Le nozze di Giustizia

Interactions between Artificial Intelligence, Logic, Law, Language and Computation

Joost J. Joosten

Monday, September 18, 2023

Many thanks to the organisers

- A big responsability

Many thanks to the organisers

- A big responsability
- Link of my talk to

may go through Article 15 on Robustness

Many thanks to the organisers

- A big responsability
- Link of my talk to

  EU Proposal for a Regulation of Artificial Intelligence

  may go through Article 15 on Robustness

Many thanks to the organisers

- A big responsability
- Link of my talk to

  EU Proposal for a Regulation of Artificial Intelligence

  may go through Article 15 on Robustness
- Which, we plea, needs formal methods

Many thanks to the organisers

- A big responsability
- Link of my talk to

  EU Proposal for a Regulation of Artificial Intelligence

  may go through Article 15 on Robustness
- Which, we plea, needs formal methods

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
  - Thinking correctly;

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
  - Thinking correctly;
  - Predictive power;

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
  - Thinking correctly;
  - Predictive power;
  - Language;

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
  - Thinking correctly;
  - Predictive power;
  - Language;
  - Adaptive to environment;

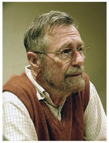
What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
  - Thinking correctly;
  - Predictive power;
  - Language;
  - Adaptive to environment;
  - Computation;

What is intelligence

- Boring answer: Intelligence is what intelligence tests measure
- 1923, Edwin Boring, pioneer of cognitive psychology
- Curious aside: ChatGPT scored very high on IQ tests ("verbal IQ"of 155, Eka Rovainen, Scientific American, 2023)
- Honest answer: No clear formal definition
- Related aspects
    - Thinking correctly;
    - Predictive power;
    - Language;
    - Adaptive to environment;
    - Computation;
    - Adequate responses;

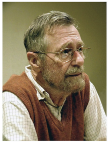
Then, what is Artificial



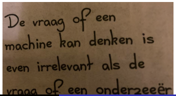
- External to humans

- Alan Turing opened his 1950 paper *Computing machinery and intelligence* with the question:
  Can machines think?
- Edsger Wybe Dijkstra: The question whether a machine can think is as relevant as the question if a submarine can swim.

Then, what is Artificial



- External to humans
- EWD:



De vraag of een
machine kan denken is
even irrelevant als de
vraag of een onderzeeër

- Alan Turing opened his 1950 paper *Computing machinery and intelligence* with the question:

  Can machines think?
- Edsger Wybe Dijkstra: The question whether a machine can think is as relevant as the question if a submarine can swim.

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);
  - Etc;

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);
  - Etc;
- Distinction: rule-based versus data-based (Hildebrandt)

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);
  - Etc;
- Distinction: rule-based versus data-based (Hildebrandt)
- For rule-based AI there is some hope for objective transparency

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);
  - Etc;
- Distinction: rule-based versus data-based (Hildebrandt)
- For rule-based AI there is some hope for objective transparency
- For machine-learning: forget it

Putting it together: Artificial Intelligence

- I am not impressed, that's not AI...
- Various processes:
  - Machine learning;
  - Data querying;
  - Automated search;
  - Computations (big data);
  - Etc;
- Distinction: rule-based versus data-based (Hildebrandt)
- For rule-based AI there is some hope for objective transparency
- For machine-learning: forget it
- Unless we are happy with ChatGPT-like answers and dialogues as being authoritative (which is not entirely ridiculous)

Should we use it?

- We use it...

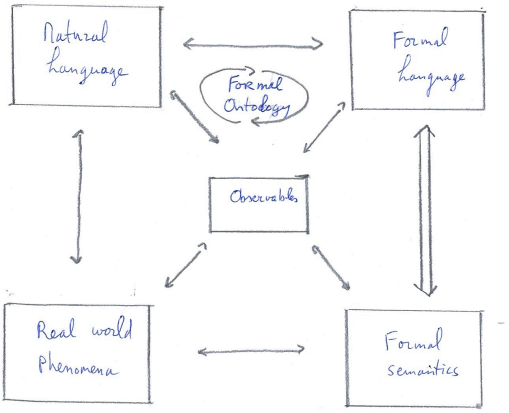
Should we use it?

- We use it...
- St. Augustine vs Ibn Taymiyya, Al Ghazali and, Nizam al-Mulk

Should we use it?

