T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KARINCA KOLONİSİ ALGORİTMASI İLE İNŞAAT PROJESİ PROGRAMININ OPTİMİZE EDİLMESİ

YÜKSEK LİSANS TEZİ

Cihan YENİGÜN

Enstitü Anabilim Dalı	**:**	**İNŞAAT MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Sedat Semih ÇAĞLAYAN**

Ekim 2023

T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KARINCA KOLONİSİ ALGORİTMASI İLE İNŞAAT PROJESİ PROGRAMININ OPTİMİZE EDİLMESİ

YÜKSEK LİSANS TEZİ

Cihan YENİGÜN

Enstitü Anabilim Dalı : İNŞAAT MÜHENDİSLİĞİ

Bu tez 10/10/2023 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

JÜRİ	BAŞARI DURUMU
Jüri Başkanı: Dr. Öğr. Üyesi Sedat Semih ÇAĞLAYAN	BAŞARILI
Üye: Dr. Öğr. Üyesi Abdulkadir ÖZDEN	BAŞARILI
Üye: Dr. Öğr. Üyesi Esra DOBRUCALI	BAŞARILI

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim

Cihan YENİGÜN

10/10/2023

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğr. Üys. Sedat Semih ÇAĞLAYAN'na teşekkürlerimi sunarım.

İÇİNDEKİLER

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KARINCA KOLONİSİ ALGORİTMASI İLE İNŞAAT PROJESİ PROGRAMININ OPTİMİZE EDİLMESİ

ÖZET

Günümüzde, inşaat sektörü, büyük projelerin karmaşıklığı, bütçe sınırları ve zaman kısıtlamaları gibi bir dizi zorluğa karşı karşıyadır. Bu zorlukların üstesinden gelmek ve projelerin başarılı bir şekilde tamamlanmasını sağlamak, inşaat sektörü için kritik bir ihtiyaçtır. Bu tez, Karınca Kolonisi Algoritması (KKA) ile inşaat projelerinin programlarının optimize edilmesinin nasıl mümkün olabileceğini incelemeyi amaçlamaktadır.Karınca Kolonisi Algoritması, doğal bir fenomen olan karıncaların yem arama davranışını temel alarak geliştirilmiş bir optimizasyon algoritmasıdır. KKA, karıncaların yolları üzerinde feromonlar bırakarak birbirleriyle iletişim kurmaları ve en kısa yolu bulmaları şeklindeki davranışlarını taklit eder. İnşaat projeleri, kaynakların etkin kullanımını, zaman yönetimini ve karmaşık süreçleri içerdiği için, bu projelerin optimize edilmesi zor bir görevdir.Bu tez, KKA'nın inşaat projeleri üzerinde nasıl uygulanabileceğini ve projelerin zaman çizelgelerinin optimize edilmesine nasıl yardımcı olabileceğini araştırmaktadır. KKA, inşaat projelerinin özel gereksinimlerini göz önünde bulundurarak uyarlanabilir ve projenin süreçlerini optimize etmek, kaynakların verimli bir şekilde kullanılmasını sağlamak ve proje tamamlama süresini minimize etmek amacıyla kullanılabilir.Araştırma, KKA'nın inşaat projelerindeki uygulanabilirliğini ve avantajlarını vurgulamaktadır. KKA ile optimize edilen bir inşaat projesinin, projenin bitiş tarihini sıkı bir şekilde takip ederken aynı zamanda kaynakları daha verimli bir şekilde kullanma yeteneğine sahip olduğu bulunmuştur. Bu, projelerin zamanında tamamlanmasını ve maliyetlerin minimize edilmesini sağlar.Ancak, KKA'nın başarılı bir şekilde uygulanabilmesi için uygun modelleme ve parametre ayarlamaları gerekmektedir. Ayrıca, inşaat projelerinin özel gereksinimleri ve kısıtlamaları göz önünde bulundurulmalıdır. Proje yöneticileri ve mühendisler, KKA'yı etkili bir şekilde kullanmak için iyi bir anlayış geliştirmelidir.Sonuç olarak, "Karınca Kolonisi Algoritması İle İnşaat Projesi Programının Optimizasyonu" başlıklı bu tez, inşaat sektöründe proje yönetimi ve zaman çizelgesinin geliştirilmesine yönelik önemli bir katkı sunmaktadır. KKA, inşaat projelerinin daha etkili bir şekilde yönetilmesine ve maliyetlerin minimize edilmesine katkıda bulunabilir. Bu çalışma, inşaat sektöründeki karar vericilere ve proje yöneticilerine, projelerini optimize etme konusunda yeni bir perspektif sunmaktadır ve sektördeki verimliliği artırma potansiyeline sahiptir. İnşaat projelerinin karmaşıklığını yönetmek ve kaynakları en iyi şekilde kullanmak için KKA gibi yenilikçi yaklaşımların önemi büyüktür.

Anahtar Kelimeler: İnşaat projesi, proje hızlandırma problemi, Karınca kolonisi algoritması

OPTIMIZING THE CONSTRUCTION PROJECT PROGRAM WITH THE ANT COLONY ALGORITHM

ABSTRACT

In today's construction industry, projects are becoming increasingly complex, facing challenges such as budget constraints and tight timelines. Overcoming these challenges and ensuring the successful completion of construction projects is of paramount importance. This thesis aims to explore how the Ant Colony Algorithm (ACA) can be applied to optimize construction project schedules.The Ant Colony Algorithm is an optimization technique inspired by the foraging behavior of ants, where they communicate by leaving pheromones on their paths to find the shortest route to food sources. Construction projects involve resource management, time constraints, and intricate processes, making their optimization a formidable task.This thesis delves into how ACA can be adapted and applied to meet the specific requirements of construction projects. By mimicking the natural foraging behavior of ants, ACA can optimize project processes, efficiently utilize resources, and minimize project completion times.The research highlights the applicability and advantages of ACA in construction projects. An ACA-optimized construction project can vigilantly track project deadlines while simultaneously optimizing resource allocation. This leads to on-time project completion and cost minimization.Nevertheless, successful implementation of ACA requires careful modeling and parameter adjustments, along with consideration of the unique requirements and constraints of construction projects. Project managers and engineers should develop a solid understanding of ACA to use it effectively.In conclusion, "Optimizing Construction Project Scheduling with Ant Colony Algorithm" offers a significant contribution to project management and scheduling in the construction sector. ACA has the potential to enhance the effective management of construction projects and cost minimization. This study provides a fresh perspective to decision-makers and project managers in the construction sector, offering the potential to increase efficiency. Innovative approaches like ACA are essential to manage the complexity of construction projects and optimize resource utilization.

Keywords: Construction project, project crashing problem, Genetic algorithm

BÖLÜM 1. GİRİŞ

Proje yönetimi, başarılı bir projenin geliştirilmesi, ölçülmesi ve kontrol edilmesi için gerekli olan güçlü bir alandır. Bu alan, yönetim biliminin, organizasyon teorisinin, operasyon yönetiminin ve sosyal psikoloji gibi farklı disiplinleri içermektedir. Bu alanlar, proje yönetiminin temelini oluşturmakta ve proje başarısının önemli bir ölçütü olarak kabul edilmektedir (Chen, 2015). Proje yönetiminde, müşteri ihtiyaçlarının karşılanması, proje etkinliğinin ölçülmesi ve proje çerçevesinin etkin bir şekilde yönetilmesi gibi çeşitli görevler yer almaktadır. Proje yöneticileri, proje ekibinin performansını ölçmek, projenin başarısını değerlendirmek ve projeleri hızla gerçekleştirmek için gerekli olan çalışmaları gerçekleştirmekle yükümlüdür. Proje yönetimine uygulanan geniş bilgi ve uzmanlık yelpazesine rağmen, birçok projenin gecikmeler ve maliyet aşımları nedeniyle genellikle amaçlandığı gibi performans gösteremediği olabilmektedir (Huo ve ark., 2018). Bu problemler, projenin uygulanabilirliği açısından önemli bir risk oluşturabilmektedir (Callegari ve ark., 2018). Proje yönetimi başarısızlıklarının nedenlerini daha iyi anlamak için, genellikle birbirleriyle bağlantılı olduklarından dolayı gecikmeler ve maliyet aşımlarını birlikte ele almak önemlidir (Heravi ve Mohammadian, 2021). Kötü programlama, proje yönetimi başarısızlıklarının en yaygın sebeplerinden biridir ve genellikle birincil problem olmaktadır (Durdyev, 2020). Proje yönetiminde, proje başarısızlığına sebep olabilecek birçok farklı faktör de bulunmaktadır. Başarılı proje sonuçlarına ulaşabilmek için, proje yönetiminin zamanlama, kaynak tahsisi ve risk yönetimi gibi birçok yönünü göz önünde bulundurmak önemlidir.

Proje çizelgeleme problemi, 1960'lardan bu yana proje yönetimindeki araştırmaların en önemli konusu olmuştur. Bu problem, projelerin ne zaman ve ne şekilde tamamlanacağını belirleyen bir karar problemidir. Proje çizelgeleme ile proje başarısının arttırılması amaçlanmaktadır. Bu amaçla, araştırmacılar maliyet minimizasyonu, süre minimizasyonu ve kaynak optimizasyonu gibi çeşitli yaklaşımlar

geliştirmişlerdir. Bu yaklaşımlar, proje çizelgeleme problemine uygun çözümlerin bulunmasını sağlamak için kullanılmaktadır. Özellikle maliyet minimizasyonu, projenin ne kadar maliyetli olacağının tahmin edilmesine ve daha sonra gerçekleşmesine yardımcı olmak için kullanılmaktadır. Ayrıca, süre minimizasyonu, projenin belirli bir süre içinde tamamlanmasının sağlanması için kullanılmaktadır. Kaynak optimizasyonu, projenin başarısını arttırmak için gereken kaynakların doğru şekilde dağıtılmasını sağlamak için kullanılmaktadır. Bütün bu yaklaşımlar, proje çizelgeleme problemini etkili bir şekilde çözmek için kullanılmaktadır (Rahman ve ark., 2020).

Büyük ölçekli projeleri etkin bir şekilde yönetmek için kuruluşlar, kaynakların en uygun şekilde tahsis edilmesini ve projenin verilen zaman dilimi içinde tamamlanmasını sağlamalıdır. Bu amaçla, araştırmacılar çizelgeleme sürecinin verimliliğini artırmak için çok sayıda sezgisel yöntem geliştirmiştir (Vega-Velázquez ve ark., 2018). Örneğin, kritik yol yöntemi ve kaynak kısıtlı proje çizelgeleme problemi, proje maliyetlerini ve süresini en aza indiren verimli çizelgeler oluşturmak için kullanılmıştır (Pellerin ve ark., 2020). Ayrıca, proje çizelgeleme problemini çözmek için yapay zeka ve makine öğrenimi tekniklerinin kullanılması önerilmiştir (Zareei, 2018). Bu, kuruluşların proje başarısını en üst düzeye çıkarabilecek daha doğru ve verimli projeleri geliştirmelerini sağlamıştır.

Proje hızlandırma, projenin toplam maliyetini en aza indirmek için maliyet ve süre değiş tokuşlarını kullanan bir proje hızlandırma tekniğidir. Bu teknik, kritik yolu kısaltarak ve kritik yoldaki belirli aktivitelere daha fazla kaynak tahsis ederek çalışmaktadır. Bunu yaparak proje süresini kısaltmak mümkündür, bu da proje gecikmeleriyle ilişkili cezalarda tasarruf sağlayabilmektedir. Bu tasarruf, aktivite maliyetlerindeki artıştan daha ağır bastığı sürece faydalı olabilmekte ve böylece projenin toplam maliyetini azaltmaktadır. Projenin hızlandırılması herkese uyan tek bir çözüm değildir ve dikkatli bir analiz ve müzakere gerektirmektedir. Proje yöneticilerinin tekniği uygulamadan önce kaynak maliyetini, proje gecikmesinin cezasını ve değişikliklerin proje zaman çizelgesi üzerindeki etkisini göz önünde bulundurmaları gerekmektedir. Bazı durumlarda, projenin hızlandırılması faydalı olabilmektedir, ancak diğerlerinde projenin toplam maliyeti üzerinde olumsuz etkisi de olabilmektedir.

Proje hızlandırma, geleneksel yaklaşım ile gerçekleştirilebilmekte ve zaman alıcı ve hataya açık bir süreç olabilmektedir. Yanlış veya yanıltıcı sonuçlara da yol

açabilmektedir. Bu problemi ele almak için araştırmacılar, karmaşık problemler için en uygun çözümleri hızlı bir şekilde belirleyebilen karınca kolonisi algoritmalar gibi meta-sezgisel algoritmalar geliştirmişlerdir. Bu algoritmalar, milyonlarca olası çözüm arasından en iyi çözümü, geleneksel yaklaşımla bunu yapmak için gereken sürenin çok altında bir sürede bulabilmektedir.

Proje yönetimi, zamanın çok önemli olduğu ve kaynakların sınırlı olduğu günümüz dünyasında güçlü bir araçtır. Kaynaklar doğru yönetilmezse istenmeyen sonuçlar ortaya çıkabilmektedir. Bu kaynakların en iyi şekilde kullanılabilmesi için projenin başından sonuna kadar bir planın uygulanması gerekmektedir. Bu plan proje yönetimi olarak bilinmektedir.

Proje yönetimi, istenen hedeflere en verimli ve etkili şekilde ulaşılmasını sağlamak için bir projede yer alan süreçlerin planlanmasını ve kontrol edilmesini içermektedir (Yıldız, 2001). Bu, proje aktivitelerinin planlanması, programlanması (çizelgelenmesi) ve kontrolünün yanı sıra finansal ve insan kaynakları aktivitelerinin organize edilmesi ve yürütülmesini de kapsamaktadır (Sivri, 2001). Proje yönetiminin temel amacı, bir projeyi planlanan süre içerisinde en az maliyetle ve en yüksek kalite seviyesinde tamamlamaktır (Kutlu, 2001). Kuruluşlar, proje yönetimini kullanarak projelerinin problemsiz ve verimli bir şekilde yürütülmesini sağlayabilmekte, böylece zamandan ve kaynaklardan tasarruf edebilmektedirler.

Yukarıda belirtilen proje yönetimi tanımlarından da anlaşılacağı üzere, projelerin planlanması ve kontrolü, projenin başarısı için kilit bir faktördür. Projelerin planlanması ve kontrolü için en yaygın kullanılan yöntemlerden biri ağ analizidir. Bu yöntem, bir dizi aktivite ve olayın grafiksel gösterimi olan ağları analiz ederek karmaşık projelerin kaynaklarının daha etkin bir şekilde yönetilmesine yardımcı olmaktadır. Ağlar, planlama amacıyla aktiviteler ve olaylar arasındaki ilişkileri göstermenin yanı sıra, projenin hedeflerine ulaşmak için tamamlanması gereken aktivite ve olayların bir çizelgesini göstermektedir. Program Değerlendirme ve Gözden Geçirme Tekniği (PERT) ve Kritik Yol Yöntemi (CPM) büyük ve karmaşık projelerin planlanması, tasarımı ve kontrolüne yardımcı olmak üzere geliştirilmiştir. Ağlar genellikle büyük ve ayrıntılı olduğundan, bu modelleri uygulayarak proje tamamlanma süresini hesaplamak için bir bilgisayar yazılımına ihtiyaç duyulmaktadır (Duncan, 1996). Bilgisayar yazılımlarının kullanımıyla, proje yöneticileri kritik yolları ve bunlarla ilişkili

aktiviteleri ve ayrıca proje süresince ortaya çıkabilecek potansiyel riskleri belirleyebilmektedirler. Buna ek olarak, bilgisayar yazılımları kaynak tahsisi ve programlamadaki değişikliklerin etkisini incelemek ve bu değişikliklerden kaynaklanabilecek problemleri belirlemek için de kullanılabilmektedir. Bilgisayar yazılımlarının kullanımı sayesinde proje yöneticileri bilinçli kararlar alabilmekte ve projelerini başarılı bir şekilde yönetebilmektedir.

Bu nedenle günümüzde karşılaştığımız karmaşık ve zor koşullara hızlı ve kolay çözümler için yeni yöntemler bulmak gerekli hale gelmiştir. Bu yöntemler genel olarak analitik veya sezgisel modeller olarak sınıflandırılabilmektedir ve bu koşulların ortaya çıkardığı zorlukların üstesinden gelmek için kullanılmaktadır. Ancak, yalnızca analitik modellere güvenildiğinde her zaman başarılı sonuçlar elde edilememektedir, çünkü bu modeller karmaşık veya zor problemleri çözmek için uygun değildir. Öte yandan sezgisel veya evrimsel modeller, probleme bağlı yapıları nedeniyle bu tür problemleri çözmek için daha donanımlıdır. Son yıllarda, zor optimizasyon tekniklerinin aksine kolay hesaplama ve evrimsel algoritmaların kullanımı giderek daha popüler hale gelmiştir (Leu ve Yang, 1999). Evrimsel algoritmalar özellikle arama ve optimizasyon problemleri söz konusu olduğunda etkilidir. Çalışmalar, evrimsel algoritmaları kullanmanın bu problemlere çeşitli etkili çözümler bulmanın oldukça verimli bir yolu olduğunu ortaya koymuştur (Deb, 2001). Bunun nedeni, evrimsel algoritmaların farklı senaryolara uyarlanabilmesi ve meydana gelebilecek herhangi bir değişikliği hesaba katmak için hızlı bir şekilde ayarlanabilmesidir. Ayrıca, çok çeşitli çözümler üretebildikleri için kullanıcıya aralarından seçim yapabileceği daha geniş seçenekler sunmaktadırlar.

Bu bölümde, araştırmanın ayrıntılarına girmeden önce incelenmekte olan Program Değerlendirme ve Gözden Geçirme Tekniği (PERT) ve Kritik Yol Yöntemi (CPM) tekniklerine kısa bir genel bakış sunulmaktadır.

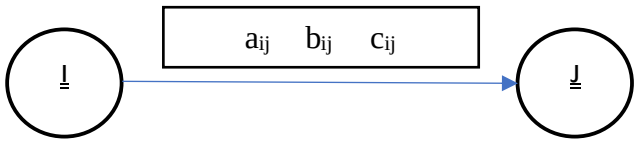
1.1. Kritik Yol Yöntemi

CPM yöntemi, sabit ve tanımlanmış işlem sürelerinin kabul edilmesi fikrine dayanmaktadır. Ayrıca bu yöntem, farklılıklar göz ardı edildiğinde, proje tamamlanma süresi, kritik aktiviteler ve ağın kritik yolunun yanı sıra ağın eksik unsurlarının tanımlanmasını da gerektirmektedir (Özdemir, 2006; Özturk, 1984). Kritik yol, en uzun

toplam aktivite süresine sahip olan ve proje ağı üzerinde başlangıç ve bitiş noktası arasında projenin en kısa sürede tamamlanması için gerekli olan yoldur. Kritik yolun önemi, uzunluğunun projenin tamamlanma süresini belirlemesidir. Bu, kritik yolu oluşturan aktivitelerdeki herhangi bir gecikmenin projenin genel olarak tamamlanmasında bir gecikmeye neden olacağı anlamına gelmektedir. Bu yolu oluşturan aktiviteler de kritik aktiviteler olarak adlandırılmaktadır. Ayrıca kritik yol, tüm projenin temelini oluşturduğu için CPM yönteminin en önemli unsurudur. Bu olmadan projenin zamanında tamamlanması mümkün değildir. Bu nedenle, projenin mümkün olduğunca hızlı ve verimli bir şekilde tamamlanmasını sağlamak için kritik yolun belirlenmesi ve etkin bir şekilde yönetilmesi kritik önem taşımaktadır.

1.2. Program Değerlendirme ve Gözden Geçirme Tekniği (PERT)

PERT yöntemi, operasyon sürelerini ve projenin tamamlanma süresini tahmin etme dayanmaktadır (Özdemir, 2006; Trietsch ve ark., 2012). Araştırma ve geliştirme projeleri gibi belirsiz projelerde kullanılmak üzere geliştirilmiştir, bu da onu Kritik Yol Yöntemi'nden (CPM) farklı kılmaktadır. Hem PERT hem de CPM aynı matematiksel tekniklerle hesaplanmasına rağmen, ikisi arasındaki tek fark PERT'in Şekil 1.1'de görüldüğü gibi her aktivite için üç tahmin yapmasıdır. Tahminler şu şekildedir: a, her şey plana uygun giderse aktivitenin minimum zamanını göstermekte; b, her şey ters giderse aktivitenin maksimum zamanını göstermekte; ve m, koşullar normal kalırsa aktivitenin yaklaşık zamanını göstermektedir. Bu üç zaman tahmini, projenin tamamlanması için geçecek süreyi daha iyi gösterdiğinden, projenin daha iyi anlaşılmasına yardımcı olmaktadır. Ayrıca, projede ortaya çıkabilecek potansiyel riskleri belirlemek için de kullanılabilmekte ve proje yöneticisinin bunları önlemek veya hafifletmek için gerekli adımları atmasına olanak tanımaktadır.



Şekil 1.1:PERT Ağları için verilen zaman dilimleri.

Her bir aktivitenin tahmini süre içinde gerçekleşme olasılığı çeşitli matematiksel ve istatistiksel yöntemler kullanılarak hesaplanmaktadır. Her bir aktivitenin beklenen süresini belirlemek için Eq 1'deki formül kullanılarak ortalama bir süre hesaplanmaktadır. Bu formül, aktivite için tahmin edilen sürenin yanı sıra kaynakların

mevcudiyeti ve personelin mevcudiyeti gibi aktivitenin süresini etkileyebilecek potansiyel dış faktörleri de dikkate almaktadır. Ortalama süre daha sonra aktivitenin beklenen süresine ilişkin bir olasılık dağılımı oluşturmak ve bu dağılım aktivitenin beklenen süresine ilişkin kararları bilgilendirmek için kullanılabilmektedir. Ayrıca, bu hesaplamaların sonuçları, söz konusu aktivite için kaynak tahsisi ve personel atamasına ilişkin kararlar için kullanılabilmektedir. Nihayetinde, bu hesaplamaların sonuçları verimliliği en üst düzeye çıkarmak ve belirli bir aktiviteyi tamamlamak için harcanan süreyi azaltmak için kullanılabilmektedir.

$$T= (a+4m+b)/6 \quad (1.1)$$

Herhangi bir aktivitenin süresi tek bir zamana indirgendiğinde, her bir düğümün Ei ve Li değerleri, ilk düğümden başlayıp son düğümle biten doğrusal bir şekilde hesaplanabilmektedir. Aynı E ve L değerlerini paylaşan düğümler, projenin tamamlanması için kritik olan aktiviteleri içeren kritik yolu göstermektedir. Bu yolun toplam uzunluğu projenin tamamlanma süresini ifade etmektedir (Trietsch ve ark., 2012). Ayrıca, kritik yol yöntemi (CPM), projenin tamamlanma zaman çizelgesini karşılamak için gerçekleştirilmesi gereken aktivitelerin görsel bir gösterimini sağlamaktadır. Bu, aktiviteler ve süreleri arasındaki bağımlılıkları dikkate aldığı için projenin daha kapsamlı bir görünümünü sağlamaktadır. Ayrıca, herhangi bir beklenmedik durum veya gecikmeyi planlamak için kullanılabilecek projenin tamamlanma süresine ilişkin bir tahmin de vermektedir. Proje yöneticileri CPM'i kullanarak projenin tamamlanmasını doğru bir şekilde planlayabilmekte ve tüm aktivitelerin zamanında tamamlandığından emin olabilmektedirler.

BÖLÜM 2. GENEL BİLGİLER

2.1. Proje Hızlandırması ve TCTP Problemi

2.1.1. TCTP problemi

İnşaat yönetimi açısından bakıldığında hem müşteri hem de yüklenici, bir inşaat projesinde zaman, maliyet ve diğer operasyonel kaynaklar gibi farklı parametreleri karşılayacak en ekonomik çizelgelemeyi aramaktadır. Bir inşaat projesindeki her aktivitenin bir normal süresi ve bir de zorunlu süresi bulunmaktadır; bir aktiviteyi zorunlu süresinde tamamlamak daha fazla doğrudan maliyet ve kaynak gerektirmektedir. Ancak bu aynı zamanda projenin toplam süresinde ve buna bağlı dolaylı maliyetlerde (şantiye, denetçi, merkez ofis giderleri vb.) Bir projenin zaman ve maliyeti arasındaki bu dengeleme literatürde zaman maliyet ödünleşim problemi (TCTP) olarak bilinmektedir ve bu problemin çözümü bir optimizasyon yönteminin uygulanmasını gerektirmektedir. Bir aktivitenin durdurulup durdurulmayacağı kararının karmaşık bir karar olduğunu ve bu karara dikkatli bir şekilde yaklaşılması gerektiğini unutmamak önemlidir. Bir aktivitenin durdurulması gerekebilmektedir ancak bu durum diğer aktiviteler üzerinde dalgalanma etkisine sebep olabilmektedir ve karar verilirken dikkate alınması gereklidir. Sonuç olarak, en iyi yaklaşımı belirlemek için zaman ve maliyet arasındaki dengenin dikkatli bir şekilde analiz edilmesi gerekmektedir.

TCTP'leri çözmek için kullanılan ilk optimizasyon yöntemleri doğrusal programlama, tamsayılı programlama ve dinamik programlama yöntemlerini içeren matematiksel çözümlere dayanmaktadır (Meyer ve Shaffer, 1965). Bu yöntemler nispeten küçük test senaryoları üzerinde uygulanmıştır, ancak değerlendirilen problemin tüm değişkenleri arasında sürekli ilişkiler olduğunu varsaymaktadırlar. Bu önemli bir kısıttır, çünkü uygulamada bir aktivitenin yürütülmesi zaman, maliyet, işçilik vb. gibi operasyonel kaynaklara ihtiyaç duymakta ve bunlarda da birkaç seçenek bulunabilmektedir. Bundan dolayı, matematiksel teoriye dayalı optimizasyon yöntemleri kesikli zaman-maliyet ilişkilerine sahip problemler için uygun değildir (Feng ve ark., 1997). Ayrıca, tamsayılı

programlama ve dinamik programlama, daha karmaşık proje ağlarını çözmek veya birçok farklı aktivite içeren projeleri çözmek için önemli hesaplama çabaları gerektirmekte ve bu da etkinliklerini daha da sınırlamaktadır. Bu durum, ayrık değişkenli problemlerin üstesinden gelmek için daha uygun olan karınca kolonisi algoritmalar, karınca kolonisi optimizasyonu ve benzetilmiş tavlama gibi alternatif optimizasyon yaklaşımlarının geliştirilmesine yol açmıştır. Bu algoritmalar, çok daha geniş bir olasılık yelpazesinde arama yaparak çok daha kısa sürede optimuma yakın bir çözüm elde etmek için kullanılabilmektedir.

TCTP'lerin çözümü için kullanılan diğer yaygın yöntemler, matematik tabanlı yöntemlerde kullanılan kesin algoritmaların aksine basit kurallar uygulayan sezgisel algoritmalara dayanmaktadır (Siemens, 1971). Bu, karmaşık problemleri daha az çabayla çözmek için avantaj sağlamaktadır. Bu algoritmalar, en uygun çözümü bulmayı garanti etmese de, karmaşık problemlere global çözümler bulmaya çalışmaktadırlar. Ayrıca farklı problem uzaylarını keşfetmek ve en az hesaplama çabasıyla en iyi çözümü bulmak için de kullanılabilmektedirler. Buna ek olarak, metasezgisel algoritmalar genellikle eldeki özel probleme göre uyarlanmakta ve bu da onları TCTP'lerin zorluklarının üstesinden gelmek için güçlü bir araç haline getirmektedir.

