T.C.
NECMETTİN ERBAKAN
ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HEMŞİRE NÖBET ÇİZELGELEME PROBLEMİNİN TAVLAMA BENZETİMİ ALGORİTMASI İLE ÇÖZÜMÜ

Bilgen AYAN KOÇ

YÜKSEK LİSANS TEZİ

Endüstri Mühendisliği Anabilim Dalı

Haziran-2019
KONYA

TEZ KABUL VE ONAYI

Bilgen AYAN KOÇ tarafından hazırlanan "Hemşire Nöbet Çizelgeleme Probleminin Tavlama Benzetimi Algoritması ile Çözümü" adlı tez çalışması 17/06/2019 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. Saadettin Erhan KESEN	.....
Danışman Prof. Dr. Mehmet AKTAN	.....
Üye Dr.Öğrt.Üyesi Kemal ALAYKIRAN	.....

Yukarıdaki sonucu onaylarım.

Prof. Dr. Süleyman Savaş DURDURAN
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Bilgen AYAN KOÇ

Tarih:

ÖZET

YÜKSEK LİSANS TEZİ

HEMŞİRE NÖBET ÇİZELGELEME PROBLEMİNİN TAVLAMA BENZETİMİ ALGORİTMASI İLE ÇÖZÜMÜ

Bilgen AYAN KOÇ

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Mehmet AKTAN

2019, 100 Sayfa

Jüri
Prof. Dr. Mehmet AKTAN
Doç. Dr. Saadettin Erhan KESEN
Dr. Öğretim Üyesi Kemal ALAYKIRAN

Hemşire çizelgeleme problemi (HÇP), personel çizelgeleme problemleri içerisinde spesifik, daha zor ve insan sağlığıyla ilgili olması nedeniyle daha hassas ve önemli bir konudur. Bu çalışmada, haftanın her günü 24 saat kesintisiz hizmet veren bir hastanede 2 vardiya halinde (08:00-16:00 arası 8 saat ve 16:00-08:00 arası 16 saat) çalışan 15 hemşire için varsayılan kısıtlar çerçevesinde tavlama benzetimi algoritması kullanılarak en iyi çalışma çizelgesinin oluşturulması amaçlanmıştır. Araştırmada yöntem olarak, HÇP'nin çözümüne yönelik geliştirilen sezgisel yöntemlerden biri olan Tavlama Benzetimi Algoritması (TBA) kullanılmıştır. Verilerin analizinde kısıtların matematiksel formüllerle ifade edilebilmesini kolaylaştırmak, ana amaç ve alt amaç fonksiyonlarının değerlerinin doğru olarak hesaplandığından emin olmak için öncelikle Microsoft Office Excel programından faydalanılmıştır. Daha sonra oluşturulan ve doğruluğu test edilen matematiksel ifadelerin kodları Matlab programıyla eşdeğer olup kullanımı ücretsiz olan GNU Octave (2018) Version 4.4.1'in Windows-64 işletim sistemi için geliştirilen versiyonunda yazılmıştır. Araştırma kapsamında geliştirilen tavlama algoritması, genel tavlama algoritmasının temel mantığına sadık kalınarak ancak geliştirilerek yeniden yazılmıştır. Algoritma ile ele alınan problemin uyması beklenen 8'i zorunlu ve 8'i esnek olmak üzere 16 kısıtın tamamını sağlayarak en iyi çizelgeyi oluşturması sağlanabilmiştir. En iyi çözüme ulaşma süresi verilen problemde 0-8 izinli hemşire durumunda yani 7-15 çalışan hemşire durumunda ortalama 34 saniye (min.150-maks.1.338 deneme) olarak hesaplanmıştır. Algoritma, 9 izinli hemşire yani 6 çalışan hemşire durumu için de test edilmiş ve 195 saniyede (min.456-maks.4.864 deneme) tüm kısıtları sağlayan en iyi çözüme ulaşmıştır. Algoritma toplamda 200 kere test edilmiş, tamamında 0-9 izinli (6-15 çalışan) hemşire için 5.000 denemenin (500 saniyenin) altında çözüme ulaşmıştır. Araştırmada tavlama algoritması için yeni bir sıcaklık düşürme tekniği (logaritmik-üstel soğutma) ve yeni bir atama tekniği (belirli kritik değerlerin altında ve üstünde farklı atama tekniği kullanma) geliştirilmiş, bu iki tekniği içeren tavlama algoritmasına "çifte atamalı çoklu tavlama algoritması" adı verilmiştir.

Anahtar Kelimeler: Atama Problemi, Hemşire Çizelgeleme, Nöbet Çizelgeleme, Personel Çizelgeleme, Vardiya Atama, Tavlama Algoritması.

ABSTRACT

MS THESIS

THE SOLUTION OF NURSE SCHEDULING PROBLEM WITH SIMULATED ANNEALING ALGORITHM

Bilgen AYAN KOÇ

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN INDUSTRIAL ENGINEERING

Advisor: Prof. Dr. Mehmet AKTAN

2019, 100 Pages

Jury
Prof. Dr. Mehmet AKTAN
Doç. Dr. Saadettin Erhan KESEN
Dr. Öğretim Üyesi Kemal ALAYKIRAN

Amongst the personnel scheduling problems, the nurse scheduling problem (HCP) is a more sensitive and important because it is specific, more difficult and related to human health. In this study, it is intended to create the best work schedule by using a simulated simulation algorithm for 15 nurses working in 2 shifts (the day shift as 8 hours between 08:00-16:00 and the night shift as 16 hours between 16:00-08:00) at a hospital operating 24 hours a day. In the study, the Simulated Annealing (SA) Algorithm, one of the heuristic methods developed for the solution of NSP, is is used as the method. In the analysis of the data, in order to make it easier to express the constraints with mathematical formulas, firstly, Microsoft Office Excel program was used to ensure that the values of the main and sub-objective functions were calculated correctly. The codes of the mathematical expressions, which are generated and tested for accuracy, are written in GNU Octave (2018) Version 4.4.1 which are free to use, developed for Windows-64 operating system and equivalent to the Matlab program. The annealing algorithm developed within the scope of the research has been rewritten in accordance with the basic logic of the general annealing algorithm but has been improved. The problem addressed by the algorithm which was expected to provide the best sheduling for the nurses by providing all 16 constraints (8 hard and 8 flexible) could be solved by the developed algorithm. The time to reach the best solution for 0-8 off (7-15 working) nurses in the given problem, on average, was calculated as 34 seconds (min.150-max.1,338 trials) in the case of 7-15 working nurses. The algorithm has also been tested for the case of 9 off (i.e. 6 working) nurses and the best solution in 195 seconds (min.456-max.4,864 iterations) providing all the constraints. The algorithm has been tested 200 times in total, reaching the best solution under the 5,000 trials (500 seconds) for the 0-9 off (i.e. 6-15 working) nurses in all tests. In the study, a new temperature reduction technique (logarithmic-exponential cooling) and a new assignment technique (using different assignment techniques above and below a certain critical values depending on the number of nurses off) were developed for the annealing algorithm. The new developed technique is called as "multiple simulated annealing with double assignment".

Keywords: Assignment Problem, Nurse Scheduling, Shift Assignment, Staff Scheduling Shift Scheduling,AnnealingAlgorithm

ÖNSÖZ

Hemşire çizelgeleme problemlerinde, çalışan hemşire sayısının az (örneğin 10'dan az), kısıt sayısının fazla (örneğin 10'dan fazla), hastanenin 24 saat kesintisiz hizmet vermesi gerektiği, günlük vardiya sayısının az (3 yerine 2 gibi), vardiya sürelerinin birbirinden farklı (örneğin gündüz 8 saat, gece 16 saat gibi), her hemşireye gece vardiyasından daha fazla gündüz vardiyası atama zorunluluğu olduğu durumlarda çözüme ulaşmak oldukça zordur. Araştırmamızın amacı; hemşirelerin çalışma sürelerini adaletli şekilde ayarlayarak hemşire memnuniyetini arttırmaktır. Çalışmamızda çok sayıda kısıt minimum sayıda çalışan hemşire ile yerine getirilerek aylık çizelge oluşturulabilmiştir.

Bu çalışma sürecinde danışmanlık yaparak bilgi ve desteğini asla esirgemeyen Sayın Prof.Dr. Mehmet Aktan'a ve tüm süreçlerde hep yanımda olup motive eden eşim ve çocuklarıma sonsuz teşekkürlerimi sunarım.

Bilgen AYAN KOÇ
KONYA-2019

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

KISALTMALAR

AF	: Amaç Fonksiyonu
CSP	: Constraint Satisfaction Problems (Kısıtlama Memnuniyet Problemleri)
EK	: Esnek Kısıtlar
GA	: Genetik Algoritma
HÇP	: Hemşire Çizelgeleme Problemi
ILP	: Integer Linear Programming (Doğrusal Tamsayı Programlama)
İK	: İnsan Kaynakları
MA	: Memetik Algoritma
NRP	: Nurse Rostering Problem (Hemşire Çizelgeleme Problemi)
NSP	: Nurse Scheduling Problem (Hemşire Çizelgeleme Problemi)
TB	: Tavlama Benzetimi (SA: Simulated Annealing)
ZK	: Zorunlu Kısıtlar

1. GİRİŞ

Bu bölümde araştırmanın konusu, önemi ve amacı, sayıltıları ve sınırlılıkları sunulacaktır.

1.1. Araştırmanın Konusu, Önemi ve Amacı

Araştırmanın konusunu hemşire çizelgeleme problemi oluşturmaktadır. Uyum sorunları, iç politikalar, hemşirelerin tatil zamanı, vardiya vb. kişisel tercihleri, hasta bakımı için farklı becerilerin bir arada olması gerekliliği, hemşire nitelikleri, hasta duyarlılık düzeyi, personel ciroları ve benzeri durumlar nedeniyle çok amaçlı bir problem oluşturan hemşire vardiyalarının çizelgelenmesi, tipik bir hastanenin ortalama toplam maliyetleri içinde %59 gibi oldukça yüksek bir oranda bulunan çalışan maliyetinin azaltılabilmesi açısından büyük önem taşımaktadır (Wang, 2016). Çalışma vardiyalarını birçok günden oluşan bir süre boyunca hemşirelere atama işi oldukça zordur. Bir organizasyon içindeki görevlerin yerine getirilmesi için, çalışanların her birini kapsayan bir program oluşturmayı gerektirir. Sağlık alanında bu problemin çözümü, farklı günlerde ve vardiyalarda bir dizi farklı personel gereksiniminin varlığı nedeniyle özellikle zordur (Kumar, Nagalakshmi ve Kumaraguru, 2014). Özellikle personel çizelgeleme problemleri içerisinde spesifik, daha zor ve insan sağlığıyla ilgili olması nedeniyle daha hassas ve önemli bir konu olan hemşire çizelgeleme probleminin önemi daha da artmaktadır.

Araştırmada, haftanın her günü 24 saat kesintisiz hizmet veren bir hastanede 2 vardiya halinde (08:00-16:00 arası 8 saat ve 16:00-08:00 arası 16 saat) çalışan 15 hemşire için varsayılan kısıtlar çerçevesinde Matlab programında tavlama benzetimi algoritması kullanılarak en iyi çalışma çizelgesinin oluşturulması amaçlanmıştır.

Araştırmaya ilişkin genel bilgi verilen birinci bölüm sonrasında ikinci bölümde kaynak araştırması yapılmıştır. Kaynak araştırması bölümünde hemşire çizelgeleme problemi ve bu problemin çözümünde kullanılan yöntem ve yaklaşımlar incelenmiştir.

Materyal ve yönteme ilişkin bilgilerin verildiği üçüncü bölümde araştırmanın yöntemi, araştırmada ele alınan problemin tanımlanması, araştırmanın amaç fonksiyonu ve varsayılan veri kısıtları ile verilerin analizine ilişkin bilgi verilmiştir.

Dördüncü bölümde araştırmanın bulguları sunularak literatürde benzer araştırmalarda elde edilen bulgularla karşılaştırılarak tartışılmıştır.

Araştırmanın son bölümünü oluşturan beşinci bölümde ise araştırmada elde edilen bulgulara dayanılarak elde edilen sonuçlar özetlenmiş, bu sonuçlar doğrultusunda birtakım önerilerde bulunularak çalışma tamamlanmıştır.

1.2. Araştırmanın Sayıltıları

Araştırmada ele alınan kısıtların bir hastanede karşılaşılan hemşire çizelgeleme sorunu için örnek alınabilecek düzeyde olduğu varsayılmıştır.

1.3. Araştırmanın Sınırlılıkları

Araştırma; bir hastanede çalışan 15 hemşireyle, hemşire çizelgeleme problemi için varsayılan kısıtlar ve amaç fonksiyonuyla, araştırmanın analizi ise Matlab programında kullanılan tavlama benzetimi algoritmasıyla sınırlıdır.

2. KAYNAK ARAŞTIRMASI

Bu bölümde hemşire çizelgeleme probleminin tanımı ve bu problemin çözümünde kullanılan yöntem ve yaklaşımlar incelenecektir.

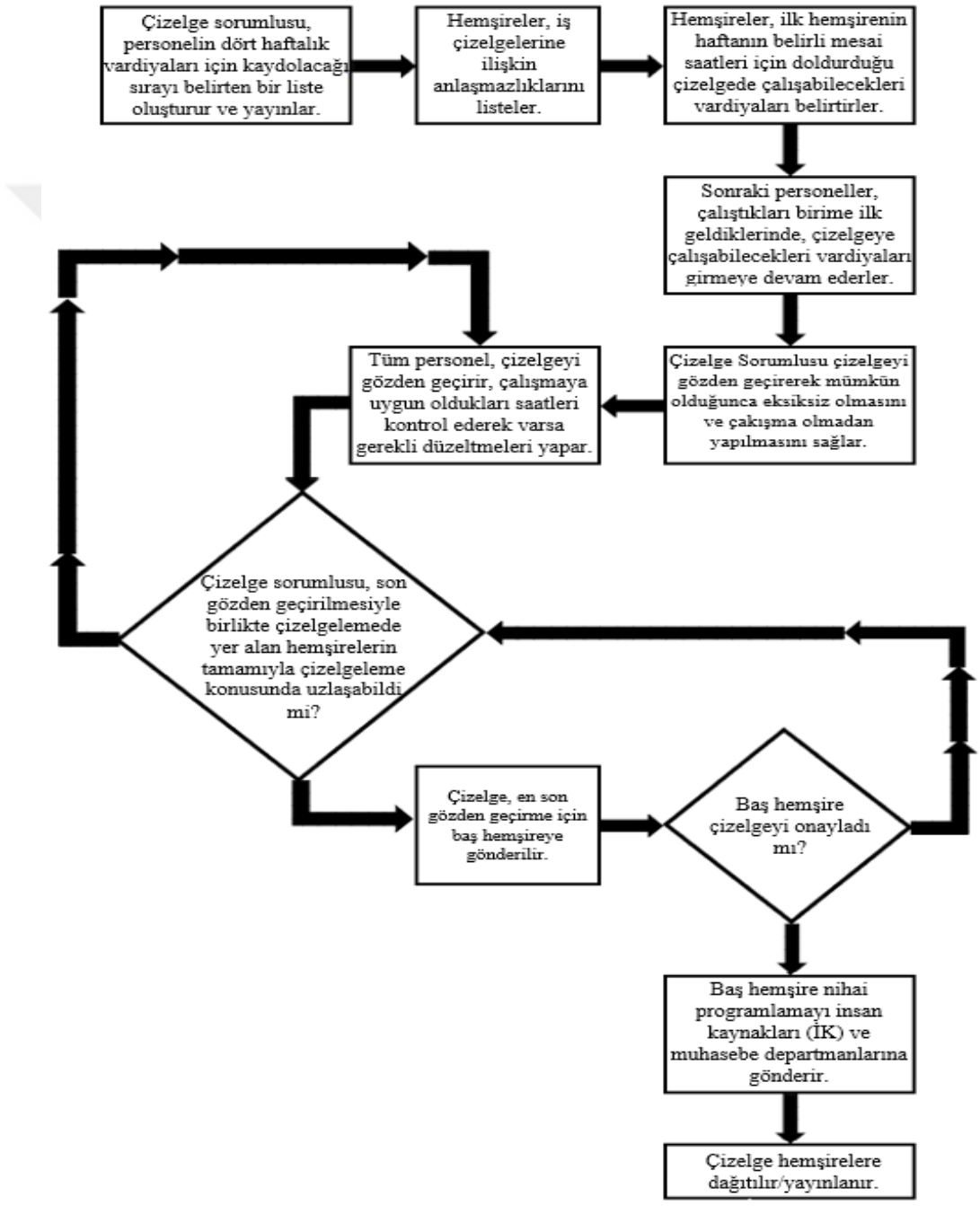
2.1. Hemşire Çizelgeleme Problemi

İngilizcede ‘nurse rostering problem’ (NRP) ya da ‘nurse scheduling problem’ (NSP) olarak geçen ve personel çizelgeleme türlerinden biri olan ‘hemşire çizelgeleme problemi’ (HÇP), uzun yıllardır ele alınan araştırma konularından biridir (Burke, Li ve Qu, 2012). Özellikle 1960'lı yıllardan itibaren sağlık personelinin bilgisayar yardımıyla mesai çizelgelemeleri üzerine çeşitli çalışmalar yayınlanmıştır (Burke vd., 2004b).

HÇP, hemşirelerin hemşire tercihleri ve hasta iş yükü gereksinimleri doğrultusunda atanması için bir karar verme aracı geliştirilmesi ihtiyacından doğmuştur (Karpuz, 2015). HÇP, bir dizi kısıtlama içeren Kısıtlama Memnuniyet Problemleri'nin (CSP: Constraint Satisfaction Problems) bir alt sınıfını temsil eder. Problem oldukça kısıtlılığa sahiptir ve çözülmesi zordur. Amaç, hemşirelerin, iş sözleşmesi kurallarına, hemşire ve sağlık kurumlarındaki işverenlerin gereksinimlerine yönelik kısıtlamaları sağlayacak şekilde yüksek kalitede vardiya atamalarını sağlamaktır. HÇP'de kısıtlamalar, önemlerine bağlı olarak zorunlu ve esnek olarak sınıflandırılmaktadır (Kundu ve Acharyya, 2008). Bu kısıtlamalar; kurumlara ve ülkelere bağlı olan yasal düzenlemelerin yanı sıra bireysel tercihlere göre önemli ölçüde farklılık gösterebilir. Uygulanabilir çözümler elde edilebilmesi için zorunlu kısıtların sağlanması gerekir. Diğer yandan zorunlu olmasa da esnek kısıtlar olması da arzu edilir, ancak esnek kısıtlar ihlâl edilebilir (Kumar vd., 2014).

Hemşire çizelgeleme, daha basit bir ifadeyle her hemşire için çalışma düzeninin belirlenmesi olarak tanımlanabilir. Bir baş hemşire tarafından manuel olarak ya da programlama için yazılım sistemleri kullanılarak yapılabilir. Çoğu hastane ünitesinde, hemşire çizelgeleri, hemşirelerden çizelgeleme sorunları hakkında bilgi toplayan ve gerektiğinde programları düzenlemeleri gereken deneyimli baş hemşireler tarafından hâlen manuel olarak geliştirilmektedir. Ancak manuel olarak adil bir programın

geliştirilmesi, göz önünde bulundurulması gereken kısıtlamalar nedeniyle zaman alıcı ve zordur. Hemşire programları tipik olarak 4 haftalık bir süre için geliştirilir. Bu programlar esnek olabilir, her ay değişebilir veya düzeltilebilir (Tekin, 2017). Döngüsel çizelgelemeler daha yaygın kullanıma sahiptir (Trilling, Guinet ve Le Magny, 2006). Sabit veya döngüsel çizelgeler genellikle iyi çözümler sunar, ancak personel tercihlerini ve dalgalanan talebi kolayca karşılayamazlar (Tekin, 2017). Şekil 1'de hemşire çizelgelemeye yönelik bir akış şeması gösterilmiştir (Creately, 2018):



Şekil 1.1. Hemşire Çizelgeleme Akış Şeması (Creately, 2017)

Şekil 1.1'e göre hemşire nöbet çizelgeleme probleminin çözümüne ilişkin temel yaklaşımı barındıran örnek akış şemasında yer alan aşamalar şunlardır:

- 1.Aşama: Çizelge sorumlusu, personelin dört haftalık vardiyaları için kaydolacağı sırayı belirten bir liste oluşturur ve yayınlar.
- 2.Aşama: Hemşireler, iş çizelgelerine ilişkin anlaşmazlıklarını listeler.
- 3.Aşama: Hemşireler, ilk hemşirenin haftanın belirli vardiyaları için doldurduğu çizelgede çalışabilecekleri vardiyaları belirtirler.
- 4.Aşama: Sonraki personeller, çalıştıkları birime ilk geldiklerinde, çizelgeye çalışabilecekleri vardiyaları girmeye devam ederler.
- 5.Aşama: Çizelge Sorumlusu çizelgeyi gözden geçirerek mümkün olduğunca eksiksiz olmasını ve çakışma olmadan yapılmasını sağlar.
- 6.Aşama: Tüm personel, çizelgeyi gözden geçirir, çalışmaya uygun oldukları saatleri kontrol ederek varsa gerekli düzeltmeleri yapar.
- 7.Aşama: Çizelge sorumlusu, son gözden geçirilmesiyle birlikte çizelgelemede yer alan hemşirelerin tamamıyla çizelgeleme konusunda uzlaşabildi mi?

  Hayır ise 6.Aşamaya geri dön.

  Evet ise 8.Aşamaya ilerle.
- 8.Aşama: Çizelge, en son gözden geçirme için baş hemşireye gönderilir.
- 9.Aşama: Baş hemşire çizelgeyi onayladı mı?

  Hayır ise 7.Aşamaya geri dön.

  Evet ise 10.Aşamaya ilerle.
- 10.Aşama: Baş hemşire nihai programlamayı insan kaynakları (İK) ve muhasebe departmanlarına gönderir.
- 11.Aşama: Çizelge hemşirelere dağıtılır/yayınlanır.

2.2. Hemşire Çizelgeleme Problemlerinin Çözümünde Kullanılan Yöntem ve Yaklaşımlar

Hemşire çizelgeleme ve HÇP'nin çözümüne yönelik literatürde birçok çalışma bulunmaktadır. Tein ve Ramli (2010), çalışmalarında, hemşire çizelgelemeye ilişkin son gelişmeleri gözden geçirmiş; Cheang vd. (2003) ve Clark vd. (2013) hemşire çizelgeleme ve yeniden çizelgeleme çalışmalarını değerlendirmişlerdir. Warner (1976) hemşire planlaması problemini ve çözümlerini ele almış; Tien ve Kamiyama (1982), Bradley ve Martin (1990), Ernst ve diğ. (2004) personel planlaması ve çizelgelemesine ilişkin kapsamlı araştırmalar yapmışlardır. Santos vd. (2015) hemşirelerin memnuniyeti sağlama; Karayel ve Atmaca (2017) hastalara kaliteli sağlık hizmeti verebilmek için hemşirelerin bölümlerde dengeli bir şekilde dağıtılmasına; Alharbi (2018) hastalara en verimli ve kaliteli hizmet verebilmeye yönelik yaklaşımlar geliştirmişlerdir.

Sitompul ve Randhawa (1990), Cheang ve diğ. (2003) ve Burke vd'nin (2004b) çalışmalarında literatürde HÇP'ye yönelik ortaya atılan çeşitli yaklaşımlar kapsamlı bir biçimde tartışılmıştır. Temel olarak, bu yaklaşımlar geleneksel matematiksel programlama yöntemlerinden (Warner ve Prawda, 1972; Beaumont, 1997; Jaumard vd., 1998) özel amaçlı sezgisel yöntemlere (Isken ve Hancock, 1990; Randhawa ve Sitompul, 1993) kadar uzanmaktadır. 2000'li yıllardan itibaren hemşire çizelgelemede en önemli araştırma yönlerinden biri meta-sezgisel yöntemlerin, özellikle evrimsel yöntemlerin incelenmesi olmuştur (Aickelin ve Dowsland, 2000, 2004). Tavlama benzetimi (Isken ve Hancock, 1990; Abramson 1991; Brusco ve Jacobs, 1993), tabu arama (Burke vd., 1998; Dowsland, 1998; Dowsland, ve Thompson, 2000; Valouxis ve Housos, 2000; Ikegami ve Niwa, 2003), memetik algoritmalar (Burke vd., 2001; Özcan, 2005), genetik algoritma (Kawanaka vd., 2001; Tanomaru, 2001; Aickelin ve Dowsland, 2004; Çivril, 2009), değişken komşuluk arama (Burke vd., 2004a) ve Bayesyen optimizasyon (Li ve Aickelin, 2006) dahil olmak üzere diğer meta-sezgisel yöntemler de araştırılmıştır. Bu ileri-sezgisel yaklaşımların birçoğu, modern hastane ortamlarında ihtiyaç duyulan karmaşıklığı ve çok çeşitli talepleri ele alan modelleri çözmeye çalışmaktadır (Burke, Li ve Qu, 2012).

Hemşire tercihlerinin ön plana alınması gerektiğini vurgulayan Constantino vd. (2014), çalışmalarında sezgisel bir yöntem ile algoritma; Wong vd. (2014) sezgisel yöntem ile algoritma, Pinheiro vd. (2015) sezgisel bir algoritma; Ramli vd. (2016) parçacık sürüsü optimizasyonu; Lim vd. (2016) özel kısıtlamalara izin veren 2 temel

model; Namoco ve Salazar (2016) genetik algoritma yöntemi; Shi ve Landa-Silva (2016) dinamik programlama; Varlı vd. (2017) hedef programlama yöntemi; Hakim ve Bakhtiar (2017) hedef programlama yöntemi; Uslu vd.(2018)hedef programlama yöntemleri kullanmışlardır.

Çizelgeleme için manuel ve otomatik yöntemler kullanılabilmekte, hemşire çizelgeleme 'döngüsel' veya 'döngüsel olmayan' çizelgeleme şeklinde düzenlenebilmektedir (Tekin, 2017). Hemşire programlaması için yaygın olarak kullanılan çözüm yaklaşımları, matematiksel programlama ve sezgisel yöntemlere dayanan çözüm yöntemleridir. Diğer yandan hemşire zamanlama problemi, pratikte talebin belirsiz olduğu stokastik bir problemdir (Karpuz, 2015).

Bu bölümde HÇP'ye yönelik geliştirilen yöntem ve yaklaşımlar 3 grupta ele alınacaktır. Bunlar; 'optimizasyon yaklaşımları', 'yapay zekâ yöntemleri' ve 'sezgisel yöntemler'dir. Bu yöntem ve yaklaşımlar arasından, meta-sezgisel yöntemlerden biri olan tavlama benzetimi algoritması, bu çalışmada ele alınan HÇP'nin çözümünde kullanılan yöntem olduğundan detaylı olarak ele alınacak olup diğer yöntem ve yaklaşımlar kısaca incelenecektir.

2.2.1. Optimizasyon yaklaşımları

Optimizasyon basitçe bir problem için en iyi çözümün bulunmasıdır (Wang, 2016). Bu bölümde; 'Tamsayılı ve doğrusal programlama yöntemleri' ile 'hedef programlama yöntemleri' gibi, analitik yaklaşımlar olarak da bilinen 'optimizasyon yaklaşımları' (Atmaca, Pehlivan, Aydoğdu ve Yakıcı, 2012) hemşire çizelgeleme konusu çerçevesinde kısaca ele alınacaktır.

2.2.1.1. Tamsayılı ve doğrusal programlama

İngilizce karşılığı 'Linear and Integer Programming' olan Tamsayılı ve Doğrusal Programlama (Tekin, 2017), Türkçe literatürde 'Doğrusal Tamsayı Programlama' (ILP: Integer Linear Programming) olarak da geçmektedir (Çoruhlu, 2007). Bunların yanı sıra yerli ve yabancı literatürde; Integer linear programming (Tamsayılı Doğrusal Programlama), Linear Programming (Doğrusal Progamlama) ve Integer Programming (Tamsayılı Programlama) olarak kullanımları da mevcuttur.

Personel çizelgelemenin optimize edilmesinde (eniyilenmesinde) tamsayılı ve doğrusal programlama, hemşirelerin farklı vardiyalara atanması ve personel gereksinimlerinin karşılanması için bir çizelge oluşturma amacıyla kullanılmaktadır. Bu yönteme göre belirli bir süre boyunca (birkaç hafta) öngörülen çalışan sayımına dayanılarak, hemşireler atanır ve son dakika personel değişikliklerine de uyumlu hale getirilir (Wang, 2016). Tüm tamsayılı ve doğrusal programlama modellerinde, kaynakların kullanılabilirliği, bir karar değişkeni biriminin kâr (veya maliyet) katkısı ve bir karar değişkeni birimi tarafından kaynakların tüketimi gibi tüm model parametreleri bilinmeli ve sabit olmalıdır (Kumar vd., 2014).

Tam sayılı ve doğrusal programlama, hemşirelerin hemşirelik departmanındaki işgücü maliyetini mümkün olan en düşük seviyeye indirmeyi sağlarken, haftalık klinik görevlerden hiçbiri açıkta kalmayacak şekilde tüm görevlerin mevcut hemşireler tarafından üstlenilmesini sağlayacak en etkili kombinasyonunu belirlemeye de yardımcı olur (Wang, 2016).

2.2.1.2. Hedef programlama

HÇP'de optimizasyon yaklaşımları arasında ele alındığı gibi ayrı bir şekilde de ele alınabilen 'Hedef Programlama' (Goal Programming), doğrusal programlamanın bir uzantısıdır. En önemli fark, hedef programlama modelinde, doğrusal programlamada mevcut olan nesnenin doğrudan hedeflenmesinin (maksimize/minimize etme) bulunmamasıdır. Bunun yerine, istenen hedefler ve gerçekleşen sonuçlar arasındaki sapmaları en aza indirmeye çalışılır. Bu hedeflere, belirli bir hiyerarşi içinde öncelik verilir. Önemli birçok kritere sahip bir optimizasyon türü olan hedef programlama, çoklu hedefler yoluyla problemleri çözmek için geliştirilmiştir (Jedicke, Wilbur ve Rifai, 1994).

Matematiksel programlarının çoğu aşırı basitleştirici varsayımların bir sonucu olarak sınırlandırılmış bir dizi kısıtlamaya tabi olan tek bir amaç fonksiyonuna sahipken, birçok matematiksel model, hedef programlamaya veya çok amaçlı karar almaya dayanmaktadır (Tekin, 2017). Hedef programlama modeli, Trivedi vd. (1981), Özkarahan, Bailey vd. (1988), Azaieza ve Sharif (2005), Tekin (2017) gibi araştırmacılar tarafından hemşire çizelgeleme problemlerini çözmede kullanılmıştır. Trivedi'nin 1981 yılında yayınlanan çalışmasında tamsayılı ve hedef programlamadan

oluşan ‘karma hedef programlama modeli’yle bir hastanedeki hemşirelerin gider bütçesi eniyilenmiştir. Özkarahan ve Bailey (1988), hedef programlama yaklaşımında üç temel amaç fonksiyonunu önermişlerdir. İlk amaç, planlanan hemşirelerin sayısı ile talep arasındaki günün sapmasını en aza indirmektir (Bu, günün her saati için zaman çizelgesi olarak adlandırılmaktadır). İkinci hedef, iş sözleşmesine göre planlamaya dayalı olarak iş örüntüleri üzerindeki günlerin toplamı ile iş gücünün büyüklüğü arasındaki sapmaları en aza indirmektir (Bu, haftanın günü planlaması olarak adlandırılmaktadır). Üçüncü hedef ise haftanın gününü ve zaman çizelgeleme sorunlarını birleştirmektedir. Azaieza ve Al Sharif’in (2005) çalışmalarında hemşire çizelgeleme probleminin çözümü için 0-1 hedef programlamayı uygulayarak hemşirelerin beceri ve yeterliliklerinin yanı sıra servis devamlılığı ve gereksiz mesai yükünün yaratacağı ek maliyetleri azaltmayı amaçlayan bir model geliştirmişlerdir. Tekin (2017) çalışmasında, hedef programlamayla ameliyathanelerde görev yapan hemşirelerin çizelgeleme probleminin çözümünü ele almış; GAMS programı üzerinde bütün kısıtlamaları içeren çok amaçlı bir tamsayı programı oluşturmuş ve HÇP’nin çözümü hedef programlama modeliyle CPLEX kullanarak gerçekleştirmiştir.

2.2.2. Yapay zekâ yöntemleri

Yapay zekâ yöntemleriyle çeşitli eğitim ve hemşire çizelgeleme problemlerinin çözülmesinde literatürdeki çalışmalarda genetik algoritma (Abramson ve Abela, 1991), tabu arama (Hertz, 1992; Costa, 1994), kısıt sağlama problemi (Brittan ve Farlet, 1971; Deris, Omatu, Ohta ve Samat, 1997; Abdennadher ve Marte, 2000) ve tavlama benzetimi (Abramson 1991) gibi çeşitli meta-sezgisel yöntemler kullanılmıştır.

2.2.3. Sezgisel yöntemler

Sezgisel optimizasyon yöntemleri, çok çeşitli karmaşık sistemlerin tasarlanması veya yönetilmesi için optimal kararları bulmada kullanılabilecek yapay zekâ arama yöntemlerinin kullanıldığı modern optimizasyon teknikleridir (Tafida, 2014). HÇP’nin çözümünde kullanılan sezgisel yöntemler; Genetik Algoritma, Tabu Arama, Memetik Algoritma ve Tavlama Benzetimi olmak üzere 4 grupta toplanabilir. Bu bölümde bu sezgisel yöntemler kısaca ele alınacaktır.

2.2.3.1. Genetik Algoritma

Temel mantığı kalıtım, mutasyon ve doğal seçim gibi evrimsel biyolojinin ilkelerine dayanan Genetik Algoritmalar, doğal seçim sürecini taklit ederek çok geniş arama alanlarına sahip karmaşık optimizasyon problemlerini çözmede kullanılabilir (Aycan, 2008). Bu algoritma basitçe şu şekilde işlemektedir: Bir rastgele çizelge kümesi (popülasyonu) oluşturularak bir genetik algoritma başlatılır. Bunlar daha sonra bazı kriterlere göre değerlendirilir. Bu değerlendirmeyi temel alarak çizelgeler, sonraki zaman çizelgeleri için ebeveynler olarak seçilir, yani bir deyişle başlangıçta rastgele oluşturulan bu çizelgeler sonradan oluşturulacak daha iyi çizelgeler için bir başlangıç noktası oluştururlar. Seçim sürecinde daha iyi zaman çizelgeleri seçilip daha kötü olanlar ortadan kaldırılırken, aynı zamanda arama, arama alanının en iyi sonucu veren alanlarına doğru yönlendirilmiş olur (Burke vd., 1994).

Kawanaka vd. (2001), Tanomaru (2001), Aickelin ve Dowsland (2004) ve Çivril'in (2009) çalışmaları, hemşire çizelgeleme probleminin çözümünde genetik algoritma kullanılan çalışmalara örnek gösterilebilir.

2.2.3.2. Tabu Arama

Ekseriyetle HÇP'de kullanılan yaklaşımlardan biri olan Tabu Arama meta-sezgisel yöntemler arasında yer almaktadır. Optimum ya da optimuma yakın çözümlerin bulunmasında çözüm uzayını araştırmaya dayanan bu yöntemde çözüm arayışı rastgele bir ilk çözümle başlamaktadır. Bu ilk çözüme ilişkin tanımı yapılan bir hareket mekanizması ile ilk çözüme komşuluk oluşturularak bu komşuluğun içerisinden optimum amaç değerini sağlayan çözümün tabu sınıfına girmemesi halinde ikinci bir çözüm olarak kabul edilmektedir (Eren ve Güner, 2007). Bu yöntemde, tabu sınıflarının belirlenmesinde kısa dönemler için geçerli olacak bir tabu listesi (hafıza) kullanılmaktadır. Belirli bir tekrarlama veya çözümde iyileşme sağlanamadığında aramaya son verilir (Karaatlı, 2010).

Tabu arama (tabu search), çok seviyeli bellek yönetimi ve tepki araştırmasına dayalı genel optimizasyon problemlerinde uygulanabilmektedir. Glover (1986), Tabu Arama'yı "başka bir sezgisel yönteme bindirilmiş bir meta-sezgisel" şeklinde tanımlamıştır. Bu yöntem Hertz (1992) ve Costa (1994) tarafından zaman çizelgeleme

problemlerine uygulanmaktadır. Ancak tabu arama, büyük bir çizelgeleme problemi alanı için çok uygun bir teknik olarak kabul edilmemektedir (Aycan, 2008).

Burke vd, (1998), Dowsland (1998), Dowsland ve Thompson (2000), Valouxis ve Housos (2000) ve Ikegami ve Niwa'nın (2003) çalışmaları, hemşire çizelgeleme probleminin çözümünde tabu arama yöntemini kullanan çalışmalara örnek gösterilebilir.

2.2.3.3. Memetik Algoritma

Memetik Algoritmalar (MA), Genetik Algoritma (GA)'lara benzeyen ancak yerel arama tekniklerini kullanmasıyla melez (hibrid) yapıya sahip, evrim yapısı üzerine kurulu, nesillere dayalı sezgisel arama yapan algoritmalar olup (Türkbey, 2002), farklı türden gelecek vaat eden genetik operatörleri ve kendine özel uyarlanmış bir ihlali yönlendiren hiyerarşik tepe tırmanma yöntemini kullanmasıyla bilinir (Özcan, 2005).

Hemşire çizelgelemede memetik algoritmaları kullanan çalışmalara örnek olarak Burke vd. (2001) ve Özcan'ın (2005) araştırmaları gösterilebilir.

