



EGE ÜNİVERSİTESİ



E. Ü. FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

**EN YAKIN NOKTA PROBLEMLERİ İÇİN
PROJEKSİYON TABANLI YAKLAŞIMLAR
ÜZERİNE BİR ÇALIŞMA**

Aycan VARGÜN

Tez Danışmanı : Prof. Dr. Mehmet Emin DALKILIÇ

İkinci Danışmanı : Yrd. Doç. Dr. Orhan DAĞDEVİREN

Uluslararası Bilgisayar Anabilim Dalı

Bilim Dalı Kodu : 619.02.04

Sunuş Tarihi : 19.07.2013

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(YÜKSEK LİSANS TEZİ)

**EN YAKIN NOKTA PROBLEMLERİ İÇİN
PROJEKSİYON TABANLI YAKLAŞIMLAR
ÜZERİNE BİR ÇALIŞMA**

Aycan VARGÜN

Tez Danışmanı : Prof. Dr. Mehmet Emin DALKILIÇ

İkinci Danışmanı : Yrd. Doç. Dr. Orhan DAĞDEVİREN

Uluslararası Bilgisayar Anabilim Dalı

Bilim Dalı Kodu : 619.02.04

Sunuş Tarihi : 19.07.2013

Bornova-İZMİR

2013

Aycan Vargün tarafından yüksek lisans tezi olarak sunulan “En Yakın Nokta Problemleri İçin Projeksiyon Tabanlı Yaklaşımlar Üzerine Bir Çalışma” 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 19 temmuz 2013 tarihinde yapılan tez savunma sınavında aday oybirliği/oyçokluğu ile başarılı bulunmuştur.

Jüri Üyeleri:

İmza

Jüri Başkanı	: Prof. Dr. Mehmet Emin Dalkılıç	
Raportör Üye	: Prof. Dr. Bahar Karaoğlu	
Üye	: Doç. Dr. Hasan Oğul	

ÖZET

EN YAKIN NOKTA PROBLEMLERİ İÇİN PROJEKSİYON TABANLI YAKLAŞIMLAR ÜZERİNE BİR ÇALIŞMA

VARGÜN, Aycan

Yüksek Lisans Tezi, Uluslararası Bilgisayar Anabilim Dalı

Tez Danışmanı: Prof. Dr. Mehmet Emin DALKILIÇ

İkinci Danışmanı: Yrd. Doç. Dr. Orhan DAĞDEVİREN

Temmuz 2013, 212 sayfa

Bu tezde en yakın çift problemi için yeni bir çözüm öneren yakın zamanda sunulmuş QuickCP algoritması (Dalkılıç ve Ergun, 2012) tekdüze, üstel ve normal dağılımlarda incelenmiştir. Veri noktaları çizgisel sıralanma sergilediğinde QuickCP algoritmasının performansı önemli ölçüde düştüğünden, QuickCP algoritmasının rastgeleleştirilmiş QuickCP adında yeni bir sürümü geliştirilmiştir. QuickCP algoritmasının rasgeleştirme süreci, Gram-Schmidt metoduyla elde edilen ortonormal baz vektörler kullanmaktadır.

QuickCP ve rastgeleleştirilmiş QuickCP algoritmaları, noktaların en yakın çiftini bulmak için klasik Böl ve Fethet algoritması ile Kaba Kuvvet algoritmasına karşı çok boyutlu uzaylarda ve farklı dağılımlarda test edilmiştir. QuickCP ve Kaba Kuvvet algoritmalarının çok boyutlu uzaylara genişletilmesi kolay iken, klasik Böl ve Fethet algoritması ile rastgeleleştirilmiş QuickCP algoritmalarının çok boyutlu uzaylara genişletilmesi bu çalışmanın bir parçası olarak geliştirilmiştir.

Geniş ölçüde yapılan karşılaştırmalar 2, 3, 4 ve 5 boyutlu öklit uzaylarında rastgeleleştirilmiş QuickCP algoritmasının çizgisel sıralamalı dağılımlar dâhil tüm dağılımlarda rakiplerinden daha iyi performans sergilediğini göstermektedir.

Anahtar sözcükler: En yakın çift problemi, böl ve fethet, olasılık dağılımı, rastgeleleştirilmiş algoritma, çok boyutlu uzay.

ABSTRACT**A STUDY ON THE PROJECTION BASED APPROACH TO THE
CLOSEST POINT PROBLEMS**

VARGÜN, Aycan

MSc in Information Technologies

Supervisor: Prof. Dr. Mehmet Emin DALKILIÇ

Co-Supervisor: Asst. Prof. Dr. Orhan DAĞDEVİREN

July 2013, 212 pages

In this thesis, recently introduced QuickCP algorithm (Dalkilic and Ergun, 2012) which proposes a new solution to the closest pair of points problem is studied under uniform, exponential and normal distributions. Since QuickCP algorithm's performance degrades significantly when input points exhibit linear alignments, a randomized version of the algorithm called randomized QuickCP is developed. QuickCP algorithm's randomization process uses orthonormal base vectors that are obtained by Gram-Schmidt method.

QuickCP and randomized QuickCP algorithms are tested against the classic Divide and Conquer and Brute Force algorithms for finding the closest pair of points under different input distributions on multidimensional spaces. Though extension of QuickCP and Brute Force algorithms to multi-dimensional space is straightforward, multi-dimensional versions of the classic Divide and Conquer and the randomized QuickCP algorithms are also developed as part of this work.

Extensive comparisons show that in 2, 3, 4 and 5 dimensional euclidean spaces randomized QuickCP algorithm performs better than its competitors under all input distributions including those with linear alignments.

Keywords: Closest pair problem, divide and conquer, probability distribution, randomized algorithm, multidimension space.

TEŞEKKÜR

Bu çalışma süresince benden görüşlerini esirgemeyen, bana yol gösteren, fikir veren değerli hocam Prof. Dr. Mehmet Emin Dalkılıç'a, bu projeden beni haberdar eden ve bana aktarılmasını sağlayan değerli hocam Yrd. Doç. Dr. Orhan Dağdeviren'e ve tüm eğitim hayatım boyunca bana destek olmuş olan canım aileme teşekkürü borç bilirim.

İÇİNDEKİLER

	<u>Sayfa</u>
ÖZET	v
ABSTRACT	vii
TEŞEKKÜR	ix
ŞEKİLLER DİZİNİ	xv
ÇİZELGELER DİZİNİ	xxvii
1. GİRİŞ	1
2. $\mathbb{R}^2$ UZAYINDA QUICKCP ANALİZİ	2
2.1 En Yakın Çift Problemi	2
2.2 Mevcut Çözümler	3
2.2.1 Kaba kuvvet algoritması	3
2.2.2 Böl ve fethet algoritması	4
2.3 $\mathbb{R}^{2^2}$ de QuickCP	5
2.4 Çeşitli Nokta Kümelerinde Algoritmaların Çalışma Sürelerinin	
Karşılaştırılması	8
2.4.1 Algoritmaların üstel dağılımlı kümelerde çalışma süreleri	8
2.4.2 Algoritmaların normal dağılımlı nokta kümelerinde çalışma süreleri	13
2.4.3 Algoritmaların tekdüze dağılımlı nokta kümelerinde çalışma süreleri	18

İÇİNDEKİLER (devam)

	<u>Sayfa</u>
2.5 Çizgisel Yığılım Sorunu ve Çözümü	23
2.5.1 QuickCP algoritmasında bitiş koşulu	23
2.5.2 Çizgisel yığılım sorunu	26
2.5.3 QuickCP algoritmasının BFA'nın gerisine düştüğü koşulları bulma	27
2.5.4 Nokta kümesindeki çizgisel yığılımı belirlemede histogram kullanımı	30
2.5.5 Üstel dağılımlı nokta kümelerinde çizgisel yığılım	32
2.5.6 Normal dağılımlı nokta kümelerinde çizgisel yığılım	36
2.5.7 Tekdüze dağılımlı nokta kümelerinde çizgisel yığılım	40
2.6 Rastgeleleştirilmiş QuickCP Algoritması	44
2.7 Rastgeleleştirilmiş QuickCP Algoritması ve Diğer Algoritmaların Çalışma..... Süresi Sonuçları	50
2.7.1 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın üstel dağılımlı kümelerde.... çalışma süreleri	51
2.7.2 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın normal dağılımlı nokta..... kümelerinde çalışma süreleri	56
2.7.3 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın tekdüze dağılımlı nokta..... kümelerinde çalışma süreleri	61
2.8 Nokta Kümesindeki Çizgisel Birikim Durumuna Göre Algoritma Seçimi ...	66
2.8.1 Histogram ile rastgeleştirme işlemleri	67

İÇİNDEKİLER (devam)

	<u>Sayfa</u>
2.8.2 Üstel dağılımlı kümelerde algoritma seçimi	68
2.8.3 Normal dağılımlı kümelerde algoritma seçimi.....	71
2.8.4 Tekdüze dağılımlı kümelerde algoritma seçimi	74
3. YÜKSEK BOYUTLU UZAYLARDA QUICKCP ANALİZİ	78
3.1 QuickCP'yi Yüksek Boyutlu Uzaylara Uyarlama.....	78
3.2 Yüksek Boyutlarda QuickCP Rastgeleleştirme.....	78
3.3 QuickCP Algoritması ile Karşılaştırılacak Algoritmaların Uyarlaması	83
3.3.1 KKA'nın yüksek boyutlu uzaylara uyarlanması	83
3.3.2 BFA'nın yüksek boyutlu uzaylara uyarlanması	84
3.4 Çeşitli Boyutlarda Çalışma Süresi Sonuçları	85
3.4.1 $\mathbb{R}^3$ 'te algoritmaların çalışma süresi sonuçları	86
3.4.2 $\mathbb{R}^4$ 'te algoritmaların çalışma süresi sonuçları	96
3.4.3 $\mathbb{R}^5$ 'te algoritmaların çalışma süresi sonuçları	106
3.4.4 $\mathbb{R}^8$ 'de algoritmaların çalışma süresi sonuçları.....	117
3.4.5 $\mathbb{R}^{12}$ 'de algoritmaların çalışma süresi sonuçları	130
3.4.6 $\mathbb{R}^{17}$ 'de algoritmaların çalışma süresi sonuçları	143
4. SONUÇLAR.....	156

İÇİNDEKİLER (devam)Sayfa

4.1	Algoritma Çalıştırma Süresi Sonuçları ve Analizleri	156
4.2	Algoritmaların Çalışma Süresi Sonuçları Üzerine Yorum.....	162
4.3	Tezin Katkıları.....	164
KAYNAKLAR DİZİNİ		165
ÖZGEÇMİŞ		167
EKLER		

ŞEKİLLER DİZİNİ

<u>Şekil</u>	<u>Sayfa</u>
2.1 KKA (Michigan Technological University'den, 2013)	4
2.2 Özyinelemeli böl ve fethet algoritması (Pereira and Lobo'dan, 2010).....	5
2.3 QuickCP algoritması kodu (Dalkilic and Ergun'dan, 2012)	6
2.4 Örnek nokta kümesi ve sıralama (Dalkilic and Ergun'dan, 2012)	7
2.5 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların	
çalışması.....	9
2.6 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların	
çalışması.....	10
2.7 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların	
çalışması.....	11
2.8 1000 elemanlı üstel dağılımlı nokta kümesi.....	12
2.9 10 000 elemanlı üstel dağılımlı nokta kümesi.....	12
2.10 100 000 elemanlı üstel dağılımlı nokta kümesi	13
2.11 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların.....	
çalışması.....	14
2.12 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların.....	
çalışması.....	15
2.13 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların.....	
çalışması.....	16
2.14 1000 elemanlı normal dağılımlı nokta kümesi	17

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
2.15 10 000 elemanlı normal dağılımlı nokta kümesi	17
2.16 100 000 elemanlı normal dağılımlı nokta kümesi	18
2.17 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların.... çalışması.....	19
2.18 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların.... çalışması.....	20
2.19 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların.... çalışması.....	21
2.20 1000 elemanlı tekdüze dağılımlı nokta kümesi	22
2.21 10 000 elemanlı tekdüze dağılımlı nokta kümesi	22
2.22 100 000 elemanlı tekdüze dağılımlı nokta kümesi	23
2.23 Örnek çizgisel yığılım Koridor-1 ve Koridor-2.....	26
2.24 40 elemanlı nokta kümesinin X ve Y koordinatları.....	28
2.25 Bir küme üzerinde örnek histogram.....	31
2.26 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların	33
2.27 Çizgisel yığılım içeren 1000 elemanlı üstel dağılımlı nokta kümesi.....	34
2.28 Çizgisel yığılım içeren 10 000 elemanlı üstel dağılımlı nokta kümesi.....	35

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
2.29 Çizgisel yığılım içeren 100 000 elemanlı üstel dağılımlı nokta kümesi	36
2.30 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması.....	37
2.31 Çizgisel yığılım içeren 1000 elemanlı normal dağılımlı nokta kümesi	38
2.32 Çizgisel yığılım içeren 10 000 elemanlı normal dağılımlı nokta kümesi	39
2.33 Çizgisel yığılım içeren 100 000 elemanlı normal dağılımlı nokta kümesi ...	40
2.34 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması.....	41
2.35 Çizgisel yığılım içeren 1000 elemanlı tekdüze dağılımlı nokta kümesi	42
2.36 Çizgisel yığılım içeren 10 000 elemanlı tekdüze dağılımlı nokta kümesi	43
2.37 Çizgisel yığılım içeren 100 000 elemanlı tekdüze dağılımlı nokta kümesi ..	44
2.38 Çizgisel Yığılım içeren örnek küme	45
2.39 Yeni koordinat sistemi örneği.....	46
2.40 Noktaların yeni koordinat sistemine izdüşümü	47
2.41 Yeni koordinat sisteminde yeni koordinatlar	48

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
2.42 Örnek kümenin rastgeleleştirme öncesi ve sonrası histogramı	49
2.43 Üstel dağılımlı çizgisel yığılım içeren 1000 elemanlı küme	52
2.44 Üstel dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü	53
2.45 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması	54
2.46 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması	55
2.47 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması	56
2.48 Normal dağılımlı çizgisel yığılım içeren 1000 elemanlı küme	57
2.49 Normal dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü	58
2.50 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması	59
2.51 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması	60
2.52 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması	61
2.53 Tekdüze dağılımlı çizgisel yığılım içeren 1000 elemanlı küme	62

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
2.54 Tekdüze dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü.....	63
2.55 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması.....	64
2.56 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması.....	65
2.57 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması.....	66
2.58 Birleştirilmiş QuickCP algoritması kodu.....	67
2.59 Histogram sözde kodu.....	68
2.60 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP	69
2.61 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP	70
2.62 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP	71
2.63 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP	72
2.64 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP	73
2.65 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP	74

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
2.66 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP	75
2.67 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP	76
2.68 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP	77
3.1 $\mathbb{R}^3$ 'te basis vektörlerinin sağlaması gereken kurallar (Lautrup'tan, 2011)	79
3.2 X ve X' pozisyonu ilişkisi (Lautrup'dan, 2011)	80
3.3 X' pozisyonunun bulunuşu (Lautrup'dan, 2011)	80
3.4 $\mathbb{R}^2$ uzayında 10 elemanlı kümede noktaların eski ve yeni koordinatları.....	82
3.5 Çevrilmiş koordinat hesaplama örneği	82
3.6 $\mathbb{R}^3$ uzayında 10 elemanlı kümede noktaların eski ve yeni koordinatları.....	83
3.7 k boyutta KKA.....	84
3.8 Seyreklik koşulu (ing. sparsity condition)..... (Bentley and Shamos'dan, 1976)	85
3.9 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	87
3.10 $\mathbb{R}^3$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve QuickCP'nin çalışma süreleri	88

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.11 $\mathbb{R}^3$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	89
3.12 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA,.. BFA ve QuickCP'nin çalışma süreleri.....	90
3.13 $\mathbb{R}^3$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	91
3.14 $\mathbb{R}^3$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	92
3.15 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	93
3.16 $\mathbb{R}^3$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	94
3.17 $\mathbb{R}^3$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	95
3.18 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	97
3.19 $\mathbb{R}^4$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve QuickCP'nin çalışma süreleri.....	98
3.20 $\mathbb{R}^4$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	99
3.21 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA,.. BFA ve QuickCP'nin çalışma süreleri.....	100

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.22 $\mathbb{R}^4$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	101
3.23 $\mathbb{R}^4$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	102
3.24 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	103
3.25 $\mathbb{R}^4$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	104
3.26 $\mathbb{R}^4$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	105
3.27 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	107
3.28 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	108
3.29 $\mathbb{R}^5$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve QuickCP'nin çalışma süreleri.....	109
3.30 $\mathbb{R}^5$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ... ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	110
3.31 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA,.. BFA ve QuickCP'nin çalışma süreleri.....	111
3.32 $\mathbb{R}^5$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	112

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.33 $\mathbb{R}^5$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	113
3.34 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri	114
3.35 $\mathbb{R}^5$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	115
3.36 $\mathbb{R}^5$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	116
3.37 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	118
3.38 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	119
3.39 $\mathbb{R}^8$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA... ve QuickCP'nin çalışma süreleri.....	120
3.40 $\mathbb{R}^8$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA... ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	121
3.41 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri	122
3.42 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	123
3.43 $\mathbb{R}^8$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	124

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.44 $\mathbb{R}^8$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	125
3.45 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri	126
3.46 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	127
3.47 $\mathbb{R}^8$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	128
3.48 $\mathbb{R}^8$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	129
3.49 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	131
3.50 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	132
3.51 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA . ve QuickCP'nin çalışma süreleri	133
3.52 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA . ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	134
3.53 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	135
3.54 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	136

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.55 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	137
3.56 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	138
3.57 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	139
3.58 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	140
3.59 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	141
3.60 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	142
3.61 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	144
3.62 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	145
3.63 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA,..... BFA ve QuickCP'nin çalışma süreleri.....	146
3.64 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA,..... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	147
3.65 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	148

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
3.66 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde..... KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri.....	149
3.67 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	150
3.68 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP.....	151
3.69 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve QuickCP'nin çalışma süreleri	152
3.70 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde..... KKA, BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	153
3.71 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve QuickCP'nin çalışma süreleri.....	154
3.72 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, ... BFA ve rastgeleştirilmiş QuickCP'nin çalışma süreleri	155

ÇİZELGELER DİZİNİ

<u>Çizelge</u>	<u>Sayfa</u>
2.1 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.5).....	9
2.2 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.6).....	10
2.3 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.7).....	11
2.4 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.11).....	14
2.5 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.12).....	15
2.6 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.13).....	16
2.7 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.17).....	19
2.8 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.18).....	20
2.9 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.19).....	21
2.10 BFA'nın QuickCP'yi geçtiği durum tablosu	29
2.11 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.26).....	33
2.12 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.30).....	37
2.13 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.34).....	41
2.14 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.45).....	54
2.15 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.46).....	55
2.16 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.47).....	56
2.17 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.50).....	59

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
2.18 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.51).....	60
2.19 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.52).....	61
2.20 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.55).....	64
2.21 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.56).....	65
2.22 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.57).....	66
2.23 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.60).....	69
2.24 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.61).....	70
2.25 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.62).....	71
2.26 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.63).....	72
2.27 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.64).....	73
2.28 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.65).....	74
2.29 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.66).....	75
2.30 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.67).....	76
2.31 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.68).....	77
3.1 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.9).....	87
3.2 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.10).....	88
3.3 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.11).....	89

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
3.4 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.12).....	90
3.5 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.13).....	91
3.6 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.14).....	92
3.7 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.15).....	93
3.8 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.16).....	94
3.9 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.17).....	95
3.10 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.18).....	97
3.11 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.19).....	98
3.12 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.20).....	99
3.13 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.21).....	100
3.14 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.22).....	101
3.15 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.23).....	102
3.16 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.24).....	103
3.17 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.25).....	104
3.18 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.26).....	105
3.19 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.27).....	107
3.20 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.28).....	108

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
3.21 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.29).....	109
3.22 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.30).....	110
3.23 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.31).....	111
3.24 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.32).....	112
3.25 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.33).....	113
3.26 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.34).....	114
3.27 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.35).....	115
3.28 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.36).....	116
3.29 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.37).....	118
3.30 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.38).....	119
3.31 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.39).....	120
3.32 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.40).....	121
3.33 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.41).....	122
3.34 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.42).....	123
3.35 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.43).....	124
3.36 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.44).....	125
3.37 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.45).....	126

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
3.38 $\mathbb{R}^{8^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.46).....	127
3.39 $\mathbb{R}^{8^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.47).....	128
3.40 $\mathbb{R}^{8^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.48).....	129
3.41 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.49).....	131
3.42 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.50).....	132
3.43 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.51).....	133
3.44 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.52).....	134
3.45 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.53).....	135
3.46 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.54).....	136
3.47 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.55).....	137
3.48 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.56).....	138
3.49 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.57).....	139
3.50 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.58).....	140
3.51 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.59).....	141
3.52 $\mathbb{R}^{12^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.60).....	142
3.53 $\mathbb{R}^{17^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.61).....	144
3.54 $\mathbb{R}^{17^c}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.62).....	145

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
3.55 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.63).....	146
3.56 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.64).....	147
3.57 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.65).....	148
3.58 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.66).....	149
3.59 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.67).....	150
3.60 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.68).....	151
3.61 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.69).....	152
3.62 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.70).....	153
3.63 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.71).....	154
3.64 $\mathbb{R}^{17'}$ de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.72).....	155
4.1 $\mathbb{R}^2$ uzayında sonuçlar	156
4.2 $\mathbb{R}^2$ 'de sonuçların analizi.....	156
4.3 $\mathbb{R}^3$ uzayında sonuçlar	157
4.4 $\mathbb{R}^3$ 'te sonuçların analizi.....	157
4.5 $\mathbb{R}^4$ uzayında sonuçlar	158
4.6 $\mathbb{R}^4$ 'te sonuçların analizi.....	158
4.7 $\mathbb{R}^5$ uzayında sonuçlar	159

ÇİZELGELER DİZİNİ (devam)

<u>Çizelge</u>	<u>Sayfa</u>
4.8 $\mathbb{R}^5$ 'te sonuçların analizi	159
4.9 $\mathbb{R}^8$ uzayında sonuçlar	160
4.10 $\mathbb{R}^{8''}$ 'te sonuçların analizi	160
4.11 $\mathbb{R}^{12}$ uzayında sonuçlar	161
4.12 $\mathbb{R}^{12}$ 'de sonuçların analizi	161
4.13 $\mathbb{R}^{17}$ uzayında sonuçlar	162
4.14 $\mathbb{R}^{17}$ 'de sonuçların analizi	162

1. GİRİŞ

Bu tez geometri terimleri kullanılarak tanımlanan algoritmaların çalışmasını inceleyen bir bilgisayar bilimi dalı olan hesaplamalı geometrinin problemlerinden en yakın çift problemine (ing. closest pair of points) dair yeni bir çözümü inceleyen çalışmalar içermektedir. En yakın çift problemine yeni bir çözüm olarak QuickCP algoritması sunulmaktadır (Dalkilic and Ergun, 2012). QuickCP algoritmasının çeşitli dağılımlarda davranışları gözlemlenmiştir. Daha sonra QuickCP, rastgeleştirilmiştir. QuickCP kelimesinde “CP” uzantısı, “closest pair” kelimelerinin kısaltılışıdır.

En yakın çift probleminin bugünkü çözümlerinden olan Böl ve Fethet Algoritması (kısaca BFA) (ing. Divide and Conquer Algorithm) ile tüm olasılıkları deneyerek çözüm getiren Kaba Kuvvet Algoritması (kısaca KKA) (ing. Brute Force Algorithm) çeşitli boyutlarda ve çeşitli dağılımlara sahip nokta kümeleri kullanılarak QuickCP algoritması ile karşılaştırılmıştır.

Algoritmaların çalışma davranışları başlangıçta iki boyutlu uzayda yani $\mathbb{R}^2$ 'de incelenmiştir. QuickCP algoritmasının performansını olumsuz etkileyen durumlar gözlemlenmiş, çözümü için çalışmalar yapılmıştır. Algoritmalar daha sonra herhangi bir $k \geq 2$ ve $k \in \mathbb{N}$ boyutundaki $\mathbb{R}^k$ uzayında çalışacak şekilde geliştirilmiştir. Uzay boyutu ve nokta kümelerinin büyüklüğü değiştirilirken, algoritmaların davranışlarındaki değişiklikler görüntülenmiş ve yorumlanmıştır. Yüksek boyutlu uzaylarda QuickCP algoritmasının performansını olumsuz etkileyen durumlar tekrar gözlemlenmiş, çözümü için daha önce $\mathbb{R}^2$ 'de yapılan çalışmalar bu uzaylara uyarlanmıştır.

Algoritmaların test sonuçlarını içeren grafiklerdeki her değer için algoritmalar 5'er kez çalıştırılmış, sonuçların ortalaması alınarak grafiklere eklenmiştir.

Algoritmaların $\mathbb{R}^2$, $\mathbb{R}^3$, $\mathbb{R}^4$, $\mathbb{R}^5$, $\mathbb{R}^8$, $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ uzaylarındaki uygulamalarında kullanılan dil C, kodun derlendiği ortam Microsoft Visual Studio 2008'dir. Kodun çalıştırıldığı bilgisayarın özellikleri 4GB RAM, 320GB HDD, Intel(R) CORE(TM)2 Duo CPU T9600 @2.80 GHz 2.80 GHz işlemcili, 32 bit Windows Vista işletim sistemli Packard Bell marka bilgisayardır. Algoritmaların çalışma sürelerini gösteren grafikler hariç tüm grafikler Wolfram Mathematica 9 kullanılarak, nokta kümelerinin gerçek ölçüleriyle hazırlanmıştır.

2. $\mathbb{R}^2$ UZAYINDA QUICKCP ANALİZİ

2.1 En Yakın Çift Problemi

Bu problem, N adet nokta içeren bir düzlemde nokta çiftleri arasında birbirine en yakın olan nokta çiftini bulmayı amaçlar. Birden fazla en yakın nokta çifti olması durumunda problemin çözümü için bu çiftlerden herhangi birini bulmak çözüm için yeterlidir.

Shamos (1978) en yakın komşu problemini şu şekilde tanımlamıştır:

“(En yakın çift) Bir düzlemde N nokta verilsin, birlikte en yakın olan ikisini bulun. Birden fazla en yakın çift olabilir. Bu probleme çözüm olarak herhangi birinin bulunması farz edilebilir.”

En yakın çifti bulma probleminin çözümü için kullanılan algoritmaları sınamak adına algoritmaların kullanabileceği farklı dağılımlara sahip veri kümeleri gerekmektedir. Bu kümeler, algoritmaların incelendiği uzay boyutuna uygun noktalardan oluşmalıdır. Algoritmaların kullandığı tüm noktalar dört yolla elde edilmiştir.

- Üstel dağılımlı nokta kümeleri: Mao’a (2013) ait veri tabanında “dbs/documents/long” ve “dbs/documents/short” klasörleri bulunmaktadır. “long” klasöründeki dosyalar birleştirilerek 16 520 347 ondalık sayı bulunmuş, aynı olanlar elenerek her biri birbirinden farklı 10 727 886 ondalık sayı elde edilmiştir. “short” klasöründeki dosyalar birleştirilerek 9 196 537 ondalık sayı bulunmuş, aynı olanları elenerek her biri birbirinden farklı 6 597 602 ondalık sayı elde edilmiştir. Buradan elde edilen sayılar üstel dağılımlı olmakla birlikte $\mathbb{R}^2$, $\mathbb{R}^3$, $\mathbb{R}^4$, $\mathbb{R}^5$, $\mathbb{R}^8$, $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ ’de nokta üretimi için kullanılmıştır. Ancak tezde yalnızca “long” klasöründen elde edilen sayılar kullanılmıştır. Ek 1’de “long” klasöründen elde edilen tüm noktaların histogramı yer almaktadır. Histogramdaki “Nokta Sayısı” ifadesi, “Nokta Aralığı” eksenindeki bir değerde ne kadar nokta olduğunu göstermektedir.

- $\mathbb{R}^2$ ’de Normal dağılımlı nokta kümeleri: İki boyutlu nokta üretimi yapan bir uygulama kullanılarak çeşitli boyutlarda iki boyutlu ve normal dağılımlı nokta kümeleri oluşturulmuştur (Zaiane and Pei, 2013). Buradan elde edilen sayılar normal dağılımlı ve 3 000 000 adet olmakla birlikte, sadece 2 boyutlu uzayda yani $\mathbb{R}^2$ ’de nokta üretimi için kullanılmıştır. Ek 2’de bu veri kümesine ait üretilen tüm

noktaların histogramı yer almaktadır. Histogramdaki “Nokta Sayısı” ifadesi, “Nokta Aralığı” eksenindeki bir değerde ne kadar nokta olduğunu göstermektedir.

- Tekdüze dağılımlı nokta kümeleri: Java dilinde `java.lang.Math.random()` fonksiyonu ile tekdüze dağılımlı ondalık sayılar üretilmiştir. Üretilen ondalık sayılar `java.util.Arrays.sort()` fonksiyonuyla sıralanmıştır. Sıralandıktan sonra birbiriyle aynı olan ondalık sayılar olup olmadığı kontrol edilmiştir. Aynı olan ondalık sayılar elenmiştir. Buradan elde edilen sayılar tekdüze (uniform) dağılımlı ve 5 000 000 adet olmakla birlikte $\mathbb{R}^2$, $\mathbb{R}^3$, $\mathbb{R}^4$, $\mathbb{R}^5$, $\mathbb{R}^8$, $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ ’de nokta üretimi için kullanılmıştır. Ek 3’te bu veri kümesine ait üretilen tüm noktaların histogramı yer almaktadır. Histogramdaki “Nokta Sayısı” ifadesi, “Nokta Aralığı” eksenindeki bir değerde ne kadar nokta olduğunu göstermektedir.

- Normal dağılımlı nokta kümeleri: Burada Java dili kullanılarak `java.util.Random.nextNormal()` fonksiyonu ile normal dağılımlı ondalık sayılar üretilmiştir. Üretilen ondalık sayılar `java.util.Arrays.sort()` fonksiyonuyla sıralanmıştır. Sıralandıktan sonra birbiriyle aynı olan ondalık sayılar olup olmadığı kontrol edilmiştir. Aynı olan ondalık sayılar elenmiştir. Buradan elde edilen sayılar normal dağılımlı ve 5 000 000 adet olmakla birlikte $\mathbb{R}^3$, $\mathbb{R}^4$, $\mathbb{R}^5$, $\mathbb{R}^8$, $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ ’de nokta üretimi için kullanılmıştır. Ek 4’te bu veri kümesine ait üretilen tüm noktaların histogramı yer almaktadır. Histogramdaki “Nokta Sayısı” ifadesi, “Nokta Aralığı” eksenindeki bir değerde ne kadar nokta olduğunu göstermektedir.

2.2 Mevcut Çözümler

En yakın çift problemi için bu güne kadar birçok çözüm sunulmuştur. Sunulan çözümlerden QuickCP algoritması ile çalışma davranışlarını karşılaştırmak için KKA ve BFA seçilmiştir.

2.2.1 Kaba kuvvet algoritması

KKA, bilgisayar biliminde kullanılan bir problem çözme tekniğidir. Bu teknikte problemin önermelerine uyan tüm olası çözümler teker teker incelenir ve çözüm bulunmaya çalışılır. Bu anlamda en yakın çift problemi için kullanılacak olan bu algoritma, bir nokta kümesindeki tüm nokta çiftlerini incelemeli ve birbirine en yakın çifti bu şekilde bulmaya çalışmalıdır. En yakın çifti bulmayı

amaçlayan KKA için gereken kod Michigan Technological University (2013)'e göre Şekil 2.1'dedir.

```

Algorithm BruteForceClosestPoints (P)

// P is list of points

 $d_{min} \leftarrow \infty$ 

for  $i \leftarrow 1$  to  $n-1$  do

    for  $j \leftarrow i+1$  to  $n$  do

         $d \leftarrow \text{sqrt}((x_i - x_j)^2 + (y_i - y_j)^2)$ 

        if  $d < d_{min}$  then

             $d_{min} \leftarrow d$ ;  $\text{index}_1 \leftarrow i$ ;  $\text{index}_2 \leftarrow j$ 

return  $\text{index}_1, \text{index}_2$ 

```

Şekil 2.1 KKA (Michigan Technological University'den, 2013)

Bu koda göre bir P kümesindeki her nokta çifti incelenir ve kümedeki en yakın iki nokta bulunur. Kodun her aşamasında noktalar arasındaki minimum uzaklığı taşıyan değişken yenilenir. Algoritmanın çalışması tamamlandığında bu değişkende tüm nokta çiftleri arasında en yakın olanının uzaklık değeri vardır.

2.2.2 Böl ve fethet algoritması

Özyinelemeli böl ve fethet metodu birçok problem için yaygın olarak kullanılmakla birlikte en yakın çift problemi için de bu durum geçerlidir. Bentley ve Shamos (1976)'a göre XY düzleminde $n \geq 2$ tane nokta içeren bir P kümesinde, P 'deki en yakın nokta çifti $O(n \log_2 n)$ karmaşıklıkta bulunabilir (Pereira and Lobo, 2010). Şekil 2.2'de böl ve fethet algoritmasının kodu yer almaktadır.

CLOSEST-PAIR(P)

- 1 Presort points in P along the x-coordinate.
- 2 Split the ordered set P into two equal-sized^a subsets by the vertical line l , defined by the equation $x = x_{median}$.
- 3 Solve the problem recursively in the left and right subsets. This will give the left-side and right-side minimal distances d_L and d_R , respectively.
- 4 Find the minimal distance d_{LR} among the pair of points in which one point lies on the left of the dividing vertical line and the other point lies to the right.
- 5 The final answer is the minimum between d_L , d_R , and d_{LR} .

^agive or take (exactly) one point, when the number of points is odd.

Şekil 2.2 Özyinelemeli böl ve fethet algoritması (Pereira and Lobo'dan, 2010)

Ön işlem olarak P kümesindeki noktalar X koordinatına göre sıralanmalıdır. Algoritmanın kod olarak uygulamaya aktarılması sırasında sıralama işlemi için $O(n \log_2 n)$ karmaşıklığı olan Merge Sort algoritması kullanılmıştır (Wayne, 2005; Cormen et al., 2009). Noktalar sıralandıktan sonra P kümesi bir L doğrusuyla eşit sayıda nokta içeren iki kümeye bölünür. Bu bölme işlemi özyinelemeli olarak devam eder. Oluşan kümelerin her birinde problem çözülür. Her bir kümede en yakın nokta çifti bulunur ve aralarındaki uzaklıklar (d_L ve d_R) saklanır. Daha sonra, birbirine komşu her kümenin, onları ayıran L doğrusunun iki yanında yer alan ve birbirine uzaklığı $d_{\min} = \min(d_L, d_R)$ olmak üzere $d_{\min}$ 'den küçük eşit d_{LR} uzaklığına sahip nokta çiftleri bulunur.

Algoritmanın daha ayrıntı kodu ise Rosetta Code (2013)'a göre Ek 5'te yer almaktadır.

2.3 $\mathbb{R}^2$ 'de QuickCP

QuickCP algoritması, iki boyutlu düzlemde birbirine yakın iki noktanın yakınlık durumunun, noktaların X veya Y koordinat eksenindeki izdüşümlerine büyük olasılıkla yansıtacağını iddia eder. Eğer iki nokta birbirine yakınsa örneğin X koordinatları birbirine uzak olsa da Y koordinatları birbirine yakındır. Dalkılıç ve Ergun (2012) bu konuda şöyle demektedir:

“Bu, eğer iki nokta birbirine yakınsa, noktaların X ve/veya Y eksenlerine sıralı izdüşümleri bu yakınlığı büyük ihtimalle yansıtır, varsayımına dayanır.”

Şekil 2.3'te QuickCP algoritmasının kodu yer almaktadır. QuickCP'ye göre $n \in \mathbb{N}$ adet nokta içeren P kümesindeki her bir noktanın önce X sonra da Y

eksenine izdüşümü yapılır. Yani P kümesindeki noktaların X ve Y koordinatları elde edilir. Ardından, P kümesindeki noktalar önce X sonra da Y koordinatlarına göre sıralanır ve A ile B olarak iki farklı yeni diziye kopyalanır. Sıralama işlemi için burada da $O(n \log_2 n)$ karmaşıklığı olan Merge Sort algoritması kullanılmıştır (Wayne, 2005; Cormen et al., 2009).

```

Sort P by y-values, store in B i.e., {B(1), B(2), ..., B(n)}

dmin = ∞

for r = 1 to n-1 do ; at most n-1 rounds

    ΔXmin = ΔYmin = ∞

    for i = 1 to n-r do

        ΔX = A(i+r).x-A(i).x ; A(j).x is the x-value of A(j)

        ΔY = A(i+r).y-A(i).y ; A(j).y is the y-value of A(j)

        d = sqrt(ΔX2 + ΔY2) ; distance between A(i) and A(i+r)

        if ΔX < ΔXmin then ; update ΔXmin

            ΔXmin = ΔX

        if d < dmin then dmin = d ; update dmin

        ΔX = B(i+r).x-B(i).x

        ΔY = B(i+r).y-B(i).y

        d = sqrt(ΔX2 + ΔY2) ; distance between B(i) and B(i+r)

        if ΔY < ΔYmin then ; update ΔYmin

            ΔYmin = ΔY

        if d < dmin then dmin = d ; update dmin

    end for

    if dmin ≤ sqrt(ΔXmin2 + ΔYmin2) then

        break ; termination condition

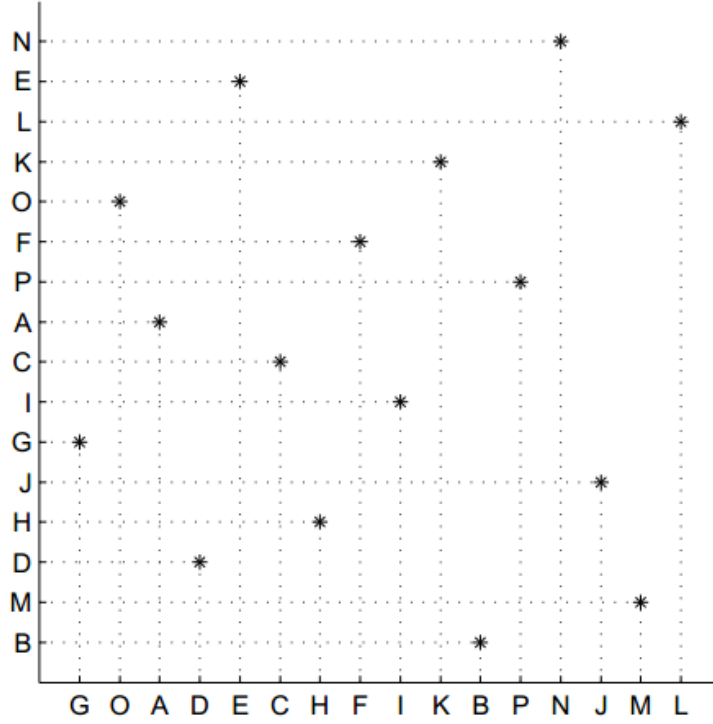
end for

```

Şekil 2.3 QuickCP algoritması kodu (Dalkilic and Ergun'dan, 2012)

Merge Sort ile elde edilen X koordinatına göre sıralı noktalardan oluşan A dizisi ve Y koordinatına göre sıralı noktalardan oluşan B dizisi, noktaların X ve Y

eksenlerine izdüşümü yapıldığında elde edilecek koordinatların sıralamasını içerir. Daha sonra sırayla önce birbirlerine bir nokta uzaklıktaki nokta çiftleri, ardından birbirine iki nokta uzaklıktaki nokta çiftleri ve işlem devam ettikçe en son $n - 1$ uzaklıktaki nokta çifti incelenir. Algoritmanın çalışma şekli, Şekil 2.4'te yer alan nokta kümesi üzerinde gösterilmiştir.



Şekil 2.4 Örnek nokta kümesi ve sıralama (Dalkilic and Ergun'den, 2012)

Bu örnekte $A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P$ noktalarının X ile Y koordinatları gösterilmektedir.

Algoritma şu şekilde çalışmaktadır:

Noktalar X eksenine izdüşümü yapıp X koordinatlarına göre sıralandığında $G, O, A, D, E, C, H, F, I, K, B, P, N, J, M, L$ şeklinde; Y eksenine izdüşümü yapıp Y koordinatlarına göre sıralandığında $B, M, D, H, J, G, I, C, A, P, F, O, K, L, E, N$ şeklinde sıralamalar oluşmaktadır. Bu noktaların koordinatları arasında herhangi bir değere yığılma ya da birbirleri üstüne çakışma bulunmamaktadır. Her bir aşamada en küçük koordinat farkları ΔX_{min} ve ΔY_{min} bulunur, bulunan değerler bitiş koşulunda kullanılarak d_{min} bulunmaya çalışılır (Bkz. Şekil 2.3).

Önce X koordinatındaki sıralamaya göre bir nokta uzaklıktaki (G, O) , (O, A) , (A, D) , (D, E) , (E, C) , (C, H) , (H, F) , (F, I) , (I, K) , (K, B) , (B, P) , (P, N) , (N, J) , (J, M) , (M, L) nokta ikilileri kullanılır. Bu ikililerin arasındaki mesafeler hesaplanır. Daha sonra aynı işlem Y koordinatı için yapılır. Y koordinatındaki sıralamaya göre yine bir nokta uzaklıktaki (B, M) , (M, D) , (D, H) , (H, J) , (J, G) , (G, I) , (I, C) , (C, A) , (A, P) , (P, F) , (F, O) , (O, K) , (K, L) , (L, E) , (E, N) nokta ikilileri kullanılır. Bu ikililerin arasındaki mesafeler hesaplanır. Bu birinci aşamadır. Birinci aşamada elde edilen ΔX_{min} ve ΔY_{min} değerleri ile d_{min} değeri bitiş koşulunda kullanılır. Henüz döngüden çıkılmamışsa ikinci aşamaya geçilir. Bu aşamada iki nokta uzaklıktaki nokta ikilileri hesaplanır. X koordinatındaki sıralamaya göre (G, A) , (O, D) , (A, E) , (D, C) , (E, H) , (C, F) , (H, I) , (F, K) , (I, B) , (K, P) , (B, N) , (P, J) , (N, M) , (J, L) nokta ikilileri kullanılır. Y koordinatındaki sıralamaya göre (B, D) , (M, H) , (D, J) , (H, G) , (J, I) , (G, C) , (I, A) , (C, P) , (A, F) , (P, O) , (F, K) , (O, L) , (K, E) , (L, N) nokta ikilileri kullanılır. Bu işlem bitiş koşuluna göre n tane nokta için $(n - 1)$. aşamaya kadar devam edebilir. Bitiş koşuluna uygun nokta çifti bulunduğunda döngüden erken çıkarılır.

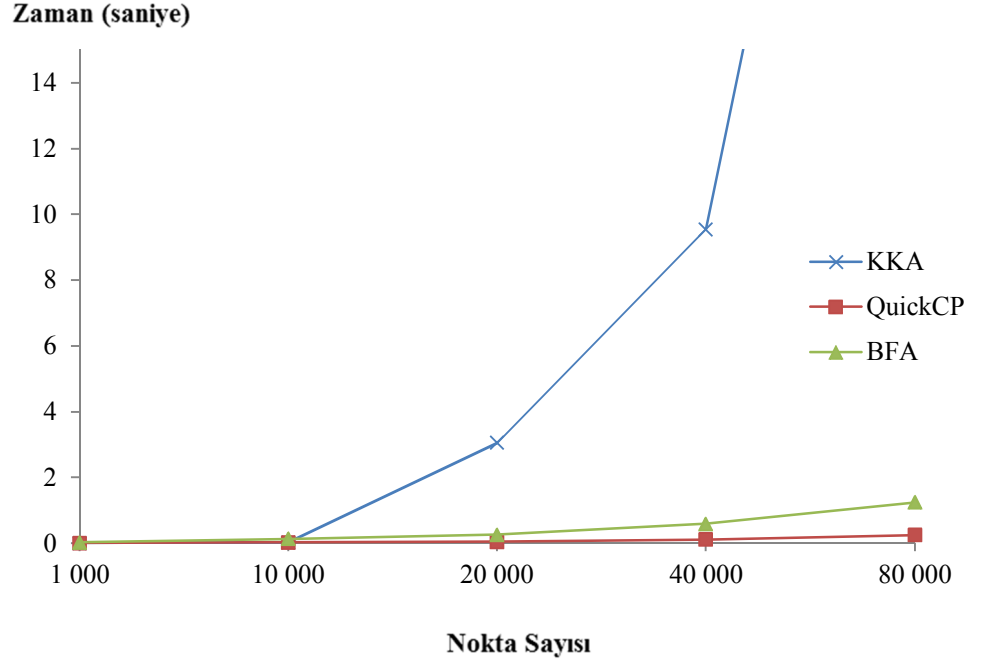
2.4 Çeşitli Nokta Kümelerinde Algoritmaların Çalışma Sürelerinin Karşılaştırılması

KKA, BFA ile QuickCP algoritması, üstel, normal ve tekdüze dağılımlı sayılarla doldurulmuş nokta kümeleri üzerinde çalıştırılmıştır. Her küme için o kümedeki en yakın iki nokta bulunmuştur. Algoritmaların en yakın nokta çiftlerini bulurken harcadıkları süre hesaplanmıştır. Nokta kümelerinin eleman sayıları farklıdır. Eleman sayısı arttıkça algoritmaların çalışma süresi de artmaktadır. Çalışma sürelerine QuickCP algoritması için gerekli ön işlem olan sıralama işlemi süreleri ve BFA için gerekli ön işlem olan sıralama işlemi süresi eklenmiştir.

2.4.1 Algoritmaların üstel dağılımlı kümelerde çalışma süreleri

Bu bölümde veri tabanından elde edilen üstel dağılımlı nokta kümeleri üzerinde çalıştırılan algoritmaların sonuçları Şekil 2.5, Şekil 2.6 ve Şekil 2.7'de grafik olarak gösterilmiştir. Grafiklerin değerleri Tablo 2.1, Tablo 2.2 ve Tablo 2.3 yer almaktadır. Bu nokta kümelerinde çalışma süresi sırasına göre iyiden kötüye doğru sıralama QuickCP, BFA, KKA olmuştur. En iyi çalışma süresi QuickCP algoritmasına aittir. Onu BFA izlemiştir. En kötü çalışma süresi ise KKA'ya aittir. QuickCP algoritması, BFA'dan daha hızlı çalışmıştır. KKA'nın

çalışma süresi 80 000 elemanlı kümeden sonra çok fazla arttığı için hesaplamalara katılmamıştır.

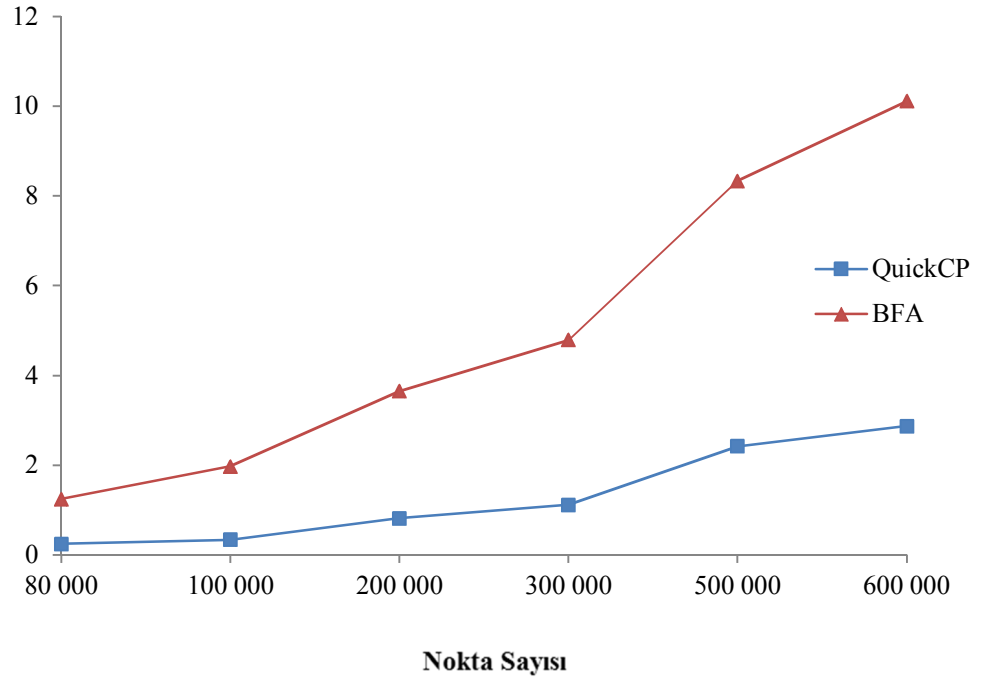


Şekil 2.5 $\mathbb{R}^{2'}$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların çalışması

Tablo 2.1 $\mathbb{R}^{2'}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.5)

Nokta Sayısı	KKA(sn)	QuickCP (sn)	BFA(sn)
1000	0.0118	0.0032	0.0300
10 000	0.0300	0.0286	0.1306
20 000	3.0534	0.0496	0.2630
40 000	9.5446	0.1176	0.5966
80 000	40.204	0.2466	1.2436

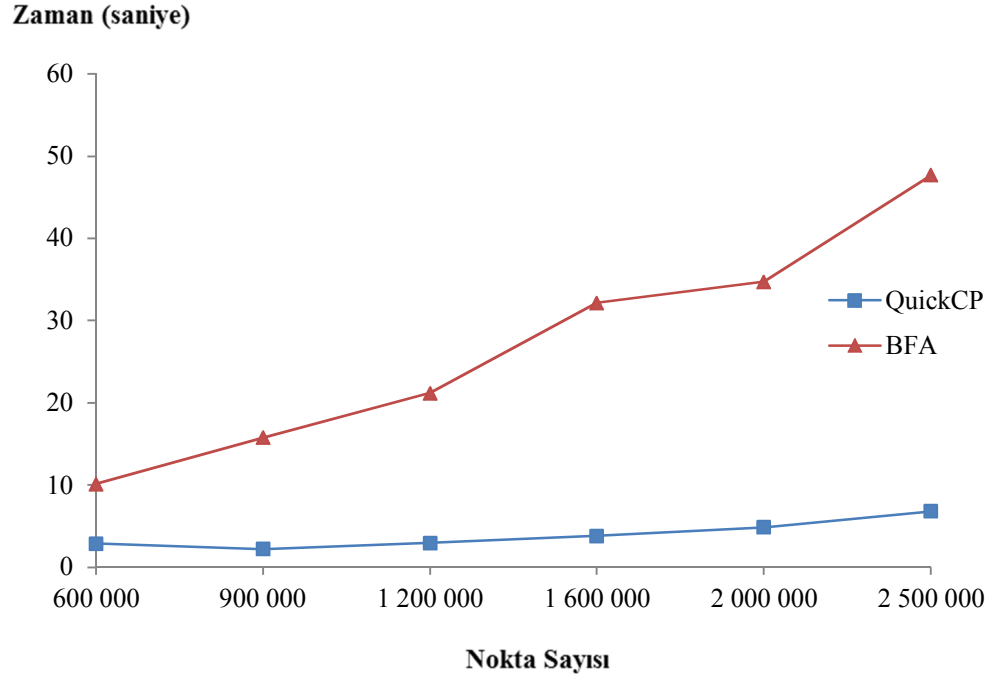
Zaman (saniye)



Şekil 2.6 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların çalışması

Tablo 2.2 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.6)

Nokta Sayısı	QuickCP (sn)	BFA(sn)
80 000	0.2466	1.2436
100 000	0.3396	1.9722
200 000	0.8172	3.6516
300 000	1.1202	4.7908
500 000	2.4210	8.3380
600 000	2.8704	10.1138

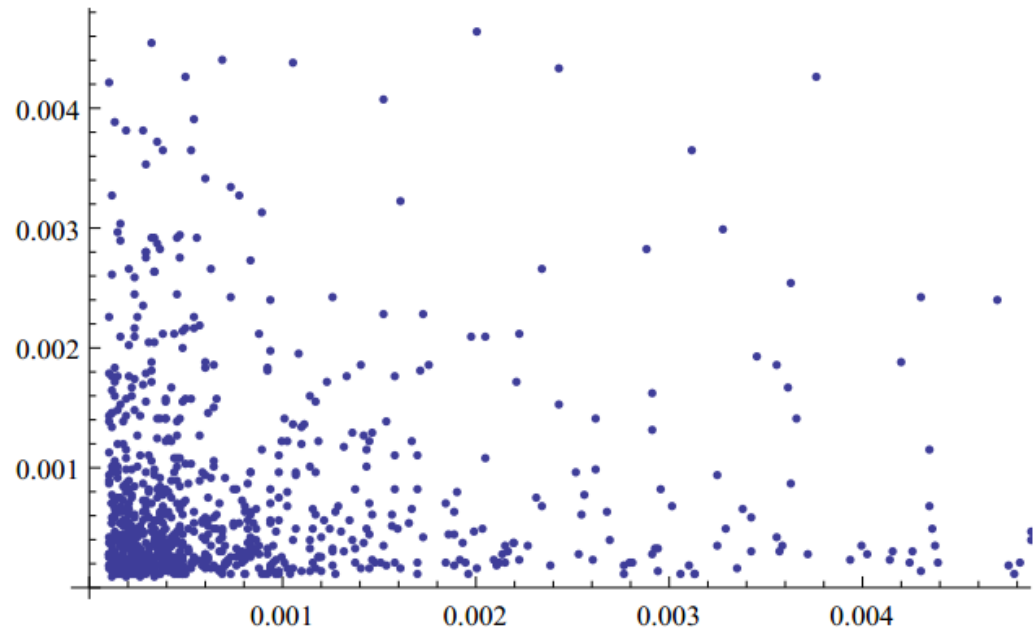


Şekil 2.7 $\mathbb{R}^{2^2}$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede algoritmaların çalışması

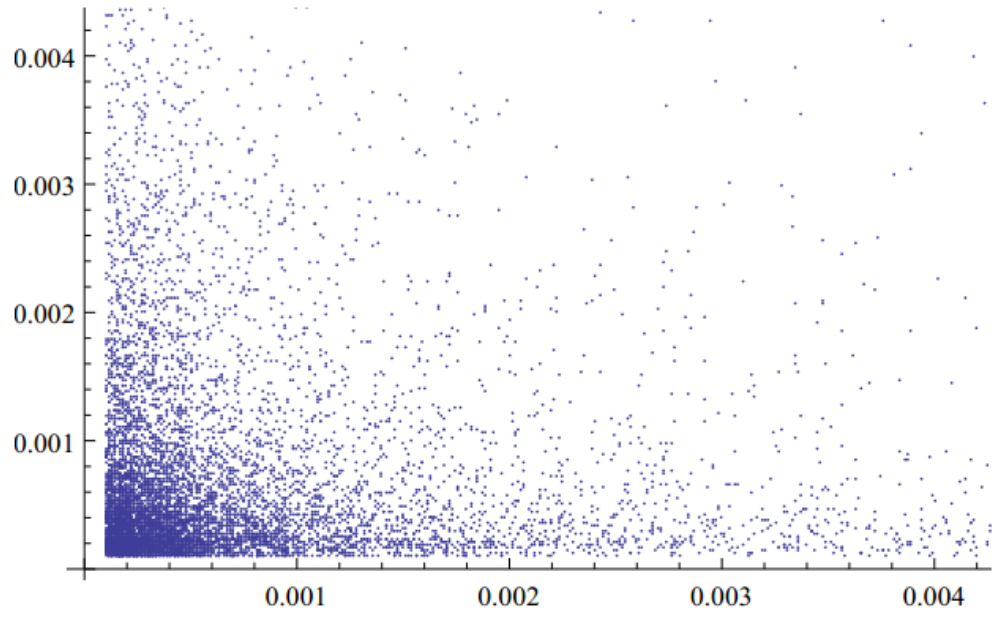
Tablo 2.3 $\mathbb{R}^{2^2}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.7)

Nokta Sayısı	QuickCP (sn)	BFA(sn)
600 000	2.8704	10.1138
900 000	2.2156	15.7762
1 200 000	2.9480	21.1768
1 600 000	3.8150	32.1648
2 000 000	4.8798	34.7100
2 500 000	6.7996	47.6990

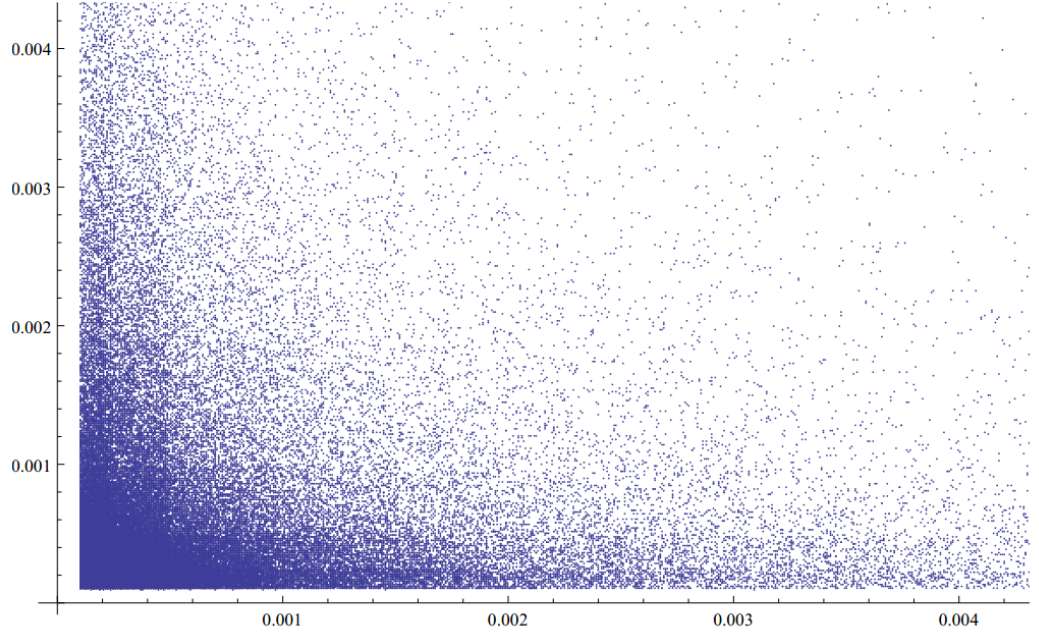
Üstel dağılımlı noktalar çeşitli boyutlarda gruplandırılarak nokta kümeleri oluşturulmuştur. 1000 elemanlı nokta kümesinden başlanarak 2 500 000 elemanlı nokta kümesine kadar algoritmalar çalıştırılmıştır. Bu nokta kümeleri 1000, 10 000 ve 100 000 elemanlı olduğunda noktaların nasıl dağılım yaptığı Şekil 2.8, Şekil 2.9 ve Şekil 2.10'de görüntülenmiştir.