- We use it...
- St. Augustine vs Ibn Taymiyya, Al Ghazali and, Nizam al-Mulk
-

- Formal versus real: Formal Language, Formal Semantics, Informal phenomenon (language and semantics)
- Limited scope (no discretionary powers, clear ontologies)



Formal methods I have been involved with

1. Logical, mathematical and algorithmic analysis of legal texts

   (please do share potential computable laws with me)

Formal methods I have been involved with

1. Logical, mathematical and algorithmic analysis of legal texts

   (please do share potential computable laws with me)

2. Proving program/algorithm correctness using proof assistants

Formal methods I have been involved with

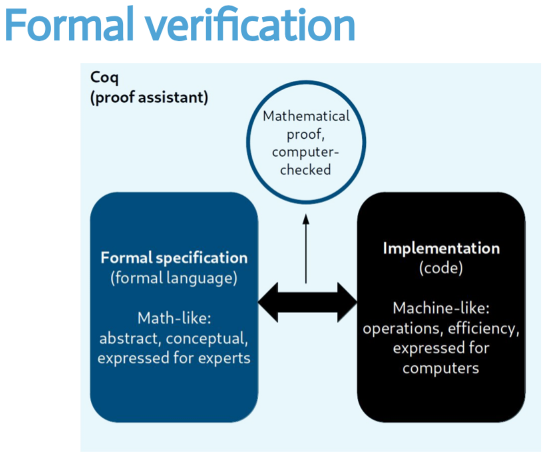
1. Logical, mathematical and algorithmic analysis of legal texts

   (please do share potential computable laws with me)

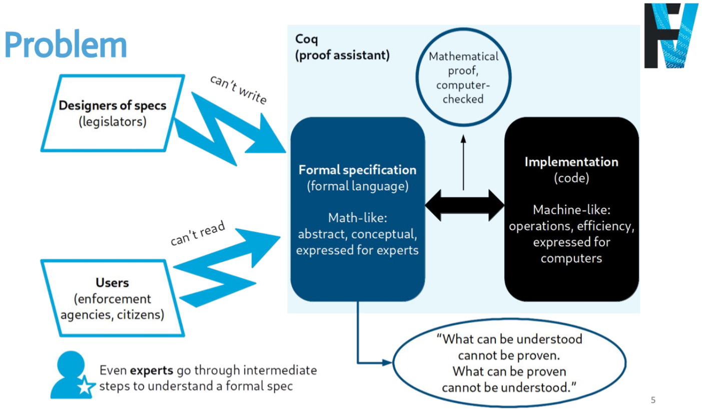
2. Proving program/algorithm correctness using proof assistants

3. Model checking

Program correctness



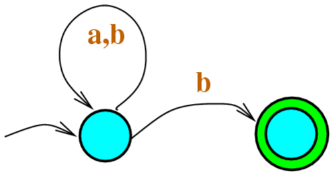
Formal and technical specification


Problem
Designers of specs
(legislators)
can't write
Users
(enforcement agencies, citizens)
can't read
Even experts go through intermediate steps to understand a formal spec
Coq
(proof assistant)
Mathematical proof, computer-checked
Formal specification
(formal language)
Math-like: abstract, conceptual, expressed for experts
Implementation
(code)
Machine-like: operations, efficiency, expressed for computers
"What can be understood cannot be proven. What can be proven cannot be understood."
5

Model checking

You must end your day with activity *b*:

Finite automata



$\Sigma = \{a,b\}$

$L = (a+b)^*b$

NFA

Limitations

- AI is often presented as an potential omnipotent salvation

Limitations

- AI is often presented as an potential omnipotent salvation
- Logic imposes severe restrictions

Limitations

- AI is often presented as an potential omnipotent salvation
- Logic imposes severe restrictions
- Undecidability: impossibility to compute by whatever means

Limitations

- AI is often presented as an potential omnipotent salvation
- Logic imposes severe restrictions
- Undecidability: impossibility to compute by whatever means
- First and most famous example: Halting Problem

Simple halting program

- Given a program Π, does it hold on input A?

