**T.C.**
**İSTANBUL AYDIN ÜNİVERSİTESİ**
**LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ**

# ÇÖZÜMSÜZ MATEMATİK TEOREMLERİNDE BİLGİSAYAR HESAPLAMALI YÖNTEMLER VE COLLATZ KONJEKTÜRÜ

**YÜKSEK LİSANS TEZİ**

**Mert ÖZKENAR**

**Bilgisayar Mühendisliği Anabilim Dalı**

**Bilgisayar Mühendisliği Programı**

**HAZİRAN, 2021**

**T.C.**
**İSTANBUL AYDIN ÜNİVERSİTESİ**
**LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ**

# ÇÖZÜMSÜZ MATEMATİK TEOREMLERİNDE BİLGİSAYAR HESAPLAMALI YÖNTEMLER VE COLLATZ KONJEKTÜRÜ

**YÜKSEK LİSANS TEZİ**

**Mert ÖZKENAR**

**(Y1713.010008)**

**Bilgisayar Mühendisliği Anabilim Dalı**

**Bilgisayar Mühendisliği Programı**

**Tez Danışmanı : Dr. Peri GÜNEŞ**

**HAZİRAN, 2021**

# ONAY BELGESİ

## YEMİN METNİ

Yüksek Lisans tezi olarak sunduğum "Çözümsüz Matematik Teoremlerinde Bilgisayar Hesaplamalı Yöntemler ve Collatz Konjektürü" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (14/12/2020)

**Mert ÖZKENAR**

***Nefes alıp verdiğim her anımda olduğu gibi tez çalışmamın her aşamasında beni yalnız bırakmayan, sevgi, anlayış ve desteklerini evlatlarından esirgemeyen annem ve babama sonsuz sevgi ve saygılarımı sunarım.***
***Aileme,***

## ÖNSÖZ

Çalışmam Lisansüstü Eğitim Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Bilgisayar Mühendisliği Yüksek Lisans Tezi olarak hazırlanan "Çözümsüz Matematik Teoremlerinde Bilgisayar Hesaplamalı Yöntemler ve Collatz Konjektürü" isimli tezi içermektedir.

Tez çalışmam süresince birlikte çalışmaktan mutluluk duyduğum, bilgi birikimi ve tecrübesiyle katkıda bulunan değerli danışman hocam sayın Dr. Peri GÜNEŞ'e teşekkürü bir borç bilir, saygı ve sevgilerimi sunarım.

Akademik yaşantımda özel bir yeri olan yüksek lisans eğitimimin yanı sıra tez çalışmam süresince de yanımda olan sevgili aileme sonsuz teşekkür ederim.

**Aralık 2020** **Mert Özkenar**

**Bilgisayar Mühendisi**

**İÇİNDEKİLER**

**Sayfa**

**SEMBOLLER**

| | |
|---|---|
| $\mathbb{Q}^+$ | **:** Pozitif rasyonel sayılar kümesidir. |
| $\omega$ | **:** Negatif olmayan tamsayılar kümesidir. |
| $\mathbb{Z}$ | **:** Tamsayılar kümesidir. |
| $\mathbb{N}$ | **:** Doğal sayılar kümesidir. |
| $\top$ | **:** Tee sembolü. |
| $\bot$ | **:** Up tack ya da falsum sembolü. |
| $\equiv$ | **:** Denk ya da eşdeğer sembolü. |
| $\mu$ | **:** Mu harfi sembolü. |
| $\wedge$ | **:** Ve sembolü. |
| $\vee$ | **:** Veya sembolü. |
| $\rightarrow$ | **:** İse sembolü. |
| $\leftrightarrow$ | **:** Ancak ve ancak sembolü. |
| $\neg$ | **:** Değil sembolü. |
| $>$ | **:** Büyüktür sembolü. |
| $f$ | **:** Fonksiyon sembolü. |
| $\mathbb{Z}^+$ | **:** Pozitif tamsayılar kümesidir. |
| $\{\}$ | **:** Küme parantezi sembolü. |
| $\%$ | **:** Mod sembolü. |
| $=$ | **:** Eşit sembolü. |
| $\neq$ | **:** Eşit değil sembolü. |
| $+$ | **:** Artı sembolü. |
| $-$ | **:** Eksi sembolü. |
| $\times$ | **:** Çarpı sembolü. |
| $/$ | **:** Bölü sembolü. |
| $\in$ | **:** Elemanıdır sembolü. |
| $\forall$ | **:** Her belirteci. |
| $\ni$ | **:** Öyle ki belirteci. |
| $\exists$ | **:** Varlık niteleyicisi. |
| $\approx$ | **:** Yaklaşık sembolü. |

**KISALTMALAR**

| | |
|---|---|
| **C** | **:** C Programming Language |
| **Caml** | **:** Categorical Abstract Machine Language |
| **CCS** | **:** Calculus of Communicating Systems |
| **CLaReT** | **:** Computer Language Reasoning Tool |
| **CPO** | **:** Complete Partial Order |
| **CSP** | **:** Communicating Sequential Processes |
| **HOL** | **:** Higher Order Logic |
| **LCF** | **:** Logic for Computable Functions |
| **Lisp** | **:** List Processing Language |
| **LSM** | **:** Logic of Sequential Machines |
| **ML** | **:** Meta Language |
| **PRL** | **:** Proof Refinement Logic |
| **PVS** | **:** Prototype Verification System |
| **RSRE** | **:** Royal Signals and Radar Establishment |
| **λ-calculus** | **:** Lambda Calculus |
| **ML** | **:** Machine Learning |
| **AI** | **:** Artificial Intelligence |

## ÇİZELGE LİSTESİ

**Sayfa**

## ŞEKİL LİSTESİ

**Sayfa**

# ÇÖZÜMSÜZ MATEMATİK TEOREMLERİNDE BİLGİSAYAR HESAPLAMALI YÖNTEMLER VE COLLATZ KONJEKTÜRÜ

## ÖZET

Bu tez çalışmasında, çözümsüz matematik teoremlerinde kullanılan bilgisayar hesaplamalı yöntemlerin neler olduğu ve collatz konjektürü ile hesaplama yöntemleri anlatılmıştır. Tez içerisinde temel bilimsel kavramlar ve ifadeler tüm ayrıntılarıyla açıklanmıştır. Çözümsüz matematik teoremlerinde yararlanılan bilgisayar hesaplamalı yöntemlere değinilmiştir. Teorem kanıtlama programları tüm yönleriye incelenmiş ve detaylandırılmıştır. Collatz konjektürüne dair kapsamlı araştırmalar ve çalışmalar yapılmış, tüm bilgilerin tezde yer alması amaçlanmıştır. Collatz konjektürünün hesaplanmasında kullanılacak üç farklı yöntem üzerinde çalışılmıştır. Her yöntem için metodolojiler oluşturulmuş, bilgisayar programları geliştirilmiş, sonuçlar bilgisayar çıktıları ve çıktılardan oluşan tablolar olarak paylaşılmış ve grafikler ile zenginleştirilmiştir. Collatz konjektürünün hesaplanmasına yönelik yöntemler karşılaştırılmış ve yöntem önerisinde bulunulmuştur. Üzerinde çalışılan yöntemlerin bilgisayar programları sayesinde etkinliği ve performansları ölçümlenmiştir. Önerilen yöntemin standart yönteme oranla daha başarılı olduğu yönünde bulgulara ulaşılmıştır. Collatz konjektürüne makine öğrenmesi ile yaklaşım geliştirilmiş ve çalışmalar yapılmıştır. Makine öğrenmesi özelinde bilgisayar programı geliştirilmiştir. Makine öğrenmesi metodolojisinde geliştirilen bilgisayar programı ile bilinmeyen değerlerin tahminlemesi yapılmıştır. Tüm çalışmalara ayrıntılı olarak tezin içerisinde yer verilmiştir. Tez çalışmasınında elde edilen yöntem özet değerleri ve karşılaştırma bilgileri, sonuç kısmında paylaşılmıştır.

**Anahtar Kelimeler :** *Collatz Konjektürü, Bilgisayar Hesaplamalı Yöntemler, Teorem Kanıtlama Programları, Bilimsel Kavramlar, HOL, LCF, ProofPower, Isabelle, Lothar Collatz, Dolu Taneleri Problemi, Makine Öğrenmesi, Denetimli Öğrenme, Lineer Regresyon.*

**COMPUTER COMPUTATIONAL METHODS IN UNSOLVABLE MATHEMATICAL THEOREMS AND COLLATZ CONJECTURE**

## ABSTRACT

In this thesis, computer computational methods used in unsolvable mathematical theorems and computation methods with the Collatz conjuncture are explained. In the thesis, basic scientific concepts and expressions are explained in detail. Computer computational methods used in unsolvable mathematical theorems are mentioned. Theorem proofing programs are examined and detailed in all aspects. Extensive researches and studies have been conducted on the Collatz conjuncture, and it is aimed to include all the information in the thesis. Three different methods have been studied to calculate the Collatz conjuncture. Methodologies were developed for each method, computer programs were developed, the results were shared as program outputs, tables consisting of that program outputs and enriched with graphics. Methods for calculating the Collatz conjuncture were compared and a method was proposed. The efficiency and performance of the methods studied were measured by means of computer programs. It has been found that the proposed method is more successful than the standard method. Machine learning approach has been developed and studies have been carried out on the Collatz conjecture. A computer program has been developed specifically for machine learning. The unknown values were estimated with the computer program developed in the machine learning methodology. All studies are included in the thesis in detail. The method summary values and comparison information obtained in the thesis study are shared in the conclusion part.

**Keywords :** *Collatz Conjecture, Computer Computational Methods, Automated Theorem Proving, Theorem Proving Programs, Scientific Concepts, HOL, LCF, ProofPower, Isabelle, Lothar Collatz, Hailstone Sequence, Machine Learning, Supervised Learning, Linear Regression.*

# 1. GİRİŞ

## 1.1 Çalışma Konusu

Tez çalışmasında çözümsüz matematik teoremlerinde kullanılan bilgisayar hesaplamalı yöntemler, teorem kanıtlama programları ve Collatz Konjektürü konu edilmiştir. Collatz Konjektürü'nün hesaplanması için farklı yöntemler üzerinde çalışılacak, yöntem önerisi sunulacak ve önerinin sağladığı fayda bilimsel kanıtlar ile ispat edilecektir.

## 1.2 Tezin Amacı

Tezin amacı çözümsüz matematik teoremlerinde yararlanılan teorem kanıtlama programları, onlara ait hesaplama yöntemlerinin detaylıca anlaşılmasını sağlamaktır. Collatz Konjektürü'nün yapısını, özelliklerini, işleyiş mekanizmasını tüm yönleriyle inceleyip üzerinde geliştirme çalışmaları yapmaktır. Collatz Konjektürü'nün hesaplanması için yöntemler belirlenecek, yöntemler arasında karşılaştırmalar yapılacak ve yöntem önerisinde bulunulacaktır. Sürecin içerisinde oluşturulan metodolojilere uygun bilgisayar programı geliştirmeleri yapılacak, sonuç kümeleri tablolar halinde paylaşılacak, grafik ve görseller ile konu pekiştirilecektir.

## 1.3 Literatür Araştırması

Tez çalışması süresince çok sayıda yerli ve yabancı kaynak incelenmiştir. Özel erişimli kütüphane veritabanları içerisinde araştırmalar yapılmıştır. Springer, IEEE, Web of Science, ScienceDirect, Times Higher Education başta olmak üzere internet kaynakları üzerinden literatür incelemeleri yapılmıştır. Uluslararası bilim, mühendislik ve matematik dergilerinde yayınlanan makaleler okunmuştur. Bazı konferans bildirilerine erişim sağlanmıştır. Yabancı üniversitelerin tez çalışmaları kontrol edilmiş, yerli tez çalışmalarına ise Ulusal Tez Merkezi aracılığıyla erişilmiştir. Kütüphane üzerinden de yayınlanan tezlere fiziksel olarak erişim fırsatı

bulunmuştur. Tüm kaynaklara ek olarak Academia ve ResearchGate üzerinde paylaşılan çalışmalar da okunarak tez çalışmasına katkıda bulunulmuştur.

## 1.4 Hipotez

Tez çalışmasında bilimsel konseptlere, çözümsüz matematik teoremlerinde başvurulan bilgisayar hesaplamalı yöntemlere ve teorem kanıtlama programlarına ait bilgiler paylaşılacaktır. Collatz Konjektürü hakkında da etraflıca araştırma ve çalışmalar yapılacak değerlendirmeleri paylaşılacaktır. Collatz Konjektürü'nün mucidi olan Lothar Collatz'a da değinilecektir.

Collatz Konjektürü için farklı yöntemler değerlendirilecek, metodolojiler belirlenecek, üzerinde çalışmalar yapılacak, bilgisayar programları geliştirilecektir.

Tez çalışmasının bir de Collatz Konjektürü özelinde hipotezi bulunmaktadır. Çalışmalar içerisinde yaklaşım önerisinde bulunulacaktır. Hipotez ise Collatz Konjektürü işlem hesaplamalarında parite sekansı yönteminin, standart yönteme göre hem toplam işlem adımı sayısı bazında hem de tüm işlemlerin tamamlanma süresi bazında daha başarılı olduğu tezidir.

Tez çalışması içerisinde "Collatz Konjektürü işlem hesaplamalarında parite sekansı yönteminin, standart yönteme göre hem toplam işlem adımı sayısı bazında hem de tüm işlemlerin tamamlanma süresi bazında daha iyi sonuçlar vermektedir." hipotezinin ispatı aranacak, oluşturulan metodolojiler temelinde geliştirilen bilgisayar programları sayesinde edinilen veriler ışığında ispatı sağlanacaktır. Yine tüm çalışmalara ait veriler, çıktılar, grafikler ve görsellere detaylıca yer verilecektir. Çalışmalara, edinilen bilgilere ve varılan karara dair sonuç açıklaması da yapılacaktır.

# 2. TEMEL KAVRAMLAR

## 2.1 Önerme Tanımı

Bir hüküm bildiren ve hakkında doğru veya yanlış denilmesi anlamlı olan ifadelere ***Önerme*** denir [1]. Önermeler genellikle $P, Q, R, S$ harfleri ile temsil edilirler. Önermede belirtilen hüküm doğru ise $T$ harfi veya $\top$ sembolü, yanlış ise $F$ harfi veya $\perp$ sembolü kullanılır. Bir önerme hem doğru hem de yanlış olamaz. Bu ara değeri dışlama kuralıdır. Bir önerme kısmen doğru ya da kısmen yanlış olamaz. Bu kurala ise çelişki kuralı adı verilir [2]. Doğruluk değerleri aynı olan önermelere denk veya eşdeğer önerme denir ve $P \equiv Q$ şeklinde gösterilir [1].

Aşağıda önerme örnekleri bulunmaktadır;

- Ay Dünya'nın çevresinde döner.
- Maymunlar uçabilirler.
- $7 + 5 = 12$

Aşağıdaki ifadeler önerme örneği olamaz;

- Bu akşam tiyatro gösterisine gidelim.
- Pastane nerede?
- $x + y = 10$

"Her sayının tersi vardır." ifadesi bir önermedir. Bu önerme;

- $(\mathbb{N},\ x,\ 1)$ için yanlış,
- $(\mathbb{Q}^{+},\ x,\ 1)$ için doğru,
- $(\omega,\ +,\ 0)$ için yanlış,
- $(\mathbb{Z},\ +,\ 0)$ için doğrudur [3].

Bir önermenin değili oluşturulabilir. Birden fazla önerme mantıksal bağlaçlar ile birleştirilebilir. Bu durumlarda *bileşik önerme* elde edilir. Önerme değişkenlerinin olası tüm değerleri için bileşik önermenin sonuçlarını listeleyen tabloya *doğruluk tablosu* adı verilir.

**Çizelge 2.1 :** Örnek $P \wedge Q$ önermesinin doğruluk tablosu üzerindeki gösterimi.

| $P$ | $Q$ | $P \wedge Q$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

**Çizelge 2.2 :** Temel mantıksal bağlaç örnekleri.

| | |
|---|---|
| $P\ ve\ Q$ | $P \wedge Q$ |
| $P\ veya\ Q$ | $P \vee Q$ |
| $P\ ise\ Q$ | $P \rightarrow Q$ |
| $P\ ancak\ ve\ ancak\ Q$ | $P \leftrightarrow Q$ |
| $P\ değil$ | $\neg P$ |

## 2.2 Hipotez Tanımı

Gözlem yapılan olay veya durumların değişkenleri arasındaki ilişkinin incelenerek, araştırmanın olası sonucuna dair yapılan tahminlerin ifadesidir [4]. Araştırmacının araştırma problemi sonucuna ilişkin yargı bildiren bilimsel önermelerdir. Araştırmacının doğru olduğunu düşündüğü, henüz ispatlanmamış geçici çözümleme veya teorik çıkarımdır. Hipotezler deneysel ve kantitatif bilimsel çalışmalarla kanıtlanabilir [5]. Bu çalışmalar sonucunda hipotez doğrulanabilir ya da yanlışlanabilir. Hipoteze varsayım, denence, faraziye de denilmektedir [6].

Hipotezler kendi içerisinde aşağıda yer alan alt gruplara ayrılır;

- Basit hipotez
- Karmaşık hipotez
- Ampirik hipotez
- Sıfır hipotezi
- Alternatif hipotez
- Mantıksal hipotez
- İstatiksel hipotez [7]

İki değişken arasındaki ilişki üzerine yapılan tahmin ***basit hipotez*** konusunu oluşturur. Bağımsız değişken etkiye sahip olan, bağımlı değişken ise bundan etkilenen konumundadır.

- Şekerli içeceklerin günlük olarak tüketimi obeziteye yol açar.

İki veya daha fazla bağımsız değişkenin, yine iki veya daha fazla bağımlı değişken ile oluşturduğu ilişki ve etkileri ***karmaşık hipotez*** konusudur.

- Yaşlı ve kırsal alanlarda yaşayan insanlar, genç ve şehirlerde yaşayan insanlara oranla daha mutludur.

Kanıta dayalı, bilimsel metodlar ile ispatlanabilen gözlem veya deneyimler ***ampirik hipotez*** konusunda incelenir. Bir dizi deneme yanılma yöntemi sonucunda varılabilen kanaatler de türe dahil edilir.

- Sıvı B vitamini katkısı ile sulanan güller, sıvı E vitamini katkısı ile sulanan güllere oranla daha hızlı gelişme gösterirler [7].

Değişkenler arasında ilişki bulunmadığı ***sıfır hipotezi*** ile ifade edilebilir. Şüphe uyandırma veya bilimsel çürütme amacı ile de oluşturulabilir. **$H_0$** sembolü ile gösterilir

- Toplumdaki eğitim seviyesi düşüklüğünün işsizlik üzerinde bir etkisi yoktur [8].

$$H_0 : \mu_{es} - \mu_{iş} = 0$$

Değişkenler arasında ilişkinin var olduğu ***alternatif hipotez*** ile ifade edilebilir. **H1** sembolü ile gösterilir

- Öğrencilerin matematik ve fen bilgisi dersi başarıları arasında bir ilişki vardır.

$$H_1 : \mu_{mat} - \mu_{fen} \neq 0$$

- Öğrencilerin matematik dersi başarıları arttıkça fen bilgisi dersindeki başarıları da artar [4].

$$H_1 : \mu_{mat} - \mu_{fen} > 0$$

Belirli koşullar altında doğru olan bir modelden, mantıksal olarak tahmini çıkarımlar yapılması ***mantıksal hipotez*** konusudur.

- Yeşil yapraklı bitkiler gelişmek için güneş ışığına ihtiyaç duyuyorsa, diğer bitkiler de gelişmek için güneş ışığına ihtiyaç duyar.

İstatistiksel olarak doğrulanabilen ifadeler ***istatistiksel hipotez*** olarak belirtilir. Bir popülasyon içerisindeki değişkene dair çözümleme ve beklentilerden oluşur [7][8].

- San Francisco şehri sakinlerinin ortalama geliri $100.000 üzerindedir.

## 2.3 Teori Tanımı

Çalışma konusunun dayandığı kuralları, gerçeği veya olayı açıklamak için önerilen görüş ve fikirlerin bilimsel bir açıklaması teori olarak adlandırılır [9]. Doğal Dünya hakkında birbirinden bağımsız çok sayıda gözlem ve deneylerle sınanmış, yanlışlanmamış gerçeklerin iyi bir şekilde ifadesidir. Bu nedenle aynı şartlar altında yapılan benzer gözlem ve deneylerle de doğrulanmaya devam etmesi beklenir [10]. Bilimde, bir teze çok bağımsız gözlem ve deney üzerinde doğrulanana kadar teori denmez. Teoriler hipotezlerden daha kesin, ama yasalardan daha kesin değildir. Bir teoriyi test etmek için takip edilecek prosedürler ve uygulanacak işlemler her bilimsel disiplinde iyi tanımlanmıştır [11]. Teori ile kuram aynı anlama gelmektedir.

"Bazı bilimsel açıklamalar o kadar iyi kurulur ki, hiçbir kanıt onları değiştiremez. Yeni bir bilimsel teori olur. Günlük kullanım dilinde teori bir önsezi veya spekülasyon anlamına gelir. Bilimde değil. Bilimde, teori kelimesine atıfta bulunur. Zamanla toplanan gerçekler tarafından desteklenen doğanın önemli bir özelliğinin kapsamlı bir açıklaması. Teoriler, bilim insanlarının henüz gözlemlenmemiş olaylar hakkında öngörülerde bulunmalarına da izin veriyor [12]."

Her bilimsel teori bir hipotez olarak başlar. Hipotez henüz kanıtlanmamış fikirdir. Hipotezi desteklemek için yeterli kanıt birikirse, bilimsel yöntemde teori olarak bilinen bir sonraki adıma geçer ve bir fenomenin geçerli bir açıklaması olarak kabul edilir.Teoriler veya yorumlanma biçimleri değişebilir, ancak gerçeklerin kendileri değişmez. Teori, olguların dikkatli ve rasyonel bir şekilde incelemesine dayanmalıdır. Gerçekler ve teoriler iki farklı şeydir. Bilimsel yöntemde, gözlemlenebilen veya ölçülebilen gerçekler ile bilim insanlarının gerçekleri açıklamaları olan teoriler arasında açık bir ayrım vardır [13].

Teori, bilimsel yöntemin sonucu değildir; teoriler tıpkı hipotezler gibi kanıtlanabilir veya reddedilebilir. Tahminler doğruluğu zamanla daha da artacak şekilde daha fazla bilgi toplandıkça teoriler geliştirilebilir veya değiştirilebilir. Teoriler, bilimsel bilgiyi ilerletmek ve toplanan bilgileri pratik kullanıma koymak için temellerdir [13]. Teoriler yasanın nasıl çalıştığını açıklar [14].

Teorilerin bazı temel özelliklerini incelemek gerekirse;

- Teoriler, doğal olayların açıklamalarıdır.
- Dahili olarak tutarlı ve kanıtlarla uyumlu
- Sıkıca kanıtlanmış ve kanıtlara dayandırılmış
- Çok çeşitli olaylara karşı test edilmiş
- Problem çözmede gözle görülür derecede etkilidir.
- Teoriler, aslında gerçekler olabilir. Teoriler değişebilir, ancak bu uzun ve zor bir süreçtir. Bir teorinin değişmesi için teorinin açıklayamadığı birçok gözlem veya kanıt bulunmalıdır [10] [11].

## 2.4 Kanun Tanımı

Kanun doğanın belirli koşullar altında nasıl davranacağına ilişkin kuralları belirtir [10]. Kanunlar geniş bir ampirik veri topluluğu tarafından desteklenir [15]. Kanunlar defalarca sınanmış, çok güvenilir kanıtların bulunduğu, oldukça doğrulanmış açıklamalardır [16]. Kanunlar aynı koşullar altında her zaman doğrudur ve bu nedenle tahminlerde bulunmak için kullanılabilir [17]. Bu tür bir güvenilir bilgi, bilim insanlarının evrenle ilgili gerçeklere ulaşabileceği en yakın ifadelerdir [16]. Kanun ve yasa kelimeleri aynı anlamda kullanılmaktadır.

Model, anlayışı geliştirmeye yardımcı olan düşüncenin tanımı, grafiği veya üç boyutlu temsilidir. Bilim insanları, neyin çok küçük (atom,molekül vb.) veya çok büyük (kainat, uzay vb.) olabileceğini anlamalarını sağlamak için bir yola ihtiyaç duyduklarında sıklıkla modelleri kullanırlar. Bir model, gerçekten üzerinde çalışılanın herhangi bir simülasyonu, temsili veya benzeridir. İyi bir model, gerçek sistemde ilgilendiğiniz temel değişkenleri içerir, gerçek sistemdeki tüm gözlemleri açıklar ve mümkün olduğunca basittir. Bir model, dünyayı temsil eden bir küre gibi sade veya ışığı temsil eden matematiksel denklemler kadar karmaşık olabilir. Modeller sadece belli koşullar altında neler olacağını tahmin etmenin bir yoludur. Modeller kavramların daha kolay anlaşılmasına yardımcı olmaktadır [10].

Bir fikrin kanun haline gelmesi, gelecekteki bilimsel araştırmalarla değiştirilemeyeceği anlamına gelmez. Günlük kullanım dilinde kanun mutlak bir şey ifade eder. Bilimsel kullanımda kanun çok daha esnektir. İstisnalar olabilir, yanlış olduğu kanıtlanabilir veya zaman içinde gelişebilir [15].
Kanunlar bir fenomenin neden var olduğunu veya buna neyin sebep olduğunu açıklamaz. Fenomenin açıklamasına teori denir. Teori nedeni, kanun ise ne olduğunu açıklar [15]. Bu nedenle bu iki kavram ne oldukları ve ne yaptıkları bakımından nitelik olarak farklıdır [14].

Yaygın bir yanılgı, bilimsel teorilerin yeterli veri ve kanıt toplandığında en sonunda bilimsel kanunlara geçecek olgunlaşmamış fikirler olduğu yönündedir. Bir teori, yeni veya daha iyi kanıtların birikimi ile bilimsel bir kanuna dönüşmez. Teoriler açıklamadır ve kanunlar sık sık bir denklem olarak yazılmış büyük miktarda veride

gördüğümüz kalıplardır. Bir teori her zaman teori olarak, bir kanun her zaman kanun olarak kalır [10].

## 2.5 Metot Tanımı

Bilimin özelliği olarak tanımlanan etkinlikler arasında sistematik gözlem ve deney, endüktif ve tümdengelimli akıl yürütme, hipotez ve teorilerin oluşumu ile test edilmesi yer almaktadır. Bunların ayrıntılı olarak nasıl uygulandığı çok değişkenlik gösterebilir, ancak bunun gibi özellikler bilimsel aktivitelerin bilimsel olmayan bir faaliyetten ayrılmasının bir yolu olarak görülmüştür. Buna bilimsel metot adı verilir. Yöntem kelimesi de metot ile aynı anlamda kullanılmaktadır [18].

Kişisel ve kültürel inançların hem algıları hem de doğal olayların yorumlarını etkileyebilme olasılığına karşın, bir teori geliştirirken bu etkileri en aza indirgemek için standart prosedürler ve kriterler kullanılması hedeflenmektedir. Bilimsel metot, kişinin önyargı veya önyargı etkisini en aza indirmeye çalışır [19]. Amacı, araştırmanın adil, tarafsız ve tekrarlanabilir bir şekilde yürütülmesidir [20].

Bilimsel metot dört adımdan oluşur;

- Bir fenomenin veya fenomen grubunun gözlemlenmesi ve tanımlanması
- Fenomeni açıklamak için bir hipotezin formüle edilmesi
- Diğer fenomenlerin varlığını öngörmek veya yeni gözlemlerin sonuçlarını nicel olarak tahmin etmek için hipotezin kullanılması
- Tahminlerin deneysel testlerinin birkaç bağımsız deneyci tarafından gerçekleştirilmesi ve deneylerin uygun şekilde yapılması [19].

Bilimsel metot bilgi, tahminler veya kontrol gibi bilimin amaçlarından ve ürünlerinden ayırt edilmelidir. Metotlar, bu hedeflere ulaşılmasının aracıdır. Bilimsel metot ayrıca meta-metodolojiden, nesnellik, tekrarlanabilirlik, basitlik veya önceki başarılı sonuçlar gibi değerlerden ayrılmalıdır. Metodu yönetmek için metodolojik kurallar önerilmiştir [18].

## 2.6 Metodoloji Tanımı

Bilimsel araştırma ve çalışmanın nasıl yürütüleceğini belirleyen stratejiye metodoloji ismi verilir. Metodoloji çok sayıda farklı prosedür, protokol ve tekniği içermektedir [21].

Metodoloji aşağıda belirtilen bileşenleri içermektedir;

- Analiz edilen verilerin seçiminde veya nitel araştırmalarda, incelenen konular ve araştırma ortamının seçiminde verilen kararları,
- Bilgi tanımlamak ve toplamak için kullanılan araç ve yöntemler ile ilgili değişkenlerin nasıl tanımlandığı,
- Verileri işleme biçimi ve bu verileri analiz etmek için kullanılan işlemleri,
- Temel hipotezi ve araştırma sorularını incelemek için kullanılan özel araştırma araçları veya stratejileri [22].

Metot araştırma yapmak için atılan teknik adımları ifade eder. Metotların açıklamaları genellikle onları tanımlamayı ve bir araştırma problemini araştırmak için neden belirli teknikleri seçtiğinizi belirtmeyi, ardından verileri sistematik olarak seçmek, toplamak ve işlemek için kullandığınız prosedürlerin ana hatlarını içerir [22].

Metodoloji, belirli metotların neden kullanıldığının altında yatan mantığın tartışılmasını ifade eder. Bu tartışma, uygulanacak yöntemlerin seçimini bildiren teorik kavramları tanımlamayı, yöntem seçimini akademik çalışmanın daha genel niteliğine yerleştirmeyi ve araştırma problemini incelemeyle alaka düzeyini incelemeyi içerir. Tartışma aynı zamanda konuyu araştırmak için diğer araştırmacıların kullandığı yöntemler hakkındaki literatürün kapsamlı bir incelemesini de içermektedir [22].

Metodoloji bilimsel çalışmalar kapsamında önemli bir role sahiptir çünkü;

- Deneylerin ve veri toplamanın tekrarlanmasına izin verir,
- Sonuçları üretmek için kullanılan koşulları detaylandırır,

- Belirli bir deney veya ölçüm için yapılan belirli operasyonel seçimlerin gerekçesini açıklar,
- Metodoloji yeterli olmadan, araştırma sonuçları düzensiz, tekrarlanamaz ve güvenilmez hale gelir [21].

## 2.7 Kanıt Tanımı

Bilimsel argümanlar temelde üç bileşenden oluşur: fikir (bir hipotez veya teori), bu fikrin ürettiği beklentiler (sıklıkla tahminler) ve bu beklentilerle ilgili gerçek gözlemler (kanıtlar). Bilim, uzak galaksilerden en küçük madde parçacıklarına, zamanın başlangıcından gelecek yılın kasırga mevsimine, küresel ekonomilerin etkileşimlerinden tek bir nöron içindeki kimyasal reaksiyonlara kadar tüm doğal olayları araştırır. Bilim insanlarının fikirlerini doğrulamak için izlediği metot, metodoloji, yaklaşımlar çeşitlilik gösterir. Bazıları deneylere, bazıları gözlemsel çalışmalara dayanır. Bilimsel argümanlar oluşturmak için bu fikirlerin test edilmesi gerekir [23]. Bir ifadenin belirtilen kurallarla doğrulanmasına kanıt adı verilir [24]. Bilim kanıtlara dayanır ve kanıtlar bilimin çekirdeğini oluşturur [23]. İspat kelimesi de kanıt ile aynı anlamda kullanılmaktadır.

Bilimsel fikirler sadece test edilebilir olmakla kalmamalı, aynı zamanda birçok farklı bağımsız deneyci tarafından test edilerek doğrulayıcı kanıtlara erişilmelidir. Bu özellik bilimin merkezinde yer almaktadır. Bilim insanları fikirlerini test etmek için aktif olarak kanıt ararlar. Bu tür testleri yapmak bilim için çok önemlidir. Çünkü bilimde, bilimsel bir fikrin kabulü veya reddedilmesi dogma, popüler görüş veya gelenek üzerine değil onunla ilgili kanıtlara bağlıdır. Bilimde kanıtlarla desteklenmeyen fikirler sonuçta reddedilir [25].

Kanıt türleri çalışmanın yapıldığı bilimsel disipline göre kendi içerisinde sınıflandırılabilir. Genel olarak, deneysel çalışmaların daha güçlü kanıtlar sağladığı ve daha açık bir sebep-sonuç ilişkisi tanımladığı düşünülmektedir. Örneğin Tıp Bilimi'nde kanıt türleri en zayıftan en güçlüye aşağıdaki sırayla listelenmektedir [26];

- Anekdot ve Uzman Görüşleri

  Anekdot kanıt, bir kişinin kendi kişisel deneyimini veya görüşünü ifade eder, mutlaka tipik deneyimleri temsil etmez. Bir uzmanın bağımsız görüşü veya yazılı bir haberde verilenler, bilimsel araştırmalar yapılmadan bunları destekleyecek zayıf kanıt biçimleri olarak kabul edilir.

- Hayvan ve Hücre Çalışmaları (deneysel)

  Hayvan araştırması faydalı olabilir ve insanlarda da görülen etkileri önceden tahmin edebilir. Bununla birlikte, gözlenen etkiler de farklılık gösterebilir, bu nedenle insanlarda belli bir etkinin görülmesi söylenmeden önce insan denemeleri gerekir. İzole edilmiş hücreler üzerinde yapılan testler ayrıca vücuttakilerden farklı sonuçlar verebilir.

- Vaka Raporları ve Vaka Serisi (gözlemsel)

  Bir vaka raporu, belirli bir konu hakkında yazılı bir kayıttır. Kanıt hiyerarşisi düşük olsa da, yeni hastalıkların veya tedavilerin yan etkilerinin tespitine yardımcı olabilirler. Bir vaka serisi benzer, ancak birden çok konuyu izler. Her iki tür çalışma da nedensellik kanıtlayamaz, sadece korelasyon kurar.

- Vaka-Kontrol Çalışmaları (gözlemsel)

  Vaka kontrolü çalışmaları, biri belirli bir koşulu veya belirtisi olan ve biri olmayan iki konu grubunu içeren retrospektiftir. Daha sonra, buna neden olabilecek bir öznitelik veya pozlama belirlemek için geri dönerler. Yine, bu çalışmalar korelasyon göstermektedir, ancak nedensellik kanıtlamak zordur.

- Kohort Çalışmaları (gözlemsel)

  Bir kohort çalışması, vaka kontrol çalışmasına benzer. Belirli bir özelliği veya tedaviyi paylaşan (örneğin bir kimyasal maddeye maruz kalma) paylaşan bir grup insanın seçimini içerir ve bunları zaman içinde bu özellik veya tedaviye sahip olmayan ve sonuçta herhangi bir farklılık göstermeyen bir grup insanla karşılaştırır.

