer yapıdaki problemlere kolayca uyarlanabilecek atama modelleri oluşturmak amaçlanmıştır. Bu kapsamda; derslerin zorunlu, şubeli zorunlu ve seçmeli olma durumlarını da dikkate alan TDP ve sezgisel modeller oluşturulmuştur. Tek aşamalı olarak sunulan TDP modelinde parametreler indislere göre gruplanarak oluşturulduğu için hem kısıt sayısı hem de karar değişken sayısı önemli ölçüde azaltılmıştır. Bu sayede, büyük ölçekli bir gerçek hayat problemi olan İktisadi ve İdari Bilimler Fakültesi (HÜİİBF) ders çizelgeleme problemi optimal olarak çözülebilmiştir.

Sezgisel çözüm için iki aşamalı algoritmalar kullanılmaktadır. İlk aşamada uygulanabilir bir başlangıç çözümü elde edilmekte, ikinci aşamada çözümün kalitesi iyileştirilmektedir. Başlangıç çözümü için, biri açgözlü sezgisel (a greedy heuristics-GH) ve bir diğeri iteratif ileri arama (iterative forward search-IFS) sezgiseli olan iki farklı yaklaşım kullanılmıştır. İyileştirme aşamasında ise birer yerel arama sezgiseli olan tabu arama (tabu search-TS) ve benzetim tavlaması (simulating annealing-SA) sezgiselleri kullanılmıştır.

Modellere ek bir kısıt (derslik sabitliği kısıtı) eklendiğinde problemin kompleksliği artmış ve TDP modeli ile optimal çözüme ulaşılamadığı görülmüştür. Sezgisel modeller kullanıldığında ise göreli olarak iyileştirilmiş çözümler elde edilmişidir. Sonuçlar, başlangıç çözümü aşamasında, çözüm süresi açısından GH algoritmasının IFS algoritmasından daha hızlı olduğunu, iyileştirme aşamasında ise SA sezgiselinin TS sezgiselinden daha iyi sonuçlar verdiğini göstermiştir.

Anahtar Sözcükler:

Üniversite Ders Çizelgeleme, Tamsayılı Doğrusal Programlama, Çizelgeleme, Tabu Arama, Benzetim Tavlaması

ABSTRACT

Özkan Akın. Solving University Course Timetabling Problems with Integer Linear Programming and Heuristic Approaches, PhD Thesis, Ankara, 2019.

University course timetabling is a problem type that is in the NP-Complete problem class and can become even more difficult due to the specific requirements of each university. In this study, it is aimed to design to a solver that can be used to solve university course timetabling problems which consists of multiple departments, have many common resources as well as have elective and multi-section courses at faculty level by using integer liner programming or heuristic approaches, and to develop assignment models that can be easily adapted to the problem in the similar structure. In this context, ILP and heuristic models that can take into account availability of mandatory, multi-section mandatory and elective of the courses were developed. In the ILP model, the number of decision variables and constraints have significantly reduced since the parameters were grouped according to indices. In this way, Faculty of Economics and Administrative Sciences at Hacettepe University course timetabling problem which is a large scale real-life problem could has been solved optimally. While this model is an exact algorithm, heuristics algorithms are two-stage. In the first stage, a feasible initial solution is obtained and in the second stage, the quality of the solution is improved. Two different approaches are proposed for the initial solution: a greedy heuristics (GH) and an iterative forward search (IFS). In the second stage, tabu search (TS) and simulating annealing (SA), which are local search heuristics, are used.

When an additional constraint (room stability constraint) has added to the models, the complexity of the problem has increased and the optimal solution could has not be reached with the ILP model. When heuristic models have used, relatively bettered solutions have obtained. The results have shown that the GH algorithm is faster than the IFS algorithm in the initial solution stage and that the SA heuristic give better results than the TS heuristic in the improvement stage.

Keywords:

University Course Timetabling, Integer Linear Programming, Timetabling, Tabu Search, Simulating Annealing

İÇİNDEKİLER

TABLOLAR DİZİNİ

ŞEKİLLER DİZİNİ

GİRİŞ

Çizelgeleme problemlerini çözmek günlük yaşamın pek çok alanında karşımıza çıkan önemli görevlerdendir. Eğitim, ulaşım, sağlık (hemşire atama) veya eğlence alanlarında sıkça gerek duyulan çizelgeleme aslında oluşturulması oldukça zor olabilen bir görevdir. Aynı zamanda çizelgeleme problemleri yapıları itibarıyla çözümü en zor olan NP-Tam problemler sınıfına girmektedir. Problemlerin boyutu ve kompleksliği artıkça problemlerin optimal çözümleri giderek zorlaşmakta ve kabul edilebilir süreler içerisinde bir sonuca ulaşılmamaktadır. Bu nedenle çoğunlukla sezgisel yaklaşımlara başvurulmaktadır. Ancak son yıllarda bilgisayardaki yazılım ve donanımların gelişmesiyle birlikte optimal sonuç veren yaklaşımlar da kullanılır olmaya başlamıştır.

Eğitimde çizelgeleme problemi üniversite çizelgeleme ve okul çizelgeleme problemleri şeklinde iki ayrı problem olarak görüldüğü gibi bu problemler de kendi içinde ders çizelgesi ve sınav çizelgesi olarak ikiye ayrılmaktadır. Ayrıca üniversite ders çizelgeleme problemi de kendi içinde kayıt temelli ve müfredat temelli üniversite ders çizelgeleme problemi olarak ikiye ayrılmaktadır. Kayıt temelli üniversite ders çizelgeleme probleminde hangi öğrencilerin hangi dersleri alacağı bilgisi önceden bilinir ve bu bilgi kullanılarak çakışmalar engellenmeye çalışılır. Müfredat temelli üniversite ders çizelgeleme probleminde ise hangi öğrencilerin hangi dersleri alacağı bilgisi önceden bilinmemektedir. Bunun yerine aynı müfredata sahip öğrencilerin (aynı dersleri alan öğrenci grubu) derslerinin çakışması engellenmeye çalışılır. Bu tez çalışmasında müfredat temelli üniversite ders çizelgeleme problemi ele alınmaktadır.

Gerçek hayat üniversite ders çizelgeleme problemleri çoğu kurumda kompleks bir yapı oluşturmaktadır. Bu problemleri elle çözmek çok sayıda personel ve iş yükü gerektirmekte, çokça zaman almakta ve çoğu kez tatmin edici bir sonuç vermemektedir. Bu nedenle, çözüm için etkin yaklaşımlar bulmak çoğu zaman zorunluluk olmaktadır. Bu problem için çok çeşitli yaklaşımların kullanıldığı görülmektedir. Bu yaklaşımlar kullanılarak geliştirilen algoritmalar bilgisayarlar yardımıyla çözülüp tatmin edici sonuçlar elde edilebilmektedir.

Üniversite çizelgeleme problemleri her kurumun yapısına göre farklılaştığı için genel bir problem yapısı oluşturmak ve bu problemler için tek bir optimizasyon yaklaşımı önermek yetersiz kalacaktır. Bunun yerine, değişen şartlara veya isteklere kolayca uyarlanabilecek modeller geliştirmek gerekir.

Bu tez çalışmasının ana amacı birçok bölüme sahip fakülte düzeyindeki ders çizelgeleme problemlerini hem tamsayılı doğrusal programlama (TDP) ve sezgisel yaklaşımlar ile çözebilecek bir çözücü tasarlamak ve benzer yapıdaki problemlere kolayca uyarlanabilecek atama modelleri oluşturmaktır. Bu amacı gerçekleştirmek için modelleme, çözücü oluşturma ve uygulama açısından birçok hedef belirlenmiştir. Aynı zamanda çalışmanın katkılarını oluşturan bu hedefler şunlardır:

Modelleme açısından hedefler:

- TDP modelini derslerin zorunlu, şubeli zorunlu ve seçmeli olma durumlarını da dikkate alacak şekilde modellemek,
- Büyük ölçekli problemlerde yetersiz hafıza (*out of memory*) durumuna düşmemek ve çözümü hızlandırmak için TDP modelinde kısıt ve karar değişkeni sayısını azaltacak yaklaşımlar bulmak ve uygulamak,
- Sezgisel modelleri derslerin zorunlu, şubeli zorunlu ve seçmeli olma durumlarını da dikkate alacak şekilde modellemek,
- Sezgisel çözüm için uygulanabilir bir başlangıç çözümü veren etkin yaklaşımlar bulmak ve ders çizelgeleme problemlerimize uyarlamak,
- Sezgisellerde kullanılacak komşuluk hareketlerini ve bu komşuluk hareketleri için uygun kombinasyonu belirlemek.
- İleri düzey komşuluk hareketi olan kempe zinciri (kempe chain) komşuluk hareketini problemimize uyarlamak ve çözüme katkısını test etmek,
- Sezgisellerin parametre testlerini yapmak ve uygun parametre değerlerini belirlemek,

- Sezgisellerinin performansını karşılaştırmak,
- TDP modeli ile sezgisel modellerin performansını karşılaştırmak,
- Problem kompleksliğinin artırılmasının çözüm üzerindeki etkisini test etmek,

Çözücü oluşturma açısından hedefler:

- Kolay anlaşılır standart bir veri giriş formatı oluşturmak,
- Veri giriş formatına uygun olarak girilen verileri modeller için uygun formatta parametrelere dönüştüren kodları yazmak,

Uygulama açısından hedefler:

- Bir gerçek hayat problemi olan Hacettepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi (HÜİİBF) ders çizelgeleme problemini bir bütün olarak ele alıp hem TDP ve hem de sezgisel modeller ile çözebilmek,
- HÜİİBF ders çizelgeleme probleminin özel gereksinimlerini modellere eklemek,

Üniversite ders çizelgeleme problemleri NP-Tam problem yapısında olması nedeniyle, bu problemlerin çözümü için yöneylem araştırması ve yapay zeka alanlarına giren pek çok yaklaşım kullanılmaktadır. Problemin kompleksliği ve boyutu artıkça optimal çözüm zorlaşmakta, sezgisel çözümlerin kalitesi de problemin yapısına göre değişkenlik göstermektedir. Dolayısıyla, bu problemler için tek bir optimizasyon yaklaşımı önermek yetersiz kalmaktadır. Bu nedenle, bu tez çalışmasında, hem TDP ve hem de sezgisel yaklaşımlar önerilmektedir. Bu yaklaşımlardan TDP modeli tek aşamalı tam bir modelken, sezgisel modeller iki aşamalıdır. Sezgisel yaklaşımlar için ilk aşamada uygulanabilir bir başlangıç çözümü elde edilmekte, ikinci aşamada çözüm kalitesi iyileştirilmektedir. İlk aşama için biri açgözlü (*a greedy heuristics*-GH) algoritma ve bir diğeri iteratif ileri arama (*iterative forward search*-IFS) algoritması olan iki farklı yaklaşım kullanılmaktadır. İkinci aşama da birer yerel arama sezgiseli olan Tabu arama (*Tabu search*-TS) ve Benzetim tavlaması (*Simulated annealing*-SA) sezgiselleri kullanılmaktadır. Bu sezgiseller son

yılarda bu tür problemler için sıkça kullanılan ve iyi sonuçlar elde edilen popüler sezgisel yaklaşımlardır.

Tezin geri kalan kısmı şu şekilde organize edilmektedir:

Birinci bölümde üniversite ders çizelgeleme probleminin genel tanımından, çeşitlerinden, bu probleme özgü genel kısıtlardan bahsedilmektedir. İkinci bölümde, üniversite ders çizelgeleme probleminde kullanılan çözüm yaklaşımlarından ve bu yaklaşımları kullanan birçok önemli çalışmadan bahsedilmektedir. Özellikle TDP veya sezgisel yaklaşım kullanan çalışmalara değinilmektedir. Bu bölümde ayrıca üniversite ders çizelgeleme probleminde kullanılan sezgisellerin bu problemi çözme prosedürleri açısından girebilecekleri temel kategorilerden ve problemi ele alış biçimlerinden bahsedilmektedir.

Üçüncü bölümde, ilk olarak çözülecek problemin genel yapısında ve veri giriş formatı verilmektedir. Devamında tamsayılı doğrusal programlama (TDP) modelinin karar değişkenleri, parametreler, kısıtları ve amaç fonksiyonu ayrıntılı bir şekilde anlatılmıştır. Son olarak, başlangıç çözümünde ve iyileştirme aşamasında kullanılan sezgisellerin işleyişleri ile komşuluk hareketlerinin yapıları anlatılmaktadır. Dördüncü bölümde, sezgisel modeller ve modellerde kullanılan komşuluk hareketleri için gerekli testler yapılmaktadır. Daha sonra, çalışmada kullanılan tüm modellerin performansları karşılaştırılmaktadır.

1. BÖLÜM
ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİ

Çizelgeleme (*timetabling*), planlama (*scheduling*), dizilim (*sequence*-ör: iş sıralama) ve listeleme (*rostering*- ör: nöbet listesi) kavramlarının arasındaki fark ve benzerlikleri anlayabilmek için Wren (1995)'nin, yapılan mevcut çalışmalardan hareketle ortaya koyduğu olduğu tanımlamalara bakılabilir. Wren (1995) bu kavramları şu şekilde tanımlamaktadır:

- **Planlama (*scheduling*),** kaynakları, kısıtlara uyacak ve bazı kaynak setlerinin toplam maliyetini minimize edecek şekilde, zaman-mekân içinde yer alan nesnelere tahsis etmektir. Bu problem türüne verilebilecek en yaygın örnekler: araç sayısını, sürücü sayısını ve/veya mesafeyi en aza indirerek toplam maliyetin minimize edildiği ulaşım planlama (*transport scheduling*) veya araç rotalama (*vehicle routing*) problemleri ve fiziksel kaynakların ve/veya zaman periyodu sayısının minimize edilmesinin istendiği atölye tipi planlama (*job-shop scheduling*) problemleridir.

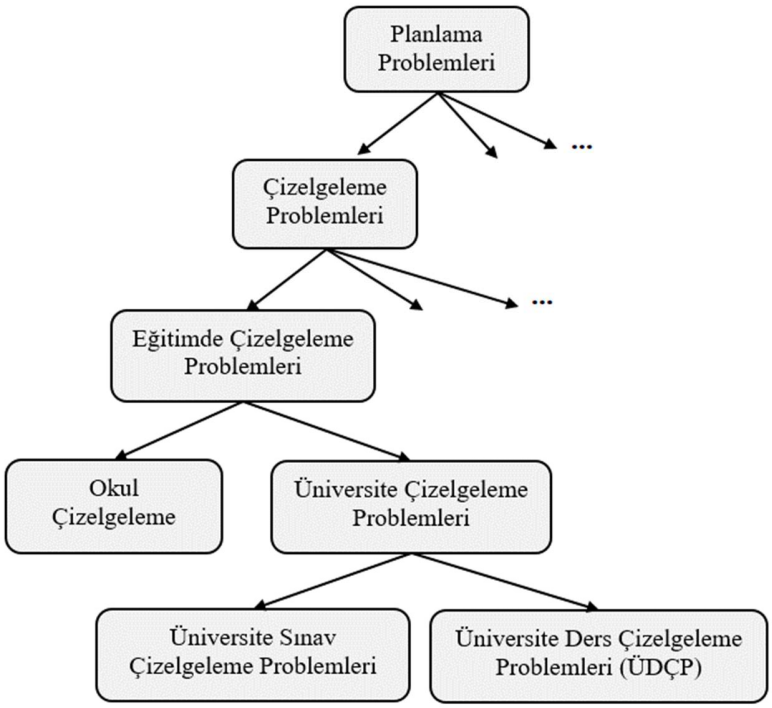
- **Çizelgeleme (*timetabling*)**, verilen kaynakları, kısıtlara uyacak ve arzulanan hedefleri mümkün olduğunca gerçekleştirecek şekilde, zaman-mekân içinde yer bulan nesnelere atamaktır. Ders ve sınav çizelgeleme ile personel atamanın bazı formları bu problem türüne örnek olarak verilebilir.

- **Dizilim (*sequence*),** kısıtlara uyarak, aktivitelerin hangi sıra ile yürütüleceğinin veya nesnelerin hangi sıra ile çözüme yerleştirileceğinin tasarlandığı problem türüdür. Bu problem türüne örnek olarak akış tipi planlama (*flow-shop scheduling*) ve gezgin satıcı problemi (*travelling salesman problem*) verilebilir.

- **Listeleme (*rostering*),** kısıtlara uyarak, kaynakları bir taslak içindeki periyotlara yerleştirmektir. Bu problemde bazı amaçlar minimize etmek ve/veya sadece uygulanabilir bir atama yapmak istenebilir. Çoğunlukla kaynaklar bir liste (ör: nöbet listesi veya nöbet cetveli) üzerinden dönmektedir. Örnek olarak hemşire atama problemi verilebilir.

40 yıla aşkın bir süredir hemşire atama (*nurse rostering*), spor müsabakalarının planlanması, ulaşımın planlanması, üniversite ders çizelgeleme ve üniversite sınav çizelgeleme gibi pek çok problem yöneylem araştırmasının konusu olmakta ve bu alanda önemli çalışmalar yapılmaktadır (Burke vd., 2007). Bu problem türleri çözümü en zor olan NP-Tam problemler sınıfına girmektedir. Bu nedenle bu tür problemleri optimal ve etkin bir şekilde çözebilmek oldukça zordur. Bir çizelgeleme problemi olan üniversite ders çizelgeleme problemi de çözüm uzayının ve karar değişkeni sayısının çok fazla büyümesinden dolayı NP-Tam problem olarak kabul edilmektedir (De Werra, 1997; Deris vd., 2000).

Eğitimde çizelgeleme problemlerinin bir çeşidi olan üniversite ders çizelgeleme problemi, belirli bir kısıt setini sağlayacak şekilde kaynakların tahsis edilmesini gerektiren bir kombinatoryal optimizasyon problemidir. Şekil 1'de görüldüğü gibi bu problem okul çizelgeleme ve sınav çizelgeleme problemleri ile birlikte eğitimde çizelgeleme ayrımının altında yer almaktadır (Schaerf, 1999).



Şekil 1. Üniversite Ders Çizelgeleme Problemi Diyagramı (Babaei vd., 2015)

Eğitimde Çizelgeleme problemleri üç ana kategoriye bölünebilir (Schaerf, 1999):

- **Okul çizelgeleme:** Aynı zaman diliminde bir öğretmenin iki farklı derslikte olmaması ve öğrenci gruplarının (sınıflarının) derslerini aynı dersliklerde işlemelerine benzer kısıtların olduğu, ilkokul veya lisenin bütün sınıfları için yapılan atama programı;

- **Üniversite ders çizelgeleme**: Ortak dersler alan öğrenci gruplarının derslerinin çakışmasını minimize edecek şekilde, üniversitedeki bütün ders setleri için yapılan program;

- **Üniversite sınav çizelgeleme:** Ortak dersler alan öğrenci gruplarının sınavlarının çakışmadığı ve öğrenciler için sınavların mümkün olduğu kadar sınav günlerine yayıldığı, üniversitedeki bütün ders setlerinin sınavları için yapılan program;

Üniversite ders çizelgeleme problemi, zaman periyotları ve dersliklere her bir dersin ders periyotları (ders saatleri) setini atamaktan oluşur. Okul çizelgeleme probleminden temel farkı üniversite dersleri ortak öğrenci grupları tarafından alınırken, okul sınıfları birbirinden ayrılmış öğrenci setlerinden oluşur. Okul eğitmenleri birden fazla dersi birden fazla sınıfa vermeleri gerekirken üniversitelerde öğretim elemanları genelde sadece bir ders verirler. Ayrıca, üniversite çizelgeleme probleminde dersliklerin (farklı boyut ve donanıma sahip olan) varlığı önemli bir rol oynarken, okul problemlerinde bunlar ihmal edilir. Çünkü çoğu durumda, her bir sınıf sadece bir dersliğe atanmıştır ve derslerin tümü o sınıf için o derslikte yürütülür (Müller, 2005).

Üniversitelerde çizelgeleme problemleri ders ve sınav çizelgeleme olmak üzere iki kategoriye ayrılmaktadır. Bu iki problem yapı itibariyle birbirine çok benzemesine rağmen, iki problem arasında bazı genel farklardan bahsedilebilir. Sınav problemi ders probleminden farklı olarak aşağıdaki karakteristik özeliklere sahiptir (Schaerf, 1999):

- Bu iki ayrım arasındaki temel fark, ders çizelgesi bir dönem için ve sadece haftanın beş günü ile sınırlı olacak şekilde planlanırken, sınav çizelgesi her sınav dönemi için birkaç güne, bir veya iki haftaya yayılabilecek şekilde planlanır.

- Her bir ders için sadece bir sınav vardır. Yani bir dersin sınavı sadece bir zaman periyodunda olmaktadır.
- Çakışmalar genellikle kabul edilemez. Bir öğrencinin bir derse girmemesi zor da olsa kabul edilebilirken, öğrencinin sınava girmemesi kabul edilemez.
- Farklı yapıda kısıtlar vardır. Örneğin her bir öğrenci için çoğunlukla günde bir sınav olur ve üst üste sınav olamaz.
- Sınav çizelgesinde zaman periyodu sayısı değişebilirken, ders çizelgesinde (yani bir haftalık) sabittir.
- Bir derslikte birden fazla sınav olabilir. Ancak ders çizelgesinde olamaz.

Üniversite ders çizelgeleme probleminin amacı, bütün kısıtları sağlayarak ders periyodlarını önceden belirlenmiş zaman periyotlarına ve dersliklere atamaktır. Dersler birden fazla periyottan ve pek çok sayıda kayıtlı öğrenciden oluşmaktadır. Öğretim elamanları tarafından yürütülen dersler, laboratuvar gibi özel ekipmanları sağlayan dersliklere de ihtiyaç duyabilir.

Üniversite ders çizelgeleme probleminin formülasyonu dersler ve zaman periyodu aralıkları açısından farklı yapılar üzerine inşa edilmektedir. Dersler açısından bakıldığında; dersler genelde dersleri alan öğrenci gruplarına göre gruplanmaktadır. Bu şekilde öğrenci gruplarının derslerinin farklı zaman dilimlerine atanması sağlanmaktadır. Bazı problemlerde de öğrencilerin hangi derslere kayıtlı oldukları önceden bilinebildiği için dersler arasında çakışma durumlarını gösteren matrisler hazırlanarak ortak öğrencilere sahip derslerin farklı zaman dilimlerine atanması sağlanmaktadır. Bu iki yapı sıklıkla iki genel yaklaşıma ayrılmaktadır; Müfredat temelli üniversite ders çizelgesi (*Curriculum-based university course timetabling*) ve Kayıt temelli üniversite ders çizelgesi (*Enrollment-based university course timetabling*) (Kristiansen ve Stidsen, 2013). Zaman periyodu aralıkları açısından bakıldığında; örneğin dört saatlik zaman dilimleri iki saatlik zaman dilimleri şeklinde gruplanarak dersler bu zaman dilimi gruplarına atanabilmektedir (Babaei vd., 2015). Bu uygulamamızda derslerin tümü üçer saatlik zaman dilimleri şeklinde tek blok olarak

işlenmektedir. Dolaysıyla üç saatlik zaman bir dilimi bir zaman periyodu olarak gruplanmaktadır.

1.1 MÜFREDAT TEMELLİ ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİ

Müfredat temelli üniversite ders çizelgeleme problemi belirli derslikler ve zaman periyotlarına derslerin haftalık olarak atanacak şekilde planlanmasından oluşmaktadır. Buradaki dersler üniversitenin yayınladığı müfredata göre setlere ayrılmaktadır. Her bir ders setindeki dersler bir öğrenci grubu tarafından alınmaktadır. Bu nedenle bu setlerdeki dersler birbiriyle çakışmayacak şekilde atanmalıdır (Di Gaspero vd., 2007; Müller ve Rudová, 2016). Dört yıllık bir fakültede her bir sınıf bir öğrenci grubu olarak kabul edilir. Çünkü bu sınıfların alacakları dersler çoğunlukla aynı olacaktır. Bu uygulamada da olduğu gibi bazı dersler seçmeli statüsünde olabilir. Bu nedenle, kısmen de olsa, bazı dersler için öğrenci grupları içerisinde alınan dersler farklılaşabilir. Böyle durumlarda seçmeli ders statüsündeki dersler için çakışmalar amaç fonksiyonunda minimize edilebilir. Aynı durum farklı müfredatlardan ders almak durumunda olunduğunda da geçerlidir. Müfredat temelli ders çizelgeleme yaklaşımı ders müfredatı belirlendikten ve ders çizelgesi hazırlandıktan sonra öğrencilerin ders kaydı yaptığı üniversiteler için uygun bir yaklaşımdır. Bu yaklaşımla ilgili pek çok çalışma bulunmaktadır; Daskalaki ve Birbas (2005); Kiefer vd. (2017); Lach ve Lübbecke (2012); Müller ve Rudová (2016).

1.2 KAYIT TEMELLİ ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİ

Kayıt temelli üniversite ders çizelgeleme problemi her bir öğrencinin bireysel kayıt verilerine dayanmaktadır. Bu yaklaşımda öğrenciler derslere kayıtlarını yaptıktan sonra ders çizelgesi hazırlanmaktadır (Lewis vd., 2007a). Derse kayıtlar önceden bilindiği için hangi derslerde çakışma olmaması gerektiği de net bir şekilde görülür. Bu çakışmalar bir matris yardımıyla modele kolayca dahil edilebilir. Bu yaklaşımla ilgili yapılan çalışmalara örnek olarak Ceschia vd. (2012), Goh vd. (2017) ve Lewis vd. (2007a) çalışmaları gösterilebilir.

Müfredat temelli üniversite ders çizelgesi ve kayıt temelli üniversite ders çizelgesi ayrımı eğitimdeki çizelgeleme problemleri üzerine düzenlenen 3 uluslararası çizelgeleme yarışması ile birlikte daha çok yapılır olmuş ve araştırmacıların eğitimde çizelgeleme problemlerine olan dikkatini artırmıştır.

İkinci uluslararası çizelgeleme yarışması 2007 (ITC-2007) yılında müfredat temelli üniversite ders çizelgesi, kayıt temelli üniversite ders çizelgesi ve sınav çizelgesi olmak üzere üç ayrı kategoride yapılmıştır (Di Gaspero vd., 2007; McCollum vd., 2010). Farklı kategorilerde yapılan bu yarışmalarda her bir kategorinin kuralları net bir şekilde belirlenmiştir. Bu kurallar ve kısıtlar gerçek uygulamalardan esinlenerek belirlenmiştir[1].

1.3 KISITLAR

Üniversite ders çizelgeleme probleminde kısıtlar, uygulanabilir bir çizelgeleme elde edebilmek için beklenen ön koşullar olarak ifade edilebilir. Literatüre bakıldığında kısıtların zorunlu ve esnek kısıtlar olarak iki kategoriye ayrıldığını görebiliriz. Zorunlu kısıtlardan kastedilen mutlaka yerine getirilmesi gereken kısıtlardır. Bu kısıtlara, bir öğretim elemanının bir zaman diliminde sadece bir dersliğe ve bir derse atanabilmesi, bir başka deyişle çakışmanın olmamasını sağlayan kısıtlar örnek gösterilebilir. Esnek kısıtlardan kastedilen şey ise gerçekleşmesi istenen fakat mecburi olmayan kısıtlardır. Bu kısıtlara öğretim elemanlarının veya öğrencilerin isteklerinin maksimum düzeyde karşılanmasını sağlayacak kısıtlar örnek gösterilebilir.

Bazı çalışmalarda yer alan zorunlu ve esnek kısıtlar aşağıda listelenmektedir (Asmuni, 2008; Babaei vd., 2015; Feizi-Derakhshi vd., 2012; Lewis vd., 2007b).

Zorunlu Kısıtlar

- Bir öğretim elemanı aynı zaman diliminde iki farklı dersliğe atanamaz.
- Bir ders aynı zaman diliminde iki farklı dersliğe atanamaz.

[1] Detaylı bilgiye şu internet sayfasından ulaşılabilir: http://www.cs.qub.ac.uk/itc2007/index.htm

- Her bir zaman periyodunda, bir öğretim elmanı sadece bir derslikte bulunmalı.
- Bazı zaman periyotları, bazı dersler için önceden ayrılmış olabilir.
- Dersi alan öğrenci sayısıyla dersin atanacağı derslik kapasitesi orantılı olmalı.

Esnek Kısıtlar

- Öğretim elemanı kendi dersi için belirli bir zaman periyodunu talep edebilir.
- Öğretim elemanının özel bir derslik isteği olabilir.
- Dersler, hem öğretim elemanlarının hem de öğrencilerin boş zaman periyotlarını minimize etmelidir.
- Dersler mümkün olduğunca son zaman periyoduna/akşam seansına atanmamalıdır.
- Öğretim elemanları için bir gün içerisinde önceden belirlenmiş bir saat yükünden daha fazla ders yükü olmamalıdır.
- Öğrenciler için bir gün içerisinde önceden belirlenmiş bir saat yükünden daha fazla ders yükü olmamalıdır.
- Planlama bir/bir grup öğrenciye bir gün veya bir zaman periyodu için ders atanmayacak şekilde yapılmalıdır.
- Aynı grup öğrencilere verilen dersler mümkün olduğunca araka arkaya atanmalıdır.
- Dersin atandığı dersliğin kapasitesi dersin kapasitesini karşılamalıdır.
- Aynı grup öğrencilere verilen dersle mümkün olduğunca aynı dersliğe atanmalıdır.

Yukarıdaki zorunlu kısıtlara üniversitelerin kendi gereksinimlerine göre başka kısıtlarda eklenebilir. Zorunlu kısıtlar bu problem için oldukça önemlidir. Çünkü bir zorunlu kısıt bile ihlal edilirse çözüm uygulanamaz (*infeasible*) olur.

Zorunlu kısıtların dışında kalan kısıtlar ise esnek kısıtlardır. Bu kısıtlar uygulanabilir bir çözüm elde etmek için gerekli değildir. Ancak çözümün kalitesi için oldukça önemlidirler. Yani esnek kısıtlar ne kadar az ihlal edilirlerse çözüm o kadar kaliteli olur. Uygulamada çözümün kalitesi ceza skoru ile ölçülmektedir. Rakamsal bir değer olan ceza skoru bir maliyet fonksiyonu ile hesaplanmaktadır. Maliyet fonksiyonu esnek kısıtların önem derecelerini gösteren ağırlık katsayıları ile bu esnek kısıtların ihlal sayılarının çarpımı

şeklinde hesaplanan doğrusal bir fonksiyondur. Böylece, çizelgeleme probleminin çözümünü iyileştirmek için amaç çizelgelemedeki atamaları değiştirerek ceza skorunu minimize etmek olabilir. Dolayısıyla, maliyet fonksiyonun değeri sıfır olan bir çözümde hiçbir esnek kısıtın ihlal edilmediğini ve en iyi çözümün elde edildiğini söyleyebiliriz. Ancak, uygulamada bu durum pek mümkün olmamaktadır. Çünkü esnek kısıtların tümü her zaman karşılanamamaktadır. Örneğin öğretim elemanlarının genelde çoğu cuma günü dersinin olmasını istememektedirler ve bu durumun tamamen sağlanması çoğu durumda imkânsızdır.

Kurumların kendine özgü gereksinimlerine göre değişkenlik gösterebilen esnek kısıtlar bir bakıma öğrenciler ve öğretim elemanları için kolaylık oluşturma isteğidir. Üniversitelerin özel gereksinimleri esnek kısıtlara verilecek önem ağırlıklarını da değiştirebilmektedir. Bazı esnek kısıtlar diğerlerine göre daha önemli olabilmekte ve üniversiteden üniversiteye bu önem düzeyleri değişebilmektedir. Bu farklılaşmayı sadece üniversite genelinde düşünmemek gerekir fakülte ve bölüm bazında da farklılaşma olabilmektedir.

Corne vd. (1995)'e göre üniversite çizelgeleme problemlerindeki zorunlu ve esnek kısıtlar beş ana kategoriye ayrılabilir: tekli, ikili, kapasite, olay (ör: dersler) yayılımı ve kullanıcı kısıtları.

- Tekli kısıtlar sadece bir olayın bulunduğu kısıtlardır. Bu kısıtlara, "bir *a* olayı Cuma gününde yer almamalı" veya "*a* olayı *b* zaman periyodunda yer almalı" örnek olarak verilebilir.
- İkili kısıtlar olay çiftlerinin bulunduğu kısıtlardır. Bu kısıtlara, "bir *a* olayı *b* olayından önce olmalı" veya "bir olay aynı zaman diliminde iki farklı yere atanmamalı" örnek olarak verilebilir.
- Kapasite kısıtları derslik kapasitelerinin dikkate alındığı kısıtlardır. Örneğin "bütün olaylar (ör: dersler veya ders periyotları) yeterli kapasitedeki dersliklere atanmalı".
- Yayılım kısıtları çizelgedeki olayların bazı günlere yayma, belli bir güne toplama veya bazı olayların arka arkaya gelmesini sağlayan kısıtlarıdır. Bu kısıtlar öğrencilere kolaylığın, öğretim elemanları iş yükünün ve/veya kurumun çizelgeleme politikasının gereği olarak düzenlenmektedir.

- Kullanıcı kısıtları çizelgeyi kullanacakların tercihlerini ve gereksinimlerini gerçekleştirmeyi sağlamaya çalışan kısıtlardır. Bu kısıtlara “bir öğretim elemanı Salı sabahları ders vermeyi seviyordur” veya “başka bir öğretim elemanın haftanın üç sabahında dersi olmamalıdır” isteği örnek olarak verilebilir.

2. BÖLÜM
ÜNİVERSİTE ÇİZELGELEME PROBLEMLERİNDE KULLANILAN ÇÖZÜM YAKLAŞIMLARI

Üniversitelerde genelde çizelgeleme problemlerini daha küçük alt problemlere (fakülte veya bölüm bazında) bölerek çözme stratejisi uygulanmaktadır. Büyük üniversitelerde çizelgeleme problemi çoğunlukla bölüm bazında ele alınmaktadır. Her bölüm kendi çizelgesini oluşturur. Daha sonra tüm bölümlerin çizelgesi birbirine entegre edilir. Neticede fakülte içerisinde ve fakülteler arasında çakışmaların olamaması ve uygulanabilir bir çözümün elde edilmesi gerekmektedir.

Bunu yapmanın iki farklı yolu olabilir; ilki, bir bölüm kendi çizelgesini oluşturduktan sonra sırasıyla diğer bölümler önceki bölümlerin çizelgelerini dikkate alarak kendi çizelgelerini oluşturabilirler. Bu yaklaşımda ortaya çıkabilecek sorun, sonraki bölümler daha kısıtlı kaynakları kullanacaklardır ve dolayısıyla çizelgeyi yapmak daha zor olacaktır. Eğer sonraki çizelge uygulanabilir olmadığı zaman önceki çizelgelerin revize edilmesi gerekecektir. İkincisi, bazı kaynaklar (ör: derslikler) önceden paylaştırıldıktan sonra her bir bölüm kendi çizelgesini oluşturabilir. Bu yaklaşımda da kaynakların doğru paylaşılamamasından kaynaklı olarak uygulanamaz çizelgeler oluşturulabilir. Neticede, uygulanabilir bir çözüm elde etmek için kaynak paylaşımının tekrar tekrar revize edilmesi gerekir. Uygulanabilir bir çözüm oluşturulsa bile esnek kısıtların çoğu sağlanamamış olabilir. Tüm bu nedenlerden dolayı problem büyüdükçe çözümün daha da zorlaştığı görülebilir. Sonuç olarak problemi bütüncül olarak ele almak ve kabul edilebilir bir çözüm süresi içinde kabul edilebilir bir çözümün elde edilme gerekliliği açıkça ortaya çıkmaktadır.

Çizelgeleme problemi arama problemi (*search problem*) veya optimizasyon problemi (*optimization problem*) olarak formüle edilebilir. Arama problemi olarak formüle edildiğinde, amaç basit bir şekilde, uygulanabilir (*feasible*) bir çözüme ulaşmak ve ulaşılan çözümü geliştirmek olmaktadır. Yani başlangıçta amaç bütün zorunlu kısıtların tam olarak sağlanması ve sonra esnek kısıtların mümkün olduğunca sağlanmasıdır. Optimizasyon

problemi olarak formüle edildiğinde, ise amaç, uygulanabilir bir çözümün yanı sıra esnek kısıtların amaç fonksiyonunda maksimize veya minimize edilmesidir (Schaerf, 1999).

Üniversite ders çizelgeleme probleminin çözüm sürecini karmaşık (kompleks) yapan bazı etmenler vardır. İlk olarak, bu problem NP-Tam problem yapısındadır. Cooper ve Kingston (1995) tamamen bağımsız beş farklı yöntemle bu problemin NP-Tam problem yapısında olduğunu kanıtlamışlardır. Yani, öğrenci ve öğretim elemanı sayılarının artışı gibi faktörlere bağlı olarak problemin boyutu üstel şekilde artabilmekte ve problem polinomsal zamanda çözülememektedir. Bu nedenle, çoğu üniversite ders çizelgeleme probleminde sezgisel yaklaşım kullanmak zorunlu olmaktadır. İkinci olarak, bir kurumdan diğerine, bu problemin kısıt (zorunlu, esnek) sayıları değişmektedir. Dolayısıyla her problem için farklı model gerekmekte ve çözüm için kullanılacak algoritmanın performansı problemden probleme farklılık gösterebilmektedir. Ancak, çözüm için kullanılan bütün algoritmaların temel amacı esnek kısıtları gerçekleştirme oranını maksimize etmektir. Bu aynı zamanda çözümün kalitesini de göstermektedir (Babaei vd., 2015; Feizi-Derakhshi vd., 2012).

Üniversite çizelgeleme problemleri her kurumun yapısına göre farklılaştığı için genel bir problem yapısı oluşturmak ve tek bir optimizasyon yaklaşımı kullanarak çözmek zordur. Hem bu nedenle ve hem de problemin NP-Tam problem yapısında olması nedeniyle, bu problemin çözümü için farklı yapıda modeller ve farklı çözüm yaklaşımları kullanılmaktadır.

Üniversite çizelgeleme probleminde kullanılan çözüm yaklaşımlarını Feizi-Derakhshi vd. (2012) ve Babaei vd. (2015)'nin çalışmalarında ifade ettiği gibi üç gruba ayırabilir: İlk grupta yöneylem araştırmasını temel alan yaklaşımalar yer almaktadır. Bu yaklaşımlara örnek olarak Graf renklendirme (*Graph coloring*) (Dandashi ve Al-Mouhamed, 2010; De Werra, 1997; Welsh ve Powell, 1967), Tamsayılı doğrusal programlama (Al-Yakoob ve Sherali, 2006, 2007; Daskalaki ve Birbas, 2005; Daskalaki vd., 2004; Dimopoulou ve Miliotis, 2001, 2004), Hedef programlama (Badri vd., 1998; Şahin, 2004) ve Kısıt programlama (*Constraint satisfaction programming*) (Deris vd., 2000; Kang ve White, 1992; Zhang ve Lau, 2005) yöntemleri gösterilebilir. İkinci grupta sezgisel yaklaşımlar yer almaktadır. Bu yaklaşımlara örnek olarak, başta Tabu arama (Aladag vd., 2009; Alvarez-Valdes vd., 2002; Hao ve Benlic,

2011) ve Genetik algoritma (Deris vd., 1999; Wang, 2002) yöntemi olmak üzere pek çok sezgisel yaklaşım gösterilebilir. Üçüncü grupta ise "modern akıllı yöntemler" (*modern intelligent methods*) yer almaktadır. Bu yöntemlere örnek olarak, Hibrit (Abdullah vd., 2007; Badoni vd., 2014; Kohshori ve Abadeh, 2012) ve Bulanık Teori (Asmuni vd., 2005; Chaudhuri ve De, 2010) temelli yöntemler gibi pek çok yöntem gösterilebilir.

