

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(DOKTORA TEZİ)

**ÇOK ROBOTLU KAPSAMA VE RANDEVU İÇİN
DAĞITIK ALGORİTMALAR**

Deniz ÖZSOYELLER

Tez Danışmanı : Prof. Dr. Kayhan ERCİYEŞ

İkinci Danışmanı : Doç. Dr. Volkan İşler

Uluslararası Bilgisayar Anabilim Dalı

Bilim Dalı Kodu : 619.03.03

Sunuş Tarihi : 12.12.2014

Bornova-İZMİR

2014

Deniz Özsoyeller tarafından **Doktora tezi** olarak sunulan “**Çok Robotlu Kapsama ve Randevu için Dağıtık Algoritmalar**” başlıklı bu çalışma E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliği ile E.Ü. Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi’nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve **12.12.2014** tarihinde yapılan tez savunma sınavında aday **oybirliği** ile başarılı bulunmuştur.

Jüri Üyeleri:

İmza

Jüri Başkanı : Prof. Dr. Kayhan Erciyeş

.....

Raportör Üye : Prof. Dr. Mustafa Gündüzalp

.....

Üye : Prof. Dr. Turhan Tunalı

.....

Üye : Doç. Dr. Muhammet Cinsdikici

.....

Üye : Prof. Dr. Mehmet Emin Dalkılıç

.....

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca **Doktora Tezi** olarak sunduğum "**Çok Robotlu Randevu ve Kapsama için Dağıtık Algoritmalar**" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

12/12/2014

Deniz Ozsoyeller

ÖZET**ÇOK ROBOTLU RANDEVU VE KAPSAMA İÇİN
DAĞITIK ALGORİTMALAR**

ÖZSOYELLER, Deniz

Doktora Tezi, Uluslararası Bilgisayar Anabilim Dalı

Tez Danışmanı: Prof. Dr. Kayhan ERCİYEŞ

İkinci Danışmanı: Doç. Dr. Volkan İşler

Aralık 2014, 148 sayfa

Mobil robotikteki gelişmeler, düşük maliyetli, güvenilir robotların karmaşık ortamlarda dolaşmasını ve karmaşık işler gerçekleştirmesini mümkün hale getirmiştir. Bir grup robotla bir işin etkin bir şekilde yürütülebilmesi için etkili robot stratejileri tasarlamak önemlidir. Bu tezde temel bir problem olan, mobil robotlarla randevu arama problemini çalışılmıştır.

Randevu arama problemi bir ortamda birbirlerinden farklı konumlarda bulunan robotların en kısa zamanda nasıl buluşabileceği ile ilgilenir. Arama ve kurtarma bu problemin önemli uygulamalarından biridir. Kurtarıcılar (robotlar) afet ve patlama gibi durumlarda kurbanları veya hayatta kalanları arayabilir. Uydu alıcılarının olmadığı, gücün sınırlı olduğu veya ortamın geometrisinin elverişli olmadığı durumlarda robotların başarılı iletişim kurmak için aralarında görüş hattının sağlanması gerekir. Bu gibi bir durumda robotların iletişim kurmak için birbirlerinin yanında olması gerekir, yani buluşmaya ihtiyaçları vardır. Böyle durumlar için, randevu arama probleminin diğer uygulamaları da robotlar arasında ağ formasyonu, çoklu robot keşif ve haritalamadır. Bu tezde problemin simetrik versiyonu çalışılmıştır. Bu versiyonda, her robot aynı stratejiyi çalıştırır, çünkü önceden buluşup arama sırasında kullanacakları stratejilere karar verme şansları yoktur.

İlk olarak problem doğru üzerinde bulunan, aralarındaki başlangıç uzaklığını bilmeyen aynı özelliklere sahip iki robotla çalışılmış ve simetrik bir randevu arama stratejisi sunulmuştur. Robotların hem senkron olduğu, yani aramaya aynı anda başladıkları hem de asenkron oldukları yani aramaya farklı zamanlarda başladığı durumu düşünülmüştür. İkinci olarak, aynı problem engelli ve engelsiz düzlemsel

ortamlarda iki ve ikiden çok robot ile çalışılmıştır. Bu ortamlar için simetrik bir randevu arama stratejisi tasarlanmıştır. Bu kısımda ayrıca engelli bir ortamın bir robot ile keşfi için de bir algoritma sağlanmıştır. Hem doğru hem de düzlemsel ortamlar için tasarlanan simetrik randevu stratejilerinin performansı teorik olarak ispatlanmış ve simülasyonlarla ve deneylerle değerlendirilmiştir.

Anahtar Sözcükler: Doğrusal arama problemi, arama problemi, simetrik randevu, robot randevu, robot keşif ve kapsama.

ABSTRACT**DISTRIBUTED ALGORITHMS FOR MULTI ROBOT
RENDEZVOUS AND COVERAGE**

ÖZSOYELLER, Deniz

PhD in Department of International Computer

Supervisor: Prof. Dr. Kayhan ERCİYEŞ

Co-Supervisor: Assoc. Prof. Dr. Volkan İŞLER

December 2014, 148 pages

The advancements in mobile robotics have made the low-cost and robust robots to navigate in complex environments and perform complex tasks. To execute a task efficiently with a group of robots, it is significant to design efficient robot strategies. In this thesis, the rendezvous search problem which is a fundamental problem in robotics is studied.

The rendezvous search problem considers how the robots that are located in different positions in an environment can meet as quickly as possible. Search and rescue tasks are one of the important applications of this problem, rescuers (robots) can search for the victims or the survivors who are in dangerous situations such as disasters and explosions. When there are no satellite receivers, the power is limited and the geometry of the environment is inconvenient, the robots should be in the line of sight of each other to communicate successfully. In such situations, robots should be close to each other to communicate, so they need to rendezvous. Other applications of rendezvous search for such situations are network formation and multi robot exploration and mapping. In this thesis, the symmetric version of this problem is studied. In this version, each robot executes the same strategy, because they do not have a chance to meet in advance to decide on the strategies they would perform during the search.

First, the rendezvous search problem is studied on the line for two identical robots that do not know the initial distance to each other and a symmetric rendezvous strategy is presented. Both the cases that is when the robots move synchronously in which they start searching at the same time and when the robots move asynchronously in which they start searching at different times are

considered. Second, the same problem is studied in planar environments with and without obstacles with two and multiple robots. A symmetric rendezvous strategy for these environments is studied. In this part, an algorithm for the exploration of an environment with obstacles by a robot is also provided. The performances of the symmetric rendezvous strategies are theoretically analyzed and their performances are also evaluated in simulations and experiments.

Keywords: Linear search problem, rendezvous search, symmetric rendezvous, robot rendezvous, robot exploration and coverage.

TEŞEKKÜR

Doktora çalışmalarım boyunca yanımda olan herkese çok teşekkür ederim:

Değerli danışman hocalarım Kayhan Erciyeş'e ve Volkan İşler'e bana kendileriyle çalışma fırsatı verdikleri, bana güvendikleri ve inandıkları ve bana verdikleri emek için. Doktora çalışmalarım boyunca verdikleri destek ve yönlendirmeleri için.

Andrew Beveridge Hocama değerli fikirleri için. Kendisiyle çalışma fırsatım oldu ve kendisinden çok şeyler öğrendim.

UMN, RSN labından arkadaşlarım Onur, Pratap, Deepak ve Nikhil'e ve labdaki diğer tüm arkadaşlarıma, yaptığımız tüm tartışmalar, çalışmalar, arkadaşlığınız ve yardımlarınız için. İyi ki sizleri tanıdım ve sizin harika arkadaşlarla çalışma fırsatım oldu.

Doktora jürimdeki hocalarım Mustafa Gündüzalp ve Muhammet Cinsdikici'e verdikleri değerli fikirleri için.

Turhan Tunalı hocama akademik kariyerim boyunca bana verdiği emek ve destekleri için.

İzmir Üniversitesindeki tüm arkadaşlarıma, özellikle bölüm arkadaşlarım Deniz, Esra, Ebru, Emre, Gökmen, Yiğit, Tayfun, Hazar, Züleyha, Senem ve Samet'e bana her zaman moral verdikleri için.

Anneannem ve Dedeme bana olan sonsuz sevgileri için.

Annem ve Babama bana olan sonsuz sevgileri, her zaman yanımda oldukları, destekledikleri ve hayatımdaki mutluluk kaynakları oldukları için. İyi ki sizin gibi bir anne ve babam var. Sizens ne yapardım bilemiyorum.

İÇİNDEKİLER

Sayfa

ÖZET	vii
ABSTRACT	ix
TEŞEKKÜR	xi
ŞEKİLLER DİZİNİ	xvii
YABANCI KELİMELELER DİZİNİ	xxii
1 GİRİŞ	1
2 Tez Araştırmasına Giriş	5
2.1 Randevu Arama Problemi Nedir?.....	5
2.2 Randevu Arama Problemlerinin Öncelikli Versiyonları	6
2.3 Sık Meydana Gelen Randevu Arama Stratejileri	7
2.4 Randevuya karşı Randevu Arama	8
2.5 Randevu Arama Probleminin Performansının Ölçülmesi	9
2.6 Katkılarımız.....	12
2.7 Taslak	14
3 İLGİLİ ÇALIŞMALAR	15
3.1 Daire/Halka Üzerinde Randevu Arama Problemi	15
3.2 Çizge/Ağ Üzerinde Randevu Arama Problemi	17
3.3 Toplanma ve Birleşme Problemi	19

İÇİNDEKİLER (devam)

Sayfa

4 DOĞRU ÜZERİNDE SİMETRİK RANDEVU ARAMA	23
4.1 İlgili Çalışmalar	25
4.2 Doğru Üzerinde Simetrik Randevu	28
4.2.1 Lost Cow Problemi	28
4.3 Senkron Durum	29
4.3.1 Simetrik randevu arama algoritması	29
4.3.2 Analiz	33
4.3.3 Ek analiz hesaplamaları	43
4.3.4 SR Algoritmasının beklenen round sayısı.....	49
4.3.5 Simülasyonlar.....	51
4.3.6 Uzantılar	56
4.3.7 Deneyler	57
4.4 Asenkron Durum	62
4.4.1 Simetrik randevu algoritması	62
4.4.2 Analiz	66
4.4.3 Simülasyonlar.....	74
4.5 Sonuçlar	80
5 DÜZLEMSEL ORTAMLARDA SİMETRİK RANDEVU ARAMA	81
5.1 İlgili Çalışmalar	84

İÇİNDEKİLER (devam)

	<u>Sayfa</u>
5.1.1 Keşif problemi	87
5.2 Engelsiz Ortamlar.....	90
5.2.1 Düzlemsel ortamlarda simetrik randevu arama	90
5.2.2 Analiz	92
5.2.3 SRP Algoritmasının Beklenen Round Sayısı	97
5.2.4 Düzlemsel ortamlarda çok robotlu simetrik randevu arama	98
5.2.5 Simülasyonlar	103
5.3 Engelli Ortamlar	112
5.3.1 Simetrik randevu arama algoritması	112
5.3.2 Coverage algoritması	113
5.3.3 Analiz	115
5.3.4 SRP _o Algoritmasının Beklenen Round Sayısı	124
5.3.5 Engelli ortamlarda çok robotlu simetrik randevu arama	124
5.3.6 Simülasyon deneyleri	127
5.4 Sonuçlar	131

İÇİNDEKİLER (devam)

	<u>Sayfa</u>
6 SONUÇ VE TARTIŞMA.....	133
6.1 Gelecek Araştırmalar	135
KAYNAKLAR DİZİNİ	137
ÖZGEÇMİŞ	144

ŞEKİLLER DİZİNİ

<u>Şekil</u>	<u>Sayfa</u>
1.1 Mobil Robotlar, Sol: Unimate. Üst: Packbot Keşif robotu Yamauchi (2004).....	2
1.2 Mobil Robotlar, Sol: Pioneer	3
4.1 Gürsel Aksel Caddesi üzerinde bulunan, birbirlerinin konumundan habersiz iki arkadaş en kısa zamanda buluşmak istiyor.	24
4.2 Round i'daki davranışı.....	36
4.3 Algoritma SR'in 100,000 simülasyondaki ortalama performansı. Burada, ϵ aramanın başında rasgele olarak seçilmiştir.	51
4.4 Algoritmamız SR'in simülasyonlardaki performansı.	52
4.5 Daha az ortalama round sayısı daha az mesafe gidileceğini ima etmez.....	53
4.6 SR'in birinci uzantısı: Algoritma SR, doğru parçası ve yarı doğru gibi sınırlandırılmış ortamlara genişletilebilir. (Lemma 11).	57
4.7 SR'in ikinci uzantısı: Algoritma SR, ayrıca kapalı yola benzer ortamlara da genişletilebilir (Lemma 12).	57
4.8 Üzerlerinde Asus EEE PC laptop olan, düz bir koridor üzerine yerleştirilmiş olan iki iRobot Create robotunu gösteren birinci ve ikinci deneyin kurulumu.	58
4.9 Düz doğru deneyleri.	62
4.10 Sınır olan düz doğru deneyleri. Deneyler düz bir koridor üzerinde yapılır. SLEB: Robot-1 veya Robot-2'e yakın olan bir sınır duvarı vardır.	63

ŞEKİLLER DİZİNİ (devam)

Şekil

Sayfa

4.11 Sınır olan düz doğru deneyleri. Deneyler doğru bir koridor üzerinde yapılır. SLEB-2b: Hem Robot-1 hem de Robot-2'e yakın olan bir sınır duvarı vardır. 64

4.12 Kapalı eğri deneyleri. Deneyler dikdörtgen şeklindeki bir koridor üzerinde yapılır. 65

4.13 Robotların ASR algoritmasını çalıştırdıklarında mümkün olan yolculukları 68

4.14 Robot-1 round i^* 'da ve robot-2 round j^* da iken, $x = 2d$ 'nin sağ tarafında randevuyu engelleyecek en kötü durum senaryosunda robotların yolculukları 69

4.15 $\Delta = 0$ iken, başarısız bir i^* da ve j^* da gelen olaylara göre robot konfigürasyonları 75

4.16 $1.35 \leq r \leq 1.85$ için t 'deki değişime göre $d = 250$ iken, Algoritma SR'in simülasyonlarla ortalama mesafe kompetitif oranı. 77

4.17 $1.35 \leq r \leq 1.85$ için, d 'deki değişime göre $t = 10$ iken, Algoritma SR'in simülasyonlarla ortalama mesafe kompetitif oranı. 78

4.18 $r = 1.195$ için d 'deki ve t 'deki değişime göre, Algoritma SR'in simülasyonlarla ortalama mesafe kompetitif oranı. 78

4.19 $r = 1.55$ için d 'deki ve t 'deki değişime göre, Algoritma ASR'in simülasyonlarla ortalama mesafe kompetitif oranı. 79

4.20 $1.2 \leq r \leq 1.55$ için, t 'deki değişime göre $d = 125$ iken, ϵ olmadığı durumda Algoritma ASR'in simülasyonlarla ortalama mesafe kompetitif oranı. 79

ŞEKİLLER DİZİNİ (devam)

Şekil

Sayfa

- 5.1 Stratejiye bakış: Her robot beklemek veya hareket etmeye karar vermek için para atar. Eğer tura atarsa, robot kendi başlangıç noktasının merkez olduğu içiçe daireler üzerinde hareket eder. Eğer yazı atarsa, şuanki round i 'da hareket etmesi gerektiği mesafe kadar bekler. 91
- 5.2 Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$ 106
- 5.3 Düzlemsel ortamlarda iki robotla simülasyonlar. $r = 1.225$. ÜST: Kominikasyon alanı R 'daki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. ALT: Kominikasyon alanı R 'daki değişime göre çeşitli d değerleri için ortalama zaman kompetitif oran. 107
- 5.4 Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$ and $r = 1.225$. Yöndeki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. Robotların oryantasyonlarının algoritma başında aynı olduğunu varsayıyoruz. 108
- 5.5 Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$ and $r = 1.225$. d 'deki değişime göre çeşitli t değerleri için ortalama mesafe kompetitif oran. Ortamdaki bir robot randevu algoritmasını diğer robottan t zaman sonra çalıştırmaya başlar. 108
- 5.6 Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $R = 1.0$. ÜST: N 'deki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. ALT: N 'deki değişime göre çeşitli d değerleri için ortalama round sayısı. 109
- 5.7 Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $N = 8$. R 'daki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. 110
- 5.8 Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $R = 1.0$. d 'deki değişime göre çeşitli N değerleri için randevu gerçekleştiğinde oluşmuş olan küme sayısı. 110

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
5.9 (a) Kapsama algoritmasının 3×3 grid üzerinde çalışmasına bir örnek. Grid hücreleri A-I olarak numaralandırılmıştır. (b) DFS Tree.	117
5.10 Farklı <i>C-obstacles</i> ile iki bölgeye ayrılan hücre örnekleri.	117
5.11 Bir quadrant örneği. Bölgeleri I-IV arasında numaralandırılmıştır. Quadrantın orijini <i>P</i> engelini köşesine sabitlenmiştir.	117
5.12 <i>P</i> engelini konkav köşesi ve <i>C-obstacle</i> 'da karşılık gelen köşesi.	117
5.13 <i>P</i> engelini konveks köşesi ve onun <i>C-obstacle</i> 'da karşılık gelen köşesi.	117
5.14 P_1 ve P_2 hücreyi iki (açık gölgelendirilmiş) bölgeye ayıran <i>C-obstacles</i> 'dir. P_3 hücrede üçüncü (koyu gölgelendirilmiş) bölgeyi yaratan <i>C-obstacle</i> 'dir.....	119
5.15 Örnek kenarların y-axis üzerindeki projeksiyonu.	121
5.16 p_{xi} , b_1 ve b_5 arasındaki <i>C-obstacle</i> kenarlarının project edildiği x-axis'i üzerindeki doğru parçalarını gösterir.	122
5.17 1. Deney Ortamı.	127
5.18 2. Deney Ortamı	128
5.19 ÜST: 3. Deney Ortamı, ALT: Robotun 3. Ortamdaki hareketi	129
5.20 Engelli ortamlarda, SRP_o algoritmasını çalıştıran iki robotla simülasyon deneyleri. Farklı ortamlarda, d' 'deki değişime göre; ÜST: randevu için ortalama adım sayısı, ALT: randevu için ortalama simülasyon zamanı.	130

ŞEKİLLER DİZİNİ (devam)ŞekilSayfa

5.21 Engelli ortamlarda CA kapsama algoritmasını çalıştıran iki robotla simülasyon deneyleri. Farklı ortamlarda, grid boyutundaki değişime göre, gridin tam kapsanması için gereken ortalama adım sayısı. 131

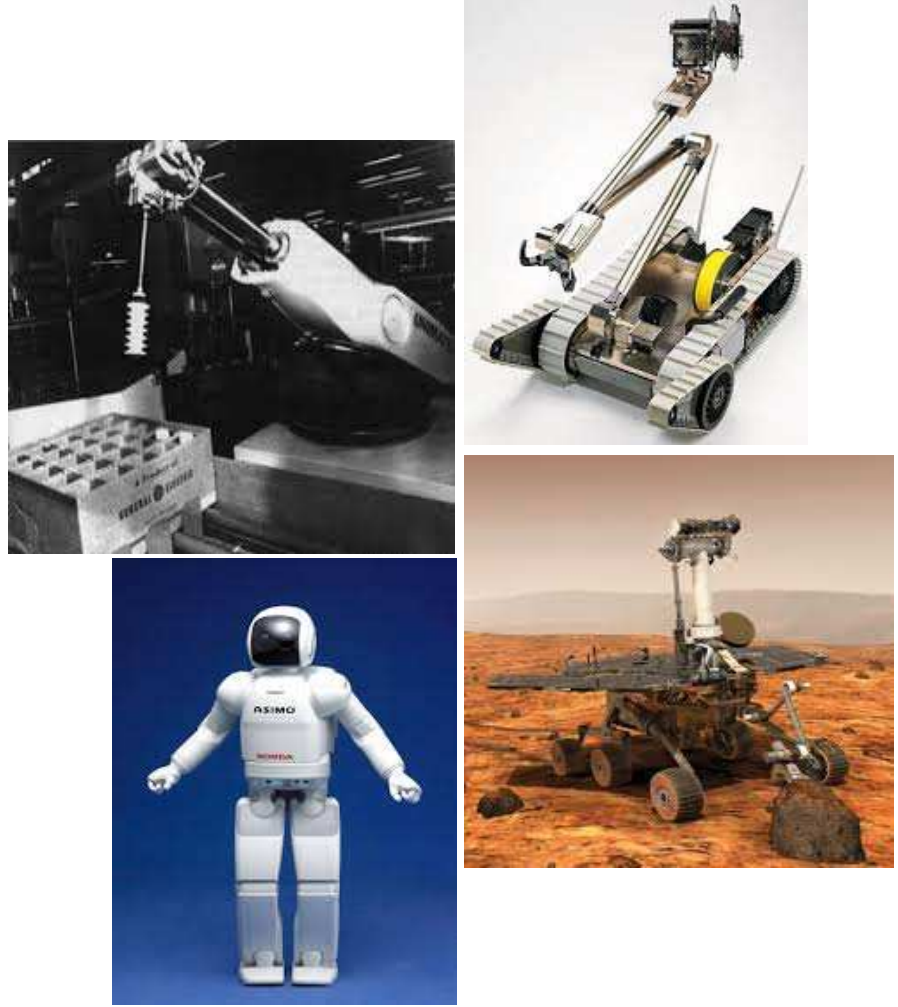
YABANCI TERİMLER DİZİNİ

<u>Yabancı Terim</u>	<u>Açıklama</u>
<i>Communication</i>	Komünikasyon, İletişim.
<i>Configuration</i>	Konfigürasyon, Düzenleniş.
<i>Coverage</i>	Kapsama.
<i>Distance Competitive Ratio</i>	Mesafe Kompetitif Oran.
<i>Expansion Radius</i>	Genişleme Yarıçapı.
<i>Expected</i>	Beklenen.
<i>Experiment</i>	Deney.
<i>Graph</i>	Çizge.
<i>Itinerary</i>	Yolculuk Programı.
<i>LostCow Problem</i>	Kaybolan İnek Problemi.
<i>Offline</i>	Çevrimdışı.
<i>Rendezvous</i>	Randevu.
<i>Round</i>	Raund.
<i>Stage</i>	Aşama.
<i>Time Competitive Ratio</i>	Zaman Kompetitif Oran.

GİRİŞ

Tam otonom robotlar, yirminci yüzyılın ikinci yarısında ortaya çıkmıştır. İlk sayısal işlem gören ve programlanabilir robot, Unimate (Şekil 1.1), 1961'de General Motors'da bir döküm makinesinden sıcak metal parçalarını kaldırmak ve onları yığmak için kurulmuştur. O zamandan beri robotik, sanayi gelişiminin önemli bir unsuru haline gelmiştir. Fabrikalarda kullanımından harekete geçilerek, ilk robotik araştırmaları kinematik, hareket ettirme, tutma gibi temel kontrol problemlerine odaklanmıştır. Sonrasında, *mobil robotik* adı verilen yeni bir çağ başlamıştır. Mobil robotikteki gelişmeler ilk önce uzay görevleri ve askeri uygulamalardaki ihtiyaçlar doğrultusunda ilerlemiştir. Bu kategorideki dünyaca bilinen mobil robotlardan bazıları (Şekil 1.1), NASA tarafından geliştirilen Mars keşif aracı Spirit Squyres (2005), iRobot tarafından geliştirilen bomba imha edici Packbot Yamauchi (2004) ve Honda tarafından geliştirilen insansı robot Asimo'dur Hirose and Ogawa (2007). Günümüzde, ticari ve endüstriyel robotlar yaygın olarak, işleri daha ucuz ve insanlardan daha güvenilir ve düzgün bir şekilde yapmak için kullanılmaktadır.

İlk robotik araştırmaları tek robotlu sistemlere odaklanmıştır. Tek robotlu sistemler tasarlamak karmaşık ve bu nedenle de pahalıdır. Dahası, bu gibi sistemlerin yetenekleri sınırlıdır. Çok robotlu sistemlerin geliştirilmesinde tetikleyici faktör, düşük maliyetli fakat sağlam, birlikte karmaşık işler yapabilen robot gruplarına ihtiyaç olması olmuştur. Robot sürülerinin daha karmaşık bireysel robotlar üzerinde çeşitli avantajları vardır. Bu tür sistemleri inşa etmek, tek bir büyük robot inşa etmekten daha ucuz ve kolaydır. Şimdi, kısaca robot sürülerinin yeteneklerini, tek bir robotun yetenekleri ile karşılaştıralım. Tek bir birim olarak çok robotlu sistemler tek robot için mümkün olmayan daha karmaşık davranışları gerçekleştirebilir. Robot sürülerinin bireysel robotlardan daha fazla hataya dayanıklılığı vardır, çünkü bu sistemlerde çalışan algoritmalar robotların birbirlerine bağımlı olmalarını gerektirmez. Tek bir modül başarısız olursa, sürünün geri kalanı bu modül asla varolmamış gibi kendi işlerini yapmaya devam edebilir. Dahası, hatalı robotlar yenileriyle yer değiştirilebilir. Ancak, tek bir robot, eğer kritik bir bileşeninde bir hata varsa görevine devam edemez. Bu bileşenlerin sağlam olması gerektiğinden bu, bu gibi sistemlerin maliyetini arttırır. Çok robotlu sistemlerin bir özelliği de ölçeklenebilir olmalarıdır. Yani etkinliği arttırmak için yapılması gereken sadece sürüye daha fazla robot eklemektir. Öte yandan, tek bir robot sisteminin etkinliğinin iyileştirilmesi her zaman belirgin değildir. Dahası, çok robotlu sistemlerdeki robotlar tekrar programlanarak farklı görevler için tek-



Şekil 1.1: Mobil Robotlar, **Sol:** Unimate. **Üst:** Packbot Keşif robotu Yamachi (2004). **Sağ:** Mars Spirit Squyres (2005). **Alt:** Asimo Hirose and Ogawa (2007).



Şekil 1.2: Mobil Robotlar, **Sol:** MobileRobots Pioneer res. **Sağ:** iRobot Create Squyres (2005). **Alt:** WillowGarage PR2 pr2.

rar kullanılabilir. Bu özellikleri, çok robotlu sistemlerin çok çeşitli ortamlarda kullanılmasına olanak sağlar. Örneğin, çok robotlu sistemler, askeri operasyonlar gibi insanların risk altında olabileceği durumlarda, uzay keşif görevlerinde, zehirli madde dökülmelerinin temizlenmesinde, yangın ile mücadelede ve arama ve kurtarma gibi görevlerde kullanılabilir Cohen and Peleg (2006). Çok robotlu sistemler tek robotlu sistemlerden daha hata toleranslı olduğu için bu gibi hayati durumlarda, çok robotlu sistemin görevi başarılı bir şekilde yerine getirme şansı tek robota göre daha fazladır.

Bu robotların yetenekleri her geçen gün daha karmaşık hale gelirken maliyetleri de düşmektedir. İlk ticari başarıya sahip ev robotu olan iRobot Roomba temizleme robotu daha düşük maliyetli, sağlam robotik platformları için verilebilecek önemli bir örnektir. Bundan dolayı, daha düşük maliyetli, karmaşık işler yapabilen sağlam mobil robotları satın almak artık mümkündür. Robotik araştırmalarında ve eğitimde yaygın olarak kullanılan standard platformlardan bazıları iRobot Create (Roomba'nın programlanabilir versiyonu), MobileRobots Pioneer res ve WillowGarage PR2 pr2'dir (Şekil 1).

Bu tezde, çok mobil robotlu sistemlerde meydana gelebilecek iki temel problemi çalışıyoruz. Bu problemler randevu arama ve keşif problemidir. Randevu arama problemi birbirlerinin konumlarını bilmeyen iki veya ikiden çok robotun en kısa zamanda buluşabilmek için nasıl bir strateji uygulamaları gerektiğini sorgular. Örneğin, iki kişinin bir alışveriş merkezinde ayrıldıklarını fark ettiklerini ve tekrar biraraya gelmek istediklerini veya bir uçaktan paraşütle ayrı zamanlarda atlayan iki arkadaşın buluşmak zorunda olduklarını düşünün. Bu ilginç problemin gerçek hayattaki problemlerle açık bağlantısı

olduđu gibi, ayrıca robotik arama ve kurtarma, ađ formasyonu, çoklu robot keşif ve haritalama gibi çeşitli uygulamaları da vardır. Robotik arama ve kurtarmada, kurtarıcılar (robotlar) afet ve patlama gibi durumlarda kurbanları veya hayatta kalanları arayabilir. Robotlar ayrıca tehlikeli veya insanların ulaşmasının mümkün olamayacağı mayın tarlaları, zehirli atıkların olduđu bölgeler veya başka gezegenler gibi bilinmeyen ortamları keşfetmek veya ortamın haritasını çıkarmak için de görevlendirilebilir. Bir olayın tespit edilmesi durumunda, alanı birlikte keşfetmek için buluşabilirler. Birçok robotun geniş bir alanı gözetlemek için görevlendirildiđini düşünelim. Bir olay tespit edildiğinde bilgiyi en kısa zamanda yaymak için ađ oluşturmaları gerekebilir. Robotlar sınırlı iletişim alanına sahip ve ortamda farklı konumlarda dağılmış olduđuanda, ađ oluşturma problemi randevu arama problemlerine yakından ilişkili hale gelir. Biz randevu arama problemini, ilk olarak doğru üzerinde ve ikinci olarak da engelli ve engelsiz düzlemsel ortamlarda çalışıyoruz. Robotların kendi lokasyonu, diđer robot/robotlarla ve ortamla ilgili hiçbir bilgisi olmadığını düşünüyoruz.

Sonraki bölümde, randevu arama probleminin öncelikli versiyonlarını, bu problemin literatürde olan randevu probleminden farklarını, randevu arama stratejilerinin performansının nasıl hesaplandığını ve bu tezdeki katkılarımızı sunuyoruz.

Tez Araştırmasına Giriş

2.1. Randevu Arama Problemi Nedir?

İki veya daha fazla ajanın bilinen bir arama bölgesine rasgele yerleştirildiğini düşünelim. **Randevu Arama Problemi**, bu ajanların bölge üzerinde en kısa zamanda başarılı bir şekilde buluşabilmesi için nasıl bir strateji yürütmesi gerektiğini sorgular. Resmi olmayarak, ilk Alpern (1976) tarafından yaklaşık 38 yıl kadar önce tanıtılsa da, bu problem Anderson ve Weber'in Anderson and Weber (1990) "*discrete location rendezvous*" üzerine olan öncü makalesi ve Alpern'in Alpern (1995) genel sürekli formülüzasyonundan (*Ing. general continuous formalization*) sonra ilgi görmeye başlamıştır. O zamandan beri bu alan hızlı bir şekilde genişlemiştir.

Randevu arama problemlerini değişik alanlar üzerinde formüle etmek mümkündür. Örneğin randevu arama problemleri, iki veya daha fazla ajanın, belirli bir düzlemsel ortamda, doğru, daire (*vb. halka*) veya bir çizge üzerinde randevusunu içerebilir.

Bu problem, bilim insanları tarafından analiz edilmeden çok önce gerçek dünyada hayvanların ve insanların karşı karşıya geldiği bir problemidir. Örneğin, periyodik olarak yeniden toplaması gereken (örneğin, yiyecekler hakkında bilgi paylaşmak için) göçmen veya denizaltı hayvanları gelişmiş randevu stratejilerine ihtiyaç duyabilir. Eş bulma şansı çok nadir olan bazı türler için, uygun bir eş bulma basit olmayan bir randevu problemidir. Uzun süre konvoy halinde denizde yolculuk eden gemiler, fırtına nedeniyle birbirlerinden ayrılmış olabilirler ve yeniden bir araya gelmek isteyebilirler. Eğer kıyı boyunca seyahat ediyor olsalardı, o zaman kıyıda belirli randevu noktalarına karar vermiş olabilirlerdi. Fakat kıyıdan çok uzak mesafede, böyle bir yaklaşım pratik değildir. Benzeri şekilde, askeriye'nin yabancı topraklarda randevu için her zaman önceden belirlemiş olduğu protokolleri vardır. Ayrıca keşifte bulunmak amacıyla bir bölgeyi dolaşan araştırmacıların da randevu için her zaman protokolleri vardır. Bir alışveriş merkezine bir arkadaşınızla gittiğinizi ve arkadaşınızla ayrı düştüğünü düşünelim. Arkadaşınızla (onun da sizi aradığını düşünerek) en kısa zamanda nasıl buluşabileceğiniz insanların karşı karşıya geldiği bir randevu problemidir. Veya bir ailenin büyük bir hayvanat bahçesini dolaşırken ayrı düştüğünüzü düşünelim. Aile bireylerinin en kısa zamanda nasıl tekrar buluşabileceği yine insanların karşılaştığı randevu

problemlerine bir örnektir. Sonraki bölümde randevu arama problemlerinin öncelikli versiyonlarından bahsediyoruz.

2.2. Randevu Arama Problemlerinin Öncelikli Versiyonları

Randevu arama problemlerinin, oyuncuların arama sırasında kullanacaklara stratejilere karar vermek için oyun öncesinde buluşup buluşmamalarına göre, iki öncelikli versiyonu vardır. Bunlardan biri olan **asimetrik randevu arama**, ajanlar arama sırasındaki rolleri üzerinde önceden anlaşılır. Eğer oyun öncesinde buluşurlarsa, o zaman farklı stratejiler seçebilirler. Mesela bir ajan diğer ajanı bekleyebilir ("Wait for Mommy" stratejisinde olduğu gibi, çocuk beklerken anne kapsamlı bir arama yapabilir). Asimetrik randevudan farklı olarak, **simetrik randevu arama**, ajanların önceden buluşup aralarında rollerine karar verme şansları yoktur, o yüzden de aynı stratejileri uygulamak zorundadırlar. Literatürde iki versiyon da doğru, daire (*vb.* halka), çizge ve geometrik ortamlar gibi çok çeşitli ortamlarda çalışılmıştır. Randevu problemlerinin simetrik versiyonu robot uygulamaları için çekicidir, çünkü her robota farklı bir strateji uygulama gereksinimini ortadan kaldırır. Bu nedenle çalışmalarımızda randevu problemlerinin simetrik versiyonuna odaklanıyoruz.

Alpern Alpern and Gal (2003) tarafından önerilen *Astronot Problemi* ve *Telefon Problemi* simetrik randevu problemlerine verilebilecek iki örnektir. **Astronot problemi**nde, iki astronot küresel bir alan üzerinde birbirlerini göremeyecekleri kadar uzak mesafede bulunmaktadır. Alanın, uzayda sabit bir yönü veya dönme eksenini yoktur, yani astronotlar koordinasyon için gerekli olan pozisyon veya yön bilgilerine sahip değildirler. Problem; her iki astronotun birim yürüme hızında ilerlediğini bilerek, beklenen randevu zamanı T 'yi en aza indirmek için astronotların nasıl hareket edeceklerine karar vermektir. **Telefon problemi** şu şekildedir; her iki odada rasgele dağılmış n tane telefon vardır. İkili olarak n tane kablo ile birbirlerine bağlanmışlardır. Kesikli (*Ing.* discrete) $t = 0, 1, 2, \dots$ zamanlarda, her oyuncu bir telefona cevap verir ve "merhaba" der. Oyuncular ilk eşleştirilmiş telefonları açtıkları ve iletişim kurabildikleri T zamanını en aza indirmek istemektedirler. Problem; oyuncuların telefonları açtıkları sırayı seçmek için nasıl bir ortak rasgeleleştirilmiş strateji (*Ing.* randomized strategy) kullanacaklarına karar vermeleridir. Her iki problem için de iki oyuncu birbirinden ayırt edilemez, o yüzden aynı

rasgeleleştirilmiş stratejiyi kullanmak zorundadır. Yani problem, oyuncuların aynı anda uyguladıklarında, randevu için gerekli olan beklenen zamanı en aza indirecek rasgeleleştirilmiş bir strateji bulmaktır. Sonraki bölümde sık meydana gelen randevu arama stratejilerinden bahsediyoruz.

2.3. Sık Meydana Gelen Randevu Arama Stratejileri

Çoğunlukla verilen bir kaynaktaki optimal olan stratejiler oldukça karmaşıktır. Fakat, bu alanda yapılan birçok çalışmada belirli randevu stratejileri çok sık meydana gelmektedir. Bu bölümde bu stratejilerden bahsediyoruz. "Odak" stratejisi (*Ing. Focal Strategy*) bu stratejilerden biridir. Oyuncuların arama bölgesi Q üzerinde, noktaları, ortak bir etiketlendirmesi varsa, odak stratejisi uygulanabilir. Bu stratejide, oyuncular önceden bir randevu noktası üzerinde anlaşır ve bulundukları konumdan doğrudan o randevu noktasına giderek diğer oyuncunun gelmesini beklerler. Bir önceki bölümde asimetrik randevuyu açıklarken de örnek olarak verdiğimiz "Anneyi bekle" stratejisinde (*Ing. Wait for Mummy Strategy (WFM)*), bir oyuncu (çocuk) yerinde beklerken, diğer oyuncu (anne) kapsamlı bir arama yapar, genelde sabit bir objeyi en az beklenen zamanda bulmak için tasarlanmıştır. "Değiştirilmiş, Anneyi Bekle" stratejisinde (*Ing. Modified Wait for Mummy Strategy (MWFM)*), çocuk annenin arama yaptığını ve ona belirli bilinen zamanlarda ulaşacağını bilir. Ona yaklaşan annesi ile buluşmak için, annesine doğru hareket ederek beklenen randevu zamanını azaltabilir. Çocuk her bilinen zamanlarda, hareketini tamamladıktan sonra, harekete başladığı noktaya geri döner. WFM nadiren optimaldir. MWFM çoğunlukla optimaldir. WFM stratejisinin bir diğer varyasyonu "Alternatif, Anneyi Bekle" stratejisidir (*Ing. Alternating Wait for Mummy*). Bu stratejide, iki oyuncu birbirlerinin olası başlangıç noktalarına doğru hareket edip, her hareketten sonra kendi başlangıç noktalarına geri döner. WFM ve MFWM stratejilerinde oyuncuların rollerine (anne ve çocuk) önceden karar vermiş olması gerektiğinden dolayı, oyuncu-simetrik randevu aramada (*Ing. player symmetric rendezvous search*) iki strateji de uygulanabilir değildir. "Rasgeleleştirilmiş, Anneyi Bekle" stratejisi (*Ing. Randomized Wait for Mummy (RWFM)*), eğer arama bölgesi olan Q 'da kapsamlı bir arama yapmak için gerekli olan sabit bir zaman varsa, oyuncular tarafından bağımsız olarak uygulanabilir. RWFM stratejisinde, bir ajan her bir periyodun uzunluğu süresince p olasılıkla arama yapar veya her bir periyodun uzunluğu süresince $1 - p$ olasılıkla olduğu yerde bekler. Eğer herhangi bir periyotta ajanların (beklemek veya arama yapmak arasında) yaptığı rasgele

seimler birbirinin tersi ise, o zaman ajanlar o periyotta kesinlikle buluşacaklardır. Her iki ajanın da rasgele yaptığı seimler aramak ise, o zaman randevu olasılıkları Q 'ya baėlıdır. Bu tip simetrik strateji birçok kez kullanılır. Sonraki bölümde *Randevu* ve *Randevu Arama* problemleri arasındaki farkları inceliyoruz.

2.4. Randevuya karşı Randevu Arama

Robotik literatüründe, randevu problemlerinin iki tipi vardır. Birinci versiyon, ajanların birbirinin durumunu gözleyebildiėi, hareketli bir objeye doėru robot takibi ve navigasyonu ile ilgilenir. Bu nedenle, robotun ve hedefin kinematik denklemlerini birleştirmek gibi kontrol-teorik yönlelere önem verilmiştir. Hedef, baska bir mobil robot, uydu, hareketli bir konvoy veya insan olabilir. Bazen, randevu isteėi tek taraflıdır (örneğin, hareketli bir konvoyla buluşmak isteyen bir ajan gibi). Dahası, robot ve hedef, büyüklük, hız, oryantasyon gibi farklı özelliklere sahip olabilir. Bu randevu tipine bir örnek hareketli bir robota ulaşmak için minimum enerji kontrolü için optimal kontrol problemi- dir Rawicz et al. (1998).

Paralel navigasyon kanunu (*Ing.* parallel navigation law) robotun hedefle randevusu için en çok kullanılan döngü kontrollerinden (*Ing.* closed loop control) birisidir. Bu yaklaşım robotun ve hedefin kinematik denklemlerini geometrik kurallar ile birleştirir. Robot, kendisini hedefe bağlayan göreceli pozisyon vektörü olan başlangıç görüş mesafesine (*Ing.* line of sight (LOS)) paralel olan doğrular içinde hareket eder. Paralel navigasyon kuralı, robotun LOS'unun hedefe olan açısının tüm zamanlarda sabit kalacağını belirtir. Agah et al. (2003). Bu nedenle, yakalama sadece LOS yönünde düşünülür. Robot, ortamı ve hedefin manevralarını önceden bilebilir veya bilmeyebilir. Genellikle, robot engelleri tespit etmek ve hedef hakkında pozisyon, doğrusal hız ve oryantasyon gibi gerekli bilgileri edinebilmek için bir sensör sistemine sahiptir. Hedef (ve engeller) hakkında edinilen bilgi randevu yol gösterme algoritmasında kullanılır.

Biz bu tezde, *randevu arama* (*Ing.* rendezvous search) problemine odaklanıyoruz. Yukarıda anlatılan robot takip ve navigasyon problemine benzer olarak, bu konuşlanma, verilen bir ortamda robot-1 ve robot-2 arasında LOS kurmak gibi düşünülebilir. Fakat, bunun dışındaki formülasyon tamamen farklıdır. 1-) Bizim çalıştığımız problemin kalbi, durum bilgisinin olmayışı-

dır. Robot iletişim kuramaz ve diğer robotun pozisyonuna karar vermeye yarayacak (randevuyu tespit edebildiği halde) bir (uzun mesafe) sensör sisteme sahip değildir. Robotlar birbirlerinin şu anki konumunu bilmek zorunda değildir ve robot, aralarındaki başlangıç uzaklığını veya diğer robota olan yönünü bilmez (veya bilemez). 2-) Bir diğer fark, bir aktif robot ve bilinmez bir hedefle karşılaştırıldığında *her iki* robotun da randevuya ulaşmak için aktif olarak çalışmasıdır. 3-) Üçüncü fark da robotların aynı özelliklere sahip olmasıdır; örneğin, aynı büyüklüğe ve hıza sahiptirler. Son anahtar sınırimız da, robotların simetrik randevu aramada aynı stratejiyi çalıştırmak zorunda olmalarıdır.

2.5. Randevu Arama Probleminin Performansının Ölçülmesi

İki robotun bir Q ortamında aramaya x ve y konumlarından başladığını düşünelim. $d(x, y)$ bu noktalar arasındaki uzaklık olsun. Bir randevu stratejisi $\mathcal{S}$ için, $S_i(x, y)$ randevudan önce robot $i = 1, 2, \dots$ tarafından (beklenen) gidilen mesafe olsun. Randevu arama stratejisi $\mathcal{S}$ 'in etkinliği çoğunlukla kompetitif oranı ile ölçülür:

$$\max_{x, y \in Q} \frac{S_1(x, y) + S_2(x, y)}{d(x, y)} \quad (2.1)$$

Burada payda randevudan önce gidilebilecek en küçük mesafedir. $\mathcal{S}$ 'in kompetitif oranı, $\mathcal{S}$ 'in performansının optimal davranışından en kötü durum sapmasıdır. Bir stratejiye, eğer kompetitif oranı sabit ise *kompetitif* denir. Önceden bir buluşma noktası belirlemenin kompetitif bir strateji olmadığını not edelim: robotların konuşlandıkları konumları birbirine çok yakın olurken önceden belirlenmiş randevu noktasından çok uzak olabilir. Denklem (2.1)'de, kompetitif oran toplam gidilen mesafeye göre verilmiştir. Maksimum gidilen mesafe gibi başka ölçümler de kullanılabilir.

Robot stratejileri duran zamanlar içermediği zaman, gidilen mesafe geçen zamanla orantılır. Biz burada robot hareketlerini duran zamanlarla karıştıran rasgeleleştirilmiş stratejileri düşünüyoruz. Bu nedenle, iki kompetitif oran düşünüyoruz: mesafe kompetitif oran ve zaman kompetitif oran. Robot hareketi robotun duran zamanından daha pahalı olduğundan, bu tezde tasarladığımız randevu stratejilerimizde mesafe kompetitif oranı en aza indirmeye odaklanıyoruz. Simetrik stratejiler için, mesafe kompetitif oran ayrıca şuna

eşittir:

$$\max_{x,y \in Q} \frac{2S_1(x,y)}{d(x,y)} = \max_{x,y \in Q} \frac{S_1(x,y)}{d(x,y)/2}. \quad (2.2)$$

Biz bu tezdeki analizlerimizde bu formülasyonu kullanıyoruz.

Şimdi örnek olarak farklı şekilde formüle edilmiş iki randevu arama problemi için tasarlanmış olan stratejilerin kompetitif oranının nasıl hesaplandığını gösteriyoruz. İlk olarak bilinen *lost-cow* probleminin kompetitif analizini yapıyoruz. Problem şu şekildedir: Uzağı göremeyen bir inek (veya ilerisinin karanlık veya sisli olduğunu düşünelim) bir çitin önünde duruyor ve çitin tek kapısının hangi yönde olduğunu bilmiyor. İnek çok dolambaçlı bir yol yürümeden kapıyı nasıl bulabilir? Bu problem randevu arama probleminin asimetrik versiyonuna bir örnektir. Problemin formülasyonu 2.3. bölümünde bahsedilen WFM stratejisine uygundur. Burada ineği arama yapan anne, kapıyı da bulunmayı bekleyen çocuk gibi düşünebiliriz. Kapının ineğe d uzaklıkta olduğunu düşünelim. İnek bu başlangıç uzaklığından habersizdir. P_1 ineğin sol tarafında olan yol ve P_2 de ineğin sağ tarafında kalan yol olsun. İnek kapının hangi yönde olduğunu bilmediğinden bu yolları dönüşümlü olarak arayacaktır.

İnek aramada kararlı online bir algoritma uygulasin. Böyle bir algoritmayı negatif olmayan bir $f_0 = 0, f_1, f_2, f_3, \dots$ dizisi olarak tanımlayabiliriz. Burada f_i , ineğin i . adımda $(P_i \bmod 2)$ yolunda ne kadar ilerlediğini gösterir. Buradaki önemli soru gidilen toplam mesafeyi en aza indirmek için f_i 'i ne seçeceğimizdir. Her adımda gidilen mesafeyi ikiye katladığımızı düşünelim ve $f_i = 2^{i+1}$ olarak belirleyelim. Aşağıda bu şekilde tasarlanmış olan stratejinin kompetitif oranının nasıl hesaplandığını gösteriyoruz.

En kötü durumda küçük ϵ ve $i \geq 1$ için $d = f_i + \epsilon$ 'dur. i . adımda ineğin kapının olduğu yolu yürüdüğünü ve kapıyı ϵ farkla kaçırdığını düşünelim. Bu durumda inek diğer yolu f_{i+1} mesafeye kadar tekrar arar ve daha sonra kapının olduğu doğru olan yolu aramaya başlar; $(i+2)$. adımda d mesafe ilerledikten sonra kapıyı bulur. Bu nedenle, gidilen mesafe aşağıdaki gibi

hesaplanır

$$\begin{aligned}
2 \sum_{j=1}^{i+1} f_j + d &= 2 \sum_{j=1}^{i+1} 2^{j+1} + d \\
j &\leftarrow j + 1 \\
&= 4 \sum_{j=0}^{i+1} 2^{j+1} - 2 + d \\
&= 4(2^{i+2} - 1) - 2 + 2^{i+1} + \epsilon = 9(2^{i+1}) + \epsilon - 6 \\
&= 9(d - \epsilon) + \epsilon - 6 \\
&\leq 9d
\end{aligned} \tag{2.3}$$

Eğer inek kapının yönünü ve kapıya olan mesafeyi bilseydi o zaman en kısa zamanda kapıya ulaşırđı. Bu en iyi durum senaryosunda, inek d mesafede kapıyı bulacaktır. Bu nedenle, ikiye katlama stratejisi *lost-cow* problemi için $9d/d = 9$ kompetitif orana sahiptir.

İkinci bir örnek olarak, randevu arama probleminin simetrik versiyonu için tasarlanmış basit bir stratejinin kompetitif oranının nasıl hesaplandığını gösteriyoruz. Birbirlerinin konumlarından habersiz iki robotun doğru üzerinde konumlandırıldığını düşünelim. Robotların aralarındaki sabit başlangıç uzaklığı olan d 'i bildiklerini ve aramaya aynı anda başladıklarını düşünelim. d 'i bildikleri durumda bile problemin zorlukları vardır. Analizin basit olması açısından $d = 2$ olarak alalım. Robotlardan birinin doğru üzerinde $x = 0$ konumuna sağa dönük olarak yerleştirildiğini ve diğer robotun da $x = \pm 2$ konumlarından birine sağa veya sola dönük olarak yerleştirildiğini düşünelim. Orijinde konumlandırılmış robotu Robot-1, diğer robotu da Robot-2 olarak adlandıralım. A , Robot-1'in sağa ilerlemesi olayı ve B , Robot-2'nin sola ilerlemesi olayı olsun. Robotların $(0, \pm 2)$ konfigürasyonu için $A \wedge B$ veya $A \wedge \overline{B}$ olayları gerçekleşebilir. Bu dört durum da eşit olasıkla gerçekleşir.

Bu problem için en basit strateji robotun 1 birim ileri rasgele bir yöne doğru ilerlemesi ve başlangıç konumuna geri dönmesi olacaktır. Robotlar sadece $(0, +2)$ konfigürasyonunda oldukları zaman, $A \wedge B$ olayı gerçekleşirse buluşurlar. Bu durumda robot bir birim mesafe ilerler. Diğer üç durumdan birisi gerçekleşirse robotlar buluşamazlar ve bu durumda robot bu durumlarından her birinde 2 birim mesafe ilerler. Robotların bir sonraki tura geçmeleri ve aramaya devam etmeleri bir önceki turda başarılı olup olmamalarına bağlıdır.

Beklenen gidilen mesafeyi şu şekilde hesaplayabiliriz

$$\sum_{i=0}^{\infty} \left(\frac{1}{4} \cdot 1 + \frac{3}{4} \cdot 2 \right) \left(\frac{3}{4} \right)^i \leq 7 \quad (2.4)$$

Eğer robotlar birbirlerinin yönünü bilselerdi birbirlerine doğru 1 birim mesafe ilerleyip ortada buluşurlardı. Bu yüzden stratejinin kompetitif oranı $7/1 = 7$ 'dir.

2.6. Katkılarımız

Bu tezde, bu bölümde detaylı olarak tanıttığımız, randevu arama problemini doğru üzerinde iki robot ile, açık düzlem ve engelli ortamlar üzerinde iki ve ikiden çok robot ile çalışıyoruz.

Problemi, tamamen aynı özelliklere (hız, büyüklük, donanım, vb.) sahip robotlar üzerinde çalışıyoruz. Ayrıca robotlar için herhangi bir etiketlendirme yapmıyoruz. Robotların birbirleri ile ilgili herhangi bir bilgiye sahip olmadıklarını düşünüyoruz; robotlar aralarındaki başlangıç uzaklığını, birbirlerinin yönlerini ve ortam üzerindeki konumlarını bilmiyorlar. Bunların dışında, ortamlarla ilgili önceden bir bilgiye sahip değiller, bu nedenle algoritmalarımızı on-line tasarlıyoruz. Düzlemsel ortamlarda buluşmak isteyen robotların sınırlı iletişim alanına ($R > 0$ olmak üzere) sahip olduğunu düşünüyoruz.

Problemin, her bir robotun buluşmak için aynı stratejiyi uyguladığı, simetrik versiyonu ile ilgileniyoruz. Böylelikle her robot için ayrı bir strateji üretme zorunluluğunu ortadan kaldırmış oluyoruz. Oyuncu simetrik versiyondan ötürü, stratejilerimizi rasgeleleştirilmiş olarak tasarlıyoruz. Aksi halde kararlı olarak tasarlanmış bir stratejide robotlar arama başında aynı yönü seçerlerse asla bulaşamazlar. Robotlar için gidilen mesafe bekledikleri zamana göre daha maliyetli olduğundan stratejilerimizde beklenen gidilen mesafeyi en aza indirmeyi hedefliyoruz.

Doğru üzerinde simetrik randevu arama problemi için yeni bir algoritma sağlıyoruz. Temel olarak problemin robotların aramaya aynı anda başladığı senkron durumuna odaklanıyoruz. Ayrıca problemi asenkron durum için de inceliyoruz ve bu durum için ön çalışmalarımızı sunuyoruz. Bizim simetrik

algoritmamız gidilen mesafe için 17.686 olan bir kompetitif orana ve toplam zaman için 24.843 olan bir kompetitif orana sahiptir. Her iki oran da daha önce en iyi bilinen, 26.650 zaman ve mesafe kompetitif oranına sahip olan algoritmadan üstündür. Algoritmamız aynı performans garantileri ile, sınırlandırılmış doğrusal ortamlar ve kapalı eğriler için adapte edilebilir. Aynı zamanda, harekete bağlı olan hatalara ve robotların başlama zamanlarındaki farklılıklara karşı da dayanıklıdır. Asenkron durum için literatürde bilginiz dahilinde sunulmuş herhangi bir çalışma yoktur. Halen açık olan bu problem için yaptığımız ön çalışmaları sunuyoruz. Simetrik stratejimizden bekleme zamanlarını kaldırarak bu duruma uygun hale getiriyoruz. Bu durum için stratejimizin performansına üst sınır buluyoruz. Problemin senkron ve asenkron durumu için elde ettiğimiz teorik sonuçlarımızı simülasyonlarla doğruluyoruz ve iç ortamlara gerçek robotları yerleştirerek aldığımız sonuçları raporlandırarak algoritmamızın uygulanabilirliğini gösteriyoruz.

Simetrik randevu arama problemini doğru olarak modellenen ortamların yanısıra düzlemsel ortamlarda da çalışıyoruz. İlk olarak, açık bir düzlemde konumlandırılmış iki robot için simetrik bir randevu stratejisi sunuyoruz ve kompetitif oranını analiz ediyoruz. Stratejimizin kompetitif karmaşıklığının $O(d/R)$ olduğunu gösteriyoruz. Burada d robotlar arasındaki başlangıç uzaklığıdır ve R robotun iletişim yarıçapıdır. İkinci olarak simetrik randevu stratejimizi poligonal engellerin olduğu bilinmeyen ortamlara genişletiyoruz. Robotlar ortamla ilgili herhangi bir bilgiye sahip değildir. Genişletilmiş stratejimiz ortamın tam olarak kapsadığını garanti eder. Stratejimizi kare, çevrilen (*Ing.* translating) robotlar için analiz ediyoruz. Genişletilmiş stratejimizin kompetitif karmaşıklığının $O(d/D)$ olduğunu gösteriyoruz. Burada D robotun bir kenarının uzunluğudur. Bu sonucu elde ederek, ayrıca rasgele poligonal ortamları kapsamak için de bir üst sınır elde etmiş oluyoruz. Son olarak, çalıştığımız bu problemlerde iki robot için tasarladığımız algoritmalarımızı ikiden çok robotun (çok-robotlu, *Ing.* multi-robots) randevusu için genişletiyoruz. Açık düzlemdeki çok-robot için genişlettiğimiz simetrik randevu stratejimizin kompetitif karmaşıklığının $O(d/\log(n)R)$ olduğunu gösteriyoruz. Burada d ortamda birbirine en uzak olan iki robot arasındaki başlangıç uzaklığıdır ve n ortamdaki robot sayısıdır. Poligonal engellerin olduğu bilinmeyen ortamlardaki çok-robot için genişletilmiş stratejimiz de $O(d/\log(n)D)$ kompetitif karmaşıklığa sahiptir. Her problem için elde ettiğimiz teorik sonuçlarımızı simülasyonlarla doğruluyoruz. Engelli ortamlarda algoritmamızın uygulanabilirliğini göstermek için birbirinden farklı ortamlarda simülasyon deneyleri yapıyoruz ve elde ettiğimiz sonuçları raporlandırıyoruz. Sonraki bölümde te-

zin nasıl tasarlandığını sunuyoruz.

2.7. Taslak

Bu tez tanıtım, sonuç ve tartışma bölümleri de dahil olmak üzere yedi bölümden oluşur. 1. ve 2 bölüm, tezin detaylı tanıtımına ayrılmıştır. Her bölüm birbirinden bağımsız olarak yazılmıştır ve ayrı olarak okunabilir.

3. bölümde, daire/halka üzerinde ve çizge/ağ üzerinde randevu arama problemi ve bu probleme benzer olan *Toplanma* (*Ing.* gathering) ve *Karşılaşma* (*Ing.* coalescence) problemi ile ilgili detaylı literatür araştırması yapıyoruz.

4. bölümde, simetrik randevu arama problemini doğru üzerinde iki robotla çalışıyoruz. Bu bölümde, temel olarak problemin senkron durumuna odaklanıyoruz. Yeni bir simetrik randevu arama stratejisi ve stratejimizin detaylı analizini sunuyoruz. Teorik sonuçlarımızı simülasyonlarla ve gerçek ortam deneyleri ile destekliyoruz ve aldığımız sonuçları yine bu bölümde sunuyoruz. Ayrıca pratik senaryolarda algoritmamızın iki uzantısını sunuyoruz: doğru parçaları ve yarım doğrular; ve basit kapalı-yol ortamları (Örneğin, dairesel bir koridor). Buna ek olarak problemin literatürde bizim formülasyonumuz ile daha önce çalışılmamış asenkron durumu için olan ön çalışmalarımızı da sunuyoruz. Problemin bu ayarlaması için simetrik randevu stratejimizi değiştiriyoruz ve değişmiş stratejimizin analizini sunuyoruz. Edindiğimiz teorik sonuçları bu durum için de simülasyonlarla destekliyoruz.

