

# Kuantum Devre Sentezi ve Optimizasyonu için Verimli Bir Algoritma

## An Efficient Algorithm to Synthesize Quantum Circuits and Optimization

Ömercan Susam<sup>1</sup>, Mustafa Altun<sup>2</sup>

<sup>1</sup>Nanobilim ve Nanomühendislik  
İstanbul Teknik Üniversitesi  
susam@itu.edu.tr

<sup>2</sup>Elektronik ve Haberleşme Mühendisliği  
İstanbul Teknik Üniversitesi  
altunmus@itu.edu.tr

### Özet

Kuantum bilgisayarlar, özellikle kuantum devreler, hesaplamalı alanda vaadettikleriyle, son yıllarda dikkatleri üzerine çekti. Bu çalışmada, kuantum devrelerin sentezini ve optimizasyonunu ayrı aşamalarda gerçekleştirdik. Birinci kısımda, verilen bir Boolean fonksiyonu kuantum kapılar ile sentezleyen hızlı bir algoritma önerdik. Verilen her fonksiyon için kapsamlı bir arama yapmak yerine, algoritmamız önce temel fonksiyonların kütüphanesini oluşturup, ardından sıralama işlemini gerçekleştirmektedir. Örnek olarak, 3 bitlik devrelerle oluşturulabilen 40320 fonksiyonu elde etmek için, sadece, temel fonksiyon olarak adlandırdığımız 24 tanesini kullandık. İkinci kısımda, devrelerimizi, oluşturduğumuz şablonları kullanarak optimize ettik. Önerilen şablonlar, genel olarak negatif ve pozitif kontrol birimli Toffoli kapılarından meydana gelmektedir. Bu şablonlar aynı zamanda bize, literatürde daha önce optimum olarak sunulmuş olan çözümlerin aslında optimum olmadığını ve iyileştirilebileceğini gösterdi.

### Abstract

Quantum computers, more specifically quantum circuits, take on the eyes in recent years, with their computational promises. In this paper, we perform synthesis and optimization of quantum circuits in separate stages. In the first part, we propose a fast synthesis algorithm that implements any given reversible Boolean function with quantum gates. Instead of an exhaustive search on every given function, our algorithm creates a library of essential functions and performs sorting operation. As an example, to implement 40320 functions that can be created with 3 bit circuits, we only use 24 of them as we call essential functions. In the second part, we optimize our circuits by using templates that we create. The proposed templates mainly consist of Toffoli gates with negative and positive controlling lines. These templates also show us that optimum area solutions proposed in the literature are not actually optimum; they can be improved.

### 1. Giriş

Geleneksel olarak bilgisayarlar 0 veya 1 değeri alabilen bitler ile hesaplamaları gerçekleştirirler. Bitler ile hesaplama genel anlamda oldukça etkin ve verimli olmasına rağmen bazı önemli problemlerin çözümünde yetersiz kalmaktadır. Bunun en önemli nedeni ise bitlerin deterministik olarak çalışması, yani belirli bir zaman aralığında sadece 0 veya 1 değeri alabilmesidir. 1982 yılında Richard Feynman tarafından önerilen kuantum bilgisayar fikri [1], kuantum mekaniğinin gücünü ve özelliklerini kullanarak hesaplama işlemlerini gerçekleştirme mantığına dayanmaktadır. Bu hesaplama yönteminde, veri saklama elemanları “kübit” olarak adlandırılır. Kuantum mekaniğinin süperpozisyon prensibi gereği, kübitler 0 veya 1 değerini almak yerine aynı anda hem 0 hem 1 konumunda bulunabilir. Bir başka deyişle bir kübit, 0 veya 1 olma olasılığını belirtir, bu da 0 ile 1 arasında sonsuz bir değer alabileceği anlamına gelmektedir. Böylece, pratik limitler dahilinde olanaksız olan bir çok problem, kuantum algoritmaları ile rahatlıkla çözelebilmektedir [2][3]. Bu problemlerin belki de en ünlüsü kriptolojide yaygın kullanılan yarı-asal sayıların çarpanlarına ayrılmasıdır. Shor’un kübit tabanlı çarpanlara ayırma algoritması [2], geleneksel anlamda çözümü yüzyıllar süren durumları, hızlıca çözmektedir.