2.1 YÖNEYLEM ARAŞTIRMASI METOTLARI

2.1.1 Tamsayılı Doğrusal Programlama

Tamsayılı doğrusal programlama (TDP) yöntemi kullanılarak üniversite çizelgeleme problemleri bir şekilde modellenebilmektedir. Ancak, güvenilir ve iyi bilinen bir teknik olan TDP dikkatli bir şekilde ele alınmalıdır. Aksi durumda bu problemlerin çözümü için çok fazla bilgisayar zamanı gerekebilir (MirHassani, 2006). Yine de fazlaca hesaplama gerektiren büyük ölçekli çizelgeleme problemleri için bu yaklaşım tek başına yetersiz kalmaktadır. Genellikle bu metot diğer metotlar ile birlikte hibrit bir şekilde kullanılmaktadır (Babaei vd., 2015).

Son yıllarda bilgisayardaki yazılım ve donanımlar gelişmesiyle birlikte TDP modelleri tekrardan pek çok kombinatoryal (*combinatorial*) problem için kullanılır olmaya başlamıştır (MirHassani ve Habibi, 2013). Üniversite ders çizelgeleme problemini TDP yardımıyla çözmeye çalışan pek çok çalışmaya rastlamak mümkündür.

De Werra (1985) çalışmasında üniversite ders çizelgeleme problemi için basit bir doğrusal model vermektedir. Bu modeldeki kısıtlar sırasıyla; (1) her dersin ders saati kadar çizelgede yer almasını, (2) tüm dersliklerin eşit kapasitede olduğu varsayımıyla birlikte bir zaman dilimindeki ders sayısının derslik sayısını geçmemesini ve (3) ortak ders alan öğrenciler birer grup olarak ele alınmasıyla birlikte gruptaki öğrencilerin derslerinin çakışmamasını sağlamaktadır. Ancak, gerçek bir uygulama düşünüldüğünde bu basit modele bir çok durumun dahil edilmediği görülecektir. Schaerf (1999) bu durumlardan yaygın olanların; kaynakların mevcut olmaması, önceden atanma durumu, dersler için çoklu şube ve zaman

periyodu uzunluğu gibi durumların olacağından bahsetmektedir. Bu tez çalışmasında ele aldığımız uygulama için de benzer durumlar söz konusudur. Örneğin bazı dersliklerin ve bazı öğretim elemanlarının belirli zaman dilimlerinde mevcut olmama durumu söz konusudur. Ayrıca bazı dersler öğrenci sayasılarının fazla olması durumundan dolayı birden fazla şubeye ayrılmışlardır. Bu gibi özel durumlar problemi daha da zorlaştırmaktadır.

Dimopoulou ve Miliotis (2001) çalışmalarında dersleri ve zaman periyotlarını gruplayarak sınav çizelgeleme problemini doğrusal programlama yaklaşımı ile çözüp elde edilen çözümün kalitesini artırmak içinde bir sezgisel yaklaşım kullanmaktadır. Daskalaki vd. (2004) ve Bakır ve Aksop (2008) ele aldıkları ders çizelgeleme problemini TDP ile çözebilmek için başlangıçta veriler indislere (ders, zaman periyodu, gün, derslik, öğrenci grubu ve öğretim üyesi gibi) göre alt gruplara ayırmak yoluyla çözüm uzayını ciddi anlamda azaltmış ve daha sonra problemi optimal olarak çözebilmişlerdir. Daskalaki ve Birbas (2005) çalışmasında ise ek olarak TDP temelli iki aşamalı bir gevşetme modeli geliştirmişlerdir. İlk aşamada, derslerin oturumları arka arkaya gelecek şekilde bir çözüm oluşturulmaktadır. İkinci aşamada ise model, derslerin araka arkaya gelecek periyotlarının birden fazla sayıda olmasını sağlamaktadır.

MirHassani (2006) gerçek bir üniversite ders çizelgeleme problemini TDP kullanarak çözmektedir. Bu problemde, çözüm uzayının büyümesinden kaçabilmek için derslik atamaları ve öğrenci istekleri modele dolaylı olarak eklenmiştir. Örneğin derslikler türlerine göre gruplanıp, daha sonra bir zaman dilimine atanacak derslerin sayısı ihtiyaç duydukları derslik sayısından daha fazla olmamasını sağlayacak kısıtlar modele eklenmiştir.

Al-Yakoob ve Sherali (2006) çeşitli ders çizelgeleme problemi örneklerini ilk önce geliştirildikleri tek aşamalı ve iki aşamalı doğrusal model ile çözmeye çalışmışlar. Bu modellerde üniversite politikasına göre derslerin atandığı zaman periyotları önceden bellidir. Amaç öğretim elemanlarının ve öğrencilerin istekilerini dikkate alarak öğretim elemanlarını derslere atamaktır. Özellikle tek aşamalı modelde daha çok olmak üzere bazı örnekler için yetersiz hafıza (*out of memory*) hatası alınmıştır. Diğer örnekler için sadece optimale yaklaşılabilmiştir. Son aşamada doğrusal programlamaya dayalı bir sezgisel geliştirilerek

daha iyi çözümlere ulaşılmaya çalışmışlardır. Al-Yakoob ve Sherali (2007) üniversite ders çizelgeleme problemini trafik yoğunluğu ve uygulanan cinsiyet politikasını da dikkate alacak şekilde TDP yaklaşımı ile çözmüşledir. Bazı senaryolar için yetersiz hafıza (*out of memory*) hatası problemi ile karşılaşılıncaya kadar çözüme devam edilmiş ve sonrasında çözüm süreleri ve optimal çözümden olan uzaklıklar kaydedilmiştir.

Schimmelpfeng ve Helber (2007) gerçek bir ders çizelgeleme problemini bir bütün olarak çözebilecek bir TDP modeli oluşturmuştur. Bu model, yaklaşık 150 ders, 100 öğretim elemanı ve 650 öğrencilik oluşan peoblemi çok sayıda kısıt ve yaklaşık 100,000 ikili değişken veya tamsayılı değişken ile birlikte dakikalar içinde çözebilmiştir. Çalışmada ayrıca, farklı senaryolar ile büyük ve küçük derslik sayılarını azaltıp çözümün kalitesi gözlemlenmiştir. Son olarak elde edilen çözüm öğretim elemanlarına ve asistanlara sorularak memnuniyet düzeylerine bakılmıştır. Elde edilen sonuçlar problemin bu yaklaşım ile çözülmesinin daha iyi olacağını ortaya koymuştur.

Lach ve Lübbecke (2008) ilk aşamada dersler zaman periyotlarına atandığı, ikinci aşamada bu derslerin derslikler ile eşleştirildiği bir sezgisel TDP yaklaşımı geliştirmişlerdir. Bu yaklaşım kullanılarak ikinci uluslararası çizelgeleme yarışması (ITC-2007) çerçevesinde kullanılan ilk 7 müfredat temelli üniversite ders çizelgeleme problemi çözülmüştür. Daha sonra, küçük orta büyük 3 problem seçilmiş modelin etkinliği test edilmiştir. 240,000 karar değişkenli ve 80,00 kısıtlı problem bir gün çalıştırılmış sonuç alınamamıştır. Diğer problemler ise sırasıyla %2 ve %7 oranında optimale yakın sonuç vermişlerdir. Ayrıca bu yaklaşımda öğretim elemanlarının istekleri amaç fonksiyonunda dikkate alınmış %80 oranında ilk zaman periyodu tercihleri gerçekleşmiştir.

Broek vd. (2009) gerçek bir üniversite ders çizelgeleme problemini TDP'ye dayalı bir yaklaşım ile çözmüşlerdir. Bu yaklaşım ekografik optimizasyon (*exicographical optimization*) kullanarak problemi dört alt probleme ayırıp çözümü bir bütün olarak vermektedir. Elde edilen çözüm öğrenci ve öğretim elemanlarını büyük oranda tatmin etmiştir.

Phillips vd. (2015) üniversite ders çizelgeleme probleminde derslik atama problemi için bir tamsayılı doğrusal programlama modeli önermişlerdir. Bilindiği gibi büyük ölçekli üniversite ders çizelgeleme problemleri TDP ile çözebilmek için genellikle önce dersleri zaman periyotlarına, sonrada dersleri dersliklere atayacak şekilde tamsayılı modeller ayrıştırılmaktadır (Phillips vd., 2015). Phillips vd. (2015) çalışmasında dersleri tercih ve özelliklere göre zaman periyotlarına atandıktan sonra dersleri dersliklere atama problemini TDP ile yapmaktadır. Bu yaklaşım hem bir gerçek uygulamada ve hem de ikinci uluslararası çizelgeleme yarışması (ITC2007) örnekleri üzerinde test edilmiştir.

2.1.2 Graf Renklendirme (*Graph Coloring*)

Graf renklendirme yaklaşımı kullanılarak çizelgeleme problemlerinin nasıl modellenebileceği De Werra (1985) çalışmasında ayrıntılı bir şekilde gösterilmektedir. Bilindiği gibi graf teorisinde düğümler (*nodes*) ve bunları birbirine bağlayan yaylar (*arcs*) vardır. Bu yaklaşımda verilen renkler ile bu düğümler boyanmaya çalışılmaktadır. Ders çizelgeleme problemini düşünürsek; her bir düğüm bir dersi, düğümler arasındaki yaylarda hangi dersler arasında çakışma (aynı öğretim elemanı tarafından verilen ve ortak öğrenci grubu tarafından alınan dersler gibi) olduğunu gösterecektir. Renk sayısı da verilen zaman periyodu sayısı kadar olacaktır. Bu yaklaşımda amaç birbirine yaylar ile bağlı olan düğümler aynı renge boyanmayacak şekilde bütün düğümleri sınırlı renkler ile boyayabilmektir.

De Werra (1996) çalışmasında graf renklendirme yaklaşımının çeşitli varyasyonlarını kullanarak ders çizelgeleme problemine uygulamıştır. Asratian ve de Werra (2002) dersleri çeşitli gruplara böldükten sonra bu yaklaşımı kullanarak çizelgeleme problemini çözmüştür. Graf renklendirme kullanılarak yapılan benzer pek çok uygulama Burke ve De Werra (2013)'nin çalışmasında incelenmektedir. Bu uygulamalara bakıldığında aslında graf renklendirmenin çeşitli modern meta-sezgisellerin bir parçası olarak kullanıldığı görülecektir (Qu vd., 2009). Örneğin; çakışmaların olmadığı bir çizelge inşa etme sürecinde kullanılan sezgisellerin genelde graf renklendirme yaklaşımına dayandığı görülmektedir. Bu sezgiseller genellikle atanması en zor olan olayları (ör: dersler) tahmin edip, daha sonra bu olayları

zorluk derecesine göre önceleyerek atama mantığı ile çalışmaktadır (Petrovic ve Burke, 2004).

2.1.3 Kısıt Programlama (*Constraint satisfaction programing*)

Kısıt programlama, kısıtlara (kaynakların sınırlı olduğunu ifade eden) dayalı bir hesaplama sistemidir. Bu metottaki amaç kısıtları ihlal etmeden karar değişkenlerine atanabilecek değerler içerisinden tutarlı bir değer seti bulmaktır (Babaei vd., 2015). Bu kısıtlar çerçevesinde birden fazla alternatif çözüm olabilir. Bu çözümlerden herhangi birine ulaşmak yeterli görülmektedir.

Bu metotta TDP ve graf renklendirme metotları gibi genelde tek başına değil diğer metotların bir parçası olarak kullanılmaktadır (Qu vd., 2009). Örneğin: Deris vd. (2000) 378 zaman periyoduna, 1673 ders periyoduna (lectures) ve 10 dersliğe sahip bir ders çizelgeleme problemine nesne yönelimli bir yaklaşımla modelleyip kısıt programlama tekniğini uygulamıştır. Deris vd. (1999) çalışmasında ise kısıt programlama metodunu genetik programlama (*genetic programing*) sezgiseli ile birleştirerek uygulanabilir çözüm uzayını önemli düzeyde küçültebilmiştir. Zhang ve Lau (2005) üniversite çizelgeleme problemini çözmek için kısıt programlamaya dayalı bir yazılım geliştirmiştir.

2.2 SEZGİSEL YAKLAŞIMLAR

Belli başlı sezgisel (*heuristic*) yaklaşımlardan bahsetmeden önce bu yaklaşımlarla ilgili bazı kavramlara değinmek faydalı olacaktır.

Sezgisel: Problemin karmaşık veya NP-Tam özeliğinde olması gibi pek çok nedenden dolayı, problemi optimal olarak çözemediğimiz veya optimal çözüm elde edebilmek için çok fazla çözüm zamanına ihtiyaç duyduğumuz durumlarda yüksek kalitede bir çözüm elde edebilmek için bir çözüm yaklaşımına ihtiyaç duyulmaktadır. Bu amaca hizmet eden sezgisel yaklaşımlar aslında optimal çözümü ve uygulanabilirliği garanti etmeyebilir.(Burke ve Kendall, 2005). Tabu arama (*tabu search*), benzetim tavlaması (*simulated annealing*),

genetik programlama (*genetic programing*) gibi yaklaşımlar bu sezgisellere örnek olarak verilebilir.

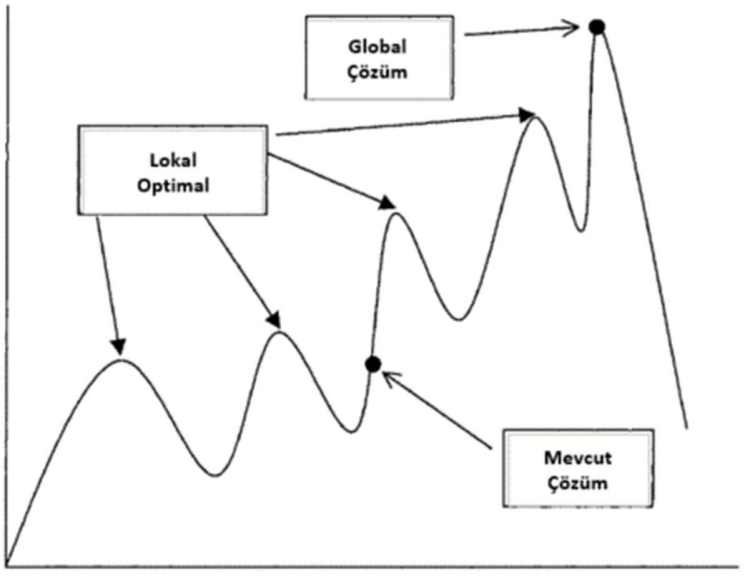
Yapıcı sezgiseleler (*constructive heuristics*): Sıfırdan bir başlangıç çözümünü inşa etmek için kullanılan sezgisellere denir (Burke ve Kendall, 2005). Bu çalışmada ele aldığımız üniversite ders çizelgeleme problemi buna iyi bir örnektir. Çünkü başlangıçta boş bir çizelge ile çözüme başlanmakta, sonra bütün dersler teker teker bu çizelgeye atanarak tamamlanmış bir başlangıç çizelgesi oluşturulmaktadır.

Yerel arama sezgiselleri (*local search heuristics*): Mevcut çözümün komşu çözümlerini potansiyel değişimler olarak değerlendirebildiğimiz bir sezgisel mekanizma olarak düşünülebilir. Bu komşu çözümlerden biri yeni bir çözüm olarak kabul edilirse, bu kez bu yeni çözümün komşu çözümleri değerlendirmeye alınır (Burke ve Kendall, 2005). Bu tanımdan da anlaşıldığı gibi yapıcı ve yerel arama sezgiselleri arasındaki fark; yapıcı sezgisellerde sıfırdan bir başlangıç çözümü inşa edilirken, yerel arama sezgisellerinde var olan bir çözümden diğerine hareket edilir. Yapıcı sezgiseller çoğunlukla yerel arama sezgiselleri için başlangıç noktası olarak kullanılır. Bu sezgiseller için prosedürü en basit olan tepe tırmanma (*hill climbing*) sezgiseli örnek olarak gösterilebilir. Ayrıca, tabu arama (*tabu search*) ve benzetim tavlaması (*simulated annealing*) gibi sezgiseller yerel arama sezgisellerine örnek olarak gösterilebilir.

Meta-sezgiseller: Yerel optimalin ötesinde çözümler üretmek için alt sezgiselleri değiştirerek ve yönlendirerek oluşturulan ana stratejiye denir. Meta-sezgiseller tarafında yönlendirilen sezgiseller, yüksek seviyede prosedürler veya bir çözümü bir diğer çözüme dönüştürmek için mevcut adımların tanımı ifade etmektedir (Burke ve Kendall, 2005). Amaç arama uzayını mümkün olduğunca hızlı arayarak en iyiye en yakın çözümleri bulmaktır. Meta-sezgisel geliştirmek, problemlerine arama algoritmaları kullanarak çözüm bulamak isteyenler için oldukça önemlidir. Literatüre bakıldığında, tabu arama (*tabu search*), benzetim tavlaması (*simulated annealing*), açgözlü arama (*greed deluge*) ve genetik programlama (*genetic programing*) gibi sezgisellerin dahil olduğu göreli olarak karmaşık meta sezgisel prosedürlerinin kullanıldığı görülebilir.

2.2.1 Tepe Tırmanma (*Hill Climbing*) Sezgiseli

Tepe tırmanma (*hill climbing*-HC) sezgiseli muhtemelen en basit yerel arama sezgisellerinden biridir. Anlaşılması ve uygulanması kolay olan bu sezgiselin dezavantajı kolayca yerel optimale (bak: Şekil 2) takılabilmesidir (Burke ve Kendall, 2005). HC sezgiseli temel olarak komşu çözümlerden biri, mevcut çözümden daha iyi ise bu iyi çözümü kabul ederek ilerler. Amaç en yüksek çözüm kalitesine ulaşmaya çalışmaktır. Ancak, çözümü iyileştirerek ilerlerken, çoğunlukla yerel optimale ulaşıldığında HC sezgiselinin yapısı itibariyle çözüm daha fazla iyileştirilemez ve çözüm yerel optimalde kalır. Bu nedenle yerel optimali aşabilmek için tabu arama (*tabu search*-TS), benzetim tavlaması (*simulated annealing*-SA) ve büyük tufan (*great deluge-GD*) gibi pek çok sezgisel geliştirilmiştir. Ancak, bazı çalışmalarda HC sezgiseli algoritmaların bir parçası olarak da kullanıldığı görülmektedir. Örneğin: Müller (2009) çalışmasında ders çizelgeleme problemi için başlangıç çözümünü elde ettikten sonra ilk olarak HC sezgiselini yerel optimuma ulaşana kadar çalıştırılmakta, sonra sırasıyla büyük tufan (*great deluge*-GD) ve benzetim tavlaması (*simulated annealing*-SA) sezgiselleri çalıştırılmaktadır. Son aşamada, belirlenen süre içerisinde çözüm iyileştirilemediğinde süreç tekrar baştan başlatılmaktadır.



Şekil 2. Sezgisel Çözüm (Burke ve Kendall, 2005).

2.2.2 Tabu Arama (*Tabu Search*-TS)

20 yılı aşkın süredir yöneylem araştırmaları literatüründe olan Tabu arama (*tabu search*-TS) ilk olarak Glover (1986) tarafından önerilmiştir. Bazı durumlarda, bu sezgisel optimale çok yakın ve etkin çözümler verebilmektedir. İyi sonuçlar vermediği durumlarda bile zor problemlerin üstesinden gelebilmektedir. Bu başarısından dolayı kısa sürede oldukça popüler bir yaklaşım olmuş, pratikte çokça kombinatoryal (*combinatorial*) problemleri çözmek için kullanılmış ve iyi sonuçlar vermiştir (Gendreau ve Potvin, 2010).

TS sezgiseli HC sezgiseli gibi rastgele bir çözüm ile başlayıp daha sonra komşu çözümlere bakarak çözümün kalitesini iyileştirmeye çalışmaktadır (MirHassani ve Habibi, 2013). Ancak Glover (1977) tarafından HC den farklı olarak bazı ek bileşenler önerilmektedir. Son hareketleri önlemek için kısa süreli bir hafıza kaydı yapmak ve faydalı hareketleri özendirmek için uzun süreli frekans kaydı yapmak bu bileşenlerden iki tanesidir. Tabu listesi oluşturmak olarak da adlandırılan bu kısa süreli hafızaya atma işlemi ziyaret edilen çözümleri bir liste şeklinde hafızada tutmaktadır. Bu sayede, yerel optimale ulaşıldığında bir döngüye düşmemek için hafızaya alınan son çözümleri tekrar ziyaret etmek engellenebilmektedir (Gendreau ve Potvin, 2010).

TS'de birden fazla tabu listesinin oluşturulması ve aynı anda kullanılması önerilmektedir. Çünkü bir problem için farklı tipte komşuluk hareketleri kullanıldığı zaman, her tip için ayrı bir tabu listesi tutmak iyi bir fikir olabilir (Gendreau ve Potvin, 2010). Örneğin ders çizelgeleme probleminde genelde kullanılan iki komşuluk hareketi düşünüldüğünde: ilk komşuluk hareketi iki zaman periyodunun zorunlu kısıtları sağlama şartıyla değiştirilmesidir; ikinci komşuluk hareketi aynı zaman periyodundaki iki dersin atandığı derslikler zorunlu kısıtları sağlama şartıyla değiştirilmesidir. Uygulamamızda bu iki komşuluk hareketi için farklı tabu listesi tutulmaktadır. Standart tabu listeleri genellikle sabit uzunlukta dairesel listeler olarak uygulanır. Ancak sabit uzunluktaki tabu listeleri bir döngüye girmeyi daima engelleyememektedirler (Gendreau ve Potvin, 2010). Vasquez ve Hao (2001) dinamik olarak değişen bir tabu liste uzunluğunun daha iyi sonuçlar vereceğini göstermektedir. Benzer şekilde, değişen tabu liste uzunluğunu öneren farklı yazarlar da olmuştur (Glover, 1989,

1990; Lü ve Hao, 2010; Taillard, 1990). Tabu listeleri ile sağlanan tabu kısıtlamaları, istisna şartı (*aspiration criteria*) adı verilen belirli koşullar altında geçersiz kılınabilir (Müller, 2005). Çünkü bazı durumlarda TS cazip bazı komşuluk hareketlerinin gerçekleşmesini bir döngüye girme tehlikesi olmadığı halde engelleyebilir. Böyle durumlar için gerektiği zaman tabu kısıtlamasını devre dışı bırakacak algoritmalar kullanmak gerekmektedir. En yaygın olarak kullanılan istisna şartı: mevcut çözümler içindeki en iyi çözümlerden biri elde edilecekse yapılan komşuluk hareketi tabu listesinde olsa bile yapılmalıdır. Bir diğer kriter ise sonlandırma kriteridir (*termination criteria*). TS'de en sık kullanılan sonlandırma kriterleri şunlardır (Gendreau ve Potvin, 2010):

- Sabit sayıdaki iterasyon sayısından veya sabit bilgisayar zamanından sonra
- Amaç fonksiyonu değerinde iyileşme olmaksızın süren sabit sayıdaki iterasyon sayısından sonra
- Amaç fonksiyonu değeri önceden belirlenmiş bir eşik değerine ulaştığında

Clark vd. (2008) çalışmalarında *QuickFix* adında bir çözücü önermişlerdir. Bu çözücü, kısıtların ihlal edilmesine izin vererek her bir dersin her bir bölümünün öğretim elemanının bir zaman periyoduna atayarak bir başlangıç çözümü oluşturmaktadır. Daha sonra, bu çözücü, ihlal edilmiş kısıtlar arasından bir kıstı ağırlık katsayısını (zorunlu kısıtlar çok yüksek, esnek kısıtlar ise amaç fonksiyonunda esnek kısıtlar için belirlenen katsayılardır) dikkate alarak seçmektedir. Seçilen kısıtlar öğretim elemanı zaman periyodu değişimi veya derslik zaman periyodu değişimi komşuluk hareketleri uygulanarak sağlanmaya çalışılmaktadır. İki tabu listesinin kullanıldığı bu algoritmada kısıt ağırlıkları çözüm boyunca güncellenmektedir. Algoritma uygulanabilir bir çözüme ulaşamadığı zamanlarda ise şimdiye kadarki en iyi çözümden algoritma tekrar başlamaktadır. Bu algoritmanın testi ITC-2007 problemleri üzerinde yapılmıştır.

Aladag vd. (2009) basit ve takas komşuluk hareketlerinin her birinden ve bileşiminden oluşan dört komşuluk hareketi türünden bahsetmektedir. Örneğin derslerin boş zaman periyotlarına atanması basit komşuluk hareketi iken, farklı zaman periyodundaki derslerin yerlerinin değiştirilmesi takas komşuluk hareketi olarak adlandırılmaktadır. Bu çalışmada dört

komşuluk hareketi tanımlanarak TS algoritması üzerinde çeşitlendirme etkisi oluşturmak amaçlanmıştır. TS algoritması kullanılarak Hacettepe Üniversitesi İstatistik Bölümü ders çizelgeleme problemi üzerinde test edilen bu komşuluk hareketleri, ayrıca aralarında çoklu karşılaştırmalara yapılarak istatistiki olarak analiz edilmiştir.

Lü ve Hao (2010) müfredat temelli üniversite ders çizelgeleme problemini çözmek için bir uyarlanabilir TS algoritması (adaptive TS algorithm (ATS)) sunmaktadırlar. Bu algoritma üç farklı aşamadan oluşmaktadır: başlatma (*initialization*), yoğunlaştırma (*intensification*) ve çeşitlendirme (*diversification*). Başlatma aşamasında uygulanabilir bir başlangıç çözümü elde edebilmek için bir sıralı açgözlü (*sequential greedy*) algoritması kullanılmıştır. Bu algoritma boş bir çizelge ile başlayıp, çeşitli atanma zorluklarına göre öncelik sırasına konan dersleri, ardışık olarak zaman periyoduna ve dersliklere atamaktadır. Uygulanabilir bir başlangıç çözümü elde edildikten sonra yoğunlaştırma ve çeşitlendirme aşaması kombine edilerek ihlal edilen esnek kısıtlar azaltılmaya çalışılmaktadır. TS algoritmasında iki farklı yapıda komşuluk tipi kullanılmıştır: ilki zaman periyotları ve dersliklerin iki farklı derse atandığı komşuluk hareketidir; ikincisi iki farklı zaman periyodu üzerinden oluşturulan kempe zincirlerinin (*Kempe chains*-detaylı bilgi için bakınız Lü ve Hao (2010)) zaman periyotlarının değiştirilmesi hareketidir. Çalışmada bu komşuluk hareketlerinin kullanım sırası algoritma yerel optimale her ulaşıldığında değişecek şekildedir. Bu süreç artık çözüm iyileştirilemeyecek aşamaya gelene kadar devam etmektedir. Son aşamada yerel optimalden kaçabilmek için çeşitlendirme aşaması devreye girmektedir. Bu aşamada, yerel optimal çözümünü bozmak için algoritmada ceza katsayılı düzensizlik (*penalty-guided perturbation*) algoritması kullanılmaktadır. Bu algoritma, çizelgeye atandığı pozisyon itibariyle yüksek ceza katsayısına sahip derslerin belirli bir listesini oluşturulduktan sonra bu liste içerisinden belli bir olasılığa göre seçilen derslere, rastgele seçilen komşuluk hareketlerini uygulamaktadır. Bu algoritma ITC-2007[2] örnekleri üzerinde test edilmiş başarılı sonuçlar alınmıştır.

[2] Detaylı bilgiye şu internet sayfasından ulaşılabilir: http://www.cs.qub.ac.uk/itc2007/index.htm

Goh vd. (2017) bu çalışmada, kayıt temelli ders çizelgeleme problemi iki aşamada sezgisel olarak çözülmektedir. İlk aşamada, tabu arama (TS) ve düzensizlik (*perturbation*) algoritması kullanılarak uygulanabilir bir başlangıç çözümü elde edilmeye çalışılmaktadır. Başlangıç çözümünde TS algoritmasının kullanımı farklı şekilde yapılmaktadır. Örneğin çizelgede uygun pozisyona yerleştirilen her bir olayın tabu listesi kendisiyle çakışabilecek dersleri içermektedir. Uygun bir çözüme ulaştıktan sonra çözüm kalitesi ikinci aşamada kullanılan benzetim tavlaması (SA) yaklaşımının geliştirilmiş bir çeşidi ile artırılmaya çalışılmaktadır. Algoritma üç önemli veri seti ile test edilmiş ve algoritmanın iyi ve rekabet edilebilir olduğu görülmüştür.

2.2.3 Benzetim Tavlaması (*Simulated Annealing*-SA)

Benzetim tavlaması (SA) algoritması, katı maddelerin yapısında, ısınma ve soğutulma süreçlerinde oluşan değişimden ilham almaktadır. Tavlama kelimesi de demirin ısıtılarak tavlanmasından gelmektedir. Örneğin demir yavaş yavaş soğutuldukça daha kararlı hale gelerek kristalize olmaktadır. SA algoritması yerel optimalden kaçabilmek için yerel arama sürecini bu fiziksel olaya benzer şekilde yapmaktadır. Kombinatoryal optimizasyon problemlerin çözümünde kullanılan bu algoritma ilk olarak Kirkpatrick vd. (1983) tarafından önerilmiştir.

SA sezgiseli tıpkı TS sezgiseli gibi HC sezgiselinin gelişmiş bir formudur. Bilindiği gibi HC algoritmasında komşu çözümler rastgele aranıyor ve sadece mevcut çözümden iyi olan çözümler yeni çözüm olarak kabul ediliyor. SA algoritmasında ise yerel optimumdan kaçabilmek için belirli bir olasılıkla mevcut çözümden kötü olan çözümlerde kabul edilebilmektedir. Kötü çözümleri kabul etme olasılığı genellikle $P = e^{-\delta/T}$ denklemiyle ifade edilir. Bu durumda, f maliyet fonksiyonunu, S^* yeni çözümü ve S mevcut çözümü gösterir iken $\delta = f(S^*) - f(S)$ olur; T de sıcaklık derecesini gösterir. Ayrıca, arama sürecinin genellikle yüksek sıcaklık derecesiyle başlaması ve arama sürecinin sonuna doğru kademeli olarak azaltılması önerilmektedir. Sıcaklık derecesini azaltmanın bir yolu olarak $T_{i+1} = T_i * \beta$ (geometrik soğutma) denklemi kullanılmaktadır. Bir problem için β katsayısı

ampirik olarak belirlenmektedir (Obit, 2010). Bu denklemlerden de anladığımız kadarıyla yeni ve mevcut çözümler arasındaki fark ne kadar büyük ve sıcaklık derecesi ne kadar yüksek olursa kötü bir çözümü kabul etme olasılığı o kadar az olacaktır. Dolayısıyla, algoritma hızlı bir şekilde çözümü iyileştirecek ve yerel optimuma yaklaştıkça kötü çözümleri kabul etmeye ve yerel optimumdan çıkmaya çalışacaktır.

SA algoritmasının soğutma planı farklı şekillerde ele alınabilmektedir. Soğutma planın seçimi temelde nihai çözümün kalitesini etkiler. Soğutma planı ne kadar hızlı olursa, yerel optimale o kadar hızlı şekilde ulaşılabilir. Öte yandan, soğutma planı ne kadar yavaş olursa, arama süreci o kadar kapsamlı olur ve bu genellikle daha yüksek kalitede bir çözüm ile sonuçlanır, ancak bunu yapmak için oldukça fazla çözüm süresi gerekir. Arama süreci sırasındaki soğutma planının seçimi, bir ilerleme formülü (geometrik soğutma planı) kullanılması, belirli sayıdaki harekete göre sıcaklığın azaltılması veya sadece başarılı hareketlere göre sıcaklığın azaltılması gibi çeşitli yollarla yapılabilir (Abdullah, 2006). Elmohamed vd. (1997) üç farklı soğutma yaklaşımından bahsetmektedir: geometrik, uyarlanabilir, yeniden ısıtmalı uyarlanabilir. Bu yaklaşımlar gerçek bir üniversite çizelgeleme verisi üzerinde test edilmiştir. Test sonucunda tekrardan ısıtmalı uyarlanabilir soğutma yaklaşımının daha iyi sonuçlar verdiği görülmüştür. Bu tez çalışmasında da kullandığımız SA sezgiseli için tekrardan ısıtmalı bir soğutma planı kullanılmaktadır.

Tuga vd. (2007) üniversite ders çizelgeleme problemini çözebilmek için kempe zincirleri (*kempe chains*) komşuluk yapısının bir kombinasyonunu SA algoritmasında kullanmıştır. Bu çalışmada, zorunlu kısıtlar gevşetmeli olarak yeniden düzenlendikten sonra gevşetmeli esnek kısıtlar gibi kullanılmaktadır. Bu gevşetilmiş problem iki adımda çözülmektedir. İlk adım, uygulanabilir bir başlangıç çözümü için graf teorisine dayalı bir sezgisel kullanılmaktadır. İkinci adımda ise esnek kısıt ihlallerini minimize etmek için SA algoritması kullanılmaktır. Bu algoritma literatürde en iyi bilinen üniversite ders çizelgeleme örnekleri üzerinde test edilmiştir. Aycan ve Ayav (2009) ise SA algoritmasını gerçek bir üniversite ders çizelgeleme problemini çözmek için kullanmışlardır. Bu çalışmada ayrıca, basit, takas ve bunların kombinasyonu olan komşuluk hareketlerinin performansları çözüm süreleri ve çözüm

kaliteleri açısından ölçülmüş en iyi performansı bahsedilen üç komşuluk hareketinin bileşiminin verdiği görülmüştür.

Cambazard vd. (2012) esnek kısıt ihlallerini minimize etmek için SA algoritmasını kullanmaktadır. Sadece uyulabilir hareketlere izin veren tek tip bir komşuluk hareketiyle derslik çakışmaları minimize edilmektedir. Başlangıç sıcaklık derecesi dinamik olarak seçilmektedir ve arama süreci ilerledikçe azalmaktadır. Her bir adımda standart geometrik soğutma süreci ($T_{i+1} = T_i * 0{,}95$) kullanılmıştır. Ayrıca, bu algoritma ITC-2007'nin kayıt temelli üniversite ders çizelgeleme problemleri üzerinden test edilmiştir.

Tarawneh vd. (2013) yerel optimalden çıkabilmek için kullanılmak üzere kabul edilmeyen çözümlerin saklandığı bir hibrit benzetim tavlaması algoritması önermektedirler. Başlangıç çözümü, Lü ve Hao (2010) tarafından kullanılan açgözlü algoritma (*greedy algorithm*) ile elde edilmektedir. Başlangıç çözümü için kullanılan açgözlü algoritma ile dersler öncelik sıralamasına konup, daha sonra derslik zaman periyodu ikilisinden oluşan pozisyonlara öncelik sırasına göre atanmaktadır. Bu algoritma, müfredata temelli üniversite ders çizelgeleme probleminde bir ders için atanmaya uygun pozisyon sayısı ne kadar az ve dersin atanmayan kısmının sayısı ne kadar fazla ise dersin atanma önceliğini o kadar yüksek belirlemektedir. Tarawneh vd. (2013) çalışmasında üç komşuluk yapısı kullanılmaktadır: (1) ders herhangi bir çakışmaya neden olmadan boş zaman periyodun atanması (basit komşuluk hareketi de denir), (2) atanmış iki dersin yer değiştirilmesi (takas komşuluk hareketi), ve (3) yüksek ceza skorlu zaman periyotlarını rastgele seçilerek yer değiştirilmesi hareketi. Bu algoritma ITC-2007 örnekleri üzerinde test edilip tatmin edici sonuçlar alınmıştır.

Bellio vd. (2016) çalışmasında müfredat temelli üniversite ders çizelgeleme problemini ele almaktadır. Bu çalışma iki yönlü katkı sunmaktadır. İlki, bu problemi çözmek için etkin bir tek aşamalı benzetim tavlaması algoritması önerilmesidir. İkincisi, parametre ayarlama yöntemi için geniş kapsamlı ve istatistiksel bir metodolojinin tasarlanıp uygulanmasıdır. Bu analizin çıktısı, örnek özelikleri ve arama yönteminin parametreleri arasında doğrusal bir regresyon modeli belirlenip derin bir istatistiki analiz geliştirilebilmesidir. Ayrıca, bu istatistiki analiz, hakkında az bilgi sahibi olunan yeni örnekler için parametrelerin

ayarlanmasına da izin verebilir. Algoritma da iki komşuluk hareketi yapısı (basit ve takas) kullanılmaktadır. Bunun yanında ikinci komşuluk hareketinin birincisine göre ne sıklıkta seçildiğini kontrol etmek için bir takas oranı parametresi de kullanılmaktadır. Bu algoritma, Lopes ve Smith-Miles (2010) tarafından üretilen büyük ölçekli örnekler üzerinde uygulanarak geliştirilmiş ve ITC-2007 örnekleri üzerinde test edilerek doğrulanmıştır.

Üniversite çizelgeleme problemlerinde kullanılan çözüm yaklaşımlarından bahsettikten sonra özellikle sezgisel yaklaşımların bu problem özelinde nasıl bir yapıda olacağına karar vermek önemli bir konu olacaktır. Çünkü bu problem zorunlu ve esnek kısıtlardan oluşmaktadır ve bu kısıtların içeriği problemlerin yapısına göre farklılık göstermektedir. Problemin çözümünde kullanılacak çözüm yaklaşımının bu kısıtları ele alış biçimi çözümün kalitesini doğrudan etkilemektedir. Bir sonraki başlık, üniversite çizelgeleme problemlerinde kullanılan sezgisel algoritmaların yapısı hakkında net bir fikir vermeyi amaçlamaktadır.

2.2.4 Ders Çizelgeleme Probleminde Sezgisel Kullanımı

Çizelgeleme problemi düşünüldüğünde, zorunlu ve esnek kısıtların varlığı bu problemi diğer kombinasyonel optimizasyon problemlerinden ayırmaktadır. Bu nedenle, sezgisel yaklaşımlardan sonuç elde edebilmek için bu iki ayırımı yapabilecek algoritmaların kullanılması gerekecekir (Lewis, 2008). Lewis (2008)'e göre çizelgeleme literatürü bu bağlamda incelendiğinde çoğu sezgisel algoritmanın aşağıda verilen üç kategorinden birine girdiği görülecektir. Lewis (2008):

1) **Tek aşamalı optimizasyon algoritmaları** hem zorunlu kısıtlar hem de esnek kısıtlar aynı anda sağlanmaya çalışılır.

2) **İki aşamalı optimizasyon algoritmaları** ilk aşamada zorunlu kısıtlar sağlanarak uygulanabilir (*feasible*) bir çizelge elde edilmeye çalışılır. Sonraki aşamada, çözümün uygulanabilirliğini bozmadan ihlal edilen esnek kısıtlar minimize edilerek çözümün kalitesi yükseltilmeye çalışılır.

