Helden van de wiskunde: L.E.J. Brouwer

Brouwers visie vanuit een logica-informatica perspectief

Herman Geuvers
Radboud Universiteit Nijmegen
Technische Universiteit Eindhoven

Helden van de wiskunde: L.E.J. Brouwer

Brouwers visie vanuit een logica-informatica perspectief

Brouwers visie vanuit een logica-informatica perspectief

Is een bewijs een programma??

Rekenen met oneidige objecten

Een klassiek bewijs

Stelling:

Er zijn irrationale getallen p en q zodat p^q is rationaal.

$$\exists p, q \in \mathbb{R}\backslash\mathbb{Q}\,(\,p^q \in \mathbb{Q}\,)$$

Een klassiek bewijs

Stelling:

Er zijn irrationale getallen p en q zodat p^q is rationaal.

Bewijs:

$\sqrt{2}$ is irrationaal.

$\sqrt{2}^{\sqrt{2}}$ is rationaal of irrationaal.

- In het eerste geval zijn we klaar: $p = q = \sqrt{2}$

- In het tweede geval: $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^2 = 2$ is rationaal en we zijn klaar: $p = \sqrt{2}^{\sqrt{2}}$, $q = \sqrt{2}$

Een klassiek bewijs

Stelling:

Er zijn irrationale getallen p en q zodat p^q is rationaal.

Bewijs:

$\sqrt{2}$ is irrationaal.

$\sqrt{2}^{\sqrt{2}}$ is rationaal of irrationaal.

- In het eerste geval zijn we klaar: $p = q = \sqrt{2}$

- In het tweede geval: $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^2 = 2$ is rationaal en we zijn klaar: $p = \sqrt{2}^{\sqrt{2}}$, $q = \sqrt{2}$

Maar wat is nou p en wat is nou q?? Ik heb u geen echte p en q gegeven . . .

"Bestaan" p en q wel??

Waar gaat wiskunde over?

Grondslagen van de wiskunde:

- Formalisme: wiskunde is een formeel spel met formele regels. Betekenis? Alleen finitaire wiskunde heeft betekenis. Een theorie is goed als hij consistent is. Hilbert

Waar gaat wiskunde over?

Grondslagen van de wiskunde:

- Formalisme: wiskunde is een formeel spel met formele regels. Betekenis? Alleen finitaire wiskunde heeft betekenis. Een theorie is goed als hij consistent is. Hilbert
- Realisme: Maar wiskunde heeft toch ook met de werkelijkheid te maken? Ook infinitaire wiskunde!

Waar gaat wiskunde over?

Grondslagen van de wiskunde:

- Formalisme: wiskunde is een formeel spel met formele regels. Betekenis? Alleen finitaire wiskunde heeft betekenis. Een theorie is goed als hij consistent is. Hilbert
- Realisme: Maar wiskunde heeft toch ook met de werkelijkheid te maken? Ook infinitaire wiskunde!
- Platonisme: Ook de abstracte en infinitaire wiskundige objecten "bestaan".
- Logicisme: Logica is universeel; bouw de wiskunde daaruit op. Frege, Russell

Waar gaat wiskunde over?

Grondslagen van de wiskunde:

- Formalisme: wiskunde is een formeel spel met formele regels. Betekenis? Alleen finitaire wiskunde heeft betekenis. Een theorie is goed als hij consistent is. Hilbert
- Realisme: Maar wiskunde heeft toch ook met de werkelijkheid te maken? Ook infinitaire wiskunde!
- Platonisme: Ook de abstracte en infinitaire wiskundige objecten "bestaan".
- Logicisme: Logica is universeel; bouw de wiskunde daaruit op. Frege, Russell
- Intuïtionisme / Constructivisme: Alleen objecten die men kan construeren (in de tijd) bestaan. Brouwer

Brouwers Intuïtionisme

Wiskunde is primair en komt vòòr de logica. Logica is beschrijvend.

Basis intuïtie: constructie van objecten in de tijd: $\mathbb{N}$

Een bewijs (redenering)is ook een constructie (in de tijd).

Brouwers Intuïtionisme

Wiskunde is primair en komt vòòr de logica. Logica is beschrijvend.

Basis intuïtie: constructie van objecten in de tijd: $\mathbb{N}$

Een bewijs (redenering)is ook een constructie (in de tijd).

Wat kunnen we dan construeren? Welke redeneringen zijn nog geldig?

$\sqrt{2}^{\sqrt{2}}$ is rationaal OF irrationaal.