Simple halting program

- Given a program Π, does it hold on input A?
- For example

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):
  - $\$x = 3$;

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):
  - $\$x = 3$;
  - $\$x = 2$;

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):
  - $\$x = 3$;
  - $\$x = 2$;
  - $\$x = 1$;

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):
  - $\$x = 3$;
  - $\$x = 2$;
  - $\$x = 1$;
  - $\$x = 0$;

Simple halting program

- Given a program Π, does it hold on input A?
- For example
- Π does the following:
  - It reads an input number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** it is larger than 0, **then** it will makes the value in the register x one smaller (If $\$x > 0$, then $\$x := \$x - 1$)

    otherwise (that is, if $\$x = 0$) it will HALT.
- A run of Π(3):
  - $\$x = 3$;
  - $\$x = 2$;
  - $\$x = 1$;
  - $\$x = 0$;
  - HALT

Simple non-halting program

- Can you device a program Π, that does not halt on input A?

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.
- A run of Π(0):

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger
    (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.
- A run of Π(0):
  - $\$x = 0$;

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.
- A run of Π(0):
  - $\$x = 0$;
  - $\$x = 1$;

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.
- A run of Π(0):
  - $\$x = 0$;
  - $\$x = 1$;
  - $\$x = 2$;

Simple non-halting program

- Can you device a program Π, that does not halt on input A?
- For example
- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It checks $\$x$ –the value in the register x– and **if** $\$x = \$x$, **then** it will makes the value in the register x one bigger
    (If $\$x = \$x$, then $\$x := \$x + 1$)

    otherwise (that is, if $\$x \neq \$x$) it will HALT.
- A run of $\Pi(0)$:
  - $\$x = 0$;
  - $\$x = 1$;
  - $\$x = 2$;
  - $\vdots$

More complicated program (Collatz conjecture)

- Π does the following:

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with \$$x = 3$;)

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with $\$x = 3$;)
  - $\$x = 10$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with $\$x = 3$;)
  - $\$x = 10$;
  - $\$x = 5$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of $\Pi(3)$: (starting with $\$x = 3$;)
  - $\$x = 10$;
  - $\$x = 5$;
  - $\$x = 16$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with \$$x = 3$;)
  - \$$x = 10$;
  - \$$x = 5$;
  - \$$x = 16$;
  - \$$x = 8$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with $\$x = 3$;)
  - $\$x = 10$;
  - $\$x = 5$;
  - $\$x = 16$;
  - $\$x = 8$;
  - $\$x = 4$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of $\Pi(3)$: (starting with $\$x = 3$;)
  - $\$x = 10$;
  - $\$x = 5$;
  - $\$x = 16$;
  - $\$x = 8$;
  - $\$x = 4$;
  - $\$x = 2$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of $\Pi(3)$: (starting with $\$x = 3$;)
  - $\$x = 10$;
  - $\$x = 5$;
  - $\$x = 16$;
  - $\$x = 8$;
  - $\$x = 4$;
  - $\$x = 2$;
  - $\$x = 1$;

More complicated program (Collatz conjecture)

- Π does the following:
  - It reads an input natural number and stores it at the register x;
  - It repeats the following:
    If the number is even, divide it by two.
    If the number is odd, triple it and add one.
  - And only HALT when we reach one.
- A run of Π(3): (starting with $\$x = 3;$)
  - $\$x = 10;$
  - $\$x = 5;$
  - $\$x = 16;$
  - $\$x = 8;$
  - $\$x = 4;$
  - $\$x = 2;$
  - $\$x = 1;$
  - HALT.

Reductio ad absurdum

- Suppose there existed a program $H(X, Y)$ with the property

$$H(X, Y) = \begin{cases} 1 & \text{if } X(Y)\downarrow \\ 0 & \text{if } X(Y)\uparrow \end{cases}$$

Reductio ad absurdum

- Suppose there existed a program $H(X, Y)$ with the property

$$H(X, Y) = \begin{cases} 1 & \text{if } X(Y) \downarrow \\ 0 & \text{if } X(Y) \uparrow \end{cases}$$

- Using this alleged H we can tweak it to get $\tilde{H}(X)$ with

$$\tilde{H}(X) = \begin{cases} 0 & \text{if } X(X) \uparrow \\ \uparrow & \text{if } X(X) \downarrow \end{cases}$$

Reductio ad absurdum

- Suppose there existed a program $H(X, Y)$ with the property

$$H(X, Y) = \begin{cases} 1 & \text{if } X(Y)\downarrow \\ 0 & \text{if } X(Y)\uparrow \end{cases}$$