Son yıllarda, karınca kolonisi algoritmalar (Zheng ve Ng, 2005), benzetilmiş tavlama (Sönmez ve Bettemir, 2012), parçacık sürü optimizasyonu (Sönmez ve Bettemir, 2012),), karınca kolonisi optimizasyonu (Afshar ve ark., 2009) ve kurbağa sıçrama algoritması (Elbeltagi ve ark., 2007) gibi çeşitli modern metasezgisel optimizasyon yöntemleri TCTP'lerin çözümü için uygulanmıştır. Metasezgisel optimizasyonun arkasındaki kavram, elde edilen çözümün kalitesini iteratif olarak iyileştirmek ve böylece algoritmaların değişken doğası nedeniyle yerel bir optimuma bağlı kalmaktan kaçınmaktır. Bu özellik, genellikle elde edilmesi zor olan global optimum çözümü bulma şansını artırmaktadır. Bu tür algoritmalar evrimsel hesaplama ve sürü zekası ilkelerine dayanmaktadır ve global bir çözümün mümkün olmadığı durumlarda optimuma yakın çözümler bulmak için çok kullanışlıdır. Yukarıda bahsedilen yöntemlere ek olarak, Diferansiyel Evrim Algoritması (Hafızoğlu, 2006) ve Dal/Sınır Algoritması (Narayanan ve Suribabu, 2014) da TCTP'lerin optimizasyonu için kullanılmaktadır.

Çeşitli metasezgisel algoritmalardan tam olarak yararlanmak için, bazıları problemin optimal çözümüne ulaşmak için gereken hesaplama çabasını optimize etmek ve çözümlerin optimalliğini artırmak amacıyla hibritleştirilmiştir (Aminbakhsh, 2013). Yukarıda açıklanan tüm metasezgiseller, benzetilmiş tavlama hariç, popülasyon tabanlı algoritmalardır. Algoritmanın iteratif sürecini başlatmak için, problem sınırları içinde rastgele oluşturulan olası çözümlerden oluşan bir başlangıç popülasyonu oluşturulmalıdır. Daha sonra, bu çözümler önceden tanımlanmış durdurma kriteri karşılanana kadar takip eden iterasyonlar yoluyla iyileştirilmektedir. Başlangıç popülasyonundaki olası çözümlerin, kullanılan optimizasyon algoritmalarının performansı üzerinde doğrudan bir etkiye sahiptir.

TCTP'lerin çok amaçlı optimizasyonu için kullanılan öğretme öğrenme tabanlı optimizasyonun (TLBO) yakınsama kabiliyetini ve performansını iyileştirmek için başlangıç popülasyonu yaklaşımı da bulunmaktadır. Bu yaklaşım, minimumun minimumu (min-min) yaklaşımının uygulanmasından elde edilen belirli çözümleri, başlangıç popülasyonunu oluşturmak için rastgele üretilen geri kalan çözümlerle bütünleştirmektedir. Min-min yaklaşımı, TCTP'nin basit tek amaçlı versiyonu için kabul edilebilir çözümleri keşfetmek için kullanılan iyi bilinen bir optimizasyon algoritmasıdır. TCTP'lerin tek amaçlı optimizasyonunun proje süresini veya maliyetini en aza indirerek yapılabileceği görülebilmektedir. Belirli bir proje maliyeti için, TCTP'nin çözüm uzayında tek bir proje süresi belirlenebilmektedir, ancak belirli bir proje süresi için birçok proje maliyeti değeri olabilmektedir. Bu sonuç, TCTP'lerin optimizasyonu ile ilgili teknik literatürde bildirilen sonuçlardan ve ayrıca ele alınan TCTP'nin çözüm uzayının incelenmesinden çıkarılabilmektedir. Yaklaşımın optimizasyon sürecinin hem yakınsama hızı hem de doğruluğu açısından faydası bulunmaktadır.

2.1.2. Proje hızlandırma problemi (project crashing)

Proje yönetimine yönelik en popüler yaklaşımlardan ikisi olan Kritik Yol Yöntemi (CPM) ve Program Değerlendirme ve Gözden Geçirme Tekniği (PERT), 1950'lerde İkinci Dünya Savaşı'ndan kısa bir süre sonra geliştirilmiştir. CPM, bir projenin süresini hesaplamak için deterministik bir yaklaşımdır ve PERT, aktivite sürelerindeki belirsizliği dikkate alarak CPM'i geliştiren ve projenin belirli bir zamanda tamamlanma olasılığını hesaplayan stokastik bir yaklaşımdır (Lee ve Arditi, 2006). J.E. Kelly ve

DuPont'tan M.B. Walker, kimyasal işleme tesislerinin bakım zamanlarının projelendirilmesine yardımcı olmak için 1957 yılında CPM'yi geliştirmiştir. PERT, kısa bir süre sonra ABD Donanması tarafından Polaris füzesinin geliştirilmesini yönetmek için geliştirilmiştir, ancak orijinal PERT Donanma raporunda (1958) geliştiricilerin isimleri belirtilmemiştir (Haga ve O'keefe, 2001). Hem CPM hem de PERT ağ tabanlı tekniklerdir ve PERT CPM'in bir uzantısıdır. Öncelik ilişkileri ve sürelerden ağ diyagramı oluşturulduktan sonra, PERT planlaması aşağıdaki adımları içermektedir: İlk olarak, kritik yolu belirlemek için ağ diyagramı incelenmeli ve analiz edilmelidir. Bu, ağdaki en uzun yoldur ve projeyi tamamlamak için mümkün olan en kısa süreyi göstermektedir. İkinci olarak, her bir aktivite için beklenen süre, her bir aktivitenin olasılığına bağlı olarak beklenen sürelerin dağılımı dikkate alınarak hesaplanmalıdır. Üçüncü olarak, projenin beklenen tamamlanma süresi, kritik yol aktivitelerinin beklenen süreleri toplanarak hesaplanabilmektedir. Projenin verilen sürede tamamlanma olasılığı, kritik yoldaki her bir aktivitenin olasılığı dikkate alınarak hesaplanabilmektedir.

2.1.2.1. Her aktivite için gereken sürenin tahmin edilmesi

PERT, aktivitelerin tamamlanma sürelerindeki belirsizliği hesaba katmak için genelleştirilmiş beta dağılımını takip ettiği varsayılan üç farklı zaman tahmini kullanmaktadır. Bu tahminler minimum zaman (a), en olası zaman (m) ve maksimum zamandır (b). İyimser zaman (a) aktivitenin tamamlanabileceği en kısa süreyi, en olası zaman (m) tamamlanma olasılığı en yüksek olan süreyi ve maksimum zaman (b) ise aktivitenin gerektirebileceği en uzun süreyi ifade etmektedir.

Her bir aktivite için beklenen süre, üç süre tahmininin ağırlıklı ortalaması hesaplanarak tahmin edilebilmektedir. Bu ortalama aşağıdaki formül kullanılarak hesaplanmaktadır:

$$\text{Beklenen süre} = [(a + 4m + b) / 6] \tag{2.1}$$

$$\text{Varyans} = [(b - a) / 6]^2 \tag{2.2}$$

şeklindedir. Beklenen süreyi ve varyansı tahmin etmeye yönelik bu yöntem PERT'in aktivite tamamlanma sürelerindeki belirsizliği hesaba katmasına olanak tanıyarak projelerin hem verimli hem de gerçekçi bir şekilde planlanmasını ve koordine edilmesini sağlamaya yardımcı olmaktadır.

2.1.2.2. Her aktivitenin slack (esnek/sarkma) süresinin hesaplanması

Esneklik süresi proje yönetiminde önemli bir kavramdır ve kritik olmayan bir yol aktivitesinin projeyi geciktirmeden aktivitenin geciktirilebileceği süreyi ifade etmektedir. Kritik yol(lar) içindeki aktivitelerin sıfır esneklik değerine sahip olduğunu ve bu nedenle bu aktivitelerdeki herhangi bir gecikmenin projenin bütününde bir gecikmeye neden olacağını unutmamak önemlidir.

Herhangi bir aktivitenin esnek zamanını hesaplamak için aşağıdaki dört değeri belirlemek gerekmektedir: ES - En Erken Başlangıç zamanı, EF - En Erken Bitiş zamanı, LS - En Son Başlangıç zamanı, LF - En Son Bitiş zamanıdır. Bu değerler genellikle projedeki aktivitelerin sırasına ve her bir aktivitenin alması beklenen süreye bakılarak belirlenebilmektedir. Bu değerler bilindiğinde, herhangi bir aktivite için esneklik süresi, en erken bitiş süresinin en son bitiş süresinden çıkarılmasıyla kolayca hesaplanabilmektedir.

Bu esnek süre anlayışı, projenin zaman yönetimini optimize etmelerini sağladığından proje yöneticileri için çok değerlidir. Kritik yolun dışındaki aktiviteler mevcut esnemiş sürelerine kadar hızlandırılabilir veya yavaşlatılabilirse, toplam proje süresi etkilenmeyecektir, bu da herhangi bir gecikmeyi veya beklenmedik problemleri yönetmek söz konusu olduğunda son derece yararlı olabilmektedir. Proje yöneticileri esnek süreden yararlanarak projenin zamanında ve mümkün olan en yüksek standartta tamamlanmasını sağlayabilmektedir.

Bu süreler, projenin geçmiş verilerine dayanan ilgili aktiviteler için beklenen süre kullanılarak hesaplanmaktadır. Her bir aktivitenin en erken başlangıç ve bitiş süreleri, ağ boyunca ileriye doğru çalışılarak ve önceki aktivitelerin bağımlılıkları göz önünde bulundurularak bir aktivitenin başlayabileceği ve bitirebileceği en erken zaman belirlenerek elde edilmektedir. En geç başlangıç ve bitiş zamanları, ağ üzerinden geriye doğru çalışılarak ve bir aktivitenin projeyi geciktirmeden başlayıp bitirebileceği en geç zaman belirlenerek bulunmaktadır. Bu, söz konusu aktivitenin önceki aktiviteleri ve ağda kaç aktivitenin ona bağımlı olduğu dikkate alınarak yapılmaktadır. Her bir aktivitenin en geç ve en erken bitişi arasındaki fark, o aktivitenin esnek süre = (Ls - Es) olarak hesaplanan esnekliğidir. Bu esnek süre, gerektiğinde geciktirilebilecek veya projeden çıkarılabilecek potansiyel aktiviteleri belirlemek için kullanılabilmektedir. Ayrıca projenin kritik olan aktiviteleri için bir zaman tamponu sağlamak amacıyla da

kullanılabilmektedir. Her bir aktivite için esnek süre hesaplanarak, projenin başarısı için en kritik olan aktiviteler belirlenebilmekte ve zamanında tamamlanmaları için bunlara yeterli öncelik ve kaynak verilmesini sağlamak mümkün olabilmektedir.

2.1.2.3. Kritik yolları belirleme

Kritik yol, projenin toplam tamamlanma süresinin belirlenmesine yardımcı olduğu için proje yönetiminin temel bir parçasıdır. Her bir sıradaki aktivitelerin süreleri toplanarak ve projedeki en uzun yol belirlenerek tanımlanmaktadır. Örneğin, bir proje ağ diyagramında 14 mevcut yol varsa ve 1, 4 ve 7 numaralı yollardaki aktivitelerin süreleri toplandığında diğerlerine kıyasla en yüksek değer elde ediliyorsa, bu değer projenin tamamlanma süresi olacaktır ve 1, 4 ve 7 numaralı yollar kritik yollar olacaktır. Projenin zamanında bitirilebilmesi için kritik yoldaki tüm aktivitelerin zamanında tamamlanması gerekmektedir. Kritik yol üzerindeki herhangi bir aktivite gecikirse, projenin tamamlanma süresi de gecikecektir. Bir projede birden fazla kritik yol olabilmekte ve projenin zamanında bitirilmesini sağlamak için bunların hepsinin yakından izlenmesi gerekmektedir. Örneğin, "C" aktivitesi kritik yolların hiçbirinde mevcut değildir, ancak 3, 6, 8 ve 11 numaralı yollarda mevcuttur. Bu dört yoldan 11 numaralı yol, 3, 6, 8 ve 11 numaralı yollardaki aktivitelerin süreleri toplandığında en büyük değeri sunmaktadır. Bu nedenle, "C" aktivitesinin boşta kalma süresi, 11 numaralı yolun toplam süresinin, proje süresi olan herhangi bir kritik yol 1, 4 veya 7'nin toplam süresinden çıkarılmasıyla belirlenebilmektedir. Ayrıca, projenin kritik yol(lar)ı, aktivitelerin hiçbirinde esneklik bulunmayan ağ üzerindeki yol(lar) incelenerek belirlenebilmektedir. Bu, ağdaki en uzun yol belirlenerek ve ardından bu yolda herhangi bir esneklik süresi olmayan aktiviteler belirlenerek yapılabilmektedir. Bunu yaparak, esnek zamanı olan herhangi bir aktivite kritik yolun bir parçası olmaktan çıkarılabilmektedir. Sonuç olarak, en uzun yolda kalan aktiviteler kritik yolun parçası olan aktivitelerdir.

2.1.2.4. Proje hızlandırma

Proje tamamlanma süresindeki varyans, kritik yoldaki aktivitelerin tamamlanma sürelerindeki varyansların toplanmasıyla hesaplanabilmektedir. Bu varyans, kritik yoldaki aktivite sayısının merkezi limit teoreminin uygulanabileceği kadar büyük olduğu varsayımı altında, projenin belirli bir tarihe kadar tamamlanma olasılığının hesaplanmasını sağlamaktadır. Bu varsayımla, istenen süre ile beklenen tamamlanma

sürelerinin birbirine eşit olması durumunda projenin zamanında tamamlanma olasılığı %50'dir.

Kritik yol projenin tamamlanma tarihini belirlediğinden, proje yöneticileri kritik yoldaki aktivitelerin süresini azaltmak için kaynak ekleyerek projeyi hızlandırmayı tercih edebilmektedir. Bu, planlanan aktivitelerin gerçekleştirilmesini hızlandırmak için projeye daha fazla para harcanmasını içeren proje hızlandırma olarak adlandırılmaktadır (Rosenau ve Githens, 2005). Proje hızlandırma, risk içerdiği için dikkatli bir şekilde yapılmalıdır. Örneğin, projeye çok fazla kaynak eklemek aşırı harcamaya ve verimsizliğe yol açabilmektedir. Ayrıca, her bir aktivitenin tamamlanma süreleriyle ilgili her zaman bir miktar belirsizlik olduğundan, projeyi hızlandırmak için gereken kaynak miktarını doğru bir şekilde tahmin etmek zor olabilmektedir.

Bu nedenle hem riskleri hem de potansiyel faydaları dikkate alan, iyi düşünülmüş bir proje hızlandırma stratejisi geliştirmek önemlidir. Bu, proje için gereken kaynak miktarının yanı sıra bu kaynakların eklenmesiyle elde edilebilecek beklenen zaman ve maliyet tasarruflarının tahmin edilmesini de içermektedir. Ayrıca proje yöneticileri, projenin hızlandırılmasının proje çıktısının kalitesi üzerindeki potansiyel etkisini ve bunun müşteri memnuniyetini nasıl etkileyebileceğini de göz önünde bulundurmalıdır. Proje yöneticileri, tüm bu faktörleri dikkatle değerlendirerek, projenin tamamlanma süresini kısaltmak için proje hızlandırılmasını takip edip etmeme konusunda bilinçli kararlar verebilmektedirler. Bir projenin hızlandırılması söz konusu olduğunda en uygun maliyetli kararı vermek için, son yirmi yılda simülasyon tabanlı çeşitli hızlandırma yöntemleri geliştirilmiştir. Bu yöntemler sezgiseldir, yani tatmin edici çözümler sunmak üzere tasarlanmışlardır, ancak her zaman en uygun çözümü sunamamaktadırlar. Bu zorluğun üstesinden gelmek için araştırmacılar, belirlenen tamamlanma süresini karşılarken gerekli maliyetleri en aza indirebilecek optimum hızlandırma için bir algoritma geliştirmeye çalışmışlardır. Bu algoritmanın, projenin verilerini doğru bir şekilde analiz edebilmesi ve projeyi hızlandırmak için en uygun maliyetli yaklaşımı belirleyebilmesi gerekliydi. Ayrıca, seçilen yaklaşımla ilişkili olabilecek potansiyel riskleri ve bunların projenin zaman çizelgesini nasıl etkileyebileceğini de dikkate alması gerekmekteydi. Böyle bir algoritma geliştirerek, herhangi bir proje hızlandırma kararının mümkün olan en uygun maliyetli şekilde alınmasını sağlamak mümkün olabilmektedir.

2.2. Optimizasyon

Optimizasyon, sınırlı kaynakların kısıtlarını dikkate alarak bir karar problemine en iyi çözümü bulma yöntemidir. Farklı seçenekleri karşılaştırarak ve bir amaç fonksiyonunu minimize veya maksimize ederek optimum çözümü bulma sürecidir. Maliyet fonksiyonu veya amaç fonksiyonu değeri olarak da bilinen bu amaç fonksiyonu, mümkün olan en iyi seçeneği belirlemek için hesaplanabilmektedir. Optimizasyon teknikleri tek veya çok amaçlı problemleri çözmek için kullanılmaktadır. Gerçek hayattaki optimizasyon problemleri genellikle birden fazla hedef içermektedir, bu da çözümün birden fazla kriteri karşılaması gerektiği anlamına gelmektedir.

Başarılı bir optimizasyon elde etmek için, problemi tanımlamak ve mevcut alternatifler kümesine karar vermek esastır. Alternatifler kümesi oluşturulduktan sonra, en iyi çözümü belirlemek için her seçenek için amaç fonksiyonu hesaplanmalıdır. Bu süreç, maksimum veya minimum amaç fonksiyonunu elde etmek için girdilerin veya değişkenlerin değiştirilmesini içermektedir. Optimizasyonun iteratif bir süreçtir ve problem değiştikçe veya karşılanması gereken kriterler değiştikçe amaç fonksiyonunun güncellenmesi gerekmektedir. Bu şekilde optimizasyon, belirli bir durumda en uygun çözümü bulmak için etkili bir yöntem olabilmektedir. Çok amaçlı optimizasyon, amacın birbiriyle çelişen bir dizi hedef için en iyi çözümü bulmak olduğu özel bir optimizasyon problemi türüdür. Tüm hedeflere ulaşabilmek için tek bir optimum çözüm elde etmek mümkün olmadığından, bunun yerine alternatif (dengeli) optimum çözümler aranmaktadır. Genel sistem performansını en üst düzeye çıkarma ve çeşitli hedefleri dengeleyen çözümler bulmayı içeren çok amaçlı optimizasyon da bulunmaktadır. Bu süreç genellikle Pareto optimallik olarak adlandırılmakta ve diğer hedeflerden ödün vermeden sistemde daha fazla iyileştirme yapılamayacağı fikrini ifade etmektedir. Bu nedenle, genel sistem performansının maksimize edilmesini sağlamak için her bir hedef arasındaki dengeleri dikkatle değerlendirmek gerekmektedir.

2.2.1. Optimizasyonu çözüm yaklaşımları

Optimizasyon problemleri, Klasik Yöntemler ve Sürü Zekası Algoritmaları olmak üzere iki ana strateji kullanılarak çözülebilmektedir.

2.2.1.1. Klasik yöntemler

Klasik yöntemler, rastgele seçilen bir başlangıç noktasıyla başlayan deterministik bir yaklaşım kullanmaktadır. Daha sonra düzenli bir şekilde bir arama yönü tanımlanmakta ve istenen sonuç elde edilene kadar aynı işlem tekrarlanmaktadır. Bu yöntemler iki kategoriye ayrılabilmektedir: sayısal yöntemler ve gradyan tabanlı yöntemlerdir. Sayısal yöntemler, amaç fonksiyonu tarafından verilen değerlere ve aramalarını yönlendirmek için herhangi bir kısıta bağlıyken, gradyan tabanlı teknikler yakınsama için bu bileşenlerin türevlerine dayanmaktadır. Sayısal yöntemler, arama sürecini yönlendirmek için çok sayıda fonksiyon gerektirdiğinden gradyan yöntemleri kadar hızlı değildir, ancak algoritmada büyük değişiklikler yapılmadan çeşitli problemler için kullanılabilmektedirler. Ayrıca, klasik yöntemler, optimum çözüme ulaşma kabiliyetlerini güçlü bir şekilde etkileyen bir başlangıç çözümüne dayanmaları nedeniyle engellenmektedir. Bu yaklaşımlar, ayrık bir çözüm uzayına sahip optimizasyon problemleriyle uğraşırken verimli olmamaktadır ve genellikle yerel çözümlerde takılıp kalmaktadırlar. Ayrıca, tüm optimizasyon problem sınıflarına uygulanabilen evrensel bir algoritma bulunmamaktadır (Deb, 2001).

2.2.1.2 Sürü zekası algoritmaları

Dorigo'nun (1992) doktora tezinde tanıtılmasından bu yana, Karınca Kolonisi Optimizasyonu (ACO) alınına ilgi artmıştır. Bu büyüme, Dorigo ve ark.'nın (1996) Karınca Sistemi (AS) ve Dorigo ve Gambardella'nın (1997) Karınca Koloni Sistemi (KKS) adlı kitaplarının yayınlanmasıyla teşvik edilmiştir. Bunun sonucunda ACO, Parçacık Sürü Optimizasyonu (PSO) ile birlikte en çok çalışılan ve uygulanan iki sürü zekası yönteminden biri haline gelmiştir (Kennedy ve Eberhart, 1995). Karınca Kolonileri Metasezgiseli, karıncaların bir kolonideki davranışlarına dayanmakta ve doğadan ilham alan bir optimizasyon algoritmaları sınıfı oluşturmaktadır. Bu davranış, nispeten basit bireylerin otonomsine dayanmakta ve yiyecek kaynakları ile yuva arasındaki en kısa yolları bulmak için kullanılmaktadır.

Sürü zekası ve öz-organizasyonun temel kavramlarını anlamak için karıncalara biyolojik bağlamda bakmak önemlidir. Karıncalar, kendi kendilerini organize etme ve görevleri yerine getirmek için bir kolonideki diğer karıncalarla birlikte çalışma konusunda yeteneğe sahiptir. Bu durum, ünlü çift köprü deneyinde açıkça görülmektedir. Bu deney ACO algoritmasının temelini oluşturmaktadır.

2.3. Sürü Zekası

Sürü zekası, bir grubun kolektif eylemlerinden doğan akıllı davranışlara odaklanan yapay zekanın bir alt bölümüdür. Bu alandaki araştırmacılar, kolektif davranış ilkelerine dayalı algoritmalar ve robotlar yaratmakta ve genellikle doğrudan geliştiriciler tarafından programlanmayan sonuçlara yol açmaktadır. Bunun nedeni, doğal sürüler ve bunların mühendislik bilimlerindeki uygulamaları aracılığıyla keşfedilen sürü üyeleri arasındaki etkileşimlerden kaynaklanan ortaya çıkan özelliklerdir.

2.3.1 Doğada sürü davranışı

Sürü zekasının ana mekanizması, bir grup içindeki bireylerin davranışlarının bir bütün olarak grubun hayatta kalmasına veya çoğalmasına katkıda bulunduğu bir olgudur. Kuş sürülerinden balık sürülerine, böcek bulutlarından antilop sürülerine ve hatta insan vücudundaki hücrelere kadar doğada çok sayıda sürü örneği bulunmaktadır. Büyük bir grup bireyin tek bir organizma gibi hareket edebileceği fikri, sürü zekasını tartışırken göz önünde bulundurulması gereken önemli bir kavramdır. Bir sürüyü neyin oluşturduğuna dair kesin bir cevap olmadığını da belirtmek önemlidir. Bu belirsizlik, bir sürü oluşturmak için birden fazla bireyin gerekli olmasından kaynaklanmaktadır, ancak iki bireyin bir sürü olarak kabul edilmesi için yeterli olup olmadığı belirsizdir. Sadece iki birey olduğunda bile, aralarındaki etkileşimlerin bir sonucu olarak ortaya çıkan bazı küresel davranışlar mevcut olabilmektedir. Bu nedenle bir sürüyü yüzlerce kuş ya da binlerce sinek gibi çok sayıda bireyden oluşan bir grup olarak düşünme eğiliminde olunmaktadır. Ayrıca, bir sürüyü sivrisinekler gibi küçük canlılardan oluşmuş olarak da düşünme eğiliminde olunabilmektedir, ancak insanların bir sürü olarak kabul edilip edilemeyeceği tartışmalı bir konudur.

Bireysel bir bakış açısıyla, yoğun bir otobüs terminaline girmek veya gece gökyüzüne bakmak, bir sürünün parçası olma hissini uyandırmayabilir. Ancak, uzaktan bakıldığında, istasyondaki insanlar, işlerine giderken, hedeflerine ulaşmak için birlikte çalışan bir grup karıncaya benziyor gibi görünebilmektedir. Benzer şekilde, bir teleskopla yıldızlara bakıldığında, sanki tek bir varlığın parçasıymışlar gibi Samanyolu'nun merkezi etrafında koordineli bir şekilde hareket ettikleri görülebilmektedir. Ancak buna gerçekten bir sürü denip denemeyeceği de aynı şekilde tartışma konusudur.

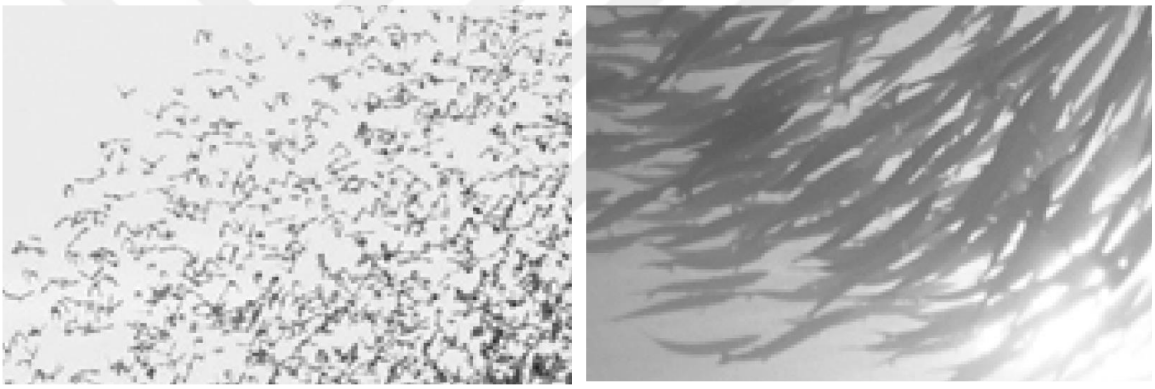
Bir sürüyü neyin oluşturduğunu anlamak için birkaç kriterin dikkate alınması gerekmektedir. İlk olarak, en az iki veya daha fazla "yeterli" birey olması gerekmektedir. İkinci olarak, boyut önemli değildir; küçük bireylerden oluşan bir sürü de büyük bireylerden oluşan bir sürü kadar sürüdür. Üçüncü olarak, bireyler birbirlerinin belirli bir menzili içindeyken birbirleriyle iletişim kurabilmeli veya birbirlerini etkileyebilmelidir. Son olarak, belirli bir düzeyde bütünlük olmalıdır, yani bireylerin iletişim topolojisi, bilginin sürü boyunca yayılabileceği şekilde bağlantılı olması gerekmektedir. Sürünün bir kısmının diğerleriyle bağlantısı kesilirse, genellikle iki farklı sürü olduğu düşünülmektedir.