2.2.3.4. Tavlama Benzetimi

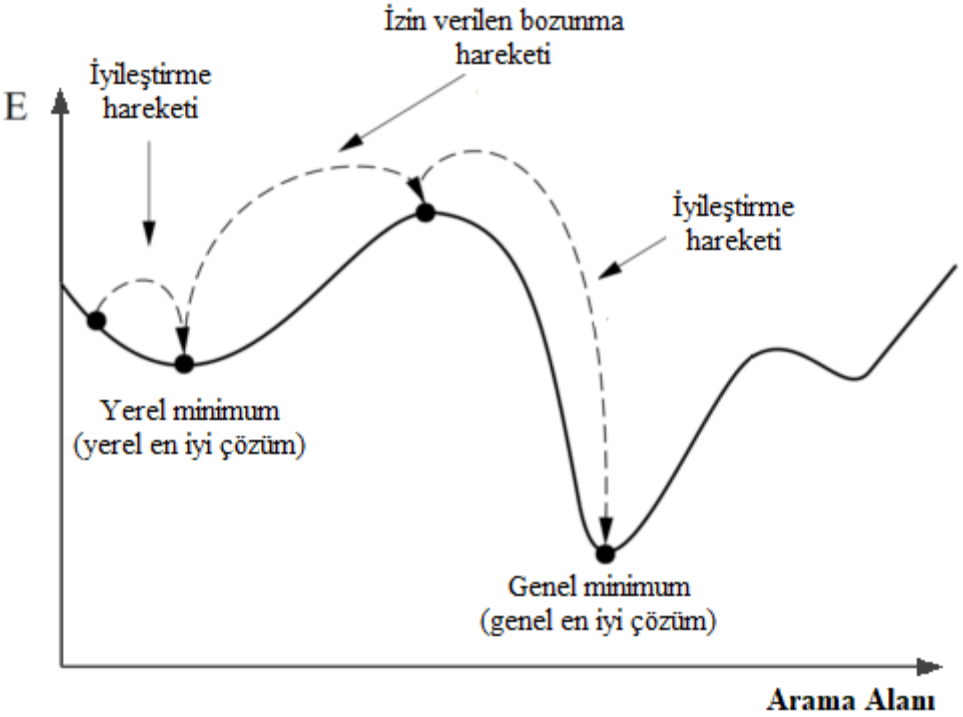
Türkçe literatürde Tavlama Benzetimi, Benzetimli Tavlama, Benzetilmiş Tavlama, Tavlama Benzeştirmesi gibi farklı tanımlamaları olan bu algoritma İngilizcede 'Simulated Annealing' olarak geçmektedir.

Tavlama Benzetimi (TB) genel minimum iç enerji konfigürasyonuna ulaşmak amacıyla tavlama işlemini kullanarak büyüyen kristallerin istatistiksel sürecinin simülasyonuna dayanan stokastik bir yaklaşımdır (Arora, 2004). Daha açık bir ifadeyle katı bir malzemenin önce belirli bir sıcaklığa kadar ısıtılıp sonra soğutulması yoluyla malzemenin kalitesinin arttırılmasını sağlayan fiziksel tavlama sürecinin taklidine dayanmaktadır (Le, 2011). Bu yaklaşımda, metal gibi bir katı madde yüksek bir sıcaklığa kadar ısıtılır ve erimiş bir hale getirilir. Atomlar, bu durumda özgürce hareket edebilirler. Bununla birlikte, atomların hareketleri sıcaklığın düşürülmesi yoluyla kısıtlanır. Sıcaklık azaldıkça, atomlar daha az hareket etmeye eğilimlidir ve kristal formları mümkün olan en az dahili enerjiye sahiptir. Kristallerin oluşumu temelde soğutma oranı ile ilgilidir. Erimiş metallerin soğutma hızı çok hızlı olduğunda, bu sıcaklığın çok hızlı bir şekilde azaldığı anlamına geleceğinden kristal halini elde

edemeyebilir. Bunun yerine, kristal duruma kıyasla daha yüksek bir enerji durumuna sahip olan bir polikristalin durumuna ulaşabilir. Mühendislik uygulamalarında, hızlı soğutmanın sonunda malzemenin içinde kusurlar meydana gelir. Bu nedenle en düşük enerji durumuna (iç enerji) ulaşmak için ısıtılmış katı (erimiş metal) sıcaklığının yavaş ve kontrollü bir oranda azaltılması gerekir. Yavaş bir oranda bu soğutma, tavlama olarak adlandırılır (Rao, 2009).

Tavlama benzetiminde soğutma işleminin kontrolünü, bir diğer deyişle hangi hızda yapılacağını ayarlamayı sağlayan fonksiyon, optimum çözümün aranması esnasında daha iyi olmasalar da genel en iyi çözümün bulunmasına yardımcı olacak çözümlerin kabul edilebilirliğine ilişkin olasılığı ifade eder. Aramaya ilk başlandığında bu olasılığın daha iyi olmayan çözümlerin kabul edildiği ve bu doğrultuda yerel en iyi çözümlerden kaçınarak genel en iyi çözümü bulmayı sağlayan yeterince yüksek bir değerde olacak şekilde ayarlanması gerekir. En iyi çözümün aranması süresince, olasılığın gittikçe azaldığı dolayısıyla yerel en iyi çözümden kaçmanın zorlaştığı görülür. Burada amaçlanan, eldeki yerel en iyi çözümün komşuları aranarak yeni bir yerel en iyi çözüme geçmektense genel en iyi çözüme ulaşılmaya çalışılmasıdır (Le, 2011).

Tavlama benzetimi algoritması rastgele başlangıç çözümüyle başlar ve her yinelemede yerel komşulukla sonraki çözümü üretir. Sistemin enerji düzeyini temsil eden 'E' amaç fonksiyonunu geliştiren (maddenin/sistemin enerjisini azaltan) yani eniyileyen yeni çözüm her zaman kabul edilir. Öte yandan, sistemin sıcaklığını arttırmaya müsade eden ya da sistemdeki amaç fonksiyonundan uzaklaşmaya/bozunmaya belirli bir düzeyde izin veren geçici çözüm önerileri de kabul edilmektedir. Algoritmanın temel işleyişi Şekil 2.1'deki gibi gösterilebilir (Rosocha, Vernerova ve Verner, 2015).



Şekil 2.1. Tavlama Benzetimi Algoritmasının Temel İşleyişi (Rosocha, Vernerova ve Verner, 2015)

Mevcut ve yeni en iyi çözüm arasındaki enerji farkı *(ΔE)*, amaç fonksiyonunun mevcut çözümün komşuluğu yoluyla oluşturulan yeni çözümü ile amaç fonksiyonunun mevcut değeri arasındaki farkı, yani ilk bulunan en iyi çözüm ile sonrasında bulunan en iyi çözüm arasındaki farkı gösterir. Daha iyi olmayan çözümün kabul edilme olasılığı Denklem 2.1'de gösterildiği gibi Boltzmann dağılımına dayanır:

$$P(\Delta E, T) = e^{\frac{\Delta E}{T}} = e^{\frac{E(n+1) - E(n)}{T}} \quad (2.1)$$

Burada; *E(n)* mevcut çözümün amaç fonksiyonunu (mevcut enerji düzeyini), *E(n+1)* yerel komşulukla elde edilen yeni çözümün enerji düzeyini, *T* ise sistemin gerçek sıcaklığını temsil eder. *T* değeri yani sıcaklık arttığında *P* değeri, yani daha iyi olmayan bir komşu çözümün kabul edilme olasılığı azalır. Bu noktada belirli bir sıcaklık seviyesinde toplamda kaç yeni çözümün araştırılacağı araştırmacı tarafından belirlenir. Ancak kararlı bir duruma ulaşıldığında (çözümlerde daha fazla ilerleme olmadığı durumlarda), sıcaklık genellikle azaltılmış olur. Bu yöntemde hem mevcut hem de en iyi elde edilen çözümler kaydedilmelidir (Rosocha vd., 2015). *T* yerine T_k'nın gerçek sıcaklık, T_0'ın ilk sıcaklık ve *k*'nın kontrol parametresi olduğu $T_k = T_0/logk$

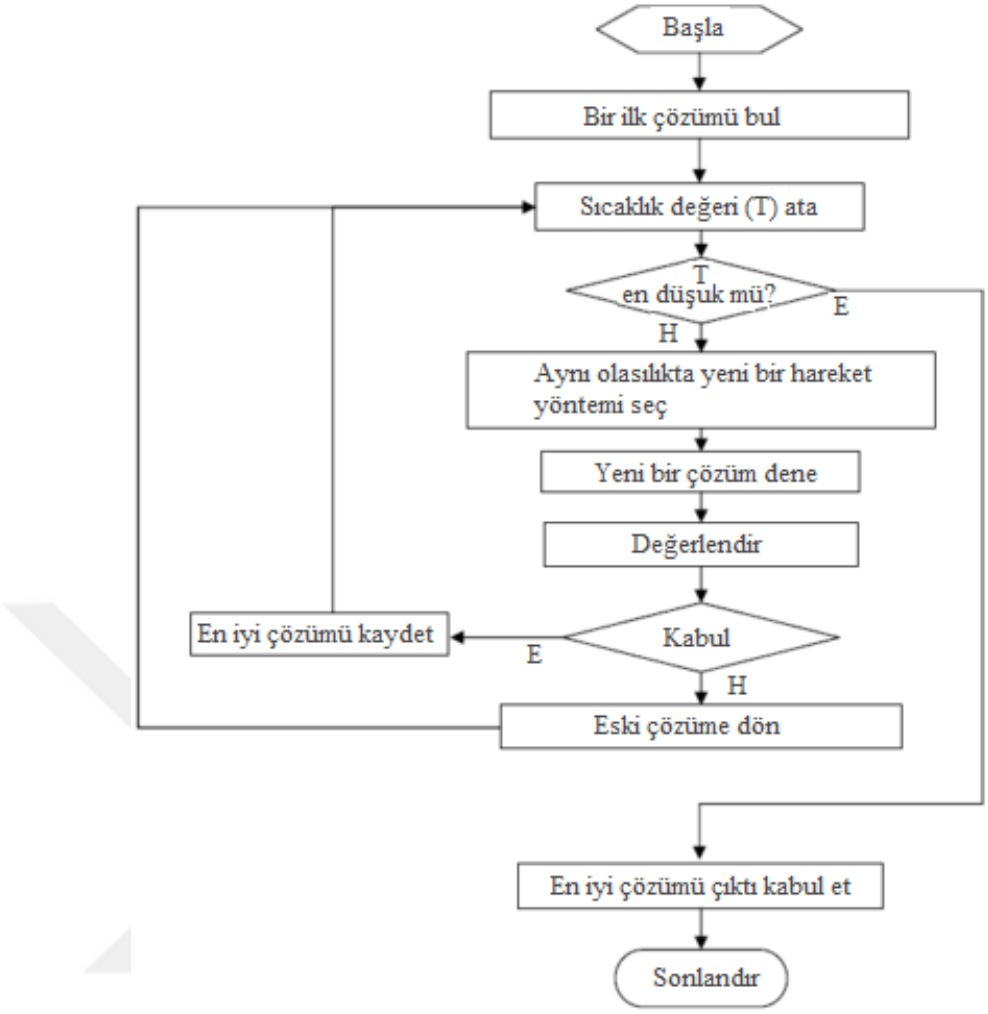
logaritmik soğutma kullanılması durumunda, tavlama benzetimi algoritmalarının genel en iyi çözüme yakınlaştığı kanıtlanmıştır (Aarts, Korst ve Michiels, 2007).

Daha iyi olmayan çözümleri kabul etme özelliği sayesinde, Tavlama Benzetimi prosedürünün temel faydası, yerel en iyi çözümden kaçabilme imkânıdır. Daha kötü çözümlere hareket etme olasılığı sıcaklığın yüksek olduğu durumda artarken, hedef fonksiyonun (E) bozunmasıyla birlikte azalır. En kötü alternatifi kabul etme olasılığı 1/1+ *P(ΔE,T)>Z olup burada P(ΔE,T)=[e^(ΔE/kT)*, ΔE amaç fonksiyonundaki fark, T mevcut sıcaklık, k Boltzmann sabiti ve Z ise 0 ile 1 arasında rastgele oluşturulmuş bir sayıdır. İlk aşamada, sistemin sıcaklığı yüksek olduğunda bozunma ya da daha kötü sonuçlara gitme olasılığı daha yüksekken sıcaklık düşüşüyle birlikte algoritma daha sert hale gelir ve sadece çözüm kalitesindeki daha küçük düşüşlere izin verir. Soğutma hızı seçimi, tavlamanın davranışını yönetir ve tekrarlamalar sırasında sıcaklığın hareketlerini belirleyerek yakınsamasında ilerler. Tavlama benzetimi algoritmasının performansı, başlangıç sıcaklığı, sonlandırma kriterleri ve soğutma fonksiyonu gibi soğutma seçiminde yer alan değişkenlere büyük ölçüde bağımlıdır. Örneğin, başlangıç sıcaklığı T_0 çok yüksekse, algoritma rastgele yerel aramaya daha yatkındır ve tam tersi yani düşük sıcaklıkta ise yerel iyileştirmeler için basit aramaya yönelir (Rosocha vd., 2015). Enerji farkının amaç fonksiyonu olarak ifadesi Denklem 2.2'deki gibidir:

$$\Delta E = \Delta f = f_{i+1} - f_i \equiv f(x_{i+1}) - f(x_i) \tag{2.2}$$

Her bir sıcaklıkta termal dengenin elde edilmesini simüle etmek için önceden belirlenmiş bir sayıda yeni nokta x_{i+1}, sıcaklık T'nin herhangi bir spesifik değerinde test edilir. Başlangıç sıcaklığı T, Tavlama Benzetimi algoritmasının başarılı bir şekilde yakınsamasında önemli bir rol oynar. Örneğin, başlangıç sıcaklığı *(T)* çok yüksekse, yakınsama için daha fazla sayıda sıcaklık düşüşüne ihtiyaç duyar. Aksine, başlangıç sıcaklığı çok düşük olacak şekilde seçilirse, arama eksik olabilir ve algoritma yerel minimumda takılabilir, yani genel minimumu bulamayabilir (Gülmez, 2014).

Tavlama Benzetimi algoritmasınn tipik akış şeması Şekil 2.2'deki gibi gösterilebilir (Gülmez, 2014).



Şekil 3.2. Tavlama Benzetimi Algoritmasının Tipik Akış Şeması (Gülmez, 2014)

Bu doğrultuda Tavlama Benzetimi algoritması, yöntemi basitleştirmek adına aşağıdaki şekilde adım adım yazılabilir (Rao, 2009):

1. Bir ilk vektör x_1 ile başlanır (f_{xi}) ve amaç fonksiyonu değeri en iyi kabul edilir. Ardından fonksiyona yüksek bir sıcaklık değeri (T_{max}) atanır.

2. Rastgele yeni bir çözüm $f(x_{(i+1)})$ oluşturularak yeni çözüm ile mevcut fonksiyon değeri arasındaki fark bulunur. Bunun için $\Delta f=f_{(i+1)}-f_i=f(x_{(i+1)})-f(x_i)$ hesaplanır.

3. Yeni çözümün mevcut çözümden daha iyi olup olmadığı belirlenir. Eğer fark (Δf) istenen yöndeyse yani yeni çözüm daha iyiyse kabul edilir, değilse bir sonraki aşamaya geçilir (Bazı durumlarda minimize etme bazı durumlarda ise maksimize etme amaçlandığından minimizasyon durumunda farkın negatif değerde olması yani yeni

bulunan çözümün daha düşük değerde olması, maksimizasyon durumunda ise farkın pozitif değerde olması yani yeni bulunan çözümün daha yüksek değerde olması istenir).

4. $1/(1+e^{(-\Delta f/T)})$ değeri rastgele üretilen bir sayının (Z) değerinden büyükse yeni çözüm kabul edilir, değilse eski çözüm halen en iyi çözüm olmaya devam eder.

5. x_{i+1} noktası reddedilirse eski çözüm yani (f_{xi}) halen en iyi çözüm olmaya devam eder, ardından algoritma tarafından rastgele yeni bir çözüm x_{i+1} üretilir (Algoritmanın kabul olasılığına göre daha kötü bir noktayı kabul edebileceği akılda bulundurulmalıdır).

Örneğin başlangıçta amaç fonksiyonu değeri 8.000 (E_n) olan ve sıcaklığı 10.000 (T_0) olarak belirlenmiş bir sisteme tavlama algoritması uygulandığını düşünelim. Sıcaklık azaltma fonksiyonu da üstel azaltma yapan $T(n+1)=T_0*0,95k$ olsun (Burada k deneme sayısını ifade edecektir). Varsayalım ilk denemede amaç fonksiyonu değeri 6.000 bulunmuş olsun. Bu durumda yeni bulunan çözüm olan 6.000 değeri 8.000'den daha iyi olduğu için yeni en iyi çözüm olarak kabul edilir. İkinci denemede ise yeni bulunan değerin 7.000 $(E_{(n+1)})$ olduğunu varsayalım. Yani bulunan değer daha önce bulunan en iyi değer olan 6.000'den daha kötüdür. Amaç fonksiyonunun değerinin optimize edilmesi yani en iyilenmesi yani minimize edilmesi istenmesine karşın tavlama algoritması genel en iyiyi bulmak için zaman zaman daha yüksek değerleri de kabul edebilmektedir. Burada yeni bulunan daha kötü değerin kabul edilebilme olasılığı verilen örneğe göre $1/(1+e^{(\Delta E/T)})$ ile hesaplandığında $1/(1+e^{((E_n- E_{n+1})/T)})$'den $1/(1+e^{(-(6000-7000)/1.0000)})$'den 0,4750 olarak bulunur. Bu değer $P(\Delta E,T)$ değeridir, yani yeni çözümün kabul edilme olasılığını göstermektedir. Sistemde 0 ile 1 arasında değişen rastgele bir Z değeri üretildiğinde ortalama 0,50 gelme olasılığı yüksektir. Varsayalım sistemde rastgele üretilen Z değeri 0,49 şeklinde oluştu. Bu durumda $P<Z$ olduğu için yeni çözüm kabul edilmez ve amaç fonksiyonu için en iyi çözüm halen 6.000 olarak kabul edilerek bir sonraki denemeye geçilir. Eğer sistemde ratsgele oluşturulan Z değeri 0,4750'den düşük olsaydı o zaman yeni çözüm en iyi çözüm olan 6000'den 1.000 fazla olmasına karşın kabul edilecek ve sonraki denemelere geçilecekti. Yani genel en iyiye ulaşmak için daha kötü bir çözüm kabul edilmiş olacaktı. İlk deneme sonunda T sıcaklık değeri $T(n+1)=T_0*0,95k$ *formülünden* $T_0=10.000$ *ve* $k=1$ *için* $T(n+1)=10.000*0,95*1=9.500$ bulunacaktır. Sonraki deneme için T değeri 10.000 yerine 9.500 olarak kullanılacaktır. Bu aşamada yeni denemede yeni çözümün yine 7.000 bulunduğunu varsayalım. Daha önce 0,4750 olarak hesaplanan yeni çözümün

kabul edilme olasılığı bu kez $1/(1+e^{((E_n- E_{n+1})/T)})$ den $1/(1+e^{(-(6000-7000)/9500)})$ den 0,4737 olarak bulunur, yani deneme sayısı arttıkça sıcaklık düşeceğinden yeni çözümün kabul edilme olasılığı da azalmaktadır. Bu durumda daha önce bulunan 0,4750 oranındaki yeni çözümün kabul edilme olasılığı ile aynı olasılığa sahip yeni bir çözüm olabilmesi için yeni çözümün değerinin 7000 değil 6950 olması gerekecektir (6000+ln((1/0,4750)-1)*9500=6950). Bir diğer deyişle deneme sayısı arttıkça sıcaklık düşmekte sıcaklık düştükçe aynı olasılıkta seçilme ihtimali için yeni çözüm değerlerinin önceki çözümle daha az farka sahip olması gerekmektedir. Deneme sayısı arttıkça hem sıcaklık değeri ve hem de daha kötü olan yeni çözümün kabul edilme olasılığı 0'a yaklaşmaktadır. Daha kötü olan yeni çözüm değerlerinin kabul edilme olasılığı hiçbir zaman 0,50'nin üzerinde olamamaktadır. Örneğin mevcut çözüm 6.000 ise ve yeni çözüm de 6.000 ise yeni çözümün kabul edilme olasılığı $1/(1+e^{((E_n-E_{n+1})/T)})$ den $1/(1+e^{(-(6000-6000)/9500)})$ den ½ yani 0,50 olarak bulunur.

Burke, Li ve Qu (2012) tarafından çok amaçlı hemşire nöbet çizelgelemesi probleminin çözümüne yönelik tavlama benzetimi yaklaşımı sunulmuştur. Öncelikle çizelgelemede yer alan tüm hemşirelerin tüm vardiyaları için yasalara uygun vardiya kalıpları oluşturulmuş; sonrasında hemşirelerin her birinin bu vardiya kalıplarına atanması sağlanarak zorunlu kısıtların tamamını sağlamaya yönelik çözüme uyarlanabilecek bir yapı sağlanmıştır. Bunu takiben esnek kısıtların karşılanabilmesi amacıyla ağırlıklı toplama ve çeşitlendirmeyi değerlendirmeye dayanan iki seçeneğe sahip bir fonksiyon kullanılmıştır. Burke vd.'nin (2012) elde ettikleri sonuçlar, önerdikleri yöntemin modern hastanelerdeki hemşire çizelgelemelerinde rahatlıkla uygulanabileceğini göstermektedir.

Isken ve Hancock (1990), Brusco ve Jacobs (1993), Kundu, Mahato, Mahanty ve Acharyya (2008), ve Ko, Kim, Jeong, Jeon, Uhmn ve Kim'in (2013a, b) çalışmaları, hemşire çizelgeleme probleminin çözümünde Tavlama Benzetimi algoritmasını kullanılan diğer çalışmalara örnek gösterilebilir.

3. MATERYAL VE YÖNTEM

Bu bölümde araştırmanın yöntemi, araştırmanın probleminin tanımlanması, araştırmanın veri kısıtları, problemin çözüm aşamaları ve verilerin analizi konuları ele alınacaktır.

3.1. Araştırmanın Yöntemi

Araştırmada yöntem olarak, HÇP'nin çözümüne yönelik geliştirilen sezgisel yöntemlerden biri olan Tavlama Benzetimi Algoritması (TBA) kullanılmıştır. Bu algoritmanın özellikleri Bölüm 2.2.3.4'te detaylı olarak açıklanmıştır.

3.2. Araştırmanın Probleminin Tanımlanması

Araştırmanın problemi; haftanın her günü 24 saat kesintisiz hizmet veren bir hastanede 2 vardiya halinde (08:00-16:00 arası 8 saat ve 16:00-08:00 arası 16 saat) çalışan 15 hemşire için varsayılan kısıtlar çerçevesinde Matlab programında tavlama benzetimi algoritması kullanılarak amaç fonksiyonlarını sağlayan en iyi çalışma çizelgesinin oluşturulmasıdır.

3.3. Araştırma Probleminin Amaç Fonksiyonu ve Veri Kısıtları

Araştırmada ele alınan hemşire nöbet çizelgeleme probleminin amaç fonksiyonu aşağıda verilmiştir:

1) Verilen kısıtlar doğrultusunda hemşireler için en iyi çalışma çizelgelerinin oluşturulması.
2) Her hemşirenin toplam atandığı vardiyaların mümkün olduğunca eşit olması.

Araştırmada ele alınan 15 hemşire için 2 vardiyaya atamanın mümkün olduğu nöbet çizelgeleme probleminin veri kısıtları ise zorunlu (sabit, katı) kısıtlar (ZK) ve esnek kısıtlar EK) olmak üzere iki grupta toplanmıştır.

Zorunlu kısıtlar (ZK) 8 adettir:

1) Günde 24 saat hizmet veriliyor.
2) Vardiya saatleri: Gündüz (08.00-16.00), Gece (16.00-08.00)
3) Haftada en az 40 saat çalışılır.
4) Hemşirelerin ardışık olarak en fazla 2 gece vardiyasında çalışmasına izin verilir.
5) Bir hemşire gece vardiyasından sonraki gün gündüz vardiyasına atanamaz.
6) Bir hemşire bir gün içinde yalnızca bir vardiyada çalışabilir.
7) Sorumlu hemşire gece nöbeti tutmaz ve hafta sonu çalışmaz.
8) 50 yaşın üstünde olan ve gebe olan hemşire nöbet tutmaz.

Esnek Kısıtlar (EK) 8 adettir:

1) Bir hafta içerisinde en az bir kere ardışık 48 saat izin yapılmalıdır.
2) Her hemşireye eşit sayıda hafta sonu tatili atanmalıdır.
3) Her hemşireye atanan toplam gündüz vardiyaları sayısı gece vardiyaları toplamından büyük veya eşit olmalıdır.
4) Personelin mazeret belirttiği günlere nöbet yazılmaz.
5) Haftada ardışık 72 saat veya daha fazla izin kullanmamalı.
6) Planlama boyunca ihtiyaç duyulan haftalık hemşire sayısı hep sabit.
7) Aynı hemşire hafta sonu 2 gün çalışmamalıdır.
8) Acil servisin yoğun olduğu akşam 19.00-23.00 saatleri arasında 4 saatlik vardiya dışı mesai yapılabilir (bu kısıt her gün için atama sağlayacak ama mesaiye çağırma görevi o akşam çalışan acil servis hekimi tarafından sağlanacak. Yoğun olmayan günlerde çağırılmayacak).

3.4. Verilerin Analizi

Verilerin analizinde kısıtların matematiksel formüllerle ifade edilebilmesini kolaylaştırmak, ana amaç ve alt amaç fonksiyonlarının değerlerinin doğru olarak hesaplandığından emin olmak için öncelikle Microsoft Office Excel programından faydalanılmıştır. Daha sonra oluşturulan ve doğruluğu test edilen matematiksel ifadelerin kodları Matlab programıyla eşdeğer olup kullanımı ücretsiz olan GNU Octave (2018) Version 4.4.1'in Windows-64 işletim sistemi için 244 MB ebatındaki versiyonunda yazılmıştır. Scriptin tamamının tek bir dosyada gösterilebilmesi için "function" özelliği kullanılmamış, tüm ana amaç ve alt amaç fonksiyonları ayrı ayrı yazılıp her biri kontrol edildikten ve doğru çalıştıklarından emin olunduktan sonra birer birer birleştirilmiş, her birleştirme sonrasında tekrar kontrol edilmişlerdir.

4. ARAŞTIRMA BULGULARI VE TARTIŞMA

Bu bölümde araştırma bulguları sunulacak, literatürde benzer araştırmalarda elde edilen bulgularla karşılaştırılarak tartışılacaktır.

4.1. Araştırma Bulguları

Bu bölümde; araştırmada ele alınan problemin çözümüne yönelik olarak alt amaç fonksiyonlarının matematiksel olarak ifade edilmesi ve bunlara uygun Matlab kodlarının yazılmasına ilişkin bulgular, genel tavlama algoritmasının çözüme ulaşmayı sağlamada yetersiz kalması nedeniyle çifte atamalı çoklu tavlama algoritmasının geliştirilmesine ilişkin bulgular, tavlama algoritmasında tek bir atama yöntemi yerine amaç fonksiyonunun kritik bir değerinin altında ve üstündeki durumlarda iki farklı atama yönteminin kullanılmasına yönelik geliştirilen algoritmaya ilişkin bulgular ayrı başlıklar halinde sunulacaktır.

4.1.1. Alt amaç fonksiyonlarının matematiksel olarak ifade edilmesi ve bunlara uygun Matlab kodlarının yazılması

Problemin çözümü için ilk aşamada amaç fonksiyonları (AF1-2), zorunlu kısıtlar (ZK1-8) ve esnek kısıtlar (EK1-8) tanımlanarak mümkün olduğu kadar matematiksel ifadeleri oluşturuldu ve excelde örnek bir çizelge oluşturularak kontrol edildi. Matematiksel ifadeleri oluşturulamayanların doğrudan excelde excel formülleriyle hesaplamaları yapıldı. Tüm kısıtlara uyan bir çizelge oluşturulup exceldeki tüm formüllerin amaç fonksiyonları ve kısıtlar için doğru sonucu verip vermediği kontrol edildi. Ardından her biri için Matlab programında ayrı ayrı kodlar yazıldı ve sonunda bu kodlar birleştirilerek tek bir script haline getirildi.

Amaç Fonksiyonları (AF1-2):

AF1 - Verilen kısıtlar doğrultusunda hemşireler için en iyi çalışma çizelgelerinin oluşturulması: Bu amaç fonksiyonu hemşirelerin nöbet çizelgelemesinin tüm zorunlu ve esnek kısıtlara uygun olarak optimum düzeyde yapılmasını gerektirdiği için, hedeflenen en iyi durum tüm kısıtlara uyulan bir çizelgeleme yapılabilmesidir.

AF2 - Her hemşirenin toplam atandığı vardiyaların mümkün olduğunca eşit olması: Bu amaç fonksiyonu, yapılacak çizelgelemenin kaç haftalık olacağına karar verilmesi sonrasında çizelgenin tamamında her hemşirenin atandığı gündüz vardiya sayısı ve gece vardiya sayısı toplamının mümkün olduğunca eşit olmasını hedeflemektedir.

$$VSH_1 = VSH_2 = \cdots = VSH_i \quad \text{burada i=1,2,…,m ve m=15} \tag{3.1}$$

Denklem 3.1'de VS 'Vardiya Sayısı'nı, H ise 'Hemşire'yi ifade etmektedir. Her hemşire için toplam vardiya sayısının hesaplaması ise Denklem 3.2'de gösterilmiştir.

$$VSH_i = \sum_{j=1}^{n} VSH_{ij} \quad \text{burada i=1,2,…,m ve j=1,2,…,n ve m=15 ve n=28} \tag{3.2}$$

Denklem 3.2'de i değerleri hemşire sayısını, j değerleri ise gün sayısını göstermektedir. Buna göre herhangi bir hemşire için vardiya sayısı toplamı, 28 gün için vardiya sayılarının toplamından oluşmaktadır ve her hemşire için ayrı ayrı hesaplanmaktadır.

AF1 ve AF2 birlikte ele alındığında amaç fonksiyonunun her alt fonksiyonu için optimum değeri verecek bir yaklaşımla hemşireler için oluşturulabilecek en iyi çizelgelemeyi ifade eden AF1 amaç fonksiyonu değerinin AF2 ve diğer alt amaç fonksiyonu değerlerinin toplamından oluşabileceği görülür (Denklem 3.3).

$$AF_1 = AF_2 + \sum_{i=1}^{m} ZK_i + \sum_{j=1}^{n} EK_j \quad \text{burada i=1,2,…,m ve j=1,2,…,n ve m=8 ve n=8} \tag{3.3}$$

Dolayısıyla AF1 için optimum amaç fonksiyonu değeri; AF2, zorunlu kısıtlar (toplam 8 adet) ve esnek kısıtların (toplam 8 adet) optimum değerlerinin toplamından, yani 17 adet alt amaç fonksiyonu değerinin toplamından oluşacaktır.

Ancak gündüz ve gece vardiyalarının sürelerinin eşit olmaması nedeniyle "AF2 - Her hemşirenin toplam atandığı vardiyaların mümkün olduğunca eşit olması" kısıtının haftalık ve 4 haftalık çizelgelerde "ZK3 - Haftada en az 40 saat çalışılır" kısıtının optimum olduğu yani "haftada 40 saat çalışılır" şeklinde düşünülmesi uygun olacaktır. Dolayısıyla ZK3 kısıtının optimum düzeyde yani her hemşire için haftalık 40 saat vardiya ataması olacak şekilde yapılması AF2 amaç fonksiyonunu da yerine getirmiş olacaktır. Dolayısıyla optimum durumda AF2 için ayrı bir tanım yapılmasına ve kod yazılmasına gerek kalmamakta ve denklemde AF2 için 0 konulduğunda (çünkü ZK3, AF2'nin gerektirdiği durumu tanımlıyor) Denklem 3.4'teki hale gelmektedir:

$$AF_1 = \sum_{i=1}^{m} ZK_i + \sum_{j=1}^{n} EK_j \qquad \text{burada i=1,2,…,m ve j=1,2,…,n ve m=8 ve n=8} \tag{3.4}$$

Denklemin açık hali aşağıdaki gibidir (Denklem 3.5):

$$AF_1 = ZK_1 + ZK_2 + ZK_3 + ZK_4 + ZK_5 + ZK_6 + ZK_7 + ZK_8 + EK_1 + EK_2 + EK_3 + EK_4 + EK_5 + EK_6 + EK_7 + EK_8 \tag{3.5}$$

Amaç fonksiyon denkleminin Matlab'taki ifadesi ise aşağıdaki gibidir (Denklem 3.6):

```
fonksiyonDegeriAF1 = fonksiyonDegeriZK1 + fonksiyonDegeriZK2 +
                     fonksiyonDegeriZK3 + fonksiyonDegeriZK4 +
                     fonksiyonDegeriZK5 + fonksiyonDegeriZK6 +
                     fonksiyonDegeriZK7 + fonksiyonDegeriZK8 +
                     fonksiyonDegeriEK1 + fonksiyonDegeriEK2 +
                     fonksiyonDegeriEK3 + fonksiyonDegeriEK4 +
                     fonksiyonDegeriEK5 + fonksiyonDegeriEK6 +
                     fonksiyonDegeriEK7 + fonksiyonDegeriEK8;
```

(3.6)

fonksiyonDegeriAF1, ana amaç fonksiyonu olarak düşünülmüştür. Zorunlu ve esnek kısıtların optimum durumu ise alt amaç fonksiyonları olarak düşünülmüştür. Zorunlu ve esnek kısıtlar için optimum durum için oluşturulan fonksiyon değerlerinin her biri için matematiksel ifadeler, ilgili kısıta uyulması halinde 0 (sıfır) değerini, uyulmaması halinde ise 0'dan büyük pozitif bir tam sayıyı vermektedir. Dolayısıyla tüm kısıtlara uyulmuş olması halinde fonksiyonDegeriAF1'in değeri 0 (sıfır) olmakta, bir ya da birden fazla kısıtın optimum durumunun ihlal edilmiş olması halinde ise sıfırdan

büyük bir değere sahip olmaktadır. Kısaca alt amaç fonksiyonlarının değerlerinin toplamı ana amaç fonksiyonunun değerini vermektedir.

Aşağıda toplamları ana amaç fonksiyon değerini (AF1) oluşturan 16 alt amaç fonksiyonu değerlerinin her biri için optimum çözümü verecek denklemlerin ve yine her biri için Matlab kodlarının oluşturulması ele alınacaktır.