Şekil 2.8 1000 elemanlı üstel dağılımlı nokta kümesi



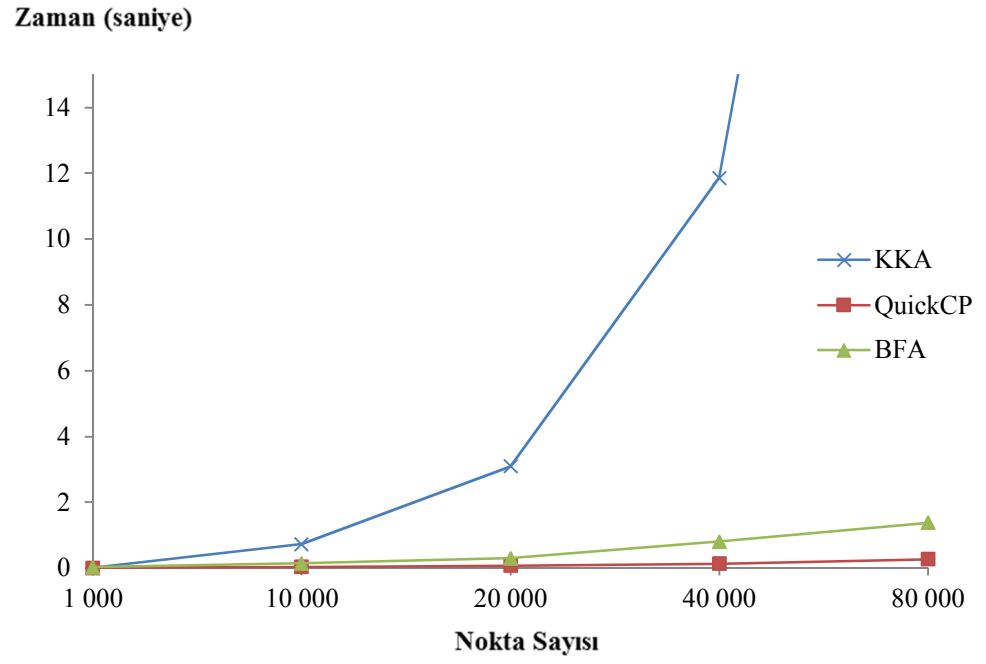
Şekil 2.9 10 000 elemanlı üstel dağılımlı nokta kümesi



Şekil 2.10 100 000 elemanlı üstel dağılımlı nokta kümesi

2.4.2 Algoritmaların normal dağılımlı nokta kümelerinde çalışma süreleri

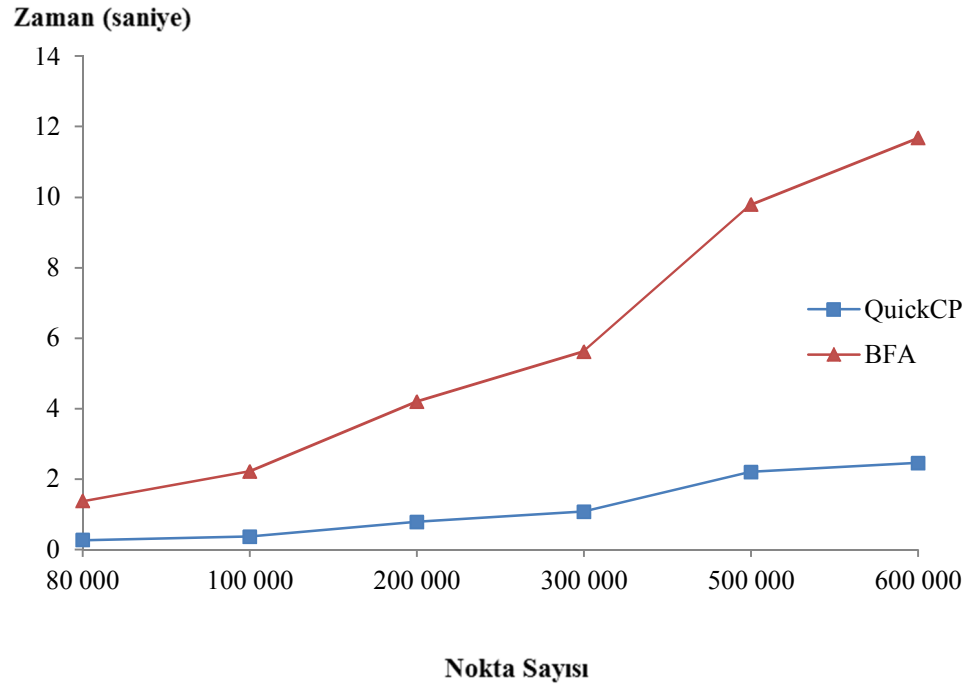
Bu bölümde iki boyutlu nokta üreten uygulama kullanılarak elde edilen normal dağılımlı nokta kümeleri üzerinde çalışan algoritmaların sonuçları Şekil 2.11, Şekil 2.12 ve Şekil 2.13'te grafik olarak gösterilmiştir. Grafiklerin değerleri Tablo 2.4, Tablo 2.5 ve Tablo 2.6'da yer almaktadır. Bu nokta kümelerinde algoritmaların çalışma sürelerine göre iyiden kötüye doğru sıralama QuickCP, BFA, KKA olmuştur. En iyi çalışma süresi QuickCP algoritmasına aittir. Onu BFA izlemiştir. En kötü çalışma süresi ise KKA'ya aittir. QuickCP algoritması, BFA'dan daha hızlı çalışmıştır. KKA'nın çalışma süresi 80 000 elemanlı kümeden sonra çok fazla arttığı için hesaplamalara katılmamıştır.



Şekil 2.11 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların çalışması

Tablo 2.4 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.11)

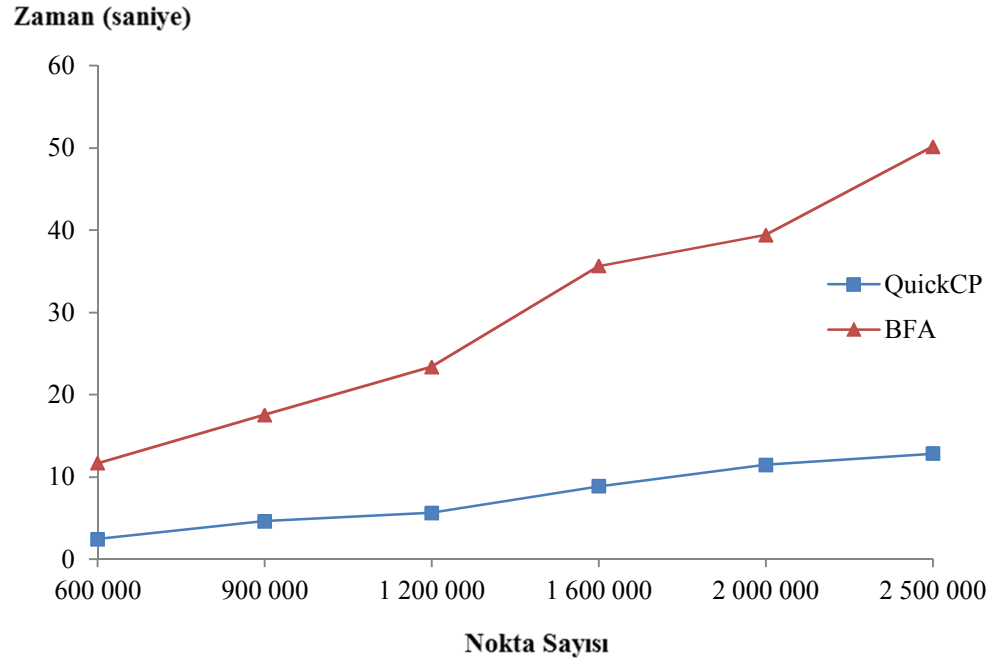
Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0064	0.0030	0.0246
10 000	0.7280	0.0352	0.1462
20 000	3.0882	0.0712	0.2998
40 000	11.8540	0.1366	0.8018
80 000	46.3778	0.2700	1.3740



Şekil 2.12 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların çalışması

Tablo 2.5 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.12)

Nokta Sayısı	QuickCP(sn)	BFA(sn)
80 000	0.2700	1.3740
100 000	0.3658	2.2246
200 000	0.7844	4.2018
300 000	1.0742	5.6196
500 000	2.2064	9.7864
600 000	2.4590	11.6790

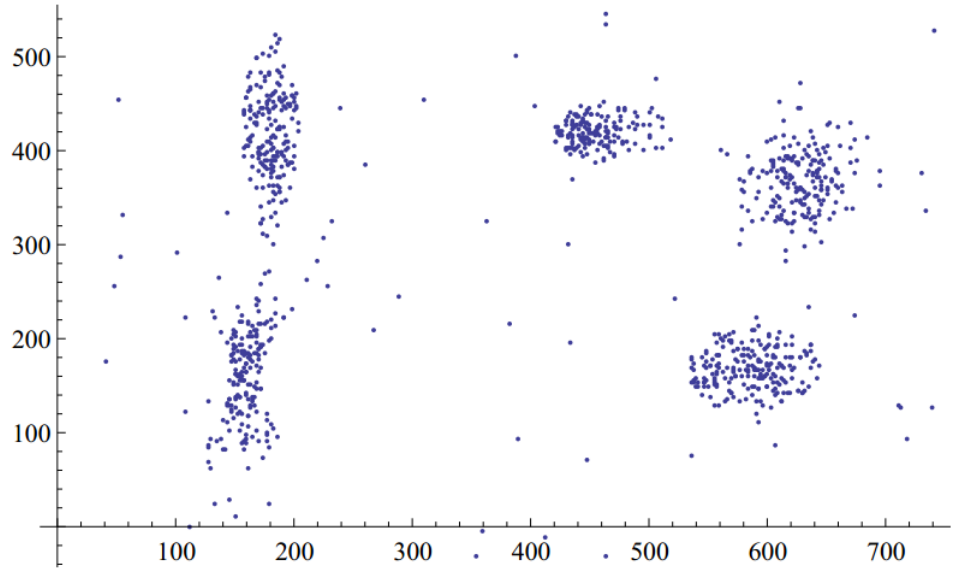


Şekil 2.13 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede algoritmaların çalışması

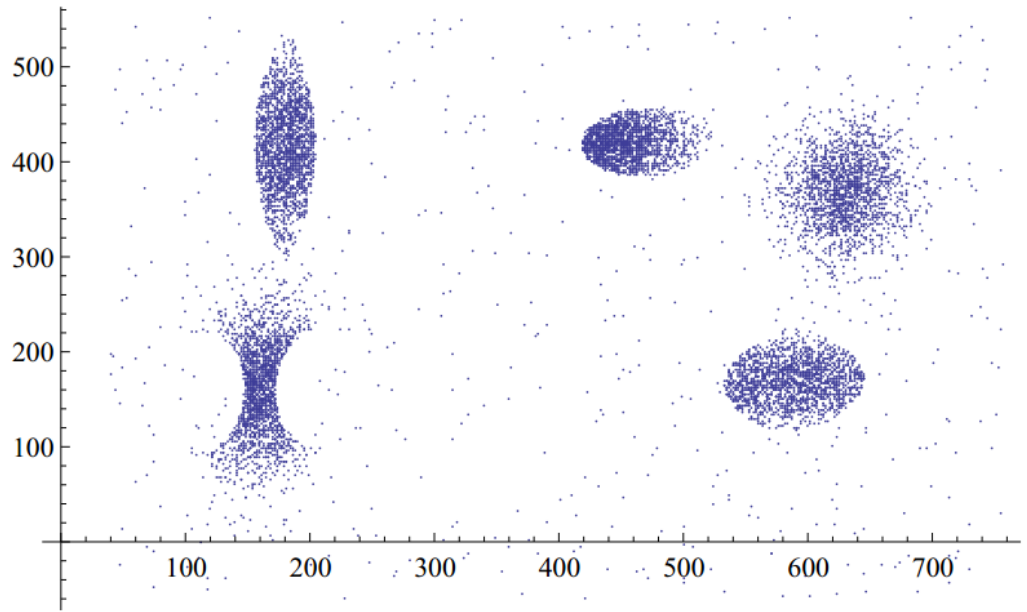
Tablo 2.6 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.13)

Nokta Sayısı	QuickCP(sn)	BFA(sn)
600 000	2.4590	11.6790
900 000	4.6238	17.5530
1 200 000	5.6184	23.3792
1 600 000	8.8828	35.6598
2 000 000	11.4690	39.4362
2 500 000	12.8532	50.1868

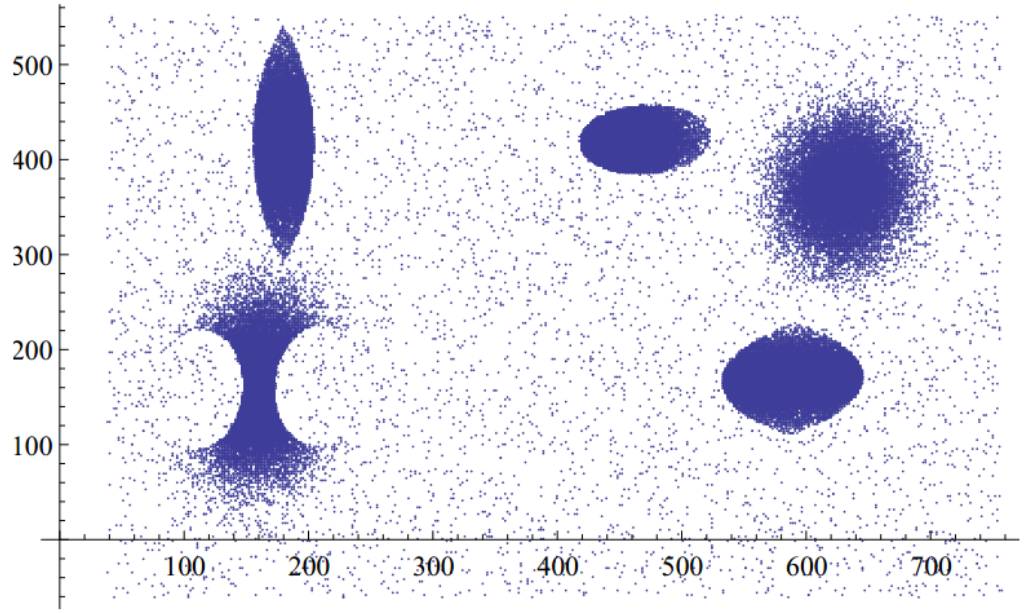
Nokta üretimi yapan uygulama kullanılarak elde edilen noktalar çeşitli boyutlarda gruplandırılarak nokta kümeleri oluşturulmuştur. 1000 elemanlı nokta kümesinden başlanarak 2 500 000 elemanlı nokta kümesine kadar algoritmalar çalıştırılmıştır. Bu nokta kümeleri 1000, 10 000 ve 100 000 elemanlı olduğunda noktaların nasıl dağılım yaptığı Şekil 2.14, Şekil 2.15 ve Şekil 2.16'da görüntülenmiştir.



Şekil 2.14 1000 elemanlı normal dağılımlı nokta kümesi



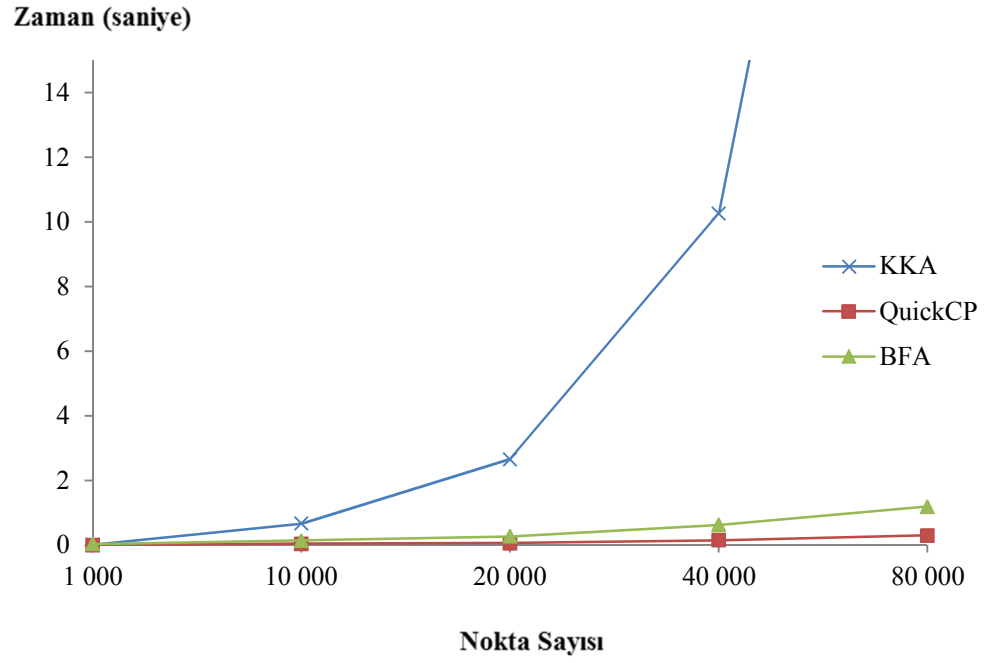
Şekil 2.15 10 000 elemanlı normal dağılımlı nokta kümesi



Şekil 2.16 100 000 elemanlı normal dağılımlı nokta kümesi

2.4.3 Algoritmaların tekdüze dağılımlı nokta kümelerinde çalışma süreleri

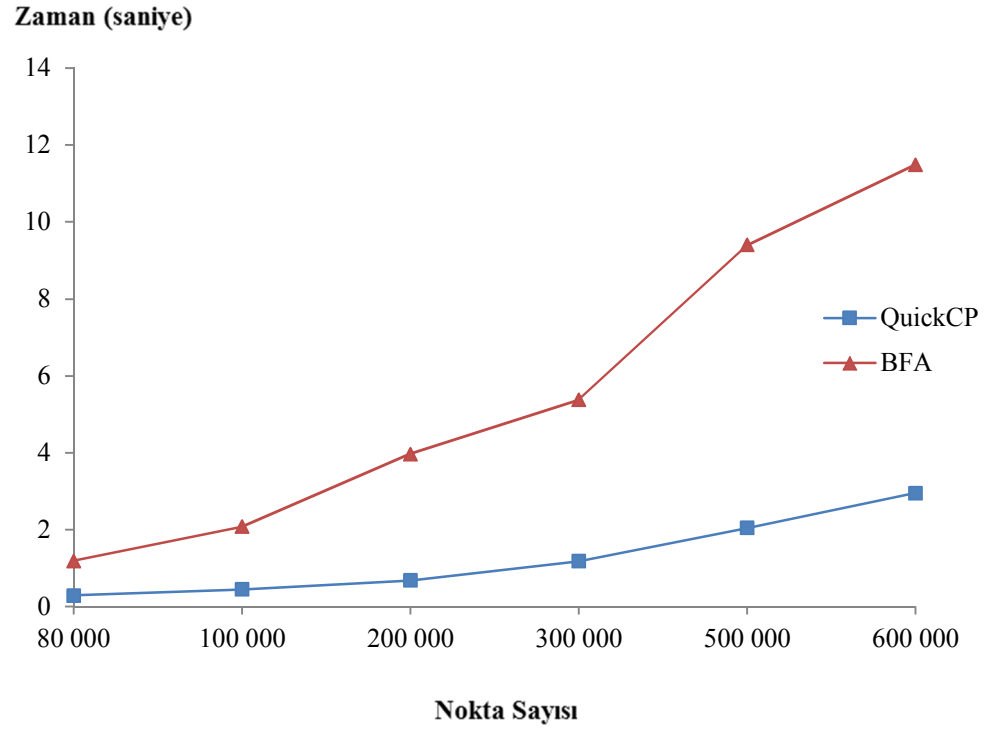
Bu bölümde Java'nın `java.lang.Math.random()` fonksiyonu ile elde edilen sayılarla oluşturulan nokta kümeleri üzerinde çalışan algoritmaların çalışma sürelerini gösteren grafikler Şekil 2.17, Şekil 2.18 ve Şekil 2.19'da gösterilmiştir. Grafiklerin değerleri Tablo 2.7, Tablo 2.8 ve Tablo 2.9'da yer almaktadır. Tekdüze dağılımlı nokta kümelerinde çalışma süresi sırasına göre iyiden kötüye doğru sıralama QuickCP, BFA, KKA olmuştur. En iyi çalışma süresi QuickCP algoritmasına aittir. Onu BFA izlemiştir. En kötü çalışma süresi ise KKA'ya aittir. QuickCP algoritması, BFA'dan daha hızlı çalışmıştır. KKA'nın çalışma süresi 80 000 elemanlı kümeden sonra çok fazla arttığı için hesaplamalara katılmamıştır.



Şekil 2.17 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların çalışması

Tablo 2.7 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.17)

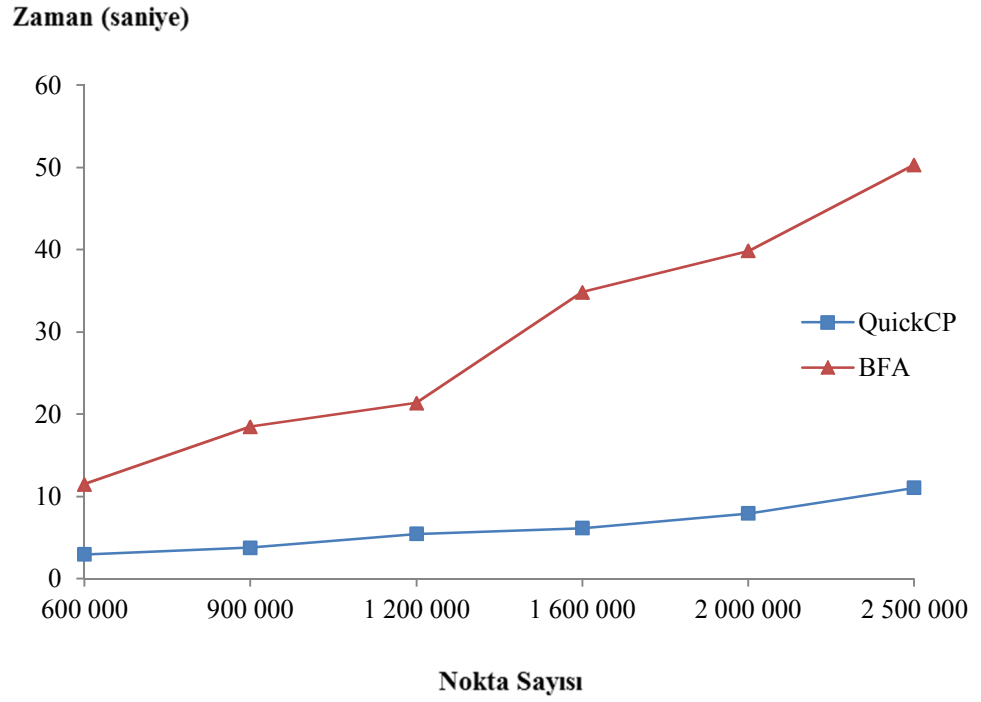
Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0132	0.0042	0.0294
10 000	0.6640	0.0362	0.1386
20 000	2.6590	0.0604	0.2650
40 000	10.2614	0.1506	0.6218
80 000	42.6212	0.2922	1.1926



Şekil 2.18 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların çalışması

Tablo 2.8 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.18)

Nokta Sayısı	QuickCP(sn)	BFA(sn)
80 000	0.2922	1.1926
100 000	0.4512	2.0830
200 000	0.6862	3.9662
300 000	1.1830	5.3752
500 000	2.0444	9.4058
600 000	2.9568	11.4890

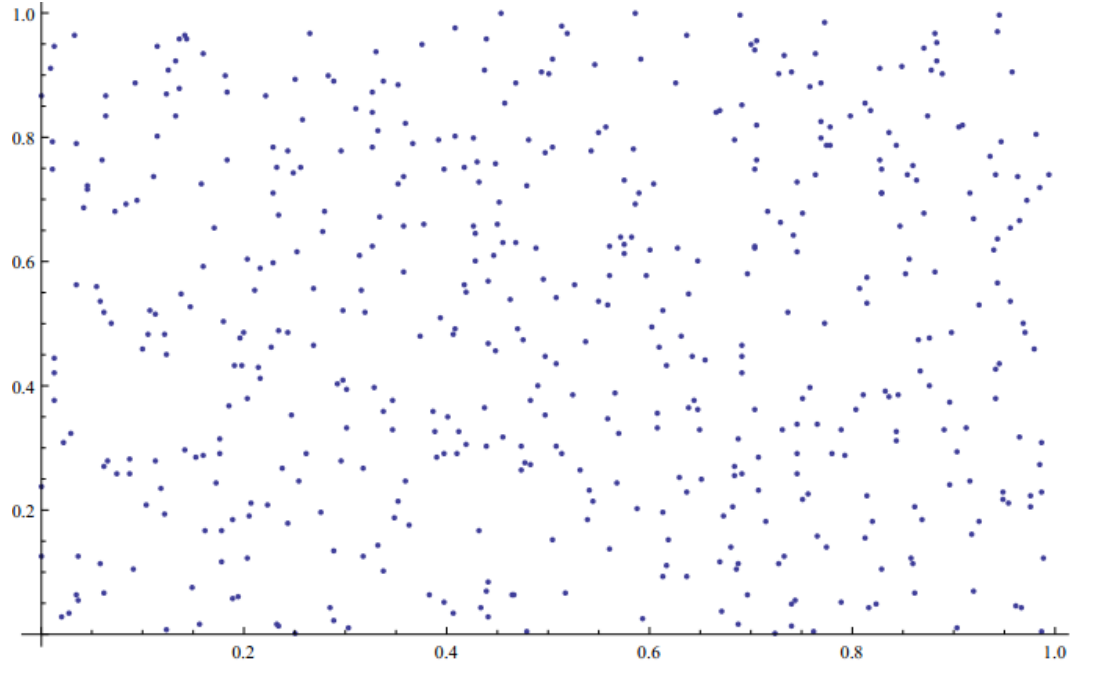


Şekil 2.19 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede algoritmaların çalışması

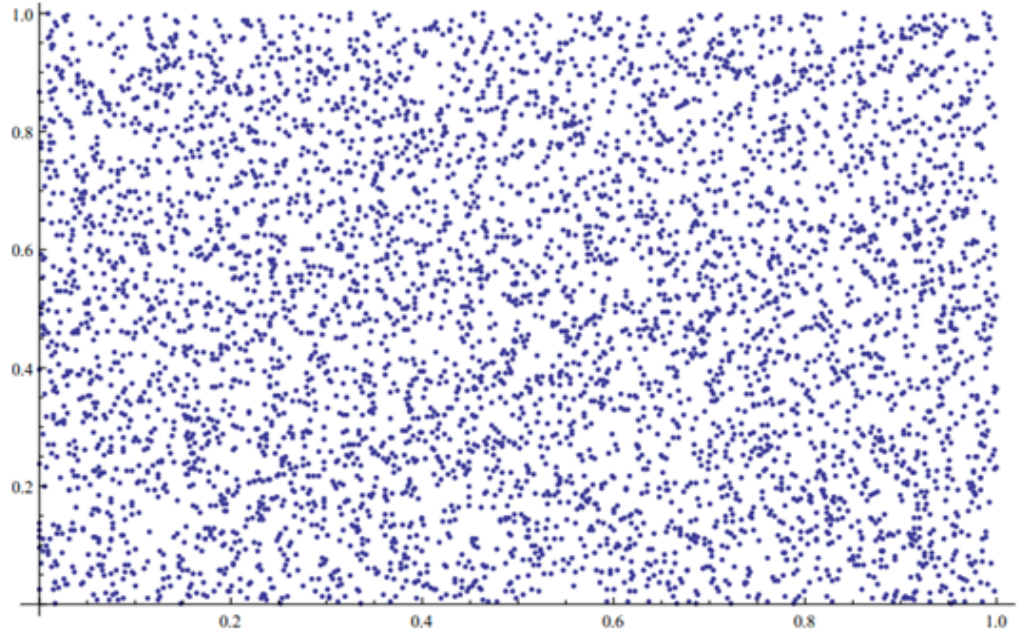
Tablo 2.9 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.19)

Nokta Sayısı	QuickCP(sn)	BFA(sn)
600 000	2.9568	11.4890
900 000	3.7702	18.4770
1 200 000	5.4486	21.3556
1 600 000	6.1338	34.8280
1 200 000	7.9402	39.8498
2 500 000	11.0610	50.3266

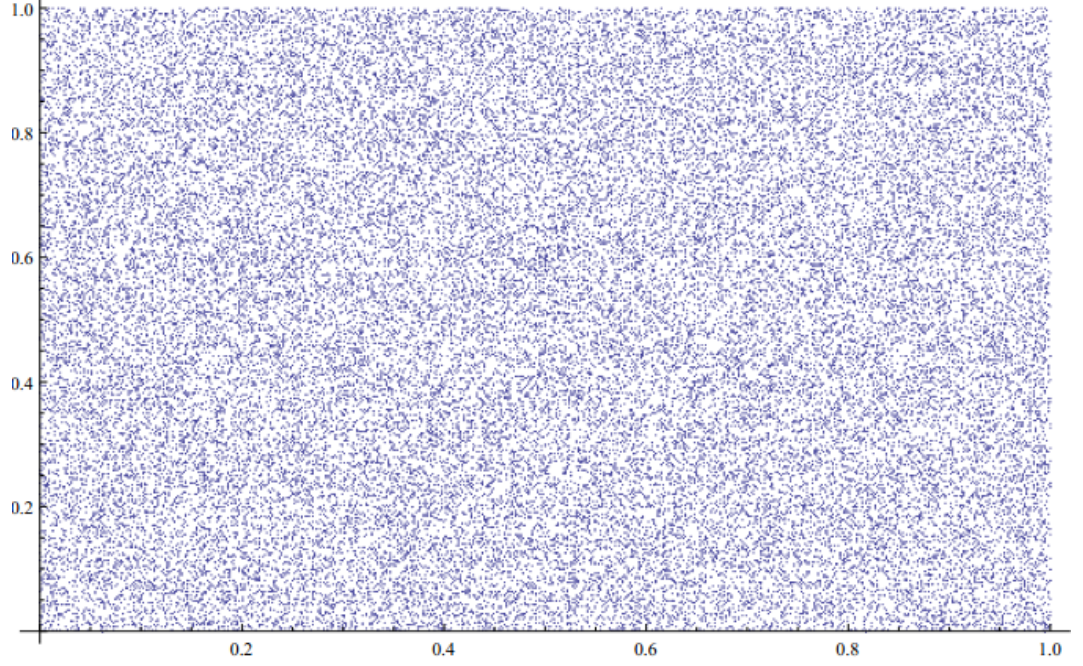
Burada kullanılan noktalar çeşitli boyutlarda gruplandırılarak nokta kümeleri oluşturulmuştur. 1000 elemanlı nokta kümesinden başlanarak 2 500 000 elemanlı nokta kümesine kadar algoritmalar çalıştırılmıştır. Bu nokta kümeleri 1000, 10 000 ve 100 000 elemanlı olduğunda noktaların nasıl dağılım yaptığı Şekil 2.20, Şekil 2.21 ve Şekil 2.22'de görüntülenmiştir.



Şekil 2.20 1000 elemanlı tekdüze dağılımlı nokta kümesi



Şekil 2.21 10 000 elemanlı tekdüze dağılımlı nokta kümesi



Şekil 2.22 100 000 elemanlı tekdüze dağılımlı nokta kümesi

2.5 Çizgisel Yığılma Sorunu ve Çözümü

2.5.1 QuickCP algoritmasında bitiş koşulu

QuickCP algoritmasında, koordinatlarına göre sıralı dizilerde en yakın nokta çifti aranırken bitiş koşulu kullanılır. Bitiş koşulunda (ing. termination condition (Bkz Şekil 2.3)) X ve Y koordinatlarına göre sıralı dizilerde arama yaparken hesaplanan ΔX_{min} ve ΔY_{min} değerleri, her aşamada noktaların o aşamadaki koordinat farklarının en küçük değerlerini taşır.

Algoritma bitiş koşuluna uygun bir nokta çifti yoksa $(n - 1)$. aşamaya kadar çalışır. Böylece dizideki tüm nokta ikilileri ikişer kere incelenmiş olur. Eğer bitiş koşuluna uygun bir nokta çifti varsa daha sonraki aşamalara gerek kalmadan döngülerden çıkılır ve en yakın nokta çifti bulunmuş olur. Bitiş koşuluna uygun bir nokta çifti bulunduğunda daha sonraki aşamalarda o nokta çiftinden daha yakın bir çift bulunamaz. Çünkü her aşamada ayrı ayrı hesaplanan bitiş koşulu değeri bir sonraki aşamada bulunacak en yakın nokta çifti uzaklığı için bir alt sınırdır (Dalkilic and Ergun, 2012).

Alt sınır olması durumunu gösterme:

QuickCP algoritması, $1 \leq a + k \leq n - 1$, $1 \leq i, j, m, n, t, z \leq n$ ve $a, k, i, j, m, n, t, z \in \mathbb{N}$ olmak üzere a . aşamada çalışmakta olsun. Bu aşamada bulunan en küçük X koordinatı farkı (x_i, y_i) ile (x_{i+a}, y_{i+a}) noktalarının X koordinatları farkı olsun. En küçük Y koordinatı farkı da (x_j, y_j) ile (x_{j+a}, y_{j+a}) noktalarının Y koordinat farkı olsun.

$$\Delta Xmin_a = |x_i - x_{i+a}|$$

$$\Delta Ymin_a = |y_i - y_{i+a}|$$

Bu durumda a . aşamadaki bitiş koşulu değeri $\sqrt{\Delta Xmin_a^2 + \Delta Ymin_a^2}$ olur.

Bu aşamada en yakın nokta çifti bulunduğu farz edilsin. Bu en yakın nokta çifti (x_t, y_t) ile (x_{t+a}, y_{t+a}) olsun ve aralarındaki uzaklık d olsun.

$$d = \sqrt{|x_t - x_{t+a}|^2 + |y_t - y_{t+a}|^2}$$

Bu durumda nokta çifti arasındaki uzaklık a . aşamanın bitiş koşulunu sağlandığından $d \leq \sqrt{\Delta Xmin_a^2 + \Delta Ymin_a^2}$ olur.

Algoritma $(a + k)$. aşamaya geldiğinde bu aşamada bitiş koşulu değeri de değişir. Bu aşamada bulunan en küçük X koordinatı farkı (x_m, y_m) ile (x_{m+a+k}, y_{m+a+k}) noktalarının X koordinatları farkı olsun. En küçük Y koordinatı farkı da (x_n, y_n) ile (x_{n+a+k}, y_{n+a+k}) noktalarının Y koordinat farkı olsun.

$$\Delta Xmin_{a+k} = |x_m - x_{m+a+k}|$$

$$\Delta Ymin_{a+k} = |y_n - y_{n+a+k}|$$

Bu durumda algoritmanın $(a + k)$. aşamadaki bitiş koşulu değeri $\sqrt{\Delta Xmin_{a+k}^2 + \Delta Ymin_{a+k}^2}$ olur.

Algoritmanın çalışma prensibindeki önermeye göre a . aşamada bulunan en yakın nokta çifti (x_t, y_t) ile (x_{t+a}, y_{t+a}) 'den daha yakın bir çift, daha sonraki aşamalar da bulunamaz. Burada önermenin tersi doğru kabul edilirse çelişki elde edilecektir:

Farz edilsin ki $(a + k)$. aşamada öyle bir nokta çifti var ki a . aşamada bulunan (x_t, y_t) ile (x_{t+a}, y_{t+a}) nokta çiftinden daha yakın. Bu nokta çiftine (x_z, y_z) ile (x_{z+a+k}, y_{z+a+k}) aralarındaki uzaklığa d_{a+k} denilsin.

$$d_{a+k} = \sqrt{|x_z - x_{z+a+k}|^2 + |y_z - y_{z+a+k}|^2}$$

$d_{a+k} \leq d$ olması gerekir.

Bu yeni en yakın nokta çifti, aynı zamanda kendi aşamasındaki bitiş koşulunu da sağlandığından $d_{a+k} \leq \sqrt{\Delta X min_{a+k}^2 + \Delta Y min_{a+k}^2}$ olmalıdır.

Noktaların sıralı olmasından ötürü bitiş koşulu değerleri her aşamada artar ya da en azından değişmez. Ancak asla bir önceki aşamaya göre azalmaz. Bu nedenle,

$$\sqrt{\Delta X min_a^2 + \Delta Y min_a^2} \leq \sqrt{\Delta X min_{a+k}^2 + \Delta Y min_{a+k}^2}$$

olur.

Algoritmanın en baştaki prensibine göre $\sqrt{\Delta X min_a^2 + \Delta Y min_a^2}$ değeri d_{a+k} değeri için bir alt sınırdır.

Sırasıyla a . ve $(a + k)$. aşamalarda bulunan en yakın çiftlerin uzaklık değerleri bu eşitsizliğe eklendiğinde,

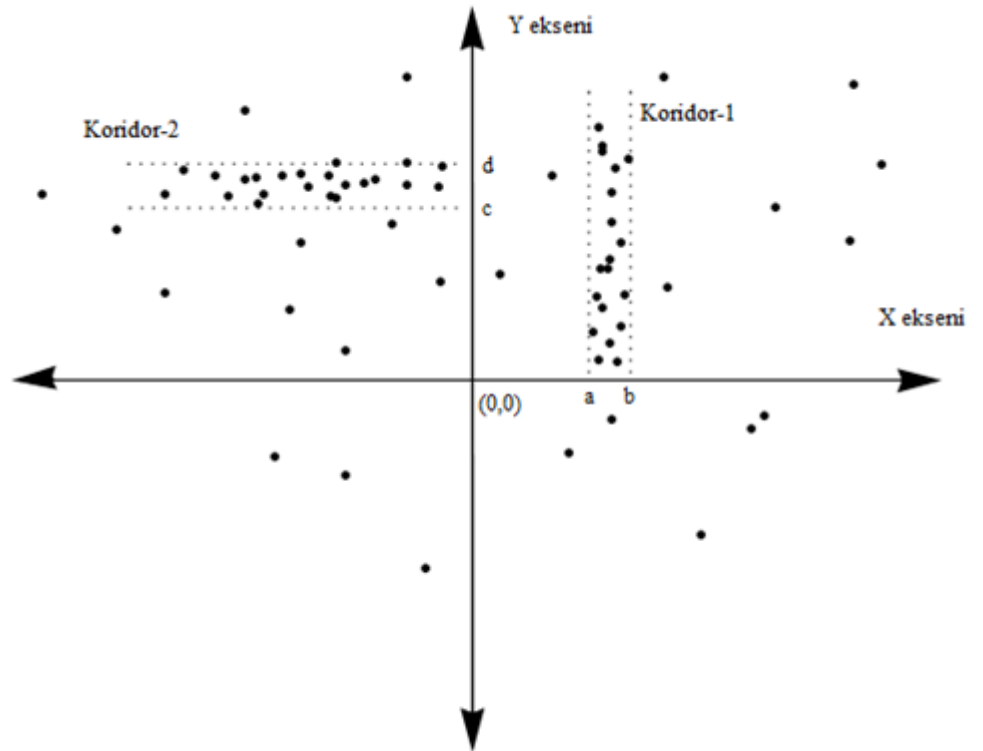
$$d \leq \sqrt{\Delta X min_a^2 + \Delta Y min_a^2} \leq d_{a+k} \leq \sqrt{\Delta X min_{a+k}^2 + \Delta Y min_{a+k}^2}$$

olur. Bu durumda $d \leq d_{a+k}$ elde edilir. Çelikçi iddiasına başlarken kabul edilen $d_{a+k} \leq d$ varsayımıyla, $d \leq d_{a+k}$ ifadesi her zaman doğru olamaz. Yani böyle bir (x_z, y_z) ile (x_{z+a+k}, y_{z+a+k}) nokta çifti yoktur. Bu, her aşamada bulunan bitiş koşulu değerinin bir sonraki aşama için alt sınır olmasından kaynaklanır.

Bitiş koşulu QuickCP algoritması için döngüden erken çıkma avantajı yaratır. Ancak bazı durumlarda bitiş koşuluna uygun bir nokta çifti bulmak zor olduğundan döngülerden erken çıkılamaz. Bu da QuickCP algoritmasının çalışma süresini kötü etkiler.

2.5.2 Çizgisel yığılm sorunu

QuickCP algoritmasının bitiş koşulu kullanılarak döngüden erken çıkılmasını engelleyen durum, noktaların koordinat eksenlerindeki izdüşümünün belli değerler arasında yığıldığı durumdur. Bu duruma uygun görsel örnek Şekil 2.23'te resmedilmiştir. Şekildeki Koridor-1 ve Koridor-2'nin içinde yer alan noktaların X ve Y eksenlerine izdüşümü yapıldığında, koordinatlar belli değerler arasında yığılır. X koordinatları (a, b) değerleri arasında, Y koordinatları (c, d) değerleri arasında birikir.



Şekil 2.23 Örnek çizgisel yığılm Koridor-1 ve Koridor-2

En yakın nokta çifti bulunacak nokta kümesinde bu durum varsa, nokta sayısı ve koordinatlardaki belli bir değere çizgisel olarak yığılm arttıkça QuickCP algoritmasının çalışma süresi kötüye gider. En kötü durum karmaşıklığı $O(n^2)$ 'ye

kadar ulaşabilir (Dalkilic and Ergun, 2012). QuickCP algoritması döngülerden erken çıkmadığı için uzun süre çalışmaktadır, bitiş koşulu yığılım nedeniyle kullanışsızlaşmaktadır.

Burada çizgisel yığılım içeren nokta kümelerinde QuickCP algoritmasının veriminin nasıl değiştiğini göstermek amaçlanmıştır. Çizgisel yığılım içeren nokta kümelerinde QuickCP algoritmasının bitiş koşulundaki, bulunan en yakın nokta uzaklığının kıyaslanacağı alt sınır değeri her aşamada 0 ya da 0'a yakın değerler olduğundan QuickCP algoritmasının erken sonlanma olasılığı ortadan kalkmaktadır.

QuickCP algoritması örneğin a . aşamada çalışırken ve bu aşamada bulunduğu en küçük uzaklık değeri d iken, bu aşamanın bitiş koşulunda alt sınır değeri $\sqrt{\Delta Xmin_a^2 + \Delta Ymin_a^2}$ olsun. Yığılım içeren kümelerde X ve Y eksenlerinde en küçük koordinat farkları 0 ya da 0'a yakın değerler olur. Bu durumda $\sqrt{\Delta Xmin_a^2 + \Delta Ymin_a^2}$ değeri de 0 veya 0'a yakın olduğundan, bitiş koşulunu çalıştıracak $d \leq \sqrt{\Delta Xmin_a^2 + \Delta Ymin_a^2}$ durumunun ortaya çıkması zorlaşır.

2.5.3 QuickCP algoritmasının BFA'nın gerisine düştüğü koşulları bulma

QuickCP algoritmasının, noktaların X veya Y eksenine izdüşümü yapıldığında çalışma süresi olarak BFA'dan daha kötü performans gösterdiği koşulları belirlemek adına histogram ve çeşitli fonksiyonlar kullanılmıştır. Kümelerde noktaların X ve Y koordinatlarındaki çizgisel yığılım deneysel olarak azdan başlayarak artırılmıştır. Örnek olarak Şekil 2.24'te 40 elemanlı bir kümede çizgisel yığılım içeren kümenin nasıl oluşturulduğu gösterilmektedir. Başlangıçta 40 elemanlı kümedeki noktaların tüm koordinatları birbirinden farklıdır ve noktalar bir değere yığılım içermemektedir. Ardından noktaların X ve Y koordinatları yığılım oluşturacak şekilde değiştirilir. Dizinin yarısına kadar X koordinatları, yarısından sonra ise Y koordinatları 1, 2, 3, 4 sayıları ile ardışık olarak doldurulur. 40 adet noktanın X ve Y koordinatlarında 1, 2, 3 ve 4 sayıları bir kereden fazla kullanılmış olur. Noktaların eksenlere izdüşümü yapıldığında X ile Y eksenlerinin (1,0), (2,0), (3,0), (4,0), (0,1), (0,2), (0,3), (0,4) noktalarında yığılım olmaktadır. Burada örneğin (0,3) noktası, sadece Y koordinatlarında ve kümenin ilk yarısından sonra kullanıldığından $40/2 = 20$ ve Y koordinatlarının

bu yarısında (0,1), (0,2), (0,3), (0,4) diye 4 adet nokta tekrar ettiğinden $20/4 = 5$ kere tekrarlanmaktadır.

[0]:	< 1.00000 , 69.59209 >
[1]:	< 2.00000 , 62.17547 >
[2]:	< 3.00000 , 94.18472 >
[3]:	< 4.00000 , 53.97263 >
[4]:	< 1.00000 , 74.04524 >
[5]:	< 2.00000 , 30.69144 >
[6]:	< 3.00000 , 28.65107 >
[7]:	< 4.00000 , 5.08341 >
[8]:	< 1.00000 , 70.86533 >
[9]:	< 2.00000 , 36.84119 >
[10]:	< 3.00000 , 59.64172 >
[11]:	< 4.00000 , 53.23415 >
[12]:	< 1.00000 , 97.89620 >
[13]:	< 2.00000 , 79.82257 >
[14]:	< 3.00000 , 27.52744 >
[15]:	< 4.00000 , 102.26342 >
[16]:	< 1.00000 , 89.05038 >
[17]:	< 2.00000 , 83.92558 >
[18]:	< 3.00000 , 63.35322 >
[19]:	< 4.00000 , 90.49868 >
[20]:	< 16.48527 , 1.00000 >
[21]:	< 98.80975 , 2.00000 >
[22]:	< 48.68232 , 3.00000 >
[23]:	< 24.30614 , 4.00000 >
[24]:	< 8.81400 , 1.00000 >
[25]:	< 52.07868 , 2.00000 >
[26]:	< 45.80161 , 3.00000 >
[27]:	< 76.07925 , 4.00000 >
[28]:	< 60.96589 , 1.00000 >
[29]:	< 25.64941 , 2.00000 >
[30]:	< 99.65009 , 3.00000 >
[31]:	< 78.33925 , 4.00000 >
[32]:	< 102.53716 , 1.00000 >
[33]:	< 12.48093 , 2.00000 >
[34]:	< 49.93964 , 3.00000 >
[35]:	< 10.40237 , 4.00000 >
[36]:	< 19.94211 , 1.00000 >
[37]:	< 60.25606 , 2.00000 >
[38]:	< 46.24406 , 3.00000 >
[39]:	< 58.88733 , 4.00000 >

Şekil 2.24 40 elemanlı nokta kümesinin X ve Y koordinatları

Tablo 2.10'daki “ n ” kolonu, üzerinde en yakın nokta çifti aranacak kümenin nokta sayısıdır. Burada “*Sınır*” kolonu ise yığılım oluştururken kaç adet tam sayı kullanıldığını gösterir. Yani örneğin, $n = 500$ ve *Sınır* = 3 iken noktaların koordinatlarını doldurmak için 1, 2, 3 sayıları kullanılır ve noktaların eksenlere izdüşümü yapıldığında (0,1), (0,2), (0,3), (1,0), (2,0), (3,0) noktalarında yığılım olur. Ayrıca, bu 6 adet noktaya $500/2 = 250$ ve $250/3 = 83.3$ ’er kere yığılım olmuştur, yani bir değere en fazla 83.3 tane nokta yığılmaktadır. Kümelerde BFA’nın QuickCP algoritmasını hangi “*Sınır*” değerinde geçtiği denenerek bulunmuştur. Örneğin 20 000 elemanlı kümede “*Sınır*” değeri 55’tir. Sınır değeri 55 ve daha küçük iken çizgisel yığılım artmakta ve QuickCP algoritması çalışma süresi olarak BFA’nın gerisine düşmektedir.

Tablo 2.10'un "Yığılım" kolonu, bir kümedeki nokta sayısını içeren " n " kolonundaki değerin yarısı, "Sınır" kolonundaki değere bölünerek (örneğin $n = 500$ iken $500/2 = 250$ ve $250/3 = 83.3$ işlemlerindeki gibi) o kümede X veya Y eksenlerindeki bir değere en fazla kaç nokta yığıldığını gösterir.

$Yığılım$ ve $\log_2 n$ değerleri kullanılarak " K " değerine ulaşılmaktadır. " K " değeri $\log_2 n$ 'nin " $Yığılım$ "ı veren katıdır. Yani $\log_2 n * K = Yığılım$ denklemindeki K değeridir. Tablo 2.10'dan hesaplandığı üzere, bulunan en küçük K değeri K için bir alt sınır olmalıdır. Bu alt sınır, nokta sayısı ne olursa olsun o kümede en az $\log_2 n * K$ tane nokta yığılım oluşturduğunda QuickCP algoritmasının BFA'nın gerisine düşebileceğini gösterir. Bu anlamda alt sınır $K = 10 \leq 10.91483$ (en küçük " K " kolonu değeri) olarak alındığında n elemanlı bir nokta kümesinde, $\log_2 n * 10$ tane nokta yığılım oluşturursa QuickCP algoritması, BFA'nın gerisine düşebilir ve çalışma süresi olarak daha kötü performans gösterebilir.

Tablo 2.10 BFA'nın QuickCP'yi geçtiği durum tablosu

n	$\log_2 n$	Sınır	$Yığılım = n / (2 * Sınır)$	$K = Yığılım / \log_2 n$
1000	9.96578	3	166.66666	16.72388
10 000	13.28771	30	166.66666	12.54291
20 000	14.28771	55	181.81818	12.72549
40 000	15.28771	112	178.57142	11.68071
80 000	16.28771	225	177.77777	10.91483
100 000	16.60964	180	277.77777	16.72388
200 000	17.60964	375	266.66666	15.14322
300 000	18.19460	518	289.57528	15.91544
500 000	18.93156	1082	231.05360	12.20467
600 000	19.19460	1038	289.01734	15.05721
900 000	19.77956	1550	290.32258	14.67790
1 200 000	20.19460	2075	289.15662	14.31850
1 600 000	20.60964	2725	293.57798	14.24469
1 200 000	20.93156	3100	322.58064	15.41120
2 500 000	21.25349	3600	347.22222	16.33718

2.5.4 Nokta kümesindeki çizgisel yığılma belirlemede histogram kullanımı

Histogram verinin dağılımını gösteren bir çeşit grafiksel temsildir. Veri kümesi içinde en büyük ve en küçük eleman arasındaki fark nokta kümesinin eleman sayısına bölünerek birbiriyle eşit aralıklar elde edilir. Veri kümesindeki noktaların bu aralıklardan hangilerine düştüğü hesaplanır (Boleman, M, 2013). Histogram burada nokta kümelerindeki noktaların koordinatlarının dağılımını incelemek için kullanılmıştır. Her bir koordinat için ayrı ayrı histogram yapılarak, o koordinatların belli değerlerde yığılma yapıp yapmadığı belirlenmektedir.

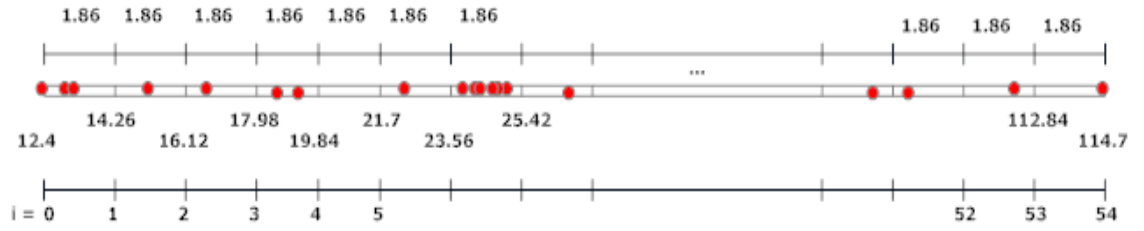
Örneğin n elemanlı bir kümenin, X ekseninde yığılma yapıp yapmadığını belirlemek adına histogram olarak yapılan işlemler şunlardır,

“ n ” elemanlı boş bir “frekans” dizisi yaratılır ve her eleman 0 ile ilklendirilir.

“ n ” elemanlı nokta kümesinde noktaların X koordinatlarının en büyüğü bulunarak “*en büyük*” değişkenine, en küçüğü bulunarak “*en küçük*” değişkenine atılır.

“*en büyük*” ve “*en küçük*” değişkenlerinin arasındaki fark bulunur ve “*frekans boyutu*” değişkenine atılır. Bu değer X koordinatlarının aralığıdır. “*frekans boyutu*” değeri toplam nokta sayısına bölündüğünde elde edilen aralık parçasında en fazla bir adet nokta olmalıdır. Yani bir noktaya düşen aralık miktarı elde edilir.

Şekil 2.25’te örnek olarak bir histogram bulunmaktadır. Burada, 55 tane nokta içeren bir nokta kümesinde noktaların X koordinatları 12.4 (“*en küçük*”), 114.7 (“*en büyük*”) değerleri arasında değişsin. Bu durumda sayı aralığı $114.7 - 12.4 = 102.3$ (“*frekans boyutu*”) olur. Bu “*frekans boyutu*” değeri toplam nokta sayısı olan 55’e bölündüğünde $102.3/55 = 1.86$ değeri en fazla bir adet nokta bulunması gereken aralık uzunluğu olur. Eğer bir aralık parçasında birden fazla nokta bulunursa, bu durumda çizgisel yığılma oluşmaya başlar. Örneğin Şekil 2.25’te, $23.56 - 25.42$ aralığındaki 6 adet nokta, bu aralıkta yığılma oluşturmuştur.



Şekil 2.25 Bir küme üzerinde örnek histogram

Boş “*frekans*” dizisinde, her bir aralıkta kaç adet nokta bulunduğu bilgisi tutulacaktır. “*frekans*” dizisindeki bir aralıkta kaç adet nokta olduğunu bulmak için noktaların X koordinatlarının “*frekans*” dizisinin kaçınıcı elemanına denk geldiği bulunmalıdır. Bu yüzden 1 sayısı “(*frekans boyutu*) / n ” değerine bölünür.

Yani $(1 / ((\text{frekans boyutu}) / n)) = n / \text{frekans boyutu}$ olur. Bu değere “*çarpım değeri*” denilirse, bu değer her bir koordinatın “*frekans*” dizisinde hangi aralığa ait olduğunu bulmak için kullanılacaktır.

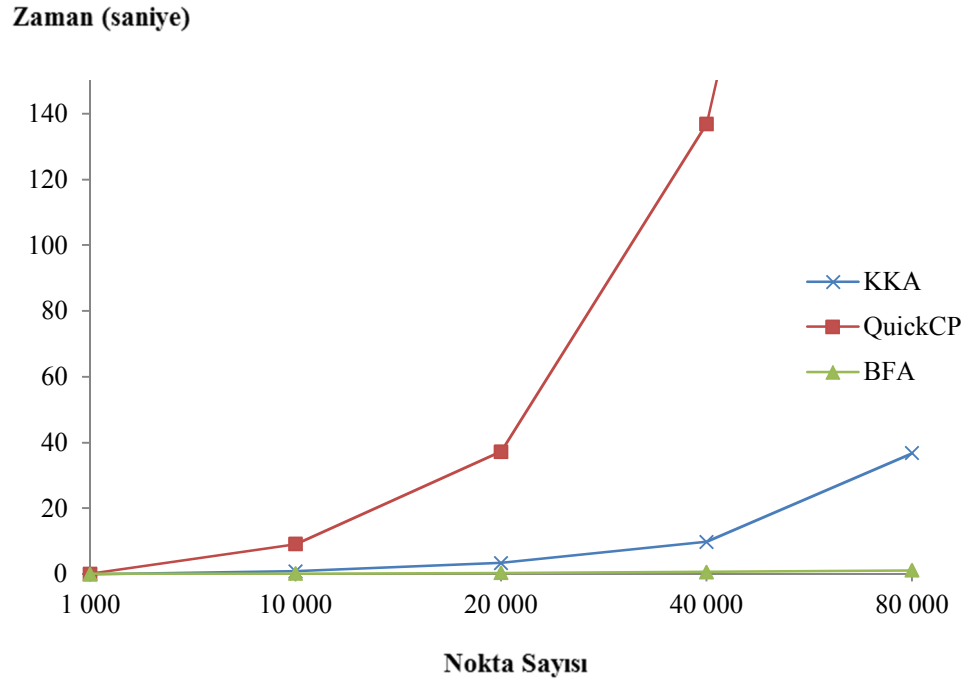
Örnek üzerinden devam edilirse, 1.86 değeri kullanılarak, 55 tane nokta koordinatının hangi aralığa ait olduğu $1/1.86 = 0.53$ sayısı kullanılarak bulunur. 0.53 sayısı “*çarpım değeri*”dir. Kümedeki tüm noktaların X koordinatlarının “*en küçük*” koordinatla arasındaki fark, 0.53 sayısı ile çarpılır ve farkın 0 ile 55 sayısı arasında bir değere düşmesi sağlanır. Ardından o indeksteki dizi elemanı değeri bir arttırılır. 55 elemanlı bu kümede örneğin (13.6, 55.1) noktasının X koordinatı olan 13.6 değerinin, kümenin en küçük X koordinatı değeri ile arasındaki fark $13.6 - 12.4 = 1.2$ olur. Bu değer “*çarpım değeri*” ile çarpılır, $1.2 * 0.53 = 0.636$. Ardından en yakın tam sayılardan küçüğüne yuvarlanır, $0.636 \cong 0$ olur. Bu durumda (13.6, 55.1) noktası 0. indeksine aittir ve bu indeksteki dizi elemanı 1 arttırılır ki ilk aralık zaten $12.4 - 14.26$ arası X koordinatına sahip noktaları içerir.

Örneğin, (24.1, 34.4) noktasının X koordinatı olan 24.1 değeri ile kümenin en küçük değeri arasındaki fark 11.7’dir, $24.1 - 12.4 = 11.7$. Bu değer “*çarpım değeri*” ile çarpılır, $11.7 * 0.53 = 6.201$. Ardından en yakın küçük tamsayıya yuvarlanır, $6.201 \cong 6$ olduğundan (24.1, 34.4) noktası 6. aralığa aittir. Bu indeksteki dizi elemanı 1 arttırılır.

Her aralıktaki nokta dağılımı “*frekans*” dizisiyle belirlendikten sonra bu dağılımların QuickCP algoritmasını, BFA’nın gerisine düşürecek miktarda olup olmadığı K değeriyle belirlenir (Bkz Tablo 2.10). Eğer “*frekans*” dizisinin elemanları arasında $\log_2 n * K$ sayısından daha büyük bir eleman bulunursa, o aralıkta $\log_2 n * K$ taneden daha fazla nokta var demektir. Bu da çizgisel yığılmaya işaret eder.

2.5.5 Üstel dağılımlı nokta kümelerinde çizgisel yığılım

Burada üstel dağılımlı sayılarla oluşturulan nokta kümeleri çizgisel yığılım içerecek hale getirilmiştir. Ardından, algoritmalar bu yığılım içeren kümelerde çalıştırılmıştır. QuickCP algoritması çizgisel yığılım içeren 80 000 noktalı kümedeki en yakın nokta çiftini 410.3164 saniyede hesapladığından daha kalabalık nokta kümeleri incelenmemiştir (Şekil 2.26).