Kuantum devreler, kübitler üzerinde işlem yapan kuantum kapılar kullanılarak inşa edilir. Kuantum devrelerin en önemli özelliklerinden birisi aynı zamanda tersinir (reversible) devreler olmalarıdır. Bu devrelerde, devrenin girişindeki ve çıkışındaki bit sayısı eşittir. Devrenin çıkışındaki değerler bize devrenin girişindeki değerler hakkında bilgi verir. Bu sayede devreleri çift yönlü olarak kullanabilmemize imkan sağlarlar. Tersinir devrelerin en büyük getirisi, anlık durumlarından bir önceki durumlarını elde edebilecek şekilde çalışmalarından dolayı, bilgisayarlarda yüksek enerji tasarrufuna olanak sağlamalarıdır [4][5]. Kuantum devreler de bu anlamda tersinir devrelerin esnekliğinden faydalanır.

Kuantum bilgisayarlar ile ilgili yapılan deneysel uygulamalar her ne kadar emekleme aşamasında olsa da, kuantum hesaplamanın uygulanabilir olduğunu göstermektedir. Bu

noktada kuantum hesaplamanın en önemli bölümü olan kuantum devre tasarımı ön plana çıkmaktadır. Çalışmalar en az sayıda kapı kullanarak optimum devreleri sentezlemeyi amaçlamaktadır [6][7]. Optimize edilmiş devreler, bir yandan güvenilirliği artırırken [8], diğer bir yandan çalışma süresini düşürmektedir. Şuana kadar yapılan çalışmalarda, optimum devre sentezi sadece 4 bite kadar gerçekleştirildi [9]. Son yapılan ve 84 kübit kullanılan [10] deneyler göz önüne alındığında, optimal devre sentezi pratik olmaktan oldukça uzak kalmaktadır. Bu, çalışmamızı hızlı sentezleme algoritmalarına yönelten motivasyonların başında gelmektedir.

Önerdiğimiz yöntem, istenilen fonksiyonu kuantum kapıları kullanarak verimli bir şekilde elde etmekte ve bunu iki ana aşamada yapmaktadır. İlk aşamada, permütasyona dayalı bir algoritma ile seçilen bit büyüklüğüne göre optimum sayıda kapı kullanılan “temel fonksiyonlar” sentezlenmektedir. Algoritmanın bu aşaması, en çok vakit alan kısım olmasına rağmen, temel fonksiyonların sayısının azlığı ve sıralama algoritmasının hızı, bu yöntemi literatürdeki çalışmalardan oldukça hızlı kılmaktadır. Devre sentezleme aşaması, istenilen fonksiyonu elde etmemizi sağlayan sıralama süreci ile devam eder. Bu süreç, algoritmamıza hızını kazandıran, yöntemimizin en önemli bölümdür. İkinci aşama, oluşturduğumuz şablonları kullanarak, sentezlenen devrelerde optimizasyon yapmaktadır. Şablonlar, aynı fonksiyonu daha az sayıda kapıyla gerçekleyen ve devredeki eşdeğeri ile değiştirilerek toplam kapı sayısında düşüş sağlayan devrelerdir. Önerdiğimiz şablonları, literatürde optimum olarak sunulan çalışmalara da uygulayarak bu devrelerde de iyileştirmeler elde ettik.

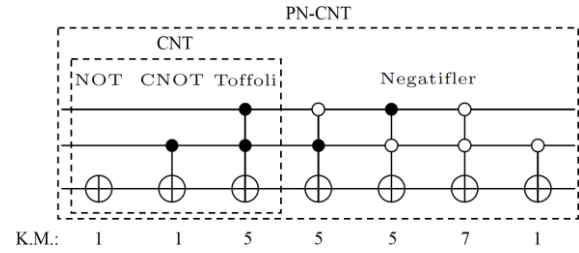
Çalışmamızın 2. Bölümünde, tersinir devreler ve tersinir kapılar ile ilgili temel bilgiler verdik. 3. Bölüm temel devrelere dayanan sentezleme algoritmamızı içermektedir. Şablonları kullanan optimizasyon yöntemimizi 4. Bölümde açıkladık. 5. Bölümde deneysel sonuçları ve literatür karşılaştırmasını sunduk. 6. Bölümde çalışmamızın neticelendirmesini yaparak, ileriye yönelik amaçlarımıza değindik.