Zorunlu kısıtlar (ZK1-8):

ZK1 - Günde 24 saat hizmet veriliyor: Hastane 7 gün 24 saat kesintisiz hizmet vermektedir. Bu, 1 haftada 7 gün 24 saatten haftada 168 saat, 4 haftada 672 saat hizmet anlamına gelmektedir. Hemşire çizelgelemede her gün için her vardiyaya en az 1 hemşire atanmış olması bu zorunlu kısıta uyulmasını sağlayacaktır. Buna göre her gün için en az 1 hemşirenin gündüz vardiyasına, 1 hemşirenin de gece vardiyasına atanmış olması, yani bu iki şartın birlikte sağlanması gerekmektedir. Bunun için öncelikle 15x28'lik yani 15 hemşire x 28 günlük bir matris oluşturularak matrisin elemanlarına 0-1-2 değerlerinden biri rastgele atandı. Böylece 15 hemşire için rastgele gündüz vardiyası (1), gece vardiyası (2) ve izinli (0) atamalar yapılmış olan 28 günlük bir başlangıç çizelgesi oluşturulmuş oldu. Sonrasında her sütun bir günü temsil ettiğinden önce matrisin her bir sütunu tek tek taranarak içinde 1 yani gündüz vardiyası ve 2 yani gece vardiyasına yapılan atamalar sayıldı, ardından bu değerler birbirleriyle çarpılarak ve en son 100 ceza katsayısı ile çarpılarak ZK1 için alt amaç fonksiyon değeri hesaplanmış oldu. ZK1'in alabileceği minimum değer 0 iken maksimum değer ise her gün için bir ihlal olabileceğinden 28 günlük çizelge için 28 ihlal x ceza katsayısı (100) = 2800'dür.

$$ZK_1 = \sum_{j=1}^{n} zk_{1j} \qquad \text{burada} \quad j=1,2,\ldots,n \quad \text{ve} \quad n=28 \qquad (3.7)$$

Bu kısıt için *zk1*'in matematiksel ifadesi yapılamadığından basit bir gösteriminin yapılabilmesi açısından excel tabanlı formülizasyonu yapılarak Denklem 3.8'de verilmiştir:

$$zk_1 = E\check{G}ER((E\check{G}ERSAY(Hi{:}Hm;1) * E\check{G}ERSAY(Hi{:}Hm;2)) = 0;100;0)$$

burada i=1 ve m=15 ve $H_i{:}H_m$=1,2,…,m (3.8)

Denklem 3.8'de *H* hücreye yapılan atama değerini göstermektedir. 15 hemşire için günlük hesaplanarak zk_1 değerini vermektedir. zk_1 değeri ise 28 gün için hesaplanıp

bu değerler Denklem 3.7'deki gibi toplanarak ZK_1 için alt amaç fonksiyon değerini vermektedir.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.1):

Çizelge 3.1. ZK1 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
m=15;
n=28;
c=100;
b=round(0+2*rand(m,n));
[m n]=size(b);
birlerinsatirfrekansi=0;
ikilerinsatirfrekansi=0;
FonksiyonDegeriZK1=0;
birlerinsatirfrekansi=sum(b==1);
frekans=birlerinsifirfrekansi;
ikilerinsatirfrekansi=sum(b==2);
for i=1
for j=1:n
frekans(i,j)= birlerinsatirfrekansi(i,j)*ikilerinsatirfrekansi(i,j);
End
End
FonksiyonDegeriZK1=c*sum(frekans==0);
fprintf('%d\n',FonksiyonDegeriZK1);
End
```

ZK2 - Vardiya saatleri: Gündüz (08.00-16.00), Gece (16.00-08.00): Hastanenin gündüz vardiyaları 8 saat, gece vardiyaları 16 saattir. Buna göre günde 2 vardiya, haftada 14 vardiya, 2 haftada 28 vardiya, 4 haftada 56 vardiya bulunmaktadır. Gece ve gündüz vardiyalarının sürelerinin farklı olması bu vardiyaların ayrı ayrı değerlendirilmesini gerektirmektedir. Bu nedenle öncelikle başlangıçta günü gündüz ve gece vardiyaları olarak iki parça halinde düşünmek, buna göre 28 günlük çizelge için 56 adet vardiyaya 15 hemşire ataması yapılmasının uygun olacağı düşünülmüş, atama yapılan vardiyalara 1 değeri, atama yapılmayan vardiyalara ise 0 değeri verilmesi planlanmıştır. Ancak gündüz vardiyasının 8 saat, gece vardiyasının 16 saat olması nedeniyle ve 15x56 (15 satır yani 15 hemşire x 56 sütun, yani 56 vardiya yani günde 2 vardiyadan 28 günde 56 vardiya) gibi büyük bir tablo (matris) yerine daha küçük bir matrisle çözüm aranmıştır. Buna göre 15x28'lik bir çizelge matrisi oluşturulması, bu

matriste hemşirelerin çalışmadıkları günler için 0, gündüz vardiyalarına atamalarında 1, gece vardiyalarına atamalarında 2 değeri verilmesi uygun görülmüştür. Bu sayede matrislerin hücre değerleri ile 8 saat çarpıldığında, çalışılmayan günler için 0x8 saat=0 saat, gündüz vardiyalarına atamalar için 1 x 8 saat = 8 saat, gece vardiyalarına atamalar içinse 2 x 8 saat = 16 saat şeklinde ilgili vardiyaların çalışma süreleri de kolaylıkla hesaplanabilmiştir. Bu sayede 15x56 matriste toplam 15x56=840 hücreye 0 ya da 1 değeri yani 2 farklı değer atandığında oluşan yüksek toplam olasılık ($2^{840}=7,3*10^{252}$) yerine 15x28=420 hücreye 0, 1 veya 2 değeri yani 3 farklı değer atandığında oluşan daha düşük bir toplam olasılık değerine ($3^{420}=2,5*10^{200}$) indirilebilmiştir. Bu da programın çalışma süresini kısaltacaktır.

Özetle bu kısıt zaten 0-1-2 atamasıyla otomatik olarak sağlanmaktadır, yani 0 ataması izinli günleri yani gün içinde toplam çalışma süresi olarak 0'ı, 1 ataması 8 saati yani gün içinde toplam çalışma süresi olarak 8 saati, 2 ataması ise gün içinde toplam çalışma saati olarak 16 saati gösteriyor. Dolayısıyla bu kısıtı ihlal eden bir durumun baştan 0-1-2 atamasıyla gerçekleşmemesi sağlanmıştır ve ZK2 için alt amaç fonksiyonu değeri her zaman 0 olmaktadır.

ZK3 - Haftada en az 40 saat çalışılır: Buna göre her hemşire çizelgelemedeki herhangi bir haftada en az 40 saat, 4 haftada (28 günde) ise en az 160 saat çalışacak şekilde atanmalıdır. Optimum durum her hemşirenin hem herhangi bir haftada 40 saat hem de 4 haftalık süreçte de 160 saat çalışmasıdır. Dolayısıyla bu kısıttaki optimum duruma göre, herhangi bir haftada herhangi bir hemşire 40 saat çalışır ve dolayısıyla da 4 haftalık süreçte 160 saat çalışmış olur. Yani optimum durumda hem haftalık 40 saat hem de 4 haftalık 160 saat atama zorunluluğu oluşturmaktadır. Haftada 40 saat, ayda 160 saat toplam çalışma süreleri, fazla mesai ücreti oluşmaması ve haftalar arasında çalışma süresi bakımından dengesizlik oluşmaması bakımından aynı zamanda hedeflenen optimum çalışma süreleridir. Çünkü herhangi bir haftada herhangi bir hemşirenin 40 saatten fazla çalışması halinde, sonraki haftalarda bu kısıt nedeniyle haftalık 40 saatin altında çalışamayacağından 4 haftalık süreç sonunda fazla mesai söz konusu olacaktır ki bu da hastanenin işgücü maliyetini arttıracaktır.

Bunun için öncelikle 15x28'lik yani 15 hemşire x 28 günlük bir matris oluşturularak matrisin elemanlarına 0-1-2 değerlerinden biri rastgele atandı. Böylece 15 hemşire için rastgele gündüz vardiyası (1), gece vardiyası (2) ve izinli (0) atamalar yapılmış olan 28 günlük bir başlangıç çizelgesi oluşturulmuş oldu. Haftalık bazda işlem

yapabilmek için h adında bir değişkene 4 değeri atandı ve haftalık döngü değişkeni hh'nin 1'den h'ye yani 1'den 4'e kadar sırasıyla taraması istendi. Bu değer çizelgede 4 hafta için işlem yapılacağını, 1, 2, 3 ve 4. haftalar için ayrı taramalar yapılacağını göstermektedir. Döngüde hh=1 için (1.hafta için) 1-7.günleri, hh=2 için (2.hafta için) 8-14.günleri, hh=3 için (3.hafta için) 15-21.günleri, hh=4 için (4.hafta için) 22-28.günleri tarayacak bir döngü için her hh değerine karşılık bu günleri tarayacak bir denklem gerektiği görüldü. Deneme yanılma yoluyla haftalık tarama yapabilmek için gereken denklemin j_{ilk} için $j_{ilk}=7*(hh-1)+1$ ve j_{son} içinse $j_{son}=7*hh$ olduğu bulundu (Çizelge 3.2).

Çizelge 3.2. Matriste haftalık bazda tarama ve işlem yapabilmek için hafta sayısına bağlı olarak j (sütun) aralıklarını ($j=j_{ilk}:j_{son}$) veren denklemin hesaplanması

	Haftalık tarama için hh'ye bağlı j_{ilk} ve j_{son} değerleri			
	$j_{ilk}=7*(hh-1)+1$	**$j_{son}=7*hh$**	**Hesaplama (j_{ilk} için)**	**Hesaplama (j_{son} için)**
hh= 1	1	7	$j_{ilk}=7*(1-1)+1$	$j_{son}=7*1$
hh= 2	8	14	$j_{ilk}=7*(2-1)+1$	$j_{son}=7*2$
hh= 3	15	21	$j_{ilk}=7*(3-1)+1$	$j_{son}=7*3$
hh= 4	22	28	$j_{ilk}=7*(4-1)+1$	$j_{son}=7*4$

Sonrasında matrisin her elemanı 8 ile çarpılarak günlük çalışma sürelerini veren yeni bir matris oluşturuldu. Bu matris içinde haftalık çalışma süreleri toplandı ve 40 çıkarıldı. Bulunan değer mutlak değer içine alındı. Örneğin herhangi bir haftada 48 saat ya da 32 saat çalışılmışsa |48-40|=|+8|=8 ve |32-40|=|-8|=8 gibi aynı değeri verseler de amaçtan uzaklaşma olarak değerlendirildiği için haftalık çalışma süresi toplamı 40 saatten farklı olanlar 0'dan farklı bir sonuç vermiş oldu. Bir sayaç oluşturularak haftalık toplam çalışma süresi ile haftalık ideal çalışma süresi olan 40 saat arasındaki fark 0'dan farklıysa sayaç 1 artırıldı. Bu işlem sonunda her satır için 4 kez, 15 satır için toplam 60 kez sayaç değerini belirleyen bir döngü oluşturuldu. Döngü sonunda sayaç değeri zorunlu kısıtlar için ceza katsayısı olan 100 ile çarpılarak ZK3 için alt amaç fonksiyon değeri bulunmuş oldu. ZK3'ün alabileceği minimum değer 0 iken maksimum değer ise kısıtın tüm haftalarda ihlal edilmesi halinde oluşan 60 döngü x 100 ceza katsayısı = 6000 olmaktadır.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab program kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.3):

Çizelge 3.3. ZK3 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
m=15;
n=28;
h=4;
c=100;
sayac=0;
fark=0;
fonksiyonDegeriZK3=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
toplam=0;
for i=1:m
  for hh=1:h
    toplam=0;
      for j=(7*(hh-1)+1):(7*hh)
      b(i, j)=8*b(i, j);
      toplam=toplam+b(i, j);
      End
      fark=abs(toplam-40);
      if fark~=0
      sayac=sayac+1;
      else
      sayac=sayac;
      End
   End
End
fonksiyonDegeriZK3=(c*sayac)+fonksiyonDegeriZK3;
fprintf('%d\n',fonksiyonDegeriZK3);
End
```

Bu kısıta ilişkin kod yazılımı yapıldığında kullanılan tavlama algoritmasında başlangıçta oluşturulan rastgele 0-1-2 atamalı çizelgeye ek olarak algoritmadaki rastgele atama yöntemi çizelgedeki tek bir hücrenin rastgele bulunarak bu hücrenin en yakın komşularından biri rastgele seçilerek değerinin mod2 tabanında bir artırılması tekniğine dayanmaktaydı. Ancak rastgele seçilen tek bir hücrenin değerinin değiştirilmesiyle çözüme ulaşmak 6-10 bin deneme aralığını bulmaktaydı. Bunun üzerine atama tekniği değiştirildi ve belirlenen haftalık vardiya kalıplarının bulunduğu matris havuzundan çekilen haftalık vardiya kalıpları kullanılarak matris oluşturulmaya başlandı. Yine de bu kodlar her durumda çizelgenin bu kısıta uygun olması halinde alt amaç fonksiyon değeri olarak 0 sonucunu verdiğinden, kodlar yukarıda verildiği şekilde kullanılmıştır. Ancak program çalıştırıldığında gerek başlangıçta gerekse takip eden tüm denemelerde ZK3 için alt amaç fonksiyon değeri hesaplama yapmakta ve her hesaplaması sonucunda her

zaman 0 sonucunu vermektedir. Bunun nedeni de yukarıda belirtildiği gibi kullanılan haftalık vardiya kalıplarının her birinin haftada 40 saatlik bir çalışmaya uygun olması ve oluşturulan çizelgelerin bu kısıtı her zaman sağlıyor olmasıdır.

ZK4 - Hemşirelerin ardışık olarak en fazla 2 gece vardiyasında çalışmasına izin verilir: Bu kısıta göre herhangi bir günün gece vardiyasında çalışan bir hemşire takip eden günün gece vardiyasına atanabiliyor ancak 3 gün peş peşe gece vardiyasına atanamıyor. Zaten atanıyor olsa o zaman herhangi bir haftada 3 gece x 16 saatten 48 saat çalışma olasılığı, dolayısıyla 40 saatin üzerinde çalışma, yani fazla mesai yapma, dolayısıyla fazla mesai ücreti ödenmesi durumu oluşuyor ki bu her ne kadar ZK3'e uygunsa da optimum çözüm için istenen bir durum değildir. Buna göre çizelgelemede herhangi bir hemşire için ardışık herhangi 3 günde atanan mesai değeri toplamının 6 olduğu durumlara (gece mesaileri için 2 değeri atandığı için 2+2+2=6) 100 ceza puanı verilmiştir. Bu durumun matematiksel ifadesi Denklem 3.6'da gösterilmiştir.

$$her\ m\ için \sum_{j=m}^{m+2} zk_{4j} = 6\ \ ise \quad zk_4 = 1 \qquad \text{Burada m=1,2,…,(n-2) ve n=28 (3.6)}$$

$$yoksa \qquad zk_4 = 0$$

$$ZK_4 = \sum_{j=1}^{n-2} c * zk_{4j} \qquad \text{Burada c=100 (ceza katsayısı)}$$

Matlab'ta kod yazılımında öncelikle 15x28'lik yani 15 hemşire x 28 günlük bir matris oluşturularak matrisin elemanlarına 0-1-2 değerlerinden biri rastgele atandı. Böylece 15 hemşire için rastgele gündüz vardiyası (1), gece vardiyası (2) ve izinli (0) atamalar yapılmış olan 28 günlük bir başlangıç çizelgesi oluşturulmuş oldu. Sonrasında ardışık 3 gün için atama değerleri toplamı 6 olanlar, yani ardışık 3 gün boyunca gece vardiyasına yapılan (2 değerli) atamalar sayac değerini 1 arttıracak şekilde düzenlendi ve sonunda sayac 100 ceza puanı ile çarpıldı. ZK4'ün alabileceği minimum değer 0 iken maksimum değer ise kısıtın toplamda 26 hafta olarak hesaplanan ardışık 3'lü günlerinde tüm hemşireler (15 hemşire) için ihlal edilmesi halinde oluşan 26 hafta x 15 x 100 ceza katsayısı = 39000 olmaktadır.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.4):

Çizelge 3.4. ZK4 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
m=15;
n=28;
u=0;
c=100;
fonksiyonDegeriZK4=0;
sayac=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
toplam=0;
for i=1:m
for u=1:(n-2)
for j=u:(u+2)
toplam=toplam+b(i, j);
End
if toplam==6
sayac=sayac+1;
toplam=0;
Else
toplam=0;
End
End
End
toplam=sayac*c;
fonksiyonDegeriZK4=toplam;
fprintf('%d\n',fonksiyonDegeriZK4);
End
```

ZK5 - Bir hemşire gece vardiyasından sonraki gün gündüz vardiyasına atanamaz: Gece vardiyası 16 saat olduğu için herhangi bir hemşirenin gece vardiyasında çalıştıktan sonra gündüz vardiyasına atanması, o hemşirenin 1 gün içinde toplam 24 saat çalışması anlamına geleceği için istenmemektedir. Buna göre bir hemşire 24 saatlik herhangi bir gün içinde en fazla 16 saat çalıştırılabilecektir. Bu kısıtın sağlanabilmesi için herhangi bir satırda 2 olan değeri takip eden günde 1 değerinin gelmemesi gerekmektedir. Yani gece vardiyasından sonraki günde gündüz vardiyasına atama yapılmaması gerekmektedir. Ardışık 2 gün için ilk güne Gün1, bunu takip eden güne Gün2 denildiğinde *X=(Gün1-Gün2)*(Gün1-Gün2+1)*(Gün1*Gün2)* formülü ile sadece gece vardiyasını takip eden gündüz vardiyalarında *X* değerinin 0'dan farklı (X=4) bir değerde olduğu görülmüştür. Çizelgenin yatay ekseninde (satırlarda) hemşirelerin H_i ile (i=1,2,3,….m burada m=15 yani toplam hemşire sayısı) ve çizelgenin dikey ekseninde (sütunlarda) günlerin Dj ile (j=1,2,3,….n burada n=28 yani toplam gün

sayısı) alındığı ve hücrelere 0 (izinli), 1 (gündüz vardiyası) ve 2 (gece vardiyası) atamalarının Aij (0, 1 ya da 2, burada i=1,2,3,….m ve m=15, j=1,2,3,….n ve n=28) gösterildiği durumda *X=[A(i,j)-A(i,j+1)]*[A(i,j)-A(i,j+1)+1]*[A(i,j)*A(i,j)]* şeklinde de ifade edilebilen denklemde X için 4 değerini veren ardışık hücrelerin değerlerine "0" verildiğinde, buna uymayan yani kendisinden önceki gün gece vardiyasına atanmış olunup gün içinde gündüz vardiyasına atanmış olan, yani bu kısıtı ihlal eden hücrelerin değerlerine ise 0'dan farklı bir ceza puanı (c) değeri verildiğinde amaç fonksiyon değerinin bu kısıta uyulması halinde 0'ı vermesi, uyulmaması halinde ise belirlenen bir ceza puanı (c) değerini alması sağlanabilmektedir. *X=(Gün1-Gün2)*(Gün1-Gün2+1)*(Gün1*Gün2)* ya da *X=[A(i,j)-A(i,j+1)]*[A(i,j)-A(i,j+1)+1]*[A(i,j)*A(i,j+1)]* denklemi deneme yanılma yöntemiyle Çizelge 3.1'deki gibi 0-1-2 için 3^2=9 olasılık için yani ardışık tüm vardiya çiftleri (0-0, 0-1, 0-2, 1-0, 1-1, 1-2, 2-0, 2-1, 2-2) için bulunmuştur (Çizelge 3.5).

Çizelge 3.5. Zorunlu Kısıt 5 (ZK5) için alt amaç fonksiyon değerinin formülize edilmesi

	X=[A(i,j)-A(i,j+1)]*[A(i,j)-A(i,j+1)+1]*[A(i,j)*A(i,j+1)]				**ZK5 için alt amaç fonksiyonu değeri)**
	j=1	**j=2**	**Formülasyonda i ve j değerlerinin yerine konulmuş hali**	**X**	**X=0 ise X=c X≠0 ise X=0**
i=1	0	0	X=(0-0)*(0-0+1)*(0*0)	0	c
i=2	0	1	X=(0-1)*(0-1+1)*(0*1)	0	c
i=3	0	2	X=(0-2)*(0-2+1)*(0*2)	0	c
i=4	1	0	X=(1-0)*(1-0+1)*(1*0)	0	c
i=5	1	1	X=(1-1)*(1-1+1)*(1*1)	0	c
i=6	1	2	X=(1-2)*(1-2+1)*(1*2)	0	c
i=7	2	0	X=(2-0)*(2-0+1)*(2*0)	0	c
i=8	2	1	X=(2-1)*(2-1+1)*(2*1)	4	**0**
i=9	2	2	X=(2-2)*(2-2+1)*(2*2)	0	c

ZK5'in alabileceği minimum değer 0 iken maksimum değer ise her hemşire için her ardışık gün çiftinin gece-gündüz yani 2-1 şeklinde vardiya atamalarıyla devam etmesi halinde 28 günün 14'ünde kısıtı ihlal etmesi anlamına geleceği için 15 hemşirede toplam 14 x 15 = 210 kez kısıt ihlal edilmiş ve 100 ceza katsayısıyla çarpıldığında 210 x 100= 21.000 olmaktadır.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.6):

Çizelge 3.6. ZK5 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
m=15;
n=28;
u=0;
c=100;
fonksiyonDegeriZK5=0;
sayac=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
toplam=0;
for i=1:m
for u=1:(n-2)
for j=u:(u+1)
toplam=(b(i,j)-b(i,j+1))*(b(i,j)-b(i,j+1)+1)*(b(i,j)*b(i,j+1));
End
if toplam~=0
sayac=sayac+1;
toplam=0;
Else
toplam=0;
End
End
End
toplam=sayac*c;
fonksiyonDegeriZK5=toplam;
fprintf('%d\n',fonksiyonDegeriZK5);
End
```

ZK6 - Bir hemşire bir gün içinde yalnızca bir vardiyada çalışabilir: Bu kısıt da bir hemşirenin 24 saatlik dilim içinde 8 ya da 16 saatlik gündüz ya da gece vardiyalarından sadece birinde çalıştırılabilmesine izin veriyor. Bazı hastanelerde 8 saatlik 3 vardiya olduğunda peş peşe iki vardiya çalıştırmak mümkün olabilmektedir ancak bu çalışmada ele alınan hastanede gündüz 8 saat, gece 16 saat şeklinde günde 2 vardiya olduğundan peş peşe vardiyalarda çalıştırma durumu 24 saatlik dilimde 24 saat çalışma anlamına geldiği için istenmemektedir. Çalışılmayan/izinli olunan günlere 0, gündüz vardiyalarına 1, gece vardiyalarına 2 değeri atanması sayesinde herhangi bir hemşireye herhangi bir gün için sadece tek bir vardiya ya da izin ataması yapılmakta olduğundan günlük bazda çalışma saatleri 0, 8 ve 16 saat şeklinde olmaktadır. 0-1-2

ataması sayesinde bu kısıt otomatik olarak yerine getirilmiş olduğundan amaç fonksiyonunda bu kısıtın oluşturacağı değer her zaman 0 olacaktır. Yani 0-1-2 ataması sayesinde bu kısıtın amaç fonksiyonunda ifade edilmesine gerek bulunmamakta, amaç fonksiyonunda yer alması halinde ise her zaman 0 değerini almaktadır.

ZK7 - Sorumlu hemşire gece nöbeti tutmaz ve hafta sonu çalışmaz: Sorumlu hemşire sayısının 1 kişi olduğu varsayılmıştır. ZK3'e göre haftada en az 40 saat çalışılır. ZK7'ye göre sorumlu hemşire sadece hafta içlerinde gündüz vardiyalarında çalışabilmektedir. Bu da 5 günde 8 saatten 40 saat yapmaktadır. Yani 15 hemşireden sadece bir hemşire (sorumlu hemşire) hafta içi 5 günün gündüz vardiyalarına atanacak, böylece bu sorumlu hemşire haftada 5 gün x 8 saat = 40 saatten 4 haftalık çizelgede toplam 160 saat çalışmış olacaktır. Diğer 14 hemşire için bu kısıt geçerli değildir. Bu amaçla hafta sonlarına denk gelen vardiya atama değerleri toplanmıştır. Hafta sonu hiç atama olmaması halinde bu değer ideal değer olan 0 çıkmaktadır. Ardından 28 günde yer alan 20 hafta içi gündeki vardiya değerleri birbiriyle çarpılarak bulunan değerden 1 çıkarılmıştır. Sorumlu hemşire için değer ideal olarak hafta içi her gün için 1 olması, çarpımlarının 1 olması, dolayısıyla bu çarpımdan 1 çıkarılmasıyla elde edilen değerin 0 olmasıdır. Sonuç olarak elde edilen toplam ve çarpım değerleri birbirleriyle toplanmış, 0'dan farklı olan durumlar için sayaç 1 artırılmıştır. Birinci sayaç 14 ise ikinci sayaç 0, birinci sayaç 14'ten farklı ise ikinci sayaca birinci sayacın değeri verilmiştir. Böylece bu kritere sadece 1 hemşirenin uyması halinde ikinci sayaç 0'ı, hiçbir hemşirenin uymaması halinde ise ikinci sayaç 15'i göstermektedir. İkinci sayaç değeri, zorunlu kriterler için belirlenen 100 ceza puanı ile çarpıldığında ZK7 için alt amaç fonksiyon değeri bulunmaktadır. ZK7'nin alabileceği minimum değer 0 iken maksimum değer ise 15 hemşire için kısıtın hafta sonu çalışmama ve gece çalışmama kısıtlarının birlikte ihlal edilmesi durumunda 15 hemşire x 100=1500'dür.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.7):

Çizelge 3.7. ZK7 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
1  sayac2ZK7=0;
2  islemZK7=0;
3  carpimZK7=0;
```

```
toplamZK7=0;
durumtoplamZK7=0;
durumcarpimZK7=0;
fonksiyonDegeriZK7=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
for i=1:m
   toplamZK7=b(i,6)+b(i,7)+b(i,13)+b(i,14)+b(i,20)+b(i,21)+b(i,27)+b(i,28);
carpimZK7=(b(i,1)*b(i,2)**b(i,3)*b(i,4)*b(i,5)*b(i,8)*b(i,9)**b(i,10)*b(i,11)*b(i,12)*b(i,15)*b
(i,16)*b(i,17)*b(i,18)*b(i,19)*b(i,22)*b(i,23)**b(i,24)*b(i,25)*b(i,26)-1);
    islemZK7=toplamZK7+carpimZK7;
   if islemZK7~=0
   sayacZK7=sayacZK7+1;
   Else
   sayacZK7=sayacZK7;
  End
End
if sayacZK7==14;
  sayac2ZK7=0;
  Else
  sayac2ZK7=sayacZK7;
End
fonksiyonDegeriZK7=sayac2ZK7*100;
fprintf('Fonksiyon degeri: %d\n',fonksiyonDegeriZK7);
sayac2ZK7=0;
islemZK7=0;
carpimZK7=0;
toplamZK7=0;
```

ZK8 - 50 yaşın üstünde olan ve gebe olan hemşire nöbet tutmaz: Çizelgelemede bu özelliği taşıyabilecek hemşire sayısının en fazla 3 olacağı varsayılmıştır. Bunun için yazılacak programın girişinde ekrana "Lütfen 50 yaş üstü ve gebe hemşirelerin toplam sayısını giriniz: " şeklinde bir yazının çıkması ve yazının sonundaki yanıp sönen imlecin olduğu kısımda kullanıcının gireceği sayıyı veri olarak kullanması sağlanacaktır. Kullanıcının 3'ten büyük bir sayı girmesi halinde ise "50 yaş üstü ve gebe hemşire sayısı 3'ü geçemez. Lütfen yeniden giriş yapınız." şeklinde bir uyarı yazısı çıkması sağlanacaktır. Kullanıcı 3 ve daha küçük bir sayı girene kadar bu uyarı devam edecektir.

Ancak bu kısıtı tek başına değerlendirmek yerine, "EK4: Personelin mazeret belirttiği günlere nöbet yazılmaz" kısıtı ile birlikte ele alınması uygun görülmüş, yazılan kodlarda en başta kullanıcıdan 50 yaşın üstünde olan ve gebe olan hemşire sayısını klavye ile girmesi istenmiştir. 50 yaş üstü, gebe ve mazeretli hemşirelerin toplam sayısından uyarı vermek yerine ayrı ayrı uyarı veren bir yapıda olması gerektiği düşünülmüştür. 50 yaş üstü ve gebe hemşire sayısı ile mazeretli hemşire sayısı bilgilerini kullanıcıdan almakta; 50 yaş üstü ve gebe hemşire sayısı 3'ten büyük olduğu

sürece uyararak yeniden giriş istemekte, mazeretli hemşire sayısı 5'ten büyük olduğu sürece uyararak yeniden giriş isteyen, şartlar sağlandığında kullanıcıdan aldığı bu verileri kullanarak diğer işlemlere geçmektedir. İlk 15x28'lik matris oluşturulup elemanlarına rastgele 0-1-2 atanması sonrasında 50 yaş üstü ve gebe hemşire sayısı ile mazeretli hemşire sayısı toplamı kadar hemşireye 0 ataması yapıldı. Bu yapılırken son satırdan geriye doğru gidildi. Örneğin çalışmayan durumdaki hemşire sayısı 3 ise matrisin son 3 satırındaki tüm elemanlar 0'landı. Bu durum amaç fonksiyonu değerinin en iyi durumda 0 olarak bulunan çözümünü değiştirdi. Sistematik bir durum söz konusu olmadığından kaç hemşirenin çalışmaması halinde amaç fonksiyonunun ne miktarda değişeceği bilinmemektedir. Bu nedenle amaç fonksiyonu başlangıç ataması ve çalışmayan hemşire sayısına bağlı olarak sıfırdan büyük ancak öngörülemeyen bir değerde çıkacaktır. Bu nedenle de tavlama algoritması uygulandıktan sonra çözümün en iyi sonucu verip vermediğinin ayrıca kontrol edilmesinde fayda bulunmaktadır. Bu kısıtın amaç fonksiyonuna katkısı diğer tüm kısıtlar üzerinden olduğu için burada ayrıca bu kısıta yönelik fonksiyon elemanı olmayacak, sadece kullanıcıdan veri girişi isteyen kodlar verilecektir.

"ZK8: 50 yaşın üstünde olan ve gebe olan hemşire nöbet tutmaz" kısıtı ve "EK4: Personelin mazeret belirttiği günlere nöbet yazılmaz" kısıtına ilişkin kullanıcıdan bu özellikleri taşıyan hemşire sayısını ayrı ayrı isteyen Matlab pogram kodları aşağıdaki şekilde yazılmış, ayrı ayrı sınırı aşan farklı değerler ve sınırın içindeki değerler için farklı varyasyonlarda denenmiş ve doğrulanmıştır (Çizelge 3.8):

Çizelge 3.8. ZK8 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
1  ElliYasUstuVeGebeHemsireSayisi=0;
2  MazeretliHemsireSayisi=0;
3  ToplamElliYasUstuGebeVeMazeretliHemsireSayisi=0;
4  ElliYasUstuVeGebeHemsireSayisi=input('Lutfen 50 yas ustu ve gebe hemsirelerin toplam
   sayisini giriniz ve enter tusuna basiniz: ');
5  while(ElliYasUstuVeGebeHemsireSayisi>4)
6  disp('50 yas ustu ve gebe hemsire sayisi 4"u gecemez. Lutfen yeniden giris yapiniz.');
7  ElliYasUstuVeGebeHemsireSayisi=input('Lutfen 50 yas ustu ve gebe hemsirelerin toplam
   sayisini giriniz ve enter tusuna basiniz: ');
8  End
9  MazeretliHemsireSayisi=input('Lutfen mazeretli hemsirelerin toplam sayisini giriniz ve enter
   tuşuna basiniz: ');
10 while(MazeretliHemsireSayisi>5)
11 disp('Mazeretli hemsire sayisi 5"i gecemez. Lutfen yeniden giris yapiniz.);
12 MazeretliHemsireSayisi=input('Lutfen mazeretli hemsirelerin toplam sayisini giriniz: ');
```

```
13 End
14 ElliYasUstuGebeVeMazeretliHemsireSayisi=ElliYasUstuVeGebeHemsireSayisi+MazeretliHems
   ireSayisi;
```

Esnek Kısıtlar (EK1-8):

EK1 - Bir hafta içerisinde en az bir kere ardışık 48 saat izin yapılmalıdır: Ardışık 48 saat izin, ardışık 2 günlük bir süre boyunca izin anlamına geldiğinden ilk bakışta her hemşire için haftalık toplam 7 günlük toplam 7 atama içinde en az ardışık 2 gün izin yapabilecekleri boşluk oluşturulması gerektiği görünmektedir. Haftada en az 1 kere 48 saat ve üzeri ardışık izin yapma durumuna uyan atamaların 00 yani ardışık günlerde çalışmama durumuyla sağlanacağı ilk bakışta görülmektedir. Ancak ardışık 102 atamaları yani ardışık 3 günde sırasıyla gündüz-izinli-gece vardiyasına atanma durumlarının da bu kısıta uyduğu ve 48 saatlik ardışık izin sağladığı görülmektedir. Herhangi bir günde gündüz vardiyası biten hemşire günün kalanındaki 16 saat izinli olmakta, ertesi gün çalışmadığında toplam izni 16+24=40 saat olmakta, sonraki gün gece vardiyasına kadar çalışmadığında ise 40+8=48 saat izin yapmış olmaktadır ki bu durum ardışık sıralı 102 vardiyalarını göstermektedir. Buna göre EK1 için herhangi bir haftada 00 ve 102 vardiyaları tespit edilecek şekilde hesaplama yapılmıştır. Her hemşire için 4 haftadan 15 hemşire için toplam 60 hafta grubu olduğundan ve esnek kısıtlar için ceza puanı 10 olarak belirlendiğinden, bu kısıttan alınabilecek maksimum alt amaç fonksiyonu değeri 4 hafta x 15 hemşire x 10 ceza puanı = 600 puandır.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.9):

Çizelge 3.9. EK1 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
1  ElliYasUstuGebeVeMazeretliHemsireSayisi=0; %kodlar birlestirlirken bu satir silinmelidir.
2  cEK1=10;
3  fonksiyonDegeriEK1=0;
4  m=15;
5  n=28;
6  b=round(0+2*rand(m,n)) %kodlar birlestirlirken bu satir silinmelidir
7  [m n]=size(b);
8  sayac1EK1=0;
9  sayac2EK1=0;
10 sayac3EK1=0;
11 Ardisik00Arama=0;
12 Ardisik102Arama=0;
13 Ardisik00ve102Arama=0;
14 h=4;
15 for i=1:m-ElliYasUstuGebeVeMazeretliHemsireSayisi;
```

```
  for hh=1:h;
    for j=(7*(hh-1)+1):((7*hh)-2);
    Ardisik00Arama=(b(i,j)+b(i,j+1))*(b(i,j+1)+b(i,j+2));
    Ardisik102Arama=abs(b(i,j)-b(i,j+1)-1)+abs(b(i,j+2)-b(i,j+1)-2);
    Ardisik00ve102Arama=Ardisik00Arama*Ardisik102Arama;
      if Ardisik00ve102Arama==0;
      sayac1EK1=sayac1EK1+1;
      else
      sayac1EK1=sayac1EK1;
      end
    End
  if sayac1EK1>=1;
  sayac2EK1=sayac2EK1+1; %sayac2EK1'in degerinin 0'dan farklı olması o haftada en az 1 adet ardışık 48 saat izin yapildigi anlamina gelmektedir.
  sayac1EK1=0;
  else
  sayac2EK1=sayac2EK1;
  sayac1EK1=0;
  end
  End
   sayac3EK1=sayac3EK1+(4-sayac2EK1);
   sayac2EK1=0;
end
fonksiyonDegeriEK1=cEK1*sayac3EK1;
```

EK2 - Her hemşireye eşit sayıda hafta sonu tatili atanmalıdır: Tamamen adil bir çizelge oluşturulabilmesi için tüm hemşireler için 4 haftalık çizelgelerde hafta sonu çalışılan ya da çalışılmayan gün sayısının eşit olması yeterli değildir. Gündüz vardiyasının 8 saat, gece vardiyasının 16 saat olması nedeniyle bazı hemşirelerin 2-4 haftalık çizelgelerde hafta sonu tamamen gündüz vardiyasına bazılarının ise tamamen gece vardiyasına atandığı durumlar oluşabilir. Bir hemşirenin 4 haftalık çizelgelemede sürekli hafta sonlarında gece vardiyalarına atanmış olması, buna karşılık başka bir hemşirenin hafta sonlarında sürekli gündüz vardiyalarına atanmış olması hafta sonu toplam çalışma ve izin süresi açısından adil olmayacaktır. Bu nedenle hafta sonu tatilleri ya da çalışma vardiyaları için hem gece hem de gündüz vardiyalarının sayısı her hemşire için eşit olmalıdır. 2-4 haftalık çizelgelemede her hemşire bir hafta sonunda gece sonraki hafta sonunda gündüz vardiyasına atanarak düzenli bir döngüyle tüm hemşireler için adil bir durum oluşturulabilmesine çalışılabilir. Ancak bu durum zaten oldukça fazla sayıda olan kısıt ve 4 hafta (28 gün) gibi geniş tutulan çizelge ve ayrıca toplamda 8 hemşireye kadar izinli yani çalışmayan durumda hemşire olması ve bu nedenle çalışan hemşire sayısının 7 kişiye kadar düşebilmesi, az sayıda hemşire ile tüm kısıtların sağlanacağı bir çizelge oluşturmayı güçleştirebilir. Bu nedenle hafta sonu atama sayısının eşit olması bu kısıta uyulduğu anlamında değerlendirilecektir.

Kod yazılımında haftalık vardiya kalıpları kullanıldığı için aday 53 adet haftalık vardiya kalıplarının tamamında hafta sonu atamaları sadece bir güne olacak şekilde yapıldığından, dolayısıyla bu kısıta uyduğundan bu esnek kısıt için ayrıca bir alt amaç fonksiyonu değeri hesaplanmasına gerek kalmamıştır. Yine de yazılımda bu değer 0 olarak gösterilmiştir.

EK3 - Her hemşireye atanan toplam gündüz vardiyaları sayısı gece vardiyaları toplamından büyük veya eşit olmalıdır: Teorik olarak 8 ve 16 saatlik vardiya süreleriyle 4 haftada toplam 160 saat çalışma süresi oluşturabilmek için sadece 20 gündüz vardiyasında çalışmadan (20 vardiya x 8 saat = 160 saat) sadece 10 gece vardiyasında çalışmaya (10 vardiya x 16 saat = 160 saat) kadar giden 11 olasılık çifti mevcuttur. Bunlar arasından EK3 kısıtında belirtilen koşulu sağlayanlar Çizelge 3.10'da yeşil renkle, kısıtı sağlayamayanlar ise kırmızı renkle gösterilmiştir.

Çizelge 3.10. 4 Haftada 160 saat toplam çalışma süresine karşılık gelen gündüz-gece vardiyaları frekansları

	Vardiya Sayısı		
	Gündüz (8 saat)	Gece (16 saat)	
İhtimaller	**D**	**N**	**Toplam Saat**
1	20	0	160
2	18	1	160
3	16	2	160
4	14	3	160
5	12	4	160
6	10	5	160
7	8	6	160
8	6	7	160
9	4	8	160
10	2	9	160
11	0	10	160

Gündüz vardiyaları 8 saat, gece vardiyaları 16 saat olduğu için her hemşire için mümkün mertebe çok sayıda gece vardiyası oluşturulması gerek ki herhangi bir vardiyada hemşire başına düşen çalışma saati ortalaması gündüz vardiyalarındaki çalışma saati ortalamasına yakın olabilsin, yani gece vardiyasına 1 kişi atandığında üzerinde 16 saatlik bir iş yükü, 2 kişi atandığında ise 8'er saatlik iş yükü oluşturacağı için ideal olarak gece vardiyası sayısının gündüz vardiyası sayısının 2 katı olması beklenebilir. Bunu sağlayan kombinasyon 4 haftalık çizelgelemede her hemşire için 4 gündüz + 8 gece vardiyası atanmasıdır. Ancak EK3 "her hemşireye atanan gündüz vardiyası sayısının gece vardiyası sayısına eşit ya da daha fazla olması"na izin verdiğinden, bu kısıtı sağlayan en yakın kombinasyonun 8 gündüz + 6 gece vardiyasına atama, yani aralarındaki fark en az olan ve gündüz vardiyasının gece vardiyası sayısından fazla olduğu sayı çifti olduğu görülmektedir. Manuel atamalarda eğer bu kombinasyon kullanıldığında zorunlu ve esnek kısıtlar sağlanamıyorsa o zaman bu kısıtı sağlayan bir sonraki kombinasyon olan 4 haftalık çizelgelemede 10 gece+5 gündüz vardiya atamasına geçilebilir. O da sağlamıyorsa sırasıyla 12D+4N, 14D+3N, 16D+2N, 18D+1N şeklinde devam edilebilir. Ancak her hafta dengeli bir atama oluşturulabilmesi için 4 haftalık çizelgelemede 8 gündüz + 6 gece ataması ile tüm kısıtların sağlanabileceği düşünülmektedir. Yukarıda bahsedilen bu durum, yani her hemşire için 8 gündüz + 6 gece ataması yapılmasının uygun olacağı varsayımı EK3 için oluşturulan alt amaç fonksiyonu matlab kod yazılımını etkilememiştir. Ancak kod yazılımı bitip algoritma çalıştırıldıktan sonra elde edilen, farklı sayıdaki izinli hemşire durumu için tüm kısıtları sağlayan yüzlerce en iyi çizelgeye bakıldığında bu yaklaşımın yerinde

olduğu görülmüştür. Elde edilen en iyi çizelgelerin tamamına yakınında hemşire başına 8 gündüz + 6 gece atamasının %90'ın üzerinde bir oranda kullanılmış olduğu, bunu takiben %10'luk bir dilim içinde en çok kullanılandan en az kullanılana doğru 10 gündüz + 5 gece, 12 gündüz + 4 gece ve 14 gündüz + 3 gece atamalarının da nadiren kullanıldığı görülmüştür. Dolayısıyla el ile atamalarda 28 günlük çizelgeler için toplamda 8 gündüz + 6 gece ataması yapılması çözüme daha kolay ulaşmayı sağlayacaktır. Ancak yukarıda da belirtildiği gibi bu varsayım, bu kısıta ilişkin program kodlarının yazılmasına herhangi bir temel teşkil etmeyecektir.

Program kodlarının yazılımında yine öncelikle 15x28'lik yani 15 hemşire x 28 günlük bir matris oluşturularak matrisin elemanlarına 0-1-2 değerlerinden biri rastgele atandı. Böylece 15 hemşire için rastgele gündüz vardiyası (1), gece vardiyası (2) ve izinli (0) atamalar yapılmış olan 28 günlük bir başlangıç çizelgesi oluşturulmuş oldu. Daha sonra satır yani hemşire bazında 1'lerin yani gündüz vardiyalarının sayısı sayıldı, benzer şekilde satır yani hemşire bazında 2'lerin yani gece vardiyalarının sayısı sayıldı. Hemşire bazında 28 günün tamamında gündüz vardiyalarının toplam sayısı gece vardiyalarının toplam sayısından az ise sayaç değeri 1 artırıldı. Böylece 15 hemşire için 0'dan 15'e kadar değişebilen bir sayaç değeri elde edilmiş oldu. En sonunda da esnek kısıtlar için belirlenen ceza katsayısı olan 10 ile sayaç değeri çarpılarak EK3 alt amaç fonksiyonu değeri elde edilmiş oldu. EK3'ün alabileceği en düşük değer 0 iken en yüksek değer ise 15 hemşire x 10 ceza katsayısı = 150'dir.

Bu kısıta ilişkin alt amaç fonksiyon değerini hesaplayan Matlab pogram kodları aşağıdaki şekilde yazılmış, farklı değerler için denenmiş ve doğrulanmıştır (Çizelge 3.11):

Çizelge 3.11. EK3 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
1  m=15;
2  n=28;
3  ToplaSatirSatir1=0;
4  ToplaSatirSatir2=0;
5  toplam1=0;
6  toplam2=0;
7  toplam12=0;
8  toplam22=0;
9  ToplaSatirSatir1=0;
10 ToplaSatirSatir2=0;
11 SatirToplamlari1=0;
```

```
SatirToplamlari2=0;
SatirToplamlari12=0;
SatirToplamlari22=0;
sayac=0;
fonksiyonDegeriEK3=0;
c=10;
b=round(0+2*rand(m,n));
for i=1:m;
for j=1:n;
ToplaSatirSatir1=sum(b(i,j)==1);
ToplaSatirSatir2=sum(b(i,j)==2);
toplam1= toplam1+ToplaSatirSatir1;
toplam2= toplam2+ToplaSatirSatir2;
toplam12= toplam12+ToplaSatirSatir1;
toplam22= toplam22+ToplaSatirSatir2;
End
SatirToplamlari1= SatirToplamlari1+toplam1;
SatirToplamlari2= SatirToplamlari2+toplam2;
if SatirToplamlari1>= SatirToplamlari2
sayac=sayac+0;
Else
sayac= sayac+1;
End
toplam1=0;
toplam2=0;
SatirToplamlari1=0;
SatirToplamlari2=0;
End
SatirToplamlari12= SatirToplamlari1+toplam12;
SatirToplamlari22= SatirToplamlari2+toplam22;
fonksiyonDegeriEK3=sayac*c;
disp(fonksiyonDegeriEK3);
```