- Eerste geval: klaar; $p = q = \sqrt{2}$
- Tweede geval: $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = 2$ is rationaal en klaar: $p = \sqrt{2}^{\sqrt{2}}$, $q = \sqrt{2}$

Brouwers Intuïtionisme

Wiskunde is primair en komt vòòr de logica. Logica is beschrijvend.

Basis intuïtie: constructie van objecten in de tijd: $\mathbb{N}$

Een bewijs (redenering)is ook een constructie (in de tijd).

Wat kunnen we dan construeren? Welke redeneringen zijn nog geldig?

$\sqrt{2}^{\sqrt{2}}$ is rationaal OF irrationaal.

- Eerste geval: klaar; $p = q = \sqrt{2}$
- Tweede geval: $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = 2$ is rationaal en klaar: $p = \sqrt{2}^{\sqrt{2}}$, $q = \sqrt{2}$

Fout redeneerpatroon:

$$A \vee \neg A$$

"Wet van de uitgesloten derde" (Tertium non-datur):

Voor iedere uitspraak A geldt A of niet-A.

Wat is intuïtionistisch waar?

Intuïtionistisch is er minder waar dan klassiek.

(Calvinistische wiskunde . . .)

Maar het levert ook wat op! ... zullen we zien.

Welke stellingen zijn nog wel waar?

Wat is intuïtionistisch waar?

Intuïtionistisch is er minder waar dan klassiek.
(Calvinistische wiskunde . . .)
Maar het levert ook wat op! ... zullen we zien.

Welke stellingen zijn nog wel waar?
Brouwer: Een stelling is waar als er een bewijs voor is.

Dus de echte vraag is:

Wat is een bewijs?

Brouwer heeft dit zelf nooit formeel precies gemaakt, want Brouwer deed niet aan logica. Zijn student Heyting en Kolmogorov wel.

Brouwer-Heyting-Kolmogorov interpretatie (BHK)

Die formalen Regeln der intuitionistischen Logik.

Von Dr. A. Heyting
in Enschede (Niederlande).

(Vorgelegt von Hrn. Bieberbach am 19. Dezember 1929 [s. Jahrg. 1929 S. 686].)

Einleitung.

Die intuitionistische Mathematik ist eine Denktätigkeit, und jede Sprache, auch die formalistische, ist für sie nur Hilfsmittel zur Mitteilung. Es ist prinzipiell unmöglich, ein System von Formeln aufzustellen, das mit der intuitionistischen Mathematik gleichwertig wäre, denn die Möglichkeiten des Denkens lassen sich nicht auf eine endliche Zahl von im voraus aufstellbaren Regeln zurückführen. Der Versuch, die wichtigsten Teile der Mathematik in Formelsprache wiederzugeben[1], wird deshalb ausschließlich gerechtfertigt durch die größere Bündigkeit und Bestimmtheit der letzteren gegenüber der gewöhnlichen Sprache, Eigenschaften, welche sie geeignet machen, das Eindringen in die intuitionistischen Begriffe und ihre Verwendung bei Untersuchungen zu erleichtern.

Zum Aufbau der Mathematik ist die Aufstellung allgemeingültiger logischer Gesetze nicht notwendig; diese Gesetze werden in jedem einzelnen Fall gleichsam von neuem entdeckt als gültig für das eben betrachtete mathematische System. Die sprachliche Mitteilung aber, nach den Bedürfnissen des täglichen Lebens gebildet, schreitet in der Form der logischen Gesetze, welche sie als gegeben voraussetzt, fort. Eine Sprache, welche dem Gang der intuitionistischen Mathematik von Schritt zu Schritt nachgebildet wäre, würde so in allen Teilen von der gewohnten Form abweichen, daß sie die obengenannten günstigen Eigenschaften wieder gänzlich verlieren müßte. Diese Überlegungen haben mich dazu geführt, die Formalisierung der intuitionistischen Mathematik doch wieder mit einem Aussagenkalkül anzufangen.

Die Formeln des formalistischen Systems entstehen aus einer endlichen Zahl von Axiomen durch Anwendung einer endlichen Zahl von Operationsregeln. Sie enthalten außer den »konstanten« Zeichen auch Variabele. Das Verhältnis zwischen diesem System und der Mathematik ist nun dieses, daß bei einer bestimmten Interpretation der Konstanten und unter bestimmten Beschränkungen hinsichtlich der Ersetzung der Variabeln jede Formel einen richtigen mathematischen Satz darstellt. (Z. B. müssen die Variabeln im Aussagenkalkül nur durch sinnerfüllte mathematische Aussagen ersetzt werden.) Ist das System so beschaffen, daß es die letztgenannte Forderung erfüllt, so

[1] Diese Abhandlung bildet eine Umarbeitung des ersten Teiles einer von dem »Wiskundig Genootschap« in Amsterdam im Anfang 1928 gekrönten Preisschrift.

