Not not to be or not to be?

Andrej Bauer

Institute for Mathematics, Physics, and Mechanics
University of Ljubljana
Slovenia

Distinguished Lecture Series
Carnegie Mellon University, February 2002

What is wrong with these theorems?

Theorem: Three points are either collinear or not.

Theorem: A non-constant polynomial has a complex root.

Theorem: Most functions are everywhere discontinuous.

What is wrong with these theorems?

Theorem: Three points are either collinear or not.

But testing for collinearity is numerically unstable.

Theorem: A non-constant polynomial has a complex root.

Theorem: Most functions are everywhere discontinuous.

What is wrong with these theorems?

Theorem: Three points are either collinear or not.

But testing for collinearity is numerically unstable.

Theorem: A non-constant polynomial has a complex root.

But classical proofs say nothing about how to compute a root.

Theorem: Most functions are everywhere discontinuous.

What is wrong with these theorems?

Theorem: Three points are either collinear or not.

But testing for collinearity is numerically unstable.

Theorem: A non-constant polynomial has a complex root.

But classical proofs say nothing about how to compute a root.

Theorem: Most functions are everywhere discontinuous.

But such functions are irrelevant for computer science.

Branches of Math tailored for Comp. Science

- Theory of computability
- Computational complexity & algorithms
- Numerical analysis
- Domain theory
- Cryptography
- Queueing theory
- Finite model theory
- Machine learning
- Type theory
- ...

Did you ever ask yourself . . .

. . . whether mathematics itself

can be tailored for computer science?

Did you ever ask yourself . . .

. . . whether mathematics itself

can be tailored for computer science?

One way to do this is *realizability theory*.

Overview

1. Building a Realizability World
2. Life in a Realizability World
3. Practical Considerations

How do we create a world of mathematics?

Use category theory, of course.

How do we create a world of mathematics?

Use category theory, of course.

Two steps:

1. Find a category that describes our view of the world.

2. Apply tools of categorical logic to study it.

How do we create a world of mathematics?

Use category theory, of course.

Two steps:

1. Find a category that describes our view of the world.

   category $=$ objects $+$ morphisms

2. Apply tools of categorical logic to study it.

   Take categorical logic course in the Philosophy Dept.

Computational Views of the World

- Everything is made of Turing machines.

Computational Views of the World

- Everything is made of Turing machines.
- Everything is made of ML programs.

Computational Views of the World

- Everything is made of Turing machines.
- Everything is made of ML programs.
- Everything is made of Scott domains.

Computational Views of the World

- Everything is made of Turing machines.
- Everything is made of ML programs.
- Everything is made of Scott domains.
- The *relative* view:
  1. Which data can be *represented*?
  2. How do we *compute* with data?