- Randomize Kontrollü Denemeler (deneysel)
  Denekler rastgele, tedaviyi alan bir test grubuna veya genellikle bir plasebo alan bir kontrol grubuna atanır. “Kör” denemelerde katılımcılar hangi grupta olduklarını bilmiyor; “çift kör” denemelerde, deneyciler de bilmez. Körleştirme denemeleri, önyargının giderilmesine yardımcı olur [26].

## 2.8 Konjektür Tanımı

Matematik’te doğru olduğu düşünülen fakat henüz kesin olarak kanıtlanmamış önerme ya da teoremler için konjektür ifadesi kullanılır. Bir çok durum için doğru olan bir model fark edildiğinde konjektürler ortaya çıkar. Bununla birlikte, bir modelin çoğu durumda doğru kalması, modelin tüm durumlar için geçerli olacağı anlamına gelmez. Matematiksel gözlemin tamamen kabul edilmesi için konjektür kanıtlanmalıdır. Bir konjektür kesin olarak kanıtlandığında, bir teorem olur [27].

Konjektür problem çözmede önemli bir adımdır; bu sadece matematikçiler için bir araç değildir. Gündelik problem çözmede, bir problemin çözümünün hemen ortaya çıkması çok nadirdir. Bunun yerine, problem çözme süreci, problem yapısını analiz etmeyi, vakaları incelemeyi, çözümle ilgili bir varsayım geliştirmeyi ve bu varsayımı kanıt ile onaylamayı içerir [27].

Konjektür geliştirme çalışmaya yönelik tutarlı bir örüntü fark edilmesi ile yapılır.

Örneğin aşağıda Pascal Üçgeni’nin $4^{\text{üncü}}$ satıra kadar olan kısmı yer almaktadır.

$$
\begin{array}{ccccccc}
 & & & 1 & & & \\
 & & 1 & & 1 & & \\
 & 1 & & 2 & & 1 & \\
1 & & 3 & & 3 & & 1 \\
1 & 4 & & 6 & & 4 & 1
\end{array}
$$

Pascal Üçgeni’nin $n^{\text{inci}}$ satırında yer alan elemanların toplamını elde edebilecek bir ifade düşünülürse, tahmin etme sürecini başlatmak için en mantıklı yaklaşım basit vakalar için ne olduğunu görmektir. İlk birkaç satır toplanarak başlanır: [27]

$0^{ıncı}\ satır:$ $1 = 1$
$1^{inci}\ satır:$ $1 + 1 = 2$
$2^{inci}\ satır:$ $1 + 2 + 1 = 4$
$3^{üncü}\ satır:$ $1 + 3 + 3 + 1 = 8$
$4^{üncü}\ satır: 1 + 4 + 6 + 4 + 1 = 16$

Ardından bu sonuçlardaki örüntü gözlemlenir. Bu olay veya nesnelerin düzenli bir biçimde birbirini takip ederek gelişmesini ifade eder. Sonuçların 2'nin kuvvetleri şeklinde devam ettiği açıktır. Modelin doğru olup olmadığını anlamak için bir sonraki satır ya da diğerleri üzerinde deneme yapılır [27].

$5^{inci}\ satır: 1 + 5 + 10 + 10 + 5 + 1 = 32$

Model bu satır için de doğru sonuç sağlamaktadır. Bu aşamadan sonra istenilen sayıda deneme yapılabilir, ancak şu ana kadar toplanan bilgiler bir varsayım yapmak için yeterlidir [27].

Bu çalışma sonucunda bir konjektür elde edilmiştir.

Konjektür: "Pascal Üçgeni'nin $n^{\text{inci}}$ satırında yer alan elemanların toplamı $2^n$'e eşittir. [27]"

# 3. TEOREM KANITLAMA PROGRAMLARI

## 3.1 HOL Teorem Kanıtlayıcısı

HOL interaktif teorem kanıtlayıcısı , yüksek dereceli mantıkla spesifikasyonlar ve formal kanıtlar oluşturmak için genel ve yaygın olarak kullanılan bir bilgisayar programıdır [28]. Teoremlerin kanıtlanabileceği ve kanıtlama araçlarının uygulanabileceği bir programlama ortamı sunar. Yerleşik karar prosedürleri ve teorem kanıtlayıcıları otomatik olarak birçok basit teorem oluşturabilir [29]. Sistem endüstri ve akademiyi donanım tasarımı ve doğrulaması, güvenlikle ilgili akıl yürütme, gerçek zamanlı sistemler hakkında kanıt oluşturma, donanım tanımlama dillerinin anlam bilgisi, derleyici doğrulaması, program doğruluğu, modelleme tutarlılığı ve program iyileştirme gibi birçok alanda formal akıl yürütme konusunda destekler [28].

HOL Cambridge Üniversitesi'nde Mike Gordon tarafından geliştirilmiş mekanize bir kanıt asistanıdır. Öncelikle dijital donanımın doğruluğu hakkında düşünmek için kullanılmıştır. Ancak donanım doğrulama için HOL'da geliştirilenlerin çoğu - örneğin aritmetik teorisi - diğer birçok uygulama için de temeldir. Sistemin altında yatan mantık ve temeller tamamen geneldir ve prensip olarak yüksek mertebeden mantıkta formalleştirilebilecek herhangi bir alanda muhakemeyi desteklemek için kullanılabilir. HOL, etkileşimli teorem kanıtlama konusunda LCF (Logic for Computable Functions) yaklaşımına dayanmaktadır ve Cambridge ile Edinburgh'da geliştirilen LCF teorem kanıtlayıcıları ile ortak birçok özelliğe sahiptir. LCF gibi, HOL sistemi de ML (Meta Language) fonksiyonel programlama dili mantığını temsil ederek güvenilir şekilde teoremi kanıtlar. Mantık önermeleri ve teoremleri ML soyut veri türleriyle temsil edilir ve teorem kanıtlayıcı ile etkileşim, bu veri türlerinin değerleri üzerinde çalışan ML prosedürleri uygulanarak gerçekleşir. HOL genel amaçlı bir programlama dilinin üzerine inşa edildiğinden, kullanıcı ispat stratejileri uygulamak için isteğine göre karmaşık programlar yazabilir. Ayrıca, mantığın ML soyut veri türleri kullanılarak temsil edilme şekli nedeniyle, bu tür kullanıcı tanımlı

kanıt stratejilerinin yalnızca geçerli mantıksal çıkarımlar gerçekleştirmesi garanti edilir. HOL sistemi, değerleri yüksek dereceli mantık teoremleri olan özel bir ML özet veri tipine sahiptir. Thm tipinde değişmez değerler yoktur; başka bir deyişle, sadece bir tane yazarak thm türünde bir nesne elde etmek mümkün değildir. Bununla birlikte, sistem oluşturulduğunda bazı thm türünde değerler verilen önceden tanımlanmış ML tanımlayıcıları vardır. Bu değerler, yüksek mertebeden mantığın aksiyomlarına karşılık gelir. Ayrıca HOL, teoremleri argüman olarak alan ve teoremleri sonuç olarak döndüren önceden tanımlanmış birkaç ML prosedürü sunar. Bu prosedürlerin her biri mantığın ilkel çıkarım kurallarından birine karşılık gelir ve sadece karşılık gelen çıkarım kuralını kullanarak giriş teoremlerinden mantıksal olarak takip eden teoremleri döndürür. ML tipe bağımlı şekilde yazılmış bir dil olduğundan, tür denetleyicisi thm türündeki değerlerin yalnızca bu önceden tanımlanmış işlevler kullanılarak oluşturulmasını sağlar. Bu nedenle HOL'da, thm türünün her değeri ya bir aksiyom olmalı ya da mantığın ilkel çıkarım kurallarını temsil eden önceden tanımlanmış fonksiyonlar kullanılarak hesaplanarak elde edilmelidir.Bu nedenle HOL'daki her teorem çıkarsama kuralları kullanılarak aksiyomlardan üretilmelidir. Bu şekilde ML tip kontrolörü, HOL teorem kanıtlayıcının doğruluğunu garanti eder. İlkel çıkarım kurallarına ek olarak, HOL'da birçok türetilmiş çıkarım kuralı vardır. Bunlar, ilkel çıkarım kurallarının uygun sırasını uygulayarak yaygın olarak kullanılan ilkel çıkarım dizilerini gerçekleştiren ML prosedürleridir. Türetilmiş çıkarım kuralları, HOL kullanıcısını bir kanıtta gereken tüm ilkel çıkarımları açıkça verme ihtiyacından kurtarır. Türetilmiş bir kural için ML kodu isteğe bağlı olarak karmaşık olabilir; ancak hiçbir zaman geçerli mantıksal çıkarımın izlemediği bir teorem döndürmez, çünkü tür denetleyicisi türetilmiş kuralların yalnızca ilkel çıkarım kurallarına bir dizi çağrı ile elde edilmişlerse teoremleri döndürebilmesini sağlar. Tarif edilen bu teorem yaklaşımı, HOL teorem kanıtlayıcısının sağlamlığını temin eder, ancak hesaplama açısından maliyetlidir. Üst düzey mantıktaki basit teoremlerin bile formal kanıtları binlerce ilkel çıkarım gerektirebilir. Ve bu kanıtlar HOL'da yapıldığında, tüm çıkarımlar gerçekte ilgili ML prosedürleri uygulanarak gerçekleştirilmelidir. Bununla beraber, HOL'ün birlikte etkili kanıt stratejilerinin programlanmasına izin veren iki önemli özelliği vardır. Bunlardan ilki şudur: HOL'da kanıtlanmış teoremler diske kaydedilebilir ve bu nedenle gelecekteki kanıtlarda her ihtiyaç duyulduğunda üretilmeleri gerekmez. İkinci özellik, yüksek dereceli mantığın kendisinin ifade

gücüdür, bu da mantıkta yararlı ve çok genel 'lemmaların' belirtilmesini sağlar. Bu nedenle, programlanmış bir kanıt kuralının yapması gereken çıkarım miktarı, istenen sonuçların nispeten az miktarda bir kesinti ile takip edildiği genel teoremleri önceden kanıtlayarak azaltılabilir. Bu teoremler daha sonra türetilmiş çıkarım kuralı tarafından gelecekteki kanıtlarda saklanabilir ve kullanılabilir. Çalışma esnasında çıkarımı önceden kanıtlanmış teoremlerle değiştirme stratejisi , tip polimorfizmi ve yeterli genelliğe sahip teoremler verecek kadar anlamlı kılan yüksek mertebeden değişkenler mantığı sayesinde mümkündür. Bu sayede herhangi bir somut özyinelemeli türün 'aksiyomatizasyonunun' etkili bir şekilde çıkarılabileceği tek bir genel teorem de elde edilebilmektedir [30].

### 3.1.1 LCF

Orijinal LCF sistemi, 1972 yılında Robin Milner tarafından Stanford Üniversitesi'nde geliştirilen bir kanıt kontrol programıdır. LCF'nin türevleri artık bilgisayar destekli akıl yürütmede gelişen bir paradigma oluşturmaktadır. Son 25 yıldaki gelişmelerin çoğunda, otomatikleştirilmiş akıl yürütme alanı üzerinde etkili çalışmaları olan Robin Milner'ın katkısı bulunmaktadır. LCF'nin türevlerinden biri olan HOL, başlangıçta donanım hakkında akıl yürütmek için geliştirilen yüksek dereceli mantık kanıt asistanıdır. Robin Milner'ın HOL gelişimine çok yönlü katkısı dikkat çekicidir. Sadece teorem ispatında LCF yaklaşımını icat etmekle kalmayıp, aynı zamanda bunun temelindeki ML programlama dilini ve  ML, LCF ve HOL mantığında kullandığı yenilikçi polimorfik tip sistemini tasarlamıştır. Milner'ın yazdığı kod bugün hala kullanımda olup, donanım doğrulama sistemi LCF LSM tasarımı Milner'ın “Calculus of Communicating Systems” isimli kitabından esinlenmiştir [31].

### 3.1.2 Stanford LCF

“LCF” Milner'ın isimlendirdiği, 1969'da Dana Scott tarafından tasarlanan ancak 1993'e kadar yayınlanmayan bir mantığın adı olan “Logic for Computable Functions - Hesaplanabilir Fonksiyonlar için Mantık” ın kısaltmasıdır. LCF mantığı λ-calculus’den terimler ve predicate calculus’den formüller içerir. Türler Scott Domain’leri (CPO) olarak yorumlanır ve mantık, sabit noktalı indüksiyon kullanılarak, anlamsal tanımlarda kullanılan türün özyinelemeli olarak tanımlanmış işlevleri hakkında akıl yürütmeye yöneliktir. Stanford'daki orijinal LCF takımı Robin

Milner ve Milner'ın Lisp'i öğrendiği Whitfield Diffie'den oluşuyordu. Diffie daha sonra kriptografiyle ilgilenmeye başladı. Daha sonra Richard Weyhrauch takıma katıldı, hemen ardından Malcolm Newey geldi. Tüm bu kişiler, şu anda Stanford LCF olarak bilinen orijinal LCF sistemini tasarlamak, uygulamak ve kullanmak için işbirliği yaptı. Ortaya çıkan sistem, Scott'ın tasarladığı mantık için bir kanıt denetleyicisidir ve Milner tarafından aşağıdaki gibi tanımlanmıştır:

"Kanıt denetimi programı, kullanıcıyla etkileşimli olarak bilgisayar bilimcilerin ilgi alanları dahil olmak üzere çeşitli alanlardan hesaplanabilir fonksiyonlar ve fonksiyoneller hakkında formal kanıtlar üretmesini sağlamak için tasarlanmıştır. Örneğin, tamsayılar, listeler ve bilgisayar programları ile bunların anlambilimi gibi. Kullanıcının görevi ise alt amaçlama imkanı ve güçlü bir sadeleştirme mekanizması sayesinde kolaylaştırılmıştır."

Kanıtlar, bir ana hedef (Scott'ın mantığında bir formül) belirlendikten sonra sabit bir alt hedef komutları seti (temeli ve adımları oluşturmak için indüksiyon gibi) kullanarak alt hedeflere bölünerek gerçekleştirilir. Alt hedefler ya sadeleştirici kullanılarak çözülür ya da doğrudan çözülünceye kadar daha basit alt hedeflere bölünür. Scott'ın mantığında formal ispatları temsil eden veri yapıları ispat komutları yorumlandığında oluşturulur. Bunlar çok fazla bellek tüketebilir.

Stanford LCF birçok örnek olay incelemesi için kullanılmıştır. Weyhrauch, imperatif bir dilin yığın tabanlı hedef bir dile derleme algoritmasının doğruluğunun kanıtı ve Newey tamsayı ile listelerin denklem teorilerinin oluşturulması üzerinde çalıştı [31].

### 3.1.3 Edinburgh LCF

1973 yılında Milner Edinburgh Üniversitesi'ne geçti ve Stanford LCF'nin bir sonraki adımını oluşturmak için bir proje başlattı. Daha sonrasında bu projeyi Edinburgh LCF olarak adlandırıldı. Başlangıçta her ikisi de Stanford'dan doktora mezunu olan Lockwood Morris ve Malcolm Newey'i araştırma görevlisi olarak işe aldı. Stanford LCF ile ilgili ispat boyutunun mevcut hafıza ile sınırlı olması ve sabit ispat komutları grubunun kolayca genişletilememesi sorunları bulunuyordu. Milner, Edinburgh LCF'deki bu eksiklikleri gidermek için girişime başladı. Bütün kanıtları

kaydetmek yerine, sistemin kanıtların sonuçlarını, yani teoremleri hatırlaması gerektiği fikrine sahipti. İspatın adımları küçük bir tahta kullanan, daha sonraki kısımlara yer açmak için ispatın önceki kısımlarını silip temizleyen bir matematik öğretim görevlisi gibi gerçekleştirilecek, ancak kaydedilmeyecekti. Teoremlerin sadece kanıtla oluşturulabilmesini sağlamak için Milner, önceden tanımlanmış değerleri aksiyom örnekleri olan ve işlemleri çıkarsama kuralları olan soyut bir veri türü kullanma fikrine sahipti. Daha sonra sıkı tip denetimi, oluşturulabilecek değerlerin sadece bir dizi çıkarım kuralı, yani teoremler uygulanarak aksiyomlardan elde edilebilen değerler olmasını sağladı. Milner kanıt komutları kümesinin genişletilmesini ve özelleştirilmesini sağlamak için, Morris ve Newey'in yardımıyla ML programlama dilini tasarladı. Bu tip denetimi sayesinde teorem güvenilirliğini sağlayan soyut tip mekanizmasını destekliyordu.

Stanford LCF'de, Scott'ın mantığının aksiyomları ve çıkarsama kuralları, basitleştirme ve alt-hedefleme mekanizmasının uygulanmasında doğrudan kodlanmıştır. Kullanıcı yalnızca hedeflerden geriye doğru kanıtlar oluşturabilir. Scott'ın yayınlanmamış makalesinde, mantığı aksiyom şemaları ve çıkarım kuralları verilerek geleneksel şekilde sunuldu. Önerdiği doğrudan formal kanıt kavramı, her biri ya bir aksiyom olan ya da önceki bir üyeden çıkarım kuralıyla takip edilen bir dizidir. Edinburgh LCF'de mantık soyut bir tür olarak kodlanarak doğrudan ileri yönde kanıtı destekledi. Tasarım hedefi ML'deki programlar tarafından hedefe yönelik kanıt araçları uygulamaktı. ML'yi buna uygun hale getirmek için dil, alt-hedefleme stratejilerinin fonksiyonlar olarak temsil edilebilmesi için fonksiyonel yapıldı. Milner bu fonksiyonları taktikler olarak adlandırıyordu. Bu sayede stratejileri birleştirme işlemleri, stratejileri argüman olarak alan ve sonucunda fonksiyonlar olarak döndüren üst düzey işlevler olarak programlanabilmekteydi.Stratejilerin yanlış hedefler uygulandığında başarısız olabileceği öngörülerek ML'ye istisna yönetim mekanizması dahil edilmiştir. Teoremin ihtiyaçları, ML'nin ilk versiyonunun tasarımını çok güçlü bir şekilde etkiledi. Birçok tasarım detayı, kanıt araçlarının programlamada kullanımları ışığında alternatifler düşünülerek çözülmüştür. Bu titiz tasarım odağı basit ve tutarlı bir dille sonuçlandı.

1975 yılında Morris ve Newey, sırasıyla Syracuse Üniversitesi ile Avustralya Ulusal Üniversitesi'nde öğretim görevlisi oldular ve onların yerine Chris Wadsworth ve Mike Gordon geçtiler. ML ve Edinburgh LCF'nin tasarımı ve uygulaması tamamlandı. Ayrıca "Edinburgh LCF" kitabı yazıldı ve yayınlandı. 1978'de ilk LCF projesi tamamlandı. Chris Wadsworth, Rutherford Appleton Laboratuvar'ındaki kalıcı işine dönüş yaptı. Mike Gordon ise Edinburgh'da doktora sonrası bir bursla desteklendi ve yeni araştırma alanı donanım doğrulama oldu. İlk LCF projesi bittikten sonra Edinburgh LCF kullanan uygulama çalışmaları devam etti. Milner'ın öğrencisi Avra Cohn, programlama dili uygulamalarını doğrulamak üzerine bir doktora yaptı. Brian Monahan ise veri LCF kullanarak veri tiplerinin teorisi ve mekanizasyonu üzerine doktora yapmışlardır, LCF ile çalışan diğer araştırmacılar ise Jacek Leszczylowski ve Stephen Sokolowski'dir [31].

### 3.1.4 Cambridge LCF

Mike Gordon 1981'de Cambridge Üniversitesi Bilgisayar Laboratuarı'nda öğretim görevlisi olarak kalıcı bir pozisyona geçti. Bir başka LCF projesi SERC tarafından finanse edildi. Yakın zamanda Stanford'dan doktora derecesi alan Larry Paulson, Cambridge'de işe alındı. Devamında David Schmidt ve Lincoln Wallen işe alındı. Bu zaman zarfında Gérard Huet, Edinburgh LCF kodunu Lelisp ve MacLisp'e taşıdı. Paulson ve Huet daha sonra bir işbirliği içerisinde birbirlerine manyetik bantlar göndererek LCF'nin çok sayıda gelişimi yaptılar. Huet ML'yi geliştirip genişletti ve LCF'nin teori dosyalarının uygulanmasını optimize etti. Huet'in grubu daha sonra CAML'ı geliştirdi. Paulson, Lisp kodunun çoğunu geliştirdi ve basitleştirdi. Edinburgh LCF'nin Edinburgh DEC-10 sisteminde mevcut olan sınırlı miktarda bellekte çalışmasını sağlamak için gerekli olan bazı alan optimizasyonlarını kaldırdı. Edinburgh LCF yorumlamalı olarak çalıştı, ancak Paulson ve Huet'in işbirliği sırasında yaklaşık yirmi kat daha fazla hız sağlayan bir ML derleyicisi uygulandı.

Cambridge'deki LCF çalışmalarının bir parçası olarak Paulson, hem kanıtlama araçlarının nasıl tasarlanacağı ve programlanacağı konusundaki anlayışta hem de Gérard Huet ile LCF'nin uygulanmasında çarpıcı iyileştirmeler yaptı. Standart dönüşüm teknikleri ve teorem devamları onun tarafından tasarlandı ve daha sonra geniş bir araç koleksiyonu uygulamak için kullanıldı. Edinburgh LCF, eski bir monolitik sadeleştiriciye sahipti. Paulson bunu basit ve temiz bir kural olarak

yeniden tasarlayıp programladı. ‘Artificial Intelligence Programming’ kitabındaki bir algoritmadan esinlenerek, yeniden yazımda kullanılan denklem setlerini verimli bir şekilde indekslemek için bir veri yapısı uyguladı. Bu, büyük kural setlerini verimli bir şekilde ele almak için önemli bir araç haline geldi.

Paulson ayrıca LCF mantığını, predicate calculus’un tüm standart yapılarını içerecek şekilde güncelledi. Scott'ın mantığında ayrışım veya varoluşsal nicelik yoktu. Ayrıca bir yığın üzerinde alt hedefleri yönetmek için basit bir paket uyguladı. Edinburgh LCF kullanıcıları genellikle alt hedefleri, g_2_1_3 gibi kullanışsız isimlerle ML değişkenlerine açıkça bağlayarak elle yönetti). Bu gelişmeler bir dizi büyük vaka çalışması tarafından yönlendirildi ve test edildi. Bu çalışmalara Paulson tarafından birleşme algoritmasının doğruluk kanıtının formalizasyonu ve kontrol edilmesi dahildi. Ortaya çıkan yeni LCF sistemi “Cambridge LCF” olarak adlandırıldı ve 1985'te tamamlandı. Bundan sonra Paulson üzerinde çok az çalışma yaptı. Rutherford Appleton Laboratuvarı’ndan Mikael Hedlund daha sonra Cambridge LCF'yi, ML'nin yeni bir uygulamasını kullanarak Standart ML'ye taşıdı. Cambridge LCF'nin ortaya çıkan Standart ML tabanlı sürümü, 1987 yılında Paulson'un ‘Logic and Computation’ isimli kitabında belgelenmiştir. Kitap vaka çalışmaları ve temel teorinin öğretisi ile desteklenmiştir [31].

## 3.2 LCF’ten HOL’a Geçiş Süreci

Paulson Cambridge LCF'yi tasarlarken ve uygularken, Mike Gordon çoğunlukla donanım doğrulamasıyla ilgileniyordu. Milner'ın İletişim Sistemleri Hesaplaması (CCS - Calculus of Communicating Systems) Genişleme Teoreminin, karma bir bileşen davranışının bireysel bileşenlerinin paralel bileşiminden hesaplanmasını doğrudan açıklamasını nasıl sağladığı Gordon’ı etkiledi. Bu, bir dijital sistemin davranışını sistemin yapısal açıklamasından türetmek için iyi bir paradigma gibi görünüyordu. Sıralı makine davranışını gösterebilmek için CCS'nin Genişleme Teoremine manipülatif benzer bir yasa ile özel bir notasyon geliştirdi.( LSM - Logic of Sequential Machines). LSM’e bir kanıt asistanı sağlamak için Cambridge'in bir versiyonu olan LCF LSM’i kullandı. Bu versiyon daha okunabilirdi, ayrıştırma mekanizması ve ek bir aksiyom şeması olarak genişleme yasası sağlamaktaydı. Bu düzenleme oldukça iyi çalıştı ve Cambridge dışında bile kullanıldı. Bunu daha sonra

Tamarack6 olarak adlandırılan bir oyuncak mikroişlemciyi doğrulamak için kullandı. LSM gösterimi, Kraliyet Sinyal ve Radar Kuruluşundaki (RSRE- Royal Signals and Radar Establishment) bir grup tarafından başarısız olan Viper işlemcisini belirlemek için kullanıldı. Bu süre zarfında Stanford'dan yeni mezun olan Ben Moskowski, Cambridge'de doktora yaptı. Moskowski, Gordon'a LSM terimlerinin yüklem hesaplamasında, LSM genişleme yasasının türetilmiş bir kural haline gelebileceği şekilde nasıl kodlanabileceğini gösterdi. Bu standart yüklem hesaplamasının bir dizisine karşılık geliyordu. Bu yaklaşım hem daha akıllıca hem de daha sağlam bir mantık temeline dayanıyordu. Gordon yaklaşımı benimsedi ve HOL meydana geldi. Bu arada CCS'nin Genişleme Teoremi sadece LSM'den HOL'a ilham verici bir basamak taşı olmakla kalmadı. Monica Nesi CCS'ye kanıt desteği sağlamak için Genişleme Teoreminin mekanizasyonu da dahil olmak üzere HOL'u kullandı.

Cambridge LCF tarafından desteklenen mantık, yüklem hesabının genel formül yapısına ve λ-calculus terim yapısına sahiptir. Milner'dan dolayı tür sistemi aslında Church'ün orijinal sistemidir, ancak tür değişkenleri meta dilden nesne diline taşınmıştır. Church'ün sisteminde, tür değişkenleri olan bir terim aslında bir meta gösterimdir, ailesini ifade eden bir terimdir, LCF'de ise tek bir polimorfik terimdir.LCF'in terimleri Scott mantığındaki üyelerin gösterimi olarak yorumlaması donanım doğrulaması için bir aşırı yükleme idi. Donanım doğrulaması için, sabit noktalı Scott indüksiyonunun ekstra karmaşıklığına nadiren ihtiyaç vardır; sıradan matematiksel tümevarım yeterlidir. HOL sistemi LCF sözdizimini korur, ancak türleri Scott kümeleri yerine sıradan kümeler olarak yorumlar.

Mevcut LCF kodunun yeniden kullanılmasını sağlamak için HOL aksiyomları ve çıkarım kuralları doğrudan Church'ün standart olanlarından alınmamıştır. Örneğin, LCF mantığı basit bir çıkarım kuralı olarak paralel yer değiştirmeye sahiptir. Bu karar Edinburgh LCF tasarlanırken bir deneme sonucu alınmıştı. Church mantığı ise farklı bir ilkeye sahipti. Gordon HOL'da mevcut olan etkin kodu kullanmak istediği için LCF ikamesini kullandı. LCF'den bir başka miras ise, doğal bir çıkarım mantığının kullanılmasıydı. Sonuç olarak, HOL mantığı oldukça özel formal bir temel ile sonuçlandı [31].

### 3.2.1 HOL'un gelişimi

Başlangıçta HOL, kayıt aktarımı düzeyinde donanım doğrulaması için oluşturuldu. Desteklenecek modelleme tekniği, giriş ve çıkış sinyalleri arasındaki ilişkiler olarak donanım cihazlarını temsil eder ve dahili sinyaller varoluşsal niceleme ile gizlenir. Sinyaller, zamandan değerlere kadar fonksiyonlarla temsil edilir, böylece daha üst düzey ilişkiler ve niceliklendirme gerekir. Bu, uygun bir formalizm olarak daha üst düzey mantığı önerir. HOL tasarımı, teorisini klasik üst düzey mantıktan ve uygulamasını ise büyük ölçüde LCF'den almıştır. Sistemin gelişimini öncelikle donanım doğrulama vakaları yönlendirdi.

HOL sisteminin ilk sürümü, üst düzey mantık somut sözdizimini desteklemek üzere Cambridge LCF ayrıştırıcısını ve okunabilir görünümü değiştirerek oluşturuldu. HOL terimleri, LCF kodunun yeniden kullanımını desteklemek için LCF yapıları olarak kodlanarak tasarlandı. LCF'nin birçok yönü, örneğin tür denetimi ve teori yönetimi değiştirilmeden HOL'a taşındı. LCF basit aksiyomları ve çıkarım kuralları, yüksek dereceli mantık için doğru olacak şekilde değiştirildi ve daha sonra teoremi kanıtlayan altyapı (dönüşümler, fonksiyonlar, fonksiyoneller, alt hedef paketi vb.) doğru çalışacak şekilde değiştirildi [31].

### 3.2.2 Temel tanımlayıcı ilkeler

HOL sistemi, LCF'den farklı olarak, teorileri geliştirmenin birincil yöntemi olarak aksiyom postülasyonundan ziyade tanımı vurgular. Yüksek dereceli mantık, birçok matematiksel nesnenin (sayılar, listeler, ağaçlar vb.) tamamen tanımlayıcı bir gelişimini mümkün kılar ve destekler.HOL tarafından sağlanan tanımlayıcı ilkeler 1980'lerde gelişmiştir. Başlangıçta, sabitler sadece $c = t$ formundaki denklemler ile tanımlanabilirdi; burada $c$ yeni bir isim ve $t$ kapalı bir terimdir. Türler, Mike Fourman tarafından tasarlanan ve yeni türlerin mevcut türlerin boş olmayan alt kümeleri olarak tanıtılabileceği bir şema ile tanımlanabilir. Daha sonra ICL'deki HOL kullanıcıları olan Roger Jones ve arkadaşları esnek özellikli sabitlerin kullanımını önerdi. Bu, $c_1, \dots, c_n$ sabitlerinin, $\exists\, x_1 \dots x_n . P(x_1 \dots x_n)$ olduğu sürece, bir $P(c_1, \dots, c_n)$ özelliğini sağlayan bir tanım ilkesine göre uygulanmıştır.Bu prensibe HOL'da sabit spesifikasyonu denir.

Tanımlama ilkelerinin ilk versiyonları doğru görünüyordu, ancak Mark Saaltink ve Roger Jones bağımsız olarak, bir tanım yaparken tutarlılığın korunmak zorunda olmamasının hatalı olduğunu fark ettiler. Tanımlama ilkelerine yan koşullar ekleyerek problemleri çözmek kolaydı. DSTO Australia'nın desteğiyle Dr. Andrew Pitts, HOL'un tanım ilkelerini doğrulamak üzere görevlendirildi. O da tutarsızlık oluşturmadıklarına dair formal olmayan kanıtlar üretti [31].

### 3.2.3 Türetilmiş tanımlayıcı ilkeler

Basit yerleşik tanımlayıcı ilkeler düşük seviyelidir, ancak yüksek düzeyde türetilmiş ilkeler ML'de programlanabilir. Bunlar, yeni bir sabitin veya türün olmasını isteyen bir özelliği alır ve sonra özelliğe sahip sabitleri ve/veya türleri otomatik olarak tanımlar. Örneğin, ilk türetilmiş sabit tanım prensibi, ilkel özyineleme teoremini somutlaştırarak kullanıcı tarafından sağlanan herhangi bir basit özyinelemeli denklemi gerçekleştirmek için aritmetik fonksiyonları tanımlamıştır.

Milner, Monahan ve Paulson tarafından LCF için geliştirilen fikirlerden yola çıkarak Melham bir yaklaşım ortaya koydu. Özyinelemeli veri türlerinin tanımlarını basit tanımlara dönüştüren, daha sonra otomatik olarak veri türü için doğal indüksiyon ve basit özyineleme ilkelerini türeten bir tür tanımlama ilkesi uyguladı.

Bu gelişme HOL algısını tamamen bir donanım doğrulama sistemi olmaktan genel amaçlı kanıt asistanı olarak değiştirmeye en çok katkıyı sağlamıştır. Örneğin, dilleri özyinelemeli soyut sözdizimi ağaçları ve basit özyinelemeli semantik işlev tanımlayarak HOL içine gömmek çok daha kolay hale geldi. Başka bir türetilmiş tanımlayıcı ilke, endüktif olarak tanımlanmış ilişkilerin bir geçiş sistemi tarafından belirlenmesine ve daha sonra otomatik olarak bir kural indüksiyon taktiğinin oluşturulmasına izin verir.Bu işlemsel anlambilimin HOL içinde kolayca tanımlanmasını sağlar. Konrad Slind tarafından geliştirilen, fonksiyonların genel özyinelemeli tanımlarını yapmak için Isabelle/HOL'da da çalışan güçlü bir araç ve bölüm türleri oluşturmak için en az iki bağımsız paket de dahil olmak üzere diğer türetilmiş tanımlayıcı ilkeler de uygulanmıştır [31].

### 3.2.4 Sadeleştirme

Cambridge LCF, terimi yeniden yazma ve formül basitleştirme ile ayrı ayrı ilgilenen güçlü bir sadeleştiriciye sahipti. HOL'da, boolean terimleri formüllerin rolünü oynadı. Bu nedenle ayrı bir sözdizimsel formül sınıfına ihtiyaç duyulmadı, formül ve terim sadeleştirme için ayrı araç setlerine artık gerek yoktu. Gordon, Paulson'un sadeleştiricisini HOL'da kullanmak için değiştirdiğinde, kodunun koşul yönetimi parçalarını yapmak için çalışmadı, bu nedenle HOL koşullu sadeleştirme olmadan sona erdi. LCF'de koşullu sadeleştirme esas olarak alan teorisinin alt öğesinden kaynaklanan tanımlanabilirlik ve katılık varsayımlarını yönetmek için kullanılmıştır. Gordon'a göre bu tür varsayımlar HOL'da ortaya çıkmazdı. Ancak bunun önemini sonradan anladı, Paulson'un sadeleştiricisi HOL uygulamaları için çok yararlı olabilirdi ve birçok insanı düşük seviyeli kanıtlardan kurtaracaktı. Yıllar boyunca birkaç kişi, HOL'a isteğe bağlı eklentiler olarak koşullu sadeleştirme paketleri ile katkıda bulunmuştur. Ancak koşullu basitleştirme sistemin çekirdeğine sonradan eklenmiştir. Artık yeniden yazma, dönüşüm ve karar prosedürlerini tek bir araca entegre eden yeni bir sadeleştiricinin parçası olmuştur [31].

