What Are Computers, Really?

Clarissa Littler

6-23-2015

Fundamental Questions

- What are the limits of computers?

Fundamental Questions

- What are the limits of computers?
- Are there problems a computer can never solve?

Fundamental Questions

- What are the limits of computers?
- Are there problems a computer can never solve?
- Do all programming languages describe the same thing?

Fundamental Questions

- What are the limits of computers?
- Are there problems a computer can never solve?
- Do all programming languages describe the same thing?
- What even are programs?

How We'll Answer Them

- Give intuitive criterion for "computability" as finite process

How We'll Answer Them

- Give intuitive criterion for "computability" as finite process
- Build up a definition of computation independent of computers

How We'll Answer Them

- Give intuitive criterion for "computability" as finite process
- Build up a definition of computation independent of computers
- Sketch out mathematical models of computation

How We'll Answer Them

- Give intuitive criterion for "computability" as finite process
- Build up a definition of computation independent of computers
- Sketch out mathematical models of computation
- Give examples of non-computable problems

How We'll Answer Them

- Give intuitive criterion for "computability" as finite process
- Build up a definition of computation independent of computers
- Sketch out mathematical models of computation
- Give examples of non-computable problems
- Discuss the implications and limits of our knowledge of computability

Computation Sounds Like Computer

- Computation is what computers do

Computation Sounds Like Computer

- Computation is what computers do
- A program *describes* a computation

Computation Sounds Like Computer

- Computation is what computers do
- A program *describes* a computation
- The limits of computation are limits of description

Computation Sounds Like Computer

- Computation is what computers do
- A program *describes* a computation
- The limits of computation are limits of description
- What processes can be described in a finite way with a finite implementation?

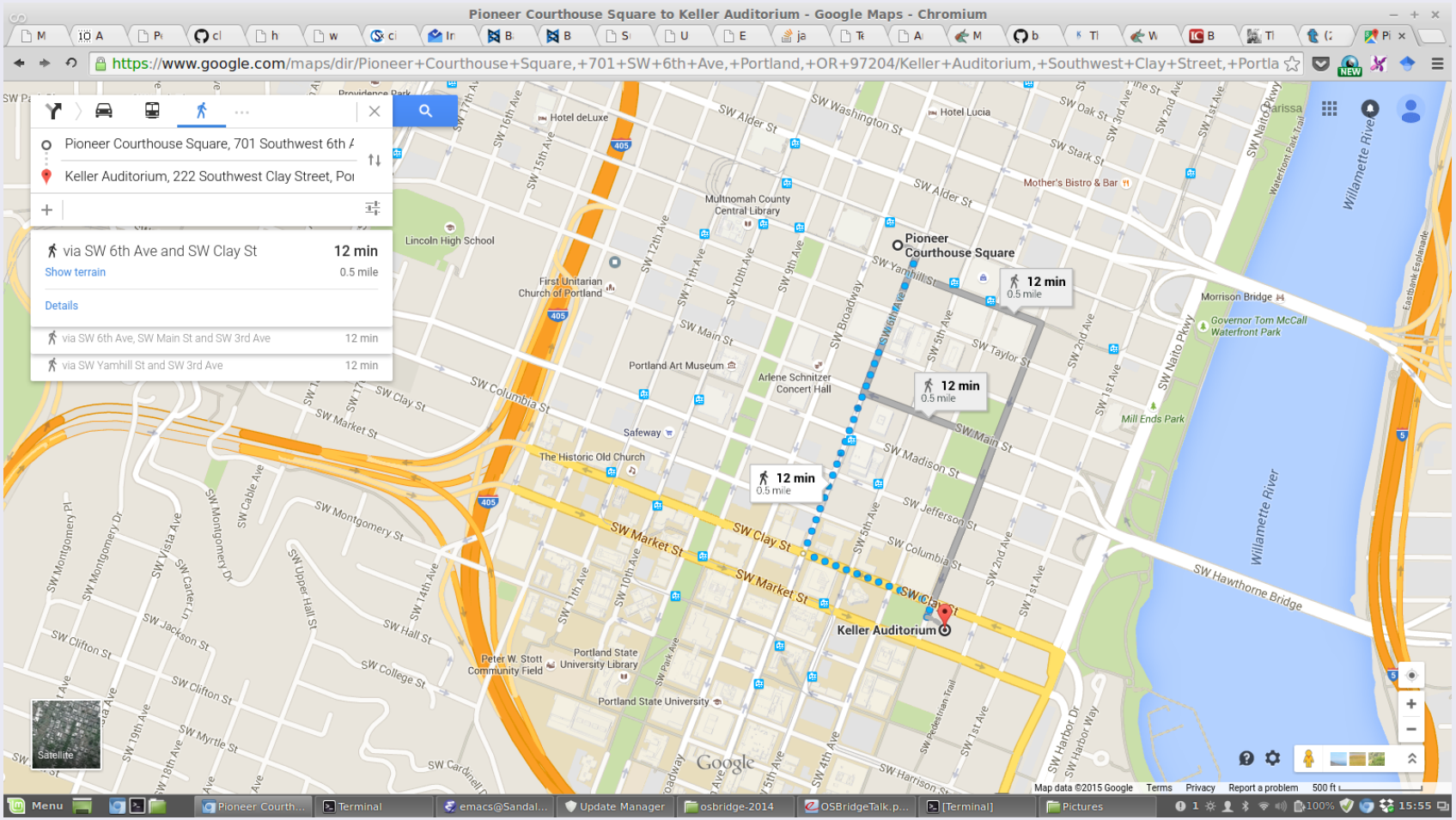
Examples of Finite Processes

Recipe as Finite Process

Cook celery and onion together til soft, then add frozen spinach and cook to get some of the moisture out and reduce volume add broth lentils cilantro and other spices, stir thoroughly, throw bay leaves on top.
Cook for 40 minutes
Turn off heat, wait til it stops bubbling and blend thoroughly.
Cook for 5-10 minutes after blending