5. bölüm, düzlemsel ortamda randevu arama problemine adanmıştır. Bu bölümde problemi iki robot ile engelli ve engelsiz ortamlarda çalışıyoruz. Simetrik bir randevu stratejisi ve stratejimizin mesafe ve zaman kompetitif analizini sunuyoruz. Randevu stratejisine ek olarak bu bölüm bir alan kapsama algoritması ve bu algoritmanın performansının hesaplanmasını da içerir. Bu bölümde ayrıca, hem engelsiz hem de engelli ortamlar için tasarladığımız stratejilerin ikiden çok robotlu uzantılarını da sunuyoruz. Simetrik randevu stratejilerimizin performanslarını simülasyonlarla da araştırıyoruz.

İLGİLİ ÇALIŞMALAR

3.1. Daire/Halka Üzerinde Randevu Arama Problemi

Randevu arama probleminin çalışıldığı ortamlardan biri dairedir Alpern; Howard (1999). Daire olarak modellenmiş bir X bölgesi üzerindeki iki robotla, asimetric randevuyu çalışan Alpern Alpern şu üç durumu düşünmüştür: birincisi, oyuncuların tüm noktaları ortak bir etiketlendirilmesinin olduğu durumdur, ki burada oyuncular aynı noktaya doğru ilerlemek için anlaşabilirler; ikincisi, oyuncuların noktaları ortak bir etiketlendirmesinin olmadığı ancak ortak bir yön kavramı olduğu durumdur, ki burada oyuncular buluşana kadar biri saat yönünde diğeri ise saat yönüne ters ilerlemek için anlaşabilirler; son durumda, oyuncular ortak bir yön kavramına ve noktaları ortak bir etiketlendirmeye sahip değildir. Bu durumların hepsinde, oyuncular çevresi 1 olan daire üzerinde rasgele konumlarda, birbirlerine $x \in [0, 1/2]$ mesafede yerleştirilmiştir.

Halka üzerinde üzerinde çalışılan randevu arama probleminde gas; Dessmark et al. (2006); Klasing et al. (2008); Kranakis et al. (2010, 2008); Marco et al. (2006) iki veya daha fazla ajanın n düğümlü bir halka ağ üzerindeki bir düğümde en kısa zamanda buluşabilmeleri için nasıl hareket etmesi gerektiğine odaklanılır.

Halka üzerinde asenkron kararlı randevu Dessmark et al. (2006) ve Marco et al. (2006)'de çalışılmıştır. Bu çalışmalarda, robotlar arasındaki simetriyi kırmak için farklı kimlikler kullanırlar. Dessmark et al. (2006) hem senkron hem de asenkron başlama zamanları için halka üzerinde randevuyu çalışmıştır. Aynı başlama zamanları için herhangi bir halka üzerinde randevunun $\Theta(D \log l)$ maliyete ve farklı başlama zamanları ve bilinmeyen n için $\Omega(n + D \log l)$ maliyete sahip olduğunu göstermişlerdir. Burada n düğüm sayısı, D ajanlar arasındaki başlangıç uzaklığı ve l robot kimliklerinden küçük olanıdır.

Marco et al. (2006)'de sonsuz bir doğru üzerinde birbirlerine D uzaklıkta yerleştirilen ajanlar için bir randevu algoritması sunmuştur. Algoritmalarının maliyeti D bilindiği zaman $O(D|L_{min}|^2)$ ve D bilinmediği zaman $O((D + |L_{max}|)^3)$ maliyete sahiptir. Bu maliyetler halka için de geçerlidir. Burada $|L_{min}|$ ve $|L_{max}|$ ajanların kısa ve uzun etiketlerini gösterir. Ayrıca

n bilindiği durum için maliyeti $O(n|L_{min}|)$ olan ve bilinmediği durum için maliyeti $O(n|L_{max}|)$ olan optimal bir algoritma da vermişlerdir.

Halka üzerinde farklı kimlikli k ajanın asenkron başlama zamanlarıyla randevusu Kowalski and Malinowski (2008)'de çalışılmıştır. Ajanlar halka üzerinde aynı düğüm üzerinde buluşmalıdır. Maliyeti $O(n \log C)$ olan kararlı bir algoritma sunmuşlardır. Burada n ağın boyutudur ve C katılan ajanların en küçük etiketidir.

Aynı kararlı algoritmayı çalıştıran, n boyutlu bir halka üzerinde dağılmış, ayırt edilemez k mobil ajanın randevusu Das (2008); Flocchini et al. (2004); Hole et al. (2009)'de çalışılmıştır. Halkanın düğümleri etiketlendirilmiştir. Bu durumda her ajan bir jeton kullanarak halka üzerindeki başlangıç konumlarını işaretler. Ajanlar tarafından bırakılan jetonların beklenmedik bir şekilde başarısız olması durumunda buluşmanın gerçekleşebilmesi için hata toleranslı çözümlere odaklanılır.

k isimsiz asenkron mobil ajanın ve bir kör deliğin olduğu n düğümlü bir ağda randevu problemi Hole et al. (2009)'de çalışılmıştır. Kör bir deliğin varlığı tüm ajanların randevusunu engeller. Bu nedenle, yazarlar hangi durumlarda kaç tane ajanın buluşabileceğini incelemişlerdir. Ajanlar böyle bir tehlikenin olduğunu bilirler ancak konumunu bilmezler. k veya n 'in bilinmediği durumda birden fazla ajanın randevusunun mümkün olmadığını göstermişlerdir. k bilindiği durum için önerdikleri randevu stratejisinin karmaşıklığı $\Theta(n)$ 'dir, ki bu problemi aynı durumda çalışmış $O(n \log n)$ sınırlı zaman karmaşıklığına sahip varolan protokollerin üzerine önemli bir iyileştirme-dir.

Kranakis Kranakis et al. (2008) isimsiz bir halka üzerinde ortak bir koordinat sistemine sahip, senkron ve hafıza gereksinimli iki aynı ajanın randevusunu çalışır. $O(n^2/2^t + 2^t)$ beklenen zamanda randevuya erişebilecek $2t$ ajan olduğunu ve her $t/2$ ajanın $\Omega(n^2/2^t)$ beklenen zamana ihtiyacı olduğunu gösterir. Lineer zamanda randevuya erişmek için $\Theta(\log \log n)$ bit hafızanın gerekli olduğunu sonuç (*Ing. corollary*) olarak incelemişlerdir.

3.2. Çizge/Ağ Üzerinde Randevu Arama Problemi

Bilinmeyen, isimsiz bir çizgenin farklı düğümleri üzerinde konumlandırılmış iki mobil ajanın (robotun) buluşması iyi çalışılmış bir randevu problemidir. Burada amaç ajanların çizge üzerindeki bir düğüm üzerinde en kısa zamanda buluşmasını sağlamaktır. Ajanların buluşma durumu problemin asenkron durumu için biraz daha rahatlatılabilir ve çizge üzerindeki bir kenar içinde buluşmalarına izin verilebilir. Çizge üzerindeki randevu arama probleminin maliyeti, randevu gerçekleşene kadar ajanlar tarafından dolaşılan toplam adım veya kenar sayısıdır.

Eğer çizge üzerindeki düğümler etiketlenirse, o zaman ajanlar önceden belirlenmiş bir düğüm üzerinde buluşmaya karar verebilir ve randevu problemi çizge keşfine indirgebilir. Fakat çoğu uygulamada, bilinmeyen bir topolojiye sahip bir ağda randevu gerekli olduğunda düğümler tek tek etiketlenilemeyebilir veya ajanlar bu etiketleri algılayamayabilir. Bu nedenle isimsiz çizge (düğümlerin ayrı etiketlenilmesinin olmadığı) üzerinde olan ajanlar için randevu algoritmaları tasarlamak daha önemlidir.

Çizge üzerindeki Dessmark et al. (2006); Marco et al. (2006), Czyzowicz et al. (2010b), Dieudonné et al. (2013), Kowalski and Malinowski (2008) ve Kowalski and Pelc (2005)'deki çalışmalarda, aynı kararlı randevu algoritmasını asenkron olarak çalıştıran birbirinin aynısı olan ajanların farklı etiketlendirmelerinin olduğu düşünülmüştür. Aksi halde randevu mümkün olmaz; ajanlar aynı yönü seçip aynı hızda hareket edebilirler (veya bir düşman ajanların hareket yönüne veya hızına karar verebilir). Ajanın arama sırasında izlediği yol, etiketine ve ortama bağlıdır. Bu çalışmalarda randevu arama probleminin asimetrik versiyonu düşünülmüştür. Bu nedenle, eğer ajanlar birbirlerinin etiketini bilirse, o zaman problem çizge keşfine indirgebilir. Küçük etikete sahip olan ajan hareket etmez ve diğer ajan çizgenin tüm düğümlerini onu bulana kadar dolaşır.

Rasgele bağlı isimsiz çizgeler üzerinde asenkron kararlı randevu Dessmark et al. (2006); Marco et al. (2006), Dieudonné et al. (2013); Kowalski and Pelc (2005); Kowalski and Malinowski (2008)'de çalışılmıştır. Dessmark et al. (2006); Kowalski and Pelc (2005); Kowalski and Malinowski (2008)'de ajanlar arasındaki senkronluk ajanların aramaya farklı zamanlarda başlamasıyla bozulurken, Dieudonné et al. (2013); Marco et al. (2006)'da senkronluk bir düşmanın arama sırasında ajanların hareketlerine müdahale etmesiyle bo-

zudur. Dessmark et al. (2006); Kowalski and Malinowski (2008) ve Kowalski and Pelc (2005)'da önerilen algoritmaların maliyeti daha geç aramaya başlayan ajandan randevu gerçekleşene kadar olan adım sayısıdır.

Marco et. al. Marco et al. (2006), eğer çizgenin boyutunun üst sınırı bilinirse randevunun mümkün olabileceğini göstermiştir. Topoloji ve ajanların başlangıç pozisyonları bilinirse, maliyeti $O(D \mid L_{min} \mid)$ olan optimal bir algoritma vermişlerdir. Burada $\mid L_{min} \mid$ ve $\mid L_{max} \mid$ sırasıyla ajanların kısa ve uzun etiketleridir. Algoritmanın maliyeti, çizge üzerinde ajanların verilen başlangıç pozisyonları için randevu gerçekleşene kadar her iki ajan tarafından dolaşılan kenar sayısı ile ölçülür. Dessmark et al. (2006) n , τ ve $\log l$ de polinom maliyetli bir kararlı randevu algoritması sunmuştur. Burada τ ajanların başlangıç zamanları arasındaki farktır. Ayrıca bazı çizge sınıfları için randevu maliyetinin $\Omega(n^2)$ alt sınırına sahip olduğunu göstermişlerdir.

Kowalski and Malinowski (2008)'de genel ağlar için n and $\log l$ de polinom maliyetli bir protokol sunulmuştur. Burada n ağ boyutu ve l katılan ajanların en küçük etiketidir. Protokolleri $O(\log^3 l + (\lambda(n))^3)$ zamanda randevuya erişir. Burada, $\lambda(N) = O(N^5 \log^4 N)$ 'dir. Maliyet ölçümleri Dessmark et al. (2006) ile aynıdır.

M kenarlı tam bir çizge üzerinde ikiden daha fazla oyuncu ile simetrik randevu arama problemi Thomas and Pikounis (2001)'de çalışmıştır. Yazarlar, arama sırasında karşılaşılan oyuncuların randevu gerçekleşene kadar birbirlerine bağlı mı hareket etmeleri, yoksa ayrılıp tekrar mı buluşmaları gerektiğini düşünen iki stratejiye bakarlar. Robotların birbirlerine bağlı kalmalarının, sabit ve hafıza gerektirmeyen strateji sınıflarında optimal olduğunu gösterirler. Fakat, iki stratejiyi karşılaştıran simülasyon sonuçları, biraz hafıza gerektiren ayrılıp tekrar buluşma stratejisinin birçok durumda daha hızlı beklenen randevu zamanlarına ulaştığını göstermiştir.

Chalopin et al. (2007), ağ üzerinde bazı kenarların tehlikeli veya bozuk olduğu ve bunlara giren herhangi bir ajanın yok olacağı durumu düşünmüştür. Yok olan ajanların sayısını en aza indirmeyi ve geride kalan ajanların buluşmasını sağlamayı amaçlamışlardır. Ağ topolojisinin önceden bilinmediği varsayılır. Ağ boyutunda sıkı bir sınıra sahip olmanın problemi çözmek için yeterli olacağını gösterirler. Eğer G çizgesi üzerinde τ tehlikeli kenar varsa ve başlangıçta k ajan mevcutsa, o zaman G de en fazla $k - \tau$ ajan buluşabilir. Ağdaki $k - \tau$ ajanın buluşmasının, n , k ve q parametrelerinin hepsi bilinse bile

$O(m(m+k))$ hareket gerektirdiğini gösterirler. Bu nedenle, algoritmalarının optimallliğini ispatlamışlardır.

Anderson ve Weber Anderson and Weber (1990) bir binada veya alışveriş merkezinde ayrı düşen iki arkadaşın simetrik randevusunu çalışmıştır. Bu kişilerin buluşabileceği n mümkün lokasyon vardır. Bu oyun iki oyuncunun başlangıçta iki farklı köşede olduğu K_n tam bağlı bir çizge olarak modellenebilir. Oyuncuların araması kesikli zamanlar içinde devam eder. Lokasyonların ortak bir etiketlendirmeleri yoktur. Böylelikle bir oyuncu her k adımda lokasyonların her birine hareket etme olasılıklarını, sadece nerde olduğunun bir fonksiyonu olarak seçmelidir. Randevuyu büyük n için $0.829n$ 'den daha az beklenen zamanda sağlayan rasgeleleştirilmiş bir strateji önermişlerdir. Önerdikleri strateji $n = 2$ ve 3 için sırasıyla ortamala 2 ve $8/3$ adım gerektirir. Daha sonra, Weber K_3 'de simetrik randevu için Anderson ve Weber (AW Stratejisi) tarafından önerilen stratejinin optimal olduğunu göstermiştir ve bu stratejiyi kullanarak randevu için gerekli en az beklenen zamanı $5/2$ olarak geliştirmiştir. Dieudonné et al. (2013), çizgenin boyutu ve daha küçük etiketin uzunluğunda polinom olan kararlı bir randevu algoritması sunmuştur.

3.3. Toplanma ve Birleşme Problemi

Toplanma problemi bir ortamda (düzlem, çizge, halka gibi) bulunan ikiden daha fazla robotun önceden belirlenmemiş tek bir konumda sonlu zamanda toplanması ile ilgilenir. Birbirinin aynı olan robotlar aynı toplama algoritmasını çalıştırılır. Toplama problemi randevu arama probleminin bir genellemesidir. Ancak kullanılan yöntemler randevu aramadan farklıdır. Bu problemde, robotlar *Bekle, Bak, Hesapla ve Hareket et* (Ing. Wait, Look, Compute and Move) işlem sırasını takip ederler. Bu nedenle, robotlar hareket kararlarını çevrelerine bakarak ve diğer robotların konumlarının konfigürasyonlarını analiz ederek belirler. Robot asenkron ise diğer robotlardan bağımsız olarak uyanır ve diğer robotların pozisyonlarına bakar. Bir robot sınırlı veya sınırsız olabilen görme mesafesindeki tüm robotların yerini tespit edebilir. Diğer robotların birbirlerine göre ve kendisine göre olan konumlarını (mesafe ve açılar) hesaplayarak ortamda gideceği noktayı hesaplar. Bu noktaya doğru hareket eder ve hareketini tamamladıktan sonra tekrar bekleme durumuna geçer. Toplanma probleminden farklı olarak randevu aramada probleminde diğer robotların pozisyonlarını hesaplamıyoruz. Ayrıca bu problemle ilgili çalışmalar robotların belirlenen koşullara göre sadece sonlu zamanda toplanıp

toplanamayacağını inceler ve bu iddianın doğruluk ispatını yapar. Önerilen toplama algoritmasının karmaşıklık analizini yapmaz.

Bu problem literatürde birçok farklı model üzerinde çeşitli varsayımlarla geniş olarak çalışılmıştır. Bu problemin zorluğu robotların çok zayıf ve basit olduğu varsayılırsa artar ve yapılan çalışmalarda sunulan modeller bu varsayımla şekillendirilir. Bu çalışmaların tümünde problem robotların yeteneklerine göre bazı ek varsayımlar yapılarak çözülür. Birçok çalışma toplama problemini asenkron modellerde çözmeye odaklanır.

Robotların Agmon and Peleg (2004); Cieliebak (2004); Cieliebak et al. (2003); Cieliebak and Prencipe (2002)'daki çalışmalarda sınırsız iletişim alanına sahip olduğu varsayılır. Bu çalışmalarda robotlar tüm alanı algılama (görme yeteneğine) sahiptir. Genellikle isimsiz ve boyutsuzdurlar, yani birbirlerinin görüşünü veya hareketini engellemeyen noktalar olarak davranırlar. Ancak Jurek et al. (2009)'de robotlar birim diskler olara temsil edilir. Robotlar unutkan olabilirler (veya hafızasız) yani daha önceki hareketlerini veya diğer robotların önceki pozisyonlarını hatırlamazlar Cieliebak et al. (2003); Cieliebak and Prencipe (2002); Flocchini et al. (2005); Jurek et al. (2009) veya unutkan olmayabilirler Cieliebak (2004).

Asenkron model olan *CORDA*'da Flocchini et al. (2000), unutkan robotlarla toplama problemini sonlu zamanda çözecek hiçbir kararlı algoritma olmadığı gösterilmiştir. Cieliebak et al. (2003); Cieliebak and Prencipe (2002); Souissi et al. (2006)'de robotlar *çokluğu* (*Ing. multiplicity*), yani düzlem üzerinde verilen bir noktanın bir veya daha fazla robot tarafından işgal edilip edilmediğini, tespit edebilirler.

Agmon and Peleg (2004) içlerinden birinin hatalı olduğu üç veya daha fazla robotun toplanması için hata toleranslı bir algoritma sunmuştur. Byzantine hatalarına meyilli bir sistemde, tüm bilinen algoritmaların başarısız olduğunu gözlemlemişlerdir. En fazla birinin Byzantine davranışında başarısız olabileceği, asenkron ortamdaki üç robotlu bir sistemde toplanmanın mümkün olamayacağını gösterirler. Problemin senkron versiyonu için en fazla bir robotun hatalı olduğu, $N \geq 3$ robotun toplanması için bir algoritma sağlarlar. Ayrıca f hataya kadar olan N -robotlu bir sistem için daha genel bir toplama algoritması sunarlar. Burada $N \geq 3f + 1$ 'dir.

Ortak bir kordinat sistemi üzerinde anlaşması olmayan, 2d bir ortamda

bağımsız olarak dolaşan, bozuk pusulalı iki mobil robotun toplanması Souissi et al. (2006) çalışılmıştır. Yazarlar aralarında $\pi/4$ fark olan pusularla donatılmış, iki unutkan robotun sonlu zamanda toplanması için öz-dengeleyici (*Ing. self-stabilizing*) bir algoritma sağlar.

Cieliebak and Prencipe (2002)'da, merkeze bağlı olmayan, asenkron, ortak bir bilgisi, kimliği ve birbirleri ile doğrudan iletişimi olmayan zayıf bir robot modeli tanıtmıştır. Robotlar unutkandır ve sınırsız iletişim alanına sahiptir. Yazarlar bu robotların bu zayıf özelliklere sahipken sonlu zamanda toplanmasının mümkün olmadığını kanıtlarlar. Bu nedenle, robotları *çokluk* tespit etme özelliğiyle güçlendirmişlerdir. Toplanma probleminin bu özelliklerle güçlendirilmiş iki robot için hala çözümsüz olduğunu kanıtlarlar ve üç ve dört robot için toplanma problemini çözen algoritmalar verirler. Dörtten fazla robot için, robotların başlangıçta belirli bir simetrik konfigürasyonda (biangular konfigürasyon) olmadan sonlu zamanda toplanmalarını sağlayacak bir algoritma sunulur. Cieliebak and Prencipe (2002)'deki robotların aynı özellikleri ile Cieliebak et al. (2003), herhangi bir konfigürasyonda başlayan $n > 2$ robot için toplama problemini çözen ilk algoritmayı sunar.

Jurek et al. (2009) birim disklerle temsil edilen aynı robotların kararlı olarak bir düzlemde dolaştığını düşünmüştür. Robotlar herhangi bir ortak koordinat sistemine sahip değildir, iletişim kurmazlar, geçmişe ait hafızaları yoktur ve tümüyle asenkronudur. Bu robotların toplanması, onları temsil eden disklerin birleşiminin birbirine bağlı olacağı bir konfigürasyon oluşturması demektir. Yazarlar en fazla dört robot için toplanma problemini çözmüştür. Bu çalışma robotların ortamda nokta olarak değilde iki boyutlu figürler olarak temsil edilerek alınan ilk algoritmik sonuçları sunar. Bu robotlar, çalışmada *şişman* (*Ing. fat*) robotlar olarak adlandırılır.

Sınırlı iletişim alanına sahip, asenkron, unutkan, isimsiz ve ortak bir koordinat sisteminde anlaşması olan robotların sonlu zamanda toplanması Floccini et al. (2005) çalışılmıştır. Bu çalışmada oryantasyonun en az robotun anlık hareketleri kadar önemli olduğunu gösterilir.

Geometrik senaryolarda çalışılan randevu arama probleminin bir diğer varyasyonu da *Birleşme* problemidir (*Ing. coalescence* problemi). Bu problemde, bağımsız olarak arama yapan mobil robotların amacı tek bir bağlı ağ oluşturmaktır. *Coalescence* kelimesi bağlı komponentin robotlar katıldıkça genişlediğini ifade etmek için kullanılmıştır. Bu problem Poduri and Sukhatme

(2007a,b)'de çalışılmıştır. Poduri ve Sukhatme çalışmalarında, robotların hafızasız (harita inşa etmiyorlar) olduğunu ve ortamla ilgili veya diğer robotların pozisyonları ile ilgili herhangi bir bilgiye sahip olmadıklarını varsaymıştır Poduri and Sukhatme (2007a). Her robot bağımsız bir rasgele arama yapar ve eğer iki robot karşılaşır, o zaman bir küme (*Ing.* cluster) içine birleşirler, bu küme içinde yapışırlar ve tek bir rasgele yürüyüş yaparlar. Rasgele hareket etme modelini kullanarak Poduri ve Sukhatme birleşme zamanının, robot sayısının, robot hızının, robot iletişim alanının ve alanın büyüklüğünün bir fonksiyonu olan, üssel (*Ing.* exponential) bir dağılıma sahip olduğunu göstermiştir. Ayrıca robot sayısı olan N arttıkça, birleşme zamanının $O(1/\sqrt{n})$ ve $\Omega(\log(n)/n)$ olarak azaldığını göstermişlerdir.

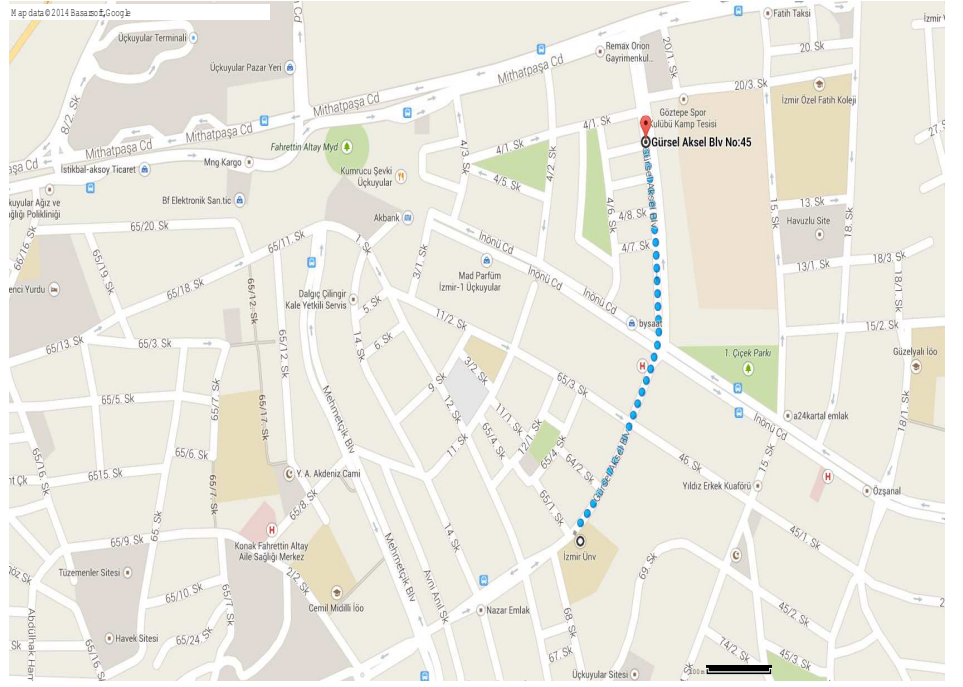
DOĞRU ÜZERİNDE SİMETRİK RANDEVU ARAMA

Giriş

Birbirleri ile buluşmak isteyen iki oyuncunun bir yol, koridor, nehir, cadde üzerinde veya daha genel olarak bir doğru olarak modellenebilecek bir arama bölgesine konumlandırıldığını düşünelim (Şekil 4.1'e bakınız). Bu oyuncular, birbirlerini aramaya başlamadan önce bir araya gelip arama sırasındaki rollerini kararlaştırmamış olsunlar. Bu yaklaşıma randevu arama probleminin oyuncu simetrik versiyonu veya kısaca simetrik versiyonu denir. Bir oyuncunun bulunduğu bölge üzerindeki kendi konumunu, diğer oyuncunun konumunu ve dahası diğer oyuncuya olan yönünü bilmeğini düşünelim. Oyuncuların beklenen en kısa zamanda buluşmalarını sağlamak için onlara nasıl bir randevu arama stratejisi uygulanmalarını önermeliyiz? Bu bölümde, bu problemi, bir doğru üzerinde konumlanmış aralarındaki başlangıç uzaklığını bilmeyen iki robot için çalışıyoruz ve yeni bir simetrik randevu algoritması sağlıyoruz. Problemin simetrik versiyonu her robot için ayrı bir strateji tasarlama gerekliliğini ortadan kaldırır, bu nedenle robotik uygulamaları için daha elverişlidir.

Problemin simetrik doğasından ötürü stratejimizi rasgeleleştirilmiş olarak tasarladık. Çünkü kararlı olarak tasarlanmış bir stratejiyi yürüten, doğru üzerinde konumlandırılmış robotlar baştan aynı yönü seçerlerse aralarındaki uzaklığı her zaman korurlar ve asla buluşamazlar. Ayrıca, ortam 1-boyutlu olduğundan ortamı ayıracak bir engel robotlar arasında olamaz, aksi takdirde randevu asla gerçekleşmez. Stratejimizin hem mesafe kompetitif oranını hem de zaman kompetitif oranını hesaplıyoruz. Ancak, robotun hareketi duran halde olduğu zamanlarından daha maliyetli olduğundan tasarladığımız stratejinin mesafe kompetitif oranını en aza indirmeye odaklanıyoruz.

Bu bölümün katkıları şu şekildedir. İlk kısımda robotların aramaya aynı anda başladığını varsaydığımız senkron durumu çalışıyoruz. Bilinmeyen (rasgele (*Ing. arbitrary*)) uzaklıkta doğrusal simetrik randevu için yeni bir algoritma sağlıyoruz. Bu algoritma, daha önce en iyi bilinen, 26.650 zaman ve mesafe kompetitif oranına sahip olan Baston ve Gal'ın algoritmasına bir iyileştirmedir: Algoritmamız 17.686-mesafe ve 24.843-zaman kompetitifdir. Teorik sonuçlarımızı simülasyonlarla doğruluyoruz ve iç ortamlara gerçek robotları



Şekil 4.1: Gürsel Aksel Caddesi üzerinde bulunan, birbirlerinin konumundan habersiz iki arkadaş en kısa zamanda buluşmak istiyor.

yerleştirerek aldığımız sonuçları raporlandırarak algoritmamızın uygulanabilirliğini gösteriyoruz. Son olarak, algoritmayı gerçek dünyadaki iki randevu görevi için adapte ediyoruz: doğrusal ortam sınırlı uzunluğa sahip olduğu zaman ve doğrusal ortam basit kapalı bir eğri olduğu zaman (dairesele döngü gibi). Algoritmamız her iki uzantıya da aynı performans garantileri ile adapte edilebilir. Algoritmamızın ilginç bir yönü her roundda gidilen mesafenin (hareket desenine ek olarak) rasgele olmasıdır. Simülasyonlar bu rasgeleliğin algoritmamızı hareketteki belirsizliğe göre, örneğin, odometri hatalarından dolayı, dayanıklı yaptığını gösterir.

Randevu arama probleminin bir diğer ilgi çekici çerçevesi de oyuncuların aramaya aynı zamanda başlamadığı asenkron durumdur. Şuana kadar olan çalışmalar problemin oyuncu asimetrik versiyonuna odaklanmıştır. Bölümün ikinci kısmında bu boşluğu problem üzerindeki ön sonuçlarımızı sunarak dolduruyoruz. İlk kısımda sunduğumuz simetrik algoritmayı bu duruma uygun şekilde değiştiriyoruz. Değişmiş simetrik algoritmamızın ön performans analizini sunuyoruz. Elde ettiğimiz bu ön sonuçlar bildiğimiz kadarıyla problemin bilinmeyen uzaklıkta, simetrik versiyonu ve asenkron durumu için literatürde sunulan ilk sonuçtur. Bu kısımdaki amacımız bu versiyon için öncelikli sonuçları elde etmektir. Bu nedenle, analizi mümkün olduğunca basit tutmaya çalıştık. Simülasyonlarla teorik sonuçlarımızı doğruluyoruz ve başlama zamanlarının aynı olduğu varsayımımızın önemini inceliyoruz. Simülasyon sonuçları,

robotların başlama zamanlarındaki farklılıklara karşı dayanıklı olduğunu gösterir; robotlar farklı zamanlarda başlasalar bile kompetitif oran önemli ölçüde artmaz.

Bu bölüm şu şekilde organize edilmiştir: İlk olarak 4.1. kısımda literatürdeki doğru üzerinde randevu arama problemi ile ilgili çalışmaları sunuyoruz. Bu problem literatürde çok çeşitli formülizasyonlar için çalışılmıştır. 4.3. ve 4.4. kısımlarında, sırasıyla problemin senkron ve asenkron durumları için yaptığımız çalışmalarımızı sunuyoruz. 4.2.1. kısımda, çalıştığımız problemle ilgili online bir problemi sunuyoruz. Daha sonra, simetrik randevu stratejimizi ve stratejimizin mesafe kompetitif oranının hesaplanmasını sırasıyla 4.3.1. ve 4.3.2. kısımlarında çalışıyoruz. Analizimize ek hesaplamaları ayrı olarak 4.3.3. kısmı altında topluyoruz. 4.3.6. kısımda, algoritmamızın sınırlı ve dairesel ortamlara olan iki uzantısını sunuyoruz. Son olarak, simülasyon ve gerçek ortam deney sonuçlarımızı sırasıyla 4.3.5. ve 4.3.7. kısımlarında raporlandırıyoruz. 4.4. kısımda, asenkron durum için ön sonuçlarımızı sunuyoruz. Simetrik randevu stratejimizi bu durum için nasıl değiştirdiğimizi 4.4.1. kısımda açıkladıktan sonra 4.4.2. kısımda değişmiş stratejimizin analizini sunuyoruz. Son olarak, simülasyon sonuçlarımızı 4.4.3. kısımda raporlandırıyoruz.

4.1. İlgili Çalışmalar

Oyuncuların bir doğru olarak modellenebilecek herhangi bir arama bölgesine yerleştirildiğini ve en kısa zamanda buluşmak istediklerini düşünelim. Doğru üzerinde randevu arama problemi hem simetrik Alpern (1995); Anderson and Essegaiier (1995); Baston (1999); Han et al. (2007); Stachowiak (2009); Uthaisombut (2006) hem de asimetrik Alpern and Beck (1999, 2000); Baston and Gal (1998); Gal (1999) versiyon için çalışılmıştır. Oyuncuların arasındaki uzaklığın veya dağılımın bilindiği durum Alpern (1995); Alpern and Beck (1997, 1999); Anderson and Essegaiier (1995); Baston (1999); Stachowiak (2009)'de, bilinmediği durum Alpern and Beck (2000); Baston and Gal (1998, 2001)'da çalışılmıştır.

Teorik önemine ek olarak, doğru üzerinde randevu robotik uygulamaları için de önemlidir: doğru, demiryolları, kabloları veya uzun koridorları modellemek için kullanılabilir. Orijinal simetrik formülasyonda, oyuncular başlangıç uzaklığı olan $2d$ 'i bilerek başlar. fakat oyuncular diğer oyuncunun yönünü veya doğru üzerindeki kendi yönünü bilmez ve bir birim hızında hareket eder Bu

formulasyon için, Alpern Alpern (1995)'de şu stratejiyi tanıtmıştır: her periyodun başında, her oyuncu bağımsız olarak rasgele bir yön seçer ve d birim ileri (*Ing. forward* (F)) ve $2d$ birim geri (*Ing. backwards* (B)), ters yönde ilerlerler. Robot bu rasgele 1F2B hareket desenini buluşana kadar takip eder. Yıllar içinde bu algoritmanın analizi iyileştirilmiştir ve Alpern tarafından başta garantilenmiş olan 5-kompetitif oranı, Anderson ve Essegai Anderson and Essegai (1995) tarafından 4.5678'e, Baston ve Gal Baston and Gal (1998) tarafından 4.4182'e, Uthaisombut Uthaisombut (2006) tarafından 4.39306'a ve Han ve arkadaşları Han et al. (2007) tarafından (4.1520, 4.2574) (üst sınır, alt sınır)'e düşürülmüştür. Han et al. (2007)'daki geliştirilmiş sınırlara ulaşmak için, markov zincir teorisi (*Ing. markov chain theory*) ve matematiksel programlama teorisinden (özellikle de, kesirli karesel programlamadan (*Ing. fractional quadratic programming*) ve yarısonlu programlamadan (*Ing. semidefinite programming*)) yararlanılmıştır.

Baston and Gal (1998)'de Baston ve Gal, başlangıç uzaklığının, beklenen değeri $E[2d] = \mu$ olan bir dağılımdan üretildiği bir simetrik randevu formulasyonu düşünür. Beklenen buluşma zamanı 13.325μ olan simetrik bir algoritma sağlanır, ki bu da 26.650 kompetitif orana sahiptir. Maksimum D mesafeli bir uzaklık dağılımı için, Baston Baston (1999) kompetitif oranı $1.71 + \mu/2D$ olan bir algoritma geliştirmiştir.

Bu tezde, doğru üzerindeki randevu problemini robotların aralarındaki başlangıç uzaklığını bilmediği durumda düşünüyoruz. Oyuncular global bir iskelet paylaşmazlar. Bilinmeyen uzaklıkta Alpern and Beck (2000); Stachowiak (2009), asimetrik randevu çalışılmıştır, fakat simetrik versiyon için iyi çalışılmamıştır. Baston and Gal (1998)'de olan Baston ve Gal'in algoritmasında, uzaklık dağılımı bilinmez ve bu nedenle bu formulasyon için 26.650-kompetitif (hem uzaklık hem de zaman için) orana sahip bir algoritmadır.

n tane kör, hızları bir birim olan oyuncunun rasgele sağ veya sol yönünü göstererek, rasgele bir permütasyon ile 1'den n 'e kadar olan tamsayıların üzerine yerleştirildiğini varsayalım. $m < n$ tanesinin tek bir noktada buluşması için gerekli olan en az beklenen zaman nedir? Lim ve arkadaşları, bu randevu probleminin, doğru üzerinde hem simetrik hem de asimetrik versiyonlarını çalışmıştır Lim et al. (1997). Asimetrik randevu değeri olan $R_{N,M}^a$ 'in değerinin $47/48$ olduğu ve simetrik randevu değeri olan $R_{N,M}^s$ 'in değerinin $n/2$ 'ye asimtotik olduğu gösterilir. Problemin n -oyunculu olan simetrik versiyonunda, her oyuncu aynı rasgeleleştirilmiş stratejiyi kullanır. Başlangıçta 1'den n 'e kadar

olan pozisyonlara yerleştirilen oyuncuların aralarındaki başlangıç uzaklığının $(n-1)$ olmasından dolayı randevu zamanı olan T , asla $(n-1)/2$ 'den daha küçük olamaz. Yazarlar problemin bu versiyonunda, n 'den bağımsız sabit bir K değeri için, $n/2 + K$ beklenen randevu zamanına ulaşacak rasgeleleştirilmiş bir strateji üretmeyi amaçlar.

Alpern ve Lim Alpern (2002), doğru üzerindeki randevu problemini, rasgele olarak sağ veya sol yöne yönelen ve doğru üzerinde üç ardışık tamsayı değere yerleştirilen üç ajan ile çalışmıştır. Elde edilen ana sonuç V. Baston tarafından verilen stratejinin bir minimax stratejisi olduğunun kanıtlanmasıdır.

Gal Gal (1999), iki oyunculu randevu arama problemi için, bir oyuncunun bir arama periyodu boyunca sabit durmasının asla optimal olmadığını gösterir. Gal aynı zamanda, en kötü durumda n -oyuncunun buluşma zamanını da düşünür.

Doğru üzerinde randevu arama probleminin asenkron durumu literatürde çok ilgi görmemiştir. Problemin bu durumu ile ilgilenen çalışmalar çizge Chalopin et al. (2007); Czyzowicz et al. (2010b); Dessmark et al. (2006); Dieudonné et al. (2013); Kowalski and Pelc (2005); Lin et al. (2006); Marco et al. (2006) veya geometrik ortamlar Bampas et al. (2010); Collins et al. (2010); Czyzowicz et al. (2010a,b); Dieudonné and Pelc (2014); Lin et al. (2007) üzerinde randevuya veya toplanmaya Agmon and Peleg (2004); Cieliebak (2004); Cieliebak et al. (2003); Cieliebak and Prencipe (2002); Flocchini et al. (2001); Ganguli et al. (2006); Jurek et al. (2009); Prencipe (2005); Souissi et al. (2006) odaklanmıştır.

İlgili bu çalışmaları 3. Bölümde tartışıyoruz. Marco et al. (2006), çizge üzerinde asenkron randevuya konsentre olsada, yazarlar doğru üzerindeki ajanlar için kararlı bir randevu algoritması sunmuştur. Bir ajanın hareketine müdahale edecek bir düşmanın (*Ing. adversary*) olduğunu düşünürler. Eğer ajanlar aynı kararlı algoritmayı çalıştıırırlarsa ve düşman onları aynı yönde ve hızda hareket ettirirse, o zaman asla buluşamazlar. Bu nedenle, ajanlar *etiket* olarak adlandırılan ayrı kimliklere sahiptir. Etiketler iki farklı boş olmayan dizidir (*Ing. string*) ve her ajan kendi etiketini bilir. Etiketine bağlı olarak, her ajan L^* etiketini üretir. Bu bit dizisi, üç ardışık parçadan oluşan ve ajan tarafından takip edilen bir hareket desenidir. Asenkron ayarlamadan dolayı, t zamanında, ajan X , p . bitin ikinci parçasını bitirirken ajan Y , p . biti yeni

çalıştırıyor olabilir. Algoritmaları, D bilindiği zaman $O(D|L_{min}|^2)$ maliyete ve bilinmediği zaman $O((D + |L_{max}|)^3)$ maliyete sahiptir. Burada $|L_{min}|$ ve $|L_{max}|$ sırasıyla ajanların kısa ve uzun etiketlerini gösterir. Bu sınır Stachowiak Stachowiak (2009) tarafından $O(D \log^2 D + D \log D |L_{max}| + D |L_{min}|^2 + |L_{max}| |L_{min}| \log |L_{min}|)$ olarak iyileştirilmiştir.

Biz bu tezde, problemin asenkron durumunu simetrik versiyon için çalışıyoruz, bu nedenle robotlar için herhangi bir etiketlendirme yapmıyoruz. Bildiğimiz kadarıyla problemin simetrik versiyonunu bilinmeyen başlangıç uzaklığı için, asenkron durumda, doğru üzerinde araştıran bir çalışma sunulmamıştır. 4.4. Bölümde bu boşluğu bir randevu stratejisi ve onun ön analizini sunarak dolduruyoruz. Çalışmamızda, robotların hareketlerindeki senkronluk, bir robotun diğer robota göre simetrik randevu algoritmasını t zaman sonra başlattığı varsayımı ile bozulur.

4.2. Doğru Üzerinde Simetrik Randevu

Randevu problemi, ayrıca *lost cow problem* olarak da bilinen *lineer arama problemini* (Ing. linear search problem) genelleştirir. Bu formülasyonda, tek bir robot bilinmeyen duran bir objeyi bulmaya çalışır. Bu lineer arama problemi orijinal olarak Beck and Newman (1970)'da çözülmüştür ve bu çözüm Baeza-Yates et al. (1993)'da yeniden bulunmuştur. Bu sonuç, Gal (1980)'daki online algoritmaların çerçevesine yerleştirilmiştir (aynı zamanda ortak bir başlama noktasından yayılan M sonsuz ışına da genelleştirilmiştir). Bizim simetrik randevu algoritmamız *lost cow* problemini genelleştirir ve *lost cow* probleminden ilham alır. Bu nedenle, lineer arama probleminin tanımı ile başlıyoruz ve algoritmamızın açıklaması ve performans analiziyle devam ediyoruz.

4.2.1 *Lost Cow* Problemi

Uzağı göremeyen bir ineğin sadece bir tane olan uzun ve düz bir çitteki kapıyı bulmaya çalıştığını varsayalım. Bu kapının, ineğe bilinmeyen başlangıç uzaklığı d 'dir ve ineğin solunda veya sağında yer almaktadır. Kapının yerini bulmak için tasarlanmış herhangi kararlı online bir algoritma, ineğin başlangıç pozisyonu olan $x = 0$ 'dan sola daha sonra da sağa doğru dönüşümlü olarak

arama yapmasına eşittir. Bu gibi spiral yapan bir algoritma negatif-olmayan $f_1, f_2, \dots$ bir dizi olarak tanımlanabilir, burada f_i , tek (çift) round i 'da ineğin başlangıç pozisyonundan sola (sağa) aranan mesafedir. Elverişlilik açısından, $f_0 = 0$ olarak belirliyoruz. i . roundda, inek $f_{i-1} + f_i$ kadar toplam mesafe ilerler.

Kararlı *Spiral LostCow* algoritması ikiye katlama, $f_i = 2^{i+1}$, stratejisi kullanır. Spiral LostCow 9-kompetitif bir algoritmadır ve bu bir kararlı on-line algoritmanın bilinen olası en iyi performansdır. Bu kompetitif oran biraz rasgelelik katılarak iyileştirilebilir. *SmartCow* algoritması, round i 'da gidilen $r^{i+\epsilon}$ mesafe hariç, aynı zig-zag stratejiyi takip eder, burada $\epsilon \in [0, 1)$ algoritmanın başında uniform olarak seçilir. Bu rasgelelik en kötü durum analizini düzeltir. Beck and Newman Beck and Newman (1970) $r \approx 3.591$ değerinin 4.591 kompetitif oranını garantilediğini kanıtladı, ki bu mümkün olan en iyi kompetitif orandır. Bu sonuç, Kao et al. Kao et al. (1994) tarafından yeniden keşfedildi.

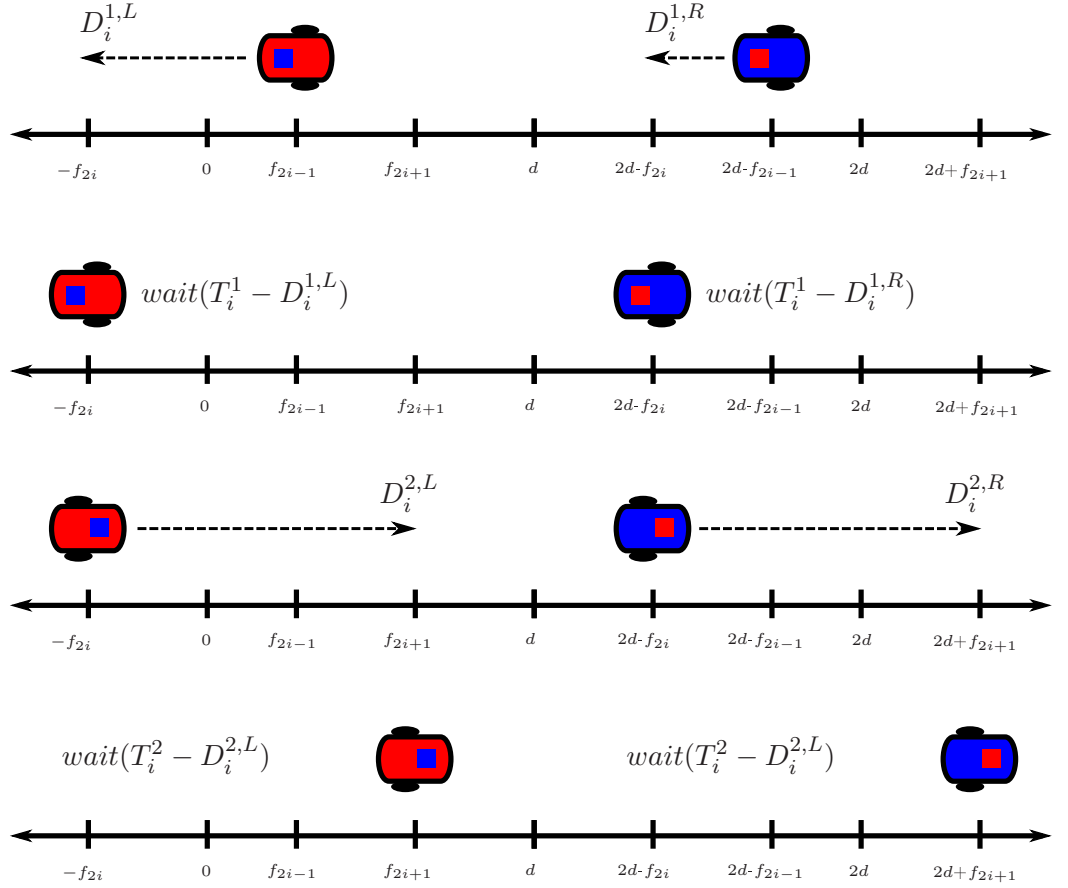
Şimdi randevu problemini *lost cow* problemi ile karşılaştırıyoruz. Tek bir hareketli inek ve duran bir kapı yerine, şimdi aralarındaki başlangıç uzaklığı $2d$ olan iki hareketli inek var. İneklerin amacı çit boyunca herhangi bir yerde buluşmaktır. Problemin simetrik versiyonu, her iki ineğin de aynı arama algoritmasını uygulamasını gerektirir. Bu kararlı stratejilere engel olur, çünkü senkron hareketleri robotların herhangi bir zamanda buluşmalarına engel olur.

4.3. Senkron Durum

4.3.1 Simetrik Randevu Arama Algoritması

Algitmamızı simetrik randevu ve robotların arasındaki başlangıç uzaklığının bilinmediği durum için sunuyoruz. Algitmamız robotların arasındaki simetriyi kırmak için rasgeleştirmeyi kullanıyor. Ayrıca, Spiral LostCow algoritmasının $2i$. ve $2i + 1$. roundlarını tek bir roundda birleştiren bir hareket deseni kullandık. Bu robotların yer değişimleri en az d birim olduğu zaman, robotların buluşma olasılığının $1/2$ olmasını garantiler.

Şimdi, robotların rasgeleleştirilmiş yolculuğunu açıklayacağız. Şekil 4.15'de



Şekil 4.2: Round i 'daki davranış. Robotlar $(f_{2i-1}, 2d - f_{2i-1})$ konfigürasyonundan başlıyor. Şuanki para atışları şuanki roundda her iki robotun da sola-sonra-sağa gitmesini gerektirir. **Birinci:** Birinci aşama hareketleri. Robot-1 $d_1 = r^{2i+\epsilon_1} + r^{2i-1+\epsilon_1}$ kadar sola hareket eder. Robot-2 $d_2 = r^{2i+\epsilon_2} - r^{2i-1+\epsilon_2}$ kadar sola hareket eder. **İkinci:** Birinci aşama durma zamanı. Senkron roundları garantilemek için, robot- j bir sonraki aşamaya başlamadan $r^{2i} + r^{2i+1} - d_j$ zaman birimi bekler. **Üçüncü:** İkinci aşama hareketi. Robot-1 $d_3 = r^{2i+\epsilon_1} + r^{2i+1+\epsilon_1}$ kadar sağa hareket eder ve robot-2 $d_4 = r^{2i+\epsilon_2} + r^{2i+1+\epsilon_2}$ kadar sağa hareket eder. **Dördüncü:** İkinci aşama durma zamanı. Robot j , $r^{2i+1} + r^{2i+2} - d_{2+j}$ zaman birimi bekler.

gösterildiği gibi, $2d$ robotlar arasındaki başlangıç uzaklığı olsun. Genellemeyi bozmadan, robot-1, $x = 0$ ve robot-2 $2d$ 'de konumlandırılmış olsun (göreceli pozisyonlarından habersiz olmalarına rağmen). Robotların konfigürasyonunu $\mathbb{R}^2$ 'da bir nokta olarak sunuyoruz, öyle ki başlangıç konfigürasyonu $(0, 2d)$ olsun. Robotlar aralarındaki başlangıç uzaklığını veya diğer robota olan yönlerini bilmiyorlar. Hiçbirşey bilmeyen robotlar için, en iyi offline algoritma robotların birbirlerine doğru ilerlemesi ve (d, d) konfigürasyonunda buluşması olacaktır. Bu nedenle, kompetitif oranımız uzaklık d 'ye göre hesaplanacaktır.

Her robot için, negatif olmayan bir $f_{-1}, f_0, f_1, f_2, \dots$ sırası oluşturduk. Burada $f_{-1} = 0$ ve

$$i \geq 0 \text{ için } f_i = r^{i+\epsilon}$$

'dir, burada $r > 0$ genişleme çapıdır (daha sonra en iyi hale edilecektir) ve $\epsilon \in (0, 1]$ uniform olarak dağıtılmış rasgele bir değişkendir. Robotlar aynı genişleme yarıçapı r 'ı kullanırlar, fakat algoritmanın başında ϵ değerlerini birbirlerinden bağımsız olarak seçerler. Robot seçtiği bu rasgele ϵ değerini algoritma boyunca kullanır (her roundda yeni bir ϵ değeri seçmek performansı arttırmaz). SmartCow algoritmasında olduğu gibi, bu rasgele değişken en kötü durum kompetitif oranı azaltarak daha iyi bir performans garantisi verir. Aslında, Beveridge et al. (2011)'da, $\mathcal{SR}$ algoritmasının, ϵ olmayan başka bir biçimini sunduk. Bu algoritmanın mesafe kompetitif oranının 19.166 ve zaman kompetitif oranının 26.888 olduğunu gösterdik. Rasgeleştirilmiş bir parametre olan ϵ performansı iyileştirir.

Eğer $r = 2$ olarak seçseydik (Spiral LostCow algoritmasında olduğu gibi), bizim simetrik randevu algoritmamızın kompetitif oranı sınırsız olurdu. Sezgisel olarak, robotların gittiği mesafenin yükselişi robotların buluşma olasılığından üstündür. r 'i ikiye katlamak yerine, algoritmamız, robotların f_i 'ini $r \approx 1.195$ 'un bir faktörü olarak arttırır (iyileştirilmiş olan bu değerin hesaplanması Teorem 1'in ispatında sunulmuştur). Başlangıç uzaklığı $2d$ verilerek, $\delta \in (0, 1]$ ve $k \in \mathbb{Z}^+$ 'i, $d = r^{k+\delta}$ 'e ulaşan özgün sayılar olarak belirleyelim.

Algoritma $i \geq 0$ olarak endekslenmiş olan roundlar halinde devam etmektedir (round 0'dan başlamak daha natürel bir endeksleme vermektedir). Her roundda robot dönüşümlü olarak hareket edecek veya bekleyecektir. Eğer robotlar bir roundda buluşmuş ise, o round başarılıdır; aksi halde basarisızdır. $x = 0$ konumundan başlayan robot-1'in hareketini açıklayarak başlıyoruz. Her round i 'da, robot bu rounddaki yolculuğuna (sağdan-sola veya soldan-sağa) karar vermek için bir bozuk para atar. Hareketin yönüne bağlı olmak üzere,

Algorithm 1 $\mathcal{SR}$: Bilinmeyen uzaklık için doğru üzerinde simetrik randevu stratejisi.

```

1:  $r \leftarrow 1.195$ 
2:  $\epsilon \leftarrow \text{random from } (0, 1]$ 
   {Note that  $f(j) = r^{j+\epsilon}$  below}
3:  $i \leftarrow 0$ 
4:  $\text{coinFlip} \leftarrow \text{random from } \{-1, 1\}$ 
5:  $\text{previousCoinFlip} \leftarrow \text{coinFlip}$ 
6: while  $\text{checkRendezvous()} \neq \text{true}$  do
7:   if  $i = 0$  then
8:      $D_{i,1} \leftarrow f(2i)$ 
9:   else if  $\text{coinFlip} \neq \text{previousCoinFlip}$  then
10:     $D_{i,1} \leftarrow f(2i) - f(2i - 1)$ 
11:   else
12:     $D_{i,1} \leftarrow f(2i - 1) + f(2i)$ 
13:   end if
14:    $W_{i,1} \leftarrow f(2i) + f(2i + 1) - D_{i,1}$ 
15:    $\text{moveTo}(\text{coinFlip} * f(2i))$ 
16:    $\text{wait}(W_{i,1})$ 
17:    $D_{i,2} \leftarrow f(2i) + f(2i + 1)$ 
18:    $W_{i,2} \leftarrow f(2i + 1) + f(2i + 2) - D_{i,2}$ 
19:    $\text{moveTo}(-1 * \text{coinFlip} * f(2i + 1))$ 
20:    $\text{wait}(W_{i,2})$ 
21:    $i \leftarrow i + 1$ 
22:    $\text{previousCoinFlip} \leftarrow \text{coinFlip}$ 
23:    $\text{coinFlip} \leftarrow \text{random from } \{-1, 1\}$ 
24: end while

```

her rounddu iki aşamaya böldük. i . roundda, robot, her bir konumun olasılığı $1/2$ olmak üzere $x = \pm f_{2i-1}$ konumlarının birisinden başlar. Eğer robot tura atarsa, o zaman birinci aşamada, $x = f_{2i}$ noktasına doğru sağa hareket eder ve ikinci aşamada $x = -f_{2i+1}$ noktasına doğru sola hareket eder. Eğer robot yazı atarsa, birinci aşamada $x = -f_{2i}$ noktasına doğru sola hareket eder ve ikinci aşamada $x = f_{2i+1}$ noktasına doğru sağa hareket eder. Başarısız bir $i \geq 0$. roundun sonunda, robotların olası konfigürasyonları $(\pm f_{2i+1}, 2d \pm f_{2i+1})$ 'dir. Bu aynı zamanda round $i + 1$ için başlangıç konfigürasyonudur.

$D_{i,1}$ ($D_{i,2}$) round i 'in birinci (ikinci) aşamasında gidilen mesafe ve $D_i = D_{i,1} + D_{i,2}$ bu roundda gidilen toplam mesafe olsun. $D_{i,1}$ 'in değeri bu rounddaki para atışı sonucunun bir önceki rounddaki para atışının sonucundan farklı olmasına bağlıdır. Eğer gelen bu paralar farklı ise, o zaman $D_{i,1} = f_{2i} - f_{2i-1}$ 'dir. Eğer aynı ise, o zaman $D_{i,1} = f_{2i} + f_{2i-1}$ 'dir. $f_{-1} = 0$ olduğundan bu ifadelerin $i = 0$ için de doğru olduğunu not edelim. Para atışından bağımsız olarak, $D_{i,2} = f_{2i+1} + f_{2i}$ 'dir. Başarısız bir round i 'da robot tara-

findan gidilen mesafe (yolculuğun uzunluğu) $D_i = f_{2i+1} + 2f_{2i} - f_{2i-1}$ veya $D_i = f_{2i+1} + 2f_{2i} + f_{2i-1}$ 'dir, her birisi eşit olasılığa sahiptir.

Robotların hareketleri için farklı ϵ değerleri kullandığını hatırlayalım. Robotların hareketlerini senkron tutabilmek için, her hareketten sonra (sağa veya sola) bekleme zamanları tanıttık. Bir robot en kötü durum olan diğer robotun $\epsilon = 1$ olarak seçtiği durumu düşünmelidir. Bu nedenle birinci aşama için $T_{i,1} = r^{2i} + r^{2i+1}$ zamana ve ikinci aşama için $T_{i,2} = r^{2i+1} + r^{2i+2}$ zamana ihtiyacı vardır. Her hareketten sonra, robot diğer robotun şuanki hareketini bitirmesine yetecek kadar bekleyecektir. Bu bekleme zamanları ile robotun hareketlerini senkronize etmek, robotların yerdeğişimleri yeterince büyük olduğunda, robotların buluşma olasılığının $1/2$ olmasını garantiliyecektir.

Robotun round i 'daki aktivitesini özetliyoruz. Robot her round başında para atışı yapar. Birinci aşamada, robot $D_{i,1}$ mesafe ilerler ve $W_{i,1} = T_{i,1} - D_{i,1}$ zaman bekler. İkinci aşamada, robot $D_{i,2}$ ilerler ve $W_{i,2} = T_{i,2} - D_{i,2}$ zaman bekler. Roundun başarısız olduğunu varsayarak, robotun ϵ değerinden ve para atışı sonucundan bağımsız olarak, round i 'da geçen zaman $T_i = T_{i,1} + T_{i,2}$ 'dir.

Algoritma, robot konfigürasyonu ilk kez (x, x) formunda olduğu zaman sonlanır. Stratejinin pseudocode'u, Algoritma 1'de sunulmuştur. Ana sonucumuz aşağıdaki teoremdedir.