3) **Gevşetme teknikleri kullanan algoritmalar** zorunlu kısıtların ihlalini önlemek için gevşetme teknikleri (ör: yapay bir zaman periyodunun eklenmesi) kullanırken aynı zamanda esnek kısıt ihlallerini minimize etmeye ve gevşetmeleri geri almaya çalışır.

2.2.4.1 Tek Aşamalı Optimizasyon Algoritmaları

Bu stratejide, sezgiseller esnek ve zorunlu kısıtları aynı anda sağlamak için tek ceza fonksiyonu kullanmaktadırlar. Bunu başarabilmek için, kısıtların önem derecelerini yansıtan ağırlıkları algoritmaya farklı verilmektedir. Sonuç olarak, zorunlu kısıtın ihlal edilmesinin ceza maliyeti esnek kısıtın ihlal edilmesinin ceza maliyetinden daha yüksek olacak ve ceza skoruna etkisi daha yüksek olacaktır (Lewis, 2008). Esnek kısıtların önemi zorunlu kısıtların öneminin yanında çok önemsiz kalmaktadır. Buna örnek olarak, her bir zorunlu kısıt için 1000 puanlık bir ağırlık atanırken, esnek kısıt olan kapasite kısıtına sadece 2 puanlık bir ağırlığın atanmasını verebiliriz. Bu algoritmalar ceza maliyetini minimize etmek için öncelikle zorunlu kısıtların hepsinin sağlandığı uygulanabilir bir çözüme ulaşmaya çalışmaktadır.

Tek aşamalı optimizasyon yaklaşımın en önemli avantajı göreli olarak daha kolay uygulanabilir ve kolay değiştirilebilir olmasıdır. Çünkü kısıtların içerikleri veya kısıtların ağırlıkları kolayca değiştirilip etkileri net olarak görülebilir (Lewis, 2008). Bu yaklaşım kısıt değişimlerinin ve kısıt eklemelerinin çok olduğu çizelgeleme problemleri için uygun bir yaklaşım olacaktır.

Bu yaklaşım doğası gereği bazı dezavantajlara da sahiptir. İlki, çözüm uzayında az sayıda çözümü olan problemlerde kısıtların ağırlık uyumu iyi işlemeyebilir. Yani, uygulanabilir çözüm sayısı az olduğu için kısıtların ağırlıklarındaki değişimler çözümde bir değişime neden olmayabilir. İkincisi, ağırlıklarda yapılan değişimler algoritmanın çözüm zamanında ve kalitesinde önemli değişimlere neden olmasına rağmen ağırlık atamada standart bir durum yoktur. Bu nedenle farklı problem versiyonlarında, hatta bazen de aynı problemin farklı boyutlarında tatmin edici olmayan sonuçlar alınabilir. Üçüncüsü, maliyet fonksiyonunda yapılacak küçük bir değişim ceza skorunda büyük değişimlere (ör: ceza skoru 1000 olan bir

zorunlu kısıtın ihlal edilmesi) neden olabilir. Bu durumda, ağırlıklar kullanarak esnek kısıtları etkilemek veya başka bir ifadeyle çözüm uzayında istikrarlı bir şekilde gezinmek oldukça zor olacaktır. Bu nedenlerden dolayı, daha kompleks problemleri çözmek için iki aşamalı yaklaşımın kullanılması daha yaygındır.

2.2.4.2 İki Aşamalı Optimizasyon Algoritmaları

İki aşamalı yaklaşım çizelgeleme problemlerinin yapısına daha uygun düşmektedir. Zorunlu ve esnek kısıtlara sahip bu problemlerde ilk aşamada uygulanabilir (*feasible*) bir çözüm elde edilmektedir. Bu aşamada, esnek kısıtların tümü ihmal edilerek sadece zorunlu kısıtlar dikkate alınmaktadır. Bir uygulanabilir çözüme veya başka bir ifadeyle bir başlangıç çözümüne ulaşıldıktan sonra esnek kısıtların dikkate alındığı ikinci aşamaya geçilmektedir. İkinci aşamada, hiçbir zorunlu kısıtın ihlal edilmesine izin verilmeden esnek kısıtlar sağlanmaya çalışılmaktadır.

Bu tekniğin ilk yararı, zorunlu ve esnek kısıtları ayırt etmek için artık ağırlık tanımlamanın gerekmediğidir. Tek aşamalı algoritma yaklaşımında, esnek ve zorunlu kısıtlar tek bir amaç fonksiyonunda ele alındığından çözüm uygulanabilir ve uygulanmaz çizelgeler arasında gidip gelebilmektedir. Bazı esnek kısıtları sağlama pahasına problem çözümsüz olabilmekte ve çözüm süreci uzayabilmektedir. İki aşamalı yaklaşımda ise ilk önce uygulanabilir bir çözüme ulaşıp daha sonra bu çözüm garanti altına alınarak çözümün kalitesi artırılmaya çalışılmaktadır.

İlk aşamada elde edilen çözüm bir başka ifadeyle başlangıç çözümü, ikinci aşamaya (çözümün kalitesinin artırılmasına) doğrudan etki edebilmektedir. Çizelgeleme problemlerinde uygulanabilir bir başlangıç çözümüne ulaşabilmek için hangi olayları (ör: dersler veya ders periyotları) hangi pozisyonlara (ör: pozisyon = (derslik, zaman periyodu)) ve hangi sırayla atayacağımız oldukça önemli olmaktadır. Örneğin gelişigüzel bir atama yaptığımızda bazı dersler için uygun sınıfların ve uygun pozisyonların kalmadığı ve bu nedenle bu derslerin atanamadığı durumlar ile sıkça karşılaşabiliriz. Bu aşamada, atanamayan olayları atayabilmek ve bir başlangıç çözümü için bazı yerel arama sezgiselleri

de (TS ve SA gibi) kullanılabilmektedir. Lü ve Hao (2010) çalışmalarında, açgözlü renklendirme (*greedy coloring*) (Brélaz, 1979) sezgiseline benzer bir algoritma kullanarak ders çizelgelemesinde başlangıç çözümüne ulaşmaya çalışmışlardır. Bu algoritma atanması en zor olan dersleri en önce atamaya çalışmaktadır. Yani bir dersin atanması için uygun pozisyon sayısı ne kadar az ve çakışma ihtimali ne kadar fazla ise bu dersin atanma önceliği o kadar yüksek olacaktır. Bu yöntem tamamlanmış bir başlangıç çözümünü her zaman garanti etmez (Lü ve Hao, 2010). Ancak, iyi bir başlangıç çözümü için süreci oldukça hızlandırır. Müller (2009) çalışmasında, çakışma temelli istatistikler (conflict-based statistics-CBS) (Müller vd., 2004) yöntemiyle başlangıç çözümlerine ulaşmaya çalışmıştır. Bu yöntem, daha önce çakışmış her olay ikilisinin çakışma sayılarını hafızada tutarak çakışma sayısı az olan olay-pozisyon seçeneğini seçerek ilerlemektedir. CSB yöntemi bir iteratif ileri arama (*iterative forward search*-IFS) algoritması ile kullanılarak tüm derslerin atanması sağlanmaktadır.

İki aşamalı yaklaşımda genelde, ilk aşamada zorunlu kısıtlar sağlanarak bir başlangıç çözümü elde edildikten sonra ikinci aşamada zorunlu kısıtlar ihlal edilmemek şartıyla esnek kısıtlar sağlanmaya çalışılmaktadır. Arntzen ve Løkketangen (2005) ilk aşamada, bir yapıcı yaklaşım kullanarak başlangıç çözümüne ulaşmışlardır. İkinci aşamada ise TS yaklaşımı kullanılarak, zorunlu kısıtları ihlal etmeme şartıyla, esnek kısıtlar sağlanmaya çalışılmıştır. Lü ve Hao (2010) yukarıda bahsedildiği gibi ilk aşamada göreli önceliklere olayları sıralayarak uygun pozisyonlara atamaya çalışıp, ikinci aşamada uyarlanabilir tabu arama (*adaptive tabu search*) yaklaşımı kullanmışlardır. Aycan ve Ayav (2009) iyi bir başlangıç çözümü için kısıt programlama (CSP) yöntemi kullanmışlardır. Bu yöntem ile tüm zorunlu kısıtlar ve bazı esnek kısıtlar sağlanıp ikinci aşamada kalan esnek kısıtlar benzetim tavlaması sezgiseli ile sağlanmaya çalışılmıştır. Goh vd. (2017) ilk aşamada TS sezgiseli ve tabu sezgiselinin iyileştirilmiş bir versiyonunu kullanarak bir başlangıç çözümüne ulaşmışlardır. İkinci aşamada ise benzetim tavlaması sezgiselinin geliştirilmiş bir versiyonu ile çözümün kalitesini artırmaya çalışmışlardır.

2.2.4.3 Gevşetme Teknikleri Kullanan Algoritmalar

Bu yaklaşım çizelgeleme problemlerinde zorunlu kısıtları ihlal etmeden esnek kısıtları sağlamayabilmek için bazı "gevşetme" (*relaxed*) metotları kullanmaktadır. Lewis (2008)'e göre çizelgeleme problemlerinde bu gevşetme teknikleri iki fraklı yolla yapılabilir: (1) Bazı olaylar için (ders) mevcut çizelgede atanacak uygun pozisyonlar bulunamadığında bu olaylar atanamayan olaylar listesine alınır. Algoritma daha sonra esnek kısıtları sağlamaya çalışırken açılabilecek yeni pozisyonlara atanamayan olaylar listesindeki olayları atamaya çalışır. (2) Atanabilecek zaman periyodu bulunmayan olaylar için çizelgeye yapay zaman periyotları eklenebilir. Algoritma, daha sonra, esnek kısıtları sağlamaya çalışırken aynı zamanda bu yapay zaman periyotlarına atanan olayları azaltmaya çalışacaktır. Problemin yapısına göre eğer gerek duyuluyorsa yapay dersliklerde eklenebilir. Gevşetme algoritmalarının amacı özetle esnek kısıtları sağlamaya çalışırken aynı zamanda eklenen gevşetmeleri kaldırmaya çalışmaktır.

2.3 MODERN AKILLI METOTLAR

Bu yaklaşımlara örnek olarak: çok kriterli/amaçlı yaklaşımalar, hibrid (*hybrid)* yaklaşımlar, yapay zeka yaklaşımları ve üst sezgisel (*hyper heuristic*) yaklaşımalar gibi bir çok yaklaşım örnek olarak gösterilebilir. Genel bir fikir vermesi açısında aşağıda bu yöntemlerden bazıları hakkında kısaca bilgi verilmektedir.

2.3.1 Çok Kriterli Yaklaşımlar

Bu yaklaşımlarda her bir kriter kendisine karşılık gelen kısıtın ihlalini ölçer. Her bir kriter farklı durumlarda ve farklı kurumlar için farklı önem seviyelerine sahip olmaktadır (Petrovic ve Burke, 2004). Yani farklı kısıtlar için yapılan basit bir maliyet toplamı daima şartları yansıtmayabilir. Son yıllarda çizelgeleme de yapılan çalışmalarda tek bir ağırlık toplamı yerine farklı kısıtlarla belirlenen bir kısıt vektörü sayesinde birbirinden farklı kısıtlar kolayca ele alınmaktadır (Qu vd., 2009). Bilindiği gibi eğitim çizelgeleme problemlerinde, ders

verenlerin, öğrencilerin ve fakültelerin bir biriyle çelişen isteklerinin olması bu problemler için çok kriterli veya bazı problemler için çok amaçlı modellerin kurulmasını gerekli kılabilmektedir. Bu problemin çok amaçlı olarak nasıl ele alınacağını ve bu konuda daha önce yapılmış çalışmalar hakkında ayrıntılı bilgi almak için Phillips vd. (2015) ve Silva vd. (2004) çalışmalarına bakılabilir.

2.3.2 Üst Sezgisel (*Hyper Heuristic*) Algoritmalar

Sezgisel seçmeye yarayan, bir başka deyişle sezgisel seçen sezgiseller olarak tanımlanmaktadır (Burke vd., 2003). Yani bir problemi için birden fazla yöntemin kullanılacağını düşünelim. Bu yöntemlerin kendilerine göre avantaj ve dezavantajlarını dikkate alarak problemin çözüm kalitesini seçim kararları alarak geliştirmeye yardımcı olacak sezgisellere üst sezgiseller denmektedir. Detaylı bilgi için okul çizelge probleminin çözümünde üç farklı üst sezgisel kullanan Pillay (2013) çalışması incelenebilir. Daha sonra bu üst sezgisellerin performanslarını karşılaştırıldığı çalışma olan Burke vd. (2003) ve Burke ve Kendall (2005) çalışmaları incelenebilir.

2.3.3 Hibrit metotlar (*Hybrid methods*)

Diğer tekniklerin veya algoritmaların en etkin ve kullanışlı özelliklerini bir araya getirerek oluşturulmaktadır. Bu şekilde diğer algoritmaların zayıf ve etkin olmayan özelliklerinden kurtularak çizelgeleme problemlerini daha etkin bir şekilde çözebilmektedir (Babaei vd., 2015). Örneğin: Le Huédé vd. (2006) çok kriterli optimizasyon yaklaşımını kısıt programlama ile entegre ederek sınav çizelgeleme problemini çözmüştür. Kriterlerin önemlerini belirlemek için bulanık ölçümler kullanılmış ve çözüm kalitesini yükseltmek için en sonuncu çözüme göre dinamik olarak hareket eden bir kriter seçim sezgiseli kullanılmıştır.

Müller (2009) çalışmasında üç aşamadan oluşan hibrit bir yaklaşım sunmaktadır. İlk aşamada, derslerin sırayla seçilip derslik ve zaman ikilisine atandığı bir iteratif ileri arama (*iterative forward search*) algoritması kullanılmıştır. Bu algoritma, Müller vd. (2004)

tarafından geliştirilen çakışma temelli istatistikler (*conflict-based statistics*-CBS) metodunu kullanarak tüm derslerin atandığı ve zorunlu kısıtların ihlal edilmediği uygulanabilir bir başlangıç çözümü vermektedir. Tüm derslerin atandığı bir başlangıç çözümü elde edildikten sonra, yerel optimale ulaşmak için HC algoritmasının kullanıldığı ikinci aşamaya geçilmektedir. HC algoritması çözümün artık iyileştirilemediği belirli bir sayıdaki tekrarın ardından durdurulur. Üçüncü aşamada sırasıyla büyük tufan (*great deluge*-GD) ve benzetim tavlaması (SA) algoritması kullanılmaktadır. Yerel optimalden kaçabilmek için ikinci ve üçüncü aşama bir döngü halinde, çözümü durdurma şartı sağlanana kadar devam etmektedir. Bu hibrit yaklaşım ITC-2007'deki üç farklı yarışma problemlerinin hepsine uygulanmıştır. Yani sadece müfredat temelli üniversite çizelgeleme problemlerine değil aynı zamanda kayıt temelli üniversite ders çizelgeleme ve sınav çizelgeleme problemlerinde de uygulanmıştır. Sonuç olarak bu yaklaşım üç yarış kategorisinin ikisinde kazanan olmuştur. Bu kategorilerden biri de müfredat temelli ders çizelgeleme problemelerdir.

Bellio vd. (2012) benzetim tavlaması (SA) dinamik tabu aramasıyla (DTS) birleştiren hibrit bir yerel arama algoritması sunmaktadır. Tabu listesinin uzunluğu ve maliyet fonksiyonu yapısı kısıtların ağırlıklarının değiştirilmesi ile dinamik bir şekilde değişmektedir. SA ve TS algoritmaları bir döngü şeklinde çalıştırılmaktadır. Bu hibrit algoritma ITC-2007[3] deki müfredat temelli üniversite ders çizelgeleme problemleri üzerinde test edilmiştir.

Bu bölümde üniversite ders çizelgeleme problemlerini çözmek için yaygın olarak kullanılan çözüm yaklaşımlarından ve bu konuda yapılan birçok çalışmadan bahsedildi. Ayrıca Tablo 1'de üniversite ders çizelgeleme problemini ele alan başlıca çalışmalar ve bu çalışmaların kullandıkları çözüm yaklaşımları özetlenmektedir.

[3] ITC-2007: 2007 yılında 3 kategoride (okul, üniversite sınav ve üniversite ders çizelgeleme) başlatılan uluslararası çizelgeleme yarışması (*international timetabling competition*). http://www.cs.qub.ac.uk/itc2007/

Tablo 1. Üniversite Ders Çizelgeleme Problemini Ele Alan Başlıca Çalışmalar

	LP/IP/MIP	GT	CSP	TS	SA	HS	DS	Kullanılan Veri
De Werra (1985)	✓							Sanal veri
De Werra (1996)		✓						Gerçek Veri
Elmohamed vd. (1997)					✓			Gerçek Veri
Deris vd. (2000)			✓					Gerçek Veri
Dimopoulou ve Miliotis (2001)	✓						✓	Gerçek Veri
Asratian ve de Werra (2002)		✓						Gerçek Veri
Daskalaki vd. (2004)	✓							Gerçek Veri
Daskalaki ve Birbas (2005)	✓							Gerçek Veri
Al-Yakoob ve Sherali (2006)	✓						✓	Gerçek Veri
MirHassani (2006)	✓							Gerçek Veri
Al-Yakoob ve Sherali (2007)	✓							Gerçek Veri
Schimmelpfeng ve Helber (2007)	✓							Gerçek Veri
Tuga vd. (2007)					✓			ITC-2007
Bakır ve Aksop (2008)	✓							Gerçek Veri
Clark vd. (2008)				✓				ITC-2007
Lach ve Lübbecke (2008)	✓							ITC-2007
Aladag vd. (2009)				✓				Gerçek Veri
Aycan ve Ayav (2009)			✓		✓			Gerçek Veri
Broek vd. (2009)	✓							Gerçek Veri
Müller (2009)						✓		ITC-2007
Lü ve Hao (2010)				✓				ITC-2007
Hao ve Benlic (2011)	✓			✓				ITC-2007
Bellio vd. (2012)						✓		ITC-2007
Cambazard vd. (2012)					✓			ITC-2007
Lach ve Lübbecke (2012)	✓							ITC-2007
Cacchiani vd. (2013)	✓						✓	ITC-2002/ITC-2007
Tarawneh vd. (2013)					✓			ITC-2007
Sørensen ve Dahms (2014)	✓							Gerçek Veri
Phillips vd. (2015)	✓							Gerçek Veri/ ITC-2007
Bellio vd. (2016)					✓			ITC-2007
Goh vd. (2017)				✓	✓			ITC-2007
Phillips vd. (2017)	✓						✓	Gerçek Veri

ITC-2007: İkinci uluslararası çizelgeleme yarışması 2007 eğitimde çizelgeleme problemleri

LP/IP/MIP: karmaşık, doğrusal ve tamsayılı programlama; GT: graf renklendirme; CSP: kısıt programlama; TS: tabu arama; SA: Benzetim tavlaması; HS: hibrit sezgiseller; DS: diğer sezgiseller

3. BÖLÜM
PROBLEMİN MODELLENMESİ

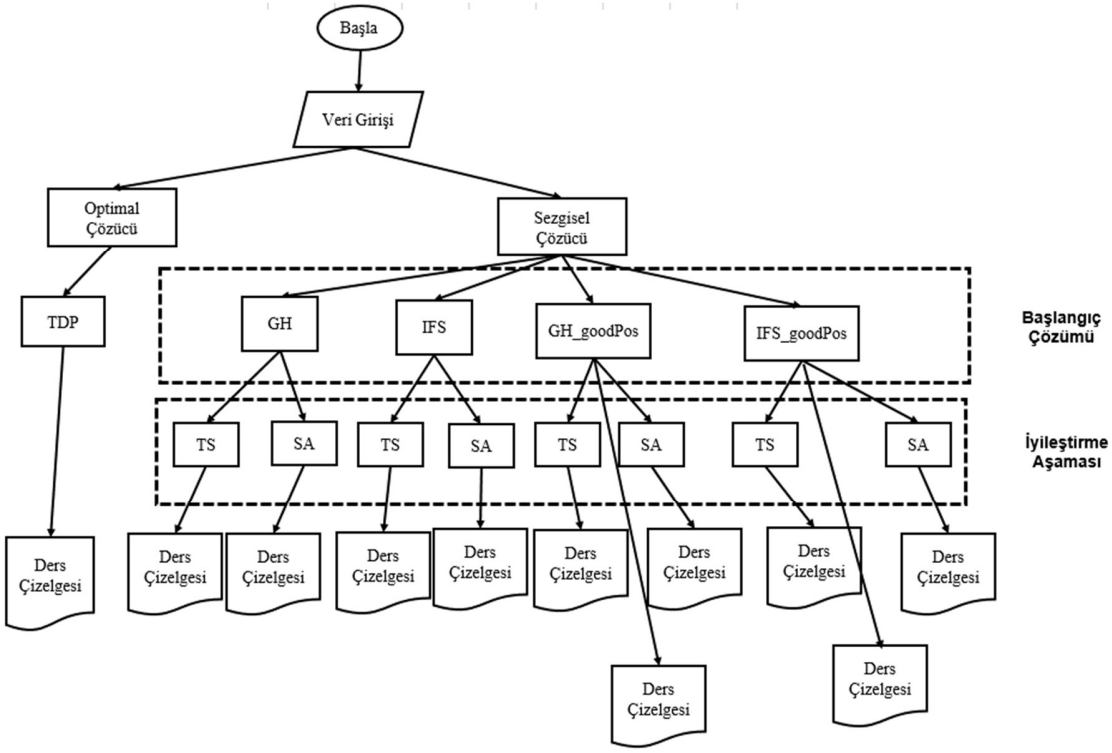
NP-Tam problem sınıfına giren üniversite ders çizelgeleme problemleri için literatürde çok sayıda optimizasyon yaklaşımı önerilmektedir. Uygulamada bu problemleri bu yaklaşımlar ile etkin bir şekilde modelleyip çözmek daha tatmin edici sonuçlar vereceği açıktır. Çoğu kurumda bu problem elle çözüme ulaştırılmaya çalışılmaktadır. Böyle durumlarda çözüm için çok fazla iş yükü gerektirmekte ve çoğu kez tatmin edici bir sonuç vermemektedir.

Birden fazla bölümü olan bir üniversite fakültesini düşünelim… Bu fakültenin ders çizelgeleme problemini elle çözmek için ilk önce ders çizelgesinin daha zor olduğunu düşündüğümüz bir bölümden başlayarak sırasıyla tüm bölümlerin ders çizelgelerini oluşturmaya çalışmak akla gelen ilk mantıklı yol olabilir. Ancak böyle büyük problemlerde, çizelgesi en son yapılacak bölümler için yeterli derslik kapasitesi kalmayabilir. Ayrıca bazı öğretim elemanları birden fazla bölüme ders verebildiği için oluşabilecek çakışmaları önlemek için çizelgesi önceden oluşturulan bölümlerin çözümlerini tekrar tekrar değiştirmek gerekebilir. En nihayetinde, uygulanabilir bir çözüme ulaşılsa bile, bu çözüm üzerinden sonradan birçok değişiklik yapılması gerekebilecektir. Bu tezde çözmeye çalıştığımız Hacettepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi (HÜİİBF) ders çizelgeleme problemi de hali hazırda elle çözülmektedir. Bu problemde fakültenin belirlediği bir koordinatör, öncelikle derslikleri bölümlere ders ve öğrenci sayılarını dikkate alarak kabaca paylaştırmakta ve bölümlerin ortak dersleri olan bazı derslerin atamasını da önceden yapmaktadır. Devamında, her bölümün belirlediği personeller (çoğu bölüm ikişer araştırma görevlisi görevlendirmektedir) bölümlerinin derslerini, zorunlu ve mümkünse esnek kısıtları da dikkate alarak yetersiz sayıda olan dersliklere atamaktadırlar. Daha sonra fakülte koordinatörü tüm bölümlerin ders çizelgeleme görevlilerinin olduğu bir toplantı düzenleyerek atanamayan dersleri boş kapasitelere atamaya çalışmaktadır. Bu toplantıda, uygun bir çözüme ulaşabilmek için çoğu dersin yeri ve zaman periyodu değiştirilmekte, uygun derslik bulunamadığı durumlarda ise fakülte dışındaki dersliklere atama yapılmaktadır. Bu çözüm sürecinden de anlaşıldığı gibi problemin çözümü oldukça

meşakkatli olmaktadır. Örneğin bu problemde 20'ye yakın personel ve çokça zaman harcanmaktadır. Buna rağmen tatmin edici bir çözüme çoğu zaman ulaşılamamaktadır. Tüm bu nedenlerden dolayı ders çizelgeleme problemini etkin bir şekilde modelleyip çözüme ulaştırmak önemlidir.

Bu tez çalışmasında üniversite ders çizelgeleme problemini çözmek için iki farklı yaklaşım önerilmektedir. İlk yaklaşım problemi optimal olarak çözen tamsayılı doğrusal programlama yaklaşımıdır. İkinci yaklaşım da optimal olmasa da yeterince iyi çözümler sunan sezgisel yaklaşımdır. Kullanım kolaylığı olması açısından her iki yaklaşım içinde tek bir veri giriş formatı tasarlanmıştır. Veri dosyasına belirlenen formatta girilen veriler, kodlanan ara fonksiyonlar vasıtasıyla kullanılacak yaklaşıma uygun şekilde dosyadan okunabilmektedir.

Şekil 3'teki diyagram ders çizelgeleme problemi için tasarladığımız çözücünün genel yapısını göstermektedir. Tasarlamak istediğimiz çözücüde, ders çizelgeleme verisi formata uygun bir şekilde girildikten sonra hangi yaklaşımla çözüleceği kullanıcının isteğine bağlı olacaktır. Aslında, kullanıcı problemi için ilk olarak optimal çözüm arayışına girecektir. Ancak, problemin kompleksliği belli bir seviyenin sütüne çıktığında tam sayılı doğrusal programlama (TDP) modelimizle kabul edilebilir süreler içerisinde optimal çözüm elde edilemeyebilir. Böyle durumlarda kullanıcı için sezgisel yaklaşımlar birer alternatif olacaktır.



Şekil 3. Ders Çizelgeleme Probleminin Çözücü Diyagramı

Veri giriş formatından bahsetmeden önce, çözmeyi amaçladığımız HÜİİBF ders çizelgeleme probleminin yapısından ve problemi oluşturan kısıtlardan bahsetmek faydalı olacaktır.

3.1 PROBLEMİNİN YAPISI

Üniversite ders çizelgeleme problemi; her kurumun yapısına, imkânlarına ve kendine öz ihtiyaçlarına göre farklılık göstermektedir. Bu nedenle, üniversite ders çizelgeleme probleminin çözümü için oluşturulan modellerin temel yapısı ne kadar benzer olsa da eklenen özel gereksinimler modelleri daha karmaşık ve farklı hale getirmektedir.

Üniversite Ders Çizelgeleme yapısı temelde gün, zaman periyodu, öğrenci grubu, öğretim elemanı, ders ve derslik boyutlarından oluşmaktadır. Bu boyutları çalışmamızda modellediğimiz Hacettepe HÜİİBF ders çizelgeleme problemi üzerinden şu şekilde tanımlayabiliriz:

Gün, hafta içindeki beş günü, zaman periyodu ise sabah (09:00-11:45) ve öğleden sonraki zaman (13:00 – 17:45) dilimlerini ifade etmektedir. HÜİİBF' de derslerin tümü üçer saatlik periyotlar şeklinde planlanmıştır. Bu derslerin tüm saatleri arka arkaya gelecek şekilde tek blok olarak işlenmektedir. Ayrıca her ders saati için 15 dakikalık aralar düşünülmüş ve bu araların nasıl dağıtılacağı öğretim elemanının inisiyatifine bırakılmıştır. Bu nedenle dersler tek bir zaman periyodu olarak düşünülebilir. Dolayısıyla bir günde sabah ve öğleden sonra olmak üzere sadece iki zaman periyodu vardır.

Öğrenci grubu ile kastedilen şey ortak dersler alan öğrenci topluluğudur. Bu nedenle, dört yıllık bir eğitim programı sunan HÜİİBF' de her bir bölüm için dönemler itibari ile ortak dersler alındığından bölümlerin her bir dönemi (müfredatı) bir öğrenci grubunu temsil etmektedir.

Müfredatta bulunan dersler öğretim elemanları tarafından verilmektedir. Öğretim elemanları, üniversite bünyesinde bulunan veya dışardan gelen öğretim üyeleri (Profesör, Doçent, Doktor Öğretim Üyesi), öğretim görevlileri ve alanında uzman kişilerden oluşmaktadır. HÜİİBF' de hangi dersi veya dersleri hangi öğretim elemanın vereceği, her bir bölümün akademik kurulu kararlarıyla önceden belirlenmektedir.

Müfredatta bulunan dersler zorunlu ve seçmeli olarak ikiye ayrılmaktadır. Zorunlu dersler, öğrencilerin programdan mezun olabilmek için mutlaka almak zorunda oldukları derslerdir. Seçmeli dersler ise öğrencilerin ilgi alanlarına göre alıp almama seçeneğine sahip olmakla birlikte mezuniyet için gerekli olan minimum krediyi tamamlamak için belli sayıda almak durumunda kaldıkları derslerdir. Aynı zamanda, bazı derslerde şubelere ayrılmaktadır. İçerikleri aynı olan bu dersler aynı veya farklı öğretim elemanları tarafından verilebilmektedir.

Sadece bazı bölümlere ayrılmış özel derslikler hariç fakültedeki çoğu derslik fakültenin bütün bölümlerinin ortak kullanımına açılmaktadır. Bazı derslikler ise özel teçhizat gerektiren dersler için özel olarak dizayn edilmiş dersliklerdir (örn: laboratuvarlar gibi).

HÜİİBF çizelge probleminin yapısını daha net görebilmek için problemin yapısını yansıtan kısıtları zorunlu ve esnek kısıtlar ayrımı altında açıklamak uygun olacaktır. Bu kısıtlar aynı zamanda modellerimizde kullanılmaktadır.

3.1.1 Zorunlu Kısıtlar

HÜİİBF ders çizelgeleme probleminde zorunlu olarak karşılanması gereken kısıtlar şöyle özetlenebilir:

Z1. Bir öğretim elemanı bir zaman periyodunda birden fazla yerde ders vermemeli ve sadece bir dersliğe atanmalıdır.

Z2. Bir öğretim elemanı zorunlu nedenlerden dolayı bulunamayacağı zaman periyotlarına atanmamalıdır.

Z3. Dersler kapasitelerini ve özel gereksinimlerini karşılayan dersliklere atanmalıdır.

Z4. Aynı zaman periyodunda bir dersliğe sadece bir ders atanmalıdır.

Z5. Müfredatlardaki tüm dersler çizelgeye atanmalıdır.

Z6. Öğrenci gruplarının müfredatlarında yer alan zorunlu dersler aynı zaman periyoduna atanmamalıdır.

Z7. Fakültenin bölümleri arasında ortak olarak alınan bazı zorunlu dersler (İngilizce dersi, Türk Dili ve Edebiyatı gibi) vardır. Bu dersler, genellikle fakülte veya bölüm dışından olan öğretim elemanları tarafından verilmektedir. Bu derslerin atanacakları zaman periyotları ve derslikler önceden belirlenmektedir. Bu nedenle, bu dersliklere bu zaman periyotları için başka dersler atanmamalıdır. Ayrıca, bir öğrenci grubunun müfredatına bu derslerden biri veya birkaçı var ise öğrenci grubunun müfredatındaki diğer dersleri bu derslerin önceden atandıkları derslik ve zaman periyotlarına atanmamalıdırlar.

Z8. HÜİİBF de beşeri ve fiziki mekân kaynağı yetersizliğinden dolayı öğrenci gruplarına verilen bazı dersler şubelere ayrılmıştır. Bir öğrenci grubunun müfredatındaki zorunlu statüsündeki şubeli dersler, şubesiz olan zorunlu derslerden farklı olarak aynı zaman periyoduna atanabilmelidir. Ancak bu esneklik öğrenciye bir şubeli dersin en

az bir şubesini seçebilme imkânı verecek şekilde olmalıdır. Örneğin aynı zaman periyoduna, bir öğrenci grubunun bir şubeli dersinin tüm şubeleri atandıysa, bu zaman periyoduna bu öğrenci grubunun müfredatındaki diğer derslerin hiç biri atanmamalıdır. Çünkü öğrenci bu zorunlu şubeli dersin herhangi bir şubesini almak zorunda olduğu için bu zaman periyodundaki alınması gereken diğer dersleri alamayacaktır. Ancak, aynı zaman periyoduna, bir öğrenci grubunun bir zorunlu şubeli dersinin tüm şubelerinin atanmaması şartıyla, zorunlu şubeli derslerin farklı şubeleri atanabilmelidir. Bu sayede, öğrenci çakışan zorunlu şubeli derslerden birini seçecek diğer dersin farklı bir şubesini farklı bir zaman periyodunda ayrıca seçebilecektir.

Z9. Öğrenci gruplarının müfredatlarında olan seçmeli dersler aynı müfredattaki zorunlu dersler ile çakışmamalıdır. Aynı şekilde, eğer öğrenci grubunun herhangi bir zorunlu şubeli dersinin tüm şubeleri aynı zaman periyoduna atandıysa, bu zaman periyoduna bu öğrenci grubunun müfredatındaki seçmeli derslerden herhangi biri atanmamalıdır.

3.1.2 Esnek Kısıtlar

HÜİİBF ders çizelgeleme probleminde mümkün olduğu kadarınca karşılanması gereken kısıtlar şöyle özetlenebilir:

E1. Birbirini takip eden müfredatların (dönemlerin) zorunlu dersleri birbiri ile mümkün olduğunca az çakışmalı. Örneğin, dört yıllık bir bölüm için birbirini takip eden müfredatlar (dönemler) sırasıyla 1-2, 2-3 ve 3-4 olmaktadır. Müfredat olarak kastedilen şey ortak dersler alan her bir öğrenci grubudur (sınıf).

E2. Birbirini takip eden müfredatların zorunlu şubeli dersleri, öğrencilerin şubeli derslerin en azından bir şubesinin seçilmesini mümkün olduğunca az engelleyecek şekilde atanmalı. Daha açık bir ifadeyle, bir müfredatta bir zorunlu şubeli dersin tüm şubeleri aynı zaman periyoduna atandıysa, aynı zaman periyoduna takip eden müfredatın zorunlu ve şubeli zorunlu dersleri ile mümkünse çakışmamalı.

Yukarıdaki kısıtla birlikte bu kısıtın amacı zorunlu dersten kalan öğrencilerin bir sonraki dönemde aynı dersi tekrardan almalarına imkân verebilmektir.

E3. Öğrenci gruplarının müfredatlarında yer alan seçmeli dersler mümkün olduğunca aynı zaman periyoduna atanmamalıdır.

E4. Eğer öğretim elemanları derslerinin atanması için zaman periyodu isteğinde bulunduysalar, bu istekler mümkün olduğunca karşılanmalıdır.

Son esnek kısıt öğretim elemanlarının memnuniyetini artırmayı amaçlayan bir kısıttır. Uygulamamızda öğretim elemanları verdikleri dersler için bir veya birden fazla zaman periyodu tercihinde bulunabilmektedirler. Modellerimiz bir ders için eğer sadece bir zaman periyodu isteği varsa bu isteği, birden fazla zaman periyodu isteği varsa da bu isteklerden herhangi birini karşılamaya çalışmaktadır. Bazı çalışmalarda ise Daskalaki ve Birbas (2005); Daskalaki vd. (2004) gibi öğretim elemanlarının tercihlerinin en çoktan en aza doğru puanlamaları ve buna göre oluşturulan matrisin yardımıyla öğretim elemanlarının memnuniyet düzeyleri artırılmaya çalışılmaktadır. Bu çalışma da dersler için zaman periyodu isteğinin toplanması modele gereğinden fazla verinin yüklenmesini engellemekte, sadece tercihte bunulan derslerin işleme alınmasını sağlamaktadır. En önemlisi öğretim elemanları çoğunlukla istemedikleri zaman periyotları vardır ve geri kalan zaman periyotları arasında önemli bir fark gözetmezler. Bu nedenle memnuniyet matrisi hazırlamak yerine doğrudan zaman periyodu isteği almak uygulama açısından daha kolay ve daha faydalı bir yaklaşım olabilir.

Son esnek kısıtın dışında kalan diğer esnek kısıtlar (E1-E3) öğrenciyi ilgilendiren kısıtlardır. Benzer yapıdaki bazı fakültelerde bu kısıtlar zorunlu kısıt olarak değerlendirebilmektedirler. Bu sayede öğrenciler istedikleri derslerin çoğunu istedikleri öğretim elemanlarından alabilme imkânları daha çok olmaktadır. Ancak HÜİİBF de insan ve fiziki mekân kaynağı yetersizliğinden dolayı bu kısıtlar zorunlu kısıtlar olarak değerlendirilememektedir. Aksi takdirde üniversite ders çizelgeleme problemimiz çözümsüz kalacaktır.

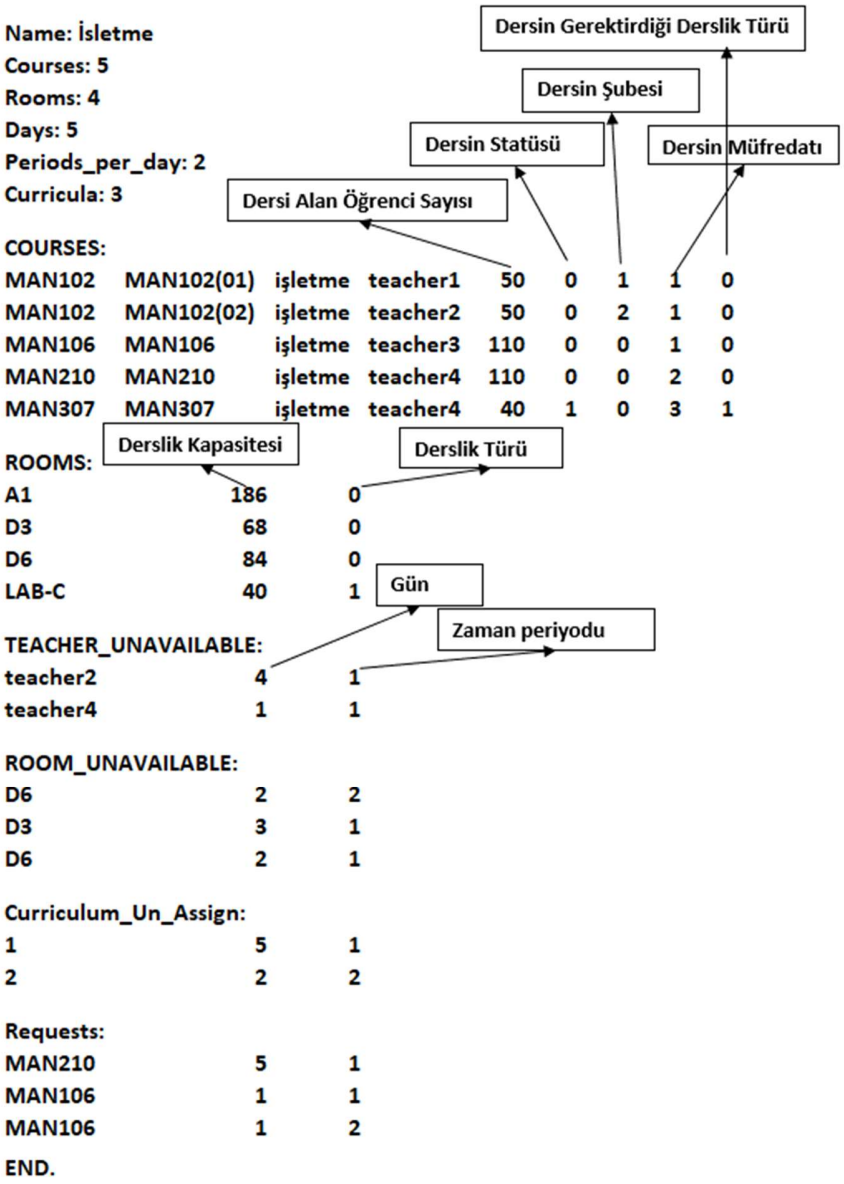
3.2 VERİ GİRİŞ FORMATI

Bu çalışmadaki veri giriş formatı oluşturulurken ITC-2007[4]'deki müfredat temelli üniversite ders çizelgeleme problemleri için kullanılan veri formatından esinlenilmiştir. Bonutti vd. (2012) çalışmasında ayrıntılı olarak bahsedilmektedir. Bu tez çalışmasında çözdüğümüz HÜİİBF ders çizelgeleme problemi ITC-2007 problemlerine göre daha kompleks yapıda olduğu için veri formatında değişikliğe gitmek gerekmiştir.