Examples of Finite Processes

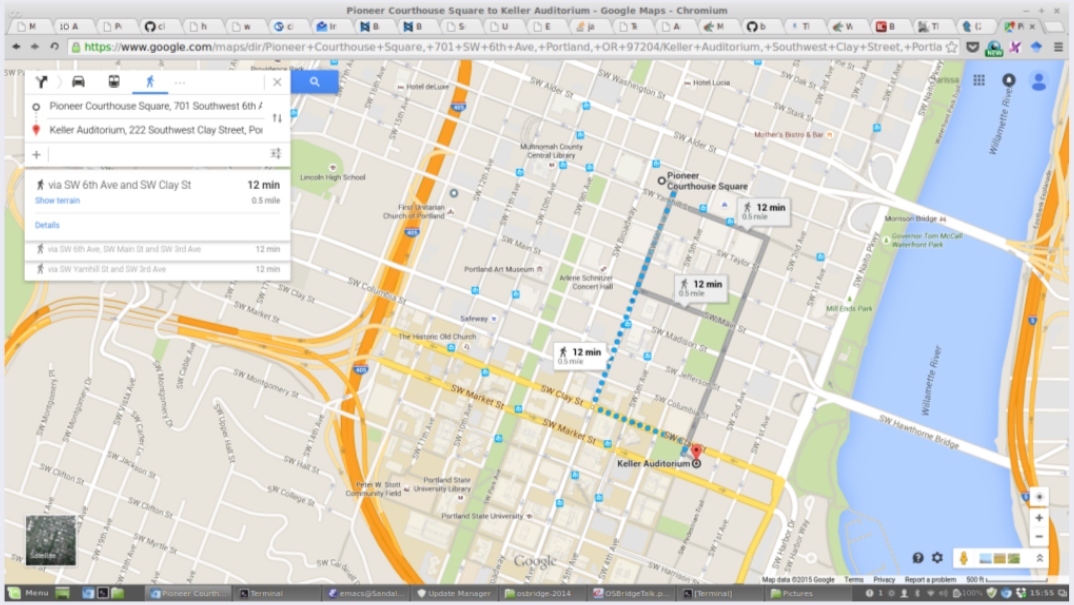
Another finite process



Examples of Finite Processes

Examples of Finite Processes

Another finite process

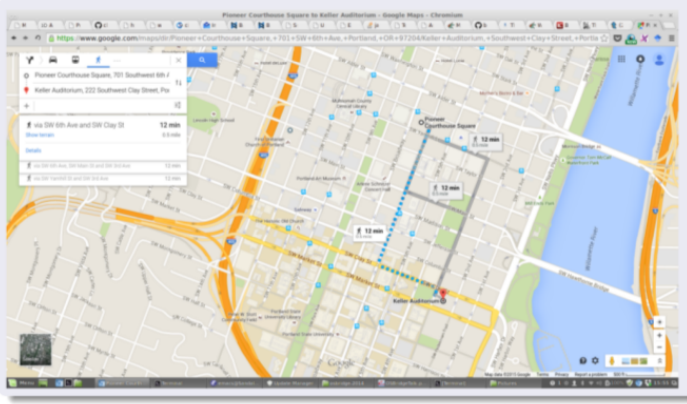


Examples of Finite Processes

Examples of Finite Processes

Examples of Finite Processes

Another finite process

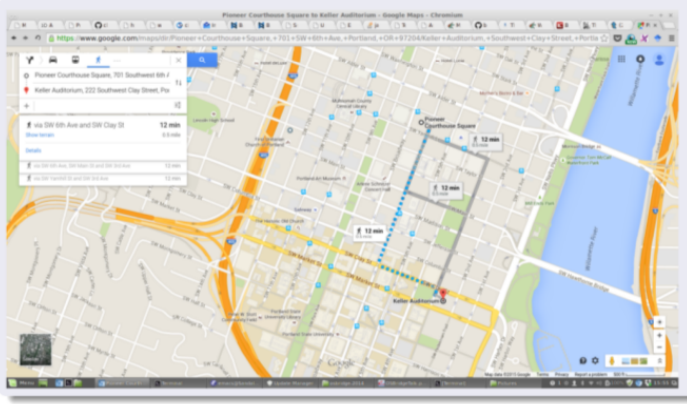


Examples of Finite Processes

Examples of Finite Processes

Examples of Finite Processes

Another finite process



I'm done with this joke now, I promise.

Examples of Finite Processes

Examples of Finite Processes

Adding

$$\begin{aligned} 5 + 10 &= 4 + 11 \\ &= 3 + 12 \\ &= 2 + 13 \\ &= 1 + 14 \\ &= 0 + 15 \\ &= 15 \end{aligned}$$

Examples of Finite Processes

Adding

$$\begin{aligned} 5 + 10 &= 4 + 11 \\ &= 3 + 12 \\ &= 2 + 13 \\ &= 1 + 14 \\ &= 0 + 15 \\ &= 15 \end{aligned}$$

Another Way

$$\begin{aligned} 5 + 10 &= 6 + 9 \\ &= 7 + 8 \\ &= 8 + 7 \\ &= 9 + 6 \\ &= 10 + 5 \\ &= 11 + 4 \\ &= 12 + 3 \\ &= 13 + 2 \\ &= 14 + 1 \\ &= 15 \end{aligned}$$

Examples of Finite Processes

The following Haskell snippet that evaluates the sum of the integers from 1 to 10 is also a finite process

Examples of Finite Processes

The following Haskell snippet that evaluates the sum of the integers from 1 to 10 is also a finite process

```
let f x = sum [1..x] in f 10
```

Examples of Finite Processes

Many more examples exist in the wild including:

Examples of Finite Processes

Many more examples exist in the wild including:

- counting on your fingers

Examples of Finite Processes

Many more examples exist in the wild including:

- counting on your fingers
- long division

Examples of Finite Processes

Many more examples exist in the wild including:

- counting on your fingers
- long division
- sorting your vinyl collection with a bucket sort

Examples of Finite Processes

Many more examples exist in the wild including:

- counting on your fingers
- long division
- sorting your vinyl collection with a bucket sort
- compiling code

Examples of Finite Processes

Many more examples exist in the wild including:

- counting on your fingers
- long division
- sorting your vinyl collection with a bucket sort
- compiling code

Qualities of Finite Processes

Informal criterion for a “finite process”

Qualities of Finite Processes

Informal criterion for a "finite process"

Finite Implementation

- Finite time
- Finite resources

Qualities of Finite Processes

Informal criterion for a "finite process"

Finite Implementation

- Finite time
- Finite resources

Finite Description

- Finite length
- Finite alphabet

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

- how massive our Sun is

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

- how massive our Sun is
- the distance between planets

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

- how massive our Sun is
- the distance between planets
- the number of lines of code in your program

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

- how massive our Sun is
- the distance between planets
- the number of lines of code in your program
- the number of words in this talk

Digression: What Does Finite Mean?

Informal Intuition

- A quantity is finite when it is "measurable"
  - Counting
  - Weighing
  - Timing

Examples

- how massive our Sun is
- the distance between planets
- the number of lines of code in your program
- the number of words in this talk
- number of other talks you'd rather be at

Finite Time

Finite process produces output in finite time

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Why Finite Time?

Only actions taking finite time can actually be finished because that's how our universe works.

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Why Finite Time?

Only actions taking finite time can actually be finished because that's how our universe works.

Produce Output?

- Some finite processes run forever

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Why Finite Time?

Only actions taking finite time can actually be finished because that's how our universe works.

Produce Output?

- Some finite processes run forever
  - Operating systems

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Why Finite Time?

Only actions taking finite time can actually be finished because that's how our universe works.

Produce Output?

- Some finite processes run forever
  - Operating systems
  - Servers

Finite Time

Finite process produces output in finite time

Examples of Finite Time

- Counting on your fingers
- Sorting vinyl
- Walking to a friend's house
- Boiling ramen

Why Finite Time?

Only actions taking finite time can actually be finished because that's how our universe works.

Produce Output?

- Some finite processes run forever
  - Operating systems
  - Servers
  - Interactive programs

Finite Resources

Finite processes only use finite resources

Examples of Resources

- scratch paper
- materials
- RAM
- disk space

Why Finite Resources

No computer and no physical process that we know of can use an infinite quantity, thus infinite resources shouldn't be allowed in computation.

Finite Descriptions

Finite processes are only allowed to have a finite number of steps in their description, i.e. a finite length as a string

Examples of Finite Descriptions

- Directions
- Recipes
- Programs
  - a `while` loop is finite!

Why Finite Directions?

Any process that has an infinite number of steps in its description would:

Finite Descriptions

Finite processes are only allowed to have a finite number of steps in their description, i.e. a finite length as a string

Examples of Finite Descriptions

- Directions
- Recipes
- Programs
  - a `while` loop is finite!

Why Finite Directions?