Theorem 1. *Simetrik randevu algoritması için optimal genişleme yarıçapı $r \approx 1.195$ 'dir. r 'in bu seçimi, 17.686-mesafe kompetitif ve 24.843-zaman kompetitif bir algoritma verir.*

Teoremi bir sonraki bölümde ispatlıyoruz.

4.3.2 Analiz

$d = r^{k+\delta}$ için Algoritma $\mathcal{SR}$ 'in performans analizi k 'nin çift veya tek olmasına bağlıdır. Çift olduğu durumu analizi etmek biraz daha zor olduğundan bu durum için mesafe kompetitif oranı analiz ettik. Diğer analizler (çift zaman, tek mesafe ve tek zaman) aynı şekilde analiz edilmektedir. Bunların analizini de özet bir şekilde yaptık.

Bundan dolayı, k 'ı çift olarak varsaydık. Algoritma $\mathcal{SR}$ 'in çalışmasını dört aşamaya böldük. *Stage-1* uzunluğu $2d = 2r^{k+\delta}$ 'ye bağlı olan baslangic

asamasidir. Bu aşama, robotların buluşabilecekleri kadar ilerlemedikleri, $0 \leq i \leq k/2 - 1$ olan önceki roundlardan oluşur. Robotların ilk buluşabileceği round $k/2$ 'dir. *Stage-2* sadece bu kritik rounddan oluşur. Round $k/2 + 1$ için ayrı bir aşama olarak *Stage-3*'ü tanımlıyoruz. Bu roundda, robotların başlangıç konumları geçici bir davranış sergiler. *Stage-4*, *Stage-3*'den sonraki tüm roundları içerir. Daha sonraki bu roundlarda, başlangıç konfigürasyonları ve buluşma davranışları istikrarlıdır, bu nedenle sonsuz toplam kullanarak beklenen gidilen mesafeyi (*Ing.* expected distance traveled) hesaplayabiliriz.

Analizimizde, her robotun hareket yönünü (sol-sonra-sağ veya sağ-sonra-sol) takip ettik. *Stage-2*'da, ϵ_1, ϵ_2 'ye göre δ 'nin göreceli büyüklüğünü düşündük. *Stage-3*'de, olası başlangıç konfigürasyonlarına dikkat etmemiz gerekir. Ayrıca *Stage-2*'dan başlayarak, robotların bu roundda buluşup buluşmama-larına göre her roundda gidilen mesafeyi hesaplamak gerekli olduğunu not edelim.

Simdi analizimizde kullanacağımız değişkenleri tanıtalım. R_i , round i 'da algoritmanın hala aktif olma olayı olsun. R_i olayının olduğunu varsayarak, şunları tanımlıyoruz: S_i , round i 'da robotların başarılı olarak buluşma olayıdır. A_i , round i 'da robot-1'in önce sağa (robot-2'ye doğru) ilerlemesi olayıdır; B_i , round i 'da, robot-2'nin önce sola (robot-1'ye doğru) ilerlemesi olayıdır.

Stratejilerin simetrik olmasından dolayı, beklenen gidilen mesafe her iki robot için de aynıdır. Bu nedenle, sadece robot-1'i analiz ediyoruz. R_i 'in hala olduğunu varsayarak, tekrar $D_i = D_{i,1} + D_{i,2}$ ve $T_i = T_{i,1} + T_{i,2}$ ve $W_i = W_{i,1} + W_{i,2}$ olarak tanımlıyoruz.

Algoritma birinci roundda her zaman aktiftir, bu nedenle $\mathbb{P}[R_0] = 1$ 'dir. Algoritmanın round i 'da hala aktif olma olasılığı robotların round i 'a kadar olan roundlarda buluşmama olaylarının ortak olasılığıdır (*Ing.* joint probability). Bu, $i > 0$ için, $\mathbb{P}[R_i] = \mathbb{P}[\overline{S}_0 \wedge \dots \wedge \overline{S}_{i-1}]$ 'dir. Bu sembolleri kullanarak bir önceki bölümdeki görüşlerimizi yeniden ifade ettik. Bunlar

$$\begin{aligned} \mathbb{E}[D_i \mid \overline{S}_i] &= \mathbb{E}[f_{2i+1} + 2f_{2i} \mid \overline{S}_i] \\ &= (r^{2i+1} + 2r^{2i})\mathbb{E}[r^\epsilon \mid \overline{S}_i], \end{aligned} \quad (4.1)$$

$$\mathbb{E}[T_i \mid \overline{S}_i] = r^{2i+2} + 2r^{2i+1} + r^{2i}, \quad (4.2)$$

$$\mathbb{E}[W_i \mid \overline{S}_i] = \mathbb{E}[T_i \mid \overline{S}_i] - \mathbb{E}[D_i \mid \overline{S}_i]. \quad (4.3)$$

Stage-1 sırasında, $\mathbb{E}[r^\epsilon \mid \overline{S}_i] = \mathbb{E}[r^\epsilon]$ 'dir. Daha sonraki aşamalar sırasında, (yani, $i \geq k/2$ iken), ϵ 'un boyutu ve S_i olayı öncelikli değişkenlerdir.

Şimdi, çift k için Algoritma $\mathcal{SR}$ 'in dört aşamasını çalışmaya hazırız. Bölüm 4.3.2, 4.3.2, 4.3.2 ve 4.3.2 sırasıyla, stage 1-4 aşamalarının mesafe analizini sunar. Gösterimin anlaşılabilirliği açısından, belirli olasılıkların ve beklenen değerlerin değerlendirmesini, çift k için mesafe kompetitif oranı ispatladığımız ana teoremimizin olduğu Bölüm 4.3.2'ye erteledik.

Çift k için Stage-1 Analizi

$d = r^{k+\delta}$ olduğunu hatırlatalım, burada $\delta \in (0, 1]$ 'dir. Stage-1'de beklenen gidilen mesafeyi hesaplıyoruz. $0 \leq i \leq k/2 - 1$ iken, robotların birleşik yer değişimleri

$$r^{2i+\epsilon_1} + r^{2i+\epsilon_2} \leq 2r^{2i+2} \leq 2r^k < 2r^{k+\delta} = d$$

'dir. Bu nedenle, robotlar Stage-1'de buluşamazlar. Başka bir deyişle, $0 \leq i \leq k/2 - 1$ için $\mathbb{P}[R_i] = 1 = \mathbb{P}[\bar{S}_i]$ 'dir.

Lemma 2. *Stage-1 sırasında beklenen gidilen mesafe şu şekildedir*

$$\sum_{i=0}^{k/2-1} \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] < (r+2) \frac{r^k}{r^2-1} \mathbb{E}[r^\epsilon].$$

Ayrıca, Round $k/2 - 1$ 'i düşünerek, dört final konfigürasyonu $(\pm f_{k-1}, 2d \pm f_{k-i})$ 'dir, her birisi $1/4$ olasılıkla olur.

İspat. Denklem (4.1)'den,

$$\begin{aligned} \sum_{i=0}^{k/2-1} \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] &= \sum_{i=0}^{k/2-1} \mathbb{E}[D_i \mid \bar{S}_i] \mathbb{P}[\bar{S}_i] \\ &= \sum_{i=0}^{k/2-1} (r^{2i+1} + 2r^{2i}) \mathbb{E}[r^\epsilon \mid \bar{S}_i] \cdot 1 \\ &= (r+2) \mathbb{E}[r^\epsilon] \sum_{i=0}^{k/2-1} r^{2i} = (r+2) \frac{r^k - 1}{r^2 - 1} \mathbb{E}[r^\epsilon]. \end{aligned}$$

Çift k için Stage-2 Analizi

Bu alt bölümde kritik round, $k/2$ 'deki beklenen gidilen mesafeyi sınırlandırıyoruz. Stage-1'den farklı olarak, robotlar sıfırdan farklı olasılıkla buluşur. Başarılı ve başarısız bir $k/2$. rounddaki gidilen mesafeyi ayrı ayrı sırasıyla

Lemma 4 ve Lemma 5'de hesaplıyoruz. Bu ifadeleri toplamak kritik rounddaki toplam beklenen gidilen mesafeyi verir.

Robotlar kritik rounddan önce buluşamazlar; bu nedenle, $\mathbb{P}[R_{k/2}] = 1$ 'dir. Ayrıca, $S_{k/2} \wedge R_{k/2} = S_{k/2}$ ve $\overline{S}_{k/2} \wedge R_{k/2} = \overline{S}_{k/2}$ 'dir. Şimdi, kritik round için başarı ve başarısızlık durumlarını inceleyeceğiz. C 'yi $2\delta \leq \epsilon_1 + \epsilon_2$ olayı olarak tanımlayalım. $A_{k/2} \wedge B_{k/2} \wedge C$ olduğu zaman, robotların round $k/2$ 'in ilk aşamasında başarılı bir şekilde buluştuklarını iddia ediyoruz. Aslında, round $k/2$ 'nin birinci aşaması, robotlar ancak ve ancak $2r^\delta \leq r^{\epsilon_1} + r^{\epsilon_2}$, veya eşit bir şekilde $\epsilon_2 \geq \log_r(2r^\delta - r^{\epsilon_1})$ olursa buluşurlar. Bu fonksiyon $0 \leq \epsilon_1 \leq 1$ için iç bükey olarak aşağıdadır, bu yüzden fonksiyon (δ, δ) noktasından geçen tanjant doğrusunun aşağısında yer alır. Bu tanjant doğrusu için formül, $\epsilon_2 = 2\delta - \epsilon_1$ 'dur. Bu nedenle, eğer $2\delta \leq \epsilon_1 + \epsilon_2$ ise, o zaman $2r^\delta \leq r^{\epsilon_1} + r^{\epsilon_2}$ 'dir.

Analizin sadeliği için, $\epsilon_2 = 2\delta - \epsilon_1$ tanjant doğrusunu birim karede $\epsilon_2 = \log_r(2r^\delta - r^{\epsilon_1})$ fonksiyonuna yaklaşmak için kullanıyoruz. Tanjant doğrusu bu bölgede eğrinin üzerinde yer alır. Bu nedenle, C olayını $A_{k/2} \wedge B_{k/2}$ olayı olduğunda başarı için bir proxy olarak kullanıyoruz. Bu algoritma sırasındaki beklenen gidilen mesafeye bir üst sınır sağlar (çünkü round $k/2$ 'deki bazı başarıları da başarısızlık olarak sayarak yanlış kategorize ediyoruz). Aşağıdaki hesaplamalarda, optimal genişleme yarıçapının $1 \leq r \leq \sqrt{2}$ 'i sağladığını buluyoruz. r in bu değerleri için, diğer faktörlerle karşılaştığımızda oluşan hata çok küçüktür.

Bunu akılda tutarak, $A_{k/2} \wedge B_{k/2}$ olayı olduğunda, başarıyı C olayı ile karşılaştırmak olarak tanımlıyoruz. C olayının daha sonraki aşamalarda analizimizde rol alacağını not edelim.

Lemma 3. *Kritik roundun başarılı olma olasılığı*

$$\mathbb{P}[S_{k/2}] = (1 + \mathbb{P}[C])/4.$$

Buna ek olarak, başarısız kritik bir round için $x = (x_1, x_2)$ bitiş konfigürasyonu şunları sağlar

$$\begin{aligned} \mathbb{P}[x = (-f_{k+1}, 2d - f_{k+1}) \mid \overline{S}_{k/2}] &= 1/4, \\ \mathbb{P}[x = (f_{k+1}, 2d + f_{k+1}) \mid \overline{S}_{k/2}] &= 1/4, \\ \mathbb{P}[x = (-f_{k+1}, 2d + f_{k+1}) \mid \overline{S}_{k/2}] &= \mathbb{P}[\overline{C}]/4. \end{aligned}$$

İspat. Robotlar iki yolla buluşurlar. Birinci yol, roundun birinci aşamasında

robotların her ikisi de birbirlerinden uzaklaştıkları zamandır. Bu olaya

$$E_1 = \overline{A}_{k/2} \wedge \overline{B}_{k/2}$$

diyelim. İkinci yol

$$E_2 = A_{k/2} \wedge B_{k/2} \wedge C$$

olayı olduğundadır. $\mathbb{P}[E_1 \vee E_2] = \mathbb{P}[E_1] + \mathbb{P}[E_2] = 1/4 + \mathbb{P}[C]/4$.

Başarısız bir round $A_{k/2} \wedge \overline{B}_{k/2}$, $\overline{A}_{k/2} \wedge B_{k/2}$ veya $A_{k/2} \wedge B_{k/2} \wedge \overline{C}$ olaylarından birisine karşılık gelir. Lemma'da listelenen bitiş konfigürasyonları ve olasılıkları bu üç olaya karşılık gelir.

İlerlemeden önce, önemli bir yorum yapalım: C olayının davranışı girdi değişkeni δ 'nin (bilinmeyen) değerine bağlıdır. Özellikle, $\Pr[C]$, $\Pr[\overline{C}]$, ve $\mathbb{E}[r^\epsilon \mid C]$, $\mathbb{E}[r^\epsilon \mid \overline{C}]$ için olan formüller δ 'nin $1/2$ 'den büyük veya küçük olmasına bağlıdır.

C ile ilgili hesaplamalar Bölüm 4.3.3'de bulunuyor. Teorem 1'nin final analizi sırasında, algoritmamızın kompetitif oranı, maksimum oranı verecek δ değerini seçerek hesaplanacaktır. δ girdi değişkenine olan bu bağımlılık, belirli olasılıkları ve beklentilerin değerlendirmelerini 4.3.2. Bölüme kadar ertelememizin nedenlerinden biridir.

Lemma 4. *Başarılı bir $k/2$. rounddaki beklenen gidilen mesafe*

$$E[D_{k/2} \mid S_{k/2}] \Pr[S_{k/2}] = \frac{1}{2} r^k \mathbb{E}[r^\epsilon] + d \left(\frac{1 + \Pr[C]}{4} \right).$$

İspat. Lemma 3'in ispatında bulunan aynı E_1, E_2 notasyonlarını kullanıyoruz. Başarılı bir $k/2$. rounddaki beklenen gidilen mesafe şöyle ifade edilir

$$\begin{aligned} & \mathbb{E}[D_{k/2} \mid S_{k/2}] \Pr[S_{k/2}] \\ &= \mathbb{E}[D_{k/2} \mid E_1] \Pr[E_1] + \mathbb{E}[D_{k/2} \mid E_2] \Pr[E_2], \end{aligned} \quad (4.4)$$

Burada $\Pr[E_1] = 1/4$ ve $\Pr[E_2] = \Pr[C]/4$ 'dir.

Daha sonra E_1 ve E_2 olaylarının her birisi için beklenen gidilen mesafeyi hesaplarız. Bu rounddaki dört olası başlangıç konfigürasyonu $(\pm f_{k-1}, 2d \pm f_{k-1})$ 'dir. E_1 olayının olduğu verilerek, eşit olasılıklı olan başlangıç konfigürasyonlarını karşılık gelen robot-1 tarafından gidilen beklenen mesafe, denklem (4.1)'e benzer bir şekilde hesaplanır, şu elde edilir

$$\mathbb{E}[D_{k/2} \mid E_1] = \mathbb{E}[2f_k + d \mid E_1] = 2r^k \mathbb{E}[r^{\epsilon_1}] + d.$$

E_2 olayının olduğu verilerek, robot-1 tarafından gidilen beklenen mesafe

$$\begin{aligned} \mathbb{E}[D_{k/2} \mid E_2] \\ = \mathbb{E} \left[\frac{1}{4} (2(d - f_{k-1}) + 2(d + f_{k-1})) \right] = d. \end{aligned}$$

Denklem (4.4)'deki bu değerleri kullanmak ispatı tamamlar.

Lemma 5. $k/2$. roundda gidilen beklenen mesafe

$$\begin{aligned} \mathbb{E}[D_{k/2} \mid \overline{S}_{k/2}] \mathbb{P}[\overline{S}_{k/2}] \\ = r^k(2 + r) \left(\frac{1}{2} \mathbb{E}[r^{\epsilon_1}] + \frac{1}{4} \mathbb{E}[r^{\epsilon_1} \mid \overline{C}] \mathbb{P}[\overline{C}] \right). \end{aligned}$$

İspat. Bu roundda, robotlar iki durum altında buluşamazlar. Birincisi, eğer aynı şekilde hareket ederlerse buluşamazlar. Başka bir deyişle,

$$E_3 = (\overline{A}_{k/2} \wedge B_{k/2}) \vee (A_{k/2} \wedge \overline{B}_{k/2})$$

olmuştur. İkincisi, eğer birbirlerine doğru hareket ederlerse ve ϵ_1, ϵ_2 robotların buluşabilecekleri kadar büyük olmadıgındadır. Bir başka deyişle, olay

$$E_4 = A_{k/2} \wedge B_{k/2} \wedge \overline{C}$$

olur. Başarısız bir $k/2$. roundda beklenen gidilen mesafe şu şekilde ifade edilir

$$\begin{aligned} \mathbb{E}[D_{k/2} \mid \overline{S}_{k/2}] \mathbb{P}[\overline{S}_{k/2}] \\ = \mathbb{E}[D_{k/2} \mid E_3] \mathbb{P}[E_3] + \mathbb{E}[D_{k/2} \mid E_4] \mathbb{P}[E_4]. \end{aligned} \quad (4.5)$$

$\mathbb{P}[E_3] = 1/2$ 'dir ve $\mathbb{P}[E_4] = \mathbb{P}[\overline{C}]/4$. Daha sonra, E_3, E_4 olaylarının her birisi için beklenen gidilen mesafeyi hesapladık. Denklem (4.1)'e benzer olarak, bu beklenen mesafeler şu şekildedir

$$\begin{aligned} E[D_{k/2} \mid E_3] &= r^k(2 + r) \mathbb{E}[r^{\epsilon_1}], \\ E[D_{k/2} \mid E_4] &= r^k(2 + r) \mathbb{E}[r^{\epsilon_1} \mid \overline{C}]. \end{aligned}$$

Denklem (4.5)'deki bu değerleri kullanmak lemmayı ispatlar.

Çift k için Stage-3 Analizi

Bu alt bölümde $(k/2 + 1)$. geçiş roundu için beklenen gidilen mesafeyi hesaplıyoruz. Robotların Stage-2'de buluşmadıklarını varsayarak, robotlar Stage-3'de ancak ve ancak

$$E_5 = (A_{k/2+1} \wedge B_{k/2+1}) \vee (\overline{A}_{k/2+1} \wedge \overline{B}_{k/2+1})$$

olayı olduğunda buluşurlar. Bu nedenle, $\Pr[S_{k/2+1} \mid R_{k/2+1}] = 1/2 = \Pr[\overline{S}_{k/2+1} \mid R_{k/2+1}]$ 'dir. Bir önceki bölümdeki olay notasyonunu kullanmaya devam ediyoruz. Stage-3'e, E_3 veya E_4 olaylarından hemen sonra giriyoruz ve bu durumları ayrı ayrı ele alıyoruz.

Lemma 6. *Round $k/2 + 1$ 'e E_3 olayından sonra girdiğimizi düşünelim. Şunlara sahibiz*

$$\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_3] = r^{k+2}\mathbb{E}[r^\epsilon] + d$$

ve

$$\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_3] = r^{k+2}(2 + r)\mathbb{E}[r^\epsilon].$$

Daha fazlası, $\overline{S}_{k/2+1}$ olduğu zaman, bitiş konfigürasyonu ya $(f_{k+3}, 2d + f_{k+3})$ 'dir yada $(-f_{k+3}, 2d - f_{k+3})$ 'dir, her birisinin olasılığı $1/2$ 'dir.

İspat. $S_{k/2+1} = E_5$ eşitliğini kullanarak, bu değerler denklem (4.1) ile aynı şekilde hesaplanır. Her iki $S_{k/2+1}$ $\overline{S}_{k/2+1}$, olayları için, iki başlangıç ve iki olası yolculuk konfigürasyonu vardır. Dört eşit olasılıklı olayın ortalamasını almak lemmayı verir.

E_4 olduğu anda, round $k/2 + 1$, pozisyon $(-f_{k+1}, 2d + f_{k+1})$ 'den başlar. Aşağıdaki lemmanın ispatı Lemma 6 ile aynı şekildedir.

Lemma 7. *Round $k/2 + 1$ 'e E_4 olayından sonra girdiğimizi düşünelim. Şuna sahibiz $\mathbb{E}[D_{k/2+1} \mid E_4] = \frac{1}{2}\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_4] + \frac{1}{2}\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_4]$, burada*

$$\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_4] = r^{k+2}\mathbb{E}[r^\epsilon \mid \overline{C}] + d$$

ve

$$\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_4] = r^{k+2}(2 + r)\mathbb{E}[r^\epsilon \mid \overline{C}].$$

Daha fazlası, $\overline{S}_{k/2+1}$ olduğu zaman, bitiş konfigürasyonu $(f_{k+3}, 2d + f_{k+3})$ veya $(-f_{k+3}, 2d - f_{k+3})$ 'dir, her birisinin olasılığı $1/2$ 'dir. $\square$

Şimdi tüm terimlerimizi tek bir ifade altında birleştiriyoruz.

Lemma 8. Round $k/2 + 1$ 'de beklenen gidilen mesafe şu şekildedir

$$\begin{aligned} & \mathbb{E}[D_{k/2+1} \mid R_{k/2+1}] \Pr[R_{k/2+1}] \\ &= \mathbb{E}[D_{k/2+1} \mid E_3] \Pr[E_3] + \mathbb{E}[D_{k/2+1} \mid E_4] \Pr[E_4] \end{aligned}$$

burada

$$\mathbb{E}[D_{k/2+1} \mid E_3] \Pr[E_3] = \frac{1}{4} (d + r^{k+2} (3 + r) \mathbb{E}[r^\epsilon])$$

ve

$$\begin{aligned} & \mathbb{E}[D_{k/2+1} \mid E_4] \Pr[E_4] \\ &= \frac{\Pr[\overline{C}]}{8} (d + r^{k+2} (3 + r) \mathbb{E}[r^\epsilon \mid \overline{C}]) . \end{aligned}$$

İspat. Round $k/2 + 1$ 'de başarı olasılığı her zaman $1/2$ 'dir, ve $\Pr[E_3] = 1/2$ ve $\Pr[E_4] = \Pr[\overline{C}]/4$ olduğunu biliyoruz. Sonuç Lemma 6 ve 7'den oluşur.

Çift k için Stage-4 Analizi

$i \geq (k/2 + 2)$ olan tüm roundlar için beklenen gidilen mesafeyi hesapladık. E_3 ve E_4 olayları hala daha sonraki roundlardaki beklenen değerleri etkiler. Özellikle, $i \geq k/2 + 2$ için, şunlara sahibiz

$$\begin{aligned} & E[r^\epsilon \mid R_i] \mathbb{P}[R_i] \\ &= E[r^\epsilon \mid R_i \wedge E_3] \mathbb{P}[R_i \mid E_3] \mathbb{P}[E_3] \\ &\quad + E[r^\epsilon \mid R_i \wedge E_4] \mathbb{P}[R_i \mid E_4] \mathbb{P}[E_4] \\ &= E[r^\epsilon] \mathbb{P}[R_i \mid E_3] \mathbb{P}[E_3] \\ &\quad + E[r^\epsilon \mid \overline{C}] \mathbb{P}[R_i \mid E_4] \mathbb{P}[E_4] . \end{aligned} \tag{4.6}$$

Gerçekte, eğer E_4 olayı olursa, o zaman $\overline{C}$ olayı da olur, bu da karşılığında r^ϵ 'nin beklenen değerini etkiler. Fakat, round $i \geq k/2 + 2$ 'in randevu davranışı, buraya varışımızdan bağımsızdır: randevuya her zaman $1/2$ olasılıkla ulaşırız. Bu davranış aşağıdaki lemmada özetleniyor.

Lemma 9. $i \geq k/2 + 2$ için, şuna sahibiz

$$\mathbb{P}[R_i \mid E_3] = \mathbb{P}[R_i \mid E_4] = \left(\frac{1}{2}\right)^{i-k/2-1} .$$

Buna ek olarak,

$$\begin{aligned} \mathbb{E}[D_i \mid R_i \wedge E_3] &= \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon]) , \\ \mathbb{E}[D_i \mid R_i \wedge E_4] &= \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon \mid \overline{C}]) . \end{aligned}$$

İspat. Kritik round $k/2$ 'de E_3 olayının olduğu verildiğinde lemmayı ispatladık. İspat E_4 için de benzer şekildedir. Round $k/2 + 1$ 'in başarısız olma olasılığı $1/2$ 'dir. Aynı şekilde, $(A_i \wedge B_i) \vee (\overline{A_i} \wedge \overline{B_i})$ olduğu zaman, her round $i \geq k/2 + 2$ yarı zamanda başarılıdır. Bu E_3 için olan ilk ifadeyi ispatlar.

Round i 'daki beklenen gidilen mesafeyi düşünerek, şunu buluruz

$$\begin{aligned}
\mathbb{E}[D_i \mid R_i \wedge E_3] &= \mathbb{E}[D_i \mid R_i \wedge E_3 \wedge S_i] \mathbb{P}[S_i] \\
&\quad + \mathbb{E}[D_i \mid R_i \wedge E_3 \wedge \overline{S_i}] \mathbb{P}[\overline{S_i}] \\
&= \frac{1}{2} (r^{2i} \mathbb{E}[r^\epsilon \mid E_3] + d) + \frac{1}{2} (r^{2i} (2 + r) \mathbb{E}[r^\epsilon \mid E_3]) \\
&= \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon]).
\end{aligned}$$

Bu E_3 için olan ispatı tamamlar. E_4 için olan ispatların tek farkı $\mathbb{E}[r^\epsilon \mid E_4] = \mathbb{E}[r^\epsilon \mid \overline{C}]$ olmasıdır.

Şimdi Stage-4 için beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 10. *Stage-4'de gidilen beklenen mesafe şudur*

$$\begin{aligned}
&\sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] \\
&= \frac{1}{4} \left(d + \frac{r^{k+4}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon] \right) \\
&\quad + \frac{\mathbb{P}[\overline{C}]}{8} \left(d + \frac{r^{k+4}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon \mid \overline{C}] \right).
\end{aligned}$$

İspat. Hesaplamamızı $R_i \wedge E_3$ and $R_i \wedge E_4$ 'ye bağlı olarak bölüyoruz. İlk olarak, $R_i \wedge E_3$ olayı olduğu zaman analizi yapıyoruz. Lemma 9'den şunlara sahibiz

$$\begin{aligned}
&\sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge E_3] \mathbb{P}[R_i \mid E_3] \mathbb{P}[E_3] \\
&= \sum_{i=k/2+2}^{\infty} \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon]) \left(\frac{1}{2} \right)^{i-k/2-1} \left(\frac{1}{2} \right) \\
&= \frac{1}{4} \sum_{j=0}^{\infty} (d + r^{2j+k+4} (3 + r) \mathbb{E}[r^\epsilon]) \left(\frac{1}{2} \right)^{j+1} \\
&= \frac{d}{8} \sum_{j=0}^{\infty} \left(\frac{1}{2} \right)^j + \frac{r^{k+4}(3+r)}{8} \mathbb{E}[r^\epsilon] \sum_{j=0}^{\infty} \left(\frac{r^2}{2} \right)^j \\
&= \frac{1}{4} \left(d + \frac{r^{k+4}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon] \right).
\end{aligned}$$

$R_i \wedge E_4$ için olan hesaplamalar tamamen aynı şekildedir ve şunları elde ederiz

$$\begin{aligned} & \sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge E_4] \mathbb{P}[R_i \wedge E_4] \mathbb{P}[E_4] \\ &= \frac{\mathbb{P}[\overline{C}]}{8} \left(d + \frac{r^{k+4}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon \mid \overline{C}] \right). \end{aligned}$$

***SR*'in Mesafe Kompetitif Oranının Hesaplanması**

Tüm dört aşamadaki beklenen gidilen mesafelere karar verdikten sonra, şimdi ana sonucumuzu ispatlayabiliriz. Algoritmanın 17.686-mesafe kompetitif ve 24.843-zaman kompetitif oranı olmasını garantileyen genişleme yarıçapı için optimal seçimin $r \approx 1.195$ olduğunu gösteriyoruz.

Proof of Theorem 1. Çift k için kompetitif oranın ispatını özet olarak yapıyoruz. Ek hesaplamalar Bölüm 4.3.3'de mevcuttur.

2, 4, 5, 8 ve 10 lemmalarındaki ifadeleri toplayarak toplam beklenen gidilen mesafe elde edilir. Bu ifade çok uzundur, o yüzden bu formülü burada dahil etmiyoruz. $d = r^{k+\delta}$ olduğunu hatırlayarak, r^k 'nin yerine, $d r^{-\delta}$ 'yi yerleştiriyoruz. Daha sonra, kompetitif oranı elde etmek için d 'ye bölüyoruz.

Daha sonra $\Pr[C]$, $\Pr[\overline{C}]$ olasılıklarını ve Bölüm 4.3.3'de bulunduğu gibi $\mathbb{E}[r^\epsilon]$, $\mathbb{E}[r^\epsilon \mid C] \Pr[C]$ ve $\mathbb{E}[r^\epsilon \mid \overline{C}] \Pr[\overline{C}]$ beklentilerini değerlendiriyoruz. C 'yi içeren değerlerin girdi değişkeni δ 'nın değerine bağımlılığı olduğunu not edelim. Özellikle, $0 < \delta \leq 1/2$ veya $1/2 < \delta \leq 1$ olmasına bağlı olarak hesaplamaların iki kümesini yapmak zorundayız. Her birisi için, mesafe kompetitif oranı en büyük yapacak δ değerini seçeceğiz.

$0 < \delta \leq 1/2$ için, bu ifade $\delta = 1/2$ olduğunda en büyük olur. Karşılığında, $r \approx 1.195$ seçimi 17.48 en küçük kompetitif oranını verir. Daha sonra $1/2 < \delta \leq 1$ için olan ifade, $\delta = 1$ için en büyük olur ve $r \approx 1.195$ seçimi 17.686 olan kompetitif oranı garantiler. Bu nedenle, herhangi bir d girdisi için, algoritmamız 17.686 kompetitif oranı garantiler.

Çift k için zaman analizi yukarıdaki hesaplamalarla aynı şekildedir. Bu 24.85 olan zaman kompetitif oranı verir. Bunun analizi özet olarak Bölüm

4.3.3'de verilmiştir. Tek k için zaman ve mesafe analizleri aynı zamanda yukarıdaki hesaplamalara çok benzerdir. Detaylar, Bölüm 4.3.3 ve 4.3.3'da özetlenmiştir. Burada tek gidilen mesafe kompetitif oranın 17.686 olduğunu ve zaman kompetitif oranın 23.58 olduğunu gösteriyoruz.

4.3.3 Ek Analiz Hesaplamaları

Bu bölümde, önceki bölümdeki analiz için gerekli olan temel hesaplamaları yapıyoruz ve o bölümde bulunan terminaloji ile devam ediyoruz. Daha sonra, çift k , için zaman kompetitif analizi özet olarak yapıyoruz ve daha sonra da tek k için mesafe ve zaman kompetitif analizi yapıyoruz.

İlgili Beklentiler ve Olasılıkları

$\epsilon = \log_r u$ 'i kullanarak, şuna sahibiz

$$\begin{aligned}\mathbb{E}[r^\epsilon] &= \int_0^1 r^\epsilon d\epsilon = \int_1^r r^{\log_r u} \frac{du}{u \log r} \\ &= \int_1^r \frac{du}{\log r} = \frac{r-1}{\log r}.\end{aligned}\tag{4.7}$$

C olayının $\epsilon_2 \geq 2\delta - \epsilon_1$ olduğunu hatırlayalım. $\mathbb{P}[C]$, $\mathbb{P}[\overline{C}]$, $\mathbb{E}[r^\epsilon \mid C]\mathbb{P}[C]$ ve $\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[\overline{C}]$ 'i hesaplıyoruz. Yalınlık açısından, ϵ_1, ϵ_2 'i, x, y olarak adlandıırıyoruz, böylece C olayı $y \geq 2\delta - x$ haline gelir, burada $x, y \in [0, 1]$ 'dir.

Bu değerleri bulmak birim kareye integrasyonu gerektirir. Önemle, integrasyonun limitlerinin δ 'ya bağlı olduğunu not edelim. Adlandıırırsak, iki durumu $0 < \delta \leq 1/2$ ve $1/2 < \delta \leq 1$ 'i ayrı şekilde düşünüyoruz, çünkü hebir durum ayrı hesaplamaları gerektirir.

Durum $0 < \delta \leq 1/2$ Bu durumda

$$\mathbb{P}[\overline{C}] = \int_0^{2\delta} \int_0^{2\delta-x} \partial y \partial x = 2\delta^2$$

ve bu nedenle

$$\mathbb{P}[C] = 1 - 2\delta^2.$$

Ayrıca şuna sahibiz

$$\begin{aligned}\mathbb{E}[r^{\epsilon_1} | \overline{C}] \mathbb{P}[\overline{C}] &= \int_0^{2\delta} \int_0^{2\delta-x} r^x \partial y \partial x \\ &= \frac{r^{2\delta} - 2\delta \log(r) - 1}{\log^2(r)}\end{aligned}\quad (4.8)$$

ve Denklem (4.7) ve (4.8)'i kullanarak, şunu elde edebiliriz

$$\mathbb{E}[r^{\epsilon_1} | C] \mathbb{P}[C] = \mathbb{E}[r^{\epsilon_1}] - \mathbb{E}[r^{\epsilon_1} | \overline{C}] \mathbb{P}[\overline{C}].$$

Durum $1/2 < \delta \leq 1$ Bu durumda

$$\mathbb{P}[C | \overline{F}] = \int_{2\delta-1}^1 \int_{2\delta-x}^1 \partial y \partial x = 2(1-\delta)^2$$

ve bu nedenle

$$\mathbb{P}[\overline{C}] = 1 - 2(1-\delta)^2.$$

Ayrıca şuna sahibiz

$$\begin{aligned}\mathbb{E}[r^{\epsilon_1} | C] \mathbb{P}[C] &= \int_{2\delta-1}^1 \int_{2\delta-x}^1 r^x \partial y \partial x \\ &= \frac{2(1-\delta)r^2 \log(r) - r^2 + r^{2\delta}}{r \log^2(r)}\end{aligned}\quad (4.9)$$

Denklem (4.7) ve (4.9)'i kullanarak, şunu elde edebiliriz

$$\mathbb{E}[r^{\epsilon_1} | \overline{C}] \mathbb{P}[\overline{C}] = \mathbb{E}[r^{\epsilon_1}] - \mathbb{E}[r^{\epsilon_1} | C] \mathbb{P}[C]$$

Çift k için Zaman Hesaplamaları

Bu bölümde, Çift k için zaman analizini özet bir şekilde yapıyoruz. Argumanlar, mesafe hesaplamalarına benzerdir. Asıl fark, her roundun zaman uzunluğu karardlıdır. Örneğin, başarısız bir round i 'da, birinci aşama $r^{i-1} + r^i$ zaman birimi sürer ve ikinci aşama $r^i + r^{i+1}$ zaman birimi sürer.

Çift k için Stage-1 Analizi Lemma 2'e benzer olarak, şuna sahibiz

$$\sum_{i=0}^{k/2-1} \mathbb{E}[T_i | R_i] \mathbb{P}[R_i] < (r^2 + 2r + 1) \frac{r^k}{r^2 - 1}. \quad (4.10)$$

Çift k için Stage-2 Analizi Lemma 4'e benzer olarak, şuna sahibiz

$$\begin{aligned} & \mathbb{E}[T_{k/2} \mid S_{k/2}] \Pr[S_{k/2}] \\ &= \frac{1}{4} (r^k + 2r^{k+1} + d) + \frac{\Pr[C]}{4} (r^k + d). \end{aligned} \quad (4.11)$$

Lemma 5'e benzer olarak, şuna sahibiz

$$\begin{aligned} & \mathbb{E}[T_{k/2} \mid \overline{S}_{k/2}] \mathbb{P}[\overline{S}_{k/2}] \\ &= r^k(r^2 + 2r + 1) \left(\frac{1}{2} + \frac{1}{4} \mathbb{P}[\overline{C}] \right). \end{aligned} \quad (4.12)$$

Çift k için Stage-3 Analizi Lemma 6'e benzer olarak, şuna sahibiz

$$\begin{aligned} \mathbb{E}[T_{k/2+1} \mid S_{k/2+1} \wedge E_3] &= r^{k+3} + d \\ \mathbb{E}[T_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_3] &= r^{k+2}(r^2 + 2r + 1). \end{aligned}$$

Lemma 7'e benzer olarak, şuna sahibiz

$$\begin{aligned} \mathbb{E}[T_{k/2+1} \mid S_{k/2+1} \wedge E_4] &= r^{k+2}(r + 1) + d \\ \mathbb{E}[T_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_4] &= r^{k+2}(r^2 + 2r + 1). \end{aligned}$$

Lemma 8'e benzer olarak, şuna sahibiz

$$\begin{aligned} & \mathbb{E}[T_{k/2+1} \mid R_{k/2+1}] \Pr[R_{k/2+1}] \\ &= \frac{1}{2} (d + r^{k+2}(r^2 + 3r + 1)) \mathbb{P}[E_3] \\ &\quad + \frac{1}{2} (d + r^{k+2}(r^2 + 3r + 2)) \mathbb{P}[E_4] \\ &= \left(\frac{1}{4} + \frac{\mathbb{P}[\overline{C}]}{8} \right) (d + r^{k+2}(r^2 + 3r + 1)) \\ &\quad + \frac{\mathbb{P}[\overline{C}]}{8} r^{k+2}. \end{aligned} \quad (4.13)$$

Çift k için Stage-4 Analizi Lemma 9'e benzer olarak, şuna sahibiz

$$\begin{aligned} \mathbb{E}[T_i \mid R_i] &= \frac{1}{2} (d + r^{2i}(r + 1)) + \frac{1}{2} (r^{2i}(r^2 + 2r + 1)) \\ &= \frac{1}{2} (d + r^{2i}(r^2 + 3r + 2)). \end{aligned}$$

Bu roundda geçen zamanın daha önceki E_3, E_4 olaylarından bağımsız olduğunu not edelim.

Lemma 10'e benzer olarak, şuna sahibiz

$$\begin{aligned}
& \sum_{i=k/2+2}^{\infty} \mathbb{E}[T_i \mid R_i] \mathbb{P}[R_i] \\
&= \mathbb{P}[E_3 \vee E_4] \sum_{i=k/2+2}^{\infty} \frac{1}{2} (d + r^{2i}(r^2 + 3r + 2)) \left(\frac{1}{2}\right)^{i-k/2-1} \\
&= \left(\frac{1}{4} + \frac{\mathbb{P}[\overline{C}]}{8}\right) \left(d + \frac{r^{k+4}(r^2 + 3r + 2)}{2 - r^2}\right). \tag{4.14}
\end{aligned}$$

Çift k için Zaman Kompetitif Oranının Hesaplanması Denklem 4.10'den Denklem 4.14'e kadar olan denklemleri topluyoruz ve Bölüm 4.3.2'deki gibi analiz ediyoruz. $0 \leq \delta \leq 1/2$ olduğu zaman, en kötü seçim $\delta = 1/2$ 'dir ve $r = 1.195$ seçmek 24.63 olan zaman kompetitif oranı verir. $1/2 < \delta \leq 1$ olduğu zaman, en kötü seçim $\delta = 1$ 'dir ve $r = 1.195$ seçmek 24.843 zaman kompetitif oranı verir.

Tek k için Mesafe Hesaplamaları

Bu bölümde, tek k için mesafe analizi özet olarak yapıyoruz. Argümanlar çift mesafe hesaplamalarına benzerdir.

Tek k için tekrar dört aşamadaki analizi yapıyoruz. Bu aşamalar: $i \leq (k-3)/2$ (Stage-1), $i = (k-1)/2$ (Stage-2), $i = (k+1)/2$ (Stage-3) ve $i \geq (k+3)/2$ (Stage-4)'dir. Çift durumdan ana fark, kritik round $i = (k-1)/2$ 'deki davranışdır. Bu round için, robotlar sadece eğer birbirlerinden uzaklaşırlarsa $(\overline{A}_{(k-1)/2} \wedge \overline{B}_{(k-1)/2})$ ve C olayı olursa buluşabilirler. Daha fazlası, eğer robotlar bu roundda buluşamazlarsa, $i \geq (k+1)/2$ için dört olası başlangıç konfigürasyonu $(\pm f_{2i-1}, 2d \pm f_{2i-1})$ için $i \geq (k+1)/2$ vardır. Bu, eğer kritik round başarısız olursa robotlar için üç olası bitiş konfigürasyonuna sahip olan çift k analizindeki Stage-2'den farklıdır.

Tek k için Stage-1 Analizi Lemma 2'a benzer olarak, şuna sahibiz

$$\sum_{i=0}^{(k-3)/2} \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] < (r+2) \frac{r^{k-1}}{r^2 - 1} \mathbb{E}[r^\epsilon]. \tag{4.15}$$

Tek k için Stage-2 Analizi Robotlar sadece $F_1 = \overline{A}_{(k-1)/2} \wedge \overline{B}_{(k-1)/2} \wedge C$ olayı olduğunda buluşabilirler. Başarı olasılığı

$$\mathbb{P}[S_{(k-1)/2}] = \mathbb{P}[F_1] = \frac{\mathbb{P}[C]}{4}$$

'dir. Lemma 4'ya benzer olarak, şuna sahibiz

$$\begin{aligned} E[D_{(k-1)/2} \mid S_{(k-1)/2}] \Pr[S_{(k-1)/2}] \\ = (2r^{k-1}E[r^\epsilon \mid C] + d)\frac{\mathbb{P}[C]}{4} \end{aligned} \quad (4.16)$$

Başarısız $(k-1)/2$ roundu iki tipe ayırdık. $F_2 = (A_{(k-1)/2} \wedge B_{(k-1)/2}) \vee (\overline{A}_{(k-1)/2} \wedge B_{(k-1)/2}) \vee (A_{(k-1)/2} \wedge \overline{B}_{(k-1)/2})$ ve $F_3 = \overline{A}_{(k-1)/2} \wedge \overline{B}_{(k-1)/2} \wedge \overline{C}$ olsun. Şuna sahibiz, $\mathbb{P}[F_2] = 3/4$ and $\mathbb{P}[F_3] = \mathbb{P}[\overline{C}]/4$.

Lemma 5'a benzer olarak şuna sahibiz

$$\begin{aligned} \mathbb{E}[D_{(k-1)/2} \mid \overline{S}_{(k-1)/2}] \mathbb{P}[\overline{S}_{(k-1)/2}] \\ = \mathbb{E}[D_{(k-1)/2} \mid F_2] \mathbb{P}[F_2] + \mathbb{E}[D_{(k-1)/2} \mid F_3] \mathbb{P}[F_3] \\ = r^{k-1}(2+r) \left(\frac{3}{4}\mathbb{E}[r^\epsilon] + \frac{1}{4}\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[\overline{C}] \right). \end{aligned} \quad (4.17)$$

Tek k için Stage-3 Analizi Robotlar Stage-3'de sadece ve sadece

$$F_4 = (A_{(k+1)/2} \wedge B_{(k+1)/2}) \vee (\overline{A}_{(k+1)/2} \wedge \overline{B}_{(k+1)/2})$$

olursa buluşur. Bu nedenle şuna sahibiz, $\Pr[S_{(k+1)/2} \mid R_{(k+1)/2}] = 1/2 = \Pr[\overline{S}_{(k+1)/2} \mid R_{(k+1)/2}]$. Bir önceki bölümdeki olay notasyonunu kullanmaya devam ediyoruz. Stage-3'e F_2 veya F_3 olayından hemen sonra giriyoruz ve bu üç durumu ayrı olarak ele alıyoruz.

Lemma 6, 7 ve 8'e benzer olarak, şuna sahibiz

$$\mathbb{E}[D_{(k+1)/2} \mid F_2] \Pr[F_2] = \frac{3}{8}d + \frac{3}{8}r^{k+1}(3+r)\mathbb{E}[r^\epsilon] \quad (4.18)$$

ve

$$\begin{aligned} \mathbb{E}[D_{(k+1)/2} \mid F_3] \Pr[F_3] \\ = \frac{\mathbb{P}[\overline{C}]}{8}d + \frac{1}{8}r^{k+1}(3+r)\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[\overline{C}]. \end{aligned} \quad (4.19)$$

Tek k için Stage-4 Analizi Lemma 9'e benzer olarak, $i \geq (k+3)/2$ için şuna sahibiz

$$\mathbb{P}[R_i \mid F_2] = \mathbb{P}[R_i \mid F_3] = \left(\frac{1}{2}\right)^{i-(k-1)/2}.$$

Daha fazlası,

$$\begin{aligned}\mathbb{E}[D_i \mid R_i \wedge F_2] &= \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon]), \\ \mathbb{E}[D_i \mid R_i \wedge F_3] &= \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon \mid \overline{C}]).\end{aligned}$$

Lemma 10'e benzer olarak, şuna sahibiz

$$\begin{aligned}& \sum_{i=(k+3)/2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge F_2] \mathbb{P}[R_i \mid F_2] \mathbb{P}[F_2] \\ &= \sum_{i=(k+3)/2}^{\infty} \frac{1}{2} (d + r^{2i} (3 + r) \mathbb{E}[r^\epsilon]) \left(\frac{1}{2}\right)^{i-(k-1)/2} \left(\frac{3}{4}\right) \\ &= \frac{3}{8}d + \frac{3}{8} \cdot \frac{r^{k+3}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon].\end{aligned}\tag{4.20}$$

Benzer ekilde, şuna sahibiz

$$\begin{aligned}& \sum_{i=(k+3)/2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge F_3] \mathbb{P}[R_i \mid F_3] \mathbb{P}[F_3] \\ &= \frac{\mathbb{P}[\overline{C}]}{8}d + \frac{1}{8} \cdot \frac{r^{k+3}(3+r)}{2-r^2} \mathbb{E}[r^\epsilon \mid \overline{C}] \mathbb{P}[\overline{C}].\end{aligned}\tag{4.21}$$

Tek k için Mesafe Kompetitif Oranının Hesaplanması Denklem (4.15)'den Denklem(4.21)'e kadar topluyoruz. $0 \leq \delta \leq 1/2$ olduğu zaman, en kötü seçim $\delta = 0$ olduğu zamandır ve $r = 1.195$ seçmek 17.69 olan mesafe kompetitif oranı verir. $1/2 < \delta \leq 1$ olduğu zaman, en kötü seçim $\delta = 1/2$ olduğu zamandır ve $r = 1.195$ seçmek 17.35 mesafe kompetitif oranını verir. Sonuçta, k için mesafe kompetitif oran 17.686'dır. Bu çift durumun mesafe kompetitif oranı ile uyuşur.

Tek k için Zaman Hesaplamaları

Bu bölümde, tek k için zaman analizi özet olarak yapıyoruz. Argümanlar tek mesafe hesaplamalarına benzerdir.

Tek k için Stage-1 Analizi Lemma 2'e benzer olarak, şuna sahibiz

$$\sum_{i=0}^{(k-3)/2} \mathbb{E}[T_i \mid R_i] \mathbb{P}[R_i] < (r^2 + 2r + 1) \frac{r^{k-1}}{r^2 - 1}.\tag{4.22}$$

Tek k için Stage-2 Analizi Şuna sahibiz

$$\begin{aligned} & E[T_{(k-1)/2} \mid S_{(k-1)/2}] \Pr[S_{(k-1)/2}] \\ &= (2r^k + r^{k-1} + d) \frac{\mathbb{P}[C]}{4} \end{aligned} \quad (4.23)$$

ve

$$\begin{aligned} & \mathbb{E}[T_{(k-1)/2} \mid \overline{S}_{(k-1)/2}] \mathbb{P}[\overline{S}_{(k-1)/2}] \\ &= \mathbb{E}[T_{(k-1)/2} \mid F_2] \mathbb{P}[F_2] + \mathbb{E}[T_{(k-1)/2} \mid F_3] \mathbb{P}[F_3] \\ &= r^{k-1}(r^2 + 2r + 1) \left(\frac{3}{4} + \frac{\mathbb{P}[\overline{C}]}{4} \right). \end{aligned} \quad (4.24)$$

Tek k için Stage-3 Analizi Şuna sahibiz

$$\mathbb{E}[T_{(k+1)/2} \mid F_2] \Pr[F_2] = \frac{3}{8} (d + r^{k+1}(r^2 + 3r + 1)) \quad (4.25)$$

ve

$$\begin{aligned} & \mathbb{E}[D_{(k+1)/2} \mid F_3] \Pr[F_3] \\ &= \frac{\mathbb{P}[\overline{C}]}{8} (d + r^{k+1}(r^2 + 3r + 1)) \end{aligned} \quad (4.26)$$

Tek k için Stage-4 Analizi Şuna sahibiz

$$\begin{aligned} & \sum_{i=(k+3)/2}^{\infty} \mathbb{E}[T_i \mid R_i] \Pr[R_i] \\ &= \left(\frac{3}{8} + \frac{\mathbb{P}[\overline{C}]}{8} \right) \left(d + \frac{r^{k+3}(r^2 + 3r + 1)}{2 - r^2} \right). \end{aligned} \quad (4.27)$$

Tek k için Zaman Kompetitif Oranının Hesaplanması Denklem (4.22)'den (4.27)'e kadar topluyoruz ve Bölüm 4.3.2'deki şekilde analiz ediyoruz. $0 \leq \delta \leq 1/2$ olduğu zaman, en kötü seçim $\delta = 0$ olduğu zamandır ve $r = 1.195 \ 23.58$ olan zaman kompetitif oranının verir. Sonuçta, tek k için, zaman kompetitif oranı 23.58'dir.

4.3.4 $\mathcal{SR}$ Algoritmasının Beklenen Round Sayısı

$\mathcal{SR}$ Algoritması randevu gerçekleşene kadar devam eder. Lemma 11'de, randevu gerçekleşene kadar çalıştırılmasını gereken round sayısının beklenen değerine alt ve üst sınır buluyoruz.

Lemma 11. *Algoritma $\mathcal{SR}$ 'daki beklenen round sayısı $k/2 - 1$ ile alt sınırlı ve $k/2 + 3$ ile üst sınırlıdır.*

Kanıt. $\mathcal{SR}$ 'ın analizinden round $i \leq k/2 - 1$ iken, robotların birleşik yer değişimleri halen $r^{2i+\epsilon_1} + r^{2i+\epsilon_2} \leq 2d$ olduğu için buluşamadıklarını ve ilk olarak round $i = k/2$ iken buluşma olasılıklarının olduğunu biliyoruz. Bu nedenle, randevunun gerçekleşmesi için en az $k/2 - 1$ round geçmesi gerekir.

Beklenen round sayısına üst sınır bulmak için robotların Stage-2 ve Stage-3'de de buluşamadıklarını varsayalım. Bu durumda $k/2 + 1$ round yapılacaktır. $k/2 + 1$ round ve Stage-4'deki beklenen round sayısı bu lemmadaki üst sınırı verir. Şimdi Stage-4'deki beklenen round sayısını hesaplıyoruz.

$rnds$ randevunun gerçekleşmesi için Stage-4'da çalıştırılan round sayısı olsun. Stage-4'da robotların her zaman $1/2$ olasılıkla buluştuklarını hatırlayalım. Bir başka deyişle, $\mathbb{P}[S_{i \geq k/2+2}] = \mathbb{P}[\overline{S}_{i \geq k/2+2}] = 1/2$ 'dir. $\mathbb{E}[rnds]$ 'i aşağıdaki gibi hesaplıyoruz. Bu, $1/2$ parametrelili geometrik dağılımın beklenen değeridir.

$$\begin{aligned} \mathbb{E}[rnds] &= \sum_{i=k/2+2}^{\infty} i \mathbb{P}[S_i] \mathbb{P}[\overline{S}_i]^{i-1} \\ &\leq \sum_{i=0}^{\infty} i \left(\frac{1}{2}\right)^i \\ &\leq 2 \end{aligned}$$

Bu nedenle, $\mathcal{SR}$ 'daki beklenen round sayısına üst sınırı $(k/2 + 1) + 2 = k/2 + 3$ 'dir. $\square$

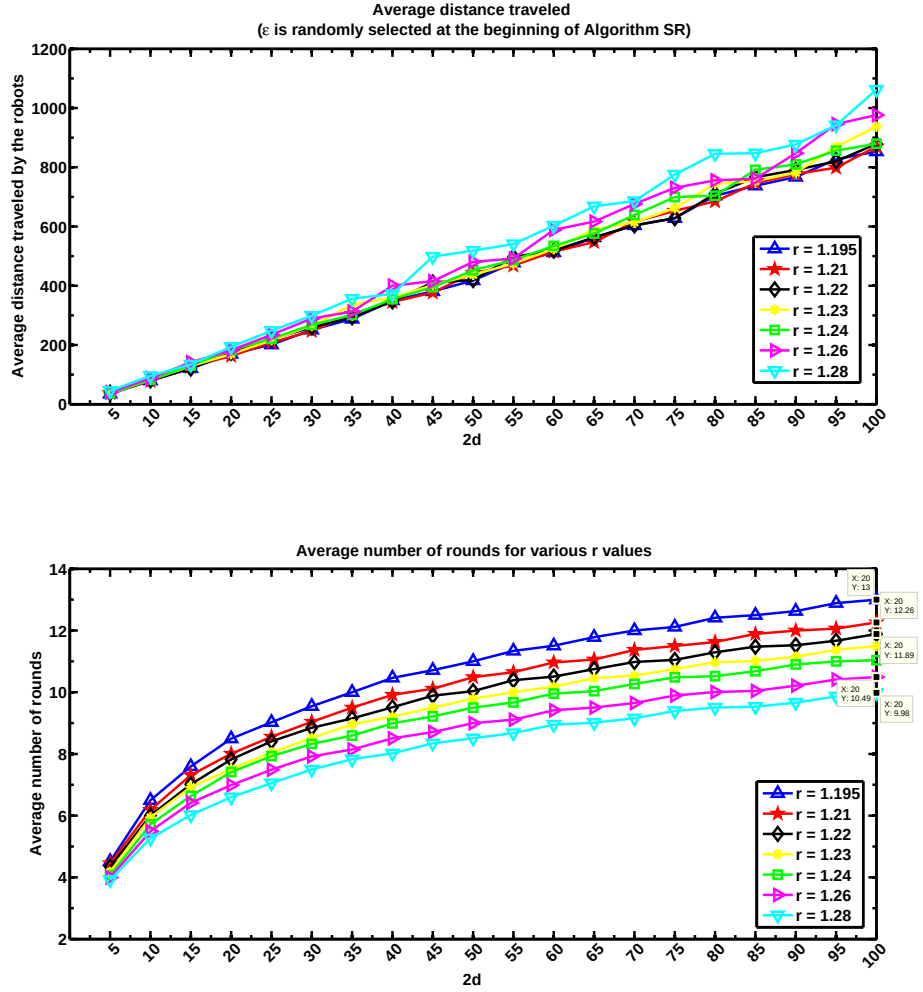
Algorithm 2 $\mathcal{SR}\text{-main}()$

```

1:  $i \leftarrow 0$  { current round}
2: while true do
3:   getNextPosition(i) { executes the lines btw 7-23 in  $\mathcal{SR}$ .}
4:    $i \leftarrow i + 1$  { next round}
5: end while
```

Lemma 12. *Algoritma $\mathcal{SR}\text{-main}$ için beklenen tekrar sayısı $O(\log d)$ 'dir.*

Kanıt. Lemma 11'de, randevunun gerçekleşmesi için gereken beklenen round sayısının en fazla $k/2 + 3$ olduğunu buluyoruz. İki robot arasındaki başlangıç uzaklığı $d = r^{k+\delta}$ 'dir ve $r = 1.195$ olarak optimize edilmiştir. Beklenen tekrar sayısını (round sayısı olarak) d 'e bağlı bulmak için, k yerine $\log_{1.195} d - \delta$ koyuyoruz. $\mathcal{SR}\text{-main}()$ Algoritmasındaki $\text{getNextPosition}(i)$ fonksiyonu $\mathcal{SR}$

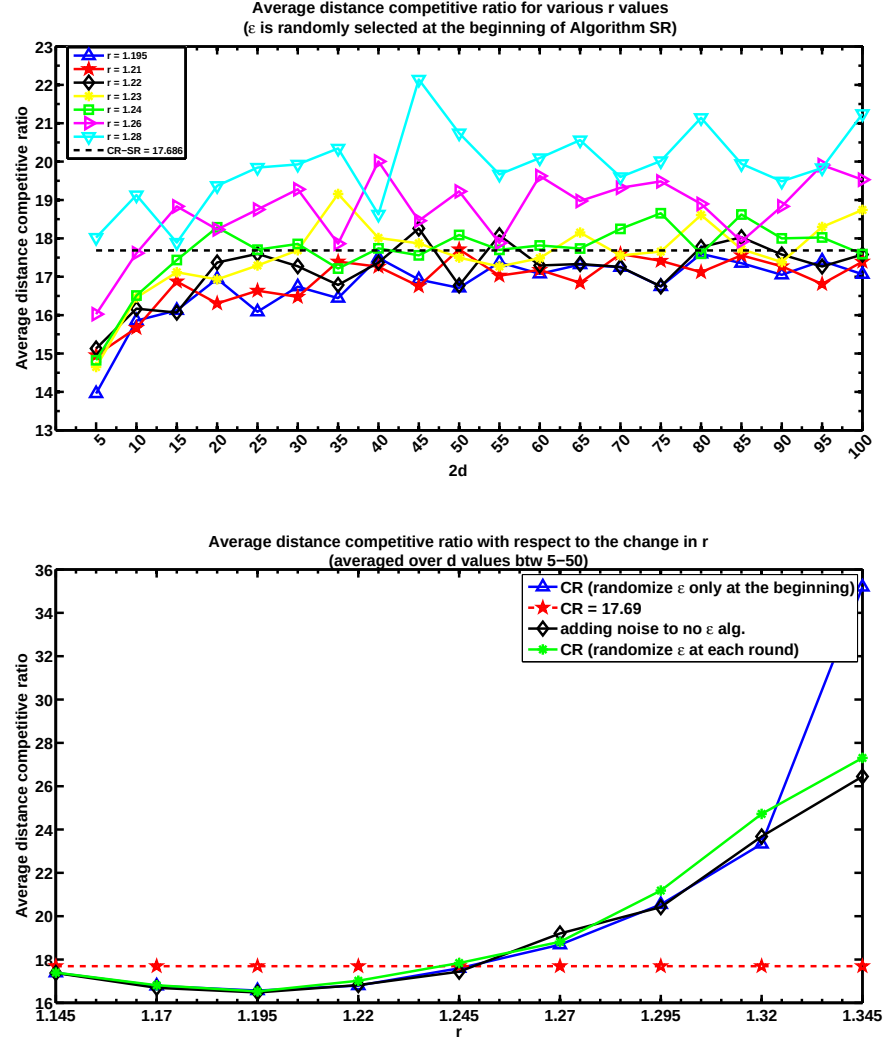


Şekil 4.3: Çeşitli r değerleri için, başlangıç uzaklığındaki değişikliğe göre Algoritma $\mathcal{SR}$ 'in 100,000 simülasyondaki ortalama performansı. Burada, ϵ robot tarafından aramanın başında rasgele olarak seçilmiştir. **ÜST**: Ortalama giden mesafe. **ALT**: Ortalama round sayısı.