Relative Computability

- Data: contents of infinite RAM
- Computation: (finite) program

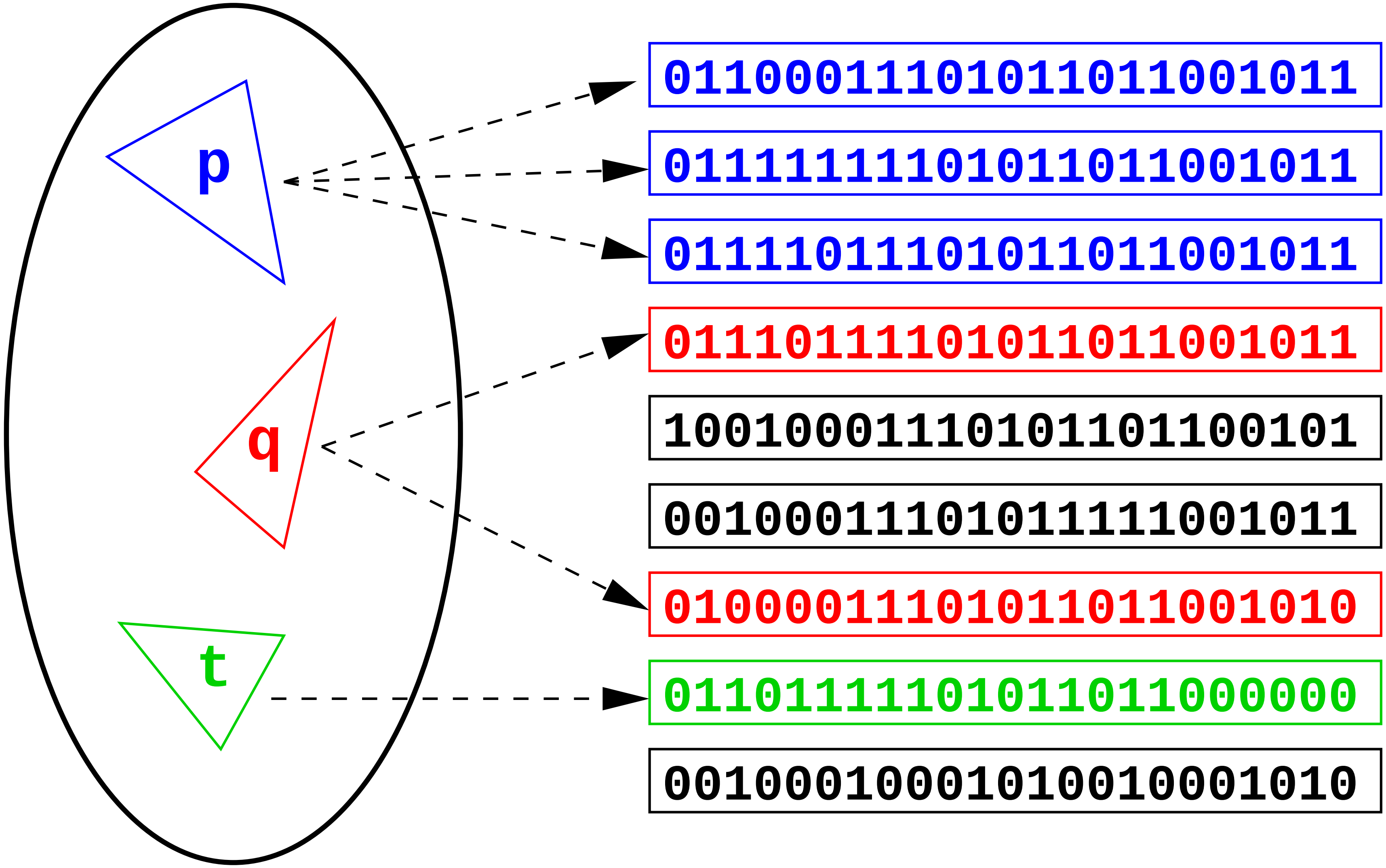
Relative Computability

- Data: contents of infinite RAM
  - *all* configurations possible, also non-computable
  - other sources of data (input streams) can be added
- Computation: (finite) program