Çözücü için veri girdisi ".ectt" uzantısına sahip bir metin dosyasına kaydedilmektedir. Örnek bir veri formatı Tablo 2'de gösterilmektedir. Çözücü Python kodlama dili yardımıyla nesne tabanlı kodlama tekniğini kullanarak dosyadan ham veriyi almakta, bu verilerden sınıflar (*class*) oluşturmakta ve sonra bunları listelere yazmaktadır. Listelerden kastedilen kodlama dilinde nesneleri köşeli parantezler içine yazmaktır. Sınıflar (*class*) ise veriyi gruplara bölmek ve gruplar içindeki değişkenleri nesneleştirmeye yaramaktadır. Bu problemimizde sınıf adları sırasıyla: *Header, Courses, Rooms, Teacher_unavailable, Room_unavailable*, *Curriculum_un_assign* ve *Requests* olarak belirlenmiştir.

[4] ITC-2007: 2007 yılında 3 kategoride (okul, üniversite sınav ve üniversite ders çizelgeleme) başlatılan uluslararası çizelgeleme yarışması (*international timetabling competition*). http://www.cs.qub.ac.uk/itc2007/

Tablo 2. Örnek Veri Formatı



Çizelge probleminin merkezinde yer alan Courses sınıfı her bir dersi birer sınıf nesnesi olarak kaydetmektedir. Her ders nesnesinde sırasıyla dersin çekirdek ismi, dersin ismi (ID), dersin bölümü, dersin öğretim elemanı, dersi alan öğrenci sayısı, dersin statüsü (zorunlu ders: 0, seçmeli ders: 1), dersin şubesi (şube yoksa: 0, birinci şube:1, ikinci şube: 2, gibi…), dersin hangi müfredatta (öğrenci grubu) olduğu (her bir rakam farklı bir müfredatı temsil

etmektedir) ve dersin ne tür derslik (normal derslik: 0, laboratuvar:1, gibi…) gerektirdiği gibi bilgiler yer almaktadır.

Rooms sınıfı her dersliğin adını (ID), kapasitesini ve derslik türünü kaydetmektedir. *Teacher_unavailable* sınıfı bir öğretim elemanın ismini ve bu öğretim elemanının mevcut olmadığı gün ve zaman periyodunu kaydetmektedir. Bazı öğretim elemanları kişisel nedenlerden veya önceden atanmış ortak derslerden bazılarını veriyor olabilmelerinden dolayı bazı zaman periyotlarına atanmamaları gerekebilir. Bu nedenle bu öğretim elemanlarını tespit etmek ve mevcut olmadıkları zaman periyotlarını kaydetmek gerekir. Bu şekilde bu zaman periyotlarına bu öğretim elemanlarının atanması engellenebilecektir.

Room_unavailable sınıfı bir dersliğin ismini ve bu dersliğin mevcut olmadığı gün ve zaman periyodunu kaydetmektedir. Çünkü bazı dersliklere bazı ortak dersler önceden atanmış olabilir veya başka kullanımlar için ayrılmış olabilir. Bu gibi nedenlerden dolayı bazı dersliklere bazı zaman periyotları için herhangi bir ders ataması yapılmamalıdır.

Curriculum_un_assign sınıfı bir müfredatın (öğrenci grubu) için uygun olmayan gün ve zaman dilimini kaydetmektedir. Fakültelerin çoğu bölümü için bazı ortak zorunlu dersler vardır. Çoğunlukla bu dersler ders çizelgesine önceden yerleştirilir ve yerleri genellikle sabittir. Bu nedenle, öğrenci gruplarının müfredatlarında olan bu dersler öğrenci gruplarının diğer dersleri ile çakışmaması gerekir. *Curriculum_un_assign* sınıfı da hangi öğrenci grubu için hangi zaman periyodunun kapalı olduğunu göstermektedir.

Requests sınıfı ise derslerin atanmasının istendiği zaman periyodu isteklerini kaydetmektedir. Öğretim elemanları dersleri için zaman periyodu tercihinde bulunabilirler. Bir ders için bir veya birden fazla zaman periyodu tercihi olabilir. Çözüm modellerimiz bu istekleri esnek kısıtlar olarak değerlendirmekte ve mümkün olduğunca gerçekleştirmeye çalışmaktadır.

Bu kısımda tanımladığımız veri girdi formatı ders çizelgeleme problemimizin çözümü için kullandığımız tüm modeller için aynıdır. Ayrıca eğer veri TDP modeli için kullanılacaksa, dosyadan okunan veri yazılan kodlar aracılığıyla bu modelin gereksinim duyduğu parametre

ve parametre listelerine otomatik olarak dönüştürülmektedir. Benzer durum sezgisel çözümler içinde geçerlidir.

3.3 TAMSAYILI DOĞRUSAL PROGRAMLAMA MODELİ

Üniversite ders çizelge problemlerini çeşitli şekillerde modelleyip tamsayılı doğrusal programlama (TDP) ile çözmek istenebilir. Ancak, büyük ölçekli problemlerde etkin çözümler elde edebilmek için modellemenin doğru şekilde yapılması gerekir. Bu problem çeşidinin iyi birer doğrusal modelleme örneği için Daskalaki vd. (2004) ve MirHassani (2006) tarafından yapılan çalışmalara bakılabilir. Bu tez çalışmasında bu problem çeşidi için geliştirdiğimiz TDP modeli Daskalaki vd. (2004) çalışmasındaki TDP modelini temel almaktadır. Ancak bizim modelimiz, bilgisayarda yetersiz hafıza (*out of memory*) hatasına düşmemek için boş karar değişkenlerini üretmeyerek karar değişkeni sayısını önemli ölçüde azaltacak ve şubeli dersler ile seçmeli dersler için ekstra kısıtlar içerecek şekilde geliştirilmiştir. Ayrıca, dersler için istenilen sayıda zaman periyodu isteği belirlenebilmektedir. Modelimiz bu istekleri mümkün olduğu kadarıyla karşılayabilecek şekilde tasarlanmıştır.

Ders çizelgeleme problemi konusunda yapılan çalışmalara bakıldığında çok az sayıda çalışmanın şubeli ders durumunu ele aldığı görülmektedir. Bu çalışmaların çizelgeleme problemlerinde şubeli derslerin ele alınış biçimlerinin farklı olduğu görülmektedir. Bu konuda yapılan çalışmaların ana fikri derslerin şube sayılarının nasıl belirleneceğini veya optimal şube sayısının ne olduğunu bulmaktır. Yani genelde derslerin kaç şubeli olması gerektiği ve hangi dersin hangi öğretim elmanı tarafından verileceği bulunmaya çalışılmaktadır. Bu konuda şu çalışmalara bakılabilir: Beyrouthy vd. (2006), Broek vd. (2009), Müller ve Murray (2010) ve Schindl (2019). Gerçek hayat ders çizelgeleme problemlerine baktığımızda bazı kurumlarda hangi dersin kaç şube olacağı ve hangi öğretim elemanı tarafından verileceği önceden belirlenmektedir. Yani, bir dersi verebilecek öğretim elemanı sayısı ve dersi alan öğrenci sayısı gibi kriterler göz önüne alınarak dersler şubelere bölünmektedir. Bu çalışmada çözmeye çalıştığımız HÜİİBF ders çizelgeleme problemi de

böyle bir problemdir. Modellere şubeli derslerin kaç şubesi olduğu parametre olarak verilmektedir. Dolayısıyla, bizim problem yapımızda şubeli derslerin problemin çözüm kalitesini artıracak ve çakışma olmayacak şekilde atanabilmesi için gerekli olan kısıtları modellere eklemek gerekir.

3.3.1 Parametreler

Tamsayılı doğrusal programlama (TDP) modelinde kullanılacak indisler ve parametreler uygun formatta hazırlanmış veri dosyasından otomatik olarak üretilmektedir.

Modelde kullanılacak indisler aşağıdaki gibi tanımlanmaktadır:

- Derslerin atanabileceği muhtemel zaman periyotları seti I harfi ile ifade edilmektedir. Örneğin beş günlük iş günü içerisinde her gün iki zaman diliminden oluşuyorsa bu set şu şekilde olur; $I = \{1,2, \dots 10\}$. Problemimizde derslerin tüm saatleri art arda işlendiği için her bir zaman periyodu teneffüslerle birlikte 3 saatlik zaman diliminden oluşmaktadır.

- Ortak dersler alan öğrenci grupları birer müfredat olarak düşünüldüğünde her bir müfredat K harfi ile ifade edilmektedir. Örneğin dörder yıllık iki bölüm $K = \{1,2, \dots , 8\}$ şeklinde olacaktır.

- Dersleri veren öğretim elemanları L harfi ile ifade edilmektedir. $L = \{hoca\#1, hoca\#2, \dots, hoca\#L\}$

- Ders çizelgesinde olacak tüm dersler M harfi ile gösterilmektedir. $M = \{ders\#1, ders\#2, \dots, ders\#M\}$

- Derslerin yapılacağı derslikler N harfi ile gösterilmektedir. $N = \{derslik\#1, derslik\#2, \dots, derslik\#N\}$

Modelin indisleri belirlendikten sonra, bu indisler aracılığıyla parametre setleri oluşturulmaktadır. Parametre setleri, öğretim elemanlarının hangi dersleri, belli zaman periyotlarında müsait olmayan derslikleri ve derslerin atanamayacakları zaman periyotları gibi pek çok bilgiyi içinde barındıran veri kümelerdir. Parametre setlerini tanımlarken, veri kümelerinin indislere göre oluşturulmuş kesişim kümeleridir de diyebiliriz. Tablo 3'de problemimizin parametreleri özetlenmektedir. Bu parametrelerin tümü geliştirdiğimiz çözücü tarafında veri giriş formatına uygun şekilde girilen veriden kolayca üretilebilmektedir.

Tablo 3. Parametreler

K_i	$=$ | $k \in K$, i zaman periyodunda uygun olan öğrenci grubu seti |.
K_l	$=$ | $k \in K$, l öğretim elemanın ders verdiği öğrenci grubu seti |.
K_{li}	$= K_l \cap K_i$
L_i	$=$ | $l \in L$, i zaman periyodunda mevcut olan öğretim elemanı seti|.
L_k	$=$ | l $\in$ L, k öğrenci grubuna ders veren öğretim elemanı seti|.
L_{ki}	$= L_k \cap L_i$,
L_{kim}	$= L_k \cap L_i \cap L_m$
M_k	$=$ | $m \in M$, k öğrenci grubuna verilen ders seti|.
M_l	$=$ | $m \in M$, l öğretim elemanı tarafından verilen ders seti|.
M_n	$=$ | $m \in M$, n dersliğinde verilebilecek ders seti|.
M_{kl}	$= M_k \cap M_l$
M_{kln}	$= M_k \cap M_l \cap M_n$
N_i	$=$ | $n \in N$, i zaman periyodunda mevcut olan derslik seti|.
N_{mk}	$=$ | $n \in N$, k öğrenci grubuna verilecek m dersi için uygun olan derslik seti|.
N_{mki}	$= N_{mk} \cap N_i$
I_n	$=$ | $i \in I$, n dersliğinin uygun olduğu zaman periyodu seti|.
I_l	$=$ | $i \in I$, l öğretim elemanının uygun olduğu zaman periyodu seti|.
I_{ln}	$= I_l \cap I_n$
I_{nk}	$= I_n \cap I_k$
I_{lnk}	$= I_l \cap I_n \cap I_k$
C_k	$=$ | $c \in C$, k öğrenci grubunun şubeli zorunlu derslerinin çekirdek (core) ders seti |.

S_{kc} = | $s \in S$, k öğrenci grubunun c şubeli zorunlu dersinin şubeleri seti|.

$M_{kl}^{zorunlu}$= | $m \in M$, k öğrenci grubuna l öğretim elemanı tarafından verilen şubesiz zorunlu ders seti|.

$M_{k}^{seçmeli}$= | $m \in M$, k öğrenci grubuna verilen seçmeli ders seti|.

$M_{kl}^{seçmeli}$= | $m \in M$, k öğrenci grubuna l öğretim elemanı tarafından verilen seçmeli ders seti|.

$M_{k}^{şubeli}$= | $m \in M$, k öğrenci grubuna verilen şubeli zorunlu derslerin seti|.

I_{m}^{tercih}= | $i \in I$, m dersinin atanması için istenen zaman periyotları seti|.

M_{kl}^{tercih}= | $m \in M$, k sınıfına l öğretim elemanı tarafından verilen ve atanması için zaman periyodu tercihinde bulunulan ders seti|.

KK_i = | $(k_a, k_b) \in K$, i zaman periyodunda uygun olup art arda gelen (k_a ile k_b ilişkili öğrenci gruplarıdır) ikili (k_a, k_b) öğrenci grubu seti |.

a_k = |k öğrenci grubuna verilen toplam ders sayısı|

b_k = |k öğrenci grubuna verilen şubeli zorunlu ders (*core* ders) sayısı |

r_{kc} = |k öğrenci grubuna verilen c şubeli zorunlu dersinin şube sayısı|

say_t = |zaman periyodu isteğinde bulunulan toplam ders sayısı |.

ter_c = *1, her bir tercihin karşılanmaması durunda oluşacak birim maliyettir.*

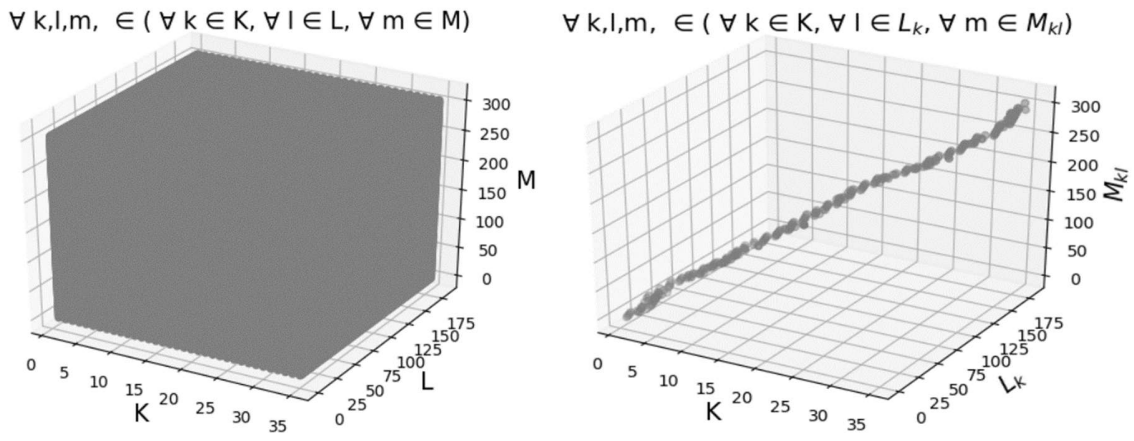
PRA = */*$(i, k, l, m, n) \in IxKxLxMxN$, *PRA, l öğretim elemanı tarafından k öğrenci grubuna verilen m dersi i zaman periyoduna n dersliğinde verilme önceliğini göstermektedir. /*

Parametre setlerinin Tablo 3'deki gibi indislere göre gruplanması hem toplam karar değişkeni sayısını ve hem de kısıt sayısını önemli ölçüde azaltmaktadır. M_k setini düşünelim… M tüm derslerin setini, k indisi eklenince de k öğrenci grubuna verilen derslerin setini göstermektedir. Benzer şekilde M_{kn} seti k öğrenci grubuna ait olan ve n dersliğine atanabilir derslerin setini, M_{kln} ise l öğretim elemanın k öğrenci grubuna ait olan derslerinden n dersliğine atanabilir olan ders setini gösterir. Örneğin, M setinde 305 ders varken, $M_{k=1}$ setinde 6, $M_{k=1,n=5}$ setinde 5 ve $M_{k=1,l=1,n=5}$ setinde ise sadece 1 ders yer almaktadır. Bu durum parametreleri indislere göre gruplamanın setleri önemli ölçüde küçülttüğünü göstermektedir. Ayrıca bu sayede birçok zorunlu kısıt ekstra kısıt gerektirmeden sağlanabilmektedir. Örneğin laboratuvar dersliği için sadece bu dersliğe atanabilecek dersler bu dersliğin parametre setinde yer alabileceği için ekstra bir kısıt yazmandan dersler sadece uygun olan dersliklere atanabilmektedir. Benzer şekilde bir

öğretim elemanı bulunmadığı günler gün setinde yer almamakta ve öğretime elemanları bulunmadıkları zaman periyotlarına atanmamaktadırlar.

Parametre setlerinin indislere göre gruplanarak oluşturulması problemin çözüm uzayını önemli ölçüde küçültebilmektedir. Bir karar değişkeninde sadece öğrenci grubu k, öğretim elemanı l, ders m indislerinin birer döngü halinde değiştiği bir kısıt varsayalım… HÜİİBF problemi için klasik durumda kısıt sayısı K, L ve M (35*184*305 = 1.964.200) setlerinin eleman sayılarının çarpımı kadar olmaktadır. Aynı kısıt, parametre setlerinin indislere göre hazırlanması durumunda ise kısıt sayısı 305'e düşmektedir. Çünkü bir parametre setinin elemanları sonraki setlerin indis değerleri olmakta ve bu şekilde sadece bu indislere uygun setler döngüde yer almakta, ilişkisiz elemanlar ise elimine edilmektedir.

Şekil 4'de kısıt sayısındaki değişim daha net görülmektedir. Soldaki grafik indissiz durumu, sağdaki grafik ise indisli durumu göstermektedir. Soldaki grafikte üç parametre setinin tüm elemanlarının birbiriyle ilişkili olduğu ve bu nedenle ilişki alanın ne kadar çok büyüdüğü görülebilmektedir. Oysaki sağdaki grafikte setlerin sadece birbiriyle ilişkili olan elemanları görülmektedir. Örneğin sağdaki grafik bize 0 – 5 bandında yer alan öğrenci gruplarının aşağı yukarı 0 – 50 bandında yer alan öğretim elemanlarından ders aldığını ve bu derslerinde de aşağı yukarı 0 – 50 bandında yer alan dersler olduğunu göstermektedir. Problemimizdeki her bir bölümün verisi arka arkaya eklenerek veri formatı oluşturulduğu için sağdaki grafik bu görünüme sahip olmuştur.



Şekil 4. Karar Değişkeni Sayısının Azalması

3.3.2 Karar Değişkenleri

Bir dersin hangi zaman periyodunda, hangi öğrenci grubuna, hangi öğretim elemanı tarafından, hangi derslikte verileceğini gösteren ikili (1-0) karar değişkeni şu şekilde tanımlanmaktadır;

$$\forall(i,k,l,m,n) \ \in (\forall i \in I, \forall k \in K_i, \forall l \in \ L_{ki}, \forall m \in M_{kl}, \forall n \in \ N_{mki})$$

$$x_{i,k,l,m,n} = \begin{cases} 1, & eğer\ m\ dersi\ l\ \text{öğretim elamanı}\ tarafından\ k\ \text{öğrenci grubuna} \\ & \quad i\ zaman\ periyodunda\ ve\ \ n\ dersliğinde\ veriliyorsa \\ 0, & diğer\ durumda \end{cases}$$

Modelimizde veriler indislere göre gruplandığı için $\boldsymbol{x_{i,k,l,m,n}}$ karar değişkenleri de gruplanmış setlere göre üretilerek, atanması mümkün olmayan boş karar değişkenleri üretilmemektedir. Bu sayede karar değişkeni sayısı önemli ölçüde azaltılabilmektedir. Şekil 4'te gösterilen örnek kısıtta bu durumu göstermektedir. Parametrelerin indislere göre gruplanmaması durumunda, karar değişkeni sayısı parametre setlerinin çarpımından oluşacağı için karar değişkeni sayısı çok büyük sayılara ulaşabilmektedir. Böyle durumlarda çoğunlukla bilgisayar yetersiz hafıza hatası (*out of memory error*) vermektedir. Yani bilgisayarda karar değişkenleri için yeterli hafıza açılamadığı için algoritma problemi çözmeye başlayamamaktadır. Örneğin; Al-Yakoob ve Sherali (2006, 2007) çalışmalarında veri boyutunun arttığı bazı ders çizelgeleme problemleri için yetersiz hafıza hatası almış ve bu durumu aşabilmek için iki aşamalı bir model geliştirmiştir. Ancak bu modelde de bazı problemler için yetersiz hafıza hatası almaya devam etmiştir.

Parametrelerin indislere göre gruplanmış setler şeklinde oluşturulması sayesinde büyük problemler için bilgisayarın yetersiz hafıza hatası verme probleminin önüne geçilebilmektedir. Aynı zamanda kısıt sayılarını da azaltabildiğimiz için çözüm süresi de önemli ölçüde azaltılabilmektedir. Aşağıdaki tabloda, problemimizdeki karar değişkeni sayısındaki önemli azalış net bir şekilde görülmektedir.

	Karar değişkeni sayısı
$x_{i,k,l,m,n} \quad (i,k,l,m,n) \in IxKxLxMxN \rightarrow$	942.816.000
$x_{i,k,l,m,n} \quad (i,k,l,m,n) \in (\forall i \in I, \forall k \in K_i, \forall l \in L_{ki}, \forall m \in M_{kl}, \forall n \in N_{mki}) \rightarrow$	57.551

Ders çizelgesinde yer alan dersler zorunlu ve seçmeli olarak iki statüye ayrılmaktadır. Bir öğrenci grubunun zorunlu derslerini, aynı öğrenci grubundaki tüm öğrenciler tarafından alınması gerektiği için bu derslerde herhangi bir çakışmanın olmaması gerekmektedir. Bu öğrenci grubunun seçmeli dersleri ise minimum sayıda çakışacak şekilde atanabilmelidir (bak: Bölüm .1). Çünkü kaynak yetersizliğinden dolayı seçmeli derslerin çakışmaması kısıtı uygulanabilir bir çözüme ulaşmayı mümkün kılmamaktadır. Seçmeli dersler için k öğrenci grubunun i zaman periyodundaki çakışma sayısını gösteren $\boldsymbol{y_{i,k}}$, $\boldsymbol{(i,k) \in (i \in I,\ k \in K_i)}$ tam sayılı karar değişkeni oluşturulmaktadır. Ayrıca her öğrenci grubu için dengeli bir seçmeli ders dağılımının olabilmesi için bu tamsayılı karar değişkenin alabileceği değerlerin üst sınırı 2 olarak belirlenmektedir. Bazı öğrencilerin alt veya üst dönemlerdeki zorunlu dersleri tekrardan alma durumları söz konusu olabilmektedir. Bu nedenle art arda gelen dönemlerin (öğrenci gruplarının) zorunlu dersleri problemimizde minimum çakışma olacak şekilde atanmalıdır. Bunun için $\boldsymbol{Z^{zorunlu}_{i,(k_a,\,k_b)}}$, $\boldsymbol{\left(i,(k_a,\,k_b)\right) \in \left(i \in I,\ (k_a,\,k_b) \in KK_i\right)}$ tamsayılı karar değişkeni oluşturulmaktadır**.** Ancak, zorunlu derslerin bazıları şubelere ayrıldığı için bu derslerin çakışma prosedürü farklıdır (bak: Bölüm 3.1). Bu nedenle şubeli zorunlu dersler için ayrıca $\boldsymbol{Z^{şubeli}_{i,(k_a,\,k_b)}}$, $\boldsymbol{\left(i,(k_a,\,k_b\right) \in (i \in I,\ (k_a,\,k_b) \in KK_i)}$ tamsayılı karar değişkeni oluşturulmaktadır.

3.3.3 Kısıtlar

Üniversite kurumları benzer özelliklerin dışında çok sayıda farklı özelliklere de sahiptirler. Bu farklılıklar fakülte ve bölüm bazında da olabilmektedir. Bu nedenle, ders çizelgeleme problemi için tüm kurumları kapsayacak genel TDP modeli sunmak pek mümkün değildir. Her model kurumların genel ve özel gereksinimlerini karşılayacak şekilde tasarlanmalıdır. HÜİİBF ders çizelgeleme probleminde fiziki mekân ve insan kaynağı yetersizliklerinden

dolayı problemimizin TDP modeli genel ders çizelgeleme modellerine göre daha karmaşık bir yapıya sahiptir. Bu karmaşık yapıya neden olan durumlar şu şekilde özetlenebilir:

Çoğu üniversite ders çizelgeleme probleminde seçmeli ve zorunlu ders ayrımı söz konusudur. Genel de modellerde seçmeli ders havuzu vardır ve bu seçmeli derslerin birbiriyle çakışmaması istenir. Ancak, bizim problemimizde uygulanabilir bir çözüm elde edebilmek için seçmeli dersler müfredatlara (öğrenci gruplarına) ayrılmıştır. Bu müfredatlardaki derslerin de mümkün olduğu kadarıyla aynı zaman periyoduna atanmaması sağlanmaya çalışılmalıdır. Bu da demek oluyor ki, bu durum esnek kısıt olarak ele alınmalıdır.

HÜİİBF ders çizelgeleme probleminde bazı dersler birbirine tam alternatif olacak şekilde şubelere ayrılmıştır. Bu sayede fiziki mekân yetersizliği olan fakültede öğrenciler zorunlu derslerinin tümünü alabilme imkânına erişmektedirler. Bu şubeli derslerinde bazıları zorunlu bazıları ise seçmeli ders statüsündedir. Şubeli seçmeli dersler normal seçmeli dersler gibi işlem görürken şubeli zorunlu derslerin durumları farklıdır. Modelimiz şubeli zorunlu derslerin şubelerinin aynı zaman periyoduna atanabilmesini sağlayacak esneklikte olmalıdır. Ancak, eğer bir k öğrenci grubunun bir şubeli zorunlu dersinin tüm şubeleri aynı zaman periyodun atandıysa, bu durum öğrenci grubunun diğer derslerinin seçimini engelleyeceği için bu öğrenci grubunun diğer dersleri bu zaman periyoduna atanmamalıdır.

Bazı öğrencilerin bazı derslerden başarısız olmasından dolayı alt dönemden ya da üst dönemden ders almak durumunda kalınabildiği için birbirini takip eden müfredatların zorunlu derslerinin çakışmaması sağlanmalıdır. Bizim problemimizde, uygulanabilir bir çözüme ulaşabilmek için birbirini takip eden müfredatların zorunlu derslerinin aynı aman periyoduna atanması minimize edilmelidir. Şubeli zorunlu dersler için de benzer durum söz konusudur. Yani bir k öğrenci grubunun bir şubeli zorunlu dersinin tüm şubeleri aynı zaman periyoduna atandıysa diğer öğrenci grubunun şubeli zorunlu dersleri bu zaman periyoduna minimum sayıda atanmalıdır. Ancak, buna rağmen bazı derslerin çakışmaması gerektiği halde çakıştıysa, bu derslerin çakışmasını engellemek için modele ekstra bir kısıt kolayca eklenebilir.

Literatürdeki şubeli dersleri optimal sayıda şubeye ayırmaya çalışan çalışmaların aksine bizim çalınmamızda da dersler eldeki imkânlar çerçevesinde kaç şubeye ayrılacağı ve şubelerin hangi öğretim elemanları tarafından verileceği önceden belirlenmiştir. Bu nedenle, bizim amacımız TDP modelimizi bir öğrenci grubunun alaması gereken şubeli zorunlu dersin en azından bir şubesini seçebilecek şekilde tasarlamak olmalıdır. Bu sayede bu tür yapıya sahip çizelgeleme problemlerinde şubeli zorunlu derslerin nasıl modellenebileceği konusunda bir yaklaşım sunulmuş olacaktır.

HÜİİBF çizelgeleme probleminin tüm gereksinimlerini sağlayan kısıtlar açıklamalarıyla birlikte aşağıda sırasıyla verilmektedir:

Her bir öğretim elemanı bir zaman periyodunda en fazla bir derslikte sadece bir ders verebilmelidir:

$$\forall i \in I, \qquad \forall l \in L_i$$

$$\sum_{k \in K_{li}} \sum_{m \in M_{kl}} \sum_{n \in N_{mki}} x_{i,k,l,m,n} \leq 1 \tag{1}$$

Kısıt (2)'de denklemin ilk kısmı her k öğrenci grubunun şubesiz zorunlu derslerinin aynı zaman periyoduna atanmasını engellemektedir. Denklemin ikinci kısmı ise k öğrenci grubunun şubeli zorunlu derslerinin her birinin ayrı ayrı şubesiz zorunlu dersler ile aynı zaman periyoduna atanmasını engellemektedir. Ayrıca bu kısıt şubeli olan zorunlu derslerin şubelerinin aynı zaman periyoduna atanabilmesine de izin vermektedir.

$$\forall i \in I, \qquad \forall k \in K_i, \ \forall m_s \in M_k^{subeli}$$

$$\sum_{l \in L_{ki}} \sum_{m \in M_{kl}^{Zorunlu}} \sum_{n \in N_{mki}} x_{i,k,l,m,n} + \sum_{l \in L_{kim_s}} \sum_{n \in N_{m_s ki}} x_{i,k,l,m_s,n} \leq 1 \tag{2}$$

Kısıt (3)'de denklemin ilk kısmı kısıt (2)'nin ilk kısmı ile aynıdır. Denklemin ikinci kısmı ise k öğrenci grubunun seçmeli derslerinin her birinin ayrı ayrı şubesiz zorunlu dersler ile aynı zaman periyoduna atanmasını engellemektedir.

$$\forall i \in I, \quad \forall k \in K_i, \ \forall m_s \in M_k^{seçmeli}$$

$$\sum_{l \in L_{ki}} \sum_{m \in M_{kl}^{Zorunlu}} \sum_{n \in N_{mki}} x_{i,k,l,m,n} + \sum_{l \in L_{kim_s}} \sum_{n \in N_{m_s ki}} x_{i,k,l,m_s,n} \leq 1 \tag{3}$$

Eğer k öğrenci grubunun herhangi bir şubeli zorunlu dersinin tüm şubeleri aynı zaman periyoduna atandıysa, bu zaman periyoduna bu öğrenci grubunun müfredatındaki diğer derslerin hiç birisinin atanmaması kısıt (4) ile sağlanmaktadır. $C_{k[z]}$ k öğrenci grubunun şubeli zorunlu çekirdek (*core*) ders setinden z. sıradaki dersi göstermektedir.

$$\forall i \in I, \ \forall k \in K_i, \ \forall z \in [1, \dots, b_k], \ m_s \in M_k, \ m_s \setminus S_{kC_{k[z]}}$$

$$\sum_{m \in S_{kC_{k[z]}}} \sum_{l \in L_{kim}} \sum_{n \in N_{mki}} x_{i,k,l,m,n} + \sum_{l \in L_{kim_s}} \sum_{n \in N_{m_s ki}} x_{i,k,l,m_s,n} \leq r_{kC_{k[z]}} \tag{4}$$

Her öğrenci grubunun müfredatında yer alan seçmeli derslerin atanma durumları problemimizin yapısı gereği esnek kısıt şeklinde düzenlenir. Yani her müfredattaki seçmeli derslerin aynı zaman periyoduna birden fazla sayıda atanmaması yerine, $y_{i,k}$, $(i,k) \in (i \in I, \ k \in K_i)$ karar değişkenin yer aldığı kısıt (5) yardımıyla minimum sayıda atanabilmesi sağlanmaktadır.

$$\forall i \in I, \quad \forall k \in K_i$$

$$\sum_{l \in L_{ki}} \sum_{m \in M_{kl}^{Seçmeli}} \sum_{n \in N_{mki}} x_{i,k,l,m,n} \leq 1 + y_{i,k} \tag{5}$$

Birbirinin devamı olan dönemlerin (her öğrenci grubu bir dönemi temsil etmektedir) öğrencileri bazı zorunlu dersleri önceki dönemden tekrardan almak durumunda olabilirler.

Modelimizde esnek kısıt olarak ele alınan bu durum art arda gelen öğrenci gruplarının zorunlu derslerinin aynı zaman periyoduna atanma sayıları kısıt (6) ile minimum sayıda tutulmaktadır.

$$\forall i \in I, \quad \forall (k_{a,}\, k_b) \in KK_i$$

$$\sum_{d \in [k_{a,}\, k_b]} \sum_{l \in L_{di}} \sum_{m \in M_{dl}^{zorunlu}} \sum_{n \in N_{mdi}} x_{i,d,l,m,n} \leq 1 + Z_{i,(k_{a,}\, k_b)}^{zorunlu} \tag{6}$$

Kısıt (7) ile kısıt (6)' daki duruma benzer durum bu kez şubeli zorunlu dersler için yapılmaktadır. Örneğin; eğer k_a/k_b öğrenci grubunun herhangi bir şubeli zorunlu dersinin tüm şubeleri aynı zaman periyoduna atandıysa, bu zaman periyoduna k_b / k_a öğrenci grubunun müfredatındaki şubeli zorunlu dersler mümkün olduğunca az sayıda atanmalıdır.

$$\forall i \in I, \forall (k_{a,}\, k_b) \in KK_i,\ \forall k_1 \in [k_{a,}\, k_b],\ \forall z \in [1, \ldots, b_{k_1}],\ \forall k_2 \in [k_{a,}\, k_b],\ \forall m_s \in M_{k_2}^{şubeli},\ k_1 \neq k_2$$

$$\sum_{m \in S_{k_1 C_{k_1[z]}}} \sum_{l \in L_{k_1 i m}} \sum_{n \in N_{m k_1 i}} x_{i,k_1,l,m,n} + \sum_{l \in L_{k_2 i m_s}} \sum_{n \in N_{m_s k_2 i}} x_{i,k_2,l,m_s,n} \leq r_{k_1 C_{k_1[z]}} + Z_{i,(k_{a,}\, k_b)}^{şubeli} \tag{7}$$

Kısıt (8) ile her bir derslik bir zaman periyodunda en fazla bir derse, bir öğretim elemanına ve bir öğrenci grubuna atanabilmektedir.

$$\forall n \in N, \quad \forall i \in I_n$$

$$\sum_{k \in K_i} \sum_{l \in L_{ki}} \sum_{m \in M_{kln}} x_{i,k,l,m,n} \leq 1 \tag{8}$$

Kısıt (9) ile her bir öğrenci grubunun müfredatında yer alan tüm dersleri ders çizelgesinde yer alması sağlanmaktadır.

$$\forall k \in K$$

$$\sum_{l \in L_k} \sum_{m \in M_{kl}} \sum_{n \in N_{mk}} \sum_{i \in I_{lnk}} x_{i,k,l,m,n} = a_k, \tag{9}$$

Kısıt (10) ile her dersin çizelgeye sadece bir kez atanması sağlanmaktadır.

$$\forall k \in K, \forall l \in L_k, \forall m \in M_{kl}$$

$$\sum_{n \in N_{mk}} \sum_{i \in I_{lnk}} x_{i,k,l,m,n} = 1 \tag{10}$$

3.3.4 Amaç Fonksiyonu

Min. Z =

$$\left\{ \begin{array}{l} (1)\ \sum_{i \in I} \sum_{k \in K_i} y_{i,k} + \\ (2)\ \sum_{i \in I} \sum_{(k_a, k_b) \in KK_i} Z^{zorunlu}_{i,(k_a, k_b)} + \\ (3)\ \sum_{i \in I} \sum_{(k_a, k_b) \in KK_i} Z^{subeli}_{i,(k_a, k_b)} + \\ (4)\ say_t * ter_c - \left(ter_c * \sum_{i \in I^{tercih}_m} \sum_{n \in N_{mki}} x_{i,k,l,m,n} \ \forall k \in K, \forall l \in L_k, \forall m \in M^{terci}_{kl} \right) \end{array} \right\} \tag{11}$$

Amaç fonksiyonu toplam memnuniyetsizliği en aza indirecek şekilde dört kısımdan oluşturulmaktadır. İlk kısım (1) öğrenci gruplarının müfredatında olan seçmeli derslerin aynı zaman periyoduna atanma sayılarının toplamını minimize etmektedir. İkinci kısım (2), birbirinin devamı olan dönemlerin zorunlu derslerinin aynı zaman periyoduna atanma sayılarının toplamını minimize etmektedir. Üçüncü kısım (3), birbirinin devamı olan

dönemlerin şubeli zorunlu derslerinin (birbirinin devamı olan iki dönemin aynı zaman periyoduna atanan şubeli zorunlu derslerinin aynı anda seçilememesi şartıyla) aynı zaman periyoduna atanma sayılarının toplamını minimize etmektedir. Dördüncü kısım (4) ise derslerin atanma tercihlerini gerçekleştirme düzeyini maksimum yapan kısımdır. Bir ders için birden fazla zaman periyodu tercihi olabilir ve model bu tercihlerden hangisi uygunsa onu gerçekleştirmeye çalışmaktadır. Görüldüğü gibi amaç fonksiyonun ilk üç kısmı öğrencilerin ders tercih seçeneklerini artırmaya yönelik olup öğrenci memnuniyetini dikkate almaktadır. Son kısımda, öğretime elemanlarının verdikleri dersler için istedikleri zaman periyotlarını sağlamaya yönelik olup öğretim elemanı memnuniyetini dikkate almaktadır. Bu kısımların önem düzeylerinin birbirinden farklı olduğu düşünülmesi durumunda, bu kısımlar için birer maliyet katsayısı belirlenerek gerçekleştirme öncelikleri değiştirilebilir. Problemimizin çözümünde öğretim elemanlarının tercihlerini gerçekleştirme katsayısı 1 (ter_c = 1) olarak belirlenmektedir.

Yukarıda tanımladığımız TDP modelini Python kodlama dili yardımıyla kodlanmakta ve çözüm için Gurobi6.2.5 optimizasyon çözücüsü kullanılmaktadır.