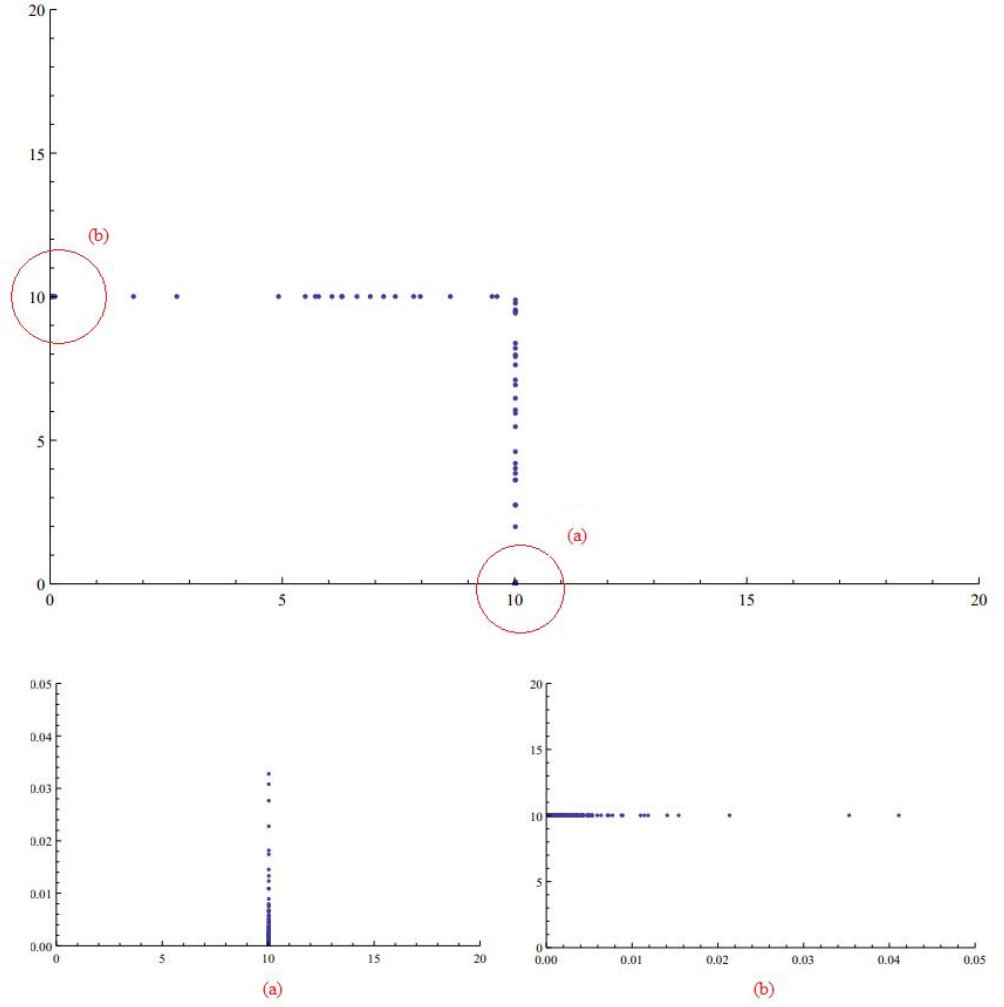
Şekil 2.26 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması

Tablo 2.11 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.26)

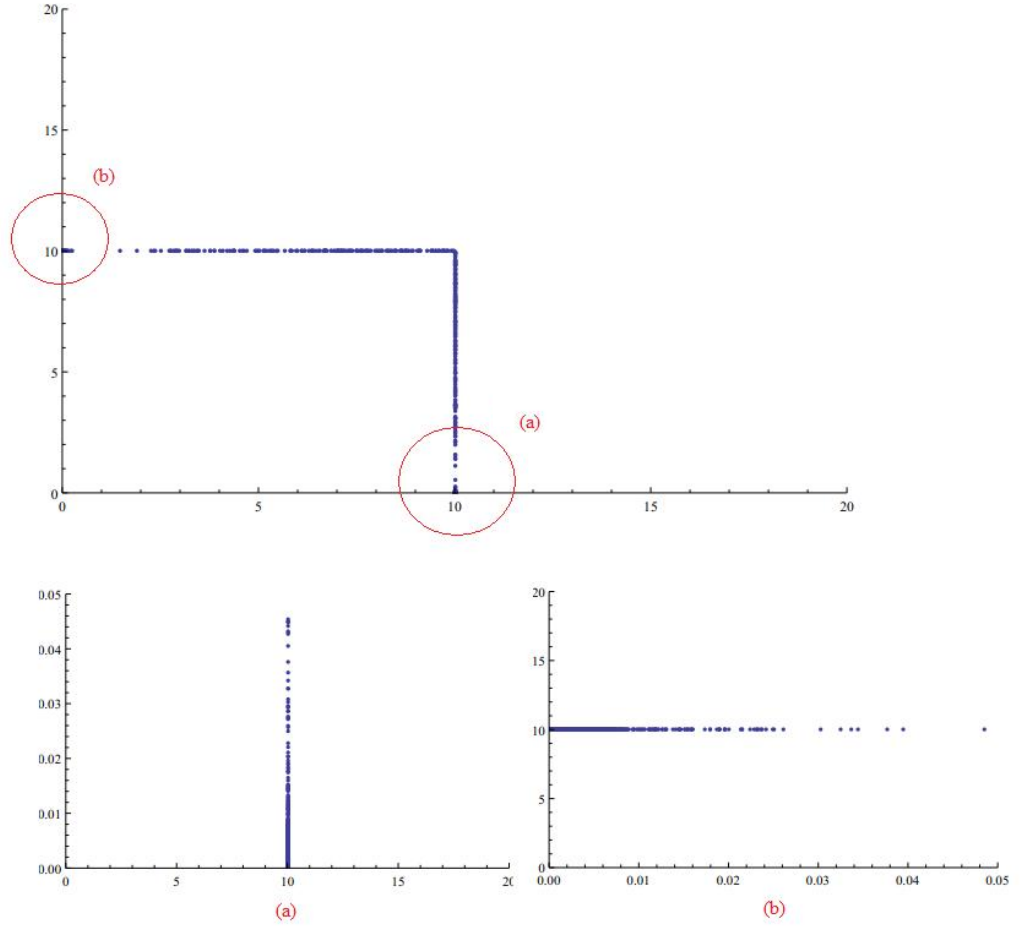
Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0080	0.0844	0.0210
10 000	0.8296	9.1538	0.1738
20 000	3.3566	37.2150	0.3322
40 000	9.7654	136.8518	0.6416
80 000	36.7288	410.3164	1.1210

Veri tabanı ile elde edilen sayılarla oluşturulan noktalar çeşitli boyutlarda gruplandırılarak nokta kümeleri oluşturulmuştur. Bu gruplandırma 1000 elemanlı nokta kümesinden başlanarak 80 000 elemanlı nokta kümesine kadar devam etmiştir. Buradaki tüm nokta kümeleri olabilecek en fazla çizgisel yığılım içerecek hale getirilmiştir. Böylece QuickCP algoritmasının çalışma süresi anlamında en kötü durumuna ulaşması hedeflenmiştir. Kümelerde oluşturulan çizgisel yığılımlar sayesinde QuickCP algoritmasının bitiş koşulunda kullanılan alt sınır değerleri hep 0 olmuştur. QuickCP algoritması döngülerinden erken çıkamadığı için çalışma süresi anlamında KKA'nın da gerisine düşmüştür. Nokta kümelerinde kümedeki noktaların yarısının X koordinatı 10.0, diğer yarısının da Y koordinatı 10.0 olarak hazırlanmıştır. Veri tabanından elde edilen 1000, 10 000 ve 100 000

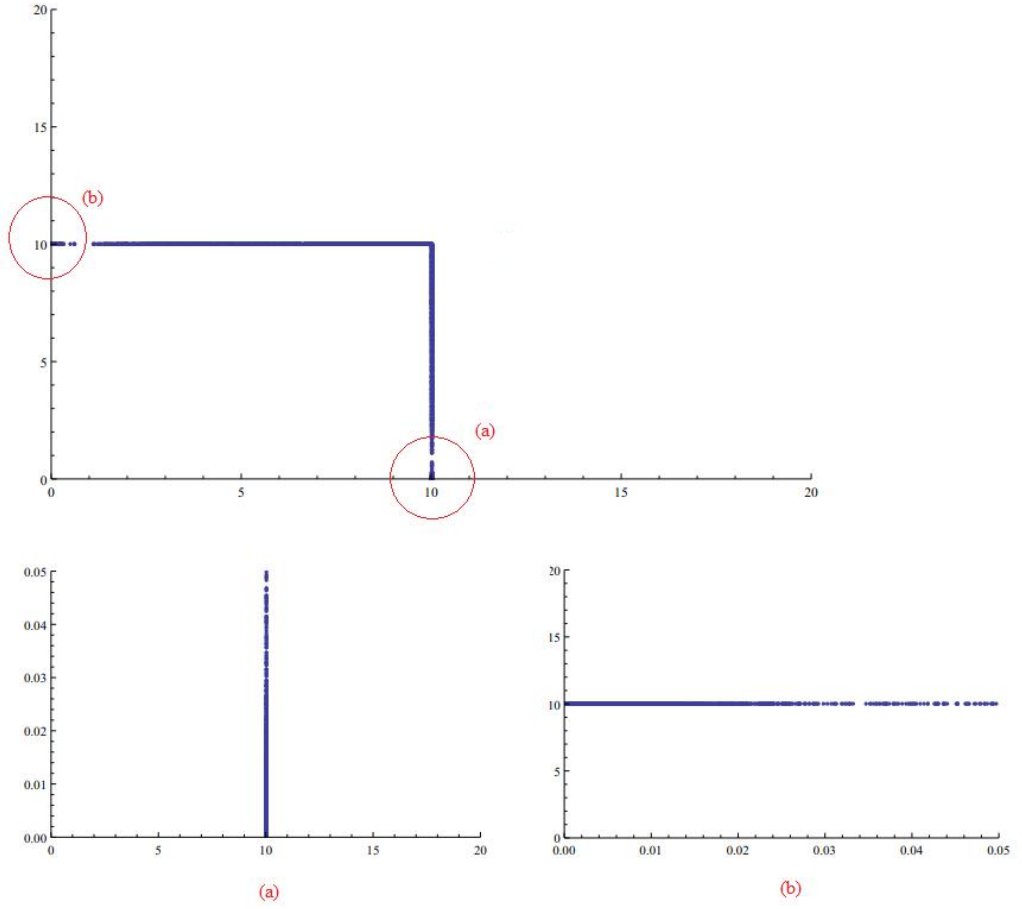
elemanlı nokta kümelerinin çizgisel yığılım içeren haliyle görüntüleri Şekil 2.27, Şekil 2.28 ve Şekil 2.29’da görülmektedir.



Şekil 2.27 Çizgisel yığılım içeren 1000 elemanlı üstel dağılımlı nokta kümesi



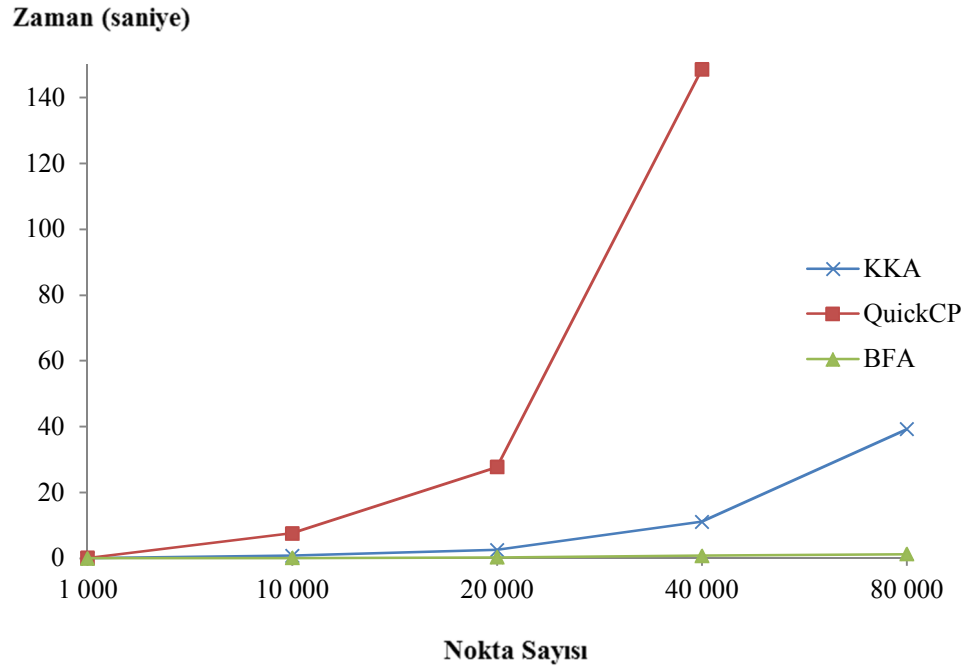
Şekil 2.28 Çizgisel yığılma içeren 10 000 elemanlı üstel dağılımlı nokta kümesi



Şekil 2.29 Çizgisel yığılım içeren 100 000 elemanlı üstel dağılımlı nokta kümesi

2.5.6 Normal dağılımlı nokta kümelerinde çizgisel yığılım

Burada iki boyutlu nokta üreten uygulama ile normal dağılımlı noktalar elde edildikten sonra çizgisel yığılım maksimum olacak şekilde noktalar düzenlenmiştir. Elde edilen nokta kümeleri 80 000 elemana kadar oluşturulup algoritmalar çalıştırıldığında, QuickCP algoritması 80 000 elemanlı kümede en yakın nokta çiftini bulmak için 445.0914 saniye harcadığından daha kalabalık kümeler incelenmemiştir.

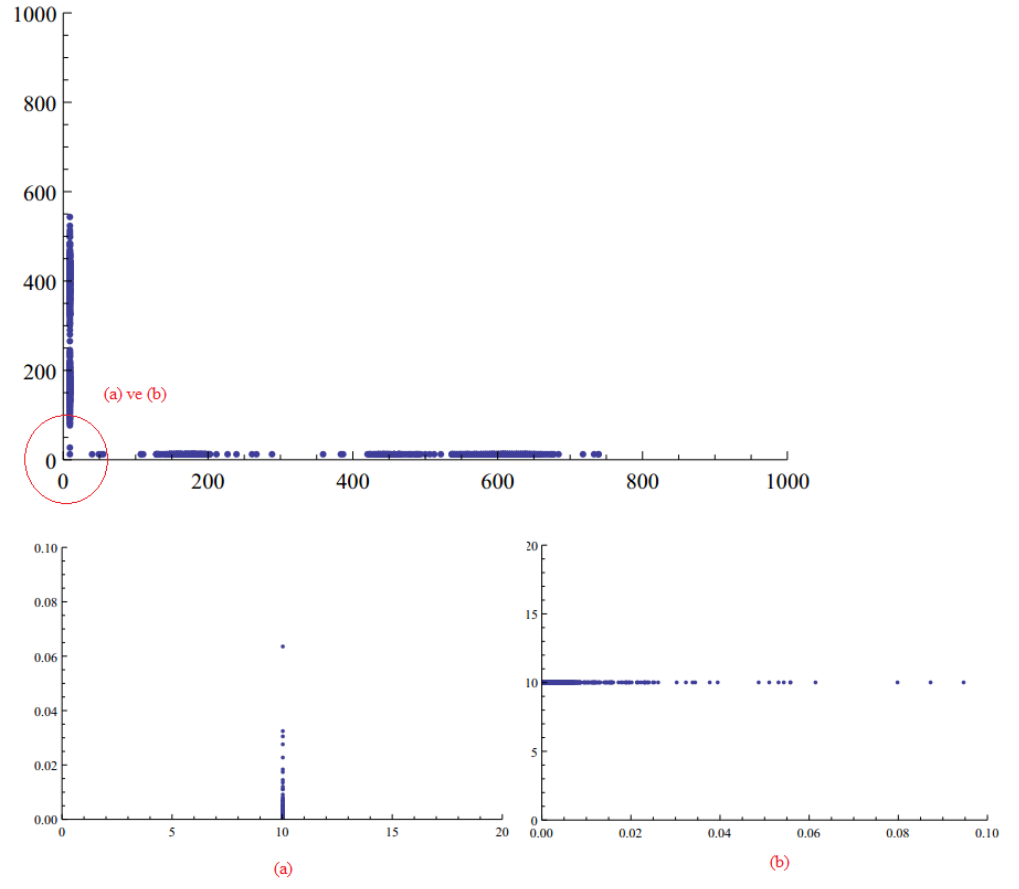


Şekil 2.30 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması

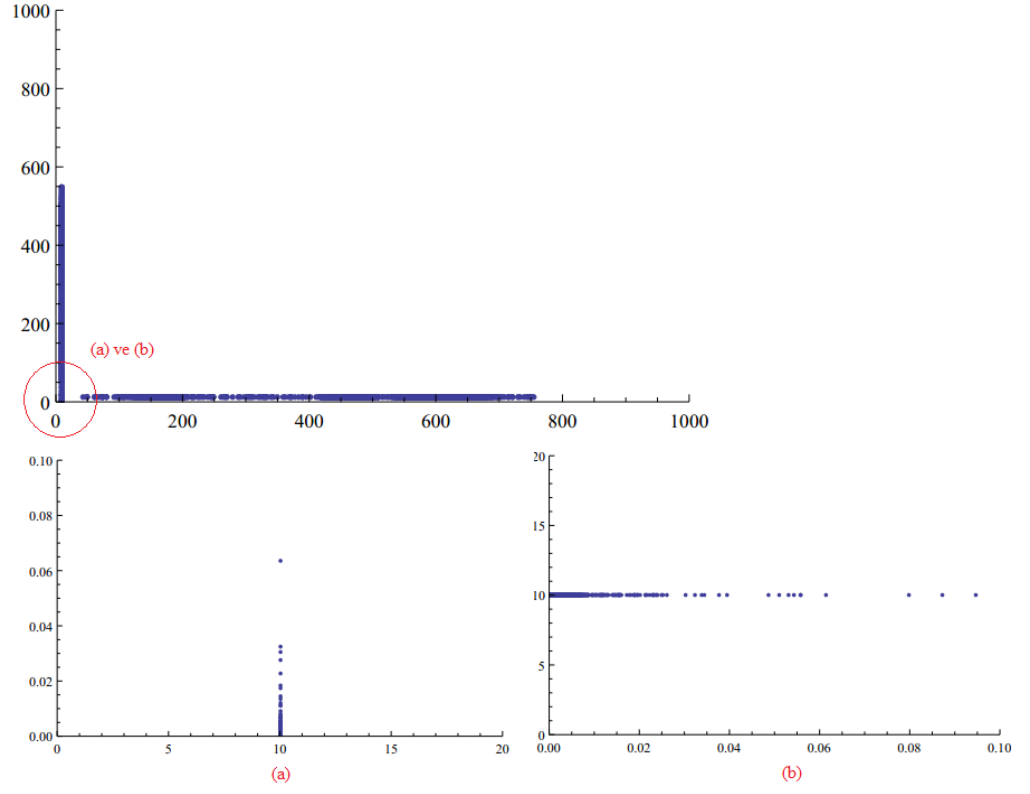
Tablo 2.12 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.30)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0062	0.0680	0.0168
10 000	0.7974	7.5542	0.1264
20 000	2.4988	27.7316	0.3030
40 000	11.1256	148.6256	0.7462
80 000	39.2580	445.0914	1.2238

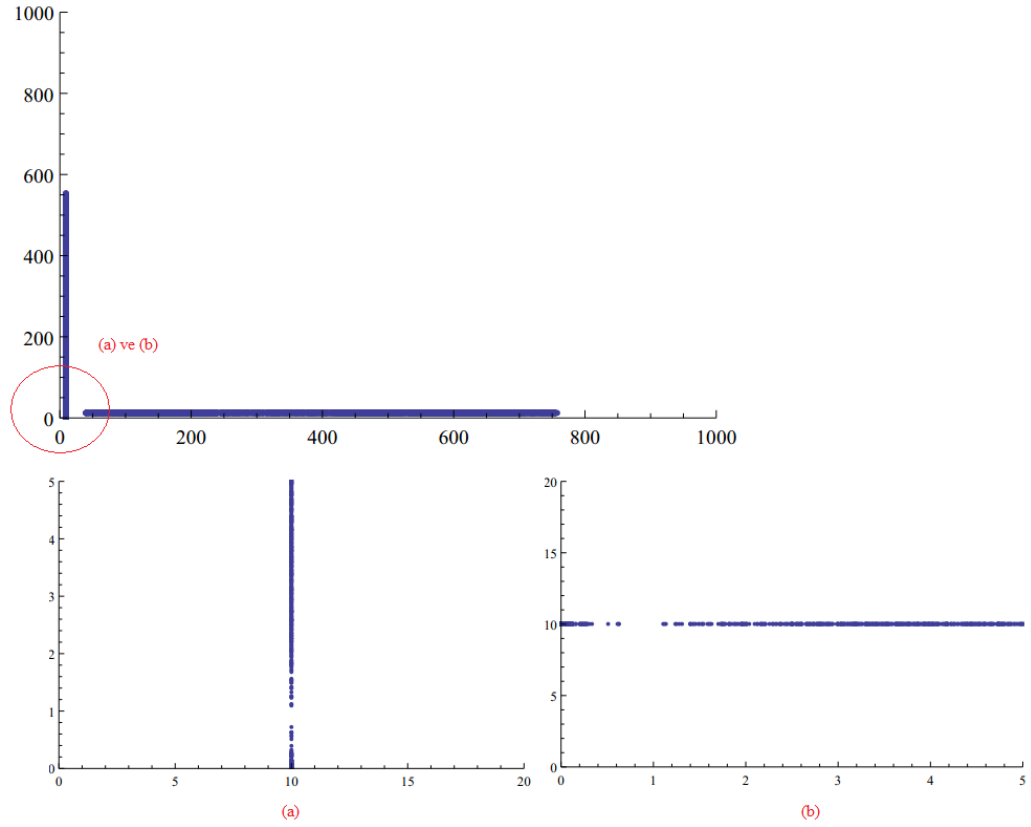
Buradaki tüm nokta kümeleri olabilecek en fazla çizgisel yığılım içerecek hale getirilmiştir. Böylece QuickCP algoritmasının çalışma süresi anlamında en kötü durumuna ulaşması hedeflenmiştir. Kümelerde oluşturulan çizgisel yığılımlar sayesinde QuickCP algoritmasının bitiş koşulunda kullanılan alt sınır değerleri hep 0 olmuştur. QuickCP algoritması döngülerinden erken çıkamadığı için çalışma süresi anlamında KKA'nın da gerisine düşmüştür. Normalde noktaların X koordinat değerleri 0 ile 800.0 arasında, Y koordinat değerleri 0 ile 600.0 arasında değişmektedir. Çizgisel yığılım oluşturmak için nokta kümelerinde kümedeki noktaların yarısının X koordinatı 10.0, diğer yarısının Y koordinatı 10.0 olarak değiştirilmiştir. Veri tabanından elde edilen 1000, 10 000 ve 100 000 elemanlı nokta kümelerinin çizgisel yığılım içeren haliyle görüntüleri Şekil 2.31, Şekil 2.32 ve Şekil 2.33'te görülmektedir.



Şekil 2.31 Çizgisel yığılım içeren 1000 elemanlı normal dağılımlı nokta kümesi



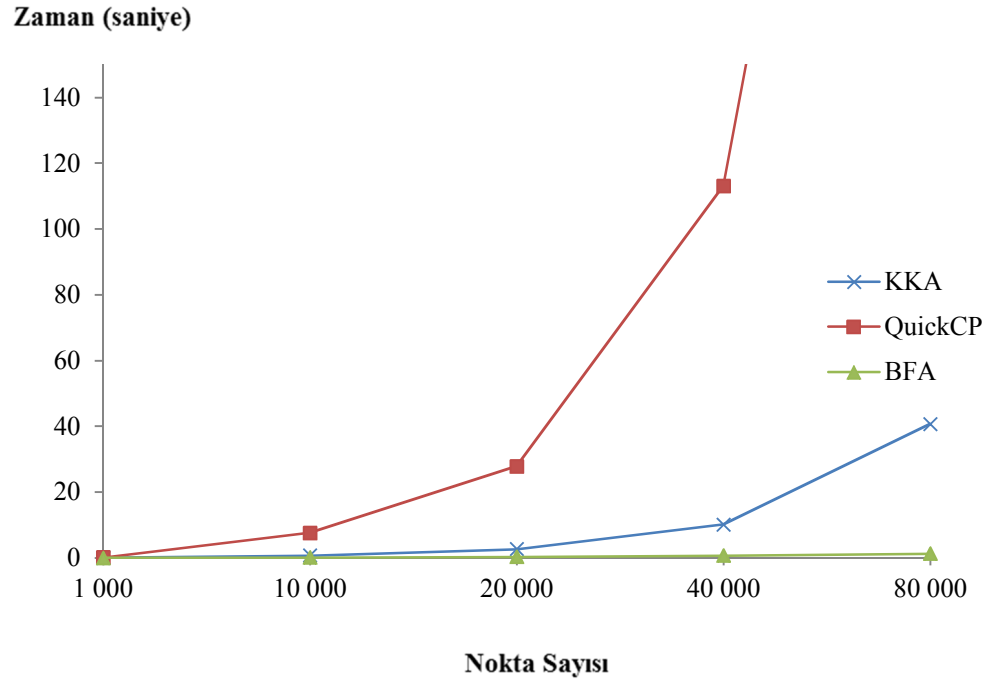
Şekil 2.32 Çizgisel yığılım içeren 10 000 elemanlı normal dağılımlı nokta kümesi



Şekil 2.33 Çizgisel yığılım içeren 100 000 elemanlı normal dağılımlı nokta kümesi

2.5.7 Tekdüze dağılımlı nokta kümelerinde çizgisel yığılım

Burada Java'daki random() fonksiyonu ile elde edilen sayılarla oluşturulan tekdüze dağılımlı nokta kümeleri çizgisel yığılım maksimum olacak şekilde düzenlenmiştir. Elde edilen nokta kümeleri 80 000 elemana kadar oluşturulup algoritmalar çalıştırıldığında, QuickCP algoritması 80 000 elemanlı kümede en yakın nokta çiftini bulmak için 449.0196 saniye harcadığından daha kalabalık kümeler incelenmemiştir.

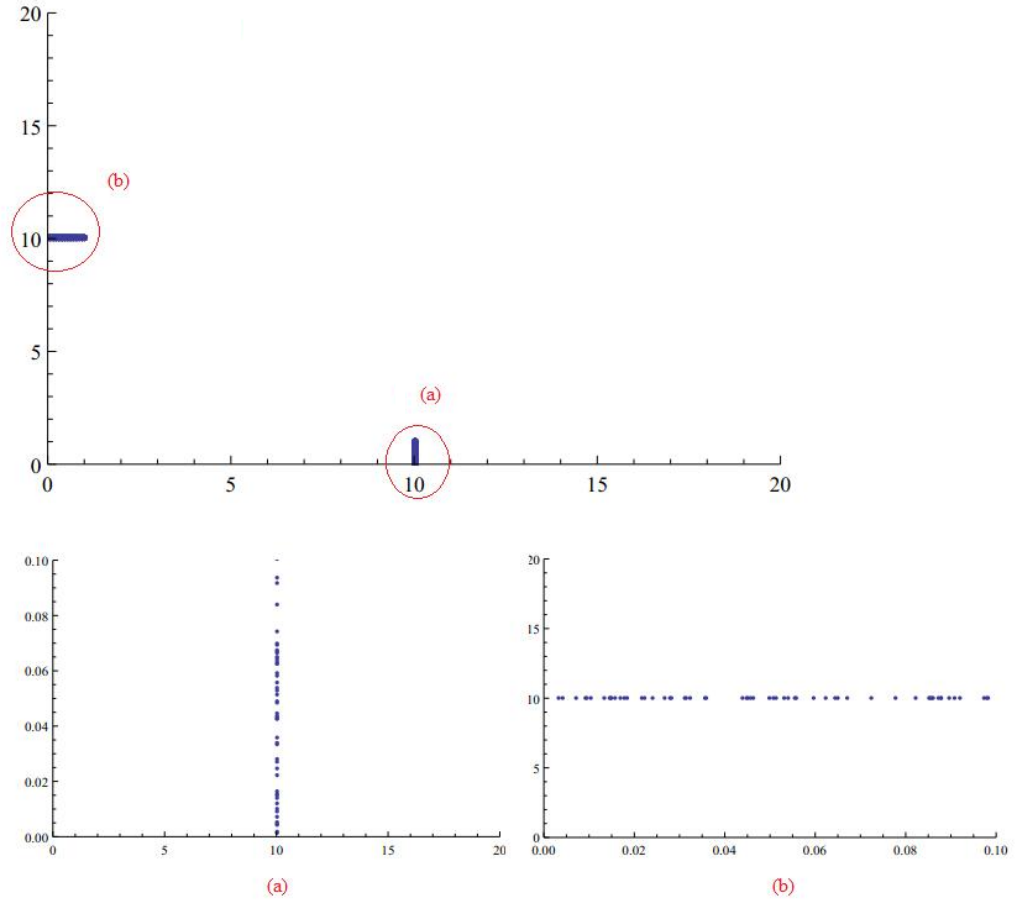


Şekil 2.34 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması

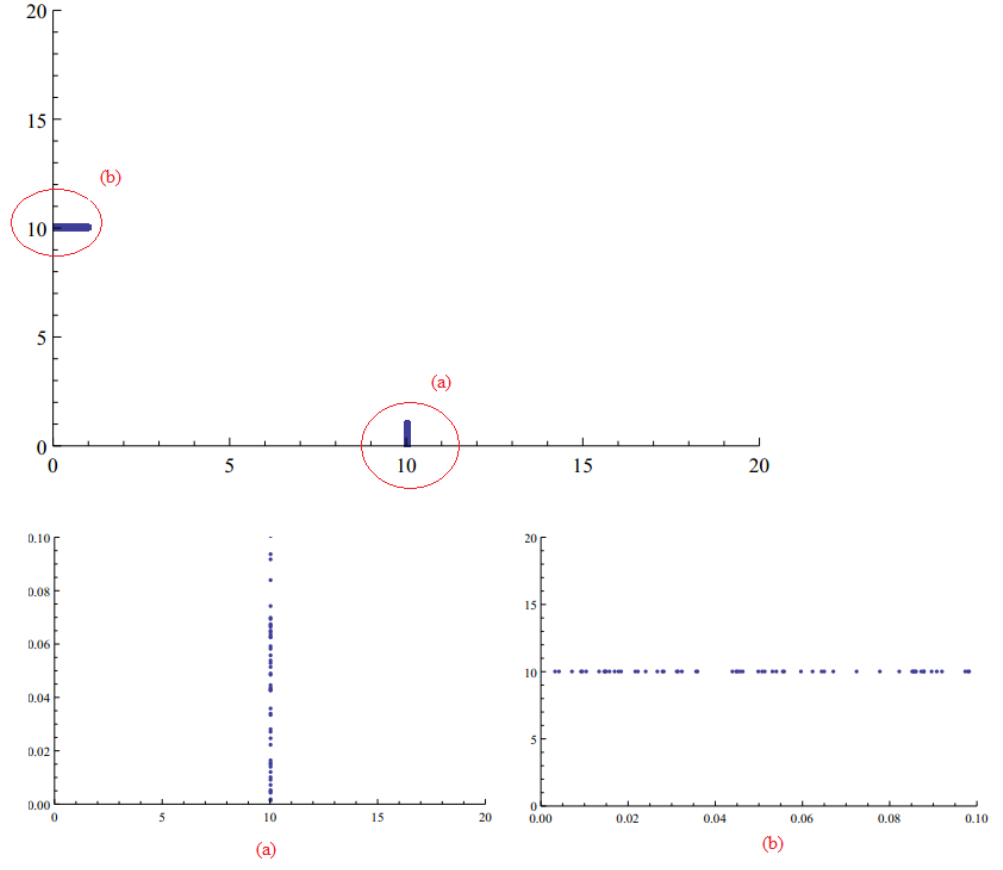
Tablo 2.13 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.34)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0064	0.0684	0.0178
10 000	0.6736	7.6058	0.1340
20 000	2.6158	27.8508	0.3256
40 000	10.1355	113.0834	0.6718
80 000	40.6530	449.0196	1.2516

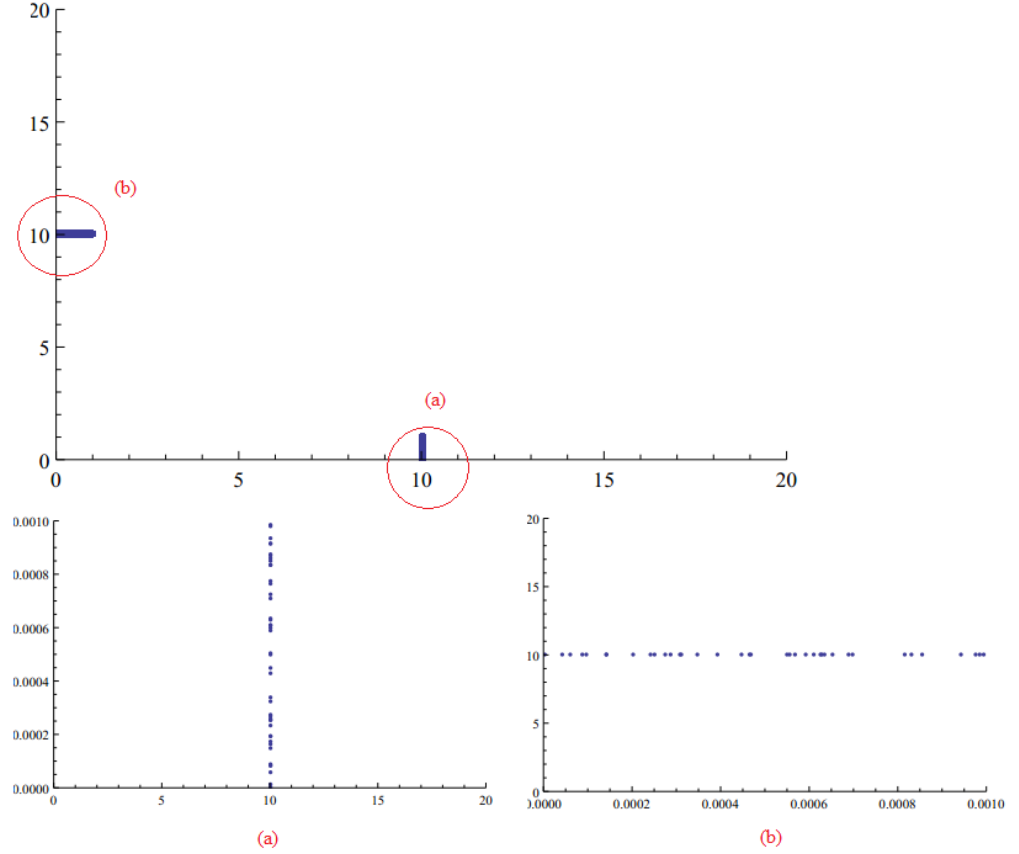
Buradaki tüm nokta kümeleri olabilecek en fazla çizgisel yığılım içerecek hale getirilmiştir. Böylece QuickCP algoritmasının çalışma süresi anlamında en kötü durumuna ulaşması hedeflenmiştir. Kümelerde oluşturulan çizgisel yığılımlar sayesinde QuickCP algoritmasının bitiş koşulunda kullanılan alt sınır değerleri burada da hep 0 olmuştur. QuickCP algoritması döngülerinden erken çıkamadığı için çalışma süresi anlamında KKA'nın da gerisine düşmüştür. Normalde noktaların X ve Y koordinat değerleri 0 ile 1.0 arasında değişmektedir. Çizgisel yığılım oluşturmak için nokta kümelerinde kümedeki noktaların yarısının X koordinatı 10.0, diğer yarısının Y koordinatı 10.0 olarak değiştirilmiştir. Veri tabanından elde edilen 1000, 10 000 ve 100 000 elemanlı nokta kümelerinin çizgisel yığılım içeren haliyle görüntüleri Şekil 2.35, Şekil 2.36 ve Şekil 2.37'de görülmektedir.



Şekil 2.35 Çizgisel yığılım içeren 1000 elemanlı tekdüze dağılımlı nokta kümesi



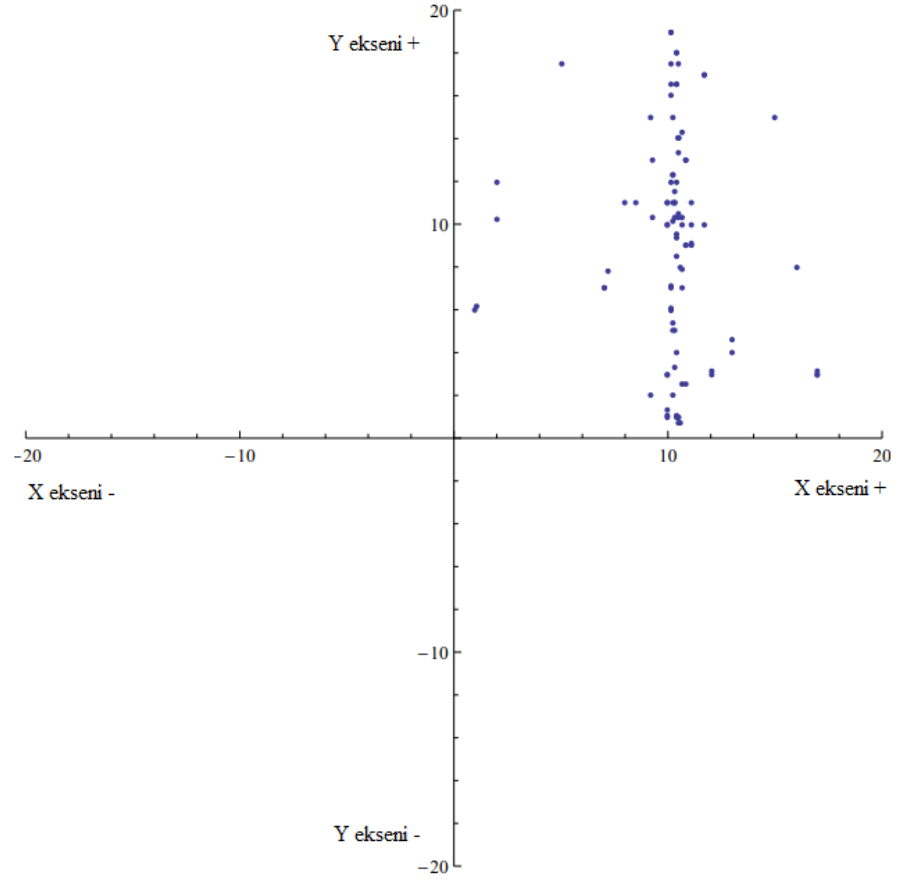
Şekil 2.36 Çizgisel yığılım içeren 10 000 elemanlı tekdüze dağılımlı nokta kümesi



Şekil 2.37 Çizgisel yığılım içeren 100 000 elemanlı tekdüze dağılımlı nokta kümesi

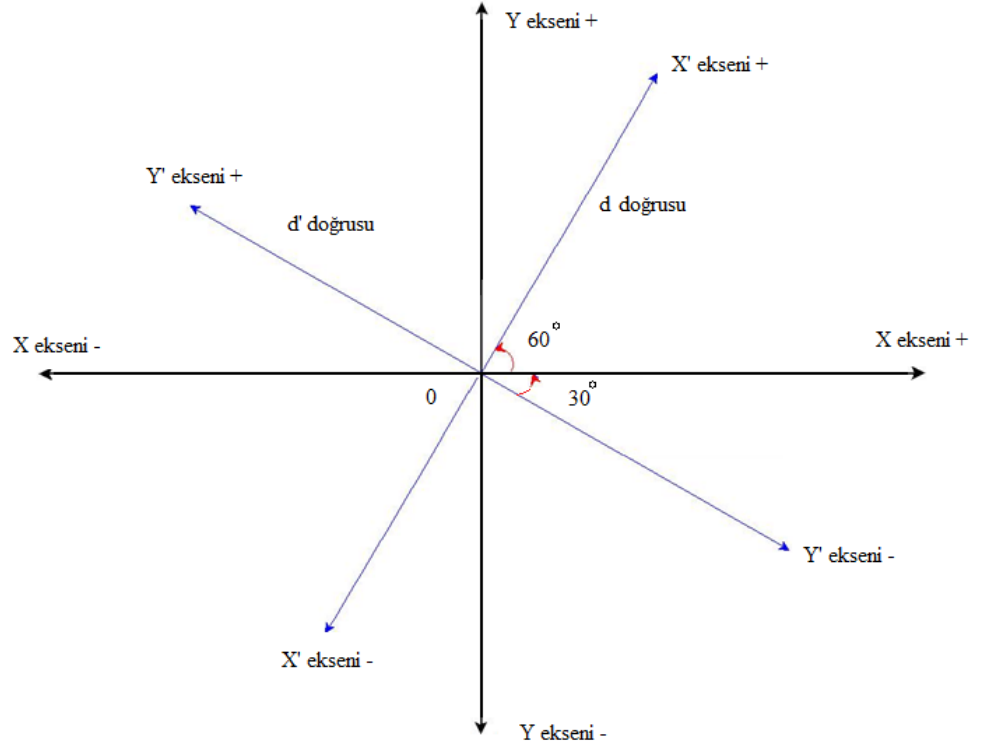
2.6 Rastgeleleştirilmiş QuickCP Algoritması

En yakın nokta çiftini QuickCP algoritması ile bulmaya başlamadan önce, nokta kümesinde yığılım olup olmadığını histogramla belirlenir. Nokta kümesinde performansı kötü etkileyecek bir çizgisel yığılım belirlenirse, QuickCP algoritmasını yavaşlatan ve hatta zaman zaman KKA'nın bile gerisine düşüren yığılım sorununun çözümü için ön işlem yapılmalıdır. Bu ön işlem, üzerinde en yakın iki nokta bulunacak nokta kümesinin koordinatlarının tanımlandığı koordinat sistemini rastgele açıyla kaydırmak ve yeni koordinat sisteminde noktaların yeni koordinatlarıyla QuickCP algoritmasını çalıştırmaktır. Bu adıma gelmeden önce noktaların farklı eksen doğrularına izdüşümünün, nokta kümesini yığılımdan nasıl kurtardığı incelenmelidir. Buna uygun örnek Şekil 2.38'de bulunmaktadır. Örnekteki nokta kümesi 100 elemanlıdır. Burada yer alan noktaların X koordinatları yaklaşık olarak 10.0 değerine yığılım göstermektedir.



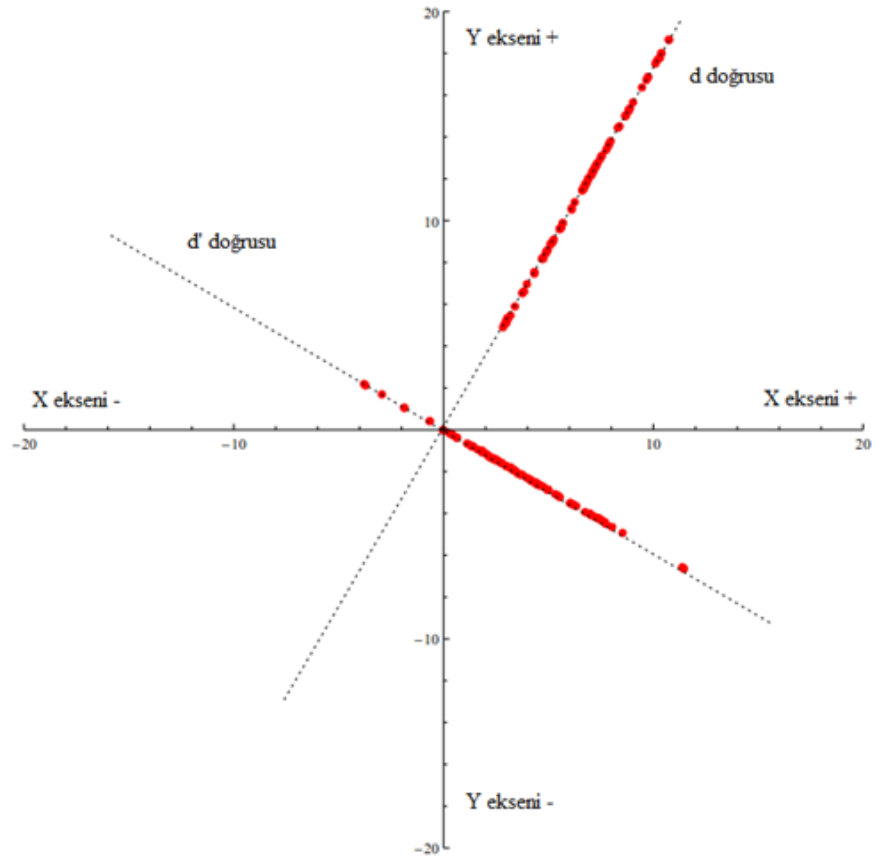
Şekil 2.38 Çizgisel Yığılım içeren örnek küme

Noktaların bulunduğu koordinat sistemi X ve Y eksenlerinden oluşmaktadır. Orijin $(0.0, 0.0)$ noktasıdır. X eksenine ile arasında α açısı olan bir d doğrusu ile bu doğruya dik d' doğrusu örnek olarak $\alpha = 60^\circ$ olacak şekilde Şekil 2.39'da resmedilmiştir. Burada yer alan d doğrusu yeni koordinat sisteminde X' eksen, d' doğrusu ise Y' eksen olacaktır.



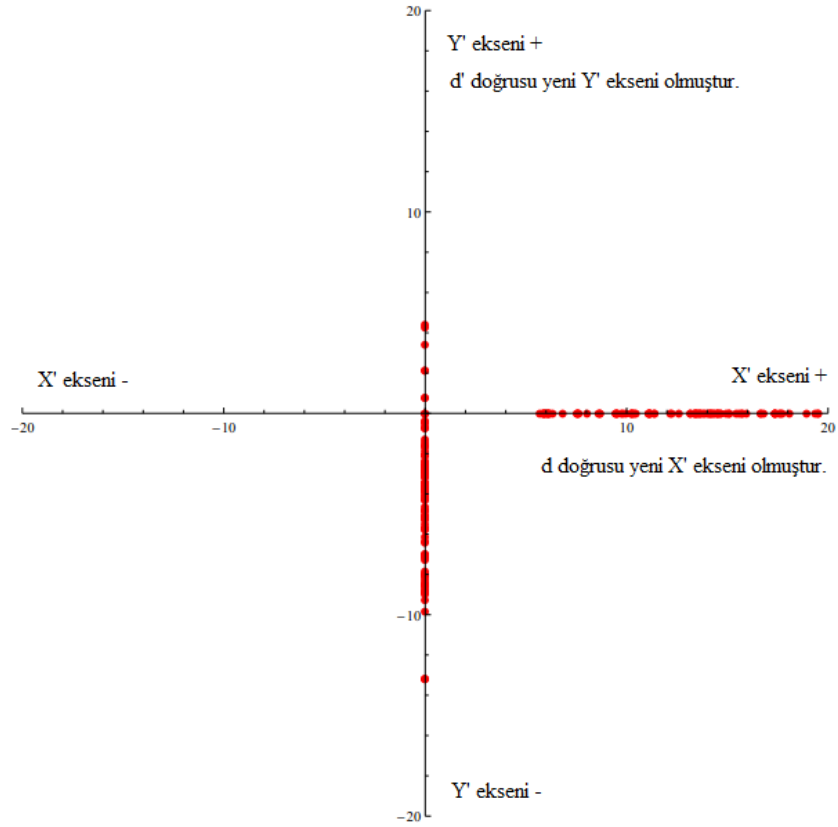
Şekil 2.39 Yeni koordinat sistemi örneği

Bununla birlikte d ve d' doğrusuna 100 elemanlı nokta kümesindeki noktaların izdüşümü yapıldığında Şekil 2.40'taki noktalar elde edilir.



Şekil 2.40 Noktaların yeni koordinat sistemine izdüşümü

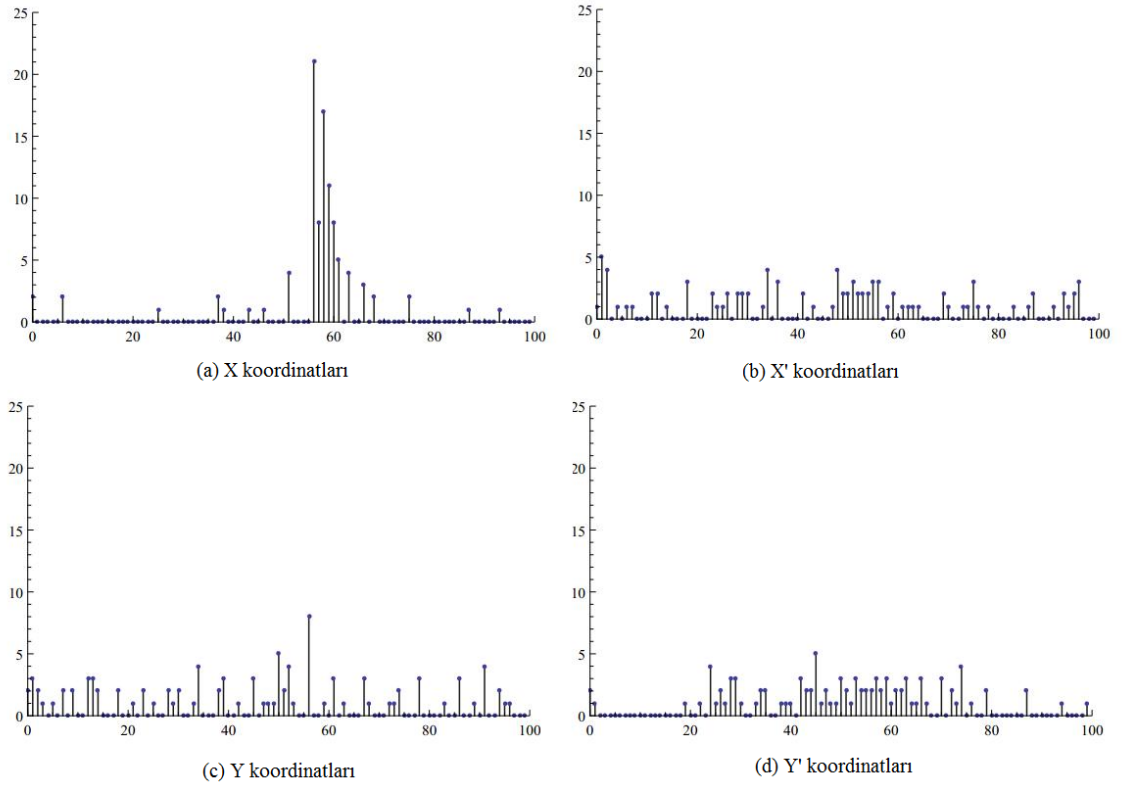
Şekil 2.41’de d ve d' doğrularından üretilen yeni X' ve Y' eksenleri ile noktaların bu eksenlere izdüşümleri resmedilmiştir. 100 elemanlı nokta kümesi X ve Y eksenlerinde 10.0 değerinde yığılım içerirken, artık yeni X' ve Y' eksenlerinde bu yığılım bulunmamaktadır.



Şekil 2.41 Yeni koordinat sisteminde yeni koordinatlar

X ve Y koordinatları ile, elde edilen yeni X' ve Y' koordinatları ayrı ayrı histogram yapılarak dağılımları Şekil 2.42'de görüntülenmiştir. Buna göre 100 elemanlı nokta kümesinde X eksenindeki 10.0 değerindeki nokta yığılması, orijinden geçen $\alpha = 60^\circ$ eğimli d doğrusuna ve ona dik olan d' doğrusuna noktaların izdüşümü yapıldığında artık yok olmaktadır. Grafiklerdeki değerler noktaların histogramdaki değerleridir. Yani örneğin Şekil 2.42'de (a)'da bulunan grafikte histogram, yatay olarak 60.0 değeri çevresinde yoğunlaşmış ve değer 56.0 iken en fazla 21'e kadar ulaşmıştır. Bu, histogramın frekans dizisinin 56. indeksine düşen 21 adet nokta bulunur, demektir. Bu küme için histogramın çarpım değeri 6.25 olarak bulunmuştur. Yığılmanın, frekans dizisinin 56. ile 64. indeksleri arasında olması ise noktaların X koordinatlarının değerlerinin 8.96 – 10.24 (56/6.25 ile 64/6.25) aralığında yoğunlaştığı anlamına gelmiştir.

Şekil 2.42'te (b)'de görüldüğü üzere elde edilen yeni koordinatlar yığılma içermemektedir ve artık QuickCP algoritmasının kullanımına hazır duruma gelmektedir.



Şekil 2.42 Örnek kümenin rastgeleleştirme öncesi ve sonrası histogramı

XY düzleminde koordinatlara sahip noktaların, yeni bir koordinat sistemine geçildiğinde sahip olacağı koordinatları bulmak için çeşitli işlemler yapılmıştır. Noktaların yerleri ve birbirleri arasındaki uzaklıklar bu işlemler sırasında hiçbir şekilde değişmemiştir. Yalnızca noktaların tanımlandığı koordinat sistemi değişmektedir.

Yeni bir koordinat sistemi için yeni X' ve Y' eksenlerine ve yeni bir orijine ihtiyaç vardır. Bu eksenler birbirine dik olmalıdır. Yani $\vec{v_1}, \vec{v_2}$ vektörlerinin yeni koordinat sistemindeki eksenlerin basis vektörleri olduğu varsayılırsa, $\vec{v_1} * \vec{v_2} = 0$ (nokta çarpım) ve $|\vec{v_1}| = 1, |\vec{v_2}| = 1$ (norm) olmalıdır (Lautrup, 2011). Yani yeni koordinat sistemini oluşturulacak basis vektörleri ortonormal olmalıdır.

İki boyutlu düzlemde XY koordinat sistemini standart basis vektörleri oluşturmaktadır. Bu vektörler X eksenini için $\vec{e_1} = (1,0)$ ve Y eksenini için $\vec{e_2} = (0,1)$ vektörleridir. Histogram sonucunda yığılım belirlenirse standart basise sahip koordinat sisteminin farkı basise sahip yeni bir koordinat sistemine çevrilmesi yeni basis vektörleri aracılığıyla sağlanır (Joy, 1999).

İki boyutlu düzlemde yani $\mathbb{R}^2$ 'de bu basis vektörlerinden ilki, normu 1 olacak şekilde rastgele olarak belirlenmiştir. Bu vektör formu a ve b reel sayı olmak üzere (a, b) ise, ikinci vektör nokta çarpımlarının (ing. dot product) 0 olması için $(-b, a)$ olarak seçilmiştir. Yüksek boyutlu uzaylarda basis vektörlerini bulmak için Gram-Schmidt yöntemi uygulanmıştır (Harvey Mudd College, 2013; Kuttler, 2012). Yeni basis vektörlerinin Gram-Schmidt yöntemi ile elde edilmesi ve bir noktanın yeni koordinat sistemindeki yeni koordinatlarının elde edilmesi işlemleri Bölüm 3'te k boyutlu ($k \geq 2$ ve $k \in \mathbb{N}$) uzay için ayrıntılı olarak anlatılmıştır.

2.7 Rastgeleleştirilmiş QuickCP Algoritması ve Diğer Algoritmaların Çalışma Süresi Sonuçları

Determinist algoritmalarda problemi doğru bir şekilde çözmek amaçlanırken, rastgeleştirme kullanan algoritmalarda girdiye ek olarak rastgele üretilen sayılar ve algoritmanın çalışması sırasında yapılan rastgele seçimler de çözüm için kullanılır. Böylece, rastgeleştirme kullanan algoritmanın davranışı aynı girdi iki kere kullanıldığında bile çeşitlilik gösterebilir.

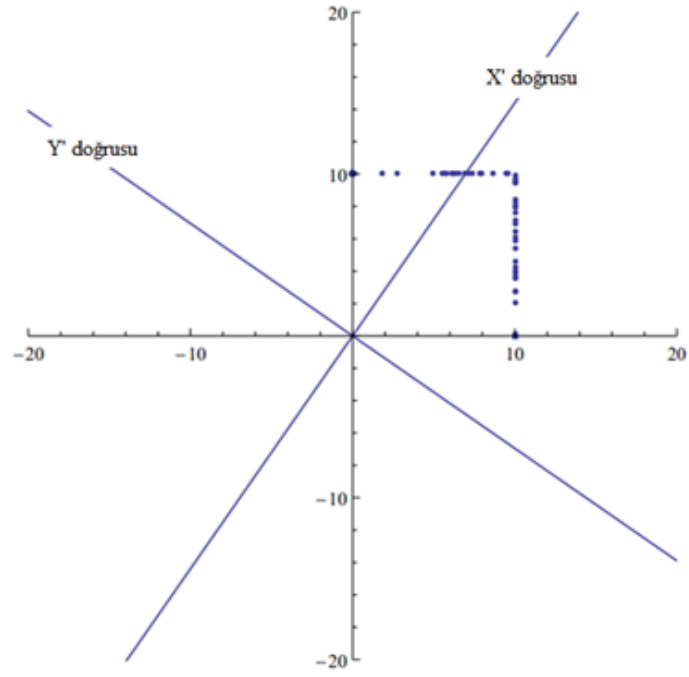
Rastgeleştirme kullanan algoritmalar Monte Carlo ve Las Vegas algoritmaları olarak da sınıflandırılır. Monte Carlo algoritmaları belirli sayıda adım içeren bir süreçte çalışır. Sonuçta elde edilen çıktının doğru olma olasılığı ise $1/2$ 'den büyüktür. Yani yine de hatalı sonuç elde etme olasılığı bulunur. Las Vegas algoritmaları ise daima doğru sonuç verir. Ancak çalışma süreleri değişkenlik gösterir (Karp, 1991).

Rastgeleştirme kullanan algoritmalar performans ve basitlik açısından deterministtik algoritmalara göre avantajlı olabilir. Bu duruma örnek olarak Quick Sort algoritması gösterilebilir. Quick Sort algoritması, sıralanması istenen sayı dizisi tersten sıralı iken çalışma süresi olarak kötü performans gösterir. Pivot olarak seçilen sayılar bu durumda diziyi etkin şekilde ayrıştıramaz. En kötü durum karmaşıklığı $O(n^2)$ olur. Ancak bu algoritmada pivot olarak seçilen sayılar rastgele olarak seçilirse, bu durumda tersten sıralı bir sayı dizisi bile her adımda dengeli ayrıştırılacağı için en kötü durumla karşılaşma olasılığı azalır. Bu anlamda Quick Sort, çalışma süresi değişkenlik gösterebilen ancak daima doğru sonuç veren Las Vegas tipinde bir rastgeleştirilmiş algoritma olur (Aspnes, 2013).

Bu anlamda QuickCP algoritmasını çalışma süresi olarak en kötü durumla karşılaşma olasılığını düşürmek adına X ve Y eksenleri yerine rastgele belirlenen ve belirli standartları taşıyan X' ve Y' eksenlerinde çalıştırmak, QuickCP'yi rastgeleleştirmek demektir. Bu işlem gerçekleştiğinde algoritmanın çalışma süresi değişse de, algoritma daima doğru sonuç vereceği için Las Vegas tipinde bir algoritma haline gelir. En kötü durumla karşılaşma olasılığı düşmekle birlikte yeni eksenlerde yığılma olmayacağından, algoritmanın performansı yığılma içermeyen durumdaki seviyesine ulaşır. Rastgele üretilen ortonormal basis vektörlerle oluşturulan yeni koordinat sisteminde, tüm noktalar yeni koordinatlarla tanımlandığında, QuickCP algoritmasının kullanımına hazır olmuş olur. Bu durumda rastgeleleştirilmiş QuickCP'nin çalışma süresine sadece ön işlem olarak noktaların yeni koordinat sistemindeki koordinatlarını hesaplamak eklenir.

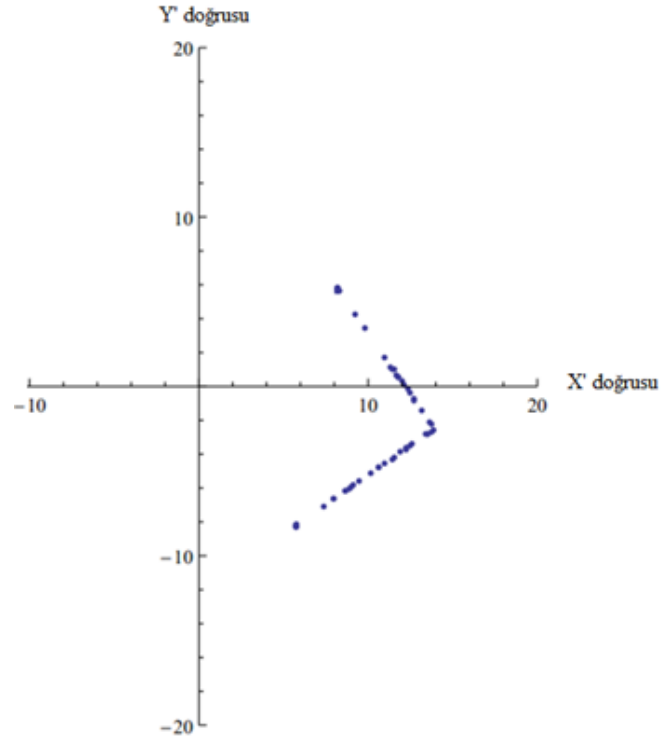
2.7.1 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın üstel dağılımlı kümelerde çalışma süreleri

Veri tabanından elde edilen nokta kümeleri çizgisel yığılma maksimum olacak şekilde değiştirilmişti. Burada, yığılma içeren nokta kümelerindeki noktaların koordinatları, ortonormal basisleri rastgele belirlenen yeni koordinat sistemindeki yeni koordinatlara dönüştürülmüştür. XY koordinat sisteminde X ve Y eksenlerinde belirli değerlere yığılan koordinatlar yeni koordinat eksenlerinde yığılma göstermemektedir. Örnek olarak üzerinde ölçüm yapılan 1000 elemanlı nokta kümesinin yığılma içeren hali Şekil 2.43'te yer almaktadır. Burada başlangıçta 1000 tane noktanın yarısının X koordinatı 10.0'dur ve Y koordinatları 0.0 ile 10.0 arasında değişmektedir. Diğer yarısının ise Y koordinatı 10.0'dur, X koordinatı ise 0.0 ile 10.0 arasında değişmektedir. Şekilde yer alan X' doğrusunun ortonormal basis vektörü $(0.5706, 0.8212)$ ve Y' doğrusunun ortonormal basis vektörü $(-0.8212, 0.5706)$ olmakla birlikte bu iki vektör, yeni koordinat sisteminin basis vektörleridir.