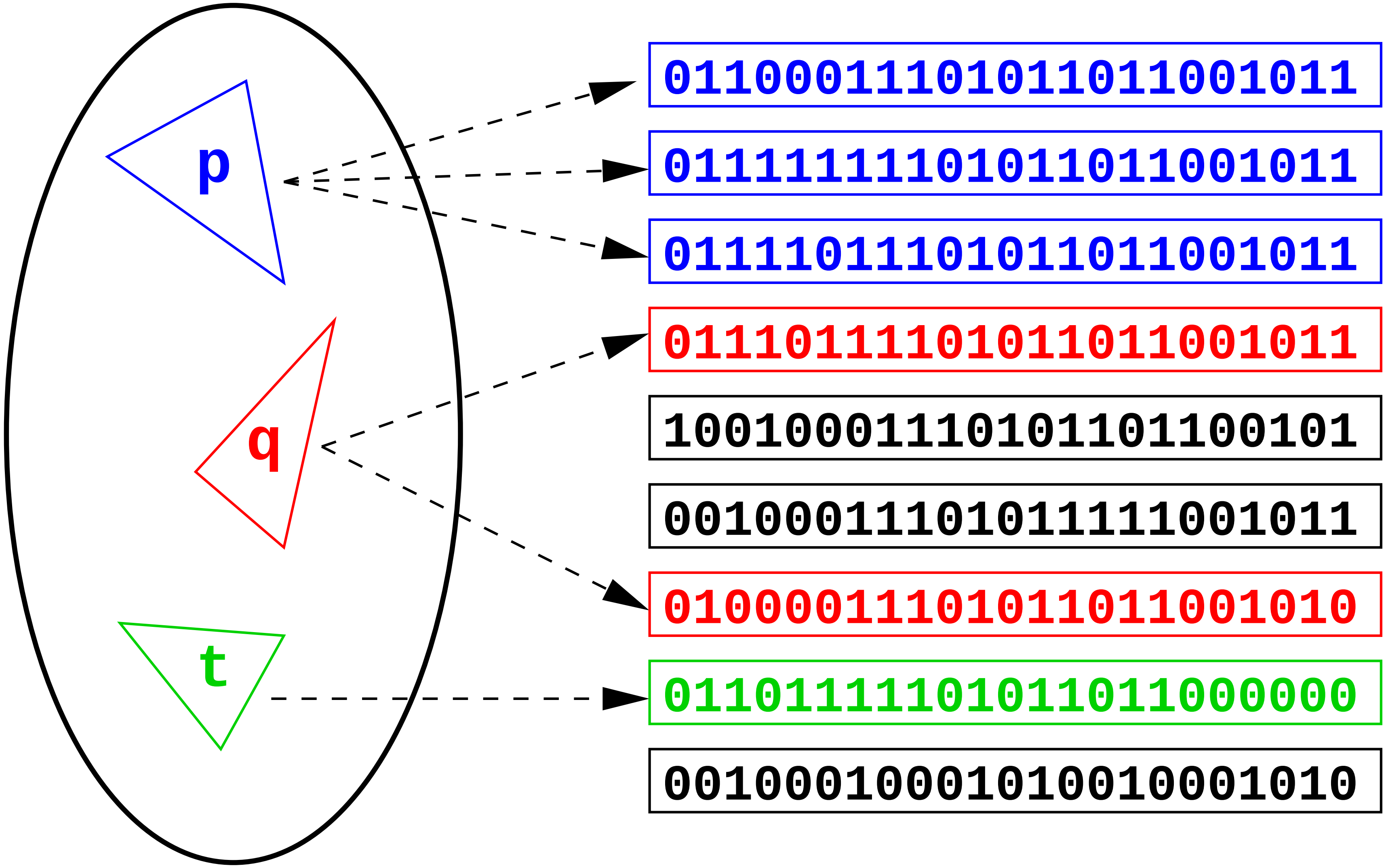
Relative Computability

- Data: contents of infinite RAM
  - *all* configurations possible, also non-computable
  - other sources of data (input streams) can be added
- Computation: (finite) program
  - any chosen general programming language
  - different language features give different worlds