3.4 SEZGİSEL ALGORİTMALAR

Bu bölümde, üniversite ders çizelgeleme problemi iki aşamada çözüme kavuşturulmaktadır. İlk aşamada sadece zorunlu kısıtlar sağlanarak uygulanabilir bir başlangıç çözümü elde edilmektedir. İkinci aşamada ise zorunlu kısıtlar ihlal edilmeden esnek kısıtlar olabildiğince sağlanarak çözüm kalitesi artırılmaya çalışılmaktadır. Başlangıç çözümü aşaması için bir açgözlü sezgisel (*greedy heuristics*) ve bir iteratif ileri arama (*iterative forward search*-IFS) algoritması kullanılmaktadır. Çözümü iyileştirme aşaması olan ikinci aşamada ise en iyi bilinen yerel arama algoritmalarından tabu arama (*tabu search*-TS) ve benzetim tavlaması (*simulated annealing*-SA) algoritmaları kullanılmaktadır.

3.4.1 Başlangıç Çözümü Aşaması

Bu aşamada amaç zorunlu kısıtları ihlal etmeden tüm derslerin atandığı uygulanabilir bir çözüm elde etmektir. Problemin çözümüne başlangıçta boş bir çizelge ile başlanır. Daha sonra dersler uygun pozisyonlara atanarak ders çizelgesi ilk aşama için tamamlanmış olur. Derslerin atandığı pozisyonlar derslik ve zaman periyodundan (r = derslik, t = zaman periyodu) oluşan ikili değişkenlerdir. Bu pozisyonlara zorunlu kısıtları ihlal etmeden dersleri atayacak iki alternatif yaklaşım aşağıda açıklanmaktadır.

3.4.1.1 Açgözlü Sezgisel

Ders çizelgeleme probleminin başlangıç çözümü için kullandığımız ilk yöntem bir açgözlü sezgiseldir (*a greedy heuristics*-GH). Bu tür sezgisellerin temel mantığı mevcut alternatiflerden en iyi olanları en önce değerlendirerek çözümde ilerlemek ve mümkün olduğunca hızlı bir çözüm elde etmekten gelmektedir. Bu çalışmaya uyarladığımız GH sezgiselinde ise dersler atanma zorluklarına göre sıralanarak, atanması en zor olan ders en önce atanmaktadır. Dersleri bu şekilde öncelikleme fikri Lü ve Hao (2010) çalışmalarında kullandığı sıralı açgözlü (*sequential greedy*) sezgiselinden esinlenilerek problemimize uyarlanmıştır.

Bu tez çalışmasında kullanılan GH sezgiseli, Lü ve Hao (2010) çalışmasında olduğu gibi, atanma zorluğuna göre seçilen dersin atanacağı pozisyonu, uygun pozisyonlar arasından rastgele seçerek belirlemektedir. Biz bu çalışmada ayrıca, GH sezgiselini, uygun pozisyonu rastgele seçmek yerine esnek kısıtları ihlal etme açısından en uygun olan pozisyonu seçecek şekilde değiştirerek alternatif bir sezgisel (*GH_goodPos*) daha kullanmaktayız. Daha açık bir ifadeyle *GH_goodPos* sezgiseli şu şekilde çalışmaktadır: Öncelikle atanacak ders GH sezgiselindeki gibi seçilmektedir. Devamında bu ders için uygun pozisyonlar belirlenmektedir. Son aşamada, seçilen dersin atanması durumunda oluşturacağı ek maliyet her bir uygun pozisyon için hesaplanıp, maliyeti en az olan pozisyonlardan biri seçilmektedir.

Tablo 4'te uygun pozisyonları maliyet açısından öncelik sıralamasına koyan algoritmanın sözde kodu gösterilmektedir. Bu algoritmaya atanacak ders ve boş pozisyonların listesi girdi olarak verilmektedir. Atanacak ders m ise algoritma boş pozisyonların her birinin m dersine uygunluğunu kontrol etmektedir (satır 3). Daha sonra uygun olan pozisyonların her birinin maliyeti, (m dersi atanması durumunda oluşacak maliyeti) hesaplanmaktadır. Son aşamada algoritma m dersi için en az maliyet oluşturacak pozisyonlardan birini seçmektedir.

Tablo 4. Pozisyon Öncelik Sıralama Sözde Kodu (*pseudo-code*)

```
Fonksiyon: orderPositionsByPriority(m, emptyPositions)
Girdiler: m, emptyPositions, #m atanacak dersi ifade etmektedir.
1: goodPositions = []
2: for pos in emptyPositions:
3:     if courseFitsIntoPosition(m, pos) is Ture: # p pozisyonu m dersi için uygunsa
4:         cost = costPosition(m, p)      # her m dersinin p pozisyonuna atanması durumunda oluşacak maliyet
5:         (cost, p) ikilisini goodPositions listesine ekle
6: goodPositions'daki pozisyonları maliyete göre sırala
7: posList = [p for (cost, p) in goodPositions]  #sadece pozisyonları listele
7: return posList
```

Ders çizelgeleme problemimizdeki derslerin zorunlu/seçmeli ve şubeli zorunlu/şubeli seçmeli statüsünde olmalarından dolayı atanma prosedürleri farklıdır. Bu derslerin atanma zorluklarını da bu farklılıklar belirlemektedir. Buradan hareketle GH sezgiselimizin parametreleri şöyle formüle edilebilir:

- TT: henüz tamamlanmamış, fakat atanmış derslerin zorunlu kısıtları ihlal etmediği güncel ders çizelgesidir.

- aps_m : $m \in M$, m dersi için TT çizelgesinde mevcut olan uygun pozisyon (dersin atanması durumunda hiçbir zorunlu kısıtın ihlal edilmediği pozisyonlardır) sayısıdır. Yukarıda bahsedildiği gibi pozisyon olarak kastedilen şey derslerin atanabileceği derslik zaman periyodu (r, t) ikilisidir. Dersler bu pozisyonlara atanmaktadır.

- nz: zorunlu dersler için öncelik katsayısı

- nsz: şubeli zorunlu dersler için öncelik katsayısı

- ns: seçmeli dersler için öncelik katsayısı

Bir dersin öncelik değeri dersin öncelik katsayısının ders çizelgesinde aynı ders için mevcut olan uygun pozisyon sayısına bölünerek elde edilmektedir. Örneğin bir zorunlu m dersinin öncelik değeri nz/aps_m bölümüne eşittir. Bu değerlerin büyükten küçüğe doğru sıralanmasıyla derslerin öncelik sırlaması belirlenir. Yani öncelik katsayısı daha yüksek, atanabilecek pozisyon sayısı daha az olan ders daha önce atanmaktadır. Uygun pozisyon sayısı daha az olan derslerin öncelik sırlaması daha yüksek olmalıdır ki bu pozisyonlar diğer dersler tarafından tüketilmesin. Dolayısıyla nz, nsz, ns katsayılarını doğru bir şekilde belirlenmesi de oldukça önemli olmaktadır. Bu katsayılar müfredatta olan tüm derlesin çizelgeye atanabilmesini, atanama hızını ve başlangıç maliyetini etkilemektedir. Bu katsayıların değerleri problemin yapısına ve boyutuna göre de değişkenlik gösterebilir.

GH sezgiselinin *GH_goodPos* sezgiselinden farkı GH atanacak ders için pozisyon seçimini uygun pozisyonlar arasından rastgele seçerken, *GH_goodPos* uygun pozisyonlar arasından esnek kısıtları da dikkate alarak pozisyon seçimi yapmaktadır. Tablo 5'te *GH_goodPos* sezgiselinin sözde kodu verilmektedir.

GH_goodPos sezgiselin aşamaları şöyle özetlenebilir: (1) dersler öncelik sırlamasına konduktan (satır 5) (2) sonra öncelik sıralaması en yüksek olan ders seçilir (satır 7). (3) Seçilen ders için atlanılabilir pozisyonlar belirlenir ve (4) bu pozisyonların her biri için dersin pozisyona atanması durumunda esnek kısıtları ihmal etme maliyetleri hesaplanır (satır 8). (5) Sonrasında seçilen ders en düşük maliyetli pozisyona atanır (satır 13). (6) Uygun pozisyon kalmamasından dolayı atanamayan derslerin olması durumunda, bu derslerin sayısı kadar atanmış ders çizelgeden çıkarılarak bu derslerin boşalan pozisyonlarına atanamayan derslerin atanması denenir (satır 15- 28). (7) Bu aşamanın sonunda da atanamayıp kalan dersler çizelgeden çıkarılan derslerin yanına eklenerek atama döngüsü baştan tekrar edilir. Döngü atanmayan ders kalmayana kadar veya belirlenen tekrar sayısına kadar devam eder.

Tablo 5. Açgözlü Sezgisel İçin Başlangıç Çözümü Sözde Kodu (*pseudo-code*)

```
Fonksiyon: constructTimetable_GH_goodPos(tt, events)
Girdiler: events = [tüm derlser],  tt = timetabling (güncel ders çizelgesi)
1: unEvents = []  #atanma denemesi başarısız olan dersler listesi
2: iteration = 0
3: while len(events) > 0 and len(unEvents) > 0 or iteration< 1000:  # bu iki listedeki ders sayıları sıfırlanıncaya kadar  veya 1000 tekrara ulaşıncaya kadar.
4:     iteration += iteration +1
5:     orderEventsByPriority(nz,nsz,ns)  #events listesindeki dersleri öncelik sırlamasına göre sırala
6:     for i in range(len(events)):  #events listesindeki ders sayısı kadar döndür
7:         m = events listesinden en yüksek önceliğe sahip olan dersi seçerek listeden çıkar
8:         posList = orderPositionsByPriority (m, emptyPozitions)
9:         if len(posList) = 0:  # m dersi için uygun pozisyon yok ise
10:                m dersini atanmayan dersler (unEvents)   listesine ekle
11:        else:                # m dersi için uygun pozisyon/pozisyonlar var ise
12:            p = m dersi için, posList 'den oluşturacağı maliyet toplamı en az olan pozisyonlardan biri seçilir.
13:            m dersini p pozisyonuna ata
14:    newPos =[]   #çözümden çıkarılan derslerin boşta akalan pozisyonları listesi
15:    for i in range(len(unEvents)): # unEvents listesindeki ders sayısı kadar döndür
16:        p = çizelgede dolu olan pozisyonlardan birini rastgele seç
17:        m = p pozisyonundaki m dersini kaldır.
18:        m dersini tekrardan events listesine ekle
19:        p pozisyonun newPos listesine ekle
20:    for i in range(len(unEvents))):
21:        m = p pozisyonundaki m dersini kaldır.
22:        assigned = False
23:        for p in newPos: # newPos listesindeki pozisyonları döndür
24:            if courseFitsIntoPosition(m, p): #eğer p pozisyonu m dersi için uygunsa
25:                    m dersini p pozisyonuna ata
26:                    newPos listesindeki p pozisyonunu kaldır.
27:                    assigned = True
28:                    break        # m dersi için uygun bir pozisyon bulunduktan sonra for döngüsünden çık
29:        if assigned = False:
30:            m dersini events listesine ekle
31: return len(events)  # events listesindeki ders sayısını döndür.
```

Açgözlü sezgisel çok hızlı bir şekilde başlangıç çözümüne ulaşsa da her zaman tamamlanmış bir çözüme ulaşmayı garanti etmemektedir (Lü ve Hao, 2010). Bu çalışmadaki ders çizelgeleme problemlerinin hepsinde bu sezgisel tamamlanmış başlangıç çözümleri vermektedir. Ancak derslerin öncelik sıralamasının doğru ayarlanmaması, problem kompleksliğinin artması veya problem yapısının değişmesi gibi nedenlerden dolayı açgözlü sezgisel ile bir başlangıç çözümüne ulaşmak mümkün olmayabilir. Bu nedenle, üniversite

ders çizelgeleme çözücümüz için alternatif bir başlangıç çözümü veren farklı bir sezgisel daha önermek faydalı olacaktır. Bu algoritma aşağıdaki başlıkta anlatılmaktadır.

3.4.1.2 İteratif İleri Arama Algoritması (*Iterative Forward Search*)

Müller vd. (2004) yılındaki çalışmada başlangıç çözümü için çakışma temelli istatistik (CBS) yöntemini tavsiye etmektedirler. Bu iteratif ileri arama algoritması büyük problemlerde uygulanabilir bir çözüm elde edebilme ihtimalini artırma ve bu süreci hızlandırma amacı gütmektedir. Bu yöntemin arkasındaki ana fikir, atama sırasında oluşan çakışmalar hafızada tutularak, sonraki aşamalarda aynı atma seçeneğinin denememesini sağlama çabasına dayanmaktadır. Algoritma bu süreçleri ezberleyerek ilerlemektedir.

Bu yaklaşımda seçilen bir V_a değişkeni v_a değerine atanacağı zaman, bu değişken $V_b = v_b$ değeri ile çakışıyorsa, $V_b \neq v_b$ çözümden çıkarılır ve V_b atanmamış değişkenler listesine eklenir. Daha sonra $V_a = v_a$, $V_b \ \ v_b$ çiftinin çakışma sayısı 1 artırılır. Bu çakışma sayılarının hafızada tutulması sayesinde V_a değişkeni için bir değer belirlenirken çakışma sayısı en az olan değerlerden biri seçilir. Bu süreç, belli bir iterasyon sayısına veya eşleştirilmeyen değişken kalmayana kadar devem eden bir döngü şeklinde ilerler. Atanmayan olay kalmadığında bu aşama sonlanmış olur. Ders çizelgeleme probleminde değişken dersleri, değerlerde derslerin atanacağı pozisyonlar olmaktadır. Aşağıdaki denklem, V_a değişkenin v_a değerine atandığı için, v_b değerine atanan V_b değişkeni ile çakıştığı sayıların toplamını (c_{ab}) göstermektir.

$$CBS[V_a = v_a \ \rightarrow \ \neg V_b = v_b] = c_{ab}.$$

Ders çizelgeme probleminde IFS algoritması başlangıçta boş olan ders çizelgesine CBS yöntemiyle dersleri atayarak tüm derslerin atandığı bir başlangıç çözümü elde etmeye çalışmaktadır. Bu süreç, bir m dersinin bir p pozisyonuna atanması durumunda, diğer bazı derslerin önceden atanmış olduğu pozisyonlar ile çakışma söz konusu oluyorsa, bu dersler çözümden kaldırılarak atanmayanlar listesine eklenir. Devamında m derside p pozisyonuna

atanır. Aynı zamanda bu dersiler ilgili çakışmalar için çakışma sayıları hafızada tutulur. Bu sayede dersler için pozisyonlar belirlenirken geçmişteki çakışma sayıları dikkate alınabilmektedir. Süreç atanacak ders kalmayana kadar devam eder ve atanacak ders kalmayınca sonlanır. Tablo 6'da IFS algoritmasının sözde kodu verilmektedir.

Tablo 6. İteratif İleri Arama Algoritması İçin Başlangıç Çözümü Sözde Kodu (*pseudo-code*)

```
Fonksiyon: cunstructTimetable_cbsIFS(tt, events)
Girdiler: events = [tüm derlser], tt = timetabling (ders çizelgesi)
unEvents = []      #atanma denemesi başarısız olan dersler listesi
α = len(events)        # güncel çözüm – henüz atanmamış ders sayısı
β = α                  # şimdiye kadarki en iyi çözüm
while len(events) > 0 :  # events listesindeki ders sayısı sıfırlanıncaya kadar döndür.
    for i in range(len(events)):  # events listesindeki ders sayısı kadar döndür
        m = selectVariable(events)   # events listesinden rastgele bir m dersini kaldır
        p = selectValue(tt, m)   # m dersi için seçilen p pozisyonu
          N  = conflicts(tt, m, p)   # m dersinin p pozisyonuna atanması durumunda çakıştığı ders pozisyon çiftleri
        N listesindeki m ile çakışan dersleri çizelgeden (tt) kaldır
        N listesindeki dersleri çözümden çıkarılan dersleri (unEvents) listesine ekle
        m dersini p pozisyonuna ata
    for ev in unEvents:      # unEvents listesindeki dersleri döndür
        ev dersini events listesine ekle
        ev dersini unEvents listesinden çıkar
    α = len(events)        # yeni çözümü güncelle
    if  α  <  β:             # yeni çözüm şimdiye kadarki en iyi çözümden daha iyiyse.
            β = α
return  β                  # en iyi çözümü döndür
```

Değişken Seçimi: Yukarıda da bahsedildiği gibi IFS algoritması için bir değişken seçimi fonksiyonuna ihtiyaç vardır. Değişken seçimi çeşitli kriterlere göre yapılabilir ya da hiçbir kriteri dikkate almadan değişkenler arasından rastgele yapılabilir. Ders çizelgeleme probleminde; ders seçimi, rastgele ya da yukarıda bahsettiğimiz GH sezgiselindeki gibi dersleri atanma önceliğine göre sıralayıp öncelik sırasına göre yapılabilir. GH algoritmasının ders seçim aşamasını IFS algoritmasına dâhil ettiğimizde, bu algoritmanın ders seçimi yapan fonksiyonun sözde kodu Tablo 7'de gösterildiği gibi olmaktadır.

Tablo 7. Değişken Seçimi Sözde Kodu (*pseudo-code*)

Fonksiyon: selectVariable(events)
Girdiler: events, nz, nsz, ns *# atanmayı bekleyen güncel ders listesi ve derslerin öncelik katsayıları*
1: → orderEventsByPriority(nz,nsz,ns) *#events listesindeki dersleri öncelik sırlamasına göre sırala*
2: → m = events listesinden en yüksek önceliğe sahip ders
3: **return** m

Değer seçimi: Değişken seçildikten sonra bu değişkenin atanacağı bir değere ihtiyaç vardır. Genelde, en faydalı tavsiye en uygun değeri seçmektir. Bu nedenle, bir değişken için en fazla tercih edilen ve en az hasara neden olacak değer bulunmaya çalışılmalıdır. Ders çizelgeleme problemi açısından bunun anlamı, diğer dersler ile gelecekte gireceği çakışmalar açısından minimum potansiyele sahip bir pozisyon bulma çabasıdır. Bir ders için bulunan pozisyonlar birden fazla sayıda olması durumunda, bu pozisyonlar arasından rastgele bir pozisyon seçimi ya da bu pozisyonlar arasından esnek kısıtları en az ihlal eden pozisyon belirlenerek seçim yapılabilir. Müller (2009) çalışmasında pozisyon seçimini uygun pozisyonlar arasından rastgele yapmaktadır. Bu tez çalışmasında kullanılan IFS sezgiseli Müller (2009) çalışmasında olduğu gibi, atanma zorluğuna göre seçilen dersin atanacağı pozisyonu, uygun pozisyonlar arasından rastgele seçerek belirlemektedir. Biz bu çalışmada ayrıca, IFS sezgiselini, uygun pozisyonu rastgele seçmek yerine esnek kısıtları ihlal etme açısından en uygun olan pozisyonu seçecek şekilde değiştirerek alternatif bir sezgisel (*IFS_goodPos*) daha kullanmaktayız. Daha açık bir ifadeyle *IFS_goodPos* sezgiseli şu şekilde çalışmaktadır: Öncelikle atanacak ders IFS sezgiselindeki gibi seçilmektedir. Devamında bu ders için uygun pozisyonlar belirlenmektedir. Son aşamada, seçilen dersin atanması durumunda oluşturacağı ek maliyet her bir uygun pozisyon için hesaplanıp, maliyeti en az olan pozisyonlardan biri seçilmektedir. Pozisyonları öncelik sırlamasına koyan algoritmanın sözde kodu yukarıda bahsettiğimiz GH algoritmasında kullanılan fonksiyonla aynıdır. Bu fonksiyonun sözde kodu Tablo 4'te gösterilmektedir.

IFS sezgiselinin *IFS_goodPos* sezgiselinden farkı atanacak ders için pozisyon seçimi uygun pozisyonlar arasından rastgele yapılmasıdır. Tablo 8'de örnek olarak *IFS_goodPos* sezgiselinin değer seçimi algoritmasının sözde kodu verilmektedir. Bu algoritmada seçilen ders için bir pozisyon belirlenirken, bu dersin daha önceki çakışma sayılarına bakılarak çakışma sayısı en az olan pozisyon veya pozisyonlar belirlenir. En uygun pozisyon bu

pozisyon arasından seçilir. Yani öncelikle ders için uygun pozisyonların her birinin çakışma sayılarına bakılarak en iyi pozisyonlar belirlenir (satır 3-10). Sonra, çakışma sayısı açısından en iyi olan pozisyonlardan esnek kısıtları ihlal etme açısından en iyi olan pozisyon seçilir (satır 11-12). Son aşamada seçilen ders belirlenen pozisyona atanır ve çakışmaya neden olan dersler çizelgeden çıkarılarak, çakışma sayıları 1 artırılır (satır 15). Döngü atanmayan ders kalmayana kadar devam eder.

Tablo 8. Değer Seçimi Sözde Kodu (*pseudo-code*)

```
Fonksiyon: selectValue(tt, A)
Girdiler: tt = timetabling (ders çizelgesi), global CBS
1:  bestValues = []   #en iyi pozisyonlar listesi
2:  bestNrConfs = 0  #güncel çakışma sayısı
3:  for a in availablePositionsList(A)      #A dersi için uygun pozisyonları döndür.
4:     nrConfs = 0
5:     N  = conflicts(tt, A, a)   # A dersinin a pozisyonuna atanması durumunda çakıştığı ders pozisyon çiftleri
6:     if not N = Ø:  # çakışma listesi boş küme değilse
7:        for  (B → b) in N.values:    # A →a ikilisinin çakışacağı ders pozisyon ikililerini döndür.
8:           nrConfs += 1 + CBS[A = a → B ≠ b] # aynı çakışmaların çakışma sayısını bir arttır.
6:     if  bestValues = []   or bestNrConfs  >  nrConfs:
7:           bestValues = [a]      # bestValues listesine a pozisyonunu ekle
8:           bestNrConfs = nrConfs
9:     else:
10:          → bestValues = bestValues ⋃ [a]   # bestValues listesindeki pozisyonlara a pozisyonunu ekle
11: posList = orderPositionsByPriority(m, bestValues)
12: a = posList[0]          # listesindeki pozisyonlardan en düşük maliyetli pozisyon seçilir
12: N  = conflicts(tt, A, a)
13: if not N = Ø:
14:    for  (B → b) in N.values:
15:       CBS[A = a → B ≠ b] += 1
16: return a
```

Çakışan atamalar: Değişkenler değerlere atanırken problemde var olan kısıtların dikkate alınması gerekir. IFS algoritması içinde kullandığımız çakışma algoritması çakışma sayılarını belirlerken sadece zorunlu kısıtları dikkate almaktadır. Bu algoritma bir "*A"* dersinin *"a*" pozisyonuna atanması durumunda oluşacak çakışmaların listesini üretmektedir. Tablo 9'da görüldüğü gibi örneğin; "*A*" dersinin öğretim elemanın atanacak zaman periyodunda hali hazırda bulunması, "*A*" dersini alan öğrenci grubunun başka bir dersinin hali hazırda atanacak zaman periyodunda bulunması ve "*a*" pozisyonunda hali hazırda başka bir dersin bulunması halinde bunlar birer çakışma olarak çakışma listesine eklenmektedir.

Tablo 9. Çakışan Atamalar Sözde Kodu (*pseudo-code*)

Fonksiyon: conflicts (tt, A, a)
Girdiler: tt = timetabling (ders çizelgesi), A (ders), a (pozisyon)
1: → N = []
2: → N[A, a] = N[A, a] ∪ timeslotHasSameTeacher(A, a) *# öğretim elmanı çakışmaları*
3: → N[A, a] = N[A, a] ∪ timeslotHasSameCurriculum (A, a) *# öğrenci grubuna ait başka derslerin olması*
3: → N[A, a] = N[A, a] ∪ timeslotNotEmpty (a) *# a pozisyonunda hali hazırda dersin olması*
4: **return** N *# çakışma listesi*

Başlangıç çözümü başlığı altında GH ve IFS algoritmalarının ders çizelgeleme problemimizde nasıl uygulanacağı aşamalar halinde ayrıntılı bir şekilde anlatılmıştır. Bir sonraki aşama çözümü iyileştirme aşamasıdır. Bu aşamada kullandığımız iki farklı yerel arama sezgiseline geçmeden önce bu iki sezgiselde de kullandığımız komşuluk yapılarından bahsetmek daha uygun olacaktır.

3.4.2 Komşuluk Yapısı (*Neighborhood Structure*)

Komşuluk hareketinin nasıl tanımlanacağı konusu yerel arama sezgisellerinin en önemli bileşenlerinden biri olduğu yaygın bir şekilde söylenmektedir (Lü ve Hao, 2010). Komşuluk sürecini çözümde zorunlu kısıtları ihlal etmeden yapılan bir değişim hareketi ile yeni bir komşu çözümün üretilmesi olarak tanımlanabilir.

Bu tez çalışmasında iki türde komşuluk hareketleri kullanılmaktadır. İlki standart komşuluk hareketi diyebileceğimiz pozisyon takası (pt) komşuluk hareketleridir. İkincisi zaman periyodu takası (zpt) ve tekli/çoklu kempe zinciri (kempe chain) komşuluk hareketleridir. İki farklı kısma ayırdığımız bu komşuluk hareketlerini sırasıyla K_1 ve K_2 ile ifade edebiliriz.

Eğitimde çizelgeleme çalışmalarının (Aladag vd., 2009; Bellio vd., 2016; Clark vd., 2008; Tarawneh vd., 2013) çoğunda standart komşuluk hareketleri kullanılmaktadır. Çalışmaların bazılarında (Burke vd., 2010; Lü ve Hao, 2010; Merlot vd., 2002; Tuga vd., 2007) ise bazı standart komşuluk hareketinin yanında kempe zinciri komşuluk hareketlerinin de kullanıldığı görülmektedir.

Bu tez çalışmasında kullandığımız komşuluk hareketleri zorunlu kısıtları ihlal etmeyecek şekilde uygulanmaktadır. Tablo 10'da, örnek olarak, iki dersin/pozisyonun takası (pt) komşuluk hareketinin sözde kodu (*pseudo-code*) gösterilmektedir. Bu komşuluk hareketinde pozisyonlar rastgele seçilmektedir. Bu pozisyonlardaki derslerin takası, zorunlu kısıtlar ihlal edilmemesi şartıyla gerçekleşebilmektedir. Aksi durumda takas algoritması ders takasını gerçekleştirmeden sonlanmakladır. Her iki pozisyonda da bir dersin olması şart değildir. Sadece pozisyonun birinde bir dersin olması takas işlemin denenmesi için yeterlidir. Fonksiyon çıktı olarak, takas gerçekleştirilebiliyorsa "*Ture*" ve pozisyon ders ikililerini, takas gerçekleştirilemiyorsa "False" değerlerini üretir. Diğer komşuluk hareketlerinin kodları da benzer süreçleri izlediği için örnek olarak sadece bu komşuluk hareketinin sözde kodu verilmektedir.

Tablo 10. Pozisyon Takası Hareketinin Sözde Kodu (*pseudo-code*)

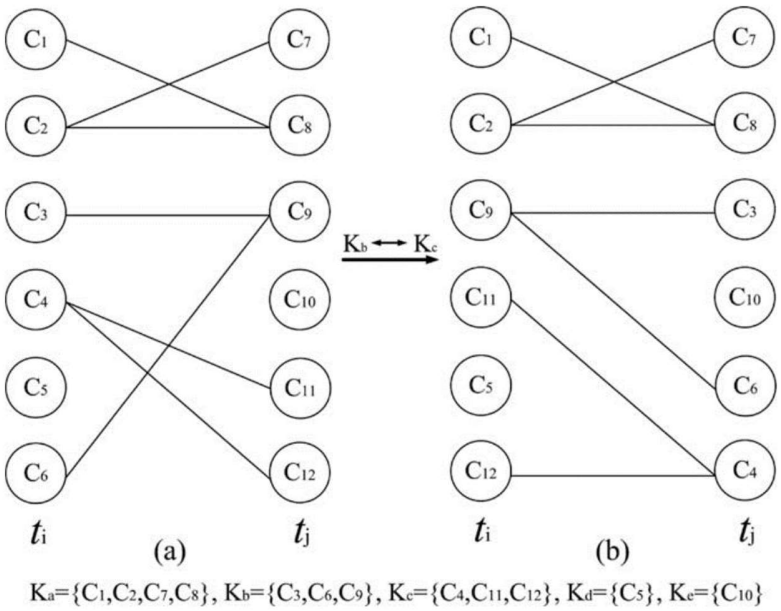
```
Fonksiyon: swapCourses(posizyon1, posizyon2)
Girdi: posizyon1, poziston2
1: course1 = pozisyon1  # her iki pozisyonu dolduran olayların yedeği oluşturulur:
2: course2 = pozisyon2
3: if (pozisyon1= None) and (pozisyon2= None):
        return False, (), ()
# course2 bir dersse ve bu ders pozisyon 1'e atanabiliyorsa fonksiyon devam eder.
#course1 bir dersse ve bu ders pozisyon 2'ye atanabiliyorsa fonksiyon devam eder.
4: if courseFitsIntoPosition (pozisyon1, course2) = False:
        return False, (), ()
5: if courseFitsIntoPosition (pozisyon2, course1) = False:
        return False
# pozisyon değerleri değiştirilir.
6: → pozisyon1 = course2
7: → pozisyon2 = course1
# fonksiyon "True " ve pozisyonların ilk değerlerini döndürür.
8: return True, (pozisyon1, course1), (pozisyon2, course2)
```

Kempe zinciri komşuluk hareketi diğer standart komşuluk hareketlerine göre daha karmaşık bir yapıya sahiptir. Kempe zinciri graf teoride kullanılan bir yaklaşımdır. Bu yaklaşım bazı düğümlerin (*nodes*) çeşitli özeliklerden dolayı bir birine yayalar (*arcs*) ile bağlanmasıyla oluşan alt kümeler olarak tanımlanabilir. Ders çizelgeleme problemine kempe zinciri yaklaşımı uyarlanırken, her bir dersin bir düğüm olduğu, aynı öğretime elemanı tarafından verilen veya ortak öğrencilere sahip derslerinde birbirine yaylar ile bağlı olduğu

düşünülmektedir. Daha açık bir ifadeyle, aynı zaman periyoduna atanamayacak dersler (düğümlerin) birbirine yaylar ile bağlanmaktadır.

Şekil 5'da bir kempe zinciri örneği gösterilmektedir. Bu örnekte t_i ve t_j zaman periyotlarına atanmış toplam 12 adet ders $(C_1, \dots, C_{12})$ 5 adet $(K_a, \dots, K_e)$ kempe zinciri oluşturmaktadır. $K_b = [C_3, C_6, C_9,]$ ve $K_c = [C_4, C_{11}, C_{12},]$ zincirlerindeki derslerin zaman periyotlarının karşılıklı olarak takas edilmesi durumunda şekil a şekil b'ye dönüşecektir.

Bu tez çalışmasında Lü ve Hao (2010) çalışmasındaki gibi iki kempe takası komşuluk hareketi kullanılmaktadır. İlki tek bir kempe zincirinden oluşan tekli kempe zinciri takası (tkz) komşuluk hareketidir. İkincisi ise iki kempe zincirinden oluşan çoklu kempe zinciri takası $(çkz)$ komşuluk hareketidir.



Şekil 5. Kempe Zinciri Gösterimi (Lü ve Hao, 2010)

Bu çalışmada geliştirdiğimiz tkz algoritması şu adımları takip etmektedir. Algoritma rastgele belirlenen iki farklı zaman periyodunun ilkinden rastgele bir ders seçmektedir. Daha sonra seçilen dersin kempe zinciri oluşturulmaktadır. Eğer bu kempe zincirindeki ders sayısı en az üçse, algoritma kempe zincirini oluşturan derslerin her biri için öğretim elemanının karşı zaman periyodunda mevcut olup olmamasını, öğrenci grubu için karşı zaman periyodunun

kapalı olup olmamasını ve karşı zaman periyodunda uygun kapasiteye sahip dersliklerin yeterli sayıda olup olmamasını kontrol etmektedir. Tüm bu şartlar olumluysa algoritma tkz komşuluk hareketini gerçekleştirmektedir. $çkz$ algoritması da tkz algoritması gibi çalışmaktadır. Bu algoritmanın tkz algoritmasından tek farkı seçilen zaman periyodundan iki farklı ders seçip iki kempe zinciri oluşturmak ve daha sonra bu iki zincirin takasını gerçekleştirmeye çalışmaktır. Özetle, iki farklı zaman periyoduna atanan derslerden kempe zinciri/zincirleri oluşturulmakta ve değişim zorunlu kısıtları ihlal etmiyorsa komşuluk hareketi gerçekleştirilmektedir.

Kempe zinciri takası komşuluk hareketlerinde ders derslik eşleştirilmesi ise karşı zaman periyoduna atanacak her bir ders için mevcut olan derslikler en az boş kapasiteden (öğrenci sayısı - derslik kapasitesi) en çok boş kapasiteye doğru sıralanmakta ve her ders kendi listesinin başındaki dersliğe atanacak şekilde yapılmaktadır.

Basit bir ders çizelgeleme örneği üzerinden grafikler yardımıyla komşuluk hareketlerini göstermek komşuluk hareketlerinin daha iyi anlaşılmasını sağlayacaktır. Bu nedenle aşağıda her bir komşuluk hareketi için bir örnek çözüm gösterilmektedir.

Bölüm 3.2'de verilen veri giriş formatı örneğindeki derslerden uygulanabilir bir başlangıç çözümünün Şekil 6'daki gibi oluştuğunu düşünelim… Bu derslerin özelikleri de veri giriş formatındaki örnekteki gibi olsun.

	Pazartesi		Salı	
Derslik/Periyot	1	2	3	4
A1	MAN106		MAN102(02)	MAN210
D3			MAN102(01)	
D6				
Lab-c		MAN307		

Şekil 6. Başlangıç Çözümü Örneği

Pozisyon takası (pt) komşuluk hareketi örneğinde algoritma tarafından rastgele (1, 1) – derslik A1 ve zaman periyodu 1- ve (1, 4) pozisyonlarının seçildiğini varsayalım. Algoritma sonraki aşamada (1, 1) pozisyonundaki dersin – MAN106 – (1, 4) pozisyonuna ve (1, 4)

pozisyonundaki dersin – MAN210 – (1, 1) pozisyonuna atanması zorunlu kısıtları ihlal etmiyorsa iki dersin yer değişimi yapılır. Yeni çizelge Şekil 7'deki gibi olur.

	Pazartesi		Salı	
Derslik/Periyot	1	2	3	4
A1	MAN210		MAN102(02)	MAN106
D3			MAN102(01)	
D6				
Lab-c		MAN307		

Şekil 7. Çizelgede Pozisyon Takası Komşuluk Hareketinin Örneği

Eğer algoritma (1, 1) ve (3,3) pozisyonlarını seçseydi (1,1) pozisyonundaki dersin (3, 3) pozisyonuna atanması hem D6 dersliğinin kapasitesinin yetersiz olması ve hem de 3 zaman periyodunda aynı müfredatın başka bir dersinin -MAN102(01)- mevcut olmasından dolayı dersin yer değişimi yapılamayacaktı.

Zaman periyodu (zpt) komşuluk hareketi örneğini düşünelim… Bu komşuluk hareketi algoritması için öncelikle iki zaman periyodu belirlenir. Bu zaman periyotlarının 2 ve 4 olduğunu varsayalım. Algoritma, 2 zaman periyodundaki derslerin 4 zaman periyoduna, 4 zaman periyodundaki derslerin de 2 zaman periyoduna atanmasının uygunluğunu teker teker kontrol eder. Öğretim elemanın diğer zaman periyodunda uygun olmaması veya diğer zaman periyodunun dersi alan öğrenci grubu için kapalı olması gibi durumlar söz konusu ise takas işlemi algoritma tarafından gerçekleştirilmez. Örneğimizde 2 ve 4 zaman periyodundaki derslerin karşılıklı takasları zorunlu kısıtları ihlal etmediği için takas gerçekleşmektedir. Şekil 8 zpt komşuluk hareketinden sonra oluşan yeni çizelgeyi göstermektedir.

	Pazartesi		Salı	
Derslik/Periyot	1	2	3	4
A1	MAN210	MAN106	MAN102(02)	
D3			MAN102(01)	
D6				
Lab-c				MAN307

Şekil 8. Çizelgede Zaman Periyodu Takası Komşuluk Hareketinin Örneği

Tekli kempe zinciri (tkz) komşuluk hareketi örneğini düşünelim… Rastgele seçilen iki zaman periyodunun sırasıyla 2 ve 3 olduğunu varsayalım. 2. zaman periyodunda seçilen (1, 2) pozisyonundaki dersin –MAN106- kempe zinciri (1, 2), (1,3) ve (2, 3) –MAN106, MAN102(02) ve MAN102(01)- pozisyonlarından oluşacaktır. Zincirdeki bu dersler aynı öğrenci grubun ait derslerdir. Bu nedenle sadece (1,2) pozisyonundaki dersi veren öğretim elemanın 3. zaman periyodunda mevcut olup olmadığı ve bu ders için uygun dersliğin olup olmadığı kontrol edilmektedir. Aynı şekilde (1, 3) ve (3, 3) pozisyonlarındaki dersleri veren öğretim elemanlarının 2. zaman periyodunda mevcut olup olmadığı ve bu dersler için uygun dersliklerin olup olmadığı kontrol edilmektedir. Zincirdeki bu derslerin karşı zaman periyoduna atanması herhangi bir zorunlu kısıtı ihlal etmediği için tkz komşuluk hareketi gerçekleşmekte ve yeni çözüm Şekil 9'daki gibi olmaktadır.

	Pazartesi		Salı	
Derslik/Periyot	1	2	3	4
A1	MAN210		MAN106	
D3		MAN102(02)		
D6		MAN102(01)		
Lab-c				MAN307

Şekil 9.Çizelgede Tekli Kempe Takası Komşuluk Hareketinin Örneği

3.4.3 Tabu Arama Sezgiseli İle Ders Çizelgeleme Probleminin Modellenmesi

Tabu arama (tabu search-TS) sezgiseli iyileştirme aşamasında kullandığımız sezgisellerden biridir. Bu sezgiselde önceki kısımda bahsedilen K_1 ve K_2 komşuluk hareketlerini kullanılarak çözüm iyileştirilmeye çalışılmaktadır. Bu komşuluk hareketleri kullanım kombinasyonu bir döngü ($K_1 \rightarrow K_2 \rightarrow K_1 \rightarrow K_2 \ldots$, gibi) şeklinde uygulanmaktadır. Daha açık bir ifadeyle, TS arama süreci bir komşuluk türüyle başlar. Arama süreci yerel optimuma ulaştığı zaman TS arama süreci durur ve diğer komşuluk hareketiyle tekrar başlar. Örnek olarak, bu komşuluk hareketlerinden pozisyon takasının (pt) gerçekleştirilmesini sağlayan komşuluk hareketinin sözde kodu aşağıda verilmektedir.