- Using this alleged H we can tweak it to get $\tilde{H}(X)$ with

$$\tilde{H}(X) = \begin{cases} 0 & \text{if } X(X)\uparrow \\ \uparrow & \text{if } X(X)\downarrow \end{cases}$$

- But now $\tilde{H}(\tilde{H})\downarrow \iff \tilde{H}(\tilde{H}) = 0 \iff \tilde{H}(\tilde{H}) =\uparrow$

Reductio ad absurdum

- Suppose there existed a program $H(X, Y)$ with the property

$$H(X, Y) = \begin{cases} 1 & \text{if } X(Y)\downarrow \\ 0 & \text{if } X(Y)\uparrow \end{cases}$$

- Using this alleged H we can tweak it to get $\tilde{H}(X)$ with

$$\tilde{H}(X) = \begin{cases} 0 & \text{if } X(X)\uparrow \\ \uparrow & \text{if } X(X)\downarrow \end{cases}$$

- But now $\tilde{H}(\tilde{H})\downarrow \iff \tilde{H}(\tilde{H}) = 0 \iff \tilde{H}(\tilde{H}) = \uparrow$
- $\lightning$

Devastating consequences of Turing

- There exists no infallible virus scanner

Devastating consequences of Turing

- There exists no infallible virus scanner
- with or without AI

Devastating consequences of Turing

- There exists no infallible virus scanner
- with or without AI
- Unrestricted program verification is impossible

Devastating consequences of Turing

- There exists no infallible virus scanner
- with or without AI
- Unrestricted program verification is impossible
- with or without AI

Devastating consequences of Turing

- There exists no infallible virus scanner
- with or without AI
- Unrestricted program verification is impossible
- with or without AI
- etc.

Devastating consequences of Turing

- There exists no infallible virus scanner
- with or without AI
- Unrestricted program verification is impossible
- with or without AI
- etc.
- Many problems are just not solvable in the mere virtue of logic (with our without AI)

Just divorce

- Imagine a *Just divorce* law that say:

Just divorce

- Imagine a *Just divorce* law that say:

  Upon divorce, each item in the common patrimony should be assigned a € amount and then the patrimony should be split between the partners in such a way that causes the minimal € difference in the division of goods between the partners

Infeasible problems

- If the partners have 1000 items in the common patrimony

Infeasible problems

- If the partners have 1000 items in the common patrimony
- The legal authority/supervisor would have perform many checks

$$2^{1000} =$$

10715086071862673209484250490600018105614048117055336
07443750388370351051124936122493198378815695858127594
67291755314682518714528569231404359845775746985748039
34567774824230985421074605062371141877954182153046474
98358194126739876755916554394607706291457119647768654
2167660429831652624386837205668069376

Infeasible problems

- If the partners have 1000 items in the common patrimony
- The legal authority/supervisor would have perform many checks

$$2^{1000} =$$

10715086071862673209484250490600018105614048117055336
07443750388370351051124936122493198378815695858127594
67291755314682518714528569231404359845775746985748039
34567774824230985421074605062371141877954182153046474
98358194126739876755916554394607706291457119647768654
2167660429831652624386837205668069376

- With the best supercomputers, this would take more than the estimated age of the universe

Infeasible problems

- If the partners have 1000 items in the common patrimony
- The legal authority/supervisor would have perform many checks

$$2^{1000} =$$

$$\begin{gathered}1071508607186267320948425049060001810561404811705 5336\\ 0744375038837035105112493612249319837881569585812759 4\\ 6729175531468251871452856923140435984577574698574803 9\\ 3456777482423098542107460506237114187795418215304647 4\\ 9835819412673987675591655439460770629145711964776865 4\\ 2167660429831652624386837205668069376\end{gathered}$$

- With the best supercomputers, this would take more than the estimated age of the universe
- not a practical law

Computable laws should be feasible

- There are tons of problems that are decidable (unlike the Halting problem)

Computable laws should be feasible

- There are tons of problems that are decidable (unlike the Halting problem)
- but are unfeasible

Computable laws should be feasible

- There are tons of problems that are decidable (unlike the Halting problem)
- but are unfeasible
- Legal stipulations and computable laws should avoid those

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour
- Due to mathematical properties implied by the regulations

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour
- Due to mathematical properties implied by the regulations
- Properties the legislators were probably simply not aware of

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour
- Due to mathematical properties implied by the regulations
- Properties the legislators were probably simply not aware of
- All come from the same regulation:

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour
- Due to mathematical properties implied by the regulations
- Properties the legislators were probably simply not aware of
- All come from the same regulation:

  European transport Regulation 561/2006 *European Parliament and Council of the European Union. Regulation (EC) no 561/2006 of the European Parliament and of the Council of 15 march 2006 on the harmonisation of certain social legislation relating to road transport. Official Journal of the European Union, 2006.*

Bad mathematical behaviour

- We will now see some examples where a real law ignites bad/undesired behaviour
- Due to mathematical properties implied by the regulations
- Properties the legislators were probably simply not aware of
- All come from the same regulation:

  European transport Regulation 561/2006 *European Parliament and Council of the European Union. Regulation (EC) no 561/2006 of the European Parliament and of the Council of 15 march 2006 on the harmonisation of certain social legislation relating to road transport. Official Journal of the European Union, 2006.*
- Our team is looking for new similar laws: please do share suggestions with me

Second labelling to minute labelling

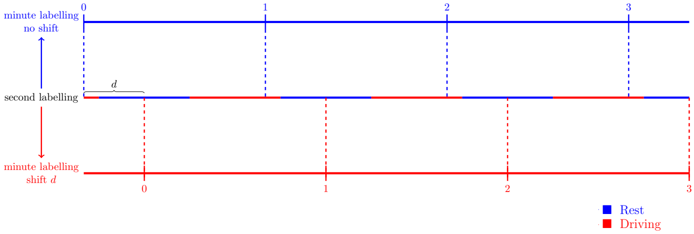
- Tachographs record activities by the second (Regulation (EU) 2016/799)

Second labelling to minute labelling

- Tachographs record activities by the second (Regulation (EU) 2016/799)
- Regulation (EU) 2016/799 tells how to transform second activities to an activity list organised by the minute

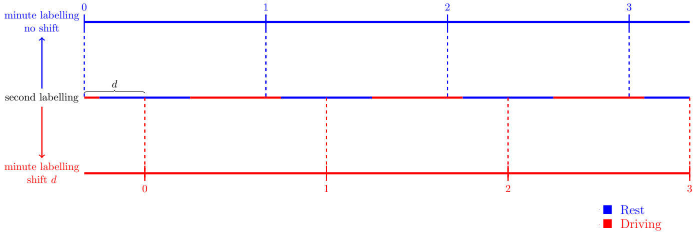
Second labelling to minute labelling

- Tachographs record activities by the second (Regulation (EU) 2016/799)
- Regulation (EU) 2016/799 tells how to transform second activities to an activity list organised by the minute
- Not shift invariant:



Second labelling to minute labelling

- Tachographs record activities by the second (Regulation (EU) 2016/799)
- Regulation (EU) 2016/799 tells how to transform second activities to an activity list organised by the minute
- Not shift invariant:



- Ignoring UTC makes this a relevant issue!!!

Hidden dynamics

Regulation (EU) 2016/799:

(51) Given a calendar minute, if DRIVING is registered as the activity of both the immediately preceding and the immediately succeeding minute, the whole minute shall be regarded as DRIVING.

(52) Given a calendar minute that is not regarded as DRIVING according to requirement 051, the whole minute shall be regarded to be of the same type of activity as the longest continuous activity within the minute (or the latest of the equally long activities).

More hidden dynamics

§4(h) ‘regular weekly rest period’ means any period of rest of at least 45 hours.

§4(i) ‘a week’ means the period of time between 00.00 on a Monday and 24.00 on the following Sunday.

§8.6. In any two consecutive weeks, a driver shall take at least:

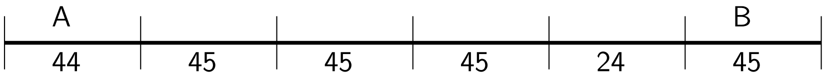
- two regular weekly rest periods, or
- one regular weekly rest period and one reduced weekly rest period of at least 24 hours. However, the reduction shall be compensated by an equivalent period of rest taken en bloc before the end of the third week following the week in question.

A weekly rest period shall start no later than at the end of six 24-hour periods from the end of the previous weekly rest period.

§8.7. Any rest taken as compensation for a reduced weekly rest period shall be attached to another rest period of at least nine hours.