Brouwer-Heyting-Kolmogorov interpretatie (BHK)

Een bewijs van

$A \wedge B$ is een paar van een bewijs van A en een bewijs van B

$A \vee B$ is een bewijs van A of een bewijs van B

Brouwer-Heyting-Kolmogorov interpretatie (BHK)

Een bewijs van

$A \wedge B$	is een paar van een bewijs van A en een bewijs van B
$A \vee B$	is een bewijs van A of een bewijs van B
$A \to B$	is een methode om, gegeven een bewijs van A, een bewijs van B te maken
$\bot$	is er niet
$\forall x \in D(A(x))$	is een methode om, gegeven een element $d \in D$, een bewijs van $A(d)$ te maken
$\exists x \in D(A(x))$	is een paar bestaande uit een element $d \in D$ en een bewijs van $A(d)$.

Brouwer-Heyting-Kolmogorov interpretatie (BHK)

Een bewijs van

$A \wedge B$	is een paar van een bewijs van A en een bewijs van B
$A \vee B$	is een bewijs van A of een bewijs van B
$A \to B$	is een methode om, gegeven een bewijs van A, een bewijs van B te maken
$\bot$	is er niet
$\forall x \in D(A(x))$	is een methode om, gegeven een element $d \in D$, een bewijs van $A(d)$ te maken
$\exists x \in D(A(x))$	is een paar bestaande uit een element $d \in D$ en een bewijs van $A(d)$.

Dus: er is geen bewijs van $A \vee \neg A$

Brouwer-Heyting-Kolmogorov interpretatie (BHK)

Een bewijs van

$A \wedge B$	is een paar van een bewijs van A en een bewijs van B
$A \vee B$	is een bewijs van A of een bewijs van B
$A \to B$	is een methode om, gegeven een bewijs van A, een bewijs van B te maken
$\bot$	is er niet
$\forall x \in D(A(x))$	is een methode om, gegeven een element $d \in D$, een bewijs van $A(d)$ te maken
$\exists x \in D(A(x))$	is een paar bestaande uit een element $d \in D$ en een bewijs van $A(d)$.

Dus: een bewijs van $\forall x \in D \exists y \in E(A(x, y))$ bevat een methode om bij iedere $d \in D$ een $e \in E$ te maken zodat $A(d, e)$ geldt.

Kleene Realiseerbaarheid, Curry-Howard Formules als Types

We kunnen de BHK interpretatie nog formeler maken

- een formule opvatten als een type (of specificatie)
- een bewijs opvatten als een als een algoritme (programma)

Kleene Realiseerbaarheid, Curry-Howard Formules als Types

We kunnen de BHK interpretatie nog formeler maken

- een formule opvatten als een type (of specificatie)
- een bewijs opvatten als een als een algoritme (programma)

Kleene realiseerbaarheid

$$m \mathsf{r} A$$

"m realiseert de formule A" ($m \in \mathbb{N}$, gezien als code van een Turing machine)

Kleene Realiseerbaarheid, Curry-Howard Formules als Types

We kunnen de BHK interpretatie nog formeler maken

- een formule opvatten als een type (of specificatie)
- een bewijs opvatten als een als een algoritme (programma)

Kleene realiseerbaarheid

$$m \mathsf{r} A$$

"m realiseert de formule A" ($m \in \mathbb{N}$, gezien als code van een Turing machine)

Curry-Howard formules als types:

$$M : A$$

"M is van type A" (M een algoritme / functioneel programma / data)

Formules als Types, Bewijzen als termen / programma's

Een bewijs van (term van type)

$A \wedge B$	is een term $\langle p, q \rangle$ met $p : A$ en $q : B$
$A \vee B$	is $\mathsf{inl}\, p$ met $p : A$ of $\mathsf{inr}\, q$ met $q : B$
$A \to B$	is een term $f : A \to B$
$\bot$	is er niet
$\forall x \in D(A(x))$	is een term $f : \Pi_{x \in D} A(x)$
$\exists x \in D(A(x))$	is een term $\langle d, p \rangle$ met $d : D$ en $p : A(d)$

Formules als Types, Bewijzen als termen / programma's

Een bewijs van (term van type)