TS algoritmasını farklı kılan bileşeni tabu listeleridir. Tabu listesi ile (bak: Bölüm 2.2.2) gerçekleşen komşuluk hareketleri hafızaya alınarak geçici süreyle tekrarlanması yasaklanmaktadır. Bu şekilde yerel optimuma ulaşıldığında tekrar eden bir döngüye girilmemesi sağlanmaya çalışılmaktadır. Tabu listelerinin her komşuluk türü için ayrı ayrı oluşturulması önerilmektedir (Gendreau ve Potvin, 2010). Aşağıdaki algoritmada da görüldüğü gibi, bizde bu çalışmada her bir komşuluk türü için ayrı bir tabu listesi oluşturmaktayız. Komşuluk hareketlerinin tabu listelerine kaydedilme biçimleri içinde farklı yaklaşımlar söz konusudur. En iyi bilinen iki yaklaşımdan birincisi, aşağıdaki algoritmada (Tablo 11- satır 5) da görüldüğü gibi, sadece değişen hareketlerin ((pos1, pos2), (pos2, pos1)) tabu listesine kaydedilmesidir. İkincisi Lü ve Hao (2010) çalışmasında kullandığı gibi her olay (ders) için ayrı bir tabu listesinin belirlenmesidir. Güçlü tabu (strong tabu) olarak adlandırılan bu yaklaşımda bir ders bir pozisyondan diğer bir pozisyona atandıysa, dersin önceki pozisyonu bu dersin tabu listesine alınır. Böylece bu dersin tekrardan eski pozisyonuna atanarak bir döngüye girmesinin önüne geçilebilmektedir. Bizim çalışmamıza, komşuluk yapıları açısından daha uygun olan ilk yaklaşım tercih edilmektedir. Tabu liste uzunluğunun belirlenmesinde çözüm sürecine doğrudan etki etmektedir. Bazı çalışmalarda sabit liste uzunluğu bazı çalışmamalarda ise değişken liste uzunlukları önerilmektedir. Bu bileşen problemin yapısıyla doğrudan ilgili bir durumdur. Bu çalışmada sabit uzunluklu tabu listeleri kullanılmakta ve farklı uzunluk parametreleri belirlenerek performans karşılaştırması yapılmaktadır.

Tablo 11. Tabu Aramada Pozisyon Takası Sözde Kodu (*pseudo-code*)

```
Fonksiyon: swapPositions (TabuListPos)
Girdi: TabuListPos            # pozisyon takası için tabu listesi
       global:  best_tt, best_cost   # şimdiye kadar ulaşılan en iyi ders çizelgesi ve en düşük maliyet
1:   pos1, pos2 = randomChose2positions()   # rastgele iki farklı pozisyon seçilir.
2:   if (pos1, pos2) in TabuListPos  or (pos2, pos1) in TabuListPos :
3:         return False
4:   else:
5:         (pos1, pos2) ve (pos2, pos1) pozisyon ikililerini TabuListPos'a ekle
6:   successful, (pos1, course1), (po2, course2) = swapCourses(pos1, pos2)  # takasın gerçekleştirilmesi
7:   if not successful:          # takas olabilir değilse fonksiyon "False" üretir
8:         return False
9:   total_cost = totalCostTimetable()     # yeni maliyet skoru hesaplanır.
10: delta_e = total_cost - best_cost
11: if delta_e  >=  0:
12:       takası tersine çevir
13:       return False
14: best_cost = total_cost
15: best_tt = timtable                      # ders çizelgesini yenile
16: return True
```

Tablo 11'de verilen pozisyon takası komşuluk hareketi fonksiyonu pozisyonları takas etmeye çalışmakta, takasın başarılı/başarısız olması durumunda da "*True*"/"*False*" değerlerini üretmektedir. Bu algoritmanın izlediği süreç şu şekilde açıklanabilir:

- İlk olarak bu algoritma rastgele iki pozisyon seçer (satır 1). Daha sonra, seçilen zaman periyotlarının tabu listesinde olup olmadığını kontrol eder (satır 2). Eğer bu pozisyonlar tabu listesinde varsa, komşuluk hareketi bu noktada sonlanır (satır 3). Aksi durumda ise bu pozisyonlar tabu listesine eklenerek süreç devam eder (satır 5).

- Pozisyon çifti tabu listesinde yok ise algoritma pozisyon takasını gerçekleştirilmeye çalışır (satır 6). Bu takas Tablo 10'da verilen fonksiyon ile yapılmaktadır. Eğer takas gerçekleştirilemiyorsa, algoritma süreci sonlandırır (satır 8). Takas gerçekleştirilebiliyorsa, algoritma maliyet hesaplama aşamasına geçer.

- Toplam maliyeti hesaplama aşamasında (satır 9), algoritma maliyet hesaplama fonksiyonu ile toplam maliyeti hesaplar. Toplam maliyet bütün esnek kısıtların (bak:

Bölüm 33.1.2, - E1, … , E4) maliyetinin komşuluk hareketinden sonra oluşan yeni çözüm için ayrı ayrı hesaplanarak toplanmasından oluşmaktadır.

- Toplam maliyet hesaplandıktan sonra, algoritma şimdiye kadarki en iyi maliyet skoru ile mevcut çözümün toplam maliyet skoru arasındaki farkı hesaplar (satır 10). Eğer yeni çözümün toplam maliyeti ile en iyi maliyet skoru arasındaki fark sıfırdan büyük eşitse, algoritma takası tersine çevirerek (satır 12) süreci sonlandırır (satır 13). Bu fark sıfırdan küçükse, algoritma en iyi maliyet skorunu yeni çözümün toplam maliyetine (satır 14) ve şimdiye kadarki en iyi ders çizelgesi çözümünü de yeni çözümle değiştirir.

Pozisyon takası komşuluk hareketinin dışında, TS sezgiseli zaman periyodu takası (zpt), tekli kempe zinciri (tkz) ve çoklu kempe zinciri ($çkz$) komşuluk hareketlerini de kullanmaktadır. Ancak komşuluk hareketlerinin genel yapıları birbirine çok benzediği için bu komşuluk hareketlerinin sözde kodlarını da vererek tezi gereğinden fazla uzatmak istenmemiştir.

Komşuluk hareketlerinin tümü üst bir algoritma olan TS algoritması ile yönetilmektedir. Bu algoritmanın sözde kodu Tablo 12'de verilmektedir. Maliyet skoru sıfırdan büyük oldukça ve belirlenen maksimum çözüm süresi dolana kadar çalışan (satır 1) bu algoritma, K_1 ve K_2 komşuluk hareketlerini bir döngü şeklinde ($K_1 \rightarrow K_2 \rightarrow K_1 \rightarrow K_2$ … , gibi) kullanmaktadır. Bu süreci belli sayıdaki tekrara rağmen toplam maliyette bir iyileşme olmaması durumunda, bir diğer komşuluk türüne atlayarak sürdürmektedir (satır 3, 12). Komşuluk hareketleri denenmeden önce eğer listedeki komşuluk hareketi sayısı belirlenen liste uzunluğunu aşıyorsa (satır 4, 13), algoritma daha önce eklenen komşuluk hareketlerini listeden çıkararak liste uzunluğunu korur (satır 5, 14). K_2 komşuluk türünde birden fazla komşuluk hareketi olduğundan dolayı, algoritma bu komşuluk hareketleri arasındaki seçimi rastgele yapmaktadır (satır 15). Son olarak, algoritma sonlanırken şimdiye kadarki ulaşılan en iyi maliyet skorunu üretmektedir (satır 26).

Tablo 12. Tabu Arama Sözde Kodu (*pseudo-code*)

```
Fonksiyon: tabuSearch(tabu_length_K1, tabu_length_K2)
Girdi: tabu_length_K1 ve tabu_length_K2 tabu liste uzunlukları
     : Liste_K1 = [ ], Liste_K2 = [ ]
1:  while best_cost > 0 and currentTime – startTime < maxTime:
2:      no_improvement = 0
3:      while no_improvement < 10:
4:          while len(Liste_K1) > tabu_length_K1:
5:              Liste_K1 tabu listesine ilk eklenen komşuluk hareketini listeden çıkar
6:          change = swapPositions(Liste_K1)
7:          if not change:
8:              no_improvement += 1
9:          else:
10:             no_improvement = 0
11:     no_improvement = 0
12:     while no_improvement < 10:
13:         while len(Liste_2) > tabu_length_K2:
14:             Liste_K2 tabu listesine ilk eklenen komşuluk hareketini listeden çıkar
15:         x = random.choice([1, 2, 3])
16:         if x = 1:
17:             change = swapTimeslots (Liste_K2)
18:         elif x = 2:
19:             change = swapKempeCahin(Liste_K2)
20:         else:
21:             change = swapKempeCahins(Liste_K2)
22:         if not change:
23:             no_improvement += 1
24:         else:
25:             no_improvement = 0
26: return best_cost
```

3.4.4 Benzetim Tavlaması Sezgiseli İle Ders Çizelgeleme Probleminin Çözümü

Benzetim tavlaması (*simulated annealing*-SA) sezgiseli iyileştirme aşamasında kullandığımız ikinci sezgiseldir. Bu sezgiselde de TS sezgiselindeki gibi K_1 ve K_2 komşuluk hareketleri kullanılarak çözüm iyileştirilmeye çalışılmaktadır. Komşuluk hareketi sürecinin SA sezgiseli için nasıl bir süreç izlediğini daha net görebilmek için örnek olarak, komşuluk hareketlerinden bir olan pozisyon takası (pt) komşuluk hareketinin sözde kodu Tablo 13'de verilmektedir:

Tablo 13. Benzetim Tavlaması Pozisyon Takası Sözde Kodu (*pseudo-code*)

```
Fonksiyon: swapPositions (T)
Girdi: T, best_tt, last_cost, best_cost   #sıcaklık derecesi, şimdiye kadar ulaşılan en iyi ders çizelgesi, en düşük ve en son maliyet skoru
1:  pos1, pos2 = randomChose2positions()   # rastgele iki farklı pozisyon seçilir.
2:  successful, (pos1, course1), (po2, course2) = swapCourses(pos1, pos2)  # takasın gerçekleştirilmesi
3:  if not successful:          # takas olabilir değilse fonksiyon "False" üretir
4:      return False
5:  total_cost = totalCostTimetable()     # yeni maliyet hesaplanır.
6:  delta_e = total_cost - last_cost
7:  p = random.random()              # her defasında 0-1 arasından rastgele bir p olasılığı seçilmektedir.
8:  if delta_e >= 0 and p > e^(-delta_e/T): # maliyet farkı sıfırdan büyük ve 0-1 p olasılığı eşitliğin diğer tarafında hesaplanan değerden büyükse
9:          takası tersine çevir
10:         return False
11: last_cost = total_cost
12: impro = False
13: if total_cost < best_cost:
14:     best_tt = timtable              # ders çizelgesini yenile
15:     best_cost = total_cost
16:     impro = True          # daha iyi bir çözüm elde edildiyse "impro" değişkenini True yap. Bu değişken Tablo 14 algoritmasında kullanılacaktır.
17: return True
```

TS ile SA sezgiseli arasındaki en önemli fark; TS sezgiseli kötü çözümleri yeni çözüm olarak kabul etmezken, SA sezgiselinde ise belli bir olasılıkla kötü çözümlerde kabul edilebilmektedir. Bir başka ifadeyle, TS sezgiselinde yerel optimumdan çıkabilmek için tabu listeleri kullanılırken, SA sezgiselinde ise bazen kötü çözümleri kabul ederek yerel optimumdan kaçabilmek amaçlanmaktadır. SA sezgiseli ile ilgili daha ayrıntılı bilgi için Bölüm 2.2.3'e bakılabilir.

Tablo 13'de verilen pozisyon takası komşuluk hareketi, Tablo 11'de TS algoritması için verilen fonksiyon gibi pozisyonları takas etmeye çalışmakta, takasın başarılı/başarısız olması durumunda da "*True*"/"*False*" değerlerini üretmektedir. Bu algoritmanın izlediği süreç şu şekilde açıklanabilir:

- İlk olarak bu algoritma rastgele iki pozisyon seçer (satır 1). Daha sonra algoritma pozisyon takasını gerçekleştirilmeye çalışır (satır 2). Bu takas Tablo 10'da verilen fonksiyon ile yapılmaktadır. Eğer takas gerçekleştirilemiyorsa, algoritma süreci

sonlandırır (satır 4). Takas gerçekleştirilebiliyorsa, algoritma maliyet hesaplama aşamasına geçer.

- Toplam maliyeti hesaplama aşamasında (satır 5), algoritma maliyet hesaplama fonksiyonu ile toplam maliyeti hesaplar. Toplam maliyet bütün esnek kısıtların (bak: Başlık 3.1.2, - E1, … , E4) maliyetinin komşuluk hareketinden sonra oluşan yeni çözüm için ayrı ayrı hesaplanarak toplanmasından oluşmaktadır.

- Toplam maliyet hesaplandıktan sonra, algoritma bundan önce kabul edilen çözümün maliyet skoru ile mevcut çözümün toplam maliyet arasındaki farkı hesaplar (satır 6). Eğer bu fark sıfırdan ve 0-1 arasından rastgele seçilen p (satır 7) olasılık değeri eşitliğin diğer tarafında hesaplanan değerden ($e^{-\mathrm{del}\ -\mathrm{e}/T}$) büyükse (satır 8), algoritma takası tersine çevirerek (satır 9) süreci sonlandırır (satır 10). Aksi durumda, algoritma bir önceki çözümün maliyet skorunu yeni çözümün toplam maliyetine (satır 11) eşitler. Sonraki adımda, eğer toplam maliyet skoru şimdiye kadarki en iyi maliyet skorundan küçükse (satır 13), algoritma mevcut çözümü en iyi çözüm olarak kaydederken toplam maliye skorunu da en iyi maliyet skoru olarak kaydeder.

SA sezgiselinde de komşuluk hareketlerinin tümü üst bir algoritma ile yönetilmektedir. Bu algoritmanın sözde kodu Tablo 14'de verilmektedir:

Tablo 14. Benzetim Tavlaması Sözde Kodu (*pseudo-code*)

Fonksiyon: simulatedAnneling(Tmax, Tmin, β, yk, $iter_1$, $iter_2$)
Girdi: Tmax, Tmin, β, sk, yk, $iter_1$, $iter_2$

```
1:  no_improvement = 0
2:  no_improvement1 = 0
3:  T = Tmax
4:  while best_cost > 0 and currentTime – startTime < maxTime:
5:      iterations = 0
6:      while iterations < iter1:
7:          if T < Tmin:
8:              T = Tmin
9:          elif T > Tmax:
10:             T = Tmax
11:         iterations += 1
12:         no_improvement = 0
13:         while no_improvement < 10:
```

```
14:             change = swapPositions(T)
15:             if not change:
16:                   no_improvement  += 1
17:                   no_improvement1  += 1
18:             else:
19:                  if impro = False:
20:                       no_improvement  +=  1
21:                       no_improvement1 += 1
22:                  else:
23:                       no_improvement  =  0
24:                       no_improvement1 = 0
25:          no_improvement = 0
26:          while  no_improvement  <  10:
27:                iterations += 1
28:                x = random.choice([1, 2, 3])
29:                if x = 1:
30:                      change = swapTimeslots (T)
31:                elif x = 2:
32:                      change = swapKempeCahin(T)
33:                else:
34:                      change = swapKempeCahins(T)
35:                if not change:
36:                      no_improvement  += 1
37:                      no_improvement1  += 1
38:                else:
39:                     if impro = False:
40:                          no_improvement  +=  1
41:                          no_improvement1 += 1
42:                     else:
43:                          no_improvement  =  0
44:                          no_improvement1 = 0
45:          if no_improvement1 > iter_2:
46:                    iter_2 = α*iter_2
47:                    T = T* β^(−1.7*yk)
48:     T = β *T
49: return best_cost
```

SA sezgiselinde soğutma planı çözüm sürecini etkileyen önemli bir bileşendir. Soğutma planı T sıcaklık derecesinin alacağı değer ve bu değerdeki değişim ile ilgili bir durumdur. T değeri ne kadar büyük olursa kötü sonuçların kabul edilme olasılığı, bir başka deyişle arama uzayında gezinme genişliği o kadar büyük olmaktadır. T değeri sıfıra yaklaştıkça, algoritma tepe tırmanma (HC) algoritmasına benzemektedir. Yani kötü sonuçların kabul edilme

olasılığı giderek düşmekte süreç kararlı hale gelmektedir. İyi bir başlangıç değeri (satır 3) için T'nin ne olacağı problem yapısına göre değişkenlik göstermektedir. Soğutma palanı çeşitli şekillerde tasarlanabilmektedir. Soğutma planı stratejileri ile ilgili olarak Nourani ve Andresen (1998) çalışmasına bakılabilir. Genelde kullanılan yaklaşım olan geometrik yaklaşımdır Bu yaklaşım $T_{i+1} = T_i * \beta$ denklemiyle ifade edilir (bak: Bölüm 2.2.3). Bu çalışmada kullanılan soğutma stratejisi Müller (2009) çalışmasında kullandığı soğutma planından da esinlenilerek şu şekilde oluşturulmuştur:

- $Tmax$ maksimum sıcaklık değeridir. Bu değer aynı zamanda başlangıç sıcaklığı olarak da kullanılır. $Tmin$ ise minimum sıcaklık değeridir. Sıcaklık değerinin bu sıcaklıktan aşağıya düşmesi engellenmektedir.

- $T_{i+1} = T_i.\beta$ soğutma denklemi (satır 48) ile her bir $iter\_1$ ($iter_1 = \frac{TL}{sk}$) iterasyon sayısından (satır 6) sonra sıcaklık düşmektedir. Sıcaklık düşüş oranını β sabit katsayısı belirlemektedir. sk soğutma katsayısıdır, TL de problem değişken boyutuna göre hesaplanan değerdir. Bu çalışmada bu değer ders sayısı olarak alınmaktadır.

- Eğer $iter\_2$ iterasyon sayısına (satır 45) kadar en iyi çözüm iyileştirilemezse $T_{i+1} = T_i.\beta^{-4.yk}$ denklemi (satır 47) ile sıcaklık derecesi arttırılır. $iter\_2$ iterasyon sayısı da her defasında $iter\_2_{i+1} = iter\_2_i.\alpha$ denklemi ile artırılmaktadır (satır 46). yk sıcaklığı yeniden artırma katsayısını, α (1.4) ise yeniden ısıtma iterasyon sayısı artış katsayısını göstermektedir.

Özetle, SA algoritması (TS algoritması gibi) K_1 ve K_2 komşuluk yapılarını kullanarak çözümü iyileştirmeye çalışmaktadır. Algoritma bir başlangıç T sıcaklığı ile çözüme başladıktan sonra iterasyon sayısı $iter\_1$ sayısına ulaştıkça (satır 6) hem T sıcaklığı düşürülmekte (satır 48) ve hem de iterasyon sayısı sıfırlanmaktadır (satır 5). T sıcaklığı $Tmin$ sıcaklık değerinin altına düştükçe $Tmin$ sıcaklık değerine tekrardan yükseltilmektedir. T sıcaklığı $Tmax$ sıcaklık değerinin üstüne çıkınca da $Tmax$ sıcaklık değerine tekrardan düşürülmektedir (satır 7-10). Ayrıca, bulunan en iyi çözüm $iter\_2$ sayısı kadar bir denemeye rağmen hala iyileştirilemediyse hem sıcaklık ve hem de $iter\_2$ tekrardan artırılmaktadır (satır

46-47). Bu sayede, maliyet skoru düştükçe algoritmanın daha ısrarlı bir çaba içerisine girmesi sağlanmaktadır. Son olarak, algoritma sonlanırken şimdiye kadar ulaşılan en iyi maliyet skorunu üretmektedir.(satır 49).

4. BÖLÜM
MODELLERİN DEĞERLENDİRİLMESİ

Bu bölüm aşağıda sıralanan sorulara cevap verebilmek için oluşturulmuştur:

- Hacettepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi (HÜİİBF) ders çizelgesinin bölüm/fakülte bazında problem boyutu ve istatistiki değerleri nedir?
- HÜİİBF ders çizelgeleme probleminden türetilen diğer problemler ve istatistiki değerleri nedir?
- Başlangıç çözümü için önerilen yaklaşımlar nasıl bir performans sergilemektedirler?
- Benzetim tavlaması (SA) ve tabu arama (TS) algoritmaları için uygun parametreler nelerdir?
- Uygun komşuluk kombinasyonu nedir?
- HÜİİBF ders çizelgeleme problemimiz için SA ve TS sezgisellerinden hangisi daha iyi performans göstermektedir?
- HÜİİBF ders çizelgeleme problemin optimal çözümü nedir?
- Problem kompleksliliğinin artırılmasının problem çözümüne etkisi nedir?

4.1 PROBLEM VERİSİ

HÜİİBF bünyesinde olan bölümlerin aynı bina içerisinde yer alması, bu nedenle çoğu dersliğin ortak kullanımda olması ve bazı öğretim elemanlarını birkaç farklı bölüme ders vermeleri, bölümlerin ortak derslerinin olması gibi nedenler bu problemin fakülte bazında tek bir problem gibi düşünülmesini gerekli kılmaktadır. Bu problem ile ilgili istatistiki bilgiler aşağıdaki tablolarda özetlenmektedir.

HÜİİBF 9 farklı bölümden oluşmaktadır. Tablo 15'de bölüm kodları, isimleri, bölümlerin ders sayıları ve öğretim elemanı sayıları gösterilmektedir. HÜİİBF bahar döneminde 164

zorunlu (72 ders şubeli zorunlu) ve 141 (4 ders seçmeli şubeli) seçmeli statüsünde olan toplam 305 dersi vardır. Bu ders sayılarına hemen hemen tüm bölümler için ortak olan ve ders çizelgesinde atandığı yeri önceden belli olan dersler (İngilizce, Türk Dili ve İktisada Giriş gibi dersler) dâhil değildir. Bazı öğretim elemanları birkaç bölüme ders vermektedirler. Bu nedenle bölümlerin toplam öğretim elemanı sayısı 202 olsa da, aslında 184 farklı öğretim elemanı vardır. Bu fakültenin ortak kullanımında 2 tanesi laboratuvar olmak üzere toplam 35 adet derslik vardır. Bunun haricinde var olan bazı dersliklerde sadece bazı bölümlere veya bazı derslerin kullanımında olabilmektedir. Ayrıca, fakülte dışından bazı dersliklerde ihtiyaç durumunda bazı dersler için kullanılabilmektedir.

Tablo 15. HÜİİBF Ders ve Öğretim Elemanı Sayıları

			Ders Sayıları				
Kod	Bölüm İsmi	ÖES	Z	S	ZŞ	SŞ	Toplam
MAN	İşletme	29	21	27	16		48
ECO	İktisat (İngilizce)	19	17	10	4		27
EKO	İktisat (Türkçe)	23	25	11	16		36
KAY	Kamu Yönetimi ve Siyaset Bilimi	29	9	25			34
MAB	Maliye	25	17	20	7	2	37
SAİ	Sağlık Yönetimi Bölümü	18	13	11			24
INR	Uluslararası İlişkiler Bölümü	16	16	10			26
ATB	Aile ve Tüketici Bilimleri	21	17	11		2	28
SHY	Sosyal Hizmet Bölümü	22	29	16	29		45
Toplam		202	164	141	72	4	305

ÖES: Öğretim Elemanı Sayısı, Z: Zorunlu, S: Seçmeli, ŞZ: Şubeli zorunlu, ŞS: Şubeli Seçmeli

Bonutti vd. (2012) çalışmasında ITC-2007[5] problemlerinin temel özelliklerini daha net görmek ve problemleri kendi aralarında kıyaslayabilmek için bazı istatistiksel göstergeler kullanmıştır. Bu çalışmada kendi problem yapımıza uyarladığımız bu istatistiki göstergeler şöyle açıklanabilir:

[5] ITC-2007: 2007 yılında 3 kategoride (okul, üniversite sınav ve üniversite ders çizelgeleme) başlatılan uluslararası çizelgeleme yarışması (*international timetabling competition*). http://www.cs.qub.ac.uk/itc2007/

- Çakışma yoğunluğu (ÇY) oranı, aynı zaman periyoduna atanamayan (aynı öğrenci grubuna ait olan zorunlu dersler veya aynı öğretim elmanı tarafından verilen dersler) ders çiftlerinin toplam ders çiftleri sayısına bölünmesiyle elde edilmektedir.

- Öğretim elemanı bulunabilirliği (ÖB) farklı ders/zaman periyodu toplam sayısına oranlanarak hesaplanan ders başına öğretim elmanı bulunabilirlik oranıdır. Bir başka ifadeyle zaman periyotlarında bulunan öğretim elemanları ders/zaman periyodu sayısının toplam ders/zaman periyodu sayısına oranın hesaplanmasıdır.

- Pozisyon uygunluğu (PU) oranı birbiri için uygun olan ders/pozisyon sayısını toplam ders/pozisyon sayısına oranlayarak bulunmaktadır. Önceden de bahsedildiği gibi pozisyonlar derslerin atandığı derslik-zaman periyodu ikilileridir. Pozisyon uygunluğu için hem dersin zaman periyoduna atanabilir olmasına ve hem de dersliğin mevcut ve kapasitesinin yeterli olmasına bakılmaktadır.

- Derslik işgali (Dİ) oranı toplam ders sayısının toplam pozisyon sayısına bölünmesiyle elde edilmektedir.

- Müfredat (öğrenci grubu) başına düşen günlük ders (GD) oranı her bir öğrenci grubu için hesaplanan günlük ders oranıdır. Bu oran öğrenci gruplarının gün başına düşen ders sayılarının toplamını öğrenci grubu sayısına bölünmesiyle elde edilmektedir.

Tablo 16'da ders çizelgeleme problemlerinin yukarıda tanımladığımız istatistiki göstergeleri bölüm/fakülte bazında gösterilmektedir. Bu tez çalışmasında bilindiği gibi ders çizelgeleme problemi fakülte bazında bir bütün olarak ele alınmaktadır. Tablodaki HÜİİBF istatistiki değerleri bu problemin istatistiki değerlerini göstermektedir. Tabloda ayrıca bölüm bazında istatistiki değerleri vermemizin ana nedeni her bir bölümün ders çizelgeleme problemine katkısını daha net görebilmektedir. Örneğin çakışma yoğunluğu (ÇY) değeri en yüksek olan SHO bölümünün ders çizelgeleme probleminin çözümünü zorlaştıran bölümlerden olduğunu söylenebilir. Benzer yorum diğer istatistiki değerler içinde yapılabilir. Bölümler birbirinden bağımsız düşünülüp PU ve Dİ istatistiki değerleri hesaplanırken her bölüm için tüm derslikler hesaba katılmaktadır. Bu şekilde derslik işgali (Dİ) istatistiki değeri ile bölümlerin derslik

kullanıp yoğunlukları gözlenebilmektedir. Örneğin MAN bölümünü diğerlerine oranla daha fazla dersliğe ihtiyaç duyduğu görülmektedir. Aynı şekilde MAN bölümü gün başına düşen ders oranı da (GD) diğer bölümlere göre fazladır. Bu değerler bölümlerin problemi zorlaştırmada katkılarının ne olduğunu görebilme açısından önemlidir.

Her bölümün ders çizelgeleme problemi ayrı ayrı çözümlense bile bütün olarak uygulanabilir bir çözüm olmayacaktır. Çünkü öğretim elemanları, derslikler veya bazı derslerin birçok bölüm için ortak olması gibi nedenlerden dolayı çakışmaların olması olasıdır. Bu nedenler, bu problemin fakülte bazında çözülmesi için önemli gerekçelerdir. ÇY değerinin fakülte bazında oldukça düşük çıkmasının nedeni aynı zaman periyoduna atanacak iki farklı dersin çakışma ihtimalinin, birbiriyle ilişkisi az olan bölümlerin sayısı arttıkça azalmasıdır.

Tablo 16. Bölüm/Fakülte Bazında İstatiksel Göstergeler

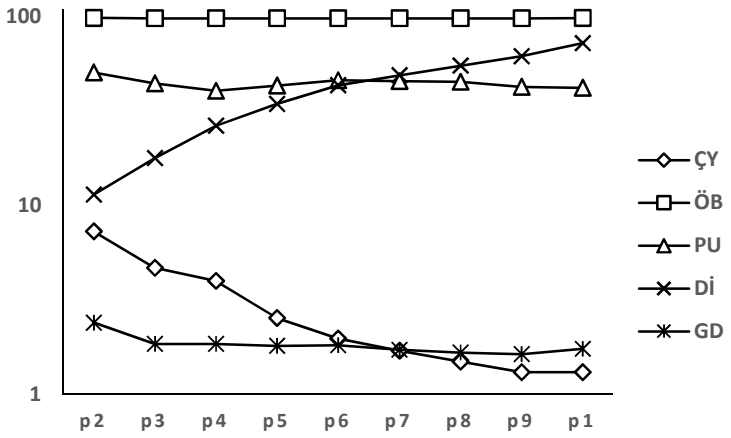
	Bölümler	ÇY	ÖB	PU	Dİ	GD
	MAN	7,3	98,75	50,67	**11,43**	**2,4**
	ECO	13,4	98,51	30,54	5,63	1,35
	EKO	14,6	99,44	29,75	7,5	1,80
	KAY	3,74	99,11	49,21	7,08	1,7
	MAB	7,66	98,37	53,62	7,08	1,85
	SAİ	9,42	98,33	38,94	5,0	1,2
	INR	12,62	97,70	40,29	5,42	1,3
	ATB	12,61	98,84	43,90	5,42	1,3
Fakülte	SHO	**16,46**	99,1	43,17	9,38	3
HÜİİBF		1,31	98,49	42,03	72,62	1,74

Bu çalışmada asıl problemimiz bir gerçek hayat problemi olan HÜİİBF ders çizelgeleme problemidir (p1). Ancak modellerimizin performanslarını daha iyi gözlemleyebilmek için bir bölümden başlayıp her defasında bir başka bolüm daha ekleyerek 8 farklı problem (p2 - p9) daha türetilmektedir. Tablo 17'de bu problemleri oluşturulan bölümler ve istatistiki değerler verilmektedir. Bu istatistiki değerlerin problemlere göre seyri Şekil 10'de daha net bir şekilde gösterilmektedir. Örneğin derslik işgali (Dİ) oranları problem boyutu arttıkça artmaktadır. Benzer şekilde çakışma yoğunluğu (ÇY) oranları problem boyutu arttıkça azalmaktadır. GD, PU ve ÖB oranları da aşağı yukarı aynı değerlerdedir. Aslında Tablo 16'daki istatistiki

değerler bakılarak da türetilen bu problemlerle ilgili çıkarımlarda bulunulabilir. Örneğin SHO, MAN ve ECO bölümlerinin diğerlerine göre problemi zorlaştırma etkileri daha fazladır. Yani türetilen bu problemler farklı bölüm bileşimlerine göre türetilirse problemlerin zorlukları da farklı olur.

Tablo 17. Problemlerin İstatistiki Göstergeleri

Problemler	Bölümler	ÇY	ÖB	PU	Dİ	GD
p2	MAN	7,3	98,75	50,67	11,42	2,4
p3	MAN+ECO	4,68	98,13	44,47	17,85	1,85
p4	MAN+ECO+EKO	3,99	98,28	40,56	26,43	1,85
p5	MAN+ECO+EKO+KAY	2,54	98,48	43,3	34,52	1,81
p6	MAN+ECO+EKO+KAY+MAB	1,97	98,46	46,21	43,33	1,82
p7	MAN+ECO+EKO+KAY+MAB+SAİ	1,7	98,44	45,64	49,05	1,72
p8	MAN+ECO+EKO+KAY+MAB+SAİ+INR	1,49	98,36	45,3	55,24	1,66
p9	MAN+ECO+EKO+KAY+MAB+SAİ+INR+ATB	1,31	98,38	42,64	61,9	1,63
p1	MAN+ECO+EKO+KAY+MAB+SAİ+INR+ATB+SHO	1,31	98,49	42,03	72,61	1,74



Şekil 10. Problem Verilerinin İstatistiki Göstergeleri

4.2 BAŞLANGIÇ ÇÖZÜMÜ SEZGİSELLERİNİN KARŞILAŞTIRILMASI

Bu çalışmada, başlangıç çözümü için iki farklı sezgisel yaklaşım kullanılmaktadır. Bölüm 3.4.1'de ayrıntılı olarak bahsedilen bu sezgisel yaklaşımlardan ilki, atanması en zor olan dersi

en önce atamaya çalışan bir açgözlü (GH) algoritmadır. Diğeri ise hafızasında tuttuğu çakışma sayılarına göre dersleri atayan bir iteratif ileri arama (IFS) algoritmadır. Bu iki sezgisel algoritmanın pozisyon seçme prosedürleri değiştirilerek başlangıç çözümü için kullanılabilecek iki sezgisel algoritma daha çalışmaya dâhil edilmektedir. Bu sezgiseller *GH_goodPos* ve *IFS_ goodPos* (bak: Bölüm 3.4.1) olarak adlandırılmaktadır.

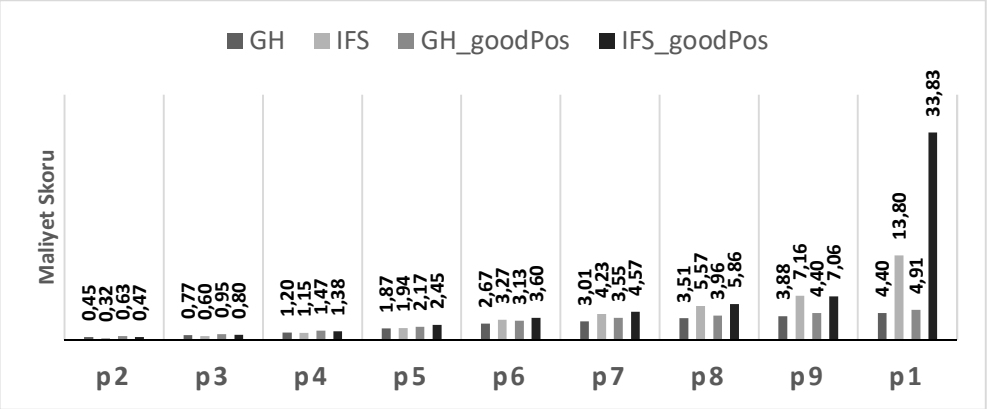
Başlangıç çözümü algoritmaları 10 problemin er biri için 10'ar kez çalıştırılarak ortalama maliyet skorları ve ortalama çözüm süreleri bulunmuştur. Bulunan bu değerler Tablo 18'de gösterilmektedir. Ayrıca, algoritmalar arasındaki farkın daha net görülebilmesi için tablodaki bu değerler ortalama maliyet ve ortalama çözüm süreleri açısından iki farklı grafikte (Şekil 11 ve Şekil 12) gösterilmektedir.

Tablo 18. Başlangıç Çözümü Sezgisellerinin Karşılaştırılması

	GH		IFS		GH_goodPos		IFS_goodPos	
Problemler	Maliyet	ÇS(sn)	Maliyet	ÇS(sn)	Maliyet	ÇS(sn)	Maliyet	ÇS(sn)
p2	12	0,45	14,6	0,32	9	0,63	8,3	0,47
p3	15	0,77	19,3	0,60	9	0,95	10,1	0,80
p4	23	1,20	25,5	1,15	11	1,47	10,8	1,38
p5	28	1,87	36	1,94	12	2,17	12	2,45
p6	43	2,67	49,9	3,27	20	3,13	19,9	3,60
p7	45	3,01	57,8	4,23	22	3,55	23,8	4,57
p8	60	3,51	67,1	5,57	21	3,96	25,5	5,86
p9	72	3,88	81,2	7,16	32	4,40	35,1	7,06
p1	69	4,40	85,2	13,80	31	4,91	34,5	33,83

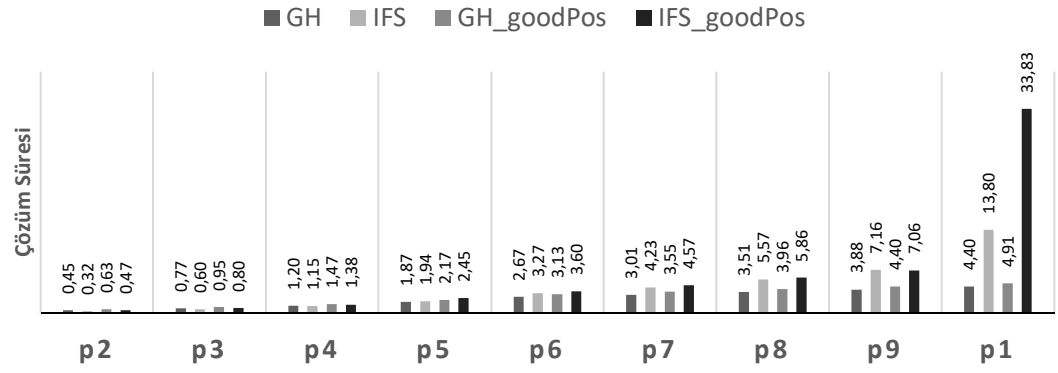
GH sezgisel yaklaşımı için nz (zorunlu dersler için), nzs (zorunlu şubeli dersler için) ve ns (seçmeli dersler için) öncelik katsayıları sırasıyla 3,3 ve 1 olarak alınmıştır (bak: Bölüm 3.4.1). ÇS: Çözüm süresi

Şekil 11'e bakıldığında, GH ve IFS algoritmalarının ortamla maliyet skorlarının çok farklı olmadığı görülmektedir. Ancak, en iyi pozisyonu seçme prosedürünün eklendiği *GH_goodPos* ve *IFS_goodPos* algoritmalarının çözümünde ortamla maliyet skorunun önemli düzeyde iyileştiği görülmektedir. Bu skorlarında her iki algoritma türü için benzer olduğu görülmektedir



Şekil 11. Başlangıç Çözümü Algoritmaları İçin Ortalama Maliyet Skorları

Şekil 12'ye bakıldığında iki algoritma türünün özelikle büyük boyutlu problemlerde çözüm süresi açısından önemli farklılık gösterdiği gözlemlenmektedir. Özellikle büyük problemler için, GH algoritmalarının tercih edilmesinin daha uygun olduğu görülmekledir. Ancak bilindiği gibi GH sezgiselleri her ne kadar bizim problemlerimizde uygulanabilir bir çözüm vermiş olsa da, tüm ders çizelgeleme problemleri için uygulanabilir bir çözüm bulacağını garanti etmemektedir. Bu nedenle, ders çizelgesi çözücümüzde IFS algoritmasının bir alternatif olarak bulunması önemlidir. İlk bakışta, ortalama maliyetler açısından daha iyi olan algoritmaların (*GH_goodPos* ve *IFS_goodPos*) ders çizelgeleme problemi için daha iyi tercihlermiş gibi görünmektedir. Ancak, bu her zaman böyle olmayabilir. Çünkü iyileştirme aşamasında asıl önemli olan durum yerel optimalle takılmamak ve çözüm kalitesini olabildiğince arttırmaktır. Her ne kadar kötü bir maliyet skoruyla da başlansa hızlıca çözüm iyileştirilebilir. Ya da iyi bir maliyet skoruyla başlandığı halde yerel optimale takılmaktan dolayı çözüm çok fazla iyileştirilemeyebilir. Çözüm için hangi algoritmanın daha iyi olacağını deneyip görmek gerekir. Ancak çözüm süresi açısından GH algoritmalarının daha iyi performans gösterdiği açıktır.