Modest Sets

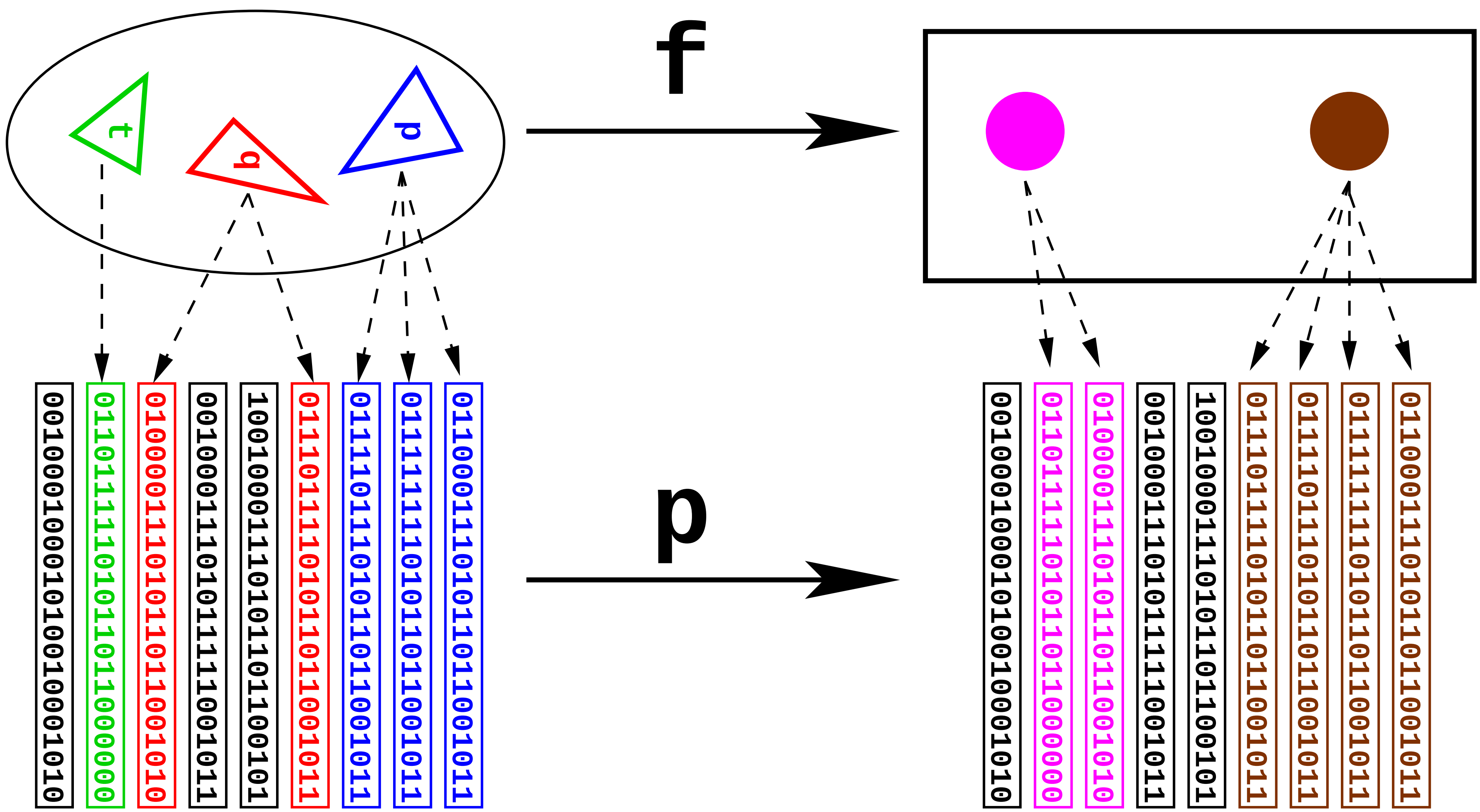

p
q
t
01100011101011011001011
01111111101011011001011
01111011101011011001011
01110111101011011001011
10010001110101101100101
00100011101011111001011
01000011101011011001010
01101111101011011000000
00100010001010010001010