## 2. Ön Hazırlık

Geleneksel Boolean fonksiyonlar, girişlerindeki bit sayısından bağımsız olarak tek bitlik bilgi taşıırken, tersinir devrelerde giriş ve çıkış kısımlarındaki bit sayısı eşit olmaktadır. Tersinir bir fonksiyon, girişindeki her bit kombinasyonu için çıkışında da ona özel bir bit kombinasyonu bulundurmaktadır. Bu durum tersten bakıldığında da geçerliliğini korur. Tersinir fonksiyonlar, matematikteki iki kümesi eşit sayıda eleman bulunduran örten fonksiyonlar gibi düşünülebilir.

Dijital devrelerin aksine, tersinir devrelerde feedback ve fun-out bulunmaz [11]. Tersinir kapılar, verilen fonksiyonları, hedefledikleri bitin lojik değerini değiştirerek gerçeklemek için kullanılırlar. CNT (CNOT, NOT, Toffoli) bu amaçla kullanılan, en çok bilinen ve kabul görmüş kapı kütüphanesidir. Toffoli kapısı, kontrol birimleri ve hedef bitinden oluşmaktadır. Tüm kontrol birimlerine 1 değeri atandığında hedef bitinin değerini tersine çevirir. CNOT ve NOT kapıları, sırasıyla tek kontrol birimli ve kontrol birimsiz Toffoli kapısı olarak düşünülebilir. Kapılar Şekil 1’de gösterilmiştir. Çizelge 1 bu kapılardan bazılarının, girilen bir

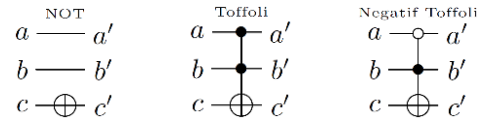
fonksiyonu nasıl değiştirdiğini göstermekte olup, Şekil 2’de aynı kapıların devre üzerindeki gösterimi bulunmaktadır.



Şekil 1: NOT, CNOT, Toffoli ve negatif kontrollü kapılar ve kuantum maliyetleri.

Çizelge 1: Kuantum Kapıların Doğruluk Tablosu

Not		Toffoli		Negatif Toffoli	
Giriş	Çıkış	Giriş	Çıkış	Giriş	Çıkış
cba	c'b'a'	cba	c'b'a'	cba	c'b'a'
000	100	000	000	000	000
001	101	001	001	001	001
010	110	010	010	010	110
011	111	011	111	011	011
100	000	100	100	100	100
101	001	101	101	101	101
110	010	110	110	110	010
111	011	111	011	111	111



Şekil 2: Kuantum Kapıların Devre Üzerinde Gösterimi

Negatif kontrol birimli Toffoli kapıları literatürdeki bazı çalışmalarda kullanılmıştır [12]. Bu kapılar yine aynı prensip ile çalışır ancak 1 yerine 0'ı hedef biti değiştirmek için kullanılırlar. Bu çalışmada biz, pozitif ve negatif kontrol birimlerini barındıran kütüphaneyi PN-CNT olarak adlandırdık ve uygulamalarımızda kullandık.

Tersinir kapıların maliyeti her kapı için 1 alınır. Bu kapılar, kuantum devrelerde, bazı alt-kapıların birleşiminden oluşmaktadır. Kullanılan alt-kapıların sayısı, bize tersinir kapıların kuantum maliyetini verir. Pozitif ve negatif kontrollü kapıların kuantum maliyetleri Şekil 1’de belirtilmiştir.

## 3. Sentezleme

Bu kısımda, verilen bir Boolean fonksiyonu, kuantum kapıların gerçekleyen hızlı bir sentezleme algoritması sunduk. Her fonksiyon için sonucu teker teker aramak yerine, algoritma öncelikle *temel fonksiyonların* kütüphanesini oluşturup, ardından sıralama işlemini gerçekleştirmektedir.