Bu kriterler kuş sürüleri, balık sürüleri veya karınca kolonileri gibi doğal sürülerin yanı sıra insan kalabalıkları, araç filoları veya bilgisayar ağları gibi yapay sürülere de uygulanabilmektedir. İnsan yapımı sürüler, karmaşık sorunları çözmek ve verimliliği artırmak için kullanılabildiklerinden günümüz dünyasında giderek daha yaygın hale gelmektedir. Sürü fikri korkutucu görünse de, uzaktan bakıldığında aslında oldukça güzel olabilmektedir.

Sürü, koordineli bir şekilde birlikte hareket eden bir grup hayvandır. Bu koordinasyon genellikle gruptaki birkaç bireyin çevrelerindeki potansiyel tehditleri fark etmesiyle tetiklenmektedir. Bu, grubun tüm üyelerinin çevreden (türlerinin diğer üyeleri de dahil olmak üzere) girdi alabilmesi ve buna göre hareket edebilmesi gerektiği anlamına gelmektedir. Girdi ile eylem arasındaki eşleme, sürünün davranışının özelliklerini belirlediği için son derece önemlidir. Eğer eşleme bireylerin birbirlerinden çok uzaklara dağılmasına neden oluyorsa, o zaman sürü uyumlu olmayacak ve dağılacaktır. Öte yandan, eşleme tüm bireylerin tepkisiz kalmasına ve durmasına neden olursa, hareket gerekliliğini ihlal ettiği için sürüden geriye pek bir şey kalmamaktadır. Bu nedenle, ancak bir grup birey hareket, uyum ve yanıt verebilirlik kriterlerinin tümünü karşılayabiliyorsa gerçek bir sürü olarak kabul edilebilmektedir. Sürüler, bir tür kolektif zeka olarak düşünülebilecek üyelerinin etkileşimlerinden ortaya çıkmaktadır. Bireylerin davranışları DNA'larında ve hafızalarında kodlanmış olup, girdilerinin ve mevcut durumlarının eylemleriyle (olasılıksal) eşleştirilmesini belirlemektedir. Bu davranış, sürünün kolektif etkileşimlerinden kaynaklanan, ortaya çıkan bir olgu olarak görülebilmektedir. Mikro ölçekte, sürü kaotik davranmaktadır: bireylerin girdilere ve etkileşimlerine tepkileri deterministik olsa da, sürüdeki tam hareketleri başlangıç

durumlarındaki küçük değişimler için çok farklı olabilmektedir. Bu kaotik davranış, bireylerin davranışları çok basit olsa bile ortaya çıkmakta ve sürünün bir bütün olarak nasıl davranacağını tahmin etmek genellikle zordur. Sürüyü mikro ölçekte incelemek bizim için sınırlı bir alandır, çünkü bireylerin davranışı genellikle sürünün bir bütün olarak ortaya çıkan büyüleyici davranışını ortaya çıkarmamaktadır.

Çoğu insan için çok daha ilginç ve büyüleyici olan şey, bir sürünün genellikle tek bir organizma gibi görünebilmesidir. Bireyler sanki tek bir amaç doğrultusunda hareket ediyormuş gibi görünmektedir. Ağaçlardan uçan bir kuş sürüsü, sanki tek bir canlı varlıkmış gibi bazen genişleyerek bazen de küçülerek ve yoğunlaşarak gökyüzünde yol alıyor gibi görünmektedir. Şekil 2.1(a) bu büyüleyici olguyu göstermektedir. Sürünün davranışı tek tek üyelerinin davranışından çok daha karmaşık olduğundan, bu tür ortaya çıkan davranışlar kolektif zekanın gücünün bir kanıtıdır.



(a) Ağaçlardan birlikte yükselen kuşların sürü olarak hareket etmesi.

(b) Bir grup balığın sürü şeklindebirlikte yüzmesi.

Şekil 2.1: Doğada iki sürü örneği: kuş sürüsü ve balık sürüsü.

Sürünün bu küresel özellikleri, tek tek üyelerinin davranışlarından çok daha fazla düzene sahip olduğunu göstermektedir. Sürü üyelerinin hareketlerini mikro ölçekte anlamak ve tahmin etmek neredeyse imkansızdır, ancak aynı şeyi makro ölçekte yapmak çok daha başarılabilirdir. Sürülerin güzelliği, sürüdeki her bir organizmanın davranışını belirleyen, mikro düzeyde karmaşık ve görünüşte rastgele davranışlar üreten, ancak daha büyük ölçekte sürünün düzenli bir davranışını üreten basit talimatlar dizisine atfedilebilmektedir.

Sürünün mikro ölçekteki kaotik davranışı, söz konusu sürünün türüne bağlı olarak değişmektedir. Örneğin, bir balık sürüsünde, bireysel balıklar sürü içinde hareket edebilmekte, ancak küresel bir perspektiften bakıldığında sürü, suda hareket eden top

şeklindeki tek bir organizma gibi olabilmektedir. Öte yandan, V şeklindeki kuş sürüsü mikro ölçekte de düzenlidir. Sürüdeki bireylerin kesin davranışlarının çok önemli olmadığını da belirtmek gerekmektedir. Önemli olan, sürünün kolektif davranışının öngörülebilir ve düzenli olmasıdır; bu da bireysel üyelerin kolektif davranışının bir sonucudur.

Bu, sürünün koordineli ve akıcı bir şekilde hareket edebilmesinde ve her bir üyenin sürünün genel davranışına yardımcı olacak şekilde hareket etmesinde görülebilmektedir. Bu, sürünün tek tek üyelerinin öngörülemez ve kaotik olabilen davranışlarının aksine bir durumdur. Sürünün koordinasyonu, tek tek üyelerin kolektif davranışlarının bir sonucudur ve sürünün hareketlerinde bu kadar etkili ve verimli olmasını sağlayan da budur.

Karmaşık algoritmalar ve karmaşık süreçler içerdiğinden, sürü zekasında ortaya çıkan davranış ve otonom kavramını anlamak zor olabilmektedir. Bu nedenle araştırmacılar arasında pek çok tartışmaya neden olmuştur. Yerel düzeyde düzensiz gibi görünen davranışların küresel düzeyde organize davranışlar üretebilmesinde sihirli bir şey varmış gibi görünebilmektedir. Ancak durum böyle değildir. Küresel davranış, iyi tanımlanmış bir dizi yerel kuralın sonucudur ve bu nedenle hiç de sihirli değildir. Buna rağmen, bireylerin kesin davranışlarını anlamak, tanımlamak veya tahmin etmek genellikle çok karmaşık ve zahmetlidir. Bu nedenle ortaya çıkan davranış genellikle alışılmışın dışında bir şey olarak algılanmaktadır.

Bir sürünün homojenliği veya heterojenliği, grubun geri kalanından farklı olan bireylerin sayısına bağlıdır. Homojen sürü durumunda, tüm üyeler birbirine benzemekte ve hiçbir birey öne çıkmamaktadır. Bu sürüler birbirlerinin kopyalarından oluşmaktadır, tek fark başlangıç koşullarındadır. Ancak heterojen sürülerde bir ya da daha fazla birey fiziksel özellikleri ya da sürü içindeki işlevleri bakımından diğerlerinden farklıdır.

Doğada genellikle, balık sürülerinde görüldüğü gibi, tüm bireylerin aynı özelliklere sahip olduğu homojen sürüleri görülmektedir. Bununla birlikte, karıncalar gibi bazı türler, farklı bireylerin grup içinde farklı görevlere sahip olduğu heterojen sürüler oluşturacak şekilde evrimleşmiştir. Örneğin, asker karıncalar işçi karıncalardan çok daha güçlüdür ve erkeklere sadece kraliçeyi dölleme görevi verilmektedir. İşçi karıncalar ise yuva yapımından larva beslemeye ve yiyecek aramaya kadar farklı

görevleri yerine getirmektedir. Önemli bir besin kaynağı bulunursa, diğer işçi karıncalar yiyecek aramaya geçmek için işe alınmakta ve yuva hasar görmüşse, yuvayı onarmaya yardımcı olmak için başka işçiler de işe dahil edilebilmektedir. Bu şekilde, heterojen sürüler farklı çevresel değişikliklere uyum sağlayabilmekte ve bu da onları homojen muadillerine göre daha dirençli hale getirmektedir.

2.3.2 Mühendislikte sürü zekâsı prensipleri

Doğadaki sürü zekâsı prensipleri anlaşılarak, bunların insanlar için faydalı bir şekilde kullanılabilecek mühendislikte kullanılan sürülere dönüştürülebilmesi mümkündür. Bu mühendislik sürüleri, ulaşmaya çalıştıkları kendi hedefleri olan otonom ajanlardan oluşmaktadır. Ajanlar, genellikle sınırlı bir menzil içinde koordine olmak ve iş birliği yapmak için birbirleriyle etkileşime girmektedir ve iletişim kurmaktadır. Merkezi bir denetleyici ya da gözetmen ve hiyerarşik bir yapı bulunmamaktadır.

Bu mühendislikte kullanılan sürülerin doğal sürülere kıyasla avantajları ölçeklenebilirlik, sağlamlık, esneklik ve üretim maliyetleridir. Özellikle ölçeklenebilirlik burada önemli bir faktördür. Ajanlar sınırlı bir etkileşim aralığına sahip olduğundan, yeni bir bireyin eklenmesi tüm sürüyü değil, yalnızca birkaç ajanın davranışını etkilemektedir. Bu, daha fazla hesaplama gücüne ihtiyaç duymadan daha fazla bireyin eklenebileceği anlamına gelmektedir.

Bazı ajanlar başarısız olsa bile sürü çalışmaya devam edebildiği için sağlamlık da önemlidir. Bunun nedeni sürünün yeni duruma adapte olabilmesi ve uyum sağlayabilmesidir. Esneklik de önemli bir faktördür, çünkü sürü çevreye uymak için davranışını ve hedeflerini gerektiği gibi değiştirebilmektedir. Son olarak, sürü daha az kaynak gerektirdiği ve daha verimli olduğu için üretim maliyetleri geleneksel yöntemlere göre daha düşüktür.Tüm sürünün iletişim topolojisi yeni birey için oldukça önemlidir, çünkü diğer bireylerle etkileşim kurarak tüm sürüyü etkileyebilmektedir. Sağlamlık açısından, bu sınırlı etkileşim aralığı, herhangi bir bireyin tüm sürü üzerinde anlık bir etkiye sahip olmasını önlemekte, bu da belirli durumlarda inanılmaz derecede yararlı olabilmektedir. Örneğin, bir birey arızalanırsa veya ortadan kaldırılırsa, menzilindeki bireyler onun arızalı davranışını telafi edebilmekte, hatta işlevselliğini tamamen devralabilmektedir. Buna ek olarak, sürünün geri kalanı sadece minimum düzeyde etkilenmektedir. Dahası, farklı koşullara göre davranışlarını değiştirebildikleri

için tasarlanmış sürüler açısından esneklik de yüksek öneme sahiptir. Örneğin, iyi tasarlanmış bir robot sürüsü bilinmeyen bir alanı keşfetmek için dağılabilmekte veya geniş bir boşluk üzerinde köprü oluşturmak için birbirlerine bağlanabilmektedir (bkz. Şekil 2.2(b)).

Sürü davranışı, son birkaç on yılda kapsamlı bir şekilde incelenen dikkate değer bir doğa olayıdır. Ayrıca laboratuvarda ve sahada başarılı bir şekilde taklit edilmiş ve bazı robotik sistemler oluşturmak için kullanılmıştır. Mühendislik sürü sistemlerinin temel avantajlarından biri, bir sürüdeki bireyler genellikle nispeten basit olduğundan ve hatta bazen ucuz bileşenler kullanılarak inşa edilebildiğinden, maliyetlerinin oldukça düşük olabilmesidir. Bu, sürülerin sağlamlık özelliğinden kaynaklanmaktadır: sürünün kolektif davranışı genellikle bireysel hatalara karşı merkezi bir sistemden daha dayanıklıdır, yani bazı parçaları arızalansa bile sistem bir bütün olarak çalışmaya devam edebilmektedir. Şekil 2.2'de 2010 yılında son teknoloji ürünü olan iki mühendislik ürünü sürü örneği gösterilmektedir. Bu örnekler sürü robotiğinin olağanüstü potansiyelini ortaya koymakta ve bize gelecekteki uygulamalara yönelik olasılıklar hakkında bir fikir vermektedir.

(a) Bir afet bölgesinde yerdeki birden fazla kişi arasında bağımsız olarak acil durum kablosuz ağları (SMAVNET'ler) oluşturabilen bir grup Mikro Hava Aracının (SMAV'ler) bir örneği.

(b) ULB'deki SWARM-BOT projesi, tek bir robotun kendi başına geçemeyeceği kadar geniş boşlukları kapatmak için robotik sistemlerin kendilerini otonom olarak bir araya getirme, düzenli bir şekilde düzenleme ve farklı şekillere dönüşme yeteneklerini araştırıyor ve test etmektedir.

Şekil 2.2: Mühendislik sürülerini hedefleyen iki proje örneği

İstenen küresel davranışı elde etmek için bireyleri ve etkileşimlerini tasarlamanın temel zorluğuna ek olarak, sürüleri tasarlamak için dikkate alınması gereken birkaç önemli sorun daha bulunmaktadır. Örneğin, sürünün performansının matematiksel olarak nasıl

garanti edilebileceği, sürünün performansının nasıl test edilip ölçüleceği, iletişimdeki gecikmeler ve hataların sürü içinde nasıl yayılıp sürü davranışını etkileyeceği gibidir.

Zorluklara rağmen, mühendislik sürüleri için hala çok fazla potansiyel bulunmaktadır. Davranışlarının karmaşıklığını anlayarak, sürülerin istenen sonuçlara ulaşmasına yardımcı olacak etkili algoritmalar ve stratejiler geliştirmek mümkündür.

Birbirine bağlı araçlardan oluşan ağların geliştirilmesi sayesinde karmaşık görevler daha verimli ve daha yüksek doğrulukla tamamlanabilmektedir. Örneğin, küresel iletişim ve konumlandırma hizmetleri sağlamak için uydu sürüleri konuşlandırılması, uzay araçları sürülerinin uzayın ve gezegenlerin bilinmeyen bölgelerini keşfetmek için kullanılması, birlikte çalışan kurtarma robotlarının geliştirilmesi, tehlike altındaki insanları kurtarmak için sürülerden yararlanılması, arabalar ve diğer araçların sürü üyeleri olarak birbirleri ile anlaşması ve trafik ağının genel performansı araçların otonomsi gibi alanlarda yararlanılması olasıdır.

Optimizasyon veya parametre uzayı olarak adlandırılan sanal bir dünyada optimizasyon problemlerini çözmek için sürü zekası algoritmaları da geliştirilmiştir. En yaygın kullanılan sürü zekası optimizasyon yöntemleri Parçacık Sürü Optimizasyonu ve Karınca Kolonisi Optimizasyonudur.

2.4. Karınca Kolonileri

Kendi kendini yöneten yapıları sayesinde yüz milyonlarca yıl boyunca yüksek direnç gösterebilen karınca kolonileri bulunmaktadır. Antarktika dışında dünyanın dört bir yanına yayılmışlardır.

2.4.1. Doğada karıncalar

Karınca kolonisi, bir kraliçe, erkek karıncalar ve (dişi) işçi karıncalardan oluşan karmaşık ve iyi organize olmuş bir toplumdur. Koloni, bakire bir kraliçe ve birkaç erkeğin koloni kurmak için yeni bir yer aramak üzere yuvadan ayrılmasıyla kurulmaktadır. Bu süreçte kraliçe erkeklerle çiftleşmekte, ardından erkekler ölüp kraliçe kendini toprağa gömmektedir ve 15 yıl kadar sürebilen hayatının geri kalan kısmında yumurtlamaya başlamaktadır. İlk yumurtalar larvaya dönüştükten sonra pupaya ve son olarak da yetişkin işçilere dönüşmektedirler. Bu işçiler, yuva içindeki ve çevresindeki görevlerin çoğundan sorumlu oldukları için kolonin işleyişi için çok önemlidir.

Karıncalar yiyecek arama konusunda organizedirler. Karıncalar yiyecek bulmak için yuvadan ayrılmakta ve dönüş yolunda feromon adı verilen özel bir kimyasal madde izi oluşturmaktadırlar. Yuvadan ayrılan karınca geri döndüğünde, diğer karıncalar yiyecek için bölgeyi keşfetmenin güvenli olduğunu bilmektedirler. Buna ek olarak, karıncalar yuvadan toprak parçaları alıp yuva girişine yığmakta ve bu da o koloniye ait özel bir kimyasal özellik ile işaretlenmeyi sağlamaktadır. Diğer karıncaların yuva girişine geri yönlendirilmesinin bir yolu olarak hizmet etmektedir. Ayrıca, bazı karıncalar asker karıncalar olarak diğer kolonilerden gelen davetsiz karıncalara saldırarak bölgeyi koruma görevi de görmektedir.

Ortalama olarak, işçilerin yaklaşık %25'i yuvanın dışında yiyecek aramakta, diğer %25'i yuvanın içindeki larvalarla ilgilenmekte ve kalan %50'si de yuvanın içinde boş durmaktadır. Bu inanılmaz bir organizasyon ve verimlilik göstergesidir ve karıncaların çevrelerine nasıl hızla uyum sağlayabildiklerini ve hayatta kalmak için kolektif güçlerini nasıl kullanabildiklerini göstermektedir.

Yuvanın dışında çalışan karıncalar yiyecek ve diğer kaynakları toplamaktan, yuvanın içindeki karıncalar ise larvaları beslemekten sorumludur. Karıncalar karşılaştıklarında birbirlerine dokunduklarında, kokuyu hissedebilir ve kendilerine verilen göreve devam etmeleri veya başka bir şeye geçmeleri gerektiğine karar verebilmektedirler. Bu dinamik süreç, karıncaların görevlerini bu kadar etkili bir şekilde değiştirmelerini sağlayan şeydir.

Örneğin, çok fazla yiyecek bulunur ve yuvaya getirilirse, yiyecek arayan karıncalar bu patikaya odaklanacak ve diğer yerlerde daha düşük yoğunlukta yiyecek arayan karınca bırakacaktır. Bu da diğer karıncaların ihtiyaç duyulduğunda yiyecek arama görevine geçmesine neden olmaktadır. Ancak, her yöne geçişin mümkün olmadığı tespit edilmiştir. İşçiler yiyecek aramaya geçebilmektedir, ancak yiyecek arayan karıncalar asla başka bir şeye geçmemektedir.

Kraliçe hayatta olduğu sürece, ilk döllenmeden itibaren yumurtlamaya devam etmekte ve bu da koloninin boyut olarak büyümesinin temelini oluşturmaktadır. Dolayısıyla koloninin büyüklüğü, koloninin yaşının bir fonksiyonudur ve koloninin belirli özelliklerini ortaya koymaktadır. Örneğin, genç koloniler yuvalarının etrafındaki ani değişikliklere daha uyumludur ve bilinen iyi besin kaynaklarına daha fazla

odaklanmaktadırlar. Nadiren risk almakta ve yuvadan daha uzakları keşfetmektedirler. Öte yandan, daha yaşlı kolonilerin yakın çevrelerindeki değişikliklere daha az uyum sağladıkları ve risk alma ve daha uzakları keşfetme olasılıklarının daha yüksek olduğu görülmüştür.

Karınca kolonilerini karınca tabanlı algoritmaların geliştirilmesi için ilginç bir ilham kaynağı haline getiren temel özellik, karıncalar arasındaki etkileşimdir. Bu etkileşim çok fazla sayıda gerçekleşmekte ve belirli bir derecede rastgelelik içermektedir. Karıncalar daha fazla seçenek keşfetmek ve en iyi çözümleri bulmak için rastgelelikten yararlanabildiklerinden, karınca tabanlı algoritmaları bu kadar güçlü ve verimli kılan da bu rastgeleliktir. Rastgelelik aynı zamanda karıncaların değişen koşullara hızla adapte olmalarını ve en verimli çözümleri bulmalarını sağlamaktadır. Karınca tabanlı algoritmaların son yıllarda bu kadar popüler hale gelmesinin nedeni budur, çünkü karmaşık problemleri geleneksel algoritmalardan daha verimli bir şekilde çözebilmektedirler.

Gordon (2003), karıncaları dışarıdan bir bakış açısıyla gözlemleyen birinin, davranışlarının çok etkili değilmiş gibi görünebileceğini ve sık sık kendini onlara yardım etme dürtüsü hissederken bulduğunu savunmaktadır. Ancak, kolonin kolektif davranışı açısından bakıldığında, ortaya çıkan davranış oldukça başarılı ve büyük ölçüde öngörülebilirdir. Tek tek karıncaların davranışları mükemmel olmasa da, kolonin kolektif davranışı göz önünde bulundurulduğunda sonuçlar çok iyidir. Ayrıca, bu kolektif davranış esneklik, ölçeklenebilirlik ve sağlamlık gibi ek faydalara da sahiptir.

2.4.2 Çift köprü deneyi

Deneubourg ve ark. (1990) ve Goss ve ark. (1989), yiyecek arayan karıncaların otonom davranışını modellemek için kullanılan ikonik çift köprü deneyini gerçekleştiren ilk kişilerdir. Bu model o zamandan beri ilk KKA algoritmalarının temelini oluşturmaktadır.

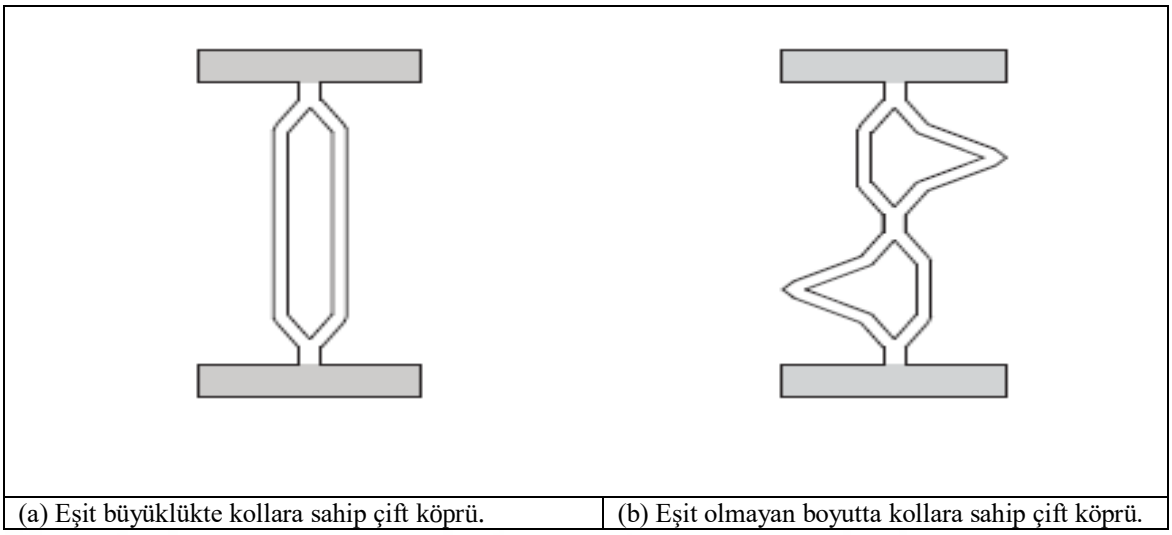
İlk çift köprü deneyi Deneubourg ve ark. (1990) tarafından gerçekleştirilmiş ve Arjantin karıncası Iridomyrmex humilis'e odaklanmıştır. Ancak 1990'ların başında yapılan bir yeniden sınıflandırma nedeniyle bu türün bilimsel adı Linepithema humile olarak değiştirilmiştir. Bu karıncanın işçileri yaklaşık 3 milimetre uzunluğundayken, kraliçe iki

ila dört kat daha uzun olabilmektedir. Kuzey Arjantin, Uruguay, Paraguay ve Güney Brezilya'ya özgü olan Arjantin karıncası, büyük ölçüde insan faaliyetleri nedeniyle Amerika Birleşik Devletleri, Avrupa ve Japonya da dahil olmak üzere dünyanın diğer birçok bölgesinde görülmüştür. Bu da onun en yaygın istilacı karınca türlerinden biri haline gelmesine yol açmıştır (Tsutsui ve ark., 2001).

Arjantin karıncası genellikle "haşere" olarak kabul edilmektedir ve diğer karınca türleriyle karşılaştırıldığında davranışı oldukça sıra dışıdır. Normalde birbirinden uzak yerlerde yaşayan karıncalar bir araya geldiklerinde agresif davranışlar sergilemektedir ancak farklı bölgelerde yaşayan Arjantin karıncaları agresif olmayan davranışlar sergilemektedirler. Bu durum Walker (2009) tarafından yürütülen ve Avrupa, Kaliforniya ve Japonya'dan karıncaların bir araya getirildiği bir deneyde gösterilmiştir. Aralarındaki coğrafi mesafeye rağmen karıncalar saldırganlaşmamış, bunun yerine eski arkadaşlarmış gibi davranmış, antenlerini ovuşturmuş ve hiçbir saldırganlık belirtisi göstermemişlerdir.

Arjantin karıncasını çift köprü deneyi için özellikle ilginç kılan şey, feromonları yalnızca yuvalarına yiyecekle dönerken değil, aynı zamanda hareket halindeyken, yiyecek ararken de bırakmalarıdır. Bu davranış, Deneubourg ve ark. (1990) tarafından yürütülen ve iki kapalı alanın bir köprü ile birbirine bağlandığı bir deneyde kullanılmıştır. Karıncalar her iki alanı da keşfetmek için köprüyü kullanabilmiş ve deney, araştırmacıların karıncaların feromon izi bırakma davranışını incelemelerine olanak sağlamıştır.

Çift köprü düzeneği, karıncaların geri dönmek zorunda kalmadan arenayı kat etmelerini sağladığı için deney düzeneğinin önemli bir parçasıdır. Köprü, karıncaların arenayı keşfettikten sonra yuvaya geri dönmemelerini sağlamak için 60 derecelik açıyla iki daldan inşa edilmiştir. Köprünün mesafesi 15 cm ve her bir dalın genişliği 1 cm'dir. Deneyin tekrarlanabilir olmasını sağlamak için köprü beyaz kumla kaplanmış ve bu kum her deneyden önce yeni, kimyasal olarak işaretlenmemiş kumla değiştirilmiştir. Deney sırasında, hareketlerini takip etmek için köprünün her bir dalındaki karınca sayısı 3 dakikada bir sayılmıştır. Çift köprü kurulumu, karıncaların yolları kesişmeden arenayı keşfetmelerini sağlamakla kalmamakta, aynı zamanda bilim insanlarına karıncaların davranışlarını kontrollü bir ortamda gözlemleme imkanı vermektedir.



(a) Eşit büyüklükte kollara sahip çift köprü.	(b) Eşit olmayan boyutta kollara sahip çift köprü.