Modest Sets

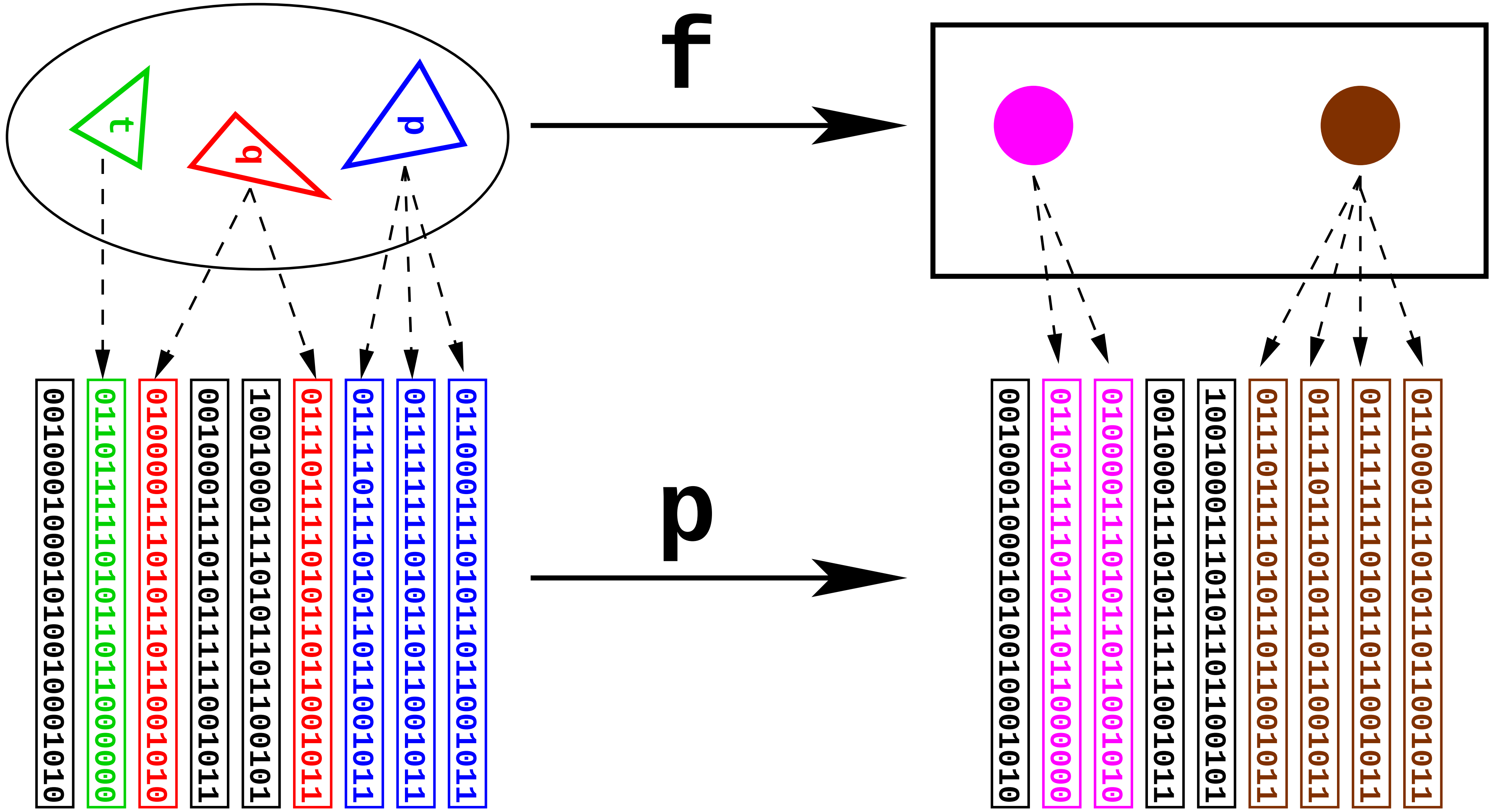


$c \Vdash x$ "c realizes (represents) x"

Realized Functions



Realized Functions



If $\quad c \Vdash x \quad$ then $\quad p(c) \Vdash f(x).$

Computation-aware Sets and Functions

Foundation of classical mathematics:

sets & functions

Computation-aware Sets and Functions

Foundation of classical mathematics:

sets & functions

Foundation of *computation-aware* mathematics:

modest sets & realized functions

Category Theory at Work

Any mathematical structure that has a universal property has a unique representation, up to isomorphism.

Category Theory at Work

Any mathematical structure that has a universal property has a unique representation, up to isomorphism.

1. Natural numbers

   initial algebra with one constant and one unary operation

Category Theory at Work

Any mathematical structure that has a universal property has a unique representation, up to isomorphism.

1. Natural numbers

   initial algebra with one constant and one unary operation

2. Real numbers

   the complete Archimedean field

Category Theory at Work

Any mathematical structure that has a universal property has a unique representation, up to isomorphism.

1. Natural numbers

   initial algebra with one constant and one unary operation

2. Real numbers

   the complete Archimedean field

3. Inductive sets (lists, trees, ...)

Category Theory at Work

Any mathematical structure that has a universal property has a unique representation, up to isomorphism.