EK4 - Personelin mazeret belirttiği günlere nöbet yazılmaz: Bu özellikleri taşıyan hemşire sayısının 4 haftalık çizelgelemede en fazla 5 hemşire olacağı varsayılmıştır. Bunun için yazılacak programın girişinde ekrana "Lütfen mazeretli hemşirelerin toplam sayısını giriniz: " şeklinde bir yazının çıkması ve yazının sonundaki yanıp sönen imlecin olduğu kısımda kullanıcının gireceği sayıyı veri olarak kullanması sağlanacaktır. Kullanıcının 5'ten büyük bir sayı girmesi halinde ise "Mazeretli hemşire sayısı 5'i geçemez. Lütfen yeniden giriş yapınız." şeklinde bir uyarı yazısı çıkması sağlanacaktır. Kullanıcı 5 ve daha küçük bir sayı girene kadar bu uyarı devam edecektir. Bu kısıtla ZK8 "50 yaşın üstünde olan ve gebe olan hemşire nöbet tutmaz" birlikte ele alındığında; haftada 7 gün x 24 saat = 168 saat, 4 haftada 672 saat hizmet veren bir

hastane için 15 hemşire bulunan ve bir hemşirenin haftada en az 40 saatten 4 haftada en az 160 saat çalıştırılabildiği durumda haftalık 40 saat x 15 hemşire = 600 saatlik, 4 haftalık ise 4 hafta x 600 saat = 2400 saatlik minimum ve optimum toplam işgücü süresi bulunuyor. Yani çizelgelemeye dahil olan 15 hemşire, 4 haftalık süreçte hastanenin kesintisiz hizmet vermesi gereken 672 saatin çok üzerinde (toplamda 2400 saatlik) bir işgücü oluşturmaktadır. Buna göre eğer 50 yaş üstü ve gebe hemşireler ile mazeret izni kullanan hemşirelerin toplam sayısı maksimum 8 olduğunda, çalışabilir durumdaki hemşire sayısı 15-8=7 olmaktadır. 7 hemşire toplamda 28 günlük çizelgede 160 saat x 7 hemşire = 1120 saatlik bir işgücü oluşturmaktadır. Bu 1120 saatlik işgücü ile hastanenin 7 gün 24 saatten 28 gün 672 saat kesintisiz hizmet sunabilmesi için çizelgelemenin 8 hemşire ile verilen tüm zorunlu kısıtları ve olabildiğince çok sayıda esnek kısıtı sağlayabilmesi beklenecektir. Verilen kısıtlar çerçevesinde çizelgeleme, olabilecek en az sayıda hemşire (1.grup diyelim) ile çözülebilirse, diğer hemşirelerin ataması da yine tüm kısıtlar sağlanacak şekilde yapılırsa (2.grup diyelim), bu durumda 1.grupta yapılan atamalar kesintisiz hizmet için sabit vardiyaları, 2.grupta yapılan atamalar ise o vardiyalarda çalışanların yeri geldiğinde mazeret izni kullanabilecekleri vardiyalar olabilir. Buna göre ZK8'de belirtilen "50 yaş üstü ve gebe hemşireler" ve EK4'te belirtilen "mazeret bildiren hemşireler" olması halinde çizelgede 1.grup vardiyalar her zaman dolu olması sağlandığı taktirde hastanenin kesintisiz hizmeti aksatılmadan bir çizelge oluşturulabilmiş olur. Eğer 50 yaş üstü, gebe ya da mazeret izni kullanan hemşire olursa ve bu hemşireler 2.gruptaki vardiyalarda bulunuyorlarsa, izin kullanmaları halinde sorun yaratmayacak ve çizelgede değişiklik yapılması gerekmeyecektir. Ancak eğer bu hemşirelerden biri ya da birden fazlası 1.grup vardiyalarda yer alıyorlarsa o zaman onların çalışmadığı 1.grup vardiyalara, 2.grup vardiyalarda çalışan hemşirelerin atanması gerekecektir, böylece yine hastane tarafından sorunsuz/kesintisiz hizmet sağlanabilecektir. ZK8'de maksimum 3 hemşire, EK4'te ise maksimum 5, yani bu iki kısıtta maksimum 8 hemşire olabileceği varsayıldığı için oluşabilecek en kötü senaryoda 15 hemşire – 8 izin kullanan maksimum sayıda hemşire = 7 hemşire ile tüm kısıtların sağlandığı bir çizelge oluşturulması hedeflenmelidir. Eğer 1.grupta 7 hemşire ile verilen tüm kısıtları sağlanabilirse, diğer kalan 8 hemşirenin de yine kısıtları sağlayacak şekilde 2.grupta ataması yapılırsa hem ideal/optimum hem de izinler olduğunda kolaylıkla yeni duruma adapte edilebilen esnek bir çizelge oluşturulabilmiş olacaktır. Hesaplandığında; 7 hemşire x haftada 40 saat=280 saat, 4 haftada ise 4 hafta x 280 saat = 1120 saat işgücü ile haftada 168 saat, 4 haftada ise 672

saat işgücü gerektiren hastanenin kesintisiz hizmetinin sağlanabileceği umulabilir/denenebilir. Bu durumun mümkün olup olmadığını tespit etmek için hastanenin 7 gün 24 saat yani haftada 7x24=168 saat hizmet vermesini sağlayacak minimum hemşire sayısının 5 olduğu, buna sorumlu hemşire de eklendiğinde minimum 6 hemşire ya da 7 hemşire ile tüm kısıtlara uyan bir çizelgeleme yapılıp yapılamayacağının test edilmesi gerekmektedir. Bu varsayımdan yola çıkılarak çalışan 7 hemşire için tüm esnek ve zorunlu kısıtlara uyan çizelge ya da çizelgeler elde edilebildiği görülmüştür.

Özetle bu kısıt ZK8 "50 yaşın üstünde olan ve gebe olan hemşire nöbet tutmaz" kısıtı ile birlikte ele alındığından sadece veri girişi için kullanıcıdan bilgi isteyen kodlar ZK8'de verilmiştir. ZK8 ve EK4 kısıtlarına yönelik alt amaç fonksiyonu tanımlaması olmayacaktır çünkü bu iki kısıtın amaç fonksiyonuna etkisi diğer kısıtlar üzerinden olmaktadır. ZK8 ve EK4 için belirtilen toplam çalışamayacak durumdaki hemşire sayısının 0 olması, yani 15 hemşirenin tamamının çalışabilir halde olması durumunda amaç fonksiyonu değeri 0 olabilecektir. Ancak "50 yaşın üstünde olan ve gebe" ve/veya "mazeret izni kullanan" en az 1 hemşire olması halinde amaç fonksiyonu değeri 0'dan farklı pozitif bir değerde olacaktır. Ancak bu değerin ne olacağı, o anki çizelgede yer alan atamalara, hangi ve kaç hemşirenin çalışmadığına bağlı olarak belirsiz bir şekilde değişecektir.

EK5 - Haftada ardışık 72 saat veya daha fazla izin kullanmamalı: Bu kısıta göre haftada en az 1 kere 72 saat ve üzeri ardışık izin yapma durumuna uyan atamaların 000 yani ardışık 3 günde çalışmama durumu ile ihlal edileceği ilk bakışta görülmektedir. Ancak ardışık 1002 atamaları yani ardışık 3 günde sırasıyla gündüz-izinli-izinli-gece vardiyasına atanma durumlarının da bu kısıta uyduğu ve 72 saatlik ardışık izin anlamına geldiği görülmektedir. Herhangi bir günde gündüz vardiyası biten hemşire günün kalanındaki 16 saat izinli olmakta, ertesi gün çalışmadığında toplam izni 16+24=40 saat olmakta, sonraki gün yine çalışmadığında toplam izni 40+24=64 saat olmakta, dördüncü ve son gün ise gece vardiyasına kadar çalışmadığında 64+8=72 saat izin yapmış olmaktadır ki bu durum ardışık sıralı 1002 vardiyalarını göstermektedir. Buna göre EK5 için herhangi bir haftada 000 ve 1002 vardiyaları tespit edilecek şekilde hesaplama yapılmıştır. Bu atama varyasyonlarını diğer atamalardan ayırt edebilmek, yani diğerleri arasından bu atamaları tespit edebilmek için sezgisel olarak ve deneme

yanılma yöntemiyle excelde formülizasyon yapılmış, sonucunda Çizelge 3.12'deki kodlar yazılmıştır:

Çizelge 3.12. EK5 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
ElliYasUstuGebeVeMazeretliHemsireSayisi=0; %kodlar birlestirlirken bu satir silinmelidir.
cEK5=10;
fonksiyonDegeriEK5=0;
m=15;
n=28;
b=round(0+2*rand(m,n)) %kodlar birlestirlirken bu satir silinmelidir.
[m n]=size(b);
sayac1EK5=0;
sayac2EK5=0;
sayac3EK5=0;
Ardisik000Arama=0;
Ardisik1002Arama=0;
Ardisik000ve1002Arama=0;
h=4;
for i=1:m-ElliYasUstuGebeVeMazeretliHemsireSayisi; %calismayan hemşirelerin olduğu satirlar için hesaplama yapılmaz
  for hh=1:h;
     for j=(7*(hh-1)+4):((7*hh)); %hh 1:4 aralığında dolaşırken j 4:7 aralığında dolaşır.
     Ardisik000Arama=(b(i,j-3)+b(i,j-2)+b(i,j-1))*(b(i,j-2)+b(i,j-1)+b(i,j));
     Ardisik1002Arama=abs(b(i,j-3)-b(i,j-2)-b(i,j-1)-1)+abs(b(i,j)-b(i,j-1)-b(i,j-2)-2);
     Ardisik000ve1002Arama=Ardisik000Arama*Ardisik1002Arama;
       if Ardisik000ve1002Arama==0;
       sayac1EK5=sayac1EK5+1;
       else
       sayac1EK5=sayac1EK5;
       end
     End
  if sayac1EK5>=1;
  sayac2EK5=sayac2EK5+1; %sayac2EK5'in değeri 0'dan farklıysa o haftada en az 1 adet ardışık 72 saat izin olduğu anlamina gelmektedir.
  sayac1EK5=0; %haftalik tarama sonrasi sayac sifirlanir.
  else
  sayac2EK5=sayac2EK5;
  sayac1EK5=0; %haftalik tarama sonrasi sayac sifirlanir.
  end
  End
   sayac3EK5=sayac3EK5+sayac2EK5;
   sayac2EK5=0; %satirin tamaminin taranmasi yani 4 haftalik tarama sonrasi sayac sifirlanir.
End
fonksiyonDegeriEK5=cEK5*sayac3EK5;
disp(fonksiyonDegeriEK3);
```

EK6 - Planlama boyunca ihtiyaç duyulan haftalık hemşire sayısı hep sabit: Bu kısıta göre 2-4 haftalık çizelge için 15 hemşirenin tamamının vardiyalara atanması sağlanmalıdır. Bu nedenle 50 yaş üstü, gebe ve/veya mazeret izinli hemşireler için çalışmayan toplam hemşire sayısı kadar satıra 0 ataması yapılmıştır (Çizelge 3.13):

Çizelge 3.13. EK6 alt amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
imin=m-ElliYasUstuGebeVeMazeretliHemsireSayisi+1;
for i=imin:m;
  for j=1:n;
    b(i,j)=0;
  End
End
```

Böylece çalışmayan hemşirelerin vardiyalarına en alt satırdan başlanarak 0 ataması yapılmış olmaktadır.

EK7 - Aynı hemşire hafta sonu 2 gün çalışmamalıdır: Bu kısıta göre sorumlu hemşire için ayrılan çizelgenin birinci satırı haricinde, hafta sonları yani çizelgenin her satırında 6-7, 13-14, 20-21 ve 27-28. sütun çiftleri taranarak her ikisine de vardiya atandığı durumlar esnek kısıtlar için belirlenen ceza puanıyla çarpılmalıdır. Diğer yandan bu kısıtın "EK2 - Her hemşireye eşit sayıda hafta sonu tatili atanmalıdır" kısıtı ile birlikte düşünülmesi uygun olacaktır. Haftalık bazda dengenin korunabilmesi için herhangi bir hemşirenin herhangi bir haftada hiç çalışmadığı ya da hafta sonu 2 günde birden çalıştığı durumların engellenmesi halinde her hemşirenin hafta sonu Cumartesi ya da Pazar günlerinden birinde gece ya da gündüz vardiyasına atanmasının uygun olacağı görülmektedir. Bu da hafta sonu vardiya çiftleri için 0-1, 1-0, 0-2 ya da 2-0 atamalarını uygun kılmaktadır. Kod yazılımında haftalık vardiya kalıpları kullanıldığı için aday 53 adet haftalık vardiya kalıplarının tamamında hafta sonu atamaları bu kısıta uyduğundan ayrıca bu esnek kısıt için alt amaç fonksiyonu değeri hesaplanmasına gerek kalmamıştır. Yine de yazılımda bu değer 0 olarak gösterilmiştir.

EK8 - Acil servisin yoğun olduğu akşam 19.00-23.00 saatleri arasında 4 saatlik vardiya dışı mesai yapılabilir (bu kısıt her gün için atama sağlayacak ama mesaiye çağırma görevi o akşam çalışan acil servis hekimi tarafından sağlanacak. Yoğun olmayan günlerde çağırılmayacak: Acil servis hekiminin o gün vardiyası olmayan ve 2.Grup vardiyalarda çalışan hemşirelerden birini çağırması uygun görünmektedir. Bu durumda 4 saatlik vardiya dışı mesaiye çağrılan hemşirenin aynı haftada ya da aynı 4 haftalık çizelgede sonraki vardiyalarından birinde 4 saat daha az çalışması yeterlidir. Planlandığı gibi 1.Grup vardiyalarda çalışan hemşireler hastanenin kesintisiz 7 gün 24 saat hizmet vermesini sağlayan minimum sayıda hemşirelerden oluşacağı düşünülmektedir. Dolayısıyla 2.Grup vardiyalara atanan hemşireler, herhangi

bir vardiyada gündüz ve gece vardiyalarına en az 1 hemşirenin atanmış olması nedeniyle vardiyada bulunan en az ikinci hemşire olacaklardır. Yani 2.Grup vardiyalarda çalışan hemşirelerin çalışmaması ya da başka vardiyalara geçmesi hastanenin kesintisiz hizmet vermesini engellemeyecektir. 2.Grup vardiyada çalışan bir hemşirenin 4 saatlik vardiya dışı mesaiye çağrılması ve aynı haftada ya da aynı 4 haftalık çizelgede sonraki vardiyalarından birinde 4 saat daha az çalışmasının sağlanması, 4 haftada optimum hedef olan 160 saat toplam çalışma süresini koruyacağından ve hastanenin kesintisiz hizmetine zarar vermeyeceğinden en uygun seçenek olarak görünmektedir. Dolayısıyla bu kısıt için bir hesaplama yapılmasına ya da kod yazılımına gerek bulunmamaktadır. Yine de yazılımda bu değer 0 olarak gösterilmiştir.

Tüm zorunlu ve esnek kısıtlar için optimum çözümü veren alt amaç fonksiyon değerleri hesaplandıktan sonra bunların toplamı AF1 ana amaç fonksiyon değerini verecek şekilde toplanmıştır.

Çizelge 3.14. AF1 ana amaç fonksiyon değerinin hesaplayan Matlab program kodları

```
411 fonksiyonDegeriAF1=fonksiyonDegeriZK1+fonksiyonDegeriZK2+fonksiyonDegeriZK3+fon
    ksiyonDegeriZK4+fonksiyonDegeriZK5+fonksiyonDegeriZK6+fonksiyonDegeriZK7+fonksiy
    onDegeriZK8+fonksiyonDegeriEK1+fonksiyonDegeriEK2+fonksiyonDegeriEK3+fonksiyon
    DegeriEK4+fonksiyonDegeriEK5+fonksiyonDegeriEK6+fonksiyonDegeriEK7+fonksiyonDeg
    eriEK8;
```

Programda, fonksiyonDegeriAF1 sıfırdan büyük olduğu sürece çizelgenin yeniden oluşturulması sağlanmıştır (Çizelge 3.15).

Çizelge 3.15. AF1 ana amaç fonksiyon değeri sıfırdan büyük olduğu sürece işleme devam edilmesini sağlayan Matlab program kodları

```
58 while(fonksiyonDegeriAF1>0)
```

Ancak Çizelge 3.15'teki durumun fonksiyonDegeriAF1 için 0 değeri bulunamadığında sonsuz bir döngü oluşturmaması için her denemede bir artan ve ana döngüyü gösteren bir sayaç (sayac) ile sınırlandırılması sağlanmıştır

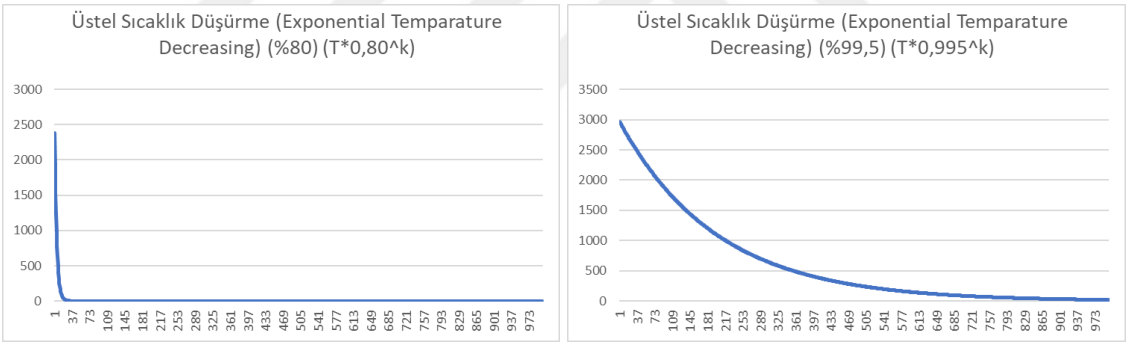
Çizelge 3.16. AF1 ana amaç fonksiyon değeri için sıfıra erişilemediğinde oluşabilecek sonsuz döngüyü engellemeye yönelik Matlab program kodları

```
55 while(sayac<10000) %ana dongu
```

4.1.2. Yeni bir sıcaklık düşürme önerisi: Çoklu tavlama

Genel tavlama algoritmasında isteğe göre üstel ya da logaritmik soğutma (sıcaklık düşürme) kullanılabilmektedir. Ancak Aarts, Korst ve Michiels'e (2007) göre T yerine T_k'nın gerçek sıcaklık, T_0'ın ilk sıcaklık ve k'nın kontrol parametresi olduğu $T_k = T_0/logk$ logaritmik soğutma kullanılması durumunda, tavlama benzetimi algoritmaları genel en iyi çözüme daha kolay ulaşmaktadır.

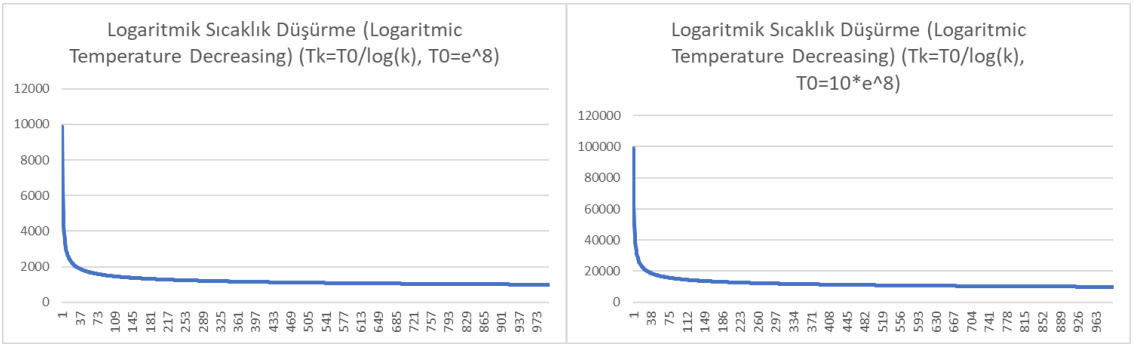
Üstel sıcaklık düşürmede sıcaklık düşürme sabiti olarak tanımlayabileceğimiz bir değişken ile ("d" diyelim) sıcaklığın hızlı ya da yavaş düşmesi sağlanabilmektedir. Şekil 3.1'de üstel sıcaklık düşürmede farklı soğutma (d) sabitlerinin kullanıldığı fonksiyonlar görülmektedir.



Şekil 4.1. Üstel Sıcaklık Düşürme (solda 0,80^k, sağda 0,995^k ile soğutma)

Şekil 3.1'de solda d=0,80, sağda ise d=0,995 ile sıcaklık düşürme fonksiyonlarında k sabiti iterasyon ya da döngü/deneme sabiti olup 1'den 1.000'e kadar birer birer artmaktadır. d soğutma sabitinin buradaki örnekteki gibi 0,80 şeklinde göreli küçük bir değerde seçilmesi 'hızlı soğutma' (veya hızlı tavlama), yine buradaki örnekteki gibi 0,995 gibi göreli yüksek bir değerde seçilmesi ise 'yavaş soğutma' (veya yavaş tavlama) olarak adlandırılmaktadır. Hızlı tavlamada sıcaklık keskin bir şekilde çok kısa sürede 0'a çok yakın bir değere inmekte, ardından uzun bir süre boyunca 0'a yavaş yavaş yaklaşmaya devam etmektedir. Yavaş tavlamada ise sıcaklık düşüşü daha yumuşaktır.

Logaritmik sıcaklık düşürmede ise sıcaklığın yakınsadığı değer ilk seçilen sıcaklık değerine ve logaritmik sabitin değerine ve değişim (k) miktarına bağlıdır. Şekil 3.2'de logaritmik sıcaklık düşürmeye 2 örnek verilmiştir.

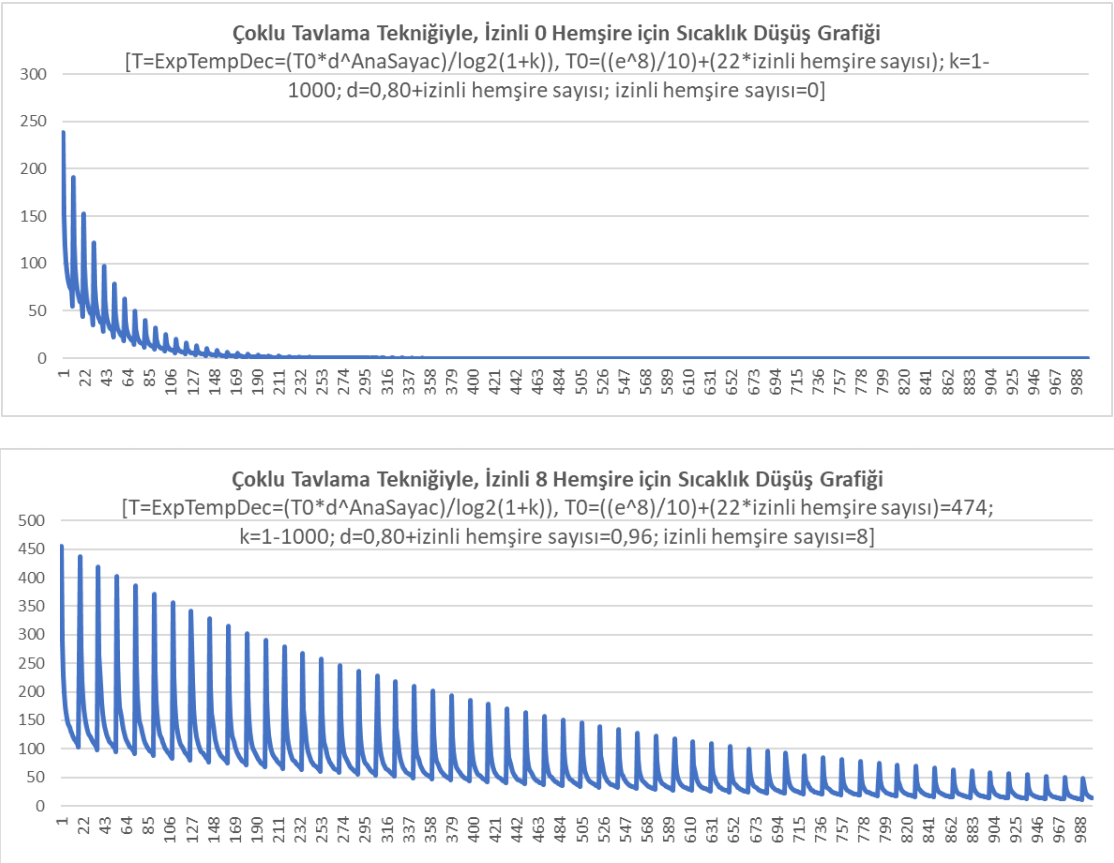


Şekil 5.2. Logaritmik Sıcaklık Düşürme (log(k) ile [k=1-1.000] solda T0=e^8 [yaklaşık 2981] ve sağda T0=10*e^8 [yaklaşık 29810] ile soğutma)

Şekil 3.2'de ilk sıcaklık (T0) olarak sabit bir değeri kullanan fonksiyonda k değişkeni iterasyon/döngü değişkeni olup 1'den 1.000'e kadar birer birer artmaktadır. Soldaki örnekte ilk sıcaklık (T0) değeri için e^8 yani yaklaşık 2981 kullanılmış, bu nedenle sistem sıcaklığı yaklaşık 10bin gibi yüksek bir değerden başlayarak k=31 değerinde 2000'in altına, k=938 için 1.000 değerinin altına inmekte k=1.0000 içinse 745 gibi bir sıcaklık değerine düşmektedir. Dolayısıyla çok hızlı bir şekilde 30 iterasyonda 10.000 gibi bir sıcaklıktan 2.000'in altına düşerek uzun süre (10.000 iterasyonda) 745 gibi bir değere varmakta, 2000-745 arası bir değerde oldukça ağır düşmekte, sıfıra değil seçilen değerler için 10bin denemede 745 gibi bir değere düşmektedir. Sıcaklığın sıfıra değil de belirli bir değer aralığına düşmesi istendiğinde logaritmik soğutma, değerlerin doğru ayarlanmış olması halinde idealdir. Yine Şekil 3.2'de yer alan sağdaki grafikte ise T0 ilk sıcaklık değeri soldakinin 10 katı değerde seçilmiş, dolayısıyla sistem sıcaklığının yakınsadığı değer de 10 kat yükselmiştir. Yeni durumda sistem sıcaklığının en fazla zaman geçireceği sıcaklık aralığı 20.000 ila 7.450 arasındaki değerlerdir ki bu da yine k için 31-10.000 iterasyon aralığındadır. Eğer doğru ilk sıcaklık (T0) seçilmezse sistem istenen sıcaklık aralığına ya da sıcaklık hedefine asla düşmeyebilir. Halbuki üstel sıcaklık düşürmede sistem 1.000 ya da 10.000 iterasyon için 0 değerine çok kısa sürelerde oldukça yaklaştığı için istenen herhangi bir sıcaklığa inmesi de daha kolay olmaktadır.

Araştırmamızda genel olarak tavlama algoritmasında kullanılan üstel ve logaritmik soğutma yöntemlerine alternatif olarak her iki yöntemi birlikte kullanan dolayısıyla çoklu tavlama yöntemi olarak adlandırabileceğimiz bir teknik geliştirilmiştir. Buna göre 500'ü aşkın deneme sonrasında, ele alınan hemşire çizelgeleme probleminde çözümü en hızlı veren sıcaklık denklemi T=(T0*d^ana sayaç)/log2(1+k) olarak bulunmuştur. Burada d sıcaklık düşürme sabiti olup d=0,80+(0,02*İzinli hemşire sayısı) ile ifade edilmektedir. Örneğin hiçbir hemşirenin izinli olmadığı durumda d=0,80 olurken 8 hemşirenin yani araştırmamızdaki maksimum sayıda hemşirenin izinli olması halinde ise d=0,96 değerini almaktadır. Bunun nedeni şöyle açıklanabilir: 15 hemşirenin tamamının çalışır durumda olduğu hallerde kısıtların sağlanması daha kolay olmakta, bir diğer deyişle yerel en iyilere takılmadan genel en iyi çözümü bulmak daha kolay olmaktadır. Bu nedenle sıcaklığın yavaş düşürülerek daha iyi olmayan çözümlerin kabul edilmesine pek ihtiyaç bulunmamaktadır. d=0,80 gibi bir sıcaklık düşürme ile izinli hemşire sayısının az olduğu ya da hiç olmadığı durumlarda çözüme hızlıca ulaşılabilmektedir. Ancak izinli hemşire sayısı artmaya başladığında bir diğer deyişle daha az sayıda çalışan hemşire ile kısıtların sağlanması gerektiğinde yerel en iyilerde takılma riski doğmakta, bir diğer deyişle yerel en iyileri seçe seçe genel en iyi çözüme ulaşabilme imkânı güçleşmektedir. Bunun nedeni de tavlama algoritmasında sıcaklığa bağlı olarak değişen P değeridir ki bu konuya Bölüm 2.2.3.4'te değinilmişti. Oradaki açıklamalardan da anlaşılacağı üzere sistemin ilk sıcaklık değerinin yeterli bir yükseklikte olması daha iyi olmayan yani yerel en iyi çözümlere takılarak genel en iyi çözüme ulaşmayı engelleyen çözümlerden daha kötü olan çözümlerin seçilebilmesini sağlamaktadır. Bir diğer deyişle sıcaklık ne kadar yüksekse, daha iyi olmayan çözümlerin kabul edilmesi yani rastsallığa izin verilmesi o denli fazla olmaktadır. Ayrıca sıcaklık düşüşü yavaşsa da daha iyi olmayan çözümlerin kabul edilme olasılığını ifade eden $1/1+P(\Delta E,T)$ olasılığı daha uzun süre 1'e yakın değerlerde olmaktadır ki bu da daha uzun süre yerel en iyi çözümlerden kaçabilmek anlamına gelmektedir. Ancak sıcaklık yeterince yüksek bir değerin çok üzerinde seçilmişse ve sıcaklık düşüşü çok yavaşsa bu durumda da her zaman yerel en iyi çözümlerden kaçan ve genel en iyiyi bulmakta da zorlanan bir durum oluşabilir. Dolayısıyla tavlama algoritmasında sıcaklık düşürme tekniğinin, ilk sıcaklığın ve sıcaklık düşürme sabitinin seçimi oldukça önemlidir. Araştırmamızda doğru seçimin yapılabilmesi için yüzlerce deneme yapılmış, nihayetinde hem logaritmik hem de üstel (logaritmik-üstel) bir sıcaklık düşürme tekniği

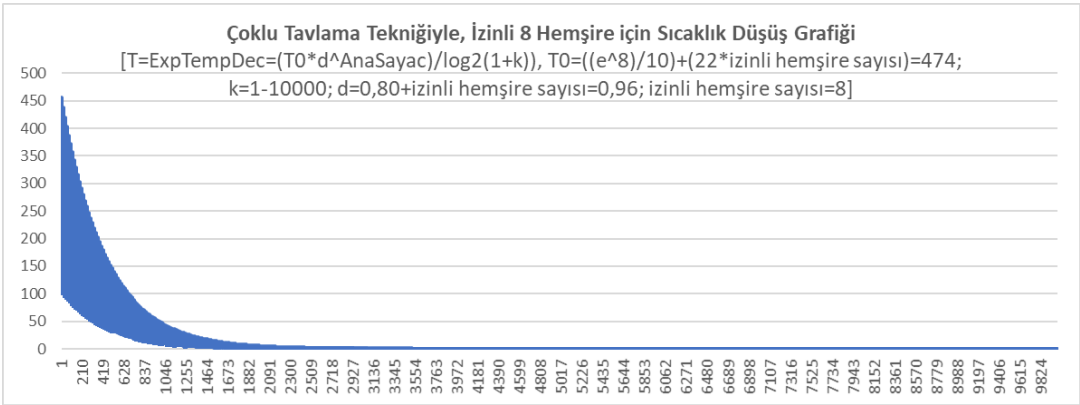
uygulanmıştır. Bu tekniğe göre, sistem sıcaklığının farklı sayıda izinli hemşireye göre değişimi Şekil 3.3'te gösterilmiştir:



Şekil 6.3. 1.000 iterasyonda 0 (üstteki grafik) ve 8 (alttaki grafik) izinli hemşire için Çoklu Tavlama (Logaritmik-üstel sıcaklık düşürme)

Bu tekniğe göre k=1'den (kmaks.=10+İzinli hemşire sayısı)'na kadar her döngüsünü tamamladığında ana sayaç bir artmış, böylece logaritmik sıcaklık düşüşü devam ederken bir anda yani her k döngüsünün tamamlanması sonrasında sıcaklık tekrar yükseltilmiştir. Şekil 3.3'te üstte yer alan grafikte 15 hemşirenin tamamının çalışıyor olması yani hiçbirinin izinli olmaması durumunda 0,80 soğutma sabiti ile hızlı tavlama yapıldığından ilk 1.000 iterasyonun 200'üncü iterasyonunda sistem sıcaklığı 0'a oldukça hızlı bir şekilde yaklaşmakta, sonrasında düşüşü yavaşlamaktadır. Birim döngü olarak adlandırdığımız k'nın kmaks'a kadarki döngüsü üstteki grafikte toplam 10 döngüdür. k değeri 10'a ulaştığında ana döngü sayısını belirleyen ana sayaç 1 arttığından sistemin sıcaklığı bir anda yükselmekte, sonrasında ise yine 0,80 oranında hızlı tavlama nedeniyle hızlıca düşmektedir. Alttaki grafikte ise izinli hemşire sayısı 8'dir. Bu sayının yüksek olması sistemin ilk sıcaklığını artırmaktadır. İlk 1.000 iterasyonun başında 474 ila 100 arasında değişen sistem sıcaklığı ilerleyen

iterasyonlarda daha az değişim göstermektedir. Ancak d soğutma sabitinin 0,80 değil de burada 0,96 olması tavlamanın yavaş soğutma ile yapılmasını sağlamaktadır. Yani sistem sıcaklığı 0,80 soğutma sabitine göre daha yavaş düşmektedir. Yine de her iki grafikte de her k iterasyonu içinde sıcaklığın düşüş eğrisi benzerdir. Aralarındaki fark izinli hemşire sayısının yüksek olduğu durumlarda tavlamanın daha yavaş olması, ilk sıcaklığın daha yüksek olması, minimum-maksimum sıcaklık farkının daha yavaş düşmesi ve sistem sıcaklığının 1.000 iterasyona doğru daha yüksek (daha iyi ve hızlı çözüme ulaşma amacıyla özellikle seçilen) bir sıcaklık aralığına yakınsamasıdır. Az sayıda çalışan hemşire için genel en iyi çözümü bulmadaki zorluk yavaş tavlamada ve daha yüksek ilk sıcaklık sayesinde çözülebilmiştir. Bir diğer deyişle tavlama çalışan hemşire sayısına göre ayarlanmış ve çoklu tavlama sayesinde de sistemin anlık hızlı sıcaklık yükselişleriyle daha kötü çözümleri anlık kabul etme olasılığı yükseltilmiştir. Bu da sıcaklığın doğru ayarlanamaması durumlarında bile sistemin çözümü bulmada rastsallık sayesinde daha rahat hareket etmesini sağlamaktadır. Şekil 3.3'te 1.000 iterasyon için verilen grafiklerin 1.0000 iterasyon için görünümleri Şekil 3.4'te verilmiştir:



Şekil 7.4. 1.0000 iterasyonda 0 (üstteki grafik) ve 8 (alttaki grafik) izinli hemşire için Çoklu Tavlama (Logaritmik-üstel sıcaklık düşürme)

Şekil 3.4'te renkli olarak görülen kısım bir önceki şekilde (Şekil 3.3'te) ilk 1.000 iterasyon için daha yakından ve dataylı görünen grafiğin 10.000 iterasyon için daha uzaktan görünümü olarak düşünülebilir. Dolayısıyla 10.000 iterasyon için grafik, çıkış ve inişleri gösterememektedir. Ancak renkli kısımdaki hareket Şekil 3.3'te olduğu gibi kısa sürede keskin çıkış ve yine kısa sürede ancak çıkış süresinden daha uzun bir sürede düşüş, ardından daha yavaş düşüş şeklinde bir sıcaklık artış azalış döngüsü söz konusudur. Şekil 3.4'te görüleceği gibi izinli hemşire sayısındaki artış durumunda ilk 1500-2000 iterasyonda sıcaklık değişimleri daha yüksek ancak iterasyon sayısına bağlı olarak düşmektedir. Gerek hızlı gerek yavaş tavlamada sıcaklık logaritmik-üstel soğutma tekniğinde alt sıcaklık değerinde daha fazla zaman geçirmekte, üst sıcaklık değeri hızlı bir şekilde düşerek sıcaklık dalgalanma miktarını azaltmaktadır.