Şekil 2.3: Çift köprü deneylerinde kullanılan kurulumların şematik görünümü (Deneubourg ve ark., 1989, 1990; Goss ve ark., 1989).

Başlangıçta her iki dalın da eşit sayıda karınca tarafından seçildiği gözlemlenmiştir. Bu olgu, aynı sayıda karıncanın her iki yolu da seçtiği simetri olarak bilinmektedir. Ancak, her karınca sürekli olarak feromon izi bıraktığından ve feromon yoğunluğu seviyesi diğer karıncaları sağ veya sol dalı seçme kararlarında etkilediğinden, dallardaki feromon yoğunlukları arasındaki küçük bir fark bu simetrinin bozulmasına neden olabilmektedir. Bu pozitif geri besleme mekanizması, karıncaları dallardan birini diğerine tercih etmeye teşvik etmektedir, öyle ki karıncaların büyük çoğunluğu hızla aynı dalı seçmektedir.

Karıncaların ilk bakışta kaotik gibi görünen karar verme süreci aslında matematiksel bir olasılık fonksiyonu ile tanımlanabilmektedir. Bu olasılık fonksiyonu, feromon izlerinin yoğunluğuna ve belirli bir dalı seçen karınca sayısına dayanmaktadır. Bu teoriye göre, bir karıncanın belirli bir dalı seçme olasılığı, karıncaların o dal üzerinde bıraktığı feromon izinin yoğunluğu ile orantılıdır. Bu, belirli bir dalı seçen karınca sayısı ne kadar fazlaysa, bir sonraki karıncanın aynı dalı seçme olasılığının o kadar yüksek olacağı anlamına gelmektedir.

Bu nedenle, karıncaların karar verme sürecinin aşağıdaki olasılık fonksiyonu ile tanımlanabileceği hipotez ortaya atılmıştır:

$$P_A = \frac{(C+N_A)^a}{(C+N_A)^a+(C+N_B)^a} = 1 - P_B \tag{2.3}$$

Yukarıda açıklanan olasılık fonksiyonu feromon değerlerini doğrudan dikkate almamaktadır, bunun yerine bir dalın geçiş sayısının feromonlar tarafından yansıtıldığını varsaymaktadır. Bu makul bir varsayımdır, çünkü Arjantin karıncası feromonunun ortalama ömrü 30 dakikadır ve bu süre bir karıncanın köprüyü geçmesi için gereken süreden çok daha uzundur. Bu nedenle, deneylerde feromon buharlaşmasının etkisi ihmal edilmiştir. α ve C parametrelerini belirlemek için model üzerinde Monte Carlo simülasyonları kullanılmış ve sonuçta $\alpha \approx 2$ ve $C \approx 20$ bulunmuştur. Bu bulgu, sadece yerel ölçüleri dikkate alan basit bir olasılık kuralının, karıncaların yiyecek ararken otonom davranışının arkasındaki ana itici güç olduğuna dair ilk deneysel kanıttır.

Model ayrıca öngörülen geçiş sayısı ile gözlemlenen davranış karşılaştırılarak test edilmiştir. Modelin dallardan geçen karınca sayısını doğru bir şekilde tahmin edebildiği görülmüştür. Bu durum, karıncaların otonom davranışlarının altında yatan mekanizmanın basit bir olasılık kuralı olduğu fikrini daha da güçlendirmiştir.

Goss ve ark. (1989) tarafından yayınlanan ilk KKA algoritmasının kurulumu, karıncaların dallardan biri için tercihi olmayacak şekilde 60 derecelik bir açıyla boşluğun her iki ucuna bağlanan farklı uzunlukta iki dal içermekteydi. Birkaç santimetre sonra, dallardan biri daha büyük bir açıyla ilerleyerek diğerinden daha uzun hale gelmiştir. Köprünün ortasında, karıncalara yine iki dal sunulmaktadır, ancak bu defa daha kısa ve daha uzun dallar yer değiştirmiştir, böylece karıncaların sol veya sağ dal için olası tercihleri iptal edilmiştir. Başlangıçta karıncaların iki daldan birini seçme olasılığı eşitti. Ancak, kısa dalı seçen karıncalar yiyeceğe daha erken ulaşmış ve geri dönerken kısa dallarda uzun dallara göre daha güçlü bir feromon konsantrasyonu bırakmışlardır. Bu feromon konsantrasyonu, karıncaların tekrar aynı yolu seçmeleri için bir ipucu görevi görmüş ve böylece pozitif bir geri besleme döngüsü yaratmıştır. Uzun daldaki feromon konsantrasyonu da üzerinden geçen karıncalar nedeniyle artmıştır, ancak yine de kısa daldaki feromon konsantrasyonundan daha zayıftır.

Feromon konsantrasyonundaki bu fark, sonunda kısa dalın daha sık seçilmesine ve dolayısıyla uzun dalın daha az tercih edilmesine yol açmıştır.

Deneubourg ve ark. (1990)'da bir adım daha ileri giderek karıncaların iki boyutlu bir arenadaki keşif ve yiyecek arama davranışlarını, çift köprü deneyindeki iki daldan birini

seçmeye benzer bir dizi ikili karar problemiyle modellemişlerdir. Bu deney, yerel bilginin küresel kararları kolaylaştırmadaki etkinliğini göstermiş ve çift köprü deneyinin sonuçlarını doğru bir şekilde yeniden üretebilmiştir. Ayrıca, karıncalar daha uzun dallarda güçlü bir feromon izi oluşturduktan sonra bu izler tanıtıldığında karıncalar daha kısa dallara geçemediğinden, güçlü bir feromon izini tersine çevirmenin zorluğunu da vurgulamıştır. Bu davranış, Monte Carlo simülasyonları ile doğrulanmış ve sadece feromonlar biriktirildiğinde ve önemli bir buharlaşma olmadığında güçlü bir izin pratikte geri döndürülemez olduğunu göstermiştir. Böylece, Deneubourg ve ark.'nın deneyleri, yerel bilgi ve çok sayıda birey temelinde küresel karar problemlerini çözmek için otonom ilkelerin nasıl kullanılabileceğinin açık bir örneğini sunmaktadır. Karıncalar, bu tür ilkeleri kullanarak, merkezi bir karar alma otoritesi olmamasına rağmen çift köprü deneyinde en kısa yolu başarıyla bulabilmişlerdir.

Karıncalar arenayı keşfederken arkalarında feromon izleri bırakarak diğer karıncaların daha hızlı karar vermesini sağlayan bir tür otoyol sistemi oluşturmaktadırlar. Bir karınca biraz yiyecek bulduğunda, aynı izler boyunca yuvaya geri dönmekte ve başarılı keşiflerin gıda açısından zengin bölgelerde güçlü izlere yol açtığı bir dallanma modelinde onları daha da güçlendirmektedir. Keşiflerin büyümesine ve feromon izlerinin güçlenmesine katkıda bulunan ikinci bir faktör, yiyecekle dönen karıncalar tarafından yuvadan yeni işçilerin işe alınmasıdır.

Çoğu karınca türünde, yiyecek arayan karıncalar sadece yuvaya geri dönerken feromon bırakmaktadır veya yuvadan çıkarken çok daha az feromon bırakmaktadır. Ayrıca, bırakılan feromon miktarı bulunan yiyeceğin kalitesine veya miktarına bağlı olabilmektedir, böylece daha iyi yiyecek kaynaklarına giden yollar daha da güçlendirilmiş olmaktadır.

Karıncalar keşfettikçe, arkalarında bıraktıkları feromonlar, daha hızlı ve verimli karar vermelerini sağlayan bir yol sistemi oluşturmaktadır. Bir işe alma ve takviye sistemi sayesinde feromon izleri daha güçlü ve daha iyi hale gelmektedir ve karıncalar yiyecek kaynaklarına giden en kısa yolları bulabilmektedirler.

2.5. Karınca Kolonisi Algoritması (KKA) Metasezgiseli

KKA Metasezgiseli, kombinatoryal optimizasyon problemlerini ele almak için oluşturulmuştur (Dorigo ve Blum, 2005). Metasezgisel, birçok farklı sorun türü için

sezgisel yöntemler oluşturmak üzere kullanılabilen algoritmik kavramlar bütünüdür. Bu nedenle, KKA'nın tanımı oldukça geniştir.

2.5.1. Kombinatoriyal optimizasyon problemleri

Bir kombinatoryal optimizasyon problemi $P = \langle S, F \rangle$ çifti olarak ifade edilmektedir; burada S potansiyel çözümlerden ($s \in S$) oluşmakta ve $F : S \rightarrow (0, \infty)$ bu çözümlere kalitelerine göre pozitif reel değerler atayan bir fonksiyondur; daha yüksek değerler daha iyi çözümlere karşılık gelmektedir. Belirli bir çözüme atıfta bulunmak için s_i alt simge indeksi kullanılmaktadır. Problemin amacı, F'den en yüksek değeri üreten $s* \in S$ çözümünü bulmaktır. Eğer birden fazla çözüm F 'den maksimum sonucu veriyorsa, bunlar $S* \subseteq S$ kümesi olarak adlandırılmakta ve optimal kabul edilmektedir. Her çözüm, optimum bir şekilde birleştirilmesi gereken çeşitli bileşenlerden oluşmaktadır – bu nedenle kombinatoryal optimizasyon olarak adlandırılmaktadır. Bir çözümün bileşenlerini ayrı ayrı belirtmek için parantez içinde üst simge kullanılmaktadır: $s_i^{(j)}$ çözümünün j^{th} bileşenini belirtmektedir.

Bu tanım, Dorigo ve Blum (2005; Dorigo ve ark., 2006) tarafından kullanılan tanımdan, karmaşıklığı ve ACL ile gösterim çatışmalarını önlemek için basitleştirilmiş olması bakımından farklıdır. Onlar tarafından kullanılan tanımda, sözde uygulanabilir çözüm kümesine daha fazla kısıtlama getirilmekte ve bunların belirli koşullara uyması gerekmektedir.

2.5.2. KKA algoritması çerçevesi

KKA'da, kombinatoryal optimizasyon problemi, bir dizi köşe ve bunları birbirine bağlayan yaylardan oluşan yapı grafiği olarak bilinen bir grafikle temsil edilmektedir. Örneğin, Gezgin Satıcı Problemi (GSP)'de köşeler şehirleri, yaylar ise aralarındaki yolları temsil etmektedir. Dinamik sistemlerin kontrolü ile ilgili olarak köşeler ayrık durumları sembolize ederken, yaylar belirli bir durumdaki girdiye sistem tepkisini göstermektedir. Bir c karıncası tarafından bulunan belirli bir çözüm, çözüm bileşenlerinin sıralı bir kümesidir, $sc = (s_c^{(1)}, s_c^{(2)}, \ldots)$ - her bileşen ij indisi ile bağlanmış iki düğümden oluşmaktadır. Bu çözüm, karıncaların üzerinde gezinerek aşamalı olarak inşa ettiği düğüm grafiği üzerinde bir yolu temsil etmektedir.

Bir ij indisine iki değişken bağlıdır: ij feromon değişkeni ve ij sezgisel değişkenidir. Uç noktaların belirlenme şekli söz konusu hususa bağlıdır. Örneğin, GSP'de son köşe, karıncanın diğer tüm köşeleri tam olarak bir kez ziyaret ettikten sonraki başlangıç noktasına eşdeğerdir. Kontrol problemlerine uygulanması için, son tepe noktası sistemin istenen durumuna karşılık gelmektedir ve tüm karıncalar için benzerdir. GSP söz konusu olduğunda karıncaların tüm köşeleri yalnızca bir kez ziyaret ettiğinden emin olmak için, bir karınca çözümünün bileşenlerini özel tabu listesine de eklemektedir. Bir karıncanın bir sonraki adımda hangi köşeye gideceğine karar vermesi gerektiğinde, tabu listesinde olmayan herhangi bir köşeyi seçebilmektedir. Bu karınca tarafından belirlenen aday çözüm, kısmi bir çözüme dayanmaktadır.

2.5.3. Gerçek karıncaların ilişkisi

KKA algoritmalarındaki yapay karıncalar, genellikle karınca olarak adlandırılmaktadırlar, gerçek olan benzerleriyle bazı benzerlik ve farklılıkları bulunmaktadır. Özellikle, karar vermek için benzer bir olasılık kuralını (2.2) takip etmektedirler, ancak gerçek karıncalarda görülmeyen, ziyaret ettikleri konumların bir hafızasına sahiptirler. Ayrıca, feromon bırakma söz konusu olduğunda, KKA karıncaları, tıpkı bazı gerçek karınca türlerinin yaptığı gibi, feromonları yalnızca hedeflerine (terminal tepe noktası) ulaştıktan sonra bırakmaktadırlar; buna ek olarak, bırakılan feromon miktarı, çözümlerinin (besin kaynağı) kalitesiyle orantılıdır. KKA algoritmaları ile gerçek karınca davranışları arasındaki bu benzerlik ve farklılıklar göz önünde bulundurularak, birbirleriyle nasıl bir ilişki içinde oldukları daha iyi anlaşılabilmektedir.

2.6. KKA Algoritmaları

Çok çeşitli KKA algoritmaları geliştirilmiştir ve bunların çoğu, belirli senaryo türleri için onları daha etkili kılan yalnızca küçük farklılıklar içermektedir. Bu bölümde en popüler ve yaygın olarak kullanılan iki KKA algoritması olan karınca sistemleri ve karınca kolonisi sistemleri incelenecektir.

2.6.1 Ant system (karınca sistemleri)

En temel KKA algoritması olan Karınca Sistemleri (Dorigo ve ark., 1996) şu şekilde çalışmaktadır: M karıncalar, inşaat grafiğinin köşelerine rastgele dağıtılmaktadır.

Problemle ilgili ön bilgileri yansıtmak üzere, bazı köşelere diğerlerinden daha fazla öncelik vermek için sezgisel değişkenler ηij oluşturulabilmektedir. Her bir c karıncası için sp,c kısmi çözümü boş olarak başlatılmakta ve tüm feromon değişkenleri τ0 > 0 gibi küçük bir başlangıç değerinden başlamaktadır. Her adımda, her karınca belirli bir olasılık dağılımına dayanarak, sp,c kısmi çözümüne hangi çözüm bileşenini (i, j) dahil edeceğini belirlemektedir. İ köşesindeki bir c karıncasının Ni,c uygulanabilir komşuluğundaki j köşesine hareket etme olasılığı pc{j|i} olarak formüle edilmektedir:

$$P_{i,J}^{l} = \frac{[T_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum_{l\in N_i^l}[T_{il}]^{\alpha}[\eta_{il}]^{\beta}} \tag{2.4}$$

ηij ve τij'nin göreceli önemi α ≥ 1 ve β ≥ 1 ile belirlenmektedir. C karıncası için bir i köşesindeki Ni,c uygulanabilir komşuluğu, problemin yapısı tarafından belirlenen i'ye bağlı olan ve henüz ziyaret etmediği vertekslerden oluşmaktadır. Karınca c, i köşesinden j köşesine geçiş yaparak ilgili çözüm bileşenini (i, j) terminal noktasına ulaşana kadar sp,c kısmi çözümüne eklemekte ve sp,c 'yi sc olarak depolayarak aday çözümünü tamamlamaktadır. Bu süreç bir deneme olarak adlandırılmaktadır. Tüm karıncalar denemelerini tamamladıktan sonra AS, her bir karıncanın aday çözümlerini bir uygunluk fonksiyonu F (s) kullanarak değerlendirmekte ve bu değer daha sonra feromon seviyelerini güncellemek için kullanılmaktadır.

Global feromon güncelleme adımı, karıncaların feromon izi bırakarak yuvalarına döndükleri adımdır. Bu, KKS ile görülen yerel feromon güncelleme adımının tersidir. KKA'daki tüm karıncalar, gerçek karıncalardan farklı olarak birlikte "geri dönmeden" önce birbirlerini "bekle"mektedirler. Feromon birikintisi Δτij (s) miktarı, buharlaşma oranı olan ρ ∈ (0, 1) ve feromonların güncellenmesi için uygun çözümlerden oluşanlar dikkate alınarak hesaplanmaktadır:

Feromon seviyeleri, ilgili çözüm parçasının kısmi çözüme dahil edilmesinin ne kadar faydalı olduğunu göstermektedir. Feromonların buharlaştırılmasıyla algoritmanın zamanından önce optimal olmayan çözümlere ulaşması engellenebilmektedir. (2.3)'te tüm feromon seviyeleri buharlaştırılmakta ve yalnızca güncelleme kümesindeki çözümlerle ilişkili köşeler feromon depozitosu almaktadır. Bir sonraki seferde, hangi köşeye doğru hareket edileceği konusunda daha bilinçli kararlar vermek için kullanılabilecek güncellenmiş feromon seviyeleri ile her karınca bu adımları tekrarlamaktadır. Bir durma noktasına ulaşıldığında- örneğin belirli sayıda deneme

sayısını geçtiğinde- grafik üzerindeki τij ve ηij değerleri bize tüm (i, j) çiftleri için bir çözüm vermektedir. Bu nihai çözümü, i köşesinden en uygun j köşesini bulma açısından elde etmek için, söz konusu grafikten aşağıdaki şekilde çıkarılabilmektedir:

$$J = arg\,max\left(T_{ij}^{a} n_{ij}^{\beta}\right) \tag{2.7}$$

Çarpımının belirli bir i için aynı maksimum değere ve farklı j değerlerine sahip olması durumunda, beraberlikler rastgele bozulmaktadır. Supd'yi oluşturmak için birçok kural bulunmaktadır, ancak en yaygın olanı Strial3 denemesinde bulunan tüm çözümleri kullanmaktır. Genellikle AS güncelleme kuralı kullanılmaktadır ancak diğer güncelleme kurallarının belirli kombinatoryal optimizasyon problemlerinde daha etkili olduğu gösterilmiştir. Son denemedeki tüm çözümleri kullanmak yerine, sadece en iyi çözüm ya da sadece başlangıçtan bu yana en iyi çözüm kullanılabilmektedir. Bu yöntemler sırasıyla iterasyon-en iyi veya deneme-en iyi ve şimdiye kadarki en iyi veya global-en iyi olarak adlandırılmaktadır (Dorigo ve Blum, 2005). Bu yaklaşımlar feromon izlerini daha önce başarısı kanıtlanmış çözümlere doğru yönlendirirken algoritmaları çalıştırmak için hesaplama gereksinimlerini de azaltmaktadır.

Elitist Karınca Sistemi: Elitist Karınca Sistemi'ndeki en önemli özelliklerden bir tanesi uzman karıncaların varlığıdır. Bu karıncalar, normal karıncaların tipik rollerinden farklı işlevleri yerine getirmek üzere tasarlanmıştır. Örneğin, karıncalar bir çözüm bulduklarında, genellikle yol boyunca bir feromon izi bırakmaktadırlar. Ancak Elitist Karınca Sistemi ile en iyi çözüme, uzman karınca sayısı ile çarpılacak bir feromon bonusu verilmektedir (Strauß ve ark., 2007). Bu, en başarılı yolların diğerlerinden daha yüksek bir feromon konsantrasyonuna sahip olacağı ve daha uzun süre alakalı kalacağı anlamına gelmektedir. Geleneksel yollar hızla kullanılmaz hale gelirken, elitist yollar uygulanabilir seçenekler olarak kalacaktır. Bunun nedeni, tek bir iyi çözümün aramanın mevcut problem alanından sapmasına neden olabilmesi ve böylece durgunluğu önleyebilmesidir.

Rank Temelli Karınca Sistemi: Rank Temelli Karınca Sistemi, uzman karıncaların en iyi yolları işaretlemek için feromonları kullandığı bir tür KKA algoritmasıdır. Yollar uzunluklarına göre sıralanmakta ve en iyi sıradaki yol en fazla feromonu, en kötü yol ise en az feromonu almaktadır. Bu, feromonları sadece bulunan en iyi yola veren diğer optimizasyon algoritmalarından farklıdır. Bununla birlikte, Rank Temelli Karınca

Sisteminin farklı bir yönü daha bulunmaktadır. Eğer normal karıncalardan daha az uzman karınca varsa, en kötü sıradaki yollar hiç feromon alamayabilmektedir. Örneğin, üç uzman karınca ve on normal karınca bulunuyorsa, on normal karıncanın her biri kendi yollarını bulurken, uzmanlar bu yolları sıralayacak ve feromon vermek için en iyi üç yolu seçecektir. Bu, en kötü sıralanmış yolların bile dikkate alınmasını sağlamaktadır ve bir veya iki kötü yolun gözden kaçabileceği durumlarda faydalı olabilmektedir.

Min-Maks Karınca Sistemi: Bu sistemde uzman karıncalar bulunmamakta, bunun yerine sadece normal karıncaların kullanıldığı geleneksel sisteme geri dönülmektedir. Ancak Min-Maks Karınca Sistemi'nde temel karınca sistemine kıyasla en dikkat çekici değişiklik feromon seviyelerinin ele alınış biçimidir. Bu sistem feromon değerleri için minimum ve maksimum değerler getirmektedir. Bu, bir yoldaki feromon değerinin maksimum değeri aşmasına izin verilmemesini ve asla yolun ilgisiz hale geleceği kadar düşmemesini sağlamaktadır. Ayrıca, bir yolun çok doygun hale gelmesini ve diğer tüm yolları gölgede bırakmasını önlemektedir. Ek bir yumuşatma mekaniği de uygulanmaktadır. Bir veya daha fazla yol maksimum değere yakın olduğunda ve diğer tüm yollar minimum değere yakın olduğunda, feromon seviyeleri yumuşatılmaktadır (Hoos ve Stützle, 2000). Bu, düşük feromon seviyelerine sahip yolları büyük ölçüde teşvik etme etkisine sahipken, yollar arasındaki sıralamaları da korumaktadır. Ayrıca feromonlar sadece en iyi yola uygulanırken diğer tüm yollar göz ardı edilmektedir. Temel karınca sistemine eklenen bu özellikler, optimum çözümü verimli bir şekilde bulabilen güçlü ve rekabetçi bir algoritma oluşturmaktadır.

2.6.2 Karınca kolonisi sistemleri

KKS (Dorigo ve Gambardella, 1997) KS'nin bir ilerlemesidir ve en popüler ve yaygın olarak kullanılan KKA algoritmalarından biridir. KS ve KKS arasında birkaç önemli fark bulunmaktadır. İlk olarak, global feromon güncelleme adımında küresel en iyi güncelleme kuralını kullanmaktadır, yani deneme sonunda feromon değişkenlerini değiştirmek için yalnızca algoritma başlangıcından bu yana en yüksek (yani önceki çözümlerden daha yüksek) uygunluğa sahip olan çözüm – sgb olarak bilinmektedir - kullanılmaktadır. Bu, KKA algoritmalarında optimal sonuçlara yakınsamayı önemli ölçüde hızlandırdığı gözlemlenen bir elitizm biçimi olarak düşünülebilmektedir. İkinci önemli fark, global feromon güncellemesi için yalnızca sgb'de bulunan (i, j) çiftlerinin hedeflenmesidir; sonuç olarak KS'de olduğu gibi tüm feromon seviyeleri değil, yalnızca

ρ ile ağırlıklandırılmış bir depozito alanlar buharlaştırılmaktadır. Bu üç değişikliğin bir sonucu olarak, global feromon güncelleme kuralı şu hale gelmektedir:

$$\tau_{ij} \leftarrow \begin{cases} (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}(s_{\mathrm{gb}}), & \text{if } (i,j) \in s_{\mathrm{gb}} \\ \tau_{ij}, & \text{otherwise.} \end{cases} \tag{2.8}$$

KKS algoritması, yetersiz çözümlere yakınsamayı önlemeye yardımcı olan yerel bir feromon güncelleme adımı içermektedir. Bu adım, her karınca bir deneme içinde hareket ettikten sonra gerçekleşimekte ve şu şekilde ifade edilmektedir:

$$\tau_{ij} \leftarrow (1-\gamma)\tau_{ij} + \gamma\tau_0, \tag{2.9}$$

Yerel feromon güncelleme oranı γ ∈ (0, 1) global feromon güncelleme kuralına (ρ) benzemektedir. İndeks ij kısmi çözüme yeni eklenen çifte karşılık gelmektedir ve τ0 izlerin başlangıç değeridir. Bu, halihazırda ziyaret edilen bileşenlerin diğer karıncalar için daha az çekici hale geldiği ve böylece onları daha az geçilen seçenekleri keşfetmeye teşvik ettiği anlamına gelmektedir.

2.7. Applications of aco (kka uygulamaları)

Temel KKA algoritması olan Karınca Sistemleri ve türevleri, gezgin satıcı problemi (Dorigo ve Stützle, 2004), yük dengeleme (Sim ve Sun, 2003), atölye çizelgeleme (Huang ve Yang, 2008; Alaykran ve ark., 2007), mobil robotlar için optimum yol planlama (Fan ve ark., 2003) ve telekomünikasyon ağlarında rotalama (Wang ve ark., 2009) gibi çeşitli optimizasyon problemlerinde yaygın olarak uygulanmıştır. Ayrıca, Purnamadjaja ve Russell (2005) tarafından gerçek robotik sistemler için KKA feromon izleri konseptinin bir uygulaması önerilmiştir. KKA'nın endüstriyel uygulamalarına ilişkin bir anket Fox ve ark. (2007) tarafından sunulmuştur. Schoonderwoerd ve ark. (1996), çağrıların düğümlerdeki feromon dağılımına göre yönlendirildiği telekomünikasyon ağlarında yük dengeleme sağlamak için bir yöntem sunan karınca tabanlı kontrolün erken bir çalışmadır (Bianchi ve ark., 2006).

Karınca Koloni Sistemi (KKS), sinir ağlarının performansını optimize etmek için de kullanılmış ve dinamik ortamlardaki optimizasyon problemlerini çözmek için genişletilmiştir. Ayrıca, Max-Min Karınca Sistemi (MMAS), NP-zor kombinatoryal optimizasyon problemlerine optimal çözümler bulmak için kullanılan KKS'nin bir

çeşididir. Son olarak, Karınca Programlama (AP), ikinci dereceden atama problemi, minimum yayılan ağaç problemi ve sırt çantası problemi dahil olmak üzere çeşitli optimizasyon problemlerine uygulanan ACO'nun bir başka çeşididir.

Sürekli optimizasyon için karınca metaforlarının kullanımı yeni bir kavram değildir ve birçok çalışmada incelenmiştir. Örneğin, Bilchev ve Parmee (1995) karınca metaforunun sürekli optimizasyon problemine erken bir uygulamasını sunmuştur. Daha yakın zamanda, Tsutsui ve ark. (2005) karınca metaforunu sürekli optimizasyon problemine uygulayan Toplama Feromonları Sistemini geliştirmiştir. Bir başka örnek de Korosec ve ark. (2007) tarafından önerilen Diferansiyel Karınca-Stigmerji Algoritmasıdır. Bu algoritmalar örüntü sınıflandırma ve sinir ağları gibi çeşitli alanlara uygulanmıştır. Aslında, KKA'nın sürekli versiyonunun gradyan tabanlı sinir ağı eğitim algoritmalarıyla karşılaştırılabilir performans elde ettiği gösterilmiştir.