### 3.1. Temel Fonksiyonlar

“n” bit büyüklüğündeki bir Boolean fonksiyonun doğruluk tablosunda  $2^n$  satır bulunmakta ve bu da  $2^n!$  olası tersinir fonksiyon oluşurabileceği anlamına gelmektedir. Bu olası fonksiyonların tümünü ayrı ayrı elde etmek yerine, toplam olasılığın çok küçük bir bölümünü oluşturan temel

fonksiyonları kullanarak, diğer fonksiyonları kolayca elde edebileceğimizi bulduk. Temel fonksiyon kısaca, girişinde bulunan iki satırı, çıkışında birbirleriyle yer değiştiren, kalan satırları yerinde kalan bir fonksiyondur. Belirli bir bit sayısı için toplam temel fonksiyon sayısı,  $C(2^n, 2) = 2^{n-1} * (2^n - 1)$  ile hesaplanabilir. Bu formül kullanılarak elde edilen Çizelge 2; toplam fonksiyon sayısının, temel fonksiyon sayısına oranının eksponansiyel olarak arttığını, bit sayısı arttıkça, sunmuş olduğumuz yaklaşımın daha faydalı olacağını göstermektedir.

Çizelge 2: Bit Sayısına Göre Temel Fonksiyon Sayısı

Bit Sayısı	Fonksiyonlar	
	Temel Fonksiyon #	Toplam Fonksiyon #
2	6	24
3	28	40320
4	120	2092278988000
5	469	2.613308e + 35
6	2016	1.268869e + 89

Temel fonksiyonları sentezlemek için geliştirdiğimiz algoritma, bu fonksiyonları, tersinir kapılar ile optimum olarak elde etmektedir. Her temel fonksiyon için, en az kapı sayısından başlayıp, tüm kapıları ve permütasyonlarını deneyerek, fonksiyonu gerçekleştiren devreyi temel fonksiyonlar kütüphanesine ekler ve bir sonraki temel fonksiyonu aramaya başlar. Kütüphane tamamlandıktan sonra sıralama aşamasına geçilmektedir. Çizelge 3 ve Şekil 3, verilen bir F fonksiyonunun,  $f_1$  ve  $f_2$  temel fonksiyonları ile nasıl elde edildiğini göstermektedir.

Çizelge 3: İstenilen Bir Fonksiyonun Temel Fonksiyonlarla Gerçeklenmesi

$f_1$		+	$f_2$		=	$F$	
In	Out		In	Out		In	Out
cba	cba		cba	cba		cba	cba
000	000		000	<b>100</b>		000	<b>100</b>
001	001		001	001		001	001
010	010		010	010		010	010
011	<b>101</b>		011	011		011	<b>101</b>
100	100		100	<b>000</b>		100	<b>000</b>
101	<b>011</b>		101	101		101	<b>011</b>
110	110		110	110		110	110
111	111		111	111		111	111

Şekil 3:  $f_1$  ve  $f_2$  temel fonksiyonları ile  $F$  fonksiyonunun oluşturulması

Optimum çözüm üretmeyen yöntemlerle [12] bu devreler çok daha hızlı bir şekilde elde edilebilir ancak maliyetleri çok daha yüksek olacaktır. Temel fonksiyonların maliyetleri, oluşturulacak olan tüm devrelerin maliyetini etkileyeceğinden, devreleri optimum olarak sentezlemeyi tercih ettik.

### 3.2. Sıralama

Temel fonksiyonların üretilmesinin ardından, verilen herhangi bir fonksiyonu elde edebilmek için bir sıralama algoritmasının uygulanması gerekmektedir. Sıralama algoritmaları, matematik