Algoritmasındaki robotun gideceği bir sonraki pozisyonu belirler ve bu fonksiyonun her çağırımı sabit zaman alır. Bu nedenle, $\mathcal{SR}$ -main'in beklenen tekrar sayısı $O(\log d)$ ile sınırlıdır. $\square$

4.3.5 Simülasyonlar

Bu bölümde $\mathcal{SR}$ 'in simülasyon sonuçlarının, optimal r seçimi ve Teorem 1'de elde ettiğimiz kompetitif oran ile uyumlu olduğunu gösteriyoruz. Daha fazlası, ϵ 'u her roundda rasgeleleştirmenin uzaklıkla orantılı hatalara benzer olduğunu gösteriyoruz. Bu, algoritmamızın navigasyonel hatalara dayanıklı olduğu gösterir.



Şekil 4.4: Algoritma $\mathcal{SR}$ 'in simülasyonlardaki performansı. **ÜST:** Çeşitli r değerleri için, d 'deki değişime göre ortalama gidilen mesafe için kompetitif oran. **ALT:** Gaussian noise'in Algoritmamızın kompetitif oranı üzerindeki etkisi. Mesafe kompetitif oranının, x -axis üzerindeki her bir r değeri için, birçok başlangıç uzaklığı değerlerinden ortalaması alınır.

```

2d = 50.0 r = 1.28
epsilon1: 0.25 epsilon1:0.25
Robot-1=> rnd: 0 toss: 1 distsofar: 2.6609704338453044
Robot-2=> rnd: 0 toss: 1 distsofar: 2.659571687469562
Robot-1=> rnd: 1 toss: 1 distsofar: 9.746901215326375
Robot-2=> rnd: 1 toss: 0 distsofar: 7.017013940219693
Robot-1=> rnd: 2 toss: 0 distsofar: 16.889889333444195
Robot-2=> rnd: 2 toss: 1 distsofar: 14.156247327125508
Robot-1=> rnd: 3 toss: 1 distsofar: 28.59296106616843
Robot-2=> rnd: 3 toss: 1 distsofar: 33.16739941247567
Robot-1=> rnd: 4 toss: 1 distsofar: 59.7572142175857
Robot-2=> rnd: 4 toss: 1 distsofar: 64.31527098911337
Robot-1=> rnd: 5 toss: 1 distsofar: 110.81672658086775
Robot-2=> rnd: 5 toss: 0 distsofar: 95.71395147835169
Robot-1=> rnd: 6 toss: 0 distsofar: 162.28738038417544
Robot-2=> rnd: 6 toss: 1 distsofar: 147.15754959191975
Robot-1=> rnd: 7 toss: 1 distsofar: 246.61689957551476
Robot-2=> rnd: 7 toss: 1 distsofar: 284.14733750786667
Robot-1=> rnd: 8 toss: 1 distsofar: 471.17900978351906
Robot-2=> rnd: 8 toss: 1 distsofar: 508.5914060293542
Robot-1=> rnd: 9 toss: 1 distsofar: 748.1419827095477
Robot-2=> rnd: 9 toss: 0 distsofar: 643.9441092669549
Avg. Dist: 696.043 Avg. # of rounds: 10.0

```

```

2d = 50.0 r = 1.195
epsilon1: 0.68 epsilon1:0.09
Robot-1=> rnd: 0 toss: 0 distsofar: 2.6632980372004234
Robot-2=> rnd: 0 toss: 0 distsofar: 2.397414549949209
Robot-1=> rnd: 1 toss: 1 distsofar: 6.466554216773559
Robot-2=> rnd: 1 toss: 0 distsofar: 8.250746508517397
Robot-1=> rnd: 2 toss: 1 distsofar: 15.752274511992841
Robot-2=> rnd: 2 toss: 1 distsofar: 13.139687077038278
Robot-1=> rnd: 3 toss: 1 distsofar: 29.012515236573357
Robot-2=> rnd: 3 toss: 0 distsofar: 20.12121643240031
Robot-1=> rnd: 4 toss: 1 distsofar: 47.94847049729245
Robot-2=> rnd: 4 toss: 1 distsofar: 30.091014890091174
Robot-1=> rnd: 5 toss: 0 distsofar: 63.764549361965564
Robot-2=> rnd: 5 toss: 1 distsofar: 54.432462193863515
Robot-1=> rnd: 6 toss: 1 distsofar: 86.35030538269038
Robot-2=> rnd: 6 toss: 1 distsofar: 89.19265747983302
Robot-1=> rnd: 7 toss: 1 distsofar: 141.49384638077652
Robot-2=> rnd: 7 toss: 1 distsofar: 138.83108535307963
Robot-1=> rnd: 8 toss: 0 distsofar: 187.55197132323823
Robot-2=> rnd: 8 toss: 0 distsofar: 180.29111623888573
Robot-1=> rnd: 9 toss: 1 distsofar: 253.32412519419714
Robot-2=> rnd: 9 toss: 1 distsofar: 239.497076844589
Robot-1=> rnd: 10 toss: 1 distsofar: 413.90805518031664
Robot-2=> rnd: 10 toss: 1 distsofar: 384.0495245463993
Robot-1=> rnd: 11 toss: 1 distsofar: 600.2578962119576
Robot-2=> rnd: 11 toss: 0 distsofar: 468.6029863258286
Avg. Dist: 534.430 Avg. # of rounds: 12.0

```

Şekil 4.5: Daha az ortalama round sayısı daha az mesafe gidileceğini ima etmez.

İlk önce SR 'in performansını, r 'nin ve başlangıç uzaklığının bir fonksiyonu olarak araştırıyoruz. Şekil 4.3'deki plotlar, herbirisi 100,000 denemenin ortalaması alınarak hesaplanan, değişik r değerleri ve başlangıç uzaklıkları için, simülasyonların sonuçlarını rapor eder. Robotların arasındaki başlangıç uzaklığı $2d$ 'dir. Biz stratejimizi, 5 ve 100 adım büyüklüğü (*Ing. step size*) arasında değişen tamsayı $2d$ değerleri için simüle ediyoruz ve robot bir birim adım büyüklüğüne sahiptir. r 'in değeri, 1.195 ve 1.28 arasında değişir.

Üstteki plot, başlangıç uzaklığının bir fonksiyonu olarak, bu r değerleri için ortalama gidilen mesafedeki değişimi gösterir. Bu sonuçları şu şekilde hesaplıyoruz: Önce her denemedeki robotlar tarafından gidilen ortalama mesafeyi buluyoruz. Daha sonra, her denemedeki bu ortalamaları birbirine ekliyoruz ve çıkan toplamın ortalamasını alıyoruz. Bu şekil, uzaklık arttıkça robotlar tarafından gidilen mesafenin arttığını gösterir. Robot tarafından gidilen mesafe robotun genişleme yarıçapı olan r arttıkça artmaktadır. r değerleri birbirine yakın olduğundan, şekilden ortalama gidilen mesafe değerlerinin de birbirine yakın olduğunu görüyoruz.

Alttaki plot, başlangıç uzaklığının bir fonksiyonu olarak, aynı r kümesi için ortalama round sayısındaki değişimi gösterir. Beklendiği gibi, simülasyon sonuçları, başlangıç uzaklığı arttıkça ve r 'in değeri düştükçe, ortalama round sayısının arttığını gösterir. Her $(2d, r)$ kombinasyonu için, ortalama round sayısı $\approx k/2 + 2$ 'dir ki bu da SR 'in beklenen round sayısının üst sınırı olan $k/2 + 3$ 'ü doğrular. Ortalama round sayısı, $r = 1.195$ iken en yüksek ve $r = 1.28$ iken en düşüktür.

Şekil 4.3'deki üstteki ortalama mesafeyi gösteren plot ve alttaki ortalama round sayısını gösteren plot arasında uyumsuzluk vardır. Üstteki plotta r azaldıkça ortalama gidilen mesafe azalırken, alttaki plotta r değeri azaldıkça ortalama round sayısı artmaktadır. Daha az sayıda ortalama round sayısına sahip olan bir r değeri, aynı değer için daha az mesafe gidileceğini ima etmez. Şekil 4.5'de bu duruma bir örnek gösteriyoruz. Bu şekildeki sonuçlar, robotların Algoritma SR 'i çalıştırmasının sonucunda elde edilmiştir. Burada, rnd şuanki round numarasını, $distsofar$ şuana kada geline roundda robot tarafından gidilen mesafeyi ve $toss$ robotun para atışından gelen sonucu göstermektedir. Robotlar arasındaki başlangıç uzaklığı $2d = 50$ 'dir. Sonuçlar, $r = 1.28$ için ortalama round sayısının 10, $r = 1.195$ için ortalama round sayısının 12 olduğunu göstermektedir. Fakat, görüldüğü gibi $r = 1.28$ iken ortalama round sayısı daha az round olmasına rağmen, ortalama gidilen mesafe

değeri (696.043), $r = 1.195$ 'in ortalama gidilen mesafe değerinden (534.430) daha büyüktür.

Şekil 4.4'deki üstteki plot, r değerlerinin aynı kümesi için, başlangıç uzaklığının bir fonksiyonu olarak ortalama kompetitif oranı gösterir. Yatay çizgili doğru, $r = 1.195$ 'in optimal değeri için, 17.696 olan teorik optimal mesafe kompetitif oranı gösterir. Siyah koyu doğru, $r = 1.195$ için $\mathcal{SR}$ 'in simülasyonlardaki performansının gösterir. Robotlar aramanın başında ϵ değerlerini rasgele olarak seçmişlerdir. Beklenildiği gibi, optimal $r = 1.195$ değeri diğer r değerlerine göre daha üstün performansa sahiptir. Her başlangıç uzaklığı için ortalama kompetitif oran $r = 1.195$ için 17.686'nın altındadır.

Daha sonra, $\mathcal{SR}$ algoritmasının, her roundda $\epsilon \in (0, 1]$ 'un yeni bir değerinin üretildiği bir varyantını düşündük. Rasgeleliğin etkisi navigasyonel hatalar olarak yorumlanabilir, örneğin odometri hatalarından kaynaklananlar. Bu varyantın performansı algoritmamızın bu tip hatalar altında dayanıklı olduğunu gösterir. Öncelikle, bu hataları $\epsilon \in (0, 1]$ 'u her roundda her robot uniform olarak rasgele seçerek simüle ediyoruz. Teorem1'dekine benzer olan analiz, bu varyantın aynı beklenen kompetitif orana sahip olduğunu gösterir. Şekil 4.4'de bulunan simülasyonlardaki sonuçlar bu iddia ile uyumludur. Üstteki plot, başlangıç uzaklığının bir fonksiyonu olarak çizilen çeşitli r değerleri için, ortalama mesafe kompetitif oranını göstermektedir. Sonuçlar Şekil 4.3'dekiler ile karşılaştırılabilir. Simülasyon sonuçları ϵ 'u her roundda rasgele olarak seçmenin $\mathcal{SR}$ 'in performansını etkilemeyeceğini göstermektedir.

Literatürde, navigasyonel hatalar nedeniyle meydana gelen gürültü, çoğunlukla standart sapması gidilen mesafeye orantılı olan Gaussian olarak modellenir Rekleitis (2004). Şekil 4.4'deki alttaki plotda, $\mathcal{SR}$ 'ı bu hataların olduğu durumda çalıştırıyoruz. Bu simülasyonlarda, hataların sadece x-axis'inde olduğunu varsayıyoruz. Rekleitis (2004)'deki Tablo 1'den $\mu = -1.843$ ve $\sigma = 0.372$ olarak kullanıyoruz. Özellikle, $f_i = r^{i+\epsilon}$ olarak almak yerine, $f_i = r^i + n(\mu, \sigma)$ olarak kuruyoruz. Burada n gaussian noise'dur. Bu değerler 0.2 m/sec hızında hareket eden bir robot içindir.

Yatay çizgili doğru, optimal $r = 1.195$. değeri için, $\mathcal{SR}$ 'ın 17.696 olan mesafe kompetitif oranını gösterir. r yarıçapı için kompetitif oran, tüm simüle edilen d değerlerinden gözlenen kompetitif oranın ortalamasıdır. Başka bir deyişle, her bir $5 \leq d \leq 50$ tamsayı uzaklığı için 10,000 simülasyon yaptık ve

daha sonra onların gözlenen kompetitif oranlarının ortalamasını aldık. Mavi koyu doğru, r 'deki değişime göre, $\mathcal{SR}$ 'in kompetitif oranındaki değişmeyi gösterir. Kırmızı koyu doğru, $\mathcal{SR}$ Gaussian noise'a eklendiği zaman, kompetitif orandaki değişimi gösterir. Simülasyon sonuçları algoritmamızın navigasyonel hatalara dayanıklı olduğunu gösterir. Gerçekte, x -axis'de gözüken r değerlerinin çoğu için, Gaussian noise algoritmamızın performansını etkilemez. Bu f_i 'a rasgele değişken ϵ 'u eklemenin $\mathcal{SR}$ 'i gürültü olduğu durumda çalıştırmakla neredeyse aynı olduğu anlamına gelir.

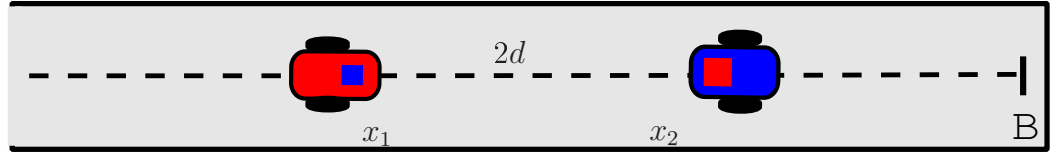
4.3.6 Uzantılar

Bu bölümde, pratik senaryolarda da karşılaşılabileceğimiz, algoritmamızın iki uzantısını sunuyoruz. Birinci uzantıda, geniş iç ortamlarda karşılaşılabilecek, doğru parçalarını (*Ing. line segments*) ve yarı doğruları (*Ing. half lines*) sunuyoruz. Bu gibi ortamlarda, robot hareketleri sınır noktaları tarafından sınırlandırılmıştır. Algoritmamızı bu sınırlardan faydalanmak için şu şekilde değiştiriyoruz. Eğer $\mathcal{SR}$ 'nin çalışması sırasında, robot sınıra çarparsa, diğer robotla karşılaşana kadar zıt yöne doğru hareket eder. Açık bir şekilde, bir robot sınıra çarptığı zaman, diğer robotun diğer tarafta olduğunu bilir.

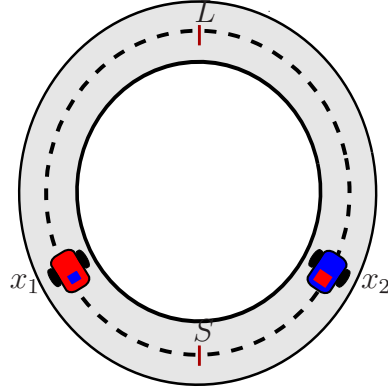
Lemma 13. *Doğru parçasının kompetitif oranı, $\mathcal{SR}$ 'in kompetitif oranını aşmaz.*

İspat. Sınırlandırılmış doğru parçası durumuna bir örnek (*Ing. instance*) verilerek, bu örneği sınırlandırılmamış randevu örneği ile eşleştiriyoruz. Her iki örnekte, robotları $2d$ uzaklıkta başlatıyoruz. Her örnek için robotlar aynı rasgele ϵ 'u paylaşırlar ve çalışma sırasında aynı para atışlarını kullanırlar. Sınırlandırılmamış versiyondaki robotlar, sınırlandırılmış versiyondaki robotlardan daha önce buluşamazlar. Aslında, eğer sınırlandırılmış robotlar asla bir bitiş noktası ile karşılaşmazlarsa, o zaman eşleştirilen örnekler aynı zamanda randevuya ulaşır. Eğer sınırlandırılmamış bir robot bir bitiş noktası ile karşılaşır, hemen yönünü değiştirir ve randevu olana kadar da durmaz. Bu durumda, sınırlandırılmış robotlar, sınırlandırılmamış robotlardan daha önce buluşur.

Daha sonra, Şekil 4.7'de gösterilen dairesel koridorlar gibi, basit kapalı yol (*Ing. simple closed path*) ortamlarını düşündük. Lineer durumda, robotların buluşması için sadece bir yol vardır: soldaki robot sağa doğru hareket etmelidir ve sağdaki robot sola doğru hareket etmelidir. Dairesel bir ortamda,



Şekil 4.6: $\mathcal{SR}$ 'ın birinci uzantısı: Algoritma $\mathcal{SR}$, doğru parçası ve yarı doğru gibi sınırlandırılmış ortamlara genişletilebilir. (Lemma 13) .



Şekil 4.7: $\mathcal{SR}$ 'ın ikinci uzantısı: Algoritma $\mathcal{SR}$, ayrıca kapalı yola benzer ortamlara da genişletilebilir (Lemma 14).

sonuç olarak, robotların buluşması için iki tane yol vardır: aynı şekilde hareket etmedikleri sürece, bir taraftan veya diğer taraftan birbirlerine yaklaşır- lar. Dairesel yol (*Ing.* circular path) ortamlarında, aynı $\mathcal{SR}$ algoritmayı takip ederiz.

Lemma 14. $\mathcal{SR}$ 'ın kompetitif oranı dairesel yol uzantısı için üst sınırdır.

İspat. Birkez daha, dairesel bir ortamın verilen bir örneğini, sınırlandırılma- mış lineer bir ortamın bir örneği ile eşleştiriyoruz. $2d$ dairesel ortam üzerinde robotlar arasındaki en kısa uzaklık olsun. Lineer bir ortamda, robotları baş- langıçta $2d$ uzaklığa yerleştirelim. Eş robotlar aynı rasgele ϵ 'u ve aynı para atışlarını kullanırlar. Bir önceki lemmaya benzer olarak, sınırlandırılmamış robotlar, dairesel ortamdaki robotların buluşmasından daha önce buluşamaz- lar. Daha sonraki çift, zig-zag hareketleri sırasındaki robotlar arasındaki daha uzun olan uzaklığı tamamen dolaşarak, önceki çiftten daha önce buluşabilir. Fakat, bu sadece randevu zamanını azaltır.

4.3.7 Deneyler

Bu bölümde, iki robotun gerçek yerleştiriminden elde edilen deneysel sonuç- ları sunuyoruz. Bu deneyler simetrik randevu stratejisinin ve uzantılarının



Şekil 4.8: **SOL:** Üzerlerinde Asus EEE PC laptop olan, düz bir koridor üzerine yerleştirilmiş olan iki iRobot Create robotunu gösteren birinci ve ikinci deneyin kurulumu. **SAĞ:** Üzerlerinde Asus EEE PC laptop olan, dikdörtgen şeklinde bir koridor üzerine yerleştirilmiş olan iki iRobot Create robotunu gösteren üçüncü grup için olan deney kurulumu.

(doğru parçası ve dairesel yol) geçerliliğini onaylar ve gerçek dünya ortamlarında uygulanabilirliğini gösterir.

Deney Kurulumu

Deneyler için, iRobot Create (Şekil 4.8'i görün) robotlarını robotik platformumuz olarak kullandık. iRobot Create wheel drop sensörleri, bumper sensörleri, cliff sensörleri ve Omnidirectional IR Receiver gibi dış sensörlere sahiptir. Deneylerimizde, sadece robotun sınır duvarına çarptığında ters yöne gitmesi için sadece bumper sensörlerini kullandık.

Engeller olmayan iç ortamlar içinde üç grupta deney yaptık. Bir grupta içindeki her bir deneyde robotlar arasındaki başlangıç uzaklığını değiştirdik. Deney grupları şunlardır: düz doğru deneyleri (*Ing.* straight line experiments (SLE)), sınır olan düz doğru deneyleri (*Ing.* straight line experiments with boundary (SLEB)) ve kapalı eğri deneyleri (*Ing.* closed curve experiments (CCE)). SLE ve SLEB'de, iki robot düz bir koridor üzerine yerleştirilir. CCE'de, iki robot dikdörtgen şekilli bir koridor üzerine yerleştirilir. Her robotun hızı 0.2 m/s'dir ve r 1.195 metredir. Robotlar $\mathcal{SR}$ algoritmasını veya ona uygun varyantını senkron olarak çalıştırır. Eğer birbirlerine çarparsa, deney sonlanır ve randevu gerçekleşmiştir diye sonuçlandırılır. Robotların

doğrudan çok az sapmasından kaynaklanan hataları göz ardı ediyoruz. Tüm deneylerde, robotlar başarıyla buluşmuşlardır. Deney şekillerindeki dikey doğrular, yeni bir roundun başlangıç noktalarını gösterir.

DeneySEL Sonuçlar

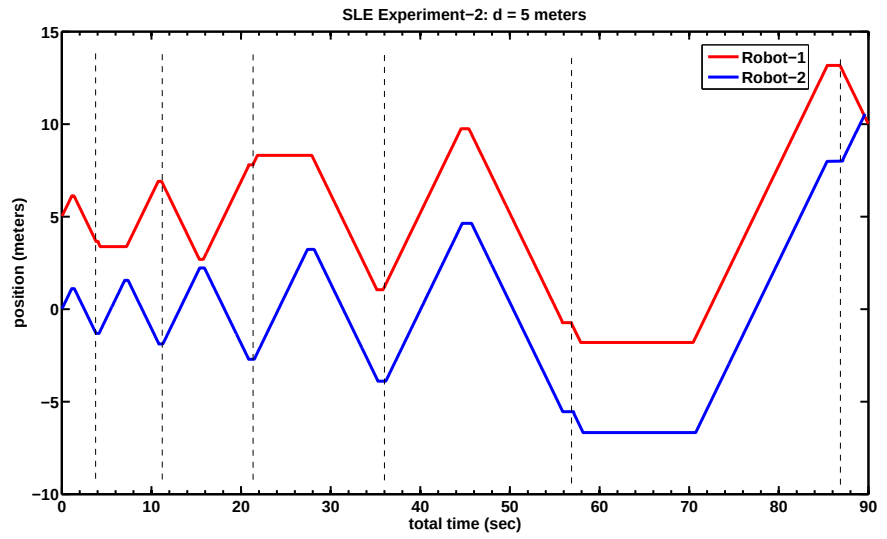
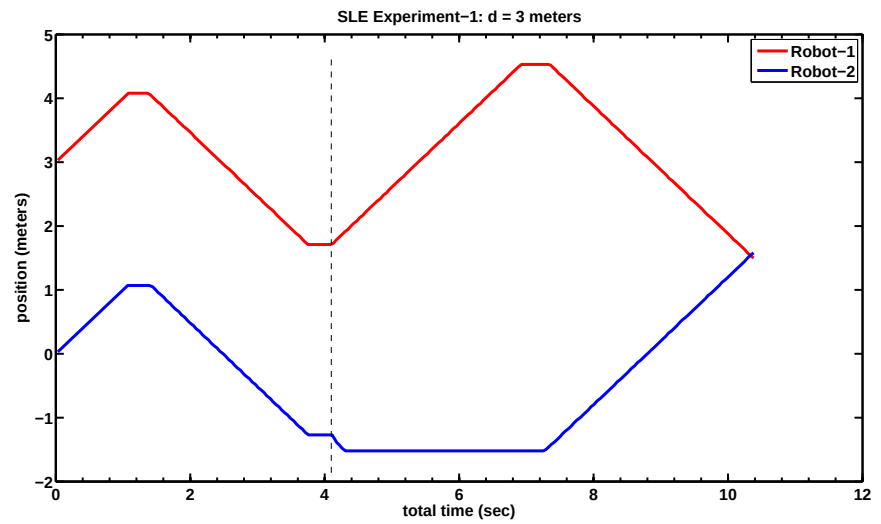
Şekil 4.9, 4.10 ve 4.12’de, robotların izledikleri yolları (*Ing.* trajectory) sunuyoruz. Şekil 4.9 ve 4.10’deki deneyler düz koridor üzerinde yapılır ve $\mathcal{SR}$ ’ın ve birinci uzantının uygulanabilirliğini göstermeyi amaçlar. Bu şekillerdeki x -ekseni saniye cinsinden zamanı gösterir ve y -ekseni robotların doğru üzerindeki x -koordinatını metre cinsinden gösterir. Şekil 4.12, $\mathcal{SR}$ ’ın ikinci uzantısının uygulanabilirliğini göstermek amacıyla dikdörtgen şekilli bir koridor üzerinde yapılan deneylerin sonuçlarını gösteriyoruz. Bu şekilde, x -ekseni ve y -ekseni, robotların x ve y -koordinatlarını gösterirken, z -ekseni saniye cinsinden zamanı gösterir. Robotların izledikleri yol kesiştiği zaman, robotlar doğru üzerinde aynı pozisyonadadır ve randevu gerçekleşmiştir.

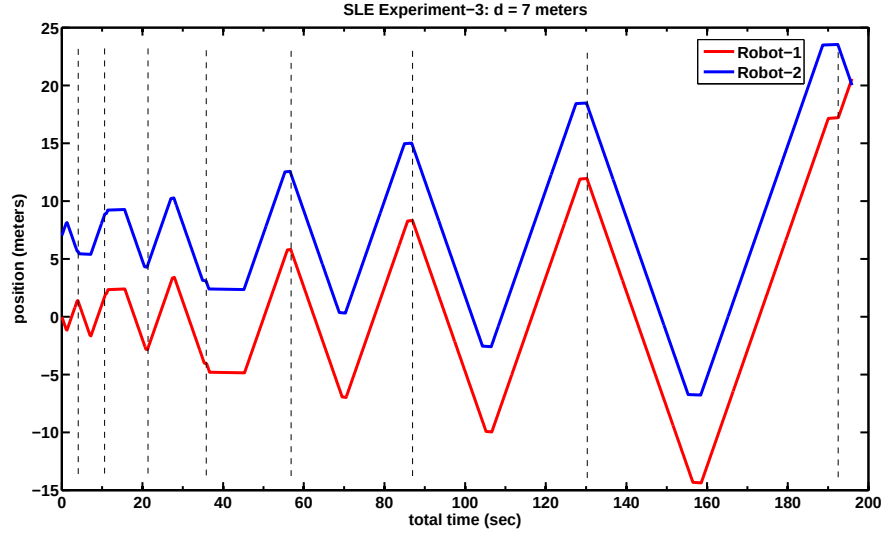
Şekil 4.9, SLE’deki deneyleri sunar. Bu deneylerde, robotlar Algoritma 1’de sunulan stratejiyi çalıştırır. Robotlar başlangıçta aralarındaki uzaklık üç, beş ve yedi metre olacak şekilde yerleştirilir. Bu şeklin en üstteki plotunda, robotlar birbirlerinden $2d = 3$ metre uzakta olacak şekilde yerleştirilir. Bu uzaklık $d = r^{k+\delta}$ için çözüldüğünde, $k = 2$ ve $\delta = 0.2239$ olur. Randevu iki roundda gerçekleşir ve 10.3 saniye sürer. Her robot tarafından rasgele seçilen ϵ değerlerinin sonucu $\epsilon_1 = 0.22$ ve $\epsilon_2 = 0.33$ ’dir. Birinci roundda, Robot-1 tura atar ve $r^{2*0+\epsilon_1} = 1.06$ metre sağa hareket eder. Diğer robotla senkron hareket etmek için, robotun, diğer robotun hareketini bitirecek kadar uzun beklediğini hatırlayalım. Bu nedenle bu deneide, robot birinci aşamanın sonunda, $r^{2*0+1} - 1.06 = 0.14$ saniye bekler. İkinci aşamada, robot $r^{2*0+\epsilon_1} + r^{2*0+1+\epsilon_1} = 1.06 + 1.27 = 2.33$ sola hareket eder ve ikinci aşamanın sonunda $r^{2*0+1} + r^{2*0+1+1} - 2.33 = 0.31$ saniye bekler. Bu arada, R_2 de tura atar ve $r^{2*0+\epsilon_2} = 1.04$ metre sağa hareket eder ve daha sonra da birinci aşamanın sonunda $r^{2*0+1} - 1.04 = 0.16$ saniye bekler. İkinci aşamada, robot $r^{2*0+\epsilon_1} + r^{2*0+1+\epsilon_1} = 1.04 + 1.24 = 2.28$ sola hareket eder ve ikinci aşamanın sonunda $r^{2*0+1} + r^{2*0+1+1} - 2.28 = 0.36$ saniye bekler.

Şekil 4.10 ve 4.11, SLEB ve SLEB-2b’deki deneyleri sunar. Hem SLEB hem de SLEB-2b için, robotları başlangıç uzaklıkları beş ve yedi metre olacak şekilde yerleştirdik. SLE’ye zıt olarak, SLEB’de, sadece bir veya iki robota

yakın olan bir sınır duvarının olduğu durumu düşündük. SLE’de olduğu gibi, robotlar Algoritma 1’in sınırlandırılmış parça olan varyantını çalıştırır. Sınır duvarını tespit etmek için, robot ön tarafında bulunan bumper sensörlerini kullanır. Sensör her zaman hareket edilen yön ile yüzleşmek zorunda olduğundan, robotun hareket ettiği yöne doğru ön yüzünü çevirmesini sağladık. SLEB’de ya Robot-1’in ya da Robot-2’nin yakınında sınır duvarı vardır. Şekil 4.10’deki üstteki plotta, robotlar arasındaki uzaklık beş metredir ve sınır duvarı Robot-1’den 2m uzağa yerleştirilmiştir. Buna rağmen, Şekil 4.10’deki alt plotta, robotlar arasındaki başlangıç uzaklığı yedi metredir ve sınır duvarı Robot-2’den iki metre uzağa yerleştirilmiştir. Şekil 4.11’deki plotlar, sınır duvarları hem Robot-1’in hem de Robot-2’nin yakınında olduğu zamanki sonuçları gösterir. Her iki deneyde de, sınır duvarları Robot-1 ve Robot-2’den yaklaşık olarak 2m uzaktadır. Şekil 4.11’nin son plotunda, Robot-2 ikinci roundda yazı atar ve sola doğru hareket etmeye başlar. Birinci aşamada, başlangıç konumundan itibaren yerdeğişimi iki metre olduğu zaman, sınır duvarına çarpar ve sağa doğru hareket etmeye başlar. Robot-1’e pozisyon 6.766’da çarpma kadar sağa doğru hareket etmeye devam eder. Robotların buluşması 18.2 saniye sürer.

Şekil 4.12 CCE’i gösterir. Doğru üzerinde yapılan deneylerden farklı olarak, CCE’de, robotların hem x hem de y koordinatları değişir. Bu gruptaki deneylerin amacı, simetrik randevu algoritmasının, dikdörtgen gibi dairesel yola benzer ortamlara uygulanabilir olmasıdır. Bu deneylerde, robotlar dikdörtgen şeklindeki bir koridora yerleştirilir. Dikdörtgenin uzunluğu ≈ 27.5 ve genişliği ≈ 18.5 metredir. Şekil 4.8’deki alt plot bu deneylerin kurulumuna bir örnektir. Şekil 4.12’deki en üst plotta, hem Robot-1 hem de Robot-2, köşeden 3m uzakta başlar. Ortadaki plotta, hem Robot-1 hem de Robot-2 köşeden 5m uzakta başlar. En alttaki plottada Robot-1 köşeden 3m uzakta başlarken, Robot-2 köşeden 7m uzakta başlar. Her robot Algoritma 1’i çalıştırır, dikdörtgenin köşesine ulaştıklarında, koridorun diğer tarafına dönüp, hareketine devam etmek zorunda olduklarının farkındadırlar. Şekil 4.12’deki en üst plotta, robotlar i köşesinde buluşurken, diğer iki plotta, robotlar koridorlardan birinde buluşurlar.





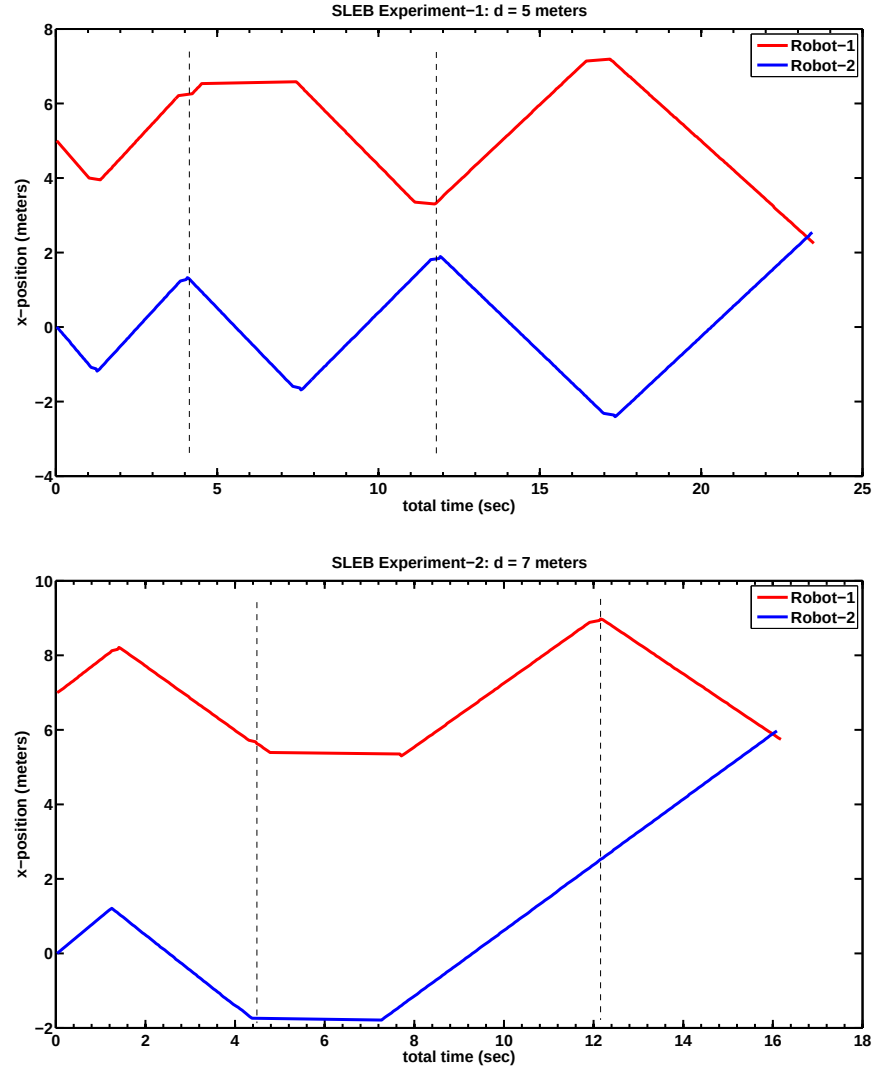
Şekil 4.9: **Düz doğru deneyleri.** Deneyler düz bir koridor üzerinde yapılır. Robotlar Algoritma 1’i çalıştırır. Buluşma robotların izledikleri yollar kesiştiği zaman gerçekleşir. Üst plottan alt plota doğru sırasıyla robotların arasındaki başlangıç uzaklığı üç, beş ve yedi metredir.

4.4. Asenkron Durum

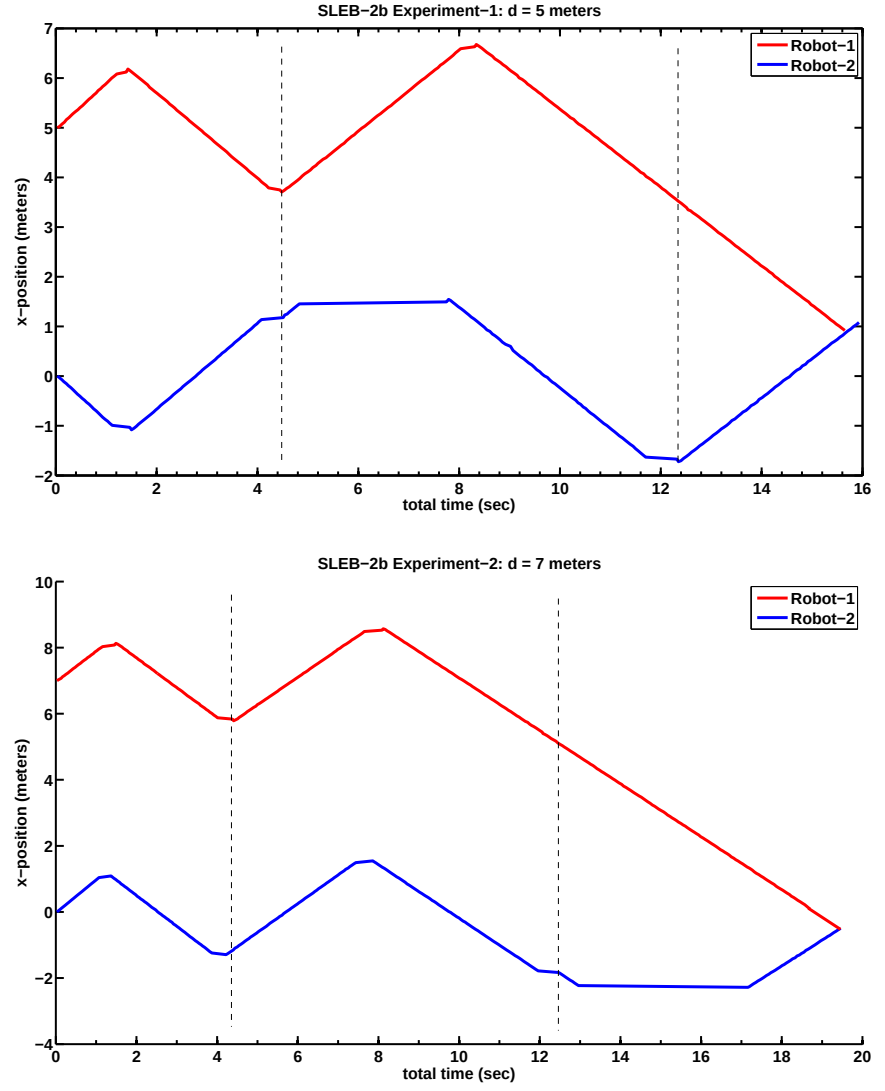
4.4.1 Simetrik Randevu Algoritması

Şuana kadar robotların *SR* algoritmasını aynı anda çalıştırdığını varsaydık. Bu standard bir varsayım olmasına rağmen genelde gerçekçi değildir; ajanlar birbirlerinden habersiz, ortamın farklı bölgelerinde farklı zamanlarda yaratılmış olabilir. Bu nedenle, bu bölümde asenkron hareket eden simetrik ajanların randevusunu çalışıyoruz. Senkronluğun robotların aramaya farklı zamanlarda başlamasıyla bozulduğunu düşünüyoruz.

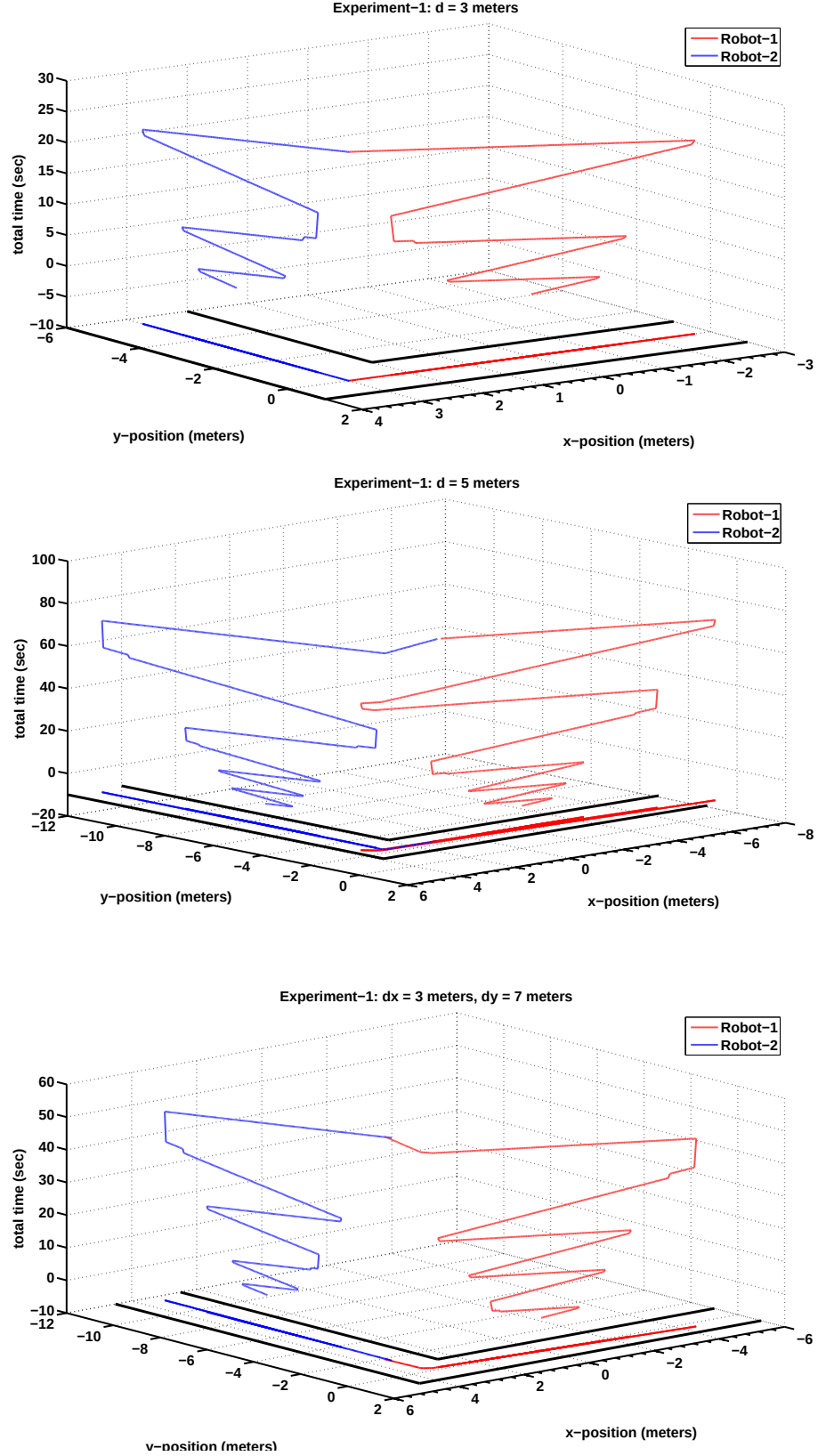
SR’da, bir roundun her aşamasının sonundaki bekleme zamanlarını, robotların hareketlerini senkron tutabilmek için kullandığımızı hatırlayalım. Bu nedenle, asenkron durum için bekleme zamanlarını kullanmıyoruz. Elde edilen bu algoritmaya *ASR* diyoruz. *ASR*’ın analizini rasgele değişken ϵ olmadan (i.e. $f_i = r^i$) yapıyoruz. Fakat algoritmamızın performansının ϵ olduğu ve olmadığı durumlar için nasıl etkilendiğini simülasyonlarla araştırıyoruz. Algoritma 3’de *ASR*’ın pseudocode’ünü sunuyoruz. Robot-1, *ASR*’ı robot-2’den t zaman sonra çalıştırmaya başlar. Robotlar t ’i bilmezler, aksi halde robot-1 t zaman bekler ve robot-2 ile başlar. Algoritma randevu gerçekleşince sonlanır. Sonraki bölümde, *ASR*’ın performansını analiz ediyoruz.



Şekil 4.10: **Sınır olan düz doğru deneyleri.** Deneyler düz bir koridor üzerinde yapılır. **SLEB:** Robot-1 veya Robot-2'e yakın olan bir sınır duvarı vardır.



Şekil 4.11: **Sınır olan düz doğru deneyleri.** Deneyler doğru bir koridor üzerinde yapılır. **SLEB-2b:** Hem Robot-1 hem de Robot-2'e yakın olan bir sınır duvarı vardır.



Şekil 4.12: **Kapalı eğri deneyleri.** Deneyler dikdörtgen şeklindeki bir koridor üzerinde yapılır. **ÜST:** Hem Robot-1 hem de Robot-2'nin köşeye olan uzaklığı üç metredir. **ORTA:** Hem Robot-1 hem de Robot-2'nin köşeye olan uzaklığı beş metredir. **ALT:** Robot-1'in köşeye olan uzaklığı üç metre iken Robot-2'nin köşeye olan uzaklığı yedi metredir.

Algorithm 3 $\mathcal{ASR}$: Asenkron durum ve bilinmeyen uzaklık için doğru üzerinde simetrik randevu stratejisi.

```

1:  $r \leftarrow 1.55$ 
   {Note that  $f(j) = r^j$  below}
2:  $i \leftarrow 0$ 
3:  $coinFlip \leftarrow \text{random from } \{-1, 1\}$ 
4:  $previousCoinFlip \leftarrow coinFlip$ 
5: while  $\text{checkRendezvous()} \neq \text{true}$  do
6:   if  $i = 0$  then
7:      $D_{i,1} \leftarrow f(2i)$ 
8:   else if  $coinFlip \neq previousCoinFlip$  then
9:      $D_{i,1} \leftarrow f(2i) - f(2i - 1)$ 
10:  else
11:     $D_{i,1} \leftarrow f(2i - 1) + f(2i)$ 
12:  end if
13:   $\text{moveTo}(coinFlip * f(2i))$ 
14:   $\text{moveTo}(-1 * coinFlip * f(2i + 1))$ 
15:   $previousCoinFlip \leftarrow coinFlip$ 
16:   $coinFlip \leftarrow \text{random from } \{-1, 1\}$ 
17: end while

```

4.4.2 Analiz

Bu bölümde, $\mathcal{ASR}$ algoritmasının ön analiz sonuçlarını sunuyoruz. Asenkron durumdaki randevu davranışları senkron durumdakinden farklılık gösterir. Yeni ayarlama, robotlar herhangi bir t' zamanda farklı roundları çalıştırıyor olabilir. Bu, SR 'in analizinde tanıttığımız randevu durumlarını etkiler. Örneğin, robot-1'in kritik round $i = k/2$ 'da olduğunu düşünelim. Eğer robot-2 $j < i$ olan bir roundda ise, o zaman robotlar yer değişimleri halen d 'den az olduğu için buluşamazlar. $\mathcal{ASR}$ 'da robotlar her rounda (ve bir roundun her aşamasına) aynı anda başlamaz. Bu nedenle, robot-1 round i 'in phase-1(2)'ine başladığında, robot-2 round i 'den farklı bir roundun farklı aşamasında hareket ediyor olabilir. Dahası, robot-1 bir aşamadaki varış noktasına ulaşmadan önce, robot-2 bulunduğu roundu tamamlayabilir, yeni bir rounda başlamak için para atabilir ve yönünü değiştirebilir..

Analizde t 'nin bilgisini kullanmıyoruz. Robotların başlangıç konfigürasyonlarının $(0, 2d)$ olduğunu hatırlayalım. Analizimizde şu değişkenleri kullanıyoruz: t^* , robot-1'in $x = 2d$ 'a ulaştığı zaman; i^* , robot-1'in t^* 'da bulunduğu round; j^* , robot-1 $x = 2d$ 'e ulaştığında (yani t^* 'da) robot-2'nin çalıştırdığı rounddur. Robotların randevu davranışlarını i^* 'dan itibaren incelemeye başlıyoruz ve bu rounddan önce robotların buluşmadıklarını varsayıyoruz. Çünkü $i < i^*$ olan roundlarda robotların kesin buluşma olasılıklarını bulmak zordur.

t^* 'da robot-1'in robot-2'den (round açısından) önde olduğunu düşünüyoruz, yani $i^* > j^*$ dir. Δ_1 , $i^* - j^*$ olarak tanımlıyoruz. $S_j^{t^*}$, robot- j 'in t^* 'deki hareket durumudur. $S_j^{t^*}$,

$$S_j^{t^*}(curr, dir, dest)$$

olarak sunulur. Burada $curr$, robotun şuanki konumunu, dir , yönünü ve $dest$, robot- j 'in t^* 'daki varmayı hedeflediği yeri gösterir. SR 'ın analizinde kullandığımız A_i , B_i ve D_i olaylarını aynı tanımlarıyla kullanıyoruz. Bu olayları şu şekilde tanımladığımızı hatırlayalım: A_i , robot-1'in (robot-2'e doğru) round i 'da başlangıçta sağa doğru hareket etmesi olayıdır; B_i , robot-2'nin (robot-1'e doğru) round i 'da başlangıçta sola doğru hareket etmesi olayıdır ve D_i round i 'daki gidilen mesafedir. Herhangi bir zamanda, robot-1'in round i 'i çalıştırırken robot-2'nin round j 'i çalıştırdığını düşünelim. Robotların para atışlarına göre mümkün olabilecek dört olay vardır. Bu olaylar: $E_1^{i,j} = A_i \wedge \overline{B_j}$, $E_2^{i,j} = \overline{A_i} \wedge \overline{B_j}$, $E_3^{i,j} = A_i \wedge B_j$ ve $E_4^{i,j} = \overline{A_i} \wedge B_j$ 'dir. Robotların round i 'daki yolculuğuna bağlı olarak, $\pm X_j^{i,j}$ 'i, robot- j 'in round i 'daki mümkün olan başlama noktalarını ifade etmek, $\pm Y_j^{i,j}$ 'i, robot- j için round i 'ın phase-1'indeki mümkün olan bitiş noktalarını ifade etmek ve $\pm Z_j^{i,j}$ 'i, robot- j için round i 'ın phase-2'deki mümkün olan bitiş noktalarını ifade etmek için kullanıyoruz. Şekil 4.13, robotların round i 'daki mümkün olan yolculuklarını gösterir.

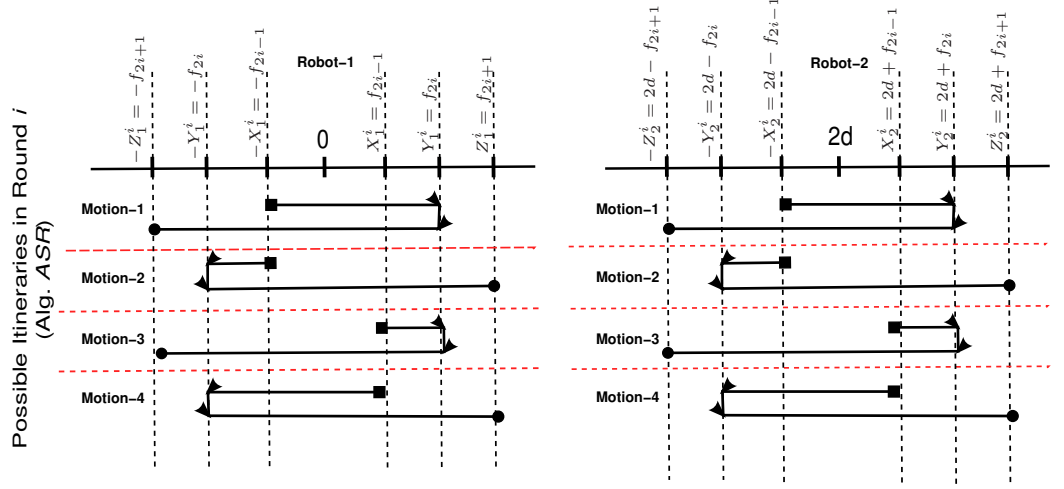
Lemma 15. t^* 'da, robot-2 $x = 2d$ 'nin sol tarafında ise, o zaman randevu t^* anından önce gerçekleşmiştir.

Kanıt. Robotların 1d bir ortamda, başlangıçta $(0, 2d)$ noktalarında konumlandırıldığını biliyoruz. Bu nedenle, eğer robot-1 $2d$ 'e vardığında, robot-2 bu noktanın sol tarafında hareket ediyorsa, o zaman robot-1 $x = 2d$ 'e varmadan yolu üzerinde robot-2 ile karşılaşmış demektir. $\square$

t^* 'dan önce randevunun gerçekleşmediğini, bu nedenle t^* 'da robot-2'nin $x = 2d$ 'nin sağında hareketli olduğunu varsayıyoruz. Sonraki lemmada robot-1'in $x = 2d$ 'e ulaşabilecek kadar ilerlemiş olduğu roundu buluyoruz.

Lemma 16. $r \geq 1.26$ seçimi için, robot $2d$ 'e round $i \geq k/2 + 2$ olduğunda ulaşır.

Kanıt. $\mathcal{ASR}$ 'da, robotlar tarafından round i 'da çalıştırılacak dört mümkün hareket deseni vardır. Bu desenler Şekil 4.13'de gösterilmiştir. Robot-1'in



Şekil 4.13: Robotların $\mathcal{ASR}$ algoritmasını çalıştırdıklarında mümkün olan yolculukları. Bir yolculuk üzerindeki oklar robotun hareket ettiği yönü gösterir.

$x = 2d$ 'e ulaşması için, bitiş noktası $2d$ 'den daha ileride olan bir aşamada sağa doğru hareket ediyor olması gerekir. Şekil 4.13'den, robot-1'in sağa-hareketli durumundaki mümkün olan bitiş noktalarının f_{2i} (Hareket 1 ve 3'e bakınız) ve f_{2i+1} (Hareket 2 ve 4'e bakınız) olduğunu görüyoruz. Bu nedenle, robot-1 $x = 2d$ 'e $f_{2i} \geq 2d$ veya $f_{2i+1} \geq 2d$ olursa ulaşabilir. Robotun en geç hangi roundda $x = 2d$ 'e ulaşabileceğini bulabilmek için, şuanda varmak istediği noktanın f_{2i} (round i phase-1 için bitiş noktası) olduğunu düşünüyoruz.

Eğer $f_{2i} \geq 2d$ ise:

$$f_{2i} \geq 2d = r^{2i} \geq 2r^{k+\delta}$$

En kötü durumda, $\delta = 1$ 'dir, ki bu aşağıdaki sonucu sağlar,

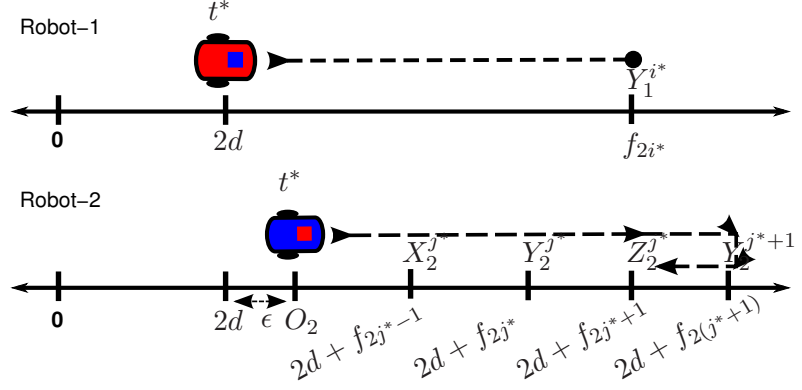
$$r^{2i} \geq 2r^{k+1} = 2i \geq \log_r 2 + k + 1 = i \geq \log_{r^2} 2 + k/2 + 1/2$$

$r \geq 1.26$ seçimi için, round $i \geq k/2 + 2$ olduğunda,

robot $2d$ 'e ulaşabilecek kadar ilerlemiştir.

□

Randevunun round $i < i^*$ 'dan önce gerçekleşmediğini varsayıyoruz. Bu nedenle, $\mathbb{P}[\overline{S}_{i < i^*}] = 1$ 'dir. $i \geq i^*$ olan sonraki roundlardaki, randevu davranışları $\Delta > 0$ ve $\Delta = 0$ olmasına bağlı olarak değişiklik gösterir. t^* anında Δ 'nın ne olacağı robotların para atışlarına ve t 'e bağlıdır. Teorem 27, ϵ olmayan $\mathcal{ASR}$ algoritmasının mesafe kompetitif oranını sunar. 4.4.2. ve 4.4.2. Bölümde sırasıyla $\Delta > 0$ ve $\Delta = 0$ durumlarındaki randevu davranışlarını çalışıyoruz.



Şekil 4.14: Robot-1 round i^* 'da ve robot-2 round j^* 'da iken, $x = 2d$ 'nin sağ tarafında randevuyu engelleyecek en kötü durum senaryosunda robotların yolculukları. Robotların t^* 'daki hareket durumları $S_1^{t^*}(2d, right, Y_1^{i^*})$ ve $S_2^{t^*}(2d + \epsilon, right, Z_2^{j^*})$ 'dir.

