Continuous Computation and Applications

Jaikrishnan Janardhanan
Indian Institute of Technology Madras
jaikrishnan@iitm.ac.in

Talk at IIIT Hyderabad, March 4, 2016

A Mathematician's viewpoint on Computer Science

- The aim of theoretical computer science is to understand the difficulty of solving a given computational problem.

A Mathematician's viewpoint on Computer Science

- The aim of theoretical computer science is to understand the difficulty of solving a given computational problem.
- Some examples:
  - Given two natural numbers n, m, compute $n + m, n \times m$ and $gcd(m, n)$.

A Mathematician's viewpoint on Computer Science

- The aim of theoretical computer science is to understand the difficulty of solving a given computational problem.
- Some examples:
  - Given two natural numbers n, m, compute $n + m, n \times m$ and $gcd(m, n)$.
  - Trisect a given angle using only a straight-edge and compass.

A Mathematician's viewpoint on Computer Science

- The aim of theoretical computer science is to understand the difficulty of solving a given computational problem.
- Some examples:
  - Given two natural numbers n, m, compute $n + m, n \times m$ and $gcd(m, n)$.
  - Trisect a given angle using only a straight-edge and compass.
  - Given a natural number p, determine if p is prime or not. More generally, given a natural number n, determine its prime factorization.

A Mathematician's viewpoint on Computer Science

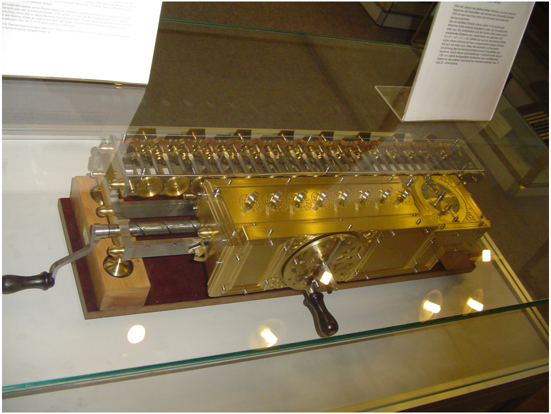
- The aim of theoretical computer science is to understand the difficulty of solving a given computational problem.
- Some examples:
  - Given two natural numbers n, m, compute $n + m, n \times m$ and $gcd(m, n)$.
  - Trisect a given angle using only a straight-edge and compass.
  - Given a natural number p, determine if p is prime or not. More generally, given a natural number n, determine its prime factorization.
- TCS has as much to do with computers as astronomy has to do with telescopes.

Origin: Leibniz's dream

In the seventeenth century, Gottfried Leibniz constructed a mechanical calculating machine known as the stepped reckoner.

Origin: Leibniz's dream

In the seventeenth century, Gottfried Leibniz constructed a mechanical calculating machine known as the stepped reckoner.



Origin: Leibniz's dream

- Leibiniz dreamt of building a machine which would decide the truth of a statement of mathematics.

Origin: Leibniz's dream

- Leibiniz dreamt of building a machine which would decide the truth of a statement of mathematics.
- He also understood that it is essential that one develops a formal language in which statements of mathematics could be expressed in, and spent a considerable amount of time in designing such a language.

Origin: Leibniz's dream

- Leibiniz dreamt of building a machine which would decide the truth of a statement of mathematics.
- He also understood that it is essential that one develops a formal language in which statements of mathematics could be expressed in, and spent a considerable amount of time in designing such a language.
- However, he never completed this project. His ideas were far ahead of his times and his contemporaries were not interested in them. Much later, in the 19th century, George Boole developed first-order logic.

Origin: Leibniz's dream

- Leibiniz dreamt of building a machine which would decide the truth of a statement of mathematics.
- He also understood that it is essential that one develops a formal language in which statements of mathematics could be expressed in, and spent a considerable amount of time in designing such a language.
- However, he never completed this project. His ideas were far ahead of his times and his contemporaries were not interested in them. Much later, in the 19th century, George Boole developed first-order logic.
- A famous quote: "Gentlemen, let us calculate!".

Hilbert's tenth problem and Entscheidungsproblem

- Hilbert's tenth problem asks for a "mechanical procedure" to determine whether a given polynomial equation with integer coefficients has an integer root.

Hilbert's tenth problem and Entscheidungsproblem

- Hilbert's tenth problem asks for a "mechanical procedure" to determine whether a given polynomial equation with integer coefficients has an integer root.