4.1.3. Yeni bir atama tekniği önerisi: Çifte atama

Araştırmamızda geliştirilen bir diğer yöntem de değişken tavlama tekniği olarak adlandırabileceğimiz yöntemdir. Bu yönteme göre ana amaç fonksiyon değerinin belirli bir kritik değere ininceye kadar atama tekniği seçimini çizelgenin tamamını haftalık vardiya kalıplarıyla rastgele değiştirmesi, ana amaç fonksiyon değeri belirlenen kritik değerin altına indiğinde ise sadece rastgele tek bir haftayı haftalık vardiya kalıplarından rastgele seçilen biriyle değişmesi sağlanmaktadır. Çifte atama tekniği olarak adlandırabileceğimiz bu iki ayrı atama tekniğinin kullanılmasındaki amaç bu tekniklerden sadece biri kullanıldığında sistemin çözüme ulaşma süresinin uzamasıdır. Çalışırken her iterasyonda alt amaç ve ana amaç fonksiyon değerleri ekrana yazdırılan algoritmada belirli değerlerde algoritmanın tıkandığı ve uzun süre o kritik değerin altına inmekte zorlandığı görülmüştür. Kritik değerin izinli hemşire sayısına göre farklılık gösterdiği tespit edilmiş, 0-8 izinli hemşire durumunda gözlemlenen kritik değerlere uygun sonuç verecek denklem *Kritik değer=140+70*log2((2*İzinli hemşire sayısı)+1)* olarak belirlenmiştir. Bu denklem 0-8 izinli hemşireye göre sırasıyla 140-250,94-302,53-336,51-361,89-382,16-399,03-413,48-426,12 kritik değerlerini vermiştir. Ancak algoritmanın 0-2 izinli hemşire için 140'tan daha yüksek değerlerde uzun süre takılması, 7-9 izinli hemşire içinse uzun süre *Kritik değer=140+70*log2((2*İzinli hemşire sayısı)+1)* formülünde belirlenen 400'lü

değerlerin çok üzerinde (750-1000 aralığında) takılması nedeniyle, izinli hemşire sayısının 0 olduğu durumlar için kritik değer istisnai olarak 251 olarak, 7 ve üzeri izinli hemşireler içinse kritik değer *Kritik değer=251+(İzinli hemşire sayısı-2)*100*; ayrıca düzenlenmiştir. Bu doğrultuda atama tekniği değişikliği için kritik değerler 0-9 izinli hemşireye göre sırasıyla 251-251-251-336,51-361,89-382,16-399,03-751-851-951 kritik değerlerini vermiştir (EK-3, 116-123. satırlar).

Böylece algoritmanın 0-2 izinli hemşire için ortalama 30 saniyenin altında, 3-5 izinli hemşire için ortalama 45 saniyenin altında, 6-8 izinli hemşire için ortalama 60 saniyenin altında en iyi çizelgeyi oluşturması sağlanabilmiştir. Algoritma 0-8 hemşire için ortalama 34 saniyede en iyi çizelgeyi bulabilmektedir. Hatta çizelgede kullanıcıdan input olarak (50 yaş üstü ve gebe hemşireler ile mazeret izni kullanan hemşire sayılarının toplamı olan) izinli hemşire sayılarını alan kodlarda 50 yaş üstü ve gebe hemşireler için üst sınır olan 3 hemşire sınırı 1 artılarak izinli toplam hemşire sayısının 8'den 9'a yükseltilmesi sonrası algoritma yeniden çalıştırılmış ve olabilecek en zorlu durum yani izinli 9 hemşire, çalışan 6 hemşire için 100 kez test edilmiş, tamamında 5.000 (min.456-maks.4.864) iterasyonun (500 saniyenin) altında çözüme ulaştığı, ortalama 1.953 iterasyonda (ort. 195 saniye) en iyi çizelgeyi bulduğu görülmüştür.

4.1.4. Algoritmanın komut ekranı görüntüsü

Algoritma çalıştırıldığında her denemede ekrana özellikle yazdırılan ve algoritmanın çözüme ulaşmada hangi aşamada olduğunu anlamaya yardımcı olan değerler bulunmaktadır (Şekil 3.5).

Şekil 8.5. Geliştirilen algoritmada ekrana sürekli yazdırılan ve algoritmanın çözüme ulaşmada hangi aşamada olduğunu anlamaya yardımcı olan değerler

```
Daha kotu olan yeni cozumun kabul edilme olasiligi: 2.95061e-05
Karsilastirilan Z olasilik degeri: 0.826669
Olasilik karsilastirmasinda dusuk olasilikli oldugu icin kabul edilmeyen kotu amac fonksiyonu degeri: 0
Amac fonksiyonunun son cozum degeri: 0
Calisabilir durumdaki toplam hemsire sayisi     :     7
Izinli durumdaki (50 yas ustu, gebe ya da mazeret izinli) toplam hemsire sayisi: 8
Deneme sayisi: 198
Bulunan en iyi cozum sayisi: 2
Daha iyi oldugu icin kabul edilen cozum sayisi: 22
Daha kotu olup kabul edilen cozum sayisi: 49
Daha kotu olup reddedilen cozum sayisi: 127
Ilk sicaklik: 474.096
Son sicaklik: 9.58691
ZK1'in zorunlu kisit amac fonksiyon degeri     :     0
ZK2'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK3'un zorunlu kisit amac fonksiyon degeri     :     0
ZK4'un zorunlu kisit amac fonksiyon degeri     :     0
ZK5'in zorunlu kisit amac fonksiyon degeri     :     0
ZK6'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK7'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK8'in zorunlu kisit amac fonksiyon degeri     :     0
EK1'in esnek kisit amac fonksiyon degeri       :     0
EK2'nin esnek kisit amac fonksiyon degeri      :     0
EK3'un esnek kisit amac fonksiyon degeri       :     0
EK4'un esnek kisit amac fonksiyon degeri       :     0
EK5'in esnek kisit amac fonksiyon degeri       :     0
EK6'nin esnek kisit amac fonksiyon degeri      :     0
EK7'nin esnek kisit amac fonksiyon degeri      :     0
EK8'in esnek kisit amac fonksiyon degeri       :     0
Zorunlu ve esnek kisitlarin toplam amac fonksiyon ilk degeri: 1210
Zorunlu ve esnek kisitlarin toplam amac fonksiyon sondan bir onceki degeri: 0
Zorunlu ve esnek kisitlarin toplam amac fonksiyon son degeri: 0
Atama yontemi degisimi icin kritik toplam amac fonksiyon degeri: 851
```

Şekil 3.5'teki komut ekranı çıktısı, izinli 8 çalışan 7 hemşire için 198 denemede en iyi sonuca ulaşan algoritmada 198 kez ekrana yazdırılan değerler grubunun 198. yani son yazdırılışını göstermektedir. Bu değerlerin çözüme ulaşmadan önceki süreçte 16.denemedeki örnek görüntüsü aşağıdaki gibidir (Şekil 3.6):

Şekil 9.6. Geliştirilen algoritmada ekrana sürekli yazdırılan ve algoritmanın çözüme ulaşmada hangi aşamada olduğunu anlamaya yardımcı olan değerlerin çözüm öncesi aşamada görünümüne örnek

```
Amac fonksiyonunun son cozum degeri: 920
Calisabilir durumdaki toplam hemsire sayisi     :     7
Izinli durumdaki (50 yas ustu, gebe ya da mazeret izinli) toplam hemsire sayisi: 8
Deneme sayisi: 16
Bulunan en iyi cozum sayisi: 0
Daha iyi oldugu icin kabul edilen cozum sayisi: 5
Daha kotu olup kabul edilen cozum sayisi: 7
Daha kotu olup reddedilen cozum sayisi: 4
Ilk sicaklik: 474.096
Son sicaklik: 92.7902
ZK1'in zorunlu kisit amac fonksiyon degeri     :   800
ZK2'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK3'un zorunlu kisit amac fonksiyon degeri     :     0
ZK4'un zorunlu kisit amac fonksiyon degeri     :     0
ZK5'in zorunlu kisit amac fonksiyon degeri     :   100
ZK6'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK7'nin zorunlu kisit amac fonksiyon degeri    :     0
ZK8'in zorunlu kisit amac fonksiyon degeri     :     0
EK1'in esnek kisit amac fonksiyon degeri       :     0
EK2'nin esnek kisit amac fonksiyon degeri      :     0
EK3'un esnek kisit amac fonksiyon degeri       :    20
EK4'un esnek kisit amac fonksiyon degeri       :     0
EK5'in esnek kisit amac fonksiyon degeri       :     0
EK6'nin esnek kisit amac fonksiyon degeri      :     0
EK7'nin esnek kisit amac fonksiyon degeri      :     0
EK8'in esnek kisit amac fonksiyon degeri       :     0
Zorunlu ve esnek kisitlarin toplam amac fonksiyon ilk degeri: 1410
Zorunlu ve esnek kisitlarin toplam amac fonksiyon sondan bir onceki degeri: 920
Zorunlu ve esnek kisitlarin toplam amac fonksiyon son degeri: 920
Atama yontemi degisimi icin kritik toplam amac fonksiyon degeri: 851
```

Şekil 3.6'da görüleceği gibi rastgele oluşturulan algoritmadan hemen sonra komple çizelgenin tamamına haftalık vardiya kalıplarından yapılan atamalar sonucunda zorunlu ve esnek kısıtların toplam amaç fonksiyon ilk değeri, yani ana amaç fonksiyon değeri (AF1) 1.410'dur. 16.denemede yani çizelgeye 16 kez rastgele komple atama yapıldıktan sonra bu değer 920'ye inmiştir (ZK1:800 + ZK5:100 + EK3:20=920). Bu değer atama yöntemi değişimi için kritik toplam amaç fonksiyon değeri olan 851'e ininceye dek haftalık vardiya kalıplarından yapılan atamalar komple çizelgedeki tüm hemşirelerin tüm haftalarına yapılmaktadır. Bu değerin yani toplam ana amaç fonksiyon değerinin 851'in altına inmesi sonrasında atama tekniği değişmekte ve rastgele herhangi bir haftanın tamamına haftalık vardiya kalıplarından rastgele seçilen birini atama şeklinde olmaktadır. Ana amaç fonksiyon değerinin 851'in altına inmesi sonrasında atama tekniği değişip yeni atama yapıldığında kritik değerin altında ve üstünde bir süre dalgalanma olabilmekte, bu süreçte atama yöntemi bazen yaklaşık 0-10 saniye kadar sürekli değişebilmektedir. Sistemin sıcaklığı düştükçe rastsallığa verilen izin olasılığı da azaldığından kritik değerin altına ilk inilmesinden en geç yaklaşık 10 saniye sonradan itibaren amaç fonksiyon değeri kademeli bir düşüş göstererek (indiği miktardan daha az yükselerek) 0'a doğru yaklaşmakta, 0'a çok yaklaştığında yani çözüme birkaç haftalık

değişiklik ile ulaşılabileceği aşamada amaç fonksiyon değerinin düşüşü tekrar yavaşlamaktadır.

Şekil 3.5’te gösterilen değerleri takiben sonucun bulunması halinde ekrana yazdırılan aşağıdaki sonuç değerleri gelmektedir (Şekil 3.7):

Şekil 10.7. Geliştirilen algoritmada komut ekranına yazdırılan örnek sonuç değerleri

```
Rastgele atama ile olusturulan baslangic cizelgesi
    0    2    1    0    2    2    1    2    2    1    1    0    2    2    1    1    0    0    2    1    1    0    0    0    0    0    0    2
    1    1    0    2    0    2    0    0    2    2    1    0    1    1    1    0    1    1    2    1    2    1    0    0    2    2    1    1
    1    1    1    0    0    2    1    1    1    1    1    0    0    0    1    1    1    1    2    1    1    0    0    0    1    0    0    2
    0    1    1    2    2    1    1    1    1    1    2    2    0    0    2    1    2    2    1    0    2    2    0    0    2    1    1    0
    1    2    1    2    0    2    1    0    2    1    1    1    1    1    1    2    1    1    0    0    1    1    1    2    1    2    0    1
    2    2    0    1    1    0    1    0    0    0    2    1    2    0    1    0    0    1    0    0    1    0    2    1    1    2    1    1
    2    0    2    0    0    0    1    1    0    1    2    0    1    2    1    2    1    1    1    1    0    0    2    0    0    2    2    1
    1    1    1    2    1    0    1    0    2    1    2    1    1    1    1    0    1    0    0    1    2    1    1    0    2    1    0    1
    1    1    1    2    0    1    2    2    0    2    0    2    1    1    2    0    1    0    0    1    1    1    2    1    1    1    2    1
    2    1    1    1    1    1    1    0    2    0    1    1    2    2    1    0    1    0    0    0    0    1    1    0    1    0    0    0
    0    1    1    1    2    1    1    2    1    1    1    1    1    0    1    1    2    2    2    0    0    1    1    1    1    0    1    2
    0    0    2    1    1    1    1    2    2    0    0    0    1    0    0    0    1    0    1    1    0    0    1    0    2    2    1    2
    1    1    1    1    2    2    1    1    2    1    2    1    2    1    1    1    2    1    1    2    1    0    1    0    1    2    1    2
    1    1    2    0    2    1    1    0    2    2    1    1    2    0    0    2    1    2    0    1    2    2    0    1    1    2    0    0
    1    2    0    2    0    0    0    0    0    2    2    1    0    2    1    2    0    2    2    1    0    2    0    0    2    1    0    1


Bulunan en iyi cozum sayisi: 2
Tavlama Algoritmasi ile elde edilen en iyi cizelgeler
    1    1    1    1    1    0    0    1    1    1    1    1    0    0    1    1    1    1    1    0    0    1    1    1    1    1    0    0
    2    0    1    1    0    0    1    2    0    0    1    1    1    0    0    1    0    2    0    0    2    0    0    1    0    2    0    2
    0    0    1    2    0    2    0    1    2    0    1    0    0    1    1    0    1    0    2    0    1    2    0    2    0    0    1    0
    1    0    1    0    2    0    1    0    0    2    0    2    0    1    1    0    2    0    1    1    0    1    0    2    0    1    1    0
    0    2    0    2    0    0    1    1    1    0    0    1    2    0    0    1    2    0    0    2    0    1    1    0    2    0    0    1
    0    2    2    0    0    1    0    0    0    1    0    2    0    2    2    0    0    1    1    1    0    1    2    0    0    1    1    0
    0    1    1    0    1    0    2    2    0    0    2    0    1    0    0    2    0    0    1    0    2    0    0    1    1    1    2    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8
    1    1    1    1    1    0    0    1    1    1    1    1    0    0    1    1    1    1    1    0    0    1    1    1    1    1    0    0
    2    0    1    1    0    0    1    2    0    0    1    1    1    0    0    1    0    2    0    0    2    0    0    1    0    2    0    2
    0    0    1    2    0    2    0    1    2    0    1    0    0    1    1    0    1    0    2    0    1    2    0    2    0    0    1    0
    1    0    1    0    2    0    1    0    0    2    0    2    0    1    1    0    2    0    1    1    0    1    0    1    2    0    0    1
    0    2    0    2    0    0    1    1    1    0    0    1    2    0    0    1    2    0    0    2    0    1    1    0    2    0    0    1
    0    2    2    0    0    1    0    0    0    1    0    2    0    2    2    0    0    1    1    1    0    1    2    0    0    1    1    0
    0    1    1    0    1    0    2    2    0    0    2    0    1    0    0    2    0    0    1    0    2    0    0    1    1    1    2    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8   -8
```

```
Calisabilir durumdaki toplam hemsire sayisi     :     7
Izinli durumdaki (50 yas ustu, gebe ya da mazeret izinli) toplam hemsire sayisi: 8
ZK1'in zorunlu kisit amac fonksiyon degeri      :     0
ZK2'nin zorunlu kisit amac fonksiyon degeri     :     0
ZK3'un zorunlu kisit amac fonksiyon degeri      :     0
ZK4'un zorunlu kisit amac fonksiyon degeri      :     0
ZK5'in zorunlu kisit amac fonksiyon degeri      :     0
ZK6'nin zorunlu kisit amac fonksiyon degeri     :     0
ZK7'nin zorunlu kisit amac fonksiyon degeri     :     0
ZK8'in zorunlu kisit amac fonksiyon degeri      :     0
EK1'in esnek kisit amac fonksiyon degeri        :     0
EK2'nin esnek kisit amac fonksiyon degeri       :     0
EK3'un esnek kisit amac fonksiyon degeri        :     0
EK4'un esnek kisit amac fonksiyon degeri        :     0
EK5'in esnek kisit amac fonksiyon degeri        :     0
EK6'nin esnek kisit amac fonksiyon degeri       :     0
EK7'nin esnek kisit amac fonksiyon degeri       :     0
EK8'in esnek kisit amac fonksiyon degeri        :     0
Zorunlu ve esnek kisitlarin toplam amac fonksiyon ilk degeri: 1210
Zorunlu ve esnek kisitlarin toplam amac fonksiyon sondan bir onceki degeri: 0
Zorunlu ve esnek kisitlarin toplam amac fonksiyon son degeri: 0
Atama yontemi degisimi icin kritik toplam amac fonksiyon degeri: 851
Deneme sayisi: 198
Bulunan en iyi cozum sayisi: 2
Daha iyi oldugu icin kabul edilen cozum sayisi: 22
Daha kotu olup kabul edilen cozum sayisi: 49
Daha kotu olup reddedilen cozum sayisi: 127
Sicaklik dusurme kontrol sabiti: 0.8
Ilk sicaklik: 474.096
Son sicaklik: 9.58691

Elapsed time is 17.6833 seconds.
>>
```

Şekil 3.7'de algoritmanın sonuç aşamasında ekrana yazdırılan bilgiler sırasıyla şunlardır: Rastgele atama ile oluşturulan başlangıç çizelgesi, bulunan en iyi çözüm sayısı, tavlama algoritmasıyla elde edilen en iyi çizelgeler, çalışabilir durumdaki toplam hemşire sayısı, izinli durumdaki (50 yaş üstü, gebe ya da mazeret izinli) toplam hemşire sayısı, zorunlu ve esnek kısıtların alt amaç fonksiyonu değerleri, zorunlu ve esnek kısıtların toplam amaç fonksiyon ilk değeri, zorunlu ve esnek kısıtların toplam amaç fonksiyon sondan bir önceki değeri, zorunlu ve esnek kısıtların toplam amaç fonksiyon son değeri, atama yöntemi değişimi icin kritik toplam amaç fonksiyon değeri, deneme sayısı,bulunan en iyi çözüm sayısı, daha iyi olduğu için kabul edilen çözüm sayısı, daha kötü olup kabul edilen çözüm sayısı, daha kötü olup reddedilen çözüm sayısı, sıcaklık düşürme kontrol sabiti, ilk sıcaklık, son sıcaklık ve toplam işlem süresi (elapsed time).

4.1.5. Birden fazla en iyi çizelgenin en hızlı sürede elde edilmesi

Sonuç komut ekranında tavlama algoritmasıyla elde edilen en iyi çizelgeler birbiri peşisıra eklenen tek bir matris halinde verilmiş, bu matrisleri ayırmak için sembolik olarak tek bir satırın tüm hücrelerinde -8 kullanılmıştır. Algoritmanın en iyi çözümü, yani tüm kısıtlara uyan ilk en iyi çizelgeyi bulduktan sonra bir kmaks döngüsü (izinli hemşire sayısına bağlı olarak kmaks=10+izinli hemşire sayısı denkleminden bulunmakta, 10-19 arasında değişmektedir) boyunca daha kötü çözüm bulununcaya dek en iyi çizelgeleri tek bir matriste toplayan ve alt alta gösteren yapıda olması sağlanmıştır. Normalde algoritma en iyi çözümü bulduğunda duracak şekilde yazılmış

ancak tekrar tekrar çalıştırılmadan birden fazla alternatif sunabilmesi için sonrasında en iyi 10 çözümü bulacak şekilde geliştirilmiş, ancak en iyi 1.çözüm çizelgesinin bulunmasından en iyi 10.çözüm çizelgesinin bulunmasına kadar geçen süre rastsallığa bağlı olarak oldukça değişken ve kimi zaman 10.000'lik döngünün tamamlanmasına kadar varabilen oldukça uzun bir sürece yol açabildiğinden hem hız hem de birden fazla alternatif ihtimali için algoritmanın ilk en iyi çizelgeyi bulduktan sonra içinde bulunduğu iç döngü yani kmaks'a kadarki döngü (10-19 arasında değişmektedir) tamamlanıncaya kadar en iyi çizelge bulduğu anda onları da çözüm matrisine ekleyeceği şekle kavuşturulmuştur. Böylece algoritmanın ilk en iyi çizelgeyi bulduktan sonra rastsal olarak 0 ila 18 arasında değişen sayıda diğer en iyi çizelgeleri de vermesi sağlanmıştır. Bu yeni durumda da bazen çözüm tek bir en iyi çizelge halinde verilmektedir ancak genel olarak verilen en iyi çizelge sayısı 2 ve üzeri sayıda olmaktadır. Bu da hem en kısa sürede çözüme erişmeyi hem de birden fazla çizelge alternatifi sunabilmesi nedeniyle bu haliyle daha etkin bir yapıda olmasını sağlamaktadır.

Çizelge 3.16. İç döngü bitinceye kadar en iyi çizelgeleri tek bir matriste toplayan Matlab program kodları

```
431     if fonksiyonDegeriAF1==0;
432       CozumSayisi=CozumSayisi+1;
433         for i=1:m;
434           for j=1:n;
435           EniyiOnMatris(i+(m+1)*(CozumSayisi-1),j)=EniyiOnMatris(i+(m+1)*(CozumSayisi-
      1),j)+b(i,j);
436           EniyiOnMatris((m+1)*CozumSayisi,j)=-8;
437           End
438         End
439       fprintf('Bulunan en iyi cizelge-%d\n', CozumSayisi);
440       EniyiCizelgeler=EniyiOnMatris(1:(m+1)*(CozumSayisi),1:n);
441       disp(EniyiCizelgeler);
442     Else
443     End
```

Çizelge 3.16'da verilen kodlar, algoritmanın 3 yerinde bulunmaktadır. İlk olarak yeni bulunan çözümün (yani yeni hesaplanan ana amaç fonksiyon değerinin) mevcut çözümden (yani mevcut ana amaç fonksiyon değerinden) küçük (yani daha iyi) olması halinde doğrudan kabul edildiği, if fonksiyonDegeriAF1-FonksiyonDegeriAF1<0; kodunun yer aldığı 425.satırdan sonra (Ek-3) 431-443.satırlar arasında (yalnızca yeni bulunan çözüm değerinin, yani ana amaç fonksiyon değerinin sıfır olması halinde

gerçekleşen); ikinci olarak yeni bulunan çözümün (fonksiyonDegeriAF1'in) mevcut çözümden (FonksiyonDegeriAF1'den) küçük olmaması durumunda geçilen 442.satırdaki else komutu sonrasında daha kötü olan yeni çözümün kabul edilmesi halinde 455-467.satırlar arasında (yalnızca yeni bulunan çözüm değerinin, yani ana amaç fonksiyon değerinin sıfır olması halinde gerçekleşen); üçüncü ve son olarak ise tüm döngülerin sona ermesi halinde ekrana yazdırılan komut ekran görüntüsünde yer alan 540-545.satırlar arasında yer almaktadır.

4.2. Tartışma

Hemşire çizelgeleme problemlerinde, çalışan hemşire sayısının az (örneğin 10'dan az), kısıt sayısının fazla (örneğin 10'dan fazla), hastanenin 24 saat kesintisiz hizmet vermesi gerektiği, günlük vardiya sayısının az (3 yerine 2 gibi), vardiya sürelerinin birbirinden farklı (örneğin gündüz 8 saat, gece 16 saat gibi), her hemşireye gece vardiyasından daha fazla gündüz vardiyası atama zorunluluğu olduğu durumlarda çözüme ulaşmak oldukça zordur. Bunlar arasından özellikle gece vardiyalarının 16 saat yani gündüz vardiya sürelerinin 2 katı olduğu bir durumda 28 günlük bir çizelgede hastanenin günde 24 saat hizmet verebilmesi için yatay eksende yani satırlarda hemşirelerin, dikey eksende yani sütunlarda günlerin gösterildiği bir çizelgede her gün en az birer adet 1 ve 2 değeri yani gündüz ve gece ataması olması gerekmektedir. Bu araştırmada ele alınan problemde özellikle her sütunda en az 1 adet 2 değerinin yani gece atamasının olmasının sağlanması çalışan hemşire sayısı azaldıkça zorlaşmaktadır. 28 günlük bir çizelge her sütunda en az bir adet 2 değeri olması, yani çizelgenin tamamında en az 28 adet 2 ataması bulunması demektir. Herhangi bir hemşire için 28 günlük atamada verilen kısıtlara uyulabilmesi için en fazla 6 adet 2 ataması bulunabilmektedir. Dolayısıyla çizelgenin tamamı için en az 28 adet 2 atamasının elde edilebilmesi için en az 5 adet hemşire bulunması gerekmektedir. Buna haftasonu çalışmayan ve sadece gündüz vardiyalarına atanabilen sorumlu hemşire de eklendiğinde, araştırmamızda ele alınan problemin çözümünün en az çalışabilir durumdaki 6 hemşireyle mümkün olabildiği görülmektedir. Çizelgenin 28 gününün her birinde yer alan gündüz ve gece vardiyalarına en az 1 kişi atanabildiğinde (ZK1) bile her hemşireye atanan gündüz vardiyalarının sayısının gece vardiyaları sayısından fazla olması (EK3) kısıtını sağlamak güçleşmektedir. Dolayısıyla biri zorunlu (ZK1) diğer esnek olmak (EK3) üzere bu iki kısıt, araştırmada ele alınan problemde izinli 9 hemşire,

yani çalışan 6 hemşire durumunda, verilen kısıtlara göre en iyi çözümü vermesi istenen algoritmanın çözüme ulaşma süresini en fazla uzatan kısıtlardır. Araştırmamızda geliştirilen algoritmada ortalama her 10 iterasyonun 1 saniye zaman aldığı göz önünde bulundurulduğunda izinli 0-8 izinli hemşire durumunda ortalama 600 iterasyonun (60 saniyenin) altında, izinli 9 hemşire durumunda ise %100 oranında 5.000 iterasyonun (500 saniyenin) altında, ortalama 1953 iterasyonda (ort. 195 saniyede) çözüme ulaşmasının, ele alınan çizelgeleme problemindeki kısıt sayısının (16 adet) ve gün sayısının (28 gün) oldukça fazla olduğu ve algoritmanın kısıtların tamamına uyan en iyi çözümü her zaman sağlayabildiği göz önünde bulundurulduğunda, çözüme ulaşılan iterasyon sayısının da oldukça başarılı olduğu düşünülmektedir. Kundu, Mahato, Mahanty ve Acharyya'nın (2008) çalışmalarında basitleştirilmiş kısıtlar kullanılarak 7 ve 14 günlük çizelgeler için tavlama benzetimi algoritmasının 7 günlük çizelgede verilen 100 problemin 88'ini ortalama 0,77 saniyede çözdüğü, 14 günlük çizelgede ise verilen 100 problemin 92'sini ortalama 2,85 saniyede çözdüğü bildirilmiştir. Ele alınan kısıt sayısının 5'ten az ve gün sayısının 7 veya 14 gün olduğu durumlarda çözüme daha hızlı ulaşmak mümkündür. Ancak 15 hemşire için 14 günlük çizelgeleme probleminde çözüme 10.000 iterasyon içinde ulaşılamayan çalışmalar da mevcuttur. Örneğin; Hatta Chen, Lu, Lu ve Zhu'nun (2017) çalışmasında, sadece 3 kısıtın sağlanması koşulunu içeren, 15 hemşire için 14 günlük istenen çizelgeleme probleminin çözümüne, her 1.000 iterasyonda sıcaklığı 0,99 (yavaş tavlama) ya da 0,95 (hızlı tavlama) yöntemiyle çözüme 100.000 iterasyon içinde erişildiği belirtilmiştir. İlgili çalışmada sıcaklığın 1.000 iterasyonda bir kez 0,95 (%95) ya da 0,99 (%99) oranında soğutulması yerine her iterasyonda soğutulması şeklinde bir yöntem izlenilmiş olması halinde daha hızlı sonuç alınabileceği düşünülmektedir. Dolayısıyla literatürde, araştırmamızda ele alınan problemle aynı özellikleri taşıyan bir çizelgeleme problemini tavlama algoritmasıyla çözen, yani araştırmamızda elde edilen bulguları doğrudan karşılaştırılabileceği eşdeğer bir çalışmaya rastlanmamıştır.

Araştırmamızla ilgili bir diğer önemli nokta da araştırmanın zorluk seviyesi yüksek hemşire çizelgeleme probleminin çözümüne özel olarak tavlama algoritması kod yazılımının yapılmış olması, bunun çalışan hemşire sayısı başta olmak üzere birden fazla değişkene ve sabite bağlı, logaritmik-üstel soğutma kullanan çoklu tavlama ve kritik değişkene bağlı tavlama tekniği değişikliği ile daha hızlı çözüme ulaşan bir yapıya kavuşturulmuş olmasıdır. Araştırmamızda geliştirilen bu teknik; "Çifte atamalı

çoklu tavlama" olarak adlandırılmıştır. Ele alınan çizelgeleme probleminin 4 hafta (28 gün) gibi geniş bir süre için yapılması, gece ve gündüz vardiya sürelerinin eşit olmaması, 8'i zorunlu, 8'i esnek olmak üzere toplamda 16 gibi uyması gereken ve/veya beklenen oldukça fazla sayıda kısıtı olması, çalışan hemşire sayısının 15'ten 7'ye düşme durumları için esnek olması gibi zorluklar nedeniyle komple satır veya komple sütun veya tek bir hücre değeri veya tek bir hücre komşusu ya da komşuları veya herhangi bir haftanın günlerine rastsal vardiya atanması veya bunların birbirleriyle değiştirilmesi yoluyla çözüme ulaşmayı güçleştirmiştir. Bu nedenle de tüm kısıtlar için gerekli formülasyonlar öncelikle excelde kurgulanmış, tüm zorunlu ve esnek kısıtlar için en iyi çözümü veren toplam 53 adet haftalık vardiya kalıpları (EK-1) bulunmuş ve gerek rastsal olarak komple tüm çizelge haftalarının gerekse yine rastsal olarak çizelgenin tek bir haftasının bu matris havuzundan rastgele çekilen haftalık vardiya kalıplarıyla değiştirilmesi yoluna gidilmiştir. Literatürde Burke, Li ve Qu'nun (2012) da hemşire çizelgeleme probleminin tavlama benzetimi yöntemiyle çözümünde tüm hemşirelerin tüm vardiyaları için yasalara uygun vardiya kalıpları oluşturulduğu ve hemşirelerin her birinin bu vardiya kalıplarına atanması yoluyla zorunlu kısıtların tamamını sağlamaya yönelik çözüme uyarlanabilecek bir yapı sağlandığı görülmektedir. Dolayısıyla araştırmamızda diğer yöntemlere göre daha hızlı sonuç veren vardiya kalıpları kullanımının Burke, Li ve Qu'nun (2012) yaklaşımı doğrultusunda uygun bir tercih olduğu anlaşılmaktadır.

5. SONUÇLAR VE ÖNERİLER

Bu bölümde araştırmanın bulguları doğrultusunda elde edilen sonuçlar özetlenerek, elde edilen sonuçlar doğrultusunda çeşitli önerilerde bulunulacaktır.

5.1. Sonuçlar

Haftanın her günü 24 saat kesintisiz hizmet veren bir hastanede 2 vardiya halinde (08:00-16:00 arası 8 saat ve 16:00-08:00 arası 16 saat) çalışan 15 hemşire için varsayılan kısıtlar çerçevesinde Matlab programında tavlama benzetimi algoritması kullanılarak en iyi çalışma çizelgesinin oluşturulmasının amaçlandığı bu araştırmada elde edilen sonuçlar şöyle özetlenebilir:

- Tavlama Benzetimi algoritmasında farklı sayıda izinli hemşire için ilk sıcaklığın, soğutma tekniği ve sabitinin, atama tekniğinin doğru seçilmesi ve ayarlanması, atamalarda haftalık bazda tüm kısıtları sağlayan (ancak doğal olarak 4 haftalık çizelge için birleştirildiklerinde çeşitli kısıtları ihlal etmesi muhtemel) haftalık vardiya kalıpları kulanılması halinde, bu araştırmada ele alınan ve zorluk seviyesi üst düzeyde olan hemşire probleminin eksik ve fazla mesai oluşturmadan ve verilen 16 kısıtın tamamını sağlayacak şekilde çözümü mümkün olabilmektedir.
- Farklı sayıdaki izinli hemşire durumları için ana amaç fonksiyon değerinin belirli değerlerde uzun süre sabit kaldığı ve o değerlerin altına inmekte zorlandığı görülmüştür. Bu değerler, kritik değer olarak adlandırılmış, farklı sayıdaki izinli hemşireler için ayrı ayrı tanımlanmış, kritik değerin altında ve üstünde farklı atama teknikleri uygulanarak algoritmanın daha hızlı çözüme ulaşması sağlanmıştır. İki atama tekniğinin kullanılması nedeniyle bu tekniğe “çifte atama” adı verilmiştir.