Any process that has an infinite number of steps in its description would:

- take necessarily infinite time to process and run
  - not absolutely a bad thing, but likely so
- take infinite resources to store
  - this is absolutely bad

Finite Alphabet

Finite processes can only be written with a finite alphabet

Examples of Alphabets

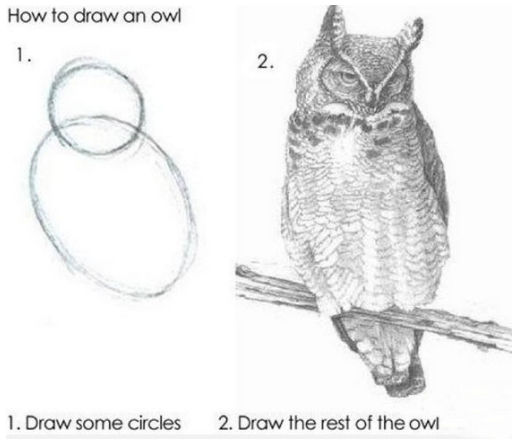
- 0 and 1
- the English alphabet (a-z, A-Z)
- unicode
- ASCII

Why a Finite Alphabet?

An infinite alphabet can't have an implementation that is, itself, a finite process.

What Next?

- These are rules of thumb
- But how do we actually specify a process?
- Most directions too broad:
  - driving directions
  - cooking directions
- Need instructions simple enough for a machine



Modelling Computation

What Is a Model?

A model of computation is a mathematically precise formulation of computation.

Modelling Computation

What Is a Model?

A model of computation is a mathematically precise formulation of computation.

What's in A Model?

- a rigorous way to describe computation

Modelling Computation

What Is a Model?

A model of computation is a mathematically precise formulation of computation.

What's in A Model?

- a rigorous way to describe computation
- a way to perform the descriptions

The Search for a Model

- 1930s logicians searched for theorem proving algorithms

The Search for a Model

- 1930s logicians searched for theorem proving algorithms
  - Proofs in 1st order logic

The Search for a Model

- 1930s logicians searched for theorem proving algorithms
  - Proofs in 1st order logic
  - The decision problem

The Search for a Model

- 1930s logicians searched for theorem proving algorithms
  - Proofs in 1st order logic
  - The decision problem
- Needed a model of computation to prove if needed computation could exist

The Search for a Model

- 1930s logicians searched for theorem proving algorithms
  - Proofs in 1st order logic
  - The decision problem
- Needed a model of computation to prove if needed computation could exist
- Turing won the race!

Turing and His Automatic Machines

- Turing's 1936 paper "On Computable Numbers, with an Application to the [Decision Problem]" [5]

Turing and His Automatic Machines

- Turing's 1936 paper "On Computable Numbers, with an Application to the [Decision Problem]" [5]
- Automatic machines weren't actually stand-ins for modern computers

Turing and His Automatic Machines

- Turing's 1936 paper "On Computable Numbers, with an Application to the [Decision Problem]" [5]
- Automatic machines weren't actually stand-ins for modern computers
- Turing was inspired by human computers

Human Computers

- Turing's day computers were people

Human Computers

- Turing's day computers were people
  - Actually, they were mostly women [3]

Human Computers

- Turing's day computers were people
  - Actually, they were mostly women [3]
- Computers worked on

Human Computers

- Turing's day computers were people
  - Actually, they were mostly women [3]
- Computers worked on
  - Physics simulations

Human Computers

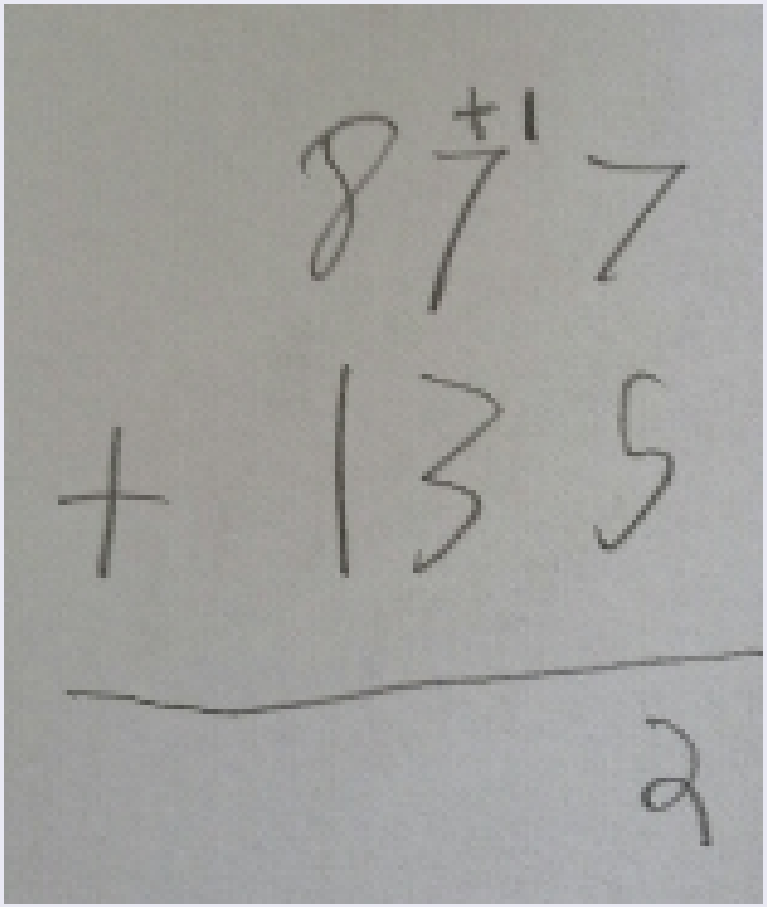
- Turing's day computers were people
  - Actually, they were mostly women [3]
- Computers worked on
  - Physics simulations
  - Scientific research

Human Computers

- Turing's day computers were people
  - Actually, they were mostly women [3]
- Computers worked on
  - Physics simulations
  - Scientific research
  - Firing tables