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0,\ a_i \in \mathbb{Z}.$$

Does there exists $z \in \mathbb{Z}$ such that $p(z) = 0$?

Hilbert's tenth problem and Entscheidungsproblem

- Hilbert's tenth problem asks for a "mechanical procedure" to determine whether a given polynomial equation with integer coefficients has an integer root.

  $$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0, \ a_i \in \mathbb{Z}.$$

  Does there exists $z \in \mathbb{Z}$ such that $p(z) = 0$?
- More generally, Hilbert's Entscheidungsproblem asks for a "mechanical procedure" to decide whether a given statement of first order logic is true or false.

Hilbert's tenth problem and Entscheidungsproblem

- Hilbert's tenth problem asks for a "mechanical procedure" to determine whether a given polynomial equation with integer coefficients has an integer root.

  $$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0, \; a_i \in \mathbb{Z}.$$

  Does there exists $z \in \mathbb{Z}$ such that $p(z) = 0$?
- More generally, Hilbert's Entscheidungsproblem asks for a "mechanical procedure" to decide whether a given statement of first order logic is true or false.
- To answer such questions one needs to have a rigorous definition of a "mechanical procedure" (nowadays know as an *algorithm*).

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \ldots, q_n$ and a set of instructions (a program).

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \ldots, q_n$ and a set of instructions (a program).
- The head scans one cell at a time and executes a step of computation that depends on the content of the current cell and its current state.

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \ldots, q_n$ and a set of instructions (a program).
- The head scans one cell at a time and executes a step of computation that depends on the content of the current cell and its current state.
- A step of computation consists of
  - replacing the contents of the scanned cell with another symbol,

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \ldots, q_n$ and a set of instructions (a program).
- The head scans one cell at a time and executes a step of computation that depends on the content of the current cell and its current state.
- A step of computation consists of
  - replacing the contents of the scanned cell with another symbol,
  - move one step to the left or right, or stay still,

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \ldots, q_n$ and a set of instructions (a program).
- The head scans one cell at a time and executes a step of computation that depends on the content of the current cell and its current state.
- A step of computation consists of
  - replacing the contents of the scanned cell with another symbol,
  - move one step to the left or right, or stay still,
  - change to a new state;

The Turing machine

- A TM consists of three parts, an infinite type divided into cells and each cell consists of symbols $0, 1, B$ and a head (or control) that is in one of finitely many states $q_1, \dots, q_n$ and a set of instructions (a program).
- The head scans one cell at a time and executes a step of computation that depends on the content of the current cell and its current state.
- A step of computation consists of
  - replacing the contents of the scanned cell with another symbol,
  - move one step to the left or right, or stay still,
  - change to a new state;
- A TM program is sequence of instructions of the form

$$(q_i, S_j) \mapsto (q_l, S_k)D.$$

A counting machine

The video demonstrates a simple machine that counts. Given any input, the machine keeps incrementing the input by 1 indefinitely. The states of the machine are also 0 and 1.

A counting machine

The video demonstrates a simple machine that counts. Given any input, the machine keeps incrementing the input by 1 indefinitely. The states of the machine are also 0 and 1.

A counting program

$(0,1) \rightarrow (0,1)$ `Right`
$(0,0) \rightarrow (0,0)$ `Right`
$(0,B) \rightarrow (1,B)$ `Left`
$(1,0) \rightarrow (0,1)$ `Right`
$(1,1) \rightarrow (1,0)$ `Left`
$(1,B) \rightarrow (0,1)$ `Right`

Computable functions and the Halting problem

- A function $f : \mathbb{N} \to \mathbb{N}$ is said to be computable if there is a Turing machine that computes f.

Computable functions and the Halting problem

- A function $f : \mathbb{N} \to \mathbb{N}$ is said to be computable if there is a Turing machine that computes f.
- Thus, the Entscheidungsproblem can be rephrased as "Is there a Turing machine that takes as its input an encoding of a first-order logic formula and returns 1 if its true and 0 if it is false?"

Computable functions and the Halting problem

- A function $f : \mathbb{N} \to \mathbb{N}$ is said to be computable if there is a Turing machine that computes f.
- Thus, the Entscheidungsproblem can be rephrased as "Is there a Turing machine that takes as its input an encoding of a first-order logic formula and returns 1 if its true and 0 if it is false?"
- As there are uncountably many functions from $\mathbb{N}$ to $\mathbb{N}$ but only countably many Turing machines it follows that there are uncountably many uncomputable functions.