## 3.3 HOL Versiyonları

HOL sistemi her zaman açıktı ve birçok insan gelişimine katkıda bulundu. Birçok grup, esasen sıfırdan başlayarak sistemin kendi versiyonlarını oluşturmuştur. Bunun iyi ve kötü yönleri vardı: çalışmayı çoğaltmak için çaba harcanması ve hangi sürümün kullanılacağı konusunda karışıklık yaşanabilmesi gibi, ancak diğer taraftan kötü tasarım kararları düzeltilmişti ve Mizar modu gibi yeni fikirler dahil olma fırsatı yakaladı. HOL'un en son sürümlerinde Isabelle, PVS ve Mizar gibi diğer başarılı sistemlerden fikirler yer alıyor [31].

### 3.3.1 HOL88

HOL sistemi yaklaşık olarak 1988'de stabil ve istikrarlı hale geldi. Daha sonra HOL88 adı verilen çeşitli değişiklikleri ve geliştirmeleri birleştiren yeni bir sürüm yayınlandı. HOL'u dokümante etmek için DSTO Avustralya'dan ve Franz Lisp'den Common Lisp'e taşınması için ise Hewlett Packard'dan destek alındı. HOL'un mevcut sürümleri ve belgeleri herkese açıktır ve internette bulunmaktadır [31].

### 3.3.2 HOL90

1980'lerin sonlarında Calgary Üniversitesi'nden Graham Birtwistle, Standard ML'de HOL'u yeniden hayata geçirmek için bir proje başlattı. Çalışma, Konrad Slind tarafından Birtwistle yönetiminde ve Cambridge'deki HOL grubunun işbirliğiyle yapıldı. HOL90 adı verilen sistem ilk olarak 1990 civarında yayınlandı. Eski kod tabanlı HOL88'e rasyonalizasyon getirdi ve önemli bir performans iyileştirmesi sağladı. 1990'lı yıllarda Slind, AT&T Bell Laboratories'den Elsa Gunter ile işbirliği yaparak HOL90'ı geliştirmeye devam etti. HOL90 artık tüm dünyada kullanılan HOL'ün ana versiyonudur, ancak HOL88 kullanıcıları hala devam etmektedir [31].

### 3.3.3 ProofPower

ICL, HOL90'ın geliştirilmesine paralel olarak ProofPower olarak adlandırılan kendi ticari HOL versiyonunu yarattı. Bu versiyonda şirket içi ve ticari kullanım hedeflenmiştir, özellikle de güvenlik uygulamaları için. ProofPower diğer HOL sistemleriyle tam olarak aynı mantığı destekler, ancak hedeflenen uygulamaların ihtiyaçlarını karşılamak için geliştirilmiş farklı bir kanıt altyapısına sahiptir. Örneğin Z notasyonu için özelleştirilmiş teorem kanıt desteği ve Z - SPARK Ada uyum notasyonu için bir doğrulama koşulu oluşturucu gibi ek özelliklere sahiptir [31].

### 3.3.4 Isabelle/HOL

HOL'un yanı sıra meta dili olarak ML ile birkaç LCF tarzı kanıt asistanı daha geliştirilmiştir. Bazı durumlarda başlangıç noktası olarak LCF kodu kullanılmıştır. Bunların bazıları yapı analizleri için bir kanıt sistemi, Nuprl ve Martin Löf'ün tür teorisi için bir kanıt sistemidir. Geliştirilen asistanlar Milner'ın LCF metodolojisini çok farklı mantıklara uyguladılar.Paulson, LCF tarzı sistemlere sistematik bir uygulama metodolojisi sağlamaya çalışmak için, genel kanıtlayıcı Isabelle'i geliştirdi. Isabelle, nesne mantıklarına ait ispat kurallarının deklaratif olarak tarif edilebildiği bir metalojik sağladı. LCF ve HOL'da, kurallar ML programları olarak temsil ediliyordu. Yani, kurallar belirtilmek yerine uygulanıyordu.İlk bakışta Isabelle, HOL gibi bir kanıt araçları koleksiyonu sunuyor gibi gözükse de, çalışma şekilleri oldukça farklıydı.Metalojik kurallar, daha yüksek mertebe birleştirmeye dayalı bir meta çıkarsama kuralı kullanılarak oluşturulur. HOL'daki ileriye ve geriye dönük kanıtlar Isabelle'deki özel kural oluşturma vakalarına karşılık gelir. Bununla

birlikte, Milner'ın teoremlerin yalnızca izin verilen kuralların izin verilen kombinasyonları ile elde edilmesini sağlamak için ML'nin soyut türlerini kullanma hakkındaki ana fikri korunur ve metalojik seviyeye yükseltilir. Paulson tarafından Isabelle için geliştirilen nesne mantıklarından biri HOL mantığıydı. Ortaya çıkan Isabelle / HOL sistemi, Isabelle'in farklı kanıt altyapısı nedeniyle orijinal HOL sisteminden biraz farklı bir görünüm ve anlayışa sahiptir. Özelleştirilebilir sadeleştiricisi ve birinci dereceden teoremi kanıtlayan araçları sayesinde HOL'dan daha iyi genel mantık otomasyonu sağlar. Bazı HOL kullanıcıları Isabelle versiyonuna geçiş yapmıştır [31].

### 3.3.5 HOL Light

Sonraki zamanlarda John Harrison ve Konrad Slind diğer çalışmaların yanı sıra basit sabitleri, aksiyomları ve çıkarım kurallarını rasyonelleştirmek için HOL tasarımını tamamen yeniden çalıştılar. Örneğin, mantık başlangıçta yapıcı olarak alınır ve ancak önemli kanıt altyapısı tanımlandıktan sonra yapıcı olmayan ilkeler eklenir. HOL'un bu yeni sürümü "HOL Light" olarak adlandırıldı. Caml Light'da uygulandı ve kişisel bilgisayarlar gibi mütevazı platformlarda çalışır durumdaydı. HOL Light Lisp tabanlı HOL88'den daha hızlıdır, ancak Standard ML'nin modern uygulamalarında çalışan HOL90'dan biraz daha yavaştır. HOL Light, kanıt aramayı kanıt denetlemeden ayıran otomatik kanıtlayıcılar da dahil olmak üzere birçok yeni imkanı içermektedir. Ayrıca normal hedef odaklı ispatlama biçimlerinin yanı sıra Mizar modu da sağlar [31].

## 3.4 HOL'un Özellikleri

HOL bir dizi temel özellik ile karakterize edilir. Basit bir temel mantık, LCF tarzı tam genişleme, artan kanıt stilleri çeşitliliği desteği ve kullanıcı tarafından sağlanan teoriler ve kanıt araçlarının büyük bir topluluğu [31].

### 3.4.1 Temel mantık

Sadece dört ayrı basit terim türü vardır: değişkenler, sabitler, fonksiyon uygulamaları ve λ-soyutlamalar. Standart teknikler kullanılarak, diğer yararlı notasyonlar bunların üzerinde ayrıştırıcı ve okunabilir bir yapı tarafından desteklenir. Örneğin, $Qx.t$ değerleri bir soyutlama sabitinin uygulaması olarak kodlanır. Yani $Q(\lambda x.t)$ ve

yerel değişkene atama yapılan $let\ x = t_1\ in\ t_2$ , (Landin'e göre) ile $(\lambda x.t_2)t_1$ eşdeğerdir. Bu nedenle, tüm değişken bağlanma, λ-bağlamaya indirgenir.

Ek notasyonları, koşullu terimler, soyut kümeler, sınırlı nicelikler, tuple notasyonu ve tuple değişken bağlama (örn. , $\lambda(x,y).t$ gibi gösterimler türetilmiş formlar olarak kabul edilir (yani sözdizimsel şeker - syntactic sugar'") ). Karmaşık notasyonları tercüme etme stratejisi oldukça işe yaradı. Tüm terimleri işlemek için tasarlanmış prosedürler sıklıkla kullanılan dört grubu (değişkenler, sabitler, uygulamalar ve soyutlamalar) dikkate alıyor, böylece prosedürlerin kısa ve anlamsal olarak şeffaf olması sağlanıyordu. Öte yandan her şeyin sadece değişkenlere, sabitlere, uygulamalara ve soyutlamalara kodlanması, yapıların doğal bileşenlerinin hesaplanmasını maliyetli hale getirir. Arayüzleri ise oldukça karmaşık hale getiriyordu. Karmaşık gösterimleri basite indirgemenin belirgin bir tehlikesi, kullanıcının basılı olarak gördüğü şeyin çıkarım mekanizmalarının gerçekte işlediğinden uzak olabilmesidir. Ayrıştırma ve okunabilir arabirimdeki hatalar, çıkarım kurallarındaki hatalar kadar tehlikeli olabilir. Tehlikeleri en aza indirmeye yönelik bir yaklaşım, bildirimsel bir girdiden ayrıştırıcılar ve okunurluk üreten güvenilir araçlar kullanmaktır. Richard Boulton'ın geliştirdiği ClaReT' bu araçlara örnek olarak gösterilebilir [31].

### 3.4.2 Tam genişleme

Teorinin kanıtlanmasına yönelik LCF yaklaşımı tüm ispatların basit çıkarımlar dizisine genişletilmesiyle tamamen geniş bir yapıya sahiptir. İlk bakışta bunun çok verimsiz göründüğü ve kabul edilebilir performans seviyesinin altında olduğu iddia edilmiştir. Bununla birlikte etkili türetilmiş kuralları ve taktikleri programlamak için bütün bir programlama metodolojisi gelişmiştir. Örneğin, belirli problem sınıfları için karar prosedürleri geliştirilmiştir. Bunlar ilkel çıkarımlara genişlese de şaşırtıcı derecede etkilidirler. HOL, kullanıcıların birçok uygulama için yeterince hızlı ve güçlü bulduğu totoloji kontrolü ile aritmetik alt kümesi elde edilmesini sağlayan bu tür karar prosedürlerine sahiptir. Verimliliği artıran önemli bir teknik, birinci dereceden mantıkta türetilmiş çıkarım kurallarını (örneğin teorem şemaları) olması gereken tek teorem olguları olarak kodlayarak daha yüksek dereceli mantığın ifade gücünden yararlanmaktır. Böylelikle kanıt adımlarının dizilerinin zaman alıcı tekrarlarından, genel bir teoremi bir kez kanıtlayıp daha sonra birçok kez

örnekleyerek kaçınılabilir. Başka bir programlama tekniği kanıt aramasını kanıt kontrolünden ayırmaktır. Bir ML programı, hatta harici C kodlu bir totoloji denetleyicisi veya bir cebir sistemi kanıt bulmak için kullanılabilir. Sonuç HOL mantığı içindeki formal çıkarım ile doğrulanır. Bu ayrımı paketlemenin ve otomatikleştirmenin bir yolu Boulton'ın tekniğidir.

Son yirmi yılda taktiklerin uygulanması ve kullanımında birçok gelişme olmuştur. Milner'ın orijinal konseptinin onları desteklemek için yeterince genel olması ise dikkat çekicidir [31].

### 3.4.3 Kanıt stillerinin çeşitliliği

HOL'un mevcut sürümleri ileriye dönük kanıtı ve hedefe yönelik ispatı desteklemektedir. Bunu Paulson tarafından Cambridge LCF için sağlanan yığın tabanlı bir alt hedef paketi aracılığıyla gerçekleştirmektedir. Diğer kanıt stilleri ise kütüphaneler olarak mevcuttur. HOL'un sabit mantığa sahip bir araç olarak, ancak giderek artan çeşitlilikte yerleşik kanıt stillerine sahip olması beklenmektedir. Örneğin son zamanlarda, doğal dil benzeri bir ders kitabı tarzında ifade edilen argümanları geliştirerek kanıtların oluşturulduğu Mizar sistemi çok fazla heyecan yaratmıştır. Bu ileri stil bazı durumlarda daha iyi olmakla beraber diğer durumlar içinse hedef odaklı bir stildir. Özellikle, hedefe yönelik bir stil kanıtın özelleştirilmiş algoritmalar aracılığıyla üretilebildiği karmaşık yapıları (örneğin mikroişlemciler) doğrulamak için işe yararken, ileri Mizar stili genel matematiksel teoriler (örneğin cebir, fonksiyonel analiz, topoloji) geliştirmek için de daha iyi bir yöntem olarak görünmektedir. Kayan nokta doğrulama, kriptografi, sinyal işleme gibi pek çok uygulama, probleme özel algoritmalar aracılığıyla uygulanacak genel bir matematik altyapısı gerektirir. Dolayısıyla teorileri geliştirmek için bir Mizar stili ve bunları uygulamak için hedefe yönelik bir stil kullanma seçeneği sunmak yararlıdır. Bu amaçla John Harrison HOL'a Mizar modu desteği ekledi [31].

### 3.4.4 Kütüphaneler ve kullanıcılar tarafından sağlanan katkılar

HOL, teorilerin ve diğer yardımcı program kodunun paylaşılmasını sağlamak için temel bir kütüphane tesisine sahiptir. Bu sayede bağımsız HOL gelişmeleri için bir dosya yapısı ve dokümantasyon formatı sağlar. Yıllar içinde dünyanın dört bir yanından kullanıcılar tarafından birçok kütüphane sağlanmıştır. Çekirdek HOL

sistemi oldukça stabil kalmasına rağmen, kütüphane seti büyüdü. HOL ile halihazırda dağıtılan kütüphaneler arasında aritmetik ve totoloji karar prosedürleri, grup teorisinin gelişimi, endüktif olarak tanımlanmış ilişkileri destekleyen bir paket, tamsayılar ve reel sayılar teorileri (çekirdek sistemde sadece doğal sayılar önceden tanımlanmıştır) n-bit kelimelerin teorileri, karakter dizeler, genel listeler ve kümeler, iyi sıralı kümeler (sonsuz indüksiyon vb.), UNITY ve Hoare tarzı programlama mantığı desteği, donanım doğrulama ve program iyileştirme araçları yer almaktadır.

Kütüphanelerin oldukça gelişmiş olması ve yüksek bir standartta dokümante edilmesi amaçlanmıştır. Ayrıca HOL ile minimum kalite kontrolüne tabi olan "katkılar" da dağıtılmaktadır. Halen paylaşılan katkılar arasında CSP izleme teorisi, ilişkisel-değişmeli birleşim için kanıt araçları, Boyer ve Moore'un otomatik kanıt buluşsal yöntemini uygulayan taktikler, sıralı bir çarpanın kanıtı (HOL için bir kıyaslama olarak kullanılmaktadır), sonsuz durum otomat teorileri, koşullu ifadeleri sadeleştirme kuralları, sabit noktaların tanımı ve Scott indüksiyonunun türetilmesi, dilin yüksek mertebeli mantığa yerleştirilmesini destekleyen araçlar, türetilmiş bir kural olarak Knuth-Bendix tamamlama, özyinelemeli türler paketinde çeşitli iyileştirmeler (örneğin iç içe ve karşılıklı özyineleme için), Standart ML tanımının önemli bir kısmının formalizasyonu, bir veritabanı sorgu dilinin HOL'a uygulanması, denklem kümelerinden verimli dönüşümler oluşturmak için bir derleyici, kısmi fonksiyonları destekleyen teoriler ve HOL teorileri için köprü bağlantılı bir kılavuz yer almaktadır.

Yapım şekli göz önüne alındığında, HOL'da nispeten az hata vardı. Sistem tamamen açık olduğundan, erken hata düzeltmeleri genellikle cesur kullanıcılar tarafından yapıldı ve daha sonra gelecekteki sürümlere dahil edilmek üzere Gordan'a posta ile iletildi [31].

# 4. COLLATZ KONJEKTÜRÜ

## 4.1 Lothar Collatz

Lothar Collatz, 6 Temmuz 1910'da Almanya'nın Vestfalya bölgesinde bulunan Arnsberg şehrinde doğdu. Stettin şehrinde büyüdü [32]. 1928'de Greifswald Üniversitesi'nde eğitime başladı [33]. 1933'e kadar Göttingen, Münih ve Berlin'de okudu. Zamanın en önde gelen matematikçilerinden ve bilim adamlarından dersler aldı. 1933'te Staatsexamen'ı matematikte Richard von Mises ve fizikte Erwin Schrödinger'ın yönetiminde tamamladı. Siyasi iklim o sırada hızla değişiyordu ve Mises ile yapılan sınav esas olarak gelecekteki bir tez planlarını tartışmaktan ibaretti. Birkaç hafta sonra von Mises Almanya'dan ayrıldı [32]. Lothar Collatz esasında von Mises'ın danışmanlığında olsa da 1935 yılında Berlin Üniversitesi'nde resmi olarak Alfred Klose ve Erhard Schmidt danışmanlığında doktorasını tamamladı [32][33]. Habilitasyonunu Karlsruhe'de Theodor Pöschl ve Wilhelm Quade altında tamamlandı. Ardından 1935-1943 yılları arasında Karlsruhe'deki Institut für Technische Mechanik'te misafir öğretim görevlisi olarak ve Darmstadt'taki Institut für Praktische Mathematik'te çalıştı. 1943'te Technische Hochschule Hannover'de matematik profesörü oldu [32]. 1952'de Hamburg Üniversitesi'ne geçti ve burada Uygulamalı Matematik Enstitüsü'nü kurdu [33]. 1978'de emekli oluncaya kadar burada kaldı. Ölümüne kadar Hamburg'da fahri profesör olarak aktif kaldı ve burada araştırma yapmaya, öğretmeye, konferanslar düzenlemeye, seyahat etmeye, konuşmalar yapmaya ve matematiğin gelişimine katkıda bulunmaya devam etti. 1978'den ölümüne kadar Aequationes Mathematicae de dahil olmak üzere çeşitli dergilerin editörlüğünü yaptı [32]. Kitapları ve akademik yayınlarının yanısıra, çalışmalarından dolayı çeşitli ödül ve nişanlara layık görüldü [33].

Lothar Collatz kariyerini uygulamalı matematik alanına ve matematiğin uygulanabilirlik aralığını genişletmeye adadı. Matematiğin temel birliğine inanan biriydi ve bu nedenle araştırmasında daha geleneksel uygulamalı tekniklerin yanı sıra saf matematik alanlarını kullanma ve geliştirme konusunda hiçbir endişesi yoktu. İlk

çalışmalarında sonlu fark yöntemleri ve özdeğer problemleri ile bunların teknik problemlere uygulanmasını ele aldı. Ayrıca kendi dahil etme teoremlerinin ilkini de kanıtladı. Bu teoremler ilk olarak literatürde Collatz Teoremi olarak anılsa da, kitaplarında hiçbir zaman onlardan bu şekilde bahsetmedi ve çalışmalarını yalnızca kitapları veya dersleri aracılığıyla bilenler pek çok güzel teoremin gerçekte onunla ilgili olduğunu asla anlayamayacaklardı. Hayatı boyunca teknik problemlere olan ilgisini sürdürdü ve amaçlarından biri modern matematiksel araştırmanın sonuçlarını uygulayıcı mühendis için erişilebilir kılmaktı. Araştırmaları ve alanının 1950'li yıllar civarında genel durumu Eigenwertaufgaben mit technischen Anwendungen, Akademische Verlagsgesellschaft ve Numerische Behandlung von Differentialgleichungen adlı kitaplarında özetlenmiştir [32].

O sıralarda sayısal matematikteki son teknolojiden açıkça memnun değildi ve özellikle hata analizinde araştırmaya çok ihtiyaç duyulan alanlara işaret etti. Burada önemli katkılarda bulundu. Ellili ve altmışlı yılların başındaki çalışmaları, hata analizi, monotonluk yöntemleri, bir ve birkaç boyutta yaklaşım teorisi ve fonksiyonel analizin sayısal analize uygulamaları ile ilgiliydi. Bu sonuçların çoğu bugün hala kullanılmaktadır ve bunların birçoğu Funktionalanalysis und numerische Mathematik adlı iyi bilinen ve geniş çapta tercüme edilmiş kitabında bulunabilir. Lothar Collatz'ın bir alanla ilgilenmeye başladığında, asla vazgeçmeyen bir yapısı vardı. Bu nedenle sonraki yıllarında araştırmasına yukarıdaki tüm alanlarda devam etti ve optimizasyon teorisi ile çatallanma teorisi gibi diğer alanlarda da çalışmalar yaptı [32].

Matematikle birlikte iki tutkusu daha bulunmaktaydı. İlk olarak oyunları ve bulmacaları severdi.1955'te oyunlar üzerine ilginç bir kurs verdi. Bu derste oyunları, yapılarını ve adaletlerini analiz etti. Oyunları kendisi icat etti. Ayrıca ifade edilmesi ve anlaşılması kolay, ancak çözülmesi oldukça zor olan Collatz Konjektürü'nü de ortaya koyan kişidir. Diğer tutkusu ise geometrik desenlerdi. Onları analiz etmek için kendi sistemini geliştirdi. İlk makalelerinden birinde, mutlak değerler aracılığıyla örtük olarak tanımlanan desenleri işlemiştir. Not ettiği yeni kalıpları araştırmaktan hoşlanıyordu. Materyallerinin bir kısmını sonraları yayınlamaya başlamıştı [32].

Muazzam miktarda enerjisi vardı. Çok sayıda konferans verdi. Bu konferanslardan biri de kısa sürede bir olaya dönüşen Collatz Konjektürü konusundaydı. Hamburg'daki enstitüsünü fikir alışverişinde bulunan, birbirine sıkı sıkıya bağlı bir grup haline nasıl getireceğini bilen biriydi. Seyahatlerinde yeni fikirler aradı ve bunları dünya çapında arkadaşları ve meslektaşları ile paylaştı. Hayattan zevk alan, ilgi çekici ve sıcak bir kişiliğe sahipti ayrıca pek çok hobisi vardı. Sonuç olarak gittiği her yerde hoş bir misafirdi. Bir konferans veya geziden sonra fotoğraflarını ve eskizlerini arkadaşları ve katılımcılarıyla paylaşırdı [32].

Profesör Lothar Collatz, 26 Eylül 1990'da 80 yaşında iken Bulgaristan'ın Varna kentinde düzenlenen Bilgisayar Aritmetiği, Bilimsel Hesaplama ve Matematiksel Modelleme Uluslararası Sempozyumu'na katılımı esnasında öldü. Orada bir makale vermişti ve aktif bir katılımcıydı [32][33].

## 4.2 Collatz Konjektürü

### 4.2.1 Ön bilgiler

Lothar Collatz doktorasını tamamladıktan 2 yıl sonra 1937 yılında bir hipotez öne sürdü. Bu hipotez 20. yy'ın başlarından beri matematik çevrelerinde ve uluslararası alanda popüler oldu. Collatz'a göre: "Herhangi bir $n$ pozitif tam sayısından başlayarak $f(n)$ fonksiyonuna sokulan $n$ iterasyonları daima 1 rakamına ulaşır.". Fonksiyona verilen n sayısı tek ise sayı 3 ile çarpılır ve 1 eklenir, n sayısı çift ise sayı 2'ye bölünür. 1 rakamına ulaşana kadar iterasyon devam eder [34].

Hipotezin ispatı için çok sayıda çalışma yapıldı. Önemli matematikçiler uzun süreler boyunca üzerinde çalıştılar. Hipotezi ispatlayana Paul Erdős $500, Bryan Thwaites £1000, Harold Scott MacDonald Coxeter $50 ve karşı örneğini bulup hipotezi çürütene ise $100 para ödülü vaadinde bulunmuşlardır. Macar matematikçi Paul Erdős "Matematik bu tip problemler için henüz hazır değil." ifadesini kullanmıştır [35].

### 4.2.2 Problemin tanımı

**Tanım:** Herhangi bir $n$ pozitif tam sayısından başlayarak $f(n)$ fonksiyonuna sokulan $n$ iterasyonları daima 1 rakamına ulaşır. Fonksiyona verilen $n$ sayısı tek ise sayı 3 ile çarpılır ve 1 eklenir, $n$ sayısı çift ise sayı 2'ye bölünür. 1 rakamına ulaşana kadar iterasyon devam eder [34].

Bu tanımda yer alan fonksiyon aşağıdaki gibi ifade edilmektedir;

- $f : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$
- $f_0 = n\,,\ n \neq 0$
- $f_{i+1} = f_i / 2 \quad ,\ f_i\ çift\ sayı\ ise$
- $f_{i+1} = 3f_i + 1\,,\ f_i\ tek\ sayı\ ise$ [36].

### 4.2.3 Collatz fonksiyonu

Hipotez $n$ pozitif tam sayısının işleme girdiği $f(n)$ fonksiyonu detaylarına açıkça yer vermektedir. Bu fonksiyon matematiksel notasyonda aşağıdaki şekilde gösterilebilir;

$$C : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$$

$$C(n) = \begin{cases} \dfrac{n}{2}, & n => \text{ n \% 2} = 0 \\ 3n + 1, & n => \text{ n \% 2} = 1 \end{cases}$$

$$F(n) = \left\{ n, C(n), C\big(C(n)\big), C\left(C\big(C(n)\big)\right) \ldots . \right\}$$

$$= \{ n, C(n), C^2(n), C^3(n) \ \ldots . \} \qquad [37].$$

Fonksiyona göre işleyiş mantığı aşağıdaki gibi özetlenebilir;

1. Herhangi bir $n$ pozitif tam sayısı seçilir.
2. Eğer $n$ sayısı çift ise $n$ 2'ye bölünür.
3. Eğer $n$ sayısı tek ise $n$ 3 ile çarpılır ve 1 eklenir.

4. Fonksiyon sonucunda çıkan sayı 1'e ulaşana kadar iterasyon devam eder [38].

### 4.2.4 Konjektür notasyonu

**Konjektür 1 (Collatz Konjektürü) :**

Her $n \in \mathbb{Z}^+$ için, $C^k(n) = 1$ olan bir $k \in \mathbb{Z}^+$ vardır.

$\forall n \in \mathbb{Z}^+ \ , \ni C^k(n) = 1 \ \ \exists k \in \mathbb{Z}^+$ [37].

Collatz Konjektürü'nün sade ve anlaşılır bir yapısı bulunmaktadır. Hipotezin tanımı, fonksiyonu ve özellikleri net olarak belirtilmiştir. Buradaki ispat bekleyen konu ise "Bu tanım tüm pozitif tam sayılara uygulandığında sonuç 1'e ulaşır mı?" veya "Bu ifade tüm pozitif tam sayılar için doğru, geçerli midir?" sorusudur.

2020 yılı itibariyle, konjektür bilgisayar tarafından $2^{68} \approx 2.95 \times 10^{20}$ 'ye kadar olan tüm başlangıç değerleri için kontrol edilmiştir [39]. Dolayısıyla karşı örnek arayan birisi yaklaşık 300 kentilyondan itibaren çalışmaya başlayabilir [40].

$2^{68}$ 'e kadar tüm başlangıç değerleri kontrol edilmiş ve karşı bir örneğe rastlanmamıştır. Her defasında fonksiyona giren başlangıç değeri 1 rakamına ulaşarak varsayımı sağlamıştır. Bu nedenle Collatz Konjektürü; "Doğru olduğu düşünülen ancak doğruluğu henüz kanıtlanmamış veya reddedilmemiş matematiksel bir varsayımdır" [41].

### 4.2.5 Senaryolar ve realizasyonları

Collatz Konjektürü'nün sayılar üzerindeki realizasyonlarına, fonksiyonun çalışma mantığına, fonksiyon iterasyonlarına ve işlem adım sayılarına örnekler üzerinden açıklama yapalım. Bununla birlikte çeşitli senaryoları inceleyelim.

Öncelikle hesaplamanın yapılacağı fonksiyonu belirtelim;

$$C : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$$

$$C(n) = \begin{cases} \frac{n}{2}, & n \rightarrow n \equiv 0 \pmod 2 \\ 3n+1, & n \rightarrow n \equiv 1 \pmod 2 \end{cases}$$

Örneğin 5 rakamını fonksiyon üzerinden adım adım hesaplayalım;

1. 5 tek sayı → $3n+1$ işlemini uygula → sonuç 16
2. 16 çift sayı → $n/2$ işlemini uygula → sonuç 8
3. 8 çift sayı → $n/2$ işlemini uygula → sonuç 4
4. 4 çift sayı → $n/2$ işlemini uygula → sonuç 2
5. 2 çift sayı → $n/2$ işlemini uygula → sonuç 1

1 (Fonksiyon iterasyonu 1 rakamına ulaştı, işlemi sonlandır.)

Bu örnekte seçilen 5 rakamı fonksiyon üzerinden işleme verildi. Toplamda 5 adımda 1 rakamına ulaştı.

Şimdi de 1'den 10'a kadar olan sayılar için hesaplanan serileri inceleyelim;

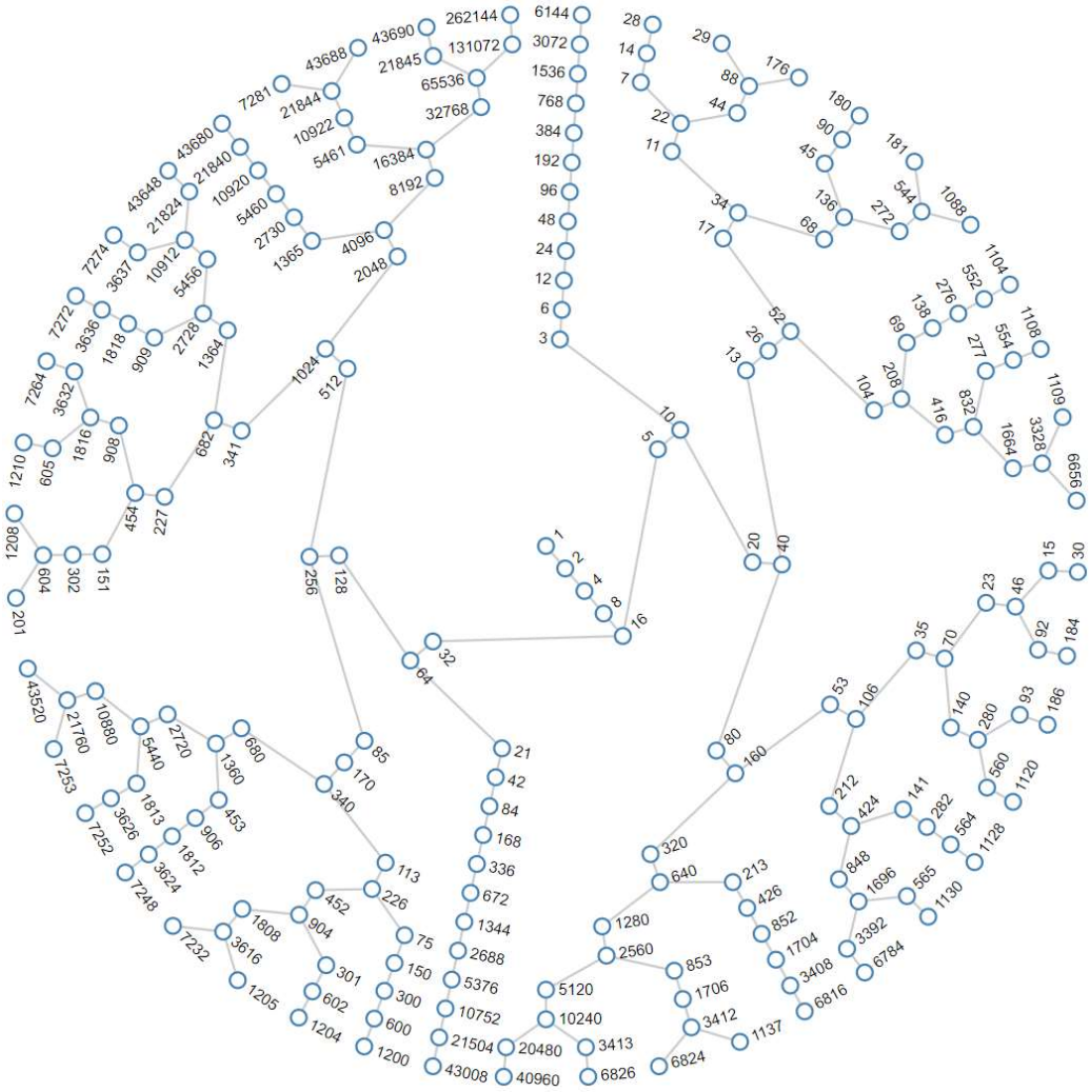
$C(1) = \{1\},$
$C(2) = \{2,1\},$
$C(3) = \{3,10,5,16,8,4,2,1\},$
$C(4) = \{4,2,1\},$
$C(5) = \{5,16,8,4,2,1\},$
$C(6) = \{6,3,10,5,16,8,4,2,1\},$
$C(7) = \{7,22,11,34,17,52,26,13,40,20,10,5,16,8,4,2,1\},$
$C(8) = \{8,4,2,1\},$
$C(9) = \{9,28,14,7,22,11,34,17,52,26,13,40,20,10,5,16,8,4,2,1\},$
$C(10) = \{10,5,16,8,4,2,1\}$

İncelenen örnekten de anlaşılacağı üzere Collatz Konjektürü'nde seride çıkan herhangi bir değerin tekrar kontrol edilmesine gerek bulunmamaktadır. Bir sonraki aşamada ise bu örnekteki $n$ sayılarını, oluşturdukları serileri, seride ulaştıkları $maximum$ sayıları, 1 rakamına ulaşana kadar iterasyon işlemlerinin adım sayılarını, ulaştıkları sonuç değerlerinin tablosunu oluşturalım;

**Çizelge 4.1 :** Collatz Konjektürü'nde örnek olarak hesaplanan sayıların tablosu.

| $n$ | $Seri$ | $Max$ | $i$ | $Sonuç$ |
|---|---|---|---|---|
| $1^*$ | $-$ | $-$ | $-$ | 1 |
| 2 | 2 , 1 | 2 | 1 | 1 |
| 3 | 3 , 10 , 5 , 16 , 8 , 4 , 2 , 1 | 16 | 7 | 1 |
| 4 | 4 , 2 , 1 | 4 | 2 | 1 |
| 5 | 5 , 16 , 8 , 4 , 2 , 1 | 16 | 5 | 1 |
| 6 | 6 , 3 , 10 , 5 , 16 , 8 , 4 , 2 , 1 | 16 | 8 | 1 |
| 7 | 7 , 22 , 11 , 34 , 17 , 52 , 26 , 13 , 40 , 20 , 10 , 5 , 16 , 8 , 4 , 2 , 1 | 52 | 16 | 1 |
| 8 | 8 , 4 , 2 , 1 | 8 | 3 | 1 |
| 9 | 9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1 | 52 | 19 | 1 |
| 10 | 10 , 5 , 16 , 8 , 4 , 2 , 1 | 16 | 6 | 1 |

$^*$ 1 rakamı fonksiyonda işleme girdiğinde { 1, 4 , 2 , 1, 4 , 2 … } iterasyonu ile karşılaşılır.




**Şekil 4.1:**Collatz Konjektürü'nde sayıların yörüngelerini gösteren harita grafiği [38].

Collatz Konjektürü hesaplanırken seride çıkan sayılar, başlangıç sayısı ve maksimum sayı arasındaki fark, başlangıç sayısına göre adım sayısının lineer olmaması gibi pek çok düzensizlik dikkat çekicidir. Kümülonimbus bulutu içerisindeki dolu taneleri yeryüzüne düşüne kadar gelişigüzel desenlerle aşağı ve yukarı sürüklenir. Bu benzetmeden yola çıkarak Collatz Konjektürü "*Dolu Taneleri Sekansı*" ya da "*Dolu Taneleri Konjektürü*" olarak da isimlendirilir [42].