$A \wedge B$	is een term $\langle p, q \rangle$ met $p : A$ en $q : B$
$A \vee B$	is $\mathsf{inl}\, p$ met $p : A$ of $\mathsf{inr}\, q$ met $q : B$
$A \to B$	is een term $f : A \to B$
$\bot$	is er niet
$\forall x \in D(A(x))$	is een term $f : \Pi_{x \in D} A(x)$
$\exists x \in D(A(x))$	is een term $\langle d, p \rangle$ met $d : D$ en $p : A(d)$

Formules en verzamelingen zijn allebei (data)types

Bewijzen en elementen zijn allebei termen (data, programma's)

Twee "lezingen" voor $M : A$:

- M is een bewijs van de formule A
- M is data van het type A

Formules als Types, Bewijzen als termen / programma's

Een bewijs van (term van type)

$A \wedge B$	is een term $\langle p, q \rangle$ met $p : A$ en $q : B$
$A \vee B$	is $\mathsf{inl}\, p$ met $p : A$ of $\mathsf{inr}\, q$ met $q : B$
$A \to B$	is een term $f : A \to B$
$\bot$	is er niet
$\forall x \in D(A(x))$	is een term $f : \Pi_{x \in D} A(x)$
$\exists x \in D(A(x))$	is een term $\langle d, p \rangle$ met $d : D$ en $p : A(d)$

Formules en verzamelingen zijn allebei (data)types

Bewijzen en elementen zijn allebei termen (data, programma's)

Twee "lezingen" voor $M : A$:

- M is een bewijs van de formule A
- M is data van het type A

Formules als Types, Bewijzen als termen in de informatica (1)

Bewijs Checken

$$\text{Bewijs checken} = \text{Type checken}$$

Er is een "type check" algoritme TC:

$$\text{TC}(p) \mapsto A \text{ als } p : A \text{ of fail als } p \text{ niet typeerbaar}$$

Bewijs zoeken (Theorem Proving) =
interactief zoeken naar een term p zodat $p : A$.

Formules als Types, Bewijzen als termen in de informatica (2)

Programmeren met bewijzen

We onderscheiden logische typen en data typen

$$
\begin{array}{rcl}
A \wedge B & \simeq & A \times B \\
A \vee B & \simeq & A \uplus B \\
A \to B & \simeq & A \to B \\
\bot & \simeq & \emptyset \\
\forall x{:}D(A(x)) & \simeq & \Pi_{x:D} A(x) = \{f \mid \forall d{:}D(f(d){:}A(d))\} \\
\exists x{:}D(A(x)) & \simeq & \Sigma x{:}D(A(x)) = \{\langle d, p\rangle \mid d{:}D \;\&\; f(d){:}A(d)\}
\end{array}
$$

Formules als Types, Bewijzen als termen in de informatica (2)

Programmeren met bewijzen

We onderscheiden logische typen en data typen

$$
\begin{array}{rcl}
A \wedge B & \simeq & A \times B \\
A \vee B & \simeq & A \uplus B \\
A \to B & \simeq & A \to B \\
\bot & \simeq & \emptyset \\
\forall x{:}D(A(x)) & \simeq & \Pi_{x:D} A(x) = \{f \mid \forall d{:}D(f(d){:}A(d))\} \\
\exists x{:}D(A(x)) & \simeq & \Sigma x{:}D(A(x)) = \{\langle d, p\rangle \mid d{:}D \;\&\; f(d){:}A(d)\}
\end{array}
$$

Deze typen gedragen zich hetzelfde, maar we gebruiken ze anders:

- data typen voor computationele informatie (berekening)
- logische typen voor specificatie informatie (correctheid)

Programmeren met constructieve bewijzen

Voorbeeld: een lijst van natuurlijke getallen sorteren

$$\mathsf{sort} : \mathsf{List}_{\mathbb{N}} \to \mathsf{List}_{\mathbb{N}}$$

Programmeren met constructieve bewijzen

Voorbeeld: een lijst van natuurlijke getallen sorteren

$$\mathsf{sort} : \mathsf{List}_{\mathbb{N}} \to \mathsf{List}_{\mathbb{N}}$$

Verfijndere specificatie (output is gesorteerd):

$$\mathsf{sort} : \mathsf{List}_{\mathbb{N}} \to \exists y{:}\mathsf{List}_{\mathbb{N}}(\mathsf{Sorted}(y))$$

$\mathsf{Sorted}(x) := \forall i < \mathsf{length}(x) - 1(x[i] \leq x[i+1])$

Programmeren met constructieve bewijzen

Voorbeeld: een lijst van natuurlijke getallen sorteren

$$\mathsf{sort} : \mathsf{List}_{\mathbb{N}} \longrightarrow \mathsf{List}_{\mathbb{N}}$$