Computable functions and the Halting problem

- A function $f : \mathbb{N} \to \mathbb{N}$ is said to be computable if there is a Turing machine that computes f.
- Thus, the Entscheidungsproblem can be rephrased as "Is there a Turing machine that takes as its input an encoding of a first-order logic formula and returns 1 if its true and 0 if it is false?"
- As there are uncountably many functions from $\mathbb{N}$ to $\mathbb{N}$ but only countably many Turing machines it follows that there are uncountably many uncomputable functions.
- Turing observed that TMs themselves might be encoded as natural numbers. He showed that the Halting problem is unsolvable and also showed that if the Entscheidungsproblem is solvable, then so is the Halting problem.

The Church–Turing thesis

- The Chruch–Turing thesis states that any function that is “effectively calculable” can be computed by a Turing machine. Note that this is not a theorem and can never be formally proved.

The Church–Turing thesis

- The Chruch–Turing thesis states that any function that is "effectively calculable" can be computed by a Turing machine. Note that this is not a theorem and can never be formally proved.
- The thesis was proposed because entirely different ways to define what it means for a function to "effectively calculable" turned out to be equivalent. Church's lambda calculus, the recursive functions of Gödel–Herbrand, register machines are all equivalent to the TM. All known models of computation that are "physically realizable" (including quantum computers) can be emulated on a Turing machine

The Church–Turing thesis

- The Chruch–Turing thesis states that any function that is "effectively calculable" can be computed by a Turing machine. Note that this is not a theorem and can never be formally proved.
- The thesis was proposed because entirely different ways to define what it means for a function to "effectively calculable" turned out to be equivalent. Church's lambda calculus, the recursive functions of Gödel–Herbrand, register machines are all equivalent to the TM. All known models of computation that are "physically realizable" (including quantum computers) can be emulated on a Turing machine
- This thesis has been the subject of much debate and philosophical speculation.

The Church–Turing thesis

- The Chruch–Turing thesis states that any function that is "effectively calculable" can be computed by a Turing machine. Note that this is not a theorem and can never be formally proved.
- The thesis was proposed because entirely different ways to define what it means for a function to "effectively calculable" turned out to be equivalent. Church's lambda calculus, the recursive functions of Gödel–Herbrand, register machines are all equivalent to the TM. All known models of computation that are "physically realizable" (including quantum computers) can be emulated on a Turing machine
- This thesis has been the subject of much debate and philosophical speculation.
- The video demonstrates an entirely different type of computing device—Conway's game of life.

Real computing

- What does it mean for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable?

Real computing

- What does it mean for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable?
- Turing in his famous 1936 paper defined a real number to be computable if its decimal expansion can be written out by a TM.

Real computing

- What does it mean for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable?
- Turing in his famous 1936 paper defined a real number to be computable if its decimal expansion can be written out by a TM.
- There are two main approaches to define what it means for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable.

Real computing

- What does it mean for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable?
- Turing in his famous 1936 paper defined a real number to be computable if its decimal expansion can be written out by a TM.
- There are two main approaches to define what it means for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable.
- The first one known as Recursive Analysis (the bit model) was developed by Andrzej Grzegorczyk, and independently by Daniel Lacombe, and the second one by Blum, Shub and Smale known as the BSS model.

Real computing

- What does it mean for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable?
- Turing in his famous 1936 paper defined a real number to be computable if its decimal expansion can be written out by a TM.
- There are two main approaches to define what it means for a function $f : \mathbb{R} \to \mathbb{R}$ to be computable.
- The first one known as Recursive Analysis (the bit model) was developed by Andrzej Grzegorczyk, and independently by Daniel Lacombe, and the second one by Blum, Shub and Smale known as the BSS model.
- Roughly speaking, a real function f is computable in the bit model if there is an TM which, given a good rational approximation to x, finds a good rational approximation to $f(x)$. One can neatly formalize this notion in terms of a TM with access to an oracle.

Definition of a computable real function

- Let $\mathbb{D}$ denote the set of dyadic rational numbers, i.e., those that of the form $\frac{p}{2^m}$.