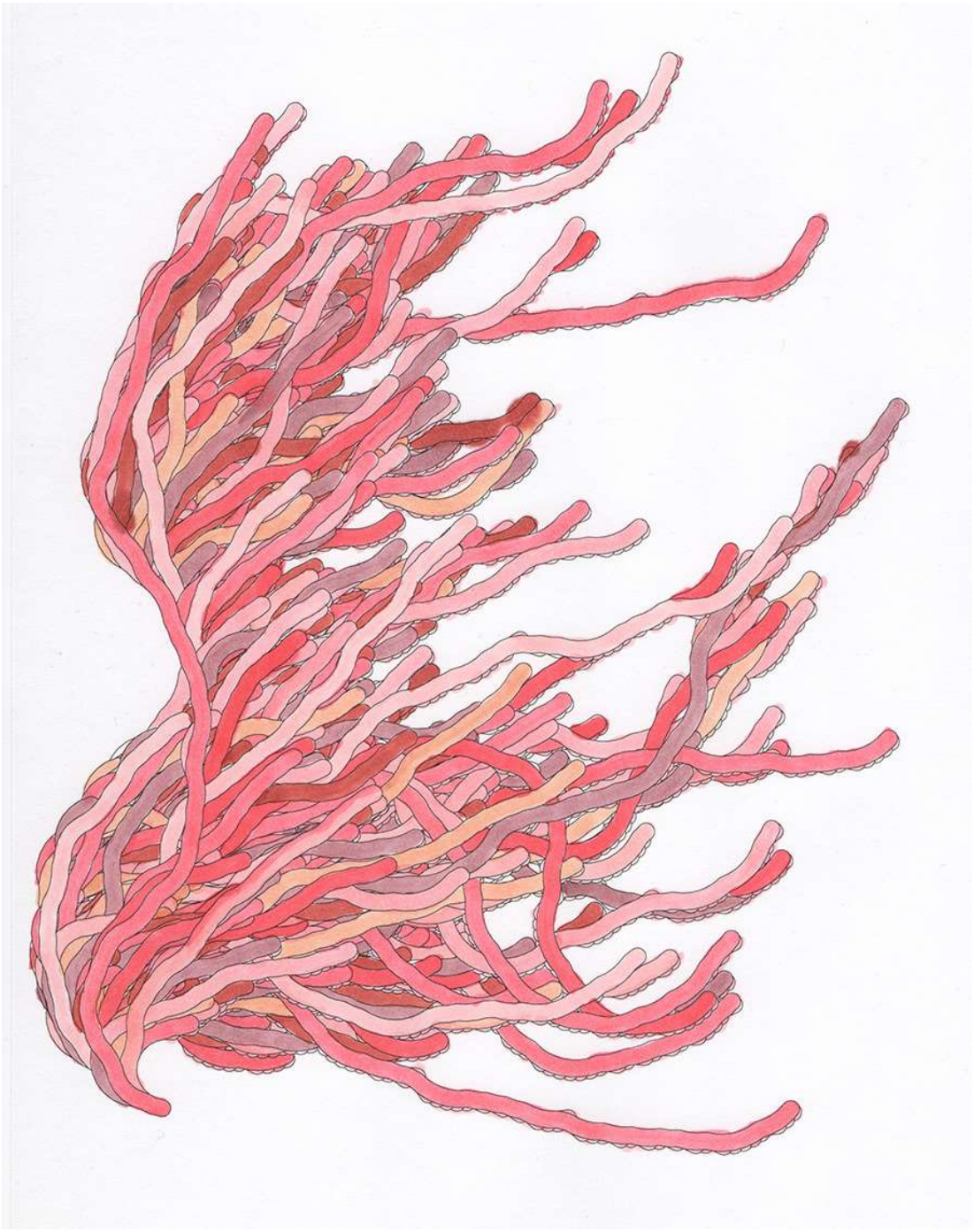
İşlemler görselleştirildiğinde inişli çıkışlı durumlar daha net belirmektedir. Belirli sayı aralıklarında hesaplama yaparken maksimum adım sayısına ulaşan bazı örnekler aşağıda paylaşılmıştır;

- 10'dan az 9 rakam 19 adımda,
- 100'den az 97 sayısı 118 adımda,
- 1,000'den az 871 sayısı 178 adımda,
- 10,000'den az 6,171 sayısı 261 adımda,
- 100,000'den az 77,031 sayısı 350 adımda,
- 1 milyon'dan az 837,799 sayısı 524 adımda,
- 10 milyon'dan az 8,400,511 sayısı 685 adımda,
- 100 milyon'dan az 63,728,127 sayısı 949 adımda,
- 1 milyar'dan az 670,617,279 sayısı 986 adımda,
- 10 milyar'dan az 9,780,657,630 sayısı 1132 adımda,
- 100 milyar'dan az 75,128,138,247 sayısı 1228 adımda,
- 1 trilyon'dan az 989,345,275,647 sayısı 1348 adımda,
- 10 trilyon'dan az 7,887,663,552,367 sayısı 1563 adımda,
- 100 trilyon'dan az 80,867,137,596,217 sayısı 1662 adımda,
- 1 katrilyon'dan az 942,488,749,153,153 sayısı 1862 adımda,
- 10 katrilyon'dan az 7,579,309,213,675,935 sayısı 1958 adımda,
- 100 katrilyon'dan az 93,571,393,692,802,302 sayısı 2091 adımda.

hesaplanabilmektedir [43].

**Çizelge 4.2 :** Belirli aralıklarda maksimum adım sayısına ulaşan örnekler tablosu.

| $n$ | $Başlangıç$ | $Bitiş$ | $Aralık$ | $i$ |
|---|---|---|---|---|
| 9 | $10^0$ | $10^1$ | $10^0 < n < 10^1$ | 19 |
| 97 | $10^1$ | $10^2$ | $10^1 < n < 10^2$ | 118 |
| 871 | $10^2$ | $10^3$ | $10^2 < n < 10^3$ | 178 |
| 6,171 | $10^3$ | $10^4$ | $10^3 < n < 10^4$ | 261 |
| 77,031 | $10^4$ | $10^5$ | $10^4 < n < 10^5$ | 350 |
| 837,799 | $10^5$ | $10^6$ | $10^5 < n < 10^6$ | 524 |
| 8,400,511 | $10^6$ | $10^7$ | $10^6 < n < 10^7$ | 685 |
| 63,728,127 | $10^7$ | $10^8$ | $10^7 < n < 10^8$ | 949 |
| 670,617,279 | $10^8$ | $10^9$ | $10^8 < n < 10^9$ | 986 |
| 9,780,657,630 | $10^9$ | $10^{10}$ | $10^9 < n < 10^{10}$ | 1132 |
| 75,128,138,247 | $10^{10}$ | $10^{11}$ | $10^{10} < n < 10^{11}$ | 1228 |
| 989,345,275,647 | $10^{11}$ | $10^{12}$ | $10^{11} < n < 10^{12}$ | 1348 |
| 7,887,663,552,367 | $10^{12}$ | $10^{13}$ | $10^{12} < n < 10^{13}$ | 1563 |
| 80,867,137,596,217 | $10^{13}$ | $10^{14}$ | $10^{13} < n < 10^{14}$ | 1662 |
| 942,488,749,153,153 | $10^{14}$ | $10^{15}$ | $10^{14} < n < 10^{15}$ | 1862 |
| 7,579,309,213,675,935 | $10^{15}$ | $10^{16}$ | $10^{15} < n < 10^{16}$ | 1958 |
| 93,571,393,692,802,302 | $10^{16}$ | $10^{17}$ | $10^{16} < n < 10^{17}$ | 2091 |



**Şekil 4.2:** Collatz Konjektürü'nün suda dalgalanan deniz yosunu illüstrasyonu [44].

Şekil 4.2'de yer alan illüstrasyon Alex Bellos ve Edmund Harriss'in Amerika Birleşik Devletleri'nde Visions of the Universe, Birleşik Krallık'ta ise Visions of Numberland ismiyle yayınlanan kitabından alıntıdır. Birçok farklı başlangıç noktasından oluşan sayıların tamamı bitkinin kökünde 1 rakamına ulaşmaktadır. Bitkinin köküne ulaşana kadar uçlar çok sayıda noktada aynı dallarda buluşmaktadır.

Sayı dizisinde 1 rakamına ulaşırkan aynı seriyi takip eden sayılardan oluşmaktadırlar. İllüstrasyon bu yönleriyle Collatz Konjektürü'nü simgelemektedir.

Günümüzde Collatz Konjetürü'nün hesaplanmasına bilgisayarlar tarafından devam edilmektedir. California Üniversitesi'nde BOINC ve Science United projelerinin çalışmaları yürütülmektedir. Bu projeler Ulusal Bilim Vakfı tarafından desteklenmektedir. BOINC programı, bilgisayarınızı kullanarak en son bilim araştırmalarına yardımcı olmanızı sağlar. Programı bilgisayarınıza indirip, kurmanız yeterlidir. Bilgisayarınızda çalışan BOINC uygulaması, bilimsel hesaplama işlerini indirir ve bunları arka planda görünmez bir şekilde çalıştırır. Yaklaşık 30 bilim projesi BOINC programını kullanmaktadır [45]. Collatz Konjektürü de bu bilimsel araştırma projelerinden birisidir. İnternete bağlı bilgisayarlar tarafından Collatz Konjektürü'nün matematik araştırmaları ve hesaplamaları yapılmaktadır. BOINC projelerinin bir kısmı Berkeley'de yer alırken, Collatz Konjektürü ise Illinois'de yerleşiktir. Özel olarak yönetilen bu BOINC projesinin temel amacı Collatz Konjektürü'nü çürütmektir. Collatz Konjektürü projesi eşlik dizisi optimizasyonunu kullanır. Program işletim sistemi üzerinde çalışır, CPU ve grafik kartlarını kullanır [46].

Proje kapsamında; proje sunucularının durumu, hesaplama statüleri altında gönderime hazır olan, işlemi devam eden, silinmesi beklenen task adetleri, validasyon bekleyen, asimilasyon bekleyen, silinmesi beklenen iş birimi adetleri, saat bazında iş yığını geçişlerinin yanısıra bilgisayar ve kullanıcı adetleri, son 24 saat içerisinde kaydı gerçekleşmiş olanların adetleri ile Collatz Sieve uygulamasına ait task adetleri gibi bilgiler paylaşılmaktadır. Liderlik tablosu altında zirvedeki katılımcılar, takımlar ve bilgisayarların bilgilerini içeren listeler yer almaktadır. En iyi CPU performansları ile en prodaktif GPU modelleri de listelenmektedir.

Projeye ait web sayfasında önemli istatistiksel bilgiler de bulunmaktadır. Bunlardan biri Collatz Konjektürü hesaplamarında, maksimum adım sayısına göre bugüne kadar alınan en iyi sonuçlardır. Güncel bilgi esas alındığında listenin birinci sırasında 2,968 adımla 7,219,136,416,377,236,271,195 sayısı bulunmaktadır [47]. Diğer önemli istatistiksel bilgi ise ulaşılan en yüksek sayıdır. Günlük olarak, günün en yüksek sayısı 1 rakamına ulaşılabildiği adım sayısı ile listelenir. Yine güncel bilgi

esas alındığında listenin birinci sırasında 2516 adımla 9,033,093,293,652,500,451,647 sayısı bulunmaktadır [48].

### 4.2.6 İterasyonlar

Collatz Konjektürü'nde sayıların hesaplama işlemleri, konjektürün tanımlı olan fonksiyonu sayesinde yapılmaktadır. Fonksiyona giren sayı işlemler sonucunda 1 rakamına ulaşana kadar iterasyon devam eder. Bu bölümde bilenen iterasyonlar ile farklı uygulama türleri incelenecektir.

İlk olarak Collatz Konjektürü'nde hesaplanmak üzere pozitif tam sayılar kümesinden 1 rakamı seçilirse;

$$C : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$$

$$C(n) = \begin{cases} \frac{n}{2}, & n \rightarrow n \equiv 0 \pmod 2 \\ 3n+1, & n \rightarrow n \equiv 1 \pmod 2 \end{cases}$$

1. 1 tek sayı → $3(1)+1$ işlemini uygula → sonuç 4
2. 4 çift sayı → $4/2$ işlemini uygula → sonuç 2
3. 2 çift sayı → $2/2$ işlemini uygula → sonuç 1
4. 1 tek sayı → $3(1)+1$ işlemini uygula → sonuç 4
5. 4 çift sayı → $4/2$ işlemini uygula → sonuç 2
6. 2 çift sayı → $2/2$ işlemini uygula → sonuç 1

.

.

.

$$C(1) = \{ 1 , 4 , 2 , 1 , 4 , 2 \dots \}$$

Bu senaryoda 1,4 ,2 ,1,4 ,2… iterasyonu ile karşılaşılır. İterasyon sonsuza kadar devam eder, sonuç kümesi ise sonsuza gider.

Aynı senaryo ile 1 rakamından farklı verilen pozitif tam sayının, işlemler sonucunda 1 rakamına ulaşana kadar iterasyonun devam etmesine rağmen 1 rakamında iterasyonun devam ettirilmesi ile de karşılaşılabilir;

1. 10 çift sayı → $10/2$ işlemini uygula → sonuç 5
2. 5 tek sayı → $3(5)+1$ işlemini uygula → sonuç 16
3. 16 çift sayı → $16/2$ işlemini uygula → sonuç 8
4. 8 çift sayı → $8/2$ işlemini uygula → sonuç 4
5. 4 çift sayı → $4/2$ işlemini uygula → sonuç 2
6. 2 çift sayı → $2/2$ işlemini uygula → sonuç 1
7. 1 tek sayı → $3(1)+1$ işlemini uygula → sonuç 4
8. 4 çift sayı → $4/2$ işlemini uygula → sonuç 2
9. 2 çift sayı → $2/2$ işlemini uygula → sonuç 1

.

.

.

$$C(10) = \{\, 10\,, 5\,, 16\,, 8\,, 4\,, 2\,, 1\,, 4\,, 2 \ldots \,\}$$

0 rakamı da farklı bir iterasyon oluşturmaktadır. Collatz Konjektürü sadece pozitif tam sayılar kümesinde yer alan sayıları incelemektedir. Collatz Konjektürü'nde hesaplanmak üzere doğal sayılar kümesinden 0 rakamı seçilirse;

$$C(0) = \{\, 0 \ldots \}$$

0 iterasyonu ile karşılaşılmaktadır.

Bir diğer senaryo ise aşağıda örnekleri yer alan negatif tam sayılar ile hesaplama yapılırken ortaya çıkmaktadır. Negatif tam sayılar kümesi içerisinde de iterasyon ile karşılaşılan sayılar bulunmaktadır. Collatz Konjektürü'nde hesaplanmak üzere negatif tam sayılar kümesinden $-1$ sayısı seçilirse;

$$-1 \rightarrow -2 \rightarrow -1 \ \ldots$$

$$C(-1) = \{-1\,,\ -2\,,\ -1 \ldots\}$$

Collatz Konjektürü'nde hesaplanmak üzere negatif tam sayılar kümesinden $-5$ sayısı seçilirse;

$$-5 \rightarrow -14 \rightarrow -7 \rightarrow -20 \rightarrow -10 \rightarrow -5 \ \ldots$$

$$C(-5) = \{-5\,,\ -14\,,\ -7\,,\ -20\,,\ -10\,,\ -5\ \ldots\}$$

Son olarak Collatz Konjektürü'nde hesaplanmak üzere negatif tam sayılar kümesinden $-17$ sayısı seçilirse;

$$-17 \rightarrow -50 \rightarrow -25 \rightarrow -74 \rightarrow -37 \rightarrow -110 \rightarrow -55 \rightarrow -164$$

$$\rightarrow -82 \rightarrow -41 \rightarrow -122 \rightarrow -61 \rightarrow -182 \rightarrow -91 \rightarrow -272 \rightarrow$$

$$-136 \rightarrow -68 \rightarrow -34 \rightarrow -17 \ldots$$

$$C(-17) = \{-17\,,\ -50\,,\ -25\,,\ -74\,,\ -37\,,\ -110\,,\ -55\,,\ -164\,,\ -82\,,$$
$$-41\,,\ -122\,,\ -61\,,\ -182\,,\ -91\,,\ -272\,,\ -136\,,\ -68\,,$$
$$-34\,,\ -17\ \ldots\}$$

iterasyonları ile karşılaşılmaktadır [49].

# 5. BİLGİSAYAR PROGRAMI GELİŞTİRMELERİ

## 5.1 Standart Yöntem

Bu bölümde Collatz Konjektürü'nün geliştirilen bilgisayar programı ile hesaplanması incelenecektir. Standart yöntemde geliştirme çalışmaları Collatz Konjektürü'nün orijinal tanımı ve fonksiyonu üzerinden kurgulanmıştır.

### 5.1.1 Metodoloji

#### 5.1.1.1 Önerme

**Önerme 1 (Collatz Konjektürü):** Herhangi bir $n$ pozitif tam sayısından başlayarak $f(n)$ fonksiyonuna sokulan $n$ iterasyonları daima 1 rakamına ulaşır. Fonksiyona verilen $n$ sayısı tek ise sayı 3 ile çarpılır ve 1 eklenir, $n$ sayısı çift ise sayı 2'ye bölünür. 1 rakamına ulaşana kadar iterasyon devam eder [34].

Bu önermede yer alan fonksiyon aşağıdaki gibi ifade edilmektedir;

- $f : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$
- $f_0 = n\,,\ n \neq 0$
- $f_{i+1} = f_i \,/\, 2 \quad ,\ f_i\ çift\ sayı\ ise$
- $f_{i+1} = 3f_i + 1\,,\ f_i\ tek\ sayı\ ise$ [36].

#### 5.1.1.2 Prosedür

Önermede yer alan fonksiyona göre işleyiş mantığı aşağıdaki gibi özetlenebilir;

1. Herhangi bir $n$ pozitif tam sayısı seçilir.
2. Eğer $n$ sayısı çift ise $n$ 2'ye bölünür.
3. Eğer $n$ sayısı tek ise $n$ 3 ile çarpılır ve 1 eklenir.

4. Fonksiyon sonucunda çıkan sayı 1'e ulaşana kadar iterasyon devam eder [38].

#### 5.1.1.3 Notasyon

Önermede ifade edilen fonksiyonun matematiksel notasyonda gösterimi aşağıdaki gibidir. Bilgisayar programında tanım ve fonksiyon esas alınacaktır.

$$C : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$$

$$C(n) = \begin{cases} \frac{n}{2}, & n \rightarrow n \equiv 0 \pmod 2 \\ 3n+1, & n \rightarrow n \equiv 1 \pmod 2 \end{cases}$$

$$F(n) = \left\{ n, C(n), C\big(C(n)\big), C\left(C\big(C(n)\big)\right) \ldots . \right\}$$

$$= \{ n, C(n), C^2(n), C^3(n) \ \ldots . \} \qquad [37].$$

### 5.1.2 Bilgisayar programı ve program mimarisi

#### 5.1.2.1 Algoritma

Algoritma programlama terminolojisinde; bir problem sınıfını çözmek veya hesaplama yapmak için iyi tanımlanmış, bilgisayar tarafından işlenecek kural veya talimatlar dizisidir. Collatz Konjektürü'nün standart yöntemi ile işlemlerini yapacak olan programın algoritması aşağıdaki şekilde geliştirilmiştir;

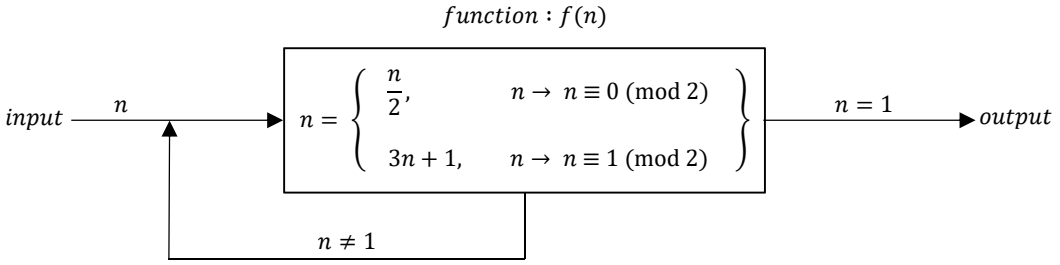
**Algoritma 1:** Collatz Konjektürü Program Algoritması

**Require** : $n\ is\ positive\ integer$
**Input** : $n$
**Output** : $CollatzConjecture(n)$

```
1  : if n ≤ 1 then return n
2  : procedure CollatzConjecture(n)
3  :   while n ≠ 1 do
4  :     if n mod 2 = 0 then n ← n / 2
5  :     else
6  :       n mod 2 ≠ 0 then n ← 3(n) + 1
7  :     end if
8  :     return n
9  :   end while
10 : end procedure
```

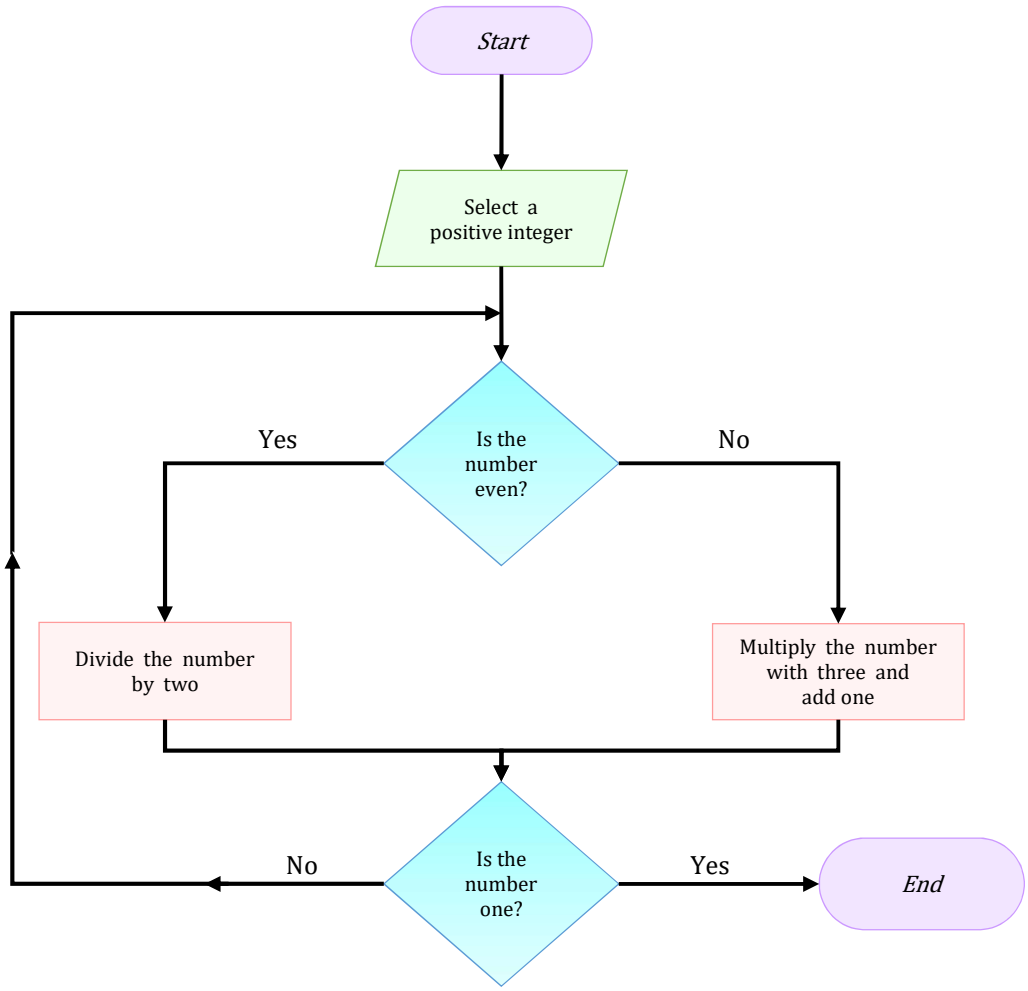
### 5.1.2.2 Akış diyagramı

Fonksiyonların çalışma mantığı ifade edilirken giriş değeri, fonksiyon ve çıkış değeri şeklinde gösterimi yapılmaktadır. Fonksiyon kısmında giriş değerinin hangi işlemlerden geçtiği, içerideki işleyiş mekanizması ve çıkış değeri oluşana kadar tüm süreç yer alır. Aşağıda Collatz Konjektürü fonksiyonunun gösterimi yapılmıştır;




**Şekil 5.1 :** Collatz Konjektürü standart yöntem fonksiyon gösterimi.

Akış şeması, iş akışını veya süreci temsil eden bir diyagram türüdür. İşlemi gerçekleştirmek için gereken adımların ve süreci ifade eden algoritmanın görsel veya sembolik bir temsilidir. Akış şeması sürecin baştan sona mantıksal olarak izlenmesine olanak tanır. Collatz Konjektürü'nün standart yöntemi ile işlemlerini yapacak olan programın algoritması aşağıdaki şekilde geliştirilmiştir;




**Şekil 5.2 :** Collatz Konjektürü standart yöntem akış diyagramı.

### 5.1.2.3 Program kodu

Collatz Konjektürü işlemlerinin bilgisayarda yapılabilmesi için bilgisayar programı geliştirilmiştir. Program standart yönteme yönelik yani Collatz Konjektürü'nün orijinal tanımı ve fonksiyonu üzerinden kurgulanmıştır.

Bilgisayar programı istenilen sayı aralığında hesaplama işlemleri yapıp, başlangıç değerleri, işlem yapılan seriler ve toplam adım sayılarından oluşan bir sonuç listesi üretebilecek şekilde tasarlanmıştır. Bu çalışma mekanizmasında algoritma daha gelişmiş hale getirilmiştir. Program kodu aşağıda paylaşılmıştır;

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace CollatzConjecture
{
    class Program
    {
        public static bool isEven(int number)
        {
            bool answer = (number % 2 == 0) ? true : false;
            return answer;
        }

        public static void CollatzConjecture()
        {
            int givenNumber = 100;
            int result = 0;
            int stepCount = 0;

            Console.WriteLine("\n-----------------------------");

            for (int i = 2; i <= givenNumber; i++)
            {
                result = i;
                stepCount = 0;

                StringBuilder sequence = new StringBuilder("");

                sequence.Append($"{result}  {result} -");

                while (result != 1)
                {
                    if (isEven(result))
                    {
                        result = result / 2;
                        sequence.Append($" {result} -");
                        stepCount++;
                    }
                    else
                    {
                        result = result * 3 + 1;
                        sequence.Append($" {result} -");
                        stepCount++;
                    }
                }

                sequence.Remove((sequence.Length - 1), 1);
                sequence.Append($"  {stepCount}");

                Console.WriteLine($"{sequence}\n");
                Console.WriteLine("----------------------------");
            }
        }
```

```
        static void Main(string[] args)
        {

            CollatzConjecture();

            Console.ReadKey();