- Bu araştırmada tavlama algoritmasının temel mantığını oluşturan çizelgedeki herhangi bir hücrenin değerinin rastgele değiştirilmesi, çizelgedeki herhangi bir satırın rastgele değiştirilmesi ya da rastgele başka bir satırla değiştirilmesi, çizelgedeki herhangi bir sütunun rastgele değiştirilmesi ya da rastgele başka bir sütunla değiştirilmesi tekniğinin temel yapısı olan çizelgede çok az değişiklik olacak şekilde rastgele atama yaklaşımına sadık kalınmış ancak rastgele atama yaklaşımı tüm kısıtlara uyan haftalık vardiya kalıpları ile yapılmıştır. Belirli bir ana amaç fonksiyonu kritik değerine kadar çizelgenin tamamına haftalık vardiya kalıplarından rastgele atama, kritik değerin altında ise çizelgenin rastgele seçilen herhangi bir hemşiresinin herhangi bir haftasına rastgele haftalık vardiya kalıplarından birini atama tekniği uygulanmıştır. Bu sayede algoritmanın çözüme ulaşmada zorlandığı anda sistem sıcaklığına bağlı olarak hesaplanan olasılık değerinin izin verdiği ölçüde daha rastsal bir çizelge ile yerel en iyiden daha kolay kurtulması ve çözüme daha hızlı ulaşması sağlanmıştır.
- Bu araştırmada geliştirilen logaritmik-üstel soğutma tekniği sayesinde daha iyi olmayan çözümlere izin verme olasılığı iç döngünün her turu başında kısa süre için yükseltilerek, üstel ya da logaritmik soğutma tekniklerinden herhangi birinin tek başına kullanıldığı durumlardan daha hızlı çözüme ulaşılabilmesi sağlanmıştır.
- Bu araştırma kapsamında yazılan ve "çifte atamalı çoklu tavlama" olarak adlandırılan algoritma ile zorluk seviyesi oldukça yüksek olan, 15 hemşire için yapılması istenen, ancak maksimum 8 izinli hemşire durumu için de uygun olması gereken, dolayısıyla 0-8 izinli (7-15 çalışan) hemşire için 8 zorunlu 8 esnek kısıta uyması istenen, 28 günlük bir hemşire çizelgeleme problemi 0-8 izinli (7-15 çalışan) hemşire için ortalama 340 iterasyona karşılık gelen 34 saniyede (min.15-maks.134 sn) çözülebilmiştir. Algoritma, 0-7 izinli (8-15 çalışan) hemşire için tamamen 600 iterasyonun (60 saniyenin) altında, 8 izinli (7 çalışan) hemşire içinse tamamen 1.500 iterasyonun (150 saniyenin) altında çözüme ulaşmıştır. Hatta algoritma, izinli 8 hemşire yerine izinli 9 hemşireye, yani daha yüksek bir zorluk seviyesine kadar işlem

yapabilecek şekilde geliştirilmiş ve çalışan 6, izinli 9 hemşire için ortalama 195 saniyede çözüme ulaşmıştır. Çalışan 6, izinli 9 hemşire için 100 kez çalıştırılan algoritma denemelerin tamamında 5.000 (min.456-maks.4.864) itrasyonun altında çözüme ulaşma başarısı göstermiştir. Araştırma kapsamında geliştirilen algoritma, ileride bu konuda yapılacak araştırmalara örnek teşkil etmesi amacıyla tek parça çalışır halde, 579 satırlık koddan oluşan script halinde Ek-3'te verilmiştir.

- Algoritmada içiçe iki döngü oluşturulması sayesinde en iyi çizelge bulunduktan sonra, iç döngü sayısı olan kmaks=10+izinli hemşire sayısı tamamlanıncaya kadar elde edilen yeni en iyi çözümlerin de ekrana yazdırılması sağlanmıştır. Böylece rastsallığa bağlı olarak herhangi bir sayıda izinli hemşire durumu için sayıları 1-19 arasında değişen ve birbirlerinden bir haftalık atama kadar farklı olan en iyi çizelgeler elde edilebilmiştir. 0-9 izinli hemşire (yani 6-15 çalışan hemşire durumu) için Ek-2'de algoritma ile oluşturulan çizelgelere 3'er örnek verilmiştir.

5.2. Öneriler

Herhangi bir problemin çözümünde en önemli hedefin doğru sonucu en kısa sürede elde etmek olduğu düşünüldüğünde, tavlama algoritmasının komşuluk ilişkisi çerçevesinde mevcut sonuçta rastsal olarak küçük bir değişiklikle yeni çözüm üretmesi ve bu çözümün daha iyi olmaması halinde yine rastsal bir yaklaşımla kabul edilip edilmemesine karar vermesi, çözüme ulaşmada tüm olasılıkları belirli bir sistematiklikle tek tek değerlendiren bazı algoritmalara göre üstün bir özellik olarak kabul edilebilir. Ancak tavlama algoritmasının temel mantığından uzaklaşmadan çözüm süresinin daha da kısaltılabileceği düşünülmektedir. Araştırmada ele alınan hemşire çizelgeleme problemi temelinde 15 hemşire x 28 gün için 15x28=420 hücre (atama) ve her hücrede (atamada) 0-1-2 şeklinde toplam 3 olasılık olabilmesi nedeniyle 3^{420} ($2{,}46 \times 10^{200}$) olasılığı içinde barındırmaktadır. Haftalık vardiya kalıplarının atanması yaklaşımı bu olasılığı toplamda 15 hemşire x 4 hafta = 60 haftaya 53 adet 7 günlük vardiya kalıplarından birinin atanması yaklaşımı ile 60^{53}'e yani $1{,}75 \times 10^{95}$'e indirmiştir. Böylece çözüm evreni $1/1{,}41^{106}$ oranında küçültülebilmiş ve çözüme ulaşma süresi

kısaltılabilmiştir. İleride bu konuda yapılacak akademik çalışmalarda bu sürenin daha da kısaltılabilmesi için şu önerilerde bulunulabilir:

- Öncelikle tüm kısıtlara uyan haftalık vardiya kalıplarını excel temelli bir yaklaşım yerine yine Matlab'ta yazılacak kodlar ile bulan ve bu çalışmada oluşturulan matris havuzunu, yine algoritmadaki kodlarla bulduğu haftalık vardiya kalıpları ile dolduran, sonrasında haftalık vardiya kalıplarından öncelikle yine tüm kısıtlara uyan 14 günlük vardiya kalıpları oluşturup bunlardan yeni bir matris havuzu oluşturan ve böylece haftalık vardiya kalıpları yerine, 2 haftalık (14 günlük) vardiya kalıpları kullanarak çözüm evrenini daha da daraltan bir yaklaşım izlenilmesi önerilebilir.
- Ayrıca çözüm süresinin daha da kısaltılabilmesi için bu önerilen yeni yaklaşımda çizelgedeki rastgele 14 günlük vardiya gruplarından birini değiştirmek yerine yalnızca kısıta uymayan 14 günlük vardiya kalıplarını tespit ederek onların rastsal olarak matris havuzundaki bir 14 günlük vardiya kalıbıyla değiştirilmesi önerilebilir.

KAYNAKLAR

Aarts, E., Korst, J. and Michiels, W., 2007, *Theoretical aspects of local search*, Berlin: Springer.

Abdennadher, S. and Marte, M., 2000, University course timetabling using constraint handling rules, *Journal of Applied Artificial Intelligence* 14 (4), 311–326. https://pdfs.semanticscholar.org/fcc4/fcd5cababd38a32aa2b0b9f113f21f28b30f.pdf [Ziyaret Tarihi: 11 Kasım 2018].

Abramson, D., 1991, Constructing school timetables using simulated annealing: Sequential and parallel algorithms, *Management Science* 37 (1), 98–113. https://pdfs.semanticscholar.org/5edb/60081a48a2ea7a2ba17b8d37ebf374b8360e.pdf [Ziyaret Tarihi: 11 Kasım 2018].

Abramson, D. and Abela, J., 1991, A parallel genetic algorithm for solving the school timetabling problem, Technical Report, *Division of Information Technology, Commonwealth Scientific and Industrial Research Organisation.* https://pdfs.semanticscholar.org/c496/91a77c40528c0d0d80e8f1912ef9b55f5562.pdf [Ziyaret Tarihi: 11 Kasım 2018].

Aickelin, U. and Dowsland, K., 2000, Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem, *Journal of Scheduling,* 3 (3), 139-153, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.63.8186&rep=rep1&type=pdf # [Ziyaret Tarihi: 29 Ekim 2018].

Aickelin, U., and Dowsland, K.A., 2004, An indirect Genetic Algorithm for a nurse-scheduling problem. Computers & Operations Research, 31(5), 761–778. doi:10.1016/s0305-0548(03)00034-0

Alharbi, M. (2018). Nurse scheduling model in saudi arabia hospitals. In International Journal of Computing and Digital Systems, 7,(2) , March 2018, University of Bahrain.

Arora, J. S., 2004, Introduction to Optimum Design, Elsevier Acedemic Press.

Atmaca, E., Pehlivan, C., Aydoğdu, C.B. ve Yakıcı, M., 2012, Hemşire çizelgeleme problemi ve uygulaması. *Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 28 (4), 351–358. http://dergipark.gov.tr/download/article-file/236155 [Ziyaret Tarihi: 10 Kasım 2018].

Aycan, E., 2008, Solving the course scheduling problem by constraint programming and simulated annealing, Yüksek Lisans Tezi, *İzmir Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü*, İzmir.

Azaieza, M.N. and Al Sharif, S.S., 2005, A 0-1 Goal Programing Model for nurse scheduling, *Computers and Operations Research*, Oxford, UK, 491-507. http://www.irantahgig.ir/wp-content/uploads/10025.pdf [Ziyaret Tarihi: 10 Kasım 2018].

Beaumont, N., 1997, Scheduling staff using mixed integer programming, *European Journal of Operational Research*, 98 (3), 473-484. doi:10.1016/s0377-2217(97)00055-6

Bradley, D., and Martin, J., 1991, Continuous personnel scheduling algorithms: A literature review, *Journal of the Society for Health Systems*, 2 (2), 8-23. PMID:1760547.

Brittan, J.N.G. and Farley, F.J.M., 1971, Collage timetable construction by computer, *The Computer Journal*, 14, 361–365. doi:10.1093/comjnl/14.4.361

Brusco, M. J., and Jacobs, L.W., 1993, A simulated annealing approach to the cyclic staff-scheduling problem, *Naval Research Logistics*, 40 (1), 69–84. doi:10.1002/1520-6750(199302)40:1<69::aid-nav3220400105>3.0.co;2-h

Burke, E.K., De Causmaecker, P. and Vanden Berghe, G., 1998, A hybrid tabu search algorithm for the nurse rostering problem, *SEAL'98 Selected papers from the Second Asia-Pacific Conference on Simulated Evolution and Learning on Simulated Evolution and Learning*, 1585, 187-194. https://pdfs.semanticscholar.org/415a/eedb2ea77f5dca6eed3bc10cbf0402a45b05.pdf [Ziyaret Tarihi: 1 Kasım 2018].

Burke, E.K., Cowling, P., De Causmaecker, P., and Vanden Berghe, G., 2001, A memetic approach to the nurse rostering problem," *Applied Intelligence*, 15 (3), 199-214. https://www.researchgate.net/profile/Patrick_De_Causmaecker/publication/220204967_A_Memetic_Approach_to_the_Nurse_Rostering_Problem/links/5563864b08ae6f4dcc98b80c/A-Memetic-Approach-to-the-Nurse-Rostering-Problem.pdf [Ziyaret Tarihi: 1 Kasım 2018].

Burke, E.K., De Causmaecker, P., Petrovic, S. and Vanden Berghe, G., 2004a, Variable neighborhood search for nurse rostering problems, in M.G.C. Resende and J.P. De Sousa (eds.), *Metaheuristics: Computer Decision-Making* (Combinatorial Optimization Book Series), Kluwer, Chapter 7, 153-172. ISBN:1-4020-7653-3.

Burke, E.K., De Causmaecker, P., Berghe, G.V., and Landeghem, H., 2004b, The state of the art of nurse rostering, *Journal of Scheduling*, 7 (6), 441-499. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.456.2058&rep=rep1&type=pdf [Ziyaret Tarihi: 20 Ekim 2018].

Burke, E.K., Li, J. and Qu, R., 2012, Pareto-based optimization for multi-objective nurse scheduling. In: Boros, E. (ed.) *Annals of Operation Research,* 196 (1), 91-109, Springer, US. https://pdfs.semanticscholar.org/ff62/32a89df804ae2188ea5d82c81160f31141e3.pdf [Ziyaret Tarihi: 14 Ekim 2018].

Cheang, B., Li, H., Lim, A. and Rodrigues, B. 2003, Nurse rostering problems–A bibliographic survey, *European Journal of Operational Research,* 151, 447–460. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1030.5363&rep=rep1&type=pdf [Ziyaret Tarihi: 27 Ekim 2018].

Chen, F., Lu, S., Lu, T., ve Zhu, S. (2017). *Nurse Scheduling*. Columbia University. www.columbia.edu/~cs2035/courses/ieor4405.S17/p23.pdf [Ziyaret Tarihi: 04 Ocak 2018].

Clark, A., Moule, P., Topping, A., & Serpell, M., 2013, Rescheduling nursing shifts: scoping the challenge and examining the potential of mathematical model based tools. *Journal of Nursing Management,* 23 (4), 411–420. doi:10.1111/jonm.12158.

Costa, D., 1994, A tabu search algorithm for computing an operational timetable. *European Journal of Operational Research,* 76 (1), 98–110. doi:10.1016/0377-2217(94)90009-4

Creately, 2018, New current nursing staff scheduling process, https://creately.com/app/?tempID=izg30ep42&login_type=demo# [Ziyaret Tarihi: 18 Ekim 2018].

Constantino, A. A., Landa-Silva, D., de Melo, E. L., de Mendonça, C. F. X., Rizzato, D. B., & Romão, W. (2014). A heuristic algorithm based on multiassignment procedures for nurse scheduling. Ann Oper Res, 218(1), 165-183.

Çivril, H., 2009, Hemşire çizelgeleme problemlerinin genetik algoritma ile çözümü. Yüksek Lisans Tezi, *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü,* Isparta.

Çoruhlu, A., 2007, Sınav personel çizelgeleme modeli. Yüksek Lisans Tezi, *Gazi Üniversitesi Fen Bilimleri Enstitüsü,* Ankara.

Deris, S.B., Omatu, S., Ohta, H., and Samat. P., 1997, University timetabling by constraint based reasoning: A case study. *Journal of the Operational Research Society*, 48 (12), 1178–1190. doi:10.1057/palgrave.jors.2600469

Dowsland, K.A., 1998, Nurse scheduling with tabu search and strategic oscillation. *European Journal of Operational Research,* 106 (2-3), 393–407. doi:10.1016/s0377-2217(97)00281-6

Dowsland, K.A., and Thompson, J.M., 2000, Solving a nurse scheduling problem with knapsacks, networks and tabu search. *Journal of the Operational Research Society*, 51 (7), 825–833. doi:10.1057/palgrave.jors.2600970

Eren, T., ve Güner, E., 2007, Sıra-bağımlı hazırlık zamanlı iki ölçütlü çizelgeleme problemi: Toplam tamamlanma zamanı ve maksimum erken bitirme, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi,* 23, (1-2), 95–105. http://dergipark.gov.tr/download/article-file/252291 [Ziyaret Tarihi: 12 Kasım 2018].

Ernst, A.T., Jiang, H., Krishamoorty, M., Owens, B., Sier, D., 2004, Staff scheduling and rostering: A review of applications, methods and models, *European Journal of Operational Research,* 153, 3–27. http://www3.eng.cam.ac.uk/~hj231/Papers/rostering.pdf [Ziyaret Tarihi: 24 Ekim 2018].

GNU Octave. 2018. *GNU Octave Version 4.4.*1. File name: octave-4.4.1-w64-installer.exe. https://ftp.gnu.org/gnu/octave/windows/octave-4.4.1-w64-installer.exe [Ziyaret Tarihi: 6 Aralık 2018].

Gülmez, E. (2014). Minimum weight design of carbon/epoxy laminated composites for maximum buckling load using simulated annealing algorithm. Master's Dissertation, *Graduate School of Engineering and Sciences of İzmir Institute of Technology*.

Hakim, L., & Bakhtiar, T. (2017). The nurse scheduling problem: a goal programming and nonlinear optimization approaches. IOP Conf Ser-Mat Sci, 166(1), 012024.

Hertz, A., 1992, Finding a feasible course schedule using tabu search. *Discrete Applied Mathematics* 35, 55–270. https://core.ac.uk/download/pdf/82288045.pdf [Ziyaret Tarihi: 11 Kasım 2018].

Ikegami, A., and Niwa, A., 2003, A subproblem-centric model and approach to the nurse scheduling problem. *Mathematical Programming*, 97 (3), 517–541. doi:10.1007/s10107-003-0426-2

Isken, I. and Hancock, W., 1990, A heuristic approach to nurse scheduling in hospital units with non-stationary, urgent demand and a fixed staff size, *J Soc Health Syst.,* 2 (2), 24-41. PMID:1836967.

Jaumard, B., Semet, F., and Vovor, T., 1998, A generalized linear programming model for nurse scheduling, *European Journal of Operational Research*, 107 (1), 1–18. doi:10.1016/s0377-2217(97)00330-5.

Jedicke, V., Wilbur, W.L., and Rifai, A.K., 1994, Goal Programming: An Effective Tool in Commercial Bank Management, *Mid-American Journal of Business*, 14 (10), 31-40. https://pdfs.semanticscholar.org/82df/11b7187039e029058e87b327ef9d6c77711f.pdf [Ziyaret Tarihi: 10 Kasım 2018].

Karayel, S.D., & Atmaca, E. (2017). Özel bir hastane için hemşire çizelgeleme problemi. Çukurova Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 21(2), 111-132.

Karaatlı, M., 2010, Bulanık ortamda çok amaçlı işgücü çizelgeleme: Hemşireler için bir uygulama, Doktora Tezi, *Süleyman Demirel Üniversitesi Sosyal Bilimler Enstitüsü,* Isparta.

Karpuz, E., 2015, Nurse scheduling and rescheduling problem under uncertainty, Yüksek Lisans Tezi, *Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü,* Ankara.

Kawanaka, H., Yamamoto, K., Yoshikawa, T., Shinogi, T., and Tsuruoka, S., 2001, Genetic algorithm with the constraints for nurse scheduling problem. *Proceedings of the 2001 Congress on Evolutionary Computation.* doi:10.1109/cec.2001.934317.

Ko, Y.-W., Kim, D.H., Jeong, M., Jeon, W., Uhmn, S., and Kim, J., 2013a, An Efficient Method for Nurse Scheduling Problem using Simulated Annealing, *20*, 82–85. Retrieved from http://onlinepresent.org/proceedings/vol20_2013/20.pdf

Ko, Y.-W., Kim, D.H., Jeong, M., Jeon, W., Uhmn, S., and Kim, J., 2013b, An improvement technique for simulated annealing and its application to nurse scheduling problem. *International Journal of Software Engineering and Its Applications*, *7*(4), 269–278.

Kumar, Bs., Nagalakshmi, G., and Kumaraguru, S., 2014, A shift sequence for nurse scheduling using linear programming problem, *IOSR Journal of Nursing and Health Science (IOSR-JNHS)*, *3* (6), 24-28. http://www.iosrjournals.org/iosr-jnhs/papers/vol4-issue6/Version-6/B04660715.pdf [Ziyaret Tarihi: 5 Kasım 2018].

Kundu, S. and Acharyya, S., 2008, A SAT approach for solving the nurse scheduling problem, *TENCON 2008 - 2008 IEEE Region 10 Conference*, 1–6. http://doi.org/10.1109/TENCON.2008.4766380 [Ziyaret Tarihi: 17 Ekim 2018].

Kundu, S., Mahato, M., Mahanty, B., and Acharyya, S., 2008, Comparative performance of simulated annealing and genetic algorithm in solving nurse scheduling problem. *IMECS 2008 - International Multiconference of Engineers and Computer Scientists (19-21 March 2018)*, *1*. http://doi.org/10.1007/3-540-45014-9

Le, K.N., 2011, A study of evolutionary multiobjective algorithms and their application to Knapsack and nurse scheduling problems (Doctoral Dissertation), *Nottingham: University of Nottingham*.

Li, J., and Aickelin, U., 2006, Bayesian Optimisation Algorithm for nurse scheduling, Pelican M, Sastry K and Cantú-Paz E. (eds), *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Chapter 17, 315-332, Springer,

Lim, G. J., Mobasher, A., Bard, J. F., & Najjarbashi, A. (2016). Nurse scheduling with lunch break assignments in operating suites. Operations Research for Health Care, 10, 35-48.

Namoco, R. A., & Salazar, R. G. (2016). Solving the nurse scheduling problem of private hospitals in the philippines using various operators for genetic algorithm. Indian Journal of Science and Technology, 9(47).

Özcan, E., 2005, *Memetic algorithms for nurse rostering* https://www.researchgate.net/profile/Ender_Ozcan/publication/221579196_Memetic_Algorithms_for_Nurse_Rostering/links/02e7e524026ab4cee6000000/Memetic-Algorithms-for-Nurse-Rostering.pdf [Ziyaret Tarihi: 6 Aralık 2018].

Pinheiro, R. L., Dario, L.S. & Atkin, J. (2015). A variable neighbourhood search for nurse scheduling with balanced preference satisfaction. In: 17th International Conference on

Randhawa, S.U., and Sitompul, D., 1993, A heuristic-based computerized nurse scheduling system, *Computers & Operations Research,* 20 (8), 837–844. doi:10.1016/0305-0548(93)90105-r

Rao, Singiresu S., 2009, *Engineering optimization: Theory and practice,* John Wiley & Sons.

Ramli, M., Abas, Z., Ibrahim, N., & Hussin, B. (2016). Solving complex nurse scheduling problems using particle swarm optimization. International Review on Computers and Software (IRECOS), 11.

Rosocha, L., Vernerova, S., & Verner, R., 2015, Medical staff scheduling using Simulated Annealing, *Quality Innovation Prosperity*, 19 (1), 1–11. http://doi.org/10.12776/qip.v19i1.405 [Ziyaret Tarihi: 21 Kasım 2018].

Santos, D., Fernandes, P., Cardoso, H.L. & Oliveira, E. (2015). A weighted constraint optimization approach to the nurse scheduling problem. 2015 IEEE 18th International Conference on Computational Science and Engineering, 233-239.

Shi, P., & Landa-Silva, D. (2016). Dynamic programming with approximation function for nurse scheduling. International Workshop on Machine Learning, Optimization and Big Data: Springer, 269-280.

Sitompul, D. and Randhawa, S., 1990, Nurse scheduling models: A state-of-the-art review, *J Soc Health Syst.,* 2 (1), 62-72. PMID: 2132334.

Tafida, A.K., 2014. Overall cost optimization of a concrete bridge using simulated annealing algorithm. Yüksek Lisans Tezi, *Atılım Üniversitesi Fen Bilimleri Enstitüsü,* İstanbul.

Tanomaru, J., 2001, Staff scheduling by a genetic algorithm with heuristic operators, *1995 IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century.* https://www.ufsj.edu.br/portal2-repositorio/File/nepomuceno/00489191.pdf [Ziyaret Tarihi: 13 Kasım 2018].

Tein, L., and Ramli, R., 2010, Recent advancements of nurse scheduling models and a potential path, *Proceedings of the 6th IMT-GT Conference on Mathematics, Statistics and its Applications (ICMSA2010).* Kuala Lumpur, Malaysia.

Tekin, E., 2017, Multi-objective nurse scheduling with shift preferences in a surgical suite. Yüksek Lisans Tezi, *Kadir Has Üniversitesi Fen Bilimleri Enstitüsü,* İstanbul.

Tien, J. and Kamiyama, A., 1982, On manpower scheduling algorithms, *SIAM Review*, 24 (3), 275-287. doi:10.1137/1024063

Trilling, L., Guinet, A., and Le Magny, D., 2006, Nurse scheduling using integer linear programming and constraint programming. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, *12* (PART 1). https://www.sciencedirect.com/science/article/pii/S1474667015360602/pdf?md5=653a4e10b24c81251ebf19996a3ebcf3&pid=1-s2.0-S1474667015360602-main.pdf [Ziyaret Tarihi: 7 Kasım 2018].

Trivedi, V.M., 1981, A Mixed-Integer Programming Model for nursing budgeting, *Operations Research*, 29, 1019-1034. https://www.jstor.org/stable/170238 [Ziyaret Tarihi: 10 Kasım 2018].

Türkbey, O., 2002, Tesis düzenlemesi probleminde yerel arama sezgiseli kullanan bir genetik algoritma: Memetik algoritma yaklaşımı. *Pamukkale Üniversitesi Mühendislik Fakültesi Mühendislik Bilimleri Dergisi,* 8 (2), 265-271.

Uslu,B..2018,0-1 Hedef Programlama Yöntemi Kullanılarak Hemşire Çizelgeleme Probleminin Çözümü.ttps://dergipark.org.tr/download/article-file/499600

Varlı, E., & Eren, T. (2017). Hemşire çizelgeleme problemi ve hastanede bir uygulama. APJES, 5(1), 34-40.

Wang, B.A., 2016, Using linear programming to flex nursing staff. http://floridahfma.squarespace.com/s/Xiao-Angelica-Wang.pdf [Ziyaret Tarihi: 4 Kasım 2018].

Warner, M., 1976, Scheduling nursing personnel according to nursing preferences: A mathematical programming approach, *Operations Research*, 24 (5), 842–856. doi:10.1287/opre.24.5.842

Warner, M. and J. Prawda, 1972, A mathematical programming model for scheduling nursing personnel in a hospital, *Management Science*, 19, 411-422. doi:10.1287/mnsc.19.4.411

Valouxis, C., and Housos, E., 2000, Hybrid optimization techniques for the workshift and rest assignment of nursing personnel. *Artificial Intelligence in Medicine,* 20 (2), 155–175. doi:10.1016/s0933-3657(00)00062-2

Wong, T., Xu, M., & Chin, K. (2014). A two-stage heuristic approach for nurse scheduling problem: A case study in an emergency department. Comput Oper Res, 51, 99-110.

EKLER

EK-1 Haftalık Bazda Tüm Kısıtları Sağlayan 7 Günlük (Haftalık) Vardiya Kalıpları

	1	2	3	4	5	6	7	1	2	3	4	5	6	7
1			N		D		N	0	0	2	0	1	0	2
2			D		N		N	0	0	1	0	2	0	2
3			D	N			N	0	0	1	2	0	0	2
4		N			N		D	0	2	0	0	2	0	1
5		N			D		N	0	2	0	0	1	0	2
6		N			D	N		0	2	0	0	1	2	0
7		N		N			D	0	2	0	2	0	0	1
8		N		D		N		0	2	0	1	0	2	0
9		D		N			N	0	1	0	2	0	0	2
10		D		N		N		0	1	0	2	0	2	0
11		D	N			N		0	1	2	0	0	2	0
12	N			N			D	2	0	0	2	0	0	1
13	N			N		D		2	0	0	2	0	1	0
14	N			D		N		2	0	0	1	0	2	0
15	N		N			D		2	0	2	0	0	1	0
16	D		N			N		1	0	2	0	0	2	0
17			D	D	N		D	0	0	1	1	2	0	1
18			D	D	D		N	0	0	1	1	1	0	2
19		D		D	D		N	0	1	0	1	1	0	2
20		D	D		N		D	0	1	1	0	2	0	1
21		D	D		D		N	0	1	1	0	1	0	2
22		D	D	N			D	0	1	1	2	0	0	1
23		D	D	D		N		0	1	1	1	0	2	0
24	N			D	D		D	2	0	0	1	1	0	1
25	N			D	D	D		2	0	0	1	1	1	0
26	N		D	D			D	2	0	1	1	0	0	1
27	D			D	N		D	1	0	0	1	2	0	1
28	D			D	D	N		1	0	0	1	1	2	0
29	D		N		D		D	1	0	2	0	1	0	1
30	D		N		D	D		1	0	2	0	1	1	0
31	D		D		N		D	1	0	1	0	2	0	1
32	D		D	N			D	1	0	1	2	0	0	1
33	D		D	D		N		1	0	1	1	0	2	0
34	D	N			D		D	1	2	0	0	1	0	1
35	D	N			D	D		1	2	0	0	1	1	0
36	D	N		D			D	1	2	0	1	0	0	1
37	D	D			D	N		1	1	0	0	1	2	0
38	D	D		N			D	1	1	0	2	0	0	1
39	D	D		N		D		1	1	0	2	0	1	0

EK-1 (devamı)

	1	2	3	4	5	6	7	1	2	3	4	5	6	7
40	D	D		D		N		1	1	0	1	0	2	0
41	D	D	N			D		1	1	2	0	0	1	0
42	D	D	D	D			D	1	1	1	1	0	0	1
43			N		N		D	0	0	2	0	2	0	1
44			N		D	N		0	0	2	0	1	2	0
45			N	N			D	0	0	2	2	0	0	1
46			N	N		D		0	0	2	2	0	1	0
47			D		N	N		0	0	1	0	2	2	0
48			D	N		N		0	0	1	2	0	2	0
49			D	D	D	N		0	0	1	1	1	2	0
50		N	N			D		0	2	2	0	0	1	0
51	D			D	D		N	1	0	0	1	1	0	2
52	D		D		D		N	1	0	1	0	1	0	2
53	D	D			D		N	1	1	0	0	1	0	2

D: Day (Gündüz vardiyası, 08:00-16:00, 8 saat),
N: Night (gece vardiyası, 16:00-08:00, 16 saat).
0: İzinli olunan gün,
1: Gündüz vardiyası, 08:00-16:00, 8 saat),
2: Gece vardiyası, 16:00-08:00, 16 saat).

EK-2 Farklı Sayıda Çalışan Hemşire İçin Tavlama Benzetimi Algoritmasıyla Elde Edilen İdeal Çizelgelerden Örnekler

0 İzinli, 15 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	1	1	0	2	0	0	0	1	1	2	0	1	0	2	0	0	1	2	0	1	0	1	1	0	2	0
1	1	0	2	0	0	1	0	1	0	1	1	0	2	0	2	0	0	1	0	2	0	2	0	0	1	0	2
0	0	1	1	2	0	1	1	0	1	0	1	0	2	0	2	0	1	0	2	0	0	2	0	2	0	0	1
1	1	1	1	0	0	1	1	2	0	0	1	0	1	0	2	2	0	0	1	0	0	1	0	1	1	0	2
2	0	0	2	0	0	1	2	0	0	1	1	0	1	1	0	1	2	0	0	1	1	1	2	0	0	1	0
0	1	1	1	0	2	0	0	0	1	2	0	2	0	0	1	1	0	2	0	1	0	0	1	1	1	2	0
0	1	2	0	0	2	0	1	1	2	0	0	1	0	1	0	1	2	0	0	1	0	1	1	0	1	0	2
0	2	0	1	0	2	0	1	0	0	1	1	0	2	2	0	2	0	0	1	0	1	1	0	1	0	2	0
0	2	0	0	1	0	2	0	0	1	1	2	0	1	0	1	0	2	0	0	2	0	0	1	1	1	0	2
0	0	1	0	2	2	0	1	2	0	1	0	0	1	0	0	2	2	0	1	0	2	0	0	1	1	0	1
2	0	0	1	1	1	0	2	0	0	2	0	0	1	1	0	2	0	1	1	0	1	0	1	0	2	0	1
1	0	1	2	0	0	1	0	2	0	2	0	0	1	1	0	1	2	0	0	1	0	0	1	1	1	0	2
1	1	0	2	0	0	1	1	0	1	0	2	0	1	0	2	0	0	1	2	0	0	2	0	2	0	0	1
0	2	0	2	0	0	1	1	1	1	1	0	0	1	1	0	1	2	0	0	1	0	1	1	2	0	0	1

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	1	2	0	2	0	1	0	1	2	0	0	1	1	0	0	1	1	2	0	0	0	1	0	2	0	2
1	0	2	0	1	1	0	1	1	0	0	1	2	0	2	0	0	2	0	0	1	1	1	1	1	0	0	1
2	0	0	2	0	1	0	0	0	2	0	1	2	0	1	2	0	1	0	0	1	0	0	1	1	1	2	0
0	0	1	1	1	2	0	0	2	0	1	0	2	0	0	0	2	2	0	0	1	1	1	2	0	0	1	0
0	1	1	0	2	0	1	1	0	1	2	0	0	1	1	1	2	0	0	1	0	0	1	0	2	0	2	0
0	0	2	0	2	0	1	1	2	0	0	1	1	0	0	1	1	0	1	0	2	2	0	2	0	0	1	0
2	0	0	1	1	0	1	1	0	0	1	2	0	1	0	0	2	2	0	1	0	0	0	1	1	2	0	1
1	1	0	0	1	2	0	0	0	1	1	1	2	0	1	0	1	0	1	0	2	0	1	1	2	0	0	1
1	2	0	0	1	1	0	0	0	1	2	0	0	2	0	0	2	0	2	0	1	1	1	1	1	0	0	1
1	0	0	1	2	0	1	0	0	2	0	1	2	0	2	0	1	1	0	0	1	1	1	0	0	1	0	2
0	1	0	2	0	0	2	2	0	0	1	1	0	1	1	1	2	0	0	1	0	1	0	0	1	1	0	2
1	1	0	0	1	2	0	0	0	2	0	1	2	0	1	2	0	0	1	1	0	2	0	0	2	0	0	1
0	2	0	0	1	2	0	1	1	0	2	0	1	0	1	1	0	1	0	2	0	1	2	0	0	1	1	0
0	0	1	1	2	0	1	0	1	0	2	0	2	0	1	2	0	0	1	0	1	0	1	1	0	1	0	2

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	2	0	2	0	1	1	0	2	0	1	1	0	2	0	0	1	1	1	0	0	2	2	0	0	1	0
0	0	1	0	2	0	2	0	0	1	0	2	0	2	0	1	1	0	1	0	2	0	1	1	0	1	0	2
1	2	0	0	1	0	1	1	1	0	1	0	2	0	0	1	2	0	0	2	0	0	2	2	0	0	1	0
0	1	1	0	1	0	2	2	0	1	1	0	0	1	0	1	0	1	1	0	2	0	1	2	0	0	2	0
0	1	1	0	2	0	1	2	0	0	1	0	2	0	1	1	0	2	0	0	1	0	1	1	2	0	0	1
1	0	1	0	2	0	1	2	0	0	1	1	1	0	2	0	0	2	0	1	0	0	0	1	0	2	2	0
1	1	0	2	0	1	0	1	0	2	0	1	0	1	2	0	0	2	0	0	1	2	0	0	2	0	0	1
1	1	1	1	0	0	1	0	0	2	2	0	1	0	0	0	2	0	1	2	0	1	0	1	2	0	0	1
2	0	0	1	1	1	0	1	0	0	1	2	0	1	1	0	1	2	0	0	1	1	0	1	2	0	0	1
0	2	2	0	0	1	0	0	0	1	1	2	0	1	0	2	0	0	2	0	1	1	0	2	0	1	0	1
1	1	0	2	0	0	1	1	1	0	0	1	0	2	0	1	1	0	2	0	1	2	0	2	0	0	1	0
0	2	0	0	2	0	1	1	1	0	1	0	2	0	1	0	1	0	1	0	2	0	1	2	0	0	2	0
0	2	0	0	1	2	0	0	2	0	0	1	0	2	0	0	1	1	2	0	1	0	1	0	1	1	0	2
1	0	2	0	0	2	0	2	0	0	1	1	0	1	1	1	0	0	1	2	0	0	2	0	0	1	0	2