Definition of a computable real function

- Let $\mathbb{D}$ denote the set of dyadic rational numbers, i.e., those that of the form $\frac{p}{2^m}$.
- A function $\phi : \mathbb{N} \to \mathbb{D}$ is said to be an *oracle* for the number $x \in \mathbb{R}$ if $|x - \phi(n)| < 2^{-n}$. In other words, ϕ provides a good dyadic approximation of x.

Definition of a computable real function

- Let $\mathbb{D}$ denote the set of dyadic rational numbers, i.e., those that of the form $\frac{p}{2^m}$.
- A function $\phi : \mathbb{N} \to \mathbb{D}$ is said to be an *oracle* for the number $x \in \mathbb{R}$ if $|x - \phi(n)| < 2^{-n}$. In other words, ϕ provides a good dyadic approximation of x.
- A Turing machine with access to an oracle ϕ is a Turing machine that is allowed at any step of its computation to query the value of $\phi(n)$ for any n.

Definition of a computable real function

- Let $\mathbb{D}$ denote the set of dyadic rational numbers, i.e., those that of the form $\frac{p}{2^m}$.
- A function $\phi : \mathbb{N} \to \mathbb{D}$ is said to be an *oracle* for the number $x \in \mathbb{R}$ if $|x - \phi(n)| < 2^{-n}$. In other words, ϕ provides a good dyadic approximation of x.
- A Turing machine with access to an oracle ϕ is a Turing machine that is allowed at any step of its computation to query the value of $\phi(n)$ for any n.

Definition

A function $f : [a, b] \to [c, d]$ is said to be computable if there is an oracle Turing machine M^ϕ such that if ϕ is an oracle for $x \in [a, b]$ then on input m, $M^\phi(m)$ is a dyadic rational with the property that $|M^\phi(m) - f(x)| < 2^{-m}$.

Definition of a computable real function

- Let $\mathbb{D}$ denote the set of dyadic rational numbers, i.e., those that of the form $\frac{p}{2^m}$.
- A function $\phi : \mathbb{N} \to \mathbb{D}$ is said to be an *oracle* for the number $x \in \mathbb{R}$ if $|x - \phi(n)| < 2^{-n}$. In other words, ϕ provides a good dyadic approximation of x.
- A Turing machine with access to an oracle ϕ is a Turing machine that is allowed at any step of its computation to query the value of $\phi(n)$ for any n.

Definition

A function $f : [a, b] \to [c, d]$ is said to be computable if there is an oracle Turing machine M^{ϕ} such that if ϕ is an oracle for $x \in [a, b]$ then on input m, $M^{\phi}(m)$ is a dyadic rational with the property that $|M^{\phi}(m) - f(x)| < 2^{-m}$.

The definition says that a function is computable if there is TM when provided with a good approximation of x outputs a good approximation of $f(x)$. It is easy to prove that computable functions are continuous.

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.
- In recent years, the study of a class of figures called **fractals** have become very popular.

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.
- In recent years, the study of a class of figures called **fractals** have become very popular.
- The pioneer of the subject was Mandelbrot who described fractals as ''beautiful, damn hard, increasingly useful. That's fractals."

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.
- In recent years, the study of a class of figures called **fractals** have become very popular.
- The pioneer of the subject was Mandelbrot who described fractals as ''beautiful, damn hard, increasingly useful. That's fractals."
- There is some disagreement amongst experts as to what the precise formal definition is but in short fractals are objects that are infinitely self-similar, iterated, and detailed having fractal dimensions.

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.
- In recent years, the study of a class of figures called **fractals** have become very popular.
- The pioneer of the subject was Mandelbrot who described fractals as ‘’beautiful, damn hard, increasingly useful. That’s fractals.”
- There is some disagreement amongst experts as to what the precise formal definition is but in short fractals are objects that are infinitely self-similar, iterated, and detailed having fractal dimensions.
- There are many computer images of theses infinitely self-similar objects.

When do we say that a picture is computable?

- Computers have been used as an aid to visualizing mathematical objects like graphs for decades.
- In recent years, the study of a class of figures called **fractals** have become very popular.
- The pioneer of the subject was Mandelbrot who described fractals as ‘’beautiful, damn hard, increasingly useful. That’s fractals.”
- There is some disagreement amongst experts as to what the precise formal definition is but in short fractals are objects that are infinitely self-similar, iterated, and detailed having fractal dimensions.
- There are many computer images of theses infinitely self-similar objects.
- In what sense are these pictures accurate representations of these complicated objects?