        }
    }
}
```

### 5.1.3 Sonuçlar

#### 5.1.3.1 Program çıktısı

Collatz Konjektürü işlemlerinin bilgisayarda yapılabilmesi geliştirilen bilgisayar programı çalıştırılmıştır. Program standart yöntemde hesaplama işlemlerini yapıp, başlangıç değerleri, işlem yapılan seriler ve toplam adım sayılarından oluşan bir sonuç listesi üretmiştir. Sonuçlar öncelikle program çıktısı, daha sonrasında ise sonuç listesinin tam tablosu olarak iki aşamada incelenmiştir.

Program çıktısı aşağıda sunulmaktadır;


**Şekil 5.3 :** Collatz Konjektürü standart yöntem program çıktısı.

Sonuç listesine ait tam tablo ise aşağıdaki gibidir;

**Çizelge 5.1 :** Standart yöntem program çıktısının tablosu.

| $n$ | $Seri$ | $i$ |
|---|---|---|
| 1 | 1 | – |
| 2 | 2 - 1 | 1 |
| 3 | 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 7 |
| 4 | 4 - 2 - 1 | 2 |
| 5 | 5 - 16 - 8 - 4 - 2 - 1 | 5 |

| | | |
|---|---|---|
| 6 | 6 - 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 8 |
| 7 | 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 16 |
| 8 | 8 - 4 - 2 - 1 | 3 |
| 9 | 9 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 19 |
| 10 | 10 - 5 - 16 - 8 - 4 - 2 - 1 | 6 |
| 11 | 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 14 |
| 12 | 12 - 6 - 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 9 |
| 13 | 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 9 |
| 14 | 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 15 | 15 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 16 | 16 - 8 - 4 - 2 - 1 | 4 |
| 17 | 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 12 |
| 18 | 18 - 9 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 20 |
| 19 | 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 20 |
| 20 | 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 7 |
| 21 | 21 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 7 |
| 22 | 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 15 |
| 23 | 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 15 |
| 24 | 24 - 12 - 6 - 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 10 |
| 25 | 25 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 23 |
| 26 | 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 10 |
| 27 | 27 - 82 - 41 - 124 - 62 - 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 111 |
| 28 | 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 18 |
| 29 | 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 18 |
| 30 | 30 - 15 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 18 |
| 31 | 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 106 |
| 32 | 32 - 16 - 8 - 4 - 2 - 1 | 5 |

| | | |
|---|---|---|
| 33 | 33 - 100 - 50 - 25 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 26 |
| 34 | 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 13 |
| 35 | 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 13 |
| 36 | 36 - 18 - 9 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 21 |
| 37 | 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 21 |
| 38 | 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 21 |
| 39 | 39 - 118 - 59 - 178 - 89 - 268 - 134 - 67 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 34 |
| 40 | 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 8 |
| 41 | 41 - 124 - 62 - 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 109 |
| 42 | 42 - 21 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 8 |
| 43 | 43 - 130 - 65 - 196 - 98 - 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 29 |
| 44 | 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 16 |
| 45 | 45 - 136 - 68 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 16 |
| 46 | 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 16 |
| 47 | 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 104 |
| 48 | 48 - 24 - 12 - 6 - 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 11 |
| 49 | 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 24 |
| 50 | 50 - 25 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 24 |
| 51 | 51 - 154 - 77 - 232 - 116 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 24 |
| 52 | 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 11 |
| 53 | 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 11 |
| 54 | 54 - 27 - 82 - 41 - 124 - 62 - 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - | 112 |

| | | |
|---|---|---|
| | 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | |
| 55 | 55 - 166 - 83 - 250 - 125 - 376 - 188 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 112 |
| 56 | 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 19 |
| 57 | 57 - 172 - 86 - 43 - 130 - 65 - 196 - 98 - 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 32 |
| 58 | 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 19 |
| 59 | 59 - 178 - 89 - 268 - 134 - 67 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 32 |
| 60 | 60 - 30 - 15 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 19 |
| 61 | 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 19 |
| 62 | 62 - 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 107 |
| 63 | 63 - 190 - 95 - 286 - 143 - 430 - 215 - 646 - 323 - 970 - 485 - 1456 - 728 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 107 |
| 64 | 64 - 32 - 16 - 8 - 4 - 2 - 1 | 6 |
| 65 | 65 - 196 - 98 - 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 27 |
| 66 | 66 - 33 - 100 - 50 - 25 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 27 |
| 67 | 67 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 27 |
| 68 | 68 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 14 |
| 69 | 69 - 208 - 104 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 14 |

| | | |
|---|---|---|
| 70 | 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 14 |
| 71 | 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 102 |
| 72 | 72 - 36 - 18 - 9 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 22 |
| 73 | 73 - 220 - 110 - 55 - 166 - 83 - 250 - 125 - 376 - 188 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 115 |
| 74 | 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 22 |
| 75 | 75 - 226 - 113 - 340 - 170 - 85 - 256 - 128 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 14 |
| 76 | 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 22 |
| 77 | 77 - 232 - 116 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 22 |
| 78 | 78 - 39 - 118 - 59 - 178 - 89 - 268 - 134 - 67 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 35 |
| 79 | 79 - 238 - 119 - 358 - 179 - 538 - 269 - 808 - 404 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 35 |
| 80 | 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 9 |
| 81 | 81 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 22 |
| 82 | 82 - 41 - 124 - 62 - 31 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 110 |
| 83 | 83 - 250 - 125 - 376 - 188 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - | 110 |

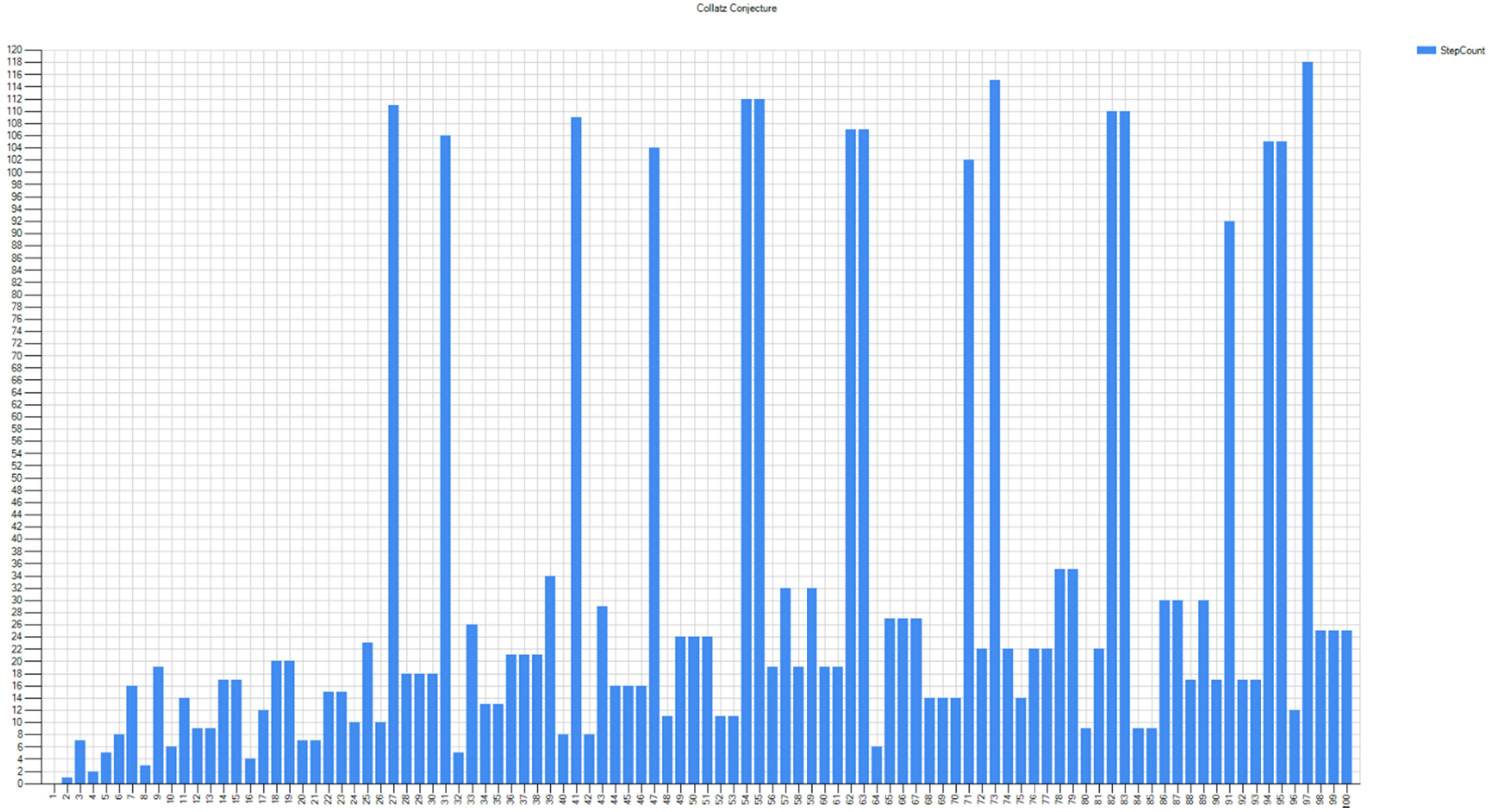
| | | |
|---|---|---|
| | 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | |
| 84 | 84 - 42 - 21 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 9 |
| 85 | 85 - 256 - 128 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 9 |
| 86 | 86 - 43 - 130 - 65 - 196 - 98 - 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 30 |
| 87 | 87 - 262 - 131 - 394 - 197 - 592 - 296 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 30 |
| 88 | 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 89 | 89 - 268 - 134 - 67 - 202 - 101 - 304 - 152 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 30 |
| 90 | 90 - 45 - 136 - 68 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 91 | 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 92 |
| 92 | 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 93 | 93 - 280 - 140 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 17 |
| 94 | 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 105 |
| 95 | 95 - 286 - 143 - 430 - 215 - 646 - 323 - 970 - 485 - 1456 - 728 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 105 |
| 96 | 96 - 48 - 24 - 12 - 6 - 3 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 12 |
| 97 | 97 - 292 - 146 - 73 - 220 - 110 - 55 - 166 - 83 - 250 - 125 - 376 - 188 - 94 - 47 - 142 - 71 - 214 - 107 - 322 - 161 - 484 - 242 - 121 - 364 - 182 - 91 - 274 - 137 - 412 - 206 - 103 - 310 - 155 - 466 - 233 - 700 - 350 - 175 - 526 - 263 - 790 - 395 - 1186 - 593 - 1780 - 890 - 445 - 1336 - 668 - 334 - 167 - 502 - 251 - 754 - 377 - 1132 - 566 - 283 - 850 - 425 - 1276 - 638 - 319 - 958 - 479 - 1438 - 719 - 2158 - 1079 - 3238 - 1619 - 4858 - 2429 - 7288 - 3644 - 1822 - 911 - 2734 - 1367 - 4102 - 2051 - 6154 - 3077 - 9232 - 4616 - 2308 - 1154 - 577 - 1732 - 866 - 433 - 1300 - 650 - 325 - 976 - 488 - 244 - 122 - 61 - 184 - 92 - 46 - 23 - 70 - 35 - 106 - 53 - 160 - 80 - 40 - | 118 |

| | | |
|---|---|---|
| | 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | |
| 98 | 98 - 49 - 148 - 74 - 37 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 25 |
| 99 | 99 - 298 - 149 - 448 - 224 - 112 - 56 - 28 - 14 - 7 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 25 |
| 100 | 100 - 50 - 25 - 76 - 38 - 19 - 58 - 29 - 88 - 44 - 22 - 11 - 34 - 17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1 | 25 |

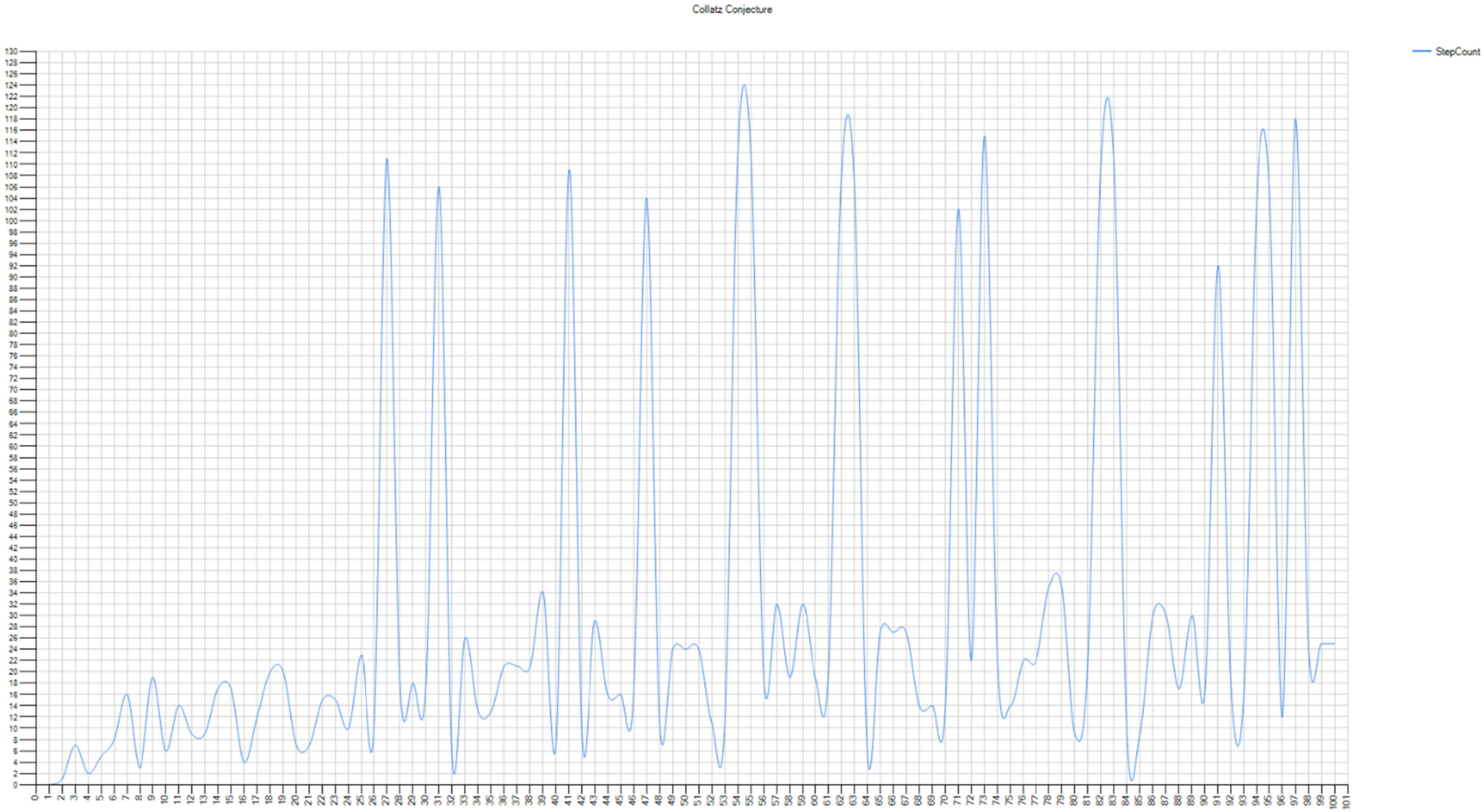
### 5.1.3.2 Program grafikleri

Collatz Konjektürü standart yöntemde hesaplama işlemlerini yapan bilgisayar programına ait tüm detaylar önceki bölümlerde incelenmiştir. Bu program haricinde yine standart yöntemde yapılan bu işlemlerin grafiklerini oluşturan bir bilgisayar programı daha geliştirilmiştir. Programda sayıların adım sayılarına göre oluşturulmuş sütun grafiği, sp çizgi grafiği, çizgi grafiği ve polar grafik geliştirmeleri yapılmıştır.

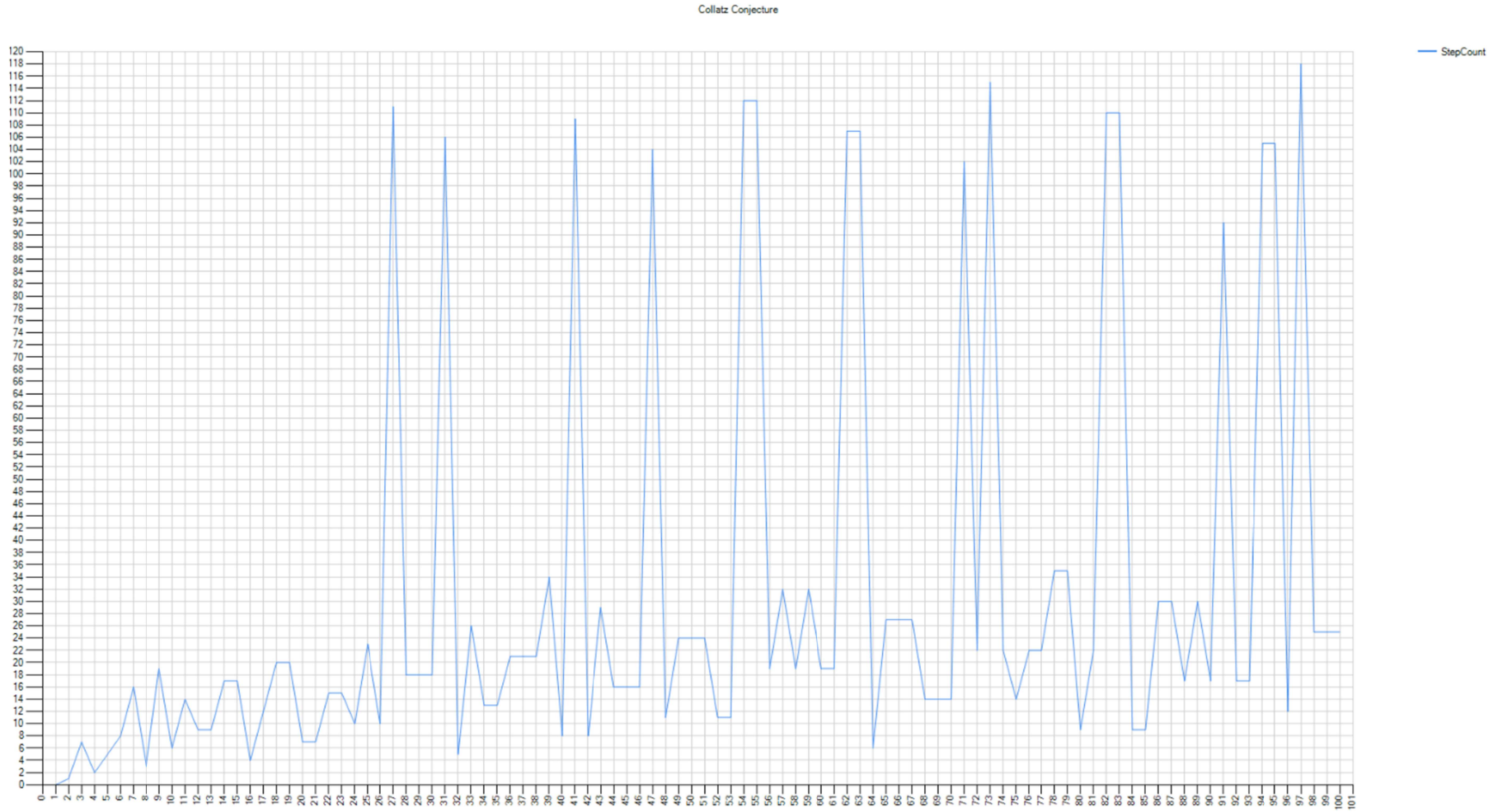
Geliştirilen program grafikleri aşağıda grafik bazında yer almaktadır;



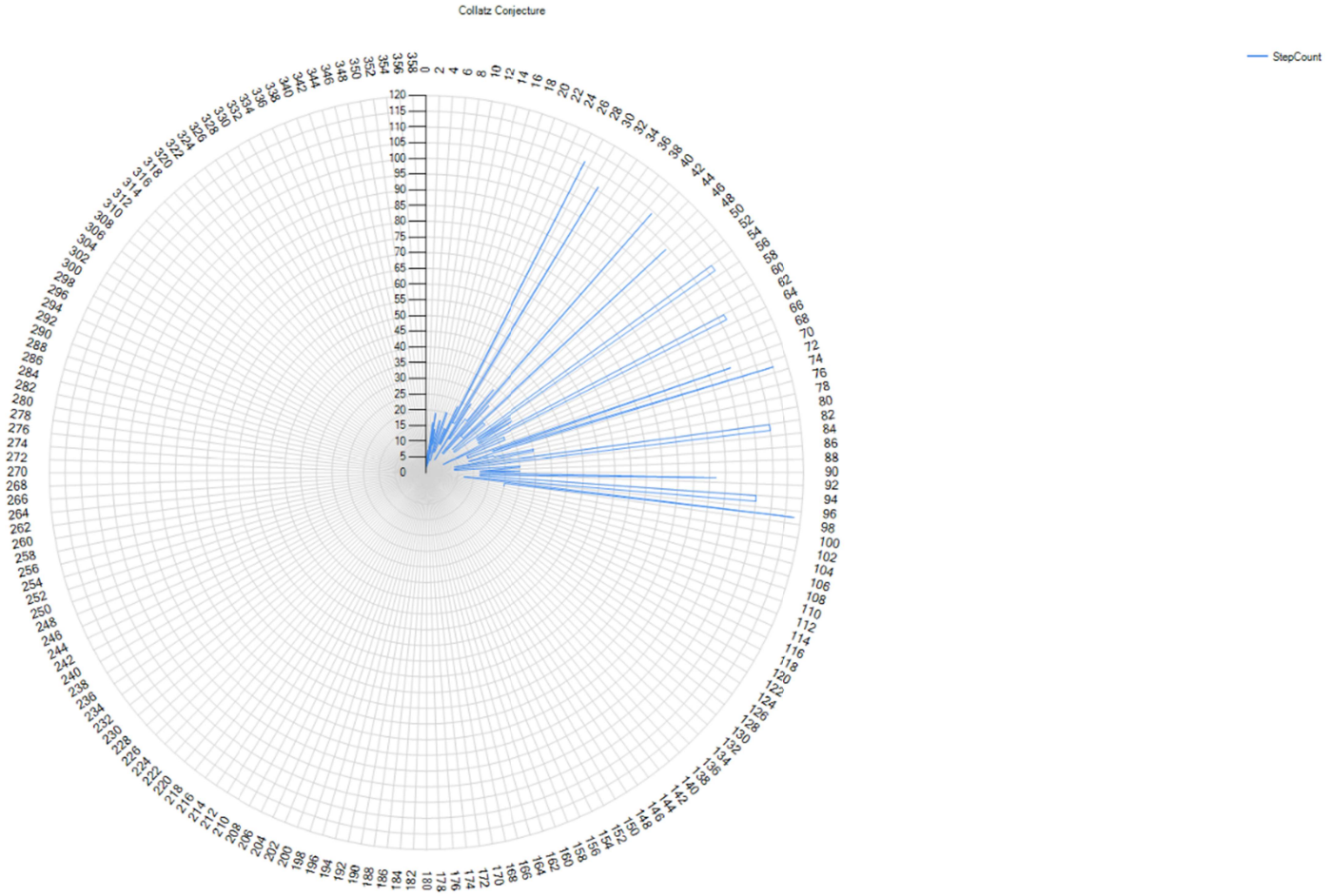

**Şekil 5.4 :** Collatz Konjektürü standart yöntem sütun grafiği




**Şekil 5.5 :** Collatz Konjektürü standart yöntem sp çizgi grafiği




**Şekil 5.6 :** Collatz Konjektürü standart yöntem çizgi grafiği




**Şekil 5.7 :** Collatz Konjektürü standart yöntem polar grafiği

### 5.1.3.3 Grafiksel gösterim program kodu

Collatz Konjektürü standart yöntemde yapılan hesaplama işlemlerinin grafiklerini oluşturan bilgisayar programı çıktıları incelenmiştir. Programda sayıların adım sayılarına göre oluşturulmuş sütun grafiği, sp çizgi grafiği, çizgi grafiği ve polar grafikleri oluşturulmuştur. Bu grafiklerin üretildiği program kodu aşağıda paylaşılmıştır;

**Program.cs**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CollatzConjecture
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new FrmCollatzConjecture());
        }
    }
}
```

**FrmCollatzConjecture.cs**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CollatzConjecture
{
    public partial class FrmCollatzConjecture : Form
    {
        public static int givenNumber = 100;
        int[] steps = new int[givenNumber];
```

```
public FrmCollatzConjecture()
{
    InitializeComponent();
}

public static bool isEven(int number)
{
    bool answer = (number % 2 == 0) ? true : false;
    return answer;
}

public void CollatzConjecture()
{
    int result = 0;
    int stepCount = 0;

    for (int i = 2; i <= givenNumber; i++)
    {
        result = i;
        stepCount = 0;

        StringBuilder sequence = new StringBuilder("");

        sequence.Append($"{result} | {result} -");

        while (result != 1)
        {
            if (isEven(result))
            {
                result = result / 2;
                sequence.Append($" {result} -");
                stepCount++;
            }
            else
            {
                result = result * 3 + 1;
                sequence.Append($" {result} -");
                stepCount++;
            }
        }

        sequence.Remove((sequence.Length - 1), 1);
        sequence.Append($" | {stepCount}");

        steps[i-2] = stepCount;
    }
}

private void FrmCollatzConjecture_Load(object sender, EventArgs e)
{
    CollatzConjecture();

    for (int i = 0; i < steps.Length - 1; i++)
    {
        var number = i + 2;
        var stepsCount = steps[i];
        this.columnChart.Series["StepCount"].Points.AddXY(number,
         stepsCount);
```

```
                this.spLineChart.Series["StepCount"].Points.AddXY(number,
                 stepsCount);
                this.lineChart.Series["StepCount"].Points.AddXY(number,
                 stepsCount);
                this.funnelChart.Series["StepCount"].Points.AddXY(number,
                 stepsCount);
            }
        }
    }
}
```

## 5.2 Soyut Makina Yöntemi

Bu bölümde Collatz Konjektürü'nün soyut makina yöntemi ile hesaplanması incelenecektir. Soyut makina yöntemi sayıların binary sisteme dönüştürülmesi, işlem yapılması ve Collatz Konjektürü'nün türetilmiş fonksiyonu üzerinden kurgulanmıştır.

### 5.2.1 Metodoloji

#### 5.2.1.1 Önerme

**Önerme 2 (Collatz Konjektürü):** Herhangi bir $n$ pozitif tam sayısından başlayarak $f(n)$ fonksiyonuna sokulan $n$ iterasyonları daima 1 rakamına ulaşır. Fonksiyona verilen $n$ sayısı binary koda dönüştürülür. $n$ sayısı çift ise sayı sondaki 0 adedi kadar 2'ye bölünür. $n$ sayısı tek ise sayı 3 ile çarpılır ve 1 eklenir, $n$ sayısı çift ise sayı 2'ye bölünür. 1 rakamına ulaşana kadar iterasyon devam eder [34].

Bu önermede yer alan fonksiyon aşağıdaki gibi ifade edilmektedir;

- $f : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$
- $f_0 = n\,,\ n \neq 0$
- $f_{i+1} = f_i \,/\, 2\ \ (sondaki\ 0\ adedi\ kadar)\ \ ,\ f_i\ \ çift\ sayı\ ise$
- $f_{i+1} = (2f_i + 1) + (f_i)\ ,\ f_i\ \ tek\ sayı\ ise$
- $f_{i+1}\ /\ 2\ \ (sondaki\ 0\ adedi\ kadar)\ \ ,f_{i+1\ i}\ \ çift\ sayı\ ise$

#### 5.2.1.2 Prosedür

Soyut makina yönteminde serideki tek sayılar üzerinde yoğunlaşılır. Önermede yer alan fonksiyona göre işleyiş mantığı aşağıdaki gibi özetlenebilir;

1. Herhangi bir $n$ pozitif tam sayısı seçilir.
2. $n$ sayısı binary koda dönüştürülür.
3. Eğer $n$ sayısı çift ise binary kod sonundaki sıfırlar atılır.
4. Üçüncü adım mantığı sonuç tek olana kadar tekrar tekrar 2'ye bölmektir.
5. Eğer $n$ sayısı tek ise binary kod sonuna 1 eklenir.
6. Beşinci adım mantığı $2n + 1$ sonucunu elde etmektir.
7. $n$ sayısı ile elde edilen sayı toplanır.
8. Yedinci adım mantığı $(2n + 1 + n) = (3n + 1)$ sonucunu elde etmektir.
9. İşlem sonucu $n$ sayısı çift ise binary kod sonundaki sıfırlar atılır.
10. Fonksiyon sonucunda çıkan sayı 1'e ulaşana kadar iterasyon devam eder.

#### 5.2.1.3 Notasyon

Önermede ifade edilen fonksiyonun işleyiş mekanizması binary (ikili) sayı sistemi üzerine kurgulanmıştır. Günlük hayatta her ne kadar 10'luk (decimal) sayı sistemini esas alıp kullansak da, bilgisayarlar tüm işlemlerini en temel seviyede 1'ler ve 0'lara dönüştürerek çalışırlar. Soyut makina yönteminde, binary koda dönüştürülmüş sayılardan oluşan serideki tek sayılar üzerinde yoğunlaşılır.

10'luk sayı sisteminde 347 sayısı aşağıdaki gibi hesaplanmaktadır;

$$347 = 3 \times 10^2 + 4 \times 10^1 + 7 \times 10^0$$
$$= 3 \times 100 + 4 \times 10 + 7 \times 1$$
$$= 300 + 40 + 7$$

347 sayısı binary sayı sistemine aşağıdaki gibi dönüştürülmektedir;

$$347 = 1 \times 2^8 + 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$
$$= 101011011$$

Soyut makina yönteminde, önermede yer alan fonksiyon esas alınarak hesaplama yapılır. Örneğin 7 sayısı için öncelikle aşağıdaki gibi binary koda dönüştürme işlemi yapılır;

$$7 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$= 111$$

Collatz Konjektürü işlemlerinde soyut makina yöntemine göre, binary koda dönüştürülen 7 sayısının adım adım hesaplanışı aşağıdaki gibidir;

| | |
|---|---|
| 111 | $n = 7$ |
| 1111 | $2n + 1 = 15$ |
| 1011~~0~~ | $(2n + 1 + n) = 22 \rightarrow 22 / 2 = 11$ |
| 10111 | 23 |
| 10001~~0~~ | $34 \rightarrow 17$ |
| 100011 | 35 |
| 110100 | $52 \rightarrow 13$ |
| 11011 | 27 |
| 101~~000~~ | $40 \rightarrow 5$ |
| 1011 | 11 |
| 1~~0000~~ | $16 \rightarrow 1$ |

## 5.3 Parite Sekansı Yöntemi

Bu bölümde Collatz Konjektürü'nün parite sekansı yöntemi ile geliştirilen bilgisayar programı sayesinde hesaplanması incelenecektir. Parite sekansı yönteminde geliştirme çalışmaları Collatz Konjektürü'nün türetilmiş tanımı ve türetilmiş fonksiyonu üzerinden kurgulanmıştır.

### 5.3.1 Metodoloji

#### 5.3.1.1 Önerme

**Önerme 3 (Collatz Konjektürü):** Herhangi bir $n$ pozitif tam sayısından başlayarak $f(n)$ fonksiyonuna sokulan $n$ iterasyonları daima 1 rakamına ulaşır. Fonksiyona verilen $n$ sayısı tek ise sayı 3 ile çarpılır, 1 eklenir ve 2'ye bölünür , $n$ sayısı çift ise sayı 2'ye bölünür. 1 rakamına ulaşana kadar iterasyon devam eder [34].

Bu önermede yer alan fonksiyon aşağıdaki gibi ifade edilmektedir;

- $f : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$
- $f_0 = n\,,\ n \neq 0$
- $f_{i+1} = f_i \,/\, 2 \quad ,\ f_i\ çift\ sayı\ ise$
- $f_{i+1} = (3f_i + 1) \,/\, 2,\ f_i\ tek\ sayı\ ise$

#### 5.3.1.2 Prosedür

Collatz Konjektürü'nün parite sekansı yöntemine özel türetilmiş önermede yer alan fonksiyona göre işleyiş mantığı aşağıdaki gibi özetlenebilir;

1. Herhangi bir $n$ pozitif tam sayısı seçilir.
2. Eğer $n$ sayısı çift ise $n$ 2'ye bölünür.
3. Eğer $n$ sayısı tek ise $n$ 3 ile çarpılır, 1 eklenir ve 2'ye bölünür.
4. Fonksiyon sonucunda çıkan sayı 1'e ulaşana kadar iterasyon devam eder.

#### 5.3.1.3 Notasyon

Önermede ifade edilen fonksiyon parite sekansı yöntemine özel türetilmiştir. Standart yöntemde belirtilen orijinal fonksiyonun $C_s(n)$ ve parite yöntemine özel türetilmiş fonksiyonun $C_p(n)$ matematiksel notasyonda gösterimi aşağıdaki gibidir. Bilgisayar programında parite fonksiyonuna özel türetilmiş tanım ve türetilmiş fonksiyon esas alınacaktır.

$$C : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$$

$$C_s(n) = \begin{cases} \frac{n}{2}, & n \rightarrow\ n \equiv 0 \pmod 2 \\ 3n+1, & n \rightarrow\ n \equiv 1 \pmod 2 \end{cases}$$

$$C_p(n) = \begin{cases} \frac{n}{2}, & n \rightarrow\ n \equiv 0 \pmod 2 \\ \frac{3n+1}{2}, & n \rightarrow\ n \equiv 1 \pmod 2 \end{cases}$$

Tanımlanan fonksiyonda yer alan $3n+1$ işlemi her zaman çift sayı sonucunu vereceği için, parite sekansı yönteminde tek sayılar için $(3n+1)\ /\ 2$ işlemi uygulanabilmektedir.

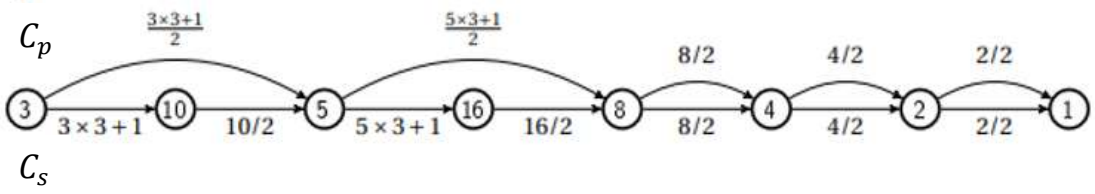
Şimdi de Collatz Konjetürü'nde 3 sayısını hem standart yöntem ile hem de parite sekansı yöntemiyle hesaplayalım. Her iki yöntem için de hesaplanan serileri inceleyelim;

$$C_s(3) = \{\, 3\,, 10\,, 5\,, 16\,, 8\,, 4\,, 2\,, 1\,\}$$

Bu örnekte seçilen 3 sayısı standart yöntem fonksiyonu üzerinden işleme verildi. Toplamda 7 adımda 1 rakamına ulaştı.

$$C_p(3) = \{\, 3\,, 5\,, 8\,, 4\,, 2\,, 1\,\}$$

Bu örnekte seçilen 3 sayısı parite sekansı yöntemi fonksiyonu üzerinden işleme verildi. Toplamda 5 adımda 1 rakamına ulaştı.




**Şekil 5.8 :** Parite sekansı yöntemi hesaplama adımları gösterimi [50].

### 5.3.2 Bilgisayar programı ve program mimarisi

#### 5.3.2.1 Algoritma

Collatz Konjektürü'nde parite sekansı yöntemi ile işlemlerini yapacak olan programın algoritması aşağıdaki şekilde geliştirilmiştir;

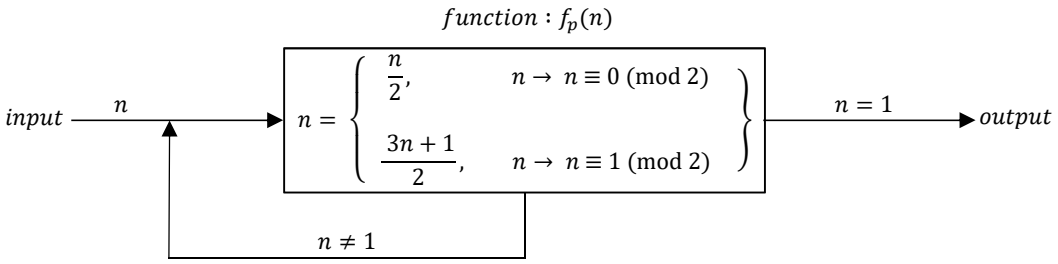
**Algoritma 2:** Parite Sekansı Yöntemi Program Algoritması

**Require** : $n\ is\ positive\ integer$
**Input** : $n$
**Output** : $CollatzConjecture(n)$

```
1  : if n ≤ 1 then return n
2  : procedure CollatzConjecture(n)
3  :   while n ≠ 1 do
4  :     if n mod 2 = 0 then n ← n / 2
5  :     else
6  :       n mod 2 ≠ 0 then n ← (3(n) + 1) / 2
7  :     end if
8  :     return n
9  :   end while
10 : end procedure
```

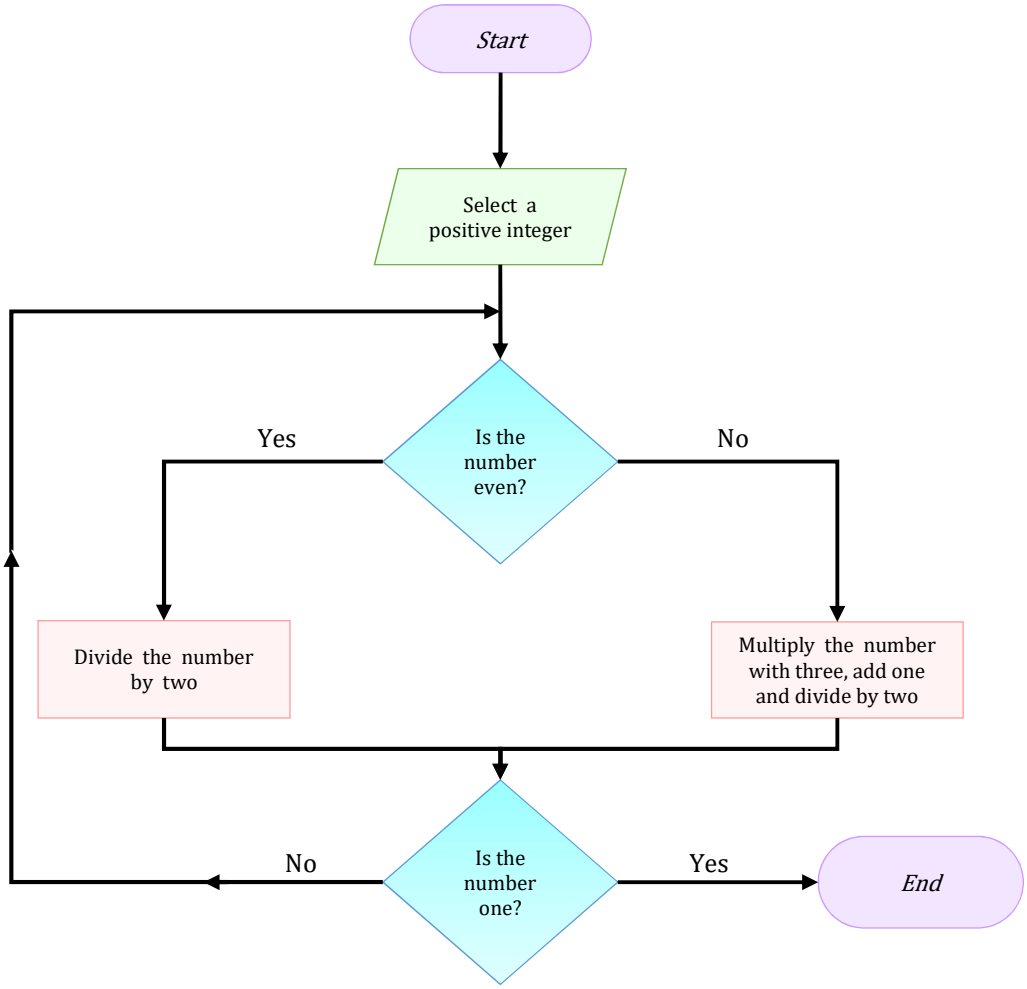
#### 5.3.2.2 Akış diyagramı

Collatz Konjektürü'nde parite sekansı yöntemi ile işlemlerini yapacak olan programın fonksiyonu aşağıdaki şekilde geliştirilmiştir;




**Şekil 5.9 :** Collatz Konjektürü parite sekans yöntemi fonksiyon gösterimi.

Collatz Konjektürü'de parite sekansı yöntemi ile işlemlerini yapacak olan programın algoritması aşağıdaki şekilde geliştirilmiştir;




**Şekil 5.10 :** Collatz Konjektürü parite sekansı yöntemi akış diyagramı.

### 5.3.2.3 Program kodu

Collatz Konjektürü'nde parite sekansı yöntemi işlemlerinin bilgisayarda yapılabilmesi için bilgisayar programı geliştirilmiştir. Program parite sekansı yöntemine yönelik yani Collatz Konjektürü'nün türetilmiş tanımı ve türetilmiş fonksiyonu üzerinden kurgulanmıştır.

Bilgisayar programı istenilen sayı aralığında hesaplama işlemleri yapıp, başlangıç değerleri, işlem yapılan seriler ve toplam adım sayılarından oluşan bir sonuç listesi üretebilecek şekilde tasarlanmıştır. Bu çalışma mekanizmasında algoritma daha gelişmiş hale getirilmiştir. Program kodu aşağıda paylaşılmıştır;

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace CollatzConjecture
{
    class Program
    {
        public static bool isEven(int number)
        {
            bool answer = (number % 2 == 0) ? true : false;
            return answer;
        }

        public static void CollatzConjecture()
        {
            int givenNumber = 100;
            int result = 0;
            int stepCount = 0;

            Console.WriteLine("\n-----------------------------");

            for (int i = 2; i <= givenNumber; i++)
            {
                result = i;
                stepCount = 0;

                StringBuilder sequence = new StringBuilder("");

                sequence.Append($"{result}  {result} -");

                while (result != 1)
                {
                    if (isEven(result))
                    {
                        result = result / 2;
                        sequence.Append($" {result} -");
                        stepCount++;
                    }
                    else
                    {
                        result = (result * 3 + 1) / 2;
                        sequence.Append($" {result} -");
                        stepCount++;
                    }
                }

                sequence.Remove((sequence.Length - 1), 1);
                sequence.Append($"  {stepCount}");

                Console.WriteLine($"{sequence}\n");
                Console.WriteLine("----------------------------");
            }
        }
```

```
        static void Main(string[] args)
        {

            CollatzConjecture();

            Console.ReadKey();

        }
    }
}
```

### 5.3.3 Sonuçlar

#### 5.3.3.1 Program çıktısı

Collatz Konjektürü'nde parite sekansı yöntemi işlemlerinin bilgisayarda yapılabilmesi için geliştirilen bilgisayar programı çalıştırılmıştır. Program parite sekansı yönteminde hesaplama işlemlerini yapıp, başlangıç değerleri, işlem yapılan seriler ve toplam adım sayılarından oluşan bir sonuç listesi üretmiştir. Sonuçlar öncelikle program çıktısı, daha sonrasında ise sonuç listesinin tam tablosu olarak iki aşamada incelenmiştir.