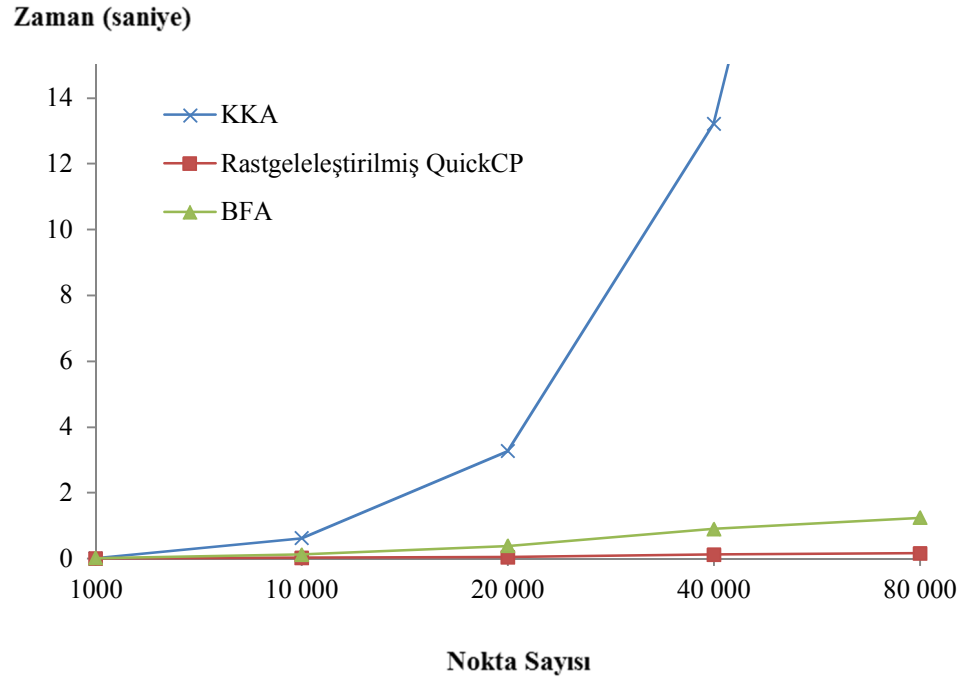
Şekil 2.43 Üstel dağılımlı çizgisel yığılma içeren 1000 elemanlı küme

Şekil 2.44'te ise çizgisel yığılma içeren 1000 elemanlı nokta kümesinin $(0.5706, 0.8212)$ ve $(-0.8212, 0.5706)$ basis vektörlü yeni koordinat sistmindeki görüntüsü yer almaktadır. Nokta kümesi yeni koordinat sisteminin X' ve Y' eksenlerinde artık çizgisel yığılma oluşturmamaktadır.



Şekil 2.44 Üstel dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü

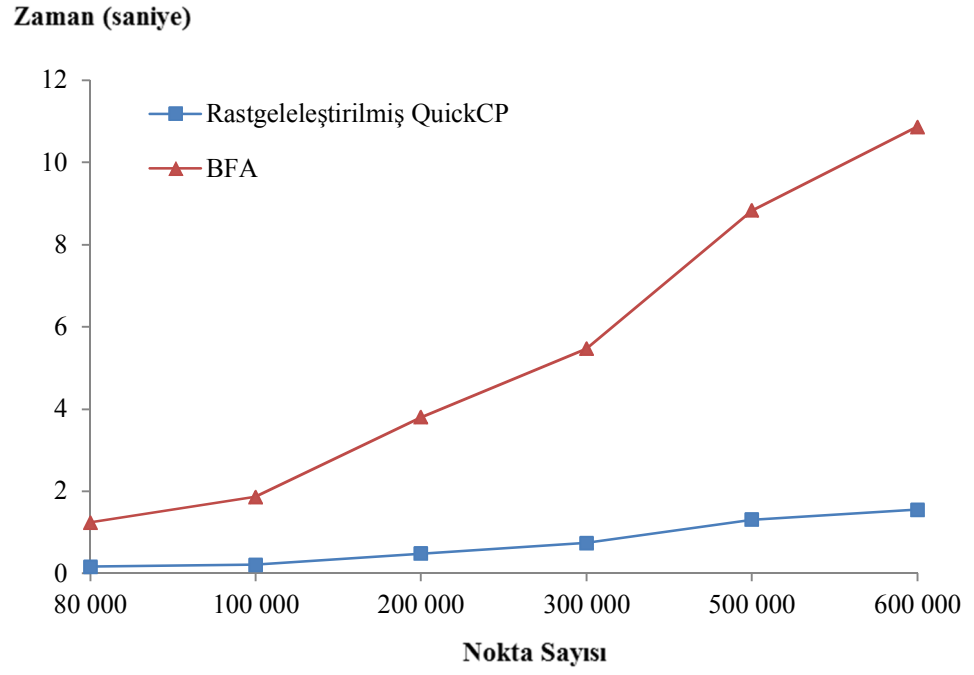
Yığılım içeren nokta kümelerindeki noktaların koordinatları yeni koordinat sistemlerine göre hazırlandıktan sonra, rastgeleleştirilmiş QuickCP algoritması, KKA ve BFA 1000 eleman içeren kümeden 2 500 000 içeren kümeye kadar çeşitli boyutlarda elemanla çalıştırılmıştır. Sonuçlar Şekil 2.45, Şekil 2.46 ve Şekil 2.47’de yer aldığı üzere, ön işleme rastgeleştirme işlemleri eklenen QuickCP, BFA ve KKA’dan çalışma süresi anlamında daha iyi performans göstermiştir. KKA, 80 000 elemanlı kümeden sonra çalışma süresi çok fazla olduğundan hesaplamalara katılmamıştır.



Şekil 2.45 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması

Tablo 2.14 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.45)

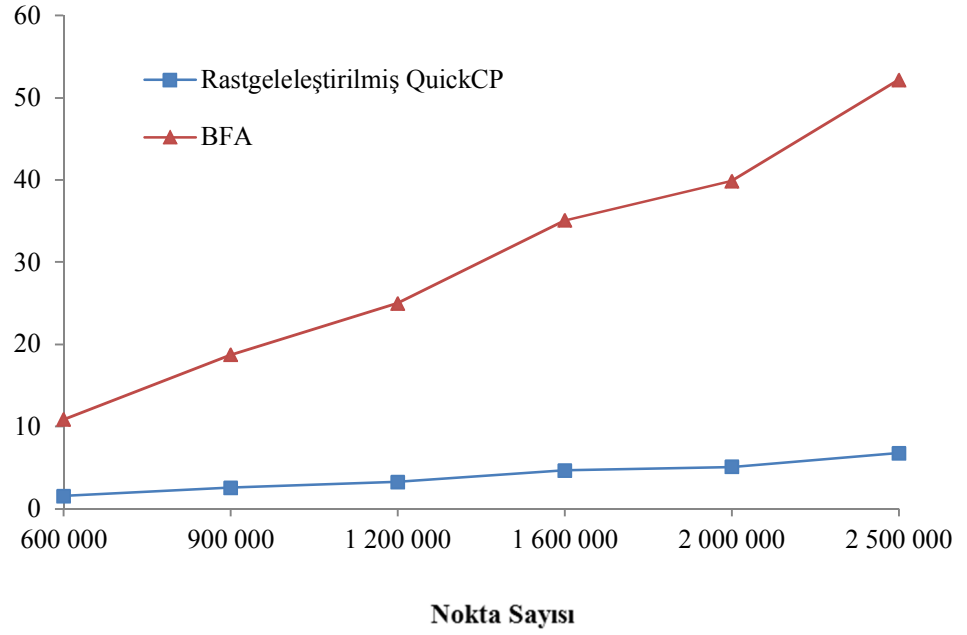
Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.0060	0.0036	0.0158
10 000	0.6170	0.0210	0.1270
20 000	3.2668	0.0490	0.3882
40 000	13.2182	0.1204	0.8988
80 000	38.4188	0.1650	1.2378



Şekil 2.46 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışması

Tablo 2.15 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.46)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
80 000	0.1650	1.2378
100 000	0.2128	1.8656
200 000	0.4844	3.8030
300 000	0.7440	5.4696
500 000	1.3086	8.8348
600 000	1.5510	10.8616

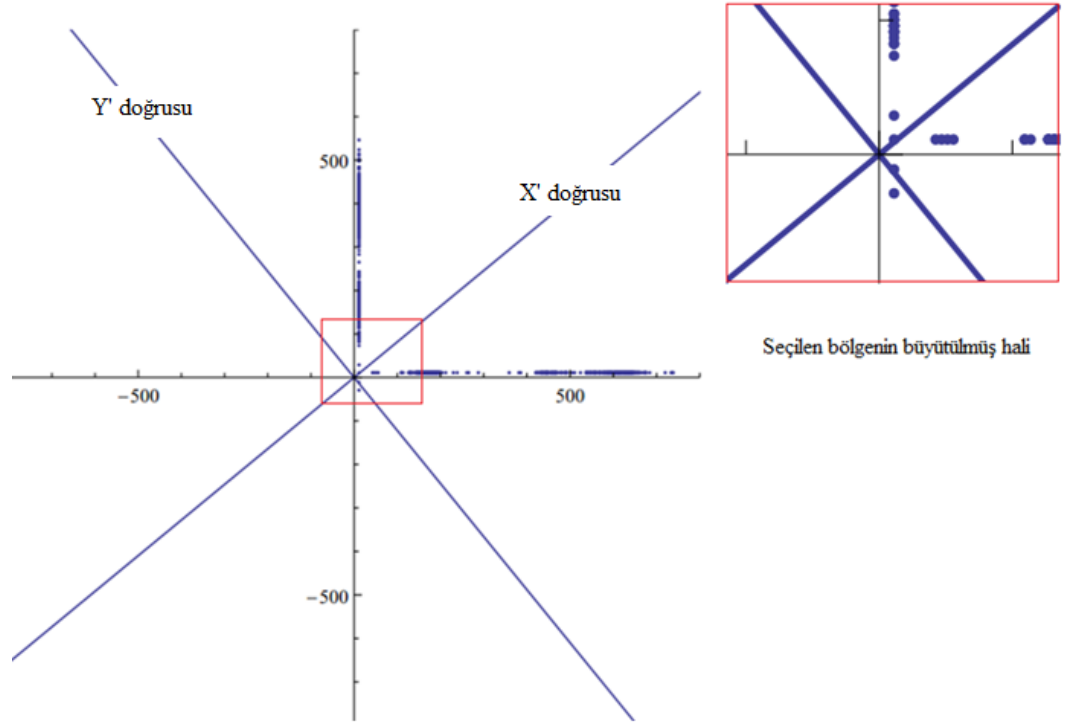
Zaman (saniye)Şekil 2.47 $\mathbb{R}^2$ 'de çizgisel birikim içeren üstel dağılımlı kümede algoritmaların çalışmasıTablo 2.16 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.47)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
600 000	1.5510	10.8616
900 000	2.5616	18.7142
1 200 000	3.2602	25.0164
1 600 000	4.6782	35.0962
2 000 000	5.1172	39.8602
2 500 000	6.7968	52.1570

2.7.2 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın normal dağılımlı nokta kümelerinde çalışma süreleri

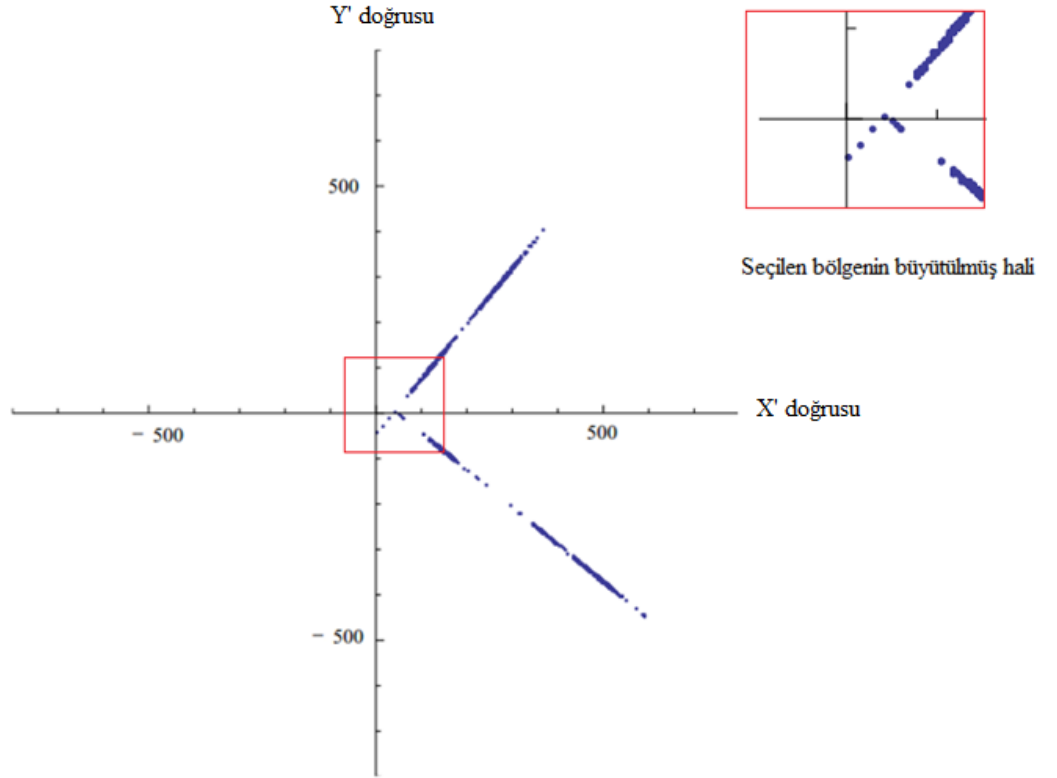
İki boyutlu nokta üreten uygulama ile elde edilen nokta kümeleri çizgisel yığılım maksimum olacak şekilde değiştirilmişti. Burada da yığılım içeren nokta kümelerindeki noktaların koordinatları, ortonormal basisleri rastgele belirlenen yeni koordinat sisteminde yeni koordinatlara dönüştürülmüştür. XY koordinat sisteminde X ve Y eksenlerinde belirli değerlere yığılan koordinatlar yeni koordinat eksenlerinde yığılım göstermemektedir. Örnek olarak üzerinde ölçüm yapılan 1000 elemanlı nokta kümesinin yığılım içeren hali Şekil 2.48'de yer almaktadır. Burada başlangıçta 1000 tane noktanın yarısının X koordinatı

10.0'dur ve Y koordinatları 0.0 ile 600.0 arasında değişmektedir. Diğer yarısının ise Y koordinatı 10.0'dur, X koordinatı ise 0.0 ile 800.0 arasında değişmektedir. Şekilde yer alan X' doğrusunun ortonormal basis vektörü $(0.7734, 0.6339)$ ve Y' doğrusunun ortonormal basis vektörü $(-0.6339, 0.7734)$ olmakla birlikte bu iki vektör, yeni koordinat sisteminin basis vektörleridir.



Şekil 2.48 Normal dağılımlı çizgisel yığılım içeren 1000 elemanlı küme

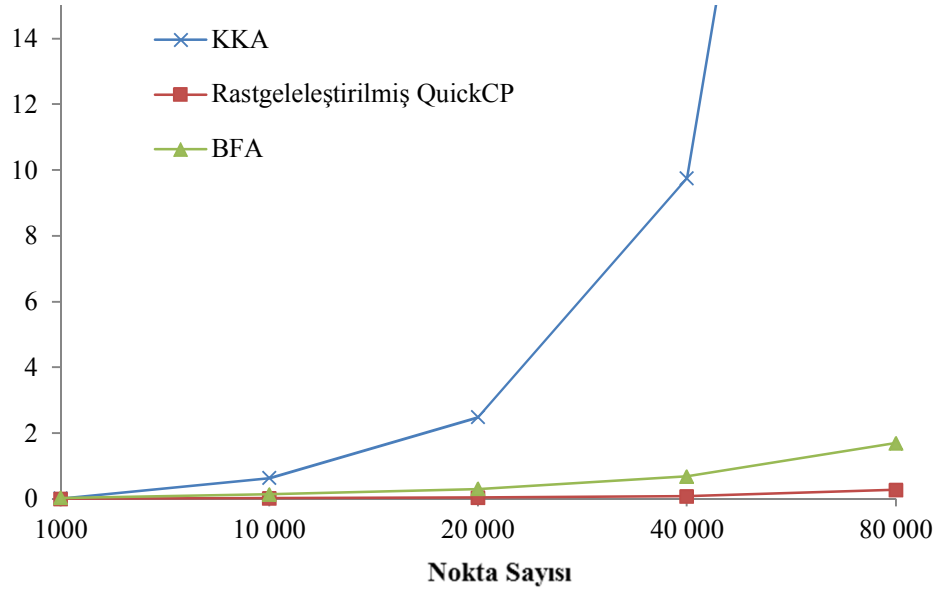
Şekil 2.49'da ise çizgisel yığılım içeren 1000 elemanlı nokta kümesinin $(0.7734, 0.6339)$ ve $(-0.6339, 0.7734)$ basis vektörlü yeni koordinat sistmindeki görüntüsü yer almaktadır. Nokta kümesi yeni koordinat sisteminin X' ve Y' eksenlerinde artık çizgisel yığılım oluşturmamaktadır.



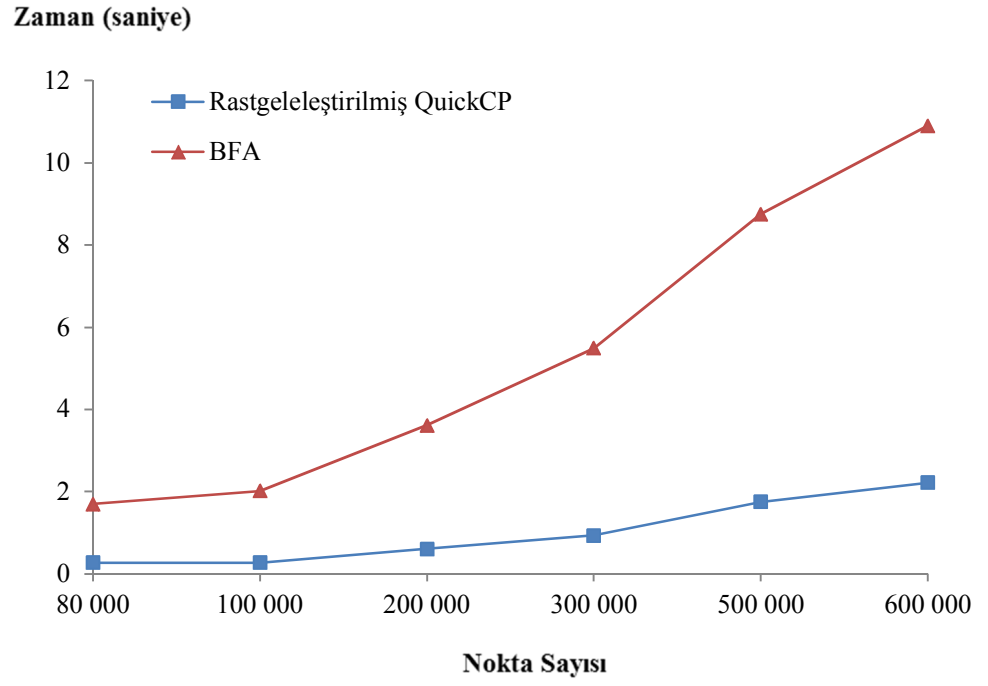
Şekil 2.49 Normal dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü

Yığılım içeren nokta kümelerindeki noktaların koordinatları yeni koordinat sistemlerine göre hazırlandıktan sonra, rastgeleleştirilmiş QuickCP algoritması, KKA ve BFA 1000 eleman içeren kümeden 2 500 000 içeren kümeye kadar çeşitli boyutlarda elemanla çalıştırılmıştır. Sonuçlar Şekil 2.50, Şekil 2.51 ve Şekil 2.52’de yer aldığı üzere, ön işlemine rastgeleleştirme işlemleri eklenen QuickCP, BFA ve KKA’dan çalışma süresi anlamında daha iyi performans göstermiştir. KKA, 80 000 elemanlı kümeden sonra çalışma süresi çok fazla olduğundan hesaplamalara katılmamıştır.

Zaman (saniye)

Şekil 2.50 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışmasıTablo 2.17 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.50)

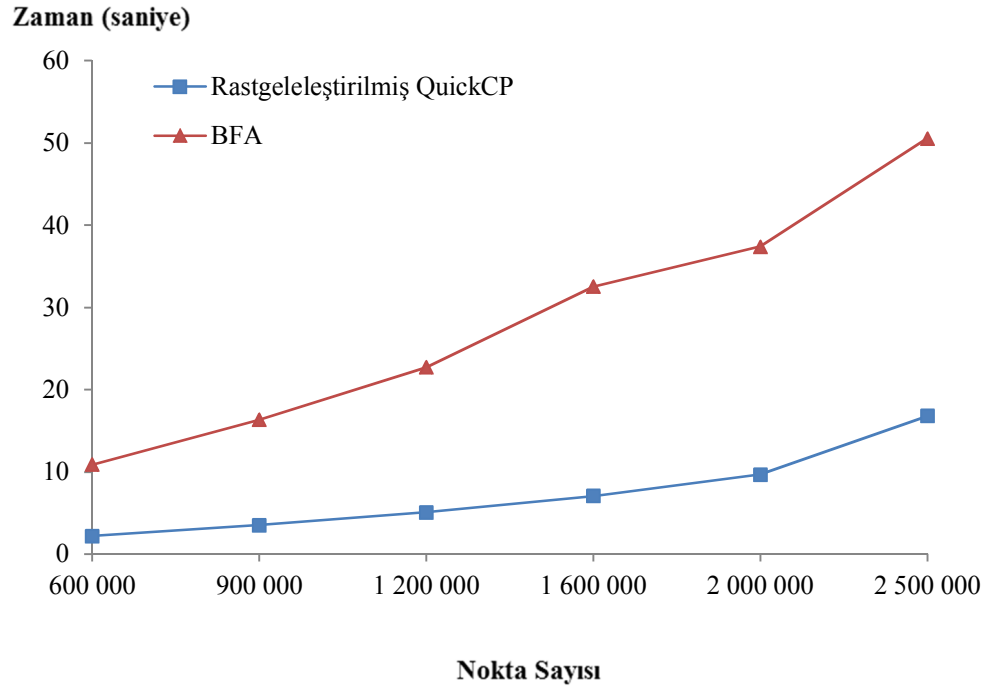
Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.0102	0.0030	0.0288
10 000	0.6328	0.0224	0.1414
20 000	2.4814	0.0472	0.3034
40 000	9.7440	0.0864	0.6846
80 000	47.6876	0.2740	1.6974



Şekil 2.51 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması

Tablo 2.18 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.51)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
80 000	0.2740	1.6974
100 000	0.2748	2.0206
200 000	0.6090	3.6168
300 000	0.9390	5.4888
500 000	1.7568	8.7562
600 000	2.2198	10.9016



Şekil 2.52 $\mathbb{R}^2$ 'de çizgisel birikim içeren normal dağılımlı kümede algoritmaların çalışması

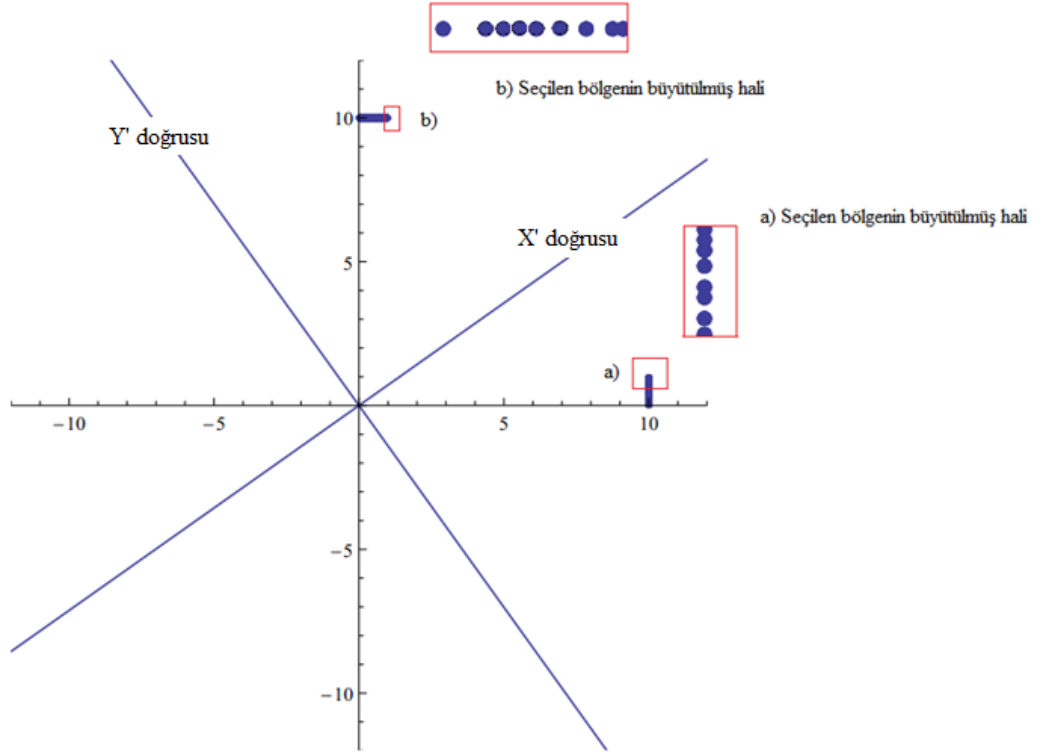
Tablo 2.19 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.52)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
600 000	2.2198	10.9016
900 000	3.5432	16.3458
1 200 000	5.0866	22.7386
1 600 000	7.0594	32.5498
2 000 000	9.6876	37.3916
2 500 000	16.8248	50.5486

2.7.3 Rastgeleleştirilmiş QuickCP, KKA ve BFA'nın tekdüze dağılımlı nokta kümelerinde çalışma süreleri

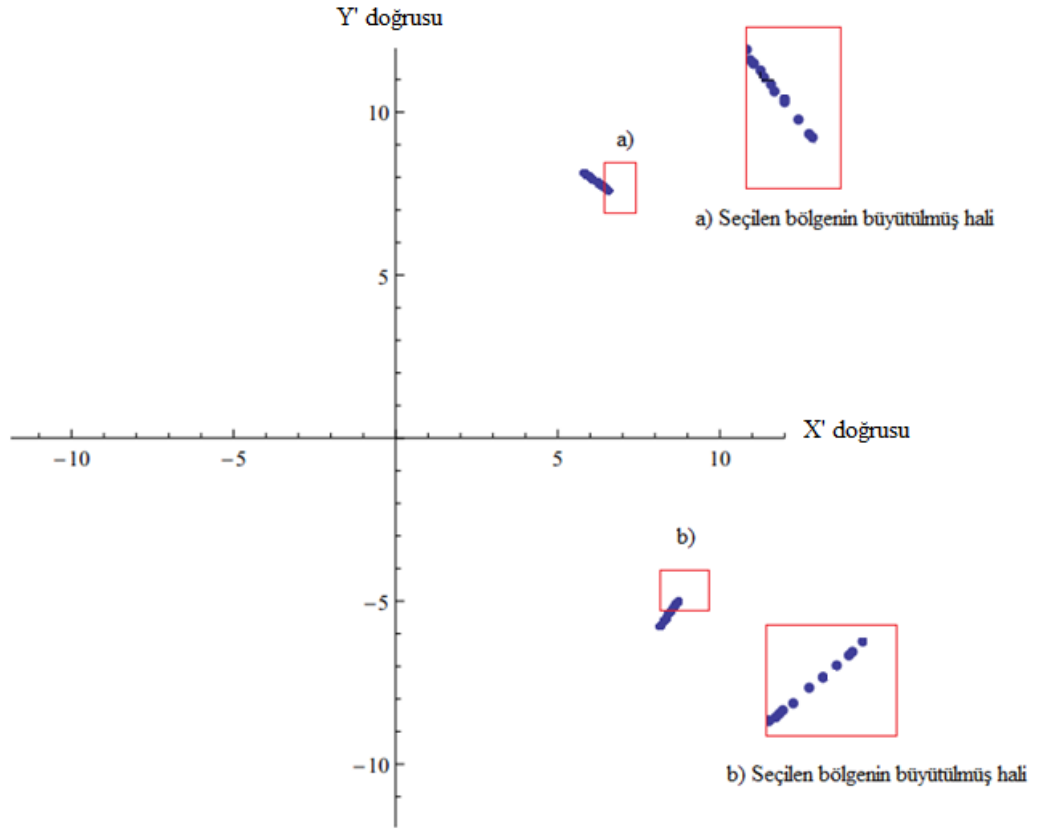
Tekdüze dağılımlı nokta kümeleri çizgisel yığılma maksimum olacak şekilde değiştirilmiştir. Burada da yığılma içeren nokta kümelerindeki noktaların koordinatları, ortonormal basisleri rastgele belirlenen yeni koordinat sisteminde yeni koordinatlara dönüştürülmüştür. XY koordinat sisteminde X ve Y eksenlerinde belirli değerlere yığılan koordinatlar yeni koordinat eksenlerinde böyle bir yığılma göstermemektedir. Örnek olarak üzerinde ölçüm yapılan 1000 elemanlı nokta kümesinin yığılma içeren hali Şekil 2.53'te yer almaktadır. Burada başlangıçta 1000 tane noktanın yarısının X koordinatı 10.0'dur ve Y koordinatları

0.0 ile 1.0 arasında değişmektedir. Diğer yarısının ise Y koordinatı 10.0'dur, X koordinatı ise 0.0 ile 1.0 arasında değişmektedir. Şekilde yer alan X' doğrusunun ortonormal basis vektörü $(0.8144, 0.5803)$ ve Y' doğrusunun ortonormal basis vektörü $(-0.5803, 0.8144)$ olmakla birlikte bu iki vektör, yeni koordinat sisteminin basis vektörleridir.



Şekil 2.53 Tekdüze dağılımlı çizgisel yığılım içeren 1000 elemanlı küme

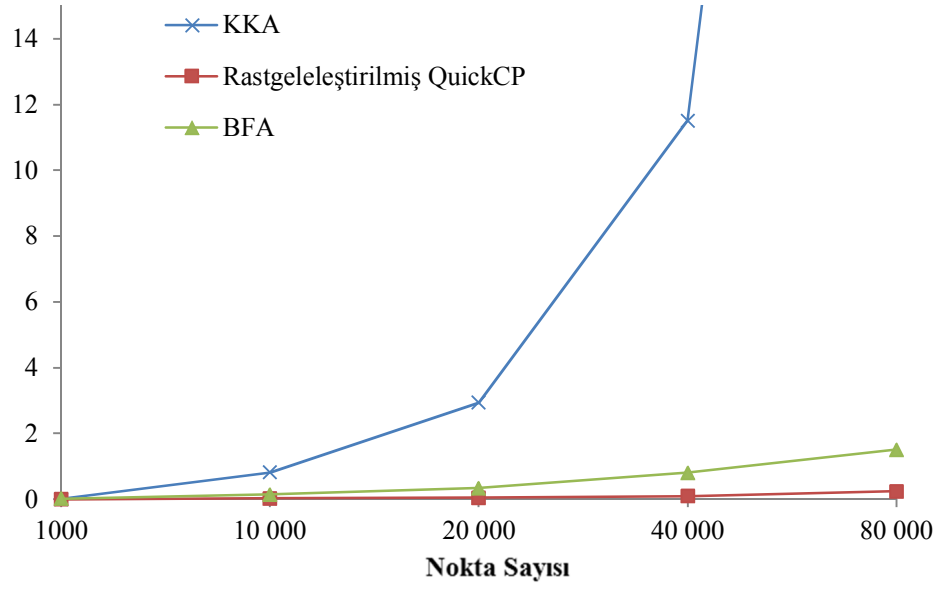
Şekil 2.54'te ise çizgisel yığılım içeren 1000 elemanlı nokta kümesinin $(0.8144, 0.5803)$ ve $(-0.5803, 0.8144)$ basis vektörlü yeni koordinat sistmindeki görüntüsü yer almaktadır. Nokta kümesi yeni koordinat sisteminin X' ve Y' eksenlerinde artık çizgisel yığılım oluşturmamaktadır.



Şekil 2.54 Tekdüze dağılımlı çizgisel yığılım içeren 1000 elemanlı kümenin yeni koordinat sistemindeki görüntüsü

Yığılım içeren nokta kümelerindeki noktaların koordinatları yeni koordinat sistemlerine göre hazırlandıktan sonra, rastgeleleştirilmiş QuickCP algoritması, KKA ve BFA 1000 eleman içeren kümeden 2 500 000 içeren kümeye kadar çeşitli boyutlarda elemanla çalıştırılmıştır. Sonuçlar Şekil 2.55, Şekil 2.56 ve Şekil 2.57’de yer aldığı üzere, ön işlemine rastgeleştirme işlemleri eklenen QuickCP, BFA ve KKA’dan daha iyi performans göstermiştir. KKA, 80 000 elemanlı kümeden sonra çalışma süresi çok fazla olduğundan hesaplamalara katılmamıştır.

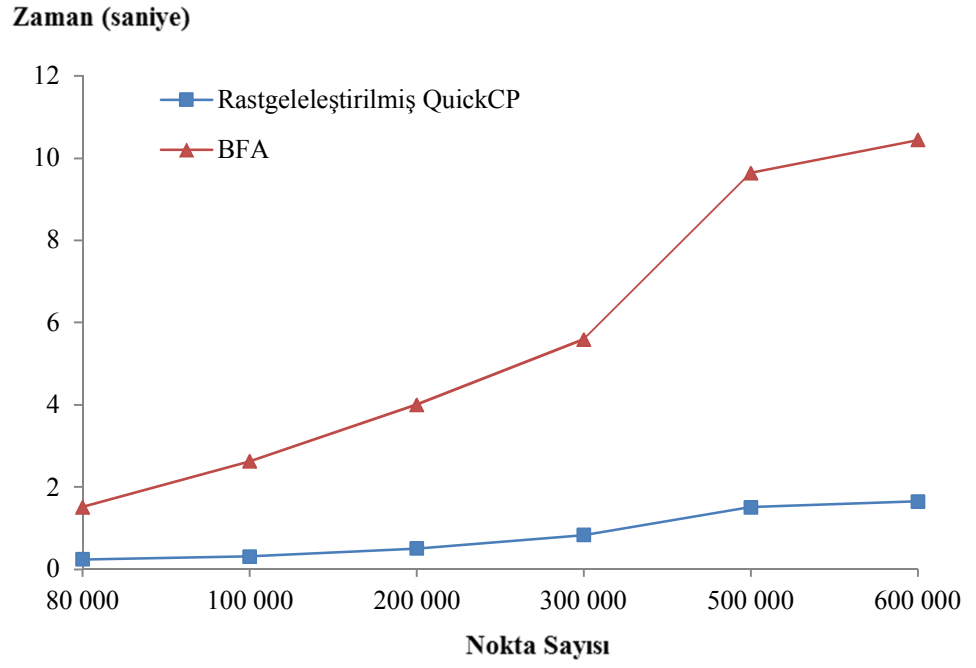
Zaman (saniye)



Şekil 2.55 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması

Tablo 2.20 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.55)

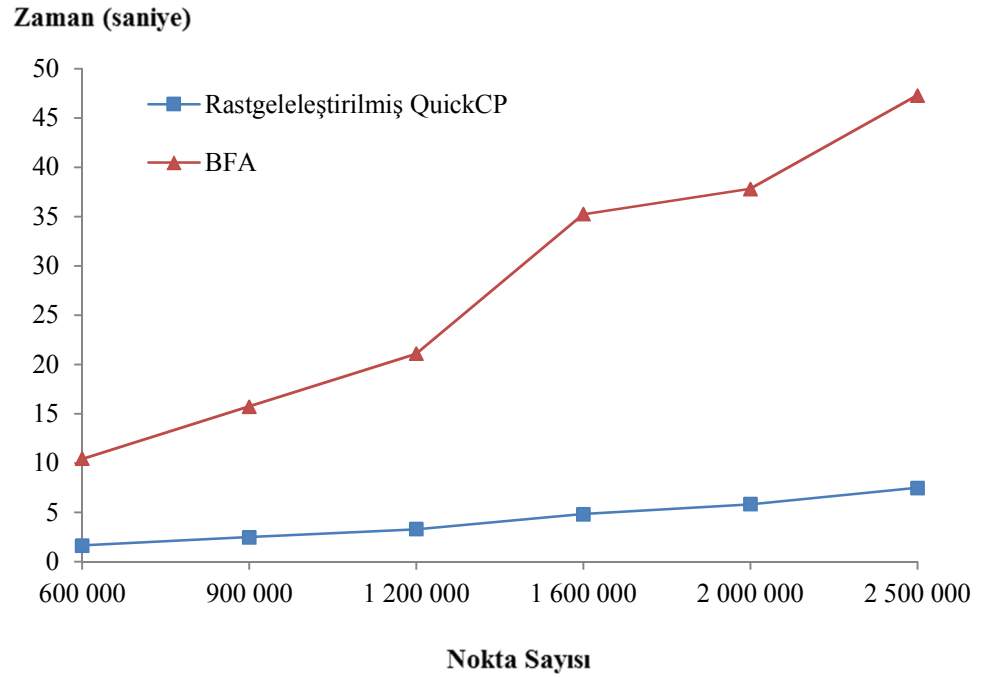
Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.0060	0.0016	0.0170
10 000	0.8116	0.0226	0.1400
20 000	2.9326	0.0500	0.3370
40 000	11.5126	0.0986	0.8020
80 000	62.3154	0.2418	1.5074



Şekil 2.56 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması

Tablo 2.21 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.56)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
80 000	0.2418	1.5074
100 000	0.3082	2.6268
200 000	0.5052	3.9986
300 000	0.8296	5.5924
500 000	1.5070	9.6408
600 000	1.6484	10.4414



Şekil 2.57 $\mathbb{R}^2$ 'de çizgisel birikim içeren tekdüze dağılımlı kümede algoritmaların çalışması

Tablo 2.22 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.57)

Nokta Sayısı	R. QuickCP(sn)	BFA(sn)
600 000	1.6484	10.4414
900 000	2.5114	15.7510
1 200 000	3.3020	21.0946
1 600 000	4.8396	35.2408
2 000 000	5.8276	37.8044
2 500 000	7.4818	47.2568

2.8 Nokta Kümesindeki Çizgisel Birikim Durumuna Göre Algoritma Seçimi

QuickCP algoritması ile rastgeleleştirilmiş QuickCP algoritması nokta kümesindeki dağılıma göre yer yer farklı performans göstermektedir. Çizgisel birikimin histogramla belirlenecek şekilde QuickCP algoritmasının performansını kötü etkilemeyeceği durumlarda rastgeleleştirme işlemine gerek yoktur. Ancak histogramla QuickCP algoritmasının performansını kötü etkileyecek bir nokta dağılımı belirlendiğinde rastgeleleştirilmiş QuickCP algoritması kullanılmalıdır. Bu anlamda histogramla incelenen nokta kümesinde QuickCP ya da rastgeleleştirilmiş QuickCP algoritması arasında seçim yapacak şekilde bir

birleştirme algoritması düşünülebilir. Şekil 2.58’de gerektiğinde QuickCP ve rastgeleştirilmiş QuickCP algoritmaları arasında seçim yapacak birleştirici bir algoritma tasarlanmıştır. Algoritmaların farklı dağılımlı nokta kümelerinde davranışları gözlemlenmiştir. Sonuç olarak, çizgisel yığılmanın QuickCP algoritmasının performansını kötü etkileyeceği durumlarda birleştirici algoritmanın kullanılması düşünülebilir.

```

/* P = {P1, P2, .... Pn} where Pi = (xi, yi) i.e.. n input
points.

"is_there_linear_accumulation" is the flag which indicates the
result whether there is any linear accumulation in P. if the
flag is 1. then there is a linear accumulation in P.

*/

is_there_linear_accumulation = Histogram(P)

if( is_there_linear_accumulation = 0 )

QuickCP(P);

else

Randomized_QuickCP(P);

```

Şekil 2.58 Birleştirilmiş QuickCP algoritması kodu

$\mathbb{R}^2$ uzayında histogram fonksiyonunun maliyeti ile noktaları farklı bir koordinat sistemine çevirme işlemi arasında çalışma zamanı süresi anlamında pek fark bulunmamaktadır.

2.8.1 Histogram ile rastgeleştirme işlemleri

QuickCP algoritması çalıştırılmadan önce öngörülen işlemler, histogram yapımı ile rastgeleştirme işlemidir. Bu iki işlemin QuickCP algoritmasının çalışma zamanına etkisi, işlemlerin karmaşıklığı belirlenerek yorumlanabilir. Şekil 2.59’da görüntülenen histogram kodunun karmaşıklığı $O(n)$ ’dir.

```

/* P = {P1. P2. .... Pn} where Pi = (xi. yi) i.e.. n input
points. */

Min = infinity; Max = -infinity;

for i from 1 to n

    if( Min > P(i) ) then Min = P(i);

    if( Max < P(i) ) then Max = P(i);

end for

freqsize = Max - Min;

hash_product = n / freqsize;

for i from 1 to n

    histogram(i) = 0;

end for

for i from 1 to n

    difference = P(i) - min;

    index = difference * hash_product;

    histogram(index)++;

end for

```

Şekil 2.59 Histogram sözde kodu

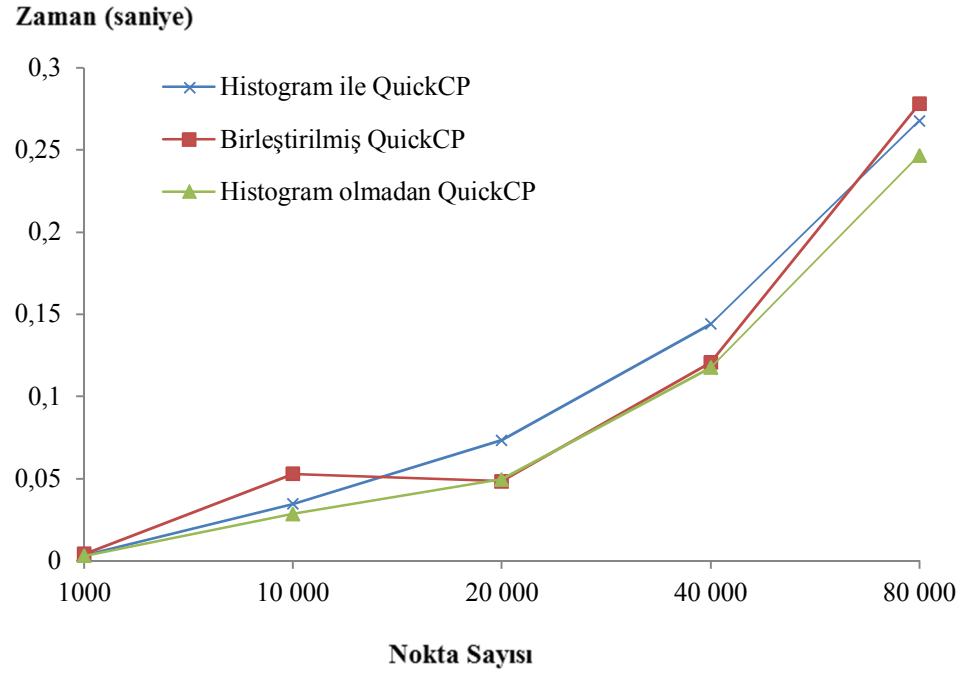
Rastgeleleştirme işlemi için yapılan iki işlem vardır. Birinci işlemde basis vektörleri elde edilir. Bu işlemin maliyeti $O(k^3)$ 'tür. İkinci işlemde elde edilen basis vektörlerinden oluşturulan bir matris nokta kümesindeki tüm noktalarla çarpılır. Bu işlemin maliyeti ise $O(k^2 * n)$ 'dir. Sonuç olarak Ek 6'da yer alan rastgeleleştirme kodunun karmaşıklığı $O(k^3 + k^2 * n)$ 'dir.

2.8.2 Üstel dağılımlı kümelerde algoritma seçimi

Şekil 2.60. Şekil 2.61 ve Şekil 2.62'de yer alan grafikte QuickCP algoritmasının üç şeklinin çalışma süreleri, üstel dağılımlı ve çeşitli büyüklükte nokta içeren kümelerde çalıştırılarak görüntülenmiştir. "Histogram ile QuickCP" algoritmasının çizgisinde, QuickCP algoritmasının çalışma süresi ile nokta kümesi için yapılan histogram fonksiyonunun çalışma süresinin toplamı yer almaktadır. "Birleştirilmiş QuickCP" algoritmasının çizgisinde QuickCP ve rastgeleleştirilmiş

QuickCP algoritmasını birleştiren algoritmanın çalışma süresi yer almaktadır (Bkz Şekil 2.58). “Histogram olmadan QuickCP” algoritmasının çizgisinde sadece QuickCP algoritmasının çalışma süresi yer almaktadır.

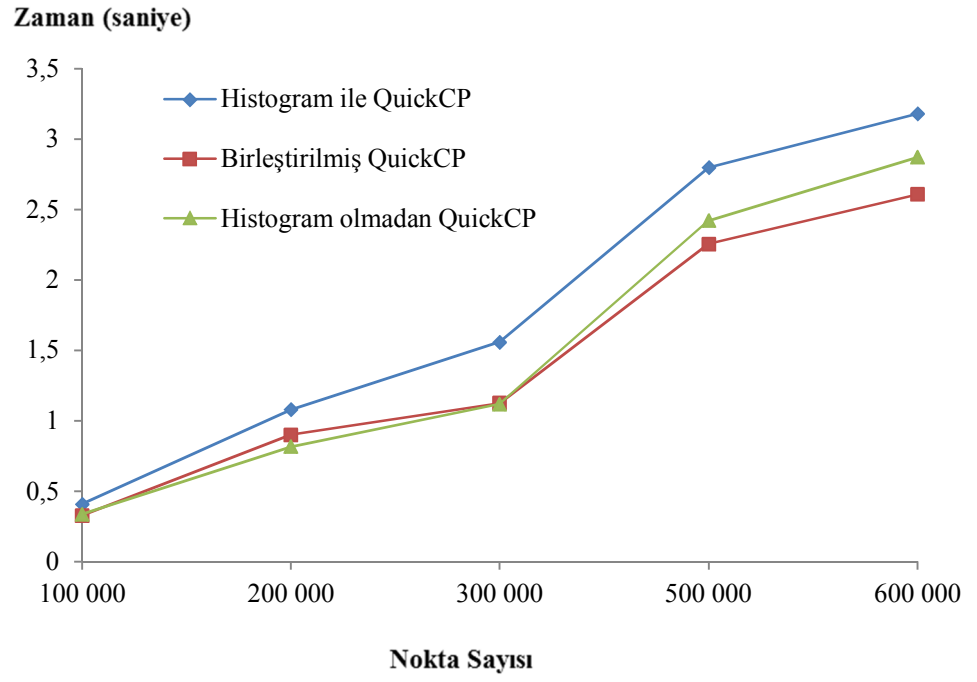
Üç algoritma da zaman zaman birbirlerini geçmiş ancak genelde benzer sürede çalışmıştır.



Şekil 2.60 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.23 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.60)

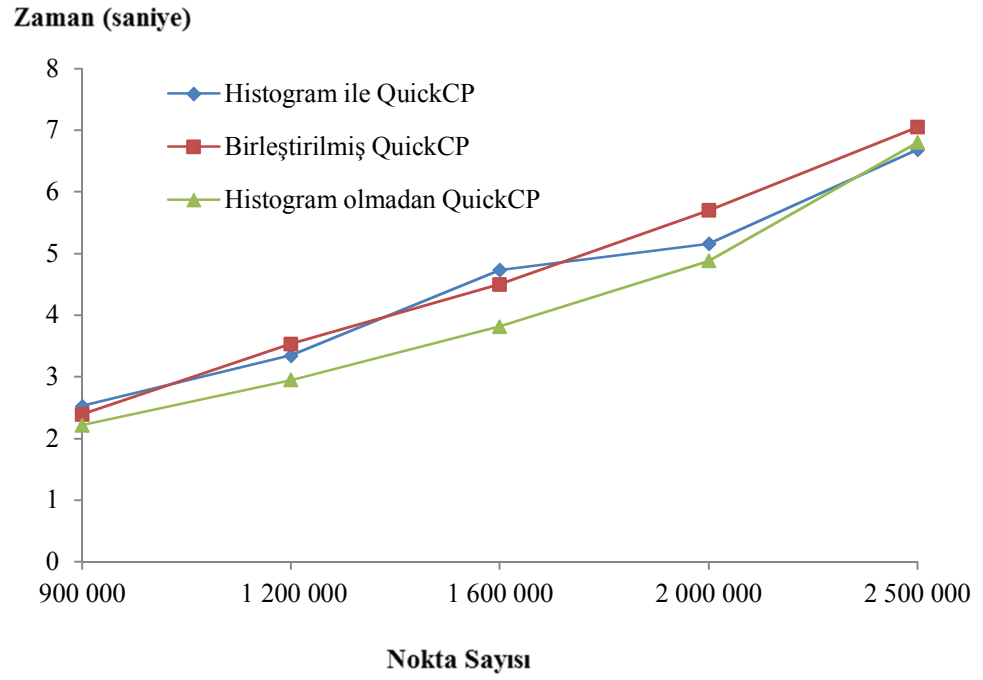
Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
1000	0.0034	0.0042	0.0032
10 000	0.0346	0.0530	0.0286
20 000	0.0734	0.0484	0.0496
40 000	0.1440	0.1206	0.1176
80 000	0.2676	0.2780	0.2466



Şekil 2.61 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.24 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.61)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
100 000	0.4110	0.3286	0.3396
200 000	1.0802	0.9010	0.8172
300 000	1.5594	1.1266	1.1202
500 000	2.7984	2.2548	2.4210
600 000	3.1790	2.6070	2.8704



Şekil 2.62 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen üstel dağılımlı kümede birleştirilmiş QuickCP

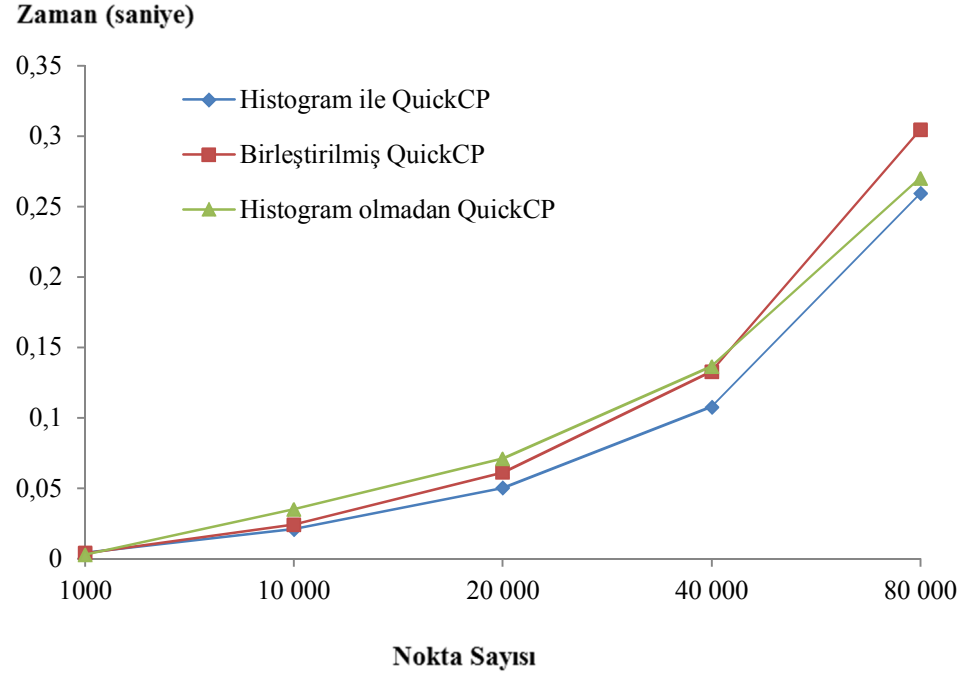
Tablo 2.25 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.62)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
900 000	2.5290	2.3906	2.2156
1 200 000	3.3468	3.5348	2.9480
1 600 000	4.7354	4.5026	3.8150
2 000 000	5.1580	5.7004	4.8798
2 500 000	6.6860	7.0502	6.7996

2.8.3 Normal dağılımlı kümelerde algoritma seçimi

Şekil 2.63. Şekil 2.64 ve Şekil 2.65'te yer alan grafikte QuickCP algoritmasının üç şeklinin çalışma süreleri, normal dağılımlı ve çeşitli büyüklükte nokta içeren kümelerde çalıştırılarak görüntülenmiştir. "Histogram ile QuickCP" algoritmasının çizgisinde, QuickCP algoritmasının çalışma süresi ile nokta kümesi için yapılan histogram fonksiyonunun çalışma süresinin toplamı yer almaktadır. "Birleştirilmiş QuickCP" algoritmasının çizgisinde QuickCP ve rastgeleleştirilmiş QuickCP algoritmasını birleştiren algoritmanın çalışma süresi yer almaktadır (Bkz Şekil 2.58). "Histogram olmadan QuickCP" algoritmasının çizgisinde sadece QuickCP algoritmasının çalışma süresi yer almaktadır.

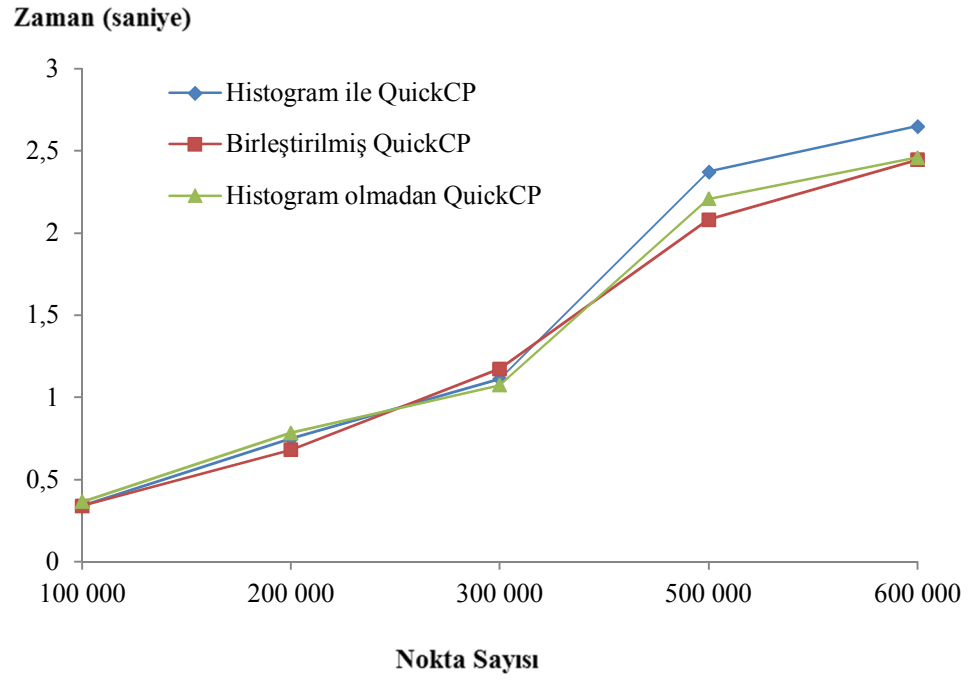
Üç algoritma da zaman zaman birbirlerini geçmiş ancak genelde benzer sürede çalışmıştır.



Şekil 2.63 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.26 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.63)

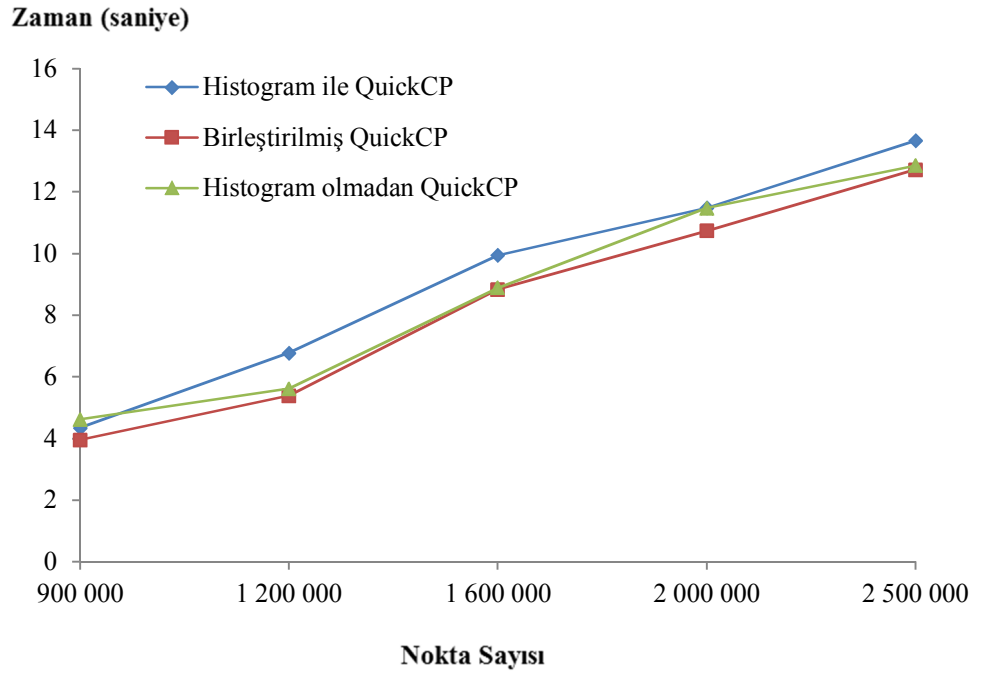
Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
1000	0.0044	0.0040	0.0030
10 000	0.0212	0.0244	0.0352
20 000	0.0504	0.0614	0.0712
40 000	0.1078	0.1328	0.1366
80 000	0.2594	0.3044	0.2700



Şekil 2.64 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.27 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.64)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
100 000	0.3398	0.3416	0.3658
200 000	0.7486	0.6814	0.7844
300 000	1.1138	1.1732	1.0742
500 000	2.3730	2.0808	2.2064
600 000	2.6486	2.4438	2.4590



Şekil 2.65 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen normal dağılımlı kümede birleştirilmiş QuickCP

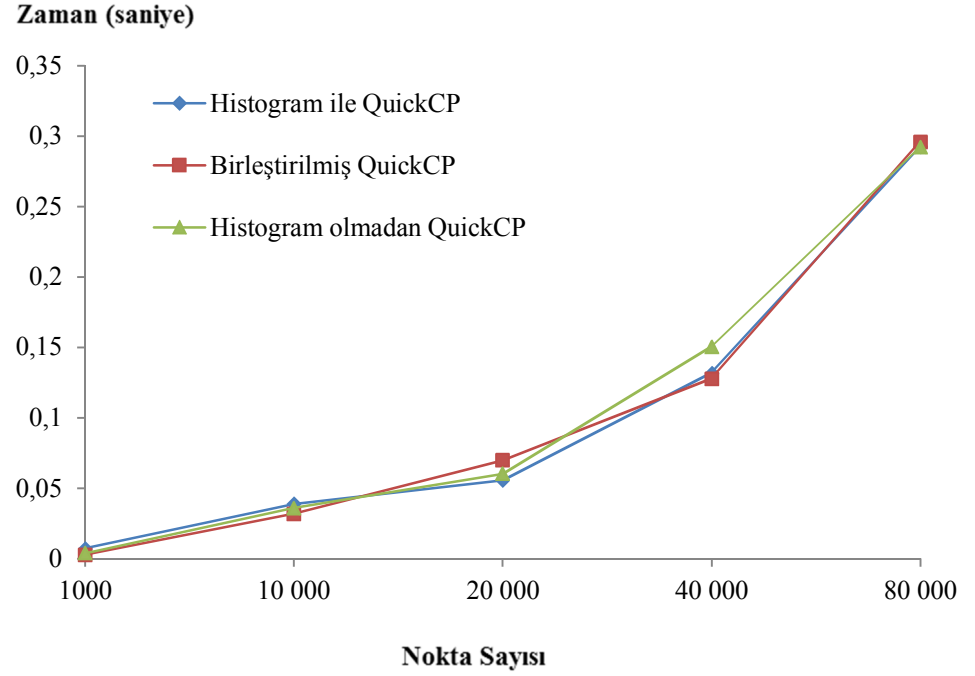
Tablo 2.28 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.65)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
900 000	4.3548	3.9594	4.6238
1 200 000	6.7812	5.3918	5.6184
1 600 000	9.9412	8.8356	8.8828
2 000 000	11.4830	10.7380	11.4690
2 500 000	13.6624	12.7124	12.8532

2.8.4 Tekdüze dağılımlı kümelerde algoritma seçimi

Şekil 2.66, Şekil 2.67 ve Şekil 2.68'de yer alan grafikte QuickCP algoritmasının üç şeklinin çalışma süreleri, tekdüze dağılımlı ve çeşitli büyüklükte nokta içeren kümelerde çalıştırılarak görüntülenmiştir. “Histogram ile QuickCP” algoritmasının çizgisinde, QuickCP algoritmasının çalışma süresi ile nokta kümesi için yapılan histogram fonksiyonunun çalışma süresinin toplamı yer almaktadır. “Birleştirilmiş QuickCP” algoritmasının çizgisinde QuickCP ve rastgeleleştirilmiş QuickCP algoritmasını birleştiren algoritmanın çalışma süresi yer almaktadır (Bkz Şekil 2.58). “Histogram olmadan QuickCP” algoritmasının çizgisinde sadece QuickCP algoritmasının çalışma süresi yer almaktadır.