The Koch snowflake

One famous example of a fractal is the Koch snowflake. The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

The Koch snowflake

One famous example of a fractal is the Koch snowflake. The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

1. Divide the line segment into three segments of equal length.

The Koch snowflake

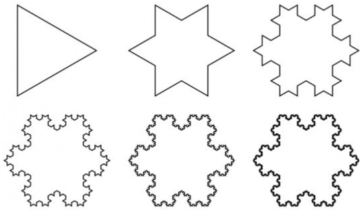
One famous example of a fractal is the Koch snowflake. The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

1. Divide the line segment into three segments of equal length.
2. Draw an equilateral triangle that has the middle segment from step 1 as its base and points outward.

The Koch snowflake

One famous example of a fractal is the Koch snowflake. The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

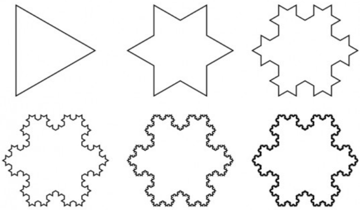
1. Divide the line segment into three segments of equal length.
2. Draw an equilateral triangle that has the middle segment from step 1 as its base and points outward.
3. Remove the line segment that is the base of the triangle from step 2.



The Koch snowflake

One famous example of a fractal is the Koch snowflake. The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

1. Divide the line segment into three segments of equal length.
2. Draw an equilateral triangle that has the middle segment from step 1 as its base and points outward.
3. Remove the line segment that is the base of the triangle from step 2.



Computability of sets in the plane

- It is clear that we can draw the Koch snowflake on a computer screen to whatever degree of accuracy we desire.

Computability of sets in the plane

- It is clear that we can draw the Koch snowflake on a computer screen to whatever degree of accuracy we desire.
- Informally, a planar figure is computable if it can be "rendered" by a computer to any desired level of accuracy.

Computability of sets in the plane

- It is clear that we can draw the Koch snowflake on a computer screen to whatever degree of accuracy we desire.
- Informally, a planar figure is computable if it can be "rendered" by a computer to any desired level of accuracy.
- We say that a set $S \subset \mathbb{R}^2$ is computable, if for any $k \in \mathbb{N}$, there is a computer program that outputs a finite set of points S_k whose coordinates are dyadic rational such that S_k is a 2^{-k}-approximation of S in the Hausdorff distance.

Computability of sets in the plane

- It is clear that we can draw the Koch snowflake on a computer screen to whatever degree of accuracy we desire.
- Informally, a planar figure is computable if it can be "rendered" by a computer to any desired level of accuracy.
- We say that a set $S \subset \mathbb{R}^2$ is computable, if for any $k \in \mathbb{N}$, there is a computer program that outputs a finite set of points S_k whose coordinates are dyadic rational such that S_k is a 2^{-k}-approximation of S in the Hausdorff distance.
- The Hausdorff distance between two compact sets S_1 and S_2 is defined as follows:

$$d_H(S_1, S_2) := \inf\{\varepsilon > 0 : S_1 \subset B(S_1, \varepsilon), S_2 \subset B(S_1, \varepsilon)\}.$$

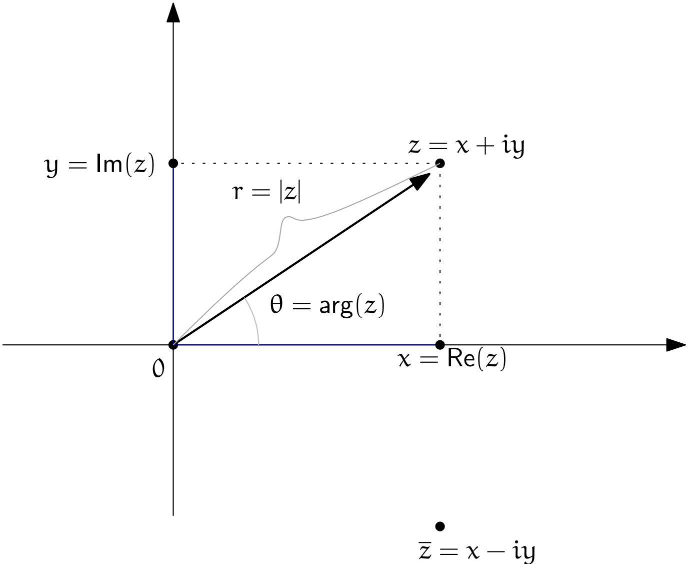
Computability of sets in the plane

- It is clear that we can draw the Koch snowflake on a computer screen to whatever degree of accuracy we desire.
- Informally, a planar figure is computable if it can be "rendered" by a computer to any desired level of accuracy.
- We say that a set $S \subset \mathbb{R}^2$ is computable, if for any $k \in \mathbb{N}$, there is a computer program that outputs a finite set of points S_k whose coordinates are dyadic rational such that S_k is a 2^{-k}-approximation of S in the Hausdorff distance.
- The Hausdorff distance between two compact sets S_1 and S_2 is defined as follows:

$$d_H(S_1, S_2) := \inf\{\varepsilon > 0 : S_1 \subset B(S_1, \varepsilon), S_2 \subset B(S_1, \varepsilon)\}.$$

- Apart from being natural this definition satisfies some nice properties. For instance, the bit computability of a continuous function $f : D \to \mathbb{R}$ ($D \subset \mathbb{R}^2$ is a computable domain) is equivalent to its graph being computable.

Complex numbers



Rational functions

- A complex polynomial is function

$$P(z) = c_0 + c_1 z + c_2 z^2 + \cdots + c_n z^n, \;\; c_i \in \mathbb{C}$$

Rational functions

- A complex polynomial is function

$$P(z) = c_0 + c_1 z + c_2 z^2 + \cdots + c_n z^n, \;\; c_i \in \mathbb{C}$$

- By a famous theorem called the Fundamental theorem of algebra, we can factorize $P(z)$ as $c(z - a_1)^{n_1}(z - a_2)^{n_2} \ldots (z - a_k)^{n_k}$ where a_i's are the roots of P.

Rational functions

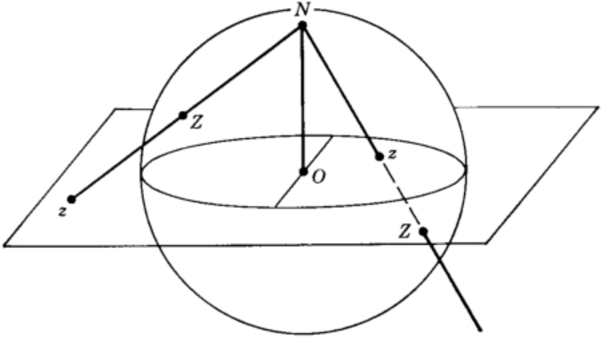
- A complex polynomial is function

$$P(z) = c_0 + c_1 z + c_2 z^2 + \cdots + c_n z^n, \;\; c_i \in \mathbb{C}$$

- By a famous theorem called the Fundamental theorem of algebra, we can factorize $P(z)$ as $c(z-a_1)^{n_1}(z-a_2)^{n_2}\ldots(z-a_k)^{n_k}$ where a_i's are the roots of P.
- A rational function is just the quotient of two complex polynomials: $\frac{P(z)}{Q(z)}$. We will assume that the P and Q do not have any common factors.

The Riemann sphere

We add a point at ∞ to the Complex plane. The result is the Riemann sphere. Points on the sphere are identified to points on the plane via stereographic projection from the north pole. The north pole is the point at ∞.



Notions from complex dynamics

- Complex dynamics is primarily about the study of iterations of rational functions on the Riemann sphere.

Notions from complex dynamics

- Complex dynamics is primarily about the study of iterations of rational functions on the Riemann sphere.
- In the past 30 years, there has been tremendous progress in the subject and this has been partially attributed to the possibility of computer experimentation

Notions from complex dynamics

- Complex dynamics is primarily about the study of iterations of rational functions on the Riemann sphere.
- In the past 30 years, there has been tremendous progress in the subject and this has been partially attributed to the possibility of computer experimentation
- We are primarily interested in the orbit $R(z), R^2(z), R^3(z), \dots$ of points $z \in \widehat{\mathbb{C}}$ under the rational map R.

Notions from complex dynamics

- Complex dynamics is primarily about the study of iterations of rational functions on the Riemann sphere.
- In the past 30 years, there has been tremendous progress in the subject and this has been partially attributed to the possibility of computer experimentation
- We are primarily interested in the orbit $R(z), R^2(z), R^3(z), \ldots$ of points $z \in \widehat{\mathbb{C}}$ under the rational map R.
- Consider $R(z) = z^2$. Then it is clear that orbit of any point in the unit disk converges to 0 and the orbit of any point outside the closed unit disk goes to ∞. The orbits of points on the unit circle are complicated and depend on the particular point.