Verfijndere specificatie (output is gesorteerd):

$$\mathsf{sort} : \mathsf{List}_{\mathbb{N}} \longrightarrow \exists y{:}\mathsf{List}_{\mathbb{N}}(\mathsf{Sorted}(y))$$

Nog verfijndere specificatie (output is permutatie van input):

$$\mathsf{sort} : \forall x{:}\mathsf{List}_{\mathbb{N}}\, \exists y{:}\mathsf{List}_{\mathbb{N}}(\mathsf{Sorted}(y) \wedge \mathsf{Permutation}(x, y))$$

Het bewijs sort bevat een sorteringsalgoritme.

Programmeren met constructieve bewijzen

Extractie van programma's uit bewijzen.

$$\mathsf{sort} : \forall x{:}\mathsf{List}_{\mathbb{N}}\, \exists y{:}\mathsf{List}_{\mathbb{N}}(\mathsf{Sorted}(y) \wedge \mathsf{Permutation}(x, y))$$

Programmeren met constructieve bewijzen

Extractie van programma's uit bewijzen.

$$\mathsf{sort} : \forall x{:}\mathsf{List}_{\mathbb{N}}\, \exists y{:}\mathsf{List}_{\mathbb{N}}(\mathsf{Sorted}(y) \wedge \mathsf{Permutation}(x, y))$$

Met onderscheid data-logica:

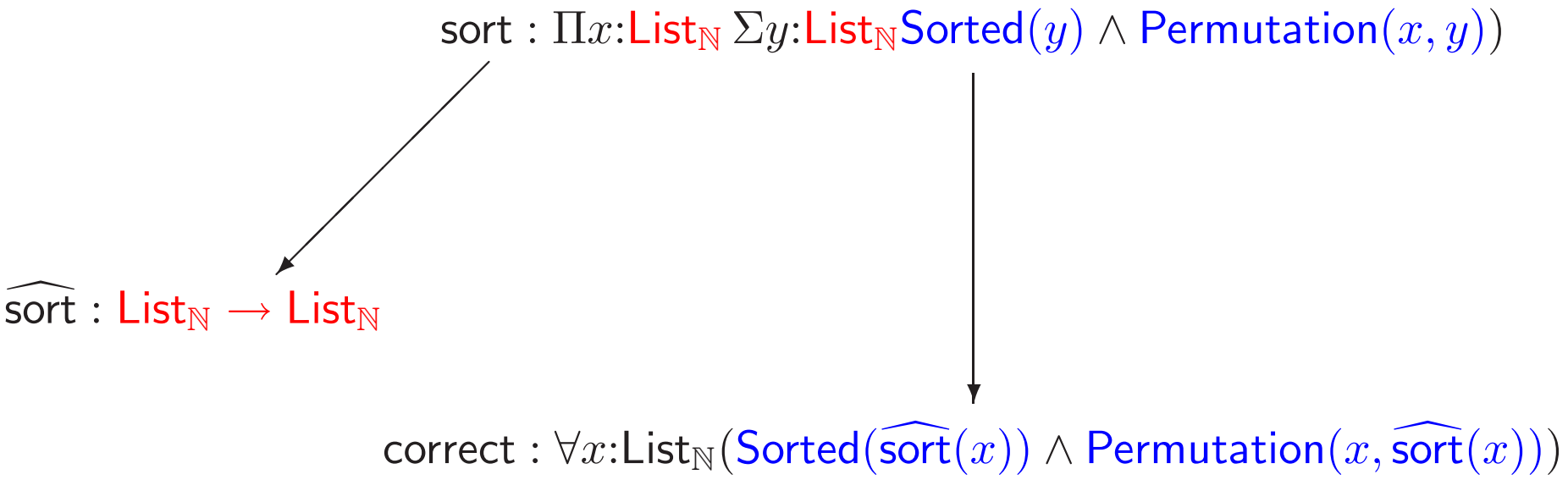
$$\mathsf{sort} : \overbrace{\Pi x{:}\mathsf{List}_{\mathbb{N}}\, \Sigma y{:}\mathsf{List}_{\mathbb{N}}}^{\text{berekening}} \underbrace{(\mathsf{Sorted}(y) \wedge \mathsf{Permutation}(x, y))}_{\text{specificatie}}$$

Programmeren met constructieve bewijzen

Extractie van programma's uit bewijzen.

$$\text{sort} : \forall x{:}\text{List}_{\mathbb{N}}\, \exists y{:}\text{List}_{\mathbb{N}}(\text{Sorted}(y) \wedge \text{Permutation}(x, y))$$

Met onderscheid data-logica en programma extractie:



Rekenen met reële getallen

Wat is een reëel getal?