Program çıktısı aşağıda sunulmaktadır;


**Şekil 5.11 :** Collatz Konjektürü parite sekansı yöntemi program çıktısı.

Sonuç listesine ait tam tablo ise aşağıdaki gibidir;

**Çizelge 5.2 :** Parite sekansı yöntemi program çıktısının tablosu.

| $n$ | $Seri$ | $i$ |
|---|---|---|
| 1 | 1 | – |
| 2 | 2 - 1 | 1 |
| 3 | 3 - 5 - 8 - 4 - 2 - 1 | 5 |
| 4 | 4 - 2 - 1 | 2 |
| 5 | 5 - 8 - 4 - 2 - 1 | 4 |
| 6 | 6 - 3 - 5 - 8 - 4 - 2 - 1 | 6 |
| 7 | 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 8 | 8 - 4 - 2 - 1 | 3 |
| 9 | 9 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 10 | 10 - 5 - 8 - 4 - 2 - 1 | 5 |
| 11 | 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 10 |
| 12 | 12 - 6 - 3 - 5 - 8 - 4 - 2 - 1 | 7 |
| 13 | 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 7 |
| 14 | 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 12 |
| 15 | 15 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 12 |
| 16 | 16 - 8 - 4 - 2 - 1 | 4 |
| 17 | 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 9 |
| 18 | 18 - 9 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |
| 19 | 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |
| 20 | 20 - 10 - 5 - 8 - 4 - 2 - 1 | 6 |
| 21 | 21 - 32 - 16 - 8 - 4 - 2 - 1 | 6 |
| 22 | 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 23 | 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 24 | 24 - 12 - 6 - 3 - 5 - 8 - 4 - 2 - 1 | 8 |
| 25 | 25 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 26 | 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 8 |
| 27 | 27 - 41 - 62 - 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 70 |
| 28 | 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 29 | 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 30 | 30 - 15 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 31 | 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 67 |

| | | |
|---|---|---|
| 32 | 32 - 16 - 8 - 4 - 2 - 1 | 5 |
| 33 | 33 - 50 - 25 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 18 |
| 34 | 34 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 10 |
| 35 | 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 10 |
| 36 | 36 - 18 - 9 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 15 |
| 37 | 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 15 |
| 38 | 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 15 |
| 39 | 39 - 59 - 89 - 134 - 67 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 23 |
| 40 | 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 7 |
| 41 | 41 - 62 - 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 -5 - 8 - 4 - 2 - 1 | 69 |
| 42 | 42 - 21 - 32 - 16 - 8 - 4 - 2 - 1 | 7 |
| 43 | 43 - 65 - 98 - 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 20 |
| 44 | 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 12 |
| 45 | 45 - 68 - 34 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 12 |
| 46 | 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 12 |
| 47 | 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 66 |
| 48 | 48 - 24 - 12 - 6 - 3 - 5 - 8 - 4 - 2 - 1 | 9 |
| 49 | 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 17 |
| 50 | 50 - 25 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 17 |
| 51 | 51 - 77 - 116 - 58 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 17 |
| 52 | 52 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 9 |
| 53 | 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 9 |
| 54 | 54 - 27 - 41 - 62 - 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 71 |
| 55 | 55 - 83 - 125 - 188 - 94 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 71 |
| 56 | 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |
| 57 | 57 - 86 - 43 - 65 - 98 - 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 22 |
| 58 | 58 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |

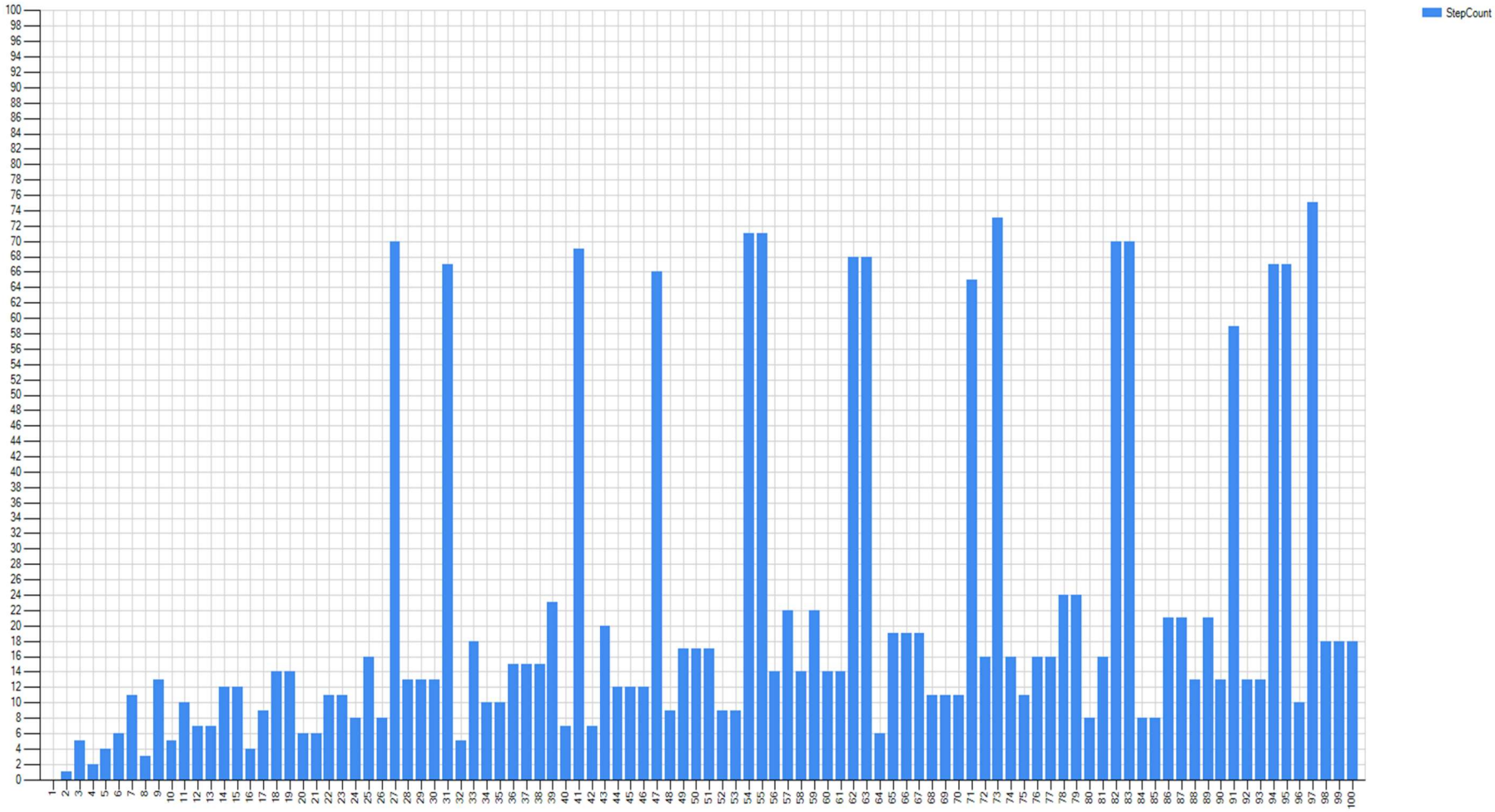
| | | |
|---|---|---|
| 59 | 59 - 89 - 134 - 67 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 22 |
| 60 | 60 - 30 - 15 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |
| 61 | 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 14 |
| 62 | 62 - 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 68 |
| 63 | 63 - 95 - 143 - 215 - 323 - 485 - 728 - 364 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 68 |
| 64 | 64 - 32 - 16 - 8 - 4 - 2 - 1 | 6 |
| 65 | 65 - 98 - 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 19 |
| 66 | 66 - 33 - 50 - 25 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 19 |
| 67 | 67 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 19 |
| 68 | 68 - 34 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 69 | 69 - 104 - 52 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 70 | 70 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 11 |
| 71 | 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 65 |
| 72 | 72 - 36 - 18 - 9 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 73 | 73 - 110 - 55 - 83 - 125 - 188 - 94 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 73 |
| 74 | 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 75 | 75 - 113 - 170 - 85 - 128 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 11 |
| 76 | 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 77 | 77 - 116 - 58 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 78 | 78 - 39 - 59 - 89 - 134 - 67 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 24 |
| 79 | 79 - 119 - 179 - 269 - 404 - 202 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 24 |
| 80 | 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 8 |
| 81 | 81 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 16 |
| 82 | 82 - 41 - 62 - 31 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - | 70 |

| | | |
|---|---|---|
| | 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | |
| 83 | 83 - 125 - 188 - 94 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 70 |
| 84 | 84 - 42 - 21 - 32 - 16 - 8 - 4 - 2 - 1 | 8 |
| 85 | 85 - 128 - 64 - 32 - 16 - 8 - 4 - 2 - 1 | 8 |
| 86 | 86 - 43 - 65 - 98 - 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 21 |
| 87 | 87 - 131 - 197 - 296 - 148 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 21 |
| 88 | 88 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 89 | 89 - 134 - 67 - 101 - 152 - 76 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 21 |
| 90 | 90 - 45 - 68 - 34 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 91 | 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 59 |
| 92 | 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 93 | 93 - 140 - 70 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 13 |
| 94 | 94 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 67 |
| 95 | 95 - 143 - 215 - 323 - 485 - 728 - 364 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 67 |
| 96 | 96 - 48 - 24 - 12 - 6 - 3 - 5 - 8 - 4 - 2 - 1 | 10 |
| 97 | 97 - 146 - 73 - 110 - 55 - 83 - 125 - 188 - 94 - 47 - 71 - 107 - 161 - 242 - 121 - 182 - 91 - 137 - 206 - 103 - 155 - 233 - 350 - 175 - 263 - 395 - 593 - 890 - 445 - 668 - 334 - 167 - 251 - 377 - 566 - 283 - 425 - 638 - 319 - 479 - 719 - 1079 - 1619 - 2429 - 3644 - 1822 - 911 - 1367 - 2051 - 3077 - 4616 - 2308 - 1154 - 577 - 866 - 433 - 650 - 325 - 488 - 244 - 122 - 61 - 92 - 46 - 23 - 35 - 53 - 80 - 40 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 75 |
| 98 | 98 - 49 - 74 - 37 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 18 |
| 99 | 99 - 149 - 224 - 112 - 56 - 28 - 14 - 7 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 18 |
| 100 | 100 - 50 - 25 - 38 - 19 - 29 - 44 - 22 - 11 - 17 - 26 - 13 - 20 - 10 - 5 - 8 - 4 - 2 - 1 | 18 |

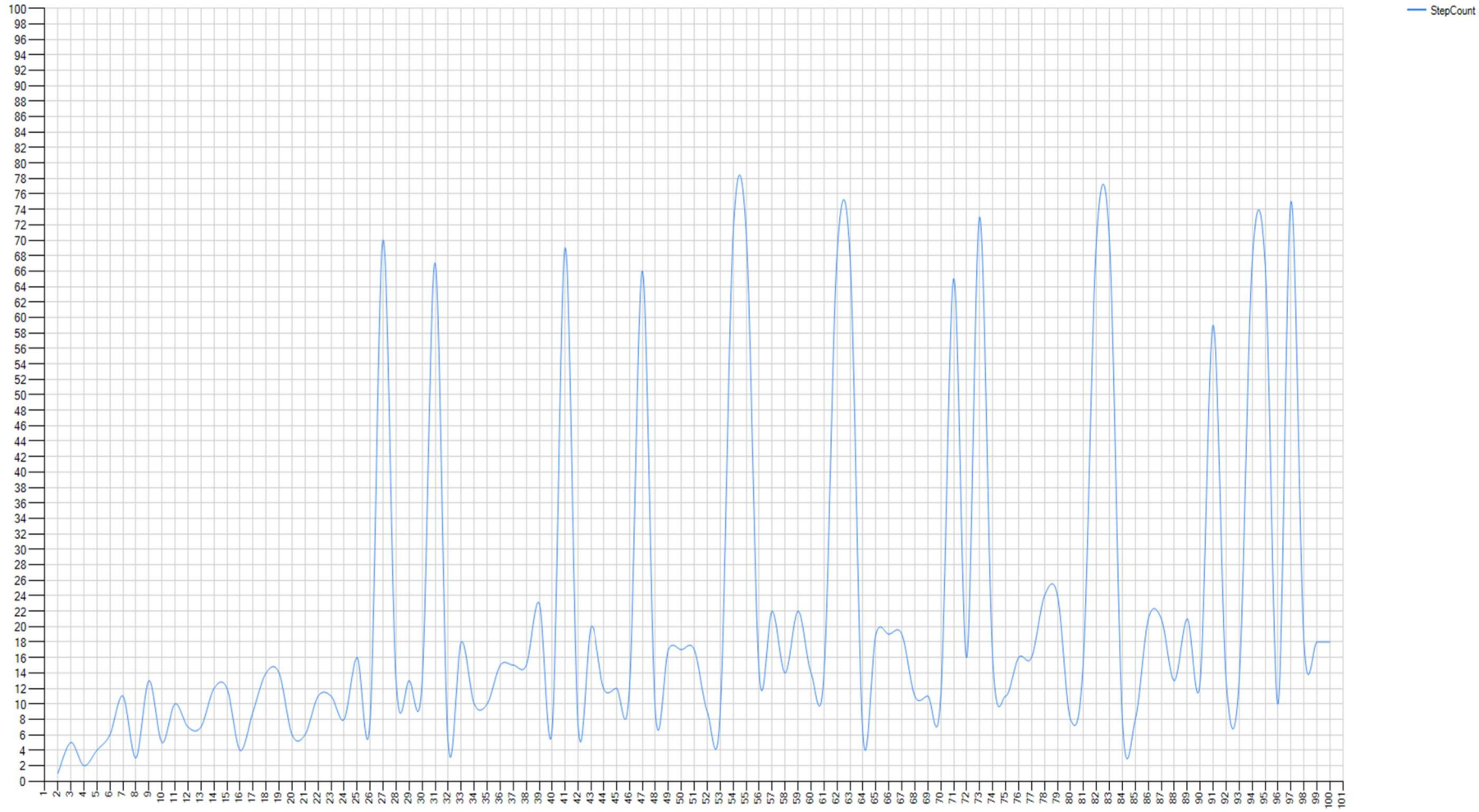
### 5.3.3.2 Program grafikleri

Collatz Konjektürü parite sekansı yönteminde hesaplama işlemlerini yapan bilgisayar programına ait tüm detaylar önceki bölümlerde incelenmiştir. Bu program haricinde yine parite sekansı yönteminde yapılan bu işlemlerin grafiklerini oluşturan bir bilgisayar programı daha geliştirilmiştir. Programda sayıların adım sayılarına göre oluşturulmuş sütun grafiği, sp çizgi grafiği, çizgi grafiği ve polar grafik geliştirmeleri yapılmıştır.

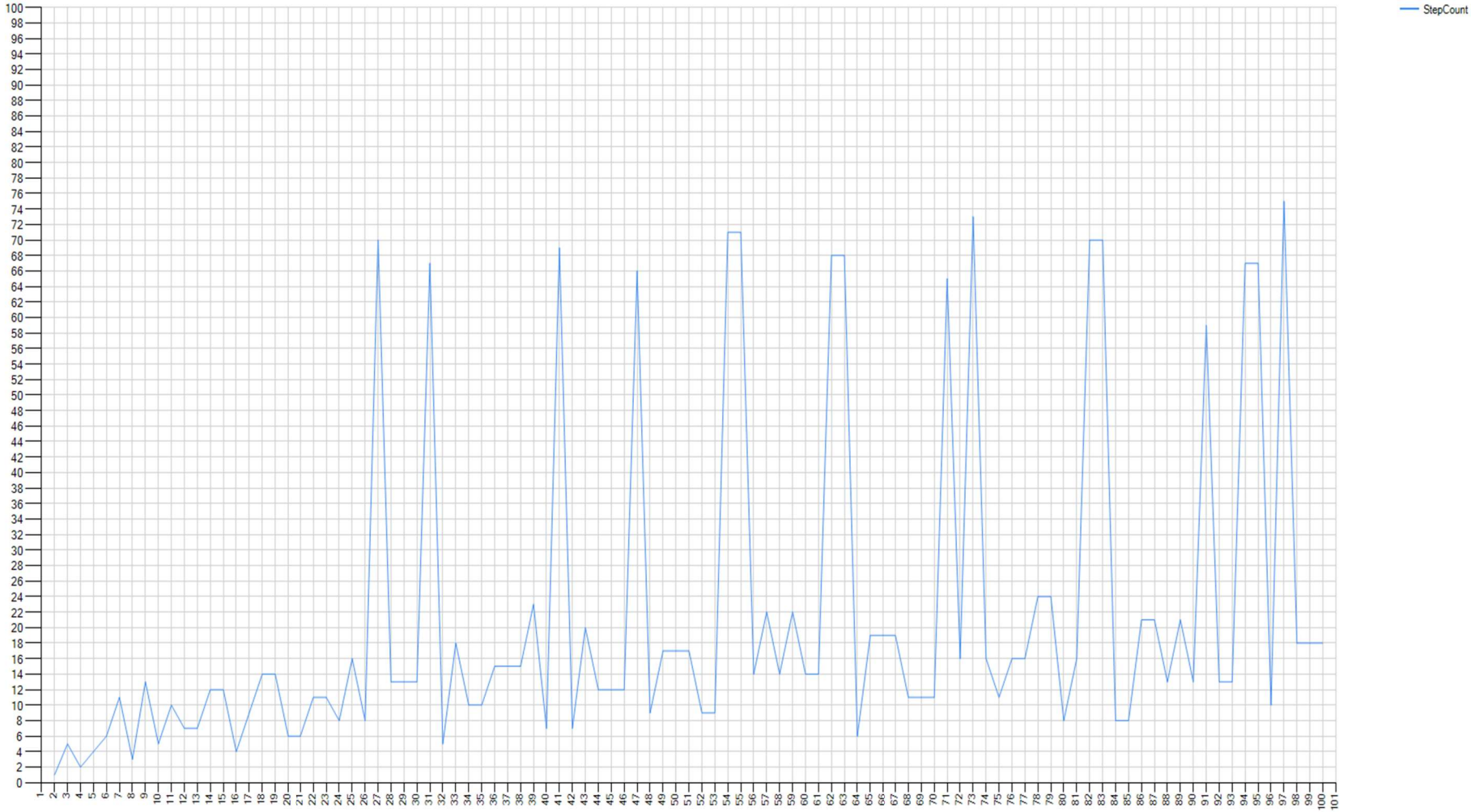
Geliştirilen program grafikleri aşağıda grafik bazında yer almaktadır;



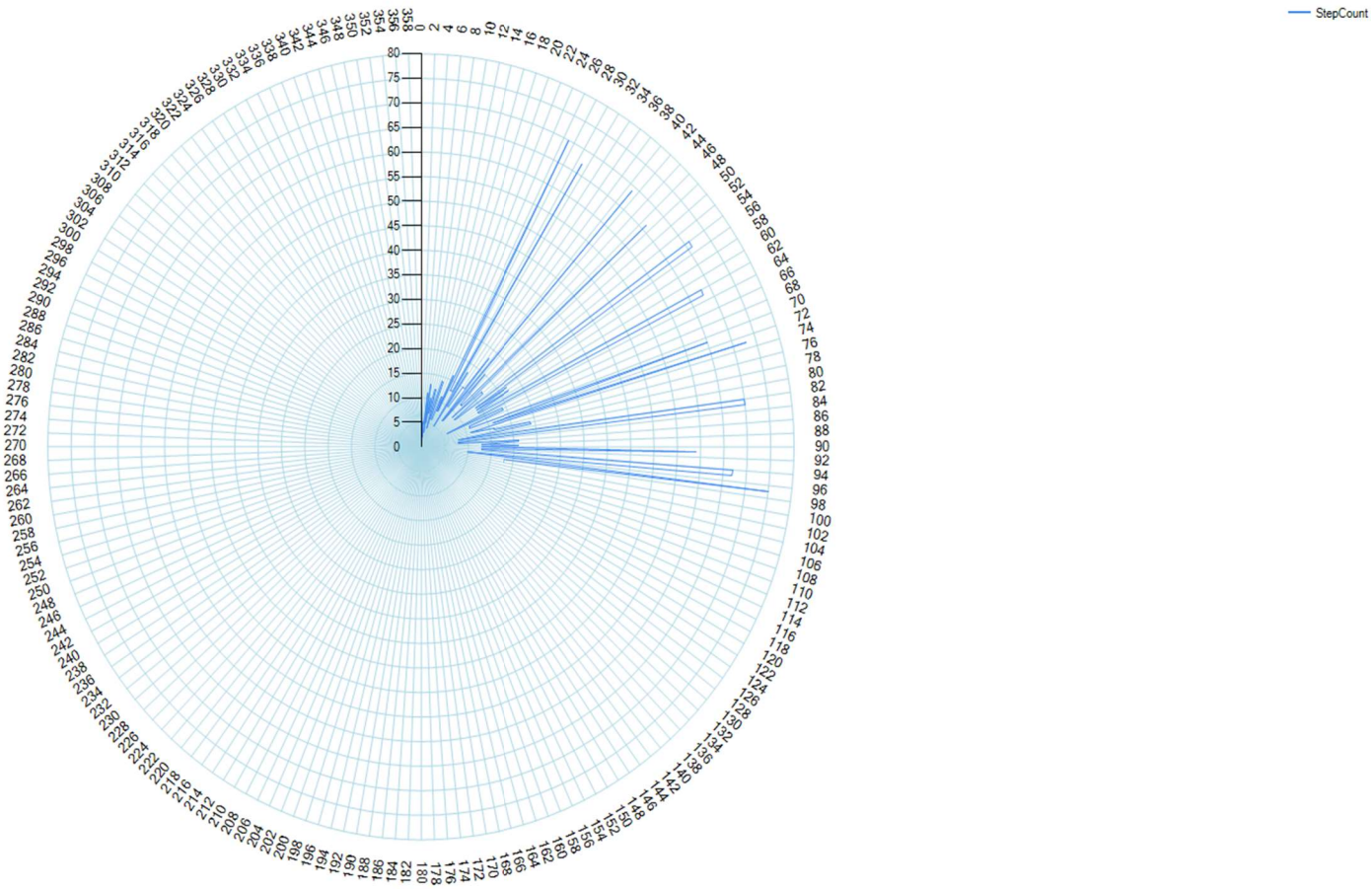

**Şekil 5.12 :** Collatz Konjektürü parite sekansı yönteminde sütun grafiği




**Şekil 5.13 :** Collatz Konjektürü parite sekansı yönteminte sp çizgi grafiği




**Şekil 5.14 :** Collatz Konjektürü parite sekansı yönteminde çizgi grafiği




**Şekil 5.15 :** Collatz Konjektürü parite sekansı yönteminde polar grafiği

### 5.3.3.3 Grafiksel gösterim program kodu

Collatz Konjektürü parite sekansı yönteminde yapılan hesaplama işlemlerinin grafiklerini oluşturan bilgisayar programı çıktıları incelenmiştir. Programda sayıların adım sayılarına göre oluşturulmuş sütun grafiği, sp çizgi grafiği, çizgi grafiği ve polar grafikleri oluşturulmuştur. Bu grafiklerin üretildiği program kodu aşağıda paylaşılmıştır;

**Program.cs**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CollatzConjecture
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new FrmCollatzConjecture());
        }
    }
}
```

**FrmCollatzConjecture.cs**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CollatzConjecture
{
    public partial class FrmCollatzConjecture : Form
    {
        public static int givenNumber = 100;
        int[] steps = new int[givenNumber];

        public FrmCollatzConjecture()
```

```
{
    InitializeComponent();
}

public static bool isEven(int number)
{
    bool answer = (number % 2 == 0) ? true : false;
    return answer;
}

public void CollatzConjecture()
{
    int result = 0;
    int stepCount = 0;

    for (int i = 2; i <= givenNumber; i++)
    {
        result = i;
        stepCount = 0;

        StringBuilder sequence = new StringBuilder("");

        sequence.Append($"{result} | {result} -");

        while (result != 1)
        {
            if (isEven(result))
            {
                result = result / 2;
                sequence.Append($" {result} -");
                stepCount++;
            }
            else
            {
                result = (result * 3 + 1) / 2;
                sequence.Append($" {result} -");
                stepCount++;
            }
        }

        sequence.Remove((sequence.Length - 1), 1);
        sequence.Append($" | {stepCount}");

        steps[i-2] = stepCount;
    }
}

private void FrmCollatzConjecture_Load(object sender, EventArgs e)
{
    CollatzConjecture();

    for (int i = 0; i < steps.Length - 1; i++)
    {
        var number = i + 2;
        var stepsCount = steps[i];
        this.columnChart.Series["StepCount"].Points.AddXY(number,
         stepsCount);
        this.spLineChart.Series["StepCount"].Points.AddXY(number,
         stepsCount);
```

```
                this.lineChart.Series["StepCount"].Points.AddXY(number,
                  stepsCount);
                this.funnelChart.Series["StepCount"].Points.AddXY(number,
                 stepsCount);
            }
        }
    }
```

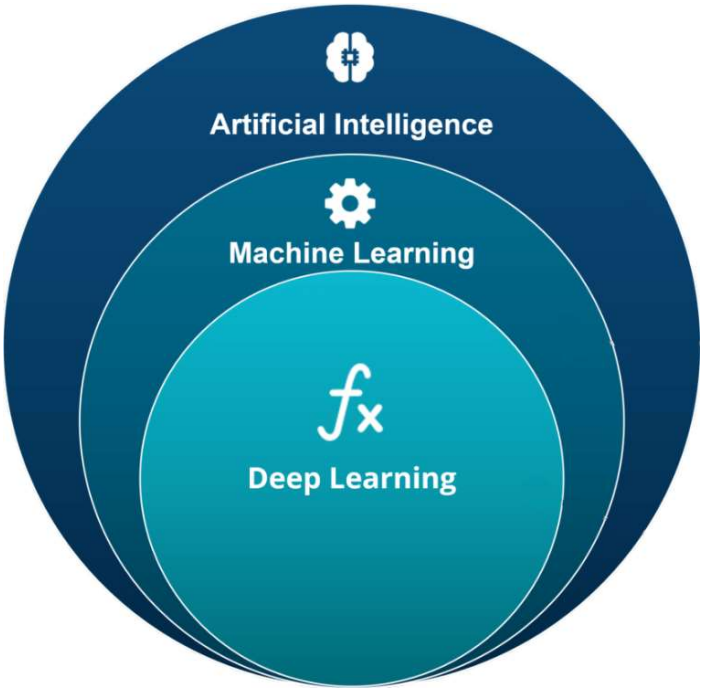
# 6. MAKİNE ÖĞRENMESİ

## 6.1 Makine Öğrenmesi

Makine Öğrenmesi (ML), bilgisayarların algılarını, bilişlerini ve eylemlerini deneyimle nasıl geliştirebileceklerini araştırdığımız, etkileyici bir Yapay Zeka (AI) araştırma ve uygulaması dalıdır. Makine öğrenmesi, verilerden, bilgiden, deneyimden ve etkileşimden gelişen makinelerle ilgilidir [51].

Yapay zeka ve bilgisayar oyunları alanında öncü olan Arthur Samuel, "Makine Öğrenmesi" ifadesini ortaya çıkaran kişidir. Makine öğrenmesini “Bilgisayarlara açıkça programlanmadan öğrenme yeteneği veren çalışma alanı” olarak tanımlamıştır.

Tom Mitchell ise makine öğrenmesinin iyi tasarlanmış bir matematiksel ve ilişkisel tanımını: "Bir bilgisayar programının P ile tanımlı performans ölçütlerine göre ölçümlenen değerlerde, bazı T ile tanımlanmış görevler bazında E ile tanımlanan deneyimle iyileşme sağlanıyorsa T görevlerinin E deneyimlerinden öğrendiği söylenir.” şeklinde ifade etmiştir. Özetle bir bilgisayarın belirli bir görevdeki performansı deneyim ile artarsa, bazı görev sınıflarıyla ilgili olarak deneyimlerden öğreniyor olduğu söylenir [52].




**Şekil 6.1:** Yapay zeka, makine öğrenmesi ve derin öğrenme gösterimi [53].

Makine öğrenmesi, deneyim ve veri kullanımıyla otomatik olarak gelişen bilgisayar algoritmaları üzerine yapılan çalışmadır [54]. Kazanım sağladığı öğrenme yeteneği sayesinde programlanmadan zaman içinde doğruluğunu artıran uygulamalar oluşturmaya odaklanır [55]. Makine öğrenmesi algoritmaları, açıkça programlanmadan tahminler veya kararlar almak için "eğitim verileri" olarak bilinen örnek verilere dayalı bir model oluşturur. Makine öğrenmesi algoritmaları, gerekli görevleri gerçekleştirmek için geleneksel algoritmalar geliştirmenin zor veya olanaksız olduğu çok çeşitli uygulamalarda da kullanılır.

Bilgisayarlara atanan basit görevler kapsamında, makineye eldeki sorunu çözmek için gereken tüm adımları nasıl uygulayacağını söyleyen algoritmalar programlamak mümkündür. Bu tür algoritmaların uygulanması için bilgisayar tarafında öğrenmeye gerek bulunmaz. Daha gelişmiş görevler içinse, bir insan tarafından gerekli algoritmaları tasarlamak ve oluşturmak zor olabilir. Pratikte, programcıların gerekli her adımı belirlemesinden ziyade, makinenin kendi algoritmasını geliştirmesine yardımcı olmak daha etkili olabilir [56].

Makine öğrenmesi aslında programlanmadan, yani herhangi bir insan yardımı olmadan bilgisayarların deneyimlerine dayanarak öğrenme sürecini

otomatikleştirmek ve iyileştirmek olarak açıklanabilir. Süreç kaliteli verileri beslemek, ardından verileri ve farklı algoritmaları kullanarak makine öğrenmesi modelleri oluşturup bilgisayarları eğitmekle başlar. Algoritma seçimi ise, ne tür veriye ve nasıl bir göreve sahip olduğumuza bağlıdır [52].

Makine öğrenmesi disiplini, bilgisayarlara tam olarak tatmin edici bir algoritmanın bulunmadığı görevleri gerçekleştirmeyi öğretmek için çeşitli yaklaşımlar kullanır. Çok sayıda potansiyel yanıtın olduğu durumlarda, doğru yanıtların bir kısmını geçerli olarak etiketlemek bir yaklaşımdır. Bu daha sonra bilgisayarın doğru yanıtları belirlemek için kullandığı algoritmaları geliştirmek için eğitim verisi olarak kullanılabilir [56].

Makine öğrenmesinde, algoritmalar yeni verilere dayalı kararlar ve tahminler yapmak için büyük miktarda veri içerisindeki kalıpları ve özellikleri bulmak üzere eğitilir. Algoritma ne kadar iyi olursa, daha fazla veri işledikçe kararlar ve tahminler o kadar doğru olur [55].

Makine öğrenmesi, istatistik, bilgi gösterimi, planlama ve kontrol, veritabanları, nedensel çıkarım, bilgisayar sistemleri, makine görüşü ve doğal dil işleme dahil olmak üzere birçok disiplinde temellere dayanan büyük ve karmaşık miktarlardaki bilgileri akıllıca işlemek için çeşitli teknikler kullanır.

Temelleri makine öğrenmesi olan yapay zeka bileşenleri, fenomenler hakkında tahminler sağlama, kararlar için önerilerde bulunma, talimatlar alma ve düzeltme dahil olmak üzere insanlarla çeşitli şekillerde etkileşim kurmayı da hedefler [51].

### 6.1.1 Makine öğrenmesi yöntemleri

#### 6.1.1.1 Denetimli öğrenme

Denetimli öğrenme, modelin etiketli bir veri kümesi üzerinde eğitilmesidir. Etiketli veri kümesi, hem giriş hem de çıkış parametrelerine sahiptir. Bu tür öğrenmede, hem eğitim hem de doğrulama veri kümeleri etiketlenir [57].

Denetimli makine öğrenmesi kendisini etiketli bir veri kümesi üzerinde eğitir. Yani veriler, makine öğrenmesi modelinin belirlemek için oluşturulduğu ve hatta modelin verileri sınıflandırması beklenen şekilde bile sınıflandırılabileceği bilgisiyle etiketlenir [55].

Optimal bir öğrenim işlevi, algoritmanın eğitim verilerinin bir parçası olmayan girdiler için çıktıları doğru bir şekilde belirlemesine izin verecektir. Zaman içinde çıktılarının veya tahminlerinin doğruluğunu artıran bir algoritmanın bu görevi gerçekleştirmeyi öğrendiği söylenir [54].

Denetimli öğrenme, diğer makine öğrenimi yöntemlerinden daha az eğitim verisi gerektirir ve modelin sonuçları gerçek etiketli sonuçlarla karşılaştırılabildiğinden eğitimi kolaylaştırır. Ancak, uygun şekilde etiketlenmiş verilerin hazırlanması daha maliyetlidir. Ayrıca aşırı uygunluk veya yeni verilerdeki varyasyonları doğru bir şekilde işlemeyecek kadar eğitim verilerine çok yakından bağlı ve önyargılı bir model oluşturma tehlikesi vardır [55].

Denetimli öğrenme model türleri arasında aşağıda açıklamalarına yer verilen sınıflandırma ve regresyon yer almaktadır.

- **Sınıflandırma:** Giriş değerlerini tanımlanmış etiketlere ya da sınıflara ayıran bir model bulma sürecidir. Başka bir deyişle, sınıflandırma giriş değerinin önceden tanımlanmış bir grubun parçası olup olmayacağını belirler [58]. Bir sınıflandırma algoritması, girdisini beklenen bir şekilde sınıflandırmasının doğruluğu hesaplanarak değerlendirilir.

- **Regresyon:** Giriş değerlerine dayalı olarak bir sürekli değeri tahmin eden model bulma sürecidir. Çıkış değeri bir etikete yani ayrık değere sahip değildir, ancak belirli aralıkta süreklidir. Buradaki amaç, modelimizin yapabileceği kadar gerçek çıktı değerine daha yakın bir değer tahmin etmektir ve ardından değerlendirme, hata değeri hesaplanarak yapılır. Hata ne kadar küçükse, regresyon modelinin doğruluğu o kadar artar [58] [57].