Şekil 12. Başlangıç Çözümü Algoritmaları İçin Ortalama Çözüm Süreleri

4.3 PARAMETRE TESTİ

Sezgisel modellerin karşılaştırmasını yapmadan önce, sezgisel parametreleri için uygun değerlerin belirlenmesi gerekmektedir. Bu parametreler başlangıç çözümü aşamasından sonraki aşama olan iyileştirme aşaması sezgisellerinde kullanılan parametrelerdir. Parametre değerleri sezgisellerin etkinliği ve çözüm kalitesi üzerinde doğrudan etkilidir. Bu nedenle problem yapısından etkilenen parametre değerlerinin doğru şekilde belirlenmesi önemlidir.

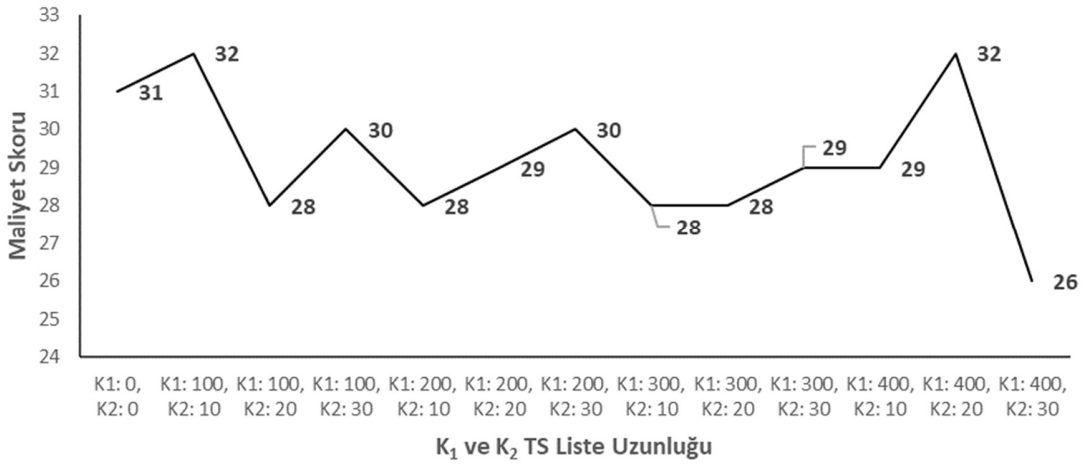
İyileştirme aşamasında kullanılan parametrelerin ayarlanması için sadece HÜİİBF ders çizelgeleme problemi (p1) kullanılmaktadır. Parametre testi için sadece bu problemin kullanılmasının nedeni diğer problemlerin bu problemle benzer yapıda olmasıdır. Çünkü diğer problemler bu problemin birer küçük parçasıdır. Yapı itibari ile önemli bir farklılık yoktur.

4.3.1 Tabu Arama Sezgiseli Parametre Ayarları

Yukarıda (Bölüm 3.4.2) bahsedildiği gibi iyileştirme aşaması sezgisellerinde iki tür (K_1 ve K_2) komşuluk hareketi kullanılmaktadır. İlk tür komşuluk hareketi pozisyonların takası (pt) komşuluk hareketi iken, ikinci tür komşuluk hareketleri ise zaman periyodu takası (zpt), tekli kempe zinciri (tkz) ve çoklu kempe zinciri ($çkz$) komşuluk hareketleridir. K_1 komşuluk

hareketi türünde pozisyon seçimi önemliyken, K_2 komşuluk hareketi türünde zaman periyodu seçimi önemli olmaktadır. Bu farklılıktan dolayı TS sezgiselinde her tür komşuluk hareketi için farklı bir tabu listesi kullanılmaktadır. Dolayısıyla tabu listesi uzunluğu da farklı olmalıdır. Çünkü olası pozisyon değişim sayısı ile olası zaman periyodu değişim sayısı birbirinden çok farklıdır. Tabu listesi uzunluğu da bu sayılar dikkate alınarak belirlenmelidir.

K_1 ve K_2 komşuluk hareketleri için test edilen tabu liste uzunlukları sırasıyla (0, 0), (100, 10), (100, 20), (100, 30), (200, 10), (200, 20), (200, 30), (300, 10), (300, 20), (300, 30), (400, 10), (400, 20) ve (400, 30) olarak belirlenmiştir. Her bir liste uzunluk ikilisi (K_1 liste uzunluğu, K_2 liste uzunluğu) için TS algoritması 10.000 iterasyonluk bir döngü ile 10'ar kez çalıştırılmış ve ulaşılan en iyi maliyet skorlarının ortalaması alınmıştır. Bu değerler Şekil 13'te gösterilmektedir



Şekil 13. Farklı Uzunluktaki Tabu Listeleri İçin Ortalama Maliyet Skorları

Tabu liste uzunluklarının 0 olarak alınması durumunda TS sezgiseli temel HC sezgiseline dönüşmektedir. Çünkü bu durumda herhangi bir yasaklı komşuluk hareketi söz konusu olmayacaktır. Algoritma HC sezgiseli gibi çalışacaktır. Liste uzunlukları arttıkça ortalama maliyet skorlarında bir değişim söz konusu olmaktadır. Bazı liste uzunluklarında düşüş olurken bazılarında ise yükseldiği görülmektedir. Bu durum problem yapısının problem çözümü üzerindeki etkisini göstermektedir. En düşük maliyet skoru değeri liste uzunlukları

sırasıyla 400 ve 30 iken elde edildiği görülmektedir. Bu nedenle bu problem için uygun tabu liste uzunlukları sırasıyla 400 ve 30 olarak alınmaktadır.

4.3.2 Benzetim Tavlaması Sezgiseli Parametre Ayarları

Benzetim tavlaması (SA) sezgiselinde soğutma prosedürünün yanında ısıtma prosedürünü de eklediğimiz için birden çok sayıda parametre gereksinimi söz konusu olmaktadır. Bu parametreler sırasıyla: maksimum sıcaklık (aynı zamanda başlangıç sıcaklık değeridir - $Tmax$) değeri, minimum sıcaklık ($Tmin$) değeri, sıcaklık düşüş oranı (β), soğutma katsayısı (sk), soğutma iterasyon sayısı ($iter\_1$), yeniden ısıtma katsayısı (yk), ısıtma için başlangıç iterasyon sayısı ($iter\_2$) ve ısıtma iterasyon sayısı artış katsayısıdır (α).

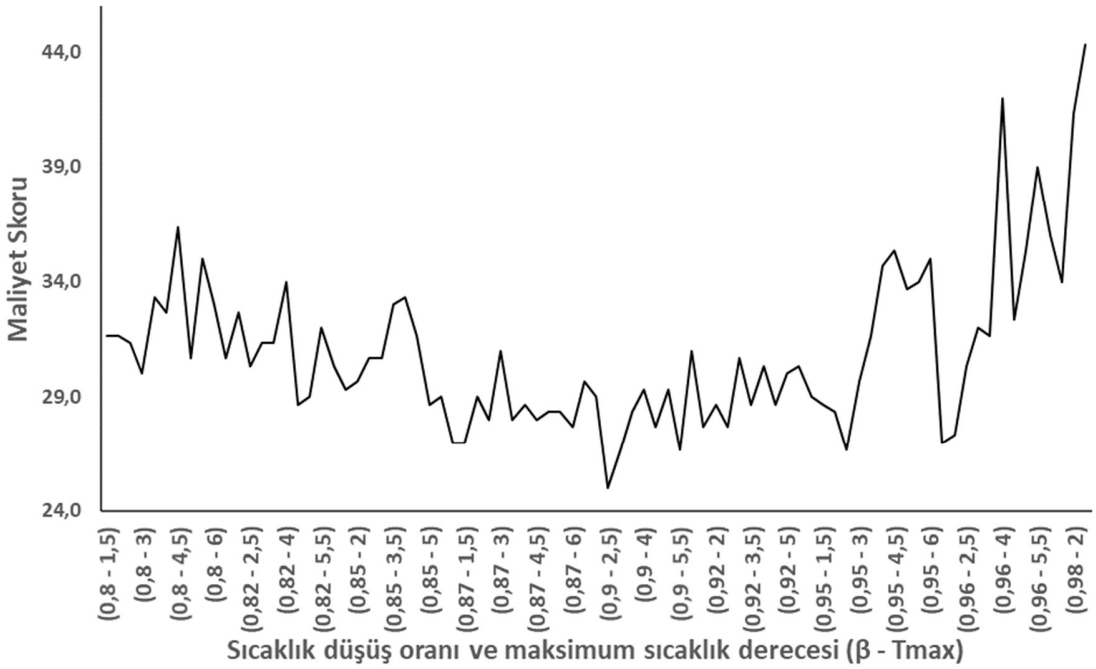
Parametre değerleri için problem üzerinde testler (her bir problem için algoritma 10'er kez çalıştırılmıştır) yapılarak değer aralıkları belirlenmiştir. Bu testler, algoritmanın mümkün olduğu kadar hızlı bir şekilde sürekli olarak iyi sonuçlar verebileceği parametre kombinasyonlarını bulmak amacıyla yapılmıştır. Tablo 19'da belirlenen parametre değerleri verilmektedir. Bu değerlerin birkaçı problem yapısına ve boyutuna göre farklılık gösterdiği için başlangıçta birer aralık değer olarak belirlenmiştir.

Tablo 19. Benzetim Tavlaması Sezgiseli Çözüm Parametreleri

Parametreler	Sembol	Aralık/Değer
Maksimum sıcaklık	$Tmax$	2,5
Minimum sıcaklık	$Tmin$	0,01
Sıcaklık düşüş oranı	β	0,9
Soğutma katsayısı	sk	[4, 5]
Yeniden ısıtma katsayısı	yk	7
Soğutma iterasyon sayısı	$iter\_1$	$[\frac{TL}{sk=4}, \frac{TL}{sk=5}]$
Yeniden ısıtma için başlangıç iterasyon sayısı	$iter\_2$	[300, 1000]
Yeniden ısıtma iterasyon sayısı artış katsayısı	α	1,4

TL: Problem boyutunu gösterir (örn: ders sayısı)

HÜİİBF ders çizelgeleme problemi (p1) için $Tmin$, sk, yk, $iter\_2$ ve α parametreleri sırasıyla 0,01, 5, 7, 1000 ve 1,4 olarak alınıp maksimum sıcaklık derecesi ($Tmax$) ve sıcaklık düşüş oranındaki (β) değişimin maliyet skoru üzerindeki etkisi hesaplanarak Şekil 14'de gösterilmektedir. Ortalama maliyet skorları, SA algoritması her β ve $Tmax$ ikilisi için 5'er (60'ar saniyelik süreler ile) kez çalıştırılıp elde edilen değerlerin ortalaması alınarak bulunmuştur. Şekle bakıldığında, ortalama maliyet skoru, sıcaklık düşüş oranı 0,9 ve maksimum sıcaklık değeri ise 2,5 iken en düşük değerini almaktadır. Sıcaklık düşüş oranı çok düşükken ve çok yüksekken ortalama maliyet skorlarının yüksek çıktığı görülmektedir.

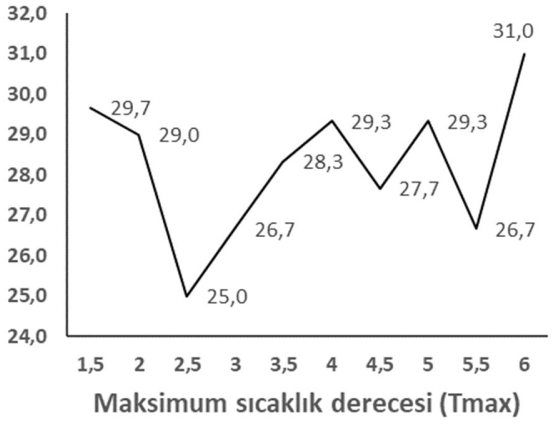


Şekil 14. Maksimum Sıcaklık Dereceleri ve Sıcaklık Düşüş Oranları İçin Ortalama Maliyet Skorları

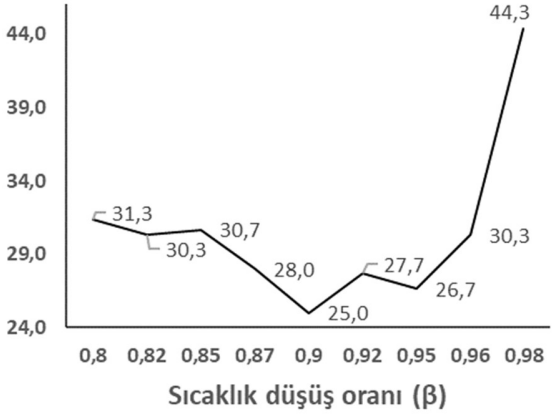
Bilindiği gibi sıcaklık derecesi ne kadar düşük olursa kötü sonuçlanın kabul edilme olasılığı o kadar azalmakta ve algoritma HC sezgiseline benzemektedir. Şekil 15'de bu durumun ortalama maliyet skorları üzerindeki etkisi görülmektedir. Sıcaklık düşüş oranın 0,9 alındığı bu şekilde $Tmax$ değerinin düşük ve yüksek olduğu değerlerde ortalama maliyet skorunun nispeten daha yüksek olduğu görülmektedir. Diğer değerlerde SA sezgiselinin ortalama maliyet skoru üzerindeki etkisi açıkça görülebilmektedir. Ayrıca şekilde $Tmax$ değeri 4 iken ortalama maliyet skoru, $Tmax$ değeri 5,5 iken elde edilen maliyet skorundan daha yüksek olması tutarsız bir durum gibi gözükse de aslında bu sezgisel algoritmaların genel bir özelliğidir. Problemin yapısı ve diğer parametrelerin değerleri yerel optimalden kaçma durumlarını etkilemektedir. Dolayısıyla parametre değerlerindeki değişim ile maliyet skorları arasında doğrusal bir ilişki aramamak gerekir.

Sıcaklık düşüş oranı (β) sıcaklık derecesinin ($Tmax$) ne kadar hızlı düşürüleceğini belirlemektedir. β değeri ne kadar yüksek olursa sıcaklık o kadar hızlı düşer ve kötü

sonuçların kabul edilme olasılığı hızlıca azalır. Bu durumda algoritma hızlı ilerler, fakat yerel optimale takılma olasılığı artar. β değeri ne kadar düşük olursa kötü sonuçların kabul edilme olasılığı o kadar daha yavaş düştüğü için algoritmanın hızı yavaşlar. Yerel optimalden çıkma olasılığı artsa da kısıtlı çözüm sürelerinde yeterince iyi sonuçlar elde edilemeyebilir. Şekil 16'da bu durumlar açıkça gözlemlenebilmektedir. β değeri düşükken ve yüksekken ortalama maliyet skorları yüksek çıktığı görülmektedir. Diğer değerlerde ise SA sezgiselin ortalama maliyet skorlarına etkisi görülmektedir. $Tmax$ değerinin 2,5 ve β değerinin 0,9 olduğu yerde en düşük ortalama maliyet skoru gerçekleşmektedir.



Şekil 15. Maksimum Sıcaklık Dercesine Göre Ortalama Maliyet Skorlarındaki Değişim



Şekil 16. Sıcaklık Düşüş Oranına Göre Ortalama Maliyet Skorlarındaki Değişim

Yukarıda p1 için belirlediğimiz parametre değerleri sk, $iter\_1$ ve $iter\_2$ parametreleri dışında diğer problemler içinde kullanılmaktadır. sk, $iter\_1$ ve $iter\_2$ parametreleri ise problemlerin boyutlarına göre güncellenerek kullanılmaktadır.

4.4 KOMŞULUK KOMBİNASYONU

Bölüm 3.4.2'de iki tür komşuluk yapısından bahsedilmektedir. İlk tür komşuluk hareketi (K_1) pozisyon takası (pt) komşuluk hareketidir. İkinci tür komşuluk hareketi (K_2) zaman periyodu takası (zpt), tekli kempe zinciri (tkz) ve çoklu kempe zinciri ($çkz$) komşuluk hareketlerinden

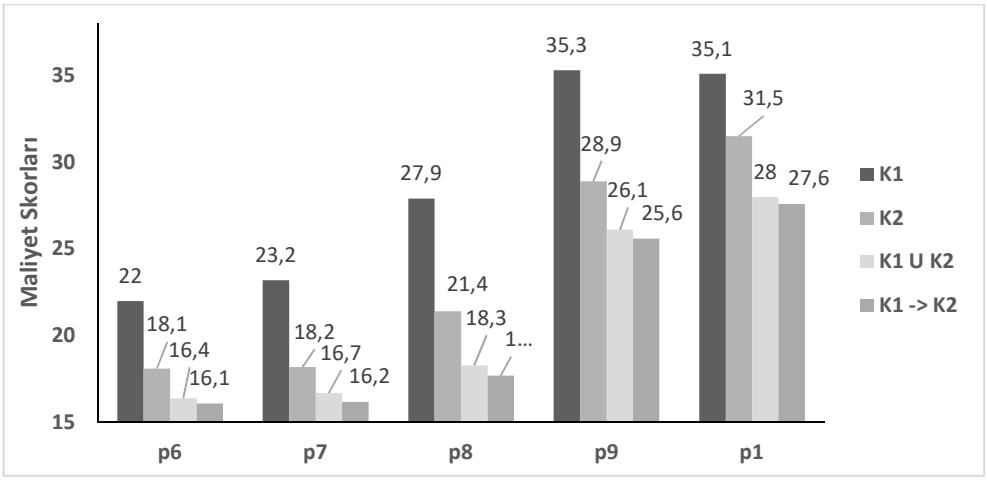
oluşmaktadır. Bu kısımda bu iki komşuluk türünün performansı ve özellikle kempe zinciri komşuluk hareketlerinin çözüme katkıları test edilmektedir.

Tablo 20'de 5 ayrı problem (p1, p6, p7, p8 ve p9) için farklı komşuluk kombinasyonlarının ortalama maliyet skorları verilmektedir. Bu maliyet skorları TS ve SA algoritması her bir problem için 10'ar kez 90 saniyelik çözüm süreleri ile çalıştırılarak bulunmuştur. Sonuçlara bakıldığında, K_2 ortalama maliyet skorları, K_1 ortalama maliyet skorlarından daha düşük çıkmıştır. Bu durum K_2 komşuluk hareketinin K_1 komşuluk hareketlerinden daha fazla maliyeti düşürücü etkisi olduğunu göstermektedir. Bilindiği gibi K_2 komşuluk türü içiresinde zpt, tkz ve $çkz$ komşuluk hareketlerini barındırmaktadır. Bu komşuluk hareketleri, K_1 komşuluk hareketine sırayla eklenerek, her birinin ortalama maliyet skoru üzerindeki etkisine bakılmıştır. Bunun sonucunda, özellikle çoklu kempe zinciri ($çkz$) komşuluk hareketinin ortalama maliyet skorlarını diğer komşuluk harekelerinden daha fazla düşürdüğü görülmüştür. Thompson ve Dowsland (1998) yılındaki çalışmasında bu sonuçlar desteklenmekte ve kempe zincirinin esnekliği arttırdığı için standart komşuluk hareketlerinden daha iyi sonuçlar verdiği söylenmektedir. $K_1 \cup K_2$ sütununda bu durum net bir şekilde görülmektedir. Algoritma $K_1 \cup K_2$ bileşimindeki komşuluk hareketlerini rastgele seçmektedir. $K_1 \rightarrow K_2$ ise algoritma komşuluk hareketlerini bir döngü şeklinde işleme koymaktadır. Yani belli sayıdaki tekrara kadar en iyi çözüm iyileştirilemiyorsa, algoritma bir diğer komşuluk hareketi türünü denenmeye başlar. Bu döngü süreç sonlanıncaya kadar devam eder. $K_1 \rightarrow K_2$ sütununa bakıldığında ortalama maliyet skorları $K_1 \cup K_2$ bileşiminden nispeten daha düşük olduğu görülmektedir. Bu nedenle, Di Gaspero ve Schaerf (2006); Lü ve Hao (2010) çalışmalarında olduğu gibi bu tez çalışmasında da komşuluk türleri bir döngü şeklinde uygulanması daha doğru olacaktır.

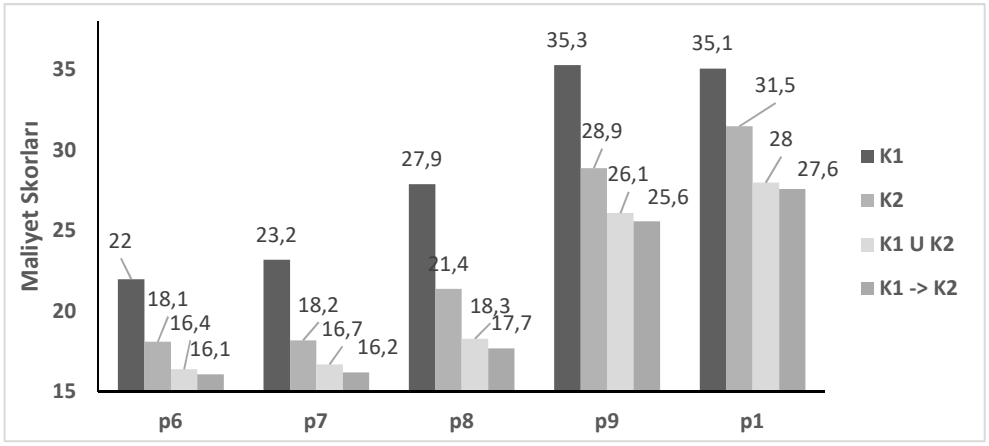
Tablo 20. Farklı Komşuluk Yapıları ve Kombinasyonları için Ortalama Maliyet Skorları

	Problemler	K_1	K_2	$K_1 \cup zpt$	$K_1 \cup zpt \cup tkz$	$K_1 \cup K_2$	$K_1 \rightarrow K_2$
TS	p6	22	18,1	20,2	17,7	16,4	16,1
	p7	23,2	18,2	19,4	19,5	16,7	16,2
	p8	27,9	21,4	23,7	22,3	18,3	17,7
	p9	35,3	28,9	33	33,4	26,1	25,6
	p1	35,1	31,5	35,4	35,1	28	27,6
SA	p6	19,3	17,6	18,4	16,5	16	16
	p7	19,9	17,6	19,4	17,4	16,2	16
	p8	22,1	19	24,6	22	19,1	16,9
	p9	32,2	27,1	33,5	29,8	24,6	24,5
	p1	33,1	28,8	37,7	34,9	26,37	25,8

Yukarıda ayrıntılı bir şekilde özetlediğimiz komşuluk kombinasyonlarının ortalama maliyet skorları üzerindeki etkisini Şekil 17 ve Şekil 18'de daha net bir şekilde gözlemlenebilir. Şekil 17 TS algoritmasının sonuçlarını özetlerken, Şekil 18 SA algoritmasının sonuçlarını özetlemektedir. Şekillerden de anlaşıldığı gibi TS ve SA algoritmalarının komşuluk kombinasyonları açısından yakın sonuçlar vermiştir. Dolayısıyla, bu sonuçlar her iki algoritma içinde komşuluk türlerinin bir döngü ($K_1 \rightarrow K_2$) şeklinde kullanılmasının daha uygun olacağını göstermektedir.



Şekil 17. Komşuluk Kombinasyonları İçin Ortalama Maliyet Skorları (TS)



Şekil 18. Komşuluk Kombinasyonları İçin Ortalama Maliyet Skorları (SA)

4.5 İYİLEŞTİRME AŞAMASI SEZGİSELLERİNİN KARŞILAŞTIRILMASI

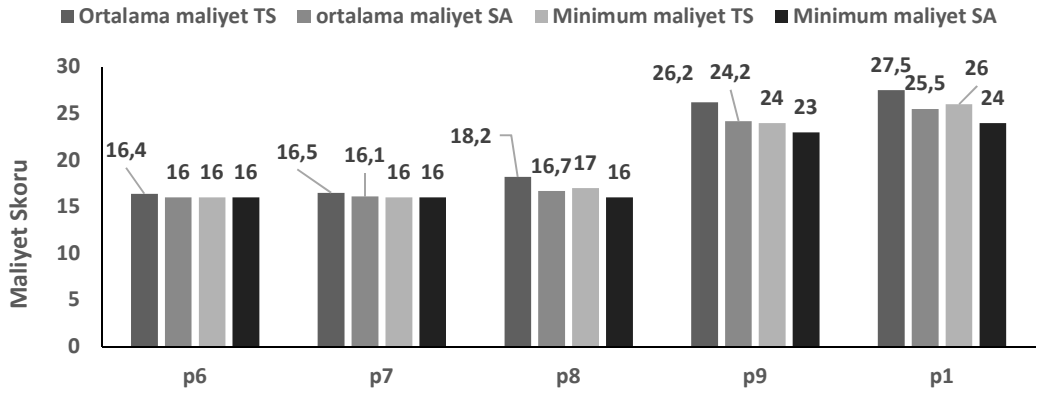
İyileştirme aşamasında, başlangıç çözümü elde edilen çizelgeleme problemlerinin esnek kısıt ihaleleri azaltılmaya çalışılmaktadır. Bu bölümde, iyileştirme aşaması için kullandığımız sezgisellerin performansları değerlendirilerek, hangi sezgisel yaklaşımın daha iyi sonuç verdiği tespit edilmektedir. Bunun için değerlendirmeye p1, p6, p7, p8 ve p9 alınmıştır. Çünkü bu problemler diğer problemlere kıyasla daha büyük boyutta ve daha kompleks yapıda olduğundan dolayı sezgisellerin çözüm üzerindeki etkileri daha net görülebilecektir. Bu problemler için GH sezgiseli kullanılarak başlangıç çözümleri elde edilmiştir. Daha sonra TS

ve SA sezgiselleri için Tablo 21'deki parametre değerleri girilerek her bir problem, bu sezgiseller ile 10'ar kez 90 saniyelik süreler boyunca çalıştırılarak çözümler iyileştirilmeye çalışılmıştır.

Tablo 21. Sezgisel Karşılaştırmaları için Parametre Değerleri

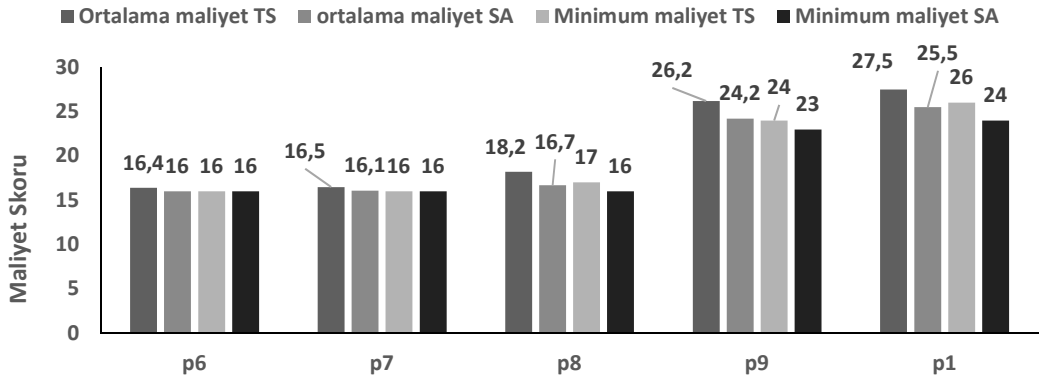
	Parametreler	Değerler
TS Sezgiseli	K_1 için liste uzunluğu	400
	K_2 için liste uzunluğu	30
SA Sezgiseli	$Tmax$	2,5
	$Tmin$	0,01
	β	0,9
	yk	7
	α	1,4
	$iter\_1$	50
	$iter\_2$	800

Her bir sezgisel için ortalama maliyet skorları Şekil 19'da verilmektedir. Sonuçlara göre SA sezgiselinin TS sezgiselinden daha düşük maliyet skorları bulabildiği görülmüştür. Problemlerin hepsinde maliyet skorları eşit veya SA sezgiseli lehine daha düşük çıkmıştır. Özellikle büyük boyutlu problemlerde bu fark belirgin olmuştur. Örneğin, HÜİİBF probleminde (p1) SA sezgiseli maliyeti 24'e kadar düşürmüştür. Bu çözümlerden, SA sezgiselinin esnek kısıtları ihlal etme maliyetini azaltma da, TS sezgiselinden daha iyi bir performans gösterdiği açık bir şekilde görülmektedir.



Şekil 19. Ortalama ve Minimum Maliyet Skorları, TS, SA, GH

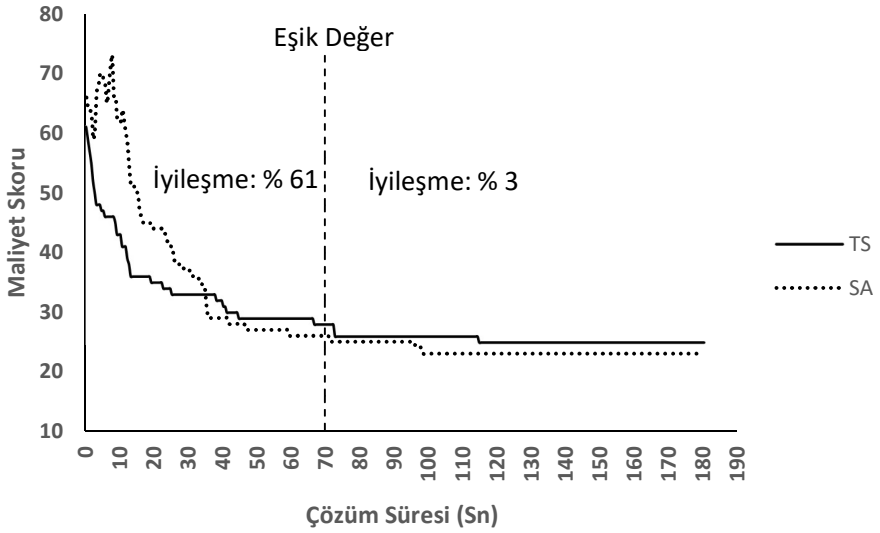
Başlangıç çözümü için *GH_goodPos* sezgiseli kullandığında TS ve SA sezgiselleri yukarıdaki sonuçlarla yakın sonuçlar vermiştir (Şekil 20). Bu durum, bu sezgisellerin her ne kadar daha iyi bir çizelge ile çözüme başlamış olsalar bile yakın noktalarda tıkandıklarını göstermektedir. Aslında bazı durumlarda kötü bir başlangıç çözümüyle başlamak yerel optimalden çıkmayı sağlayabilmekledir. Dolayısıyla, *GH_goodPos* sezgiseli her ne kadar küçük boyutlu problemler için çözüm süresinde avantaj sağlayabilse de, en azından bu problemler için maliyet açısından önemli bir avantajı yoktur. Ancak, *GH_goodPos* ve *IFS_goodPos* sezgiselleri esnek kısıtları da hesaba kattığından, bu sezgiselleri bazı problemlerde kullanmak kullanıcılar için yeterli bir çözüm olabilir. Yani iyileştirme sezgisellerini (TS ve SA) kullanmaya gerek kalmadan sadece bu sezgisellerden elde edilen çözümler tatmin edici çözümler olarak kabul edilebilir. Bu nedenle, *GH_goodPos* ve *IFS_goodPos* sezgisellerini daha hızlı çözümler verdiği için TS ve SA sezgisellerine bir alternatif olarak kabul edebiliriz.



Şekil 20. Ortalama ve Minimum Maliyet Skorları, TS, SA, *GH_GoodPos*.

Şekil 21'de TS ve SA sezgisellerinin çözüm sürelerinin çözüm kalitesi üzerindeki etkisi gösterilmektedir. Bu şekil, HÜİİBF (p1) problemi -180'er saniye çalıştırılarak- çözülerek elde edilmektedir. Yani, iyileştirme aşamasının başlangıcında maliyet skoru hızlı bir şekilde düşmekte, algoritmanın çözüm süresi uzadıkça da maliyet skoru yavaş bir düşüş göstermektedir. Bir noktadan sonra sezgisellerin yerel optimal ulaştığı ve maliyet skorunun

artık düşmediği görülmektedir. Diğer problemlerinde çözüm sürelerinin çözüm kalitesi üzerindeki etkisi benzer şeklide olmaktadır. SA sezgiselinin TS sezgiselinden daha iyi bir performans sergilediği bu şekilden de görülebilir. 180 saniyelik çözüm süresi sonunda, SA sezgiseli ile 23 maliyet skoruna, TS sezgiseli ile en düşük 26 maliyet skoruna inilebilmiştir.



Şekil 21. Çözüm Süresine Bağlı Maliyet Skorları

Çözüm süresinin 70 olduğu yer eşik değeri olarak kabul edildiğinde, 70 saniyelik çözüm süresine kadar maliyet skorundaki ortalama düşüşün % 61civarında olduğu, 70 saniyeden 180 saniyeye kadar olan zaman aralığında ise maliyet skorundaki ortalama düşüşün sadece %3'lerde kaldığı görülmektedir. Bu durum bize HÜİİB ders çizelgeleme probleminin sezgisel çözümünde ilk 70 saniyelik çözüm süresinde çözümün hızlıca iyileştiğini, ondan sonraki süre zarfında çözüm kalitesinin belli bir seviyede kaldığını göstermektedir.

SA sezgiselini belli bir olasılıkla kötü sonuçları da kabul ettiği, sıcaklık düştükçe kararlı hale geldiği ve bu nedenle kötü sonuçları kabul etmediği şekilde görülebilmektir. Önceden bahsedildiği gibi SA algoritmamızda yeniden ısıtma prosedürü de bulunmaktadır. Bu demek oluyor ki, eğer yerel optimalde kalma süresi uzamaya devam ederse sıcaklık tekrardan artacak ve bunun sonucunda kötü sonuçlar kabul edilmeye başlanacaktır. Devamında sıcaklığın düşürülme süreciyle birlikte yerel optimalden çıkmak tekrardan denenecektir.

4.6 TAMSAYILI DOĞRUSAL PROGRAMLAMA YAKLAŞIMI İLE SEZGİSEL YAKLAŞIMLARIN KARŞILAŞTIRILMASI

Bir önceki başlıkta, TS ve SA sezgisel yaklaşımlarının performansları karşılaştırılmışken, bu kısımda ise bu sezgisellerin performansı tamsayılı doğrusal programlama (TDP) yaklaşımı ile karşılaştırılmaktadır.

Tablo 22'de tüm problemlerin TDP, SA ve TS yaklaşımları ile elde edilen çözümleri yer almaktadır. TDP yaklaşımı ile tüm problemlerin kabul edilebilir bir çözüm süresi içinde optimal olarak çözülebildiği görülmektedir. Asıl problemimiz olan HÜİİBF (p1) probleminin de 95 saniyelik bir çözüm süresi sonucunda optimal olarak çözülebilmektedir. Sezgisel yaklaşımlar (SA, TS) ile sadece ilk dört problem (p2, p3, p4 ve p5) TDP yaklaşımının çözüm süresinden çok da kötü olmayan çözüm sürelerinde optimal olarak çözülebildiği görülmektedir. Diğer problemlerde ise optimal çözümlerin elde edilemediği görülmektedir. Bu durum, sezgisel yaklaşımlarımızın kötü performansa sahip olduğunu göstermez. Aslında TDP yaklaşımının kolayca çözüme ulaşabilecek şekilde iyi modellendiği gösterir.

Tablo 22. Modellerin Karşılaştırılması (TDP, SA, TS)

	TDP		SA		TS	
Problemler	OMS	ÇS (sn)	MS	ÇS (sn)	MS	Çs (sn)
p2	8	2,96	**8**	9,9	**8**	2,67
p3	8	1,57	**8**	3,3	**8**	4,81
p4	8	3,34	**8**	7,4	**8**	41
p5	8	3,59	**8**	26	**8**	29
p6	9	51	16	90	16	90
p7	9	67	16	90	16	90
p8	9	34	16	90	17	90
p9	10	69,2	23	90	24	90
p1	10	95,27	24	90	26	90

OMS: optimal maliyet skoru, LB: alt sınır maliyet skoru, MS: maliyet skoru, ÇS: çözüm süresi

Bir problemde, kabul edilebilir çözüm süresi içerisinde optimal çözüm elde edilebiliyorsa, o problem içim sezgisel yaklaşımların kullanılmasına gerek kalmayacaktır. Sezgisel yaklaşımlar, kompleks bir yapıya sahip ve/veya büyük boyutlu olmalarından dolayı optimal çözüm elde edilemeyen veya kabul edilebilir bir çözüm zamanı içerisinde çözülemeyen

problemler için iyi bir çözüm alternatifi olmaktadır. Bizim bu çalışmada ele aldığımız problemlerde bu şekliyle optimal çözülebildiği için çözüm için TDP yaklaşımının kullanılması en doğru seçenek olacaktır.

HÜİİBF (p1) ders çizelgeleme problemi aslında boyutu itibarıyla oldukça büyük ve kompleks yapıda bir problemedir. Bu problem TDP yaklaşımı ile standart şekilde modellendiği zaman 90 milyonu aşkın karar değişkeni ve yüz binlerle ifade edilen kısıt sayısı söz konusu olmaktadır. Problem bu şekliyle TDP yaklaşımı kullanılarak çözülmek istendiği zaman, bilgisayar yetersiz hafıza (*out of memory*) hatası verecektir. Ancak biz bu çalışmada, parametreleri ve karar değişkenleri gruplayarak karar değişkeni sayısını 55 binler civarına indirebildik. Ayrıca, veriyi organize ederek birçok kısıtı sağladığımız için kısıt sayısını da önemli ölçüde azaltmış olduk. Bu sayede problemiz TDP yaklaşımı ile kabul edilebilir bir çalışma süresi içerisinde optimal olarak çözülebilmiştir. Buna rağmen, probleme eklenecek yeni kısıtlar veya problemin boyutunun artması, yani problemin daha kompleks bir yapıya girmesi TDP yaklaşımı ile çözümü mümkün kılmayabilir. Bu nedenle, bir üniversite ders çizelgeleme çözücüsü tasarlamayalı amaç edindiğimiz bu çalışmada sezgisel yaklaşımlar da önermek bir gereklilik olmaktadır.

Bir sonraki kısımda derslik sabitliği (*room stability*) ksıtının modellere eklenmesinin çözüm üzerindeki etkisi incelenecektir. Bu kısıt esnek bir kısıt olarak düşünülmektedir.

4.7 PROBLEM KOMPLEKSLİĞİNİN ARTIRILMASININ PROBLEM ÇÖZÜMÜNE ETKİSİ

Çizelgeleme problemlerini kompleksliğinin veya boyutunun artırılması bu problemlerin optimal çözümünü zorlaştırabilmekte yada imkansız hale getirebilmektedir. Bu tez çalışmasına, bu başlığı eklememizin amacı HÜİİBF (p1) probleminin daha kompleks hale gelmesinin çözüme olan etkisini görmektir. Bunun için Bölüm 3.1.2'de bahsettiğimiz esnek kısıtlara (E1-E4) derslik sabitliği kısıtı bir diğer esnek kısıt (E5) olarak eklenmiştir.