ve bilgisayar biliminde uzun süredir çalışılan bir konu olduğundan, birçok farklı sıralama algoritması geliştirilmiştir. Biz çalışmamızda, “Seçmeli Sıralama” algoritmasını kullandık. Bu sıralama yöntemi, verilen fonksiyonu, doğruluk tablosu ile karşılaştırarak satır satır kontrol edip, eşleşmeyen her durum için temel fonksiyonlardan birini kullanarak, fonksiyonu adım adım birim fonksiyona çevirmektedir. Diğer sıralama algoritmalarının aksine kaydırma veya bölme işlemlerini uygulamadığından, fazladan temel fonksiyon kullanımını önleyerek, devre maliyetini düşük tutmaktadır. Örneğin, birleştirmeli sıralama, verilen sıralama kümesini öncelikle alt kümelerine ayırıp, bu alt kümeleri sıralamaktadır. Ardından, oluşturulan alt kümeler, parça parça birleştirilerek her yeni birleşmede yeni bir sıralama yapılmaktadır. Sıralamalardaki yer değiştirme işlemlerinin her biri ek bir temel fonksiyon kullanımına neden olmaktadır. Aynı şekilde, eklemeli sıralama algoritmasında kullanılan kaydırma işlemlerinin her biri, bir temel fonksiyona karşılık gelmektedir.

Seçmeli algoritma ile oluşturulan devrelerin maliyeti, yer değiştirilecek olan satırların değiştirilme sırasının, doğru bir şekilde belirlenmesiyle iyileştirilebilir. Çizelge 4’de bu durum ile ilgili bir örnek gösterilmiştir; kalın yazılmış satırlar birim fonksiyonun doğruluk tablosu ile eşleşmeyen satırları göstermektedir. Çizelgede dört satır yanlış yerde bulunmaktadır. Çizelge 4a, yukarıdan aşağıya doğru adım ilerleyen değişimi göstermektedir. Buna göre sırasıyla, 1. ve 7., 2. ve 7., 3. ve 7. satırları yer değiştiren temel fonksiyonlar kullanılmıştır. Her bir temel fonksiyonun maliyeti aynı sıraya göre 4, 4 ve 1 olduğundan, devrenin toplam maliyeti 9 kapı olmaktadır. Bu fonksiyon için optimum değişim sırası, tüm olası değişimler denenerek elde edilmiş ve Çizelge 4b de gösterilmiştir.

Çizelge 4: İki Farklı Değişim Sırasının Maliyetleri

(a)					(b)				
$f$					$f$				
000	0	0	0	0	000	0	0	0	0
<b>111</b>	<b>7</b>	1	1	1	<b>111</b>	<b>7</b>	<b>3</b>	1	1
<b>001</b>	<b>1</b>	<b>7</b>	2	2	<b>001</b>	<b>1</b>	<b>1</b>	<b>3</b>	2
<b>010</b>	<b>2</b>	<b>2</b>	<b>7</b>	3	<b>010</b>	<b>2</b>	<b>2</b>	<b>2</b>	3
100	4	4	4	4	100	4	4	4	4
101	5	5	5	5	101	5	5	5	5
110	6	6	6	6	110	6	6	6	6
<b>011</b>	<b>3</b>	<b>3</b>	<b>3</b>	7	<b>011</b>	<b>3</b>	7	7	7
T.F.	1-7	2-7	3-7	✓	T.F.	7-3	1-3	2-3	✓
M.	4	4	1	<b>9</b>	M.	1	2	2	<b>5</b>

Eşleşmeyen satır sayısını ‘m’ ile ifade edersek, tüm olası değişimlerin sayısı  $m!/2$  formülü ile hesaplanabilir. Örneğin, Çizelge 4 de bulunan fonksiyon için toplam oniki olası değişim sırası var demektir. Bunlardan iki tanesi çizelgede gösterilmiştir. Bu çalışmada biz, yukarıdan-aşağıya (Çizelge 4a) ve aşağıdan-yukarıya olmak üzere iki ayrı değişim sırası kullandık. Optimum değişimi bulan algoritmalar üzerine çalışmalarımız devam etmektedir.

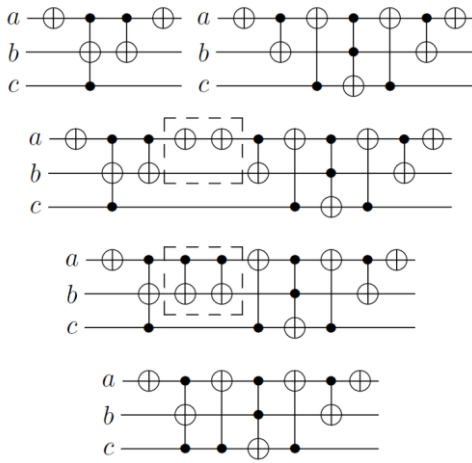
## 4. Optimizasyon

Ürettiğimiz devreleri, oluşturduğumuz şablonlar yardımıyla optimize ettik. Şablonlarımızı iki farklı yolla oluşturduk.