1. Natural numbers

   initial algebra with one constant and one unary operation

2. Real numbers

   the complete Archimedean field

3. Inductive sets (lists, trees, . . .)
4. Cartesian product $A \times B$
5. Function space $A \rightarrow B$

Example: real numbers $\mathbb{R}$

A real number $x \in \mathbb{R}$ is realized by a pair $\langle d, e \rangle$:

$$\langle d, e \rangle \Vdash_{\mathbb{R}} x \ ,$$

Example: real numbers $\mathbb{R}$

A real number $x \in \mathbb{R}$ is realized by a pair $\langle d, e \rangle$:

$$\langle d, e \rangle \Vdash_{\mathbb{R}} x \ ,$$

where:

- mantissa $d = d_0 d_1 d_2 \ldots$, with $d_i \in \{-1, 0, 1\}$

Example: real numbers $\mathbb{R}$

A real number $x \in \mathbb{R}$ is realized by a pair $\langle d, e \rangle$:

$$\langle d, e \rangle \Vdash_{\mathbb{R}} x \ ,$$

where:

- mantissa $d = d_0 d_1 d_2 \ldots$, with $d_i \in \{-1, 0, 1\}$
- exponent $e \in \mathbb{Z}$

Example: real numbers $\mathbb{R}$

A real number $x \in \mathbb{R}$ is realized by a pair $\langle d, e \rangle$:

$$\langle d, e \rangle \Vdash_{\mathbb{R}} x \; ,$$

where:

- mantissa $d = d_0 d_1 d_2 \ldots$, with $d_i \in \{-1, 0, 1\}$
- exponent $e \in \mathbb{Z}$
- *signed* binary digit representation:

$$x = 2^e \cdot \sum_{k=0}^{\infty} \frac{d_i}{2^{k+1}}$$

Overview

✓ Building a Realizability World

☞ Life in a Realizability World

3. Practical Considerations

The Particle Physics of Realizability

"Everything is made of tiny invisible realizers."

The Particle Physics of Realizability

"Everything is made of tiny invisible realizers."

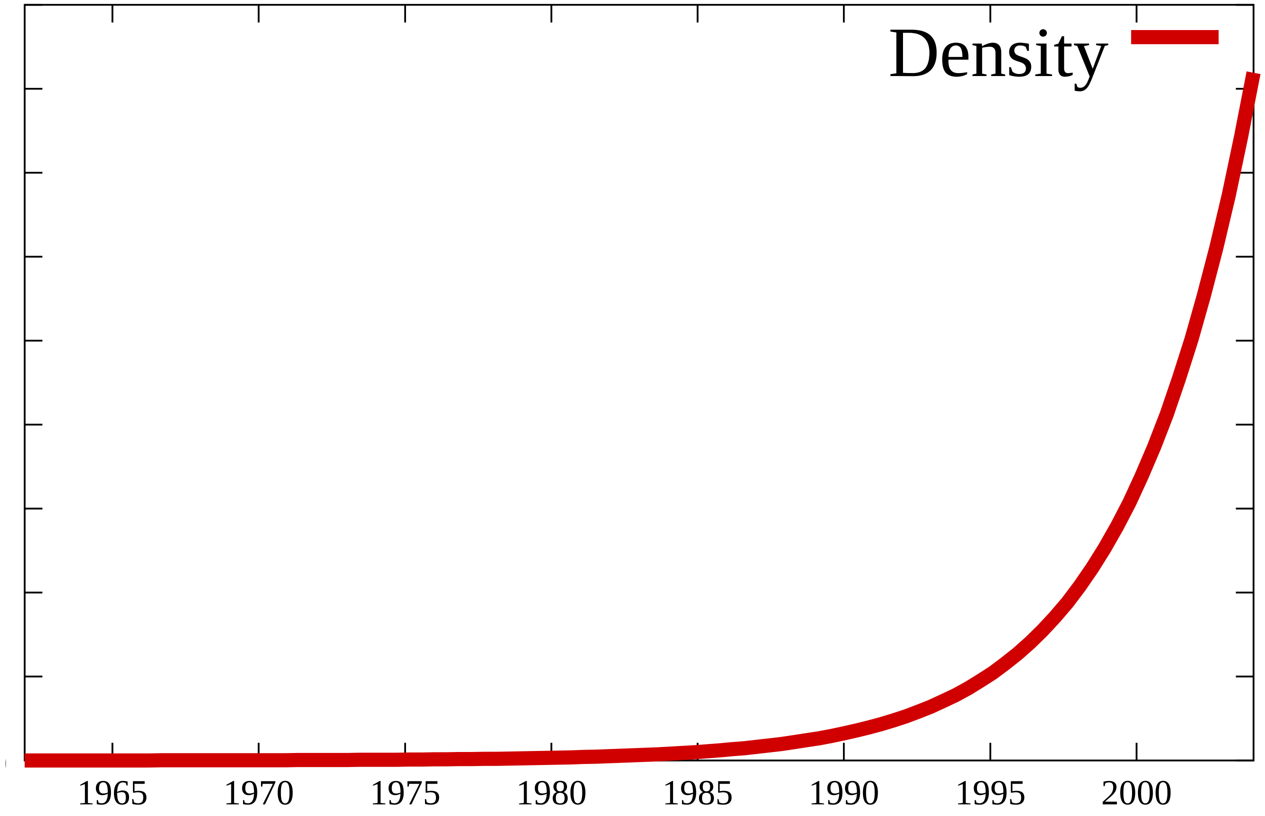
"The two basic realizers are S and K."