$\Delta > 0$ iken Randevü Davranışı

Bu bölümde, $\Delta > 0$ durumdaki randevü davranışlarını inceliyoruz. Robotların t^* 'da varış noktalarının kombinasyonları, $St_1 = (Y_1^{i^*}, Y_2^{j^*}) (= (f_{2i^*}, 2d + f_{2j^*}))$, $St_2 = (Z_1^{i^*}, Y_2^{j^*}) (= (f_{2i^*+1}, 2d + f_{2j^*}))$, $St_3 = (Y_1^{i^*}, Z_2^{j^*}) (= (f_{2i^*}, 2d + f_{2j^*+1}))$ ve $St_4 = (Z_1^{i^*}, Y_2^{j^*}) (= (f_{2i^*+1}, 2d + f_{2j^*+1}))$ olsun. St_1 , St_2 , St_3 ve St_4 durumları sırasıyla $E_1^{i^*,j^*}$, $E_2^{i^*,j^*}$, $E_3^{i^*,j^*}$ ve $E_4^{i^*,j^*}$ olaylarına karşılık gelir. Robotların para atışları birbirinde bağımsız olduğu için bu olayların her birisi $1/4$ olasılıkla gerçekleşir.

Robot-2 t^* 'da $2d$ 'nin solunda ise randevunun bu andan önce gerçekleşmiş olduğunu hatırlayalım (ispatı Lemma 15'de). Eğer t^* anından önce robotlar buluşmamışsa, o zaman Robot-1 $x = 2d$ 'e vardığı zaman, robot-2 $(2d, Z_2^{j^*})$ aralığında sağa/sola doğru hareket ediyor olacaktır.

- **RC-1:** Eğer $S_2^{t^*}(dest) = Y_2^{i^*}$ ise, o zaman randevü her zaman $S_1^{t^*}(dest) \geq S_2^{t^*}(dest)$ olduğunda gerçekleşir. Çünkü, robot-2 $S_2^{t^*}(dest)$ 'e ulaştığında sola doğru hareket etmeye başlayacaktır. Bu nedenle, robotlar birbirlerine doğru hareket ederken buluşurlar.
- **RC-2:** Eğer $S_2^{t^*}(dest) = Z_2^{j^*}$ ise, o zaman randevü her zaman $S_1^{t^*}(dest) \geq Z_2^{j^*+1}$ olduğunda gerçekleşir. Bu durumda için $x = 2d$ 'nin sağında randevunun olmasını engelleyecek *en kötü durum senaryosunu* düşünelim. Senaryo, Şekil 4.14'de gösterilmiştir: Burada ϵ 'nın çok küçük bir değere sahip olduğunu varsayalım. Robot-1'in $Y_1^{i^*}/Z_1^{i^*}$ 'a doğru hareket eder-

ken, robot-2'nin $Z_2^{j^*}$ 'a ulaştığını düşünelim. Robotun vardığı bu nokta, round j^* 'in 2. aşamasının bitiş ve round $j^* + 1$ 'in başlama noktasıdır. Robot-2 bu anda $j^* + 1$ 'a başlamak için para atar ve sağa doğru hareket etmeye karar verebilir. Bu durumda robotun yeni varmak istediği nokta $Z_2^{j^*+1}$ olacaktır.

St_1 ve St_2 için RC-1'in tutma durumunu inceliyoruz.

$$\begin{aligned}
S_1^{t^*}(dest) &\geq S_2^{t^*}(dest) \\
Y_1^{i^*} &\geq Y_2^{j^*} \\
f_{2i^*} &\geq 2d + f_{2j^*} \\
\delta &\leftarrow 1, i^* \leftarrow k/2 + x \text{ ve } i^* - \Delta \leftarrow j^* \\
r^{2(k/2+x)} &\geq 2r^{k+1} + r^{2(k/2+x-\Delta)} \\
r^{2x-1}(1 - r^{-2\Delta}) &\geq 2
\end{aligned} \tag{4.28}$$

Eğer $S_1^{t^*} \leftarrow Z_1^{i^*}$ ($= f_{2i^*+1}$) olursa, Denklem 4.28 aşağıdaki şekilde değişir

$$r^{2x-1}(r - r^{-2\Delta}) \geq 2 \tag{4.29}$$

St_3 ve St_4 için RC-2'in tutma durumunu inceliyoruz.

$$\begin{aligned}
S_1^{t^*}(dest) &\geq X_2^{j^*+1} \\
Y_1^{i^*} &\geq 2d + f_{2(j^*+1)} \\
f_{2i^*} &\geq 2d + f_{2(j^*+1)} \\
\delta &\leftarrow 1, i^* \leftarrow k/2 + x \text{ ve } i^* - \Delta \leftarrow j^* \\
r^{2(k/2+x)} &\geq 2r^{k+1} + r^{2(k/2+x-\Delta+1)} \\
r^{2x-1}(1 - r^{2(1-\Delta)}) &\geq 2
\end{aligned} \tag{4.30}$$

Eğer $S_1^{t^*} \leftarrow Z_1^{i^*}$ ($= f_{2i^*+1}$) olursa, Denklem 4.30 aşağıdaki şekilde değişir

$$r^{2x-1}(r - r^{2(1-\Delta)}) \geq 2 \tag{4.31}$$

Lemma 17. $\Delta \geq 2$ ve $r \geq 1.35$ için $\mathbb{P}[S_{i \geq k/2+3}] = 1$ 'dir.

Kanıt. $\Delta \geq 2$ için, Eşitsizlik 4.28, 4.29, 4.30 ve 4.31, $r \geq 1.35$ ve round $i \geq k/2 + 3$ iken her zaman tutar. Bu nedenle, robotlar $i = k/2 + 3$ 'da mümkün dört olaydan herhangi birisi olduğunda her zaman buluşurlar. $\square$

Lemma 18. $\Delta = 1$ ve $r \geq 1.39$ için $\mathbb{P}[S_{i \geq k/2+3}] = 3/4$ 'dir.

Kanıt. $\Delta \geq 1$ için, Eşitsizlik 4.28, 4.29 ve 4.31, $r \geq 1.39$ ve round $i \geq k/2+3$ iken her zaman tutar. Bu eşitsizliklerin herbirisi bir roundda robotların

mümkün olan para atışlarından birine denk gelir ve her birisinin olma olasılığı $1/4$ 'dür. Bu nedenle, robotlar $i = k/2 + 3$ itibaren $3/4$ olasılıkla buluşurlar. $\square$

$i \leq k/2 + 2$ iken randevunun gerçekleşmediğini varsayıyoruz ve bu roundlar sırasındaki beklenen gidilen mesafeyi Lemma 19'de hesaplıyoruz.

Lemma 19. $P[R_{i \leq k/2+2}] = 1$ varsayımı için, round $i \leq k/2+2$ iken beklenen gidilen mesafe

$$< r^{k+6} \frac{r+2}{r^2-1}, \text{dir.}$$

Kanıt. Başarısız bir round i 'da beklenen gidilen mesafe $\mathbb{E}[D_i \mid \bar{S}_i] = f_{2i+1} + 2f_{2i}$ 'dir. $i \leq k/2 + 2$ için $\mathbb{P}[R_i] = 1$ 'dir. Bu nedenle, $\mathbb{E}[D_{\leq k/2+2}]$ 'in

$$\begin{aligned} \sum_{i=0}^{k/2+2} \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] &= \sum_{i=0}^{k/2+2} \mathbb{E}[D_i \mid \bar{S}_i] \mathbb{P}[\bar{S}_i] \\ &= \sum_{i=0}^{k/2+2} \mathbb{E}[f_{2i+1} + 2f_{2i} \mid \bar{S}_i] \cdot 1 = \sum_{i=0}^{k/2+2} (r^{2i+1} + 2r^{2i}) \\ &< r^{k+6} \frac{r+2}{r^2-1} \end{aligned} \quad (4.32)$$

olduğunu buluyoruz, ki bu ifade $\delta = 0$ için en büyük olur. $\square$

Lemma 20. $\Delta > 1$ iken beklenen gidilen mesafe aşağıdaki gibi verilir

$$< r^{k+6}(r+2) \left(1 + \frac{1}{r^2-1} \right) \quad (4.33)$$

Kanıt. Lemma 17'den, $r \geq 1.35$ ve $i^* \geq k/2 + 3$ ise randevunun her zaman gerçekleştiğini biliyoruz. Bu nedenle, $\Delta > 1$ için beklenen gidilen mesafe $\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* = k/2+3}]$ 'dir. Lemma 19'de $\mathbb{E}[D_{i \leq k/2+2}]$ 'i hesaplamıştık, şimdi $\mathbb{E}[D_{i^* = k/2+3}]$ 'ü hesaplıyoruz.

$$\mathbb{E}[D_i] = \mathbb{E}[f_{2i+1} + 2f_{2i} \mid R_i] \text{ ve } \mathbb{P}[R_i] = 1$$

$$\mathbb{E}[D_{i=k/2+3}] = r^6(r+2)$$

$\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* = k/2+3}]$, Lemma 20'deki sınırı verir. $\square$

Lemma 21. $\Delta = 1$ iken beklenen gidilen mesafe aşağıdaki gibi verilir

$$< r^{k+6}(r+2) \left(\frac{1}{r^2-1} + \frac{1}{4-r^2} \right)$$

Kanıt. Lemma 18'dan, $r \geq 1.39$ ve $i^* \geq k/2 + 3$ ise randevunun $3/4$ olasılıkla gerçekleştiğini biliyoruz. Bu nedenle, $\Delta = 1$ için beklenen gidilen mesafe $\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* \geq k/2+3}]$ 'dir. Lemma 19'de $\mathbb{E}[D_{i \leq k/2+2}]$ 'i hesaplamıştık, şimdi $\mathbb{E}[D_{i^* \geq k/2+3}]$ 'ü hesaplıyoruz.

$$\begin{aligned} & \sum_{i=k/2+3}^{\infty} (\mathbb{E}[D_i \mid \bar{S}_i \wedge R_i] \mathbb{P}[\bar{S}_i] + \mathbb{E}[D_i \mid S_i \wedge R_i] \mathbb{P}[S_i]) \mathbb{P}[R_i] \\ &= \sum_{i=k/2+3}^{\infty} (r^{2i+1} + 2r^{2i}) \left(\frac{1}{4}\right)^{i-k/2-2} \\ &= \frac{r^6(r+2)}{(4-r^2)} \end{aligned}$$

$\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* \geq k/2+3}]$, Lemma 21'deki sınırı verir. $\square$

$\Delta = 0$ iken Randevu Davranışı

Bu bölümde, $\Delta = 0$ durumdaki randevu davranışlarını inceliyoruz. Bu durumda, t^* anında robotlar aynı roundu çalıştırılar. Öncelikle, başarısız bir roundda olan bir olaya göre robotların bitiş konfigürasyonlarını düşünüyoruz, Bunlar Şekil 4.15'de gösterilmiştir. Daha sonra da, bu konfigürasyonlara göre robotların bir sonraki rounddaki para atışlarına bakarak randevu olasılıklarını buluyoruz. $i^* \geq k/2 + 2$ olarak belirlediğimizi hatırlayalım. Geçici olarak, $i = i^* + 1$ ve $j = j^* + 1$ olarak belirleyelim.

Lemma 22. *Eğer $E_2^{i^*,j^*}$ olayı gerçekleşmişse, robotlar $E_4^{i,j}$ olayı hariç diğer tüm mümkün olaylarda (yani, $E_1^{i,j}$, $E_2^{i,j}$ ve $E_3^{i,j}$) buluşabilirler. Bir başka deyişle, $\mathbb{P}[S_{i \geq k/2+3} \mid E_2^{i^*,j^*}] = 3/16$ 'dir.*

Kanıt. Eğer $E_1^{i,j} \vee E_3^{i,j}$ olursa, robotlar $f_{2i+2} \geq 2d + f_{2j}$, $r \geq 1.45$ için tuttuğundan buluşurlar. Eğer $E_2^{i,j}$ olayı olursa, robotların t^* 'daki durumu Eşitsizlik 4.29'deki gibi olur, ki bu da $\Delta = 0$ iken $r \geq 1.39$ için her zaman tutar. $\square$

Lemma 23. *Eğer $E_4^{i^*,j^*}$ olayı olursa, robotlar $E_2^{i,j}$ veya $E_3^{i,j}$ olaylarından birisi olursa buluşur. Bir başka deyişle, $\mathbb{P}[S_{i \geq k/2+3} \mid E_4^{i^*,j^*}] = 1/8$ 'dir.*

Kanıt. $E_3^{i,j}$ olursa, robot-2 sola doğru $f_{2j} - 2d$ 'e hareket edecektir ve robot-1 $-f_{2i-1}$ 'e vardığında sağa doğru hareket etmeye başlayacaktır. Bu nedenle, eğer $f_{2j} - 2d \geq f_{2i-1}$ ise robotlar buluşurlar. Bu eşitsizlik de $r \geq 1.45$ için her zaman tutar. Eğer $E_2^{i,j}$ olursa, robotların t^* 'daki durumu Eşitsizlik 4.29'deki gibi olur, ki bu da $\Delta = 0$ iken $r \geq 1.39$ için her zaman tutar. $\square$

Lemma 24. *Eğer $E_1^{i^*,j^*}$ olayı olursa, robotlar $E_2^{i,j}$ veya $E_3^{i,j}$ olaylarından birisi olursa buluşur. Bir başka deyişle, $\mathbb{P}[S_{i \geq k/2+3} | E_1^{i^*,j^*}] = 1/8$ 'dir.*

Kanıt. $E_3^{i,j}$ olursa, robot-1 f_{2i-1} 'den sağa doğru f_{2i} 'a hareket edecektir ve robot-2 $2d + f_{2j-1}$ 'e vardığında sola doğru hareket etmeye başlayacaktır. Bu nedenle, eğer $f_{2i} \geq 2d + f_{2i-1}$ ise robotlar buluşurlar. Bu eşitsizlik, $r \geq 1.45$ için her zaman tutar. Eğer $E_2^{i,j}$ olursa, robotların t^* 'daki durumu Eşitsizlik 4.29'deki gibi olur, ki bu da $\Delta = 0$ iken $r \geq 1.39$ için her zaman tutar. $\square$

Lemma 25. *Eğer $E_3^{i^*,j^*}$ olayı gerçekleşmişse, robotlar $E_4^{i,j}$ olayı hariç diğer tüm mümkün olaylarda (yani, $E_1^{i,j}$, $E_2^{i,j}$ ve $E_3^{i,j}$) buluşabilirler. Bir başka deyişle, $\mathbb{P}[S_{i \geq k/2+3} | E_3^{i^*,j^*}] = 3/16$ 'dir.*

Kanıt. Eğer $E_1^{i,j} \vee E_3^{i,j}$ olursa, robotlar $f_{2i}/f_{2i+1} \geq 2d + f_{2j-1}$, $r \geq 1.45$ için tuttuğundan buluşurlar. Eğer $E_2^{i,j}$ olayı olursa, robotların t^* 'daki durumu Eşitsizlik 4.29'deki gibi olur, ki bu da $\Delta = 0$ iken $r \geq 1.39$ için her zaman tutar. $\square$

Lemma 26. $\Delta = 0$ iken beklenen gidilen mesafe aşağıdaki gibi verilir

$$< r^{k+6}(r+2) \left(\frac{1}{r^2-1} + \frac{3}{8-3r^2} \right)$$

Kanıt. Lemma 24, 22, 25, 25'den, $r \geq 1.45$ ve $i^* \geq k/2+3$ ise randevunun $5/8$ olasılıkla gerçekleştiğini biliyoruz. Bu nedenle, $\Delta = 0$ için beklenen gidilen mesafe $\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* \geq k/2+3}]$ 'dir. Lemma 19'de $\mathbb{E}[D_{i \leq k/2+2}]$ 'i hesaplamıştık, şimdi $\mathbb{E}[D_{i^* \geq k/2+3}]$ 'i hesaplıyoruz.

$$\begin{aligned} & \sum_{i=k/2+3}^{\infty} (\mathbb{E}[D_i | \bar{S}_i \wedge R_i] \mathbb{P}[\bar{S}_i] + \mathbb{E}[D_i | S_i \wedge R_i] \mathbb{P}[S_i]) \mathbb{P}[R_i] \\ &= \sum_{i=k/2+3}^{\infty} (r^{2i+1} + 2r^{2i}) \left(\frac{3}{8} \right)^{i-k/2-2} \\ &= \frac{3r^6(r+2)}{(8-3r^2)} \end{aligned}$$

$\mathbb{E}[D_{i^* \leq k/2+2}] + \mathbb{E}[D_{i^* \geq k/2+3}]$, Lemma 26'deki sınırı verir. $\square$

Robot tarafından gidilen mesafenin en büyük olduğu durum $\Delta = 0$ olduğu durumdur. Bu nedenle, mesafe kompetitif oranı en kötü durum için

hesaplıyoruz, bu da Teorem 27'deki sonucu verir.

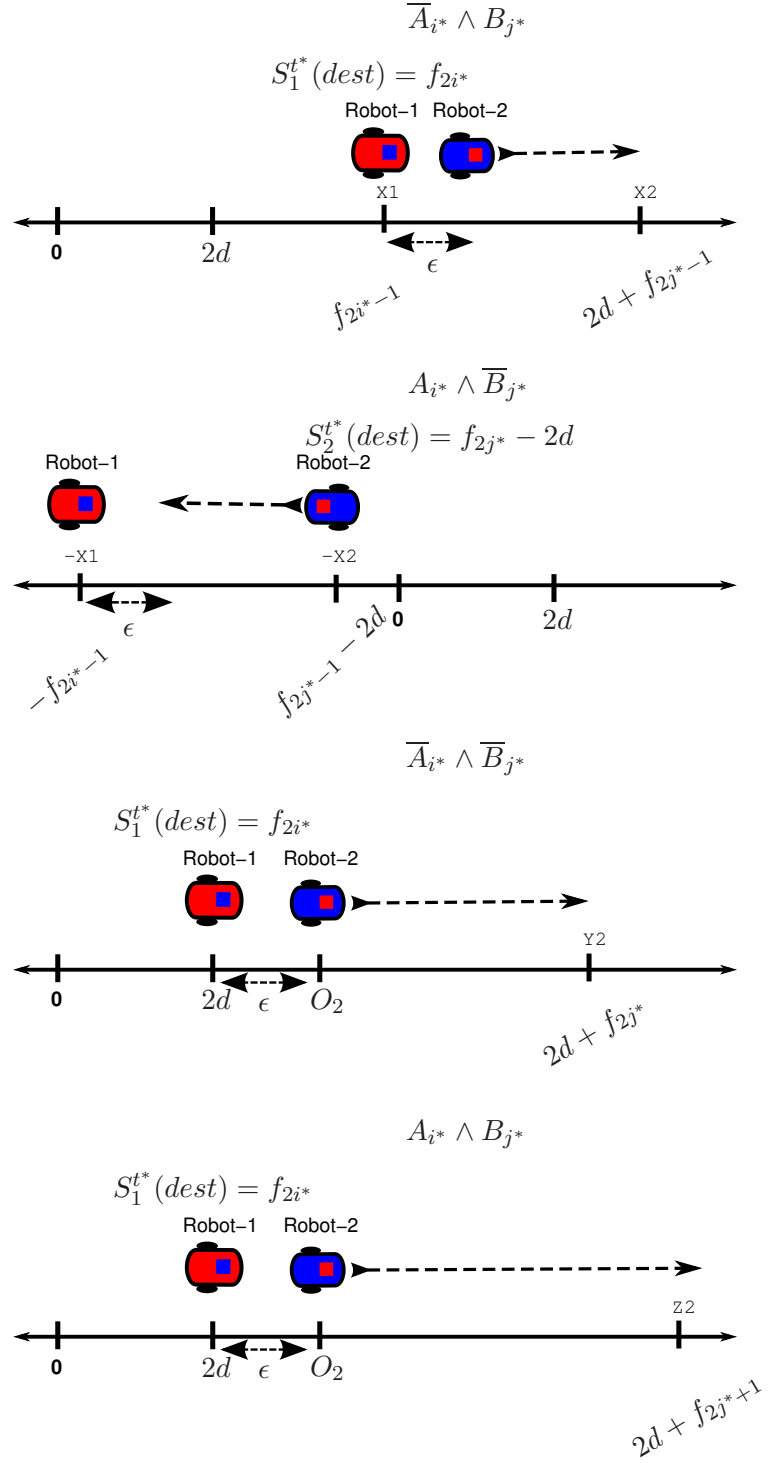
Theorem 27. *ASR için genişleme yarıçapının seçimi $r = 1.45$ 'dir, ki bu 86 kompetitif oran garantisini verir.*

Kanıt. Beklenen gidilen mesafe Lemma 26'deki sınırla aynıdır. $d = r^{k+\delta}$ olduğunu hatırlayarak, bu sınırdaki r^k 'nin yerine, $d r^{-\delta}$ 'i yerleştiriyoruz. Daha sonra, kompetitif oranı elde etmek için d' 'ye bölüyoruz. Kompetitif oranı elde etmek için, bu toplamı d' 'e bölüyoruz. Çıkan ifadeye, $r = 1.45$ yerleştirerek 86 kompetitif oranını elde ediyoruz. $\square$

4.4.3 Simülasyonlar

Geçen bölümde, sunduğumuz *SR* algoritmasının, robotlar tarafından aynı zamanda çalıştırılmaya başlandığını varsaymıştık. Bu standard bir varsayım olmasına rağmen, pratikte ihlal edilebilir. Bu nedenle, bu bölümde doğru üzerinde simetrik randevu arama probleminin asenkron durumunu simülasyonlarla çalışıyoruz.

Tezin bu bölümünde yaptığımız asıl katkı, randevu stratejisi *SR* ve onun, robotların stratejiyi eş zamanlı olarak çalıştırmaya başladıkları varsayımı altındaki performans analizidir. Bu senkron başlama, robotların pratikte her zaman mümkün olmayan global kontrolünü gerektirir. Bu bölümde, algoritma *SR*'in bu varsayımın ihlal edildiği durumdaki performansını çalışıyoruz. Robot-2'nin robot-1'den t zaman birimi daha sonra başladığı simülasyonları düşünüyoruz. Şekil 4.18 $5 \leq d \leq 1000$ mesafeleri ve $0 \leq t \leq 1000$ zaman gecikmeleri için ortalama kompetitif oranı gösterir. Bu simülasyon sonuçları algoritmamızın senkron hatalara karşı dayanıklı olduğunu gösterir. Çok küçük zaman gecikmelerinin performans üzerinde çok az etkisi vardır. Başlangıç uzaklığı olan d 'nin daha büyük değerleri için, mesafe kompetitif oranı 22'nin altında kalır. Ayrıca, çok uzun gecikmeleri aslında performansı iyileştirir. Aslında bu davranışın olmasını bekliyoruz, çünkü bizim simetrik randevu problemimiz bir robotun diğer robotu beklediği ("waits for mommy.") asimetrik duruma yaklaşır. Bir başka deyişle çok büyük t için (d 'e kıyasla), *lost cow* (Ing. kaybolan inek) problemine sahibiz. t arttıkça kompetitif orandaki azalma eğilimini açıkça görebiliyoruz. Gerçekte, d 'nin daha küçük değerleri için, kompetitif oran 10'un hemen üzerinde bir değerde dengelenir.



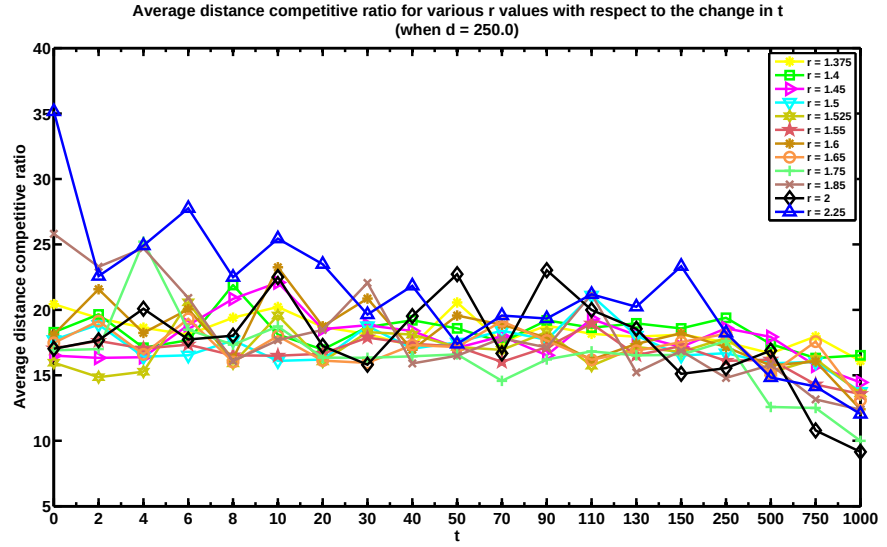
Şekil 4.15: $\Delta = 0$ iken, başarısız bir i^* 'da ve j^* 'da gelen olaylara göre robot konfigürasyonları.

Bu çizgiden devam edersek, eğer büyük senkronizasyon hataları beklersek, o zaman algoritmamız bu senaryoya daha iyi bir şekilde uyması için adapte edilebilir. $\mathcal{SR}$ 'in duran zamanları robotların hareketlerini tamamen senkronize etmek için tasarlanmıştır. Her robot her roundu diğer robotla senkron olarak çalıştırmak için bir roundun her aşamasının sonunda bekler. Eğer yakın-senkron bir başlangıç zamanı garanti edilmezse, bu duran zamanlar etkisizdir. Bu nedenle, $\mathcal{SR}$ algoritmasından duran zamanları kaldırarak ikinci bir simülasyon seti düşündük. Bu ortaya çıkan algoritmayı $\mathcal{ASR}$ olarak adlandırıyoruz, asenkron simetrik randevu (*asynchronous symmetric rendezvous*).

$\mathcal{ASR}$ 'in performansını iki aşamada çalışıyoruz. İlk olarak, asenkron durum için r in optimal seçimine deneysel olarak karar verdik. Daha sonra, r 'in bu belirlediğimiz değeri için $\mathcal{ASR}$ 'in performansını, başlangıç uzaklığı ve zaman gecikmesinin bir fonksiyonu olarak çalıştık. Özellikle, simülasyonlarımızın birinci aşamasında, uzaklık d ve zaman gecikmesi t 'deki değişimlere göre çeşitli r değerleri için robot-1 için ortalama kompetitif oranları bulduk. Her (r, d, t) kombinasyonu için 5,000 denemenin ortalaması alınmıştır. Simülasyon sonuçları (figure omitted) $\mathcal{ASR}$ 'in en iyi performansı $1.35 \leq r \leq 1.85$ olduğu zaman verdiğini gösterir. Bu r değerleri seti için, $\mathcal{ASR}$ algoritmasının ortalama mesafe kompetitif oranı, $5 \leq d \leq 1000$ mesafeleri için $t = 10$ iken Şekil 4.16'de ve $0 \leq t \leq 1000$ zaman gecikmeleri için $d = 250$ iken Şekil 4.17'de sunulmuştur.

İkinci aşamada, $r = 1.55$ değerini daha detaylı araştırmak için seçiyoruz. Bu r değeri için, (d, t) kombinasyonları için ek simülasyonlar çalıştırdık. $5 \leq d \leq 1000$ mesafeleri ve $0 \leq t \leq 1000$ zaman gecikmeleri için olan ortalama kompetitif oranı gösteren sonuçlar Şekil 4.19'de sunulmuştur. Bu oranlar 22 ile üst sınırlıdır. Hatta, $d = 500$, $t = 250$ olduğu istisna durum hariç, tüm değerler 20'nin altındadır.

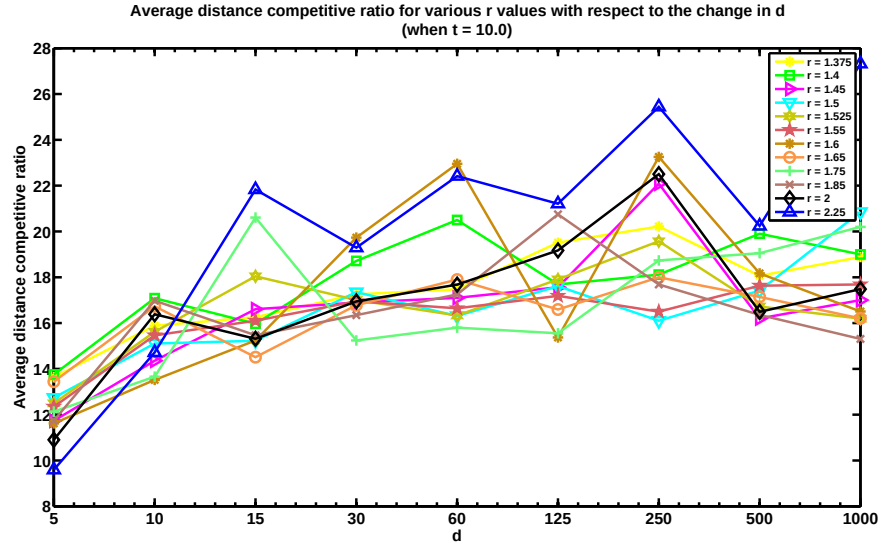
Sonuçlar en-kötü-durum zaman gecikmesinin $t = 0$ yakınında olduğunu ortaya çıkarır. Aslında, çok uzun bir gecikme robot davranışını asimetrik randevu arama problemine (yukarıdaki $\mathcal{SR}$ simülasyonlarında not edildiği gibi) yakınlaştırır. Aynı şekilde, t arttıkça kompetitif oranda açıkça bir azalma eğilimi görülür. Daha küçük d değerleri için, kompetitif oran 6'a yakın dengede durur. Bu da kabaca $r = 1.55$ için SmartCow algoritmasının kompetitif oranına eşittir.



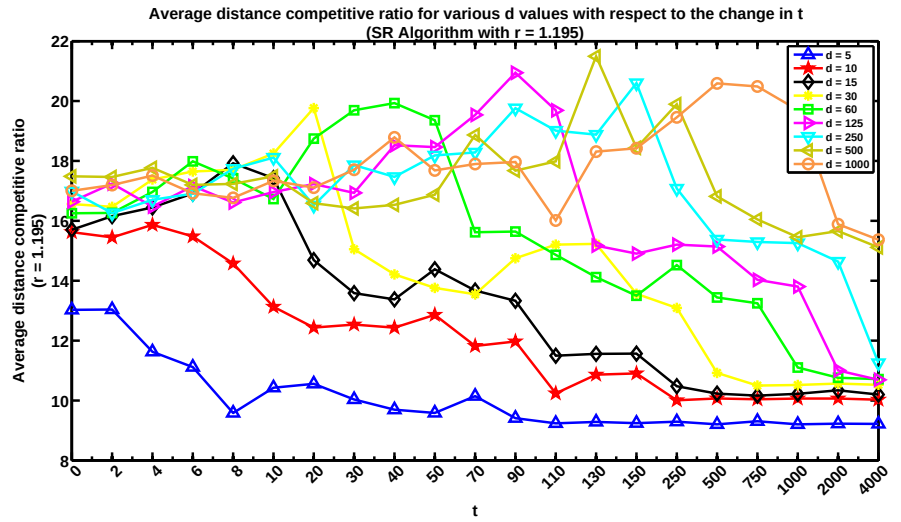
Şekil 4.16: $1.35 \leq r \leq 1.85$ için t 'deki değişime göre $d = 250$ iken, Algoritma $\mathcal{SR}$ 'ın simülasyonlarla ortalama mesafe kompetitif oranı.

Son olarak, simülasyonlarla $\mathcal{ASR}$ 'ın performansını ϵ olmadığı durum için araştırıyoruz. Böylece, teorik analizini ϵ olmadan yaptığımız $\mathcal{ASR}$ 'ın kompetitif performansına bulduğumuz üst sınırı doğrulamış oluyoruz. Simülasyon sonuçları rasgelelik eklemenin senkron durumda olduğu gibi asenkron durum için de algoritmanın performansında iyileştirme sağladığını gösterir. ϵ eklemek $\mathcal{ASR}$ 'ın performansını iyileştirir. Şekil 4.20, çeşitli r değerleri ve $0 \leq t \leq 1000$ zaman gecikmeleri için $d = 125$ iken, $\mathcal{ASR}$ 'ın algoritmasının ortalama mesafe kompetitif oranını gösterir.

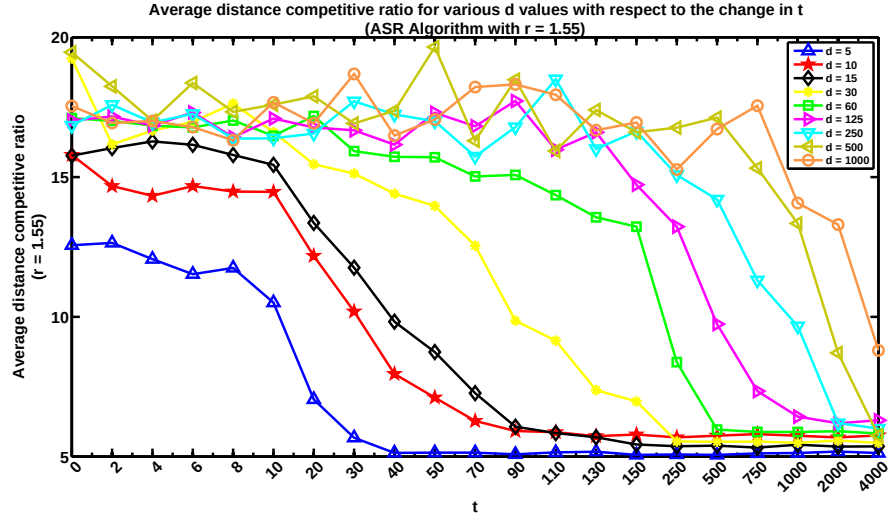
Özetle, $\mathcal{SR}$ ve $\mathcal{ASR}$ algoritmalarının simülasyon sonuçları zaman gecikmeleri bakımından güçlü performans gösterir. $\mathcal{SR}$ ve $\mathcal{ASR}$ için olan simülasyon grafiklerine göre, $\mathcal{SR}$ algoritmasının çok küçük zaman gecikmeleri için (senkron başlamaya yakın) daha iyi performans sergilerken, $\mathcal{ASR}$ algoritmasının çok uzun zaman gecikmeleri için daha iyi performans sergilediğini görüyoruz.



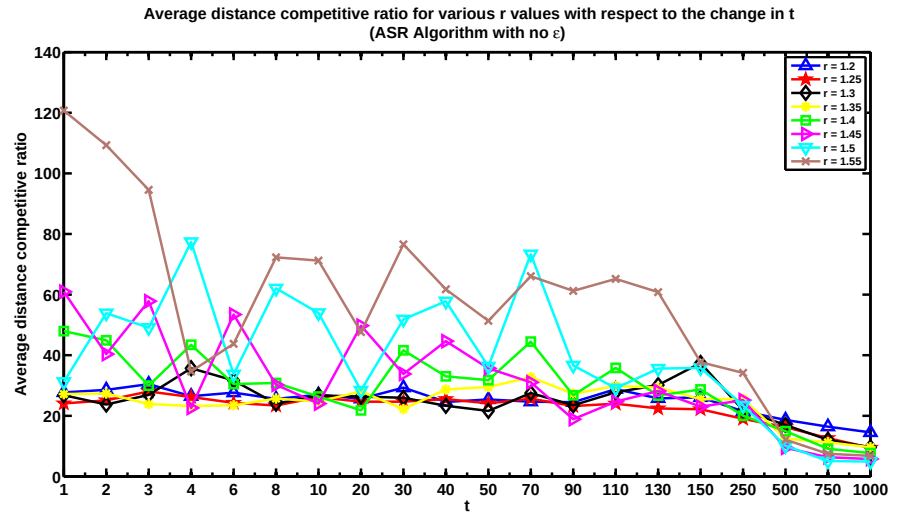
Şekil 4.17: $1.35 \leq r \leq 1.85$ için, d 'deki değişime göre $t = 10$ iken, Algoritma $\mathcal{SR}$ 'ın simülasyonlarla ortalama mesafe kompetitif oranı.



Şekil 4.18: $r = 1.195$ için d 'deki ve t 'deki değişime göre, Algoritma $\mathcal{SR}$ 'ın simülasyonlarla ortalama mesafe kompetitif oranı.



Şekil 4.19: $r = 1.55$ için d 'deki ve t 'deki değişime göre, Algoritma $\mathcal{ASR}$ 'ın simülasyonlarla ortalama mesafe kompetitif oranı.



Şekil 4.20: $1.2 \leq r \leq 1.55$ için, t 'deki değişime göre $d = 125$ iken, ϵ olmadığı durumda Algoritma $\mathcal{ASR}$ 'ın simülasyonlarla ortalama mesafe kompetitif oranı.

4.5. Sonuçlar

Bu bölümde, doğru üzerinde simetrik randevu arama problemini çalışıyoruz. 4.3. kısımda, problemi senkron durumda çalışıyoruz. Çalışmamızda, Spiral LostCow algoritmasını simetrik robotlar için genelleştiriyoruz. Teorik olarak optimal değer olan $r = 1.195$ 'i belirledik. Bu genişleme yarıçapı ile, simetrik stratejimiz toplam gidilen mesafe için 17.686 ve toplam zaman için 24.843 kompetitif orana sahip olduğunu gösterdik. Algoritmamız, daha önce en iyi bilinen, 26.650 zaman ve mesafe kompetitif oranına sahip olan Baston ve Gal'in algoritmasına bir iyileştirmedir. Algoritmamızın, aynı performans garantileriyle, sınırlı lineer ortamlara ve basit kapalı ortamlara adapte edilebileceğini gösterdik. Simülasyonlarda, $\mathcal{SR}$ 'ın performansını çok çeşitli r değerleri ve başlangıç uzaklıkları ile karşılaştırdık. Ayrıca, algoritmamızın navigasyonel hatalara da dayanıklı olduğunu gösterdik. Sonuçlarımız teorik analiz ile tutarlıdır. Son olarak, iki robotun koridor üzerinde bulunduğu deneylerden aldığımız sonuçları raporlandık.

Başlıca odaklandığımız problemin senkron ayarları olsa da, problemi ayrıca asenkron ayarlama da çalıştık. 4.4. kısımda, ilk olarak senkron simetrik randevu algoritmamızı asenkron duruma uyarladık (Algoritma $\mathcal{ASR}$). Daha uyarlanan algoritmamızın ön analiz sonuçlarını sunduk. Bilğimiz kadarıyla, literatürde bu doğru üzerinde simetrik randevu arama problemini bilinmeyen uzaklık ve asenkron robotlar için sunulan bir çalışma yoktur, bu nedenle çalışmamız ilk ön sonuçları sunar. Simülasyonlarda, $\mathcal{ASR}$ 'ın performansını çeşitli r ve t değerleri ve çeşitli başlangıç uzaklıkları için araştırdık. Buna ek olarak, $\mathcal{SR}$ 'ın ve $\mathcal{ASR}$ 'ın performanslarını çeşitli başlama zamanları için karşılaştırdık.

DÜZLEMSEL ORTAMLARDA SİMETRİK RANDEVU ARAMA

Randevu arama problemi gerçek hayatta da sıkça karşılaştığımız temel bir problemdir. Bu problemde, bilinmeyen bir ortamda birbirlerinin konumlarından habersiz uzak mesafelerde iletişim kuramayan oyuncular en kısa zamanda buluşmak isterler, bunun için de bulundukları alanı keşfetmeleri gerekir. Örneğin, bir bölgede farklı konumlara iniş yapan iki paraşütçü (Schelling, 1960) veya süpermarkette alışveriş yaparken ayrı düşen bir anne ve çocuk en kısa zamanda bir araya gelmek isteyebilir. Bunlara benzer birçok örnek verilebilir. Bu problemin öncelikli uygulamaları arama ve kurtarma operasyonlarıdır. Bu uygulamaya odaklanan Thomas ve Hulme Thomas and Hulme (1997) problemi özelliksiz bir çölde, arazide veya denizde kaybolan bir yayayı arayan bir arama kurtarma helikopterini düşünen gerçek bir senaryo üzerinde çalışmıştır.

Randevu, daha karmaşık davranışları gerçekleştirmek için de kullanılabilir. Uydu alıcılarının olmadığı, gücün sınırlı olduğu veya ortamın geometrisinin elverişli olmadığı durumlarda robotların başarılı iletişim kurmak için aralarında görüş hattının sağlanması gerekir. Bu gibi bir durumda robotların iletişim kurmak için birbirlerinin yanında olması gerekir, yani buluşmaya ihtiyaçları vardır. Keşif sırasında her robot tarafından yaratılan haritalar, randevu gerçekleştikten sonra birleştirilerek alanın haritasının (Örneğin, bir düşman arazinin haritası) çıkartılması için kullanılabilir. Birçok koşulda haritaların birleştirilebilmesi için robotların ortak bir referans noktası paylaşması gerektiğini not edelim. Eğer ortam tehlikeli veya insanlar için ulaşılması güçse mobil robotlar alanı keşfetmek için kullanılabilir. Nükleer alanlarda işlem yapıldığı, kirli atıkların temizlendiği veya denizaltı operasyonlarının gerçekleştirildiği durumlar buna örnektir. Farklı konumlardaki robotların bu gibi durumlarda keşif sırasındaki görevlerine karar vermek için önceden buluşmaları veya keşif bittikten sonra iletişim kurmak için biraraya gelmeleri gerekebilir. Ayrıca, birçok veri toplama uygulamaları robotların bir bölgeye konumlandırılmasını ve sonradan onları toplanmayı gerektirir. Bir randevu algoritması kolay veri toplamayı sağlayabilir. Bu uygulamaların hepsindeki amaç robotun randevuya ulaşmak için izlediği yolun uzunluğunu en aza indirmektir.

Bu bölümde, engelli ve engelsiz düzlemsel ortamlarda aynı özelliklere sahip iki veya ikiden çok robotun simetrik randevusunu çalışıyoruz. Simetrik versiyonda, her robot aynı randevu stratejisini takip eder, çünkü önceden bir

aramaya gelip arama sırasındaki rollerine karar verme şansları yoktur. Robotlar aralarındaki başlangıç uzaklığını, ortam üzerindeki kendi konumları ve birbirlerinin konumlarını bilmezler. Her robot bir görme sensörü ile donatılmıştır ve sınırlı $R > 0$ iletişim alanına sahiptir. Ancak, robotların birbirlerini aramaya başlamadan önce herhangi bir iletişim kurmadıklarını not edelim. Bilinmeyen uzaklık için, iki boyutlu senaryoda eğer iletişim alanı sıfır ise, randevu olmasını bekleyemeyiz (ispat Anderson and Fekete (2001) Lemma-1'dedir).

Robotlar önceden ortamla ilgili bir bilgiye sahip değildir. Ortamı ve engelleri sensörleri tarafından sağlanan görsel bilgiler ile öğrenirler. Robotların aramadaki kararları sadece alanın o anki gördükleri bölümüne bağlıdır, bu yüzden robotların araması online bir problem olarak görülebilir. Bunun dışında, robotlar unutkandır, yani geçmişe ait bir hafızaları yoktur. Robotların ortak bir zamanlaya bağlı olduğunu ve aramaya aynı anda başladığı senkron durumu çalışıyoruz.

Mobil bir robot hareketindeki en kritik parametre fiziksel hareket ettiği zamandır. Bu nedenle, çalışmamızda robotun beklenen gidilen mesafesini en aza indirmeyi hedefliyoruz. Simetrik randevu stratejimizin performansını ölçmek için kompetitif analiz taslağını kullanıyoruz. Ortamı bilmeyen robotun randevu için ilerlediği mesafeyi robotun herşeyi bildiği optimal durumdaki ilerlediği mesafe ile karşılaştırıyoruz.

İlk olarak, başlangıçta açık (engelsiz) bir düzleme yerleştirilen robotlar için simetrik bir randevu arama stratejisi sunuyoruz ve onun kompetitif performansını analiz ediyoruz. Stratejimizin kompetitif karmaşıklığının $O(d/R)$ olduğunu gösteriyoruz. Burada d robotlar arasındaki başlangıç uzaklığı ve R iletişim yarıçapıdır. İkinci olarak, açık düzlem için olan simetrik randevu stratejimizi poligonal engellerin olduğu bilinmeyen ortamlar için genişletiyoruz. Genişletilmiş stratejimiz ortamın tamamen kapsandığını garantiler. Stratejimizi kare ve çevrilen robotlar için analiz ediyoruz ve kompetitif oranının $O(d/D)$ olduğunu gösteriyoruz. Burada, D robotun bir kenarının uzunluğudur. Bu sonucu alırken, ayrıca rasgele poligonal ortamların kapsanması için de bir üst sınır da elde ediyoruz. Ayrıca hem açık düzlem hem de engelli ortamlar için sunduğumuz stratejilerimizi ikiden çok robotun olduğu durum için genişletiyoruz ve analizlerini sunuyoruz. n robot sayısı olmak üzere, açık düzlem için stratejimizin kompetitif karmaşıklığının $O(d/\log(n)R)$ ve engelli ortamlar için $O(d/\log(n)D)$ olduğunu gösteriyoruz.

Bu bölümde düzlemsel ortamlarda çalıştığımız randevu arama probleminin iki yönü vardır. Birinci yönü en kısa randevunun gerçekleşmesi ki biz buna odaklanıyoruz. İkinci ilgi çekici yönü de, robotun bilinmeyen ortamı kapsamayı ve keşfetmesidir.

Problemin ilk yönü literatürde ağırlıklı asimetrik versiyon için çalışılmıştır. Asenkron durumda randevu için olan çoğu çalışma randevu için kararlı bir algoritma tasarlamaya odaklanmıştır. Bu çalışmalarda aynı kararlı algoritmayı çalıştıran ajanların randevusunun mümkün olabilmesi için simetrinin kırılması gerekir. Bu da ajanların *etiket* adı verilen birbirlerinden ayrı kimliklerinin olduğu varsayılarak yapılır.

Problemin ikinci yönünde, tek robot ile poligonal engellerle donatılmış bir ortamı keşif problemini ele alıyoruz. Keşif problemi ağırlıklı ortamın haritasının oluşturulması için kullanılan öncelikli bir uygulamadır. Harita oluşturma, mobil robotların üzerindeki sensörler tarafından edinilen ölçümlerden ortamın planını çıkarmak anlamına gelir. Ortam bilinmediğinde keşif, *online bir problem*dir ve online algoritma içerisinde kapsamlı olarak çalışılmıştır. Online bir algortmada, tüm inputa önceden erişim yoktur. Algoritmaya çalışması sırasında sunulur. Ortamın keşfi ağırlıklı tek robot ile çalışılmış olsa da birden daha fazla robot ortamı keşfi için sunulan algoritmalar da vardır. Biz bu bölümde sunduğumuz keşif algoritmamızın analizinde kompetitif oran çatısı kullanıyoruz. Literatürde ortamın rectilinear veya poligonal engeller ile işgal edildiği durumda bu yaklaşımı kullanarak keşif algoritmalarının analizini yapan çalışmalar vardır. Bunlardan 5.1. Bölümde bahsediyoruz.

Bu bölüm şu şekilde organize edilmiştir: İlk olarak 5.1. kısımda literatürde iki boyutlu ortamlarda randevu arama problemi ile ilgili çalışmaları sunuyoruz. Bu problem literatürde çok çeşitli formülizasyonlar için çalışılmıştır. 5 ve 5.2.5. kısımlarında, sırasıyla problemi açık düzlemde ve poligonal engellerin olduğu ortamlarda çalışıyoruz. Açık düzlemde iki ve ikiden çok robot için olan randevu stratejimizin analizlerini sırasıyla 5.2.2. ve 5.2.4. Bölümlerde sunuyoruz. Engelli ortamlarda iki ve ikiden çok robot için olan randevu stratejimizin analizlerini sırasıyla 5.3.3. ve 5.3.5. Bölümde sunuyoruz. Son olarak, simülasyon ve gerçek ortam deney sonuçlarımızı sırasıyla 5.2.5. ve 5.3.6. Bölümde sunuyoruz.

5.1. İlgili Çalışmalar

Literatürdeki randevu problemini iki boyutlu ortamlarda ikiden ve ikiden çok robot için araştıran çalışmalar, randevu arama stratejileri veya randevu için lokal kontrol stratejileri geliştirmeye odaklanmıştır. Randevu problemlerinin bu iki tipi birbirinden farklıdır. Bu farklardan 2.4. Bölümde bahsediyoruz. Bizim bu tezde odaklandığımız randevu arama problemidir. Diğer versiyon ajanların birbirinin durumunu gözleyebildiği, hareketli bir objeye doğru robot takibi ve navigasyonu ile ilgilenir. Bu nedenle, robotun ve hedefin kinematik denklemlerini birleştirmek gibi kontrol-teorik yönler önem verilmiştir. Bu versiyon için bilinmeyen ortamlarda çok robotlu randevu problemine odaklanılan literatürde bir çok çalışma Belkhouche et al. (2005); Fang et al. (2008); Ganguli et al. (2006); Lin et al. (2003, 2007); Yu et al. (2008) vardır. Bu çalışmalar ajanlar arasında aktif iletişim olmadan tüm ajanları önceden belli olmayan tek bir noktada buluşturmayı hedefler ve bunu sağlayacak her ajan için lokal kontrol stratejileri geliştirmeye odaklanılır. Lin ve Morse problemin çözümü için iki tip strateji önermiştir. Birincisi, ortak bir saate bağlı olarak senkron kalan ajan stratejilerini içerir. İkinci versiyonda ise, ortak bir saat olmadan birbirlerinden bağımsız olarak uygulanabilen stratejileri içerir. Fangy Fang et al. (2008) çok robotlu randevu problemini düzlem üzerinde *bekleme-noktaları* (*Ing. dwell-points*) adını verdiği bir noktalar kümesi olduğunu düşünerek çalışmıştır. Sınırlı iletişime sahip bu ajanların her birisi başlangıçta bir bekleme noktası üzerinde konumlandırılır. Ajanlar algı alanları içinde olan bekleme noktalarına olan uzaklıklarına ve bir ajanın lokal referansına göre bu bekleme noktalarının yönlerine karar verebilir. Bir ajan, bir bekleme noktasında birden fazla ajanın olup olmadığına bakabilir, fakat kaç tane ajan olduğunu sayamaz. Randevu ajanlar bir hepsi birbirlerinin algı alanı içerisinde olan bekleme noktalarına hareket ettiğinde gerçekleşir.

Düzlemsel ortamlarda randevu arama problemine odaklanan çalışmaların çoğu engellerin olmadığı durum ve asimetric versiyon Alpern and Baston (2006); Anderson and Fekete (2001); Thomas and Hulme (1997) için düşünülmüştür. Simetrik versiyonu düşünen çalışmalar kafes (*Ing. lattice*) üzerindedir. Problem robotların asenkron olduğu: robotların ortak bir saate bağlı olmadığı durum (senkronluk robotların stratejileri farklı zamanlarda çalıştırılması veya hareketlerinin bir düşman tarafından kontrol edilmesiye bozulabilir) Bampas et al. (2010); Collins et al. (2010); Czyzowicz et al. (2010b)'de çalışılmıştır.

Dudek ve Roy Roy and Dudek (2001), harita çıkarmak için sınırlı iletişim alanına sahip bilinmeyen bir ortamda keşif yapan iki robotun randevusunu düşünmüştür. Keşif devam ederken randevuyu sağlayan birçok alternatif algoritma önermişlerdir. Algoritmaları dönüm noktası (*Ing. landmark*) dedikleri potansiyel randevu lokasyonları varsayımı üzerinedir. Bu noktalar robotlar tarafından keşif sırasında seçilebilir. Bu noktaların konumları rasgele bir sensör ile hesaplanan ayırt edicilik (*Ing. distinctiveness*) ölçüsüne bağlıdır.

Anderson ve Weber Anderson and Weber (1990), bir bina veya bir alışveriş merkezinde birbirinden ayrılan iki arkadaşın buluşmasını çalışmıştır. Kişilerin buluşabileceği n mümkün lokasyon vardır ve aramada aynı stratejileri kullanmak zorundadır. Aramaların kesikli zamanlarda devam ettiğini düşünülür. Önerdikleri rasgeleleştirilmiş stratejinin beklenen buluşma zamanının üst sınırı büyük n için $0.829n$ 'dir. $n = 2$ ve $n = 3$ olduğu durumlarda beklenen buluşma zamanları sırasıyla 2 ve $8/3$ adımdır.

Anderson Anderson and Fekete (2001), düzlem üzerinde hareket eden iki oyuncunu asimetric randevusunu çalışmıştır. İlk olarak oyuncuların aralarındaki vektör uzaklığının sonlu bir kümeden eşit olasılıkla seçilerek konumlandırıldıkları durumu düşünmüşlerdir. Ayrıca oyuncuları aralarındaki başlangıç uzaklığı olan d 'i bildiği, ancak diğer oyuncuya olan yönünü bilmediği durumu da çalışmışlardır. Son olarak, birinci oyuncunun ikinci oyuncunun konumu bildiği ancak ikinci oyuncunun sadece birinci oyuncuya olan başlangıç uzaklığı bilgisine sahip olduğu durumda aldıkları sonuçları sunmuşlardır.

n boyutlu tamsayı kafes üzerinde simetrik randevu Alpern and Street (2001)'de ve asimetric randevu Alpern and Baston (2006)'de çalışılmıştır. İki kör oyuncu bir kafesin düğümleri üzerine aralarındaki vektör uzaklığı 2 ve herhangi bir koordinat eksenine paralel olacak şekilde yerleştirilir. Bir komşu düğümde en kısa beklenen zamanda ($R(n)$) randevunun gerçekleşmesi için oyuncular her periyotta bir komşu düğüme hareket etmek zorundadır. Oyuncular hiçbir yer ve yön kavramına sahip değildir. Oyuncuların birbirlerinin başlangıç konumlarını bilmedikleri varsayılmıştır. Asimetric randevu değeri $R^a(n)$ ve simetrik randevu değeri $R^s(n)$ olarak gösterilir. Alpern and Gal tarafından bulunan $R^a(1) = 3.25$ sonucu Alpern and Baston (2006)'de genişletilmiş ve $R^a(2) = 197/32 = 6.16$ ve $2n \leq R(n) \leq (32 + 12n^2 - 2n - 3)/12n^2$ olarak bulunmuştur.

Alpern aynı problem üzerine olan çalışmasında Alpern and Street (2001), oyuncuların aralarındaki başlangıç uzaklığını bildiğini düşünmüştür ve en düşük beklenen randevu zamanlarının $\lim R^a(n)/n \leq 8/3$ ve $\lim R^s(n)/n \leq 56/9$ olarak elde etmiştir. Alpern'in bu çalışması, Gal ile $R^a(1) = 3.25$ olarak sundukları çalışmanın ve Baston'un $R^s(1) = 4.5$ olarak sunduğu çalışmanın bir devamıdır.

Thomas ve Hulme Thomas and Hulme (1997), bir çölde kaybolan ve bulunmak isteyen bir yayayı arayan bir kurtarma helikopterinin yapacağı arama tiplerinin modelleri için olan simülasyon sonuçlarını raporlandırır. Yaya helikopterden daha yavaştır, fakat yaya helikopteri helikopterin onu tespit edebileceğinden daha uzak bir mesafede tespit edebilir. Mümkün olan arama stratejileri helikopterin azalan spiraller içinde uçuşması veya bölgeyi gide gele taramasıdır. Yaya sabit kalabilir veya helikopterin sesini duyduğunda ona doğru rasgele hareket edebilir veya yürüyebilir. Bu stratejilerden hangisinin daha iyi olacağı simülasyonlarla araştırılır.

Czyzowicz et al Czyzowicz et al. (2010b), bilinmeyen bağlı bir arazi üzerinde farklı etiketlenmiş iki ajanın (robotun) buluşmasını çalışmıştır. Ajanlar nokta olarak modellenmiştir ve hareketleri etiketlerine ve bilinmeyen ortama bağlıdır. Her ajanın hareketi aynı zamanda asenkron bir düşmana bağlıdır. Düşman ajanının hızını değiştirebilir durdurabilir veya ileri geri bile hareket ettirebilir. Ajanın rotasındaki her parçadaki yürüyüşü sürekli ve tamamlayana kadar bunun dışına çıkmaz. Yol bağlı iç kısma sahip bir kapalı T bölgesinde rasgele rasyonel içteki noktalardan başlayan rasgele iki ajanın asenkron randevusunu garantileyen kararlı algoritmalar sunulmuştur.

Collins et. al. Collins et al. (2010), 2d bir alanda asenkron olarak hareket eden sınırlı iletişim alanına sahip iki ajanın randevusunu çalışmıştır. Her ajanın düzlem üzerindeki kendi başlangıç pozisyonunu bildiği varsayılır. Fakat ajanlar birbirlerinin pozisyonlarından habersizdir. Ajanlar aynı koordinat sistemi üzerindeki o anki konumlarını gösterecek senkron pusulalara ve aynı birim uzunluğa sahiptir. Randevu algoritmasının maliyeti, randevu gerçekleşene kadar robotların başlangıç pozisyonlarından itibaren yaptığı yolculuklarını uzunluklarının toplamıdır. Bu maliyet düşman tarafından kontrol edilen ajanların tüm mümkün asenkron hareketlerinin maksimumu olarak alınır. d ajanlar arasındaki başlangıç uzaklığı olmak üzere, önerilen randevu algoritmasının maliyeti $O(d^2)$ 'dir.

$N > 0$ boyutlu bir grid üzerinde asenkron olarak hareket eden iki isimsiz ajanın randevusu Bampas et al. (2010)'de çalışılmıştır. Ajanlar aramaya aralarındaki başlangıç uzaklığı en fazla d olan rasgele konumlardan başlarlar. Genel ayarlamasıyla problemin çözümü mümkün olmadığından gridin N boyutlu euclid bir alana gömülü olduğu ve her ajanın kendi başlangıç pozisyonunun kartezyen kordinatlarını bildiği, fakat diğer ajaninkini bilmediği varsyılmıştır. Ajanların uzunlupu $O(d^\delta poly \log d)$ olan bir yolculuk yaptıktan sonra buluşmalarını sağlayan bir algoritma önerilmiştir. Aynı problemin geometrik versiyonuna olan uyarlaması için, ajanların $O(((\frac{d^\delta}{r})poly \log(\frac{d^\delta}{r})))$ mesafe ilerledikten sonra buluşmaların mümkün olacağı bir algoritma tasarlamışlardır. Burada, $r \geq 1$ için $r = \min(r_1, r_2)$ 'dir. r_1 ve r_2 sırasıyla ajanların iletişim alanını gösterir. Her ajan kendi iletişim alanının bildiği düşünülmüştür.