2.8. Değelendirme

Sürüdeki tüm bireylerin kolektif davranışı bireyler için faydalar sağlamaktadır, öyle ki bireyler küresel davranışın farkındaymış gibi hareket ediyor gibi görünmektedir. Bu otonom olgusu, otonom ajanların merkezi olmayan kontrolünün, tek başına bir birey için imkansız olan görevleri tamamlayabilen kolektif davranışla sonuçlanabileceği mühendislik bilimlerinde uygulanabilmektedir. Ayrıca, bireylerin arızalanması veya ortadan kaldırılmasına karşı sağlamlık, ölçeklenebilirlik, esneklik ve büyük miktarlarda benzer birimlerin üretilmesiyle ilişkili maliyet avantajları gibi mühendislik sürülerinin doğasında bulunan özellikler, büyük bir robot yerine küçük robotlardan oluşan bir sürü tasarlamanın ek faydalarını sağlamaktadır.

Optimizasyon alanında, Parçacık Sürü Optimizasyonu ve KKA'nın sürü zekası algoritmaları, yirminci yüzyılın başlarında tanıtılmalarından bu yana çok başarılı olmuştur. Bu algoritmalar Gezgin Satıcı Problemi, Sırt Çantası Problemi ve Tesis Yerleşim Problemi gibi çeşitli optimizasyon problemlerini çözmek için kullanılmaktadır. Optimizasyon problemlerini çözmek için güçlü ve verimli bir araç oldukları ve genellikle optimum çözüme yakın çözümler sağlayabildikleri görülmüştür. Buna ek olarak, sürü zekası algoritmaları çok sayıda parametre ve değişkenle başa çıkabildikleri için genellikle çok sayıda serbestlik derecesine sahip zor problemleri çözmek için kullanılmaktadır. Bu nedenle, sürü zekası algoritmaları çok çeşitli

uygulama alanlarında optimizasyon problemlerini çözmek için popüler bir araç haline gelmiştir. Bazı karınca türlerinin yiyecek arama davranışları uzun yıllardır incelenmektedir ve karıncaların feromon adı verilen özel bir kimyasal bırakarak yiyecek kaynaklarına giden yolların kalitesini bildirdikleri bilinmektedir. Bu olgu, karıncaların karar sürecinin matematiksel bir modelinin geliştirilmesiyle sonuçlanan çift köprü deneyi aracılığıyla araştırılmıştır. Bu model daha sonra, karmaşık kombinatoryal optimizasyon problemlerini çözmek için çok popüler hale gelen Karınca Kolonisi Algoritması (KKA) çerçevesi için temel olarak kullanılmıştır.

KKA çerçevesini kullanmanın, basit sezgisel yöntemler kullanarak hızlı ve verimli bir şekilde optimuma yakın çözümler bulma yeteneği gibi avantajları bulunmaktadır. Ayrıca, KKA çerçevesinin robotikten veri madenciliğine ve yöneylem araştırmasına kadar çeşitli uygulamalarda kullanım şekilleri de görülmektedir.

BÖLÜM 3.GELİŞTİRİLEN KARINCA KOLONİSİ ALGORİTMASI VE İNŞAAT PROJESİNE UYGULANMASI

Kritik Yol Yöntemi (CPM) ve Program Değerlendirme ve Gözden Geçirme Tekniği (PERT) ağları kullanılarak proje yönetimi için ağ analizi gerçekleştirilebilmektedir. Özellikle ağın boyutu çok büyük olduğunda ağ analizinde zorluklar meydana gelebilmektedir. Ağ analizi problemlerinin çözümünde KKA kullanılabilmektedir. Bu tür karmaşık problemler sezgisel yöntemlerle çözülebilmektedir. Sezgisel bir yaklaşım olan KKA'nın uygulanması üzerine araştırmalar yapılmaya devam etmektedir ve bu strateji için uygun olan yeni algoritmalar da ortaya çıkmaktadır. Bu algoritmalar, proje yönetimi için ağ analizinde karşılaşılan problemlere potansiyel bir çözüm olarak geliştirilmiş ve önerilmiştir. Ayrıca, bu algoritmalar büyük ağlar için mümkün olan en iyi sonuçları sağlamak ve ağ analizi sürecinin karmaşıklığını yönetmek için tasarlanmıştır.

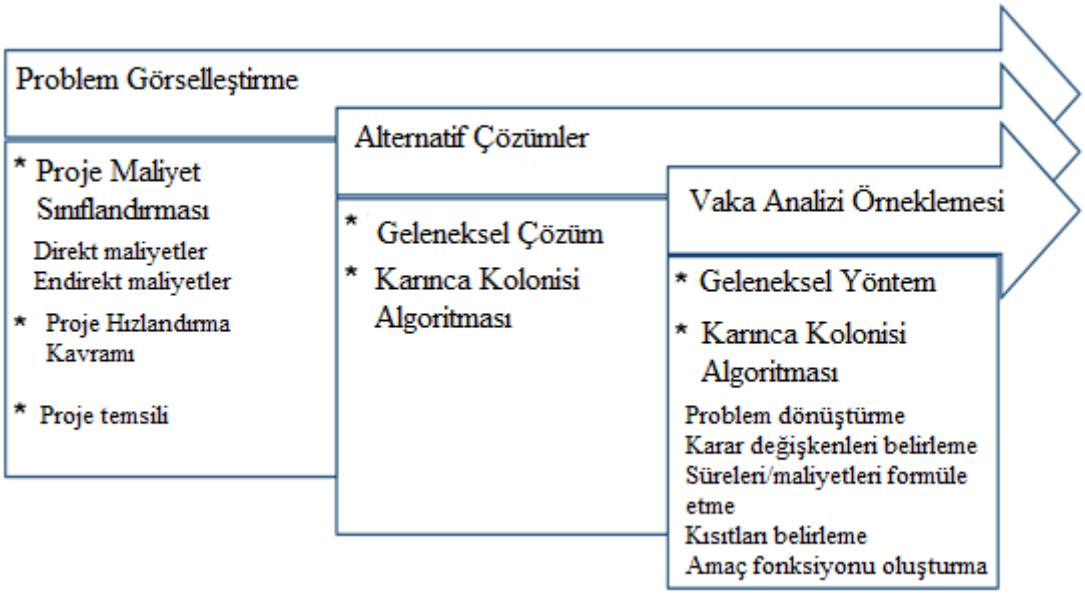
3.1. Karınca Kolonisi Algoritması

Proje yönetimi kapsamında CPM ve PERT ağları üzerinde KKA ile optimizasyon sağlamak için MATLAB programında algoritma yazılmıştır. Tasarlanan algoritmanın kodlanması için MATLAB programı kullanılmıştır. Tüm simülasyonlar 2.90 GHz işlemciye sahip bilgisayarda gerçekleştirilmiştir. Algoritma tüm CPM ve PERT ağlara bakmak yerine bazı örnek çözümlere bakmaya göre tasarlandığından, uygulama öncelikle ağlarda kullanılan düğümleri ve daha sonra söz konusu düğümlerin öncül-sonraki durumlarını belirlemektedir. Aktivitelerin operasyon süreleri bir tabloya aktarılmaktadır. CPM ağları için, kullanıcı ağda kullanmak üzere operasyon sürelerini eklemektedir. Operasyon süreleri eklendikten sonra, Karınca Kolonisi Algoritması bulunabilen en iyi çözümü üretmeye çalışmaktadır. Bulunan en iyi çözüm, algoritma tarafından üretilen en iyi çözümdür ve projeyle ilişkili süreyi ve maliyeti azaltmak için kullanılabilmektedir.

3.2. Yöntem

Şekil 3.1'de sunulan metodoloji akış şeması, proje hızlandırma problemlerini tanımlamayı ve çözmeyi sağlayan üç ana aşamadan oluşmaktadır. İlk aşama, proje hızlandırma problemlerinin anlaşılması kolay bir şekilde görselleştirilmesini sağlayan problem gösterimini içermektedir. İkinci aşamada, bu problemleri ele almak için kullanılabilecek farklı yaklaşımların ana hatları çizilmektedir. Bu, mevcut kaynakların ve uygulanabilecek stratejilerin belirlenmesini içermektedir. Son olarak, üçüncü aşamada, ikinci aşamada belirlenen yaklaşımların gerçek dünyada nasıl uygulanabileceğini göstermek için bir vaka çalışması örneği sunulmaktadır. Bu, problemin karmaşıklığını ve nasıl ele alınacağını daha iyi anlamaya yardımcı olmaktadır. Akış şemasında özetlenen adımlar takip edilerek, proje hızlandırma problemlerinin etkili bir şekilde belirlenmesi ve çözülmesi sağlanabilmektedir.

Projenin ilk aşaması, proje hızlandırması kavramının kapsamlı bir şekilde anlaşılmasını sağlamaya odaklanmaktadır. Bu, proje maliyetlerinin kategorizasyonunun netleştirilmesini, proje hızlandırması kavramının etkili bir şekilde açıklanmasından oluşmaktadır. İkinci aşama daha derinlere inmekte ve proje hızlandırma problemleri için hem geleneksel hem de önerilen karınca kolonisi algoritma yaklaşımları için izlenen adımları açıklamaktadır. Burada, geleneksel çözüm tanıtılmakta ve ana hatlarıyla açıklanmaktadır. Ayrıca, karınca kolonisi algoritmanın benimsenmesi için gerekenler de ayrıntılı bir şekilde ifade edilmektedir. Üçüncü ve son aşama, bu iki farklı yaklaşım tarafından izlenen adımların bir vaka çalışması üzerinde gösterilmesidir. Bu, iki yaklaşım arasındaki farkları ve karınca kolonisi algoritmanın proje hızlandırma problemlerine verimli ve etkili bir çözüm sağlamak için nasıl kullanılabileceğini göstermek için bir örnek teşkil etmektedir.



Şekil 3.1: Metodolojinin akış şeması

3.3. Proje Hızlandırma Probleminin Gösterimi

Doğrudan maliyetler, ürünün üretimiyle doğrudan ilgili maliyetlerdir. Bu maliyetlere malzeme, işçilik ve ürünün üretimiyle ilgili diğer maliyetler dahildir. Dolaylı maliyetler, ürünle doğrudan ilgili olmayan, ancak üretim süreciyle ilişkili olan maliyetlerdir. Dolaylı maliyetlere örnek olarak kira, kamu hizmetleri ve bakım maliyetleri verilebilmektedir. Bu maliyetler, ürünün verimli ve uygun maliyetli bir şekilde üretilmesini sağlamak için gerekmektedir. Ayrıca bu maliyetler, pazarlama ve reklam maliyetlerinin yanı sıra üretim süreciyle ilişkili diğer maliyetleri de ifade edilmektedir. Projenin toplam maliyeti belirlenirken tüm bu maliyetler göz önünde bulundurulmalıdır. Proje hızlandırma problemlerinde, her bir aktivitenin bireysel maliyeti bu kategori altında değerlendirilmektedir.

Doğrudan maliyetler, işçilik ve malzeme gibi projenin oluşturulmasında açıkça ortaya çıkan maliyetlerdir. Ancak dolaylı maliyetler, ürünün üretilmesinde gerçekleştirilen harcamaların da fazlasını ifade etmektedir. Bunlar, ofis malzemeleri ve bina kiralama ücretleri gibi genel yönetim giderlerinin yanı sıra proje süresince ortaya çıkan ek maliyetleri de içerebilmektedir. Proje hızlandırma problemlerinde, bu dolaylı maliyetlerin proje süresiyle doğrusal olarak ilişkili olduğu, yani proje süresi arttıkça orantılı olarak arttığı varsayılmaktadır. Bu, bir projenin tamamlanması ne kadar uzun sürerse, dolaylı maliyetlerin de o kadar yüksek olacağı anlamına gelmektedir.

Bu nedenle, proje yöneticilerinin projelerinin maliyetlerini minimumda tutmak için doğrudan ve dolaylı maliyetlerini doğru bir şekilde hesaplamaları ve tahmin etmeleri çok önemlidir.

Proje hızlandırma, proje maliyetini minimumda tutmayı hedeflerken bir projenin daha kısa sürede tamamlanmasını sağlayan, proje yönetiminde yaygın olarak kullanılan bir tekniktir. Bu, ek personel atamak veya prim ödemek gibi kritik yol üzerindeki belirli aktivitelere daha fazla kaynak sağlayarak gerçekleştirilebilmektedir. Ek kaynaklar nedeniyle projenin doğrudan maliyeti artsa da, projenin süresiyle doğrudan ilişkili olduğu için dolaylı maliyet azalmaktadır.

Proje maliyetini en aza indirmek ve projenin optimum süresini ve maliyetini elde etmek için yaygın olarak iki yaklaşım kullanılmaktadır. Geleneksel yaklaşım ve önerilen karınca kolonisi algoritma yaklaşımı en popüler iki yaklaşımdır ve her ikisi de istenen sonucu hesaplamak için farklı adımlar içermektedir. Geleneksel yaklaşım, aktivitelerin analiz edilmesini ve projenin minimum maliyetinin ve süresinin hesaplanmasını içerirken, karınca kolonisi algoritma yaklaşımı optimum proje maliyetini ve süresini belirlemek için karınca kolonisi algoritmaları kullanmaktadır.

Genel bir proje hızlandırma problemi gösterimi Tablo 3.1'de gösterilmiştir. İlk iki sütun aktivitelerin listesini ve aralarındaki ilişkileri göstermektedir. Aktivitelerin genellikle yukarıdan aşağıya doğru düzenlendiği, başlangıç ve bitiş aktivitelerinin sırasıyla üstte ve altta verildiği görülebilmektedir. Üçüncü sütun her bir aktivitenin normal süresini (hızlandırma meydana gelmediği süre) göstermektedir. Bu normal süre, süreyi kısaltmak için herhangi bir girişimde bulunulmadığında aktiviteyi tamamlamak için gereken tahmini süredir. Dördüncü sütun olan ezilmiş süre, tüm olası hızlandırmalar uygulandıktan sonra her bir aktivitenin süresini ifade etmektedir. Dolayısıyla, normal ve ezilmiş süre arasındaki fark, ilgili aktivite için geçerli olan ezme sayısını ifade etmektedir. Örnek vermek gerekirse, fark ikiye eşitse, aktivite iki kereden fazla hızlandırılamamaktadır (süre iki günden fazla azaltılamamaktadır). Beşinci ve altıncı sütunlar sırasıyla normal maliyetler ve her bir çakışmanın maliyetidir. Her bir çakışmanın maliyeti, sürenin tek bir gün azaltılması için ödenmesi gereken para miktarıdır. Projenin doğrudan maliyeti, normal maliyetler ile gerçekleşen çakışmaların maliyetlerinin toplamına eşittir. Son olarak, yedinci sütun, genel maliyetin CPM ile belirlenen proje süresi ile çarpılmasıyla elde edilen dolaylı maliyettir.

Tablo 3.1: Tipik bir proje hızlandırma probleminin gösterimi

Aktivite	Öncelik	Normal Zaman(gün)	Normal Maliyet(TL)	1.hızlandırma maliyeti	3.hızlandırma maliyeti	3.hızlandırma maliyeti
A	...	...	...	...	...	...
B	...	...	...	...	...	...
C	...	...	...	...	...	...
...	...	...	...	...	...	...

*Genel Gider: … TL/gün

3.4. Proje Hızlandırma Problemleri İçin Alternatif Yaklaşımlar

Projenin hızlandırmasına yönelik geleneksel yaklaşım birkaç adıma ayrılabilmektedir. İlk olarak, tüm aktivitelerin normal sürelerinde/maliyetlerinde tamamlanacağı varsayılmakta ve proje süresi/maliyeti buna göre hesaplanmaktadır. Bunu, proje süresini hesaplamak ve kritik yolları belirlemek için Kritik Yol Yöntemi (CPM) kullanımı takip etmektedir. Kritik yol belirlendikten sonra, bir sonraki adım proje süresini bir birim azaltarak dolaylı maliyetlerde tasarruf elde etmektir. Bu önemli bir adımdır çünkü proje süresini önemli ölçüde kısaltarak teslim tarihlerine uyulmasına ve genel maliyetlerin düşürülmesine yardımcı olabilmektedir. Bu adımdan sonra, geleneksel yaklaşım, hangi aktivitelerin en fazla hızlandırma potansiyeline sahip olduğunu belirlemek için kritik yol üzerindeki aktiviteleri analiz etme adımına geçmektedir. Proje yöneticisi her bir aktiviteyi analiz ederek hangi aktivitelerin ne kadar aksayacağına karar verebilmektedir. Son olarak, proje hızlandırma süreci tamamlanmakta ve proje süresi/maliyeti azaltılmaktadır. Bu nedenle, minimum aktivite zamanına sahip kombinasyonu, bu aktivitelerin hızlandırılmasını belirtilen koşulu karşılayabileceği şekilde belirlenmektedir. Proje süresinin bir birim kısaltılması, tüm kritik yolların hızlandırılmasını gerektirmektedir. Bir aktivitenin hızlandırılması ile bir kritik yolun hızlandırılması arasındaki farkın da açıklanası gerekmektedir. Birincisi aktivite süresinin bir birim azaltılması anlamına gelirken, ikincisi ilgili yoldaki aktivitelerden en az birinin hızlandırılması anlamına gelmektedir. Seçilen aktiviteleri durdurmanın maliyeti, dolaylı maliyetlerden elde edilen tasarrufla karşılaştırılmaktadır. Tasarruf, hızlandırma maliyetini aşarsa; seçilen aktiviteler hızlandırılmaktadır, yeni proje süresi ve maliyeti hesaplanmakta ve süreç tekrarlanmaktadır. Bu süreç, kritik yollardaki tüm aktiviteler hızlandırılana ve en kısa proje süresine ulaşılana kadar tekrarlanmaktadır. Buna ek olarak, aktivitelerin hızlandırılmasının maliyeti, malzeme maliyetleri ve işçilik

maliyetleri gibi projeyle ilişkili dolaylı maliyetlerden elde edilen tasarrufla karşılaştırılmaktadır. Tasarruf, hızlandırma maliyetinden büyükse, optimum proje süresi ve maliyet kombinasyonu elde edilene kadar hızlandırma işlemi tekrarlanmaktadır. Aksi takdirde, mevcut proje süresi ve maliyeti optimum olarak kabul edilmektedir. Aktiviteleri hızlandırdıktan ve optimum proje süresi ve maliyetine ulaştıktan sonra, proje yöneticisi, projenin kalitesinden ve güvenliğinden ödün vermeden proje hedeflerinin karşılandığından emin olmalıdır.

Önerilen karınca kolonisi algoritma yaklaşımı, geleneksel problem çözme yöntemlerinden tamamen farklı bir süreçtir. Önerilen yaklaşımın düzgün bir şekilde uygulanabilmesi için öncelikle Tablo 3.2'de görüldüğü gibi problemin dönüştürülmesi gerekmektedir. Hızlandırmaların maliyetine odaklanmak yerine, çeşitli aktivitelerin süresi ve maliyeti dikkate alınmaktadır. Bu dönüşüm, ilerlemek ve karınca kolonisi algoritma yaklaşımını uygulamak için gerekli bir adımdır. Problem uygun şekilde değiştirildikten sonra, algoritma aşağıdaki adımlarda probleme uygulanabilmektedir. Bu adımlar, problemi haritalandırmak ve bir çözüm bulmak için algoritmayı kullanmak gibi karmaşık ve girift bir süreci içermektedir. Sadece problem uygun şekilde dönüştürüldüğünde ve algoritma uygulandığında bir çözüm bulunabilmektedir. Bir karar değişkeni, bir aktivitenin hızlandırma derecesinin sayısal bir temsilidir ve her aktivite için tanımlanmaktadır. Hızlandırma derecesi, ilgili aktivitenin hızlandırma sayısı ile gösterilmektedir. Karar değişkenleri belirlendikten sonra, her bir aktivitenin süresi ve maliyeti karar değişkeninin bir fonksiyonu olarak ifade edilebilmektedir. Bu, her bir aktivitenin kaç kez hızlandırılacağının karar değişkenleri için kısıt olarak kullanılmasına olanak tanımaktadır. Amacı maliyeti en aza indirmektir; bu da proje maliyetinin karar değişkenlerinin bir fonksiyonu olacak şekilde formüle edilmesiyle elde edilmektedir. Bu nedenle optimizasyon süreci, karar değişkenlerinin kısıtları kullanılarak proje maliyetinin minimize edilmesiyle yürütülebilmektedir.

Tablo 3.2: Problemin dönüştürülmüş versiyonunun gösterimi

Aktivite	Öncelik	Normal Zaman Süre/ Maliyet (Gün/TL)	1.hızlandırma Süre/ Maliyet(Gün/ TL)	2.hızlandırma Süre/ Maliyet(Gün/ TL)	2.hızlandırma Süre/ Maliyet(Gün/ TL)
A	...	.../...	.../...	.../...	.../...
B	...	.../...	.../...	.../...	.../...
C	...	.../...	.../...	.../...	.../...
...	...	.../...	.../...	.../...	.../...

*Genel Gider: … TL/gün

3.5. İnşaat Projesinin Gösterimi

Proje hızlandırma problemleri için iki farklı yaklaşımın izlediği adımlar bir inşaat projesi kullanılarak gösterilebilmektedir. Toplam 12 aktivite içeren böyle bir problem örneği Tablo 3.3'te sunulmuştur. Aktiviteler arasındaki ilişkiler ikinci sütunda verilmiştir ve sonlandırma-başlatma türündedir. Bu, herhangi bir aktivitenin ancak bir önceki aktivite tamamlandıktan sonra başlatılabileceği anlamına gelmektedir. Her bir aktivitenin normal ve hızlandırma süreleri sırasıyla üçüncü ve dördüncü sütunlarda gösterilmektedir. Örneğin, Aktivite 1'in normal süresi 8 gün, hızlandırma süresi ise 5 gündür. Bu, aktivitenin 3 gün boyunca (8 gün - 5 gün) hızlandırma potansiyeline sahip olduğu anlamına gelmektedir. Beşinci sütun aktivitenin normal süresinde tamamlanmasının maliyetini gösterirken, son üç sütun hızlandırma maliyetini göstermektedir. Hızlandırma maliyetinin her ilave hızlandırma ile birlikte arttığı görülebilmektedir. Bunun nedeni, aktiviteyi daha kısa bir zaman diliminde tamamlamak için gereken ek çaba ve kaynaklardır. Bu nedenle, bir aktivitenin hızlandırılıp hızlandırılmayacağına karar verirken, hızlandırma maliyeti ile aktiviteyi normal süresinde tamamlama maliyetini göz önünde bulundurmak gerekmektedir.

Tablo 3.3: Metodolojinin akış şeması

Aktivite	Öncelik	Normal Zaman(gün)	Hızlandırma Süresi	Normal Maliyet(TL)	1.hızlandırma maliyeti	3.hızlandırma maliyeti	3.hızlandırma maliyeti
1	-	8	5	2500	20	25	35
2	-	6	4	1200	30	40	-
3	-	7	5	1450	30	35	-
4	1,2,3	6	4	1350	30	30	-
5	1,2,3	4	2	1250	20	30	-
6	2,3	5	3	2300	40	45	-
7	3	4	2	2000	40	45	-
8	4,5	6	5	2750	40	-	-
9	3,7	5	2	1800	20	35	50
10	4,5,9	4	2	850	45	55	-
11	7	8	5	3800	50	50	50
12	3	3	10	7	2500	15	20

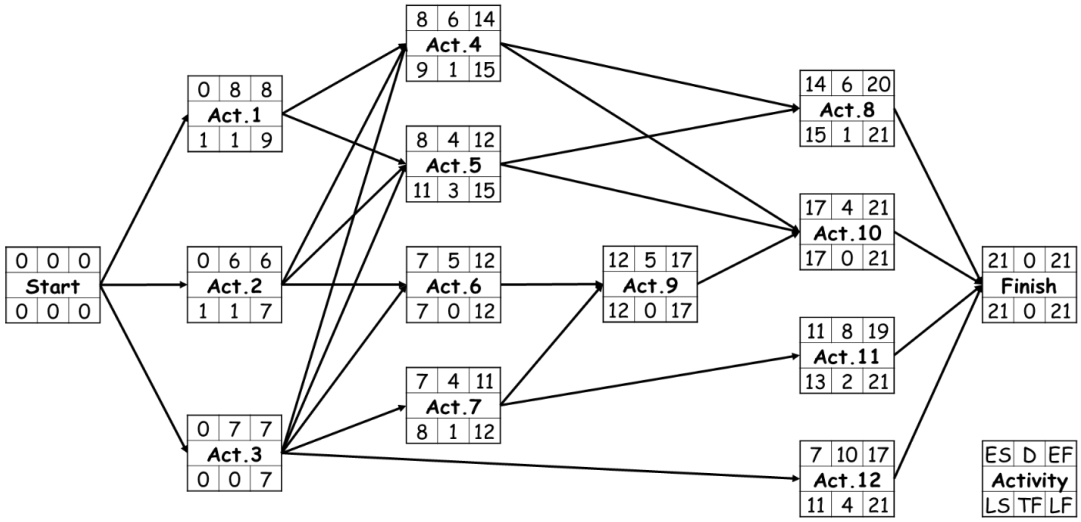
*Genel Gider: 100 TL/gün

3.5.1. Geleneksel yöntem

Geleneksel yöntemde ilk adım, aktivitelerin normal sürelerine ve maliyetlerine dayalı olarak proje maliyetini hesaplamaktır. Daha önce de belirtildiği gibi, proje maliyeti doğrudan ve dolaylı maliyetlerden oluşmaktadır. Doğrudan maliyet, Tablo 3.3'ün beşinci sütununda belirtilen maliyetlerin toplanmasıyla hesaplanmaktadır ve 23750TL olarak belirlenmiştir. Dolaylı maliyet, CPM'e göre belirlenen proje süresi ile genel gider maliyetinin çarpılmasıyla hesaplanmaktadır (Şekil 3.2). Proje süresini ve maliyetini hesaplamak için, Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10 olan kritik yol boyunca her bir aktivitenin süresi dikkate alınmaktadır. Projenin süresi ve maliyeti daha sonra sırasıyla 21 gün ve 25850TL (doğrudan ve dolaylı maliyetler) olarak belirlenmiştir.