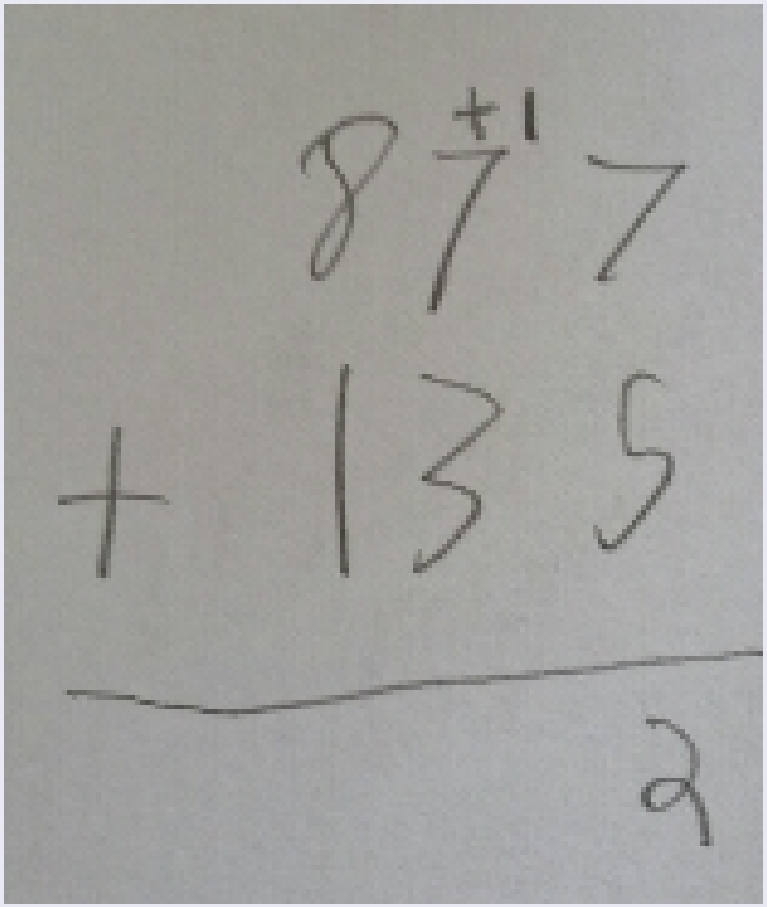
How Humans Compute

An Example



How Humans Compute

An Example

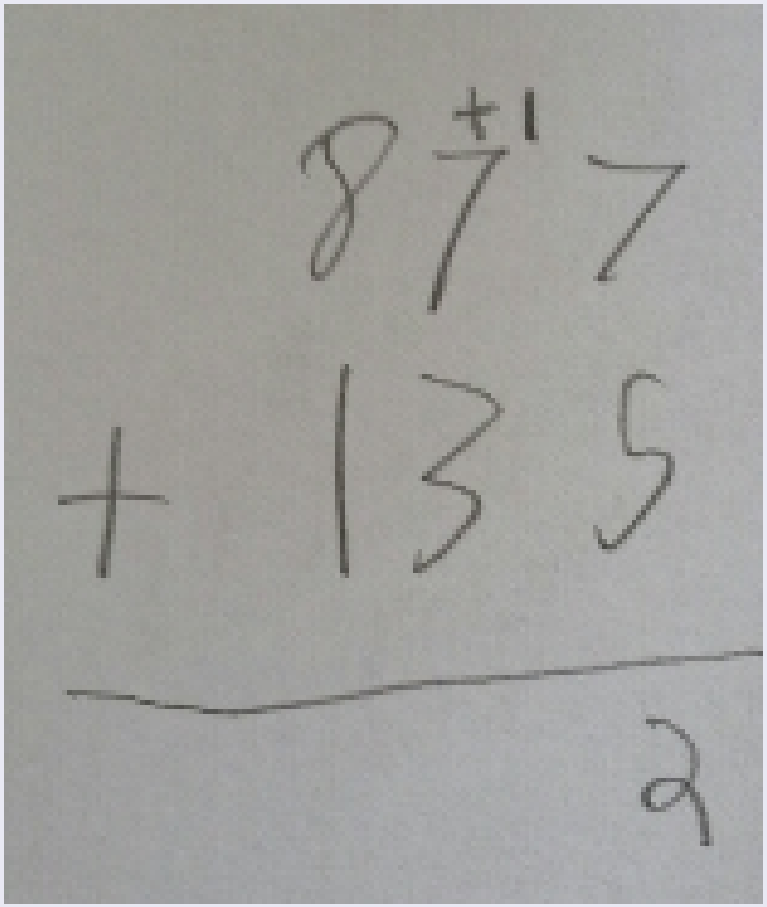


How We Think

- Finite scratch paper

How Humans Compute

An Example

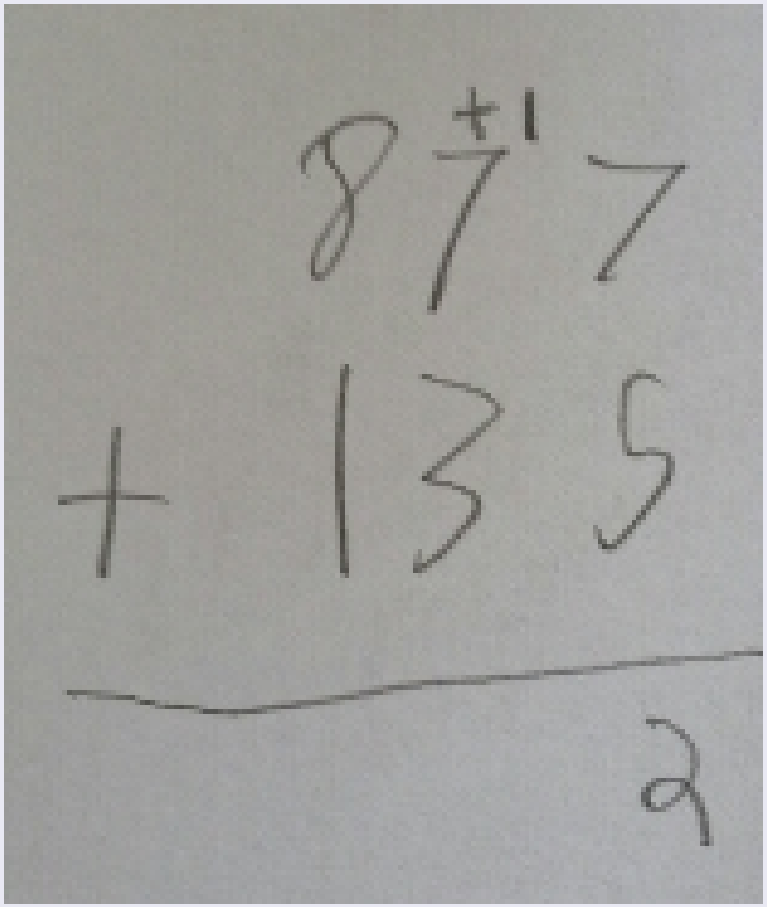


How We Think

- Finite scratch paper
- Finite steps

How Humans Compute

An Example

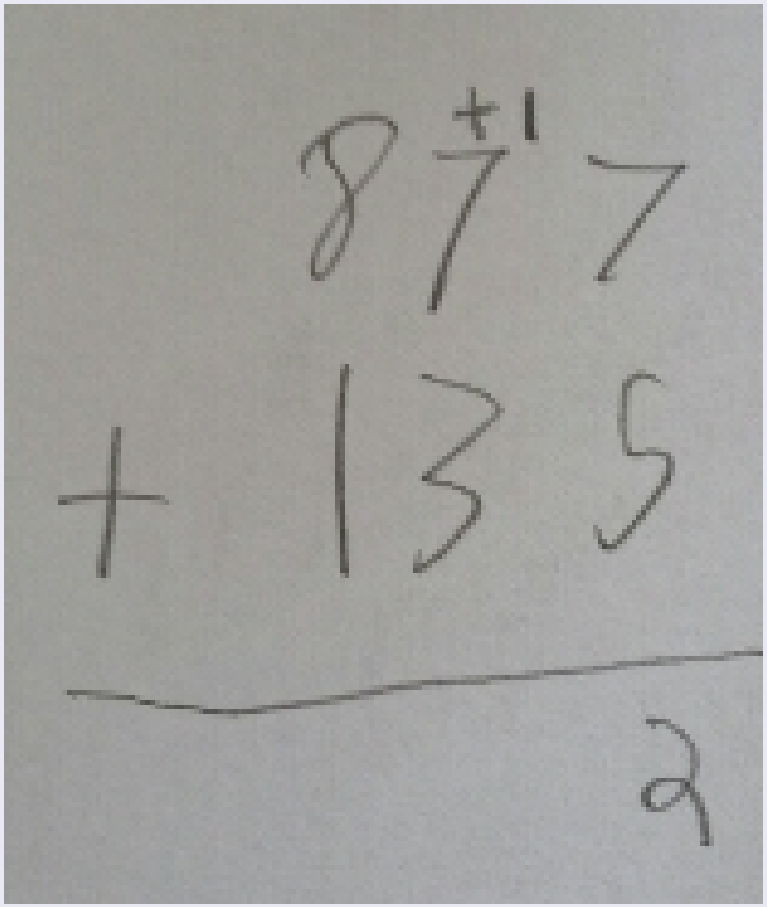


How We Think

- Finite scratch paper
- Finite steps
- You can look at it and pick up where I left off