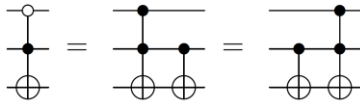
Birincisi, tersinir kapı kütüphanemizi kullanarak. İkincisi de bu kütüphanedeki kapıların içlerinde bulunan kuantum kapıları göz önüne alarak.

#### 4.1. Kütüphaneyi Kullanan Şablonlar

Şablonlar, sentezlenen devrelerin içerisinde aranarak, optimum olan eşdeğerleri ile yer değiştirilmekte böylece devrenin maliyeti düşürülmektedir. İki eş kapının yanyana geldiği durum, şablonlarımızdan biri olup, eşdeğeri birim fonksiyon, yani boş devredir. Değiştirilme sırasında kaynaklanan bazı durumlarda, iki tane aynı kapının komşu olduğu devreler oluşabilmektedir (bir temel fonksiyonun son kapısı ile diğerinin ilk kapısı). Bu durumlarda şablonumuz, devreden en az iki kapı eksiltmektedir (Şekil 4). Diğer şablonumuz, Toffoli-CNOT komşuluğuna dayanmaktadır. 3 bit için toplam 12 adet kombinasyonu olan bu şablonun Şekil 5’de bir örneği gösterilmiştir.



Şekil 4: Aynı iki kapının komşu olduğu sıralama dizilimi, toplam 4 kapı devreden çıkarıldı.



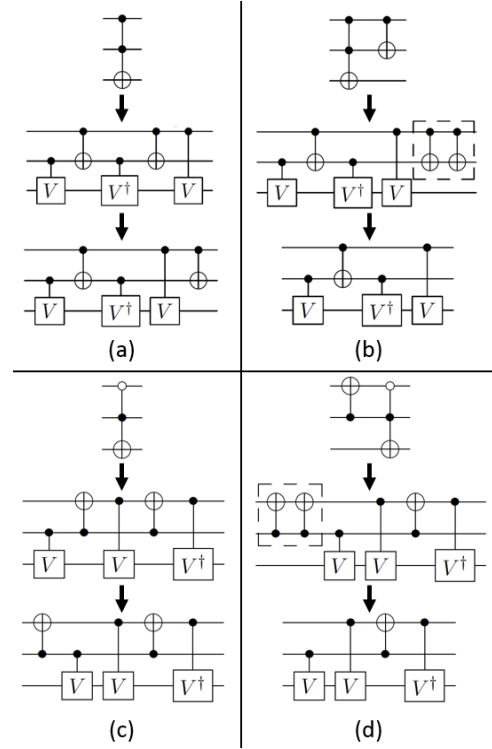
Şekil 5: Negatif kontrollü kapılar için örnek bir şablon

#### 4.2. Kapıların Kuantum Yapısını Kullanan Şablonlar

Kapı kütüphanesinden elde ettiğimiz şablonları kullanarak yaptığımız optimizasyonun ardından, devrede kullanılan kapıların kuantum yapıları göz önünde bulundurularak ikinci optimizasyon gerçekleştirilmektedir. Oluşturduğumuz şablonların, devre içerisinde bulunması halinde, Toffoli kapısı, kuantum devre yapısına genişletilerek (Şekil 6a, Şekil 6c) kapı eksiltme işlemi yapılır (Şekil 6b, Şekil 6d). 3 bit için toplam 12 adet Toffoli-CNOT kuantum şablonu bulunmaktadır.

### 5. Deneyisel Sonuçlar

Deneyler, C programlama dili kullanılarak, 3.20 GHz Intel i5 CPU’lu (tek çekirdek üzerinde) 4.00 GB RAM’e sahip bilgisayar ile gerçekleştirilmiştir.