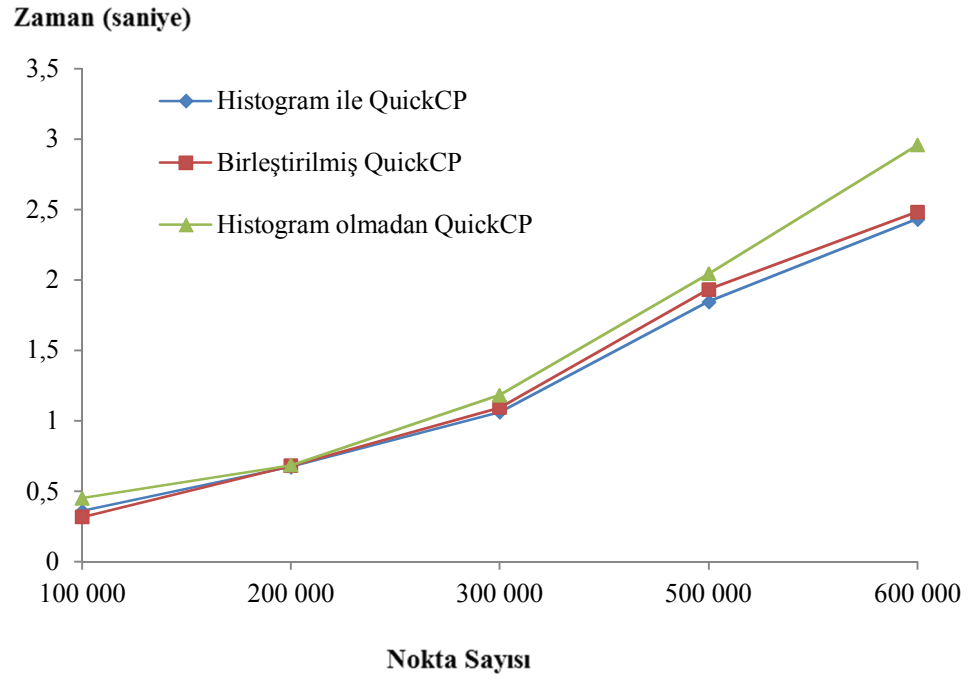
Üç algoritma da zaman zaman birbirlerini geçmiş ancak genelde benzer sürede çalışmıştır.



Şekil 2.66 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.29 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.66)

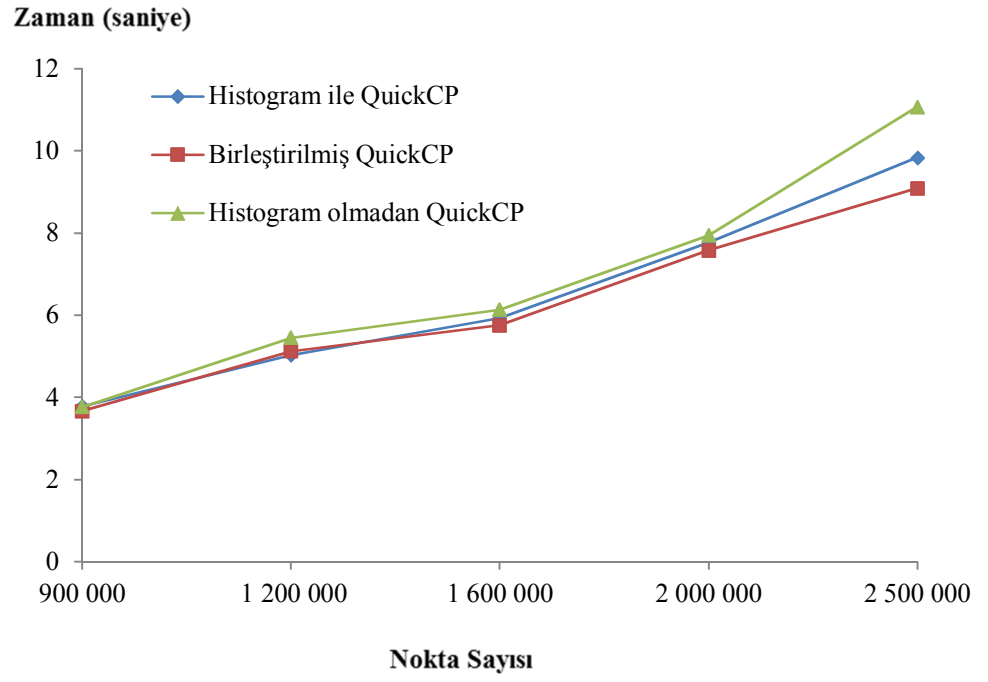
Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
1000	0.0076	0.0030	0.0042
10 000	0.0390	0.0320	0.0362
20 000	0.0558	0.0700	0.0604
40 000	0.1320	0.1278	0.1506
80 000	0.2924	0.2960	0.2922



Şekil 2.67 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.30 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.67)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
100 000	0.3616	0.3184	0.4512
200 000	0.6752	0.6804	0.6862
300 000	1.0634	1.0952	1.1830
500 000	1.8458	1.9324	2.0444
600 000	2.4312	2.4820	2.9568



Şekil 2.68 $\mathbb{R}^2$ 'de çizgisel birikim içermeyen tekdüze dağılımlı kümede birleştirilmiş QuickCP

Tablo 2.31 $\mathbb{R}^2$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 2.68)

Nokta Sayısı	Histogram ile QuickCP	Birleştirilmiş QuickCP	Histogram olmadan QuickCP
900 000	3.7810	3.6676	3.7702
1 200 000	5.0288	5.1272	5.4486
1 600 000	5.9302	5.7566	6.1338
2 000 000	7.7700	7.5742	7.9402
2 500 000	9.8306	9.0882	11.061

3. YÜKSEK BOYUTLU UZAYLARDA QUICKCP ANALİZİ

3.1 QuickCP'yi Yüksek Boyutlu Uzaylara Uyarlama

QuickCP algoritması, noktaların birbirine yakın olması durumunu $\mathbb{R}^2$ 'de noktaların X ve Y koordinatlarını inceleyerek gözlemler. Bitiş koşulunda ise X ve Y koordinatlarından elde ettiği en küçük koordinat farkı değerlerini kullanır. Bu işlemler herhangi bir $2, 3, 4, \dots, k$ ($k \in \mathbb{N}$) boyutuna uyarlanabilir. Bunun için $k = 2$ iken X ve Y koordinatları için yapılan işlemleri boyut $3, 4, \dots, k$ ($k \in \mathbb{N}$) iken tüm koordinatlara uyarlamak ve algoritma çalıştırılmadan önce yapılan ön işlemler ile bitiş koşulunu tüm koordinatlar için düzenlemek gereklidir.

QuickCP algoritması iki boyutlu uzayda ön işlem olarak nokta kümesindeki noktaları önce X sonra da Y koordinatına göre sıralayarak A ve B adında iki diziye kopyalar. Uzay boyutu arttırıldığında nokta kümesindeki noktalar her bir koordinata göre ayrı ayrı sıralanmalı ve dizilere kopyalanmalıdır. Örneğin k boyutlu $\mathbb{R}^k$ uzayında, bir nokta kümesindeki noktalar tüm koordinatlarına göre sıralanarak $A_1, A_2, \dots, A_k$ dizilerine ayrı ayrı kopyalanmalıdır. QuickCP algoritmasının bu bölümünün uyarlanmasında sıralama algoritması olarak Merge Sort algoritması kullanılmıştır.

$\mathbb{R}^2$ 'de nokta çiftleri incelenirken X ve Y koordinat farklarının en küçükleri bulunarak bitiş koşulunda kullanılıyordu. Uzay boyutu herhangi bir $k \in \mathbb{N}$ boyutuna genişletildiğinde, her bir koordinat için en küçük koordinat farkı bulunmalı ve bitiş koşulunda kullanılmalıdır. Örneğin k boyutlu $\mathbb{R}^k$ uzayında tüm koordinatların $\Delta X_{min_1}, \Delta X_{min_2}, \dots, \Delta X_{min_k}$ değerleri bulunmalıdır. Ek 7'de QuickCP algoritmasının $k \in \mathbb{N}$ boyuta uyarlanmış hali bulunmaktadır.

3.2 Yüksek Boyutlarda QuickCP Rastgeleleştirme

$\mathbb{R}^2$ 'de bir nokta kümesinin, QuickCP algoritmasının performansını olumsuz etkileyecek şekilde çizgisel yığılım içerip içermediğini belirlemek için uygulanan histogram işlemi, burada tüm koordinatlar için uygulanmıştır. Nokta kümesindeki noktaların koordinatları, her bir ekseninde incelenmiştir. Noktaların eksenlere izdüşümünde çizgisel yığılım belirlenirse, burada da rastgeleleştirme yapılmalıdır. $\mathbb{R}^2$ uzayındaki noktaları yeni basis vektörler ile elde edilen yeni bir koordinat sisteminde tanımlama işlemi, boyut $3, 4, \dots, k$ ($k \in \mathbb{N}$) iken tüm koordinat sistemi yeni bir koordinat sistemine dönüştürülerek uygulanmıştır. Histogram sonucunda

çizgisel birikim belirlenirse yeni bir koordinat sisteminin kullanılması yeni basis vektörleri ile sağlanır.

Yeni bir koordinat sistemi için k boyutlu uzayda $X_1, X_2, \dots, X_k$ eksenlerine ve yeni bir orijine ihtiyaç vardır. Bu eksenler k ne olursa olsun birbirine dik olmalıdır, öyle ki $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_k$ vektörlerinin koordinat sistemindeki eksenlerin yön vektörleri (ing. direction vector) olduğu varsayılırsa,

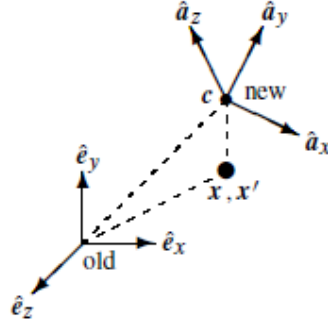
Her $i, j \in \{1, 2, \dots, k\}$ için $\vec{v}_i \cdot \vec{v}_j = 0$ (ing. dot product) ve her $i \in \{1, 2, \dots, k\}$ için $|\vec{v}_i| = 1$ olmalıdır (Lautrup, 2011).

$\mathbb{R}^3$ uzayında a_x, a_y, a_z vektörlerinin basis vektörü olabilmesi için Şekil 3.1’de yer alan iki özelliği taşıması gerekir. Bu özellikleri taşıyan vektörler ortonormaldir.

$$|\hat{a}_x| = |\hat{a}_y| = |\hat{a}_z| = 1 \quad \hat{a}_x \cdot \hat{a}_x = \hat{a}_y \cdot \hat{a}_z = \hat{a}_z \cdot \hat{a}_x = 0$$

Şekil 3.1 $\mathbb{R}^3$ ’te basis vektörlerinin sağlaması gereken kurallar (Lautrup’tan, 2011)

$\mathbb{R}^3$ ’teki bir P noktasının $x, y, z, x', y', z' \in \mathbb{R}$ olmak üzere eski koordinat sistemindeki pozisyonu $\mathbf{X} = (x, y, z)$ ve yeni koordinat sistemindeki pozisyonu $\mathbf{X}' = (x', y', z')$ olsun. Bu iki pozisyon arasında Şekil 3.2’te yer alan ilişki bulunur. Yani, yeni koordinat sisteminin basis vektörlerinin oluşturduğu matris ile (x', y', z') noktası çarpılır. Bulunan vektör yeni orijin vektörü ile toplanarak $\mathbf{X}$ pozisyonunun X koordinatı elde edilir.



$$x = c + x' \hat{a}_x + y' \hat{a}_y + z' \hat{a}_z.$$

Şekil 3.2 X ve X' pozisyonu ilişkisi (Lautrup'dan, 2011)

P noktasının X' pozisyonu ise Şekil 3.3'te görülen şekilde bulunur. Eski koordinat sistemindeki X pozisyonundan orijin C çıkarılarak bir vektör elde edilir. Bu vektör, yeni basis vektörleri ile çarpılarak (ing. dot product) X' pozisyonu yani yeni koordinat sistemindeki koordinatlar bulunur.

$$\begin{aligned} x' &= \hat{a}_x \cdot (x - c) = a_{xx}(x - c_x) + a_{xy}(y - c_y) + a_{xz}(z - c_z), \\ y' &= \hat{a}_y \cdot (x - c) = a_{yx}(x - c_x) + a_{yy}(y - c_y) + a_{yz}(z - c_z), \\ z' &= \hat{a}_z \cdot (x - c) = a_{zx}(x - c_x) + a_{zy}(y - c_y) + a_{zz}(z - c_z), \end{aligned}$$

$$x'_i = \sum_j a_{ij}(x_j - c_j).$$

Şekil 3.3 X' pozisyonunun bulunuşu (Lautrup'dan, 2011)

Görüldüğü gibi bir koordinat sistemini diğerine dönüştürmek için yeni bir ortonormal basis'e ihtiyaç vardır. Yeni ortonormal basis vektörleri yaratmak için Gram-Schmidt yöntemi kullanılmıştır (Harvey Mudd College, 2013; Kuttler, 2012).

Yöntemi uygulamak için gereken $\vec{v}_1$ vektörü normu 1 olacak şekilde rastgele olarak belirlenir. A ve B , $\mathbb{R}^k$ uzayında iki nokta olmak üzere, $A = (0, 0, \dots, 0)$ ile $B = (x_1, x_2, \dots, x_k)$ iken $\vec{v}_1 = \overrightarrow{AB}$ olsun. Öncelikle $\vec{v}_1$ vektörünün

x_1 koordinatı 0 ile 1.0 arasında rastgele olarak belirlenir. Daha sonra 0 ile $\sqrt{1 - x_1^2}$ arasında rastgele bir onlalık sayı x_2 koordinatı olarak belirlenir. Bu işlem en son x_k koordinatı $\sqrt{1 - x_1^2 - x_2^2 - \dots - x_{k-1}^2}$ ile belirlenene kadar devam eder. Bu vektör kullanılarak uzayın boyutuna göre k boyutlu uzay için $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_k$ vektörleri Gram-Schmidt yöntemi ile elde edilir.

$$\vec{u}_1 = \vec{v}_1 \text{ ve } proj_u(v) = (\langle u, v \rangle / \langle u, u \rangle)u \text{ olmak üzere,}$$

$$\vec{u}_2 = \vec{v}_2 - proj_{u_1}(v_2)$$

$$\vec{u}_3 = \vec{v}_3 - proj_{u_1}(v_3) - proj_{u_2}(v_3)$$

...

$\vec{u}_k = \vec{v}_k - \sum_{j=1}^{k-1} proj_{u_j}(v_k)$ adımlarıyla vektörler bulunur. Bulunan vektörler normalize edilerek, yeni koordinat sisteminin ortonormal basis vektörleri elde edilir.

$$\vec{e}_1 = \vec{u}_1 = \vec{v}_1$$

Burada, $\|\vec{v}_1\| = 1$ olduğundan $\vec{v}_1$ vektörü normalize durumdadır.

$$\vec{e}_2 = \vec{u}_2 / \|\vec{u}_2\|$$

Burada, $\vec{u}_2$ vektörü normalize edilerek ikinci ortonormal vektör elde edilir. Bu normalizasyon işlemi elde edilen tüm $\vec{u}_i$ ($i \in \{2, 3, \dots, k\}$) vektörleri için yapılır. En son $\vec{e}_k = \vec{u}_k / \|\vec{u}_k\|$ vektörü elde edilir. Sonuç olarak elde edilen $\{\vec{e}_1, \vec{e}_2, \dots, \vec{e}_k\}$ vektör kümesi ortonormal basis kümesidir. Bu küme yeni koordinat sisteminde noktaların yeni koordinatlarını elde etmek için kullanılır.

Örnek olarak $\mathbb{R}^2$ ve $\mathbb{R}^3$ uzayları için yeni ortonormal basise sahip yeni koordinat sistemleri oluşturulmuştur. $\mathbb{R}^2$ uzayında yer alan 10 tane nokta içeren bir küme için, kümedeki noktaların eski ve yeni koordinatları (ing. projected coordinates), kullanılan basis (ing. the orthonormal basis matrix), basisin tersi (ing. the inverse of the matrix) ve basisin determinanı (ing. the determinant) ekrana yazdırılmıştır (Şekil 3.4).

```

DIMENSION k=2

-----
Number of points: 10

THE DETERMINANT IS=1.000000
THE ORTHONORMAL BASIS MATRIX:
    0.916118    0.400908
   -0.400908    0.916118

THE INVERSE OF THE MATRIX:
    0.916118   -0.400908
    0.400908    0.916118

ARRAY:

< 0.020276 0.000383 > projected coordinates: < 0.018728 -0.007778 >
< 0.000402 0.000403 > projected coordinates: < 0.000530 0.000208 >
< 0.000403 0.000401 > projected coordinates: < 0.000530 0.000206 >
< 0.000400 0.000394 > projected coordinates: < 0.000525 0.000201 >
< 0.000397 0.000401 > projected coordinates: < 0.000525 0.000208 >
< 0.002062 0.000400 > projected coordinates: < 0.002049 -0.000461 >
< 0.000399 0.000399 > projected coordinates: < 0.000525 0.000206 >
< 0.000401 0.000402 > projected coordinates: < 0.000528 0.000207 >
< 0.000399 0.000403 > projected coordinates: < 0.000527 0.000209 >
< 0.000402 0.000402 > projected coordinates: < 0.000529 0.000207 >

```

Şekil 3.4 $\mathbb{R}^2$ uzayında 10 elemanlı kümede noktaların eski ve yeni koordinatları

$\mathbb{R}^2$ uzayında yer alan noktaların, yeni basis kullanılarak yeni koordinat sistemindeki koordinatlarının bulunması işlemi Şekil 3.5'te yer almaktadır. Burada 10 adet noktaya sahip bir dizi ekrana yazdırılmıştır. Standart basise sahip XY koordinat sistemindeki bir $(x, y) = (0.020276, 0.000383)$ noktası ortonormal basis ile çarpılarak yeni koordinat sisteminde $(x', y') = (0.018728, -0.007778)$ olmuştur (Lautrup, 2011):

Basis		Point		Converted Point
$\begin{bmatrix} 0.916118 & 0.400908 \\ -0.400908 & 0.916118 \end{bmatrix}$	×	$\begin{bmatrix} 0.020276 \\ 0.000383 \end{bmatrix}$	=	$\begin{bmatrix} 0.018728 \\ -0.007778 \end{bmatrix}$

Şekil 3.5 Çevrilmiş koordinat hesaplama örneği

Benzer bir hesaplama örnek olarak $\mathbb{R}^3$ uzayı için de yapılmıştır. Şekil 3.6'da $\mathbb{R}^3$ 'te yer alan 10 adet nokta yeni basis ile yeni koordinat sistemine dönüştürülmüştür.

```

DIMENSION k=3
-----
Number of points: 10
THE DETERMINANT IS=1.000000
THE ORTHONORMAL BASIS MATRIX:
      0.000000      0.066686      0.997774
      0.995525     -0.094285      0.006302
      0.094496      0.993309     -0.066388

THE INVERSE OF THE MATRIX:
      -0.000000      0.995525      0.094496
      0.066686     -0.094285      0.993309
      0.997774      0.006302     -0.066388

ARRAY:
< 0.020276 0.000383 0.000402 > projected coordinates: < 0.000427 0.020151 0.002269 >
< 0.000403 0.000403 0.000401 > projected coordinates: < 0.000427 0.000365 0.000412 >
< 0.000400 0.000394 0.000397 > projected coordinates: < 0.000423 0.000364 0.000403 >
< 0.000401 0.002062 0.000400 > projected coordinates: < 0.000536 0.000207 0.002060 >
< 0.000399 0.000399 0.000401 > projected coordinates: < 0.000427 0.000362 0.000408 >
< 0.000402 0.000399 0.000403 > projected coordinates: < 0.000428 0.000365 0.000408 >
< 0.000402 0.000402 0.000405 > projected coordinates: < 0.000431 0.000365 0.000411 >
< 0.000403 0.000405 0.000403 > projected coordinates: < 0.000429 0.000365 0.000414 >
< 0.000403 0.000404 0.002010 > projected coordinates: < 0.002032 0.000375 0.000306 >
< 0.000403 0.000406 0.000403 > projected coordinates: < 0.000430 0.000366 0.000414 >

```

Şekil 3.6 $\mathbb{R}^3$ uzayında 10 elemanlı kümede noktaların eski ve yeni koordinatları

Rastgeleleştirme ile elde edilen yeni koordinatlar QuickCP algoritması değiştirilmeden kullanılır. Algoritmada noktalar arası mesafe hesaplamaları ve bitiş koşulunun uyarlanması bu yeni elde edilen koordinatlar temel alınarak yapılır.

3.3 QuickCP Algoritması ile Karşılaştırılacak Algoritmaların Uyarlaması

3.3.1 KKA'nın yüksek boyutlu uzaylara uyarlanması

KKA'nın tanımını gereği k boyutlu uzayda yer alan n elemanlı bir P kümesinin en yakın iki noktasını bulmak için her bir noktanın diğer noktalara olan uzaklığı ayrı ayrı hesaplanmalıdır. Bu durumda P kümesindeki n adet nokta $C(n, 2)$ tane nokta çifti oluşturur. Her bir çiftteki iki noktanın birbirine uzaklığını hesaplandıktan sonra bu uzaklıklar karşılaştırılır. Aralarından en küçüğü bulunur (Şekil 3.7). Bu da $O(n^2)$ tane karşılaştırma ve noktalar arası uzaklık hesaplama işlemi demektir.

```

//P is the array which contains n points in  $\mathbb{R}^k$ 

dmin = infinity

for i from 1 to n

    for j from i + 1 to n

        let p = P[i], q = P[j]

        if dist(p, q) < dmin

            minDist = dist(p, q)

            closestPair = (p, q)

        endif

    endfor

endfor

return closestPair

```

Şekil 3.7 k boyutta KKA

3.3.2 BFA'nın yüksek boyutlu uzaylara uyarlanması

BFA bir k boyutlu uzaya uyarlanabilir bir algoritmadır. Bentley and Shamos (1976)'a göre BFA, yüksek boyutlu bir uzaya uyarlanırken noktalar ön işlem olarak X_1 koordinatına göre $O(\log_2 n)$ karmaşıklıkta kendi içinde sıralanır. Burada sıralama işlemi için sıralama algoritması olarak Merge Sort kullanılmıştır.

$\mathbb{R}^k$ uzayı $X_1, X_2, \dots, X_k$ ekseninden oluşmaktadır. Noktaları bölmek için X_1 eksenini kesen ve noktaları A ve B olarak iki koleksiyona ayıran $k - 1$ boyutlu bir hiperdüzlem (ing. hyperplane) hazırlanır. $\mathbb{R}^2$ 'de yani $k = 2$ iken, bu hiperdüzlem bir doğrudur. Bu hiperdüzlemin X_1 eksenini bölerek oluşturduğu A ve B koleksiyonlarında $n/2$ adet nokta bulunmalıdır. Bu hiperdüzlemin X_1 eksenini kestiği nokta, noktaların o koordinattaki medyanıdır.

Bu bölme işlemleri özyinelemeli olarak çalıştığından $n \leq 3$ adet nokta içeren koleksiyonlar oluşana kadar devam eder. Koşul sağlandığında koleksiyonlardaki en yakın nokta çiftleri ve noktalar arasındaki uzaklıklar KKA kullanılarak hesaplanır. Ancak bu elde edilen değerler en yakın çifti bulmak için

yeterli değildir. “ing. The fixed radius-near-neighbor problem” olarak bilinen ve belli sabit bir δ yarıçap için, bir sorgulama noktası q 'ya δ uzaklıkta olan noktaları araştıran problemin özel bir durumu olan ing. sparsity olarak bilinen duruma göre verilen bir nokta koleksiyonunda δ yarıçaplı hiçbir küre (ing. ball) bir c sabitinden daha fazla nokta içermez. Bu anlamda A koleksiyondaki en yakın nokta çifti arası uzaklık δ_A ve B koleksiyonundaki en yakın nokta çifti arası uzaklık δ_B kullanılarak $A \cup B$ kümesinin en yakın nokta çifti arasındaki uzaklık δ_{AB} bulunabilir. Eğer $A \cup B$ 'de $\delta = \min(\delta_A, \delta_B)$ olarak seçilen δ 'dan daha yakın bir nokta çifti varsa bu çift, Şekil 3.8'de görüldüğü şekilde 2δ genişliğindeki dikey levhada (ing. vertical slab) yer alır. $\mathbb{R}^k$ uzayında $k = 2$ için bu levhada en fazla 12 adet nokta olabilir. Herhangi bir k boyutu için de bu değer $4 * (3^{k-1})$ 'dir ve yine sabittir.

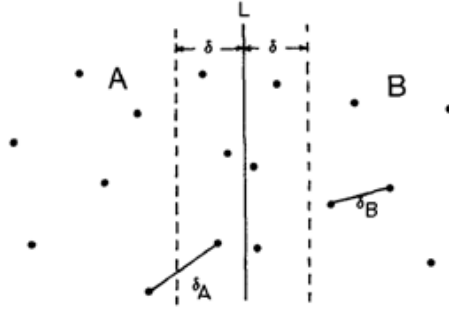


Figure 1.

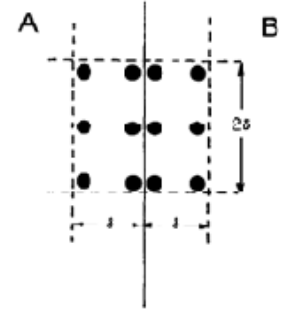


Figure 2.

Şekil 3.8 Seyreklik koşulu (ing. sparsity condition) (Bentley and Shamos'dan, 1976)

A ve B koleksiyonları, tüm noktaları içeren kümenin ikiye bölünmesiyle oluşur. Bunun için tüm noktaları içeren kümedeki noktaların X_1 koordinatlarının medyanı bulunmuş ve kullanılmıştır. Özyinelemeli olarak bölme işleminin devam etmesi için yeni koleksiyonların yeni medyanları bulunur. Elde edilen alt koleksiyonlardaki (ing. subcollections) en yakın çiftler belirlendikten sonra, her alt koleksiyonun yanındaki koleksiyonla oluşturduğu 2δ genişliğindeki levhadaki noktalar incelenir. Daha yakın bir çift bulunursa saklanır. Rosetta Code (2013)'dan k boyutta uyarlanmış BFA, Ek 8'de yer almaktadır.

3.4 Çeşitli Boyutlarda Çalışma Süresi Sonuçları

Çeşitli boyutlardaki uzaylarda KKA, BFA, QuickCP ve rastgeleleştirilmiş QuickCP uyarlanarak çalıştırılmıştır. Sırasıyla uzay boyutu $k = 3$, $k = 4$, $k = 5$,

$k = 8$, $k = 12$ ve $k = 17$ iken algoritmaların çalışma davranışları gözlemlenmiştir.

Her bir boyut için önce noktaların herhangi bir ekseninde çizgisel yığılım yapmadığı kümelerde QuickCP, KKA ve BFA ile karşılaştırılmıştır. Karşılaştırma sonucunda QuickCP eğer BFA'nın gerisinde kalmışsa rastgeleleştirilmiştir, sonra da KKA ve BFA ile karşılaştırılmıştır. Ardından çizgisel birikim maksimum olacak şekilde hazırlanmış kümelerde algoritmaların davranışları gözlemlenmiştir. KKA'nın çalışma süresi çok çabuk arttığı için yüksek boyutlu uzaylarda bu algoritmanın çalışma sürelerinin bazıları hesaplanmamış, tablolarda ilgili yerlere “*” doldurulmuştur.

3.4.1 $\mathbb{R}^3$ 'te algoritmaların çalışma süresi sonuçları

$\mathbb{R}^3$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP algoritması çalıştırılmıştır. Belirtilen yaklaşık ortalama değerler yaklaşık değerlerdir:

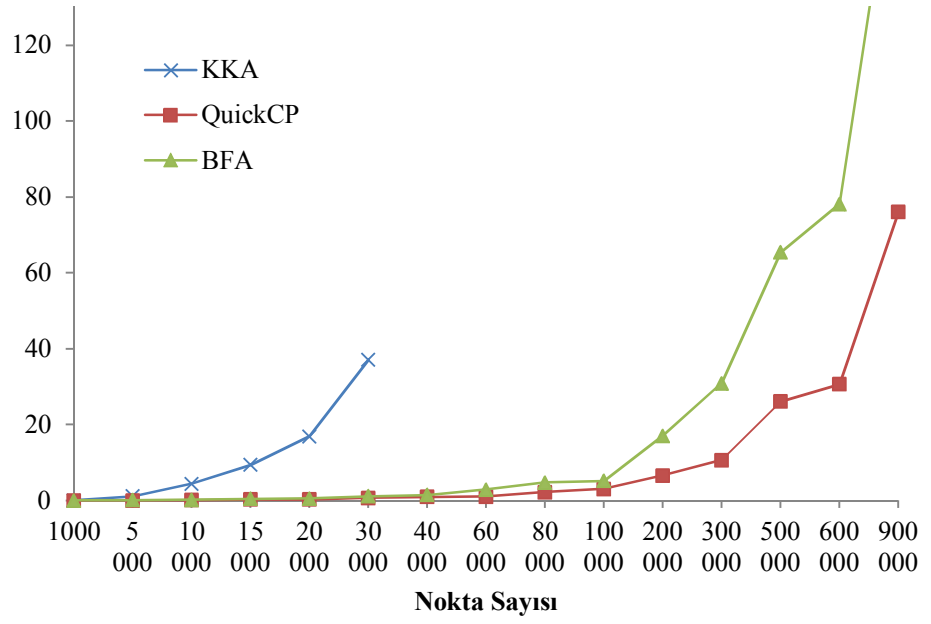
Çizgisel yığılım olmayan kümelerde en iyi performans QuickCP'ye aittir. BFA, üstel dağılımlı kümelerde QuickCP'den 2 kat, normal dağılımda 4.5 kat, tekdüze dağılımda 7 kat daha fazla süre harcamıştır; KKA ise 30 000 elemanlı kümelerde, üstel dağılımda 40 kat, normal dağılımda 100 kat, tekdüze dağılımda da 100 kat daha fazla süre harcamıştır. 30 000 elemanlı kümelerde bile çok fazla süre harcadığı için daha kalabalık olanlarda çalıştırılmamıştır.

Çizgisel yığılım olan kümelerde üstel dağılımlı kümelerde BFA, KKA'nın 1.2 katı, QuickCP 3.5 katı, rastgeleleştirilmiş QuickCP ise 1/500 katıdır. Üstel, normal ve tekdüze dağılımlı nokta kümelerinde çizgisel yığılım maksimum durumda iken yapılan rastgeleştirme işlemi QuickCP'nin performansına olumlu etki etmiştir. Rastgeleleştirilmiş QuickCP çizgisel yığılım içeren kümelerde en iyi algoritma olmuştur. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.1.1 $\mathbb{R}^3$ 'te üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Şekil 3.9'da KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

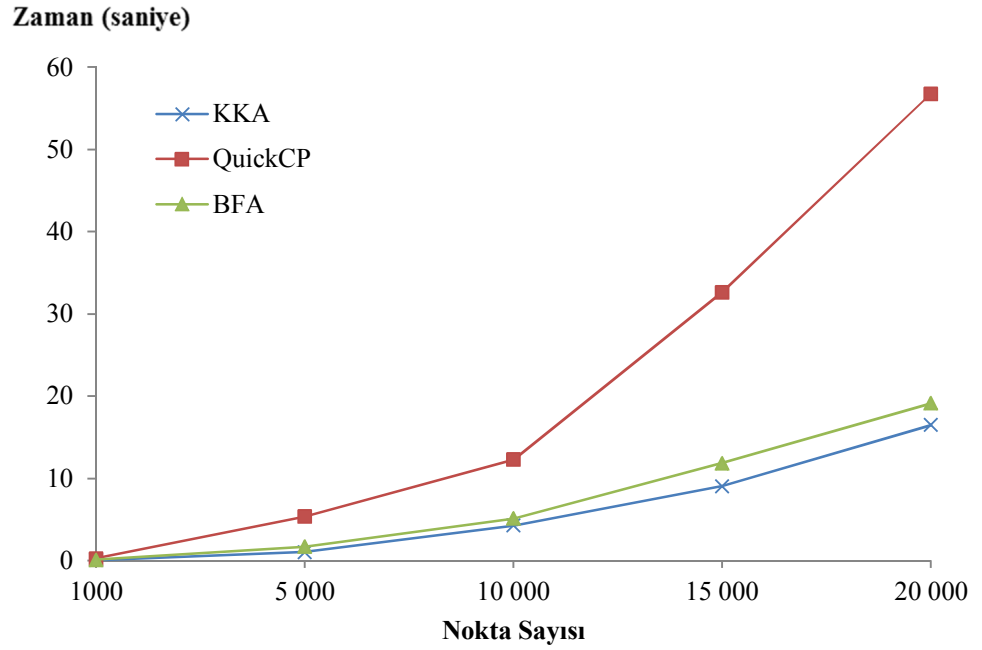
Zaman (saniye)



Şekil 3.9 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Tablo 3.1 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.9)

Nokta sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.048	0.010	0.015
5 000	1.117	0.049	0.101
10 000	4.430	0.114	0.223
15 000	9.429	0.265	0.449
20 000	16.929	0.324	0.551
30 000	37.063	0.739	1.136
40 000	*	0.931	1.485
60 000	*	1.061	2.895
80 000	*	2.213	4.750
100 000	*	3.039	5.215
200 000	*	6.570	17.108
300 000	*	10.693	30.892
500 000	*	26.124	65.403
600 000	*	30.680	78.102
900 000	*	75.995	180.131

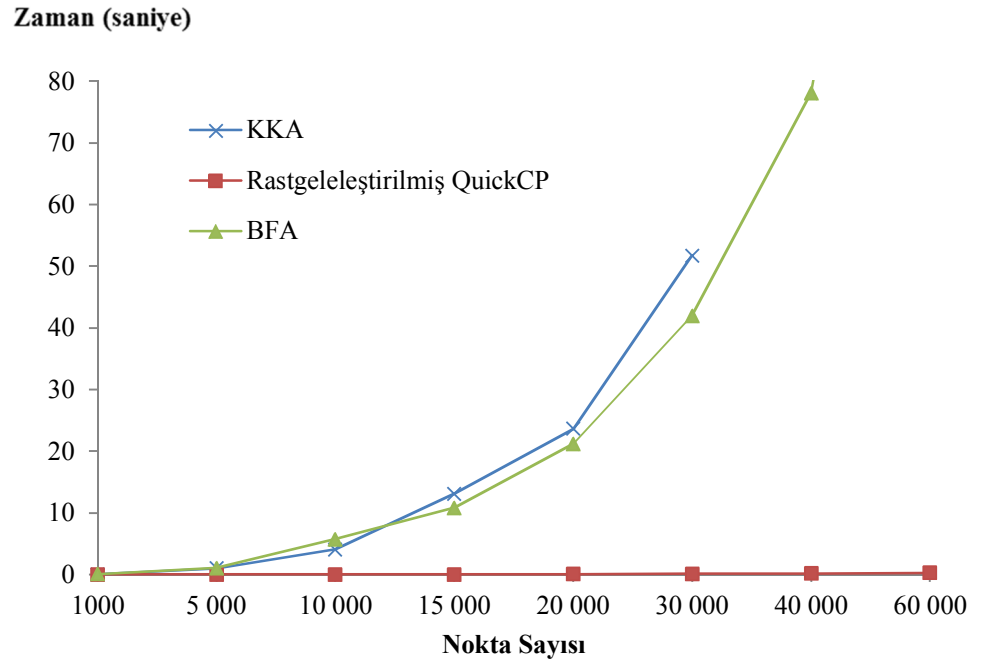


Şekil 3.10 $\mathbb{R}^3$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.10'da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.2 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.10)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.046	0.236	0.079
5 000	1.019	5.348	1.688
10 000	4.255	12.259	5.097
15 000	9.064	32.629	11.868
20 000	16.486	56.769	19.135



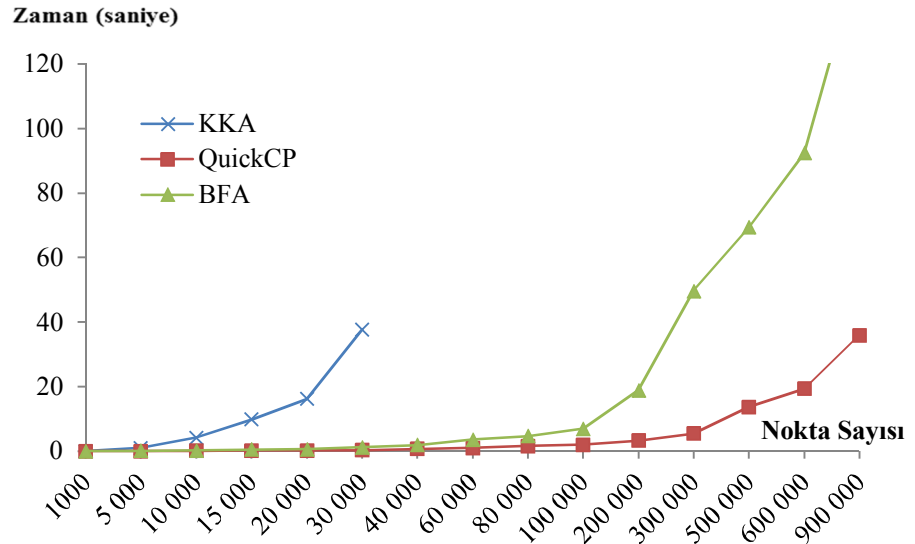
Şekil 3.11 $\mathbb{R}^3$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.11'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.3 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.11)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.044	0.006	0.086
5 000	0.986	0.015	1.044
10 000	4.070	0.030	5.752
15 000	13.089	0.045	10.841
20 000	23.650	0.059	21.206
30 000	51.711	0.099	41.978
40 000	*	0.169	78.083
60 000	*	0.272	183.282

3.4.1.2 $\mathbb{R}^3$ 'te normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

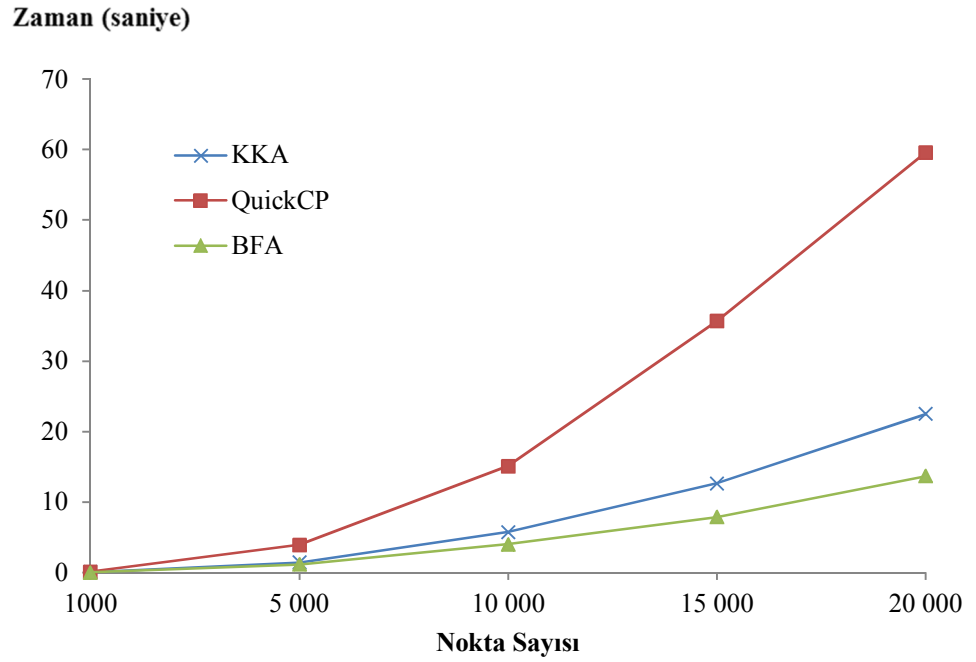


Şekil 3.12 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.12'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.4 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.12)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.041	0.007	0.013
5 000	1.043	0.037	0.086
10 000	4.131	0.091	0.227
15 000	9.842	0.157	0.546
20 000	16.196	0.235	0.648
30 000	37.631	0.349	1.241
40 000	*	0.703	1.923
60 000	*	1.041	3.569
80 000	*	1.611	4.713
100 000	*	1.990	6.960
200 000	*	3.286	18.813
300 000	*	5.463	49.630
500 000	*	13.659	69.321
600 000	*	19.335	92.390
900 000	*	35.951	153.073

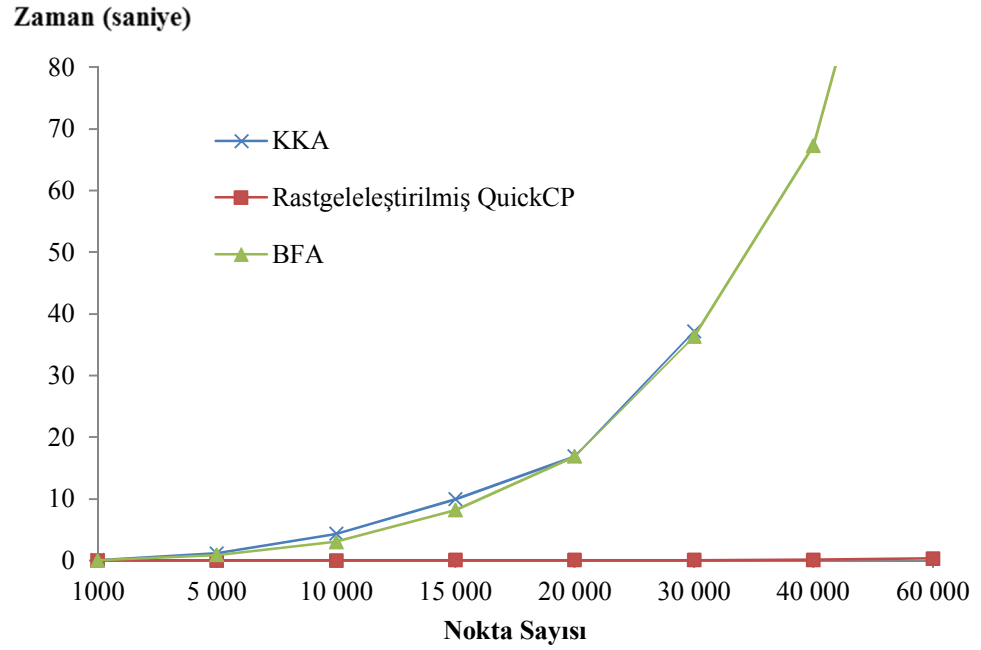


Şekil 3.13 $\mathbb{R}^3$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.13'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.5 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.13)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.057	0.110	0.035
5 000	1.408	3.937	1.151
10 000	5.727	15.118	4.009
15 000	12.653	35.653	7.895
20 000	22.486	59.584	13.690



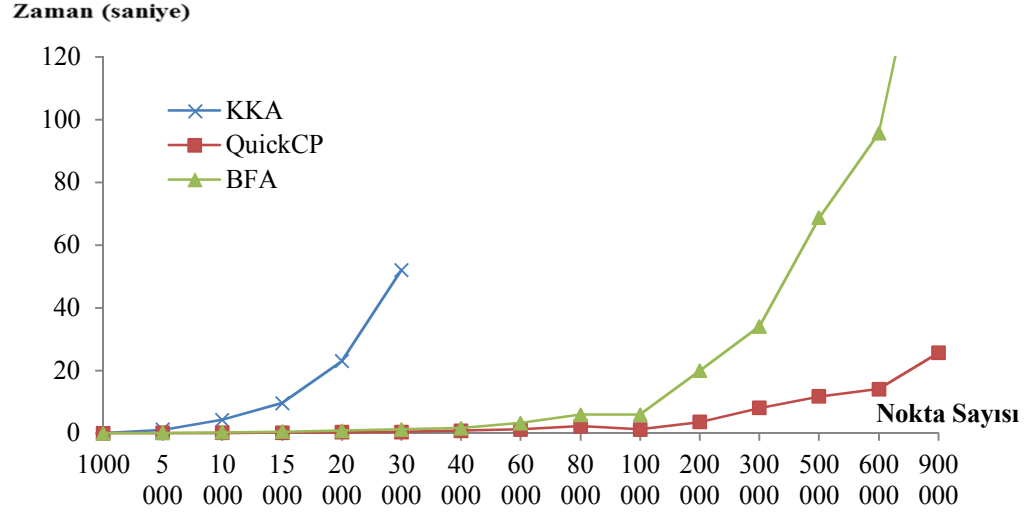
Şekil 3.14 $\mathbb{R}^3$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.14'te çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.6 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.14)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.048	0.003	0.036
5 000	1.166	0.018	0.894
10 000	4.344	0.024	3.030
15 000	9.912	0.059	8.171
20 000	16.899	0.070	16.876
30 000	37.143	0.096	36.276
40 000	*	0.129	67.277
60 000	*	0.340	137.334

3.4.1.3 $\mathbb{R}^3$ 'te tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

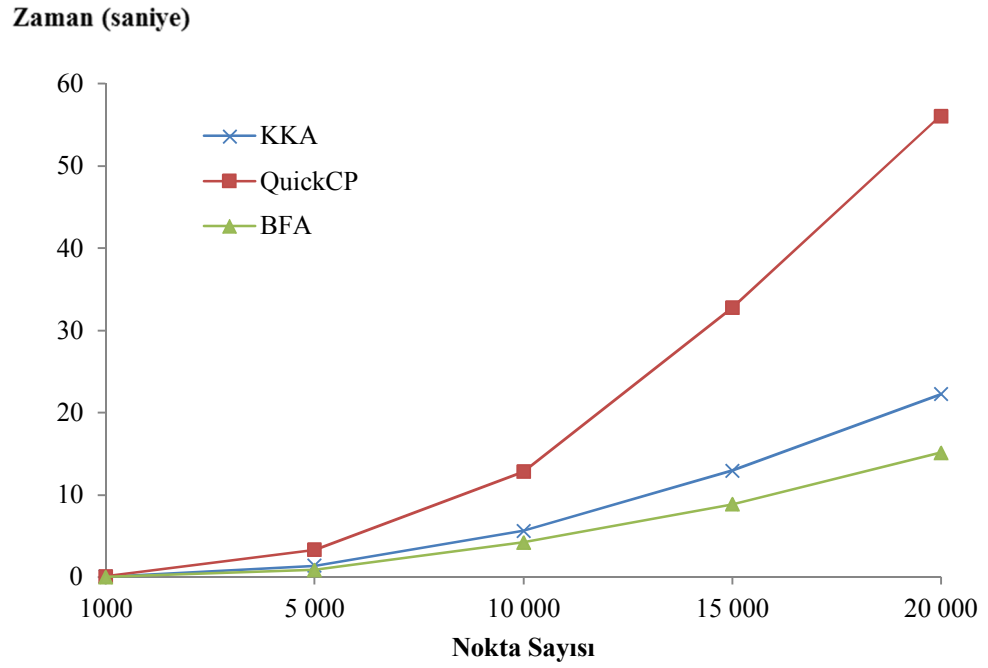


Şekil 3.15 $\mathbb{R}^3$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.15'te KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.7 $\mathbb{R}^3$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.15)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.050	0.006	0.013
5 000	1.053	0.058	0.098
10 000	4.239	0.070	0.227
15 000	9.508	0.131	0.443
20 000	22.944	0.231	0.748
30 000	51.959	0.342	1.211
40 000	*	0.805	1.616
60 000	*	1.156	3.240
80 000	*	2.167	5.844
100 000	*	1.158	5.931
200 000	*	3.525	19.834
300 000	*	7.989	34.105
500 000	*	11.680	68.666
600 000	*	14.090	95.627
900 000	*	25.575	183.571

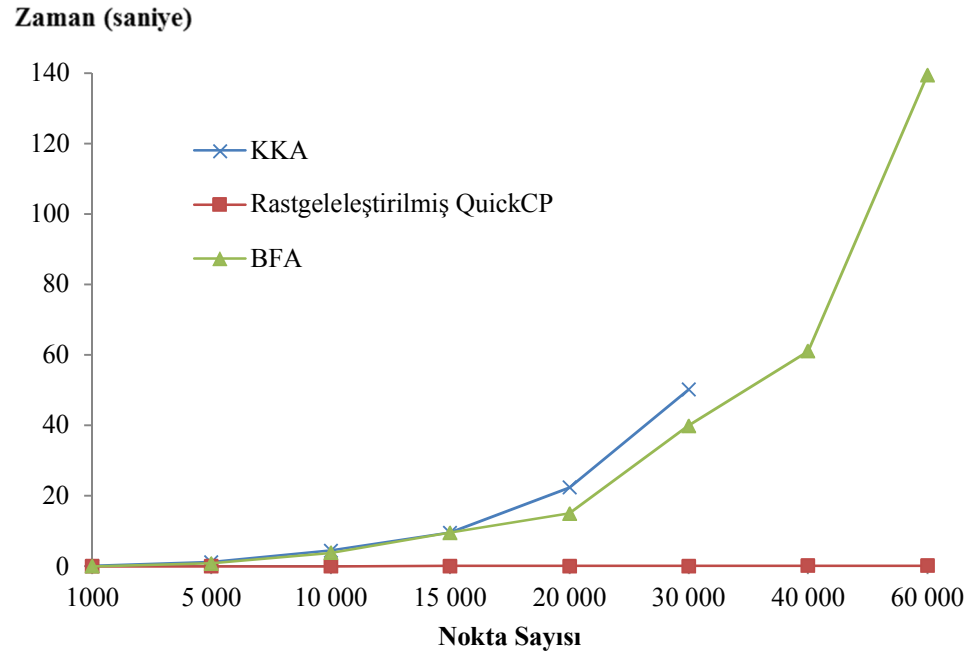


Şekil 3.16 $\mathbb{R}^{3^4}$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.16'da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.8 $\mathbb{R}^{3^4}$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.16)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.056	0.120	0.038
5 000	1.394	3.328	0.876
10 000	5.645	12.825	4.226
15 000	12.945	32.747	8.868
20 000	22.270	56.097	15.137



Şekil 3.17 $\mathbb{R}^{3'}$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.17'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.9 $\mathbb{R}^{3'}$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.17)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.046	0.003	0.036
5 000	1.142	0.025	0.814
10 000	4.405	0.028	3.839
15 000	9.492	0.044	9.514
20 000	22.295	0.109	15.003
30 000	50.172	0.093	39.859
40 000	*	0.133	60.994
60 000	*	0.193	139.288

3.4.2 $\mathbb{R}^4$ 'te algoritmaların çalışma süresi sonuçları

$\mathbb{R}^4$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA, QuickCP ve rastgeleştirilmiş QuickCP çalıştırılmıştır. Çalışma sonuçları resmedilmiştir. Farklı dağılıma sahip nokta kümelerinde algoritmaların çalışma davranışları da farklı olmuştur. Algoritmanın çalışma sonuçlarına dair yorumlar yaklaşık ortalama değerlerdir:

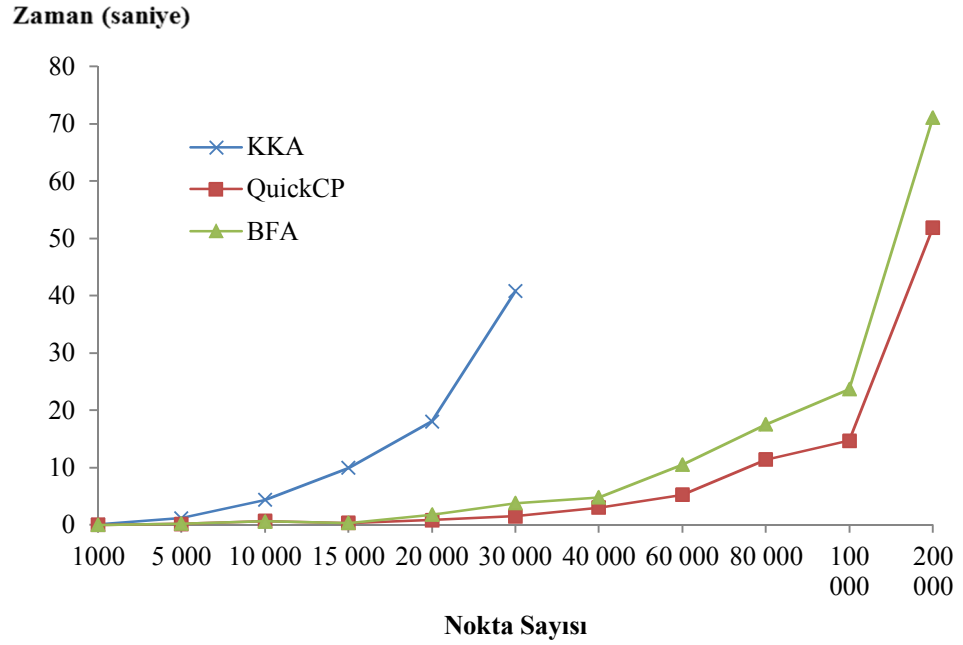
Çizgisel yığılım içermeyen üstel dağılımlı kümelerde KKA, QuickCP'nin 25 katı, BFA ise 1.5 katı kadar süre harcamıştır. Normal dağılımlı kümelerde KKA QuickCP'nin 30 katı, BFA ise 3 katı kadar süre harcamıştır. Tekdüze dağılımlı kümelerde KKA QuickCP'nin 30 katı, BFA 2.5 katı süre harcamıştır.

Çizgisel yığılım içeren üstel dağılımlı kümelerde BFA KKA'nın 1.5 katı, QuickCP ise 5 katı süre harcamıştır. Rastgeleştirilmiş QuickCP ise KKA'nın 1/250 katı kadar süre harcamıştır. Normal dağılımlı kümelerde BFA KKA'nın 0.7 katı, QuickCP ise 4 katı süre harcamıştır. Rastgeleştirilmiş QuickCP ise KKA'nın 1/100 katı süre harcamıştır. Tekdüze dağılımlı kümelerde BFA KKA'nın 1.6 katı, QuickCP ise 4 katı süre harcamıştır. Rastgeleştirilmiş QuickCP ise KKA'nın 1/110 katı süre harcamıştır.

Çizgisel yığılım içermeyen tüm nokta kümelerinde dağılım fark etmeksizin (üstel, normal, tekdüze) QuickCP en iyi çalışan algoritma olmuştur. Ancak normal ve tekdüze dağılımlı kümelerde QuickCP üstel dağılımlı olanlara göre daha iyi performans göstermiştir.