How Humans Compute

An Example

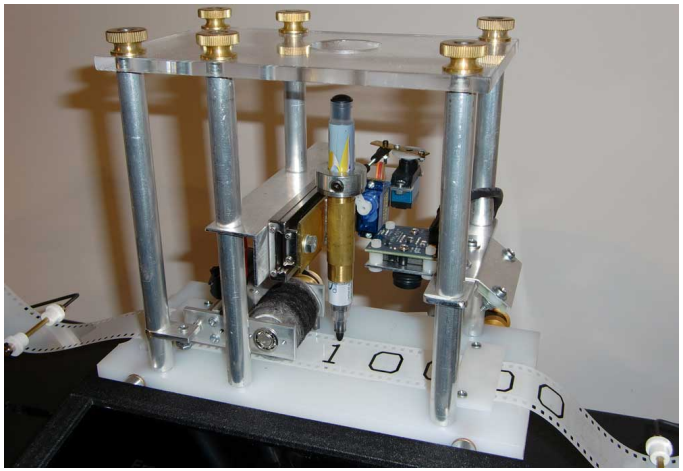


How We Think

- Finite scratch paper
- Finite steps
- You can look at it and pick up where I left off
- Only requires a finite number of brain states to perform

A Turing Machine

- Arbitrary amount of tape
- Reads and writes from only once cell at a time
- Only has a finite "alphabet" of symbols
- Has a finite number of states for deciding next move



Courte

of

`http://aturingmachine.com/hardware.php`

Historic Importance of Turing Machines

- Church's lambda calculus was slightly first [2]

Historic Importance of Turing Machines

- Church's lambda calculus was slightly first [2]
  - Mere months!

Historic Importance of Turing Machines

- Church's lambda calculus was slightly first [2]
  - Mere months!
- The lambda calculus wasn't obviously computable

Historic Importance of Turing Machines

- Church's lambda calculus was slightly first [2]
  - Mere months!
- The lambda calculus wasn't obviously computable
- Turing's machines are obviously computable

Historic Importance of Turing Machines

- Church's lambda calculus was slightly first [2]
  - Mere months!
- The lambda calculus wasn't obviously computable
- Turing's machines are obviously computable
  - Completely analogous to human processes

The Church-Turing Thesis

Original Formulation

There is no model of computation more expressive than Turing machines (equivalently, the lambda calculus). [4]

Equivalent Formulation

Equivalent formulation: no programming language can be more powerful than a Turing machine

Programs As Computations

- A programming language is a model of computation

Programs As Computations

- A programming language is a model of computation
- A program a description of a computation

Programs As Computations

- A programming language is a model of computation
- A program a description of a computation
- An interpreter/compiler is the implementation of the description

Programs As Computations

- A programming language is a model of computation
- A program a description of a computation
- An interpreter/compiler is the implementation of the description
- The real meaning of Turing Complete

The Halting Problem

Formal Specification

Is there a Turing machine that, when fed the description of another Turing machine, determines if that machine will halt on a given input?

The Halting Problem

Formal Specification

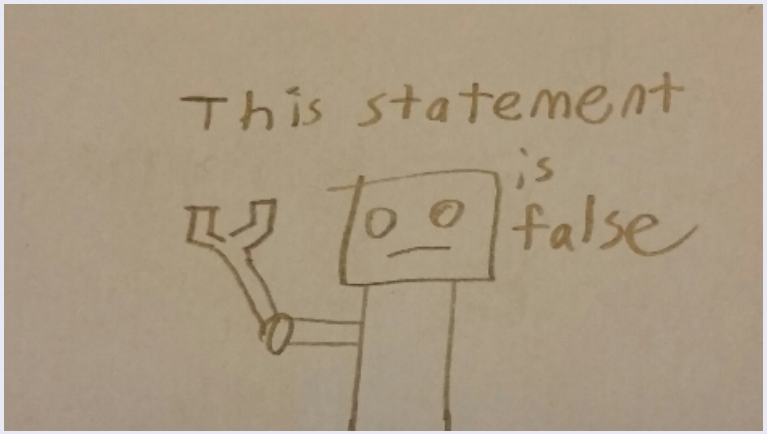
Is there a Turing machine that, when fed the description of another Turing machine, determines if that machine will halt on a given input?

Informal Implication

Can you write a program that can detect if other programs have infinite loops?

Why Can't We Solve The Halting Problem?

Liar's Paradox

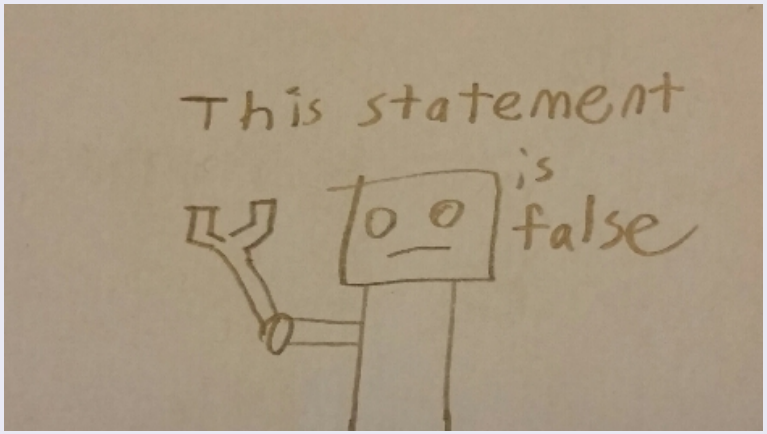


Proof Idea

- Assume we have a program that can solve the halting problem

Why Can't We Solve The Halting Problem?

Liar's Paradox

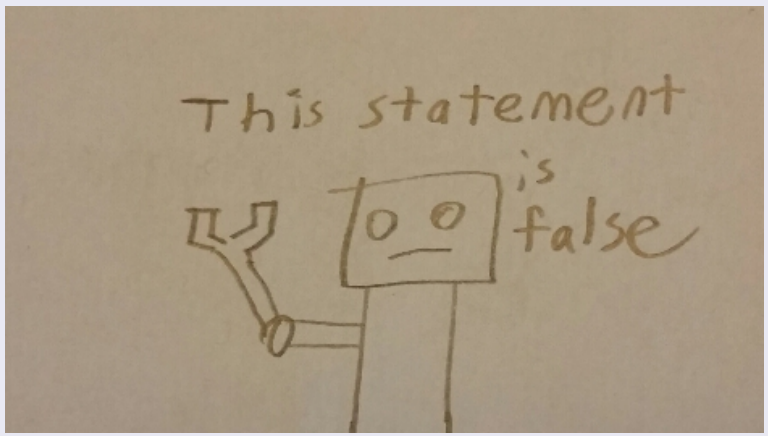


Proof Idea

- Assume we have a program that can solve the halting problem
- Use it to make a liar-compiler that

Why Can't We Solve The Halting Problem?