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.
- Roughly, the points of the Julia set are the points at which we have chaotic behaviour and the Fatou set consists of points of stability.

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.
- Roughly, the points of the Julia set are the points at which we have chaotic behaviour and the Fatou set consists of points of stability.
- The Julia set of the rational function z^2 is the unit circle.

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.
- Roughly, the points of the Julia set are the points at which we have chaotic behaviour and the Fatou set consists of points of stability.
- The Julia set of the rational function z^2 is the unit circle.
- One central question in complex dynamics is to understand the structure of the Julia sets of the rational functions $R_c(z) := z^2 + c$.

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.
- Roughly, the points of the Julia set are the points at which we have chaotic behaviour and the Fatou set consists of points of stability.
- The Julia set of the rational function z^2 is the unit circle.
- One central question in complex dynamics is to understand the structure of the Julia sets of the rational functions $R_c(z) := z^2 + c$.
- It turns out that for most $c \in \mathbb{C}$, the associated Julia set J_c is a fractal!

Julia and Fatou set

- Any rational function partitions $\widehat{\mathbb{C}}$ into two complementary pieces the Julia set and the Fatou set.
- Roughly, the points of the Julia set are the points at which we have chaotic behaviour and the Fatou set consists of points of stability.
- The Julia set of the rational function z^2 is the unit circle.
- One central question in complex dynamics is to understand the structure of the Julia sets of the rational functions $R_c(z) := z^2 + c$.
- It turns out that for most $c \in \mathbb{C}$, the associated Julia set J_c is a fractal!
- One can show that the $R_c(z)$ is the boundary of the filled Julia set given by $\{z \in \mathbb{C} : R_c^n(z) \text{ remains bounded}\}$.

The Mandelbort set

- Arguably the most famous object in the study of complex dynamics is the Mandelbrot set.

The Mandelbort set

- Arguably the most famous object in the study of complex dynamics is the Mandelbrot set.
- Some of the basic questions about this set remain open to this day.

The Mandelbort set

- Arguably the most famous object in the study of complex dynamics is the Mandelbrot set.
- Some of the basic questions about this set remain open to this day.
- Formally, the Mandelbrot set is the set of all points $c \in \mathbb{C}$ such that the sequence
$$c, c^2 + c, (c^2 + c)^2 + c, \ldots$$
remains bounded.

The Mandelbort set

- Arguably the most famous object in the study of complex dynamics is the Mandelbrot set.
- Some of the basic questions about this set remain open to this day.
- Formally, the Mandelbrot set is the set of all points $c \in \mathbb{C}$ such that the sequence

$$c, c^2 + c, (c^2 + c)^2 + c, \ldots$$

  remains bounded.

A famous result

If $c \in M$ then J_c is connected.

Some pretty pictures

- There are many programs available that “draw” the Mandelbrot set.

Some pretty pictures

- There are many programs available that “draw” the Mandelbrot set.
- The simplest method to “draw” the set is to start with a point $c \in \mathbb{C}$ and perform a fixed number of iteration of the sequence in the definition of the Mandelbrot set till an escape condition is reached. A simple escape condition is the orbit escaping the disk of radius 2.

Some pretty pictures

- There are many programs available that “draw” the Mandelbrot set.
- The simplest method to “draw” the set is to start with a point $c \in \mathbb{C}$ and perform a fixed number of iteration of the sequence in the definition of the Mandelbrot set till an escape condition is reached. A simple escape condition is the orbit escaping the disk of radius 2.
- The color of each point represents how quickly the values reached the escape point. Often black is used to show values that fail to escape before the iteration limit, and gradually brighter colours are used for points that escape. This gives a visual representation of how many cycles were required before reaching the escape condition.

Some pretty pictures

- There are many programs available that “draw” the Mandelbrot set.
- The simplest method to “draw” the set is to start with a point $c \in \mathbb{C}$ and perform a fixed number of iteration of the sequence in the definition of the Mandelbrot set till an escape condition is reached. A simple escape condition is the orbit escaping the disk of radius 2.
- The color of each point represents how quickly the values reached the escape point. Often black is used to show values that fail to escape before the iteration limit, and gradually brighter colours are used for points that escape. This gives a visual representation of how many cycles were required before reaching the escape condition.
- There are other more sophisticated colouring techniques. The Wikipedia article on the Mandelbrot set gives brief descriptions of some of them.