- Een rationaal getal $\frac{p}{q}$ is een reëel getal.
- Een limiet van een rij rationale getallen is een reëel getal

Bijvoorbeeld

$$\begin{aligned}
\sum_{i=0}^{\infty} \frac{1}{i!} = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{i!} &= e \\
\lim_{n\to\infty} (1 + \frac{1}{n})^n &= e \\
\sum_{i=0}^{\infty} \frac{1}{i^2} = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{i^2} &= \frac{\pi^2}{6}
\end{aligned}$$

$\Rightarrow$ Een reëel getal is (in het algemeen) een oneindig object

Rij rationale getallen $q_0, q_1, q_2, \ldots$ die convergeert.

Wat is een reëel getal?

Rij rationale getallen $q_0, q_1, q_2, \ldots$ die convergeert.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \tfrac{1}{k})$$

$$\forall \varepsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \varepsilon)$$

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p (|q_{N+m} - q_{N+p}| < \tfrac{1}{k})$$

Wat is een reëel getal?

Rij rationale getallen $q_0, q_1, q_2, \ldots$ die convergeert.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \tfrac{1}{k})$$
$$\forall \varepsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \varepsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p (|q_{N+m} - q_{N+p}| < \tfrac{1}{k})$$

Met N uitgedrukt in k als $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m, p (|q_{\beta(k)+m} - q_{\beta(k)+p}| < \frac{1}{k})$$

Fundamentele rij: paar bestaande uit $(q_i)_{i \in \mathbb{N}}$ en $\beta : \mathbb{N} \to \mathbb{N}$.

Wat is een reëel getal?

Rij rationale getallen $q_0, q_1, q_2, \ldots$ die convergeert.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \tfrac{1}{k})$$
$$\forall \varepsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \varepsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p (|q_{N+m} - q_{N+p}| < \tfrac{1}{k})$$

Met N uitgedrukt in k als $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m, p (|q_{\beta(k)+m} - q_{\beta(k)+p}| < \frac{1}{k})$$

Fundamentele rij: paar bestaande uit $(q_i)_{i \in \mathbb{N}}$ en $\beta : \mathbb{N} \to \mathbb{N}$.
Voorbeeld: decimale notatie

$$3,\ 3.1,\ 3.14,\ 3.141,\ 3.1415,\ 3.14159,\ \ldots$$

Rekenen op een computer met reële getallen(1)

Floating point?

Een getal is een "mantissa" + de positie van de decimale punt.

Bijvoorbeeld:

$$(1.20 \times 10^{-1}) \times (1.20 \times 10^{-1}) = (1.44 \times 10^{-2})$$

Rekenen op een computer met reële getallen(1)

Floating point?

Een getal is een "mantissa" + de positie van de decimale punt.

Bijvoorbeeld:

$$(1.20 \times 10^{-1}) \times (1.20 \times 10^{-1}) = (1.44 \times 10^{-2})$$

Probleem: Operaties worden afgerond; nauwkeurigheid van het antwoord wordt niet bijgehouden.

$$a_0 := \frac{11}{2}, a_1 := \frac{61}{11}, a_{n+1} := 111 - \frac{1130 - \frac{3000}{a_{n-1}}}{a_n}$$

n	2	5	6	7	8	10	11	12
a_n	5.6	5.6	4.3	-29.0	125.7	100.1	100.0	100.0

Rekenen op een computer met reële getallen(2)

Interval rekenkunde (of Floating points met "precisie")?

Een getal is een interval.

Bijvoorbeeld:

$$\begin{aligned} [3,6] + [0.5,1] &= [3.5,7] \\ [3,6] - [0.5,1] &= [2,5.5] \end{aligned}$$

Rekenen op een computer met reële getallen(2)

Interval rekenkunde (of Floating points met "precisie")?

Een getal is een interval.

Bijvoorbeeld:

$$\begin{aligned} [3,6] + [0.5,1] &= [3.5,7] \\ [3,6] - [0.5,1] &= [2,5.5] \end{aligned}$$

Wel precies, maar het nauwkeurigheidsinterval wordt snel (te) groot.

$$[3,6] - [3,6] = [-3,3] \;\; ??$$

Rekenen op een computer met reële getallen(3)

Exacte reële rekenkunde

Een getal x is een methode om een willekeurig goede benadering van x te krijgen.

Meestal weergegeven door een oneindige rij "digits" of rationale getallen.