Liar's Paradox

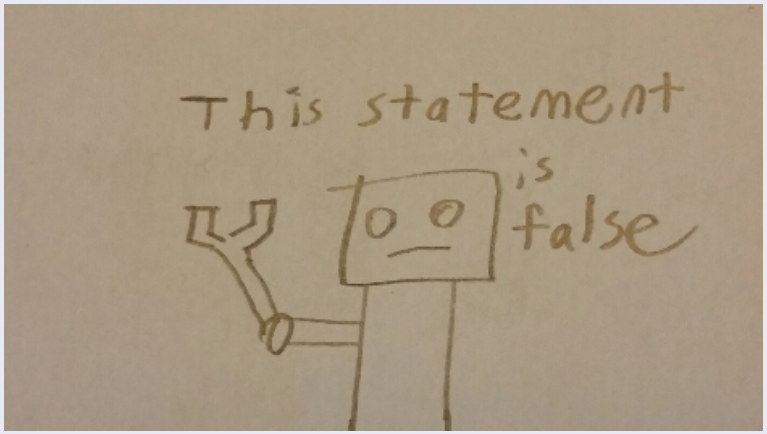


Proof Idea

- Assume we have a program that can solve the halting problem
- Use it to make a liar-compiler that
  - loops if the input program halts when fed its own source code
  - halts if the input program loops when fed its own source code

Why Can't We Solve The Halting Problem?

Liar's Paradox



Proof Idea

- Assume we have a program that can solve the halting problem
- Use it to make a liar-compiler that
  - loops if the input program halts when fed its own source code
  - halts if the input program loops when fed its own source code
- What does the liar say about itself?

Virus Scanners

Full Employment Theorem

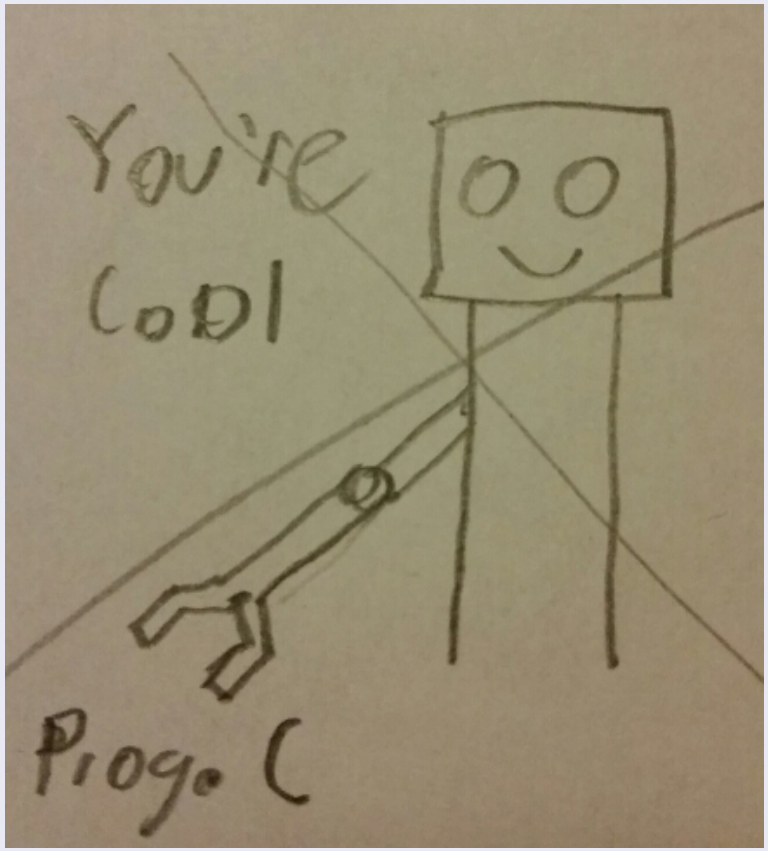
Is there a program that can perfectly detect if another program is carrying a viral payload?

Virus Scanners

Full Employment Theorem

Is there a program that can perfectly detect if another program is carrying a viral payload?

No!

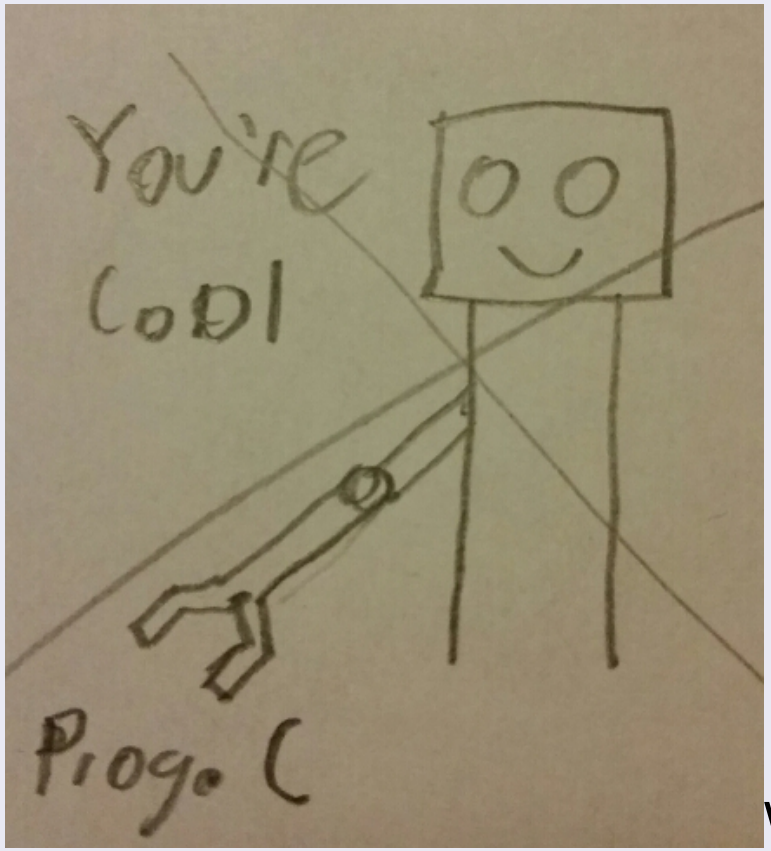


Virus Scanners

Full Employment Theorem

Is there a program that can perfectly detect if another program is carrying a viral payload?

No!



Why not?

Rice's Theorem

Semi-Formally

Non-trivial properties of all programs

Church-Turing as Physics

Strong Church-Turing Thesis

The laws of physics are inherently computable and there is no physical process that cannot be computed by some algorithm.

- is this actually true?

Church-Turing as Physics

Strong Church-Turing Thesis

The laws of physics are inherently computable and there is no physical process that cannot be computed by some algorithm.

- is this actually true?
- we literally have no idea!

Church-Turing as Physics

Strong Church-Turing Thesis

The laws of physics are inherently computable and there is no physical process that cannot be computed by some algorithm.

- is this actually true?
- we literally have no idea!
- strong Church-Turing thesis has major implications for physics

Church-Turing as Physics

Strong Church-Turing Thesis

The laws of physics are inherently computable and there is no physical process that cannot be computed by some algorithm.

- is this actually true?
- we literally have no idea!
- strong Church-Turing thesis has major implications for physics
  - reality must be "discrete"

Church-Turing as Physics

Strong Church-Turing Thesis

The laws of physics are inherently computable and there is no physical process that cannot be computed by some algorithm.

- is this actually true?
- we literally have no idea!
- strong Church-Turing thesis has major implications for physics
  - reality must be "discrete"
  - real numbers are approximations at scale

Church-Turing as Cognition

- Are brains computable?

Church-Turing as Cognition

- Are brains computable?
- Currently an unknown question

Church-Turing as Cognition

- Are brains computable?
- Currently an unknown question
- Does free will actually exist or is it an illusion?

Is Strong AI Possible?

- Can we make a machine intelligence comparable to our own?

Is Strong AI Possible?

- Can we make a machine intelligence comparable to our own?
- Is human intelligence computable?

Is Strong AI Possible?