Ayrıca, proje yöneticisi projeyi hızlandırmadan önce durumu değerlendirmelidir. Bu, aktivite süresindeki değişikliklerin proje süresi ve maliyeti üzerindeki etkisinin dikkate alınmasını gerektirmektedir. Başka bir deyişle, yönetici her bir aktivitenin süresindeki değişikliklerin genel proje süresi ve maliyeti üzerindeki etkisini analiz etmelidir. Hızlandırmadan önceki durum şu şekilde özetlenebilmektedir:

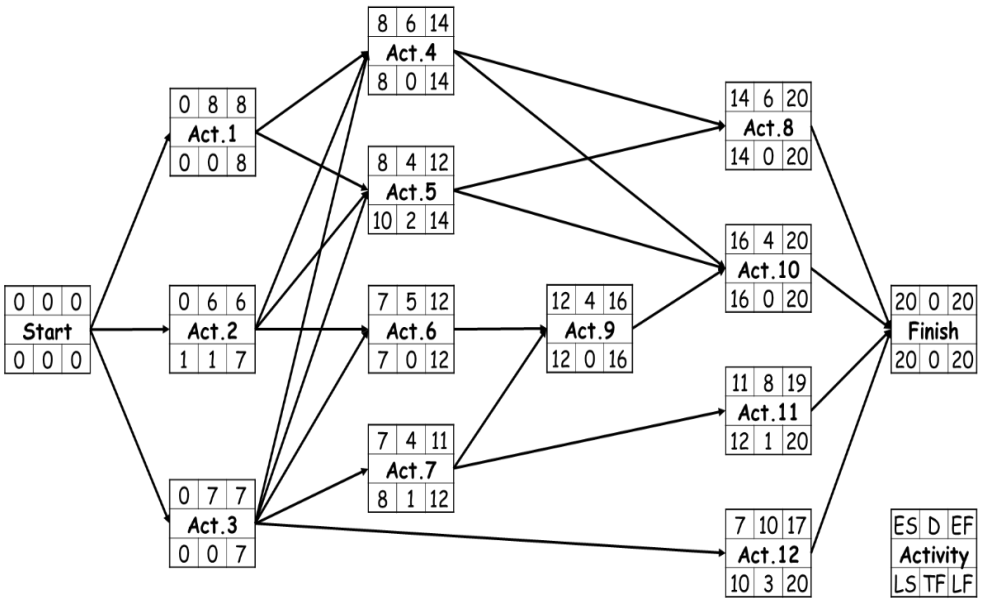
- Süre: 21 gün
- Maliyet: 23750 + 21 * 100 = 25850TL
- Kritik yol(lar):
    - Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10



Şekil 3.2 :Hızlandırmadan önceki CPM analizi

Proje süresini bir gün azaltarak istenen sonuca ulaşmak için tüm kritik yolların hızlandırılmasıu gerekmektedir. Bunu yapmak için, tüm kritik yolları hızlandıracak en ucuz aktivite kombinasyonu belirlenmeli ve bu durumda 100TL olan dolaylı maliyetteki tasarrufla karşılaştırılmalıdır. Değerlendirmenin ardından, Aktivite 9'u bir gün azaltmanın bu görevi yerine getirmenin en uygun maliyetli yolu olduğu belirlenmiştir, çünkü bunun maliyeti sadece 20 dolardır ve bu da tasarruf edilen 100 dolardan önemli ölçüde daha azdır. Aktivite hızlandırıldıktan sonra, proje süresi ve maliyeti yeniden hesaplanmalı ve yeni kritik yollar belirlenmelidir (Şekil 3.3). Bu süreç, projenin bir gün önce tamamlanmasını ve dolaylı maliyetlerdeki tasarrufun gerçekleştirilebilmesini sağlamak için gerekmektedir. İlk hızlandırmadan sonraki durum aşağıdaki gibi özetlenebilmektedir:

- Birinci hızlandırma: Aktivite 9'un 1 günlük hızlandırılması
- Ortaya Çıkan Süre: 20 gün
- Ortaya Çıkan Maaliyet: 25850 + 20 – 100 = 25770TL
- Ortaya Çıkan Kritik yol(lar):
- Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10
- Aktivite 1 - Aktivite 4 - Aktivite 8

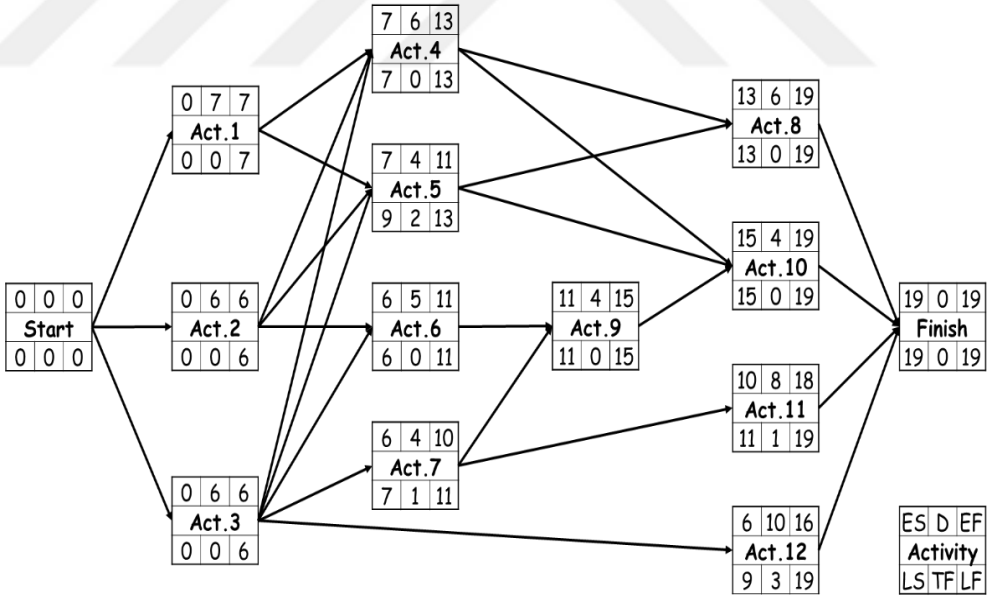


Şekil 3.3: Birinci Hızlandırmadan sonra CPM analizi

İlk hızlandırmanın sonunda iki kritik yol belirlenmektedir. Bu kritik yolları en uygun maliyetli şekilde hızlandırmak için Aktivite 1 ve Aktivite 3 en iyi kombinasyon olarak

belirlenmektedir. Bu aktivitelerin her ikisinin toplam maliyeti 50TL (20TL+30TL) olup, bu fiyat bu aktivitelerin hızlandırılmasıyla elde edilecek ek tasarruflardan daha azdır. Bu nedenle, bu aktivitelerin hızlandırılmasına ve proje süresinin bir gün azaltılmasına karar verilmiştir (Şekil 3.4). Bu aktivitelerin durdurulması sadece maddi tasarruf sağlamakla kalmamakta, aynı zamanda projenin zaman çizelgesini kısaltarak beklenenden daha erken tamamlanmasını sağlamaktadır. Bu sadece proje ekibine değil, aynı zamanda projeye dahil olan paydaşlara da fayda sağlamaktadır, çünkü projenin getirilerini çok daha erken alabileceklerdir. İkinci hızlandırmadan sonraki durum ise aşağıda özetlenmiştir:

- İkinci hızlandırma: Aktivite 1 ve Aktivite 3'ün bir gün hızlandırılması
- Ortaya çıkan süre: 19 gün
- Ortaya çıkan maliyet: 25770 + 20 + 30 – 100 = 25720TL
- Ortaya çıkan kritik yol(lar):
    - Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10
    - Aktivite 1 - Aktivite 4 - Aktivite 8
    - Aktivite 2 - Aktivite 6 - Aktivite 9 - Aktivite 10

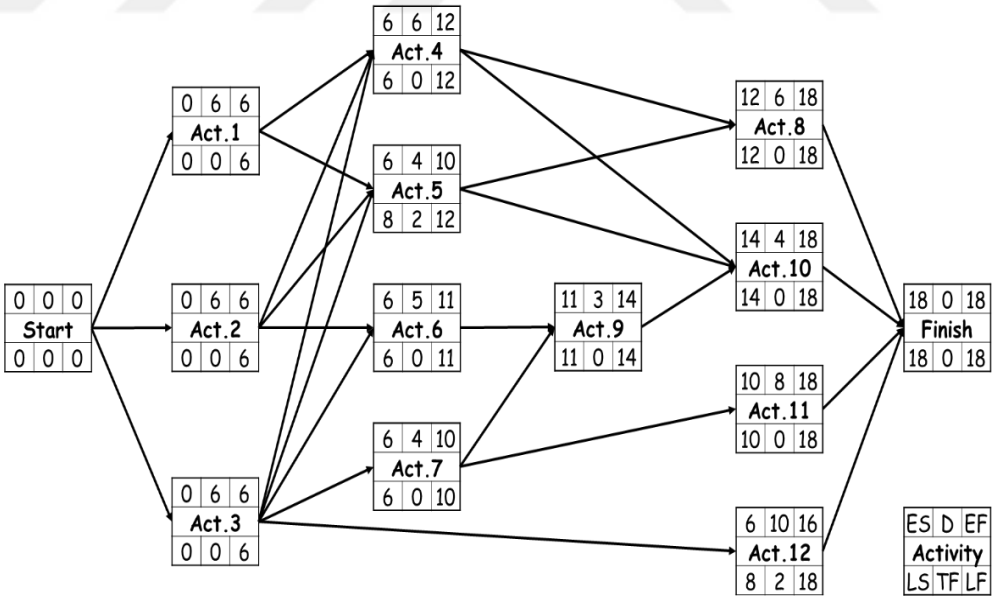


Şekil 3.4: İkinci Hızlandırmadan sonra CPM analizi

İkinci hızlandırma etkisinin farkına varıldığında, artık dikkate alınması gereken üç kritik yol olduğu anlaşılmaktadır. Bu yolları en uygun maliyetli şekilde hızlandırmak için Aktivite 1 ve Aktivite 9 en iyi kombinasyon olarak belirlenmiştir. Bu iki aktivitenin daha önceki hızlandırmalarda zaten bir günlüğüne hızlandırıldığı görülmektedir, bu

nedenle daha fazla hızlandırmanın maliyeti Aktivite 1 için 25TL ve Aktivite 9 için 35TL olacaktır. Bu iki aktivite aynı zamanda birinci ve üçüncü kritik yollardaki ortak aktiviteler olduğundan, her üç yolu da aynı anda hızlandırmak mümkündür. Bu iki aktiviteyi hızlandırmanın maliyeti, elde edilebilecek ek tasarruftan daha az olduğu için, devam edilmesine ve hızlandırılmasına karar verilmektedir (Şekil 3.5). Bu şekilde, üç kritik yolun da başarılı bir şekilde hızlandırılması ve ek tasarruftan tam olarak yararlanılması mümkündür. Üçüncü hızlandırmadan sonraki durum ise aşağıdaki gibi özetlenmektedir:

- Üçüncü hızlandırma: Aktivite 1 ve Aktivite 9'un 1 gün hızlandırılması
- Ortaya çıkan süre: 18 gün
- Ortaya çıkan maliyet: 25720 + 25 + 35 – 100 = 25680TL
- Ortaya çıkan kritik yol(lar):
    - Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10
    - Aktivite 1 - Aktivite 4 - Aktivite 8
    - Aktivite 2 - Aktivite 6 - Aktivite 9 - Aktivite 10
    - Aktivite 3 - Aktivite 4 - Aktivite 8
    - Aktivite 3 - Aktivite 7 - Aktivite 11

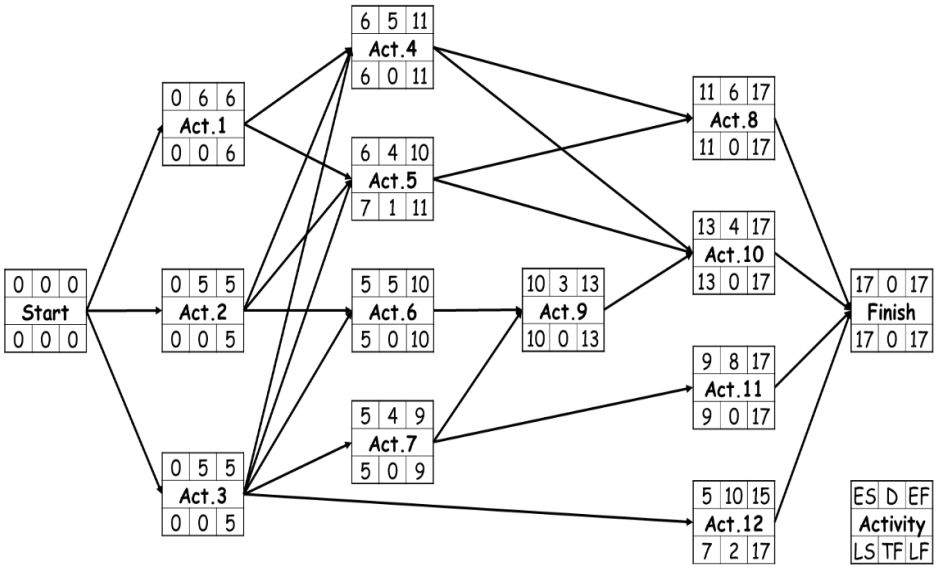


Şekil 3.5: Üçüncü Hızlandırmadan Sonra CPM Analizi

Toplam beş kritik yola ulaşmak için dördüncü hızlandırma gereklidir ve Aktivite 2, Aktivite 3 ve Aktivite 4'ün hızlandırılmasıyla gerçekleştirilmektedir. Aktivite 3, ikinci hızlandırmada zaten bir günlüğüne hızlanmış olan tek aktivitedir ve bu nedenle

hızlandırma maliyeti normalde 30TL yerine 35TL olmaktadır. Hızlandırma maliyeti biraz artmasına rağmen (toplamda 30TL+35TL+30TL), dolaylı maliyetlerdeki potansiyel tasarruftan hala daha ucuzdur. Bu nedenle Şekil 3.6'da gösterildiği gibi dördüncü hızlandırma gerçekleşmektedir. Dördüncü hızlandırma, geri kalan aktiviteleriniş yükünü ve projenin toplam süresini azaltmaya yardımcı olacağı için faydalıdır. Bu da hem doğrudan hem de dolaylı maliyetler açısından bir maliyet tasarrufu sağlamaktadır. Kritik yolların bir parçası olan aktiviteler hızlandırıldığında, bunlara tahsis edilen kaynaklar daha verimli kullanılabilmekte ve bu da daha uygun maliyetli proje sağlamaktadır. Dördüncü hızlandırma ile proje ekibi, projenin planlanan bütçe ve zaman çizelgesi içinde tamamlanacağından daha emin olabilmektedir. Dördüncü hızlandırmadan sonraki durum ise aşağıdaki gibi özetlenmektedir:

- Dördüncü hızlandırma: Aktivite 2, Aktivite 3 ve Aktivite 4'ün 1 gün hızlandırılması
- Ortaya çıkan süre: 17 gün
- Ortaya çıkan maliyet: 25680 + 30 + 35 + 30 – 100 = 25675TL
- Ortaya çıkan kritik yol(lar):
    - Aktivite 3 - Aktivite 6 - Aktivite 9 - Aktivite 10
    - Aktivite 1 - Aktivite 4 - Aktivite 8
    - Aktivite 2 - Aktivite 6 - Aktivite 9 - Aktivite 10
    - Aktivite 3 - Aktivite 7 - Aktivite 11



Şekil 3.6: Dördüncü hızlandırmadan sonra CPM Analizi

Dördüncü hızlandırmadan sonra dört kritik yol belirlenmiştir ve proje süresini bir gün azaltmanın en ucuz yolu Aktivite 4, Aktivite 6 ve Aktivite 7'yi hızlandırmaktır. Ancak, bu aktiviteleri hızlandırmanın maliyeti (30 TL + 40 TL + 40 TL) dolaylı maliyetlerdeki potansiyel tasarrufu aşmaktadır. Bu durum, bu aktivitelerin durdurulması halinde toplam proje maliyetinin artacağı ve sürecin sonlandırılacağı anlamına gelmektedir. Bundan sonra, mevcut durum optimum olarak tanımlanmaktadır. Optimum proje süresi ve proje maliyeti sırasıyla 17 gün ve 25675 TL olarak elde edilmiştir. Bu optimum durum, Aktivite 1'in 2 gün, Aktivite 2'nin 1 gün, Aktivite 3'ün 2 gün, Aktivite 4'ün 1 gün ve Aktivite 9'un 2 gün azaltılmasıyla elde edilirken, diğer tüm aktiviteler normal sürelerinde ve maliyetlerinde gerçekleştirilmektedir. Bu süreç, işin kalitesinden ödün vermeden proje süresini ve maliyetini azaltmaya yardımcı olmaktadır.

3.5.2. Karınca kolonisi algoritma yaklaşımı

Önerilen karınca kolonisi algoritma yaklaşımı, proje hızlandırma problemlerinin çözümü için yenilikçi bir yöntemdir. Karınca kolonisi algoritma yaklaşımının uyarlanması, problemin karınca kolonisi algoritma optimizasyonunu destekleyen bir programlama platformu tarafından yorumlanabilecek bir forma dönüştürülmesini gerektirmektedir. MATLAB, bu süreci kolaylaştırma kabiliyeti nedeniyle tercih edilen ticari programlama platformu olarak belirlenmiştir. Karınca kolonisi algoritma konseptinin kullanımını göstermek için karar değişkenleri, süreler/maliyetler, kısıtlamalar ve amaç fonksiyonu tanımlanmalı ve formüle edilmelidir. Bu unsurlar yerine getirildikten sonra, karınca kolonisi algoritma bir çözüm üretmek için probleme uygulanabilmektedir. Bu yaklaşımın özellikle geleneksel yöntemlerin optimal bir çözüm üretemediği durumlarda etkili olduğu görülmüştür. Ayrıca, karınca kolonisi algoritma çok çeşitli potansiyel çözümleri belirleme ve değerlendirme yeteneğine sahiptir, bu da en verimli sonucu bulma olasılığını artırmaktadır.

3.5.2.1. Problemi dönüştürme

Proje hızlandırma problemleri genellikle her bir aktivitenin normal maliyetleri ve süresinin yanı sıra varsa hızlandırma maliyetlerini özetleyen Tablo 3.2'deki gibi sunulmaktadır. Problemi bir karınca kolonisi algoritma yaklaşımı kullanarak çözmek için, her bir aktivitenin alternatifleri sunulmalıdır. Her bir hızlandırmadan sonra bir

aktivitenin maliyeti ve süresi bir paket olarak sunulmalıdır. Bu amaçla problem, her bir aktivite için alternatiflerin daha açık bir şekilde belirtildiği Tablo 3.4'e dönüştürülmüştür. Her bir aktivite için alternatif sayısı iki ile dört arasında değişebilmekte ve bu alternatiflerin çarpımı bir milyondan fazla çözümle sonuçlanmaktadır. Spesifik olarak, karınca kolonisi algoritmanın toplam 1.119.744 çözüm arasından en uygun çözümü bulması beklenmektedir. Bu, özellikle çok sayıda potansiyel çözüm düşünüldüğünde zor bir görev olabilmektedir. Mümkün olan en iyi sonuçları elde etmek için, problemin doğru ve kesin bir şekilde sunulduğundan emin olmak önemlidir. Bu, karınca kolonisi algoritmanın en iyi çözümü tanımlamasına izin vermek için gerekli verilerin sağlanmasıyla gerçekleştirilebilmektedir. Bu adımları takip ederek, proje hızlandırma problemine en uygun çözümü bulmak mümkündür.

Tablo 3.4: Aktivite maliyetlerinin/sürelerinin dönüştürülmüş hali

Aktivite	Öncelik	Normal Süre/ Maliyet (Gün/TL)	1.hızlandırma Süre/ Maliyet(Gün/ TL)	2.hızlandırma Süre/ Maliyet(Gün/ TL)	2.hızlandırma Süre/ Maliyet(Gün/ TL)
1	-	8/2500	7/2520	6/2545	5/2580
2	-	6/1200	5/1230	4/1275	-
3	-	7/1450	6/1480	5/1515	-
4	1,2,3	6/1350	5/1380	4/1410	-
5	1,2,3	4/1250	3/1270	2/1300	-
6	2,3	5/2300	4/2340	3/2395	-
7	3	4/2000	3/2040	2/2085	-
8	4,5	6/2750	5/2790	-	-
9	3,7	5/1800	4/1820	3/1855	2/1905
10	4,5,9	4/850	3/895	2/950	-
11	7	8/3800	7/3850	6/3900	5/3950
12	3	10/2500	9/2515	8/2535	7/2560

*Genel Gider: 100TL/gün

3.5.2.2. Karar değişkenlerini tanımlama

Karınca kolonisi algoritmalar kullanılırken karar değişkeninin tanımı büyük önem taşımaktadır. Karar değişkeninin, diğer tüm değişkenlerin ve amaç fonksiyonunun karar değişkeninin bir fonksiyonu olarak ifade edilebileceği şekilde tanımlanması gerekmektedir. Bunu başarmak için, karar değişkeni ile diğer değişkenler arasındaki ve karar değişkeni ile amaç fonksiyonu arasındaki ilişkilerin tanımlanması önemlidir.

Karar değişkeni, problemi çözmek için gerekli olan tüm ilgili bilgileri karşılayacak şekilde tanımlanmalıdır. Bu amaçla, karşılanması gereken çeşitli kısıtlamaları ve problemin amacını göz önünde bulundurmak faydalı olacaktır. Karar değişkeni belirlendikten sonra, optimizasyon problemini formüle etmek ve karınca kolonisi algoritmayı oluşturmak için kullanılabilmektedir. Karar değişkeni, problemi etkili bir şekilde temsil etmek için aşağıdaki gibi tanımlanmaktadır:

$x_i =$ hızlandırılmış aktivite sayısı

$x_i \in Z^n$

Karar değişkeni, aktivitelerine ek kaynaklar atayarak bir projenin süresini azaltan bir optimizasyon tekniği olan, ilgili aktivitenin kaç kez hızlandırılmasını gerektiğini ifade eden bir tamsayıdır. Bu yaklaşım sayesinde, proje hızlandırma problemi esasen her bir aktivitenin kaç kez hızlandırılması gerektiğine dair bir seçime dönüştürülmektedir. Bu teknik, proje maliyeti ile proje süresi arasında bir değiş tokuş olduğu varsayımına dayanmaktadır; dolayısıyla, hızlandırma yoluyla amaç, proje süresini en verimli şekilde azaltan en uygun maliyetli çözümü bulmaktır.

Karar değişkenlerinin bir fonksiyonu olarak ifade edilecek diğer değişkenler maliyet, süre ve kaynak mevcudiyetidir. Maliyet, projeyi tamamlamak için gereken para miktarı; süre, projeyi bitirmek için gereken toplam zaman miktarı; kaynak mevcudiyeti ise proje için mevcut olan kaynak miktarıdır. Tüm bu değişkenler birbirine bağlıdır, bu nedenle proje hızlandırma probleminin çözümü tüm bu değişkenleri aynı anda dikkate almalıdır.

$d_i = i$ aktivitesinin süresi

$c_i = i$ aktivitesinin maliyeti
(10)

$ES_i = i$ aktivitesinin en erken başlama zamanı
(11)

$EF_i = i$ aktivitesinin en bitiş zamanı
(12)

3.5.2.3. Kısıtlamaların belirlenmesi

Optimizasyon problemlerini formüle ederken, değişkenlerin tabi olduğu kısıtlamaları dikkate almak önemlidir. Proje hızlandırma problemlerinde, kısıtlamalar temel olarak

bir aktivitenin kaç kez hızlandırılacağı ile ilgilidir. Tablo 3.4'te gösterildiği gibi, herhangi bir aktivite için minimum hızlandırma sayısı sıfırdır (normal süre ve maliyete karşılık gelir), maksimum hızlandırma sayısı ise genellikle 1 ila 3 arasındadır. Bu tür problemleri modellemek için MATLAB kodu yazarken, kısıtlamalar koda dahil edilmelidir. Örneğin, kodda bir aktivitenin hızlandırma sayısının Tablo 3.4'te belirtilen minimum ve maksimum değerler arasında olması gerektiği yazılabilmektedir. Ayrıca, optimizasyon probleminde mevcut olabilecek sınırlı bütçe, maksimum işçi sayısı gibi diğer kısıtlamaların da dikkate alınması önemlidir. Bunun yapıldığı aşağıdaki gibi bir kod daha doğru ve güvenilir bir optimizasyon modeli sağlayacaktır:

$$
\begin{aligned}
0 \leq x_1 \leq 3 \\
0 \leq x_2 \leq 2 \\
0 \leq x_3 \leq 2 \\
0 \leq x_4 \leq 2 \\
0 \leq x_5 \leq 2 \\
0 \leq x_6 \leq 2 \\
0 \leq x_7 \leq 2 \\
0 \leq x_8 \leq 1 \\
0 \leq x_9 \leq 3 \\
0 \leq x_{10} \leq 2 \\
0 \leq x_{11} \leq 3 \\
0 \leq x_{12} \leq 3
\end{aligned}
\tag{3.1}
$$

3.5.2.4. Amaç fonksiyonu

Karınca kolonisi algoritma yaklaşımının son adımı amaç fonksiyonunu tanımlamak ve ifade etmektir. Proje hızlandırma probleminin amacı proje maliyetini c_{proje} minimize etmektir. Doğrudan maliyetler, her bir aktivitenin ayrı ayrı maliyetinin toplanmasıyla hesaplanmaktadır ($c_1 + c_2 + c_3 \ldots + c_{12}$). Dolaylı maliyetler, genel gider maliyetinin (100 TL/gün) proje süresi d_{proje} ile çarpılmasıyla hesaplanmaktadır. Böylece amaç fonksiyonu, toplam proje maliyetini en aza indirmek amacıyla doğrudan ve dolaylı maliyetleri birleştiren matematiksel bir ifade olarak formüle edilmektedir. Bunu

başarmak için, aktivite maliyetleri ve proje süresinin en uygun kombinasyonunu belirlemek amacıyla matematiksel ifadenin optimize edilmesi gerekmektedir. Bu değerler belirlendikten sonra proje maliyeti aşağıdaki gibi hesaplanabilmekte ve optimum çözüm belirlenebilmektedir:

$$c_{proje} = c_1+c_2+c_3+c_4+c_5+c_6+c_7+c_8+c_9+c_{10}+c_{11}+c_{12}+100*d_{proje} \quad (3.2)$$

Bu denklemde, her bir aktivitenin bireysel maliyeti karar değişkenlerinin bir fonksiyonudur ve genel gider maliyeti 100 TL/gün'dür. Proje süresi de karar değişkenlerinin bir fonksiyonu olarak ifade edilmelidir. Proje süresi, CPM aracılığıyla hesaplanmaktadır. Bu yaklaşımda, proje süresini her hızlandırmadan sonra hesaplamak yerine, aktiviteler arasındaki ilişkiler yalnızca bir kez formüle edilmektedir. Her bir aktivitenin en erken başlangıç (ES) ve en erken bitiş (EF) süreleri, karar değişkenlerinin (x_i) bir fonksiyonu olarak ifade edilen aktivite sürelerinin (d_i) bir fonksiyonu olarak ifade edilmektedir. Proje süresi, son aktivitenin EF'sine eşittir.

Karınca kolonisi algoritma yaklaşımında aktivitelerin en geç başlama (LS) ve en geç bitiş (LF) süreleri formüle edilmemektedir çünkü toplam kayma (TF) sürelerinin ve kritik yol(lar)ın belirlenmesine gerek yoktur. Bunun nedeni, karınca kolonisi algoritma yaklaşımının kritik yolların belirlenmesinden ziyade sadece proje süresinin hesaplanmasını gerektirmesidir. Buna göre, proje süresi aşağıdaki gibi formüle edilmektedir:

$$ES_{\text{başlangıç}} = 0$$

$$EF_{\text{başlangıç}} = 0$$

$$EF_i = ES_i + d_i, \qquad t = 1, 2, 3, \dots, 12$$

$$ES_1 = EF_{\text{başlangıç}}$$

$$ES_2 = EF_{\text{başlangıç}}$$

$$ES_3 = EF_{\text{başlangıç}}$$

$$ES_4 = \max(EF_1, EF_2, EF_3)$$

$$ES_5 = \max(EF_1, EF_2, EF_3)$$

$$ES_6 = \max(EF_2, EF_3)$$

$$ES_7 = EF_3$$

$$ES_8 = \max(EF_4, EF_5)$$

$$ES_9 = \max(EF_6, EF_7)$$

$$ES_{10} = \max(EF_4, EF_5, EF_9)$$

$$ES_{11} = EF_7$$

$$ES_{12} = EF_3$$

$$ES_{\text{bitiş}} = \max(EF_8, EF_{10}, EF_{11}, EF_{12})$$

$$EF_{\text{bitiş}} = ES_{\text{bitiş}}$$

$$d_{\text{proje}} = EF_{\text{bitiş}} \quad (3.3)$$

Bu, hem CPM hem de PERT ağı oluşturarak ve ardından ilk bölümdeki Denklem 1 ile operasyon sürelerini hesaplayarak projenin ilerlemesinin bir temsilini sağlamaktadır. Ayrıca, istenen sonuca bağlı olarak, çeşitli karınca kolonisi algoritma varsayımları (popülasyon büyüklüğü, elitizm oranı, nesil sayısı ve iterasyon sayısı) dahil edilebilmektedir. Bu amaçla, tüm KKA varsayımları için çaprazlama kullanılmaktadır. Bu, KKA aşamalarının hızlı bir şekilde tamamlanmasını sağlarken aynı zamanda en uygun çözüme ulaşılabilmesi için değerlerin optimize edilmesine olanak tanımaktadır.