5.1.1 Keşif Problemi

Bu tezde randevu arama problemini iki boyutlu engelli ve engelsiz ortamlar için de çalışıyoruz. Randevunun bu ortamlarda gerçekleşmesi için robotun ortamı keşfetmesi gerekir. Şimdi literatürde keşif problemini araştırmış çalışmaları inceliyoruz. Keşif problemi ağırlıklı ortamın haritasının oluşturulması için kullanılan öncelikli bir uygulamadır. Harita oluşturma mobil robotların üzerindeki sensörler tarafından edinilen ölçümlerden ortamın planını çıkarmak anlamına gelir ve robotların hafızasının olmasını gerektirir. Bu problemin motivasyonunu şu uygulamayla verelim: Bir robotun depremde dolayı yıkılmış bir binanın içini hayatta kalan insanların yerlerini tespit edebilmek için keşfetmesi gerektiğini düşünelim. Keşiften önce robot binanın içinin geometresini bilmez. Robotun, üzerindeki sensörleri kullanarak keşif sırasında binanın içerisindeki tüm noktaları görmesi gerekir. Bu yolla tüm potansiyel hayatta kalan kişilerin yeri, keşif yapan robot tarafından tespit edilebilir. Bu uygulamada olduğu gibi birçok gerçek durumda robotlar ortamla ilgili tam bilgiye sahip değildir, yani çevrenin haritasına sahip değildir. Ortam bilinmediğinde keşif online bir problemdir ve online algoritma içerisinde kapsamlı olarak çalışılmıştır. Online bir algoritmada tüm inputa önceden erişim yoktur, algoritmaya çalışması sırasında sunulur. Ortamın keşfi ağırlıklı tek robot ile çalışılmış olsa da, birden daha fazla robot ortamı keşfi için sunulan algoritmalar da vardır.

Bilinmeyen bir ortamı mobil robotlar tarafından keşfetmek hem sınırlı

hemde sınırsız görüş alanı için geniş olarak çalışılmıştır. Bu alandaki birçok araştırma kompetitif çatı ile ilgilenir. Sınırsız görüş için olan çalışmalarda, bir görüş sensörünün kendinden herhangi uzaklıktaki bir objeyi görebileceği varsayılır. Sınırsız görüş olan en önemli çalışmalardan biri Deng et al. (1998)'dir. Burada yazarlar engelsiz bir rectilinear poligon için 2 kompetitif bir algoritma vermişlerdir. Yol uzunluğunun L_1 metriktaki ölçümü için geçerli olan bu değer, daha sonra Hammar et al. (2002) tarafından $5/3$ 'e geliştirilmiştir. Hammar et al. (2006) tarafından yapılan başka bir çalışmada bu $3/2$ olarak daha da geliştirilmiştir. Kleinberg Kleinberg (1994) bu problem için olan herhangi bir kararlı stratejinin kompetitif oranının alt sınırının $5/4$ olduğunu kanıtlar ve aynı kompetitif orana sahip rasgeleştirilmiş bir keşif algoritması sunar.

Rasgele sayıda poligonal engellerle dolu olan poligonal ortamlarda keşif yapan, sınırsız görüşe sahip robotlar için tüm engeller paralelogramlar olsa bile hiçbir kompetitif strateji olmadığı gösterilmiştir Deng et al. (1998). Daha sonra bu sonuç Albers et al. (2002)'de, k tane engel olan bir poligonu keşfeden online bir algoritma için $\Omega(k)$ alt sınır kompetitif oranı sağlanarak geliştirilmiştir. Bu sınır karesel engeller için de aynen geçerlidir. Diğer bir taraftan, $O(k)$ kompetitif oranına sahip bir algoritma da vardır Deng et al. (1991); Kalyanasundaram and Pruhs (1993). Dahası Kalyanasundaram and Pruhs (1993)'de, engellerin belirli şekilleri için konveks ve sınırlı aspect oranlı ($\sqrt{k}$) optimal kompetitif oranı sağlanır.

Sınırlı görüşe sahip bir robot tarafından poligonların keşfi Czyzowicz et al. (2013); Fekete and Schmidt (2010); Gabriely and Rimon (2001,?, 2003); Ghosh et al. (2008,?); Icking et al. (2000); Kolenderska et al. (2010); Ntafos (1992)'de çalışılmıştır. Gabriely and Rimon (2001)'de kompetitif oranı $1 + 3(\frac{\pi S}{A})$ olan bir algoritma sunulur, burada π poligonun çevresine bağlı bir miktar ve A poligonun alanıdır. Gabriely and Rimon (2001, 2003) belirli tip poligonlarda keşif başarısız olur (dar koridorlar gibi). Kesikli adımlarda keşif Ghosh et al. (2008)'de düşünülmüştür. Robot hareketsizken alanı keşfedebilir ve keşif algoritmasının maliyeti keşif sırasındaki durma sayısı ile ölçülür. Fekete and Schmidt (2010) kesikli görüş altında deliklerin olmadığı rectilinear poligonların keşfini düşünmüştür. Keşfin toplam maliyeti, L_1 metrikte keşif yolunun uzunluğu ve keşif yolu boyunca görüş noktalarının lineer bir kombinasyonu olarak ölçülür. Basit (engelsiz) rectilinear poligonlar için kompetitif oranı $O(\log A)$ olan online bir keşif algoritması önerilir. Burada A poligonun aspect oranını gösterir.

Bilinmeyen hücresel bir odanın keşfi Ghosh et al. (2008)'de çalışılmıştır. Oda geçilemeyen engellerin olabileceği bir tamsayı gridin kare hücrelerini içeren bir P poligonu olarak tanımlanır. Bir hücreyi keşfetmek için robot içerisine girmelidir. Robot bir kere hücrenin içeride olduğunda, hangi dört komşu hücrenin var olduğunu ve sınır kenarlarının hangisi olduğunu bilir. C hücreli E kenarlı ve H engelli bir P poligonunu $S \leq C + \frac{1}{2}E + H - 3$ adımlı tur uzunlukları ile keşfeden bir *CellExplore* stratejisi sağlamışlardır.

Şimdi önceden bahsettiğimiz sınırlı iletişim alanına sahip bir robotun engellerle donatılmış bir ortamın keşfedilmesini ele alan, bizim çalışmamıza yakın çalışmaları Czyzowicz et al. (2013); Gabriely and Rimon (2005) daha yakından inceliyoruz.

Gabriely ve Rimon Gabriely and Rimon (2005) bir online navigasyon algoritması (*SAD1*) sunmuştur. Algoritmalarında D boyutlu mobil bir robot daha önceden konumunu bilmediği bir hedef T 'i bulmak için bilinmeyen bir ortamı sınırlı görüş alanına sahip sensörleri ile keşfeder. *SAD1* için üst sınır $l \leq \frac{8\pi}{D}l_{opt}^2 + \frac{2\pi R_0^2}{D}$ dir. Burada D robotun boyutu, R_0 başlangıç arama yarıçapı ve l_{opt} S 'den T 'e kadar olan optimal offline yolun uzunluğudur. *SAD1* daki fikir şu şekildedir: Bizim formülizasyonumuza benzer olarak, robot S merkezli R_0 başlangıç yarıçaplı bir disk seçer ve diskin S 'den erişilebilecek kısmını arar. Robot daha sonra diskin yarıçapını $2^{i/2}R_0$ olmak üzere $i = 1, 2, \dots$ için değiştirir ve hedefi bulana kadar aramaya devam eder veya S 'den erişilebilecek tüm bölge T 'i bulmadan keşfedilir. Disk içindeki aram işlemi şu şekildedir: Robot sürekli olan alanı online olarak D boyutlu hücrelerden oluşan bir gride ayırarak kesikli hale getirir. Grid sadece boş ve kısmen işgal edilmiş hücreleri içerir. Robot boş grid hücreleri üzerinde bir alan kapsama turu için basit bir DFS algoritması çalıştırır. Her yeni hücreyi hedefi bulmak için tarar. Bir kez yeni bir hücreye girdiğinde, robot T 'i bulmak için ek olarak komşu olan kısmen işgal edilmiş hücreleri de dolaşır.

Arama modelleri bizimkisinden farklıdır, çünkü kısmen işgal edilmiş bir hücreyi ve boş bir hücreyi tarama maliyetinin aynı olduğunu varsaymışlardır. Dahası, robot gridin boş hücreleri üzerinde alan kapsama turu için basit bir DFS algoritması çalıştırdığından, sadece engellerin olduğu belirli arazi sınıfları keşfedilebilir, örneğin dar koridorların olmadığı veya tam aynı karelerden oluşan araziler gibi. Bizim algoritmamızın bir alt rutini bu arama stratejisini genişletir ve kısmen işgal edilmiş olan hücreleri kapsamanın maliyetinin üst sınırını ayrı olarak hesaplar. Genişletilmiş olan stratejimiz ortamın tam

kapsanmasını garantiler.

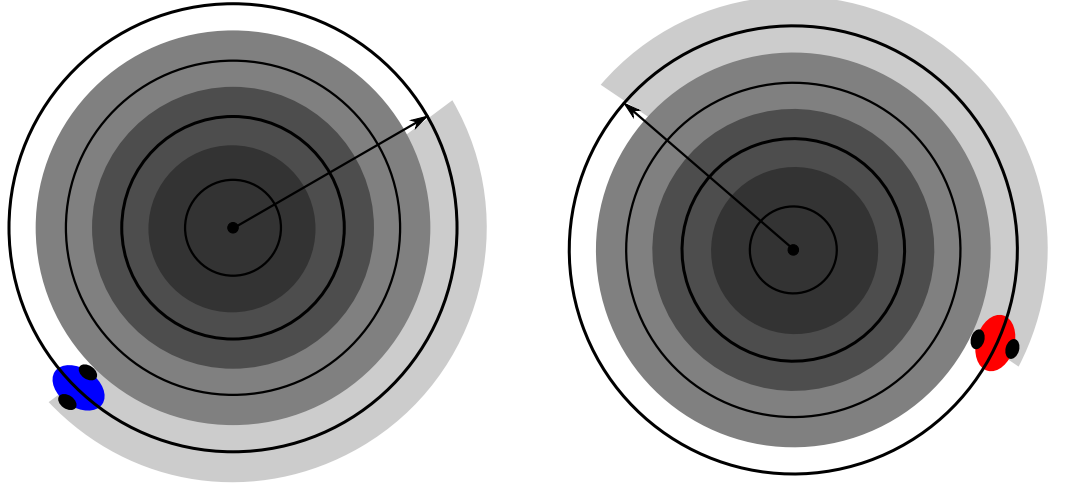
Czyzowicz et al. (2013)'da, görüş mesafesi 1 olan düzlem üzerinde hareketli bir nokta olarak temsil edilen mobil bir robot geçilemeyen engeller olan bilinmeyen bir araziye keşfetmek zorundadır. Hem arazi hem de engeller rasgele poligonlar olarak modellenmiştir. Engeller hariç arazideki tüm noktalar keşfedilmelidir. Arazi $\mathcal{P}_0, \dots, \mathcal{P}_k$ içinde kalan bağlı olmayan poligonal $\mathcal{P}_0, \dots, \mathcal{P}_k$ engeller olan rasgele bir $\mathcal{P}_0$ poligonu olarak sunulur. Yani, arazi $\mathcal{T} = \mathcal{P}_0 \cup \dots \cup \mathcal{P}_k$ 'dir. Her $\mathcal{P}_i$ poligonunun sınırlarının arazi içinde olduğu varsayılır. Karmaşıklığı $O(P + A + \sqrt{Ak})$ olan bir keşif algoritması gösterirler. Burada A arazinin alanıdır, P arazinin çevresidir (engellerin çevresinde dahil) ve k engellerin sayısıdır. Algoritmadaki fikir, ortamı F kenar boyutlu küçük kısımlara bölmek (F , pozitif real $F \leq \sqrt{2}/2$) ve bu parçaları DFS kullanarak dolaşmaktır. Bir T parkesi ve $C \subseteq T$ hücresi düşünelim. A_C , C nin alanı ve B_C , onun sınırlarının uzunluğu olsun. P_C , C 'nin sınırının $\mathcal{T}$ nin içerisindeki kısmı ve R_C , C sınırının geride kalan parçası olsun. Yani $R_C = B_C - P_C$ 'dir. Herhangi bir C hücresi için $R_C \leq 8(A_C/F + P_C)$ 'nin tuttuğu gösterilir.

Bizim keşif algoritmamızda $A_C = D^2$ ve $F = D$ 'dir. Bir hücreyi kapsama zamanının $T_c \leq 8D$ olduğunu gösteriyoruz. Bir engelle kesişen bir hücrenin sınırını kapsamının uzunluğu olmadan bile $R_c \geq T_c$ 'dir. Bu da bizim stratejimizin daha iyi bir hücre kapsama zamanına sahip olduğunu gösterir. Bizim algoritmamız bir T arazisini $8A/D$ uzunlukta bir yol alarak keşfeder, ki bu $27P + 24A/F + 12\sqrt{2}F + F \sum_{i=1}^n \sqrt{k_{T_i}}$ (Herhangi bir $F \leq \sqrt{2}/2$) yol uzunluğu kullanan Algoritma LET'den daha üstündür.

5.2. Engelsiz Ortamlar

5.2.1 Düzlemsel Ortamlarda Simetrik Randevu Arama

Bu bölümde, simetrik randevu problemini iki robot ile engellerin olmadığı bir düzlem üzerinde çalışıyoruz. Robotlar düzlem üzerinde birbirlerinin konumlarını veya kendi global pozisyonlarını bilmiyorlar. Aynı zamanda birbirleri arasındaki uzaklığı da bilmiyorlar. $r > 1$ sabit olsun (r için olan optimal seçime analizde karar veriyoruz). Robotlar arasındaki mesafe $d = r^{k+\delta}$ 'dir. Burada $k \in +\mathbb{Z}$ ve $\delta \in (0, 1]$ 'dir. Sıfırdan farklı bir iletişim alanı (*Ing.*



Şekil 5.1: Stratejiye bakış: Her robot beklemek veya hareket etmeye karar vermek için para atar. Eğer tura atarsa, robot kendi başlangıç noktasının merkez olduğu içiçe daireler üzerinde hareket eder. Eğer yazı atarsa, şuanki round i 'da hareket etmesi gerektiği mesafe kadar bekler. Daire çizerken, robot $2R$ genişliğinde olan bir halka süpürür. Burada R , robotun iletişim alanıdır.

communication range) R olduğunu varsayıyoruz, öyle ki randevu, robotlar arasındaki uzaklık R 'dan küçük olduğunda gerçekleşir.

Algoritma 4, düzlem üzerindeki simetrik senkron stratejimizi gösterir. Stratejimizi roundlara bölüyoruz. Her roundun başında, her robot hareket etmek veya beklemeye karar vermek için para atar. Algoritma 5, eğer round i 'daki atılan paranın sonucu tura olursa robotların nasıl hareket edeceğini gösterir. Robotun başlangıç konumu merkez olacak şekilde, içiçe daireler (içerden dışarıya doğru) çizerek robot round i 'da $r^i + R$ yarıçaplı bir disk kaplar. İletişim yarıçapı R olduğundan, en üstteki dairenin yarıçapı $r^i + R$ 'dir. Bir daireyi çizerken, robot genişliği $2R$ olan bir halkayı süpürür. Bu nedenle, robot en üst daireye ulaşana kadar, robotun çizdiği dairenin yarıçapını $2R$ artırıyoruz. Robot Algoritma 5'de verilen hareket desenini takip eder. Robot ilk önce R birim kuzeye çıkar ve R yarıçaplı bir disk çizer. Daha sonra $2R$ birim kuzeye çıkar ve yarıçapı $3R$ olan bir daire çizer.

$N = \left\lfloor \frac{r^i}{2R} \right\rfloor$ olmak üzere, round i için, robot başlangıç konumundan kuzeye doğru yarıçapı $(2N + 1)R$ birime ulaşana kadar aynı hareket desenini takip eder. Daha sonra, robot $(2N + 1)R$ mesafe güneye doğru inerek, başlangıç konumuna geri döner. Şekil 5.1, algoritmayı çalıştıran robotun hareketini gösterir. Eğer para atışının sonucu yazı ise, robot diğer robotla senkron ola-

bilmek için (Algoritma- 4) şuan ki round i 'da hareket etmesi gereken mesafe kadar bekler. Sonraki bölümde, algoritmamızın kompetitif oranını ispatlıyoruz.

Algorithm 4 $\mathcal{SRP}$: Düzlem üzerinde simetrik randevu arama stratejisi

```

1:  $r \leftarrow 1.225$ 
2:  $R \leftarrow 1$ 
3:  $i \leftarrow 1$ 
4:  $head \leftarrow 0$ 
5:  $tail \leftarrow 1$ 
6:  $waitTime \leftarrow 0$ 
7:  $outcome \leftarrow \text{random from } \{0,1\}$ 
8: while  $\text{checkRendezvous()} \neq \text{true}$  do
9:    $N \leftarrow \left\lfloor \frac{r^i}{2R} \right\rfloor$ 
10:  if  $outcome = tail$  then
11:     $waitTime \leftarrow (2N + 1)R + 2\pi(N + 1)^2R$ 
12:     $\text{wait}(waitTime)$ 
13:  else if  $outcome = head$  then
14:     $\text{move}(R, N)$ 
15:  end if
16:   $i \leftarrow i + 1$ 
17:   $outcome \leftarrow \text{random from } \{0,1\}$ 
18: end while

```

Algorithm 5 $Procedure_1$: $\text{move}(R, N)$

```

1: for  $j = 0$  to  $N$  do
2:   if  $j = 0$  then
3:      $\text{move } R \text{ units to north}$ 
4:   else
5:      $\text{move } 2R \text{ units to north}$ 
6:   end if
7:    $\text{move in a circle that has a radius } (2j + 1)R$ 
8: end for
9:  $\text{move } (2N + 1)R \text{ to south}$ 

```

5.2.2 Analiz

Bu bölümde, algoritmamızın performansının analiz ediyoruz ve robot tarafından beklenen gidilen mesafeye bir üst sınır buluyoruz. Analizimizde sadece robot-1'in performansını gösteriyoruz. Simetrik stratejilerden dolayı, robot-2'nin performansı da aynıdır. Algoritma 4'in (mesafe-kompetitif-oran) performansı üzerinde sınırlar elde etmek için, çalışmasını iki aşamaya bölüyoruz: round $i \leq k$ ve round $i \geq (k + 1)$. Analize başlamadan her aşamanın tanımını veriyoruz. Stage-1'in uzunluğu (bilinmeyen) başlangıç uzaklığı olan

$d = r^{k+\delta}$ 'e bağlıdır. Bu aşama robotun buluşabilecek kadar çok ilerlemediği önceki roundları kapsar. Stage-2'de, randevu davranışı tutarlıdır, o yüzden beklenen gidilen mesafeyi sonsuz toplam kullanarak hesaplayabiliriz. 5.2.4. ve 5.2.4. Bölümde sırasıyla Stage-1 ve Stage-2'nin analizlerini sunuyoruz.

Simdi analizimizdeki değişkenleri tanımlıyoruz. R_i algoritmanın round i 'da hala aktif olma olayı olsun. R_i 'in olduğunu varsayarak, şunları tanımlıyoruz: A_i robot-1'in round i 'da hareket etme olayı olsun. B_i robot-2'in round i 'da hareket etme olayı olsun. S_i olayı başarı için proxy olayımız olacaktır. $i \leq k$ için, $\Pr[S_i] = 0$ 'dir, çünkü robotlar buluşabilecekleri kadar çok ilerleememişlerdir. $i > k$ için, $S_i = (A_i \wedge \overline{B_i}) \vee (\overline{A_i} \wedge B_i)$ 'i tanımlıyoruz. Bu tanımın her iki robotta hareket ediyorkenki olayı kasıtlı olarak içermediğini not edelim, fakat bu olayın olasılığı, başlangıç robot konfigürasyonu harici bir geometriye uymadığı sürece sıfırdır. Algoritma birinci roundda her zaman aktiftir, bu nedenle $\mathbb{P}[R_0] = 1$ 'dir. Algoritmanın round i 'da hala aktif olma olasılığı, robotların round i 'a kadar buluşmama olaylarının birleşik olasılığıdır. Bu da, $i > 0$ için $\mathbb{P}[R_i] = \mathbb{P}[\overline{S}_0 \wedge \cdots \wedge \overline{S}_{i-1}]$ 'dir. Daha fazlası, şuna sahibiz

$$\mathbb{P}[R_i] = \begin{cases} 1 & 0 \leq i \leq k, \\ 2^{-i+k} & k < i. \end{cases} \quad (5.1)$$

Robot tarafından beklenen gidilen mesafeye bir üst sınır bulmak için, randevunun sadece bir robotun hareketli diğer robotun beklediği ve robotlar buluşacak kadar ilerledikleri durumda olduğunu düşünüyoruz.

Şimdi Algorithm 4'i çalışmaya hazırız. Sonraki iki bölüm, sırasıyla Stage-1 ve 2'nin analizini sunar. Algoritmanın mesafe ve zaman kompetitif oranının şu sınır olduğunu ispatlıyoruz.

Theorem 28. *Algorithm 4'in genişleme yarıçapı için en iyi seçim $r = \sqrt{1.5} \approx 1.225$ 'dir. Bu şu mesafe kompetitif oranı verir*

$$7.854 \frac{d}{R} + 40.8577 \frac{R \log d}{d} + 16.0664 \frac{R}{d} + 55.3236$$

Zaman kompetitif oran bu sınırın iki katıdır.

Burada, $\log d$ doğal logaritmadır.

Stage-1 Analizi

Round $i \leq k$ sırasındaki beklenen gidilen mesafeyi hesaplıyoruz. Robotlar yerdeğişimleri en fazla $r^k < r^{k+\delta}$ olduğundan bu roundda buluşamazlar. A_i olayının olduğunu düşünün: robot round i sırasında aktiftir.

$$N = \left\lfloor \frac{r^i}{2R} \right\rfloor \leq \frac{r^i}{2R}.$$

olsun. Roundda izlenen yol boyunca, robot başlangıç noktasına dönmeden önce $(2N + 1)R \leq r^i + R$ kuzeye ilerler. $0 \leq j \leq N$ için, robot yarıçapı $(2j + 1)R$ olan her dairenin çevresini dolaşır. Bu aşamada beklenen gidilen mesafe şudur

$$\begin{aligned} \mathbb{E}[D_i \mid A_i] &\leq 2(r^i + R) + 2\pi R \sum_{j=0}^N (2j + 1) \\ &= 2(r^i + R) + 2\pi R(N + 1)^2 \\ &\leq 2(r^i + R) + \frac{\pi(r^i + 2R)^2}{2R} \\ &= \frac{\pi}{2R} r^{2i} + 2(1 + \pi)(r^i + R). \end{aligned} \quad (5.2)$$

Şimdi bu aşamadaki beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 29. *Stage-1 sırasındaki beklenen gidilen mesafe aşağıdaki ile sınırlandırılmıştır*

$$\begin{aligned} \sum_{i=1}^k \mathbb{E}[D_i] &< d^2 \left(\frac{\pi}{4R} \right) \frac{r^2}{r^2 - 1} \\ &\quad + (1 + \pi) \left(d \cdot \frac{r}{r - 1} + (1 + \log_r d) R \right). \end{aligned}$$

İspat. Robotlar buluşabilecekleri kadar ilerlemediklerinden, round k 'a kadar olan her round yapılır. Bir başka deyişle, $0 \leq i \leq k$ için $\mathbb{P}[R_i] = 1$ 'dir. Verilen bir roundda, robot $1/2$ olasılıkla aktiftir. Denklem (5.2)'i kullanarak, beklenen gidilen mesafenin aşağıdaki gibi olduğunu buluyoruz

$$\begin{aligned} \sum_{i=0}^k \mathbb{E}[D_i] &= \sum_{i=0}^k \mathbb{E}[D_i \mid R_i] \mathbb{P}[R_i] = \frac{1}{2} \sum_{i=0}^k \mathbb{E}[D_i \mid A_i] \\ &< \frac{\pi}{4R} \sum_{i=0}^k r^{2i} + (1 + \pi) \left(\sum_{i=0}^k r^i + (k + 1)R \right) \\ &= \frac{\pi}{4R} \cdot \frac{r^{2k+2} - 1}{r^2 - 1} + (1 + \pi) \left(\frac{r^{k+1} - 1}{r - 1} + (k + 1)R \right) \\ &< d^2 \left(\frac{\pi}{4R} \right) \frac{r^{2-2\delta}}{r^2 - 1} + (1 + \pi) \left(d \frac{r^{1-\delta}}{r - 1} + (1 + \log_r d) R \right), \end{aligned}$$

Bu ifade $\delta = 0$ en büyük olur.

Stage-2 Analizi

Bu bölümde, round $i \geq (k + 1)$ olan roundları düşünüyoruz. Başarısız roundlardaki beklenen gidilen mesafeyi ve başarılı (final) rounddaki beklenen gidilen mesafeyi ayrı hesaplıyoruz.

Lemma 30. *Stage-2'deki başarısız roundlardaki beklenen gidilen mesafe aşağıdaki ile sınırlıdır*

$$\sum_{i=k+1}^{\infty} \mathbb{E}[D_i | \bar{S}_i] < d^2 \frac{\pi r^2}{8R(2-r^2)} + \frac{(1+\pi)}{2} \left(d \cdot \frac{r}{2-r} + R \right).$$

İspat. Robot-1 tarafından gidilen beklenen gidilen mesafeyi sınırlandırıyoruz; robot-2 için olan hesaplama benzerdir. Algoritmanın round i 'da aktif olduğunu varsayarsak; robotların $S_i = (A_i \wedge \bar{B}_i) \vee (\bar{A}_i \wedge B_i)$ olduğu zaman kesinlikle bulunduğunu biliyoruz. Dahası, robot-1 tarafından gidilen mesafe sadece A_i olduğu zaman sıfırdan farklıdır. Bu $\mathbb{E}[D_i | \bar{S}_i \wedge R_i] = \mathbb{E}[D_i | A_i \wedge B_i \wedge R_i]$, ve $\mathbb{E}[D_i | S_i \wedge R_i] = \mathbb{E}[D_i | A_i \wedge \bar{B}_i \wedge R_i]$ anlamına gelir.

Denklem (5.1) ve (5.2)'i kullanarak, Stage-2'deki başarısız roundlardaki toplam beklenen gidilen mesafenin aşağıdaki gibi olduğunu buluyoruz

$$\begin{aligned} & \sum_{i=k+1}^{\infty} \mathbb{E}[D_i | A_i \wedge B_i \wedge R_i] \mathbb{P}[A_i \wedge B_i \wedge R_i] \\ & < \sum_{i=k+1}^{\infty} \left(\frac{\pi}{2R} r^{2i} + 2(1+\pi)(r^i + R) \right) \left(\frac{1}{2} \right)^{i-k+2} \\ & = \sum_{j=0}^{\infty} \left(\frac{\pi}{2R} r^{2(j+k+1)} + 2(1+\pi)(r^{j+k+1} + R) \right) \left(\frac{1}{2} \right)^{j+3} \\ & = \frac{\pi r^{2k+2}}{16R} \sum_{j=0}^{\infty} \left(\frac{r^2}{2} \right)^j + \frac{(1+\pi)}{4} \left(r^{k+1} \sum_{j=0}^{\infty} \left(\frac{r}{2} \right)^j + 2R \right) \\ & = \frac{\pi r^{2k+2}}{8R(2-r^2)} + \frac{(1+\pi)}{2} \left(\frac{r^{k+1}}{2-r} + R \right) \\ & = d^2 \frac{\pi r^{2-2\delta}}{8R(2-r^2)} + \frac{(1+\pi)}{2} \left(d \cdot \frac{r^{1-\delta}}{2-r} + R \right), \end{aligned}$$

bu ifade $\delta = 0$ için en büyük olur.

Lemma 31. *Başarılı bir roundda beklenen gidilen mesafe aşağıdaki ile sınırlıdır*

$$\sum_{j=k+1}^{\infty} E[D_j \mid S_j] \leq \frac{\pi}{8R} d^2 + \frac{1+2\pi}{4} (d+R).$$

İspat. $S_j = (A_j \wedge \overline{B_j}) \vee (\overline{A_j} \wedge B_i)$ olduğunu biliyoruz ve sadece $A_j \wedge \overline{B_j}$ olayı robot-1 tarafından gidilen mesafeye katkıda bulunur. Başarılı bir roundda, robot-1 robot-2'i bulmadan önce yarıçapı $r^{k+\delta}$ olan dairesel bir alanı arar. Robot-1 tarafından toplam gidilen mesafe Denklem (5.2)'e benzer olarak $r^i = r^{k+\delta} = d$ 'i kullanarak hesaplanır. j 'den bağımsız olarak şuna sahibiz

$$\begin{aligned} E[D_j \mid A_j \wedge \overline{B_j} \wedge R_j] &\leq (d+R) + \frac{\pi(d+2R)^2}{2R} \\ &= \frac{\pi}{2R} d^2 + (1+2\pi)(d+R). \end{aligned}$$

Bu nedenle, Stage-2 sırasındaki başarılı bir rounddaki beklenen gidilen mesafe aşağıdaki gibidir

$$\begin{aligned} &\sum_{j=k+1}^{\infty} \mathbb{E}[D_j \mid A_j \wedge \overline{B_j} \wedge R_j] \mathbb{P}[A_i \wedge \overline{B_i} \wedge R_i] \\ &= \left(\frac{\pi}{2R} d^2 + (1+2\pi)(d+R) \right) \sum_{i=k+1}^{\infty} \left(\frac{1}{2} \right)^{i-k+2} \\ &= \frac{\pi}{8R} d^2 + \frac{1+2\pi}{4} (d+R). \end{aligned}$$

Mesafe Kompetitif Oranın Hesaplanması

Her iki aşamadaki beklenen gidilen mesafeler için sınırları bulduktan sonra, algoritmamızın mesafe kompetitif oranını hesaplamaya hazırız.

Proof of Theorem 28. Beklenen gidilen mesafe Lemma 29, 30 ve 31'daki sınırların toplamıdır. Kompetitif oranı elde etmek için, bu toplamı $d/2$ ile bölüyoruz, bu aşağıdaki sonucu verir

$$\begin{aligned} &\frac{d\pi}{2R} \left(\frac{r^2}{r^2-1} + \frac{r^2}{2(2-r^2)} + \frac{1}{2} \right) + \\ &\left(\frac{2(1+\pi)r}{r-1} + \frac{(1+\pi)r}{(2-r)} + \frac{1}{2} + \pi \right) + \\ &\frac{R}{d} \left(2(1+\pi) \log_r d + 4\pi + \frac{7}{2} \right) \end{aligned}$$

$\Theta(d)$ büyük d için bu ifadeyi domine eder, bu yüzden d katsayısını en küçüğe indirecek r değerini seçiyoruz. En iyi seçim $r = \sqrt{1.5} \approx 1.225$ 'dir. Bu seçim teoremden belirtilen mesafe kompetitif oranı verir.

Zaman kompetitif oran için analiz mesafe hesaplamalarına yakındır. Her roundda, robot $1/2$ olasılıkla inaktiftir. Bu durumda, robotun round i 'daki bekleme zamanı aktif robotun o roundu tamamlaması için geçen süre ile aynıdır (çünkü robotlar birim zamanda birim mesafe ilerlerler). Bu nedenle, her üç lemmadaki sınırları ikiye katlamak beklenen zaman için sınır verir. Tabiki, en iyi $r = \sqrt{1.5} \approx 1.225$ zaman kompetitif oran için de aynıdır.

5.2.3 $\mathcal{SRP}$ Algoritmasının Beklenen Round Sayısı

$\mathcal{SRP}$ Algoritması randevu gerçekleşene kadar devam eder. Lemma 32'de, randevu gerçekleşene kadar çalıştırılmasını gereken round sayısının beklenen değerine alt ve üst sınır buluyoruz.

Lemma 32. *Algoritma $\mathcal{SRP}$ 'deki beklenen round sayısı k ile alt sınırlı ve $k + 2$ ile üst sınırlıdır.*

Kanıt. $\mathcal{SR}$ 'in analizinden round $i \leq k$ iken, robotların buluşmadıklarını ve ilk olarak round $i = k + 1$ iken buluşma olasılıklarının olduğunu biliyoruz. Bu nedenle, randevunun gerçekleşmesi için en az k round geçmesi gerekir.

Beklenen round sayısına üst sınır bulmak için robotların Stage-2'deki beklenen round sayısını hesaplıyoruz.

$rnds$ randevunun gerçekleşmesi için Stage-2'de çalıştırılan round sayısı olsun. Stage-2'nin $i \geq k + 1$ olan roundlarını içerdiğini ve bu aşamada robotların her zaman $1/2$ olasılıkla buluştuklarını hatırlayalım. Bir başka deyişle, $\mathbb{P}[S_{i \leq k+1}] = \mathbb{P}[\overline{S}_{i \geq k+1}] = 1/2$ 'dir. $\mathbb{E}[rnds]$ 'i aşağıdaki gibi hesaplıyoruz. Bu, $1/2$ parametrelili geometrik dağılımın beklenen değeridir.

$$\begin{aligned} \mathbb{E}[rnds] &= \sum_{i=k+1}^{\infty} i \mathbb{P}[S_i] \mathbb{P}[\overline{S}_i]^{i-1} \\ &\leq \sum_{i=0}^{\infty} i \left(\frac{1}{2}\right)^i \\ &\leq 2 \end{aligned}$$

Bu nedenle, $\mathcal{SRP}$ 'daki beklenen round sayısına üst sınır $k + 2$ 'dir. $\square$

Algoritma $\mathcal{SRP}$ 'nin beklenen tekrar sayısı bu algoritma için gerekli olan randevu zamanı ile üst sınırlıdır. Randevu zamanı Bölüm 5.2.2'de analiz edilmiştir.

5.2.4 Düzlemsel Ortamlarda Çok Robotlu Simetrik Randevu Arama

Şimdiye kadar simetrik randevu arama problemini düzlemsel ortamlarda iki robot ile çalıştık. Engelli ve engelsiz ortamlar için sunduğumuz simetrik randevu stratejileri $\mathcal{SRP}$ ve $\mathcal{SRP}_o$, küçük değişikliklerle çoklu-robotlu ortamlara da genişletilebilir. Değişiklik yapılan stratejide, robot round i 'da artık aynı olasılıkla hareket etmez ve beklemez. Bunun yerine her roundun başında, tura gelme olasılığı $1/n$, yazı gelme olasılığı $n - 1/n$ olan hileli bir para atar. Eğer tura gelirse robotlar hareket ederler, aksi halde robotlar beklerler. Burada, n ortamdaki robot sayısını gösterir.

Robotların arama başında aynı yönü seçtiklerini ve $\mathcal{MSRP}$ ve $\mathcal{MSRP}_o$ stratejilerini senkron olarak çalıştırdıklarını düşünüyoruz. Bu varsayımlar, o roundda hareketli olan herhangi iki robotun aramadan önce R içinde yer almıyorsa, buluşamayacakları sonucunu doğurur. Bu nedenle, randevu sadece bir robot hareketli diğer robot beklerken ve robotlar R içindeyken gerçekleşir. Bu yaklaşım robot tarafından beklenen gidilen mesafeye bir üst sınır verir. Eğer bir grup robot ($< n$) round i 'da R içinde olsa bile, aramaya ayrı devam ederler. Sonraki bölümde, ikiden çok robot için genişlettiğimiz simetrik stratejimizin analizini sunuyoruz.

Analiz

Bu bölümde, algoritmamızın performansını analiz ediyoruz ve robot tarafından beklenen gidilen mesafeye üst sınır buluyoruz. Randevunun sadece bir robotun aktif diğer robotların ($n - 1$ robot) beklediği ve hareketli robotun diğer tüm robotlarla buluşacak kadar ilerlediği durumda gerçekleştiğini düşünüyoruz.

Şimdi çoklu-robot durumu için en iyi offline çözüme karar veriyoruz. C , bir A ortamındaki tüm robotları içine alacak mümkün olan en küçük yarıçaplı daire olsun. C 'nin çapı A ortamındaki birbirine en uzak iki robot tarafından

belirlenir. d , bu robotlar arasındaki uzaklık olsun. O zaman, C 'nin yarıçapı $d/2$ olur. Randevu olma durumuna yaklaşımımıza bağlı olarak, robotlar C içerisindeki herhangi bir $p \in R$ noktasında buluşacaklardır. Randevu zamanı (gidilen mesafeye göre) p 'e en uzak robot tarafından belirlenir. Bu robotun konumu C 'nin çapının uç noktalarından biri üzerindedir. o , C 'nin merkezi olsun. Robotlar $p = o$ noktasında buluşurlarsa en az mesafe ilerler. Bu durumda $p = o$ 'a en uzak robot tarafından bu noktaya gidilen mesafe $d/2$ olacaktır. Bu nedenle algoritmamızın mesafe kompetitif performansını $d/2$ 'e göre karşılaştıracğız.

Çoklu-robot stratejimizin çalışmasını iki aşamaya bölüyoruz: round $i \leq k$ ve round $i \geq (k + 1)$. Stage-1, robotun diğer robotlarla buluşabilecek kadar ilerlemediği daha önceki roundları kapsar. Stage-2'de, randevu davranışı tutarlıdır, bu yüzden beklenen gidilen mesafeyi sonsuz toplam kullanarak hesaplayabiliriz.

Simdi analizimizdeki değişkenleri tanıtıyoruz. S_i ve A_i olaylarını Bölüm 5.2.2'deki tanımları ile kullanıyoruz. Fakat, çoklu-robot genişletmesinde, $\mathbb{P}[A_i] = 1/n$ 'dir. Yeni bir değişken olarak, M_i sadece bir robotun round i 'da aktif olma olayı olsun. Robotun round i 'da $1/n$ olasılıkla hareket ettiğini veya $(n - 1)/n$ olasılıkla beklediğini biliyoruz. Bu nedenle M_i olayının olasılığı,

$$\begin{aligned}\mathbb{P}[M_i] &= n \left(\frac{1}{n} \right) \cdot \left(\frac{n-1}{n} \right)^{n-1} = \left(\frac{n-1}{n} \right)^{n-1} \\ \mathbb{P}[\overline{M}_i] &= 1 - \mathbb{P}[M_i] = \frac{n^{n-1} - (n-1)^{n-1}}{n^{n-1}}\end{aligned}\tag{5.3}$$

R_i olayının, algoritmanın round i 'da hala aktif olma olayı olduğunu hatırlayalım. Bu, $i > 0$ için $\mathbb{P}[R_i] = \mathbb{P}[\overline{S}_0 \wedge \dots \wedge \overline{S}_{i-1}]$ 'dir. Robotlar sadece M_i olayı olduğunda buluştuğundan, $\mathbb{P}[R_i]$,

$$\mathbb{P}[R_i] = \begin{cases} 1 & 0 \leq i \leq k, \\ \mathbb{P}[\overline{M}_i]^{i-k} & k < i. \end{cases}\tag{5.4}$$

Sonraki bölümde, sadece engelsiz düzlemsel ortamlar için robot tarafından beklenen gidilen mesafenin analizini sunuyoruz. Mesafe hesaplamaları engelli ortamlar için aynıdır. Engelsiz düzlemsel ortamlar için algoritmanın mesafe kompetitif oranının aşağıdaki gibi sınırlandırıldığını kanıtıyoruz.

Theorem 33. *Genişleme yarıçapı için en iyi seçim $r = 1.14$ 'dir. Bu seçim $O(\frac{d}{\log(n)R})$ mesafe kompetitif oranını verir. Zaman kompetitif oran bu sınırın n katıdır.*

Stage-1 Analizi Bu bölümde, round $i \leq k$ olduğu zaman robot tarafından beklenen gidilen mesafeyi hesaplıyoruz. Hareketli robotun bu aşamada yer değişimi en fazla $r^k < r^{k+\delta}$ olduğundan tüm robotlar ile buluşamaz. Simdi bu aşamadaki beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 34. *Stage-1’de robot tarafından beklenen gidilen mesafe şununla sınırlandırılır*

$$\sum_{i=1}^k \mathbb{E}[D_i] < d^2 \frac{1}{n} \left(\frac{\pi}{2R} \right) \frac{r^2}{r^2 - 1} + \frac{2(1 + \pi)}{n} \left(d \cdot \frac{r}{r - 1} + (1 + \log_r d)R \right).$$

İspat. Robotlar buluşabilecek kadar ilerlemediklerinden, k ’ya kadar her round yapılır. Bir başka deyişle, $0 \leq i \leq k$ için $\mathbb{P}[R_i]=1$ ’dir. Eğer round i ’da hareket ederse, robotun Algorithm 5’i çalıştırdığını biliyoruz. Bu nedenle, round i ’da beklenen gidilen mesafe Denklem (5.2)’deki gibidir. Bu denklemi $\mathbb{E}[D_i|R_i]$ yerine kullanarak, bu aşamadaki beklenen gidilen mesafenin aşağıdaki gibi olduğunu buluyoruz

$$\begin{aligned} \sum_{i=0}^k \mathbb{E}[D_i] &= \sum_{i=0}^k \mathbb{E}[D_i|R_i]\mathbb{P}[R_i] \\ &= \sum_{i=0}^k \mathbb{E}[D_i|A_i]\mathbb{P}[A_i] = \frac{1}{n} \sum_{i=0}^k \mathbb{E}[D_i|A_i] \\ &= \frac{1}{n} \sum_{i=0}^k \frac{\pi}{2R} r^{2i} + 2(1 + \pi)(r^i + R) \\ &< \frac{\pi}{2nR} \sum_{i=0}^k r^{2i} + \frac{2(1 + \pi)}{n} \left(\sum_{i=0}^k r^i + (k + 1)R \right) \\ &= \frac{\pi}{2nR} \cdot \frac{r^{2k+2} - 1}{r^2 - 1} + \frac{2(1 + \pi)}{n} \left(\frac{r^{k+1} - 1}{r - 1} + (k + 1)R \right) \\ &< d^2 \left(\frac{\pi}{2nR} \right) \frac{r^{2-2\delta}}{r^2 - 1} + \frac{2(1 + \pi)}{n} \left(d \frac{r^{1-\delta}}{r - 1} + (1 + \log_r d)R \right), \end{aligned} \quad (5.5)$$

Bu $\delta = 0$ için en büyük olur.

Stage-2 Analizi

Bu bölümde, $i \geq (k + 1)$ olan roundları düşünüyoruz. Stage-1’den farklı olarak, robotlar sıfırdan farklı olasılıkla buluşurlar. Şimdi Stage-2’de beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 35. *Stage-2’de beklenen gidilen mesafe aşağıdaki ile sınırlıdır*

$$\begin{aligned} \sum_{i=k+1}^{\infty} \mathbb{E}[D_i \mid \bar{S}_i] &< \frac{\mathbb{P}[\bar{M}_i] \pi r^2 d^2}{2nR} \left(1 + \frac{\mathbb{P}[\bar{M}_i]}{1 - r^2 \mathbb{P}[\bar{M}_i]} \right) + \\ &\quad \frac{\mathbb{P}[\bar{M}_i] 2(1 + \pi)d}{n} \left(1 + \frac{r \mathbb{P}[\bar{M}_i]}{1 - r \mathbb{P}[\bar{M}_i]} \right) + \\ &\quad \frac{\mathbb{P}[\bar{M}_i] 2(1 + \pi)R}{n} \left(1 + \frac{\mathbb{P}[\bar{M}_i]}{\mathbb{P}[M_i]} \right). \end{aligned}$$

İspat. Robot tarafından beklenen gidilen mesafe aşağıdaki ile sınırlıdır

$$\begin{aligned} &\sum_{i=k+1}^{\infty} (\mathbb{E}[D_i \mid S_i] \mathbb{P}[S_i] + \mathbb{E}[D_i \mid \bar{S}_i] \mathbb{P}[\bar{S}_i]) \mathbb{P}[R_i] \\ &= \sum_{i=k+1}^{\infty} \mathbb{E}[D_i \mid S_i \wedge R_i] \mathbb{P}[S_i \wedge R_i] + \\ &\quad \mathbb{E}[D_i \mid \bar{S}_i \wedge R_i] \mathbb{P}[\bar{S}_i \wedge R_i] \end{aligned} \tag{5.6}$$

M_i ’in round i ’da sadece bir robotun aktif olma olayı olduğunu hatırlayalım. M_i olasılığı Denklem (5.3)’teki gibi verilir. Algoritmanın round i ’da aktif olduğunu varsayarak, robotlar sadece M olayı olduğunda buluşurlar. Dahası, robot tarafından beklenen gidilen mesafe sadece A_i olayı olduğunda sıfırdan farklıdır. Bu $\mathbb{E}[D_i \mid \bar{S}_i \wedge R_i] = \mathbb{E}[D_i \mid \bar{M}_i \wedge A_i \wedge R_i]$ ve $\mathbb{E}[D_i \mid S_i \wedge R_i] = \mathbb{E}[D_i \mid M_i \wedge A_i \wedge R_i]$ olduğu anlamına gelir. Bu değerleri Denklem 5.6’de kullanarak,

Stage-2'de beklenen gidilen mesafeyi aşağıdaki gibi buluyoruz

$$\begin{aligned}
& \sum_{i=k+1}^{\infty} \mathbb{E}[D_i \mid M_i \wedge A_i \wedge R_i] \mathbb{P}[M_i \wedge A_i \wedge R_i] + \\
& \quad \mathbb{E}[D_i \mid \overline{M}_i \wedge A_i \wedge R_i] \mathbb{P}[\overline{M}_i \wedge A_i \wedge R_i] \\
&= \frac{\mathbb{P}[M_i]}{n} \sum_{i=k+1}^{\infty} \left(\frac{\pi}{2R} r^{2(k+1)} + 2(1+\pi)(r^{k+1} + R) \right) \mathbb{P}[\overline{M}_i]^{i-k} + \\
& \quad \frac{\mathbb{P}[\overline{M}_i]}{n} \sum_{i=k+1}^{\infty} \left(\frac{\pi}{2R} r^{2i} + 2(1+\pi)(r^i + R) \right) \mathbb{P}[\overline{M}_i]^{i-k} \\
& i \leftarrow i + k + 1 \\
&= \frac{\mathbb{P}[M_i] \mathbb{P}[\overline{M}_i]}{n} \left(\frac{\pi}{2R} r^{2(k+1)} + 2(1+\pi)(r^{k+1} + R) \right) \sum_{i=0}^{\infty} \mathbb{P}[\overline{M}_i]^i + \\
& \quad \frac{\pi r^{2(k+1)} \mathbb{P}[\overline{M}_i]^2}{2nR} \sum_{i=0}^{\infty} (r^2 \mathbb{P}[\overline{M}_i])^i + \\
& \quad \frac{2(1+\pi) \mathbb{P}[\overline{M}_i]^2}{n} \left(r^{k+1} \sum_{i=0}^{\infty} (r \mathbb{P}[\overline{M}_i])^i + R \sum_{i=0}^{\infty} \mathbb{P}[\overline{M}_i]^i \right) \\
&= \frac{\mathbb{P}[\overline{M}_i] \pi r^2 d^{2-\delta}}{2nR} \left(1 + \frac{\mathbb{P}[\overline{M}_i]}{1 - r^2 \mathbb{P}[\overline{M}_i]} \right) + \\
& \quad \frac{\mathbb{P}[\overline{M}_i] 2(1+\pi) d^{-\delta}}{n} \left(1 + \frac{r \mathbb{P}[\overline{M}_i]}{1 - r \mathbb{P}[\overline{M}_i]} \right) + \\
& \quad \frac{\mathbb{P}[\overline{M}_i] 2(1+\pi) R}{n} \left(1 + \frac{\mathbb{P}[\overline{M}_i]}{\mathbb{P}[M_i]} \right)
\end{aligned}$$

Bu da $\delta = 0$ için en büyük olur.

Mesafe Kompetitif Oranın Hesaplanması Her iki aşamada da beklenen gidilen mesafe için sınırları bulduktan sonra, algoritmamızın mesafe kompetitif oranını hesaplamaya hazırız.

Proof of Theorem 33. Beklenen gidilen mesafe Lemma 34 ve 35'deki sınırların toplamıdır. Kompetitif oranı elde etmek için, bu toplamı $d/2$ 'e bölüyoruz. $\Theta(d)$ büyük d için çıkan ifadeyi domine eder. Domine eden ifade

şudur

$$\begin{aligned}
& \frac{d\pi}{nR} r^2 \left(\frac{1}{r^2 - 1} + \mathbb{P}[\overline{M}_i] \left(1 + \frac{\mathbb{P}[\overline{M}_i]}{1 - r^2 \mathbb{P}[\overline{M}_i]} \right) \right) \\
& \frac{4(1 + \pi)}{n} \left(\mathbb{P}[\overline{M}_i] \left(1 + \frac{r \mathbb{P}[\overline{M}_i]}{1 - r \mathbb{P}[\overline{M}_i]} \right) + \frac{r}{r - 1} \right) + \\
& \frac{4R(1 + \pi)}{dn \log(r)} + \\
& \frac{4(1 + \pi)R}{dn} \left(\mathbb{P}[\overline{M}_i] \left(1 + \frac{\mathbb{P}[\overline{M}_i]}{\mathbb{P}[M_i]} \right) + 1 \right)
\end{aligned} \tag{5.7}$$

Bu nedenle, d katsayısının en küçük hale getirecek r değerini seçiyoruz. En iyi değer, $r = 1.14$ 'dir. Bu da $O\left(\frac{d}{\log(n)R}\right)$ mesafe kompetitif oranını verir.

Zaman kompetitif oranı için olan analiz, mesafe hesaplamaları ile aynıdır. Zaman kompetitif oranı, mesafe kompetitif oranının n katıdır ve optimal r değeri zaman kompetitif oran için de aynıdır.

5.2.5 Simülasyonlar

Bu bölümde, Algoritma $\mathcal{SRP}$ 'ın ve $\mathcal{MSRP}$ 'ın performanslarını simülasyonlarla araştırıyoruz. Simülasyon sonuçlarının Teorem 28'de (iki robotlu durum) ve Teorem 33'de (ikiden çok robotlu durum) elde ettiğimiz mesafe kompetitif oranlara uyumlu olduğunu gösteriyoruz.

İki Robotlu Durum

Bu bölümde, $\mathcal{SRP}$ için olan simülasyon sonuçlarını sunuyoruz. İlk önce $\mathcal{SRP}$ 'ın performansını r ve başlangıç uzaklığının bir fonksiyonu olarak araştırıyoruz. Şekil 5.2'deki grafikler, çeşitli r değerleri ve başlangıç uzaklıkları için olan simülasyonların sonuçlarını raporlar, her birinin 5000 deneme yapılarak ortalaması alınmıştır. Robotlar arasındaki başlangıç uzaklığı d 'dir. Stratejimizi $5 \leq d \leq 1025$ uzaklıkları için simüle ediyoruz. Robot birim adım büyüklüğünde hıza sahiptir. r 'in değeri 1.125'den başlatılır ve 1.725'e kadar 0.1 olarak arttırılır.

Şekil 5.2'deki en üstteki plot, aynı r değerlerinin kümesi için ortalama mesafe kompetitif oranı başlangıç uzaklığının bir fonksiyonu olarak gösterir.

Kırmızı koyu çizgi $\mathcal{SRP}$ 'ın performansını $r = 1.225$ için simülasyonlarla gösterir. Bu simülasyonlarda, $r = 1.225$ 'un çoğu başlangıç değerleri için daha üstün bir performansa sahip olduğunu gözlüyoruz. Aslında, $\approx 10d$ olan ortalama mesafe kompetitif oranı en yüksek $r = 1.725$ ve en düşük $r = 1.225$ olduğu zamandır.

En alttaki plot, bu r değerleri için ortalama round sayısındaki değişimi başlangıç uzaklığının bir fonksiyonu olarak gösterir. Beklendiği gibi, simülasyon sonuçları ortalama round sayısının r 'in değeri düştükçe ve başlangıç uzaklığı arttıkça artacağını gösterir. Aslında, ortalama round sayısı en yüksek $r = 1.125$ ve en düşük $r = 1.725$ olduğu zamandır.

Bundan sonra, iletişim alanı R 'daki değişimin Algoritma $\mathcal{SRP}$ 'ın performansı üzerindeki etkilerini araştırıyoruz. Şekil 5.3'deki grafikler, çeşitli R değerleri ve başlangıç değerleri için simülasyon sonuçlarını raporlar, her birinin 5000 deneme yapılarak ortalaması alınmıştır. R , 1 ve 8 adım büyüklüğü arasında değiştirilmiştir. Grafikler algoritmanın mesafe ve zaman kompetitif performanslarını gösterir. Grafiklerden ortalama kompetitif oranın R arttıkça azalacağını gözlüyoruz.

Şekil 5.4'de, robot daire çizerken $\mathcal{SRP}$ 'ın performansını üç farklı formülasyonda araştırıyoruz: (1) Robotlar aramanın başında aynı yönü seçerler (saat yönünde/ters saat yönünde) ve randevu gerçekleşene kadar aynı yönü kullanmaya devam ederler. (2) Robot aramanın başında rasgele bir yön seçer ve randevu gerçekleşene kadar bu yönü kullanmaya devam eder. (3) Robot her round başında rasgele bir yön seçer. Tüm simülasyonlarda, robotların başlangıç rotasyonlarının aynı olduğunu varsayıyoruz. Simülasyon sonuçları her round başında yönü rasgeleleştirmenin, arama boyunca aynı yöne bağlı kalmaya göre kompetitif oranı iyileştirdiğini gösterir. $\mathcal{SRP}$ 'nin analizini ilk formülasyona göre yapıyoruz. Bu nedenle, teorik analizde elde ettiğimiz kompetitif oran $\mathcal{SRP}$ 'nin performansına bir üst sınır verir.

Bu bölümde çalıştığımız problemlerde robotların randevu stratejilerini aynı zamanda çalıştırmaya başladıklarını varsayıyoruz. Bu senkron başlama, robotların pratikte her zaman mümkün olmayabilen global kontrolünü gerektirir. Bu bölümde, $\mathcal{SRP}$ 'ın performansını ayrıca bu varsayımın ihlal edildiği durum için de çalışıyoruz. Simülasyonlarda robot-2'in robot-1'den t zaman birimi daha geç başladığını düşünüyoruz. Şekil 5.5, $5 \leq d \leq 1025$ başlangıç uzaklıkları ve $2 \leq t \leq 4000$ zaman gecikmeleri için ortalama mesafe kom-

petitif oranı gösterir. Bu simülasyon sonuçları algoritmamızın senkron hatalar altında dayanıklı olduğunu gösterir. Zaman gecikmelerinin performans üzerinde çok az etkisi vardır. Daha uzun gecikmeleri aslında performansı iyileştirir. t arttıkça kompetitif oranda açıkça bir azalma eğilimi görüyoruz. Daha büyük başlangıç uzaklığı d için, kompetitif oran $\approx 7d$ 'dir.

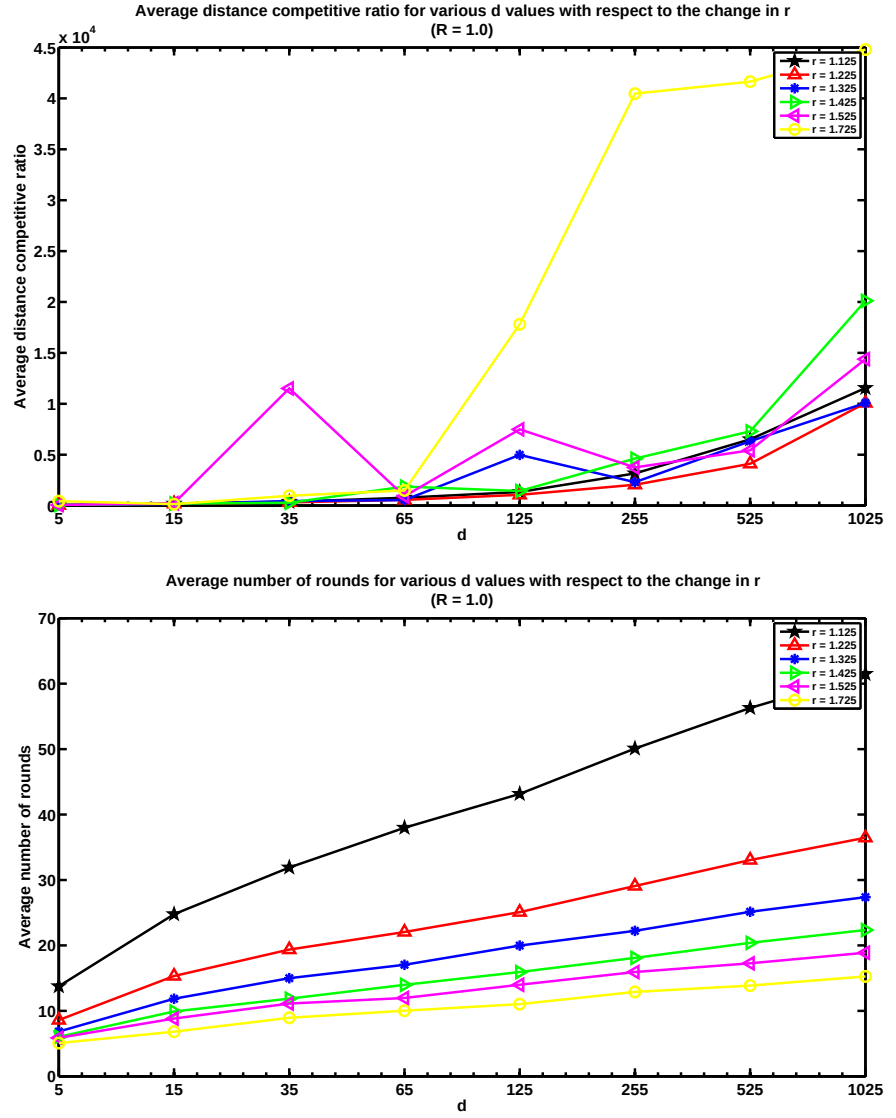
İkiden çok Robotlu Durum

Bu bölümde, $MSRP$ algoritması için olan simülasyon sonuçlarını sunuyoruz. İlk olarak SRP 'in performansını N ve başlangıç uzaklığının bir fonksiyonu olarak araştırıyoruz. Şekil 5.6'deki grafikler, $r = 1.14$ olduğu zaman çeşitli N değerleri ve başlangıç uzaklıkları için olan simülasyonların sonuçlarını raporlar, her birinin 5000 deneme yapılarak ortalaması alınmıştır. Ortamdaki en uzak iki robot arasındaki başlangıç uzaklığı d 'dir. Tüm simülasyonlarda, robotlar birbirinden eşit uzaklıkta konumlandırılmıştır. Stratejimizi $5 \leq d \leq 225$ uzaklıkları için simüle ediyoruz. Robot birim adım büyüklüğünde hızı sahiptir ve r 'in değeri 1.14'dir.