0=İzinli, 1=Gündüz, 2=Gece

1 İzinli, 14 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	1	0	2	0	1	0	0	2	2	0	1	0	0	0	2	2	0	0	1	1	2	0	0	1	0	1
1	2	0	1	0	0	1	1	1	0	0	1	0	2	0	2	0	2	0	0	1	1	1	2	0	0	1	0
2	0	0	2	0	1	0	2	0	0	1	1	0	1	1	0	2	0	1	1	0	1	0	2	0	0	2	0
1	1	0	2	0	0	1	1	1	0	2	0	1	0	0	1	1	1	0	2	0	0	0	1	1	1	0	2
0	0	2	0	1	2	0	0	0	1	1	1	2	0	1	0	2	0	1	0	1	2	0	0	2	0	0	1
1	1	0	0	1	0	2	2	0	0	1	1	0	1	2	0	0	2	0	0	1	0	0	1	0	2	2	0
1	1	0	0	1	2	0	0	0	2	0	1	2	0	1	1	0	1	0	2	0	1	1	0	1	0	2	0
2	0	1	1	0	0	1	0	2	0	0	1	0	2	0	1	1	2	0	0	1	1	0	2	0	1	0	1
0	0	2	2	0	0	1	2	0	1	1	0	0	1	1	0	0	1	2	0	1	0	2	0	1	0	2	0
0	0	1	1	1	0	2	0	0	2	2	0	0	1	0	0	1	1	1	0	2	0	1	1	0	2	0	1
1	1	0	1	0	2	0	2	0	1	1	0	0	1	0	0	1	2	0	2	0	0	0	2	0	1	2	0
1	0	0	1	2	0	1	0	2	0	0	2	0	1	2	0	0	2	0	1	0	2	0	0	1	1	0	1
0	0	1	1	1	0	2	0	1	1	2	0	0	1	1	0	0	1	2	0	1	0	0	1	0	2	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
2	0	0	1	0	2	0	1	2	0	0	1	1	0	0	1	0	2	0	2	0	1	2	0	0	1	0	1
2	0	0	2	0	1	0	1	2	0	1	0	0	1	0	0	1	1	2	0	1	0	1	1	2	0	0	1
0	2	0	0	2	0	1	0	0	1	2	0	2	0	0	2	0	2	0	0	1	1	1	1	1	0	0	1
1	1	2	0	0	1	0	0	1	0	2	0	2	0	1	2	0	0	1	1	0	2	0	0	2	0	0	1
1	1	0	0	1	2	0	1	1	0	2	0	1	0	1	0	0	1	1	0	2	0	0	2	0	2	0	1
0	2	2	0	0	1	0	0	2	0	0	2	0	1	1	0	1	1	0	2	0	1	1	0	2	0	1	0
0	0	1	0	2	0	2	0	0	2	0	2	0	1	1	0	2	0	1	1	0	1	2	0	0	1	1	0
0	0	2	2	0	0	1	0	1	1	2	0	0	1	1	0	1	0	2	0	1	0	0	2	0	1	0	2
0	2	2	0	0	1	0	1	0	2	0	1	0	1	0	0	1	1	2	0	1	1	0	1	1	0	2	0
2	0	0	2	0	1	0	2	0	0	1	1	1	0	1	0	1	0	2	0	1	0	1	1	0	2	0	1
1	0	1	0	1	0	2	0	1	1	0	1	0	2	2	0	2	0	0	1	0	0	0	1	0	2	2	0
1	1	0	2	0	0	1	0	1	1	2	0	0	1	1	0	1	0	1	0	2	0	2	0	0	1	0	2
0	0	1	1	2	0	1	1	1	0	2	0	1	0	1	1	0	0	1	2	0	0	0	1	0	2	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	2	0	0	1	0	2	0	0	1	1	2	0	1	0	1	1	0	1	0	2	2	0	0	2	0	0	1
1	2	0	0	1	1	0	1	1	0	0	1	2	0	1	0	0	1	2	0	1	2	0	1	1	0	0	1
1	2	0	1	0	0	1	0	0	1	1	1	0	2	0	1	1	0	2	0	1	0	0	1	2	0	2	0
0	0	1	1	1	0	2	0	2	0	2	0	0	1	0	2	0	0	2	0	1	1	0	0	1	1	2	0
0	2	0	0	2	0	1	1	0	0	1	2	0	1	0	1	1	2	0	0	1	1	2	0	0	1	1	0
0	0	2	0	2	0	1	1	0	0	1	1	2	0	1	2	0	0	1	0	1	0	0	1	1	1	0	2
0	1	0	2	0	0	2	0	1	1	1	0	2	0	1	2	0	0	1	1	0	0	2	0	0	2	0	1
2	0	0	1	0	2	0	1	0	1	0	2	0	1	2	0	0	1	1	0	1	0	0	1	2	0	0	2
2	0	0	1	0	2	0	2	0	0	1	1	1	0	0	0	1	2	0	2	0	0	0	1	1	1	2	0
2	0	0	2	0	1	0	2	0	0	1	1	0	1	2	0	1	1	0	0	1	1	0	2	0	1	1	0
1	1	2	0	0	1	0	0	2	0	2	0	0	1	2	0	2	0	0	1	0	0	1	1	1	0	2	0
0	0	2	0	1	0	2	0	2	2	0	0	1	0	1	0	2	0	1	1	0	2	0	1	1	0	0	1
0	0	2	0	2	0	1	1	0	2	0	1	0	1	2	0	0	1	1	0	1	2	0	0	2	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

2 İzinli, 13 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	2	0	2	0	0	1	0	1	1	0	2	0	1	0	0	2	2	0	1	0	2	0	0	1	1	1	0
0	2	0	0	2	0	1	1	0	1	0	2	0	1	0	0	1	0	2	0	2	0	1	0	1	1	0	2
1	1	2	0	0	1	0	1	1	2	0	0	1	0	0	1	2	0	0	2	0	0	2	0	2	0	0	1
1	0	1	0	1	0	2	0	2	0	1	0	2	0	1	1	1	1	0	0	1	0	1	2	0	0	2	0
0	0	1	1	1	0	2	2	0	0	1	0	2	0	1	1	2	0	0	1	0	0	0	2	2	0	0	1
0	1	1	1	0	2	0	0	2	0	2	0	0	1	1	1	0	2	0	1	0	2	0	0	1	1	0	1
0	1	1	0	1	0	2	0	1	0	2	0	2	0	0	1	0	2	0	2	0	1	0	1	0	2	0	1
2	0	0	1	1	0	1	0	0	1	2	0	0	2	0	2	0	2	0	0	1	1	0	1	2	0	0	1
0	1	1	0	2	0	1	1	0	2	0	0	2	0	1	1	2	0	0	1	0	1	2	0	0	1	1	0
0	1	2	0	0	2	0	1	1	0	2	0	0	1	0	0	1	1	1	2	0	1	1	0	2	0	1	0
2	0	0	1	1	1	0	1	1	0	1	0	2	0	0	1	2	0	0	2	0	2	0	0	2	0	1	0
2	0	0	1	1	0	1	2	0	0	1	1	0	1	2	0	0	1	1	0	1	1	2	0	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	0	1	1	0	2	0	0	1	0	2	0	2	2	0	2	0	0	1	0	0	0	1	1	2	0	1
2	0	1	1	0	0	1	1	1	0	0	1	2	0	1	1	0	0	1	0	2	0	1	1	0	1	0	2
0	1	1	2	0	0	1	1	0	2	0	1	1	0	0	0	2	0	2	0	1	1	0	0	1	1	2	0
2	0	0	1	0	2	0	0	1	0	2	0	2	0	0	1	1	0	2	0	1	1	1	0	1	0	2	0
1	0	1	0	1	0	2	0	0	2	2	0	1	0	0	1	0	2	0	2	0	2	0	0	1	1	1	0
2	0	0	1	1	1	0	1	0	1	1	0	2	0	0	2	0	0	1	2	0	0	0	2	2	0	0	1
2	0	0	2	0	0	1	0	2	0	1	0	2	0	0	1	1	1	0	2	0	1	0	1	0	1	0	2
0	1	2	0	0	2	0	0	1	1	1	0	2	0	1	0	1	1	0	2	0	0	0	1	0	2	0	2
1	1	1	1	0	0	1	0	2	0	2	0	0	1	0	2	0	1	0	2	0	2	0	0	2	0	1	0
1	1	0	2	0	0	1	2	0	0	1	1	1	0	0	0	1	1	2	0	1	0	2	0	0	2	0	1
2	0	0	1	1	1	0	1	0	1	1	0	2	0	2	0	1	1	0	0	1	0	2	0	0	1	0	2
0	2	0	0	2	0	1	0	1	2	0	0	2	0	0	1	1	0	2	0	1	1	1	0	0	1	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	1	2	0	0	1	0	1	1	2	0	0	1	0	0	2	0	1	2	0	1	1	2	0	0	1	0
0	0	1	1	2	0	1	0	1	1	2	0	0	1	0	0	2	0	1	0	2	0	0	2	2	0	1	0
2	0	0	1	1	0	1	0	2	2	0	0	1	0	2	0	0	1	1	1	0	1	0	1	0	2	0	1
1	1	0	1	0	2	0	0	2	2	0	0	1	0	0	0	2	2	0	0	1	0	1	1	0	1	0	2
0	0	1	1	1	0	2	2	0	1	1	0	0	1	0	2	0	1	0	2	0	0	2	2	0	0	1	0
2	0	0	1	1	1	0	1	1	2	0	0	1	0	2	0	1	1	0	0	1	0	2	0	0	1	2	0
0	2	2	0	0	1	0	1	1	0	1	0	2	0	1	0	0	1	1	0	2	0	0	1	0	2	2	0
1	0	2	0	0	2	0	0	1	0	2	0	0	2	2	0	1	1	0	0	1	2	0	1	1	0	0	1
0	1	1	2	0	0	1	1	2	0	0	1	0	1	0	1	1	1	0	2	0	0	0	1	2	0	2	0
0	2	0	2	0	0	1	1	1	0	1	0	2	0	0	1	1	0	2	0	1	0	1	1	2	0	0	1
1	0	1	1	0	2	0	2	0	0	1	0	2	0	1	2	0	0	1	1	0	1	1	1	1	0	0	1
0	0	2	0	1	2	0	1	0	1	0	2	0	1	0	0	2	2	0	1	0	1	0	1	0	1	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

3 İzinli, 12 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	1	2	0	0	1	0	1	0	1	0	2	0	1	0	1	2	0	0	2	0	1	1	0	2	0	0	1
1	0	0	1	1	2	0	1	2	0	1	0	0	1	1	0	2	0	1	0	1	0	2	2	0	0	1	0
2	0	1	1	0	0	1	1	0	1	1	0	2	0	1	0	0	1	1	2	0	1	0	2	0	1	1	0
0	0	1	2	0	2	0	2	0	0	1	0	2	0	0	1	0	1	1	0	2	0	1	1	0	1	0	2
2	0	0	1	1	1	0	1	2	0	1	0	0	1	0	1	0	2	0	2	0	1	0	0	1	1	2	0
0	1	0	1	1	0	2	0	2	2	0	0	1	0	1	0	0	1	1	2	0	0	1	0	1	1	0	2
0	2	0	1	0	2	0	0	1	1	0	1	0	2	2	0	0	1	1	0	1	1	0	1	0	2	0	1
1	0	2	0	0	2	0	1	0	0	1	1	0	2	2	0	2	0	0	1	0	1	1	0	1	0	2	0
1	0	0	1	2	0	1	1	0	1	2	0	0	1	0	1	1	0	2	0	1	0	1	1	0	2	0	1
0	1	1	1	0	2	0	1	1	1	1	0	0	1	1	2	0	1	0	0	1	2	0	1	1	0	0	1
1	0	0	1	1	2	0	0	0	1	1	2	0	1	1	0	0	1	1	2	0	1	1	0	2	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
2	0	1	1	0	0	1	2	0	0	2	0	0	1	1	0	0	1	2	0	1	0	0	1	1	1	0	2
0	1	0	1	1	0	2	0	2	0	1	0	2	0	0	1	0	1	1	0	2	0	1	1	0	1	0	2
1	0	2	0	0	2	0	1	2	0	0	1	0	1	1	1	0	0	1	0	2	0	1	1	1	0	2	0
0	1	1	2	0	0	1	1	0	2	0	1	0	1	2	0	0	2	0	0	1	0	0	1	1	1	0	2
1	0	0	1	1	2	0	0	0	2	2	0	1	0	2	0	0	1	1	1	0	2	0	0	1	1	0	1
1	0	1	0	2	0	1	2	0	0	1	1	0	1	2	0	0	2	0	1	0	0	0	2	2	0	1	0
0	1	0	2	0	2	0	0	0	2	0	2	0	1	1	0	2	0	1	1	0	2	0	0	1	1	0	1
0	0	1	1	2	0	1	2	0	0	1	1	1	0	1	1	0	1	0	2	0	0	0	2	0	2	0	1
1	2	0	0	1	1	0	0	1	0	2	0	0	2	0	0	1	1	2	0	1	0	1	1	0	1	0	2
1	0	2	0	0	2	0	0	0	1	2	0	2	0	1	2	0	0	1	0	1	1	2	0	0	1	1	0
1	1	0	1	0	2	0	1	0	1	0	2	0	1	1	2	0	1	0	0	1	0	1	0	1	1	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	2	0	0	2	0	0	2	0	0	1	2	0	2	0	0	1	1	0	1	1	0	1	2	0	0	1
0	0	2	2	0	0	1	0	0	1	1	1	0	2	0	1	1	0	1	0	2	0	1	2	0	0	2	0
0	0	2	0	1	0	2	2	0	0	1	1	1	0	1	1	0	2	0	0	1	1	1	0	0	1	0	2
2	0	0	2	0	0	1	1	2	0	0	1	0	1	1	2	0	0	1	1	0	0	2	0	0	1	0	2
2	0	0	1	1	1	0	0	1	1	0	2	0	1	0	0	2	0	2	0	1	1	0	2	0	1	1	0
0	0	1	0	2	0	2	0	1	0	2	0	2	0	2	0	1	1	0	0	1	0	1	1	0	2	0	1
2	0	0	1	1	1	0	0	1	1	0	2	0	1	0	0	1	1	1	2	0	0	0	1	1	1	0	2
1	2	0	0	1	1	0	1	1	0	2	0	0	1	0	1	1	1	0	2	0	1	1	2	0	0	1	0
1	1	0	2	0	0	1	1	0	1	1	0	2	0	2	0	0	1	1	1	0	1	1	0	2	0	0	1
1	0	1	1	0	2	0	0	0	2	0	1	2	0	2	0	0	1	1	1	0	2	0	0	2	0	0	1
1	0	1	2	0	0	1	1	0	0	1	2	0	1	0	2	0	2	0	0	1	1	0	2	0	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

4 İzinli, 11 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	2	0	0	1	1	0	0	2	0	0	1	0	2	2	0	0	1	1	0	1	0	2	2	0	0	1	0
0	0	1	1	2	0	1	0	1	1	1	0	2	0	0	1	1	1	0	2	0	2	0	0	2	0	0	1
1	0	1	0	1	0	2	0	0	1	1	1	0	2	2	0	0	2	0	1	0	2	0	1	1	0	0	1
1	1	0	1	0	2	0	0	0	2	2	0	1	0	0	0	1	1	1	0	2	0	0	2	0	2	0	1
1	0	0	1	2	0	1	0	0	1	1	1	2	0	1	2	0	0	1	0	1	1	1	0	1	0	2	0
0	0	1	1	1	0	2	0	2	0	0	1	2	0	1	1	0	0	1	2	0	1	0	1	0	1	0	2
2	0	0	2	0	1	0	2	0	0	1	1	0	1	2	0	0	1	0	2	0	1	1	0	1	0	2	0
0	2	2	0	0	1	0	0	1	1	0	1	0	2	0	0	2	2	0	0	1	1	2	0	0	1	1	0
1	1	0	0	1	2	0	1	0	1	0	2	0	1	2	0	1	1	0	0	1	1	0	2	0	1	1	0
0	1	0	1	1	0	2	0	0	2	0	2	0	1	1	0	0	1	2	0	1	0	0	1	0	2	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	2	0	2	0	1	0	0	1	1	2	0	1	2	0	0	1	1	0	1	0	1	0	2	0	0	2
1	1	0	1	0	2	0	1	2	0	0	1	0	1	0	1	1	1	0	2	0	0	0	2	0	1	2	0
1	0	2	0	0	2	0	1	0	0	1	1	0	2	0	0	2	2	0	0	1	1	0	1	2	0	0	1
0	0	1	0	2	2	0	1	1	1	1	0	0	1	0	2	0	0	1	0	2	0	2	0	0	2	0	1
2	0	2	0	0	1	0	2	0	0	1	1	0	1	0	2	2	0	0	1	0	0	0	1	1	2	0	1
0	0	2	2	0	0	1	1	1	2	0	0	1	0	1	0	0	1	1	0	2	0	2	0	0	2	0	1
0	1	0	1	1	0	2	0	0	1	1	1	2	0	0	1	0	2	0	2	0	0	0	1	1	2	0	1
0	0	1	1	1	2	0	0	2	0	2	0	0	1	0	1	1	0	1	0	2	2	0	0	2	0	0	1
0	2	0	0	1	2	0	1	1	0	0	1	2	0	1	1	0	0	1	0	2	0	2	0	0	1	0	2
0	1	0	1	1	0	2	0	0	1	1	1	2	0	0	2	0	0	2	0	1	0	0	2	2	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
2	0	0	2	0	0	1	1	0	2	0	1	1	0	1	1	0	0	1	0	2	0	2	2	0	0	1	0
0	0	2	2	0	0	1	1	1	2	0	0	1	0	0	1	1	0	2	0	1	1	1	0	2	0	0	1
1	1	0	2	0	0	1	1	2	0	1	0	0	1	2	0	0	2	0	1	0	1	0	0	1	1	0	2
0	2	0	2	0	0	1	1	2	0	0	1	1	0	0	0	1	2	0	2	0	1	2	0	0	1	0	1
1	0	0	1	1	0	2	0	0	2	0	2	0	1	1	1	2	0	0	1	0	0	2	0	0	2	0	1
0	2	0	1	0	2	0	0	0	2	0	1	2	0	1	1	1	1	0	0	1	0	1	1	0	1	0	2
1	1	0	2	0	1	0	0	1	1	0	1	0	2	2	0	0	2	0	1	0	0	0	1	1	1	2	0
0	1	1	0	2	0	1	2	0	0	2	0	0	1	1	1	0	1	0	2	0	2	0	0	1	0	2	0
0	1	1	0	2	0	1	1	1	1	1	0	0	1	0	2	0	1	0	2	0	2	0	0	2	0	1	0
0	0	2	2	0	0	1	1	0	2	0	1	0	1	0	2	0	0	2	0	1	0	1	1	1	0	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

5 İzinli, 10 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	1	1	0	2	0	1	2	0	0	1	0	1	0	0	1	2	0	0	2	2	0	0	1	1	0	1
2	0	2	0	0	1	0	1	1	2	0	0	1	0	2	0	0	1	0	2	0	0	1	1	0	1	0	2
0	0	1	1	1	2	0	0	1	1	2	0	0	1	0	0	2	2	0	0	1	0	2	2	0	0	1	0
1	0	1	2	0	0	1	1	0	1	2	0	0	1	0	0	1	0	2	2	0	1	1	0	0	1	0	2
1	2	0	0	1	0	1	0	0	1	0	2	0	2	0	0	1	1	2	0	1	1	0	0	1	1	2	0
0	0	1	1	2	0	1	0	0	2	2	0	1	0	0	2	0	0	1	2	0	0	1	1	0	2	0	1
2	0	0	2	0	0	1	1	0	1	1	0	2	0	0	2	2	0	0	1	0	1	0	0	1	2	0	1
0	0	1	0	2	0	2	0	2	0	0	2	0	1	1	0	0	1	1	0	2	0	1	0	1	1	0	2
1	1	2	0	0	1	0	2	0	0	1	1	0	1	0	1	1	1	0	2	0	1	1	0	2	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	2	0	2	0	1	0	2	0	1	0	2	0	0	0	1	1	2	0	1	1	0	2	0	1	1	0
0	0	2	2	0	1	0	0	1	1	2	0	0	1	0	1	1	1	0	2	0	2	0	0	1	0	2	0
0	1	0	1	1	0	2	0	0	1	2	0	0	2	2	0	0	1	1	1	0	1	1	0	0	1	2	0
1	1	0	0	1	2	0	1	2	0	1	0	0	1	0	2	0	0	2	0	1	0	2	0	1	0	2	0
2	0	1	1	0	0	1	1	0	1	2	0	0	1	1	1	2	0	0	1	0	1	0	1	0	2	0	1
1	1	0	1	0	2	0	0	0	1	2	0	0	2	0	0	1	1	1	2	0	0	0	1	2	0	2	0
0	1	0	1	1	0	2	2	0	2	0	0	1	0	0	0	1	1	1	0	2	0	0	1	2	0	0	2
0	2	2	0	0	1	0	1	0	0	1	2	0	1	0	1	1	2	0	0	1	1	0	1	2	0	0	1
1	1	1	1	0	0	1	0	2	0	2	0	0	1	0	0	2	2	0	0	1	1	2	0	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	0	1	2	0	1	0	0	1	1	1	2	0	1	0	0	1	2	0	1	0	1	0	2	0	0	2
0	0	1	0	2	2	0	2	0	0	1	1	1	0	1	0	1	1	0	2	0	0	2	0	1	0	2	0
1	0	0	1	2	0	1	2	0	0	1	1	0	1	1	1	0	1	0	2	0	1	0	0	1	2	0	1
0	1	1	0	2	0	1	0	0	2	0	1	0	2	0	2	2	0	0	1	0	0	1	1	2	0	0	1
1	0	1	0	1	0	2	0	1	1	1	0	2	0	1	0	0	1	1	0	2	0	0	2	0	2	0	1
0	1	1	0	1	0	2	2	0	0	2	0	1	0	0	0	1	1	1	0	2	0	2	0	2	0	0	1
1	1	2	0	0	1	0	0	2	0	0	2	0	1	2	0	0	2	0	0	1	2	0	0	1	1	1	0
1	2	0	0	1	0	1	1	1	0	2	0	0	1	0	2	0	0	2	0	1	1	0	1	0	2	0	1
2	0	0	2	0	0	1	2	0	0	1	1	1	0	2	0	1	1	0	0	1	1	0	2	0	0	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

6 İzinli, 9 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	0	1	1	0	2	0	0	2	2	0	1	0	1	1	0	2	0	0	1	0	0	1	1	1	0	2
2	0	1	1	0	0	1	0	2	0	1	0	2	0	1	1	0	2	0	0	1	0	0	2	2	0	0	1
2	0	0	1	1	1	0	1	0	1	0	2	0	1	0	2	0	0	1	0	2	0	0	1	0	2	2	0
0	2	0	0	2	0	1	0	1	1	0	2	0	1	0	2	2	0	0	1	0	1	0	1	1	0	2	0
1	0	0	1	2	0	1	0	2	0	2	0	0	1	1	2	0	0	1	0	1	1	0	2	0	1	1	0
0	0	2	0	1	2	0	0	1	1	0	1	0	2	0	0	1	1	2	0	1	0	2	0	1	0	2	0
0	1	1	2	0	0	1	2	0	2	0	0	1	0	0	0	2	0	1	2	0	0	1	0	1	1	0	2
1	1	0	1	0	2	0	0	0	2	2	0	0	1	2	0	0	2	0	1	0	2	0	1	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	2	0	0	2	0	2	0	0	1	1	0	1	1	0	2	0	1	0	1	0	0	1	1	2	0	1
0	2	0	0	2	0	1	0	1	1	0	2	0	1	2	0	0	2	0	1	0	1	0	2	0	1	0	1
0	0	2	2	0	1	0	1	0	1	0	2	0	1	0	0	1	0	2	2	0	1	2	0	0	1	1	0
2	0	0	2	0	1	0	0	0	1	1	1	2	0	0	0	2	2	0	0	1	1	1	0	0	1	2	0
0	0	1	0	2	2	0	0	1	1	0	2	0	1	1	0	1	0	1	0	2	0	0	1	2	0	2	0
0	1	0	1	1	0	2	0	0	1	0	2	0	2	0	0	1	1	2	0	1	1	0	1	0	2	0	1
0	2	0	2	0	0	1	1	0	2	0	1	1	0	1	2	0	0	1	1	0	2	0	0	1	0	2	0
0	1	1	1	0	2	0	0	2	0	2	0	0	1	0	0	2	2	0	1	0	0	1	0	1	1	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	2	0	0	2	0	1	1	0	0	1	2	0	1	1	0	1	1	0	2	0	0	0	2	2	0	0	1
1	0	2	0	1	1	0	1	2	0	0	1	1	0	0	2	0	0	1	0	2	0	2	0	0	2	0	1
2	0	0	1	1	0	1	0	1	0	2	0	0	2	2	0	0	1	1	0	1	1	0	1	2	0	0	1
0	0	1	0	2	0	2	2	0	0	1	1	1	0	1	0	0	1	1	0	2	0	0	1	2	0	0	2
1	1	0	2	0	0	1	0	0	2	0	2	0	1	1	2	0	0	1	1	0	2	0	0	2	0	1	0
2	0	0	1	1	1	0	1	0	0	1	1	2	0	0	0	2	0	2	0	1	0	0	2	0	1	2	0
1	1	0	1	0	2	0	0	0	2	2	0	0	1	0	0	2	2	0	1	0	1	2	0	0	1	1	0
2	0	0	1	1	1	0	0	2	0	2	0	0	1	1	0	1	0	2	0	1	0	2	2	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

7 İzinli, 8 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
2	0	1	1	0	0	1	1	0	2	0	0	2	0	0	2	2	0	0	1	0	0	1	0	1	1	0	2
1	1	0	2	0	0	1	2	0	1	1	0	0	1	0	0	1	2	0	0	2	2	0	2	0	0	1	0
0	2	0	0	1	2	0	0	1	1	2	0	0	1	0	0	1	0	2	2	0	1	0	2	0	1	0	1
0	0	1	1	1	0	2	0	1	0	1	1	0	2	2	0	1	1	0	0	1	0	2	0	2	0	0	1
1	0	1	0	2	0	1	2	0	1	1	0	0	1	1	0	1	1	0	2	0	2	0	1	1	0	0	1
0	1	0	1	1	0	2	0	0	1	0	2	0	2	2	0	0	1	0	2	0	1	1	0	0	1	0	2
0	2	2	0	0	1	0	1	2	0	0	1	1	0	1	1	0	0	1	0	2	0	0	1	0	2	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	1	2	0	0	2	2	0	1	1	0	0	1	2	0	0	1	1	1	0	0	2	0	0	1	0	2
1	2	0	1	0	0	1	0	1	2	0	0	2	0	1	0	2	0	1	0	1	1	0	2	0	1	0	1
2	0	0	1	1	0	1	0	2	0	0	2	0	1	1	1	0	2	0	0	1	2	0	0	2	0	1	0
2	0	0	1	1	1	0	1	1	0	1	0	2	0	0	0	2	0	1	2	0	0	0	2	0	1	2	0
0	0	1	0	2	2	0	0	1	0	2	0	0	2	0	0	1	1	1	0	2	2	0	0	1	1	0	1
0	0	2	0	1	2	0	1	0	0	1	1	2	0	0	1	1	0	2	0	1	0	0	1	0	2	2	0
2	0	0	1	1	1	0	1	1	0	2	0	1	0	1	2	0	1	0	0	1	0	2	2	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	2	0	1	0	2	2	0	1	1	0	0	1	0	0	1	0	2	2	0	1	0	2	0	1	0	1
2	0	2	0	0	1	0	1	1	0	0	1	2	0	1	1	0	0	1	0	2	0	0	2	2	0	1	0
1	2	0	1	0	0	1	0	2	2	0	0	1	0	2	0	0	2	0	0	1	2	0	0	1	1	0	1
1	2	0	0	1	1	0	0	0	2	0	1	0	2	0	0	1	0	2	2	0	1	0	1	0	2	0	1
0	0	1	0	2	0	2	0	2	2	0	0	1	0	1	1	2	0	0	1	0	1	0	0	1	1	0	2
0	0	1	2	0	2	0	1	0	1	0	2	0	1	1	2	0	0	1	1	0	1	0	2	0	0	2	0
1	0	1	1	0	2	0	1	1	0	2	0	0	1	2	0	0	1	0	2	0	0	2	0	2	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

8 İzinli, 7 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
2	0	0	1	1	0	1	2	0	0	2	0	0	1	1	1	1	1	0	0	1	0	2	2	0	0	1	0
0	0	1	2	0	0	2	0	0	1	1	1	0	2	2	0	0	1	1	1	0	0	2	0	0	1	2	0
1	0	1	0	2	0	1	2	0	2	0	0	1	0	0	2	0	1	0	2	0	1	0	0	1	1	0	2
1	1	2	0	0	1	0	0	0	1	1	1	0	2	0	1	0	2	0	0	2	0	0	2	0	2	0	1
0	2	0	0	2	0	1	0	0	1	1	1	2	0	0	0	1	1	2	0	1	2	0	0	2	0	0	1
0	0	1	1	1	2	0	0	2	0	0	2	0	1	0	0	2	0	1	2	0	1	1	0	0	1	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	1	0	2	0	1	1	1	0	0	1	2	0	1	0	1	0	1	0	2	2	0	1	1	0	0	1
0	2	0	2	0	0	1	1	0	0	1	1	0	2	0	0	1	1	2	0	1	1	2	0	1	0	0	1
1	1	0	0	1	2	0	0	1	2	0	0	2	0	2	0	2	0	0	1	0	1	0	2	0	1	1	0
0	0	1	1	1	0	2	0	1	1	2	0	0	1	0	2	0	2	0	0	1	1	1	0	0	1	2	0
2	0	2	0	0	1	0	0	2	0	0	2	0	1	1	0	0	1	1	2	0	0	1	1	0	2	0	1
1	0	2	0	1	0	1	2	0	0	2	0	1	0	2	0	0	1	1	0	1	0	0	1	2	0	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
1	0	1	1	0	2	0	0	0	1	0	2	0	2	0	2	2	0	0	1	0	1	0	1	0	2	0	1
0	0	1	1	2	0	1	0	0	1	1	1	2	0	0	0	1	1	2	0	1	0	2	0	0	1	2	0
2	0	0	1	1	0	1	1	1	0	0	1	0	2	2	0	1	1	0	0	1	0	0	2	2	0	0	1
1	2	0	1	0	0	1	2	0	0	1	1	0	1	1	1	0	2	0	1	0	2	0	0	2	0	1	0
0	1	0	1	1	0	2	0	2	0	1	0	2	0	0	1	0	1	1	0	2	2	0	2	0	0	1	0
0	0	2	2	0	1	0	0	0	2	2	0	1	0	1	0	1	1	0	2	0	0	0	1	1	1	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

9 İzinli, 6 Çalışan Hemşire İçin Optimum Çizelge Örnekleri:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	2	0	0	1	0	2	2	0	0	1	0	2	0	1	2	0	0	1	0	1	2	0	1	1	0	0	1
0	0	2	2	0	1	0	0	0	2	2	0	1	0	1	1	0	1	0	2	0	1	1	0	1	0	2	0
1	0	0	1	2	0	1	0	1	1	0	1	0	2	0	0	2	2	0	1	0	0	2	2	0	0	1	0
2	0	0	1	1	0	1	1	0	0	1	2	0	1	0	0	1	0	2	0	2	0	0	1	0	2	0	2
1	0	2	0	0	2	0	0	2	0	0	1	2	0	2	0	1	1	0	0	1	0	1	1	2	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	0	1	0	2	0	2	0	1	1	1	0	2	0	1	1	0	1	0	2	0	0	2	0	1	0	2	0
0	0	2	0	1	2	0	1	2	0	1	0	0	1	0	2	0	0	2	0	1	1	1	0	2	0	0	1
2	0	1	1	0	0	1	0	0	1	0	2	0	2	0	0	2	2	0	0	1	1	0	1	0	1	0	2
0	1	1	2	0	0	1	2	0	0	2	0	1	0	2	0	2	0	0	1	0	2	0	0	1	1	1	0
1	2	0	0	1	1	0	1	0	2	0	1	0	1	0	2	0	0	1	0	2	0	0	2	0	2	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0
0	1	1	0	2	0	1	0	0	1	2	0	0	2	2	0	1	1	0	0	1	0	0	1	2	0	2	0
0	0	1	2	0	2	0	1	2	0	0	1	1	0	0	0	1	0	2	2	0	0	1	0	1	1	0	2
0	2	2	0	0	1	0	1	0	0	1	2	0	1	0	0	1	1	1	0	2	0	2	0	0	2	0	1
2	0	0	2	0	0	1	2	0	0	1	1	1	0	1	2	0	0	1	1	0	0	0	2	0	1	2	0
1	0	0	1	1	0	2	0	0	2	0	1	2	0	0	0	2	2	0	1	0	2	0	0	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0=İzinli, 1=Gündüz, 2=Gece