$$Kxy = x \qquad Sxyz = (xz)(yz)$$

The Cosmology of Realizability

"The universe is becoming denser at an exponential rate."

(Gordon Moore, 1965)



The Language of Realizability

Computational understanding of truth:

"A statement is true when it is witnessed by a program."

The Language of Realizability

Computational understanding of truth:

"A statement is true when it is witnessed by a program."

Example: a witness for

$$\forall x \in \mathbb{R}.(x < 0 \lor x \geq 0)$$

The Language of Realizability

Computational understanding of truth:

"A statement is true when it is witnessed by a program."

Example: a witness for

$$\forall x \in \mathbb{R}.(x < 0 \vee x \geq 0)$$

is a program p such that, for $\langle d, e \rangle \Vdash_{\mathbb{R}} x$,

$$\begin{aligned} p(d, e) &= 0 \qquad \text{if } x < 0 \\ p(d, e) &= 1 \qquad \text{if } x \geq 0 \end{aligned}$$

Intuitionistic Logic

The logic of realizability is *intuitionistic*.

Intuitionistic Logic

The logic of realizability is *intuitionistic*.

The Law of Excluded Middle is not generally valid:

$$\varphi \vee \neg\varphi$$

Intuitionistic Logic

The logic of realizability is *intuitionistic*.

The Law of Excluded Middle is not generally valid:

$$\varphi \vee \neg\varphi$$

Proof by contradiction in not generally valid:

$$\neg\neg\varphi \implies \varphi$$

Intuitionistic Logic

The logic of realizability is *intuitionistic*.

The Law of Excluded Middle is not generally valid:

$$\varphi \vee \neg\varphi$$

Proof by contradiction in not generally valid:

$$\neg\neg\varphi \Longrightarrow \varphi$$

This is a good thing!

Markov's Principle

If elements of a set A can be enumerated and $\varphi(x)$ is a semi-decidable predicate then

$$(\neg\forall x \in A.\, \neg\varphi(x)) \quad \Longrightarrow \quad \exists x \in A.\, \varphi(x)$$

Markov's Principle

If elements of a set A can be enumerated and $\varphi(x)$ is a semi-decidable predicate then

$$(\neg\forall x \in A.\, \neg\varphi(x)) \quad \Longrightarrow \quad \exists x \in A.\, \varphi(x)$$

Witnessed by a program which iterates through all elements $x_1, x_2, \ldots$ of A and tests $\varphi(x_i)$ until one is found to hold.

All functions are continuous

“All $f : \mathbb{R} \longrightarrow \mathbb{R}$ are continuous.”

All functions are continuous

"All $f : \mathbb{R} \to \mathbb{R}$ are continuous."

"Only finite precision of input is required
for given finite precision of output."

All functions are continuous

"All $f : \mathbb{R} \to \mathbb{R}$ are continuous."

"Only finite precision of input is required
for given finite precision of output."

Witnessed by a program which, given $p \Vdash f$ and $n \in \mathbb{N}$, finds a $k \in \mathbb{N}$ such that p reads only k digits of input to produce n digits of output.