Benzerlik öğrenimi, regresyon ve sınıflandırmayla yakından ilgili denetimli makine öğrenmesinin bir alanıdır, ancak amaç iki nesnenin ne kadar benzer veya ilişkili olduğunu ölçen bir benzerlik fonksiyonu kullanarak örneklerden öğrenmektir. Sıralama, öneri sistemleri, görsel kimlik izleme, yüz doğrulama ve konuşmacı doğrulama gibi uygulamaları vardır.

Denetimli öğrenme algoritmaları aşağıda maddeler halinde sıralanmıştır;

- Linear Regression (Doğrusal Regresyon)
- Nearest Neighbor (En Yakın Komşu)
- Guassian Naive Bayes (Gauss Naive Bayes)
- Decision Trees (Karar Ağaçları)
- Support Vector Machine (SVM) (Destek Vektör Makinesi)
- Random Forest (Rastgele Orman)

#### 6.1.1.2 Denetimsiz öğrenme

Denetimsiz öğrenme, yalnızca giriş değerleri içeren bir veri kümesini alır ve veri noktalarının gruplanması veya kümelenmesi için veri içerisindeki yapıyı bulur. Bu nedenle algoritmalar etiketlenmemiş, sınıflandırılmamış veya kategorize edilmemiş verilerden öğrenir. Denetimsiz öğrenme algoritmaları verilerdeki ortak noktaları belirler ve her yeni veri parçasındaki bu tür ortak özelliklerin varlığına veya yokluğuna dayalı olarak yanıt verir.

Denetimsiz öğrenme etiketlenmemiş verileri alır ve bu verilerin insan müdahalesi olmadan gerçek zamanlı olarak gruplanması veya kümelenmesi amacıyla gereken anlamlı özellikleri çıkarmak için algoritmalar kullanır [55]. Burada makinenin görevi, önceden herhangi bir veri eğitimi almadan kendi başına, sıralanmamış bilgileri modellere, benzerliklere ve farklılıklara göre gruplamaktır [59].

Denetimsiz öğrenmenin amacı, veriler hakkında daha fazla bilgi edinmek için verilerdeki temel yapıyı veya dağılımı modellemektir [60]. Denetimsiz öğrenme kararların ve tahminlerin otomatikleştirilmesinden çok, insanların veriler içerisinde

gözden kaçırabileceği kalıpları ve ilişkileri belirlemeye yöneliktir [55]. Anormallik algılama gibi görevler için de kullanışlıdır.

Denetimsiz öğrenmede, denetimli öğrenmenin aksine doğru cevaplar yoktur ve model eğitimi yapılmaz [60]. Denetimsiz öğrenme algoritmaları, verilerdeki temel gizli özelliklere dayalı olarak verileri etiketlenmemiş bir veri kümesinde gruplandırır. Etiket olmadığı için sonucu değerlendirmenin bir yolu yoktur [55].

Denetimsiz öğrenme model türleri arasında aşağıda açıklamalarına yer verilen kümeleme ve ilişkilendirme yer almaktadır.

- **Kümeleme:** Veri kümesi içerisinde aranıp elde edilen çeşitli kalıplara göre, verileri gruplamak için uygulanır. Aynı küme içindeki veriler bir veya daha fazla kritere göre benzer olurken, farklı kümelerde yer alan veriler birbirine benzemez. Verileri, ölçütlere göre benzer olan gruplar halinde düzenleme modeli olarak da tanımlanabilir.

- **İlişkilendirme:** Büyük bir veri kümesinde yer alan özellikler arasındaki ilişkileri keşfetmek için kullanılır. İlginç bağlantılar ve ilişkiler bulunmasını sağlayan kural tabanlı bir model türüdür. Bazı "ilginçlik" ölçütlerini kullanarak veri kümesinde keşfedilen güçlü kuralları belirlemeyi amaçlar. İlişkilendirme kuralları adı verilen 'ise - o zaman' ilişkilerini tanımlar. İlginç kuralları diğer tüm ilişkilendirme kurallarından filtrelemek için önem düzeylerine uygulanan minimum eşikler vardır [61].

Denetimsiz öğrenme algoritmaları aşağıda maddeler halinde sıralanmıştır;

- K-Means Clustering (K-Ortalamalar Kümeleme)

- DBSCAN : Density-Based Spatial Clustering of Applications with Noise (Gürültülü Uygulamaların Yoğunluğa Dayalı Konumsal Kümelenmesi)

- BIRCH : Balanced Iterative Reducing and Clustering using Hierarchies (Hiyerarşileri Kullanarak Dengeli Yinelemeli Azaltma ve Kümeleme)

- Hierarchical Clustering (Hiyerarşik Kümeleme)

- Apriori (Doğrusal Regresyon)

- Eclat : Equivalence Class Transformation (Eşdeğerlik Sınıf Dönüşümü)

- FP-Growth : Frequent Pattern Growth (Sık Kalıp Büyümesi)

### 6.1.1.3 Pekiştirmeli öğrenme

Pekiştirmeli öğrenme bir ortamda kümülatif ödül kavramını en üst düzeye çıkarmak için nasıl harekete geçilmesi gerektiğiyle ilgili davranışsal bir makine öğrenmesi modelidir. Veri kümeleri kullanılarak model eğitimi yapılmaz. Model deneme yanılma yöntemini kullanarak ilerledikçe öğrenir. Belirli bir durumda en iyi davranışı geliştirmek için başarılı sonuçlar pekiştirilir [55].

Pekiştirmeli öğrenmede, bir ortamdaki durumlar ve belirli bir durumda olası eylemler söz konusudur. Öğrenme süreci sırasında, algoritma ortamdaki durum-eylem çiftlerini araştırır. Bu sayede durum-eylem çifti tablosu oluşturulur. Sonra öğrenilen bilginin uygulamasında, belirli bir durumda en iyi eylemi seçmek için bu durum-eylem çifti ödüllerini kullanır. Model, eğitim veri kümesinin yokluğunda deneyimlerinden öğrenmesi sayesinde gelişme gösterir [62].

Genelliği nedeniyle oyun teorisi, kontrol teorisi, işlem araştırması, bilgi teorisi, simülasyon tabanlı optimizasyon, çok ajanlı sistemler, sürü zekası, istatistik ve genetik algoritmalar gibi diğer birçok disiplinde incelenmektedir. Takviye öğrenme algoritmaları, otonom araçlarda veya bir insan rakibe karşı oyun oynamayı öğrenmede de kullanılır [63].

Pekiştirmeli öğrenme model türleri arasında aşağıda açıklamalarına yer verilen olumlu ve olumsuz pekiştirme yer almaktadır.

- **Olumlu Pekiştirme:** İyi sonuç alma olasılığını artırmak amacıyla her iyi sonuç için bir ödül eklenir. Davranışın gücünü, sıklığını ve tamamlanma olasılığını artırması olarak tanımlanır. Davranış üzerinde olumlu bir etkisi vardır.

Olumlu pekiştirme modelinin avantajları aşağıda maddeler halinde belirtilmiştir;

  - Performansı en üst düzeye çıkarır.
  - Değişimin daha uzun süre sürdürülmesini sağlar.

Olumlu pekiştirme modelinin dezavantajları ise aşağıdaki gibidir;

  - Olumlu pekiştirmenin çok fazla olması, sonuçları etkileyebilecek durumların aşırı yüklenmesine neden olabilir [64].

- **Olumsuz Pekiştirme:** Performansı artırmak için olumsuz bir durumun ortadan kaldırılmasına yönelik ilerlenir. Olumsuz koşulların durdurulması veya önlenmesi nedeniyle bir davranşın güçlendirilmesi de söz konusudur.

Olumsuz pekiştirme modelinin avantajları aşağıda maddeler halinde belirtilmiştir;

  - Görevi yerine getirme davranışını artırır.
  - Daha iyi sonuçlar elde edilmesini sağlar.

Olumsuz pekiştirme modelinin dezavantajları ise aşağıdaki gibidir;

  - Davranışı gereken minimum gereksinimi karşılamaya yetecek kadar zorlamasıdır [64].

Pekiştirmeli öğrenme algoritmaları aşağıda maddeler halinde sıralanmıştır;

- Q-Learning (Q-Öğrenme)

- SARSA : State Action Reward State Action
  (Durum Eylem Ödül Durum Eylem)

- DQN : Deep Q Network
  (Derin Q Ağı)

- TD : Temporal Difference
  (Zamansal Fark)

- GAN : Generative Adversarial Network
  (Üretken Çekişmeli Ağ)

### 6.1.2 Collatz Konjektürü'ne makine öğrenmesi ile yaklaşım

#### 6.1.2.1 Metodoloji

Bu bölümde Collatz Konjektürü'ne makine öğrenmesi ile yaklaşım incelenecektir. Yaklaşım makine öğrenmesi metotlarından denetimli öğrenme kapsamında yer almaktadır. Model, denetimli makine öğrenmesi algoritmalarından doğrusal regresyon (linear regression) özelinde uygulanacaktır. Gerek metot seçimi gerekse algoritma seçimi ise, ne tür veriye ve nasıl bir göreve sahip olduğumuza bağlıdır. Bu nedenle seçenekler fonksiyonun ve uygulamanın genel yapısı ile veri seti dikkate alınarak değerlendirilmiş ve en uygun olan seçim yapılmıştır.

#### 6.1.2.2 Eğitim verisi

Denetimli makine öğrenmesi metodunda algoritmalar, açıkça programlanmadan tahminler veya kararlar almak için "eğitim verileri" olarak bilinen örnek verilere dayalı bir model oluşturur. Süreç ise modelin etiketli bir veri kümesi üzerinde eğitilmesidir. Etiketli veri kümesi, hem giriş hem de çıkış parametrelerine sahiptir. Bu tür öğrenmede, hem eğitim hem de doğrulama veri kümeleri etiketlenir.

Bölüm 5.1.2.3'de yer alan "Program kodu" başlığı altında Collatz Konjektürü'nün standart yöntem ile hesaplanması için geliştirilen bilgisayar programına ait kaynak kodlarına yer verilmiştir. Bilgisayar programı üzerinde ekstra geliştirmeler yapılarak, program çıktılarından .csv formatında bir sonuç listesi üretecek şekilde tasarlanmıştır.

Bilgisayar programından elde edilen .csv formatlı dosyada veri, fonksiyona giren sayıları temsil eden " $Number$ " başlığında ve sayının toplam kaç adımda hesaplandığı bilgisini içeren "$StepCount$" başlığı ile etiketlenerek sınıflandırılmıştır.

Bu bilgisayar programı sayesinde üretilen .csv formatındaki dosya makine öğrenmesi yaklaşımında kullanılmak üzere eğitim verisi olarak hazırlanmıştır.

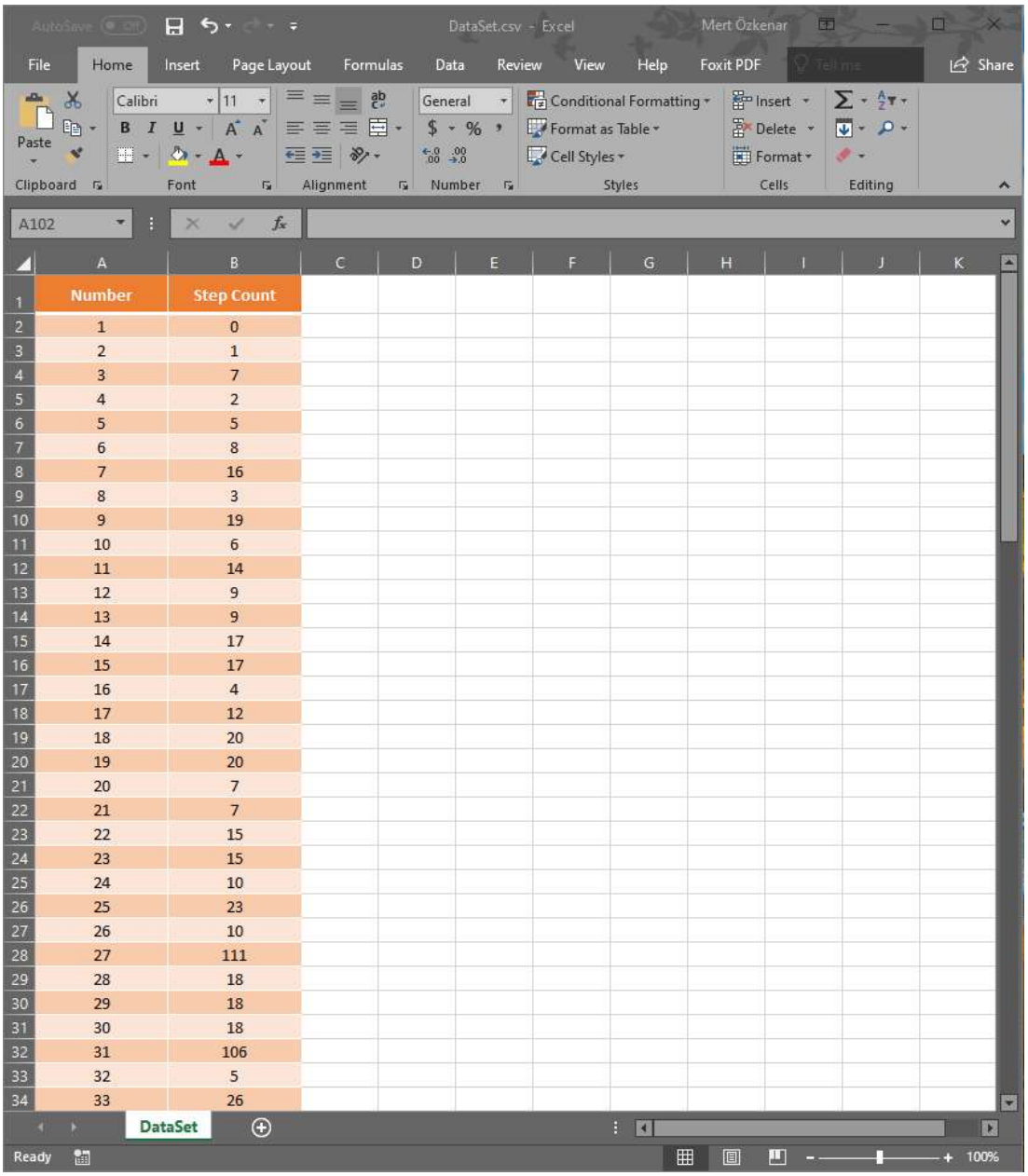
Bilgisayar programına ait program kodu aşağıda paylaşılmıştır;

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.Office.Interop.Excel;



namespace CollatzConjecture
{
    class Program
    {
        public static bool isEven(int number)
        {
            bool answer = (number % 2 == 0) ? true : false;
            return answer;
        }

        public static void CollatzConjecture()
        {
            Application excel = new Application();

            excel.Visible = true;

            object Missing = Type.Missing;
            Workbook workbook = excel.Workbooks.Add(Missing);

            Worksheet sheet1 = (Worksheet)workbook.Sheets[1];
```

```
int StartingColumn = 1;
int StartingRow = 1;

((Range)sheet1.Cells[StartingRow, StartingColumn + 0]).Value2
  = "Number";
((Range)sheet1.Cells[StartingRow, StartingColumn + 1]).Value2
  = "Step Count";

Range FirstColumnSelectedRow = (Range)sheet1.Cells[StartingRow
  + 1, StartingColumn + 0];
FirstColumnSelectedRow.Value2 = 1;
FirstColumnSelectedRow.Select();

Range SecondColumnSelectedRow = (Range)sheet1.Cells[StartingRo
  w + 1, StartingColumn + 1];
SecondColumnSelectedRow.Value2 = 0;
SecondColumnSelectedRow.Select();


int givenNumber = 100;
int result = 0;
int stepCount = 0;

Console.WriteLine("\n-----------------------------");

for (int i = 2; i <= givenNumber; i++)
{
    result = i;
    stepCount = 0;

    StringBuilder sequence = new StringBuilder("");

    sequence.Append($"{result} | {result} -");

    while (result != 1)
    {


        if (isEven(result))
        {
            result = result / 2;
            sequence.Append($" {result} -");
            stepCount++;
        }
        else
        {
            result = result * 3 + 1;
            sequence.Append($" {result} -");
            stepCount++;
        }
    }

    sequence.Remove((sequence.Length - 1), 1);
    sequence.Append($" | {stepCount}");

    Console.WriteLine($"{sequence}\n");
    Console.WriteLine("-----------------------------");
```

```
                FirstColumnSelectedRow = (Range)sheet1.Cells[StartingRow +
                 i, StartingColumn + 0];
                FirstColumnSelectedRow.Value2 = i;
                FirstColumnSelectedRow.Select();

                SecondColumnSelectedRow = (Range)sheet1.Cells[StartingRow
                 + i, StartingColumn + 1];
                SecondColumnSelectedRow.Value2 = stepCount;
                SecondColumnSelectedRow.Select();
            }
        }



        static void Main(string[] args)
        {

            CollatzConjecture();

            Console.ReadKey();

        }
    }
}
```

Aşağıda hazırlanan eğitim verisine ait ekran görüntüsü paylaşılmıştır.



| Number | Step Count |
|---|---|
| 1 | 0 |
| 2 | 1 |
| 3 | 7 |
| 4 | 2 |
| 5 | 5 |
| 6 | 8 |
| 7 | 16 |
| 8 | 3 |
| 9 | 19 |
| 10 | 6 |
| 11 | 14 |
| 12 | 9 |
| 13 | 9 |
| 14 | 17 |
| 15 | 17 |
| 16 | 4 |
| 17 | 12 |
| 18 | 20 |
| 19 | 20 |
| 20 | 7 |
| 21 | 7 |
| 22 | 15 |
| 23 | 15 |
| 24 | 10 |
| 25 | 23 |
| 26 | 10 |
| 27 | 111 |
| 28 | 18 |
| 29 | 18 |
| 30 | 18 |
| 31 | 106 |
| 32 | 5 |
| 33 | 26 |

**Çizelge 6.1 :** Hazırlanan eğitim verisini içeren kümenin kısmi gösterimi.

### 6.1.2.3 Model mekanizması

Eğitim verisinde yer alan "*Number*" bağımsız değişkendir. Tahmin edici, açıklayıcı değişken olarak da ifade edilebilmektedir. X eksenini temsil edecektir. "*StepCount*" ise bağımlı değişkendir. Hedef değişkendir ve tahmin etmeye çalışılacak olan değişken olarak yer alacaktır. Y eksenini temsil edecektir.

Denetimli öğrenme model türleri arasında yer alan sınıflandırma ve regresyon seçeneklerinden yapıya uygun olan regresyon ile çalışma yapılacaktır. Doğrusal

regresyon giriş değerlerine dayalı olarak bir sürekli değeri tahmin eden model bulma sürecidir. Buradaki amaç, modelimizin yapabileceği kadar gerçek çıktı değerine daha yakın bir değer tahmin etmektir ve ardından değerlendirme, hata değeri hesaplanarak yapılır. Hata ne kadar küçükse, regresyon modelinin doğruluğu o kadar artar.

Doğrusal regresyonda eğitim veri kümesinde yer alan veri noktaları incelenir. Veri noktalarına en uygun olan regresyon doğrusu çizilir.

Bilinmeyen değerlerin tahmini yapılırken ise hesaplama esnasında gerçekleştirilen matematiksel işlem aşağıdaki şekilde formüle edilmektedir.

$$\hat{y} = a + mx$$

$$a = \frac{[(\sum y)(\sum x^2) - (\sum x)(\sum xy)]}{[n(\sum x^2) - (\sum x)^2]}$$

$$m = \frac{[n(\sum xy) - (\sum x)(\sum y)]}{[n(\sum x^2) - (\sum x^2)]}$$

$\boldsymbol{\hat{y}}$ : $tahmin\ edilen\ adım\ sayısı\ (bağımlı\ değişken)$

$\boldsymbol{a}$ : $y\ eksenini\ kesme\ noktası\ (intercept)$

$\boldsymbol{m}$ : $eğim\ (slope\ coefficient)$

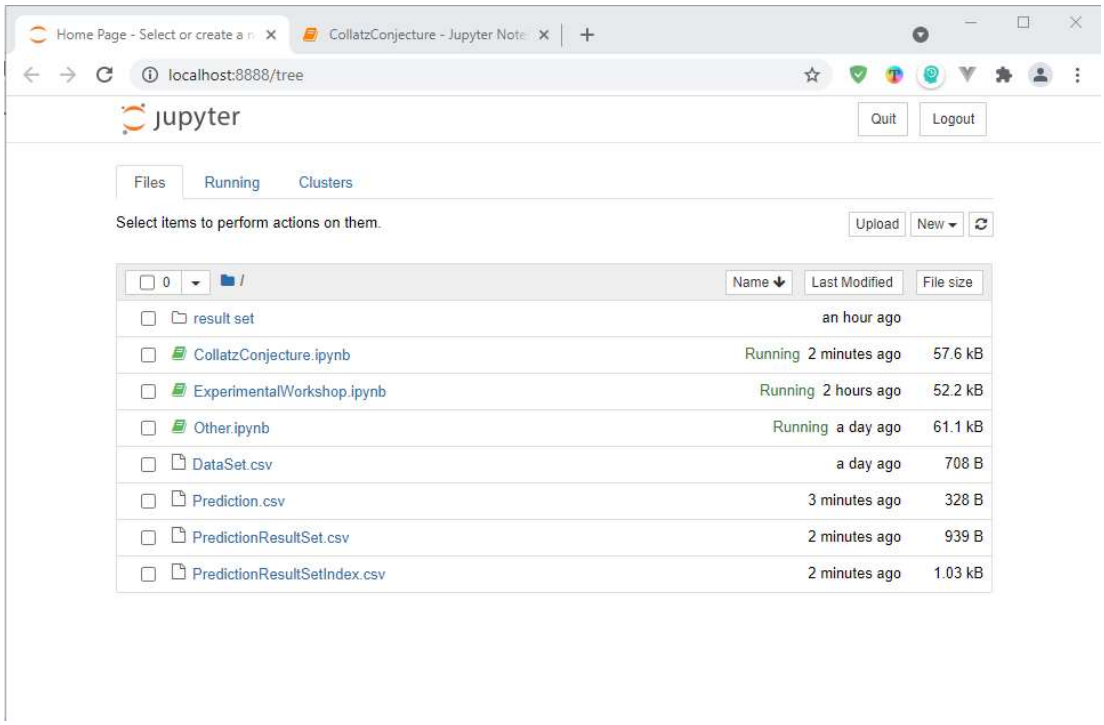
$\boldsymbol{x}$ : $sayı\ (bağımsız\ değişken)$

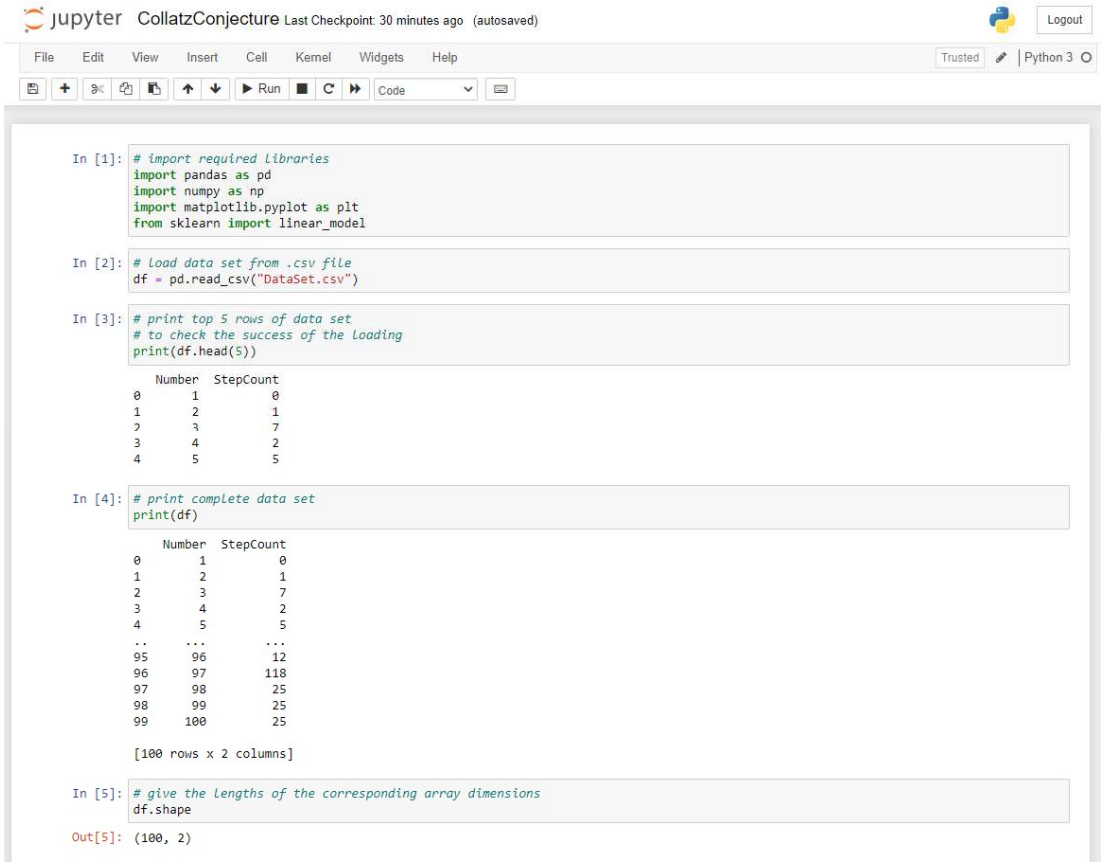
**(6.1)**

Doğrusal regresyonda veri ile en iyi uyum sağlayan doğrunun $a$ ve $m$ değerlerini bulunarak yapılan matematiksel hesaplamada kullanılır. Formülde yerini alan tüm değerler işleme alınır, sonuç olarak tahmini değer elde edilir.

Doğrusal hata veri noktası ile regresyon doğrusu arasındaki uzaklıktır. Hata her veri noktası için bulunur ve toplamı alınır. Toplam hatanın en az olduğu noktada en iyi regresyon doğrusu bulunmuş olur. Matematikte bir ifadeyi en aza indirmek için o ifadenin türevi alınır ve sıfıra eşitlenir.

### 6.1.2.4 Program kodu



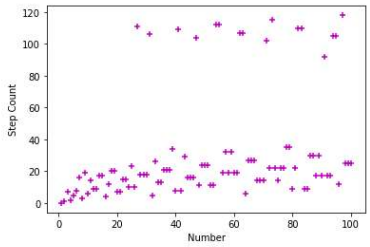
In [6]:
```
# view some basic statistical details
df.describe()
```

Out[6]:

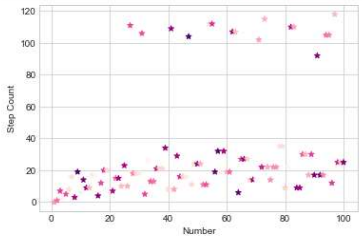
| | Number | StepCount |
|---|---|---|
| count | 100.000000 | 100.000000 |
| mean | 50.500000 | 31.420000 |
| std | 29.011492 | 34.460981 |
| min | 1.000000 | 0.000000 |
| 25% | 25.750000 | 11.750000 |
| 50% | 50.500000 | 19.000000 |
| 75% | 75.250000 | 27.500000 |
| max | 100.000000 | 118.000000 |

In [7]:
```
# plot the actual points as scatter plot
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.style.use("default")
plt.scatter(df.Number,df.StepCount,color='m',marker='+')
```

Out[7]:
```
<matplotlib.collections.PathCollection at 0x19db8e2ab20>
```




In [8]:
```
# plot the actual points as scatter plot
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.style.use("seaborn-whitegrid")
colors = np.random.randint(100, size=(100))
plt.scatter(df.Number,df.StepCount, c=colors, cmap='RdPu',marker='*')
plt.savefig('ActualPointsScatterStyle.png')
```




In [9]:
```
# create linear regression object
reg=linear_model.LinearRegression()
```

In [10]:
```
# train the model using the training sets
reg.fit(df[['Number']],df.StepCount)
```

Out[10]:
```
LinearRegression()
```

In [11]:
```
# predict the step count of given number
reg.predict([[121]])
```

Out[11]:
```
array([59.70714671])
```

In [12]:
```
# regression coefficient
reg.coef_
```

Out[12]:
```
array([0.40123612])
```

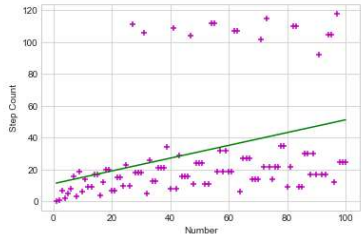
In [13]:
```
# regression interception
reg.intercept_
```

Out[13]:
```
11.157575757575746
```

In [14]:
```
# proof the accuracy of the predicted value
# ŷ = a+mx
11.157575757575746 + 0.40123612 * 121
```

Out[14]:
```
59.707146277575745
```

In [15]:
```
# plot the regression line
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.scatter(df.Number,df.StepCount,color='m',marker='+')
plt.plot(df.Number,reg.predict(df[['Number']]),color='g')
plt.savefig('RegressionLine.png')
```




In [16]:
```
# Load dataset of numbers to be predicted from .csv file
d = pd.read_csv("Prediction.csv")
```

In [17]:
```
# print top 5 rows of data set
# to check the success of the loading
print(d.head(5))
```

```
   Number
0     212
1     580
2     391
3    1903
4     750
```

In [18]:
```
# print complete data set
print(d)
```

```
          Number
0            212
1            580
2            391
3           1903
4            750
5           2021
6           9729
7          38467
8         461072
9         869664
10        781484
11       2439596
12       6616000
13      57055364
14     530987841
15      21720392
16     994308764
17      56786824
18     687686553
19       2216330
20     896536534
21        301302
22     146758597
23    3416546565
24    1684656326
25  76565321312
26  96134465518
27  41258768419
28  23784964287
29  19736485234
30  47182937821
31  52794618375
32  72589031457
```

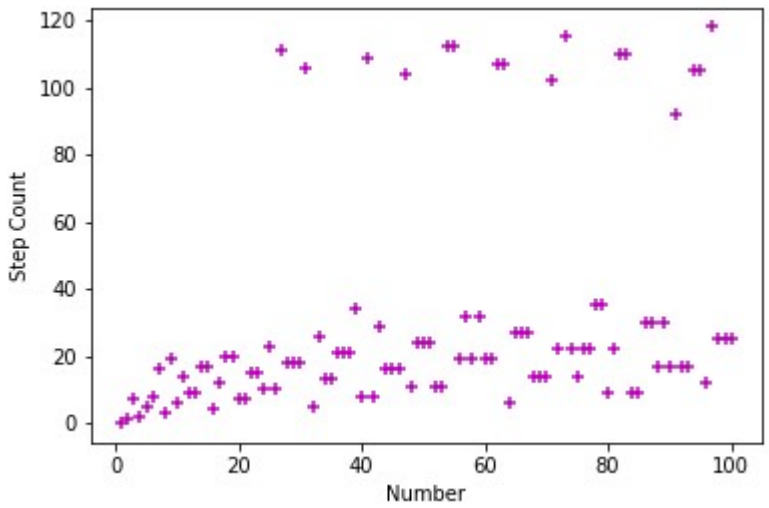
In [19]:
```
# predict the estimated step counts of given numbers in data set
p = reg.predict(d)
```

In [20]:
```
# entitle predicted step counts as StepCount
d['StepCount'] = p
```

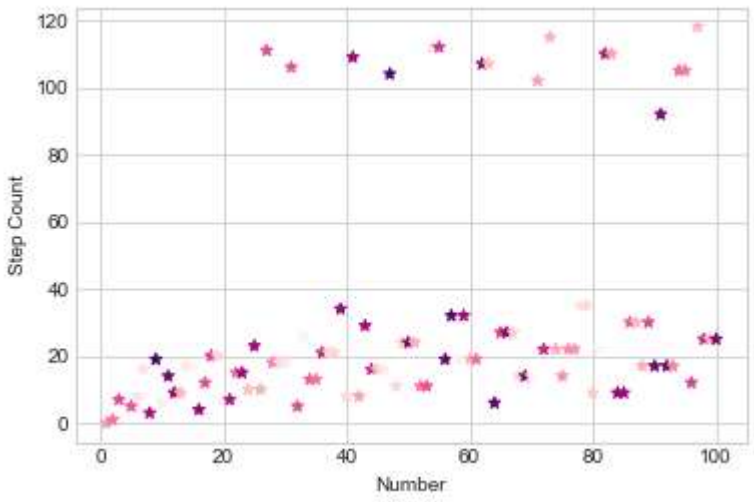
```
In [21]: # print the complete result set
         print(d)
```

| | Number | StepCount |
|---|---|---|
| 0 | 212 | 9.621963e+01 |
| 1 | 580 | 2.438745e+02 |
| 2 | 391 | 1.680409e+02 |
| 3 | 1903 | 7.747099e+02 |
| 4 | 750 | 3.120847e+02 |
| 5 | 2021 | 8.220558e+02 |
| 6 | 9729 | 3.914784e+03 |
| 7 | 38467 | 1.544551e+04 |
| 8 | 461072 | 1.850099e+05 |
| 9 | 869664 | 3.489518e+05 |
| 10 | 781484 | 3.135708e+05 |
| 11 | 2439596 | 9.788652e+05 |
| 12 | 6616000 | 2.654589e+06 |
| 13 | 57055364 | 2.289268e+07 |
| 14 | 530987841 | 2.130515e+08 |
| 15 | 21720392 | 8.715017e+06 |
| 16 | 994308764 | 3.989526e+08 |
| 17 | 56786824 | 2.278494e+07 |
| 18 | 687686553 | 2.759247e+08 |
| 19 | 2216330 | 8.892828e+05 |
| 20 | 896536534 | 3.597229e+08 |
| 21 | 301302 | 1.209044e+05 |
| 22 | 146758597 | 5.888486e+07 |
| 23 | 3416546565 | 1.370842e+09 |
| 24 | 1684656326 | 6.759450e+08 |
| 25 | 76565321312 | 3.072077e+10 |
| 26 | 96134465518 | 3.857262e+10 |
| 27 | 41258768419 | 1.655451e+10 |
| 28 | 23784964287 | 9.543387e+09 |
| 29 | 19736485234 | 7.918991e+09 |
| 30 | 47182937821 | 1.893150e+10 |
| 31 | 52794618375 | 2.118311e+10 |
| 32 | 72589031457 | 2.912534e+10 |

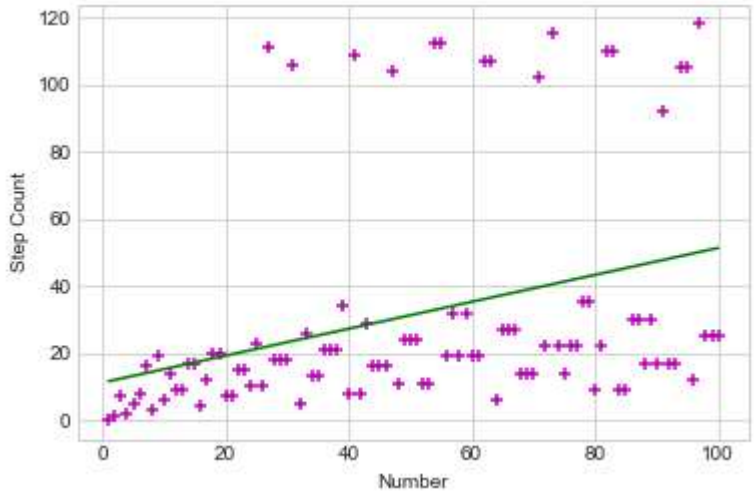
```
In [22]: # export the complete result set to .csv file
         d.to_csv("PredictionResultSetIndex.csv")