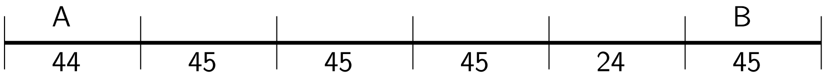
§8.9. A weekly rest period that falls in two weeks may be counted in either week, but not in both.

More hideous behaviour

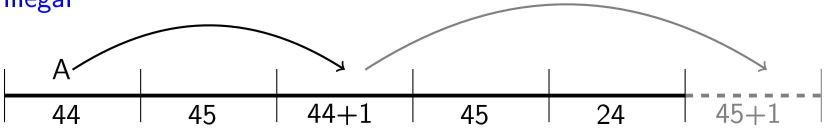


Illegal

More hideous behaviour

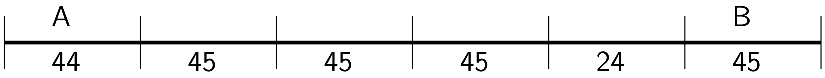


Illegal

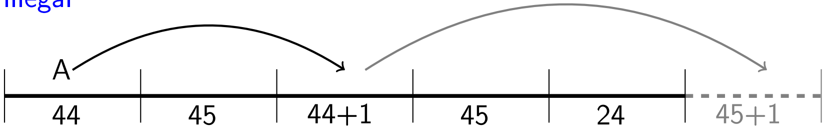


Legal

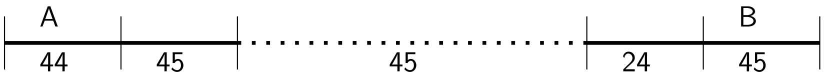
More hideous behaviour



Illegal

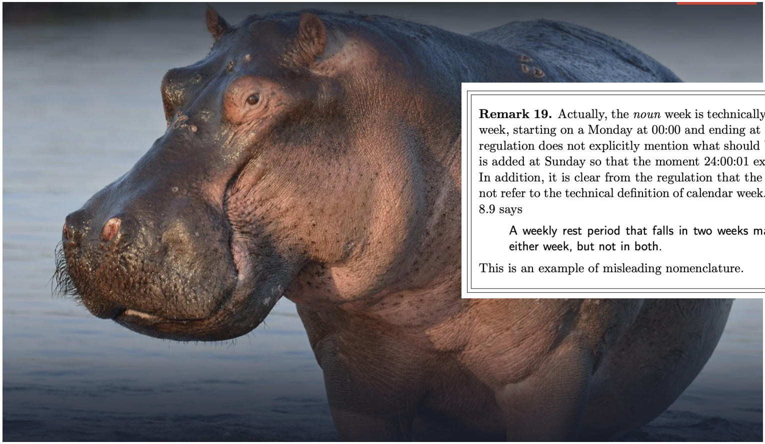


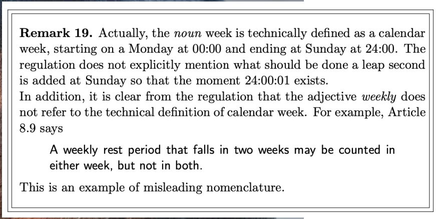
Legal



This can be iterated indefinitely: non-locality

Bad formal ontology





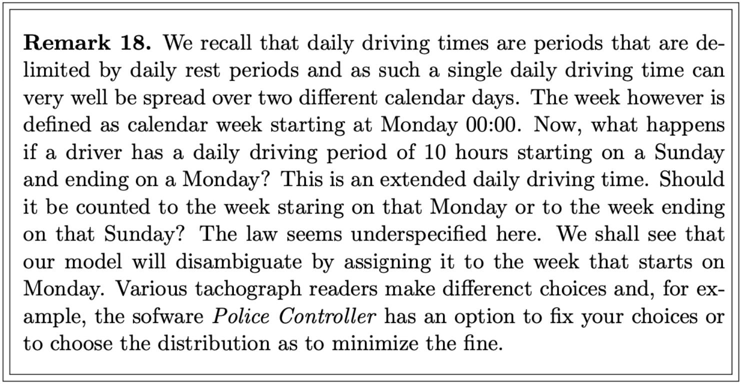
Remark 19. Actually, the *noun* week is technically defined as a calendar week, starting on a Monday at 00:00 and ending at Sunday at 24:00. The regulation does not explicitly mention what should be done a leap second is added at Sunday so that the moment 24:00:01 exists.
In addition, it is clear from the regulation that the adjective *weekly* does not refer to the technical definition of calendar week. For example, Article 8.9 says

> A weekly rest period that falls in two weeks may be counted in either week, but not in both.