Derslik sabitliği kısıtı ile aynı öğrenci grubuna (aynı müfredata) ait derslerin mümkün olduğunca aynı dersliklere atanması sağlanmaktadır. HÜİİBF ders çizelgeleme probleminin mevcut durumunda bu kısıt ihmal edilmektedir. Ancak, ileriki zamanalar da bu kısıtın dikkate alınması söz konusu olabilir. Ayrıca bu kısıtın dikkate alındığı çokça kuruma rastlanması mümkündür. Dolayısıyla, bu çalışmanın, diğer üniversite kurumlarının ders çizelgeleme problemlerine kolayca uyarlanabilecek modeller geliştirme amacı çerçevesinde derslik sabitliği kısıtının seçilmesi önemli olmaktadır.

Literatüre bakıldığında derslik sabitliği kısıtının ITC-2007 problemlerinin ve birçok gerçek hayat probleminin çözümünde kullanıldığı görülmektedir. Bu problemin çözümü için TDP yaklaşımları ile yazılmış iki aşamalı modeller öneren bazı çalışmalarda, derslik sabitliği kısıtının ilk aşamaya eklemenin kolay bir yolunun olmadığından ve hatta geliştirdikleri modeller için bunun mümkün olmadığından bahsedilmektedir. Bu çalışmalara örnek olarak Hao ve Benlic (2011), Lach ve Lübbecke (2008) ve Lach ve Lübbecke (2012) çalışmaları verilebilir. Bu çalışmalarda, genelde ilk aşamada dersler zaman periyotlarına atandıktan sonra, ikinci aşamada, derslik sabitliği kısıtı da dikkate alınarak derslik-ders eşleştirmesi yapılmaktadır. Buradaki dezavantaj problem iki aşamalı bir şekilde çözüldüğü için derslik sabitliği için en iyi çözüm bulunmayabilmektedir. Bu durumu çözebilmek için bu kısıtın ilk aşamaya dâhil edilebilmesi gerekir. Bu kısımda, tek aşamalı TDP modelimiz E5 kısıt dahil edilerek geliştirilmiştir.

E5 kısıtının TDP modelimize dahil edilebilmesi için gerekli olan ek karar değişkenleri ve ek kısıtlar şunlardır:

Karar değişkenleri:

- $\boldsymbol{sum\_RS_k}$, $(\boldsymbol{k}) \in (\boldsymbol{k} \in \boldsymbol{K})$, k öğrenci grubunun derslerinin atandığı farklı dersliklerinin sayısını veren karar değişkenidir. Sadece tamsayılı değerler alabilmelidir.

- $\boldsymbol{sum\_RS_{kn}}$, $(\boldsymbol{k}) \in (\boldsymbol{k} \in \boldsymbol{K}, \boldsymbol{n} \in \boldsymbol{N})$, n dersliğine atanan k öğrenci grubunun derslerinin sayısını vermektedir. Sadece tamsayılı değerler alabilmelidir.

- $\boldsymbol{RS_{kn}}$, $(\boldsymbol{k}) \in (\boldsymbol{k} \in \boldsymbol{K},\ \boldsymbol{n} \in \boldsymbol{N})$, n dersliğine k öğrenci grubunun herhangi bir dersi atandıysa 1 değerini alır. Atanmadıysa 0 değerini alır.

- $\boldsymbol{comp_{kn}}$, $(\boldsymbol{k}) \in (\boldsymbol{k} \in \boldsymbol{K},\ \boldsymbol{n} \in \boldsymbol{N})$, tamamlayıcı özelliği sahip bir karar değişkendir. Sadece [0- 0,999] arasında değer alabilen sürekli bir değişkendir.

Kısıtlar:

$$\sum_{i \in I_{nk}} \sum_{l \in L_{ki}} \sum_{m \in M_{kln}} x_{i,k,l,m,n} = sum\_RS_{kn} \qquad \forall k \in K, \forall n \in N \qquad (12)$$

$$\frac{sum\_RS_{kn}}{a_k} + comp_{kn} = RS_{kn} \qquad \forall k \in K, \forall n \in N \qquad (13)$$

$$\sum_{n \in N} RS_{kn} = 1 + sum\_RS_k \qquad \forall k \in K \qquad (14)$$

Derslik sabitliğini dikkate alan TDP modeli için Bölüm 3.3.4'de verilen kısıtlara 12,13 ve14 numaralı kısıtların eklenmesi gerekmektedir. Bu kısıtlardan 12 numaralı kısıt bir k öğrenci grubunun n dersliğine atanan derslerinin toplam sayısını vermektedir. Kısıt 13'de bu sayının k öğrenci grubunun toplam sayısına bölünmesiyle 1'e küçük eşit olan bir değer elde edilebilmektedir. Bu değerin, denklemin sağ tarafındaki 0-1 ikili değişkenine eşitleyebilmek için 0-0,999 arasında değer alabilen tamamlayıcı bir karar değişkeni ($comp_{kn}$) kullanılmaktadır. Yani, k öğrenci grubunun en az bir dersi n sınıfına atandıysa, denklemin sağ tarafı 1 değerine eşit olacaktır. Kısıt 14'te her k öğrenci grubu için bu değerlerin toplamı hesaplanmaktadır. Daha sonra, Bölüm 4.3.5'deki amaç fonksiyonuna bu toplamları minimize edecek olan denklem (15) eklenerek derslik sabitliğini dikkate alan TDP modelimiz oluşturulmuş olmaktadır.

$$denklem\ (11) + \left(\sum_{k \in K} sum\_RS_k \right) \qquad (15)$$

TDP modelinde olduğu gibi TS ve SA modellerinin maliyet hesaplama algoritması E5 esnek kısıtını da hesaba katacak şekilde değiştirilmiştir. Oluşturulan bu yeni modellerin sonuçları 23'te gösterilmektedir. E5 kısıtının TDP modelinin çözüm süresini önemli düzeyde etkilediği görülmektedir. Sadece ilk dört problemde optimal çözüme ulaşılmıştır. Diğer problemlerde ise uzun çözüm sürelerine rağmen optimal çözüme ulaşılamamıştır. Bu problemlerden biri olan HÜİİBF (p1) ders çizelgeleme problemi de yaklaşık 15000 saniye çalıştırılmasına rağmen optimal çözümde olan uzaklık yaklaşık olarak % 14,3'e kadar inebilmiştir.

Tablo 23. Yeni Modellerin Karşılaştırılması (TDP, SA, TS)

	TDP			SA		TS	
Problemler	OMS	Gap	ÇS(sn)	MS	ÇS(sn)	MS	ÇS(sn)
p2	15	-	127	18	34	19	360
p3	17	-	2402	23	3820	26	1000
p4	20	-	4017	26	3918	29	1050
p5	20	-	11715	31	1400	43	1089
p6	-	6,08%	15878	46	1500	65	1500
p7	-	15,1%	15475	48	1500	73	1500
p8	-	18,6%	15097	58	1500	84	1500
p9	-	18,9%	15070	70	1500	95	1500
p1	-	14,3%	15043	85	1500	116	1500

OMS: optimal maliyet skoru, AB: üst sınır maliyet skoru, MS: maliyet skoru, ÇS: çözüm süresi Gap: optimal çözüme göreli yaklaşma yüzdesi

TDP modeli ile bir çözüm elde edilemediği zaman sezgisel yaklaşımlar çözüm için önemli bir alternatif olmaktadır. SA ve TS algoritmaları ile elde edilen maliyet skorları Tablo 23'te yer almaktadır. Sezgisel modellere E5 kısıtı eklendiğinde de SA sezgiselinin TS sezgiselinden daha iyi performans gösterdiği görülmektedir. p1 için SA sezgiseli 1500 saniyelik bir çözüm süresinde 85 maliyet skoruna ulaşırken TS sezgiseli 116 maliyet skoruna ulaşabilmiştir.

Bu kısımda, problem kompleksliğinin ve boyutunun artması durumunda, TDP modelinin nasıl kolayca çözümsüz kalabileceği görülmüştür. Böyle durumlarda sezgisel yaklaşımların önemi ortaya çıkmaktadır. Çünkü optimal çözümlere ulaşılamasa da sezgisel yaklaşımalar ile göreli olarak iyileştirilmiş çözümlere ulaşılabilmiştir. Çoğu zaman sezgisel çözümler kullanıcılar için kabul edilebildiler çözümler olabilir. Bu çözümler optimal çözüm olmasalar

bile optimale yakın çözümler oldukları için problemin manuel olarak elde edilecek çözümlerinden çok daha iyi ve tatmin edici çözümler olacaklardır.

SONUÇ VE ÖNERİLER

Üniversite ders çizelgeleme oluşturmak çoğu kurum için oldukça zor bir problemdir. Bu problemleri elle çözmek çok sayıda personel ve iş yükü gerektirmekte, çokça zaman almakta ve çoğu kez tatmin edici bir sonuç vermemektedir. Bu problem için optimizasyon yaklaşımlarının modellenip kullanılması tatmin edici çözümlerin oluşturulmasına önemli katkı sağlayacaktır.

Bu çalışma da birçok bölüme sahip fakülte düzeyindeki ders çizelgeleme problemlerini TDP ve sezgisel yaklaşımlar ile çözebilecek bir çözücü tasarlanırken aynı zamanda benzer yapıdaki problemlere kolayca uyarlanabilecek atama modelleri oluşturulmuştur. Python programlama dili kullanılarak oluşturulan bu çözücü ile Hacettepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi (HÜİİBF) ders çizelgeleme problemi tüm modeller ile çözülerek elde edilen çözümler karşılaştırılmıştır.

Üniversite ders çizelgeleme problemi ile ilgili yapılan çalışmalarda genelde tercihler en çoktan en aza doğru verilen puanlara göre oluşturulan matrisler yardımıyla gerçekleştirilmeye çalışılmaktadır. Uygulama da ise çoğunlukla istenilmeyen veya istenilen zaman periyotları söz konuş olmaktadır. Geri kalan zaman periyotları arasında önemli bir fark gözetilmemektedir. Bu çalışma da önerilen modeller, hem modellere gereğinden fazla verinin yüklenmesinin önüne geçilebilmek ve hem de uygulama kolaylığı olması amacıyla dersler için bir veya birden fazla zaman periyodu tercihinde bulunabilecek şekilde tasarlanmıştır. Bu şekilde çözüm modelleri bir ders için eğer sadece bir zaman periyodu isteği varsa bu isteği, birden fazla zaman periyodu isteği varsa da bu isteklerden herhangi birini karşılamaya çalışmaktadır.

Bu çalışmada ele aldığımız uygulamada şubeli zorunlu dersler söz konusudur. Şubeli ders durumunu ele alan çok az sayıdaki çalışmaya bakıldığında temel odak noktasının derslerin optimal sayıda şubelere ayırmak olduğu görülmektedir. Oysaki bizim uygulamamızda dersler eldeki imkânlar çerçevesinde kaç şubeye ayrılacağı ve şubelerin hangi öğretim elemanları tarafından verileceği önceden belirlenmiştir. Bu nedenle, modellerimiz bir öğrenci grubunun

alması gereken şubeli zorunlu dersin en azından bir şubesini seçebilecek şekilde tasarlanmıştır. Bu sayede bu yapıya sahip çizelgeleme problemlerinde şubeli zorunlu derslerin nasıl modellenebileceği konusunda bir yaklaşım sunulabilmiştir.

Ders çizelgemle problemini çeşitli yaklaşımalar ile çözebilecek bir çözücü tasarlama amacı çerçevesinde ilk olarak, kullanıcının kolayca veri girebileceği, standart bir veri giriş formatı oluşturulmuştur. Geliştirilen çözücü nesne tabanlı kodlama tekniğini kullanarak dosyadan ham veriyi almakta, bu verilerden sınıflar (*class*) oluşturmakta ve sonra bu verileri modellerin gereksinim duyduğu veri formatına dönüştürmektedir. Bu sayede, ders çizelgeleme verisi girildikten sonra istenilen yaklaşım seçilerek modelin çözümüne geçilebilmektedir.

Çalışmada önerilen ilk model bir TDP modelidir. Daskalaki vd. (2004) kısıt sayısını küçültmek amacıyla kullandıkları, parametreleri indislere göre gruplama tekniği, bu model de, karar değişkenlerini de içine alacak şekilde geliştirilmiştir. Bu sayede, HÜİİBF ders çizelgeleme problemi için 90 milyondan fazla olan karar değişkeni sayısı 55 binler civarına düşürülebilmiştir. Bununla birlikte, bilgisayarda yetersiz hafıza (*out of memory*) hatasına düşme sorunu da çözülebilmiştir. Ayrıca TDP modeli derslerin zorunlu, şubeli zorunlu ve seçmeli olma durumlarını hesaba katmanın yanı sıra dersler için zaman periyodu isteklerini de mümkün olduğunca gerçekleştirebilecek şekilde tasarlanmıştır.

Üniversite ders çizelgesi probleminin sezgisel çözümü için iki aşamalı bir çözüm yaklaşımı önerilmiştir. Bu yaklaşımın, ilk aşamasında sadece zorunlu kısıtlar sağlanarak uygulanabilir bir başlangıç çözümü elde edilmektedir. İyileştirme aşaması olan ikinci aşamada ise, zorunlu kısıtlar ihlal edilmemesi şartıyla, esnek kısıtlar mümkün olduğunca sağlanarak çözümün kalitesi artırılmaktadır.

Başlangıç özümü için, bir açgözlü sezgisel (GH) ve bir iteratif ileri arama (IFS) sezgiseli olmak üzere birbirine alternatif olabilecek iki farklı sezgisel önerilmiştir. GH sezgiseli Lü ve Hao (2010) çalışmasından, IFS sezgiseli ise Müller vd. (2004) çalışmasından problemimizin yapısına uyarlanmıştır. İki sezgiselde, zorunlu kısıtları ihlal etmeden çalışmada kullanılan tüm problemler için uygulanabilir bir başlangıç çözümü vermiştir. Çözümlerde, GH sezgiselinin IFS sezgiselinden daha kısa sürede çözüme ulaşabildiği görülmüştür. Ancak, GH

sezgiselinin her problem için bir başlangıç çözümü vermeyi garanti etmediği için, çözücümüz için IFS sezgiseli iyi bir alternatif olacaktır. Ayrıca çalışma da bu başlangıç sezgiselleri esnek kısıtları da hesaba katacak şekilde değiştirilmiştir. Bu yeni sezgiseller (*GH_goodPos, IFS_goodPos*) daha düşük maliyette ancak daha uzun çözüm süresinde başlangıç çözümleri elde edebilmiştir. Daha düşük maliyetli başlangıç çözümlerin ulaşılabildiği için bu yeni sezgisellerin iyileştirme aşaması sezgisellerine alternatif olabileceği görülmüştür. Yani karar vericinin tercihine bağlı olarak bu sezgisellerin çözümleri yeterli görülerek iyileştirme aşamasına geçilmeyebilir.

İyileştirme aşaması için tabu arama (TS) ve benzetim tavlaması (SA)sezgiselleri kullanılmıştır. Gerçekleştirilen çözümlerde, SA sezgiselinin TS sezgiselinden daha iyi çözümler verdiği görülmüştür.

SA ve TS sezgisellerinde standart komşuluk hareketleri ve ileri düzey komşuluk hareketi olan kempe zinciri (*kempe chain*) komşuluk hareketleri kullanılmıştır. Yapılan testler sonucunda, kempe zinciri komşuluk hareketlerinin yerel optimalden çıkmak için çözüme önemli etkisinin olduğu görülmüştür. Bu testler, literatürdeki kempe zinciri komşuluk hareketleriyle ilgili tespitleri doğrulamıştır. Ayrıca, yapılan testler sonucunda komşuluk türlerinin bir döngü şeklinde uygulanmasının daha verimli olacağı görülmüştür.

Kempe zinciri komşuluk hareketleri çizelgeleme problemleri için ileri düzey komşuluk hareketleri olarak görülmektedir. Karmaşık yapısından dolayı modellenmesi zor olan bu komşuluk hareketi problemimize uyarlanarak modellenebilmiştir. Bu sayede, sezgisel modellerin çözüm arayışını çeşitlendirme özellikleri artırılmıştır.

TS sezgiselinde yerel optimalde kaçabilmek için iki farklı tabu listesi kullanılmıştır. Literatürde de önerildiği gibi her komşuluk türü için farklı bir tabu listesi belirlenmiştir. SA sezgiselinde kullanılan soğutma yaklaşımı Müller (2009) çalışmasında kullandığı soğutma ve yeniden ısıtma planından da esinlenilerek oluşturulmuştur. Bu yaklaşım sayesinde SA sezgiseli yerel optimale takıldığında sıcaklığın tekrar artırılabilmesi sayesinde modelin yerel optimalden çıkma olasılığı artırılmıştır. Yapılan çözümlerde yeniden ısıtma prosedürünün çözüm üzerindeki olumlu etkisi görülmüştür.

TDP yaklaşımı ile tüm problemler kabul edilebilir bir süre içerisinde optimal çözülebilmiştir. Aynı problemler SA ve TS algoritmaları ile çözüldüğünde ise sadece küçük boyutlu problemlerde (p2, p3, p4 ve p5) optimal çözüme ulaşıldığı, diğer problemlerde ise aynı çözüm süreleri içerisinde daha kötü sonuçlar elde edildiği görülmüştür. Eğer bir problem TDP yaklaşımı ile kabul edilebilir çözüm süreleri içerisinde optimal olarak çözülebiliyorsa, bu yaklaşımın tercih edilmesi mantıklı olan seçenektir. Ancak, ders çizelgeleme problemleri, problem kompleksliğinin ve problem boyutunun artmasıyla TDP modelleri kolayca çözümsüz kalabilmektedir. Bu durumda, optimal olmasa da çoğu zaman yeterince iyi sonuçlar veren sezgisel yaklaşımların kullanılması gerekir. Bu nedenle, üniversite ders çizelgeleme problemi için bir çözücü tasarladığımız bu çalışmada, sezgisel yaklaşımların önerilmesi gerekli bir durumdur. Nitekim, modellere ders sabitliği kısıtının eklenmesiyle problemin kompleksliği artırılmış ve özellikle HÜİİBF ders çizelgeleme probleminin optimal olarak çözülemediği görülmüştür. Bunun yerine SA ve TS sezgisel yaklaşımları kullanılarak çözümler elde edilmiştir. Bu çözümlerde de SA sezgiselinin TS sezgiselinden daha iyi sonuçlar vermiştir. Dolayısıyla, HÜİİBF ders çizelgeleme problemi için optimal çözüm elde edilmediği durumlarda en iyi tercihin SA olduğu söylenebilir.

Yapılan çalışmalara bakıldığında derslik sabitliği kısıtının sadece iki aşamalı modellerde ele alındığı görülmüştür. Genelde ilk aşamada dersler zaman periyotlarına atanmış, ikinci aşamada, derslik sabitliği kısıtı da dikkate alınarak derslik-ders eşleştirmesi yapılmıştır. Bu çalışmalarda kendi modelleri için bu kısıtı ilk aşamaya dâhil edilmesi için bir yol bulunamamış olsa da bizim tek aşamalı TDP modelimize bu kısıt dâhil edilebilmiştir. Ancak bu kısıt modelin kompleksliğini artırdığı için çözüm süreleri artmıştır. Burada kastedilen çalışmalar TDP yaklaşımı ile ders çizelgeleme problemini çözmeye çalışan çalışmalardır. Sezgisel modellere bu kısıtı eklemenin bir zorluğu yoktur.

Gelecekte yapılacak çalışmalarla ilgili şu önerileri verebiliriz: (1) Tasarladığımız bu çözücüye verilerin kolayca girilebildiği ve bazı bilgilerin sistemden kolayca çekilebildiği bir arayüz tasarlanabilir. (2) Bu çalışma da HÜİİBF problemi sadece lisans programları için çözülmektedir. Lisansüstü programlardan dolayı derslik ve/veya öğretim elemanın uygun olamama durumları veri olarak modelimize girilmektedir. Ancak, lisansüstü programlarının

dâhil olduğu bir problem, geliştirdiğimiz çözücüye kolayca dahil edilerek çözüm elde edilebilir. (3) Ders çizelgeleme modellerimiz, dersleri blok olarak işlenen problemler için tasarlanmıştır. Bu modeller dersleri bölerek de atayabilecek şekilde geliştirilebilir.

KAYNAKÇA

Abdullah, S. (2006). *Heuristic approaches for university timetabling problems.* University of Nottingham Nottingham,

Abdullah, S., Burke, E. K., & McCollum, B. (2007). *A hybrid evolutionary approach to the university course timetabling problem.* Paper presented at the 2007 IEEE congress on evolutionary computation.

Al-Yakoob, S. M., & Sherali, H. D. (2006). Mathematical programming models and algorithms for a class–faculty assignment problem. *European Journal of Operational Research, 173*(2), 488-507.

Al-Yakoob, S. M., & Sherali, H. D. (2007). Mixed-integer programming models for an employee scheduling problem with multiple shifts and work locations. *Annals of Operations Research, 155*(1), 119-142.

Aladag, C. H., Hocaoglu, G., & Basaran, M. A. (2009). The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem. *Expert Systems with Applications, 36*(10), 12349-12356.

Alvarez-Valdes, R., Crespo, E., & Tamarit, J. M. (2002). Design and implementation of a course scheduling system using Tabu Search. *European Journal of Operational Research, 137*(3), 512-523.

Arntzen, H., & Løkketangen, A. (2005). A tabu search heuristic for a university timetabling problem. In *Metaheuristics: progress as real problem solvers* (pp. 65-85): Springer.

Asmuni, H. (2008). *Fuzzy methodologies for automated university timetabling solution construction and evaluation.* University of Nottingham,

Asmuni, H., Burke, E. K., & Garibaldi, J. M. (2005). *Fuzzy multiple heuristic ordering for course timetabling.* Paper presented at the Proceedings of the 5th United Kingdom workshop on computational intelligence (UKCI 2005).

Asratian, A. S., & de Werra, D. (2002). A generalized class–teacher model for some timetabling problems. *European Journal of Operational Research, 143*(3), 531-542.

Aycan, E., & Ayav, T. (2009). *Solving the course scheduling problem using simulated annealing.* Paper presented at the Advance Computing Conference, 2009. IACC 2009. IEEE International.

Babaei, H., Karimpour, J., & Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers Industrial Engineering, 86*, 43-59.

Badoni, R. P., Gupta, D., & Mishra, P. (2014). A new hybrid algorithm for university course timetabling problem using events based on groupings of students. *Computers & Industrial Engineering, 78*, 12-25.

Badri, M. A., Davis, D. L., Davis, D. F., & Hollingsworth, J. (1998). A multi-objective course scheduling model: Combining faculty preferences for courses and times. *Computers & operations research, 25*(4), 303-316.

Bakır, M. A., & Aksop, C. (2008). A 0-1 integer programming approach to a university timetabling problem. *Hacettepe Journal of Mathematics Statistics, 37*(1), 41-55.

Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A., & Urli, T. (2016). Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem. *Computers Operations Research, 65*, 83-92.

Bellio, R., Di Gaspero, L., & Schaerf, A. (2012). Design and statistical analysis of a hybrid local search algorithm for course timetabling. *Journal of scheduling, 15*(1), 49-61.

Beyrouthy, C., Burke, E. K., Landa-Silva, D., McCollum, B., McMullan, P., & Parkes, A. J. (2006). *The teaching space allocation problem with splitting.* Paper presented at the International Conference on the Practice and Theory of Automated Timetabling.

Bonutti, A., De Cesco, F., Di Gaspero, L., & Schaerf, A. (2012). Benchmarking curriculum-based course timetabling: formulations, data formats, instances, validation, visualization, and results. *Annals of Operations Research, 194*(1), 59-70.

Brélaz, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM, 22*(4), 251-256.

Broek, J., Hurkens, C., & Woeginger, G. (2009). Timetabling problems at the TU Eindhoven. *European Journal of Operational Research, 196*(3), 877-885.

Burke, E., & De Werra, D. (2013). Applications to timetabling. In *Handbook of graph theory* (pp. 530-562): Chapman and Hall/CRC.

Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., & Schulenburg, S. (2003). Hyper-heuristics: An emerging direction in modern search technology. In *Handbook of metaheuristics* (pp. 457-474): Springer.

Burke, E. K., Eckersley, A. J., McCollum, B., Petrovic, S., & Qu, R. J. E. J. o. O. R. (2010). Hybrid variable neighbourhood approaches to university exam timetabling. *206*(1), 46-53.

Burke, E. K., & Kendall, G. (2005). *Search methodologies* (E. K. Burke & G. Kendall Eds.): Springer.

Cacchiani, V., Caprara, A., Roberti, R., & Toth, P. (2013). A new lower bound for curriculum-based course timetabling. *Computers Operations Research, 40*(10), 2466-2477.

Cambazard, H., Hebrard, E., O'Sullivan, B., & Papadopoulos, A. (2012). Local search and constraint programming for the post enrolment-based course timetabling problem. *Annals of Operations Research*

194(1), 111-135.

Ceschia, S., Di Gaspero, L., & Schaerf, A. (2012). Design, engineering, and experimental analysis of a simulated annealing approach to the post-enrolment course timetabling problem. *Computers Operations Research, 39*(7), 1615-1624.

Chaudhuri, A., & De, K. (2010). Fuzzy genetic heuristic for university course timetable problem. *Int. J. Advance. Soft Comput. Appl, 2*(1), 100-123.

Clark, M., Henz, M., & Love, B. (2008). *QuikFix—A Repair-based Timetable Solver.* Paper presented at the Proceedings of the Seventh International Conference on the Practice and Theory of Automated Timetabling, http://www. comp. nus. edu. sg/~ henz/publications/ps/PATAT2008. pdf

Cooper, T. B., & Kingston, J. H. (1995). *The complexity of timetable construction problems.* Paper presented at the International Conference on the Practice and Theory of Automated Timetabling.

Corne, D., Ross, P., & Fang, H. (1995). Evolving timetables. *The practical handbook of genetic algorithms, 1*, 219-276.

Dandashi, A., & Al-Mouhamed, M. (2010). *Graph Coloring for class scheduling.* Paper presented at the ACS/IEEE International Conference on Computer Systems and Applications-AICCSA 2010.

Daskalaki, S., & Birbas, T. (2005). Efficient solutions for a university timetabling problem through integer programming. *European Journal of Operational Research, 160*(1), 106-120.

Daskalaki, S., Birbas, T., & Housos, E. (2004). An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research, 153*(1), 117-135.

De Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research, 19*(2), 151-162.

De Werra, D. (1996). Extensions of coloring models for scheduling purposes. *European Journal of Operational Research, 92*(3), 474-492.

De Werra, D. (1997). The combinatorics of timetabling. *European Journal of Operational Research, 96*(3), 504-513.

Deris, S., Omatu, S., & Ohta, H. (2000). Timetable planning using the constraint-based reasoning. *Computers & operations research, 27*(9), 819-840.

Deris, S., Omatu, S., Ohta, H., & Saad, P. (1999). Incorporating constraint propagation in genetic algorithm for university timetable planning. *Engineering applications of artificial intelligence, 12*(3), 241-253.

Di Gaspero, L., McCollum, B., & Schaerf, A. (2007). *The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (track 3)*. Retrieved from

Di Gaspero, L., & Schaerf, A. (2006). Neighborhood portfolio approach for local search applied to timetabling problems. *Journal of Mathematical Modelling Algorithms, 5*(1), 65-89.

Dimopoulou, M., & Miliotis, P. (2001). Implementation of a university course and examination timetabling system. *European Journal of Operational Research, 130*(1), 202-213.

Dimopoulou, M., & Miliotis, P. (2004). An automated university course timetabling system developed in a distributed environment: A case study. *European Journal of Operational Research, 153*(1), 136-147.

Elmohamed, M. S., Coddington, P., & Fox, G. (1997). *A comparison of annealing techniques for academic course scheduling.* Paper presented at the International Conference on the Practice and Theory of Automated Timetabling.

Feizi-Derakhshi, M.-R., Babaei, H., & Heidarzadeh, J. (2012). *A survey of approaches for University course timetabling problem.* Paper presented at the Proceedings of 8th international symposium on intelligent and manufacturing systems, Sakarya University Department of Industrial Engineering, Adrasan, Antalya, Turkey.

Gendreau, M., & Potvin, J.-Y. (2010). *Handbook of metaheuristics. International series inoperations research and management science.* (Vol. 2): Springer.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision sciences, 8*(1), 156-166.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers Operations Research, 13*(5), 533-549.

Glover, F. (1989). Tabu search—part I. *ORSA Journal on computing, 1*(3), 190-206.

Glover, F. (1990). Tabu search—part II. *ORSA Journal on computing, 2*(1), 4-32.

Goh, S. L., Kendall, G., & Sabar, N. R. (2017). Improved local search approaches to solve the post enrolment course timetabling problem. *European Journal of Operational Research, 261*(1), 17-29.

Hao, J.-K., & Benlic, U. (2011). Lower bounds for the ITC-2007 curriculum-based course timetabling problem. *European Journal of Operational Research, 212*(3), 464-472.

Kang, L., & White, G. M. (1992). A logic approach to the resolution of constraints in timetabling. *European Journal of Operational Research, 61*(3), 306-317.

Kiefer, A., Hartl, R. F., & Schnell, A. (2017). Adaptive large neighborhood search for the curriculum-based course timetabling problem. *Annals of Operations Research, 252*(2), 255-282.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science, 220*(4598), 671-680.

Kohshori, M. S., & Abadeh, M. S. (2012). Hybrid genetic algorithms for university course timetabling. *International Journal of Computer Science Issues(IJCSI), 9*(2).

Kristiansen, S., & Stidsen, T. R. (2013). *A comprehensive study of educational timetabling-a survey*. Retrieved from Department of Management Engineering, Technical University of Denmark. (DTU Management EngineeringReport; No. 8.2013).

Lach, G., & Lübbecke, M. E. (2008). *Optimal university course timetables and the partial transversal polytope.* Paper presented at the International Workshop on Experimental and Efficient Algorithms.

Lach, G., & Lübbecke, M. E. (2012). Curriculum based course timetabling: new solutions to Udine benchmark instances. *Annals of Operations Research, 194*(1), 255-272.

Le Huédé, F., Grabisch, M., Labreuche, C., & Savéant, P. (2006). MCS—A new algorithm for multicriteria optimisation in constraint programming. *Annals of Operations Research, 147*(1), 143-174.

Lewis, R. (2008). A survey of metaheuristic-based techniques for university timetabling problems. *OR spectrum, 30*(1), 167-190.

Lewis, R., Paechter, B., & McCollum, B. (2007a). *Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition*. Retrieved from

Lewis, R., Paechter, B., & Rossi-Doria, O. (2007b). Metaheuristics for university course timetabling. In *Evolutionary scheduling* (pp. 237-272): Springer.

Lopes, L., & Smith-Miles, K. (2010). *Pitfalls in instance generation for Udine timetabling.* Paper presented at the International Conference on Learning and Intelligent Optimization.

Lü, Z., & Hao, J.-K. (2010). Adaptive tabu search for course timetabling. *European Journal of Operational Research, 200*(1), 235-244.

McCollum, B., Schaerf, A., Paechter, B., McMullan, P., Lewis, R., Parkes, A. J., . . . Burke, E. K. (2010). Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing, 22*(1), 120-130.

Merlot, L. T., Boland, N., Hughes, B. D., & Stuckey, P. J. (2002). *A hybrid algorithm for the examination timetabling problem.* Paper presented at the International Conference on the Practice and Theory of Automated Timetabling.

MirHassani, S. (2006). A computational approach to enhancing course timetabling with integer programming. *Applied Mathematics Computation, 175*(1), 814-822.

MirHassani, S., & Habibi, F. (2013). Solution approaches to the course timetabling problem. *Artificial Intelligence Review, 39*(2), 133-149.

Müller, T. (2005). Constraint-based timetabling.

Müller, T. (2009). ITC2007 solver description: A hybrid approach. *Annals of Operations Research, 172*(1), 429.

Müller, T., Barták, R., & Rudová, H. (2004). *Conflict-based statistics.* Paper presented at the J. Gottlieb, D. Landa Silva, N. Musliu, and E. Soubeiga, editors, EU/ME Workshop on Design and Evaluation of Advanced Hybrid Meta-Heuristics. University of Nottingham.

Müller, T., & Murray, K. (2010). Comprehensive approach to student sectioning. *Annals of Operations Research, 181*(1), 249-269.

Müller, T., & Rudová, H. (2016). Real-life curriculum-based timetabling with elective courses and course sections. *Annals of Operations Research, 239*(1), 153-170.

Nourani, Y., & Andresen, B. (1998). A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical General, 31*(41), 8373.

Obit, J. H. (2010). *Developing novel meta-heuristic, hyper-heuristic and cooperative search for course timetabling problems.* University of Nottingham,

Petrovic, S., & Burke, E. K. (2004). University Timetabling. In.

Phillips, A. E., Walker, C. G., Ehrgott, M., & Ryan, D. M. (2017). Integer programming for minimal perturbation problems in university course timetabling. *Annals of Operations Research, 252*(2), 283-304.

Phillips, A. E., Waterer, H., Ehrgott, M., & Ryan, D. M. (2015). Integer programming methods for large-scale practical classroom assignment problems. *Computers Operations Research, 53*, 42-53.

Pillay, N. (2013). *A comparative study of hyper-heuristics for solving the school timetabling problem.* Paper presented at the Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference.

Qu, R., Burke, E. K., McCollum, B., Merlot, L. T., & Lee, S. Y. (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of scheduling, 12*(1), 55-89.

Schaerf, A. (1999). A survey of automated timetabling. *J Artificial intelligence review, 13*(2), 87-127.

Schimmelpfeng, K., & Helber, S. (2007). Application of a real-world university-course timetabling model solved by integer programming. *OR spectrum, 29*(4), 783-803.

Schindl, D. (2019). Optimal student sectioning on mandatory courses with various sections numbers. *Annals of Operations Research, 275*(1), 209-221.

Silva, J. D. L., Burke, E. K., & Petrovic, S. (2004). An introduction to multiobjective metaheuristics for scheduling and timetabling. In *Metaheuristics for multiobjective optimisation* (pp. 91-129): Springer.

Sørensen, M., & Dahms, F. H. (2014). A two-stage decomposition of high school timetabling applied to cases in Denmark. *Computers Operations Research, 43*, 36-49.

Şahin, T. (2004). *Goal programming Approach to Solve The Timetabling problem at Turkish Military Academy.* Bilkent Universty Ankara,

Taillard, E. (1990). Some efficient heuristic methods for the flow shop sequencing problem. *European Journal of Operational Research, 47*(1), 65-74.

Tarawneh, L., Ayob, M., & Ahmad, Z. (2013). A hybrid simulated annealing with solutions memory for curriculum-based course timetabling problem. *Journal of Applied Sciences, 13*(2), 262-269.

Thompson, J. M., & Dowsland, K. A. (1998). A robust simulated annealing based examination timetabling system. *Computers Operations Research, 25*(7-8), 637-648.

Tuga, M., Berretta, R., & Mendes, A. (2007). *A hybrid simulated annealing with kempe chain neighborhood for the university timetabling problem.* Paper presented at the 6th IEEE/ACIS international conference on computer and information science (ICIS 2007).

Vasquez, M., & Hao, J.-K. (2001). A "logic-constrained" knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite. *Computational optimization applications, 20*(2), 137-157.

Wang, Y.-Z. (2002). An application of genetic algorithm methods for teacher assignment problems. *Expert Systems with Applications, 22*(4), 295-302.

Welsh, D. J., & Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal, 10*(1), 85-86.

Wren, A. (1995). *Scheduling, timetabling and rostering—a special relationship?* Paper presented at the International Conference on the Practice and Theory of Automated Timetabling.

Zhang, L., & Lau, S. (2005). *Constructing university timetable using constraint satisfaction programming approach*. Paper presented at the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06).

EK 1. ETİK KURUL MUAFİYET FORMU

HACETTEPE ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
TEZ ÇALIŞMASI ETİK KOMİSYON MUAFİYETİ FORMU

HACETTEPE ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME ANABİLİM DALI BAŞKANLIĞI'NA

Tarih: 01/11/2019

Tez Başlığı: ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİNİN TAMSAYILI DOĞRUSAL PROGRAMLAMA VE SEZGİSEL YAKLAŞIMALR İLE ÇÖZÜMÜ

Yukarıda başlığı gösterilen tez çalışmam:

1. İnsan ve hayvan üzerinde deney niteliği taşımamaktadır,
2. Biyolojik materyal (kan, idrar vb. biyolojik sıvılar ve numuneler) kullanılmasını gerektirmemektedir.
3. Beden bütünlüğüne müdahale içermemektedir.
4. Gözlemsel ve betimsel araştırma (anket, mülakat, ölçek/skala çalışmaları, dosya taramaları, veri kaynakları taraması, sistem-model geliştirme çalışmaları) niteliğinde değildir.

Hacettepe Üniversitesi Etik Kurullar ve Komisyonlarının Yönergelerini inceledim ve bunlara göre tez çalışmamın yürütülebilmesi için herhangi bir Etik Kurul/Komisyon'dan izin alınmasına gerek olmadığını; aksi durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Gereğini saygılarımla arz ederim.

01/11/2019

Adı Soyadı: Akın ÖZKAN
Öğrenci No: N15149512
Anabilim Dalı: İşletme
Programı: İşletme-Doktora
Statüsü: ☐ Yüksek Lisans ☒ Doktora ☐ Bütünleşik Doktora

DANIŞMAN GÖRÜŞÜ VE ONAYI

Prof. Dr. Aydın ULUCAN

Telefon: 0-312-2976860 **Detaylı Bilgi:** http://www.sosyalbilimler.hacettepe.edu.tr **Faks:** 0-3122992147 **E-posta:** sosyalbilimler@hacettepe.edu.tr

EK 2. ORİJİNALLİK RAPORU

HACETTEPE ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
DOKTORA TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

HACETTEPE ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME ANABİLİM DALI BAŞKANLIĞI'NA

Tarih: 01/11/2019

Tez Başlığı : ÜNİVERSİTE DERS ÇİZELGELEME PROBLEMİNİN TAMSAYILI DOĞRUSAL PROGRAMLAMA VE SEZGİSEL YAKLAŞIMALR İLE ÇÖZÜMÜ

Yukarıda başlığı gösterilen ve Danışmanlığımda hazırlanan tez çalışmasının a) Kapak sayfası, b) Giriş, c) Ana bölümler ve d) Sonuç kısımlarından oluşan toplam 119 sayfalık kısmına ilişkin, 01/11/2019 tarihinde Turnitin adlı intihal tespit programından aşağıda işaretlenmiş filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezin benzerlik oranı % 3 'tür.

Uygulanan filtrelemeler:

1- ☒ Kabul/Onay ve Bildirim sayfaları hariç
2- ☒ Kaynakça hariç
3- ☒ Alıntılar hariç
4- ☐ Alıntılar dâhil
5- ☒ 5 kelimeden daha az örtüşme içeren metin kısımları hariç

Hacettepe Üniversitesi Sosyal Bilimler Enstitüsü Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Uygulama Esasları'nı inceledim ve bu Uygulama Esasları'nda belirtilen azami benzerlik oranlarına göre tez çalışmasının herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

TEZ DANIŞMANI: Prof. Dr. Aydın LUCUAN

İMZA:

TEZİ HAZIRLAYAN ÖĞRENCİ BİLGİLERİ:

01/11/2019

Adı Soyadı:	Akın ÖZKAN
Öğrenci No:	N15149512
Anabilim Dalı:	İşletme
Programı:	İşletme-Doktora
Statüsü:	☒ Doktora ☐ Bütünleşik Dr.