EK-3 Verilen 8 Zorunlu ve 8 Esnek Kısıtın Tamamını Sağlayacak Şekilde Tavlama Benzetimi Algoritmasıyla Matlab Programında Yazılan 15 hemşire için 28 Günlük Optimum Çizelgeleme Scripti

```
clc;
ElliYasUstuVeGebeHemsireSayisi=input('Lutfen 50 yas ustu ve gebe hemsirelerin toplam sayisini giriniz: ');
while(ElliYasUstuVeGebeHemsireSayisi>4)
disp('50 yas ustu ve gebe hemsire sayisi 4''u gecemez. Lutfen yeniden giris yapiniz');
ElliYasUstuVeGebeHemsireSayisi=input('Lutfen 50 yas ustu ve gebe hemsirelerin toplam sayisini giriniz: ');
End
MazeretliHemsireSayisi=input('Lutfen mazeretli hemsirelerin toplam sayisini giriniz: ');
while(MazeretliHemsireSayisi>5)
disp('Mazeretli hemsire sayisi 5''i gecemez. Lutfen yeniden giris yapiniz');
MazeretliHemsireSayisi=input('Lutfen mazeretli hemsirelerin toplam sayisini giriniz: ');
End
tic;
ElliYasUstuGebeVeMazeretliHemsireSayisi=ElliYasUstuVeGebeHemsireSayisi+MazeretliHemsireSayisi;
m=15;
n=28;
c=100;
k=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
v=b;
sayac3=0;
sayac2=0;
sayac1=0;
AnaSayac=0;
sayaceniyicozumsayisi=0;
CozumSayisi=0;
AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1=0;
    m=15;
    n=28;
    EniyiOnMatris=zeros((300+20),28);
d=0.80+(0.02*ElliYasUstuGebeVeMazeretliHemsireSayisi);
if ElliYasUstuGebeVeMazeretliHemsireSayisi>=7;
 d=0.80;
 Else
End
T0=((e.^8)/10)+(22*ElliYasUstuGebeVeMazeretliHemsireSayisi); %Calismayan hemsire sayisiyla carpilmasinin cozumu hizlandirip hizlandirmadigina bakilacak
E0=71650;
fonksiyonDegeriAF1=E0;
FonksiyonDegeriAF1=E0;
```

```
40   fonksiyonDegeriZK1=2800;
41   fonksiyonDegeriZK2=1;
42   fonksiyonDegeriZK3=6000;
43   fonksiyonDegeriZK4=39000;
44   fonksiyonDegeriZK5=21000;
45   fonksiyonDegeriZK6=1;
46   fonksiyonDegeriZK7=1500;
47   fonksiyonDegeriZK8=1;
48   fonksiyonDegeriEK1=600;
49   fonksiyonDegeriEK2=1;
50   fonksiyonDegeriEK3=150;
51   fonksiyonDegeriEK4=1;
52   fonksiyonDegeriEK5=600;
53   fonksiyonDegeriEK6=1;
54   fonksiyonDegeriEK7=1;
55   fonksiyonDegeriEK8=1;
56 sayac=0;
57 LogaritmikUstelSicaklikT=T0;
58 while(fonksiyonDegeriAF1>0)
59 while(sayac<10000) %ana dongu
60 AnaSayac=AnaSayac+1;
61 kmaks=10+ElliYasUstuGebeVeMazeretliHemsireSayisi;
62 for k=1:kmaks;
63   sayac=sayac+1;
64 LogaritmikUstelSicaklikT=(T0*d^AnaSayac)/log2(1+k);
65   if fonksiyonDegeriAF1>=0;
66   FonksiyonDegeriAF1=fonksiyonDegeriAF1; %ana amaç fonksiyonu değeri, yeni bulunan
     çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
67   FonksiyonDegeriZK1=fonksiyonDegeriZK1; %ZK1 ana amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
68   FonksiyonDegeriZK2=fonksiyonDegeriZK2; %ZK2 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
69   FonksiyonDegeriZK3=fonksiyonDegeriZK3; %ZK3 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
70   FonksiyonDegeriZK4=fonksiyonDegeriZK4; %ZK4 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
71   FonksiyonDegeriZK5=fonksiyonDegeriZK5; %ZK5 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
72   FonksiyonDegeriZK6=fonksiyonDegeriZK6; %ZK6 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
73   FonksiyonDegeriZK7=fonksiyonDegeriZK7; %ZK7 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
74   FonksiyonDegeriZK8=fonksiyonDegeriZK8; %ZK8 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
```

```
75   FonksiyonDegeriEK1=fonksiyonDegeriEK1; %EK1 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
76   FonksiyonDegeriEK2=fonksiyonDegeriEK2; %EK2 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
77   FonksiyonDegeriEK3=fonksiyonDegeriEK3; %EK3 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
78   FonksiyonDegeriEK4=fonksiyonDegeriEK4; %EK4 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
79   FonksiyonDegeriEK5=fonksiyonDegeriEK5; %EK5 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
80   FonksiyonDegeriEK6=fonksiyonDegeriEK6; %EK3 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
81   FonksiyonDegeriEK7=fonksiyonDegeriEK7; %EK3 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
82   FonksiyonDegeriEK8=fonksiyonDegeriEK8; %EK3 alt amaç fonksiyonu değeri, yeni
     bulunan çözümün kabul edilmediği hallerde kullanılmak üzere yedekleniyor.
83   fonksiyonDegeriZK1=0;
84   fonksiyonDegeriZK2=0;
85   fonksiyonDegeriZK3=0;
86   fonksiyonDegeriZK4=0;
87   fonksiyonDegeriZK5=0;
88   fonksiyonDegeriZK6=0;
89   fonksiyonDegeriZK7=0;
90   fonksiyonDegeriZK8=0;
91   fonksiyonDegeriEK1=0;
92   fonksiyonDegeriEK2=0;
93   fonksiyonDegeriEK3=0;
94   fonksiyonDegeriEK4=0;
95   fonksiyonDegeriEK5=0;
96   fonksiyonDegeriEK6=0;
97   fonksiyonDegeriEK7=0;
98   fonksiyonDegeriEK8=0;
99   m=15;
100  n=28;
101  [m n]=size(b);
102   b(1,:)=[1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0];
103  fprintf('\n');
104  %Asagidaki 6 satirlik islemle calismayan durumdaki hemsirelerin vardiyalarina en alt satidaki
     hemsireden baslanarak sifir ataniyor.
105  imin=m-ElliYasUstuGebeVeMazeretliHemsireSayisi+1;
106  for i=imin:m;
107   for j=1:n;
108     b(i,j)=0;
109   End
110  End
111  %Yukaridaki 6 satirlik islemle calismayan durumdaki hemsirelerin vardiyalarina en alt satidaki
     hemsireden baslanarak sifir ataniyor.
112  B=b; %degistirilmis b matrisi B'ye atandi, bu daha sonra eger amac fonksiyon degeri
     karsilastirmasinda yeni cozum kabul edilmezse o zaman B matrisi yani baslangic matrisi b'ye
```

```
      geri atanacak.
113   a=b; %bu ikinci yedek yani b'nin yedegi olan a matrisi ise her alt amac fonksiyonunun
      hesaplanmasinda bazi durumlarda b matrisi uzerinde degisiklik olabilme riski bulundugundan
      alt amac fonksiyon degeri hesaplamalari sonrasi a matrisi bye geri ataniyor, bu a matrisi B ile
      ayni icerikte ancak yine de onlem olarak bu sekilde yapiliyor. Yani bnin yedegi olan B matrisi
      en sonda bir kere daha kullaniliyor, bnin diger yedegi a matrisi ise her alt amac fonksiyonunda
      eger b matrisini degistirecek riskler bulunuyorsa o zaman hesaplama sonunda ya da takip eden
      alt amac fonksiyon degeri hesaplamasinin basinda kullaniliyor, dogru olan her birinin sonunda
      kullanilmasidir.
114   %komple matrisin tamamına rastgele 7gunluk kısıtlara uyan atamalar yapma buradan basliyor
115   AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1=140+70*log2((2*ElliYasUstuGebeVe
      MazeretliHemsireSayisi)+1);
116   if ElliYasUstuGebeVeMazeretliHemsireSayisi<=2;
117   AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1=251;
118   Else
119   end
120   if ElliYasUstuGebeVeMazeretliHemsireSayisi>=7;
121   AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1=251+(ElliYasUstuGebeVeMazeretliH
      emsireSayisi-2)*100;
122   Else
123   end
124   if fonksiyonDegeriAF1>AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1;
125   M=[0,0,2,0,1,0,2; 0,0,1,0,2,0,2; 0,0,1,2,0,0,2; 0,2,0,0,2,0,1; 0,2,0,0,1,0,2; 0,2,0,0,1,2,0;
      0,2,0,2,0,0,1; 0,2,0,1,0,2,0; 0,1,0,2,0,0,2; 0,1,0,2,0,2,0; 0,1,2,0,0,2,0; 2,0,0,2,0,0,1;
      2,0,0,2,0,1,0; 2,0,0,1,0,2,0; 2,0,2,0,0,1,0; 1,0,2,0,0,2,0; 0,0,1,1,2,0,1; 0,0,1,1,1,0,2;
      0,1,0,1,1,0,2; 0,1,1,0,2,0,1; 0,1,1,0,1,0,2; 0,1,1,2,0,0,1; 0,1,1,1,0,2,0; 2,0,0,1,1,0,1;
      2,0,0,1,1,1,0; 2,0,1,1,0,0,1; 1,0,0,1,2,0,1; 1,0,0,1,1,2,0; 1,0,2,0,1,0,1; 1,0,2,0,1,1,0;
      1,0,1,0,2,0,1; 1,0,1,2,0,0,1; 1,0,1,1,0,2,0; 1,2,0,0,1,0,1; 1,2,0,0,1,1,0; 1,2,0,1,0,0,1;
      1,1,0,0,1,2,0; 1,1,0,2,0,0,1; 1,1,0,2,0,1,0; 1,1,0,1,0,2,0; 1,1,2,0,0,1,0; 1,1,1,1,0,0,1;
      0,0,2,0,2,0,1; 0,0,2,0,1,2,0; 0,0,2,2,0,0,1; 0,0,2,2,0,1,0; 0,0,1,0,2,2,0; 0,0,1,2,0,2,0;
      0,0,1,1,1,2,0; 0,2,2,0,0,1,0; 1,0,0,1,1,0,2; 1,0,1,0,1,0,2; 1,1,0,0,1,0,2];
126   h=4;
127   for i=2:m-ElliYasUstuGebeVeMazeretliHemsireSayisi; %sorumlu hemsireye atama yapilan
      1.satir ve calismayan durumdaki hemsirelere atama yapilan son satirlar degistirilmiyor
128     for hh=1:h;
129         xi=round(1+52*rand);    %yani xi sayısı 1'den 53'e kadar rastgele bir sayı olacak
130         for j=(7*(hh-1)+1):(7*hh);
131           xj=j-7*(hh-1); %bu sayede j sayısı 1-7, 8-14, 15-21 ve 22-28 arasında dolaşırken buna
      karşılık xj her zaman 1-7, 1-7, 1-7, 1-7 aralığında dolaşmış olacak.
132         b(i, j)=M(xi,xj); %böylece 15x28'lik b matrisinin ilk olarak 2.satırının 1-7 arası
      sütunlarındaki (yani 2.haftasındaki) elemanlar tek tek M matrisinin döngü başında rastgele
      belirlenen xi. satırından sırasıyla 1,2,3,4,5,6 ve 7.elemanlar ile değiştirilmiş olacak.
133         End
134     End
135   End
136   else %Eğer kritik fonksiyon degeri kritik degerin altindaysa o zaman matrisin tamamına degil
      sadece rastgele tek bir haftasina 7gunluk vardiya gruplarindan biri atanacak
137   %matrisin sadece bir haftasına rastgele 7gunluk kisitlara uyan atamalar yapma buradan
      basliyor. Aşağıda başlangıç matrisini 53 adet 7günlük atama gruplarından rasgele seçtikleriyle
      oluşturan kodlar yer alıyor.
```

```
M=[0,0,2,0,1,0,2; 0,0,1,0,2,0,2; 0,0,1,2,0,0,2; 0,2,0,0,2,0,1; 0,2,0,0,1,0,2; 0,2,0,0,1,2,0;
0,2,0,2,0,0,1; 0,2,0,1,0,2,0; 0,1,0,2,0,0,2; 0,1,0,2,0,2,0; 0,1,2,0,0,2,0; 2,0,0,2,0,0,1;
2,0,0,2,0,1,0; 2,0,0,1,0,2,0; 2,0,2,0,0,1,0; 1,0,2,0,0,2,0; 0,0,1,1,2,0,1; 0,0,1,1,1,0,2;
0,1,0,1,1,0,2; 0,1,1,0,2,0,1; 0,1,1,0,1,0,2; 0,1,1,2,0,0,1; 0,1,1,1,0,2,0; 2,0,0,1,1,0,1;
2,0,0,1,1,1,0; 2,0,1,1,0,0,1; 1,0,0,1,2,0,1; 1,0,0,1,1,2,0; 1,0,2,0,1,0,1; 1,0,2,0,1,1,0;
1,0,1,0,2,0,1; 1,0,1,2,0,0,1; 1,0,1,1,0,2,0; 1,2,0,0,1,0,1; 1,2,0,0,1,1,0; 1,2,0,1,0,0,1;
1,1,0,0,1,2,0; 1,1,0,2,0,0,1; 1,1,0,2,0,1,0; 1,1,0,1,0,2,0; 1,1,2,0,0,1,0; 1,1,1,1,0,0,1;
0,0,2,0,2,0,1; 0,0,2,0,1,2,0; 0,0,2,2,0,0,1; 0,0,2,2,0,1,0; 0,0,1,0,2,2,0; 0,0,1,2,0,2,0;
0,0,1,1,1,2,0; 0,2,2,0,0,1,0; 1,0,0,1,1,0,2; 1,0,1,0,1,0,2; 1,1,0,0,1,0,2];
m=15;
n=28;
i=round(2+(m-ElliYasUstuGebeVeMazeretliHemsireSayisi-2)*rand); %2 ila (15-calismayan
hemsire sayisi) arası rastgele bir sayi belirlendi (b matrisi satiri icin)
hh=round(1+(h-1)*rand); %1-4 arası rastgele bir sayi belirlendi (b ve M matrisleri satir
haftalari icin)
xi=round(1+52*rand);    %1-53 arası rastgele bir sayi belirlendi (M matrisi satiri icin). Not: M
matrisi 53x7 ebatlarinda.
for j=(7*(hh-1)+1):(7*hh); %hhnin 1-4 arası rastgele bir degerinde j sayisi 1-7, 8-14, 15-21 ya
da 22-28 araliginda dolaşacak
xj=j-7*(hh-1); %bu sayede j sayısı 1-7, 8-14, 15-21 ya da 22-28 araliginde dolaşırken buna
karşılık xj her zaman 1-7, 1-7, 1-7, 1-7 aralığında dolaşmış olacak.
b(i, j)=M(xi,xj); %böylece 15x28'lik b matrisinin rastgele secilen bir satisindaki rastgele
secilen bir haftasinda yer alan 7 eleman tek tek M matrisinin döngü başında rastgele belirlenen
rasgele satırından sırasıyla 1,2,3,4,5,6 ve 7.elemanlar ile değiştirilmiş olacak.
End
% Başlangıç matrisinin rastgele tek bir haftasini 53 adet 7günlük atama gruplarından biriyle
degistiren kodlar burada bitiyor.
end
% matrise komple ya da tek bir haftasina rastgele atamalar burada bitiyor ancak su
unutulmamali, anasayacin tek sayi olmasi ya da cift sayi olmasi cok onemli degil, onemli olan
donusumlu olarak her k ust sinir degerine kadar dongulerin bu farkli atama stiliyle donusumlu
olarak yapilmasidir. Ornegin calismayan durumda 5 hemsire icin k degeri 15 oluyor. Bu
durumda 15 deneme boyunca komple matrisin tamamina 7gıunluk haftalari rattgele atama
yapiliyor, bunu takip eden 15 denemede ise matrisin sadece bir haftasina ratgele 7gunluk
vardiya gruplarindan biri ataniyor. Bu devir daim izin verilen maksimum sayidaki denemeye
kadar devam ediyor
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
bbb=b; %rastgele bir haftaya ya da tum haftalara atama sonrasi matris bbb'ye yedekleniyor.
if sayac1+sayac2+sayac3==1;
m=15;
n=28;
c=100;
k=0;
b=round(0+2*rand(m,n));
[m n]=size(b);
bbb=v;
End
m=15;
n=28;
[m n]=size(b);
birlerinsatirfrekansi=0;
ikilerinsatirfrekansi=0;
fonksiyonDegeriZK1=0;
```

```
168  birlerinsatirfrekansi=sum(b==1);
169  frekansZK1=0;
170  ikilerinsatirfrekansi=sum(b==2);
171  for i=1
172  for j=1:n
173  frekansZK1(i,j)= birlerinsatirfrekansi(i,j)*ikilerinsatirfrekansi(i,j);
174  End
175  End
176  fonksiyonDegeriZK1=c*sum(frekansZK1==0);
177  fonksiyonDegeriZK2=0;
178  m=15;
179  n=28;
180  h=4;
181  c=100;
182  [m n]=size(b);
183  sayacZK3=0;
184  farkZK3=0;
185  fonksiyonDegeriZK3=0;
186  toplamZK3=0;
187  for i=1:m
188   for hh=1:h
189     toplamZK3=0;
190     for j=(7*(hh-1)+1):(7*hh)
191     b(i, j)=8*b(i, j);
192     toplamZK3=toplamZK3+b(i, j);
193     End
194   farkZK3=abs(toplamZK3-40);

195   if farkZK3~=0
196      sayacZK3=sayacZK3+1;
197   else
198      sayacZK3=sayacZK3;
199   End
200  End
201  End
202  fonksiyonDegeriZK3=(c*(sayacZK3-
     h*ElliYasUstuGebeVeMazeretliHemsireSayisi))+fonksiyonDegeriZK3;
203  m=15;
204  n=28;
205  c=100;
206  [m n]=size(b);
207  u=0;
208  fonksiyonDegeriZK4=0;
209  sayacZK4=0;
210  toplamZK4=0;
211  for i=1:m
212  for u=1:(n-2)
```

```
213 for j=u:(u+2)
214 toplamZK4=toplamZK4+b(i, j);
215 End
216 if toplamZK4==6
217 sayacZK4=sayacZK4+1;
218 toplamZK4=0;
219 Else
220 toplamZK4=0;
221 End
222 End
223 End
224 toplamZK4=sayacZK4*c;
225 fonksiyonDegeriZK4=toplamZK4;
226 m=15;
227 n=28;
228 c=100;
229 [m n]=size(b);
230 u=0;
231 fonksiyonDegeriZK5=0;
232 sayacZK5=0;
233 toplamZK5=0;
234 b=bbb; %zk4teki islemler sonrasi b matrisinde degisim olmasi ihtimaline karsi onceki
    yedeginden geri cagriliyor. Bu yapilmadiginda ya da bbb matrisi degil de b'nin B ya da a
    yedeginden alindiginda zk5 amac fonksiyon degeri hesaplamasi hata veriyor. Bu satir onemli.
235 for i=1:m
236 for j=1:(n-1)
237 toplamZK5=(b(i,j)-b(i,j+1))*(b(i,j)-b(i,j+1)+1)*(b(i,j)*b(i,j+1));
238 if toplamZK5~=0
239 sayacZK5=sayacZK5+1;
240 Else
241 End
242 End
243 End
244 toplamZK5=sayacZK5*c;
245 fonksiyonDegeriZK5=toplamZK5;
246 fonksiyonDegeriZK6=0;
247 m=15;
248 n=28;
249 c=100;
250 sayacZK7=0;
251 sayac2ZK7=0;
252 islemZK7=0;
253 carpimZK7=0;
254 toplamZK7=0;
255 durumtoplamZK7=0;
256 durumcarpimZK7=0;
257 fonksiyonDegeriZK7=0;
```

```
[m n]=size(b);
  b(1,:)=[1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0];
fprintf('\n');
if sayac1+sayac2+sayac3==1;
  b=v;
  bbb=v;
End
for i=1:m
  toplamZK7=b(i,6)+b(i,7)+b(i,13)+b(i,14)+b(i,20)+b(i,21)+b(i,27)+b(i,28);
  if toplamZK7==0;
    durumtoplamZK7=0;
    Else
    durumtoplamZK7=durumtoplamZK7+1;
   End
carpimZK7=(b(i,1)*b(i,2)**b(i,3)*b(i,4)*b(i,5)*b(i,8)*b(i,9)**b(i,10)*b(i,11)*b(i,12)*b(i,15)*
b(i,16)*b(i,17)*b(i,18)*b(i,19)*b(i,22)*b(i,23)**b(i,24)*b(i,25)*b(i,26)-1);
    if carpimZK7~=0;
    durumcarpimZK7=0;
    Else
    durumcarpimZK7=1;
   End
    islemZK7=toplamZK7+carpimZK7;
    if islemZK7~=0
    sayacZK7=sayacZK7+1;
    Else
    sayacZK7=sayacZK7;
  End
End
if sayacZK7==14;
  sayac2ZK7=0;
  Else
  sayac2ZK7=sayacZK7;
  End
fonksiyonDegeriZK7=sayac2ZK7*100;
fonksiyonDegeriZK8=0;
%Burada esnek kisitlarin amac fonksiyon degeri hesaplanmasi basliyor
%Burada EK1 esnek kisitinin amac fonksiyon degeri hesaplanmasi basliyor
cEK1=10;
fonksiyonDegeriEK1=0;
m=15;
n=28;
b=bbb;
[m n]=size(b);
sayac1EK1=0;
sayac2EK1=0;
sayac3EK1=0;
Ardisik00Arama=0;
```

```
304 Ardisik102Arama=0;
305 Ardisik00ve102Arama=0;
306 h=4;
307 for i=1:m-ElliYasUstuGebeVeMazeretliHemsireSayisi;
308  for hh=1:h;
309    for j=(7*(hh-1)+1):((7*hh)-2);
310    Ardisik00Arama=(b(i,j)+b(i,j+1))*(b(i,j+1)+b(i,j+2));
311    Ardisik102Arama=abs(b(i,j)-b(i,j+1)-1)+abs(b(i,j+2)-b(i,j+1)-2);
312    Ardisik00ve102Arama=Ardisik00Arama*Ardisik102Arama;
313      if Ardisik00ve102Arama==0;
314      sayac1EK1=sayac1EK1+1;
315      else
316      sayac1EK1=sayac1EK1;
317      end
318    end %(jnin endi yani haftanın günleri tek tek taranarak haftalık sayac yani sayac1EK1
       olustu);
319  if sayac1EK1>=1;
320  sayac2EK1=sayac2EK1+1; %(sayac2EK1'in değeri 0'dan farklıysa o haftada en az 1 adet
       ardışık 48 saat izin vardır)
321  sayac1EK1=0;
322  else
323  sayac2EK1=sayac2EK1;
324  sayac1EK1=0;
325  end
326  end %(hh'nin endi yani satırdaki 4 hafta da tarandı ve satir icin (hemsire icin) sayac2EK1
       olusturuldu, sayac2EK1 0'dan 4'e değişebilir, sayac2EK1 0 ise ilgili satırdaki 4 haftada hiç 48
       saat ardışık izin bulunmamaktadır, sayac2EK1 1 ise ilgili satırdaki 4 haftadan birinde en az 1
       kez ardışık 48 saat izin vardır, sayac2EK1 2 ise ilgili satırdaki 4 haftadan 2sinde en az 1'er kez
       ardışık 48 saat izin vardır, vs, olması gereken bu rakamın 4 olması yani her hafta için en az bir
       adet ardışık 48 saat izin olmasıdır);
327  %(şu an i döngüsüne girildi)
328   sayac3EK1=sayac3EK1+(4-sayac2EK1);
329   sayac2EK1=0;
330 end %(i sayacından da çıkıldı yani tüm satırlar taranmış oldu her satır taraması sonrası eger
       herhangi bir satırdaki 4 haftanın herhangi birinde 48 saatlik izin yoksa sayac3EK1 o kadar
       artıyor, yani diyelim ki ilk satırda 4 haftadan 3'ünde en az 1 adet ardışık 48 saat izin yok o
       zaman sayac2EK1=1 oluyor ve sayac3EK1'i 3 artırıyor, diyelim ki ikinci satırda da 4 haftadan
       1'ünde en az 1 adet ardışık 48 saat izin yok o zaman sayac2EK1=3 oluyor ve sayac3EK1'i 1
       artırıyor, boylece 2 satır taraması sonunda sayac3EK1=3+1=4 artmış oluyor, bu şekilde tüm
       satırlar taranıyor ve sayac3EK1 kümülatif değeri oluşuyor. i sayacından da çıkıldığı için tüm
       satırlar için kümülatif değer olan ve 1 satırda 4 haftayı, 15 satırda 60 haftayı tarayarak haftalık
       bazda en az 1 adet ardışık 48 saat izin olmayan hafta sayısı kadar artan sayac3EK1 kullanılarak
       fonksiyonDegeriEK1 oluşturulabilir)
331 fonksiyonDegeriEK1=cEK1*sayac3EK1;
332 %(EK1 esnek kisitinin amac fonksiyon degeri hesaplanmasi burada bitti)
333 fonksiyonDegeriEK2=0;
334 %Burada EK3 esnek kisitinin amac fonksiyon degeri hesaplanmasi basliyor
335 m=15;
336 n=28;
337 b=bbb;
338 ToplaSatirSatir1EK3=0;
```

```
ToplaSatirSatir2EK3=0;
toplam1EK3=0;
toplam2EK3=0;
toplam12EK3=0;
toplam22EK3=0;
ToplaSatirSatir1EK3=0;
ToplaSatirSatir2EK3=0;
SatirToplamlari1EK3=0;
SatirToplamlari2EK3=0;
SatirToplamlari12EK3=0;
SatirToplamlari22EK3=0;
sayacEK3=0;
fonksiyonDegeriEK3=0;
cEK3=10;
for i=1:m;
for j=1:n;
ToplaSatirSatir1EK3=sum(b(i,j)==1);
ToplaSatirSatir2EK3=sum(b(i,j)==2);
toplam1EK3= toplam1EK3+ToplaSatirSatir1EK3;
toplam2EK3= toplam2EK3+ToplaSatirSatir2EK3;
toplam12EK3= toplam12EK3+ToplaSatirSatir1EK3;
toplam22EK3= toplam22EK3+ToplaSatirSatir2EK3;
End
SatirToplamlari1EK3= SatirToplamlari1EK3+toplam1EK3;
SatirToplamlari2EK3= SatirToplamlari2EK3+toplam2EK3;
if SatirToplamlari1EK3>= SatirToplamlari2EK3;
sayacEK3=sayacEK3+0;
Else
sayacEK3= sayacEK3+1;
End
toplam1EK3=0;
toplam2EK3=0;
SatirToplamlari1EK3=0;
SatirToplamlari2EK3=0;
End
SatirToplamlari12EK3= SatirToplamlari1EK3+toplam12EK3;
SatirToplamlari22EK3= SatirToplamlari2EK3+toplam22EK3;
fonksiyonDegeriEK3=sayacEK3*cEK3;
[m n]=size(b);
%EK3 esnek kisitinin amac fonksiyon degeri hesaplanmasi burada sonlandi
%Burada EK5 esnek kisitinin amac fonksiyon degeri hesaplanmasi basliyor
cEK5=10;
fonksiyonDegeriEK5=0;
m=15;
n=28;
b=bbb;
```

```
385 [m n]=size(b);
386 sayac1EK5=0;
387 sayac2EK5=0;
388 sayac3EK5=0;
389 Ardisik000Arama=0;
390 Ardisik1002Arama=0;
391 Ardisik000ve1002Arama=0;
392 h=4;
393 for i=1:m-ElliYasUstuGebeVeMazeretliHemsireSayisi;
394  for hh=1:h;
395    for j=(7*(hh-1)+4):((7*hh));
396    Ardisik000Arama=(b(i,j-3)+b(i,j-2)+b(i,j-1))*(b(i,j-2)+b(i,j-1)+b(i,j));
397    Ardisik1002Arama=abs(b(i,j-3)-b(i,j-2)-b(i,j-1)-1)+abs(b(i,j)-b(i,j-1)-b(i,j-2)-2);
398    Ardisik000ve1002Arama=Ardisik000Arama*Ardisik1002Arama;
399      if Ardisik000ve1002Arama==0;
400      sayac1EK5=sayac1EK5+1;
401      else
402      sayac1EK5=sayac1EK5;
403      end
404    end %(jnin endi yani haftanın günleri tek tek taranarak haftalık sayac yani sayac1EK5
       olustu);
405  if sayac1EK5>=1;
406  sayac2EK5=sayac2EK5+1; %(sayac2EK5'in değeri 0'dan farklıysa o haftada en az 1 adet
       ardışık 72 saat izin vardır)
407  sayac1EK5=0;
408  else
409  sayac2EK5=sayac2EK5;
410  sayac1EK5=0;
411  end
412  end %(hh'nin endi yani satırdaki 4 hafta da tarandı ve satir icin (hemsire icin) sayac2EK5
       olusturuldu, sayac2EK5 0'dan 4'e değişebilir, sayac2EK5 0 ise ilgili satırdaki 4 haftada hiç 72
       saat ardışık izin bulunmamaktadır, sayac2EK5 1 ise ilgili satırdaki 4 haftadan birinde en az 1
       kez ardışık 72 saat izin vardır, sayac2EK5 2 ise ilgili satırdaki 4 haftadan 2sinde en az 1'er kez
       ardışık 72 saat izin vardır, vs, olması gereken bu rakamın 0 olması yani her hafta için 0 adet
       ardışık 72 saat izin olmasıdır);
413  %(şu an i döngüsüne girildi)
414   sayac3EK5=sayac3EK5+sayac2EK5;
415   sayac2EK5=0;
416 end %(i sayacından da çıkıldı yani tüm satırlar taranmış oldu her satır taraması sonrası eger
       herhangi bir satırdaki 4 haftanın herhangi birinde 72 saatlik izin yoksa sayac3EK5 artmıyor,
       varsa kaç haftada en az 1 adet 72 saat izin mevcutsa sayac3EK5 o kadar artıyor, yani diyelim
       ki ilk satırda 4 haftadan 3'ünde en az 1 adet ardışık 72 saat izin var o zaman sayac2EK5=3
       oluyor ve sayac3EK5'i 3 artırıyor, diyelim ki ikinci satırda da 4 haftadan 1'inde en az 1 adet
       ardışık 72 saat izin var o zaman sayac2EK5=1 oluyor ve sayac3EK5'i 1 artırıyor, boylece 2
       satır taraması sonunda sayac3EK5=3+1=4 artmış oluyor, bu şekilde tüm satırlar taranıyor ve
       sayac3EK5 kümülatif değeri oluşuyor. i sayacından da çıkıldığı için tüm satırlar için kümülatif
       değer olan ve 1 satırda 4 haftayı, 15 satırda 60 haftayı tarayarak haftalık bazda en az 1 adet
       ardışık 72 saat izin olan hafta sayısı kadar artan sayac3EK5 kullanılarak fonksiyonDegeriEK5
       oluşturulabilir)
417 fonksiyonDegeriEK5=cEK5*sayac3EK5;
418 %(EK5 esnek kisitinin amac fonksiyon degeri hesaplanmasi burada bitti)
419 % Esnek kisitlarin amac fonksiyon degeri hesaplanmasi burada sonlandi.
```

```
420 fonksiyonDegeriAF1=fonksiyonDegeriZK1+fonksiyonDegeriZK2+fonksiyonDegeriZK3+fonksiyonDegeriZK4+fonksiyonDegeriZK5+fonksiyonDegeriZK6+fonksiyonDegeriZK7+fonksiyonDegeriZK8+fonksiyonDegeriEK1+fonksiyonDegeriEK2+fonksiyonDegeriEK3+fonksiyonDegeriEK4+fonksiyonDegeriEK5+fonksiyonDegeriEK6+fonksiyonDegeriEK7+fonksiyonDegeriEK8;
421  if sayac==1;
422  s=fonksiyonDegeriAF1;
423  End
424 PET=e.^((fonksiyonDegeriAF1-FonksiyonDegeriAF1)/(LogaritmikUstelSicaklikT));
425 if fonksiyonDegeriAF1-FonksiyonDegeriAF1<0;
426  fonksiyonDegeriAF1=fonksiyonDegeriAF1;
427  sayac1=sayac1+1;
428  fprintf('Ilk fark hesaplamasinda daha dusuk bulunan ve KABUL EDILEN amac fonksiyonu degeri: %d\n',fonksiyonDegeriAF1);
429  fprintf('Ilk fark hesaplamasinda daha buyuk bulunan ve kabul edilmeyen amac fonksiyonu degeri: %d\n',FonksiyonDegeriAF1);
430 %*********en iyi matrisleri tek bir matriste topla ve goster*****************
431    if fonksiyonDegeriAF1==0;
432     CozumSayisi=CozumSayisi+1;
433       for i=1:m;
434         for j=1:n;
435         EniyiOnMatris(i+(m+1)*(CozumSayisi-1),j)=EniyiOnMatris(i+(m+1)*(CozumSayisi-1),j)+b(i,j);
436         EniyiOnMatris((m+1)*CozumSayisi,j)=-8;
437         End
438       End
439     fprintf('Bulunan en iyi cizelge-%d\n', CozumSayisi);
440     EniyiCizelgeler=EniyiOnMatris(1:(m+1)*(CozumSayisi),1:n);
441     disp(EniyiCizelgeler);
442    Else
443    End
444 %*********en iyi matrisleri tek bir matriste topla ve goster*****************
445  Else
446  Z=rand;
447  DahaKotuOlanYeniCozumunKabulEdilmeOlasiligi=1/(1+PET);
448  if (DahaKotuOlanYeniCozumunKabulEdilmeOlasiligi)>Z;
449  fprintf('Daha kotu olan yeni cozumun kabul edilme olasiligi: %d\n',DahaKotuOlanYeniCozumunKabulEdilmeOlasiligi);
450  fprintf('Karsilastirilan Z olasilik degeri: %d\n',Z);
451  fonksiyonDegeriAF1=fonksiyonDegeriAF1;
452  sayac2=sayac2+1;
453  fprintf('Olasilik karsilastirmasinda yuksek olasilikli oldugu icin daha kotu olmasina ragmen KABUL EDILEN amac fonksiyonu degeri: %d\n',fonksiyonDegeriAF1);
454   %*********en iyi cozumden sonra daha kotu cozum bulununcaya dek en iyi cizelgeleri tek bir matriste topla ve alt alta goster*****************
455    if fonksiyonDegeriAF1==0;
456     CozumSayisi=CozumSayisi+1;
457       for i=1:m;
458         for j=1:n;
459         EniyiOnMatris(i+(m+1)*(CozumSayisi-1),j)=EniyiOnMatris(i+(m+1)*(CozumSayisi-
```

```
1),j)+b(i,j);
460             EniyiOnMatris((m+1)*CozumSayisi,j)=-8;
461             End
462           End
463         fprintf('Bulunan en iyi cizelge-%d\n', CozumSayisi);
464         EniyiCizelgeler=EniyiOnMatris(1:(m+1)*CozumSayisi,1:n);
465         disp(EniyiCizelgeler);
466       Else
467       End
468     %*********en iyi cozumden sonra daha kotu cozum bulununcaya dek en iyi cizelgeleri tek bir matriste topla ve alt alta goster*****************
469   Else
470     fonksiyonDegeriAF1=FonksiyonDegeriAF1;
471     fonksiyonDegeriZK1=FonksiyonDegeriZK1;
472     fonksiyonDegeriZK2=FonksiyonDegeriZK2;
473     fonksiyonDegeriZK3=FonksiyonDegeriZK3;
474     fonksiyonDegeriZK4=FonksiyonDegeriZK4;
475     fonksiyonDegeriZK5=FonksiyonDegeriZK5;
476     fonksiyonDegeriZK6=FonksiyonDegeriZK6;
477     fonksiyonDegeriZK7=FonksiyonDegeriZK7;
478     fonksiyonDegeriZK8=FonksiyonDegeriZK8;
479     fonksiyonDegeriEK1=FonksiyonDegeriEK1;
480     fonksiyonDegeriEK2=FonksiyonDegeriEK2;
481     fonksiyonDegeriEK3=FonksiyonDegeriEK3;
482     fonksiyonDegeriEK4=FonksiyonDegeriEK4;
483     fonksiyonDegeriEK5=FonksiyonDegeriEK5;
484     fonksiyonDegeriEK6=FonksiyonDegeriEK6;
485     fonksiyonDegeriEK7=FonksiyonDegeriEK7;
486     fonksiyonDegeriEK8=FonksiyonDegeriEK8;
487     fprintf('Daha kotu olan yeni cozumun kabul edilme olasiligi: %d\n',DahaKotuOlanYeniCozumunKabulEdilmeOlasiligi);
488     fprintf('Karsilastirilan Z olasilik degeri: %d\n',Z);
489     fprintf('Olasilik karsilastirmasinda dusuk olasilikli oldugu icin kabul edilmeyen kotu amac fonksiyonu degeri: %d\n',fonksiyonDegeriAF1);
490     sayac3=sayac3+1;
491     b=B;
492   End
493   End
494       if fonksiyonDegeriAF1<=FonksiyonDegeriAF1;
495       fprintf('Amac fonksiyonunun son cozum degeri: %d\n',fonksiyonDegeriAF1);
496       Else
497       fprintf('Amac fonksiyonunun son cozum degeri: %d\n',FonksiyonDegeriAF1);
498       End
499   fprintf('Calisabilir durumdaki toplam hemsire sayisi      : %5d\n',m-ElliYasUstuGebeVeMazeretliHemsireSayisi);
500   fprintf('Izinli durumdaki (50 yas ustu, gebe ya da mazeret izinli) toplam hemsire sayisi: %d\n',(ElliYasUstuGebeVeMazeretliHemsireSayisi));
501   fprintf('Deneme sayisi: %d\n',sayac);
```

```
502 fprintf('Bulunan en iyi cozum sayisi: %d\n', CozumSayisi);
503 fprintf('Daha iyi oldugu icin kabul edilen cozum sayisi: %d\n',sayac1);
504 fprintf('Daha kotu olup kabul edilen cozum sayisi: %d\n',sayac2);
505 fprintf('Daha kotu olup reddedilen cozum sayisi: %d\n',sayac3);
506 fprintf('Ilk sicaklik: %d\n',T0);
507 fprintf('Son sicaklik: %d\n',LogaritmikUstelSicaklikT);
508 fprintf('ZK1"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK1);
509 fprintf('ZK2"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK2);
510 fprintf('ZK3"un zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK3);
511 fprintf('ZK4"un zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK4);
512 fprintf('ZK5"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK5);
513 fprintf('ZK6"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK6);
514 fprintf('ZK7"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK7);
515 fprintf('ZK8"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK8);
516 fprintf('EK1"in esnek kisit amac fonksiyon degeri       : %5d\n',fonksiyonDegeriEK1);
517 fprintf('EK2"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK2);
518 fprintf('EK3"un esnek kisit amac fonksiyon degeri       : %5d\n',fonksiyonDegeriEK3);
519 fprintf('EK4"un esnek kisit amac fonksiyon degeri       : %5d\n',fonksiyonDegeriEK4);
520 fprintf('EK5"in esnek kisit amac fonksiyon degeri       : %5d\n',fonksiyonDegeriEK5);
521 fprintf('EK6"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK6);
522 fprintf('EK7"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK7);
523 fprintf('EK8"in esnek kisit amac fonksiyon degeri       : %5d\n',fonksiyonDegeriEK8);
524 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon ilk degeri: %d\n',s);
525 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon sondan bir onceki degeri:
    %d\n',FonksiyonDegeriAF1);
526 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon son degeri:
    %d\n',fonksiyonDegeriAF1);
527 fprintf('Atama yontemi degisimi icin kritik toplam amac fonksiyon degeri:
    %d\n',AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1);
528 fprintf('\n');
529 Else
530 End
531 End
532 fprintf('\n');
533 Break
534 End
535 End
536 fprintf('Rastgele atama ile olusturulan baslangic cizelgesi\n');
537 disp(v);
538 fprintf('\n');
539    %*********en iyi cozumden sonra daha kotu cozum bulununcaya dek kmaks dongusu
    boyunca en iyi cizelgeleri tek bir matriste topla ve alt alta goster*****************
540       fprintf('\n');
541       fprintf('Bulunan en iyi cozum sayisi: %d\n', CozumSayisi);
542       disp('Tavlama Algoritmasi ile elde edilen en iyi cizelgeler');
543       EniyiCizelgeler=EniyiOnMatris(1:(m+1)*CozumSayisi,1:n);
544       disp(EniyiCizelgeler);
545       fprintf('\n');
```

```
546     %*********en iyi cozumden sonra daha kotu cozum bulununcaya dek en iyi cizelgeleri tek bir matriste topla ve alt alta goster**********
547 fprintf('\n');
548 fprintf('Calisabilir durumdaki toplam hemsire sayisi     : %5d\n',m-ElliYasUstuGebeVeMazeretliHemsireSayisi);
549 fprintf('Izinli durumdaki (50 yas ustu, gebe ya da mazeret izinli) toplam hemsire sayisi: %d\n',(ElliYasUstuGebeVeMazeretliHemsireSayisi));
550 fprintf('ZK1"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK1);
551 fprintf('ZK2"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK2);
552 fprintf('ZK3"un zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK3);
553 fprintf('ZK4"un zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK4);
554 fprintf('ZK5"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK5);
555 fprintf('ZK6"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK6);
556 fprintf('ZK7"nin zorunlu kisit amac fonksiyon degeri    : %5d\n',fonksiyonDegeriZK7);
557 fprintf('ZK8"in zorunlu kisit amac fonksiyon degeri     : %5d\n',fonksiyonDegeriZK8);
558 fprintf('EK1"in esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK1);
559 fprintf('EK2"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK2);
560 fprintf('EK3"un esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK3);
561 fprintf('EK4"un esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK4);
562 fprintf('EK5"in esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK5);
563 fprintf('EK6"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK6);
564 fprintf('EK7"nin esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK7);
565 fprintf('EK8"in esnek kisit amac fonksiyon degeri      : %5d\n',fonksiyonDegeriEK8);
566 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon ilk degeri: %d\n',s);
567 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon sondan bir onceki degeri: %d\n',FonksiyonDegeriAF1);
568 fprintf('Zorunlu ve esnek kisitlarin toplam amac fonksiyon son degeri: %d\n',fonksiyonDegeriAF1);
569 fprintf('Atama yontemi degisimi icin kritik toplam amac fonksiyon degeri: %d\n',AtamaYontemiDegisimiicinKritikfonksiyonDegeriAF1);
570 fprintf('Deneme sayisi: %d\n',sayac1+sayac2+sayac3);
571 fprintf('Bulunan en iyi cozum sayisi: %d\n', CozumSayisi);
572 fprintf('Daha iyi oldugu icin kabul edilen cozum sayisi: %d\n',sayac1);
573 fprintf('Daha kotu olup kabul edilen cozum sayisi: %d\n',sayac2);
574 fprintf('Daha kotu olup reddedilen cozum sayisi: %d\n',sayac3);
575 fprintf('Sicaklik dusurme kontrol sabiti: %d\n',d);
576 fprintf('Ilk sicaklik: %d\n',T0);
577 fprintf('Son sicaklik: %d\n',LogaritmikUstelSicaklikT);
578 fprintf('\n');
579 toc;
```

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Bilgen AYAN KOÇ
Uyruğu	**:**	T.C.
Doğum Yeri ve Tarihi	**:**	MUT/19.04.1990
Telefon	**:**	05399292983
Faks	**:**	-
e-mail	**:**	bilgenayan@hotmail.com

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Mut Anadolu Lisesi	2008
Üniversite	:	Selçuk Üniversitesi	2013

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2013-2014	Kayahan Hidrolik A.Ş.	Üretim Planlama Mühendisi
2014-2016	Anadolu Birlik Holding (TORKU)	Planlama ve Ana Veri Kontrol Mühendisi

YABANCI DİLLER

İngilizce

YAYINLAR

Artificial Neural Network Models for Predicting the Energy Consumption of the Process of Crystallization Syrup in Konya Sugar Factory

The Solution of Nurse Scheduling Problem with Simulated Annealing Algorithm