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

Conjecture

The Mandelbrot set is locally-connected.

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

Conjecture

The Mandelbrot set is locally-connected.

There are many reasons why this conjecture is of central importance. If the conjecture is true we get several significant consequences:

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

Conjecture

The Mandelbrot set is locally-connected.

There are many reasons why this conjecture is of central importance. If the conjecture is true we get several significant consequences:

- Using a classical result of Caratheodory combined with powerful recent results of Thurston, Douady, Hubbard and others, we can get a complete description of the Mandelbrot set.

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

Conjecture

The Mandelbrot set is locally-connected.

There are many reasons why this conjecture is of central importance. If the conjecture is true we get several significant consequences:

- Using a classical result of Caratheodory combined with powerful recent results of Thurston, Douady, Hubbard and others, we can get a complete description of the Mandelbrot set.
- A central conjecture in complex dynamics called the density hyperbolicity conjecture will also be true.

The MLC conjecture

One of the most important open questions in complex dynamics is the following conjecture:

Conjecture

The Mandelbrot set is locally-connected.

There are many reasons why this conjecture is of central importance. If the conjecture is true we get several significant consequences:

- Using a classical result of Caratheodory combined with powerful recent results of Thurston, Douady, Hubbard and others, we can get a complete description of the Mandelbrot set.
- A central conjecture in complex dynamics called the density hyperbolicity conjecture will also be true.
- The Mandelbrot set will be computable.

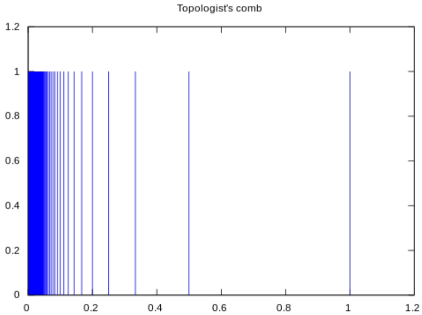
Connectedness and local-connectedness

In this slide we will briefly describe the notions of connectedness and local connectedness through pictures.

Connectedness and local-connectedness

In this slide we will briefly describe the notions of connectedness and local connectedness through pictures.

Connectedness just means that the set is in one piece. Local-connectedness means that the set is connected in the vicinity of each point. The difference between the two notions is best illustrated by the following picture.



Computability of the Mandelbrot set and Julia sets

- It is currently unknown whether the Mandelbrot set is computable. But if we could somehow prove that it is *not* computable then the MLC conjecture would be false. This would be a tremendous breakthrough.

Computability of the Mandelbrot set and Julia sets

- It is currently unknown whether the Mandelbrot set is computable. But if we could somehow prove that it is *not* computable then the MLC conjecture would be false. This would be a tremendous breakthrough.
- On the other hand, if the Mandelbrot set is computable then it is quite possible that the techniques used in the proof could be useful in resolving the MLC conjecture.

Computability of the Mandelbrot set and Julia sets

- It is currently unknown whether the Mandelbrot set is computable. But if we could somehow prove that it is *not* computable then the MLC conjecture would be false. This would be a tremendous breakthrough.
- On the other hand, if the Mandelbrot set is computable then it is quite possible that the techniques used in the proof could be useful in resolving the MLC conjecture.
- It has also been established that for most values of $c \in \mathbb{C}$, the Julia set J_c is computable.

Computability of the Mandelbrot set and Julia sets

- It is currently unknown whether the Mandelbrot set is computable. But if we could somehow prove that it is *not* computable then the MLC conjecture would be false. This would be a tremendous breakthrough.
- On the other hand, if the Mandelbrot set is computable then it is quite possible that the techniques used in the proof could be useful in resolving the MLC conjecture.
- It has also been established that for most values of $c \in \mathbb{C}$, the Julia set J_c is computable.
- In 2005, Braverman and Yampolsky proved that there are values of $c \in \mathbb{C}$ for which the associated Julia set is not computable.

Reference

The subject of computability of fractals is very new and I have not discussed many of the interesting topics. For instance, I have not talked about complexity issues at all. The following recent book gives an excellent overview of the subject and is written with a diverse audience in mind.

Reference

Mark Braverman and Michael Yampolsky, *Computability of Julia Sets*, Berlin: Springer, 2009.

THANK YOU