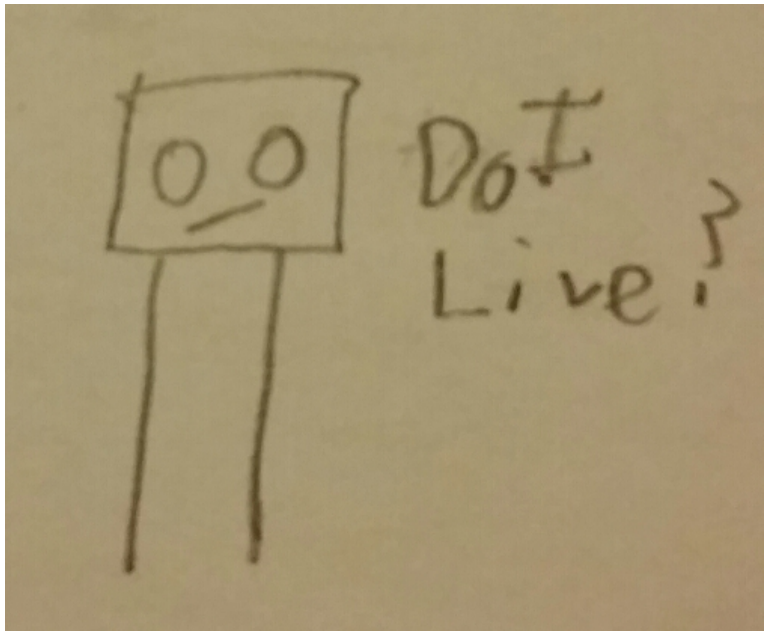
- Can we make a machine intelligence comparable to our own?
- Is human intelligence computable?
- Science fiction assumes so!

Is Strong AI Possible?

- Can we make a machine intelligence comparable to our own?
- Is human intelligence computable?
- Science fiction assumes so!
- Deep philosophic and physical implications

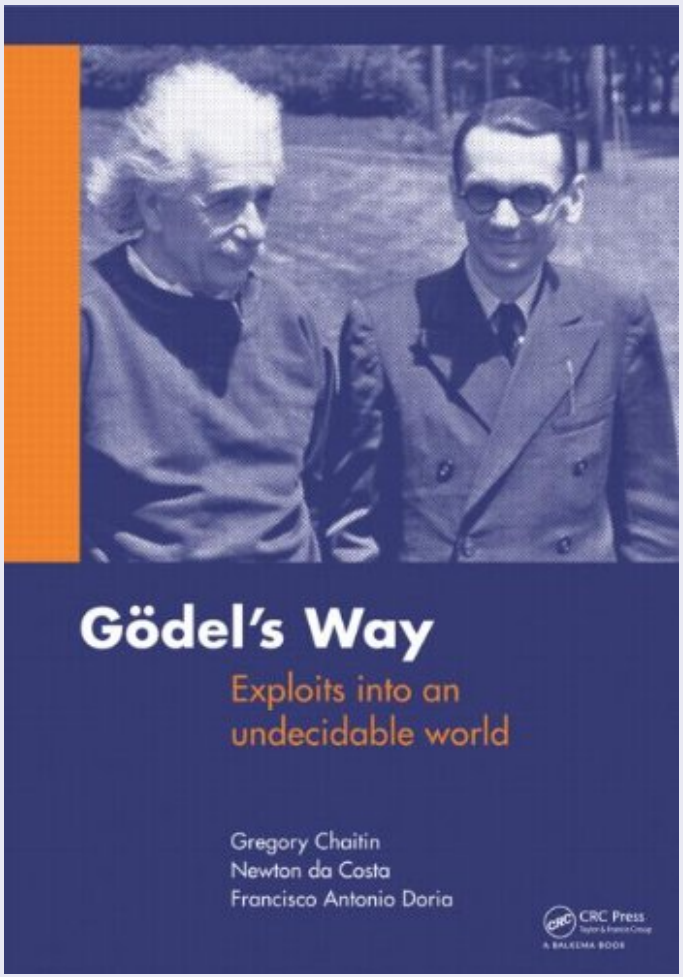
Is Strong AI Possible?

- Can we make a machine intelligence comparable to our own?
- Is human intelligence computable?
- Science fiction assumes so!
- Deep philosophic and physical implications



Skirting Computability

Goedel's Way

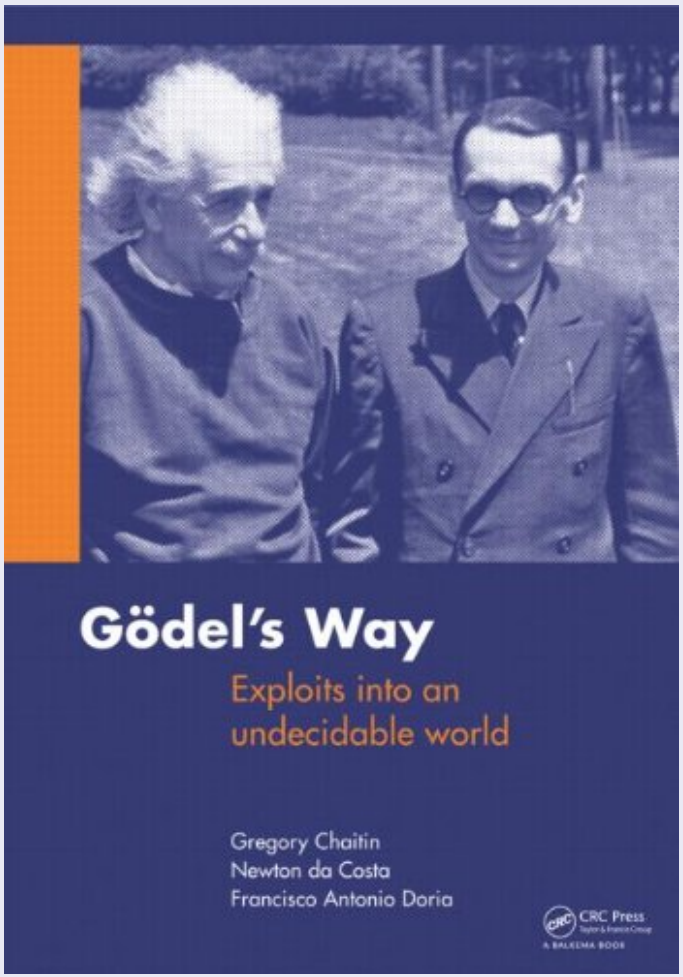


Hyper-Turing Computation

- Is computation (in the Church-Turing sense) complete?