Üstel, normal ve tekdüze dağılımlı nokta kümelerinde çizgisel yığılım maksimum durumda iken yapılan rastgeleştirme işlemi QuickCP'nin performansına olumlu etki etmiştir. Rastgeleştirilmiş QuickCP çizgisel yığılım içeren kümelerde en iyi algoritma olmuştur. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.2.1 $\mathbb{R}^4$ 'te üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

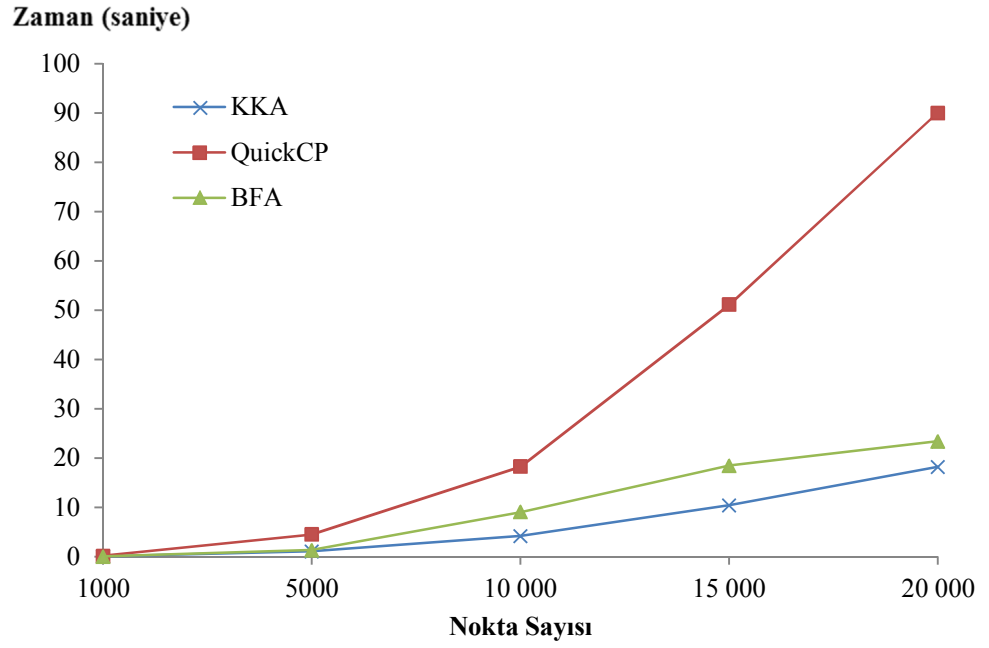


Şekil 3.18 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.18'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.10 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.18)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.050	0.021	0.032
5 000	1.184	0.173	0.203
10 000	4.381	0.689	0.611
15 000	9.928	0.355	0.355
20 000	18.074	0.832	1.777
30 000	40.866	1.526	3.802
40 000	*	3.005	4.787
60 000	*	5.255	10.518
80 000	*	11.410	17.571
100 000	*	14.725	23.721
200 000	*	51.885	71.156

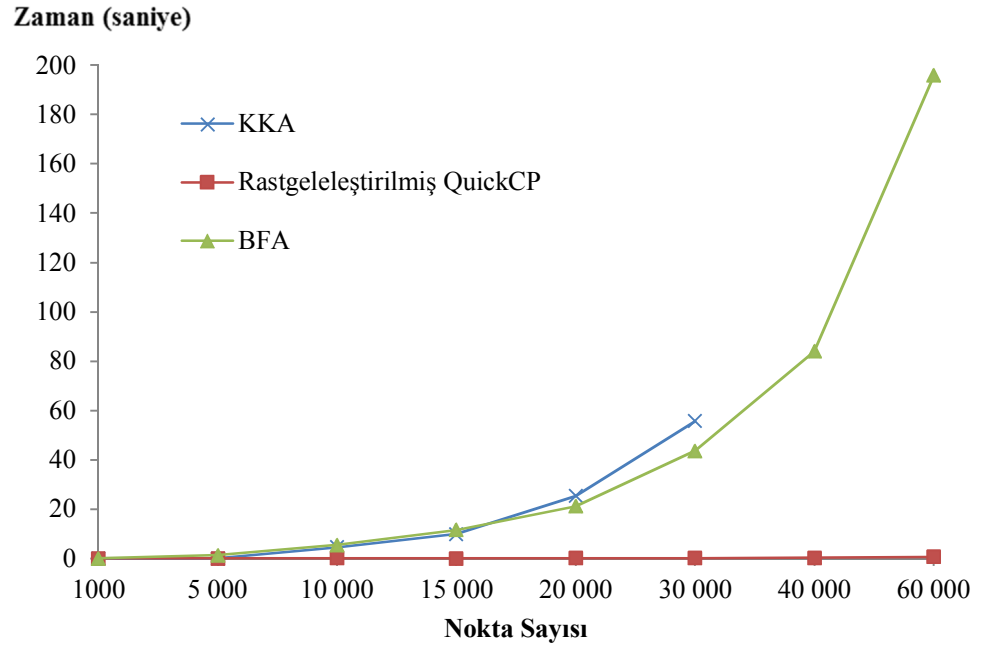


Şekil 3.19 $\mathbb{R}^4$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.19'da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.11 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.19)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.047	0.183	0.048
5000	1.126	4.558	1.379
10 000	4.208	18.299	9.066
15 000	10.470	51.152	18.493
20 000	18.212	90.005	23.450



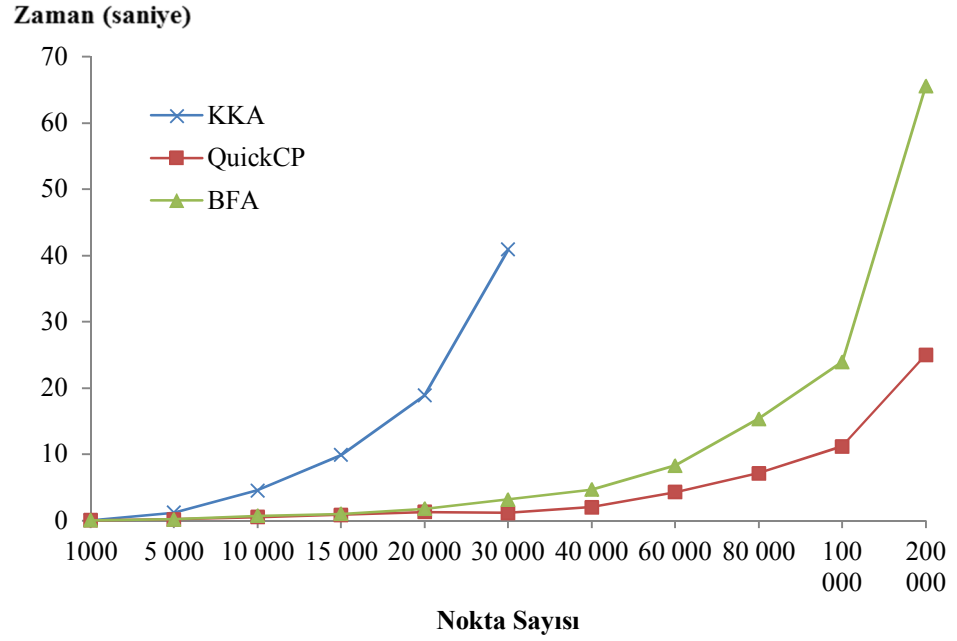
Şekil 3.20 $\mathbb{R}^4$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.20'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.12 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.20)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.051	0.010	0.089
5 000	0.048	0.023	1.369
10 000	4.571	0.120	5.479
15 000	9.936	0.082	11.569
20 000	25.361	0.132	21.185
30 000	55.732	0.184	43.589
40 000	*	0.331	84.017
60 000	*	0.656	195.871

3.4.2.2 $\mathbb{R}^4$ 'te normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

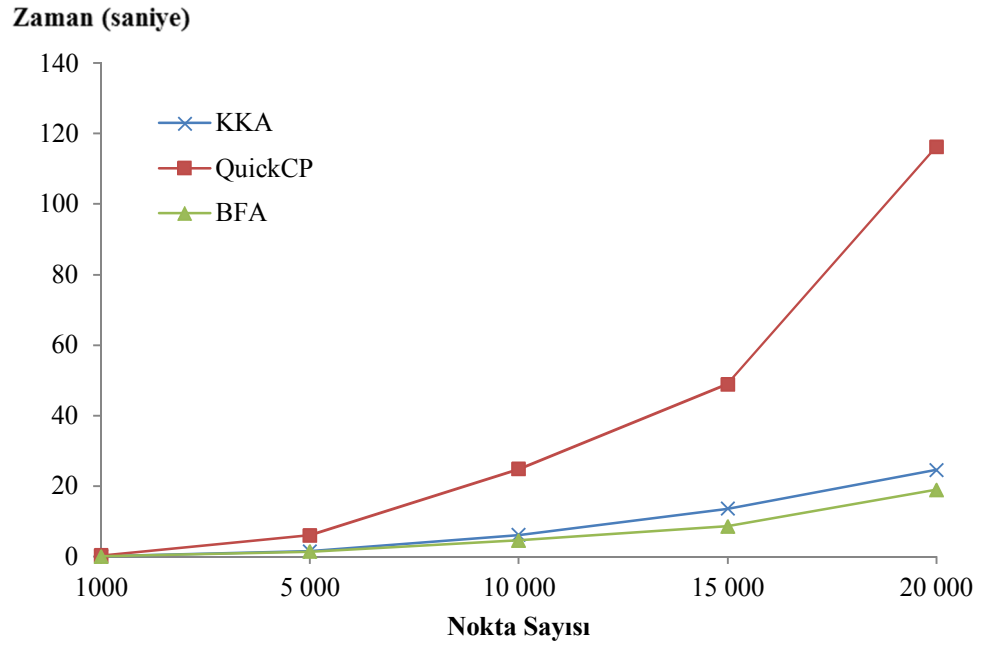


Şekil 3.21 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.21'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.13 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.21)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.046	0.014	0.022
5 000	1.168	0.161	0.180
10 000	4.544	0.970	0.694
15 000	9.903	0.847	1.002
20 000	18.859	1.259	1.786
30 000	40.884	1.132	3.170
40 000	*	1.993	4.662
60 000	*	4.295	8.279
80 000	*	7.151	15.365
100 000	*	11.193	23.928
200 000	*	24.958	65.541

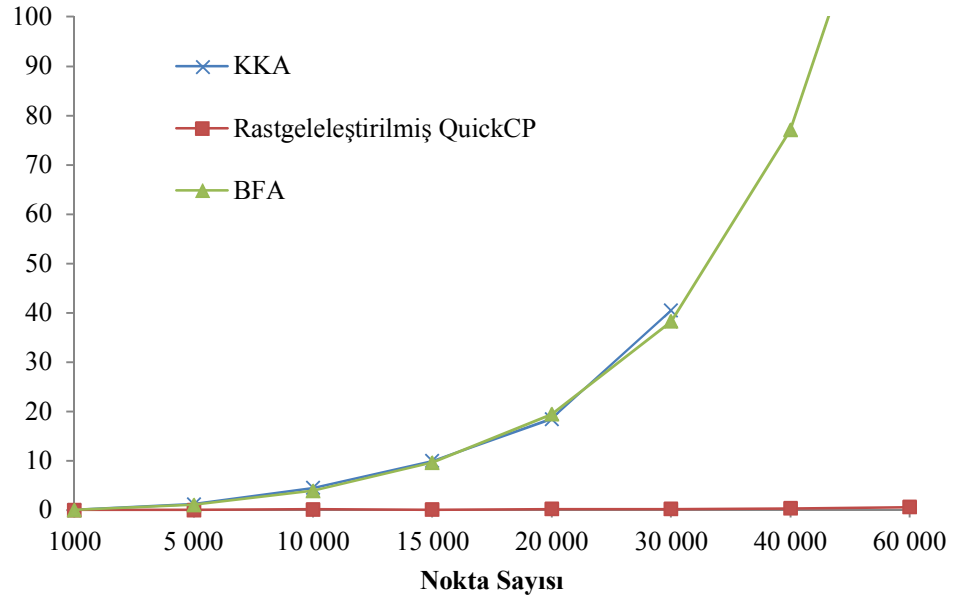


Şekil 3.22 $\mathbb{R}^4$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.22'de nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.14 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.22)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.063	0.269	0.068
5 000	1.530	6.044	1.390
10 000	6.105	24.815	4.553
15 000	13.602	48.888	8.585
20 000	24.591	116.241	18.972

Zaman (saniye)

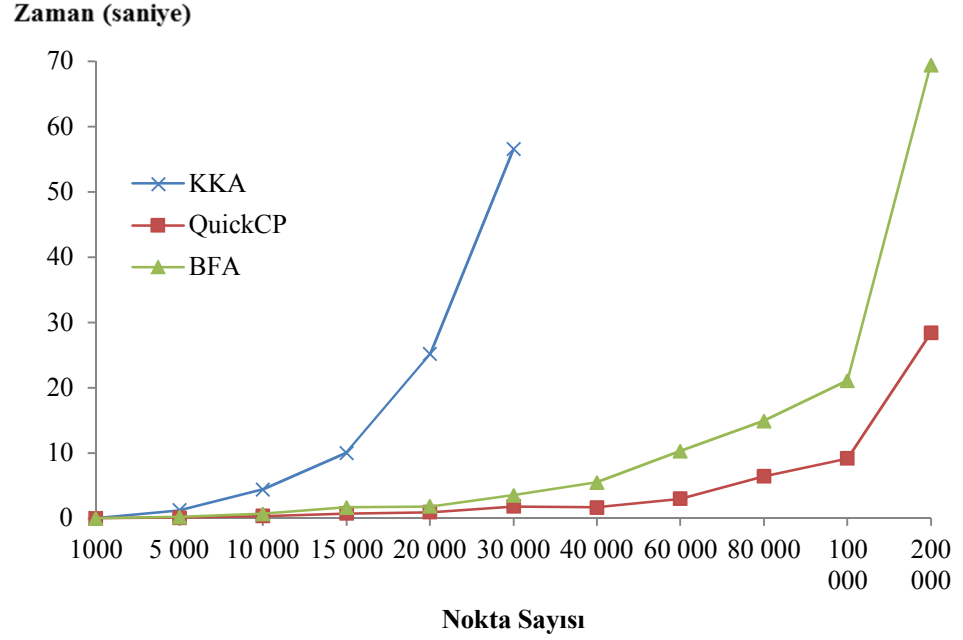
Şekil 3.23 $\mathbb{R}^4$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.23'te çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.15 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.23)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.045	0.016	0.079
5 000	1.203	0.040	1.099
10 000	4.493	0.165	3.970
15 000	9.997	0.143	9.675
20 000	18.443	0.218	19.418
30 000	40.447	0.255	38.236
40 000	*	0.369	77.074
60 000	*	0.620	149.016

3.4.2.3 $\mathbb{R}^4$ 'te tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları



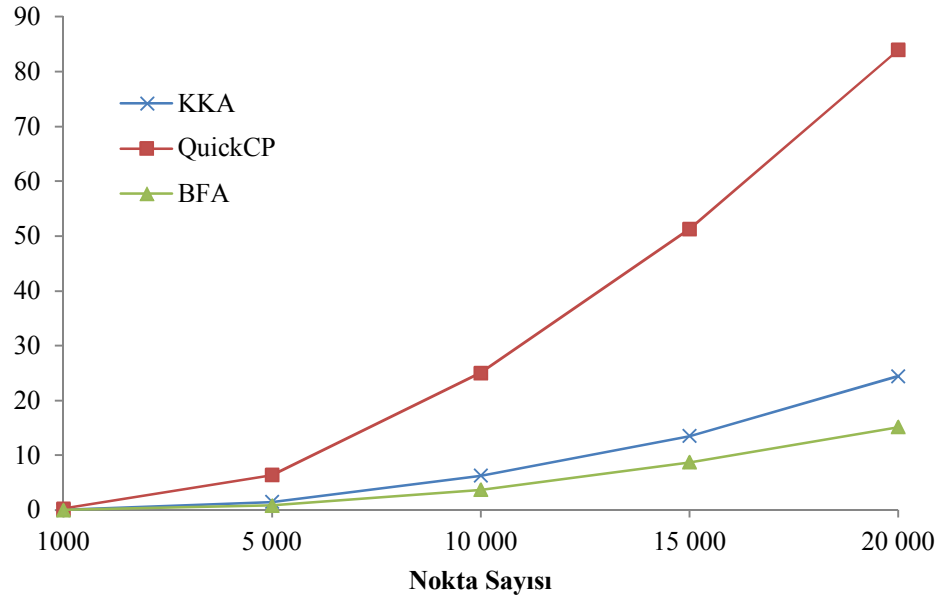
Şekil 3.24 $\mathbb{R}^4$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.24'te KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.16 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.24)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.049	0.008	0.022
5 000	1.242	0.108	0.183
10 000	4.412	0.324	0.694
15 000	10.002	0.751	1.690
20 000	25.132	0.911	1.821
30 000	56.552	1.793	3.532
40 000	*	1.681	5.504
60 000	*	3.006	10.272
80 000	*	6.422	14.908
100 000	*	9.191	21.103
200 000	*	28.411	69.414

Zaman (saniye)

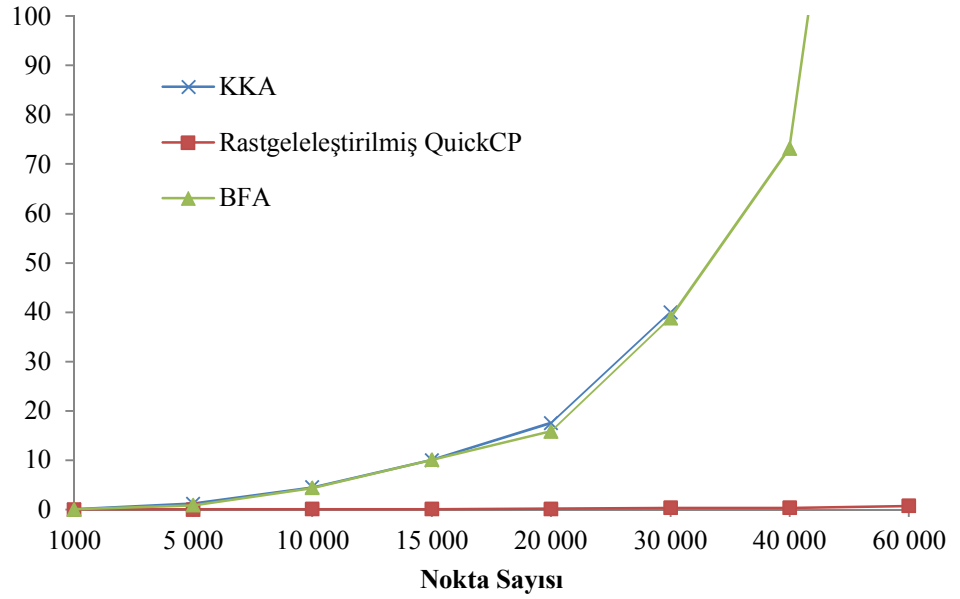


Şekil 3.25 $\mathbb{R}^4$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.25'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.17 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.25)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.060	0.268	0.058
5 000	1.523	6.431	0.849
10 000	6.309	25.031	3.713
15 000	13.545	51.336	8.752
20 000	24.469	83.970	15.179

Zaman (saniye)

Şekil 3.26 $\mathbb{R}^4$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.26'da çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.18 $\mathbb{R}^4$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.26)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.050	0.006	0.041
5 000	1.173	0.025	0.844
10 000	4.513	0.065	4.386
15 000	10.045	0.088	10.073
20 000	17.531	0.144	15.877
30 000	39.991	0.347	38.835
40 000	*	0.333	73.181
60 000	*	0.767	245.143

3.4.3 $\mathbb{R}^5$ 'te algoritmaların çalışma süresi sonuçları

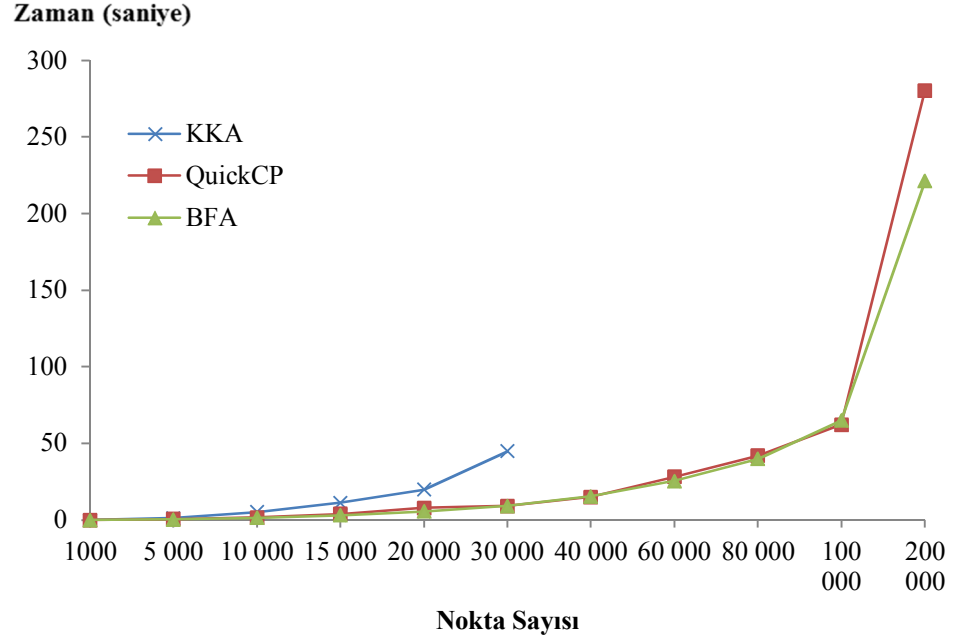
$\mathbb{R}^5$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP algoritması çalıştırılmıştır. Çalışma sonuçları resmedilmiştir. Farklı dağılıma sahip nokta kümelerinde algoritmaların çalışma davranışları da farklı olmuştur. Algoritmanın çalışma sonuçlarına dair yorumlar yaklaşık ortalama değerlerdir:

QuickCP çizgisel yığılım içermeyen normal ve tekdüze dağılımlı nokta kümelerinde en iyi çalışan algoritma olmuştur. Üstel dağılımlı kümelerde KKA QuickCP'nin 3.5 katı, rastgeleleştirilmiş QuickCP'nin 7 katı daha fazla süre harcamıştır. BFA ise QuickCP ile hemen hemen eşit süre harcamıştır. Üstel dağılımlı kümelerde BFA ile QuickCP hemen hemen eşit süre harcadığı için rastgeleleştirmiş QuickCP de denenmiş ve rastgeleleştirilmiş QuickCP BFA'nın yarısı kadar süre harcamıştır. Normal dağılımlı kümelerde KKA QuickCP'nin 5 katı, BFA ise 1.5 katı süre harcamıştır. Tekdüze dağılımlı kümelerde KKA QuickCP'nin 15 katı, BFA ise 2 katı süre harcamıştır. Çizgisel yığılım içermeyen normal ve tekdüze dağılımlı nokta kümelerinde QuickCP rastgeleleştirilirse, QuickCP'nin BFA ve KKA ile arasındaki fark açılacaktır.

Çizgisel yığılım içeren kümelerde en iyi algoritma rastgeleleştirilmiş QuickCP olmuştur. Üstel dağılımlı kümelerde QuickCP KKA'nın 6 katı, BFA 1.2 katı süre harcamıştır. Rastgeleleştirilmiş QuickCP ise KKA'nın 1/150 katı süre harcamıştır. Normal dağılımlı kümelerde QuickCP KKA'nın 5 katı, BFA 0.8 katı süre harcamıştır. Rastgeleleştirilmiş QuickCP ise KKA'nın 1/150 katı süre harcamıştır. Tekdüze dağılımlı kümelerde QuickCP KKA'nın 5 katı, BFA 0.6 katı süre harcamıştır. Rastgeleleştirilmiş QuickCP ise KKA'nın 1/150 katı süre harcamıştır.

Üstel, normal ve tekdüze dağılımlı nokta kümelerinde çizgisel yığılım maksimum durumda iken yapılan rastgeleştirme işlemi QuickCP'nin performansına olumlu etki etmiştir, bu durumdaki kümelerde en iyi çalışan algoritma olmuştur. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.3.1 $\mathbb{R}^5$ 'te üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları



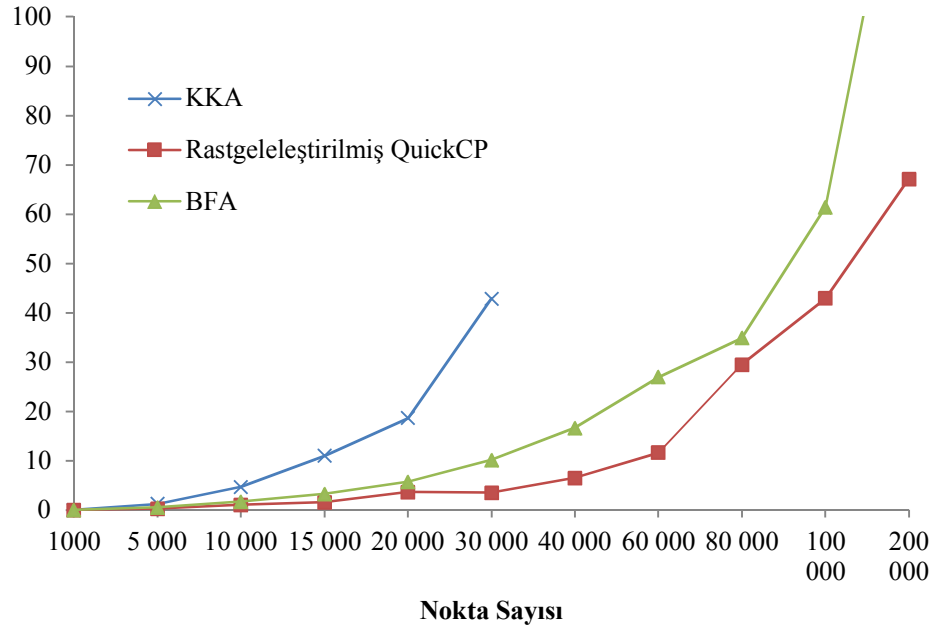
Şekil 3.27 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.27'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. BFA ile QuickCP benzer sürede çalışmış en son kümede BFA, QuickCP'yi geçmiştir.

Tablo 3.19 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.27)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.048	0.052	0.030
5 000	1.205	0.641	0.438
10 000	5.220	1.600	1.552
15 000	11.174	3.976	3.166
20 000	19.887	7.856	5.713
30 000	44.847	9.258	9.152
40 000	*	15.061	15.432
60 000	*	28.130	25.388
80 000	*	42.039	39.906
100 000	*	62.248	64.912
200 000	*	280.298	221.144

Zaman (saniye)

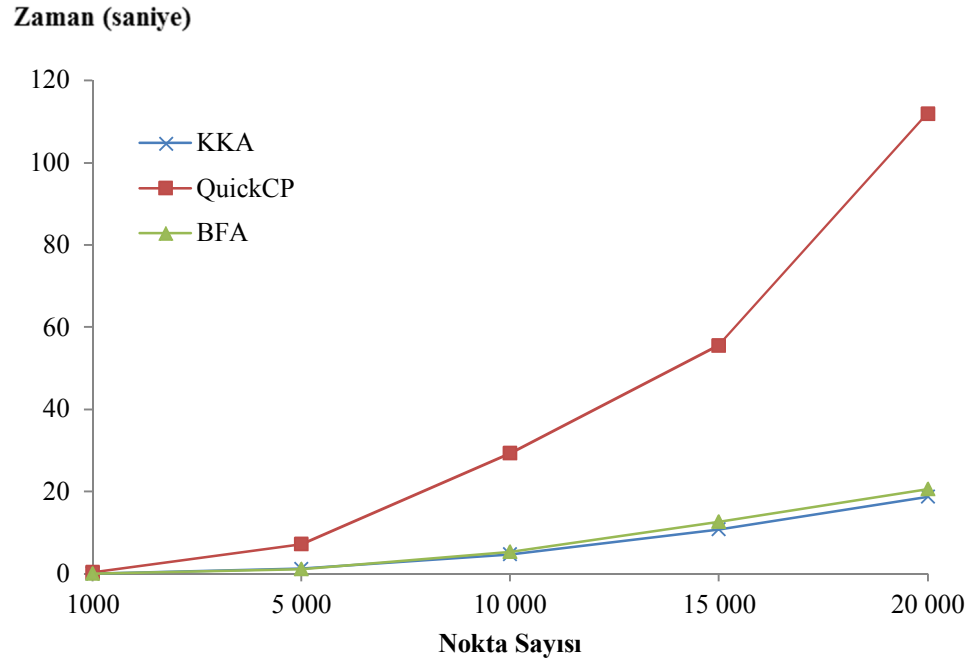


Şekil 3.28 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.28'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. Rastgeleştirme işleminden sonra QuickCP, BFA'yı geçerek en iyi algoritma olmuştur.

Tablo 3.20 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.28)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.050	0.038	0.064
5 000	1.285	0.301	0.571
10 000	4.716	1.103	1.747
15 000	11.042	1.597	3.277
20 000	18.694	3.670	5.753
30 000	42.860	3.531	10.146
40 000	*	6.560	16.636
60 000	*	11.632	26.940
80 000	*	29.520	34.878
100 000	*	42.938	61.421
200 000	*	67.142	146.745

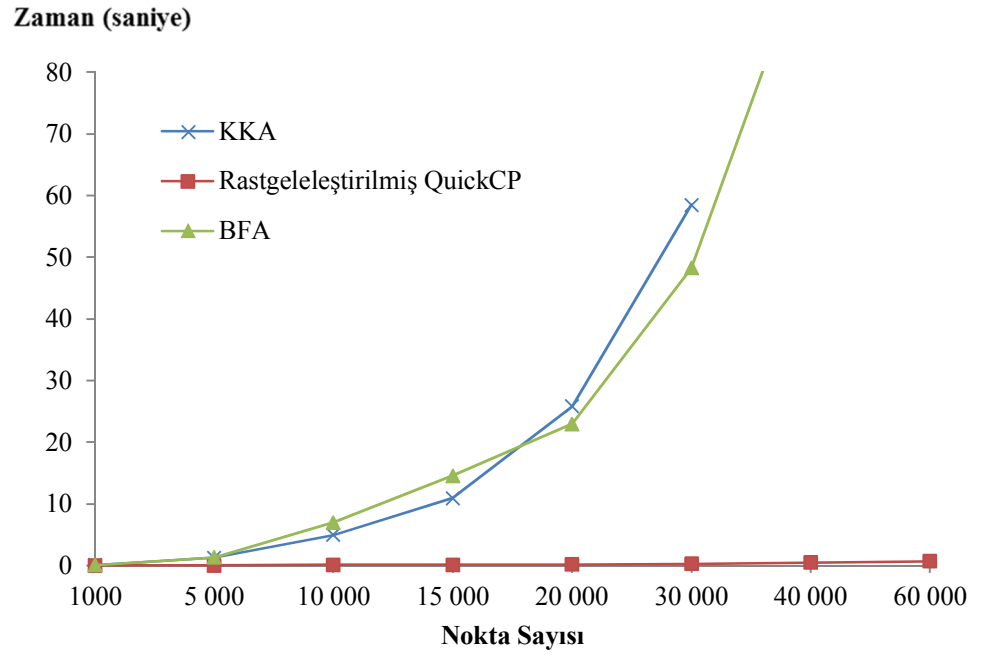


Şekil 3.29 $\mathbb{R}^5$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.29'da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.21 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.29)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.052	0.394	0.054
5 000	1.229	7.212	1.203
10 000	4.772	29.325	5.341
15 000	10.846	55.557	12.690
20 000	18.778	111.971	20.626



Şekil 3.30 $\mathbb{R}^5$ 'te çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

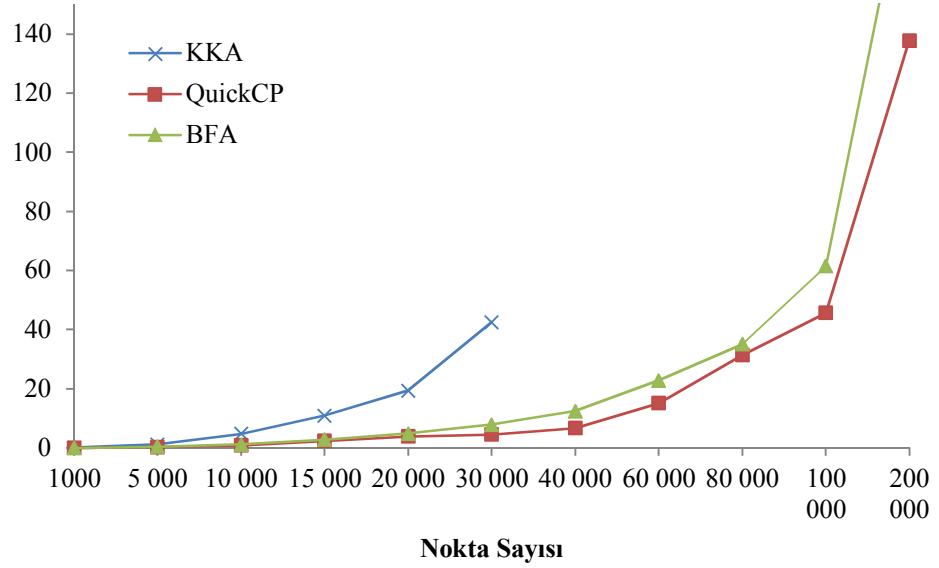
Şekil 3.30'da çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.22 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.30)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.048	0.014	0.091
5 000	1.257	0.031	1.302
10 000	4.962	0.130	6.672
15 000	10.900	0.120	14.573
20 000	25.809	0.165	22.943
30 000	58.451	0.255	48.221
40 000	*	0.513	101.502
60 000	*	0.704	212.546

3.4.3.2 $\mathbb{R}^5$ 'te normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Zaman (saniye)

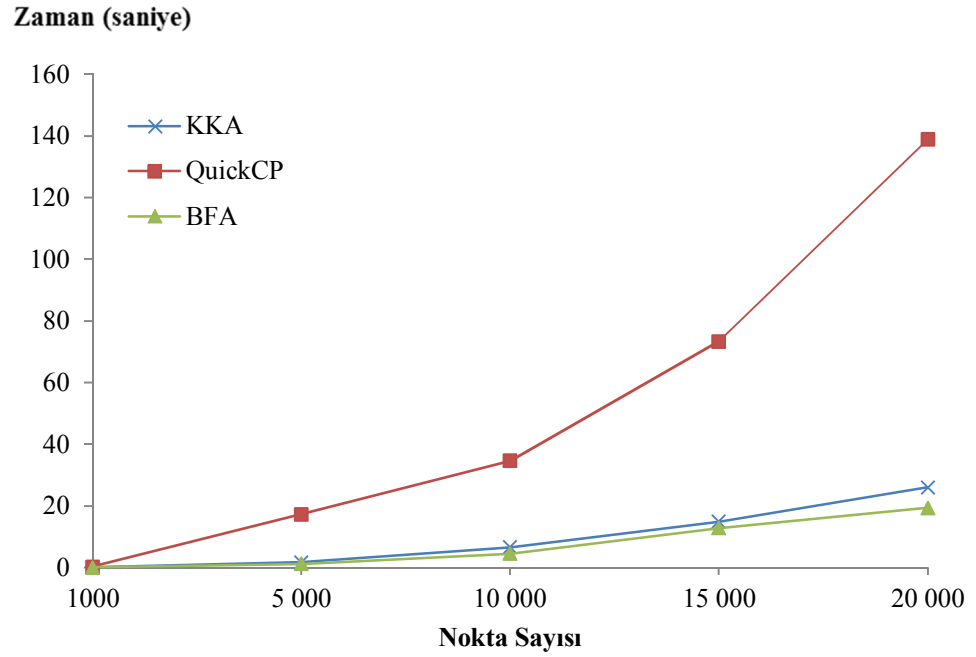


Şekil 3.31 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.31'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.23 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.31)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.051	0.033	0.031
5 000	1.223	0.346	0.366
10 000	4.760	0.867	1.159
15 000	10.918	2.370	2.805
20 000	19.411	3.911	4.863
30 000	42.563	4.537	7.882
40 000	*	6.655	12.379
60 000	*	15.191	22.824
80 000	*	31.429	35.068
100 000	*	45.671	61.421
200 000	*	137.74	200.359

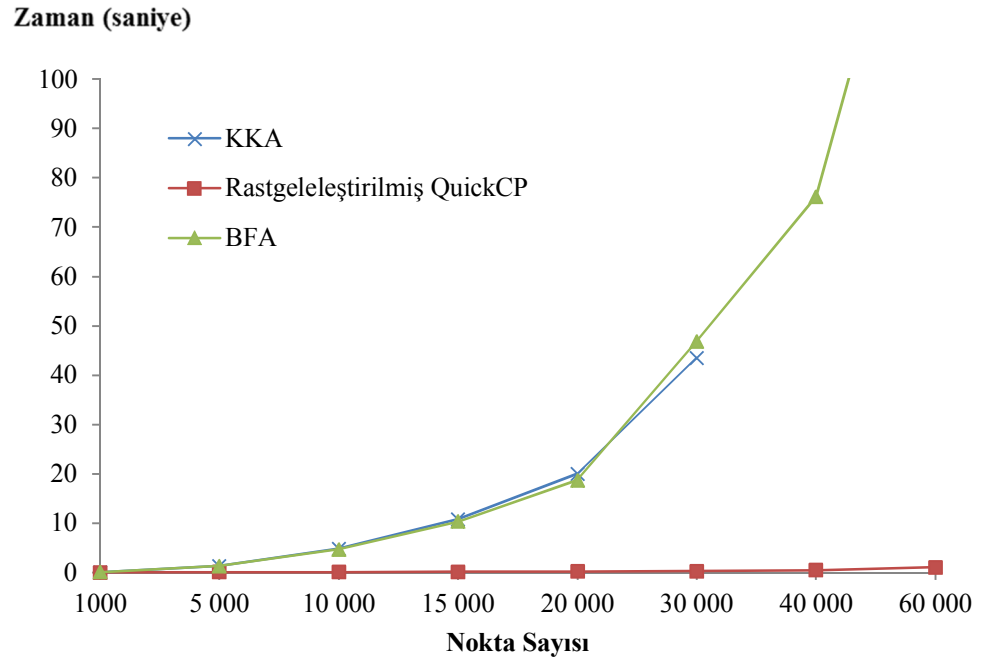


Şekil 3.32 $\mathbb{R}^5$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.32'de nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.24 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.32)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.068	0.256	0.053
5 000	1.695	17.314	1.191
10 000	6.562	34.655	4.527
15 000	14.834	73.304	12.788
20 000	26.027	138.837	19.314



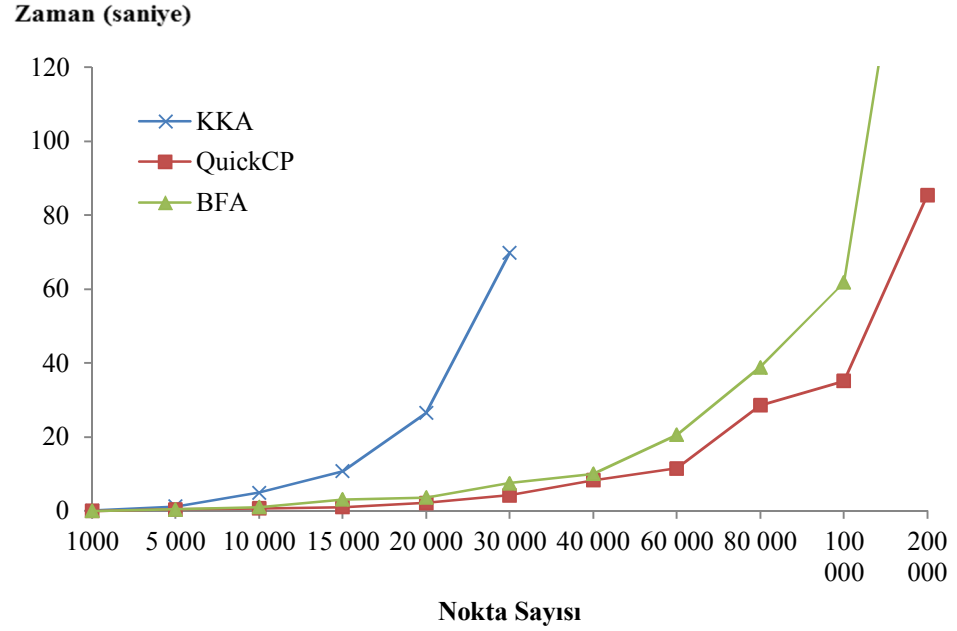
Şekil 3.33 $\mathbb{R}^5$ 'te çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.33'te çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.25 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.33)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.049	0.018	0.088
5 000	1.298	0.059	1.352
10 000	4.816	0.078	4.720
15 000	10.812	0.185	10.373
20 000	20.019	0.229	18.731
30 000	43.516	0.311	46.835
40 000	*	0.480	76.198
60 000	*	1.064	163.467

3.4.3.3 $\mathbb{R}^5$ 'te tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

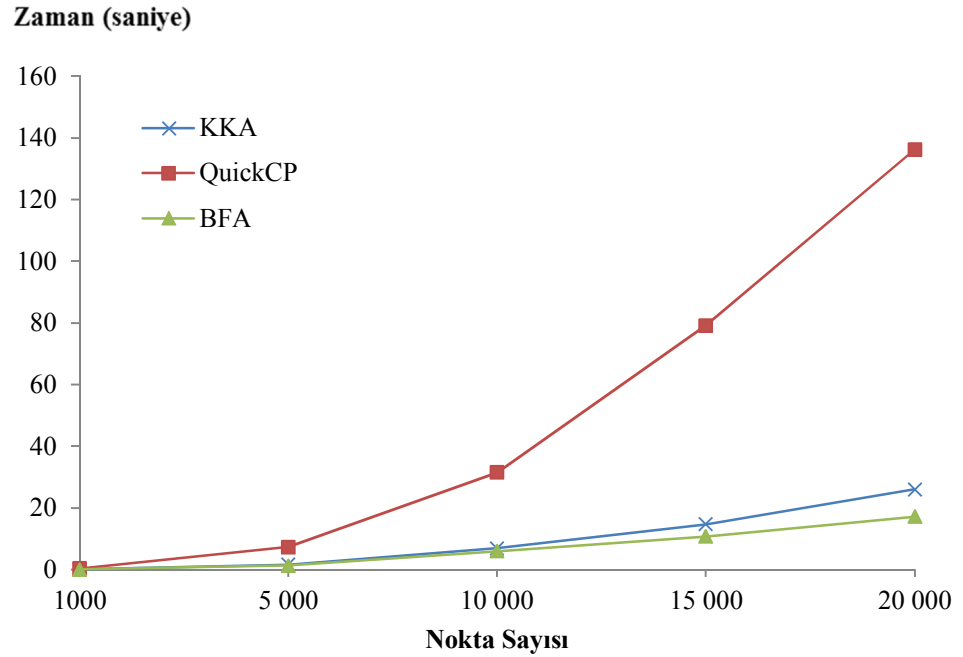


Şekil 3.34 $\mathbb{R}^5$ 'te çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.34'te KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. QuickCP algoritması en iyi çalışan algoritma olmuştur.

Tablo 3.26 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.34)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.052	0.022	0.029
5 000	1.226	0.417	0.443
10 000	4.948	0.692	1.023
15 000	10.705	0.984	3.091
20 000	26.519	2.143	3.599
30 000	69.899	4.191	7.489
40 000	*	8.277	10.059
60 000	*	11.541	20.618
80 000	*	28.554	38.866
100 000	*	35.176	61.829
200 000	*	85.372	203.119

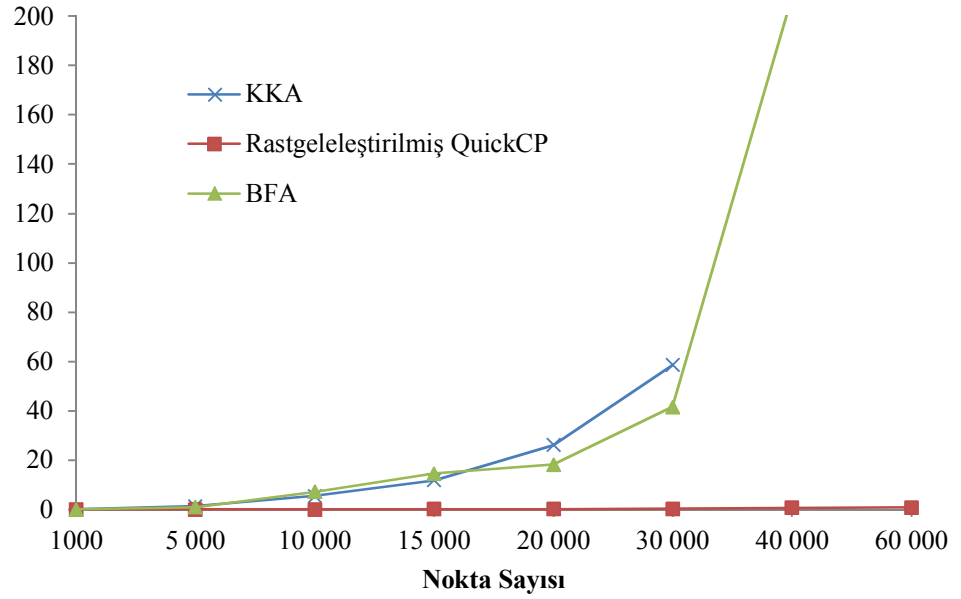


Şekil 3.35 $\mathbb{R}^5$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.35'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.27 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.35)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.070	0.331	0.070
5 000	1.642	7.271	1.244
10 000	6.859	31.541	5.980
15 000	14.722	79.116	10.772
20 000	25.981	136.107	17.173

Zaman (saniye)

Şekil 3.36 $\mathbb{R}^5$ 'te çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.36'da çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.28 $\mathbb{R}^5$ 'te Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.36)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.053	0.010	0.049
5000	1.330	0.055	1.022
10000	5.524	0.106	7.149
15000	11.835	0.12	14.573
20000	26.084	0.163	18.209
30000	58.621	0.275	41.629
40000	*	0.663	207.944
60000	*	0.798	250.278

3.4.4 $\mathbb{R}^8$ 'de algoritmaların çalışma süresi sonuçları

$\mathbb{R}^8$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP algoritması çalıştırılmıştır. Çalışma sonuçları resmedilmiştir. Farklı dağılıma sahip nokta kümelerinde algoritmaların çalışma davranışları da farklı olmuştur. Algoritmanın çalışma sonuçlarına dair yorumlar yaklaşık ortalama değerlerdir:

Çizgisel yığılım içermeyen üstel kümelerde en iyi algoritma rastgeleştirilmişmiş QuickCP, normal dağılımlılarda BFA, tekdüze dağılımlılar ise QuickCP olmuştur. Üstel dağılımlı kümelerde KKA QuickCP'nin 0.7 katı, BFA 0.5 katı, rastgeleştirilmiş QuickCP ise 0.3 katı süre harcamıştır. Normal dağılımlı kümelerde KKA QuickCP'nin 1.1 katı, BFA 0.7 katı, rastgeleştirilmiş QuickCP ise 0.9 katı süre harcamıştır. Tekdüze dağılımlı kümelerde KKA QuickCP'nin 1.5 katı, BFA 1.1 katı, rastgeleştirilmiş QuickCP ise 1.4 katı süre harcamıştır.

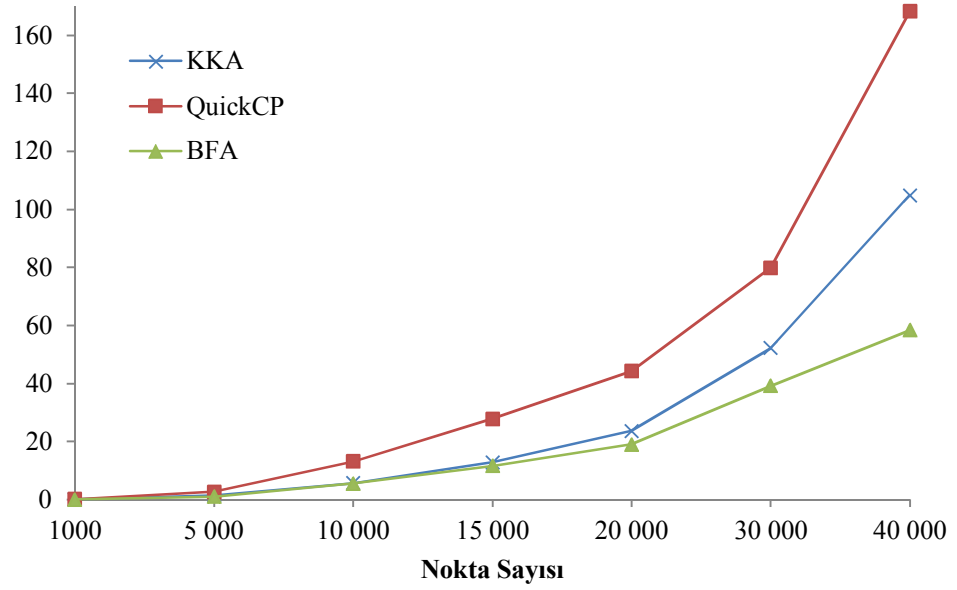
Üstel dağılımlı çizgisel birikim içermeyen kümelerde BFA'nın gerisinde kalan QuickCP, rastgeleştirme işleminden sonra BFA'yı geçerek en iyi algoritma olmuştur. Rastgeleştirme işlemi tekdüze dağılımlı kümelerde QuickCP'nin performansını kötü etkilemiştir.

Çizgisel yığılım içeren kümelerde en iyi algoritma rastgeleştirilmiş QuickCP olmuştur. Üstel dağılımlı kümelerde QuickCP KKA'nın 10 katı, BFA 1.2 katı, rastgeleştirilmiş QuickCP ise 1/20 katı süre harcamıştır. Normal dağılımlı kümelerde QuickCP KKA'nın 8 katı, BFA 0.8 katı, rastgeleştirilmiş QuickCP ise 1/7 katı süre harcamıştır. Tekdüze dağılımlı kümelerde QuickCP KKA'nın 12 katı, BFA 1.5 katı, rastgeleştirilmiş QuickCP ise 1/7 katı süre harcamıştır.

Üstel, normal ve tekdüze dağılımlı nokta kümelerinde çizgisel yığılım maksimum durumda iken yapılan rastgeleştirme işlemi QuickCP'nin performansına olumlu etki etmiştir, bu tür kümelerde rastgeleştirilmiş QuickCP en iyi çalışan algoritma olmuştur. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.4.1 $\mathbb{R}^8$ 'de üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Zaman (saniye)

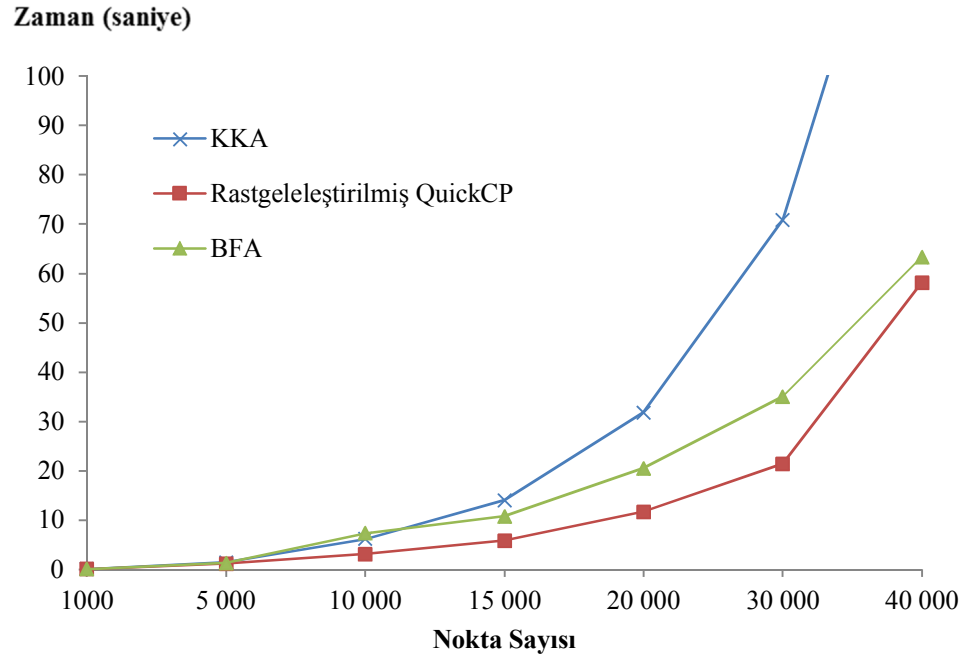


Şekil 3.37 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.37'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir.

Tablo 3.29 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.37)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.057	0.174	0.074
5 000	1.454	2.656	1.044
10 000	5.619	13.187	5.560
15 000	12.845	27.804	11.616
20 000	23.635	44.271	19.050
30 000	52.253	79.949	39.169
40 000	104.838	168.207	58.380

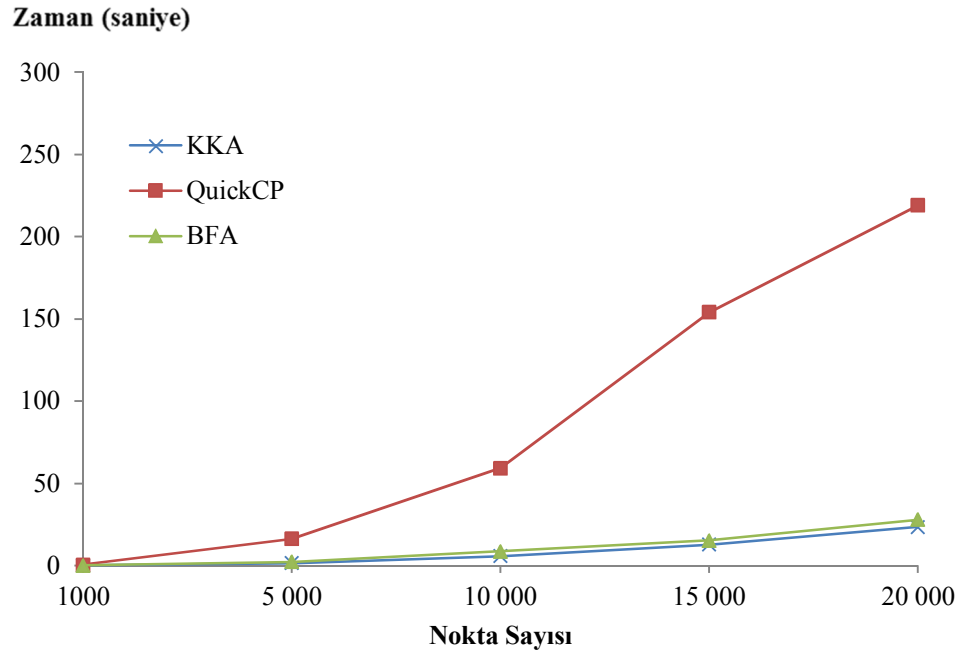


Şekil 3.38 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.38'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı KKA göstermiştir. Rastgeleştirme işleminden sonra QuickCP en iyi çalışan algoritma olmuştur.

Tablo 3.30 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.38)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.060	0.080	0.073
5 000	1.465	1.167	1.342
10 000	6.180	3.147	7.370
15 000	14.019	5.841	10.811
20 000	31.820	11.672	20.504
30 000	70.894	21.458	35.054
40 000	161.511	58.159	63.332

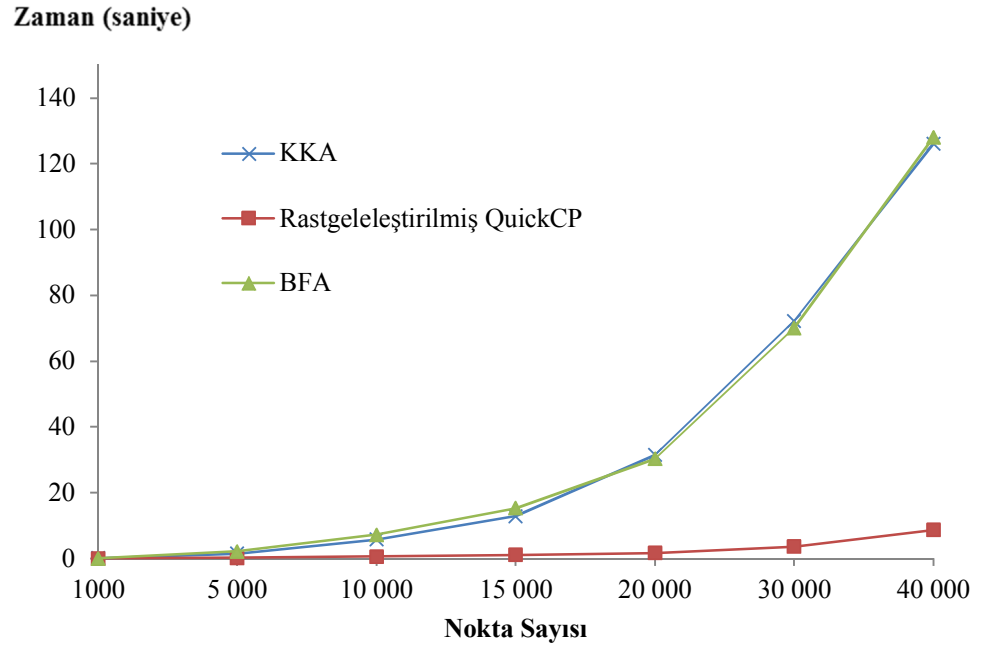


Şekil 3.39 $\mathbb{R}^{8^4}$ ’de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP’nin çalışma süreleri

Şekil 3.39’da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP’nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.31 $\mathbb{R}^{8^4}$ ’de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.39)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.063	0.524	0.082
5 000	1.508	16.142	2.212
10 000	5.710	59.168	8.604
15 000	12.746	154.067	15.219
20 000	23.448	219.091	27.983



Şekil 3.40 $\mathbb{R}^{8^c}$ ’de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP’nin çalışma süreleri

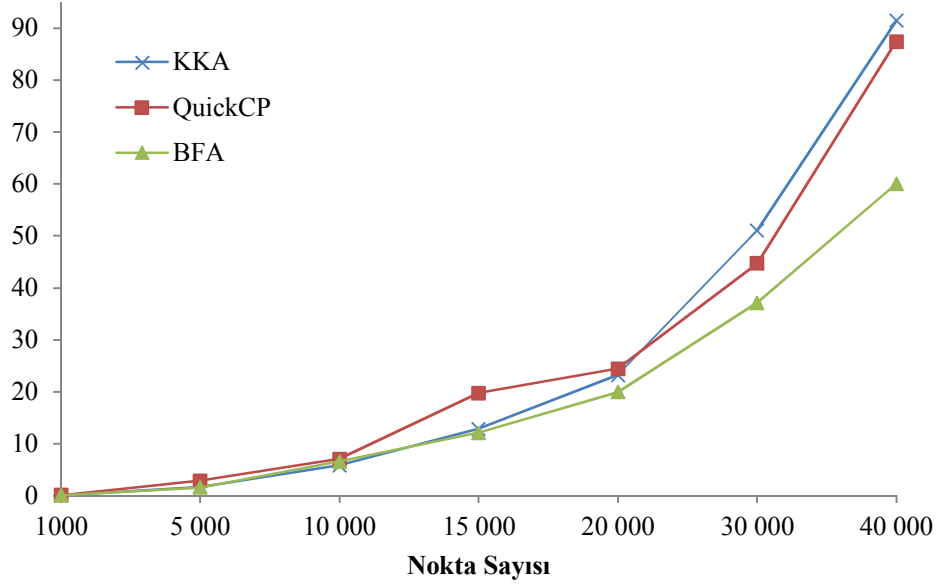
Şekil 3.40’ta çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.32 $\mathbb{R}^{8^c}$ ’de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.40)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.145	0.028	0.075
5 000	1.528	0.270	2.145
10 000	5.814	0.636	7.193
15 000	12.922	1.073	15.307
20 000	31.437	1.650	30.279
30 000	72.273	3.630	70.068
40 000	126.077	8.710	128.016

3.4.4.2 $\mathbb{R}^8$ 'de normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Zaman (saniye)

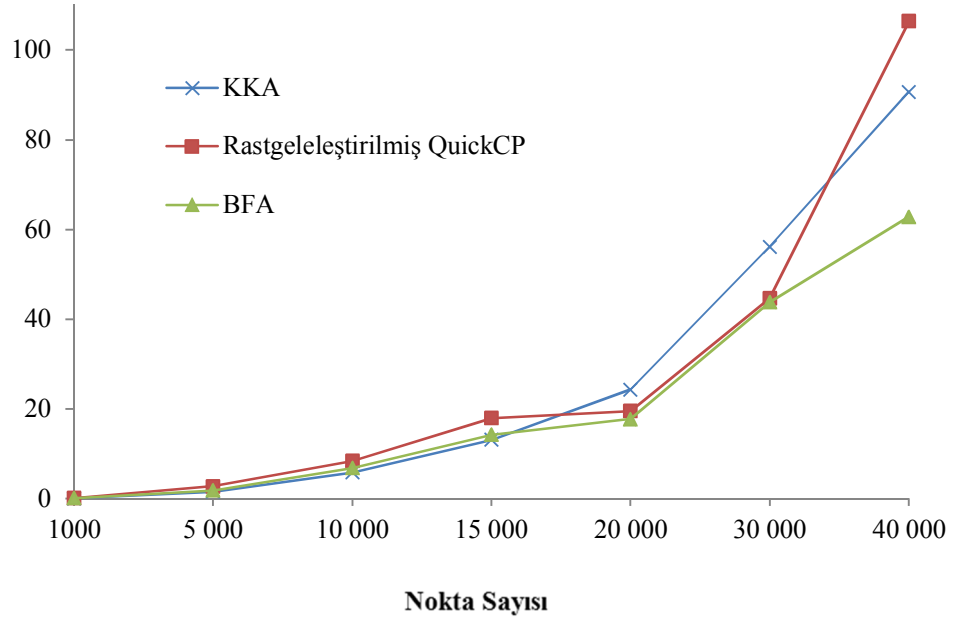


Şekil 3.41 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.41'de KKA, QuickCP algoritması ve BFA çalışma süresi anlamında benzer sürede çalışmakla birlikte BFA az farkla da olsa diğerlerinden daha iyi performans göstermiştir.

Tablo 3.33 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.41)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.060	0.123	0.093
5 000	1.686	2.868	1.582
10 000	5.798	7.031	6.582
15 000	12.847	19.788	12.127
20 000	23.223	24.435	19.951
30 000	51.043	44.733	37.137
40 000	91.455	87.431	59.994

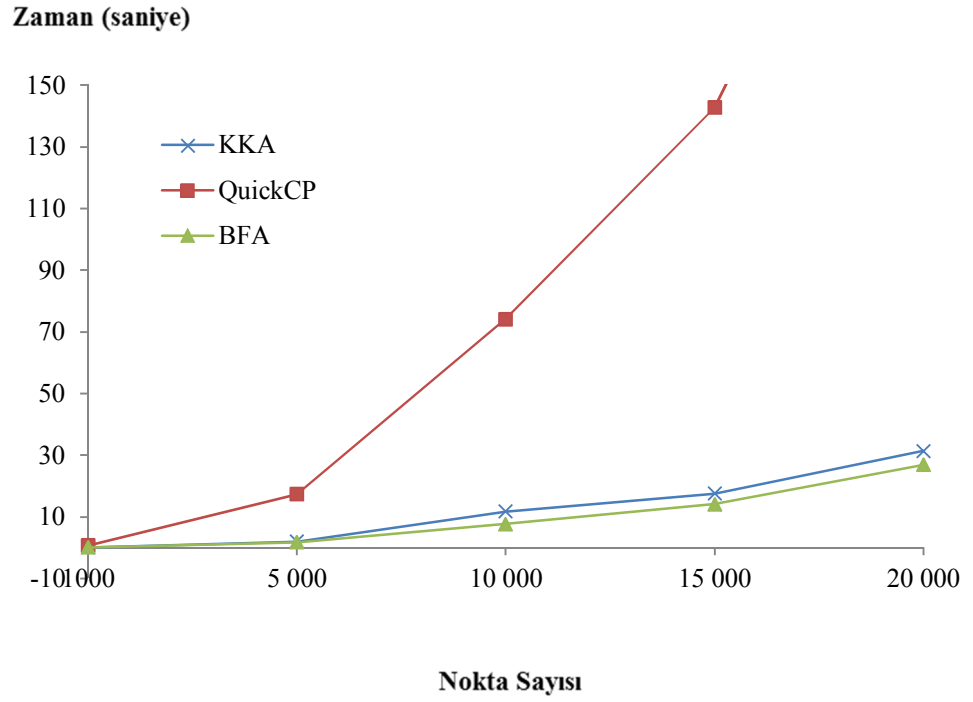
Zaman (saniye)

Şekil 3.42 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.42'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA benzer sürede çalışmıştır.