3.6. Algoritmanın Yapısı Ve İşleyişi

Geliştirilen ve önerilen algoritma CPM ve PERT ağlarına uygulanabilmektedir. Algoritma, karmaşık problemleri çözmek için güçlü bir araç olan Karınca Kolonisi Algoritması çerçevesinde sistematik olarak açıklanmıştır. KKA, bir probleme en uygun çözümü bulmak için farklı stratejiler kullanan sezgisel bir arama tekniğidir. KKA, proje çizelgeleme problemleri de dahil olmak üzere çok çeşitli problemlere uygulanabilmesi açısından avantajlıdır. Önerilen KKA'nın herhangi bir projeye uygulanması Şekil 3.9'da verilen basit bir ağ örneği ile gösterilebilmektedir. Bu örnekte uygulama CPM ağ diyagramı üzerinde gösterilmiştir. Ancak CPM ve PERT ağları arasındaki temel fark, PERT ağlarında operasyon süresinin iyimser, kötümser ve yaklaşık süreler

hesaplandıktan sonra belirlenmesidir. Bu hesaplama, her bir görevin süresi için üç olası senaryo göz önünde bulundurularak ve en olası süre hesaplanarak yapılmaktadır.

3.6.1. Karınca turunun oluşturulması

Karınca kolonisinde ilk olarak kaç tane karınca olacağı belirlenir. Daha sonra her bir karınca rastgele olarak bir düğüme yerleştirilir ve bütün düğümleri tek tek ziyaret ederek turunu tamamlar. Her bir karıncanın mevcut düğümden bir sonraki düğüme gidebilmesinin matematiksel formülü Denklem 1'de verilmiştir (Özdemir, 2008; Serin, 2009; Dikmen, 2014).

$$P_{i,J}^{l} = \frac{[T_{ij}]^{a}[n_{ij}]^{\beta}}{\sum_{l \in N_i^l}[T_{il}]^{a}[\eta_{il}]^{\beta}} \tag{3.4}$$

Burada,

P_{ij}^{l}: ℓ karıncasının i düğümünden j düğümüne geçme olasılığı,

τ_{ij}: i ve j düğümleri arasındaki feromon değeri

$\square_{ij}$: i ve j düğümleri arasındaki sezgisel değeri

α: feromon katsayısı

β: sezgisel katsayısı

N : düğümler kümesi

3.6.2. Karınca kolonisi algoritması parametreleri

Karınca Sayısı: Kolonide kaç tane karıncanın olacağını belirleyen parametredir (Dikmen, 2014).

İterasyon Sayısı: Arama işleminin kaç iterasyon (adım) gerçekleşeceğini belirleyen parametredir.

Zeta: Karınca kolonisi algoritmasında sapma-uzaklık oranını belirler. Yeni çözüm adaylarının oluşturulması sırasında sapma ve uzaklık arasındaki ilişkiyi kontrol eder.

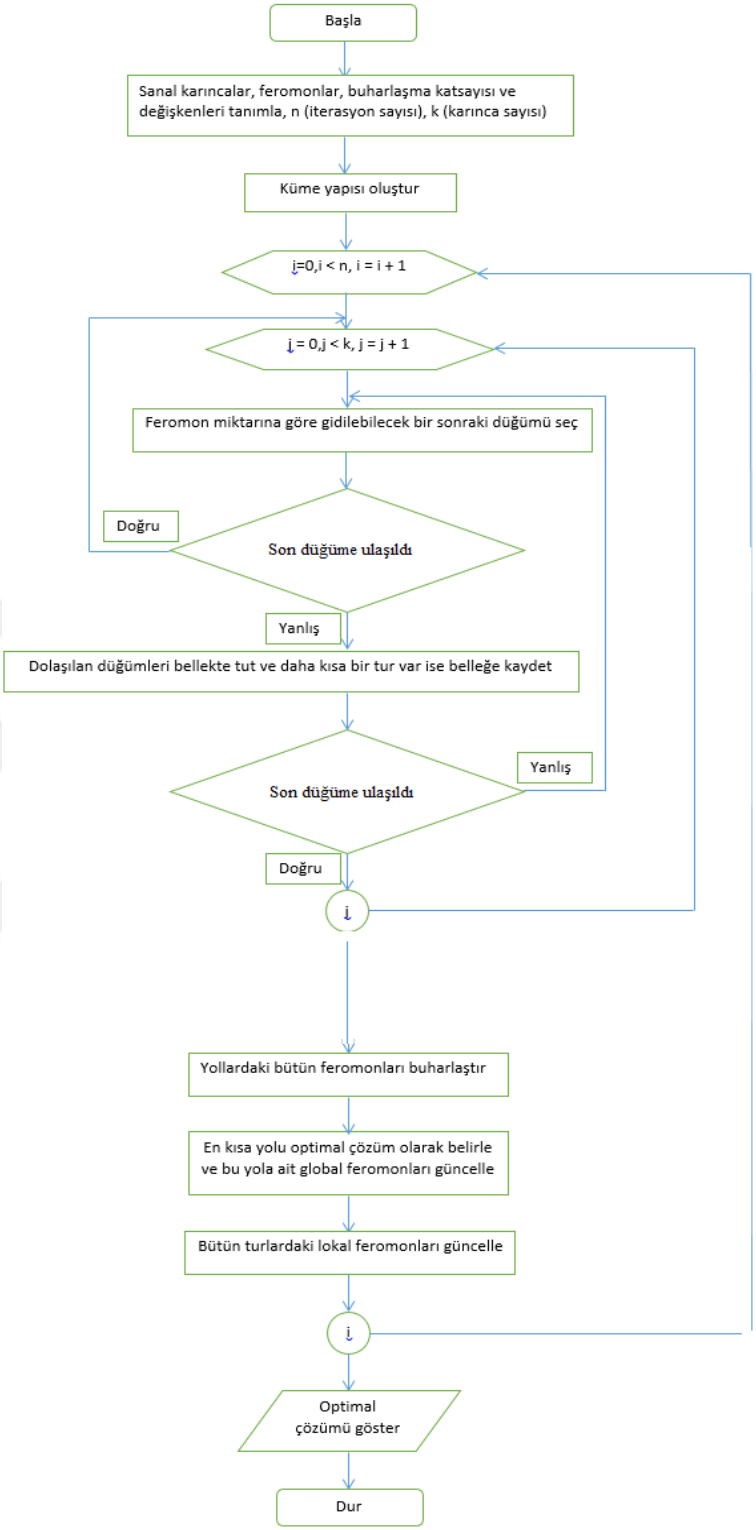
nSample: Karınca kolonisi algoritmasında örneklem boyutunu belirler. Her iterasyonda seçilecek örneklem sayısını ifade eder. Bu örneklem, çevredeki bilgileri toplamak ve yeni çözüm adayları oluşturmak için kullanılır.

Feromon Buharlaşma Oranı (ρ): Her iterasyon sonunda düğümler arasındaki feromonların hangi oranda buharlaşacağını belirleyen parametredir.

Tablo 3.5'te çalışma kapsamında kullanılan parametre değerleri verilmektedir.

Tablo 3.5: Kullanılan parametreler

Parametre	Değer
ρ	0,5
nSample	28
Karınca sayısı	50
İterasyon sayısı	50
zeta	50



Şekil 3.7: Karınca Kolonisi Algoritmasının Akış Çizgesi

BÖLÜM 4. BULGULAR

Bu algoritmada oluşturulan karıncaların en iyi yolları belirlemek için kullanılan bir yöntemdir. Sonuç olarak bu uygulamanın son iterasyonunda en iyi optimum sonuç elde edilmiştir. Sonuçların gösterimi: Karınca Kolonisi Algoritması işlemleri, kullanıcının iterasyonlar için belirlediği sayıda gerçekleştirilmekte ve ardından elde edilen en avantajlı çözüm sunulmaktadır.

Tablo 4.1: Sonuçların gösterimi

Aktivite	1	2	3	4	5	6	7	8	9	10	11	12
En İyi Çözüm	2	1	2	1	0	0	0	0	2	0	0	0

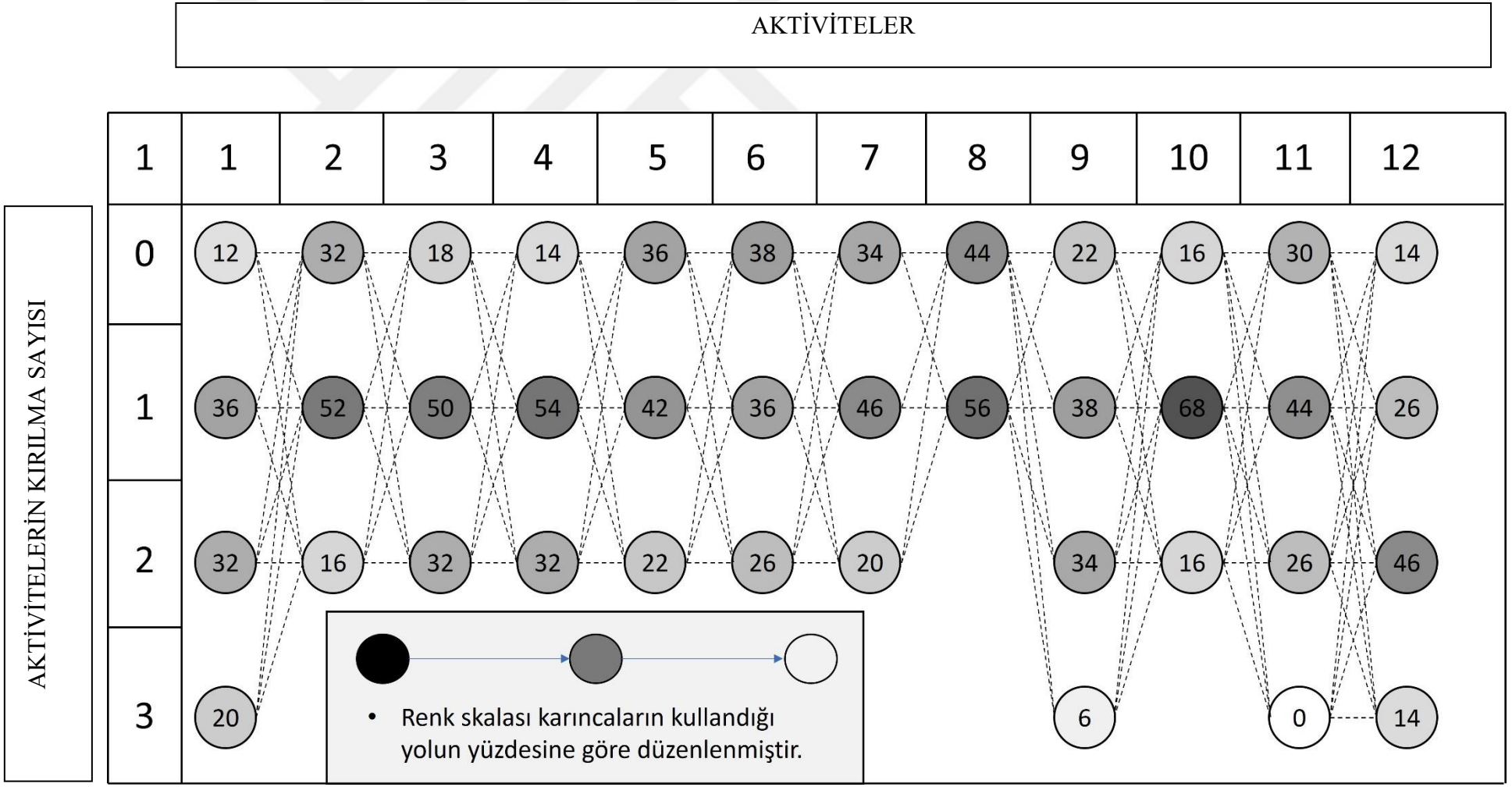
Karınca Kolonisi Algoritması uygulaması operasyon sürelerine dayalı olarak belirlenen amaç fonksiyonunu dikkate alarak sonucu belirlemiştir ve sonuçlar bu uygulama ile sunulmuştur. Çalışma sonucunda optimum çözüm 1. Aktivitenin 2 kez, 2. Aktivitenin 1 kez, 3. Aktivitenin 2 kez, 4. Aktivitenin 1 kez, 9. Aktivitenin 1 kez kırılması sonucuna varılmıştır.

4.1. İterasyon sayısına göre karınca yollarındaki değişimler

Yukarıda verilen şekillerde (Şekil 4.1, Şekil 4.2, Şekil 4.3, Şekil 4.4, Şekil 4.5, Şekil 4.6) iterasyon sayılarına bağlı olarak değişen karınca yolları gösterilmiştir. Karıncaların her iterasyonda hangi aktivitenin kaç kere kırılması gerektiğini söylediği gözlemlenmiştir.

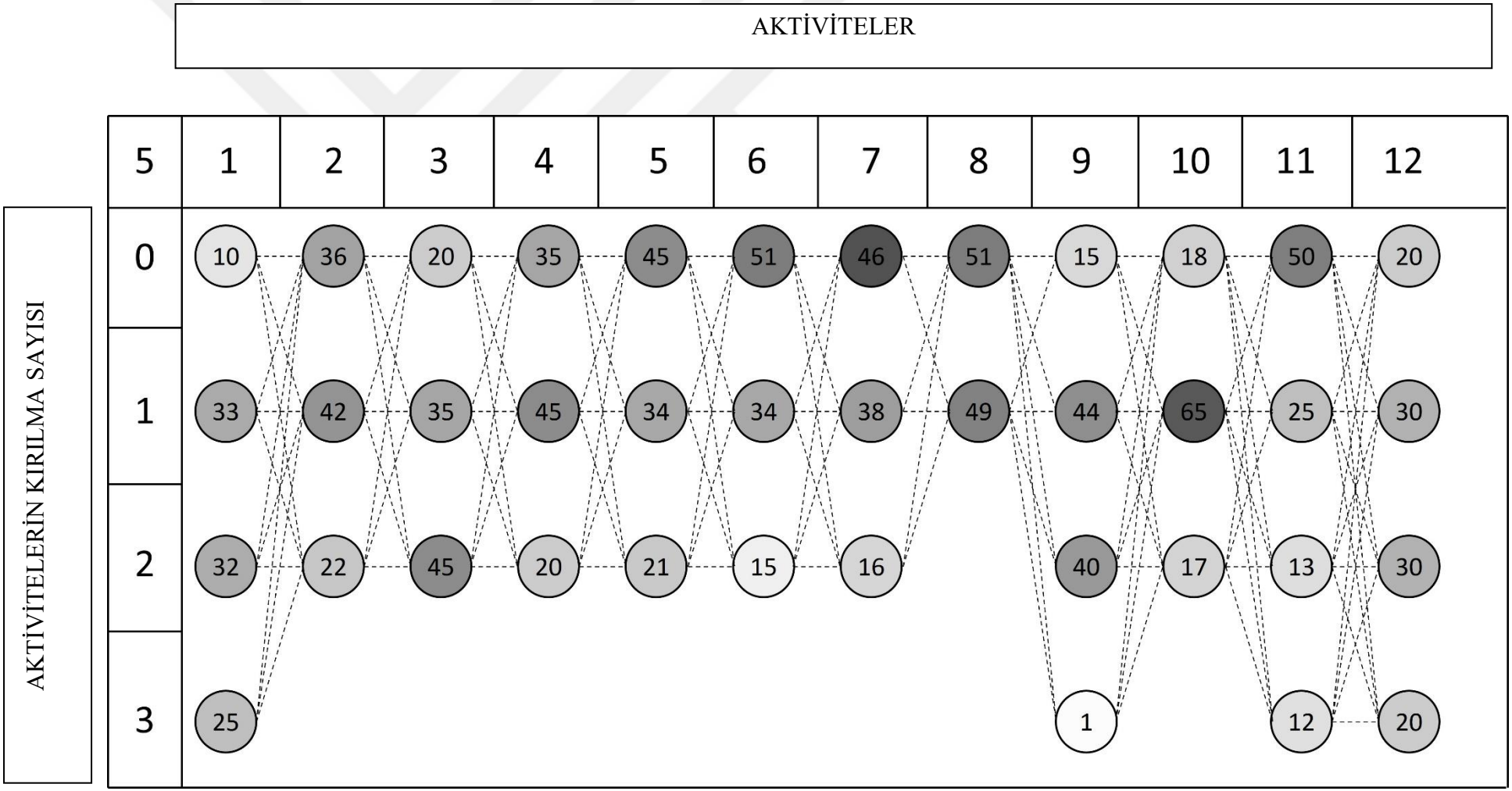
Bu çözümleme yapılırken karınca sayısı 50 alınmış olup yuvarlak şekillerin içinde gösterilen karınca sayısı '%' şeklinde ifade edilmiştir. Karıncaların sayısına göre şekillerin içerisindeki dolgu karınca yüzdesi değerince koyulaşacaktır. Örneklendirmemiz gerekirse 50. iterasyonda birinci aktiviteyi 2 kere kırma yolunu

seçen karıncaların sayısı 38 olup bunun gösterimi %76 şeklinde ve %76 beyaz-siyah koyuluğundadır. Buradan yola çıkarak karıncaların 1'den 50'ye kadar izlemiş olduğu karınca yollarını görebilirsiniz.



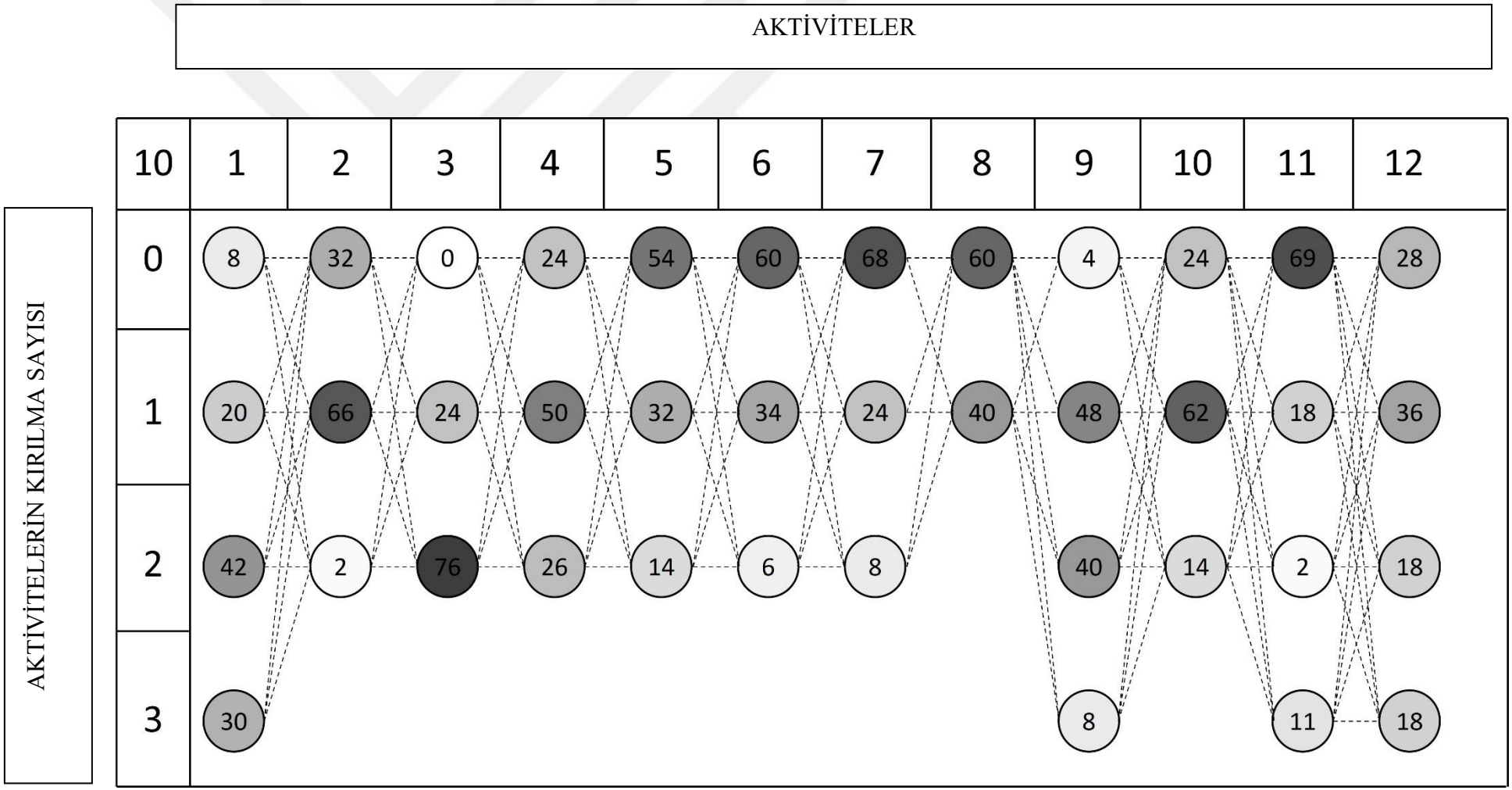
NOT: 1. İterasyonda maliyet 25750 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.1: Birinci iterasyondaki karınca yolları



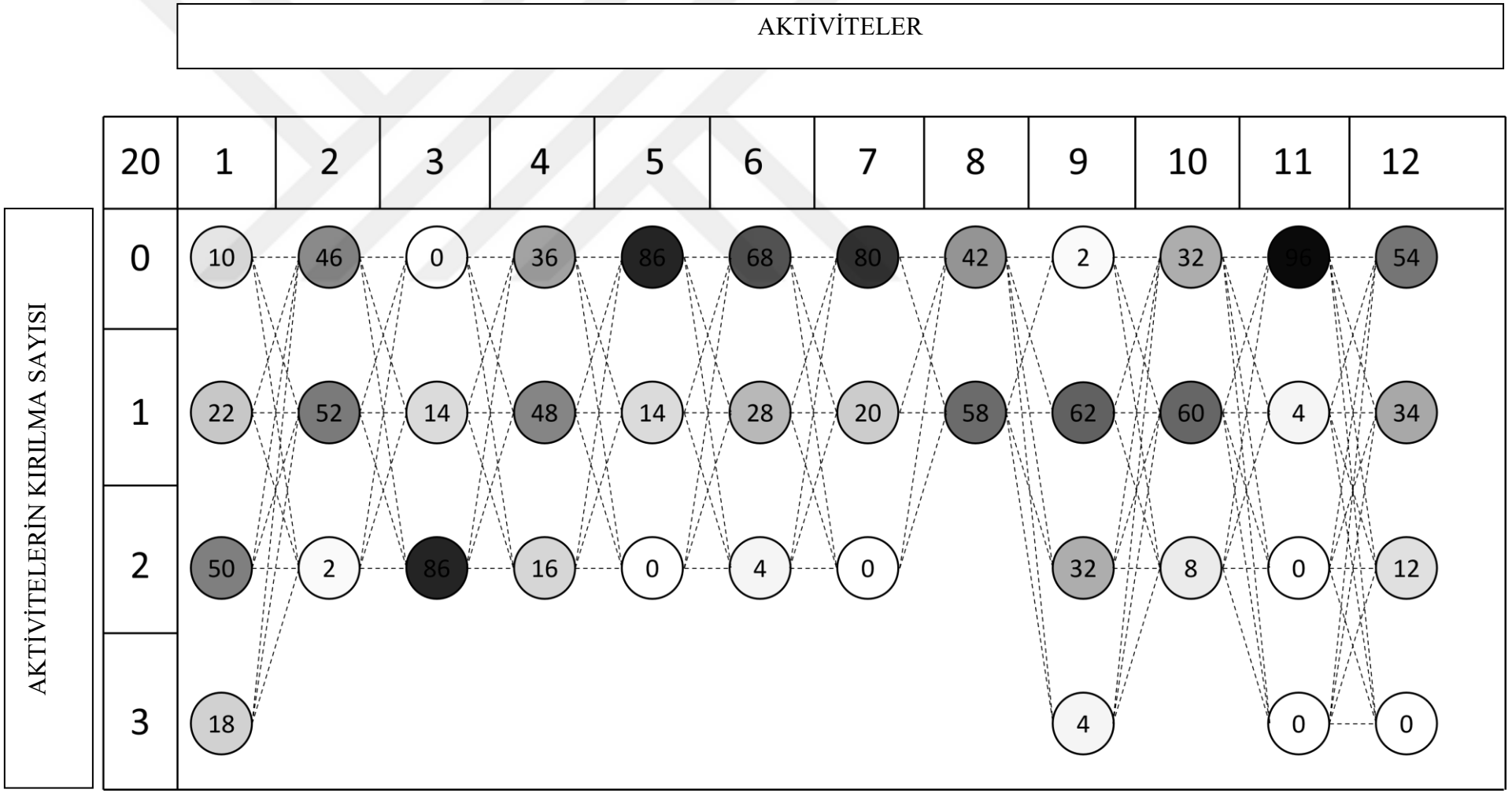
NOT: 5. İterasyonda maliyet 25750 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.2 Beşinci iterasyondaki karınca yolları