Şekil 6: (a)(c) Toffoli kapısının, kuantum kapılarıyla oluşturulan iç yapısı, (b)(d) Toffoli için bir şablon

Çizelge 5, bu makalede önerdiğimiz yöntemin sonuçları ile literatürde bulunan optimum metodların sonuçlarını göstermektedir. Belirtilen süreler saniye cinsindendir. Sonuçlar 3 bit içindir. Tersinir maliyetleri (T.M.), her tersinir kapının maliyeti 1 alarak hesapladık. Kuantum maliyetleri (K.M.) hesaplar, daha önce yapılmış olan çalışmalarını göz önüne aldık [13]. Bu değerler Şekil 2’de belirtilmiştir. “Boyut” sütunundaki rakamlar, kullanılan tersinir kapı sayısını göstermektedir. Diğer sütunlardaki rakamlar, kaç adet fonksiyonun, boyut sütununda karşılık geldiği kapı sayısı ile çözümlendiğini göstermektedir. Örneğin ilk satır, Tek Değişimli Sıralama (TDS) metodu kullanıldığında, 4 fonksiyonun 28 kapıyla gerçekleştirildiğini belirtmektedir. “Önerilen Yöntem” sütunu bizim algoritmamızın sonuçlarını göstermektedir. CNT kütüphanesi kullanılarak yapılan sentezler CNT sütununun altında yer almaktadır. TDS metodu sadece, Bölüm 3’de belirttiğimiz yukarıdan-aşağıya değişim sırasını kullanmaktadır. Çift Değişimli Sıralama (ÇDS) metodu ise, yukarıdan-aşağıya ve aşağıdan-yukarıya değişimlerinin maliyetlerini kıyaslayarak, en düşük olan ile sentezlemeyi gerçekleştirmektedir. Bu iki yaklaşım kıyaslandığında ÇDS, TDS’ye göre tersinir devre maliyetinde, %5.13’lük bir iyileşme sağlamıştır.

PN-CNT kütüphanesinin kullanıldığı yaklaşım, tersinir ve kuantum maliyeti açısından en iyi sonucu verdiğini gördük. “Optimum” adlı sütunda, literatürde bulunan tersinir kapıların sentezlenmiş optimum sonuçlar gösterilmektedir. CNT kütüphanesiyle sentezlenmiş olan optimum devreleri, kuantum devrelere çevirip önerdiğimiz şablonları kullanarak, kuantum maliyeti açısından %5.61 oranında iyileştirdik. Bunun sonucunda, 40320 fonksiyonda kullanılan tüm kapıların ağırlıklı ortalaması, 13.88’den 13.10’a düşürdük.

Çizelge 5: 3 Bit için Belirli Sayıda Kapı ile Elde Edilmiş Tersinir Fonksiyonlar

Boyut	Önerilen Yöntem		Optimum		
	CNT		PN-CNT	CNT	PN-CNT
	TDS	ÇDS			
28	4				
27	29	1			
26	90	11			
25	207	55			
24	436	149			
23	791	327			
22	1252	712			
21	1954	1362			
20	2523	1936	1		
19	3349	2938	27		
18	3772	3477	105		
17	4125	4140	381		
16	4211	4388	956		
15	3842	4202	1827		
14	3522	3974	2965		
13	2835	3313	4260		
12	2308	2771	4910		
11	1706	2169	5659		
10	1239	1529	5288		
9	843	1150	4479		
8	547	682	3793	577	
7	340	495	2420	10253	
6	194	248	1619	17049	3236
5	111	167	928	8921	20480
4	52	63	423	2780	13282
3	25	41	198	625	2925
2	9	10	68	102	369
1	3	9	12	12	27
0	1	1	1	1	1
T.M. A.O.	15.97	15.15	10.58	5.86	4.57
K.M. A.O.	34.26	32.57	30.43	13.88	-
Süre	8	13	14	40 [14]	? [15]

Önerdiğimiz sentezleme yöntemini, optimum yöntemlerle kıyasladığımızda, çalışma sürelerimizin her zaman daha iyi çıktığını gördük. Daha büyük devreler için daha verimli sonuçlar alınabileceğini Çizelge 2’de gösterdik. Literatürde henüz 5 bit devreler için dahi optimum sentezleme algoritması bulunmamaktadır. 4 bit devreleri optimum sentezlemek için sunulmuş olan en iyi algoritmayı [9], 5 bit için uygulaysaydık, tüm fonksiyonları çözmemiz  $447 \times 10^{18}$  yılımızı alacaktı. Bu çıkarım, çalışmamızın amacını destekler niteliktedir.