Skirting Computability

Goedel's Way

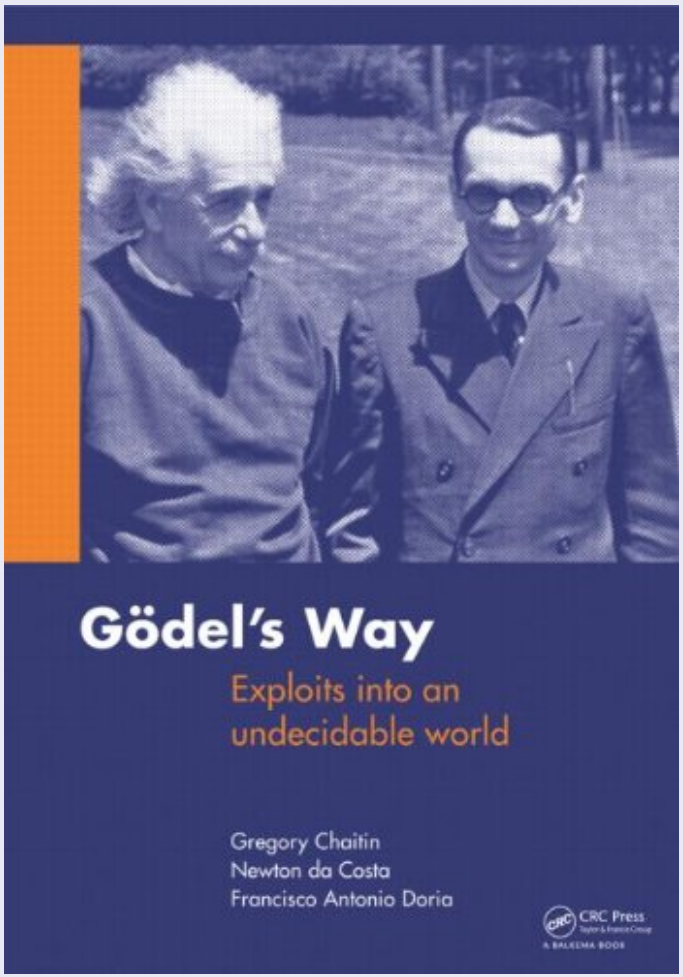


Hyper-Turing Computation

- Is computation (in the Church-Turing sense) complete?
- Is physics computable?

Skirting Computability

Goedel's Way

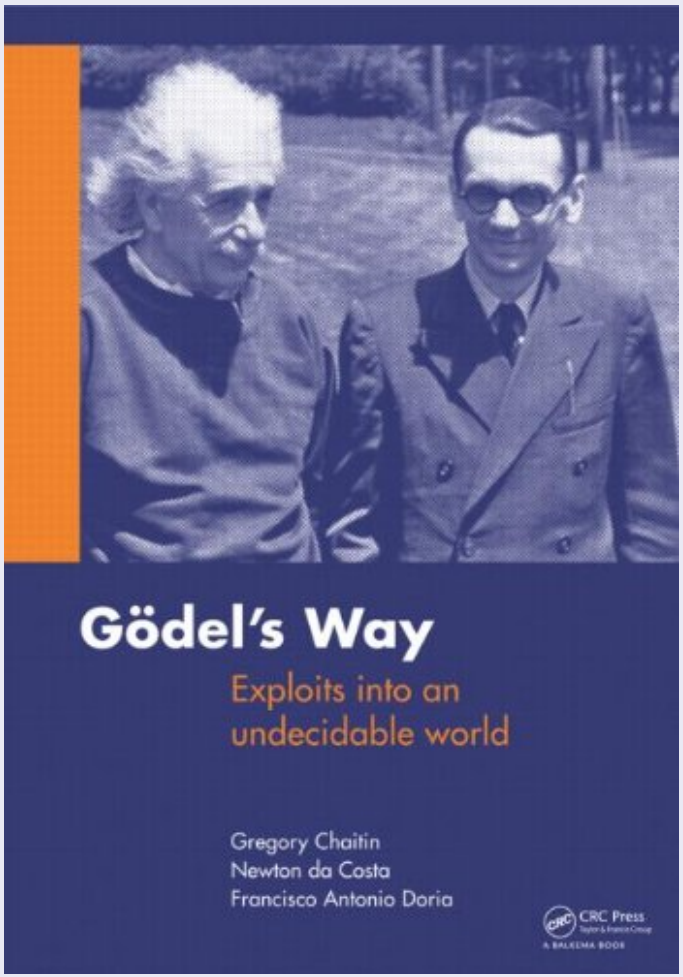


Hyper-Turing Computation

- Is computation (in the Church-Turing sense) complete?
- Is physics computable?
  - If yes, then no

Skirting Computability

Goedel's Way

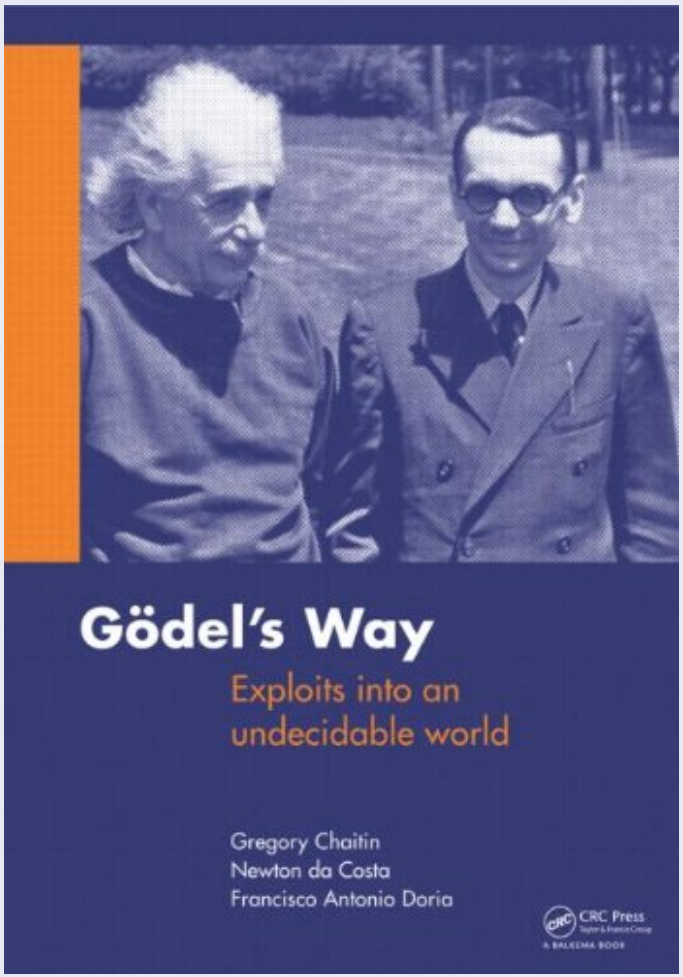


Hyper-Turing Computation

- Is computation (in the Church-Turing sense) complete?
- Is physics computable?
  - If yes, then no
  - If no, then maybe [1]

Skirting Computability

Goedel's Way



Hyper-Turing Computation

- Is computation (in the Church-Turing sense) complete?
- Is physics computable?
  - If yes, then no
  - If no, then maybe [1]
- Hyper-Turing computation

In Conclusion

- Computation is a thing that exists outside computers

In Conclusion

- Computation is a thing that exists outside computers
- The mathematics of finite methods

In Conclusion

- Computation is a thing that exists outside computers
- The mathematics of finite methods
- Computation has limits

In Conclusion

- Computation is a thing that exists outside computers
- The mathematics of finite methods
- Computation has limits
- The limits of computation are understood

In Conclusion

- Computation is a thing that exists outside computers
- The mathematics of finite methods
- Computation has limits
- The limits of computation are understood
- How computation relates to the laws of the universe?

In Conclusion

- Computation is a thing that exists outside computers
- The mathematics of finite methods
- Computation has limits
- The limits of computation are understood
- How computation relates to the laws of the universe?
  - Much more unknown

Any Questions?

Bibliography


Gregory Chaitin, Francisco A Doria, and Newton CA da Costa.
Gödel's way: exploits into an undecidable world.
CRC Press, 2011.


Alonzo Church.
An unsolvable problem of elementary number theory.
American journal of mathematics, pages 345–363, 1936.


David Alan Grier.
When computers were human.
Princeton University Press, 2013.


Stephen Cole Kleene, NG de Bruijn, J de Groot, and Adriaan Cornelis Zaanen.
Introduction to metamathematics.
1952.


Alan Mathison Turing.