Bijvoorbeeld

$$[q_0^l, q_0^r], [q_1^l, q_1^r], [q_2^l, q_2^r], \qquad \ldots \text{ met } q_i^l < q_i^r \text{ en } |q_i^r - q_i^l| < \frac{1}{i}$$

Rekenen op een computer met reële getallen(3)

Exacte reële rekenkunde

Een getal x is een methode om een willekeurig goede benadering van x te krijgen.

Meestal weergegeven door een oneindige rij "digits" of rationale getallen.

Bijvoorbeeld

$[q_0^l, q_0^r], [q_1^l, q_1^r], [q_2^l, q_2^r], \quad \ldots$ met $q_i^l < q_i^r$ en $|q_i^r - q_i^l| < \frac{1}{i}$

$3,\ 3.1,\ 3.14,\ 3.141, \ldots$ decimale ontwikkeling

$a_0, a_1, a_2, a_3, \ldots$ $a_0 \in \mathbb{Z}$, alle andere $a_i \in \mathbb{N}^+$ kettingbreuken

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}}$$

Exacte reële rekenkunde

Functies over exacte reële getallen.

F : benadering van de input $\mapsto$ benadering van de output

Exacte reële rekenkunde

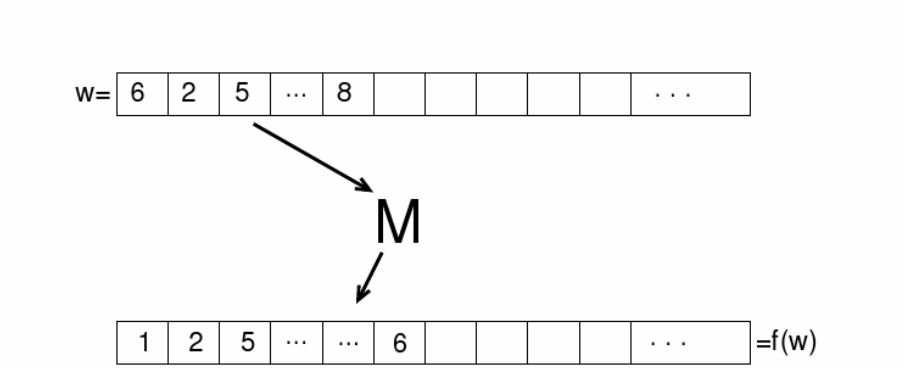
Functies over exacte reële getallen.

F : benadering van de input $\mapsto$ benadering van de output

Maar daar nog bij:

F : gewenste precisie van de output $\mapsto$ benodigde benadering van de input

Exacte reële rekenkunde



Exacte reële rekenkunde

Voorbeeld: optellen op decimale getallen.

x	y	$x+y$
0	0	$-$
$0,1$	$0,2$	$0,$
$0,12$	$0,23$	$0,3$
$0,124$	$0,235$	$0,3$

Exacte reële rekenkunde

Voorbeeld: optellen op decimale getallen.

x	y	$x+y$
0	0	$-$
$0,1$	$0,2$	$0,$
$0,12$	$0,23$	$0,3$
$0,124$	$0,235$	$0,3$
$0,1241$	$0,2351$	$0,359$
$\vdots$	$\vdots$	$\vdots$

Werkt dit altijd?

Vraag: Kun je altijd door een betere benadering van de input meer output leveren?

Exacte reële rekenkunde

Voorbeeld: optellen op decimale getallen.

x	y	$x+y$
0	0	$-$
$0,1$	$0,2$	$0,$
$0,12$	$0,23$	$0,3$
$0,124$	$0,235$	$0,3$
$0,1241$	$0,2351$	$0,359$
$\vdots$	$\vdots$	$\vdots$

Werkt dit altijd?

Vraag: Kun je altijd door een betere benadering van de input meer output leveren? Nee ...

Exacte reële rekenkunde

Voorbeeld: optellen op decimale getallen.

x	y	$x+y$
0	0	$-$
$0,3$	$0,6$	$-$
$0,33$	$0,66$	$-$
$0,333$	$0,666$	$-$
$\vdots$	$\vdots$	geen output

Een 4 bij x? Dan $1,\ldots$ Een 2 bij x? Dan $0,\ldots$

Exacte reële rekenkunde

Voorbeeld: optellen op decimale getallen.

x	y	$x+y$
0	0	–
$0,3$	$0,6$	–
$0,33$	$0,66$	–
$0,333$	$0,666$	–
$\vdots$	$\vdots$	geen output

Een 4 bij x? Dan $1,\ldots$ Een 2 bij x? Dan $0,\ldots$

De decimale representatie is niet geschikt!