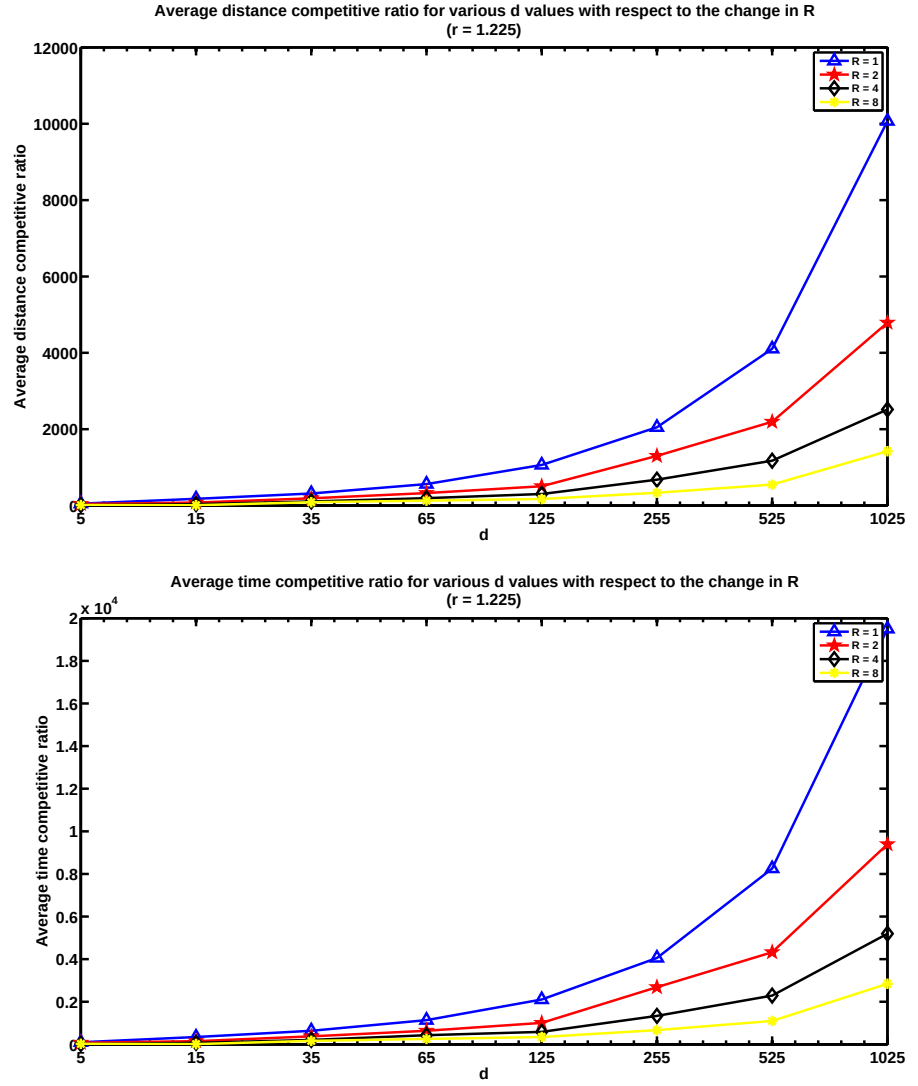
Şekil 5.7'deki grafikler, $r = 1.14$ olduğu zaman çeşitli N ve başlangıç uzaklığının bir fonksiyonu için olan simülasyonların sonuçlarını raporlar. Şekil 5.6'deki aynı başlangıç uzaklıkları kümesini kullanıyoruz. N , 4 ve 20 arasında değiştirilir. Başlangıç uzaklığı ve bu r değerlerinin kümesi için en üstteki grafik ortalama mesafe kompetitif oranı ve en alttaki grafik ortalama round sayısını gösterir. Bu grafiklerden ortalama mesafe kompetitif oranının ve round sayısının robot sayısı arttıkça azaldığını gözlemliyoruz.

Daha sonra, iletişim alanı R 'daki değişimin Algoritma $MSRP$ 'in performansı üzerindeki etkilerini araştırıyoruz. Şekil 5.8'deki grafikler iletişim alanları $1 \leq R \leq 16$ ve başlangıç uzaklıkları $5 \leq d \leq 225$ için simülasyonların sonuçlarını raporlar, her birinin 5000 deneme yapılarak ortalaması alınmıştır. Beklenildiği gibi, simülasyon sonuçları R arttıkça ortalama kompetitif oranının azaldığını gösterir.

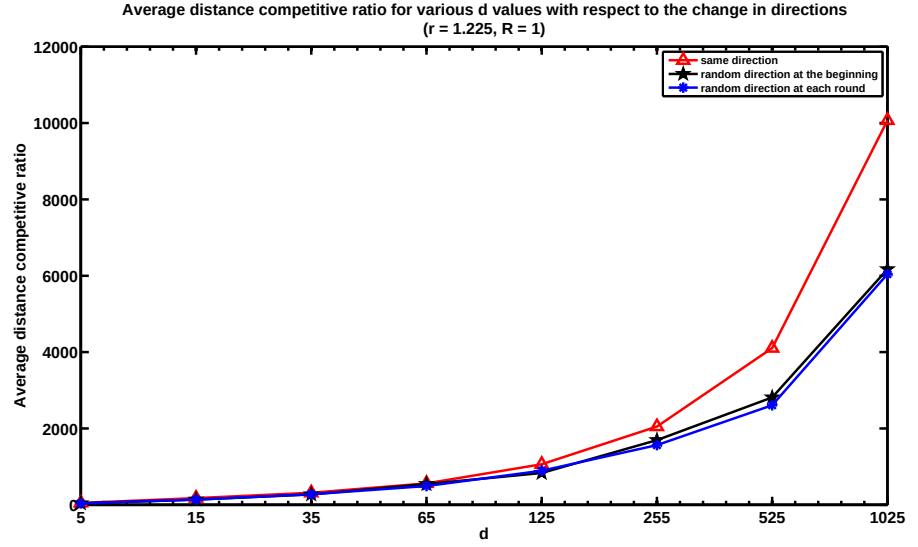
Son olarak, robotların sayısı ve başlangıç uzaklığının bir fonksiyonu olarak randevu gerçekleşene kadar oluşan küme sayısını araştırıyoruz. Robotlar birbirine eşit uzaklıkta olduğundan dolayı, oluşan küme sayısındaki değişim başlangıç uzaklığından çok robot sayısındaki değişimden etkilenmektedir.



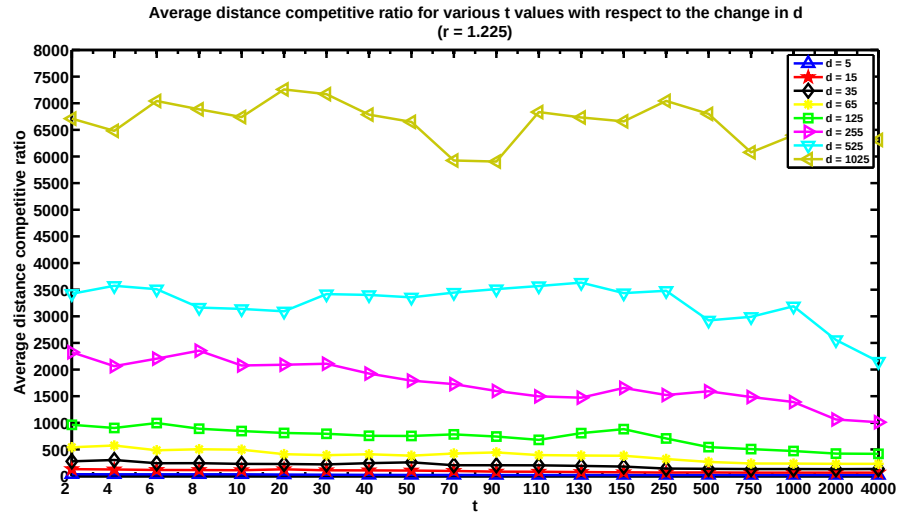
Şekil 5.2: Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$. **ÜST:** Genişleme yarıçapı r 'deki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. **ALT:** Genişleme yarıçapı r 'deki değişime göre çeşitli d değerleri için ortalama round sayısı.



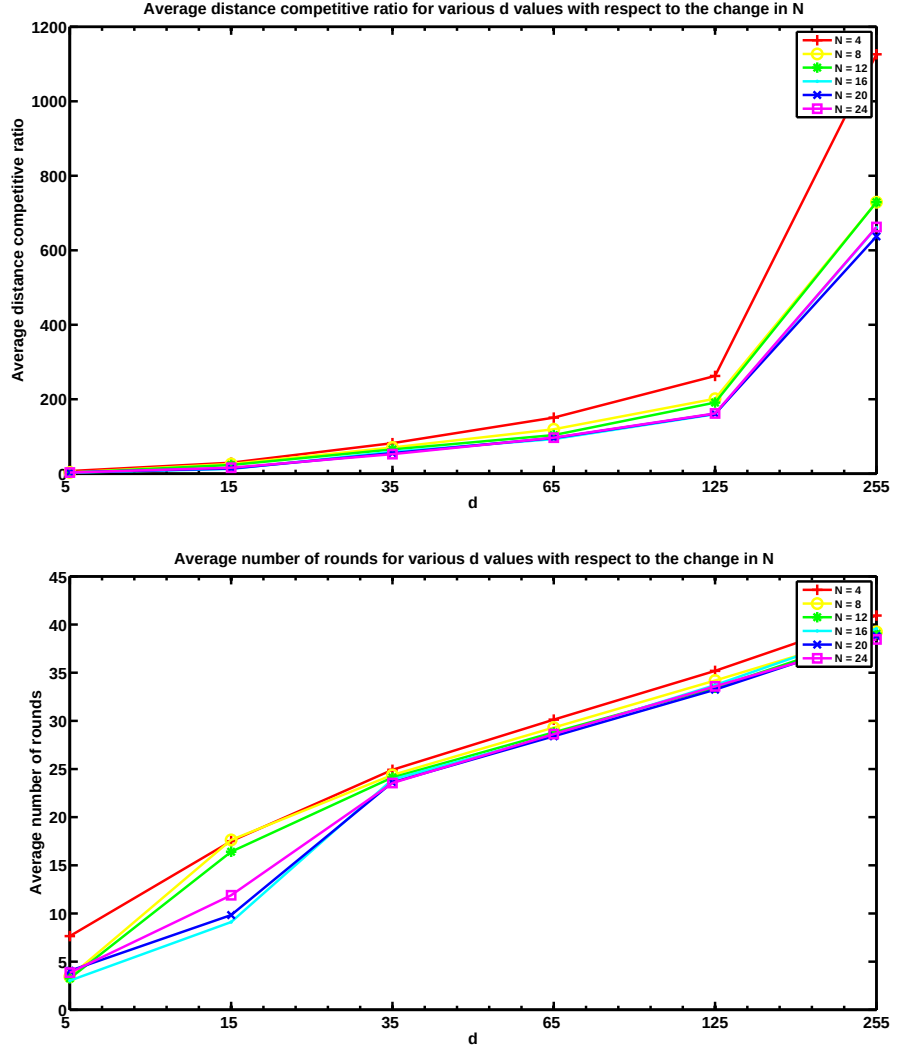
Şekil 5.3: Düzlemsel ortamlarda iki robotla simülasyonlar. $r = 1.225$. **ÜST:** Komünikasyon alanı R 'daki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. **ALT:** Komünikasyon alanı R 'daki değişime göre çeşitli d değerleri için ortalama zaman kompetitif oran.



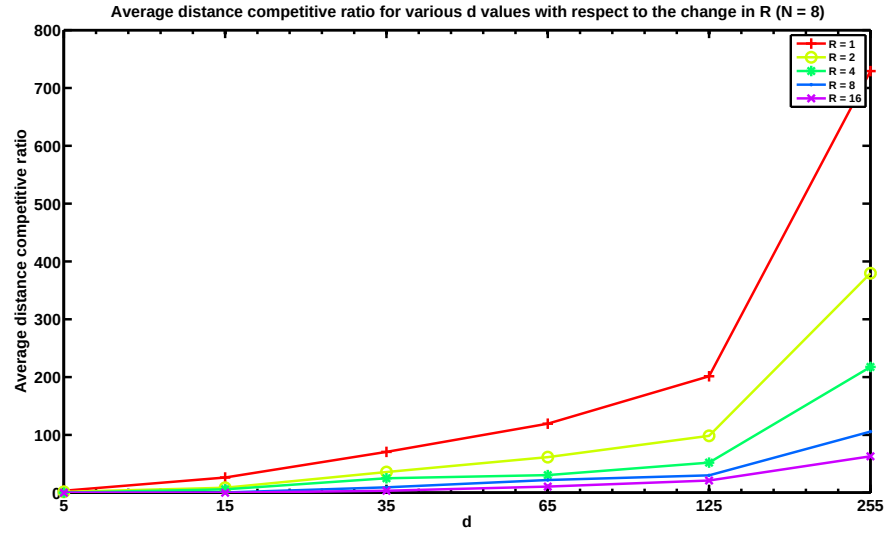
Şekil 5.4: Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$ and $r = 1.225$. Yöndeki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. Robotların oryantasyonlarının algoritma başında aynı olduğunu varsayıyoruz. **Aynı yön:** Robotlar bir roundda hareket etmeye karar verirse, her iki robot da aynı yönde içiçe daireler çizer. **Rasgele yön:** Robotlar içiçe daireleri ya saat yönünde yada ters saat yönünde çizer. (a) Robotlar randevu arama başında saat yönünde/ters saat yönünde hareket edeceklerine karar verirler. (b) Robotlar yönlerini her roundda rasgeleleştirirler.



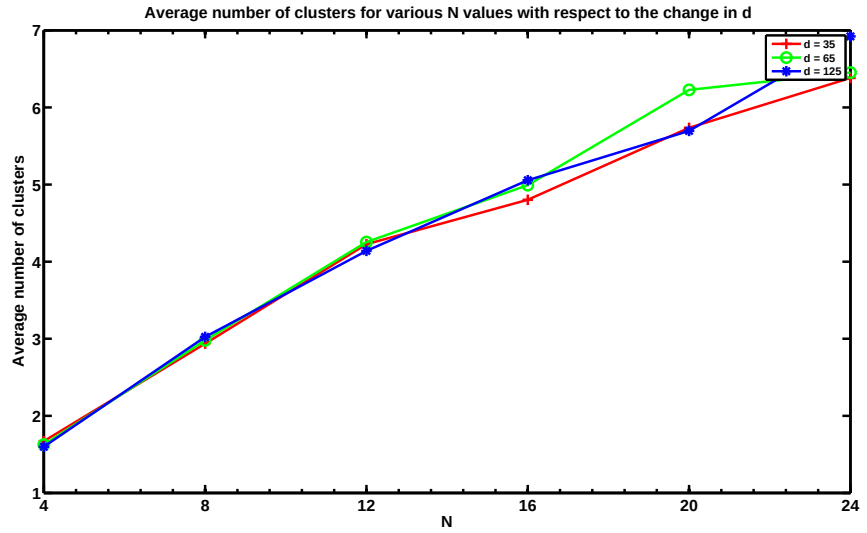
Şekil 5.5: Düzlemsel ortamlarda iki robotla simülasyonlar. $R = 1.0$ and $r = 1.225$. d 'deki değişime göre çeşitli t değerleri için ortalama mesafe kompetitif oran. Ortamdaki bir robot randevu algoritmasını diğer robottan t zaman sonra çalıştırmaya başlar.



Şekil 5.6: Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $R = 1.0$.
ÜST: N 'deki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran. **ALT:** N 'deki değişime göre çeşitli d değerleri için ortalama round sayısı.



Şekil 5.7: Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $N = 8$. R 'daki değişime göre çeşitli d değerleri için ortalama mesafe kompetitif oran.



Şekil 5.8: Düzlemsel ortamlarda ikiden çok robotla simülasyonlar. $R = 1.0$. d 'deki değişime göre çeşitli N değerleri için randevu gerçekleştiğinde oluşmuş olan küme sayısı.

Özetle, $\mathcal{SRP}$ 'ın simülasyon sonuçları Teorem 28'de elde ettiğimiz r 'ın optimal seçimi ve mesafe kompetitif oran ile uyumludur. Dahası, $\mathcal{SRP}$ ve $\mathcal{MSRP}$ simülasyonlarından iletişim alanı arttıkça ortalama mesafe kompetitif oranın azaldığını gözlemliyoruz. Ayrıca, ortalama mesafe kompetitif oran robot sayısı arttıkça azalır. Randevuya kadar aynı yöne bağlı kalmaktansa robotların yönlerini rasgeleştirmelerinin ortalama mesafe kompetitif oranı iyileştirdiğini gözlemliyoruz. Bu nedenle, algoritmamız bir üst sınır sağlar. Bunların dışında, Şekil 5.2 ve 5.5'deki simülasyon grafiklerini karşılaştırdığımızda, $\mathcal{SRP}$ 'ın zaman gecikmelerine karşı dayanıklı bir performans gösterdiğini görüyoruz.

5.3. Engelli Ortamlar

Bu bölümde, simetrik randevu problemini iki robot ile engellerin olduğu ortamlarda çalışıyoruz. Bu gibi ortamlarda daha önceki çalıştığımız (engelsiz düzlemsel ortamlar için tasarlanan) problemde sunulmuş olan strateji uygulanabilir değildir. Bunun nedeni robotun engellerden dolayı dairesel şekilde hareket ederek tüm ortamı kapsayamayacak olmasından kaynaklanır. Fakat, bu strateji, aynı problemi engellerin olduğu ortamlar için çözmek üzere genişletilebilir. Engelsiz durumda olduğu gibi, robotlar arasındaki başlangıç uzaklığı (en kısa yolun uzunluğu) $\delta \in (0, 1]$ ve $k \in \mathbb{Z}^+$ olmak üzere $d = r^{k+\delta}$ olarak verilir. Problemi formülize etmek için, şu varsayımları yapıyoruz.

1. Ortam bilinmiyor.
2. Ortamdaki engeller poligonal.
3. Robot translating bir robot (Bir başka deyişle dönmüyor).
4. Robotlar bir kenar uzunluğu D olan kare robotlardır ve engelleri tespit edecek sensörler ile donatılmışlardır.

Algoritmayı, kare bir robotun bir nokta robot olarak temsil edildiği, bir konfigürasyon uzayı C 'de (*Ing.* configuration-space (C)) sunuyoruz. C -obstacle'leri her engelin D ile Minkowski toplamını (*Ing.* Minkowski sum) alarak oluşturuyoruz. Daha sonra, her C -obstacle'in sınırlarını kaldırıyoruz. C -obstacle'leri açık küme (*Ing.* open set) olarak düşünmenin nedeni şudur. Workspace (W)'de, aralarında d uzaklık olan iki engelin olduğunu düşünelim. Bu durumda, robot bu iki engel arasından geçebilir. Bu engellerin Minkowski toplamını aldıktan sonra, aralarında hiç boşluk kalmayacaktır. Bu nedenle, bir nokta robot bu iki C -obstacle arasından geçemeyecektir. Bu problemin üstesinden gelmek için, her C -obstacle'in sınırlarını kaldırıyoruz. Sonraki bölümde, engelli ortamlar için olan simetrik randevu algoritmasını sunuyoruz.

5.3.1 Simetrik Randevu Arama Algoritması

SR algoritmasına benzer olarak, algoritma $i \in \mathbb{Z}^+$ olarak numaralandırılan roundlar halinde devam eder. Her roundun başında, robot o round için ha-

reket etmeye veya beklemeye karar vermek için bir para atar. Robotları, C konfigürasyon uzaylarında noktalar olarak sunduğumuzu tekrar hatırlatalım. Robotların rasgele davranışları şu şekildedir. *Eğer robot tura atarsa*, kendi başlangıç uzaklığı merkez olan r^i yarıçaplı bir disk çizer. Burada r robotun genişleme yarıçapıdır (*Ing.* expansion radius) (Teorem 44'de optimize edilecektir). $Disk_i$ round i 'daki r^i yarıçaplı disk olsun. Robot önce $Disk_i$ içindeki sürekli (*Ing.* continuous) alanı, D boyutlu hücrelere sahip olan bir grid ile kesikli hale getirir. Daha sonra, $Disk_i$ içindeki grid hücrelerini coverage algoritmasıyla kapsar (bir sonraki bölümde açıklıyoruz). *Eğer robot yazı atarsa*, robotun $Disk_i$ etmesi için geçecek maksimum zaman kadar bekler (bir sonraki bölümde, $Disk_i$ 'in kapsama zamanı için bir sınır buluyoruz). Randevu robotlar aynı hücre içinde olduğu zaman ve bir robot beklerken diğeri hareket ederken gerçekleşir. Randevu stratejisinin pseudocode'u Algoritma 6'de ve waitCoverage prosedürü Algoritma 7'de sunulmuştur. Sonraki bölümde, coverage algoritmasını açıklıyoruz.

Algorithm 6 $\mathcal{SRP}_o$: Engelli ortamlar için simetrik randevu arama algoritması.

Input: C -obstacle with removed boundary of the expanded obstacles.

```

1:  $r \leftarrow 1.2$ 
2:  $i \leftarrow 0$ 
3:  $coinFlip \leftarrow \text{random from } \{-1, 1\}$ 
4: while checkRendezvous()  $\neq$  true do
5:   if  $coinFlip = -1$  then
6:     Draw a disk with a radius  $r^i$ 
7:      $C_{current} \leftarrow$  cell the robot is initially located.
8:     createGrid( $C_{current}, r^i$ )
9:     CA( $Disk_i$ ) //cover disk
10:  else if  $coinFlip = 1$  then
11:    waitCoverage(i)
12:  end if
13:   $i \leftarrow i + 1$ 
14:   $coinFlip \leftarrow \text{random from } \{-1, 1\}$ 
15: end while

```

5.3.2 Coverage Algoritması

Robotlar coverage algoritmasını (CA) $Disk_i$ içindeki grid hücreleri içinde çalıştırır. Grid hücreleri engeller içermeyebilir (boş hücreler (*Ing.* free cells)) veya tamamen veya kısmen engeller tarafından işgal edilmiş olabilir (Şekil 5.9). CA aslında DFS (Depth First Search) algoritmasının değiştirilmiş bir versiyonudur ve ortamdaki her noktanın kapsandığını garantiler. Şimdi

CA 'in detaylarını sunuyoruz.

Algoritmamızda kullandığımız şu değişkenleri tanımlıyoruz: C_{center} robotun başlangıçta grid içinde bulunduğu hücredir; $C_{current}$ robotun şu anda içinde hareket ettiği hücredir; C_{parent} robotun $C_{current}$ 'a girdiği hücredir; C_{child}^j , $C_{current}$ 'in komşu hücrelerinden birisidir. İki hücre eğer ortak bir kenar içeriyorsa komşudur, Komşu hücreler aralarında paylaşılan engeller ile kesişebilir. Örneğin, Şekil 5.9'de Cell-A'nın komşuları (B, E, F and H) hücreleridir. Başlangıçta $C_{current}$, C_{center} olarak belirlenir.

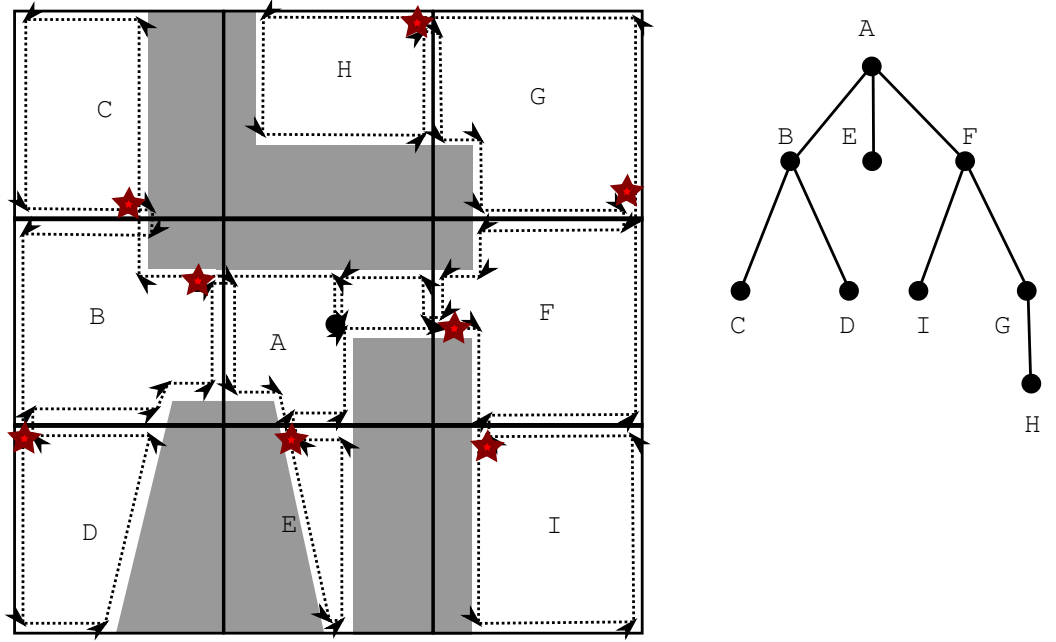
Robot, DFS ağacındaki her grid hücresini ters saat yönünde engel/hücre sınırlarını takip ederek kapsar. Bu hareket planına, raporun geri kalamında "CCWBF" (counter clockwise boundary following) olarak hitap edeceğiz. Bir hücreye sadece bir giriş olurken, birçok çıkış noktası olduğunu not edelim. e_j 'i, C_{child}^j 'in girişi olarak tanımlıyoruz (bu $C_{current}$ 'in çıkışına eşittir). Eğer $C_{current}$ C_{child}^j 'in sınırına çarparsa, C_{parent} $C_{current}$ olur ve C_{child}^j yeni $C_{current}$ olur. Robot daha sonra yeni $C_{current}$ 'i C_{child}^j 'in sınırına çarpma kadar CCWBF ile kapsar. Daha sonra C_{child}^j 'i ve tüm çocuklarını recursive olarak kapsar ve e_j 'e ulaşana kadar CCWBF'i takip eder. Robot e_j 'e geri geldiğinde, hücre kapsanmıştır deriz. Robot daha sonra C_{parent} 'a, C_{parent} 'in *exit*'den girer. Bu noktada, C_{parent} $C_{current}$ olur ve robot $C_{current}$ 'in girişine ulaşana kadar, $C_{current}$ 'i CCWBF yaparak kapsamaya devam eder. Grid içerisindeki tüm diğer hücreler benzer biçimde kapsanır.

Şekil 5.9, 3×3 bir grid üzerinde coverage algoritmasının çalışmasına bir örnek gösterir. Hücreler $A-I$ arasında numarandırılır ve robot başlangıçta Cell-A'dadır. Oklar, robotun bir hücre içinde hareket ettiği yönü gösterir ve yıldızlar, her hücrenin girişini gösterir. Sonuçtaki DFS ağacı robotun Şekil 5.9 (b)'deki grid hücrelerini hangi sırada ziyaret ettiğini sunar. Robot CCWBW'a başlamak için Cell-A içerisinde önce yukarı yöne doğru hareket eder. Cell-C'in hiç çocuğu olmadığından, robot Cell-C'i kapsamayı tamamladığında *entrance*'a geri gelir. Robot daha sonra, Cell-D'nin sınırına çarpma kadar Cell-C içerisinde CCWBF'e devam eder. Robot tarafından tüm diğer hücreler benzer biçimde kapsanır.

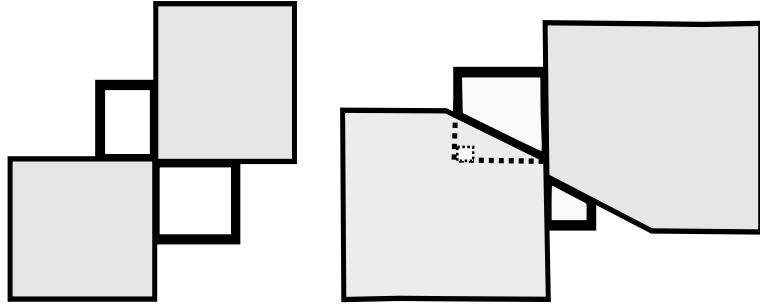
Procedure 7 waitCoverage(i)

Input: D

- 1: $N_i \leftarrow \pi r^{2i} / D^2$
 - 2: wait($8N_i D$)
-



Şekil 5.9: (a) Kapsama algoritmasının 3×3 grid üzerinde çalışmasına bir örnek. Grid hücreleri A-I olarak numaralandırılmıştır. (b) DFS Tree.



Şekil 5.10: Farklı *C-obstacles* ile iki bölgeye ayrılan hücre örnekleri.

5.3.3 Analiz

Engeller bir hücreyi birden fazla bölgeye ayırabilir (Şekil 5.10). Bir hücreyi tamamen kapsamak için robot hücre içerisindeki her bölgeyi kapsamalıdır. Lemma 38'de, *C-obstacles*'ların bir hücreyi en fazla iki bölgeye ayırabileceğini gösteriyoruz. Bu sınırı elde etmek için, bir sonraki bölümde *C-obstacles*'ın tüm köşelerini çalışıyoruz.

Bir *C-Obstacle*'ın mümkün olabilecek kenarları

Bir *P* engelini mümkün olabilecek köşelerini ve bir quadrantteki karşılık gelen *C-obstacle*'ını çalışıyoruz. Quadranın originini *P*'nin köşesine sabit-

liyoruz. E_i , P 'nin köşesine gelen ve E_o , P 'nin köşesinden çıkan kenar olsun. Şekil 5.11, bu kurulumu bir örnek gösterir. E_i ve E_o 'in aynı zamanda ters yönde etiketlendirilebileceğini not edelim. Zıt quadrantlar (I-III) ve (II-IV) olarak tanımlanır. C -obstacle kenarı D olan kare robotun şekli ile engellerin Minkowski toplamının alınarak oluşturulduğunu hatırlayalım. Minkowski toplamı alırken, engeli çevrilen robotun köşelerinden veya kenarlarından biriyle süpürüyoruz. Robotun kenarını sadece eğer engelin kenarı dik veya yatay bir kenar ise kullanıyoruz. Aksi halde, engeli süpermek için robotun köşelerinden birini kullanıyoruz. İlk olarak C -obstacle'lar için yapısal bir özellik sağlıyoruz.

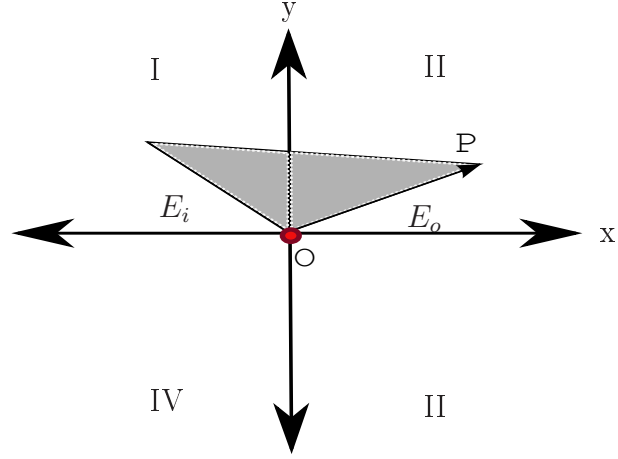
Proposition 36. u , C -obstacle C 'nin bir köşesi olsun ve ℓ u 'dan geçen dikey/yatay doğru olsun. $v \in \ell$, u 'ya en uzak nokta olsun, öyle ki $v \cap C \neq \emptyset$. u ve v arasındaki uzaklık D 'den daha büyük veya eşittir.

İspat. ℓ 'nin mümkün olabileceği iki durumu düşünelim.

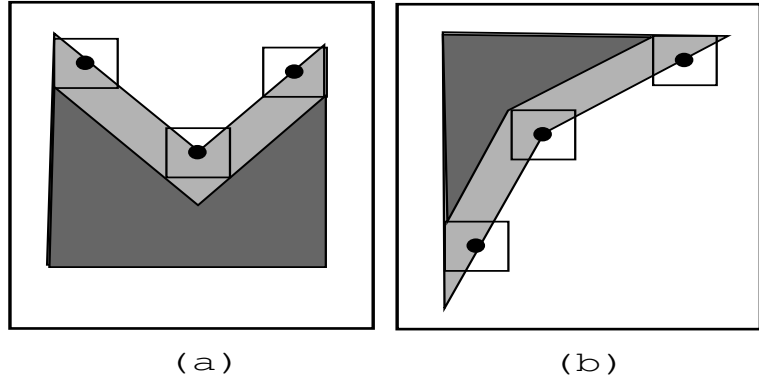
1. **Case-1.** Eğer ℓ engelin kendisinden geçiyorsa, o zaman engelin içinde olan ℓ 'nin bir parçası vardır. ℓ kare robot ile birlikte bu doğru parçasının Minkowski toplamına eşittir. Minkowski toplamı bu parçayı her iki ucundan en az $D/2$ kadar uzattığından, ℓ 'nin D 'ye eşit veya D 'den daha fazla olması gerekir.
2. **Case-2.** Eğer ℓ engelin kendinden geçmiyorsa, o zaman C -obstacle'in dikey/yatay bir kenarından geçiyor olmalıdır. Bu tip kenarlar eğer Şekil 5.13 (a) veya (c)'de gösterilen tiplerden biriye gerçekleşir. Bu robotun engelin köşesini iki farklı köşesi ile süpürmesinden dolayıdır, bu nedenle Minkowski toplamı, C -obstacle'da uzunluğu D olan yeni dikey/yatay bir kenar tanıtır.

Lemma 37. Bir C -obstacle'in köşesi her zaman 90 dereceden büyüktür.

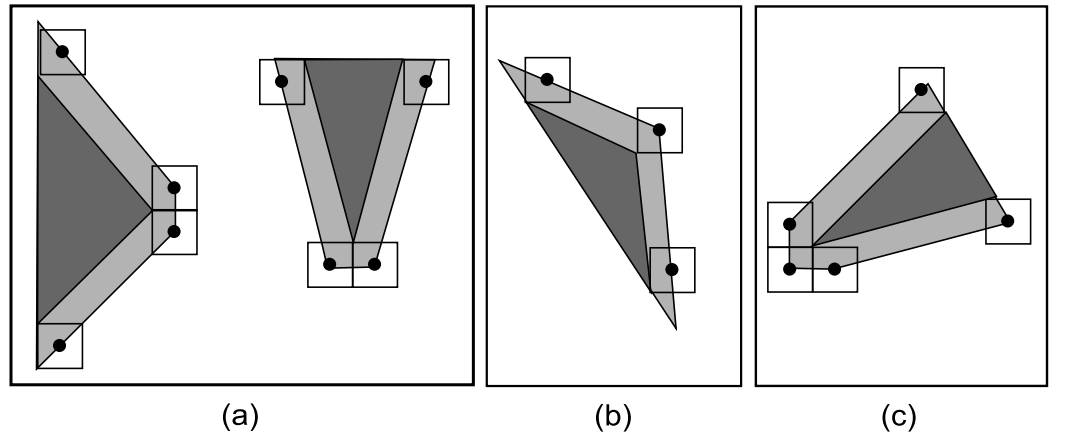
İspat. P 'nin köşesi convex (≤ 180 derece olan iç açıya sahiptir) veya konkavdır (> 180 olan iç açıya sahiptir). v , P 'nin konkav bir köşesi olsun. Önce v 'nin bir C -Obstacle köşesine nasıl eşlendiğini inceliyoruz. Eğer v 'nin E_i ve E_o 'i zıt quadrantlarda değilse, robot E_i ve E_o 'i iki farklı köşesi ile süpürür. Şekil 5.12 (a) bu durum için bir örnek gösterir. Eğer v 'nin E_i ve E_o 'i zıt quadrantlarda ise, o zaman robot E_i ve E_o 'i aynı köşesi ile süpürür. Bu duruma bir örnek Şekil 5.12 (b)'de gösterilmiştir. Her iki durumda da, v C -obstacle'in konkav bir köşesi olarak kalır.



Şekil 5.11: Bir quadrant örneği. Bölgeleri I-IV arasında numaralandırılmıştır. Quadrantın orijini P engelini köşesine sabitlenmiştir. E_i ve E_o sırasıyla P 'e gelen ve P 'den çıkan kenarları gösterir.



Şekil 5.12: P engelini konvex köşesi ve C -obstacle'da karşılık gelen köşesi. (a) Eğer v 'nin E_i ve E_o 'su zıt quadrantlarda değil ise. (b) Eğer v 'nin E_i ve E_o 'su zıt quadrantlarda ise.



Şekil 5.13: P engelini konvex köşesi ve onun C -obstacle'da karşılık gelen köşesi. (a) Eğer E_i ve E_o farklı fakat zıt quadrantlarda değilse. (b) Eğer E_i ve E_o zıt quadrantlarda ise. (c) Eğer E_i ve E_o aynı quadrantlarda ise.

u P 'nin konveks bir köşesi olsun. İkinci olarak, u 'nun bir C -Obstacle köşesine nasıl eşlendiğini inceliyoruz. İkinci durumu E_i ve E_o 'ın aynı veya farklı quadrantlarda olmasına göre üç alt duruma bölüyoruz.

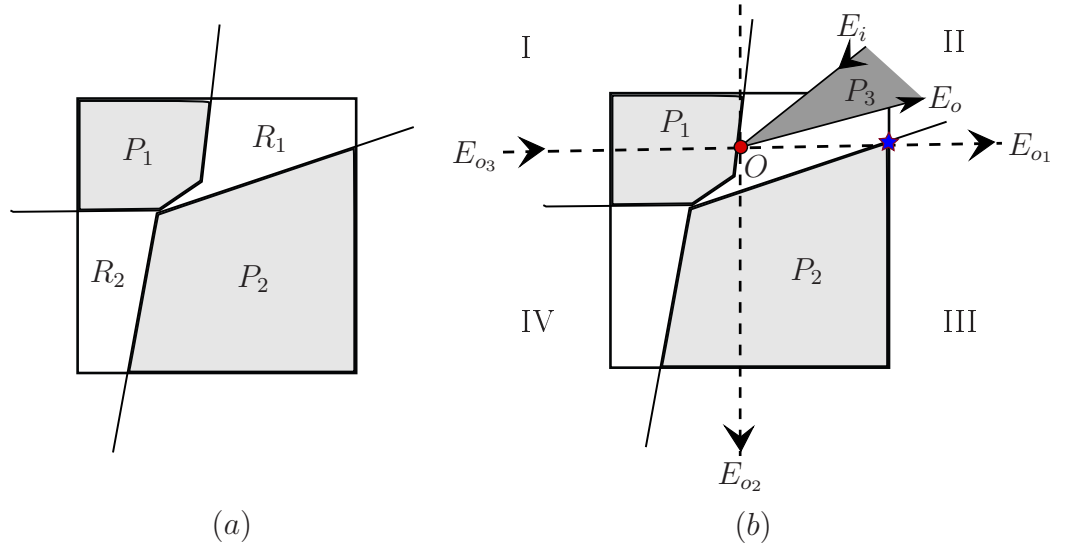
1. **Subcase-1 (farklı fakat zıt quadrantlarda değil):** Eğer E_i ve E_o farklı quadrantlarda ise fakat zıt quadrantlarda değilse, o zaman Minkowski toplamı C -obstacle'a yeni bir kenar tanıtır. Bu kenar yatay veya dikeydir ve D uzunluğuna sahiptir. Bu nedenle, bu durumda u , C -obstacle'da iki geniş açılı (> 90) köşeye karşılık gelir. Şekil 5.13 (a) bu durum için iki örnek gösterir.
2. **Subcase-2 (zıt quadrantlar):** Eğer E_i ve E_o zıt quadrantlarda ise (yani u geniş açılıdır), o zaman robot E_i ve E_o 'i aynı köşesi ile süpürür. Bu nedenle, v 'nin karşılık gelen C -obstacle köşesi hala geniş açılıdır. Şekil 5.13 (b) bu durum için bir örnektir.
3. **Subcase-3 (aynı quadrantlar):** Eğer E_i ve E_o aynı quadrantlarda ise, Minkowski toplamı C -obstacle'a iki yeni dikey ve yatay kenar tanıtır. Bu geniş açılı takiben dik bir köşeye ve bunu takiben de bir başka geniş açılı köşeye sonuçlanır. Yatay ve dikey kenarlar D uzunluğuna sahiptir. Şekil 5.13 (c) E_i ve E_o 'in aynı quadrantta olduğu duruma bir örnek gösterir.

Coverage Algoritmasının Analizi

Bu bölümde, coverage algoritmasının $Disk_i$ 'in tamamen coverage'ini garantilediğini ispatlıyoruz. Ayrıca robotun gittiği mesafeye bağlı olarak üst sınır buluyoruz. Bir hücreyi kapsama maliyetini hesaplamak için Lemma 38'de, bir hücrenin C -obstacle'lar tarafından bölünebileceği maksimum bölge sayısını analiz ediyoruz.

Lemma 38. *Bir C -obstacle bir grid hücrelerini en fazla iki bölgeye ayırabilir.*

İspat. P_1 ve P_2 bir hücreyi R_1 ve R_2 (Şekil 5.14 (a)) olarak iki bölgeye ayıran C -obstacle'lar olsun. Üçüncü bir bölge yaratmak için bir C -obstacle'in gelen bir kenarı, bu bölgelerin kenarlarından birine çarpmalıdır. Buna ek olarak, bu C -obstacle'ın çıkan kenarı, bölgeyi öyle terketmeli ki yaratılan bölge hücrenin bir kenarını içersin. Bu Şekil 5.14'de gösterilir. Bu şekilde, P_3 'ün E_i 'i, R_2 'nin sol kenarına çarpar ve E_o bu bölgeyi terk eder.



Şekil 5.14: P_1 ve P_2 hücreyi iki (açık gölgelendirilmiş) bölgeye ayıran C -obstacles'dir. P_3 hücrede üçüncü (koyu gölgelendirilmiş) bölgeyi yaratan C -obstacle'dir. R_1 ve R_2 hücredeki boş bölgeleri gösterir. E_i ve E_o P_3 'e gelen ve ayrılan kenarları gösterir. E_{o1} , E_{o2} ve E_{o3} , P_3 'ün hayali kenarlarını gösterir.

P_3 'ün R_2 'daki kösesi u olsun. Hayali bir quadrant çiziyoruz ve u 'yu bu quadrantın orijinine sabitliyoruz. Bu yatay ve dikey doğruların axisleri gösterdiği ve O noktasının bu quadrantın orijini gösterdiği şekil, Şekil 5.14 (b)'de gösterilir.

İspata, E_o 'nin mümkün olabilecek yerleşimlerini düşünerek ve bu üçüncü engelin çıkmayacağını göstererek devam ediyoruz. Öncelikle, eğer E_o 'i (u etrafında) quadrant-I içerisinde saat yönünde döndürürsek, u hala dar açı olarak kalır. Fakat Lemma 37'den, bir hücrenin içerisinde ki bir C -obstacle'ın herhangi bir kösesi geniş açığa sahip olmalıdır. Bu lemmanın üçüncü case'inde, ayrıca eğer u convex bir köşe olursa ve E_i ve E_o aynı quadrantta olursa, o zaman E_o 'in D uzunluğunda yatay/dikey bir kenar olacağını gösterdik (Şekil 5.13 (c)'e bakınız). Bu nedenle, E_o Quadrant II'de olamaz. Lemma 37'in birinci case'inde, Minkowski toplamının C -obstacle'a, uzunluğu D olan yeni dikey/yatay bir kenar tanıttığını gösteriyoruz.

E_o 'i (u etrafında) saat yönünde E_{o2} 'e kadar (E_{o2} 'i içermeyen) döndürürsek, o zaman u 'nin şekli Lemma 37'un birinci case'i ile çelişir. Çünkü bu durumda, E_o D uzunluğunda yatay bir kenar olmalıdır. Bu nedenle, E_o Quadrant III'de de olamaz. Son olarak, eğer E_o quadrant-III/IV ise, o zaman Lemma 37'un hiçbir case'i ihlal edilmez. Fakat, E_i ve E_o arasındaki köşe konkav olacaktır ve Proposition 36 nedeniyle, bu yeni C -obstacle, R_1 'deki bir kısım haricindeki hücrenin geride kalan kısmını işgal edecektir. Bu nedenle,

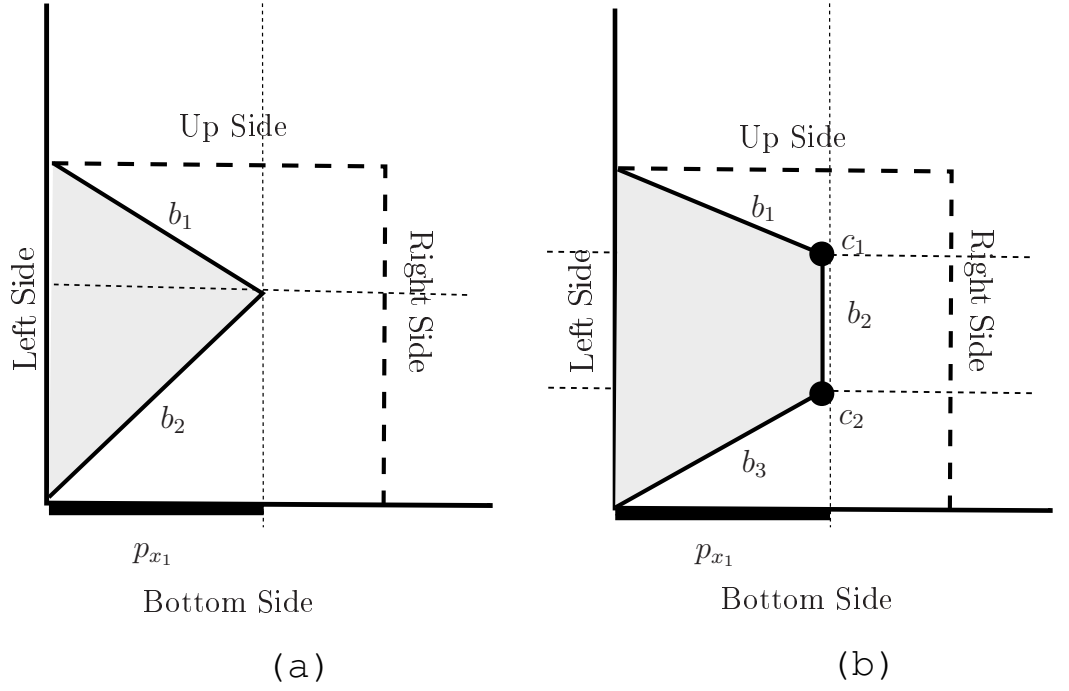
bu yapılandırma üçüncü bir bölge için hiç alan bırakmayacaktır. Yukarıdaki üç durum, bir *C-obstacle*'in hücreden ayrılabilen tüm mümkün çıkan (*Ing. outgoing*) kenarlarını kapsar. Tüm bu durumlar için, eğer hücre zaten iki bölgeye ayrılmış ise, bir başka bölgeyi yaratmanın mümkün olamayacağını gösteriyoruz.

Bundan sonra, bir hücreyi kapsamanın maliyeti için sınır buluyoruz. Öncelikle şu değişkenleri tanımlıyoruz: ρ robotun bir hücreyi kapsaması için izlediği yolu belirtir; a hücre sınırındaki kenarların kümesidir; b hücre içerisindeki *C-obstacle* sınırları üzerindeki kenarların kümesidir ve B *C-obstacle* ile kesişen hücre sınırını belirtir. Robot, bir grid hücrenin kapsanması sırasında CCWBF'i takip eder. Bu hareket planında eğer hücre engel içermiyor ise, robot hem a 'ı hem de b 'i içeren bir yol takip eder. Her $a_j \in a$, B 'i içermeyen hücre sınırındaki bir doğru parçasına bir projeksiyondur. Bu nedenle, b 'deki kenarların B 'deki doğru parçaları üzerine projeksiyonlarına odaklanıyoruz. Bundan sonra, bu doğru parçalarının non-overlapping olduğunu ve b 'deki tüm kenarları dolaşmanın maliyetinin B 'i dolaşmanın maliyetini aşmadığını gösteriyoruz, Nonoverlapping'in bu durumda her $b_j \in b$ 'in B 'deki farklı bir doğru parçası üzerine projekt edilebildiği anlamına geldiğini not edelim.

Lemma 39. *Her $b_j \in b$, B üzerindeki farklı bir doğru parçasına project edilebilir.*

İspat. Öncelikle b 'deki kenarların B 'deki doğru parçaları üzerine project edilmesini çalışıyoruz. Eğer b_j yatay/dikey bir kenar ise o zaman doğrudan B üzerindeki bir doğru parçasına projekt edilir. Aksi halde, hipotenüsü b_j olan bir üçgen tanımlarız. b_j üçgendeki iki dik kenarın kümesi olsun. b_j 'deki kenarları B 'de olan birbirlerine dik olan taraflarda olan doğru parçalarının üzerine project ediyoruz. Örneğin, Şekil 5.16'de, b_2 , b_3 ve b_4 'un dik kenarları sırasıyla (p_{x_2}, p_{y_1}) , (p_{x_3}, p_{y_2}) ve (p_{x_4}, p_{y_3}) doğru parçaları üzerine projekt edilir. Şekil 5.16'den görebileceğimiz gibi, overlapping olmuyor, çünkü b 'deki hiçbir kenar aynı doğru parçası üzerine projekt etmiyor.

Hücrenin alt tarafında overlap edebilecek olan b 'nin kenarları için ispatı çalışıyoruz (hücrenin diğer tarafları için de ispat simetriktir). b 'deki kenarların projeksiyonu hücrenin alt tarafında ancak farklı fakat zıt olmayan quadrantlarda yer alıyor ise overlap edebilirler. Overlap'ın mümkün olabileceği iki durum Şekil 5.16'de gösterilmiştir. Lemma 37'in birinci durumunda, E_i ve E_o eğer farklı fakat zıt olmayan quadrantlarda ise, o zaman Minkowski sum'ın *C-obstacle*'da uzunluğu D olan yeni dikey/yatay bir kenar tanıtaçağını



Şekil 5.15: Örnek kenarların y -axis üzerindeki projeksiyonu.

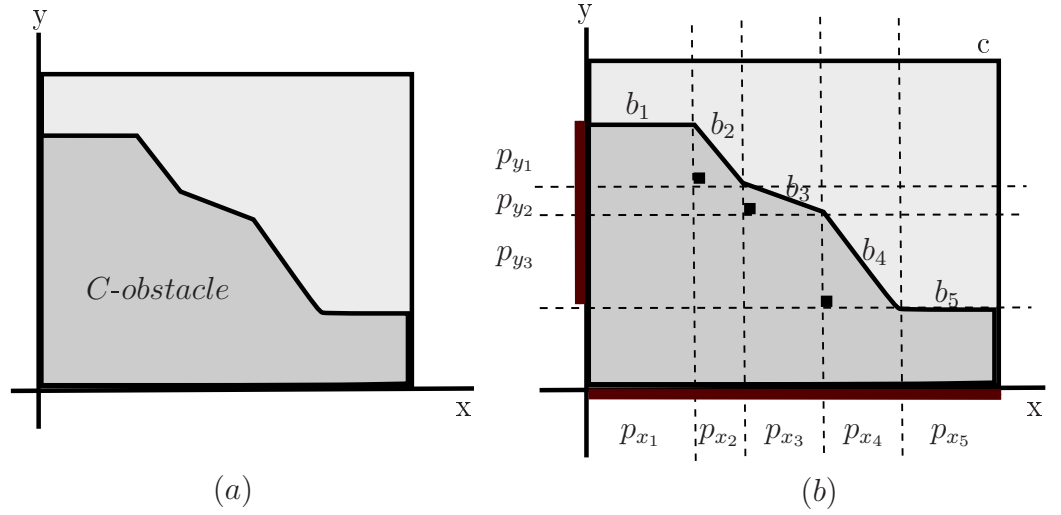
gösteriyoruz. b_2 'nin uzunluğu D olan yatay bir kenar olması gerektiğinden, Şekil 5.15 (a)'daki ilk overlapping durumunun olması mümkün değildir. Şekil 5.15 (b)'de, c_1 köşesinden hayali bir dikey ℓ doğrusu çizelim. Bu şekilden ℓ üzerinde olan c_1 'e en uzak noktanın c_2 olduğunu gözlemleriz. Fakat proposition 36'den, c_1 ve c_2 arasındaki uzaklığın D 'e eşit veya daha büyük olması gerektiğini biliyoruz. Bu şu anlama gelir; b_3 , b_2 'nin arkasından gelmeden, b_2 hücrede ayrılmalıdır. Bu nedenle, Şekil 5.15 (b)'de gösterilen ikinci overlapping durumunun da olması mümkün değildir. Sonuç olarak, b 'deki kenarların projeksiyonu overlap edemez.

Lemma 40. $b_j \in b$ 'nin yatay veya dikey bir kenar olmadığını varsayalım. b_j 'nin, b_j 'nin hipotenüs olduğu üçgendeki dik kenarların kümesi olduğunu tekrar hatırlayalım. O zaman b_j 'deki kenarlar B 'ye projekt edilebilir ve b_j 'yi dolaşma maliyeti b_j 'deki kenarları dolaşmanın maliyetinden daha fazla olamaz.

İspat. Üçgen eşitsizliğinden, hipotenüsün uzunluğu dik üçgendeki iki dik kenarın uzunluklarının toplamından daha küçük olmalıdır.

Lemma 41. b 'deki tüm kenarları dolaşmanın maliyeti, B 'i dolaşmanın maliyetini aşmaz.

İspat. b_j 'nin eğer yatay veya dikey bir kenar olursa, B üzerindeki bir doğru parçasına doğrudan projekt edildiğini biliyoruz. Ayrıca b_j eğer dik veya yatay



Şekil 5.16: p_{x_i} , b_1 ve b_5 arasındaki C -obstacle kenarlarının project edildiği x -axis'i üzerindeki doğru parçalarını gösterir. p_{y_i} , b_1 ve b_5 arasındaki C -obstacle kenarlarının project edildiği y -axis'i üzerindeki doğru parçalarını gösterir.

bir kenar değilse, o zaman b'_j 'deki kenarlar B 'nin farklı taraflarındaki doğru parçaları üzerine project edilir. b'_j 'nin dikey veya yatay bir kenar olmadığını varsayarak, Lemma 39'de, her $b_j \in b'$ 'nin B 'deki ayrı bir doğru parçasına projekt edilebildiğini gösteriyoruz. Bu nedenle, yatay/dikey kenarları ve yatay/dikey olmayan kenarları dolaşmanın maliyeti B 'yi dolaşmanın maliyetini aşmaz.

Lemma 42. *Bir hücreyi kapsama maliyeti (gidilen mesafe bakımından) $T_c \leq 8D$ 'dir.*

İspat. Şuan için, hücrenin sadece bir bölgeden oluştuğunu varsyalım. $Cost_a$, a 'daki kenarları dolaşmanın maliyeti ve $Cost_b$, b 'deki kenarları dolaşmanın maliyeti olsun. Eğer bir hücre engel içermiyorsa (*Ing.* obstacle-free) ise, robotun hem a 'i hem b 'i içeren bir yol izlediğini hatırlayalım. a 'daki ve b 'deki her kenar ayrık olduğundan (*Ing.* disjoint) ve hücrenin üzerindeki doğru parçaları üzerine projeksiyonları non-overlapping olduğundan, a ve b 'nin toplamı, hücrenin büyüklüğü olan $4D$ 'den daha fazla olamaz. Bir başka deyişle, bir bölge bir hücreyi kapsama maliyeti $4D$ ile üst sınırlandırılmıştır.

Lemma 38'de, bir hücrenin en fazla iki bölgeye ayrılabilirliğini göstermiştik. Bu nedenle, bir bölge bir hücreyi kapsama maliyetini iki ile çarparsak bir hücreyi kapsama maliyetini $8D$ olarak buluruz.

Theorem 43. *Bir $Disk_i$ 'i kapsama zamanı $T_{D_i} < 8N_iD$ 'dir. Burada N_i $Disk_i$ içindeki hücre sayısını gösterir.*

İspat. Lemma 42'den, bir hücreyi kapsama maliyetinin $\leq 8D$ olduğunu kanıtlıyoruz. Bu nedenle, bir $Disk_i$ kapsama zamanı, $8N_iD$ ile sınırlandırılır. N_i , $\pi r^{2i}/D^2$ ile sınırlıdır. N_i 'i, $Disk_i$ 'i kapsamak için olan zamanın sınırına yerleştirirsek $T_{D_i} < 8\pi r^{2i}/D$ 'i elde ederiz.

$\mathcal{SRP}_o$ Algoritmasının Analizi

Bu bölümde, Algoritma $\mathcal{SRP}_o$ 'ın analizini yapıyoruz. Argümanlar açık düzlem üzerinde iki robot ile çalıştığımız problemde tasarladığımız Algoritma SR 'ın analizi ile aynıdır. Tek fark robotun başarılı ve başarısız bir round i 'daki beklenen gidilen mesafesidir. Bu CA 'ın $8N_iD$ ile üst sınırlandırılmış olmasındandır. Bu nedenle, round i 'da beklenen gidilen mesafeyi, hem Stage-1'de hem de Stage-2'de aşağıdaki sınırı elde edebilmek için $8N_iD$ ile yer değiştiriyoruz.