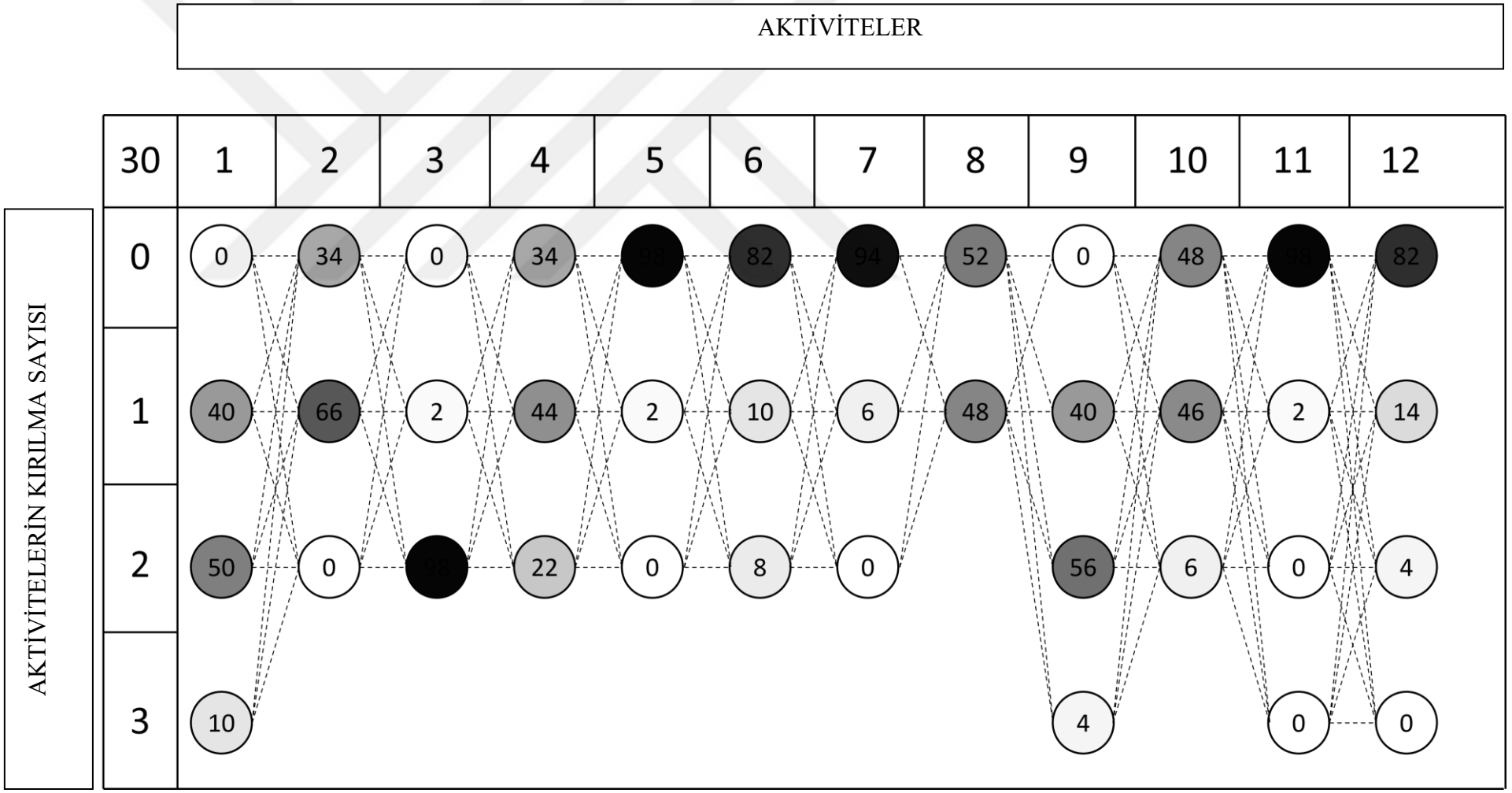
NOT: 10. İterasyonda maliyet 25685 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.3: Onuncu iterasyondaki karınca yolları



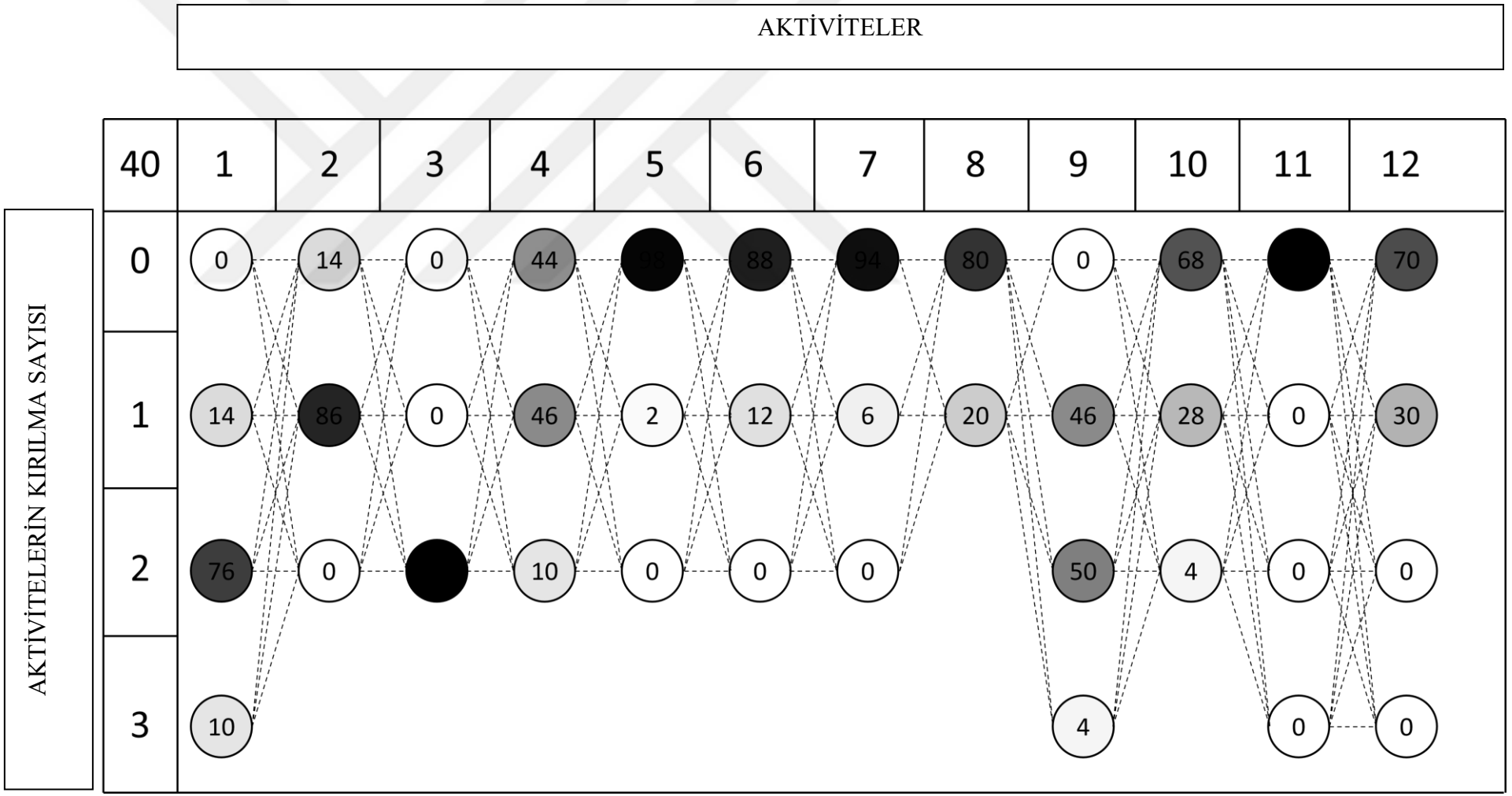
NOT: 20. İterasyonda maliyet 25685 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.4: Yirminci iterasyondaki karınca yolları



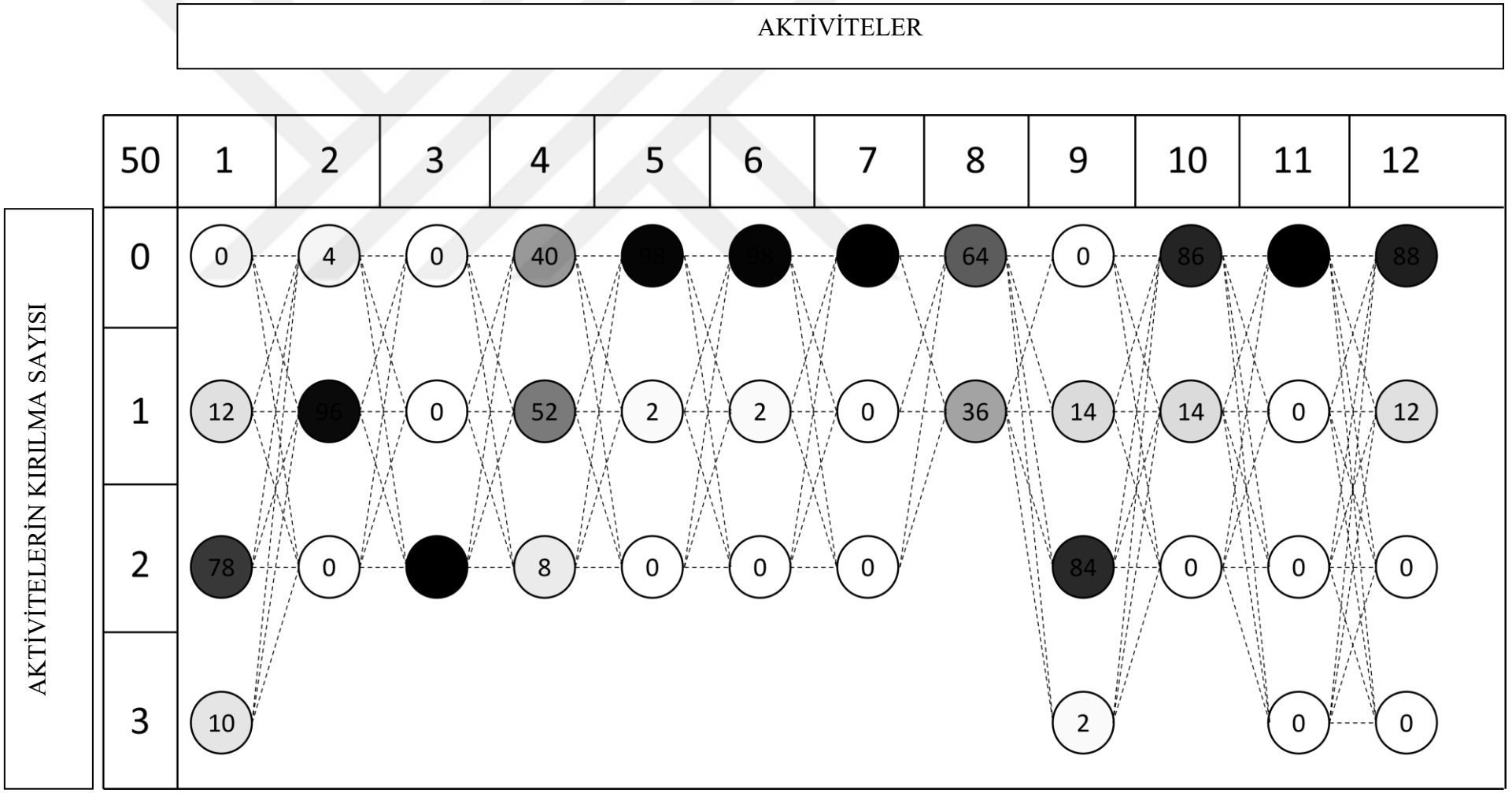
NOT: 30.iterasyonda maliyet 25675 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.5: Otuzuncu iterasyondaki karınca yolları



NOT: 40.iterasyonda maliyet 25675 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.6: Kırkıncı iterasyondaki karınca yolları



NOT: 50.iterasyonda maliyet 25675 TL olarak bulunmuştur. Şekildeki sayılar % cinsinden verilmiştir.

Şekil 4.7: Ellinci iterasyondaki karınca yolları

4.2. Popülasyon ve iterasyon arasındaki ilişkisi

Aşağıdaki tabloda (Tablo 4.1) popülasyon ve iterasyon arasındaki ilişkiyi gösterilmiştir. Bu tabloda iterasyon ve popülasyon için optimum değerler olduğu görülmektedir. Tek başına sadece bir değerin artması değeri optimum sonuçtan uzaklaştırıyor.

Yapılan bu deneyin sonunda popülasyon ve iterasyon değerinin birbirine yakın olması bizim sonucumuzu optimum hale getirdiğini göstermektedir.

Bu tablo KKA'ları başka problemlerde kullanırken başlangıç değerlerini seçmemize yardımcı olacaktır. Her problemde bu doğruluk oranını sağlamasa bile yine en yüksek doğruluk oranları popülasyon ve iterasyonun yakın olduğu seçmelerde çıkacaktır.

Yapılacak yeni çalışmalarda bu optimizasyonlar kullanılırken sonucu bilinen bir problemle optimum parametreleri oluşturup bu parametre değerleri sonucunda yeni ve çözümünü bulmak istediğimiz projeyi tasarlarsak sonucun daha hızlı ve doğru olacağını görmekteyiz.

Tablo 4.2: İterasyon ve popülasyon arasındaki bağlantı

		Popülasyon									
Doğruluk Oranı (%)		10	20	30	40	50	60	70	80	90	100
İterasyon	10	17	11	5	4	2	2	1	1	1	0
	20	4	48	34	31	19	16	11	10	8	7
	30	62	67	70	56	51	50	32	30	29	20
	40	55	79	80	82	77	72	70	64	55	42
	50	56	82	87	89	91	87	87	79	77	72
	60	57	81	89	90	96	97	95	90	90	82
	70	57	91	93	95	96	96	98	97	96	87
	80	58	89	89	97	97	98	99	100	93	90
	90	56	91	94	96	97	96	97	100	100	100
	100	64	90	94	95	99	97	98	100	100	100

BÖLÜM 5. SONUÇ

Geleneksel olan yöntemlerin bizim için ne kadar zahmet verici ve özellikle büyük problemlerin karmaşıklığından çıkamayacağımız bir durumun içine girmemize sebep olabilir.

Bu çalışma özelinde, her bir aktivite için alternatif sayısı iki ile dört arasında değişebilmekte ve bu alternatiflerin çarpımı 1.119.744 çözüm arasından en uygun çözümü bulması beklenmektedir. Bu, özellikle çok sayıda potansiyel çözüm düşünüldüğünde zor bir görev olabilmektedir.

Metasezgisel yöntemler ise günümüzde yaygın olarak kullanılan önemli optimizasyon yaklaşımları arasında bulunmaktadır. Bu tür yöntemlerin önemli özelliği, farklı problemlere kolayca uyarlanabilme yetenekleri ve geleneksel yöntemlerle çözülemeyen sorunların etkin bir şekilde ele alınabilmesidir. Özellikle karmaşık problemlerde mutlak en iyi çözümü garanti etmese de kabul edilebilir süreler içinde optimuma yakın sonuçlar elde edebilme yetenekleri bulunmaktadır.

Çok sayıda algoritma literatürde geliştirilmiş olsa da uzun yıllardır yapılan çalışmalara rağmen, her türlü senaryoya uyarlanabilecek mükemmel bir yöntem henüz oluşturulamamıştır. Bu, araştırmacıların bu tarz problemlerinde en etkili sonuçları elde etmek için çabalarını sürdürmelerine neden olmaktadır.

Süre-maliyet optimizasyonu, proje planlama ve yönetimi alanında karşılaşılan zorluklardan birini temsil etmektedir ve büyük boyutlu matematiksel modeller içerdiğinden, metasezgisel yöntemlerle etkili bir şekilde çözülebilir.

Karınca Kolonisi Optimizasyonu Algoritması, paralel çalıştırmaya son derece elverişli bir yapıya sahiptir ve hızlı ve doğru sonuçlara ulaşabilmek için yüksek performansa ihtiyaç duyar.

Bu çalışmada, proje yönetiminde daha geleneksel olan CPM ve PERT yerine kritik yolları, kritik aktiviteleri ve proje tamamlanma süresini hesaplamak için KKA'ları

kullanan yenilikçi bir algoritma geliştirilmiştir. Bu algoritmanın uygulanması kolaydır ve karmaşıklıklarına bakılmaksızın tüm CPM ve PERT ağları için uygundur.

Ayrıca bu parametreleri hesaplamak için kullanılan tipik yöntemlerden çok daha hızlı ve basittir. Bu nedenle, bu algoritma herhangi bir kurum veya kuruluş tarafından, kendi kaynaklarını ve kısıtlamalarını ve kullanılan yöntemi dikkate alarak bir projeyi tamamlamak için gereken süreyi doğru bir şekilde belirlemek için kullanılabilecektir.

Elde edilen sonuçların optimal olduğu görüşmüştür. Bu algoritma proje yönetiminde büyük bir ilerleme sağlayacaktır, çünkü sadece zaman kazandırmakla kalmamakta, aynı zamanda proje tamamlanma süresini ve kritik aktiviteleri belirlemenin daha verimli ve doğru bir yolunu sağlamaktadır.

Bu araştırma, daha anlamlı ve verimli sonuçlar elde edilmesini ve önerilen algoritmanın başarısının daha net bir şekilde gözlemlenebilmesini sağlayacaktır. Ayrıca, iki yöntemin karşılaştırılması, her bir yaklaşımın avantaj ve dezavantajlarının belirlenmesine ve her birinin performansının daha kapsamlı bir değerlendirmesinin yapılmasına olanak tanıyacaktır. Bu hibrit yaklaşımın ve KKA algoritmasının çok yönlü etkinliğini daha iyi kavranmasını sağlayacak ve en verimli ve yeterli sürecin belirlenmesine yardımcı olacaktır.

Bu çalışmada kullanılan yöntem, problemin büyüklüğü ve karmaşıklığına göre başarı yüzdesi değişecektir. Bizim aktivite sayımız yani karar değişkenlerimiz 12 tane olup bu sayı azaldığında başarı yüzdesi artacak, bu sayı azaldığında ise doğruluk yüzdemiz azalacaktır.

Bu yöntemin, ne kadar büyük problemlerde doğruluk yüzdesinin tatmin edici boyutta olduğu ise başka bir araştırmanın konusu olacaktır.

Sonuç olarak metasezgisel yöntemlerin sonuçları en hızlı olarak optimize ettiği açıkça görülmektedir. Ancak yöntemlerin verdiği sonuçların her zaman en doğrusu olmadığı gözükmektedir. Bu yöntemleri seçerken verilen parametrelerin optimum değerlerine getirilip verilen sonucun optimize edilmesini beklemek gerekir.

Bu çalışmanın bir diğer kazançlarından biri de popülasyon ve iterasyon arasındaki bağlantıyı ortaya koymuştur. Bu bağlantıya göre bir problemin başlangıç parametreleri seçilirken bu çalışmadan faydalanabileceği açıkça görülmektedir.

KAYNAKLAR

Albayrak, G. ve Özdemir, İ. (2009) Yapı projelerinin süre-maliyet optimizasyonunda metasezgisel algoritma kullanımı. *Adıyaman Üniversitesi Mühendislik Bilimleri Dergisi*, Cilt: 3, Sayı: 5, 39 – 49.

Aminbakhsh, S. (2013). *Hybrid particle swarm optimization algorithm for obtaining Pareto front of discrete time–cost trade-off problem.* (M.Sc. Thesis), Middle East Technical University, Ankara, Turkey.

Bianchi, L., Dorigo, M., Gambardella, L. M., ve Gutjahr, W. J. (2006). Metaheuristics in stochastic combinatorial optimization: a survey. *Technical Report 08, IDSIA*, Manno, Switzerland.

Bilchev, G. ve Parmee, I. C. (1995). The ant colony metaphor for searching continuous design spaces. In T. Fogarty (Ed.), *Selected Papers from AISB Workshop on Evolutionary Computing, volume 993 of Lecture Notes in Computer Science*, s. 25–39. London, UK: Springer-Verlag.

Callegari, C., Szklo, A., & Schaeffer, R. (2018). Cost overruns and delays in energy megaprojects: How big is big enough? *Energy Policy*, 114, 211-220.

Chen, H. L. (2015). Performance measurement and the prediction of capital project failure. *International Journal of Project Management*, 33(6), 1393-1404.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms.* John Wiley & Sons.

Deneubourg, J.-L., Aron, S., Goss, S., ve Pasteels, J.-M. (1990). The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior*, 3(2), 159–168.

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms* (in Italian). (PhD thesis), Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.

Dorigo, M. ve Blum, C. (2005). Ant colony optimization theory: a survey. *Theoretical Computer Science*, 344(2-3), 243–278.

Dorigo, M. ve Gambardella, L. (1997). Ant Colony System: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1), 53–66.

Dorigo, M. ve Stützle, T. (2004). *Ant Colony Optimization*. The MIT Press, Cambridge, MA, USA.

Dorigo, M., Maniezzo, V., ve Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 26(1), 29–41.

Duncan, W. R. (1996). *A Guide to the Project Management Body of Knowledge*. Project Management Institute, USA, 3-27.

Durdyev, S. (2020). Review of construction journals on causes of project cost overruns. *Engineering, Construction and Architectural Management*, 28(4), 1241-1260. doi: https://doi.org/10.1108/ECAM-02-2020-0137.

Elbeltagi, E., Hegazy, T., Grierson, D. (2007). A modified shuffled frog-leaping optimization algorithm: Applications to project management. *Structure and Infrastructure Engineering*, 3(1), 53–60.

Fan, X., Luo, X., Yi, S., Yang, S., ve Zhang, H. (2003). Optimal path planning for mobile robots based on intensified ant colony optimization algorithm. In *Proceedings of the IEEE International Conference on Robotics, Intelligent Systems and Signal Processing (RISSP 2003)*, s. 131–136, Changsha, Hunan, China.

Feng, C.W., Liu, L., Burns, S.A. (1997). Using genetic algorithms to solve construction time-cost trade-off problems. *Journal of Computing in Civil Engineering*, 11(3), 184–189.

Fox, B., Xiang, W., ve Lee, H. P. (2007). Industrial applications of the ant colony optimization algorithm. *International Journal of Advanced Manufacturing Technology*, 31(7-8), 805–814.

Gordon, D. M. (2021), *Measuring collective behavior: an ecological approach*, Theory in Biosciences, 140, 353–360.

Goss, S., Aron, S., Deneubourg, J.-L., ve Pasteels, J.-M. (1989). Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 76(12), 579–581.

Hoos, H., Stützle, T. (2000). MAX-MIN Ant System, *Future Generation Computer Systems*, 16:8, 889-914.

Hafızoğlu, A. B. (2006). *Discrete time/cost trade-off problem in project scheduling.* (M.Sc. Thesis), Middle East Technical University, Turkey.

Haga, W. A. & Tim O'keefe. (14 Temmuz 2001). Crashing Pert Networks: A Simulation Approach. *4th International conference of the Academy of Business and Administrative Sciences Conference*, Quebec City, Canada.

Heravi, G., & Mohammadian, M. (2021). Investigating cost overruns and delay in urban construction projects in Iran. *International Journal of Construction Management*, 21(9), 958-968.

Huang, R. H. ve Yang, C. L. (2008). Ant colony system for job shop scheduling with time windows. *International Journal of Advanced Manufacturing Technology*, 39(1-2), 151–157.

Huo, T., Ren, H., Cai, W., Shen, G. Q., Liu, B., Zhu, M., & Wu, H. (2018). Measurement and dependence analysis of cost overruns in megatransport infrastructure projects: Case study in Hong Kong. *Journal of Construction Engineering and Management*, 144(3), 05018001.

Kennedy, J. ve Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, s. 1942–1948, Perth, Western Australia.

Korosec, P., Silc, J., Oblak, K., ve Kosel, F. (2007). The differential ant-stigmergy algorithm: an experimental evaluation and a real-world application. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC 2007)*, s. 157–164, Singapore.

Kutlu, N. T. (2001). Project planning techniques and a study on application of PERT technique in construction sector. *Dokuz Eylul University Journal of Social Sciences Institute,* 3 (2), 164-207.

Lee, D., ve Arditi, D. (2006). Automated statistical analysis in stochastic project scheduling simulation. *Journal of Construction Engineering and Management*, 132(3), 268-277.

Leu, S. S., & Yang, C. H. (1999). A genetic-algorithm-based resource-constrained construction scheduling system. *Construction Management & Economics*, 17(6), 767-776.

Meyer, W.L., Shaffer, L.R. (1965). Extending CPM for Multiform Project Time-Cost Curves. *Journal of Construction Division*, 91(1), 45-68.

Narayanan, A.S., Suribabu, C.R. (2014). Multiobjective optimization of construction project time cost-quality trade-off using differential evolution algorithm. *Journal of Civil Engineering*, 8(4), 375-392.

Özdemir, G. (2006). *The genetic algorithm methods used in resource constrained project scheduling problems and their comparison.* (MSc. Thesis), Ankara University, Institute of Social Sciences, Ankara.

Öztürk, A. (1984). *Operations Research*. Uludag University Press.

Pellerin, R., Perrier, N., & Berthaut, F. (2020). A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 280(2), 395-416.

Purnamadjaja, A. H. ve Russell, R. A. (2005). Pheromone communication in a robot swarm: necrophoric bee behaviour and its replication. *Robotica*, 23(6), 731–742.

Rahman, H. F., Chakrabortty, R. K., & Ryan, M. J. (2020). Memetic algorithm for solving resource constrained project scheduling problems. *Automation in Construction*, 111, 103052.

Rosenau, M. D. ve Githens, G. D. (2005), *Successful Project Management: a Step-by-step Approach with Practical Examples.* 4th Ed., Wiley, Hoboken, N.J.

Schoonderwoerd, R., Holl, O., Bruten, J., ve Rothkrantz, L. (1996). Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, 5, 169–207.

Siemens N. (1971). A Simple CPM Time-Cost Trade off Algorithm. *Management Science*, 17(6), 354–363.

Sim, K. M. and Sun, W. H. (2003). Ant colony optimization for routing and load-balancing: survey and new directions. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 33(5), 560–572.

Sivri, G. (2001). Monitoring and controlling progress in construction projects and an application for project information management system. *Master Thesis, Istanbul Technical University Institute of Social Sciences, Istanbul*, 1-48.

Sönmez, R., Bettemir, O.H. (2012). A hybrid genetic algorithm for the discrete time-cost trade-off problem. *Expert Systems with Applications*, 39(13), 11428-11434.

Strauß B. Bullnheimer R. Hartl. (1997). *A new rank based version of the Ant System. A computational study*. WU Working Papers, Vienna.

Trietsch, D., Mazmanyan, L., Gevorgyan, L., Baker, K.R. (2012). Modeling activity times by the Parkinson distribution with a lognormal core: theory and validation. *European Journal of Operational Research*, 216, 386–396.

Tsutsui, N. D., Suarez, A. V., Holway, D. A., and Case, T. J. (2001). Relationships among native and introduced populations of the argentine ant (linepithema humile) and the source of introduced populations. *Molecular Ecology*, 10(9), 2151–2161.

Tsutsui, S., Pelikan, M., and Ghosh, A. (2005). Performance of aggregation pheromone system on unimodal and multimodal problems. In *Proceedings of the 2005 Congress on Evolutionary Computation (CEC 2005)*, volume 1, pages 880–887, Edinburgh, Scotland.

Vega-Velázquez, M. Á., García-Nájera, A., & Cervantes, H. (2018). A survey on the software project scheduling problem. *International Journal of Production Economics*, 202, 145-161.

Walker, M. (2009). *Ant mega-colony takes over world*. The Two-Way Publishing.

Wang, J., Osagie, E., Thulasiraman, P., and Thulasiram, R. K. (2009). HOPNET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network. *Ad Hoc Networks*, 7(4), 690–705.

Yıldız, S. (2001). *Resource Leveling and Earned Value Analysis in Project Management: an Application in Construction Sector.* (MSc. Thesis), Baskent University Institute of Social Sciences, Ankara.

Zareei, S. (2018). Project scheduling for constructing biogas plant using critical path method. *Renewable and Sustainable Energy Reviews*, 81, 756-759.

Zheng, D., Ng, S., Kumaraswamy, M. (2005). Applying pareto ranking and niche formation to genetic algorithm-based multiobjective time–cost optimization. *Journal of Construction Engineering and Management*, 131(1), 81–91.

Yılmaz, Ş., (2008), *Çok depolu araç rotalama probleminin karınca kolonisi optimizasyonu ile modellenmesi ve bir çözüm önerisi,* (Yüksek Lisans Tezi), Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.