In [23]: # export the complete result set to .csv file without indexing
         d.to_csv("PredictionResultSet.csv",index=False)

In [ ]:
```




**Şekil 6.2:** Eğitim verisinde gerçekleşen değerlerin grafiği.

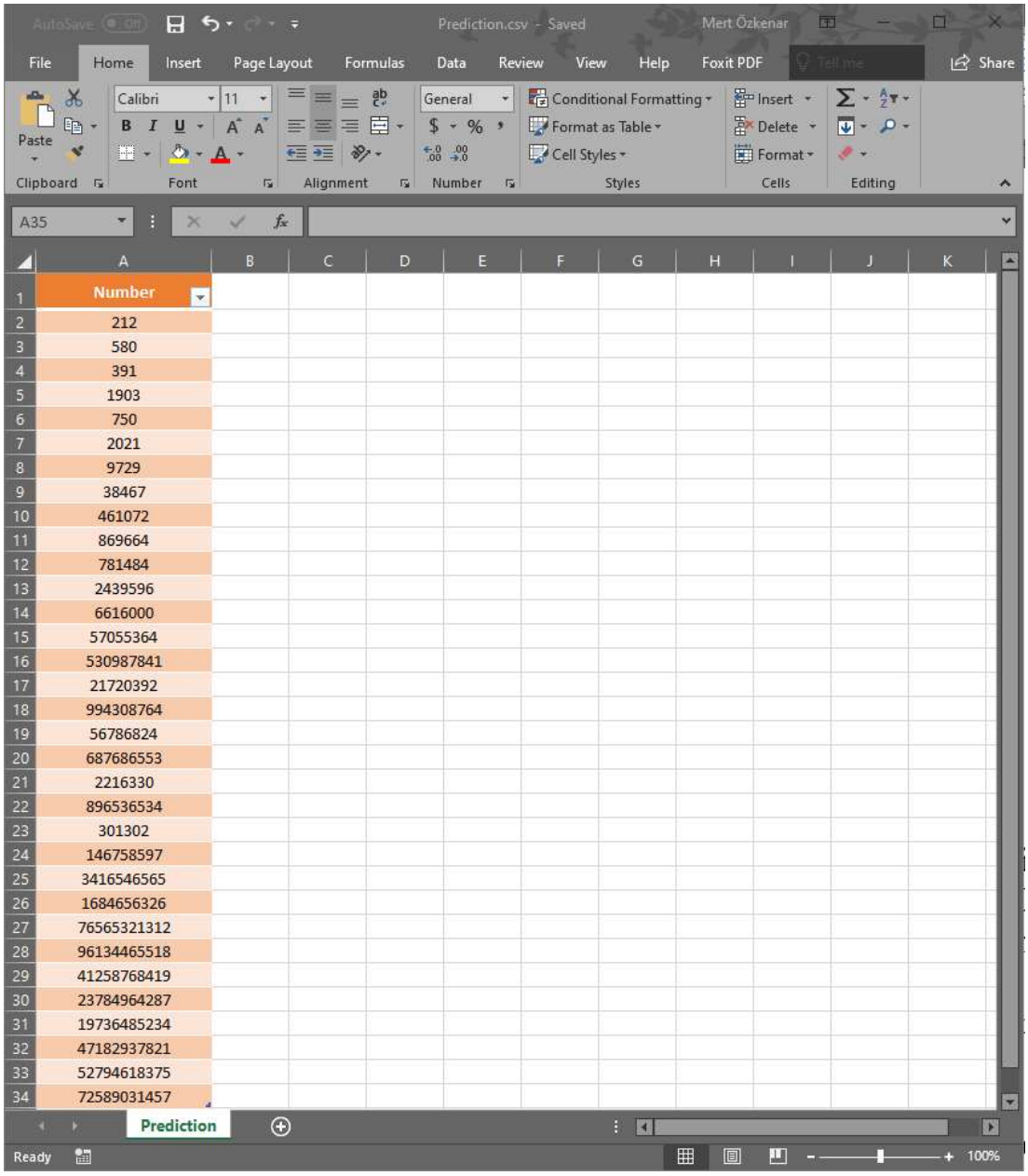



**Şekil 6.3:** Eğitim verisini oluşturan değerlerin noktasal gösterimi.
(Kılavuz çizgiler üzerinde ve çeşitli renk değerleri ile stillendirilmiş versiyon)



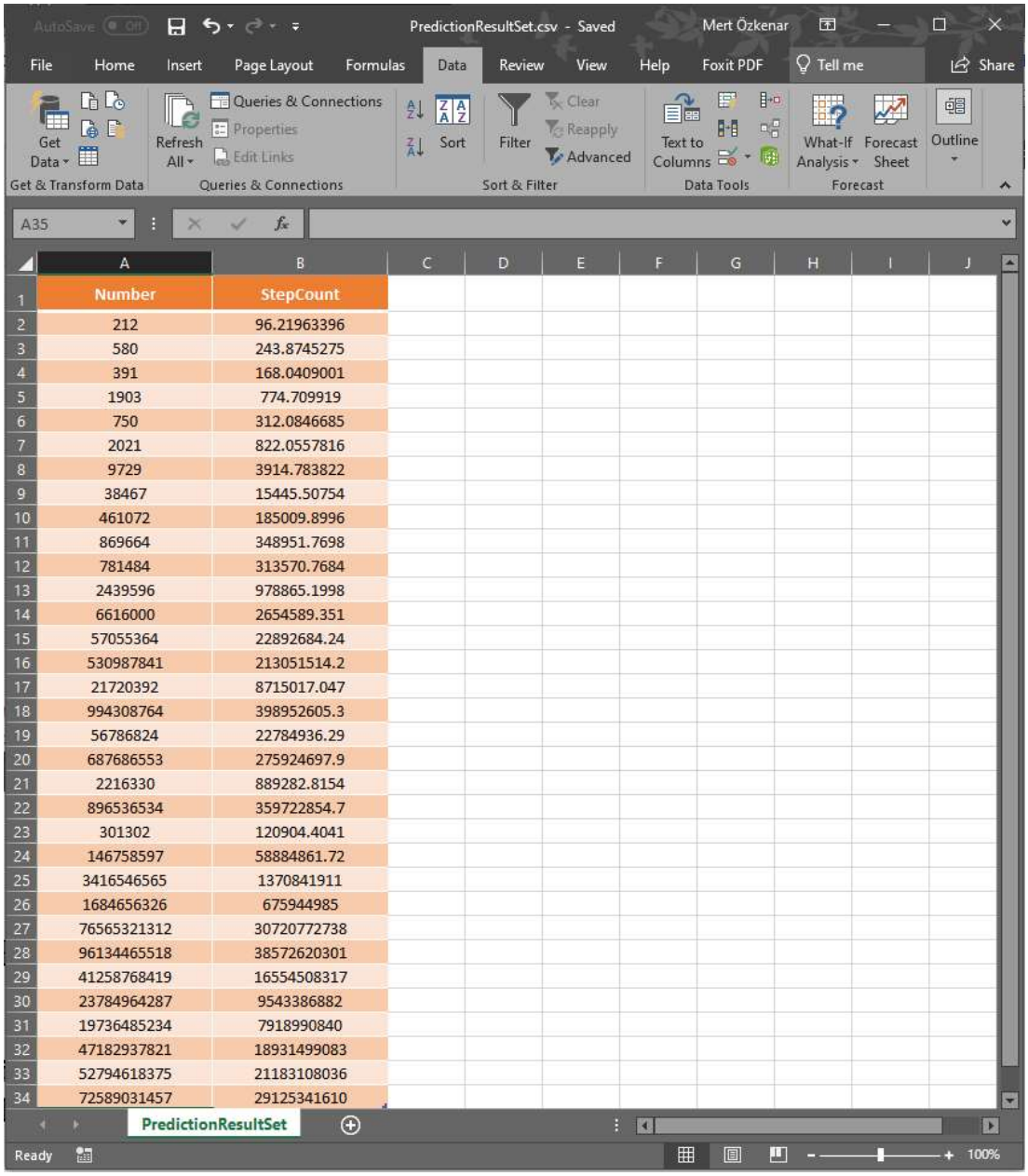

**Şekil 6.4:** Regresyon çizgisinin grafik üzerinde gösterimi.

**Çizelge 6.2 :** Adım sayısı tahmin edilmesi istenen sayılardan oluşan veri kümesi.



| | Number |
|---|---|
| 2 | 212 |
| 3 | 580 |
| 4 | 391 |
| 5 | 1903 |
| 6 | 750 |
| 7 | 2021 |
| 8 | 9729 |
| 9 | 38467 |
| 10 | 461072 |
| 11 | 869664 |
| 12 | 781484 |
| 13 | 2439596 |
| 14 | 6616000 |
| 15 | 57055364 |
| 16 | 530987841 |
| 17 | 21720392 |
| 18 | 994308764 |
| 19 | 56786824 |
| 20 | 687686553 |
| 21 | 2216330 |
| 22 | 896536534 |
| 23 | 301302 |
| 24 | 146758597 |
| 25 | 3416546565 |
| 26 | 1684656326 |
| 27 | 76565321312 |
| 28 | 96134465518 |
| 29 | 41258768419 |
| 30 | 23784964287 |
| 31 | 19736485234 |
| 32 | 47182937821 |
| 33 | 52794618375 |
| 34 | 72589031457 |

**Çizelge 6.3 :** Programın istenilen tahminleri yaparak oluşturduğu veri kümesi.



PredictionResultSet.csv - Saved — Mert Özkenar

| Number | StepCount |
|---|---|
| 212 | 96.21963396 |
| 580 | 243.8745275 |
| 391 | 168.0409001 |
| 1903 | 774.709919 |
| 750 | 312.0846685 |
| 2021 | 822.0557816 |
| 9729 | 3914.783822 |
| 38467 | 15445.50754 |
| 461072 | 185009.8996 |
| 869664 | 348951.7698 |
| 781484 | 313570.7684 |
| 2439596 | 978865.1998 |
| 6616000 | 2654589.351 |
| 57055364 | 22892684.24 |
| 530987841 | 213051514.2 |
| 21720392 | 8715017.047 |
| 994308764 | 398952605.3 |
| 56786824 | 22784936.29 |
| 687686553 | 275924697.9 |
| 2216330 | 889282.8154 |
| 896536534 | 359722854.7 |
| 301302 | 120904.4041 |
| 146758597 | 58884861.72 |
| 3416546565 | 1370841911 |
| 1684656326 | 675944985 |
| 76565321312 | 30720772738 |
| 96134465518 | 38572620301 |
| 41258768419 | 16554508317 |
| 23784964287 | 9543386882 |
| 19736485234 | 7918990840 |
| 47182937821 | 18931499083 |
| 52794618375 | 21183108036 |
| 72589031457 | 29125341610 |

```
# import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model

# load data set from .csv file
df = pd.read_csv("DataSet.csv")

# print top 5 rows of data set
# to check the success of the loading
print(df.head(5))

# print complete data set
print(df)

# give the lengths of the corresponding array dimensions
df.shape

# view some basic statistical details
df.describe()

# plot the actual points as scatter plot
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.style.use("default")
plt.scatter(df.Number,df.StepCount,color='m',marker='+')
plt.savefig('ActualPointsScatter.png')

# plot the actual points as scatter plot
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.style.use("seaborn-whitegrid")
colors = np.random.randint(100, size=(100))
plt.scatter(df.Number,df.StepCount, c=colors, cmap='RdPu',marker='*')
plt.savefig('ActualPointsScatterStyle.png')

# create linear regression object
reg=linear_model.LinearRegression()

# train the model using the training sets
reg.fit(df[['Number']],df.StepCount)

# predict the step count of given number
reg.predict([[121]])

# regression coefficient
reg.coef_

# regression interception
reg.intercept_

# proof the accuracy of the predicted value
# ŷ = a+mx
11.157575757575746 + 0.40123612 * 121
```

```
# plot the regression line
%matplotlib inline
plt.xlabel('Number')
plt.ylabel('Step Count')
plt.scatter(df.Number,df.StepCount,color='m',marker='+')
plt.plot(df.Number,reg.predict(df[['Number']]),color='g')
plt.savefig('RegressionLine.png')

# load dataset of numbers to be predicted from .csv file
d = pd.read_csv("Prediction.csv")

# print top 5 rows of data set
# to check the success of the loading
print(d.head(5))

# print complete data set
print(d)

# predict the estimated step counts of given numbers in data set
p = reg.predict(d)

# entitle predicted step counts as StepCount
d['StepCount'] = p

# print the complete result set
print(d)

# export the complete result set to .csv file
d.to_csv("PredictionResultSetIndex.csv")

# export the complete result set to .csv file without indexing
d.to_csv("PredictionResultSet.csv",index=False)
```

# 7. SONUÇ VE ÖNERİLER

Bilgisayarlar günümüz dünyasının ilerleyişinde gelinen noktada ve gelecekle ilgili yapılan çalışmalarda önemli bir rol oynamaktadır. Son derece hızlı olmaları ,yüksek işlem kapasiteleri, düşük hata oranları bu konuda pay sahibidir. İnternet kullanımının dünya genelinde yaygınlaşması da bilgisayarların insan yaşantısında bir yer edinmesine ivme kazandırmıştır. Bilgisayar hayatın her alanında olduğu gibi bilim ve mühendislik alanlarında da etkisini göstermektedir.

Tez çalışmasının birinci bölümünde; tezin amacı, literatür araştırması ve hipotez kısımlarından oluşan giriş bölümü paylaşılmıştır. İkinci bölümde bilimsel temel kavramlar tanımları, örnekleri, aşamaları, statüleri ayrıntılı bir şekilde ele alınmıştır. Karşılaştırmalardan yararlanılmış, sadece matematik terminolojisinde kullanılan kavramlar da ayrıca ifade edilmiştir. Üçüncü bölümde ise teorem kanıtlama programları işlevleri, detaylı özellikleri ve versiyonlarıyla kronolojik sırada incelenmiştir.

Tez çalışmasının dördüncü bölümünde; Collatz Konjektürü'nün çıkış noktası, problemin tanımı, fonksiyonu ve matematiksel notasyonda gösterimine istinaden açıklamaları bulunmaktadır. Ayrıca Collatz Konjektürü senaryo ve realizasyonları, bilinen iterasyonları, görselleri ve günümüzde geldiği son noktadaki güncel bilgilere de yer verilmiştir.

Tez çalışmasının beşinci bölümünde; Collatz Konjektürü işlemlerinin yapılabilmesi için geliştirilen bilgisayar programları yer almaktadır. Üç farklı yöntem üzerinde oldukça detaylı bir şekilde çalışılmış ve bir yöntem önerisinde bulunulmuştur. Her yöntem özelinde metodolojiler oluşturulmuş, önermeler, prosedürler ve notasyon çalışmaları yapılmıştır. Bilgisayar programları mimarisi ile açıklanmış, algoritma ve akış diyagramları geliştirilmiştir. Bilgisayar programlarının yanı sıra, program kodları, program çıktıları, çıktılardan oluşturulan tablolar paylaşılmış ve grafikler ile zenginleştirilmiştir.

Tez çalışmasının altıncı bölümünde; Collatz Konjektürü'ne makine öğrenmesi ile yaklaşım geliştirilmiş ve çalışmalar yapılmıştır. Makine öğrenmesi özelinde bilgisayar programı geliştirilmiştir. Makine öğrenmesi metodolojisinde geliştirilen bilgisayar programı ile bilinmeyen değerlerin tahminlemesi yapılmıştır.

Yaklaşım makine öğrenmesi metotlarından denetimli öğrenme kapsamında yer almıştır. Model, denetimli makine öğrenmesi algoritmalarından doğrusal regresyon özelinde kurgulanarak uygulanmıştır. Gerek metot seçimi gerekse algoritma seçimi esnasında ise, ne tür veriye ve nasıl bir göreve sahip olduğumuza bağlı olması nedeniyle bu kriterler detaylı olarak incelenmiştir. Bu nedenle seçenekler fonksiyonun ve uygulamanın genel yapısı ile veri seti dikkate alınarak değerlendirilmiş ve en uygun olan seçim yapılmıştır.

Collatz Konjektürü'ne makine öğremesi ile geliştirilen yaklaşımın uygulanması sonucu çeşitli bulgular elde edilmiştir. Makine öğrenmesinin geleneksel programlama teknikleri ile karşılaştırıldığında muhtelif kazanımları yer almaktadır. Çalışma gerçekleştirilen veri kümesinde yer almakla birlikte toplamda işlem adım sayısı bilinmeyen sayıların, yahut henüz hesaplaması yapılmamış hatta hesaplaması devam eden sayıların tahminlemesi yapılırken avantaj sağlamaktadır. Collatz Konjektürü fonksiyonunda hesaplanan sayıların, toplamda kaç işlem adımında sonuca ulaştığını belirten adım sayıları oldukça farklılık göstermektedir. Collatz Konjektürü'ne "Dolu Taneleri Problemi" ismi verilmesine de neden olan sekansın lineer bir yapıda olmaması tahminlerin isabeti açısından dezavantaj sağlamaktadır. Makine öğrenmesi yaklaşımının Collatz Konjektürü kapsamında söz konusu özelliği belirleyici rol oynamaktadır.

Bilgisayar programı özelinde, öncelikle oluşturulan makine öğrenmesi modeli hazırlanan eğitim verisi ile eğitilmiştir. Doğrusal regresyon algoritması ile tahmin yapabilmesi için kullanılan matematiksel formülasyon parametrelerinin değerleri hesaplanmış ve tahmin sonuçlarının kanıtı sağlanmıştır. Program, tekil değerlerin tahminlemesini yapabildiği gibi, yüklenen veri seti için de sonuç kümeleri üreterek belirtilen dosya formatında çıktı verebilecek şekilde tasarlanmıştır. Eğitim verisi için kullanılan $n$ sayıları $[1 \times 10^0 , 1 \times 10^2]$ aralığında yer almıştır. Tahminleme çalışması kapsamında bilgisayar programına eğitim verisi aralığı dışında yer alan

$[1.01 \times 10^2 , 1 \times 10^{11}]$ aralığındaki çeşitli değerlerden oluşan veri setinin tahminlemesi yaptırılmıştır. Program tarafından üretilen sonuç kümesi üzerinde indeks değerleri, tahmin edilmesi istenen $n$ sayıları ve tahminlemesi yapılan adım sayıları yer almıştır. Veriler üzerinde gerçek ve tahmin edilen değerler oluşturulmuş, tahmin edilen değerlerin gerçek değerlere oranı hesaplanmıştır. Yüzdelik bazda hata oranları elde edilmiştir. Veri setinde yer alan değerlerin hata oranları toplanmış ve aritmetik ortalaması alınmıştır. Tahmin edilen değerlerin ortalama isabeti açısından $\% \, 2.78 \times 10^9$ sapma sonucuna varılmıştır.

Collatz Konjektürü'nün hesaplanması için "Parite Sekansı Yöntemi" önerilmektedir. Önerilen yöntem standart yöntem ile karşılaştırılmıştır. 1 ve 100 arasındaki sayılar için ölçüm yapılmıştır. Standart yöntemde toplamda 3142 adımda 1 rakamına ulaşılabilirken , parite sekansı yönteminde ise toplamda 2137 adımda 1 rakamına ulaşılabilmektedir. Aradaki toplam adım sayısı farkı 1005 olarak hesaplanmıştır. Yine standart yöntemde tüm işlemlerin tamamlanması 100 milisaniye yani 0.1 saniye sürerken, parite sekansı yönteminde ise tüm işlemlerin yapılması 84 milisaniye yani 0.084 saniye sürmektedir. Aradaki toplam tamamlanma süresi farkı 16 milisaniye yani 0.016 saniye olarak hesaplanmıştır. Parite sekansı yönteminde standart yöntemin yaklaşık %68.01 'i kadar işlem adımında 1 rakamına ulaşılabilmektedir. Yani yaklaşık %31.99 oranında daha az toplam adım sayısı ile 1 rakamına ulaşılabilmektedir. Yine parite sekansı yönteminde standart yönteme göre tüm işlemlerin tamamlanması %84 oranı süresinde, yani %16 oranında daha kısa sürede ölçülmüştür.

## KAYNAKLAR


**[1] T. Uyar, A. Yayımlı, E. Harmancı** (2013). Ayrık matematik, Önermeler. https://ninova.itu.edu.tr/en/courses/faculty-of-computer-and-informatics/142/blg-112/ekkaynaklar?g331087 Son Erişim Zamanı : 04 Mart 2020.

**[2] D. Pierce** (2014). Önermeler Mantığı. Son Erişim Zamanı : 04 Mart 2020. http://mat.msgsu.edu.tr/~dpierce/Dersler/113/2014/mantik-4.pdf

**[3] D. Pierce** (2012). Önermeler Mantığı. Son Erişim Zamanı : 04 Mart 2020. http://mat.msgsu.edu.tr/~dpierce/Dersler/Modeller-kurami/mantik-2.pdf

**[4] C. D. Doğan** (2017). Hipotez ve Kurulması. Son Erişim Zamanı : 04 Mart 2020. https://acikders.ankara.edu.tr/pluginfile.php/13885/mod_resource/content/0/5.%20Hipotez%20ve%20Kurulmas%C4%B1.pdf

**[5] M. Sayan** (2017). Proje Araştırma Konusu Seçimi ve Hipotez Oluşturma. https://neu.edu.tr/wp-content/uploads/2017/02/Murat-Sayan-Proje-Ara%C5%9Ft%C4%B1rma-Konusu-Se%C3%A7imi-ve-Hipotez-Olu%C5%9Fturma.pdf
Son Erişim Zamanı : 04 Mart 2020.

**[6] Ü. Horozcu, A. Arıkan** (2015). Akademik Danışmanlık ve Araştırma Teknikleri. http://xn--ismailbadatl-kyb0x.com.tr/attachments/File/ARAS__TIRMA_TEKNI__KLERI___VE_AKADEMI__K_DANIS__MANLIK_DERS_NOTU.pdf
Son Erişim Zamanı : 04 Mart 2020.

**[7] Examples of Hypothesis** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://examples.yourdictionary.com/examples-of-hypothesis.html

**[8] Hypothesis** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://www.studyandexam.com/hypothesis.html

**[9] Theory** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://dictionary.cambridge.org/dictionary/english/theory

**[10] S. Bewick, R. Parsons, T. Forsythe, S. Robinson, J. Dupon** (2019). Hypothesis, Law, and Theory. Son Erişim Zamanı : 04 Mart 2020. https://chem.libretexts.org/Under_Construction/Purgatory/Textmaps_and_Wikitexts/Introductory_Chemistry/CK-12_Chemistry_(Version_I)/Chapter_1%3A_Introduction_to_Chemistry_and_the_Nature_of_Science/1.2%3A_Hypothesis%2C_Law%2C_and_Theory

**[11] Scientific Hypothesis, Theories and Laws** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://sci.waikato.ac.nz/evolution/Theories.shtml

**[12] Using Some Common Terms Carefully and Accurately In Scientific Speech and Writing** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://oregonstate.edu/instruction/bb317/scientifictheories.html

**[13] A. Bradford** (2017). What Is a Scientific Theory?. https://www.livescience.com/21491-what-is-a-scientific-theory-definition-of-theory.html Son Erişim Zamanı : 04 Mart 2020.

**[14] P. Narguizian** (t.y.). Laws, Theories and Hypotheses: Revealing Science through Words. Son Erişim Zamanı : 04 Mart 2020. https://nhm.org/site/sites/default/files/for_teachers/NHMLA_Laws_Theories_and_Hypotheses_Revealing_Science_through_Words.pdf

**[15] A. Bradford** (2017). What Is a Law in Science?. https://www.livescience.com/21457-what-is-a-law-in-science-definition-of-scientific-law.html Son Erişim Zamanı : 04 Mart 2020.

**[16] A. Dunedin** (t.y.). Hypothesis, Theory, Model and Law. https://www.raggeduniversity.co.uk/2014/03/16/hypothesis-theory-model-law-alex-dunedin/ Son Erişim Zamanı : 04 Mart 2020.

**[17] E. Wyman** (t.y.). Hypothesis, Theory & Law in Science. Son Erişim Zamanı : 04 Mart 2020. https://study.com/academy/lesson/hypothesis-theory-law-in-science.html

**[18] H. Andersen, B. Hepburn** (2016). Scientific Method. Son Erişim Zamanı : 04 Mart 2020. https://plato.stanford.edu/entries/scientific-method/

**[19] Introduction to the Scientific Method** (t.y.). Son Erişim Zamanı : 04 Mart 2020. http://teacher.pas.rochester.edu/phy_labs/AppendixE/AppendixE.html

**[20] M. Ryan, A. O'Callaghan** (t.y.). The Scientific Method. Son Erişim Zamanı : 04 Mart 2020. https://www.unce.unr.edu/publications/files/cd/2002/fs0266.pdf

**[21] Introduction to Methodology for Scientific Research** (2015). Son Erişim Zamanı : 04 Mart 2020. http://dr-monsrs.net/2015/09/29/introduction-to-methodology-for-scientific-research/

**[22] Organizing Your Social Sciences Research Paper: 6. The Methodology** (2019). Son Erişim Zamanı : 04 Mart 2020. https://libguides.usc.edu/writingguide/methodology

**[23] The Core of Science: Relating Evidence and Ideas** (t.y.). Son Erişim Zamanı : 04 Mart 2020. https://undsci.berkeley.edu/article/coreofscience_01

**[24] Nature of Science Terms** (t.y.). Son Erişim Zamanı : 04 Mart 2020.
https://www.nhm.ac.uk/content/dam/nhmwww/about-us/visitor-research/nature-of-science-terms.pdf

**[25] Science Relies on Evidence** (t.y.). Son Erişim Zamanı : 04 Mart 2020.
https://undsci.berkeley.edu/article/whatisscience_06
**[26] Types of Scientific Evidence** (t.y.). Son Erişim Zamanı : 04 Mart 2020.
https://www.sciencemediacentre.co.nz/coveringscience/types-of-scientific-evidence/

**[27] Conjectures** (t.y.). Son Erişim Zamanı : 04 Mart 2020.
https://brilliant.org/wiki/conjectures/

**[28] T. Melham** (2005). The HOL Theorem Prover for Higher Order Logic
Son Erişim Zamanı : 04 Mart 2020.
https://www.cs.ox.ac.uk/tom.melham/res/hol.html

**[29] What is HOL?** (t.y.).
Son Erişim Zamanı : 04 Mart 2020.
https://hol-theorem-prover.org/

**[30] T. F. Melham** (t.y.). Automating Recursive Type Definitions in Higher Order Logic. Son Erişim Zamanı : 04 Mart 2020.
http://www.cs.ox.ac.uk/tom.melham/pub/Melham-1989-ART.pdf

**[31] M. Gordon** (1996). From LCF to HOL: A Short History.
Son Erişim Zamanı : 04 Mart 2020.
https://www.cl.cam.ac.uk/archive/mjcg/papers/HolHistory.pdf

**[32] R. B. Guenther** (1992). Lothar Collatz, 1910–1990.
Son Erişim Zamanı : 04 Mart 2020.
https://link.springer.com/article/10.1007/BF01840490

**[33] J. J. O'Connor, E. F. Robertson** (2006). Lothar Collatz.
Son Erişim Zamanı : 04 Mart 2020.
https://mathshistory.st-andrews.ac.uk/Biographies/Collatz/

**[34] M. Hammett** (t.y.). The Collatz Conjecture: A Brief Overview.
Son Erişim Zamanı : 04 Mart 2020.
http://online.sfsu.edu/meredith/301/Papers/FinalDraft_Hammett.pdf

**[35] J. C. Lagarias** (2011). The 3x+1 Problem: An Annotated Bibliography (1963-1999). Son Erişim Zamanı : 04 Mart 2020.
https://arxiv.org/pdf/math/0309224.pdf

**[36] R. Deloin** (2019). Proof of Collatz Conjecture.
Son Erişim Zamanı : 04 Mart 2020.
https://www.journalarjom.com/index.php/ARJOM/article/view/30123/56520

**[37] A. J. Phillips** (2010). Parity Periodicity: An Eliminative Approach to the Collatz Conjecture. Son Erişim Zamanı : 04 Mart 2020.
https://scholarlycommons.obu.edu/honors_theses/53/

**[38] J. Davies** (2012). Collatz Graph: All Numbers Lead to One.
Son Erişim Zamanı : 04 Mart 2020.
https://www.jasondavies.com/collatz-graph/

**[39] D. Barina** (2020). Convergence Verification of the Collatz Problem.
Son Erişim Zamanı : 04 Mart 2020.
https://link.springer.com/article/10.1007/s11227-020-03368-x

**[40] P. Honner** (2020). The Simple Math Problem We Still Can't Solve.
Son Erişim Zamanı : 04 Mart 2020.
https://www.quantamagazine.org/why-mathematicians-still-cant-solve-the-collatz-conjecture-20200922/

**[41] O. D. O. Santos** (2018). Proving the Collatz Conjecture with Binaries Numbers.
Son Erişim Zamanı : 04 Mart 2020.
article.sciencepublishinggroup.com

**[42] Cliff Pickover's Patterns in the Mysterious Hailstone (3n+1) Numbers** (t.y.).
Son Erişim Zamanı : 04 Mart 2020.
http://sprott.physics.wisc.edu/pickover/hailstone.html

**[43] 3x+1 Delay Records** (t.y.).
Son Erişim Zamanı : 04 Mart 2020.
http://www.ericr.nl/wondrous/delrecs.html

**[44] B. Haran** (2017). The Collatz Conjecture in Colour.
Son Erişim Zamanı : 04 Mart 2020.
https://www.bradyharanblog.com/blog/the-collatz-conjecture-in-colour

**[45] Compute for Science** (2020).
Son Erişim Zamanı : 04 Mart 2020.
https://boinc.berkeley.edu/

**[46] J. Sonntag** (2020). What is Collatz Conjecture?
Son Erişim Zamanı : 04 Mart 2020.
https://boinc.thesonntags.com/collatz/

**[47] Collatz Best Results** (2020).
Son Erişim Zamanı : 04 Mart 2020.
https://boinc.thesonntags.com/collatz/highest_steps.php

**[48] Today's High Steps** (2020).
Son Erişim Zamanı : 04 Mart 2020.
https://boinc.thesonntags.com/collatz/high_steppers.php

**[49] E. W. Weisstein** (t.y.). Collatz Problem.
Son Erişim Zamanı : 04 Mart 2020.
https://mathworld.wolfram.com/CollatzProblem.html

**[50] B. Bairrington, A. Okano** (2019). New Experimental Investigations for the 3x+1 Problem: The Binary Projection of the Collatz Map.
Son Erişim Zamanı : 04 Mart 2020.
https://scholar.rose-hulman.edu/cgi/viewcontent.cgi?article=1393&context=rhumj

**[51] H. A. Simon** (t.y.). What is Machine Learning?
Son Erişim Zamanı : 17 Nisan 2021.
https://www.ml.cmu.edu/

**[52] An Introduction to Machine Learning** (2020).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.geeksforgeeks.org/introduction-machine-learning/

**[53] AI vs Machine Learning vs Deep Learning** (2020).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.edureka.co/blog/ai-vs-machine-learning-vs-deep-learning/

**[54] T. Mitchell** (1997). Machine Learning.
Son Erişim Zamanı : 17 Nisan 2021.
http://www.cs.cmu.edu/~tom/mlbook.html

**[55] Machine Learning** (2020).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.ibm.com/cloud/learn/machine-learning

**[56] E. Alpaydın** (2020). Introduction to Machine Learning.
Son Erişim Zamanı : 17 Nisan 2021.
https://mitpress.mit.edu/books/introduction-machine-learning-fourth-edition

**[57] Types of Learning - Supervised Learning** (2020).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/

**[58] What is the Difference Between Regression and Classification?** (t.y.).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.educative.io/edpresso/what-is-the-difference-between-regression-and-classification

**[59] Supervised and Unsupervised Learning** (2021).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/

**[60] J. Brownlee** (2020). Supervised and Unsupervised Machine Learning Algorithms. Son Erişim Zamanı : 17 Nisan 2021.
https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

**[61] What is Unsupervised Learning?** (t.y.).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.educative.io/edpresso/what-is-unsupervised-learning

**[62] M. T. Jones** (2017). Models for Machine Learning.
Son Erişim Zamanı : 17 Nisan 2021.
https://developer.ibm.com/articles/cc-models-machine-learning/

**[63] M. V. Otterlo, M. Wiering** (2012). Reinforcement Learning and Markov Decision Processes. Son Erişim Zamanı : 17 Nisan 2021.
https://link.springer.com/chapter/10.1007%2F978-3-642-27645-3_1

**[64] Reinforcement Learning** (2020).
Son Erişim Zamanı : 17 Nisan 2021.
https://www.geeksforgeeks.org/what-is-reinforcement-learning/

# ÖZGEÇMİŞ

Computer Engineer

## MERT ÖZKENAR

### Languages

- Turkish – Native
- English – Upper-Intermediate
- Italian – Elementary

### Experience

LC WAIKIKI

**Software Developer** **Nov 2014 – Present**

- Software Common Services Team (Software Architecture Management) Jan 2019 – Present
- Software ERP Team Jul 2015 – Jan 2019
- Software Infrastructure Team Nov 2014 – Jul 2015

### Education

**Istanbul Aydin University** **(2017 - 2021)**

Master of Science - MS, Computer Engineering (with Thesis)

**Istanbul Aydin University** **(2012 - 2015)**

Bachelor's Degree, Computer Engineering (English)

**Anadolu University** **(2011 - 2013)**

Bachelor's Degree, Business Administration and Management

**Istanbul Gelisim University** **(2009 - 2011)**

Associate's Degree, Computer Programming

### Publications Extracted From The Thesis

**Istanbul University,** Acta Infologica

Özkenar, M. (2020). The Parity Sequence Method Approach in Computing Collatz Conjecture via a Computer Program. Acta Infologica, 4(2), 1-00.

https://doi.org/10.26650/acin.843275