Tablo 3.34 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.42)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.059	0.127	0.077
5 000	1.537	2.746	1.807
10 000	5.811	8.390	6.819
15 000	13.136	17.930	14.277
20 000	24.298	19.549	17.720
30 000	56.117	44.667	43.803
40 000	90.643	106.460	62.871

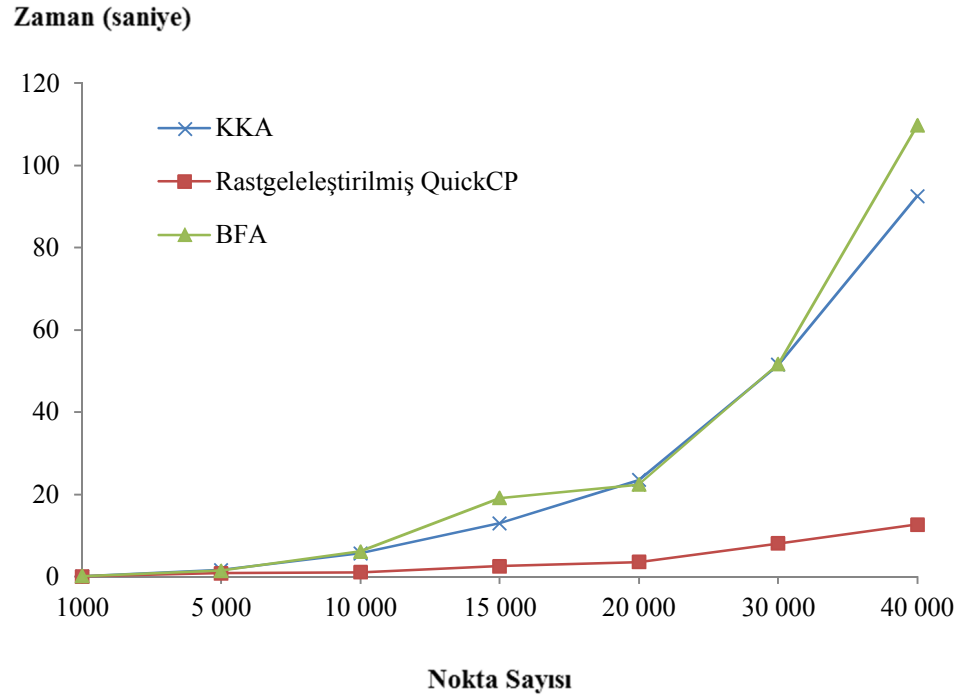


Şekil 3.43 $\mathbb{R}^8$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.43'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.35 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.43)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.078	0.741	0.095
5 000	1.937	17.36	1.817
10 000	11.713	74.153	7.655
15 000	17.564	142.737	14.087
20 000	31.351	279.649	26.872



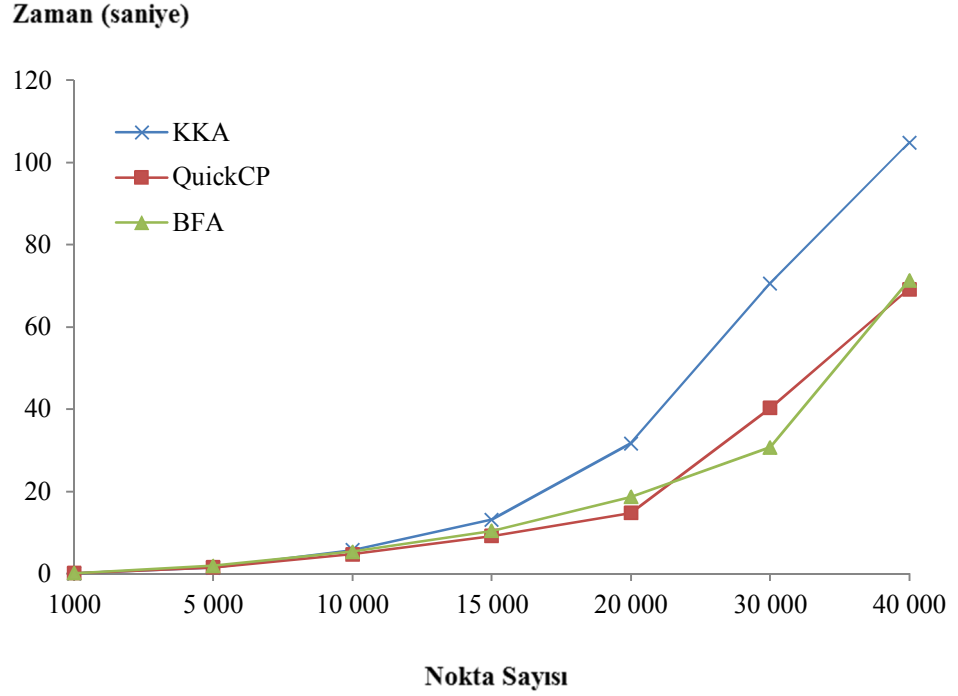
Şekil 3.44 $\mathbb{R}^8$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.44'te çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.36 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.44)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.066	0.066	0.085
5 000	1.585	0.774	1.402
10 000	5.666	1.037	6.114
15 000	12.965	2.544	19.119
20 000	23.457	3.526	22.407
30 000	51.475	8.056	51.658
40 000	92.494	12.634	109.750

3.4.4.3 $\mathbb{R}^8$ 'de tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

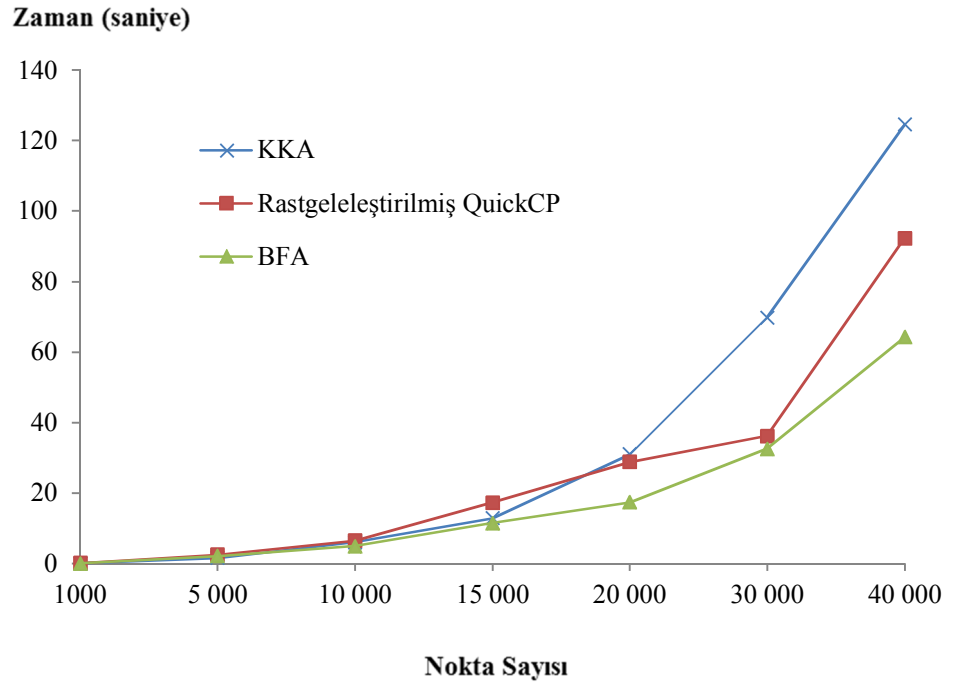


Şekil 3.45 $\mathbb{R}^8$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.45'te KKA ve BFA, QuickCP algoritmalarının performansları arasında pek fark yoktur. KKA diğerlerinden az farkla daha kötü performans göstermiştir.

Tablo 3.37 $\mathbb{R}^8$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.45)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.059	0.095	0.077
5 000	1.452	1.522	1.907
10 000	5.750	4.704	5.312
15 000	13.078	9.120	10.362
20 000	31.629	14.722	18.600
30 000	70.583	40.315	30.729
40 000	*	69.136	71.336

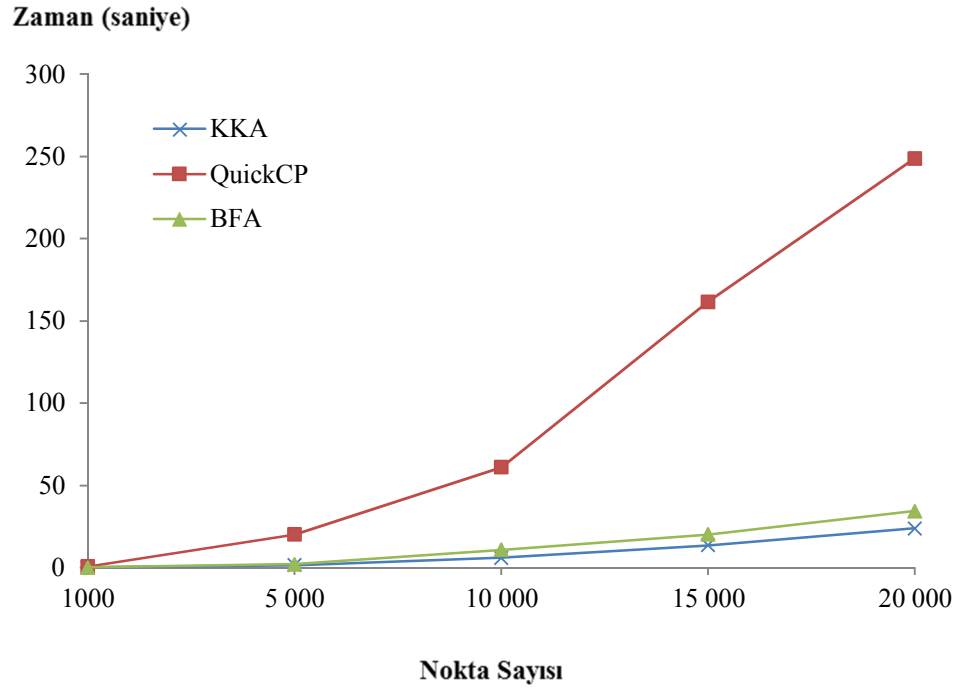


Şekil 3.46 $\mathbb{R}^{8^4}$ ’de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP’nin çalışma süreleri

Şekil 3.46’da KKA, BFA ve rastgeleleştirilmiş QuickCP algoritmaları burada benzer sürede çalışmıştır. En kötü çalışan algoritma az farkla KKA, en iyi çalışan algoritma az farkla BFA olmuştur.

Tablo 3.38 $\mathbb{R}^{8^4}$ ’de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.46)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.063	0.105	0.078
5 000	1.553	2.452	2.158
10 000	6.035	6.518	4.959
15 000	12.881	17.335	11.438
20 000	30.982	28.831	17.412
30 000	69.793	36.179	32.562
40 000	124.552	92.285	64.255

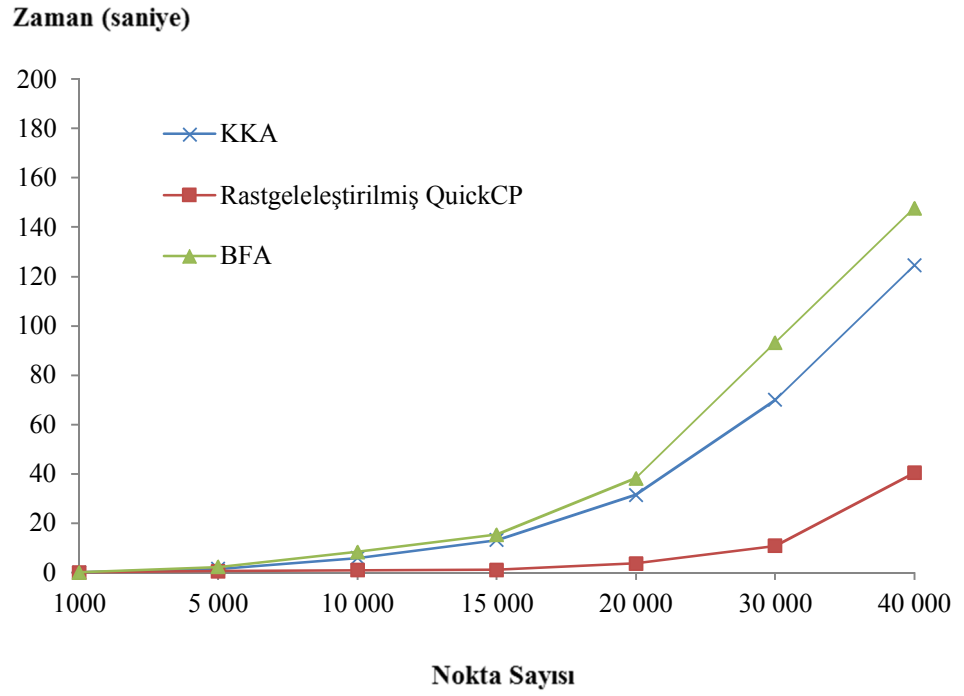


Şekil 3.47 $\mathbb{R}^{8^c}$ ’de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP’nin çalışma süreleri

Şekil 3.47’de nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP’nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.39 $\mathbb{R}^{8^c}$ ’de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.47)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.066	0.602	0.093
5 000	1.519	20.196	2.062
10 000	5.983	60.962	10.798
15 000	13.543	161.676	20.121
20 000	23.871	248.908	34.258



Şekil 3.48 $\mathbb{R}^{8^c}$ ’de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP’nin çalışma süreleri

Şekil 3.48’de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir.

Tablo 3.40 $\mathbb{R}^{8^c}$ ’de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.48)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.065	0.047	0.109
5 000	1.536	0.594	2.211
10 000	5.938	0.998	8.329
15 000	13.110	1.073	15.307
20 000	31.481	3.730	38.213
30 000	70.043	10.865	93.158
40 000	124.543	40.461	147.705

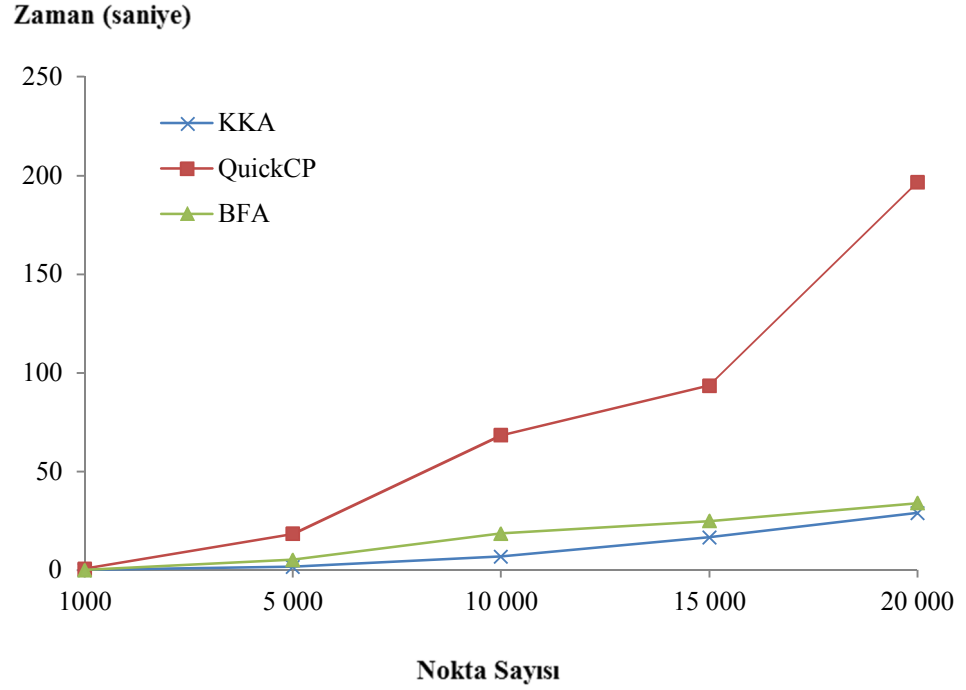
3.4.5 $\mathbb{R}^{12}$ 'de algoritmaların çalışma süresi sonuçları

$\mathbb{R}^{12}$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP algoritması çalıştırılmıştır. Çalışma sonuçları resmedilmiştir. Farklı dağılıma sahip nokta kümelerinde algoritmaların çalışma davranışları da farklı olmuştur. Algoritmanın çalışma sonuçlarına dair yorumlar yaklaşık ortalama değerlerdir:

Çizgisel yığılım içermeyen üstel, normal ve tekdüze dağılımlı nokta kümelerinde en iyi performansı KKA göstermiştir, rastgeleleştirme işlemi QuickCP algoritmasını performans anlamında iyi etkilemiştir. Ancak QuickCP'nin KKA'yı geçmesine yetmemiştir. Üstel dağılımlı kümelerde KKA QuickCP'nin 1/7 katı, BFA 1/6 katı, rastgeleleştirilmiş QuickCP ise 1/6 katı süre harcamıştır. Normal dağılımlı kümelerde KKA QuickCP'nin 1/4 katı, BFA 1/2 katı, rastgeleleştirilmiş QuickCP ise 1.2 katı süre harcamıştır. Tekdüze dağılımlı kümelerde KKA QuickCP'nin 1/2 katı, BFA 0.6 katı, rastgeleleştirilmiş QuickCP ise 1.5 katı süre harcamıştır.

Çizgisel yığılım içeren üstel dağılımlı kümelerde en iyi çalışan algoritma rastgeleleştirilmiş QuickCP; normal ve tekdüze dağılımlılarda KKA olmuştur. Üstel dağılımlı kümelerde QuickCP KKA'nın 20 katı, BFA 1.5 katı, rastgeleleştirilmiş QuickCP ise 1/2 katı süre harcamıştır. Normal dağılımlı kümelerde QuickCP KKA'nın 15 katı, BFA 1.1 katı, rastgeleleştirilmiş QuickCP ise 1.5 katı süre harcamıştır. Tekdüze dağılımlı kümelerde QuickCP KKA'nın 20 katı, BFA 2 katı, rastgeleleştirilmiş QuickCP ise 1.2 katı süre harcamıştır. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.5.1 $\mathbb{R}^{12}$ 'de üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

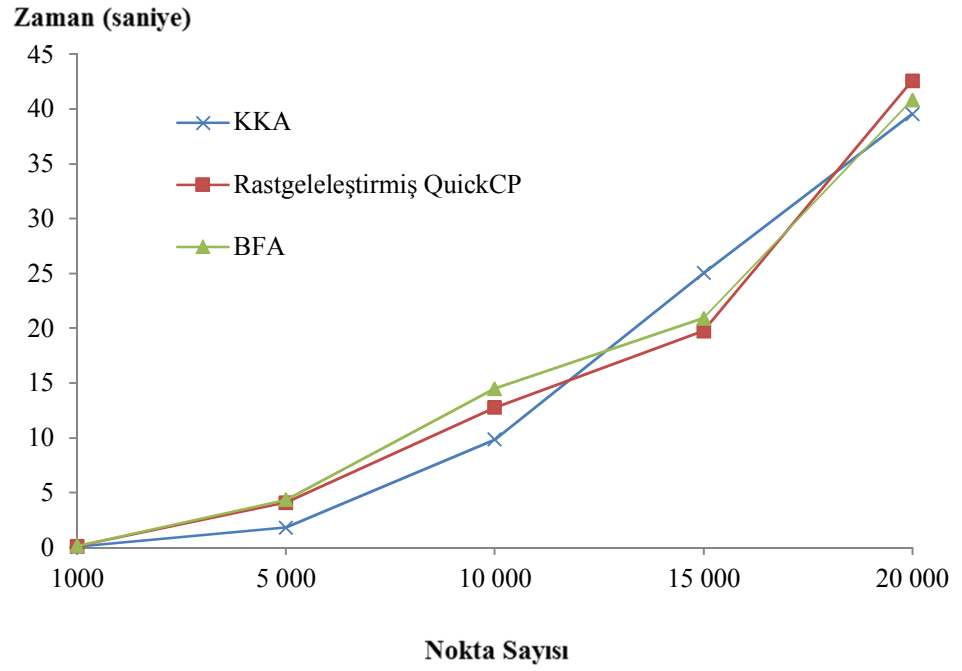


Şekil 3.49 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.49'da KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.41 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.49)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.070	0.726	0.170
5 000	1.771	18.456	5.263
10 000	6.912	68.446	18.634
15 000	16.677	93.507	24.928
20 000	28.967	196.502	34.008

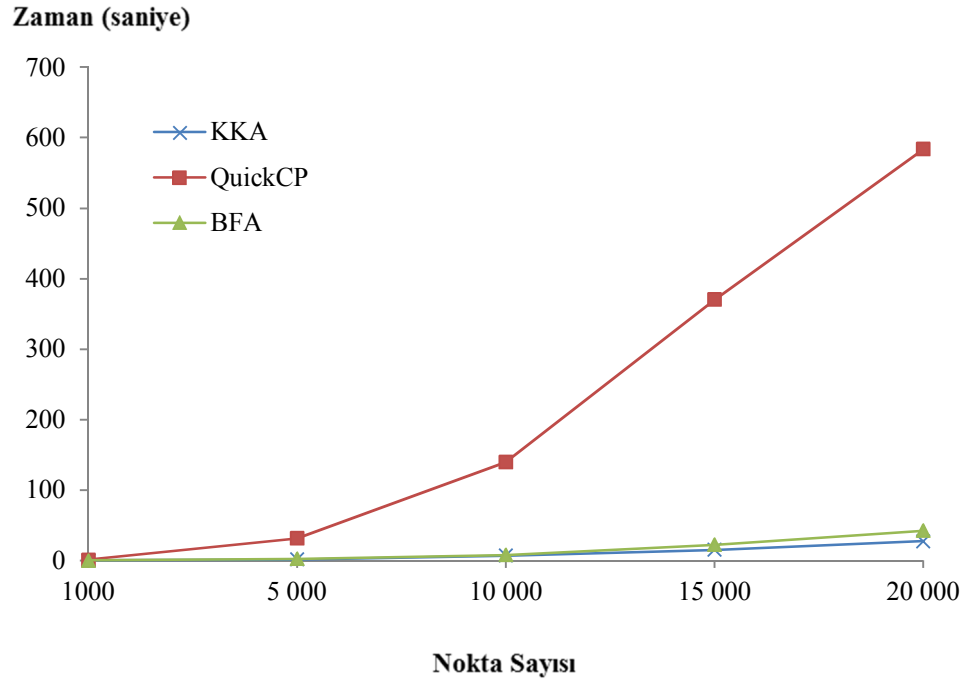


Şekil 3.50 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.50'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA benzer sürede çalışmıştır.

Tablo 3.42 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.50)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.069	0.141	0.156
5 000	1.818	4.087	4.355
10 000	9.842	12.761	14.507
15 000	25.042	19.758	20.905
20 000	39.547	42.593	40.841

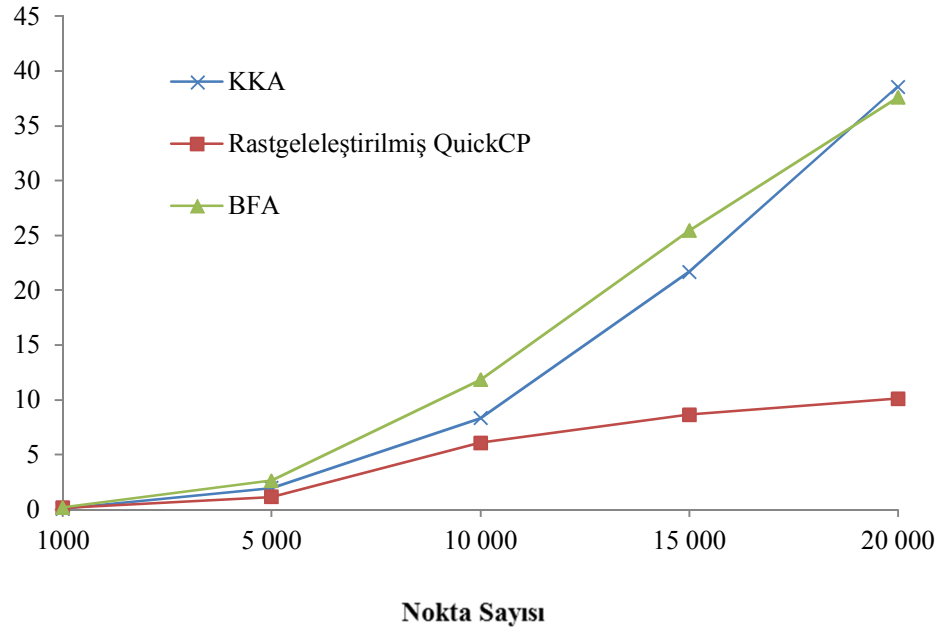


Şekil 3.51 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.51'de nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.43 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.51)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.078	0.985	0.133
5 000	1.778	31.486	2.581
10 000	6.895	139.692	8.202
15 000	15.399	370.286	22.702
20 000	27.546	583.881	42.533

Zaman (saniye)

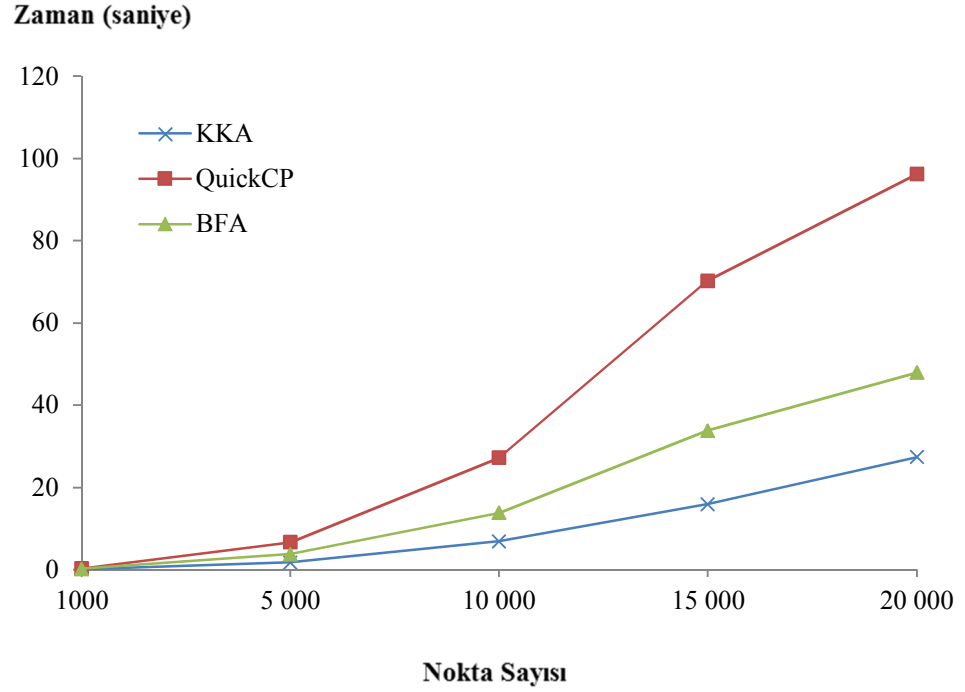
Şekil 3.52 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.52'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Nokta kümesinde çizgisel birikimden kurtulan rastgeleleştirilmiş QuickCP, üç algoritma içinde en iyi performansı göstermiştir. Çalışma süresi anlamında en kötü performansı ise BFA göstermiştir.

Tablo 3.44 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.52)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.076	0.171	0.196
5 000	1.967	1.144	2.652
10 000	8.340	6.080	11.847
15 000	21.668	8.644	25.442
20 000	38.553	10.115	37.600

3.4.5.2 $\mathbb{R}^{12}$ 'de normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları



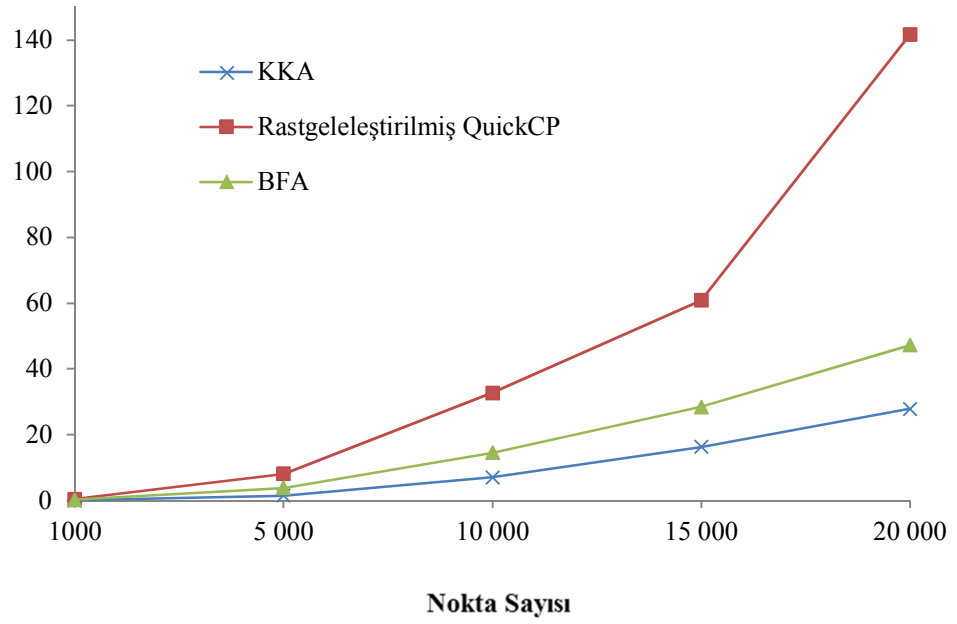
Şekil 3.53 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.53'te KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.45 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.53)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.072	0.281	0.155
5 000	1.764	6.676	3.776
10 000	6.887	27.194	13.795
15 000	15.892	70.270	33.789
20 000	27.370	96.148	47.911

Zaman (saniye)

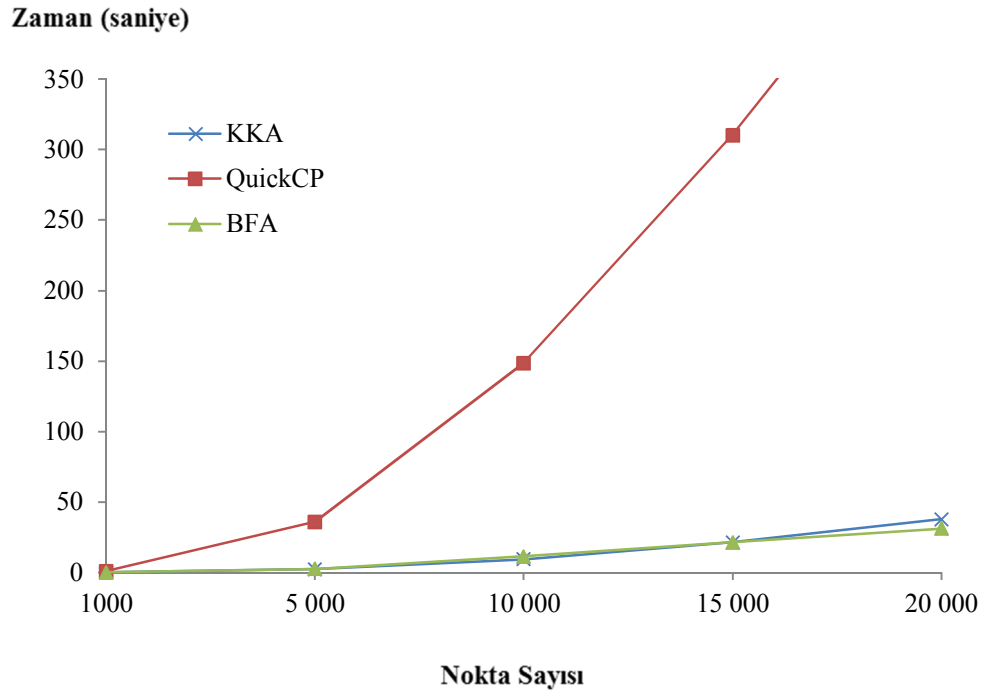


Şekil 3.54 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.54'te KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.46 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.54)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.059	0.411	0.195
5 000	1.537	8.145	3.862
10 000	7.025	32.714	14.529
15 000	16.288	60.855	28.458
20 000	27.857	141.613	47.196

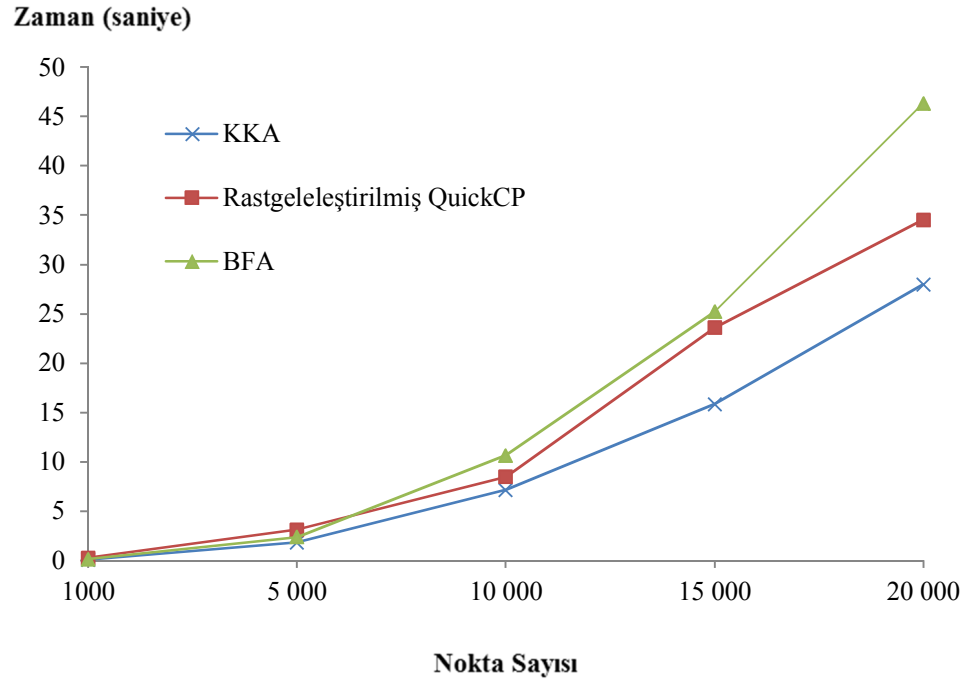


Şekil 3.55 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.55'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.47 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.55)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.095	1.051	0.112
5 000	2.357	35.913	2.590
10 000	9.506	148.663	11.440
15 000	21.331	310.252	21.499
20 000	37.784	495.936	31.255



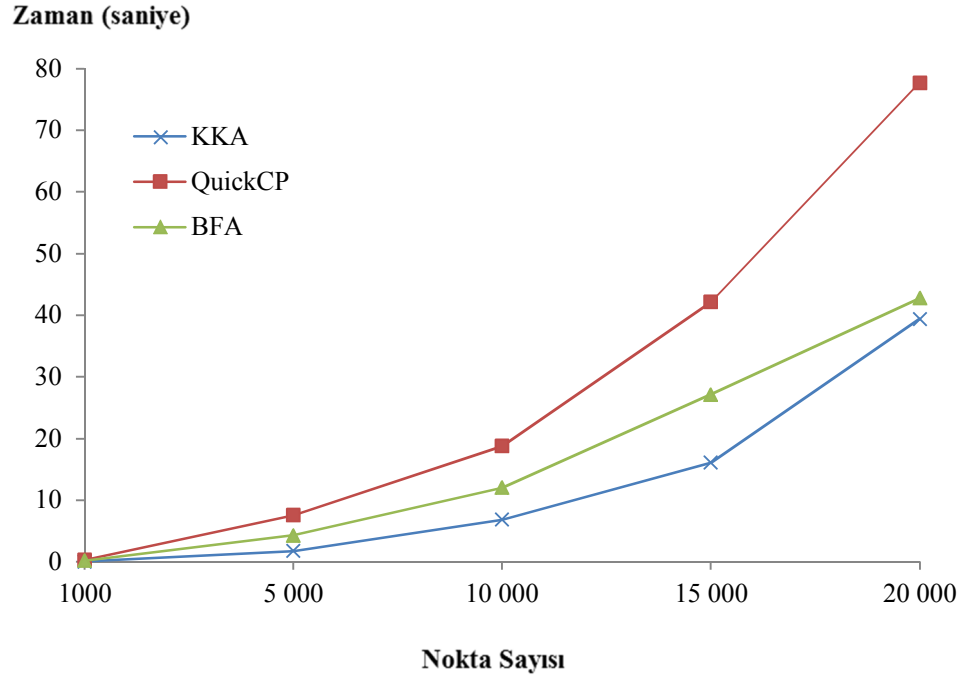
Şekil 3.56 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.56'da çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Rastgeleştirme işlemine rağmen QuickCP KKA'nın gerisinde kalmıştır.

Tablo 3.48 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.56)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.072	0.263	0.148
5 000	1.839	3.119	2.365
10 000	7.139	8.465	10.663
15 000	15.828	23.594	25.224
20 000	27.972	34.536	46.340

3.4.5.3 $\mathbb{R}^{12}$ 'de tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

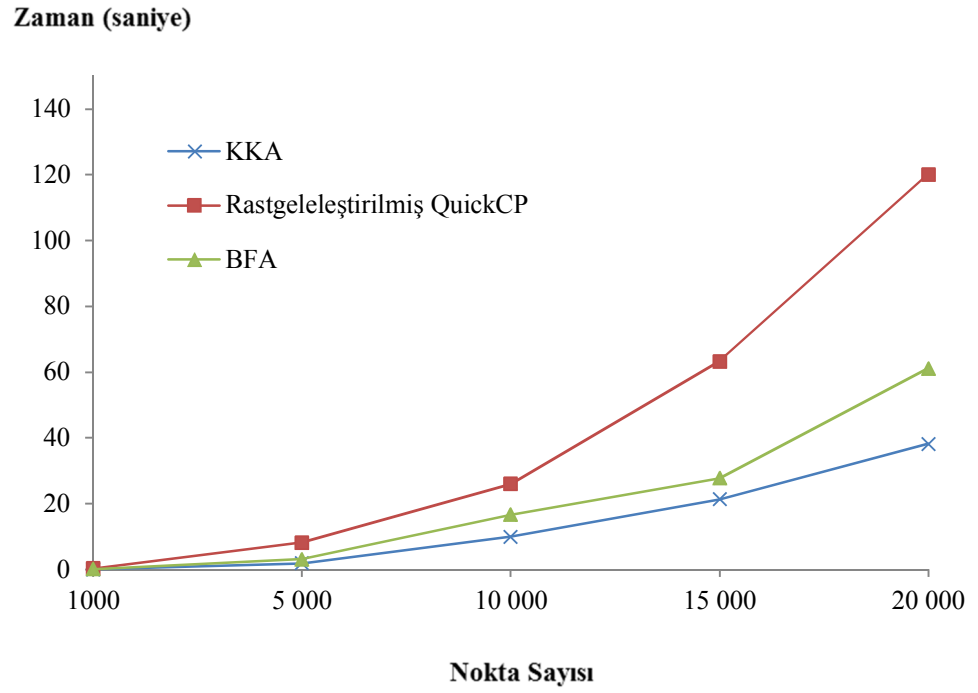


Şekil 3.57 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.57'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.49 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.57)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0702	0.328	0.197
5 000	1.7518	7.605	4.304
10 000	6.8704	18.784	12.023
15 000	16.1102	42.142	27.141
20 000	39.3750	77.678	42.782

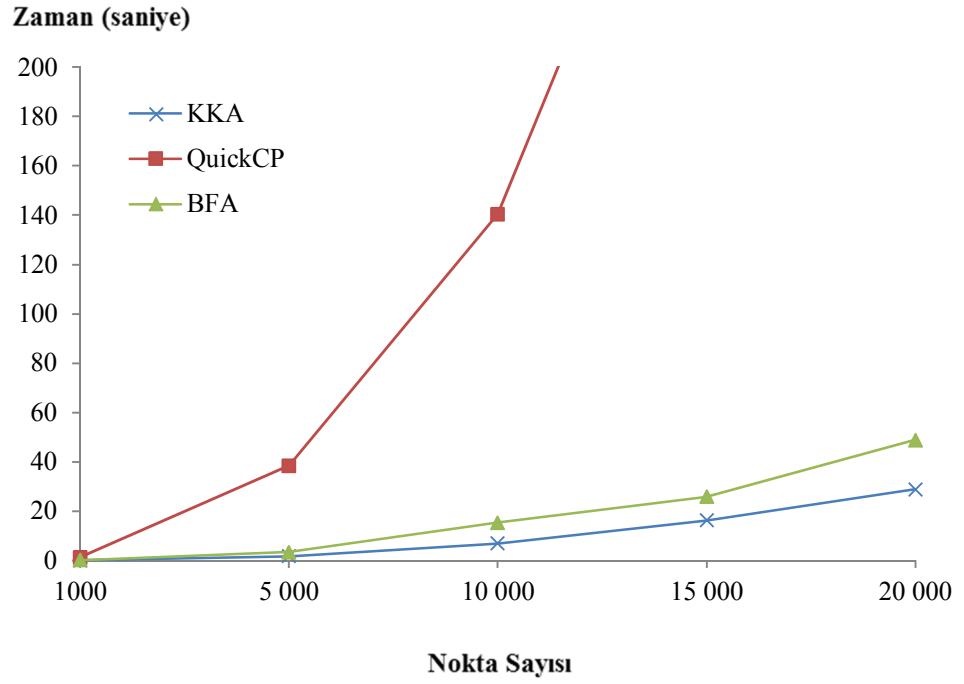


Şekil 3.58 $\mathbb{R}^{12}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.58'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en iyi performansı KKA göstermiştir. En kötü performansı rastgeleleştirilmiş QuickCP göstermiştir.

Tablo 3.50 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.58)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.072	0.362	0.164
5 000	1.841	8.257	3.181
10 000	9.965	26.03	16.671
15 000	21.329	63.249	27.735
20 000	38.175	120.097	61.162

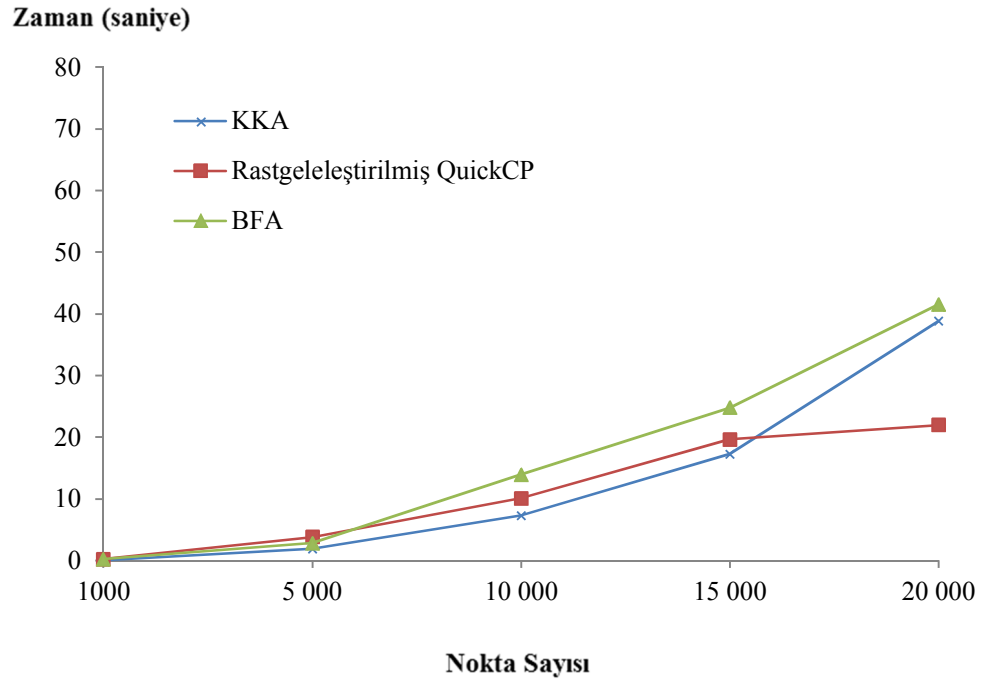


Şekil 3.59 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.59'da nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir. Burada çalışma süresi anlamında en kötü performansı QuickCP algoritması göstermiştir.

Tablo 3.51 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.59)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.070	1.358	0.211
5 000	1.794	38.455	3.393
10 000	6.892	140.451	15.376
15 000	16.331	346.824	25.815
20 000	28.863	517.264	48.831



Şekil 3.60 $\mathbb{R}^{12}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.60'ta çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Üç algoritma hemen hemen benzer sürede çalışmıştır.

Tablo 3.52 $\mathbb{R}^{12}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.60)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.078	0.213	0.215
5 000	1.903	3.789	2.831
10 000	7.283	10.096	13.930
15 000	17.263	19.620	24.774
20 000	38.865	21.963	41.524

3.4.6 $\mathbb{R}^{17}$ 'de algoritmaların çalışma süresi sonuçları

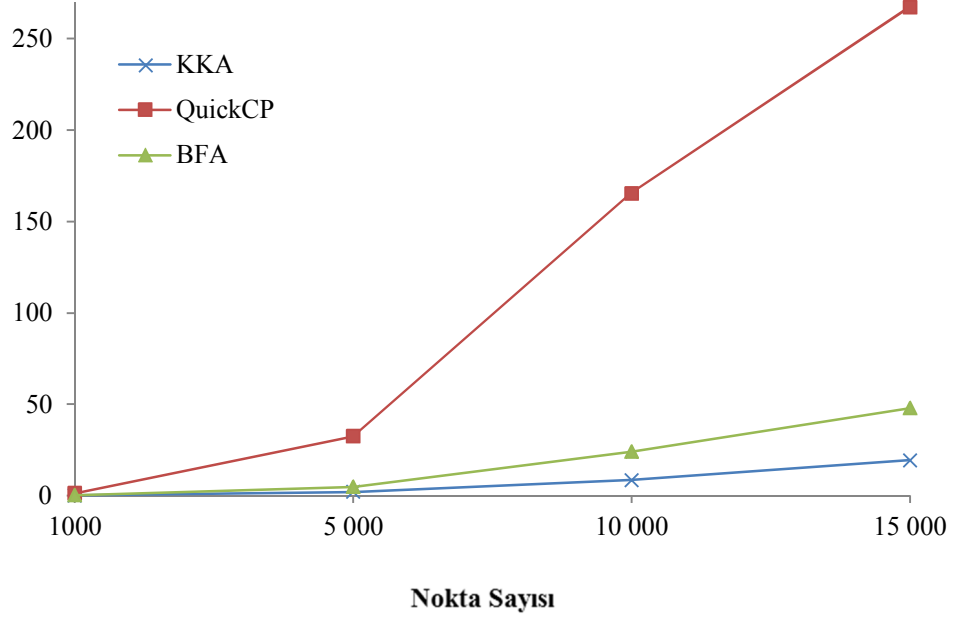
$\mathbb{R}^{17}$ uzayında üstel, normal ve tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP algoritması çalıştırılmıştır. Çalışma sonuçları resmedilmiştir. Farklı dağılıma sahip nokta kümelerinde algoritmaların çalışma davranışları da farklı olmuştur. Algoritmanın çalışma sonuçlarına dair yorumlar yaklaşık ortalama değerlerdir:

Çizgisel yığılım içermeyen nokta kümelerinde en iyi performansı KKA göstermiştir, en kötü performansı ise QuickCP göstermiştir. Rastgeleleştirmenin QuickCP algoritmasına çizgisel yığılım içermeyen kümelerde performans anlamında çok fazla etkisi olmamıştır. Üstel dağılımlı kümelerde KKA QuickCP'nin 1/15 katı, BFA 1/6 katı, rastgeleleştirilmiş QuickCP ise 1/4 katı süre harcamıştır. Normal dağılımlı kümelerde KKA QuickCP'nin 1/8 katı, BFA 1/3 katı, rastgeleleştirilmiş QuickCP ise 1.2 katı süre harcamıştır. Tekdüze dağılımlı kümelerde KKA QuickCP'nin 1/6 katı, BFA 1/3 katı, rastgeleleştirilmiş QuickCP ise 0.9 katı süre harcamıştır.

Çizgisel birikim içeren kümelerde rastgeleştirme işlemi QuickCP'nin performansını olumlu etkilemiştir. Ancak yine de KKA'yı geçmesine yetmemiştir. Üstel dağılımlı kümelerde QuickCP KKA'nın 25 katı, BFA 1.5 katı, rastgeleleştirilmiş QuickCP ise 1.2 katı süre harcamıştır. Normal dağılımlı kümelerde QuickCP KKA'nın 30 katı, BFA 1.5 katı, rastgeleleştirilmiş QuickCP ise 3.5 katı süre harcamıştır. Tekdüze dağılımlı kümelerde QuickCP KKA'nın 28 katı, BFA 2 katı, rastgeleleştirilmiş QuickCP ise 3 katı süre harcamıştır. Tezin sonuç bölümünde bu değerlerin analizi bulunmaktadır.

3.4.6.1 $\mathbb{R}^{17}$ 'de üstel dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Zaman (saniye)

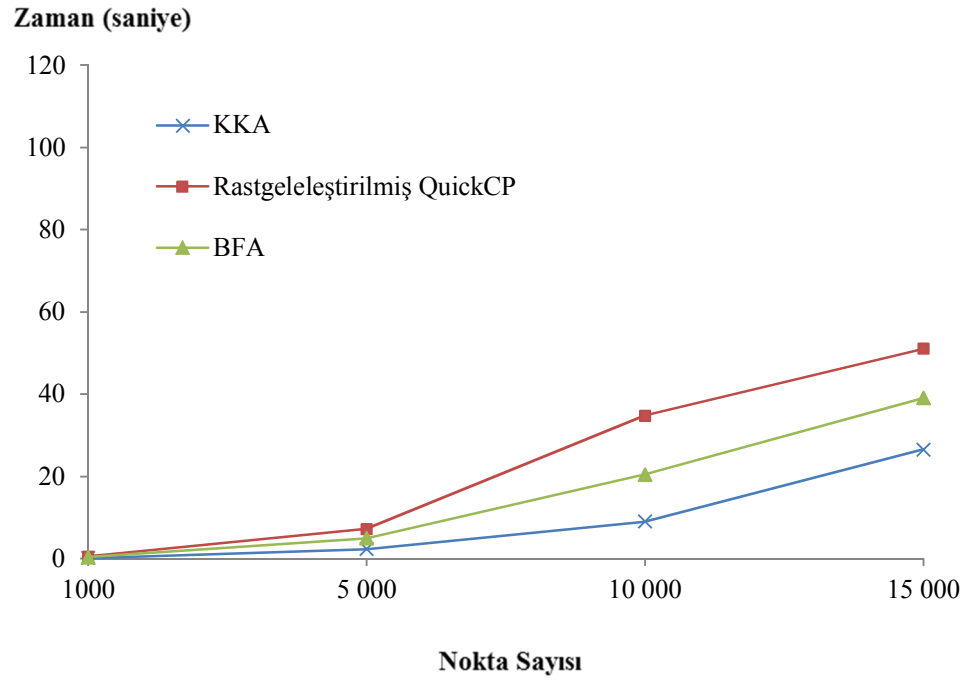


Şekil 3.61 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.61'de KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.53 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.61)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.088	1.308	0.242
5 000	2.101	32.457	4.775
10 000	8.547	165.396	24.139
15 000	19.423	267.358	47.955

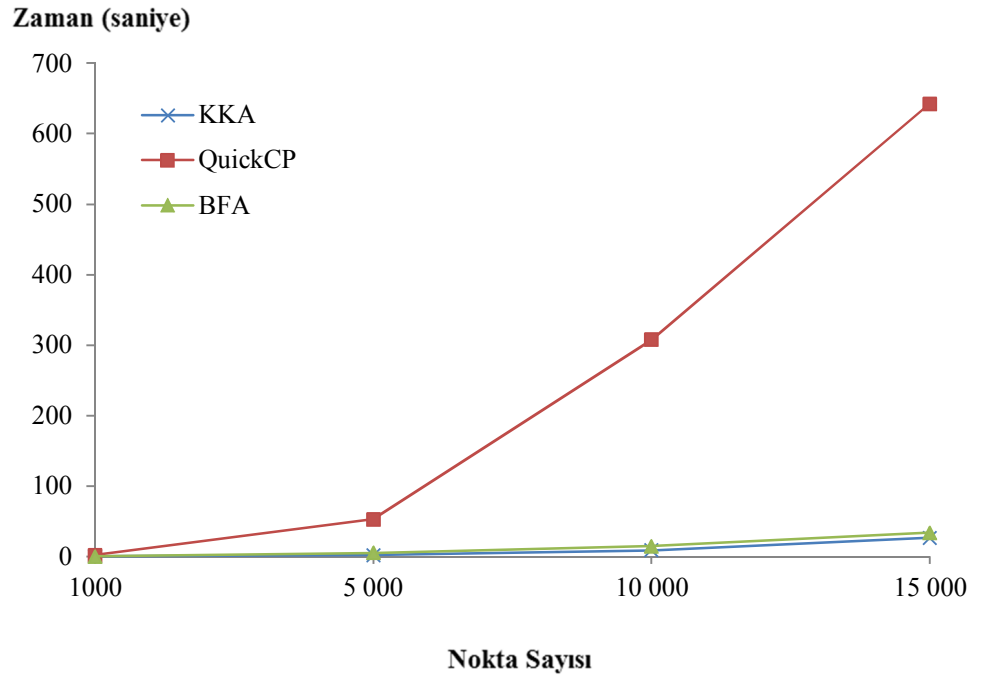


Şekil 3.62 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.62'de KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performansı KKA göstermiştir.

Tablo 3.54 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.62)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.091	0.479	0.334
5000	2.298	7.217	4.914
10000	8.956	34.784	20.400
15000	26.527	51.061	39.017

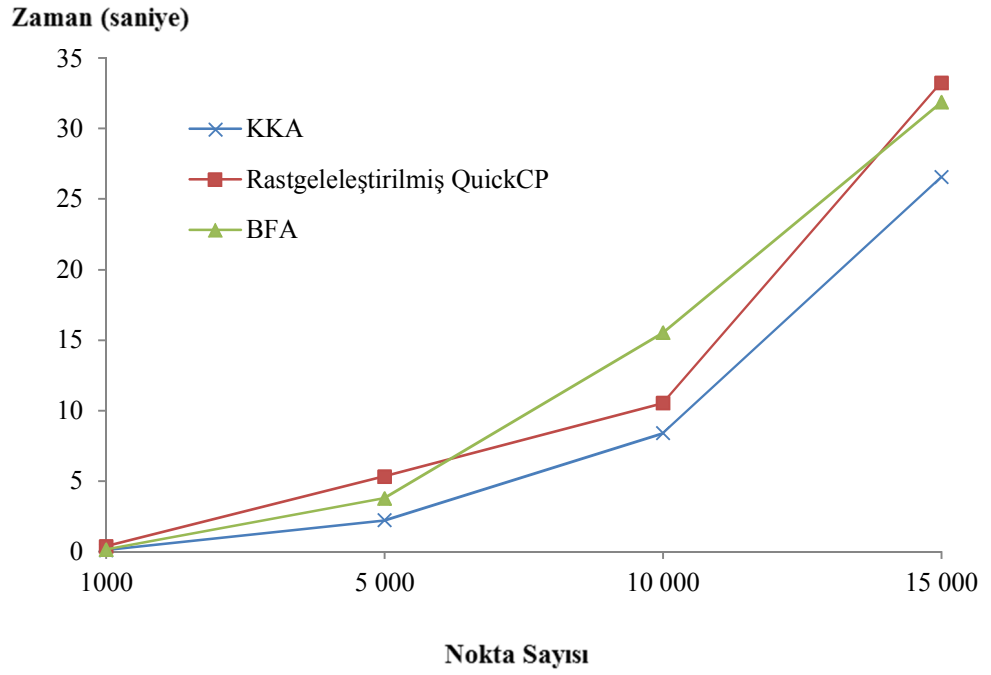


Şekil 3.63 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.63'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.55 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.63)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.0880	2.087	0.150
5 000	2.1532	52.863	5.102
10 000	8.7898	307.697	14.908
15 000	26.540	642.654	33.725



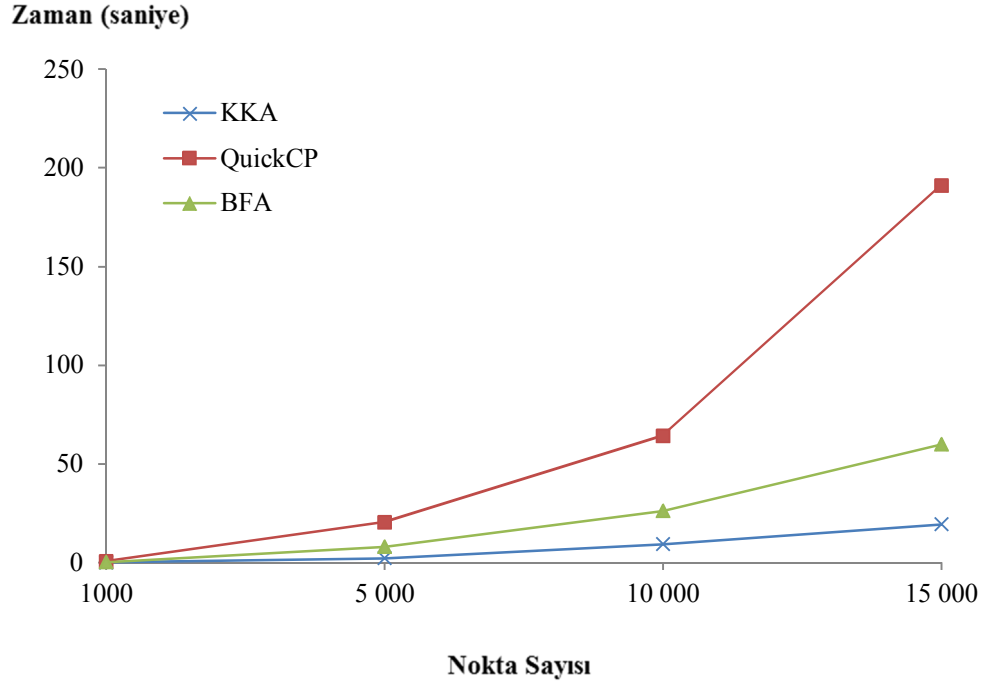
Şekil 3.64 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren üstel dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.64'te çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. En iyi performans KKA'ya aittir. Rastgeleştirme işlemi QuickCP'nin KKA'yı geçmesine yetmemiştir.