Brouwer wist dit al. "Besitzt jede reelle Zahl eine Dezimalbruchentwikklung?" (Mathematische Annalen, 1921)

Exacte reële rekenkunde: getallensystemen

Een "goed" getallen systeem heeft veel redundantie

Decimale getallen:

$$\ldots \mid \quad 0.0 \quad \mid \quad 0.1 \quad \mid \quad 0.2 \quad \mid \ldots$$

0.0 beschrijft $[0.0, 0.1]$ en 0.1 beschrijft $[0.1, 0.2]$.

Wat als we 0.1 blijken te willen benaderen?

Exacte reële rekenkunde: getallensystemen

Een "goed" getallen systeem heeft veel redundantie

Decimale getallen:

$$\ldots \mid \quad 0.0 \quad \mid \quad 0.1 \quad \mid \quad 0.2 \quad \mid \ldots$$

0.0 beschrijft $[0.0, 0.1]$ en 0.1 beschrijft $[0.1, 0.2]$.

Wat als we 0.1 blijken te willen benaderen?

Oplossing: Voeg één extra digit toe: -1

$$\begin{array}{ccccccccc} \ldots & & [& 0.0 &] & & [& 0.2 &] & & \ldots \\ \hline \ldots & & & & [& 0.1 &] & & [& 0.3 &] \ldots \end{array}$$

Exacte reële rekenkunde: getallensystemen

Een "goed" getallen systeem heeft veel redundantie

Decimale getallen:

$$\ldots \mid \quad 0.0 \quad \mid \quad 0.1 \quad \mid \quad 0.2 \quad \mid \ldots$$

0.0 beschrijft $[0.0, 0.1]$ en 0.1 beschrijft $[0.1, 0.2]$.

Wat als we 0.1 blijken te willen benaderen?

Oplossing: Voeg één extra digit toe: -1

$$\begin{array}{ccccccc} \ldots & & [\quad 0.0 \quad] & & [\quad 0.2 \quad] & & \ldots \\ \hline \ldots & & & [\quad 0.1 \quad] & & [\quad 0.3 \quad] & \ldots \end{array}$$

Stelling: Exacte reële rekenkunde kan alleen met een redundant representatiesysteem.

Exacte reële rekenkunde

Probleem (?): "$x \leq y$" kun je niet (algoritmisch) beslissen.

Klassiek: $x \leq y \vee y < x$

Exacte reële rekenkunde

Probleem (?): "$x \leq y$" kun je niet (algoritmisch) beslissen.

$$\text{Klassiek: } x \leq y \vee y < x$$

Brouwer: De wet $x \leq y \vee y < x$ geldt niet voor de reële getallen.
Wat hebben we dan wel?

$$x \leq_\varepsilon y \quad \vee \quad y <_\varepsilon x$$

$$x \leq y + \varepsilon \quad \vee \quad y - \varepsilon < x$$

$$\ldots \qquad [y - \varepsilon \qquad\qquad y + \varepsilon] \qquad \ldots$$

Exacte reële rekenkunde

Probleem (?): "$x \leq y$" kun je niet (algoritmisch) beslissen.

Klassiek: $x \leq y \vee y < x$

Brouwer: De wet $x \leq y \vee y < x$ geldt niet voor de reële getallen.
Wat hebben we dan wel?

$$x \leq_\varepsilon y \quad \vee \quad y <_\varepsilon x$$

$$x \leq y + \varepsilon \quad \vee \quad y - \varepsilon < x$$

$$\ldots \qquad [y - \varepsilon \qquad\qquad y + \varepsilon] \qquad \ldots$$

$$x < y \to x < z \vee z < y$$

$$\ldots \qquad [x \qquad\qquad y] \qquad \ldots$$

Exacte reële rekenkunde

Is dit niet problematisch in de praktijk?

Exacte reële rekenkunde

Is dit niet problematisch in de praktijk?

Nee: fysische variabelen (metingen) zijn ook altijd "bij benadering"

Specificaties die berusten op precieze gelijkheids-checks zijn niet robuust.

Exacte reële rekenkunde

Is dit niet problematisch in de praktijk?

Nee: fysische variabelen (metingen) zijn ook altijd "bij benadering"

Specificaties die berusten op precieze gelijkheids-checks zijn niet robuust.

Toestand ON $T' = 2$	$\overset{T=22}{\longrightarrow}$ $\overset{T=18}{\longleftarrow}$	Toestand OFF $T' = -T$

EINDE