1. Lemma 29'e benzer olarak, Stage-1'deki beklenen gidilen mesafe şu şekildedir

$$\sum_{i=0}^k \mathbb{E}[D_i|R_i] \mathbb{P}[R_i] < \frac{4\pi r^{2(k+1)}}{D(r^2 - 1)} \quad (5.8)$$

2. Lemma 30 ve 31'e benzer olarak, Stage-2'deki beklenen gidilen mesafe şu şekildedir

$$\begin{aligned} & \sum_{i=k+1}^{\infty} (E[D_i|S_i]Pr[S_i|R_i] + E[D_i|\bar{S}_i]Pr[\bar{S}_i|R_i]) Pr[R_i] \\ & < \frac{16\pi r^{2(k+1)}}{D(2 - r^2)} \end{aligned} \quad (5.9)$$

Theorem 44. *Optimal genişleme yarıçapı r engelli ortamlarda simetrik randevu algoritması için $r = 1.12$ 'dir. r 'in bu seçimi kompetitif oranı $293.058\frac{d}{D}$ olan bir algoritma verir.*

İspat. Beklenen gidilen mesafe Denklem(5.8) ve (5.9)'deki ifadeleri toplayarak elde edilir. Robotların arasındaki başlangıç uzaklığının (en kısa yolun) $d = r^{k+\delta}$ olduğunu hatırlayarak, r^k 'yi her gördüğümüz yere $dr^{-\delta}$ yerleştiriyoruz. Daha sonra da robotlar arasındaki optimal offline yolun uzunluğu olan $d/2$ 'ye bölüyoruz. Bu ifade $\delta = 0$ 'da en büyük olur. Bunun karşılığında, $r = 1.2$ seçimi $293.058\frac{d}{D}$ minimum kompetitif oran garantisini verir.

5.3.4 $\mathcal{SRP}_o$ Algoritmasının Beklenen Round Sayısı

$\mathcal{SRP}_o$ Algoritması randevu gerçekleşene kadar devam eder. Lemma 45’de, randevu gerçekleşene kadar çalıştırılmasını gereken round sayısının beklenen değerine alt ve üst sınır buluyoruz.

Lemma 45. *Algoritma $\mathcal{SRP}_o$ ’daki beklenen round sayısı k ile alt sınırlı ve $k + 2$ ile üst sınırlıdır.*

Kanıt. Lemma 32’ün ispatı ile aynıdır. □

Algoritma $\mathcal{SRP}_o$ ’nin beklenen tekrar sayısı bu algoritma için gerekli olan randevu zamanı ile üst sınırlıdır. Randevu zamanı Bölüm 5.3.3’de analiz edilmiştir.

5.3.5 Engelli Ortamlarda Çok Robotlu Simetrik Randevu Arama

Analiz

Theorem 46. *Genişleme yarıçapı için en iyi seçim $r = 1.14$ ’dir. Bu seçim $O(\frac{d}{\log(n)R})$ mesafe kompetitif oranını verir. Zaman kompetitif oran bu sınırın n katıdır.*

Stage-1 Analizi Bu bölümde, round $i \leq k$ olduğu zaman robot tarafından beklenen gidilen mesafeyi hesaplıyoruz. Hareketli robotun bu aşamada yer değişimi en fazla $r^k < r^{k+\delta}$ olduğundan tüm robotlar ile buluşamaz. Şimdi bu aşamadaki beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 47. *Stage-1’de robot tarafından beklenen gidilen mesafe sununla sınırlıdır*

$$\sum_{i=1}^k \mathbb{E}[D_i] < \frac{d^2 4\pi}{nD} \frac{r^{2-2\delta}}{r^2 - 1}.$$

İspat. Robotlar buluşabilecek kadar ilerlemediklerinden, k ’ya kadar her round yapılır. Bir başka deyişle, $0 \leq i \leq k$ için $\mathbb{P}[R_i]=1$ ’dir. Eğer round i ’da hareket ederse, robotun Algoritma 5’i çalıştırdığını biliyoruz. Bu nedenle, round i ’da

beklenen gidilen mesafe Denklem (5.2)'deki gibidir. Bu denklemi $\mathbb{E}[D_i|R_i]$ yerine kullanarak, bu aşamada beklenen gidilen mesafenin aşağıdaki gibi olduğunu buluyoruz

$$\begin{aligned}
\sum_{i=0}^k \mathbb{E}[D_i] &= \sum_{i=0}^k \mathbb{E}[D_i|R_i]\mathbb{P}[R_i] \\
&= \sum_{i=0}^k \mathbb{E}[D_i|A_i]\mathbb{P}[A_i] = \frac{1}{n} \sum_{i=0}^k \mathbb{E}[D_i|A_i] \\
&= \frac{1}{n} \sum_{i=0}^k \frac{8\pi r^{2i}}{D} = \frac{8\pi}{nD} \sum_{i=0}^k r^{2i} \\
&= \frac{8\pi}{nD} \cdot \frac{r^{2k+2} - 1}{r^2 - 1} \\
&< \frac{d^2 4\pi}{nD} \frac{r^{2-2\delta}}{r^2 - 1}
\end{aligned} \tag{5.10}$$

Bu $\delta = 0$ için en büyük olur.

Stage-2 Analizi Bu bölümde, $i \geq (k+1)$ olan roundları düşünüyoruz. Stage-1'den farklı olarak, robotlar sıfırdan farklı olasılıkla buluşurlar. Şimdi Stage-2'de beklenen gidilen mesafeyi hesaplıyoruz.

Lemma 48. *Stage-2'de beklenen gidilen mesafe aşağıdaki ile sınırlıdır*

$$\sum_{i=k+1}^{\infty} \mathbb{E}[D_i | \overline{S}_i] < \frac{8\pi d^{2-\delta} r^2 \mathbb{P}[\overline{M}_i]}{nD(1 - r^2 \mathbb{P}[\overline{M}_i])}$$

İspat. Robot tarafından beklenen gidilen mesafe aşağıdaki ile sınırlıdır

$$\sum_{i=k+1}^{\infty} (\mathbb{E}[D_i | S_i]\mathbb{P}[S_i] + \mathbb{E}[D_i | \overline{S}_i]\mathbb{P}[\overline{S}_i])\mathbb{P}[R_i] \tag{5.11}$$

$$\begin{aligned}
&= \sum_{i=k+1}^{\infty} \mathbb{E}[D_i | S_i \wedge R_i]\mathbb{P}[S_i \wedge R_i] + \\
&\quad \mathbb{E}[D_i | \overline{S}_i \wedge R_i]\mathbb{P}[\overline{S}_i \wedge R_i]
\end{aligned} \tag{5.12}$$

M_i 'in round i 'da sadece bir robotun aktif olma olayı olduğunu hatırlayalım. M_i olasılığı Denklem (5.3)'deki gibi verilir. Algoritmanın round i 'da aktif olduğunu varsayarak, robotlar sadece M olayı olduğunda buluşurlar. Dahası, robot tarafından beklenen gidilen mesafe sadece A_i olayı olduğunda sıfırdan farklıdır. Bu $\mathbb{E}[D_i | \overline{S}_i \wedge R_i] = \mathbb{E}[D_i | \overline{M}_i \wedge A_i \wedge R_i]$ ve $\mathbb{E}[D_i | S_i \wedge R_i] = \mathbb{E}[D_i | M_i \wedge A_i \wedge R_i]$ olduğu anlamına gelir. Bu değerleri Denklem 5.6'de kullanarak,

Stage-2'de beklenen gidilen mesafeyi aşağıdaki gibi buluyoruz

$$\begin{aligned}
& \sum_{i=k+1}^{\infty} \mathbb{E}[D_i \mid M_i \wedge A_i \wedge R_i] \mathbb{P}[M_i \wedge A_i \wedge R_i] + \\
& \quad \mathbb{E}[D_i \mid \overline{M}_i \wedge A_i \wedge R_i] \mathbb{P}[\overline{M}_i \wedge A_i \wedge R_i] \\
&= \sum_{i=k+1}^{\infty} \left(\frac{8\pi r^{2i}}{D} \mathbb{P}[M_i] \mathbb{P}[A_i] + \frac{8\pi r^{2i}}{D} \mathbb{P}[\overline{M}_i] \mathbb{P}[A_i] \right) \mathbb{P}[\overline{M}_i]^{i-k} \\
&= \frac{8\pi r^{2i}}{D} \sum_{i=k+1}^{\infty} r^{2i} \mathbb{P}[\overline{M}_i]^{i-k} \\
& \quad i \leftarrow i + k + 1 \\
&= \frac{8\pi}{nD} \sum_{i=0}^{\infty} r^{2(i+k+1)} \mathbb{P}[\overline{M}_i]^{i+1} = \frac{8\pi r^{2(k+1)} \mathbb{P}[\overline{M}_i]}{nD} \sum_{i=0}^{\infty} (r^2 \mathbb{P}[\overline{M}_i])^i \\
&= \frac{8\pi r^{2(k+1)} \mathbb{P}[\overline{M}_i]}{nD(1 - r^2 \mathbb{P}[\overline{M}_i])} = \frac{8\pi d^{2-\delta} r^2 \mathbb{P}[\overline{M}_i]}{nD(1 - r^2 \mathbb{P}[\overline{M}_i])}
\end{aligned}$$

Bu da $\delta = 0$ için en büyük olur.

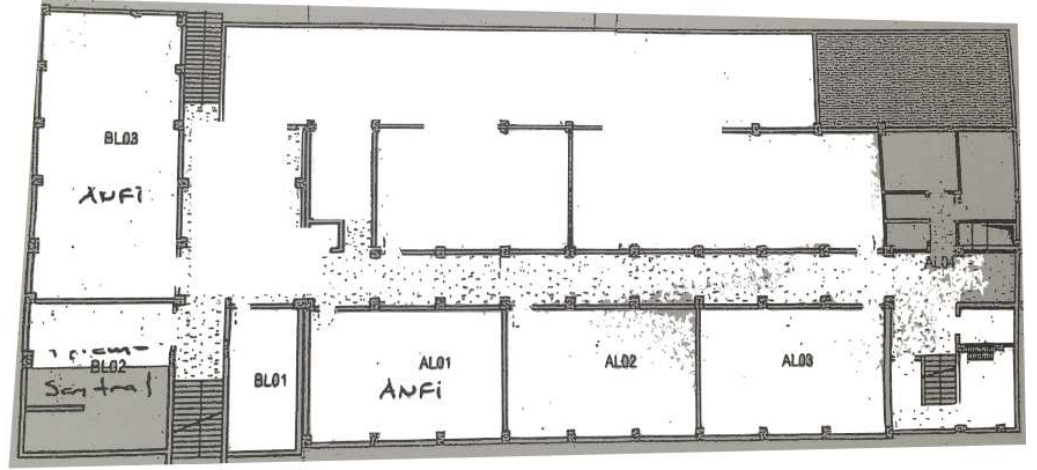
Mesafe Kompetitif Oranın Hesaplanması Her iki aşamada da beklenen gidilen mesafe için sınırları bulduktan sonra, algoritmamızın mesafe kompetitif oranını hesaplamaya hazırız.

Proof of Theorem 46. Beklenen gidilen mesafe Lemma 47 ve 48'deki sınırların toplamıdır. Kompetitif oranı elde etmek için, bu toplamı $d/2$ 'e bölüyoruz. $\Theta(d)$ büyük d için çıkan ifadeyi domine eder. Domine eden ifade şudur

$$\frac{d^2 r^2 4\pi}{nD} \left(\frac{1}{r^2 - 1} + \frac{2\mathbb{P}[\overline{M}_i]}{1 - r^2 \mathbb{P}[\overline{M}_i]} \right)$$

Bu nedenle, d katsayısını en küçük hale getirecek r değerini seçiyoruz. En iyi değer $r = 1.14$ 'dir. Bu da $O\left(\frac{d}{\log(n)R}\right)$ mesafe kompetitif oranını verir.

Zaman kompetitif oranının analizi mesafe kompetitif hesaplamaları ile aynıdır. Zaman kompetitif oranı, mesafe kompetitif oranının n katıdır ve optimal r değeri zaman kompetitif oran için de aynıdır.

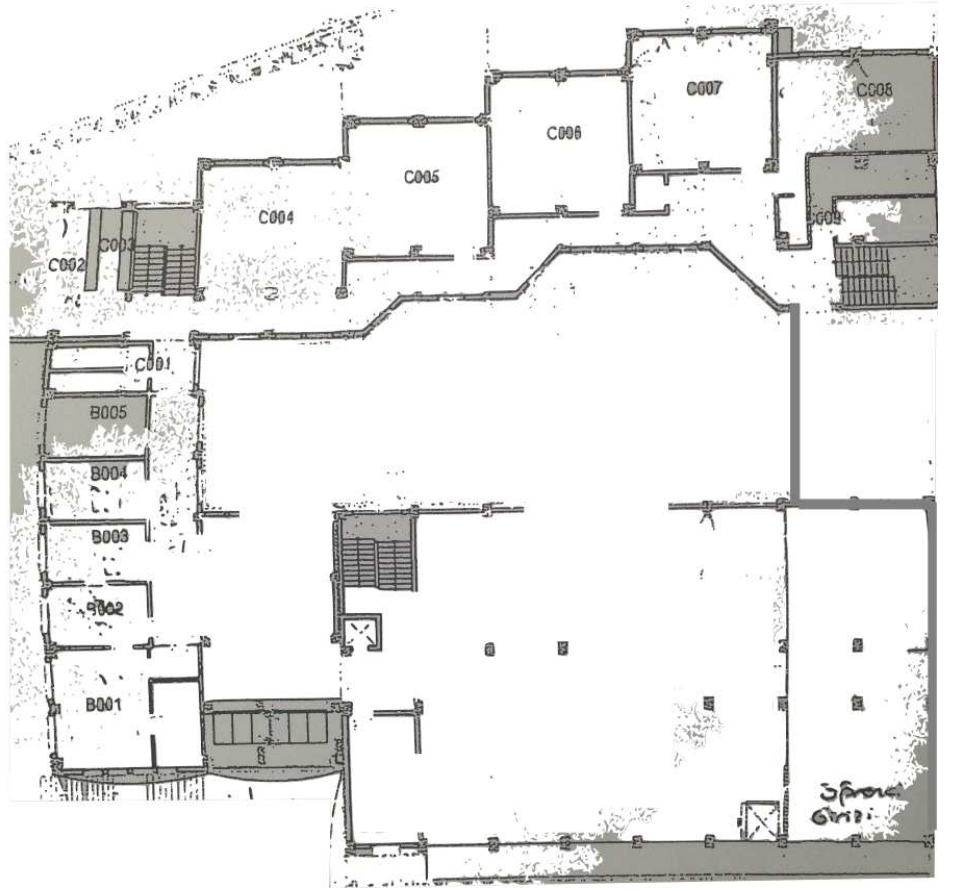


Şekil 5.17: Ortam-1.

5.3.6 Simülasyon Deneyleri

Bu bölümde, Algoritma SRP_o 'ın algoritmasının, üç farklı iç ortam üzerinde çalıştırılarak elde edilen deneysel sonuçlarını sunuyoruz. Sonuçlar teorik analizde elde ettiğimiz sonuçlar ile uyumludur. Deneyler, ROS (The Robot Operating System ROS) platformunda yürütülmüştür. Ortamların hepsi $19m \times 19m$ üzerinde genişlektir. $D = 1m$ 'dir ve herbir ortam üzerine 19×19 grid yerleştirilmiştir. Bu örnek ortamlar Şekil 5.17 (Ortam-1), 5.18(Ortam-2) ve 5.19(Ortam-3)'de sunulmuştur. Ortam örnekleri İzmir Üniversitesinin kat planlarından alınmıştır. Simülasyonlar sırasında $r = 1.12$ ve robotun ortalama hızı $0.5m/sec$ 'dir. Simülasyonun aralık zamanı (her adımın güncellenmesi) $200msec$ 'dir. Şekil 5.19, robotun Ortam-3'deki hareketini gösterir. Robot hukoyo laser sensör ile donatılmıştır. Robotun hareketi sırasındaki laseri ile ortamda gördüğü alan, şekilde yeşil taranmış olarak görünür.

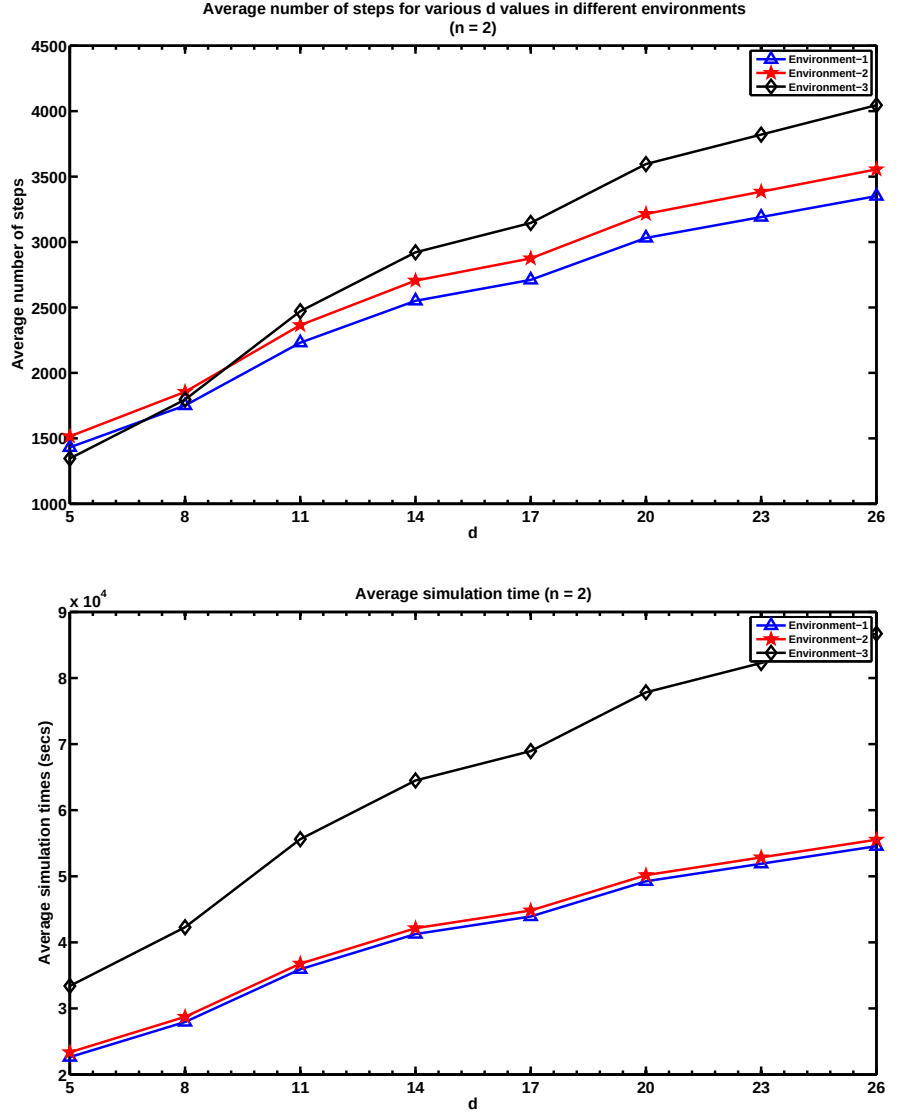
Şekil 5.20 ve 5.21'deki sonuçlar 5000 denemenin ortalaması alınarak elde edilmiştir. Şekil 5.20'deki grafikler, robotların konumlandırıldığı üç farklı ortamda çeşitli d değerleri için ortalama adım sayısını ve ortalama simülasyon zamanını gösterir. Sonuçlar, randevu zamanların 1. ve 2. ortamlar için birbirine yakın olduğunu ve 3. ortamdakine göre daha iyi olduğunu göstermiştir. Şekil 5.21, üç farklı ortama yerleştirilen çeşitli boyutlardaki gridlerin robot tarafından tamamen kapsanması için gerekli olan ortalama adım sayısını göstermektedir. Gridin şekilde test edilen herbir boyutu teorik analizde kullandığımız hücre sayısı olan N 'e eşittir. Sonuçlar, CA algoritmasının performansına bulduğumuz $8ND$ üst sınırı ile uyumludur.



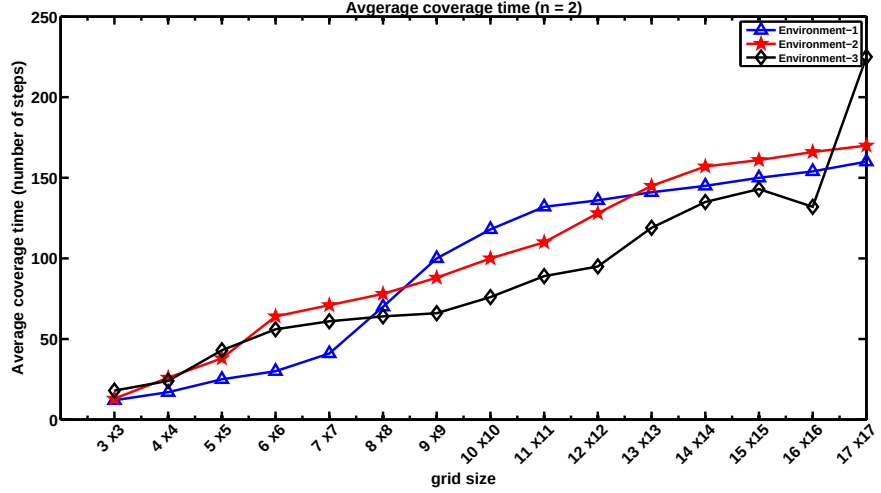
Şekil 5.18: Ortam-2



Şekil 5.19: ÜST: Ortam-3, ALT: Robotun Ortam-3'deki hareketi



Şekil 5.20: Engelli ortamlarda, SRP_o algoritmasını çalıştıran iki robotla simülasyon deneyleri. Farklı ortamlarda, d 'deki değişime göre; **ÜST**: randevu için ortalama adım sayısı, **ALT**: randevu için ortalama simülasyon zamanı.



Şekil 5.21: Engelli ortamlarda *CA* kapsama algoritmasını çalıştıran iki robotla simülasyon deneyleri. Farklı ortamlarda, grid boyutundaki değişime göre, gridin tam kapsanması için gereken ortalama adım sayısı.

5.4. Sonuçlar

Bu bölümde, düzlemsel bir ortamda birbirlerinden farklı pozisyonlarda konuşlandırılmış sınırlı iletişim alanına sahip robotların nasıl buluşabileceği problemini çalıştık. Aynı özelliklere sahip olan bu robotların aynı randevu stratejisini çalıştırdığını düşünüyoruz. Robotlar alan üzerindeki kendi pozisyonlarından, birbirlerinin pozisyonlarından ve aralarındaki başlangıç uzaklığından habersizdir.

Bu problemi ilk olarak engelsiz ortamlarda iki robot ile çalıştık ve simetrik bir randevu arama stratejisi sunduk. Stratejimizin detaylı mesafe ve zaman kompetitif analizini yaptık ve $O(d/R)$ karmaşıklığa sahip olduğunu gösterdik. Algoritmamızı daha sonra ikiden çok robot için genişlettik. Sunduğumuz performans analizinde robot sayısı (n) arttıkça beklenen buluşma zamanının azaldığını ve genişletilmiş stratejimizin $O(d/\log(n)R)$ karmaşıklığa sahip olduğunu gösterdik. Teorik sonuçlarımız yaptığımız simülasyonlarla destekledik. Bölümün ikinci kısmında, simetrik randevu algoritmamızı poligonal engellerin olduğu ortamlar için genişlettik. Robot ortamla ilgili önceden bir bilgiye sahip değildir, fakat sensörleri ile online olarak ortamla ilgili bilgi edinir. Genişletilmiş stratejimizi kare ve çevrilen robotlar için analiz ediyoruz ve kompetitif karmaşıklığının iki robot için $O(d/D)$ ve ikiden çok robot için $O(d/\log(n)D)$ olduğunu gösterdik. Burada D kare robotun bir kenarının

uzunluğudur. Bu sonucu elde ederken ayrıca bir robotla bilinmeyen rasgele poligonal bir ortamı keşfetmek için de bir üst sınır elde ettik. Son olarak algoritmamızın performansını araştıran simülasyonların ve ROS platformunda farklı iç ortamlarda yaptığımız deneylerinin sonuçlarını sunduk.

SONUÇ VE TARTIŞMA

Son iki yüzyılda, robotik topluluğundaki çoğu araştırmacı çok robotlu sistemlerin tasarlanması ve kullanımı ile ilgili konuları araştırmaktadır. Robotikte ilerlemeler mobil robotların daha önceden etkili şekilde yapılması mümkün olmayan birçok karmaşık işte kullanımını mümkün hale getirmiştir. Örneğin, robotlar afet ve patlama gibi durumlarda kurbanları veya hayatta kalanları arayabilir. Bunun dışında, insanların ulaşmasının mümkün olamayacağı tehlikeli ortamları keşfetmek veya ortamın haritasını çıkarmak için de görevlendirilebilir.

Bu tezde, çok mobil robotlu sistemlerde meydana gelebilecek temel bir problem olan randevu arama problemini çalışıyoruz. Randevu arama problemi, birbirlerinin konumlarını bilmeyen bir ortamdaki iki veya daha fazla oyuncunun en kısa zamanda nasıl buluşabileceğini sorgular. Biz bu tezde, buluşmak isteyen oyuncuların bilinmeyen bir ortamda konumlandırılmış robotlar olduğunu düşünüyoruz. Problemi öncelikle, doğru üzerinde çalışıyoruz. Daha sonra aynı problemi engelli ve engelsiz düzlemsel ortamlarda çalışıyoruz. Randevu arama problemi iki boyutlu ortamlarda çalışıldığında robotun bulunduğu statik ortamı sensörleri ile algılamasını da gerektirir. Bu nedenle, düzlemsel ortamlarda randevuya odaklanırken aslında önceden bilinmeyen bir ortamın bir robot tarafından keşfedilmesi problemini de çalışmış oluyoruz.

3. bölümde, daire/halka üzerinde ve çizge/ağ üzerinde randevu arama problemi ile ilgili yapılan çalışmaları sunuyoruz. Buna ek olarak, bu bölümde randevu arama probleminin bir genelleştirmesi olan *Toplanma* (Ing. gathering) ve *Karşılaşma* (Ing. coalescence) problemi ile ilgili yapılan çalışmaları da sunuyoruz.

4. bölümde, simetrik randevu arama problemini doğru üzerinde iki robotla çalışıyoruz. Doğruyu bir cadde, yol, nehir vb. şekilde modellemek mümkündür. Bu problemi robotların kendileri ve birbirleri hakkında herhangi bir bilgiye sahip olmadığı durumda çalışıyoruz. Yani, robotlar doğru üzerindeki kendi konumlarını veya birbirlerinin konumlarını, birbirlerine olan başlangıç uzaklığını bilmezler. Ortamın bir boyutlu olduğunu düşündüğümüz için robotların arasında herhangi bir engel yoktur. Bu bölümdeki en önemli katkımız, problemi birbirleri arasındaki başlangıç uzaklığını bilmeden ve robotların simetrik olduğunu yani buluşmak için aynı stratejiyi kullandıklarını düşünerek çalışmaktır.

4. bölümün ilk kısmında, problemi robotların senkron olduğu durumu için çalışıyoruz. Yeni bir simetrik randevu arama stratejisi ve stratejimizin detaylı analizini sunuyoruz. Çalışmamızda, Spiral LostCow algoritmasını simetrik robotlar için genelleştiriyoruz. Genişleme yarıçapının optimal değerini, $r = 1.195$ olarak belirliyoruz. r 'ın bu seçimi için, simetrik stratejimizin toplam gidilen mesafe için 17.686 ve toplam zaman için 24.843 kompetitif orana sahip olduğunu gösteriyoruz. Algoritmamız, daha önce en iyi bilinen, 26.650 zaman ve mesafe kompetitif oranına sahip olan Baston ve Gal'in algoritmasına bir iyileştirmedir. Simulasyonlarda, $\mathcal{SR}$ 'ın performansını çok çeşitli r değerleri ve başlangıç uzaklıkları ile karşılaştırıyoruz. Ayrıca, algoritmamızın navigasyonel hatalara da dayanıklı olduğunu gösteriyoruz. Sonuçlarımız teorik analiz ile tutarlıdır. Ayrıca pratik senaryolarda algoritmamızın iki uzantısını sunuyoruz: doğru parçaları ve yarım doğrular ve basit kapalı-yol ortamları (Örneğin, dairesel bir koridor).

4. bölümün ikinci kısmında, aynı problemi robotların aramaya farklı zamanlarda başladığı, robotların asenkron durumu için çalışıyoruz. Problemin bu ayarlaması için simetrik randevu stratejimizi değiştiriyoruz ve değişmiş stratejimizin ön analiz sonuçlarını sunuyoruz. Bilgimiz kadarıyla, çalışmamız literatürdeki doğru üzerinde simetrik randevu arama problemini bilinmeyen uzaklık ve asenkron robotlar için olan ilk ön sonuçları sunar. $\mathcal{ASR}$ 'ın performansını çeşitli r ve t değerleri ve çeşitli başlangıç uzaklıkları için araştırıyoruz. Buna ek olarak, $\mathcal{SR}$ 'ın ve $\mathcal{ASR}$ 'ın performanslarını çeşitli başlama zamanları için karşılaştırıyoruz.

5. bölümde, engelli ve engelsiz düzlemsel ortamlarda, aynı özelliklere sahip, farklı pozisyonlarda konuşlandırılmış iki veya ikiden çok robotun simetrik randevu arama problemini çalışıyoruz. Her robot bir görme sensörü ile donatılmıştır ve sınırlı $R > 0$ iletişim alanına sahiptir. Robotlar önceden ortamla ilgili bir bilgiye sahip değildir. Ortamı ve engelleri sensörleri tarafından sağlanan görsel bilgiler ile öğrenirler. Robotların aramadaki kararları sadece alanın o anki gördükleri bölümüne bağlıdır, bu yüzden robotların araması online bir problem olarak görülebilir. Bunun dışında, robotlar unutkan, yani geçmişe ait bir hafızaları yoktur. Robotların ortak bir zamanlaya bağlı olduğunu ve aramaya aynı anda başladığı senkron durumu çalışıyoruz. Robotlar alan üzerindeki kendi pozisyonlarından, birbirlerinin pozisyonlarından ve aralarındaki başlangıç uzaklığından habersizdir.

5. bölümün birinci kısmında, problemi ilk olarak engelsiz ortamlarda iki

robot ile çalışıyoruz ve simetrik bir randevu arama stratejisi sunuyoruz. Stratejimizin detaylı mesafe ve zaman kompetitif analizini yapıyoruz ve $O(d/R)$ kompetitif karmaşıklığa sahip olduğunu gösteriyoruz. Daha sonra, algoritmamızı ikiden çok robot için genişletiyoruz. Sunduğumuz performans analizinde robot sayısı (n) arttıkça beklenen buluşma zamanının azaldığını ve genişletilmiş stratejimizin $O(d/\log(n)R)$ karmaşıklığa sahip olduğunu gösteriyoruz. Bu kısmın sonunda hem iki robot için olan stratejimiz hem de ikiden çok robot için genişlettiğimiz stratejimiz için elde ettiğimiz teorik sonuçlarımızı simülasyonlarla destekliyoruz.

5. bölümün ikinci kısmında, simetrik randevu algoritmamızı poligonal engellerin olduğu düzlemsel ortamlar için genişletiyoruz. Genişletilmiş stratejimizi kare ve çevrilen robotlar için analiz ediyoruz ve kompetitif karmaşıklığının iki robot için $O(d/D)$ olduğunu gösteriyoruz. Burada D kare robotun bir kenarının uzunluğudur. Bu bölümün birinci kısmında olduğu gibi stratejimizi ikiden çok robot için genişletiyoruz ve $O(d/\log(n)D)$ kompetitif karmaşıklığa sahip olduğunu gösteriyoruz. Bu sonucu elde ederken ayrıca bir robotla bilinmeyen rasgele poligonal engeller olan düzlemsel bir ortamı keşfetmek için bir üst sınır elde ediyoruz. Son olarak algoritmamızın performansını araştıran simülasyonların ve ROS platformunda farklı iç ortamlarda yaptığımız deneylerin sonuçlarını sunuyoruz.

6.1. Gelecek Araştırmalar

Bu tezde, açık olan randevu arama problemleri için yeni randevu stratejileri sunuyoruz ve bu stratejilerin kompetitif oranlarını hesaplıyoruz. Bu bölümde, bu tezde çalışılan problemler için gelecekte yönlenilebilecek araştırmaları tartışıyoruz.

4. bölümde, doğru üzerindeki simetrik randevu arama problemini çalıştık. Bu problemde iki robotun randevusunu düşündük. Aynı problem $n > 2$ robotun olduğunu durum için hala açıktır. Lim et al. (1997) oyuncuların birbirlerine eşit uzaklıkta yerleştirildiğini ve her oyuncunun arasında bir birim uzaklık olduğunu düşünmüştür. $n > 2$ robotun aralarındaki uzaklığı bilmediği ve oyuncular arasındaki uzaklığın rasgele bir değer olduğu durumda simetrik randevu zamanı ne olduğu araştırılabilir. Ayrıca bu tezde önerdiğimiz simetrik randevu stratejisi halka ve daire olarak modellenen ortamlara genişletilebilir. Bu bölümün ikinci kısmında, asenkron durumu ele almıştık ve

sunduğumuz simetrik randevu stratejisinin analizi yapmıştık. Analizde elde ettiğimiz sonuçlar, bu problem için ön üst sınırları sunar. Bu sonuçlar analiz üzerinde daha detaylı çalışılarak daha da iyileştirilebilir. İlginç bir bir randevu arama problemi de iki robotun olduğu ve bu robotların, gerçek bir cadde üzerindeki n tane birbirlerinden ayırt edilemeyen buluşma noktasının birinde buluşmak istediği durum için modellenmesidir.

5. bölümde, düzlemsel ortamlarda simetrik randevu arama problemini çalıştık. Problemi ilk olarak açık düzlemde çalıştık. Bu problem için önerdiğimiz stratejiye üst sınır bulmak için buluşmanın bir robotun beklerken ve diğer robotun hareketliken gerçekleştiğini düşündük. Bu sınır, iki robotun her round başında hareket ettikleri yön rasgeleleştirilerek ve robotların her ikisinin de hareketli olduğu durumdaki buluşması dahil edilerek iyileştirilebilir. Bu bölümün ikinci kısmında randevu arama problemi poligonal engellerin olduğu düzlemsel ortamlarda çalıştık. Ortamın deliklerden oluşan poligon olarak sunulduğu ve ortamdaki engellerin şekillerinin rasgele olduğu durumda robotların randevusu hala açık olan zor bir problemdir. Bu kısımda ortamı gride bölerek kesikli hale getirmiştik, tüm ortamı bölümlere ayırmak da mümkündür ve randevu zamanı üzerindeki etkisi araştırılabilir. Bu bölümde robotları senkron olduğunu düşündük. Aynı problem robotların asenkron durumu için hala açıktır ve literatürde kompetitif yaklaşım kullanılarak herhangi bir sonuç sunulmamıştır. Buna dışında, senkronluğun sadece robotların başlama zamanlarındaki farklılıktan dolayı değil, ayrıca ortamda bir düşmanın olduğu ve robotların hareketlerine müdahale ettiği durumda da bozulduğu düşünülebilir ve her iki durum için simetrik randevu zamanı araştırılabilir.

Engelli ortamları belirli sayıda sensör yerleştirildiğini düşünelim. Bir başka ilginç araştırma konusu sensörlerden faydalanarak robotlar daha kısa sürede buluşup buluşamayacağıdır. İki boyutlu ortamlarda randevunun gerçekleşmesi için robotun ortamı keşfetmesi gerekir. Robot ortamı ne kadar hızlı keşfederse buluşma o kadar hızlı gerçekleşir. Poligonal engellerin olduğu poligonal ortamlarda etkili keşif için global stratejilerin tasarımı hala açık bir araştırma konusudur. Randevu ayrıca 3D ortamlarda da düşünülebilir. Örneğin Thomas and Hulme (1997)'deki çalışmada bir kurtarma helikopterinin çölde kaybolan bir kişiyi ne kadar zamanda bulabileceği simülasyonlarla araştırılır.

Literatürdeki randevu arama problemi ile ilgili her çalışmada robotların buluşana kadar aramaya devam ettiklerini düşünülür. Fakat, iki robotun

buluşmak istediđi geniş bir ortamda bir robot bozulabilir. Böyle bir durum olabileceđi düşünülerek arama ne zaman sonlandırılmalıdır? Bu problem Alpern (2013)'de tartışılmıştır.

KAYNAKLAR DİZİNİ

- Adept MobileRobots research robots, <http://www.mobilerobots.com/ResearchRobots/ResearchRobots.aspx>, note = Accessed: 2011.
- Optimal memory rendezvous of anonymous mobile agents in a unidirectional ring.
- Ros robotic operating system (ros), <http://www.ros.org/>.
- Willow Garage personal robot 2 (pr2), <http://www.willowgarage.com>, Accessed: 2010-09-30.
- Agah, F., Mehrandezh, M., Fenton, R., and Benhabib, B.**, 2003, Rendezvous-guidance based robotic interception, In Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, volume 3, 2998-3003 pp. IEEE.
- Agmon, N. and Peleg, D.**, 2004, Fault-tolerant gathering algorithms for autonomous mobile robots, In Proceedings of the fifteenth annual ACM/SIAM symposium on Discrete algorithms, 1078 p. Society for Industrial and Applied Mathematics.
- Albers, S., Kursawe, K., and Schuierer, S.**, 2002, Exploring unknown environments with obstacles, *Algorithmica*, 32(1), 123-143.
- Alpern, S.**, Asymmetric rendezvous search on the circle, *Dynamics and Control*, 10(1).
- Alpern, S.**, 1995, The rendezvous search problem, *SIAM Journal on Control and Optimization*, 33(3), 673-683.
- Alpern, S.**, 2002, Rendezvous search: A personal perspective, *Operations Research*, 50(5), 772-795.
- Alpern, S.**, 2013, Ten open problems in rendezvous search, 223-230 pp.
- Alpern, S. and Baston, V.**, 2006, Rendezvous in higher dimensions, *SIAM journal on control and optimization*, 44(6), 2233-2252.
- Alpern, S. and Beck, A.**, 1997, Rendezvous search on the line with bounded resources: expected time minimization, *European Journal of Operational Research*, 101(3), 588-597.
- Alpern, S. and Beck, A.**, 1999, Asymmetric rendezvous on the line is a double linear search problem, *Mathematics of Operations Research*, 24(3), 604-618.

KAYNAKLAR DİZİNİ (devam)

- Alpern, S. and Beck, A.**, 2000, Pure strategy asymmetric rendezvous on the line with an unknown initial distance, *Operations Research*, 48(3), 498.
- Alpern, S. and Gal, S.**, 2003, *The Theory of Search Games and Rendezvous*, Springer.
- Alpern, S. and Street, H.**, 2001, Rendezvous in One and More Dimensions.
- Anderson, E. and Essegaiier, S.**, 1995, Rendezvous search on the line with indistinguishable players, *SIAM Journal on Control and Optimization*, 33, 1637.
- Anderson, E. and Fekete, S.**, 2001, Two dimensional rendezvous search, *Operations Research*, 107-118 pp.
- Anderson, E. J. and Weber, R.**, 1990, The rendezvous problem on discrete locations, *Journal of Applied Probability*, 839-851 pp.
- Baeza-Yates, R., Culberson, J., and Rawlins, G.**, 1993, Searching in the plane, *Information and Computation*, 106, 234-234.
- Bampas, E., Czyzowicz, J., Gȓsieniec, L., Ilcinkas, D., and Labourel, A.**, 2010, Almost optimal asynchronous rendezvous in infinite multidimensional grids, 297-311 pp.
- Baston, V.**, 1999, Note: Two rendezvous search problems on the line, *Naval Research Logistics*, 46(3), 335-340.
- Baston, V. and Gal, S.**, 1998, Rendezvous on the line when the players' initial distance is given by an unknown probability distribution, *SIAM Journal on Control and Optimization*, 36, 1880.
- Baston, V. and Gal, S.**, 2001, Rendezvous search when marks are left at the starting points, *Naval Research Logistics*, 48(8), 722-731.
- Beck, A. and Newman, D. J.**, 1970, Yet more on the linear search problem, *Israel J. Math.*, 8, 419-229.
- Belkhouche, F., Belkhouche, B., and Rastgoufard, P.**, 2005, Multirobot rendezvous in the plane, In 2005 IEEE International Conference on Systems, Man and Cybernetics, 2293-2298 pp.

KAYNAKLAR DİZİNİ (devam)

- Beveridge, A., Ozsoyeller, D., and Isler, V.,** 2011, Symmetric rendezvous on the line with an unknown initial distance, Technical Report, 11-008, University of Minnesota, http://www.cs.umn.edu/research/technical_reports.php?page=report&report_id=11-008.
- Chalopin, J., Das, S., and Santoro, N.,** 2007, Rendezvous of mobile agents in unknown graphs with faulty links, *Distributed Computing*, 108-122 pp.
- Cieliebak, M.,** 2004, Gathering non-oblivious mobile robots, LATIN 2004: *Theoretical Informatics*, 577-588 pp.
- Cieliebak, M., Flocchini, P., Prencipe, G., and Santoro, N.,** 2003, Solving the robots gathering problem, *Automata, Languages and Programming*, 192-192 pp.
- Cieliebak, M. and Prencipe, G.,** 2002, Gathering autonomous mobile robots, In Proc. 9th Int. Colloq. on Structural Information and Communication Complexity, 57-72 pp. Citeseer.
- Cohen, R. and Peleg, D.,** 2006, Convergence of autonomous mobile robots with inaccurate sensors and movements, 549-560 pp.
- Collins, A., Czyzowicz, J., Gȩsieniec, L., and Labourel, A.,** 2010, Tell me where i am so i can meet you sooner, 502-514 pp.
- Czyzowicz, J., Ilcinkas, D., Labourel, A., and Pelc, A.,** 2010a, Asynchronous deterministic rendezvous in bounded terrains, *Structural Information and Communication Complexity*, 72-85 pp.
- Czyzowicz, J., Ilcinkas, D., Labourel, A., and Pelc, A.,** 2013, Worst case optimal exploration of terrains with obstacles, *Information and Computation*, 225, 16-28.
- Czyzowicz, J., Labourel, A., and Pelc, A.,** 2010b, How to meet asynchronously (almost) everywhere, Arxiv preprint arXiv:1001.0890.
- Das, S.,** 2008, Mobile agent rendezvous in a ring using faulty tokens, In Proceedings of the 9th international conference on Distributed computing and networking, 292-297 pp. Springer-Verlag.
- Deng, X., Kameda, T., and Papadimitriou, C.,** 1991, How to learn an unknown environment, In Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on, 298-303 pp. IEEE.

KAYNAKLAR DİZİNİ (devam)

- Deng, X., Kameda, T., and Papadimitriou, C.,** 1998, How to learn an unknown environment. i: the rectilinear case, *Journal of the ACM (JACM)*, 45(2), 215-245.
- Dessmark, A., Fraigniaud, P., Kowalski, D., and Pelc, A.,** 2006, Deterministic rendezvous in graphs, *Algorithmica*, 46(1), 69-96.
- Dieudonné, Y. and Pelc, A.,** 2014, Price of asynchrony in mobile agents computing, *Theoretical Computer Science*, 524, 59-67.
- Dieudonné, Y., Pelc, A., and Villain, V.,** 2013, How to meet asynchronously at polynomial cost, In *Proceedings of the 2013 ACM symposium on Principles of distributed computing*, 92-99 pp. ACM.
- Fang, J., Morse, A., and Cao, M.,** 2008, Multi-agent rendezvousing with a finite set of candidate rendezvous points, In *American Control Conference*, 2008, 765-770 pp.
- Fekete, S. P. and Schmidt, C.,** 2010, Polygon exploration with time discrete vision, *Computational Geometry*, 43(2), 148-168.
- Flocchini, P., Kranakis, E., Krizanc, D., Luccio, F., Santoro, N., and Sawchuk, C.,** 2004, Mobile agents rendezvous when tokens fail, *Structural Information and Communication Complexity*, 161-172 pp.
- Flocchini, P., Prencipe, G., Santoro, N., and Widmayer, P.,** 2000, Distributed coordination of a set of autonomous mobile robots, In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, 480-485 pp. IEEE.
- Flocchini, P., Prencipe, G., Santoro, N., and Widmayer, P.,** 2001, Gathering of asynchronous oblivious robots with limited visibility, *STACS 2001*, 247-258 pp.
- Flocchini, P., Prencipe, G., Santoro, N., and Widmayer, P.,** 2005, Gathering of asynchronous robots with limited visibility, *Theoretical Computer Science*, 337(1), 147-168.
- Gabriely, Y. and Rimon, E.,** 2001, Spanning-tree based coverage of continuous areas by a mobile robot, *Annals of Mathematics and Artificial Intelligence*, 31(1-4), 77-98.
- Gabriely, Y. and Rimon, E.,** 2003, Competitive on-line coverage of grid environments by a mobile robot, *Computational Geometry*, 24(3), 197-224.

KAYNAKLAR DİZİNİ (devam)

- Gabriely, Y. and Rimon, E.**, 2005, Competitive complexity of mobile robot on line motion planning problems, 155-170 pp.
- Gal, S.**, 1980, Search Games, Academic Press.
- Gal, S.**, 1999, Rendezvous search on the line, Operations Research, 974-976 pp.
- Ganguli, A., Cortes, J., and Bullo, F.**, 2006, Multirobot rendezvous with visibility sensors in nonconvex environments, Arxiv preprint cs/0611022.
- Ghosh, S. K., Burdick, J. W., Bhattacharya, A., and Sarkar, S.**, 2008, Online algorithms with discrete visibility-exploring unknown polygonal environments, *Robotics & Automation Magazine*, IEEE, 15(2), 67-76.
- Hammar, M., Nilsson, B. J., and Persson, M.**, 2006, Competitive exploration of rectilinear polygons, *Theoretical computer science*, 354(3), 367-378.
- Hammar, M., Nilsson, B. J., and Schuierer, S.**, 2002, Improved exploration of rectilinear polygons, *Nordic Journal of Computing*, 9(1), 32-53.
- Han, Q., Du, D., Vera, J., and Zuluaga, L.**, 2007, Improved bounds for the symmetric rendezvous value on the line, In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, 78 p. Society for Industrial and Applied Mathematics.
- Hirose, M. and Ogawa, K.**, 2007, Honda humanoid robots development, Philosophical Transactions of the Royal Society A: Mathematical, *Physical and Engineering Sciences*, 365(1850), 11-19.
- Hole, B., Dobrev, S., Flocchini, P., and Santoro, N.**, 2009, Multiple Agents RendezVous in a Ring in Spite of a Black Hole. Howard, J., 1999, Rendezvous search on the interval and the circle, *Operations Research*, 47(4), 550-558.
- Icking, C., Kamphans, T., Klein, R., and Langetepe, E.**, 2000, Exploring an unknown cellular environment., In EuroCG, 140-143 pp.
- Jurek, C., Leszek, G., and Pelc, A.**, 2009, Gathering few fat mobile robots in the plane, *Theoretical Computer Science*, 410(6-7), 481-499.
- Kalyanasundaram, B. and Pruhs, K.**, 1993, A competitive analysis of algorithms for searching unknown scenes, *Computational Geometry*, 3(3), 139-155.

KAYNAKLAR DİZİNİ (devam)

- Kao, M., Ma, Y., Sipser, M., and Yin, Y.**, 1994, Optimal construction of hybrid algorithms, Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA '94), 372-381 pp.
- Klasing, R., Kosowski, A., and Navarra, A.**, 2008, Taking advantage of symmetries: gathering of asynchronous oblivious robots on a ring, *Principles of Distributed Systems*, 446-462 pp.
- Kleinberg, J. M.**, 1994, On-line search in a simple polygon, In Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms, 8-15 pp. Society for Industrial and Applied Mathematics.
- Kolenderska, A., Kosowski, A., Malafiejski, M., and Zylinski, P.**, 2010, An improved strategy for exploring a grid polygon, 222-236 pp.
- Kowalski, D. and Pelc, A.**, 2005, Polynomial deterministic rendezvous in arbitrary graphs, *Algorithms and Computation*, 644-656 pp.
- Kowalski, D. R. and Malinowski, A.**, 2008, How to meet in anonymous network, *Theoretical Computer Science*, 399(1), 141-156.
- Kranakis, E., Krizanc, D., and Markou, E.**, 2010, The Mobile Agent Rendezvous Problem in the Ring, *Synthesis Lectures on Distributed Computing Theory*, 1(1), 1-122.
- Kranakis, E., Krizanc, D., and Morin, P.**, 2008, Randomized rendezvous with limited memory, 605-616 pp.
- Lim, W., Alpern, S., and Beck, A.**, 1997, Rendezvous search on the line with more than two players, *Operations Research*, 45(3), 357-364.
- Lin, J., Morse, A., and Anderson, B.**, 2003, The multi-agent rendezvous problem, In Proceedings of the 42nd IEEE Conference on Decision and Control, volume 2, 1508-1513 pp.
- Lin, J., Morse, A., and Anderson, B.**, 2007, The multi-agent rendezvous problem. Part 2: The asynchronous case, *SIAM Journal on Control and Optimization*, 46(6), 2120-2147.
- Lin, Z., Francis, B., and Maggiore, M.**, 2006, Getting mobile autonomous robots to rendezvous, *Control of Uncertain Systems: Modeling, Approximation, and Design*, 119-137 pp.

KAYNAKLAR DİZİNİ (devam)

- Marco, G., Gargano, L., Kranakis, E., Krizanc, D., Pelc, A., and Vaccaro, U.**, 2006, Asynchronous deterministic rendezvous in graphs, *Theoretical Computer Science*, 355(3), 315-326.
- Ntafos, S.**, 1992, Watchman routes under limited visibility, *Computational Geometry*, 1(3), 149-170.
- Poduri, S. and Sukhatme, G.**, 2007a, Achieving connectivity through coalescence in mobile robot networks, In *Proceedings of the 1st international conference on Robot communication and coordination*, 4 p. IEEE Press.
- Poduri, S. and Sukhatme, G.**, 2007b, Latency Analysis of Coalescence for Robot Groups, In *2007 IEEE International Conference on Robotics and Automation*, 3295-3300 pp.
- Prencipe, G.**, 2005, On the feasibility of gathering by autonomous mobile robots, *Structural Information and Communication Complexity*, 246-261 pp.
- Rawicz, P., Kalata, P., and Murphy, K.**, 1998, On the stability of mobile robot rendezvous, In *Intelligent Control (ISIC)*, 1998. Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS), *Proceedings*, 570-575 pp. IEEE.
- Rekleitis, I.**, 2004, A particle filter tutorial for mobile robot localization, Technical Report, CIM-04-02, McGill University.
- Roy, N. and Dudek, G.**, 2001, Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations, *Autonomous Robots*, 11(2), 117-136.
- Souissi, S., Défago, X., and Yamashita, M.**, 2006, Gathering asynchronous mobile robots with inaccurate compasses, *Principles of Distributed Systems*, 333-349 pp.
- Squyres, S.**, 2005, *Roving mars: Spirit, Opportunity and the Exploration of the Red Planet* Hyperion Books.
- Stachowiak, G.**, 2009, Asynchronous Deterministic Rendezvous on the Line, *SOFSEM 2009: Theory and Practice of Computer Science*, 497-508 pp.

KAYNAKLAR DİZİNİ (devam)

- Thomas, L. and Hulme, P.**, 1997, Searching for targets who want to be found, *Journal of the Operational Research Society*, 48(1), 44-50.
- Thomas, L. and Pikounis, M.**, 2001, Many-player rendezvous search: Stick together or split and meet?, *Naval Research Logistics*, 48(8), 710-721.
- Uthaisombut, P.**, 2006, Symmetric rendezvous search on the line using move patterns with different lengths.
- Yamauchi, B. M.**, 2004, Packbot: a versatile platform for military robotics, In *Defense and Security*, 228-237 pp. *International Society for Optics and Photonics*.
- Yu, J., LaValle, S. M., and Liberzon, D.**, 2008, Rendezvous without coordinates, In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 1803-1808 pp. IEEE.

ÖZGEÇMİŞ

Deniz Özsoyeller

Tel. : (232) 246 4949

Fax : (232) 232-224 0909

E-mail : deniz.ozsoyeller@izmir.edu.tr,
denizozsoyeller@gmail.com

Bilgisayar Mühendisliği Bölümü

İzmir Üniversitesi

Gürsel Aksel Bulvarı, No: 14,
Üçkuyular/İzmir, Türkiye, 35350

Eğitim

- Ege Üniversitesi – Uluslararası Bilgisayar Enstitüsü
- ❖ Araştırma Bursiyeri: Minnesota Üniversitesi, Bilgisayar Bilimleri ve Mühendisliği Bölümü, Minneapolis, Amerika, (2010 - 2012)
- Doktora, (2008 – 2014)

Tez: Çok Robotlu Randevu ve Kapsama için Dağıtık **Algoritmalar** (*Ing. Distributed Algorithms for Multi Robot Rendezvous and Coverage*)

- Ege Üniversitesi – Uluslararası Bilgisayar Enstitüsü
- Yüksek Lisans, Bilgisayar Bilimleri, 2008

Tez: Kablosuz Sensör Ağlarında Senkronizasyon için Dağıtık Kapsayan Ağaç ve Kümeleme Algoritmaları (*Ing. Distributed Spanning Tree And Clustering Algorithms For Synchronization In Wireless Sensor Networks*)

- İzmir Ekonomi Üniversitesi, Türkiye
- Lisans, Bilgisayar Mühendisliği, 2006

Akademik Görevler

- **Araştırma Görevlisi.** Bilgisayar Mühendisliği Bölümü, İzmir Üniversitesi (2009 – present)
- **Öğretim Görevlisi.** Meslek Yüksekokulu, Bilgisayar Programcılığı, İzmir Üniversitesi (2008 – 2009)
- **Araştırma Görevlisi.** Bilgisayar Mühendisliği Bölümü, İzmir Ekonomi Üniversitesi (2006 - 2008)

Öğretim Deneyimi

İzmir Ekonomi Üniversitesi:

- Computer Networks. Bahar 2007, Summer 2007 (Araştırma Görevlisi).
- C# .NET Programming. Bahar 2007, Güz 2006, Bahar 2006 (Araştırma Görevlisi).
- Introduction to C programming . Güz 2007 (Araştırma Görevlisi).
- Introduction to Computer Science. Güz 2007 (Araştırma Görevlisi).

İzmir Üniversitesi:

- Algorithms and Programming I (C programming language). Güz 2008. (Öğretim Görevlisi)
- Algorithms and Programming II (C++ programming language). Bahar 2008. (Öğretim Görevlisi)
- Algorithms and Programming I (Racket programming language). Güz 2009, Güz 2013, Güz 2014 (Araştırma Görevlisi).
- Introduction to Computer Science. Güz 2009 (Araştırma Görevlisi).
- Algorithms and Programming II (C programming language) Bahar 2009 (Araştırma Görevlisi).
- Discrete Structures. Bahar 2009 (Notlandırma).
- Web Programming. Bahar 2009 (Araştırma Görevlisi).
- Object Oriented Programming (Java programming language) Güz 2013 (Araştırma Görevlisi).
- Data Structures and Algorithms (Java). Güz 2013, Güz 2014 (Araştırma Görevlisi).
- Algorithms and Programming II (C++, Java programming language) . Bahar 2013 (Araştırma Görevlisi).
- Data Organization and Management. Bahar 2013 (Araştırma Görevlisi).
- Computer Network Programming. Bahar 2013 (Araştırma Görevlisi).
- Parallel Programming . Güz 2014 (Notlandırma).

Yayınlar

Makale Yayınları

- Ozsoyeller, D., Beveridge, A., and Isler V., July 2013, Symmetric Rendezvous Search on the Line with an Unknown Initial Distance, IEEE Transactions on Robotics, 99, pp. 1-14.
- Erciyes, K., Dagdeviren, O., Cokuslu, D., and Ozsoyeller, D., 2007, Graph theoretic clustering algorithms in mobile ad hoc networks and wireless sensor networks. Appl. Comput. Math, 6(2), pp. 162-180.

Konferans Yayınları (Hakemli)

- Ozsoyeller, D., Isler V. and Beveridge A., 2012, Symmetric Rendezvous in Planar Environments. Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12). (Kabul Oranı: = 26%, 1129 makale içinden 294 makale seçilmiştir)
- Erciyes, K., Ozsoyeller, D., and Dagdeviren, O., 2008, Distributed algorithms to form cluster based spanning trees in wireless sensor networks. In Computational Science-ICCS 2008, pp. 519-528. Baharer Berlin Heidelberg.
- Ozsoyeller, D., Erciyes, K., 2008, A Hierarchical Architecture to Implement a γ Synchronizer in Wireless Sensor Networks, IEEE 23rd International Symposium on Computer and Information Sciences. ISCIS 2008, 1 – 6 pp..
- Ozsoyeller, D., Erciyes, K., 2008, Design and Implementation of a Gamma Synchronizer for Wireless Sensor Networks (in Turkish), 16th IEEE Conference in Signal Processing, Communication and Applications, Didim.

Teknik Raporlar

- Andrew Beveridge, Deniz Ozsoyeller and Volkan Isler. Symmetric Linear Rendezvous with an Unknown Initial Distance. Technical Report, University of Minnesota-Computer Science and Engineering, 2011 (Report Number: 11-008).

Ödüller

- İzmir Ekonomi Üniversitesi, Bilgisayar Mühendisliği Bölümü Birincisi (2006)
- İzmir Ekonomi Üniversitesi, Lisans Eğitim Bursu
- TÜBİTAK Kariyer 104E064 Proje Bursu (2007 - 2008)
- TÜBİTAK Uluslararası Doktora Araştırma Bursu (2010)
- Lorentz Center Workshop Daveti, "Search and Rendezvous" , Leiden University, the Netherlands, Organizers: Steve Alpern (2012)
- TÜBİTAK Yayın Teşvik: 5000TL (2013)