All functions are continuous

"All $f : \mathbb{R} \to \mathbb{R}$ are continuous."

"Only finite precision of input is required
for given finite precision of output."

Witnessed by a program which, given $p \Vdash f$ and $n \in \mathbb{N}$, finds a $k \in \mathbb{N}$ such that p reads only k digits of input to produce n digits of output.

Such a witness exists only if we can use a `throw–catch` programming construct, or a similar control mechanism.

The “not not” translation

Classical logic can be translated into intuitionistic logic.

The "not not" translation

Classical logic can be translated into intuitionistic logic.

Classical	Intuitionistic
$\varphi \implies \psi$	$\neg\neg(\varphi^* \implies \psi^*)$
$\varphi \wedge \psi$	$\neg\neg(\varphi^* \wedge \psi^*)$
$\varphi \vee \psi$	$\neg\neg(\varphi^* \vee \psi^*)$
$\exists x.\ \varphi(x)$	$\neg\neg\exists x.\ \varphi(x)^*$
$\forall x.\ \varphi(x)$	$\neg\neg\forall x.\ \varphi(x)^*$

Not not a Classic Masterpiece

Not not, not to not be, or not to be:
not that is not the question:
Not not, whether not 'tis not nobler in the mind to suffer
The slings and arrows of outrageous fortune,
Or not to not take arms against a sea of troubles,
And not by not opposing not not end them?

(not not Hamlet)

What is still the same?

The “not not” stable statements do not change:

What is still the same?

The “not not” stable statements do not change:

1. All equations and inequations:

What is still the same?

The “not not” stable statements do not change:

1. All equations and inequations:

$$p \text{ witnesses } ``a_1 = a_2" \text{ if, and only if, } a_1 \text{ is } a_2$$

What is still the same?

The "not not" stable statements do not change:

1. All equations and inequations:

$$p \text{ witnesses } \text{“}a_1 = a_2\text{”} \text{ if, and only if, } a_1 \text{ is } a_2$$

2. Any statement built from $=$, $<$, $\wedge$, $\Longrightarrow$ and $\forall$.

What is still the same?

The "not not" stable statements do not change:

1. All equations and inequations:

   $$p \text{ witnesses } ``a_1 = a_2\text{'' if, and only if, } a_1 \text{ is } a_2$$

2. Any statement built from $=$, $<$, $\wedge$, $\implies$ and $\forall$.

Finite combinatorics is pretty much the same.

Overview

- ✓ Building a Realizability World
- ✓ Life in a Realizability World
- ☞ Practical Considerations

Tailored for Computer Science

From the proof of a statement

we obtain an associated program witnessing it.

Tailored for Computer Science

From the proof of a statement
we obtain an associated program witnessing it.

From the construction of a set or a function
we obtain an associated implementation.

Tailored for Computer Science

From the proof of a statement
we obtain an associated program witnessing it.

From the construction of a set or a function
we obtain an associated implementation.

We prove correctness of an implementation
by showing it realizes the desired specification.

A Question for Hardware Designers

Real numbers are represented with *signed* binary digits.

A Question for Hardware Designers

Real numbers are represented with *signed* binary digits.

Would negative digits be useful in hardware implementation of floating point arithmetic?

A Challenge for Computational Geometers

Testing for collinearity is not just numerically unstable, it is *non-constructive*.

A Challenge for Computational Geometers

Testing for collinearity is not just numerically unstable, it is *non-constructive*.

Prove theorems without using the dichotomy

$$\forall x \in \mathbb{R}.\,(x < 0 \vee x \geq 0)$$

Use instead

$$\forall \epsilon > 0.\,\forall x \in \mathbb{R}.\,(x < \epsilon \vee x > -\epsilon)$$

A Task for Programming Language Designers

Ordinary `if-then-else` control mechanism is inappropriate for exact arithmetic.

A Task for Programming Language Designers

Ordinary `if–then–else` control mechanism is inappropriate for exact arithmetic.

Design practical data structures for real numbers *and* invent new control mechanisms for programming with them.

In Conclusion

The world of realizability is *your* world.