Tablo 3.56 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.64)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.093	0.377	0.153
5 000	2.216	5.329	3.785
10 000	8.382	10.531	15.508
15 000	26.548	33.263	31.876

3.4.6.2 $\mathbb{R}^{17}$ 'de normal dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

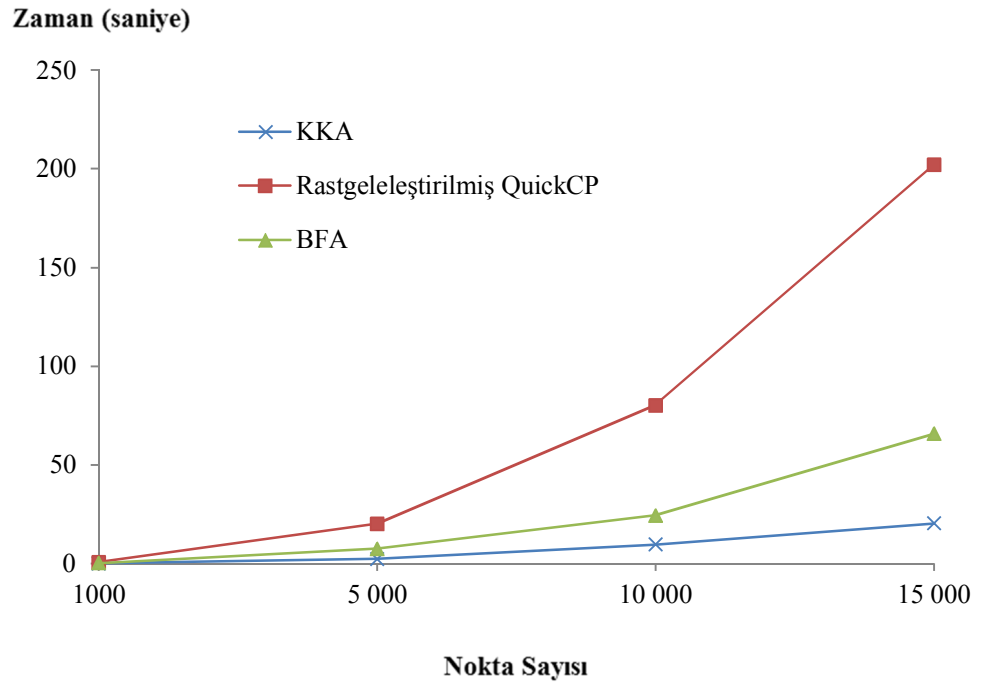


Şekil 3.65 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.65'te KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.57 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.65)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.100	0.780	0.276
5 000	2.208	20.431	8.090
10 000	9.329	64.333	26.223
15 000	19.339	191.197	59.970

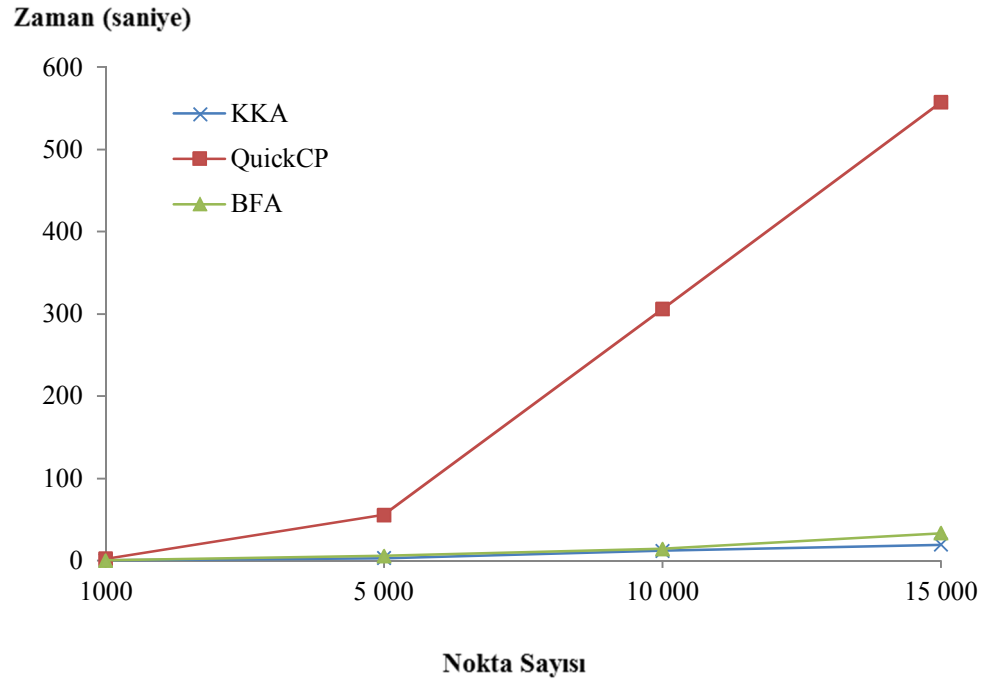


Şekil 3.66 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.66'da KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en iyi performansı KKA göstermiştir. En kötü performans ise rastgeleleştirilmiş QuickCP'ye aittir.

Tablo 3.58 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.66)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.093	0.772	0.303
5 000	2.293	20.249	7.448
10 000	9.582	80.184	24.348
15 000	20.364	201.959	65.704

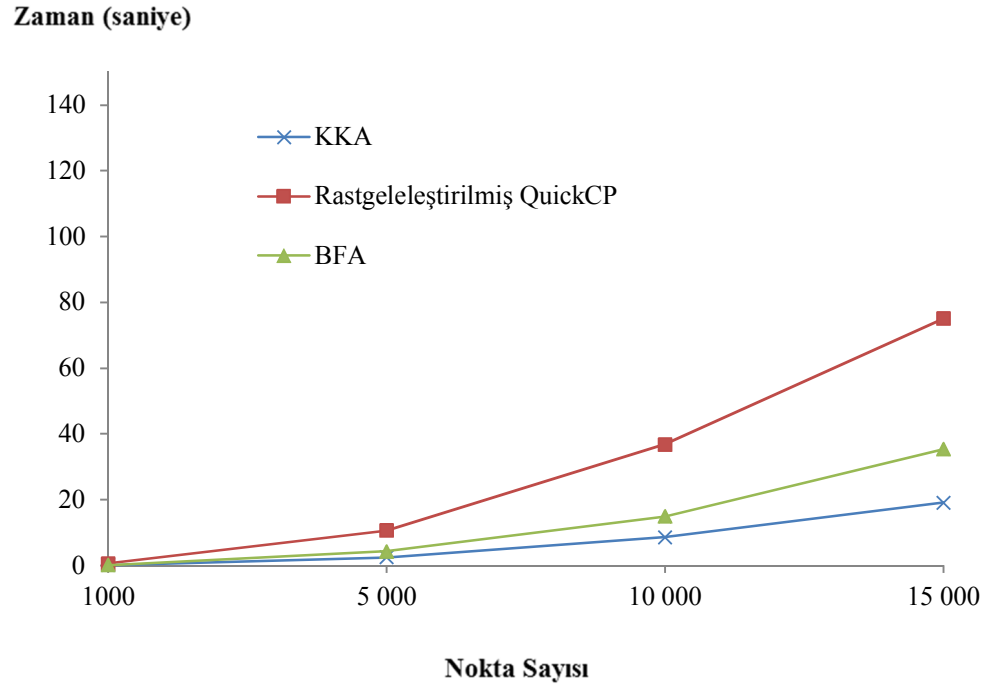


Şekil 3.67 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.67'te nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.59 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.67)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.118	2.262	0.281
5 000	2.926	55.289	5.576
10 000	11.667	305.580	14.353
15 000	19.051	557.268	33.403



Şekil 3.68 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren normal dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP

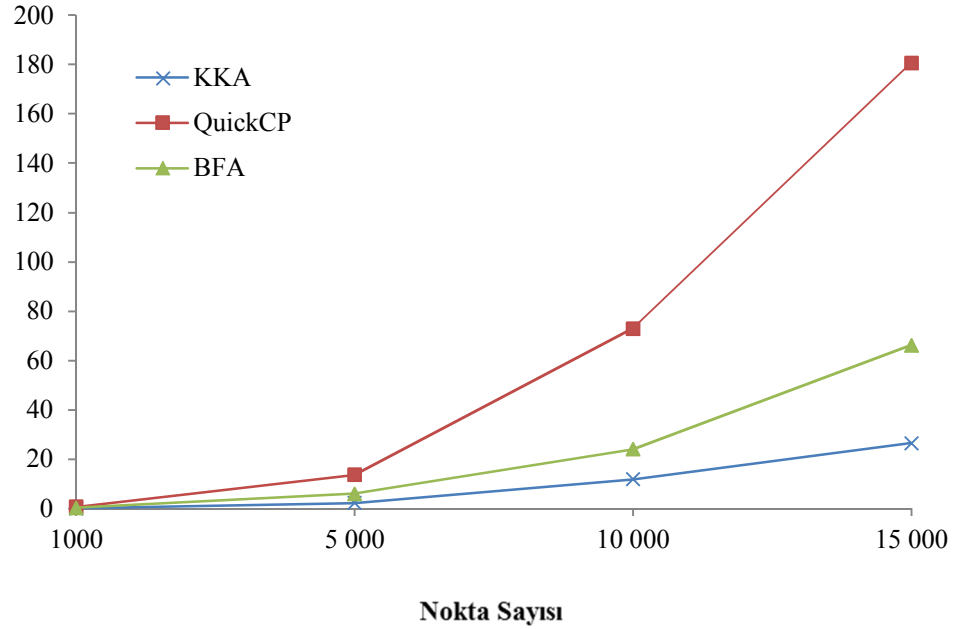
Şekil 3.68'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Rastgeleştirme işlemi QuickCP'nin, BFA'nın ve KKA'nın önüne geçmesine yetmemiştir.

Tablo 3.60 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.68)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.098	0.667	0.182
5 000	2.467	10.676	4.302
10 000	8.555	36.866	14.929
15 000	19.141	75.061	35.345

3.4.6.3 $\mathbb{R}^{17}$ 'de tekdüze dağılımlı kümelerde algoritmaların çalışma süresi sonuçları

Zaman (saniye)

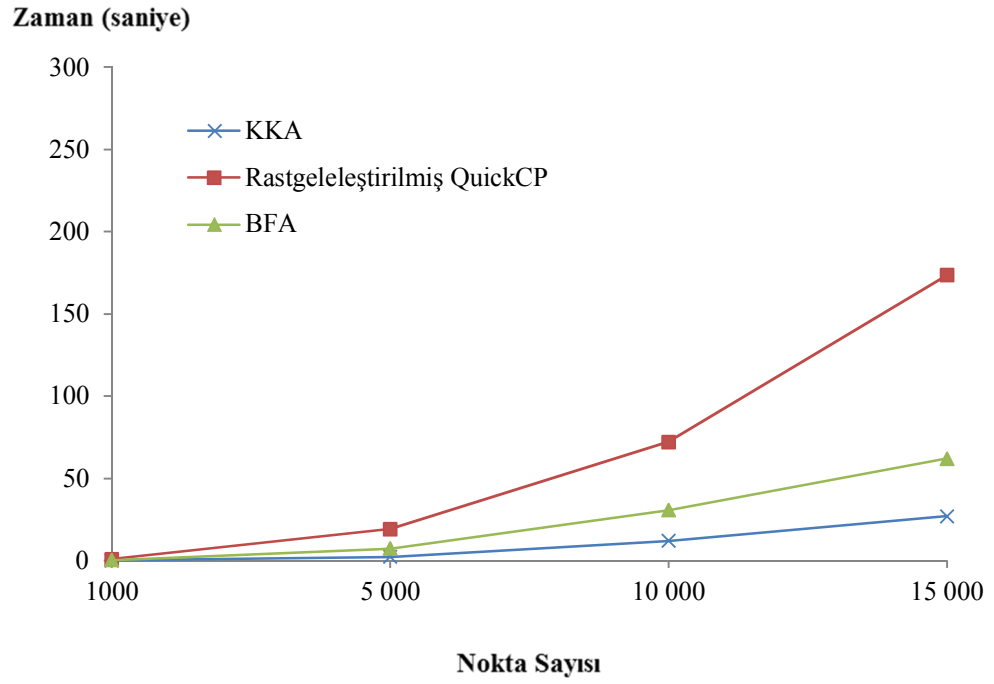


Şekil 3.69 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.69'da KKA, QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performans ise KKA'ya aittir.

Tablo 3.61 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.69)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.098	0.727	0.276
5 000	2.214	13.739	5.978
10 000	11.869	72.978	24.040
15 000	26.508	180.643	66.284

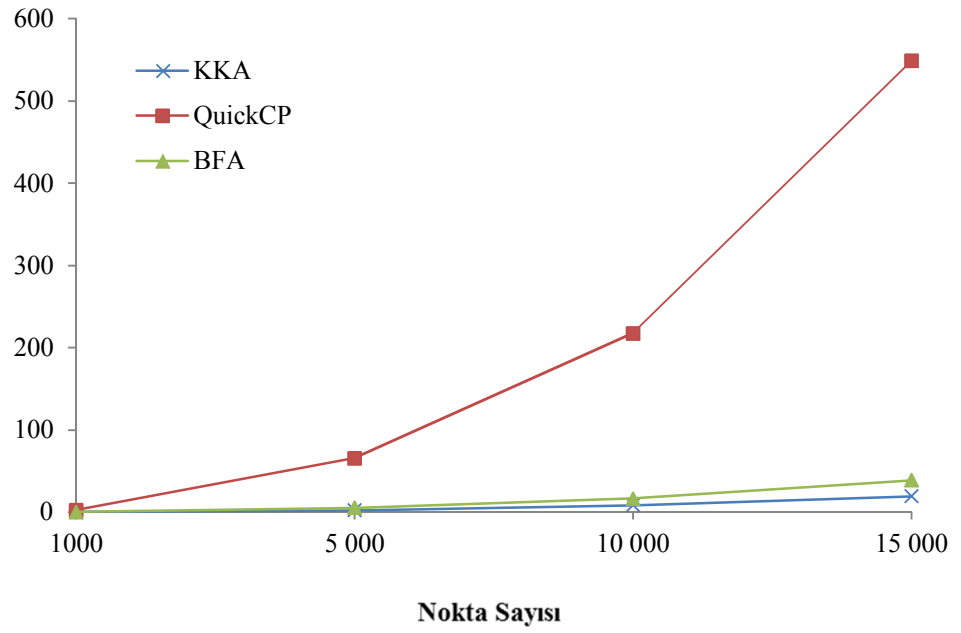


Şekil 3.70 $\mathbb{R}^{17}$ 'de çizgisel birikim içermeyen tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.70'te KKA, rastgeleleştirilmiş QuickCP algoritması ve BFA arasında çalışma süresi anlamında en kötü performansı QuickCP göstermiştir. En iyi performansı KKA göstermiştir. Rastgeleleştirme işlemi QuickCP'nin BFA'nın ve KKA'nın önüne geçmesine yetmemiştir.

Tablo 3.62 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.70)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.098	0.911	0.304
5 000	2.214	19.193	7.221
10 000	11.916	72.171	30.696
15 000	26.913	173.510	61.941

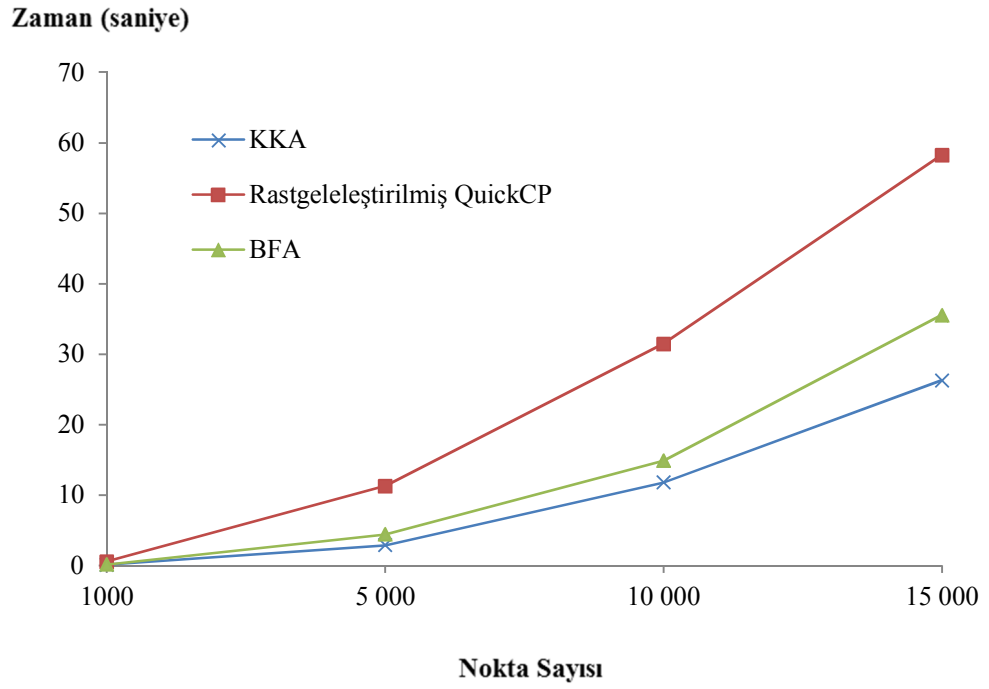
Zaman (saniye)

Şekil 3.71 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve QuickCP'nin çalışma süreleri

Şekil 3.71'de nokta kümeleri çizgisel birikim maksimum olacak şekilde hazırlanmıştır. Burada amaç QuickCP'nin en kötü durumla karşılaştığında göstereceği performansı gözlemlemektir.

Tablo 3.63 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.71)

Nokta Sayısı	KKA(sn)	QuickCP(sn)	BFA(sn)
1000	0.092	2.454	0.216
5 000	2.176	65.594	5.343
10 000	8.497	217.317	16.535
15 000	19.443	548.685	38.901



Şekil 3.72 $\mathbb{R}^{17}$ 'de çizgisel birikim içeren tekdüze dağılımlı nokta kümelerinde KKA, BFA ve rastgeleleştirilmiş QuickCP'nin çalışma süreleri

Şekil 3.72'de çizgisel birikim maksimum olacak şekilde hazırlanan nokta kümelerinde QuickCP algoritması rastgeleleştirilmiştir. Rastgeleleştirilmiş QuickCP, KKA'nın ve BFA'nın gerisinde kalmıştır.

Tablo 3.64 $\mathbb{R}^{17}$ 'de Nokta sayısı - Zaman (saniye) (Bkz Şekil 3.72)

Nokta Sayısı	KKA(sn)	R. QuickCP(sn)	BFA(sn)
1000	0.114	0.569	0.186
5 000	2.900	11.313	4.454
10 000	11.791	31.443	14.864
15 000	26.265	58.243	35.551

4. SONUÇLAR

4.1 Algoritma Çalıştırma Süresi Sonuçları ve Analizleri

Üstel, normal ve tekdüze dağılımlı farklı boyutlu uzaylardaki nokta kümelerinde incelenen KKA, BFA, QuickCP ve rastgeleştirilmiş QuickCP algoritmalarına dair test sonuçları listelenmiştir. Örneğin Tablo 4.1’de $\mathbb{R}^2$ uzayında “üstel dağılım- çizgisel yığılım olduğunda” hücrelerinde KKA:1, BFA:1/15, QuickCP: 13, R.QuickCP: 1/120 değerleri bulunmaktadır. Bu şu demektir: KKA’nın en yakın nokta çiftini bulmak için ortalama yaklaşık 1 birim zaman çalıştığı düşünülürse BFA 1/15 birim zamanda aynı nokta çiftini bulmuştur. QuickCP 13 birim zamanda, R.QuickCP yani rastgeleştirilmiş QuickCP 1/120 birim zamanda en yakın çifti bulmuştur. Bu durumda en hızlı çalışan algoritma R. QuickCP’dir. Değerler yaklaşık ortalama değerlerdir.

Tablo 4.1 $\mathbb{R}^2$ uzayında sonuçlar

$\mathbb{R}^2$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 200 BFA: 10 QuickCP: 1	KKA: 200 BFA: 5 QuickCP: 1	KKA: 150 BFA: 5 QuickCP: 1
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1/15 QuickCP: 13 R. QuickCP: 1/120	KKA: 1 BFA: 1/15 QuickCP: 11 R. QuickCP: 1/60	KKA: 1 BFA: 1/20 QuickCP: 9 R. QuickCP: 1/120

Tablo 4.2 $\mathbb{R}^2$ ’de sonuçların analizi

$\mathbb{R}^2$ ’de Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^2$ ’de Çizgisel Birikim İçeren Kümelerde
- En kötü performans daima KKA’ya aittir. - Nokta dağılımı fark etmeksizin en iyi performans QuickCP’ye aittir. BFA, performans anlamında QuickCP’den sonra gelmektedir.	- En kötü performans daima QuickCP’ye aittir. - En iyi performans daima rastgeleştirilmiş QuickCP’ye aittir.

Tablo 4.3 $\mathbb{R}^3$ uzayında sonuçlar

$\mathbb{R}^3$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 40 BFA: 2 QuickCP: 1	KKA: 100 BFA: 4.5 QuickCP: 1	KKA: 100 BFA: 7 QuickCP: 1
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.2 QuickCP: 3.5 R. QuickCP: 1/500	KKA: 1 BFA: 0.5 QuickCP: 3 R. QuickCP: 1/400	KKA: 1 BFA: 0.7 QuickCP: 2.5 R. QuickCP: 1/200

Tablo 4.4 $\mathbb{R}^3$ 'te sonuçların analizi

$\mathbb{R}^3$ 'te Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^3$ 'te Çizgisel Birikim İçeren Kümelerde
- En kötü performans daima KKA'ya aittir. - QuickCP'nin davranışları nokta kümesindeki dağılıma göre değişmektedir. QuickCP en iyi performansını normal ve tekdüze dağılımlı kümelerde göstermekle birlikte üstel dağılımlı kümeler de dâhil, bu boyutta en iyi çalışan algoritma olmuştur.	- En kötü performans daima QuickCP'ye aittir. - En iyi performans daima rastgeleleştirilmiş QuickCP'ye aittir.

Tablo 4.5 $\mathbb{R}^4$ uzayında sonuçlar

$\mathbb{R}^4$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 25 BFA: 1.5 QuickCP: 1	KKA: 30 BFA: 3 QuickCP: 1	KKA: 30 BFA: 2.5 QuickCP: 1
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.5 QuickCP: 5 R. QuickCP: 1/250	KKA: 1 BFA: 0.7 QuickCP: 4 R. QuickCP: 1/100	KKA: 1 BFA: 1.6 QuickCP: 4 R. QuickCP: 1/110

Tablo 4.6 $\mathbb{R}^4$ 'te sonuçların analizi

$\mathbb{R}^4$ 'te Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^4$ 'te Çizgisel Birikim İçeren Kümelerde
- En kötü performans daima KKA'ya aittir. - QuickCP'nin davranışları nokta kümesindeki dağılıma göre değişmektedir. QuickCP en iyi performansını normal ve tekdüze dağılımlı kümelerde göstermekle birlikte üstel dağılımlı kümeler de dâhil, bu boyutta en iyi çalışan algoritma olmuştur. - Algoritmalar $\mathbb{R}^3$ uzayındaki davranışlarının aynısını burada da göstermiştir. Yalnızca boyut artışından dolayı çalışma süreleri artmıştır.	- En kötü performans daima QuickCP'ye aittir. - En iyi performans daima rastgeleleştirilmiş QuickCP'ye aittir. - Algoritmalar $\mathbb{R}^3$ uzayındaki davranışlarının aynısını burada da göstermiştir. Yalnızca boyut artışından dolayı çalışma süreleri artmıştır.

Tablo 4.7 $\mathbb{R}^5$ uzayında sonuçlar

$\mathbb{R}^5$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 3.5 BFA: 1 QuickCP: 1 R. QuickCP: 0.5	KKA: 5 BFA: 1.5 QuickCP: 1	KKA: 15 BFA: 2 QuickCP: 1
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.2 QuickCP: 6 R. QuickCP: 1/150	KKA: 1 BFA: 5 QuickCP: 0.8 R. QuickCP: 1/150	KKA: 1 BFA: 5 QuickCP: 0.6 R. QuickCP: 1/150

Tablo 4.8 $\mathbb{R}^5$ 'te sonuçların analizi

$\mathbb{R}^5$ 'te Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^5$ 'te Çizgisel Birikim İçeren Kümelerde
- QuickCP normal ve tekdüze dağılımlı kümelerde en iyi çalışan algoritmadır. Bu dağılıma sahip kümelerde QuickCP rastgeleştirilirse QuickCP'nin BFA ve KKA ile arasındaki fark açılacaktır. - Üstel dağılımlı kümelerde BFA ve QuickCP benzer sürede çalışmıştır. Daha sonra QuickCP rastgeleştirilmiştir. Bu işlem sonucunda QuickCP, BFA'yı geçerek en iyi algoritma olmuştur.	- En kötü performans daima QuickCP'ye aittir. - En iyi performans daima rastgeleştirilmiş QuickCP'ye aittir.

Tablo 4.9 $\mathbb{R}^8$ uzayında sonuçlar

$\mathbb{R}^8$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 0.7 BFA: 0.5 QuickCP: 1 R. QuickCP: 0.3	KKA: 1.1 BFA: 0.7 QuickCP: 1 R. QuickCP: 0.9	KKA: 1.5 BFA: 1.1 QuickCP: 1 R. QuickCP: 1.4
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.2 QuickCP: 1 R. QuickCP: 1/20	KKA: 1 BFA: 0.8 QuickCP: 8 R. QuickCP: 1/7	KKA: 1 BFA: 1.5 QuickCP: 12 R. QuickCP: 1/7

Tablo 4.10 $\mathbb{R}^8$ 'te sonuçların analizi

$\mathbb{R}^8$ 'de Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^8$ 'de Çizgisel Birikim İçeren Kümelerde
- Üstel dağılımlı çizgisel birikim içermeyen kümelerde BFA'nın gerisinde kalan QuickCP, rastgeleleştirme işleminden sonra BFA'yı geçerek en iyi algoritma olmuştur. - Normal ve tekdüze dağılımlı çizgisel birikim içermeyen kümelerde BFA ile QuickCP benzer sürede çalışmıştır, az farkla en iyi çalışan algoritma olmuştur.	- En kötü performans daima QuickCP'ye aittir. - En iyi performans rastgeleleştirilmiş QuickCP'ye aittir.

Tablo 4.11 $\mathbb{R}^{12}$ uzayında sonuçlar

$\mathbb{R}^{12}$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 1/7 BFA: 1/6 QuickCP: 1 R. QuickCP: 1/6	KKA: 1/4 BFA: 1/2 QuickCP: 1 R. QuickCP: 1.2	KKA: 1/2 BFA: 0.6 QuickCP: 1 R. QuickCP: 1.5
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.5 QuickCP: 20 R. QuickCP: 1/2	KKA: 1 BFA: 1.1 QuickCP: 15 R. QuickCP: 1.5	KKA: 1 BFA: 2 QuickCP: 20 R. QuickCP: 1.25

Tablo 4.12 $\mathbb{R}^{12}$ 'de sonuçların analizi

$\mathbb{R}^{12}$ 'de Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^{12}$ 'de Çizgisel Birikim İçeren Kümelerde
- Çizgisel yığılım içermeyen üstel, normal ve tekdüze dağılımlı nokta kümelerinde en iyi performansı KKA göstermiştir. - Rastgeleleştirme işlemi QuickCP algoritmasını performans anlamında üstel dağılımlı kümelerde iyi etkilemiştir. Ancak normal ve tekdüze dağılımlı kümelerde olumsuz etkilemiştir.	Üstel, normal ve tekdüze dağılımlı nokta kümelerinde çizgisel yığılım maksimum durumda iken yapılan rastgeleleştirme işlemi QuickCP'nin performansına olumlu etki etmiştir, ancak yine üstel dağılımlı kümeler hariç en iyi algoritma olmasına yetmemiştir.

Tablo 4.13 $\mathbb{R}^{17}$ uzayında sonuçlar

$\mathbb{R}^{17}$	Üstel Dağılım	Normal Dağılım	Tekdüze Dağılım
Çizgisel Yığılım Olmadan	KKA: 1/15 BFA: 1//6 QuickCP: 1 R. QuickCP: 1/4	KKA: 1/8 BFA: 1/3 QuickCP: 1 R. QuickCP: 1.2	KKA: 1/6 BFA: 1/3 QuickCP: 1 R. QuickCP: 0.9
Çizgisel Yığılım Olduğunda	KKA: 1 BFA: 1.5 QuickCP: 25 R. QuickCP: 1.2	KKA: 1 BFA: 1.5 QuickCP: 30 R. QuickCP: 3.5	KKA: 1 BFA: 2 QuickCP: 28 R. QuickCP: 3

Tablo 4.14 $\mathbb{R}^{17}$ 'de sonuçların analizi

$\mathbb{R}^{17}$ 'de Çizgisel Birikim İçermeyen Kümelerde	$\mathbb{R}^{17}$ 'de Çizgisel Birikim İçeren Kümelerde
Çizgisel yığılım içermeyen nokta kümelerinde en iyi performansı KKA göstermiştir, en kötü performansı ise normal dağılım hariç QuickCP göstermiştir. Normal dağılımda rastgeleştirme işlemi QuickCP'yi olumsuz etkilemiştir.	- Çizgisel birikim içeren kümelerde rastgeleştirme işlemi QuickCP'nin performansını olumlu etkilemiştir. Ancak yine de KKA'yı geçmesine yetmemiştir. - En iyi performansı KKA göstermiştir.

4.2 Algoritmaların Çalışma Süresi Sonuçları Üzerine Yorum

KKA, BFA, QuickCP ve rastgeleştirilmiş QuickCP, $\mathbb{R}^k$ ($k \geq 2$) uzayında tekdüze dağılıma ek olarak üstel ve normal dağılımlı kümelerde çalıştırılmış ve test edilmiştir.

QuickCP algoritması $\mathbb{R}^2$ uzayında çizgisel yığılım içermeyen nokta kümelerinde dağılım şekli fark etmeksizin (üstel, normal, tekdüze) en yakın nokta çiftini BFA ve KKA'dan daha hızlı bulmuştur. Herhangi bir nokta kümesinde çizgisel yığılım arttıkça QuickCP'nin performansı kötüye gitmiştir. QuickCP için yapılan rastgeleştirme işlemi QuickCP'ye eski performansını kazandırmıştır. Çizgisel yığılım içeren kümelerde dağılım şekli fark etmeksizin en iyi algoritma rastgeleştirilmiş QuickCP olmuştur.

$\mathbb{R}^k$ ($k = 3, k = 4, k = 5, k = 8$) uzayında, farklı dağılıma sahip nokta kümelerinde QuickCP algoritmasının davranışları incelenmiş ve algoritmanın daha iyi performans sağladığı dağılımlar gözlemlenmiştir. Özellikle normal ve tekdüze dağılımlı kümelerde QuickCP aynı boyutta ama üstel dağılıma sahip kümedekilerden daha iyi performans göstermiştir. Ancak, rastgeleştirme işlemi QuickCP'yi uzay boyutu fark etmeksizin üstel dağılımlı kümelerde daha olumlu etkilemiştir.

Kümelerdeki çizgisel yığılma durumu $O(n)$ karmaşıklıkta belirlenmiştir. BFA'nın, QuickCP'yi performans anlamında geçtiği durum belirlenmiştir. Bunun çözümü için QuickCP algoritması rastgeleştirilmiştir. Sonuçta QuickCP, BFA'nın gerisine düştüğü durumdan kurtulmuştur. Çizgisel yığılma sorunu çözülmüştür. Rastgeleştirme işlemi yapmanın gerekliliği histogram kullanılarak bu duruma göre seçilebilir.

$\mathbb{R}^2, \mathbb{R}^3, \mathbb{R}^4$ ve $\mathbb{R}^5$ uzayında QuickCP ve rastgeleştirilmiş QuickCP algoritması en iyi performans gösteren algoritmalar olmuştur. Daha yüksek boyutlu uzaylarda durum değişmektedir.

$\mathbb{R}^8$ uzayındaki normal ve tekdüze dağılımlı çizgisel birikim içermeyen kümelerde BFA, KKA ve rastgeleştirilmiş QuickCP benzer sürede çalışmıştır. Bu boyutta üstel dağılımlı çizgisel birikim içermeyen kümelerde rastgeleştirilmiş QuickCP en iyi çalışan algoritma olmuştur.

$\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ uzaylarında KKA, BFA ve QuickCP'yi performans anlamında geçmiştir. Çizgisel yığılma içeren kümelerde rastgeleştirme işlemi QuickCP'yi olumlu etkilemiştir. Ancak bu olumlu etkileme $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ uzaylarında QuickCP'nin KKA'yı geçmesine yetmemiştir.

$\mathbb{R}^2, \mathbb{R}^3, \mathbb{R}^4$ ve $\mathbb{R}^5$ uzayında çizgisel birikim içeren ve içermeyen kümelerde çalışma süresi anlamında en iyi performans QuickCP ve rastgeleştirilmiş QuickCP algoritmasına aittir. En yakın çift problemi için çözüm gereken koşullarda, nokta kümesi bu uzaylarda tanımlanmışsa, problemi içeren küme histogramla incelenebilir, çizgisel yığılma durumuna göre QuickCP ya da rastgeleştirilmiş QuickCP algoritması tercih edilebilir.

Uzay boyutunun artması BFA'yı, QuickCP'yi ve rastgeleştirilmiş QuickCP'yi performans anlamında kötü etkilemektedir. Bu boyutlarda KKA'nın

performansı BFA, QuickCP ve rastgeleleştirilmiş QuickCP'ye göre daha iyi olmuştur.

BFA da çizgisel yığılma ve uzay boyutu arttıkça kötü performans sergilemiştir. Çizgisel yığılma karşı, BFA'nın kullanacağı noktalar da farklı koordinat sistemlerinde tanımlanabilir.

4.3 Tezin Katkıları

Bu tezin bilimsel yazına iki temel katkısı bulunmaktadır.

Bunların ilki en yakın çifti bulma problemi için yeni geliştirilmiş bir algoritma olan QuickCP'nin rastgeleleştirilmiş sürümünün tasarlanıp gerçekleştirilmesidir. Rastgeleleştirme işlemi için Gram-Schmidt yöntemi ve koordinat sistemleri arasında dönüşüm kullanılmıştır. Rastgeleleştirilmiş QuickCP adı verilen bu algoritma, QuickCP'nin performansını çalışma süresi anlamında kötü etkileyen noktaların çizgisel olarak yığılması durumunu içeren dağılımlarda QuickCP'nin performansını çok büyük ölçüde iyileştirmektedir.

İkinci temel katkı olarak, QuickCP algoritmasının ve rastgeleleştirilmiş QuickCP algoritmasının çok boyutlu uzaylarda çalışılacak şekilde genişletilmesi ve uyarlanmasıdır. QuickCP ve rastgeleleştirilmiş QuickCP algoritmaları $\mathbb{R}^2$, $\mathbb{R}^3$, $\mathbb{R}^4$, $\mathbb{R}^5$, $\mathbb{R}^8$, $\mathbb{R}^{12}$ ve $\mathbb{R}^{17}$ uzaylarında üstel, normal ve tekdüze dağılımlı kümelerde test edilmiştir.

KAYNAKLAR DİZİNİ

- Aspnes, J.** 2013, “Notes on Randomized Algorithms”, <http://www.cs.yale.edu/homes/aspnes/classes/469/notes.pdf> (Erişim tarihi: 9 Temmuz 2013)
- Bentley, J.L. and Shamos, M.I.**, 1976, Divide-and-conquer in multidimensional space, STOC '76: Proceedings of the eighth annual ACM symposium on Theory of Computing, New York, 10 p.
- Boleman, M.**, “Histogram” <http://www.southalabama.edu/physics/lectures/boleman/histogram.html> (Erişim tarihi: 9 Temmuz 2013)
- Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C.**, 2009, Introduction To Algorithms (3. Baskı), The MIT Press, London, 1292p.
- Dalkilic, M.E and Ergun, S.**, 2012, A simple yet fast algorithm for the closest-pair problem using sorted projections on multi-dimensions, Ege University, 12p (unpublished).
- Harvey Mudd College**, “The Gram-Schmidt Algorithm”, <http://www.math.hmc.edu/calculus/tutorials/gramschmidt/gramschmidt.pdf> (Erişim tarihi: 9 Temmuz 2013)
- Joy, K.I.**, 1999, “Coordinate Conversion Between The Cartesian Frame And An Arbitrary Frame”, <http://www.idav.ucdavis.edu/education/GraphicsNotes/Cartesian-Frame-to-Frame.pdf> (Erişim tarihi: 9 Temmuz 2013)
- Karp, R.M.**, 1991, An introduction to Randomized algorithms, Discrete Applied Mathematics, 34p.
- Kuttler, K.**, 2012, Linear Algebra, Theory And Applications (1. Baskı), The Saylor Foundation, USA, 913p.
- Lautrup, B.**, 2011, Physics of Continuous Matter (2. Baskı), CRC Press, ABD, 696p.
- Mao, R.**, “Metric Spaces Library - Databases”, <http://www.sisap.org/library/dbs.tar.gz> (Erişim tarihi: 9 Temmuz 2013)

KAYNAKLAR DİZİNİ (devam)

- Michigan Technological University**, "Brute Force Closest Pair and Convex-Hull", <http://www.csl.mtu.edu/cs4321/www/Lectures/Lecture%206%20-%20Brute%20Force%20Closest%20Pair%20and%20Convex%20and%20Exhaustive%20Search.htm> (Erişim tarihi: 9 Temmuz 2013)
- Pereira, J.C. and Lobo, F.G.**, 2012, An optimized divide-and-conquer algorithm for the closest-pair problem in the planar case, *Journal of Computer Science and Technology*, 27(4):891-896 pp.
- Rosetta Code**, "Closest-pair problem", http://rosettacode.org/wiki/Closest-pair_problem (Erişim tarihi: 9 Temmuz 2013)
- Wayne, K.**, 2005, "Divide-and-Conquer", <http://www.cs.princeton.edu/courses/archive/spr05/cos423/lectures/05divide-and-conquer.pdf> (Erişim tarihi: 9 Temmuz 2013)
- Zaïane, O. and Pei, Y.**, "Synthetic Data Generation", <http://www.cs.ualberta.ca/~yaling/Cluster/Generator2D/DataGenerator.jnlp> (Erişim tarihi: 9 Temmuz 2013)

ÖZGEÇMİŞ

Aycan Vargün, 31 Ağustos 1985'te İzmir'de doğmuştur. Türkiye Cumhuriyeti vatandaşıdır. Lise öğretimini 1999-2003 yılları arasında İzmir Bornova Anadolu Lisesi'nde tamamlamıştır. Lisans eğitimini 2003-2008 yılları arasında Dokuz Eylül Üniversitesi Fen-Edebiyat Fakültesi İngilizce Matematik bölümünde almıştır. 2010'dan bu yana Ege Üniversitesi Uluslararası Bilgisayar Enstitüsü tezli yüksek lisans öğrencisidir.

EKLER

Ek 1 Üstel Dağılımlı Nokta Kümesi Histogramı

Ek 2 Normal Dağılımlı Nokta Kümesi Histogramı

Ek 3 Tekdüze Dağılımlı Nokta Kümesi Histogramı

Ek 4 Normal Dağılımlı Nokta Kümesi Histogramı (java ile elde edilen)

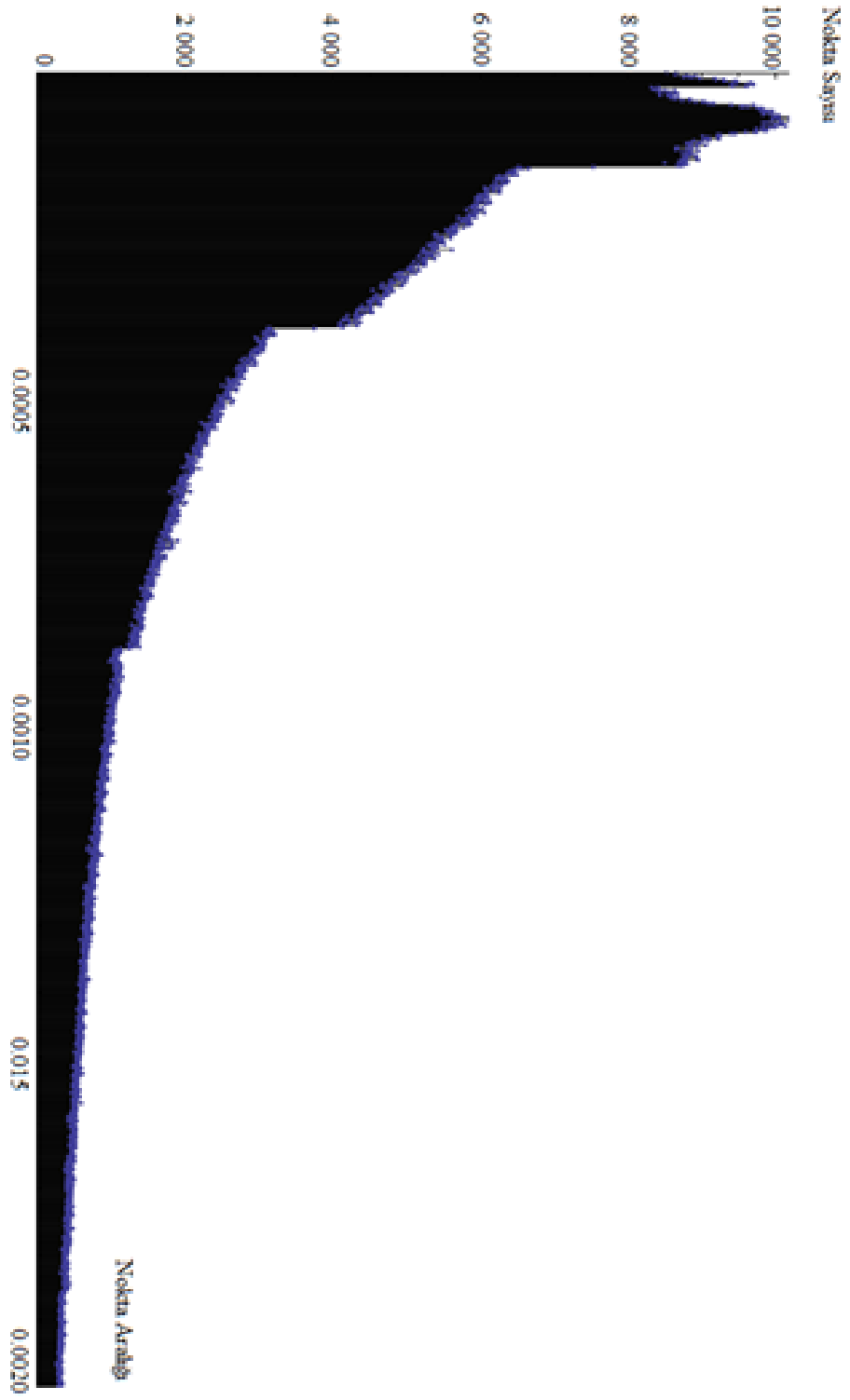
Ek 5 Rosetta Code (2013)'da BFA Kodu

Ek 6 QuickCP İçin Rastgeleleştirme Kodu

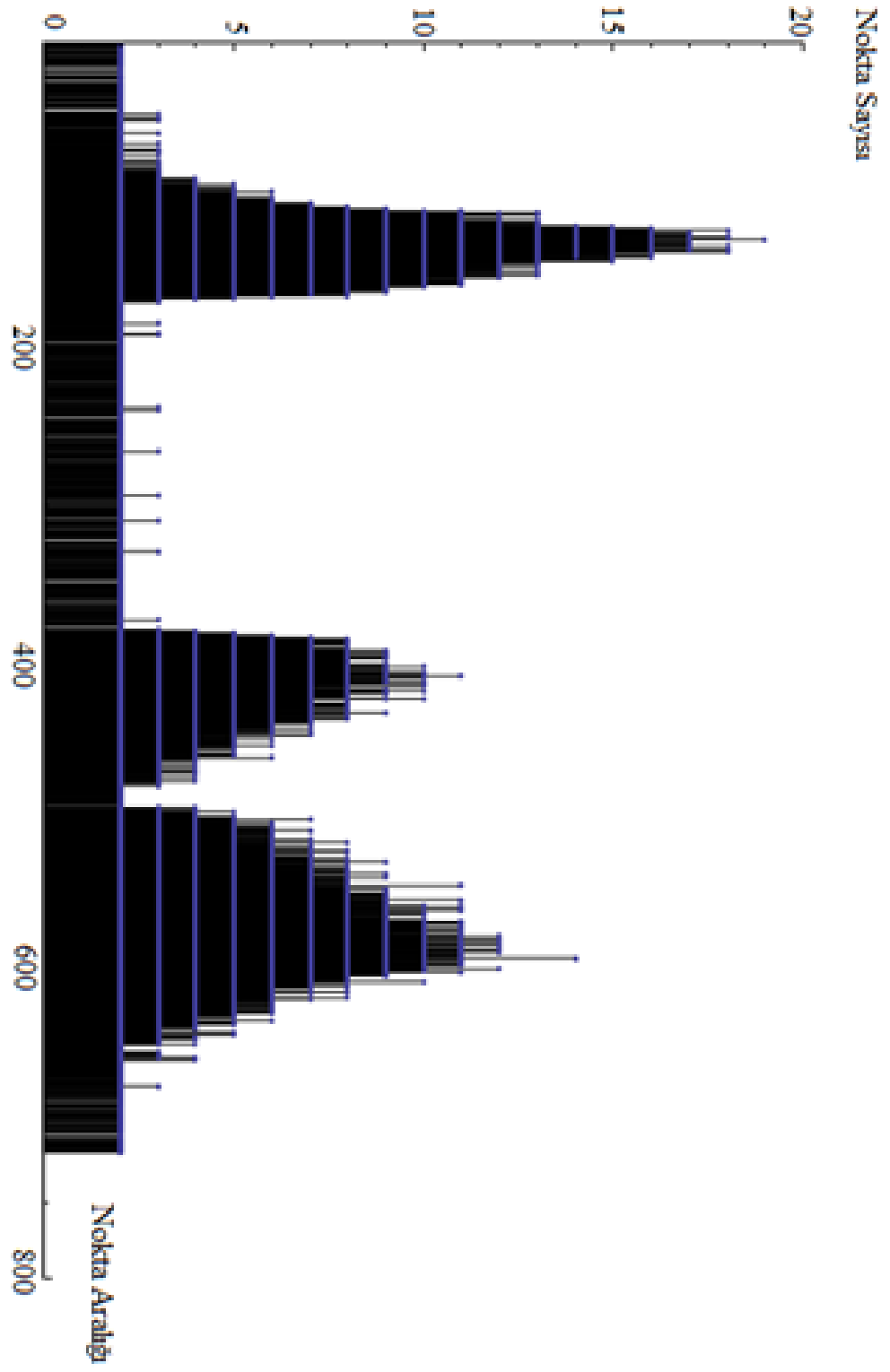
Ek 7 QuickCP'nin "k" Boyuta Uyarlanmış Kodu

Ek 8 Rosetta Code (2013)'dan "k" Boyutta Uyarlanmış BFA Kodu

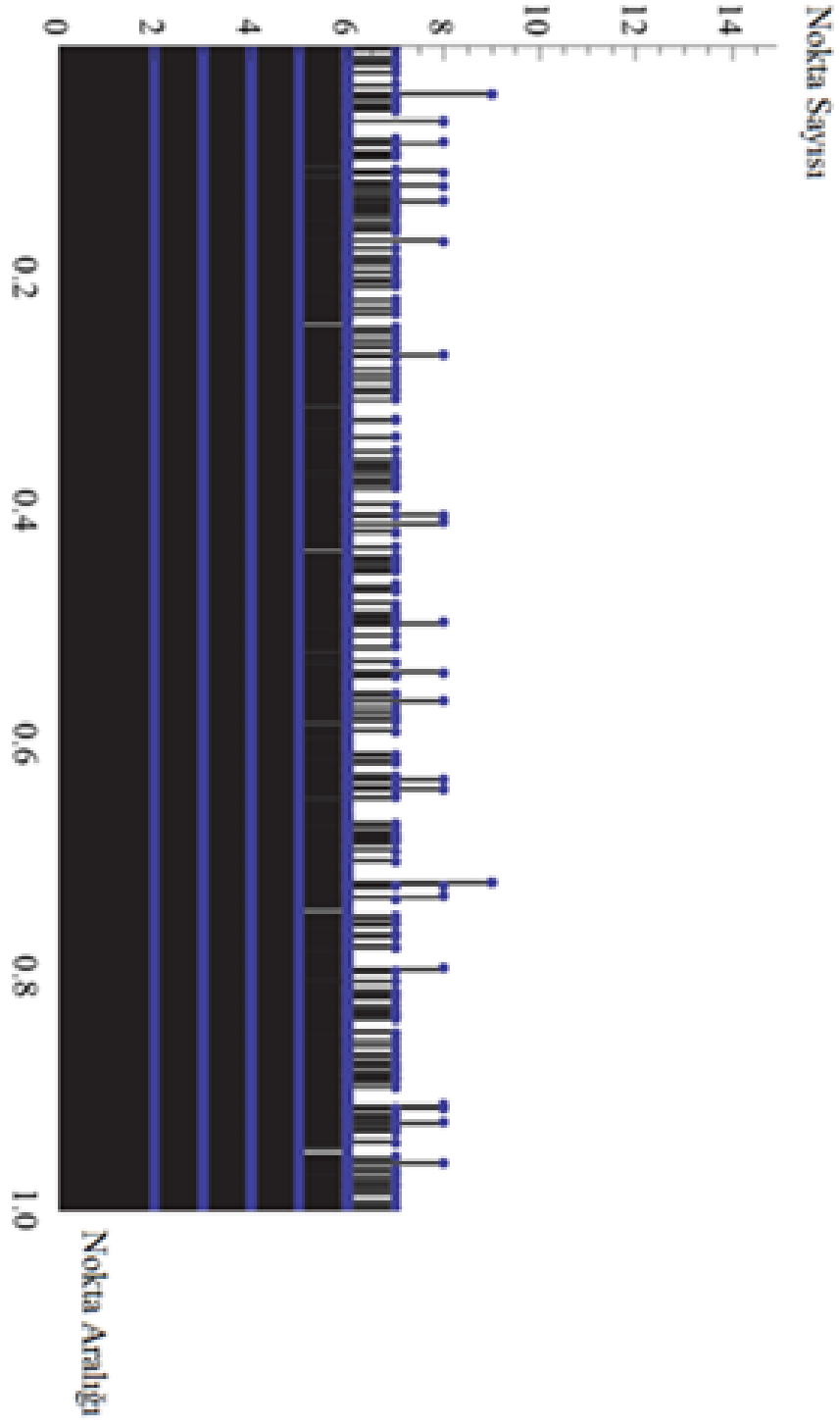
Ek 1 Üstel Dağılımlı Nokta Kümesi Histogramı



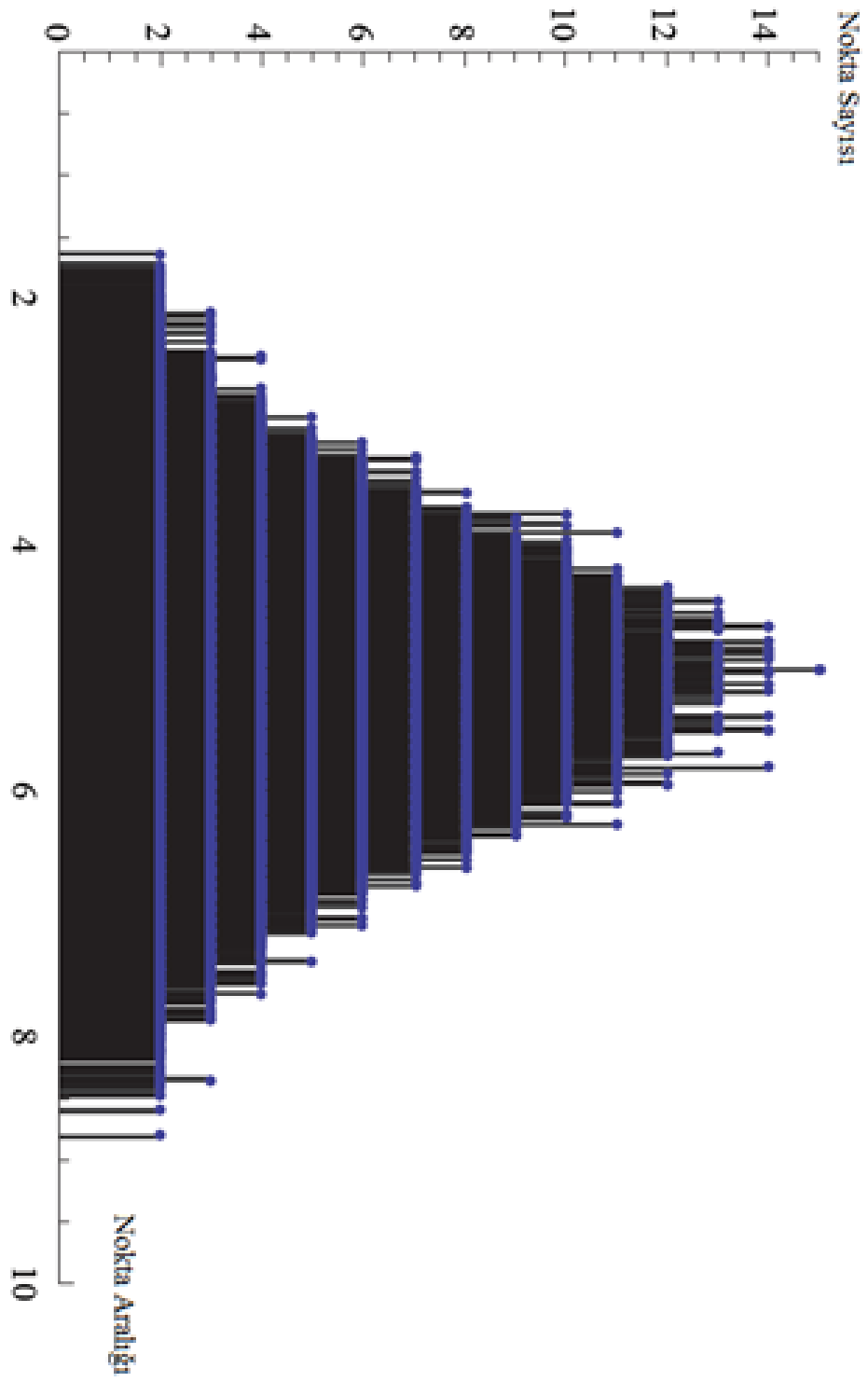
Ek 2 Normal Dağılımlı Nokta Kümesi Histogramı



Ek 3 Tekdüze Dağılımlı Nokta Kümesi Histogramı



Ek 4 Normal Dağılımlı Nokta Kümesi Histogramı (java ile elde edilen)



Ek 5 Rosetta Code (2013)'da BFA Kodu

```

closestPair of (xP, yP)
    where xP is P(1) .. P(N) sorted by x coordinate,
           and
           yP is P(1) .. P(N) sorted by y coordinate
           (ascending order)

if N ≤ 3 then
    return closest points of xP using brute-force algorithm
else
    xL ← points of xP from 1 to [N/2]
    xR ← points of xP from [N/2]+1 to N
    xm ← xP([N/2])x
    yL ← { p ∈ yP : px ≤ xm }
    yR ← { p ∈ yP : px > xm }

    (dL, pairL) ← closestPair of (xL, yL)
    (dR, pairR) ← closestPair of (xR, yR)
    (dmin, pairMin) ← (dR, pairR)

    if dL < dR then
        (dmin, pairMin) ← (dL, pairL)
    endif

    yS ← { p ∈ yP : |xm - px| < dmin }
    nS ← number of points in yS
    (closest, closestPair) ← (dmin, pairMin)

    for i from 1 to nS - 1
        k ← i + 1

        while k ≤ nS and yS(k)y - yS(i)y < dmin
            if |yS(k) - yS(i)| < closest then
                (closest, closestPair) ← (|yS(k) - yS(i)|, {yS(k),
                                                                    yS(i)})
            endif

            k ← k + 1
        endwhile
    endfor

    return closest, closestPair
endif

```

Ek 6 QuickCP İçin Rastgeleleştirme Kodu

```
/* P = {P1, P2, .... Pn} where Pi = (xi, yi) i.e.. n input
points. k is the space dimension */

for i from 1 to k
    origine[i] = 0;
end for

/* generating basis */

for i from 1 to k
    coefficients(i) = random_number;
    temp_vector(i) = 0;
    temp(i) = 0;
end for

for i from 1 to k
    for j from 1 to k
        if( i == j )
            standard_basis(i)(j) = 1;
            temp_basis(i)(j) = 0;
        else
            standard_basis(i)(j) = 0;
            temp_basis (i)(j) = 0;
        end for
    end for

for i from 1 to k
    for l from 1 to k
        for j from 1 to k
            if( i==j )
                continue;
            else
                temp_vector(l) = temp_vector(l) +
                    standard_basis(j)(l)* coefficients(j);
            end for
        end for

        for m from 1 to k
            temp_basis(i)(m) = temp_vector(m) ;
            temp_vector(m) = 0;
        end for
    end for
end for
```

Ek 6 QuickCP İçin Rastgeleleştirme Kodu (Devam)

```
for i from 1 to k
    basis(1)(i) = temp_basis(1)(i);
end for

for i from 1 to k
    for j from 1 to i
        temp = sum_of_vectors( temp, projection_vector(
            basis(j), temp_basis(i) ) );
    end for

    for m from 1 to k
        basis(i)(m) = temp_basis(i)(m) - temp(m) ;
    end for

    for l from 1 to k
        temp(l)=0;
    end for
end for

for i from 1 to k
    coefficients(i) = 0;

    for j from 1 to k
        coefficients(i) = coefficients(i) + (basis(i)(j))^2
    end for

    coefficients(i) = sqrtf(coefficients(i));
end for

for i from 1 to k
    for j from 1 to k
        basis(i)(j) = basis(i)(j) / coefficients(i);
    end for
end for
```

Ek 6 QuickCP İçin Rastgeleleştirme Kodu (Devam)

```
sum_of_vectors(u,v)
for i from 1 to k
    projected_vector(i) = u(i) + v(i);
end for
projection_vector(u,v)
for i from 1 to k
    inner_product = inner_product + u(i) * v(i);
    norm_u = norm_u + u(i)^2;
end for
for i from 1 to k
    projected_vector(i) = u(i) * (inner_product / norm_u);
end for

/* Using basis to change coordinate system */
for i from 1 to n
    for index from 1 to k
        P_converted(i).converted_coordinate(index) = 0;
    end for
end for
for i from 1 to n
    for index from 1 to k
        for j from 1 to k
            P_converted(i).converted_coordinate(index) =
                P_converted(i).converted_coordinate(index) +
                basis(index)(j) * P(i).coordinate(j) - origine(j);
        end for
    end for
end for
```

Ek 7 QuickCP'nin "k" Boyuta Uyarlanmış Kodu

```
/* In k-dimensional real number space  $R^k$  ( $k \in \mathbb{N}$  and  $k \geq 2$ ), P is a
set which contains n points as {P1, P2, ..., Pn}. All points in
P are in the form of  $(x_1, x_2, \dots, x_k)$  */

for j from 1 to k

Sort P by  $x_j$ -coordinate, store in  $A_j$ 

// $A_j$  is an array with n points.

 $d_{\min} = \text{infinity}$ 

for r from 1 to n-1 // at most n-1 rounds

     $\Delta X_{\min_1} = \Delta X_{\min_2} = \dots = \Delta X_{\min_k} = \text{infinity}$ 

    for i = 1 to n-r do

        for j=1 to k

             $d = \text{dist}(A_k(i), A_k(i+j))$ 

            // distance between  $A_k(i)$  and  $A_k(i+r)$ 

             $\Delta X_k = A_k(i+r).x_k - A_k(i).x_k$ 

            //  $A_k(j).x_k$  is the  $x_k$ -value of  $A_k(j)$ 

            if  $\Delta X_k < \Delta X_{\min_k}$  then // update  $\Delta X_{\min_k}$ 

                 $\Delta X_{\min_k} = \Delta X_k$ 

            endif

            if  $d < d_{\min}$  then // update  $d_{\min}$ 

                 $d_{\min} = d$ 

            endif

        endfor

    endfor

endfor

if  $d_{\min} \leq \sqrt{\sum_k \Delta X_{\min_k}^2}$  then // termination condition

    break

endif

endfor
```

Ek 8 Rosetta Code (2013)'dan "k" Boyutta Uyarlanmış BFA Kodu

```
closestPair of (P)

/* N is the number of points in the set P

   where x1P is an array which is P sorted by X1 coordinate
   (ascending order) and k is dimension with  $k \in \mathbb{N}$  and  $k \geq 2$  */

if  $N \leq 3$  then

    return closest points of x1P using BFA
else

    x1L ← points of x1P from 1 to  $\lceil N/2 \rceil$ 

    x1R ← points of x1P from  $\lceil N/2 \rceil + 1$  to N

    xm ← x1P( $\lceil N/2 \rceil$ ).x1

    // x1P( $\lceil N/2 \rceil$ ).x1 is x1 coordinate value of x1P( $\lceil N/2 \rceil$ )

    x1L ← { p ∈ x1P : p.x1 ≤ xm }

    x1R ← { p ∈ x1P : p.x1 > xm }

    (dL, pairL) ← closestPair of (x1L)

    (dR, pairR) ← closestPair of (x1R)

    (dmin, pairMin) ← (dR, pairR)

    if dL < dR then

        (dmin, pairMin) ← (dL, pairL)

    endif

    x1S ← { p ∈ x1P : |xm - p.x1| < dmin }

    nS1 ← number of points in x1S

    (closest, closestPair) ← (dmin, pairMin)

    for i from 1 to nS1 - 1

        m ← i + 1

        while m ≤ nS1 and x1S(m).x1 - x1S(i).x1 < dmin

            if |x1S(m) - x1S(i)| < closest then

                (closest, closestPair) ← (|x1S(m) - x1S(i)|,

                                           {x1S(m), x1S(i)})

            endif

            m ← m + 1

        endwhile

    endfor

    return closest, closestPair

endif
```