## 6. Sonuç

Bu çalışmada, kuantum devre sentezi ve optimizasyonu için yeni bir yöntem önerdik. Temel fonksiyonlar kütüphanesini tersinir kapılarla elde ettikten sonra, istenilen bir Boolean fonksiyonu, temel fonksiyonları sıralayarak hızlı bir biçimde sentezledik. Sentezleme algoritmamızı, literatürdeki optimum sentezleme algoritmaları ile kıyasladığımızda oldukça hızlı çalıştığını gördük. Sentezlenen devreleri, önerdiğimiz şablonlarla optimize ettik. Bu önerilen şablonları, literatürdeki optimum sentezlere uyguladık. Daha önce optimum olarak sunulmuş olan CNT sentezinde, önerdiğimiz şablonları kullanarak %5.61’lik bir iyileştirme sağladık.

Önerdiğimiz bu yöntemi geliştirmek için çalışmalarımız, sıralama aşamasında optimum değişim sırasını bulan algoritmalar üzerine devam etmektedir. Bu sayede, devrelerin

toplam maliyetlerinde büyük iyileştirmeler yapılabileceğini Çizelge 4’de gösterdik. Ayrıca, kuantum bilgisayarlardaki işlem yükünü azaltmak için daha düşük seviyeli devre sentezleme ve optimizasyon algoritmaları (sinyal ve elementer kapı bazlı), ileriye yönelik amaçlarımız arasında bulunmaktadır.

## 7. Teşekkürler

Bu proje TÜBİTAK BİDEB 2210-C programı ile ve İTÜ BAP kapsamında desteklenmektedir.

## 8. Kaynaklar

- [1] R.P. Feynman, "Simulating physics with computers." International journal of theoretical physics, pp. 467-488, June 1982.
- [2] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," AT&T Research, Santa Fe, NM, 1995.
- [3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in 28th Annual ACM Symposium on the Theory of Computing, Murray Hill NJ, 1996.
- [4] Frank, Michael P. "Introduction to reversible computing: motivation, progress, and challenges." Proceedings of the 2nd conference on Computing frontiers. ACM, 2005.
- [5] Landauer, Rolf. "Irreversibility and heat generation in the computing process." IBM journal of research and development 5.3 (1961): 183-191.
- [6] P. Gupta, A. Agrawal, N.K. Jha. "An algorithm for synthesis of reversible logic circuits." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions, pp. 2317-2330, November 2006.
- [7] Maslov, Dmitri, Gerhard W. Dueck, and D. Michael Miller. "Toffoli network synthesis with templates." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 24.6 (2005): 807-817.
- [8] P.W. Shor, "Scheme for reducing decoherence in quantum computer memory." Physical review A 52, October 1995.
- [9] O. Golubitsky, D. Maslov, "A study of optimal 4-bit reversible toffoli circuits and their synthesis." Computers, IEEE Transactions, pp. 1341-1353, September 2012.
- [10] Z. Bian, F. Chudak, W.G. Macready, L. Clark, F. Gaitan "Experimental determination of Ramsey numbers." Physical review letters 111, 25 September 2013.
- [11] M.A. Nielsen, I.L. Chuang. Quantum computation and quantum information. Cambridge University press, 2010.
- [12] Schönborn, Eleonora, et al. "Optimizing DD-based Synthesis of Reversible Circuits using Negative Control Lines," unpublished.
- [13] A. Barenco, et al. "Elementary gates for quantum computation." Physical Review A 52, November 1995.
- [14] V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, "Synthesis of reversible logic circuits." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions, pp. 710-722, June 2003.
- [15] Wille, Robert, et al. "Exact synthesis of Toffoli gate circuits with negative control lines." Multiple-Valued Logic (ISMVL), 2012 42nd IEEE International Symposium on. IEEE, 2012.