This is an example of misleading nomenclature.

Programmer as judge

Article 6.1: The daily driving time shall not exceed nine hours. However, the daily driving time may be extended to at most 10 hours not more than twice during the week.

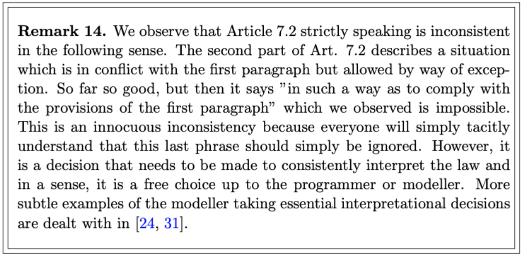


Remark 18. We recall that daily driving times are periods that are delimited by daily rest periods and as such a single daily driving time can very well be spread over two different calendar days. The week however is defined as calendar week starting at Monday 00:00. Now, what happens if a driver has a daily driving period of 10 hours starting on a Sunday and ending on a Monday? This is an extended daily driving time. Should it be counted to the week staring on that Monday or to the week ending on that Sunday? The law seems underspecified here. We shall see that our model will disambiguate by assigning it to the week that starts on Monday. Various tachograph readers make differenct choices and, for example, the sofware *Police Controller* has an option to fix your choices or to choose the distribution as to minimize the fine.

(Potentially) inconsistent laws

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes, unless he takes a rest period.

Article 7 (2nd part): This break may be replaced by a break of at least 15 minutes followed by a break of at least 30 minutes each distributed over the period in such a way as to comply with the provisions of the first paragraph.

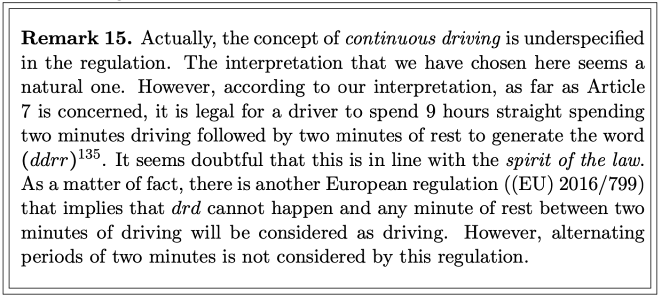


Remark 14. We observe that Article 7.2 strictly speaking is inconsistent in the following sense. The second part of Art. 7.2 describes a situation which is in conflict with the first paragraph but allowed by way of exception. So far so good, but then it says "in such a way as to comply with the provisions of the first paragraph" which we observed is impossible. This is an innocuous inconsistency because everyone will simply tacitly understand that this last phrase should simply be ignored. However, it is a decision that needs to be made to consistently interpret the law and in a sense, it is a free choice up to the programmer or modeller. More subtle examples of the modeller taking essential interpretational decisions are dealt with in [24, 31].

Violation of the spirit of the law

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes, unless he takes a rest period.

Article 7 (2nd part): This break may be replaced by a break of at least 15 minutes followed by a break of at least 30 minutes each distributed over the period in such a way as to comply with the provisions of the first paragraph.

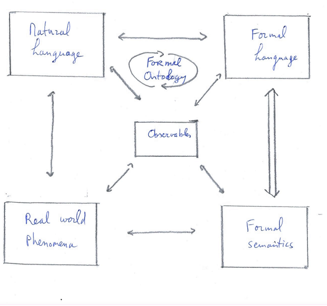


Remark 15. Actually, the concept of *continuous driving* is underspecified in the regulation. The interpretation that we have chosen here seems a natural one. However, according to our interpretation, as far as Article 7 is concerned, it is legal for a driver to spend 9 hours straight spending two minutes driving followed by two minutes of rest to generate the word $(ddrr)^{135}$. It seems doubtful that this is in line with the *spirit of the law*. As a matter of fact, there is another European regulation ((EU) 2016/799) that implies that *drd* cannot happen and any minute of rest between two minutes of driving will be considered as driving. However, alternating periods of two minutes is not considered by this regulation.

Formal ontologies not good

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes, unless he takes a rest period.

Article 7 (2nd part): This break may be replaced by a break of at least 15 minutes followed by a break of at least 30 minutes each distributed over the period in such a way as to comply with the provisions of the first paragraph.

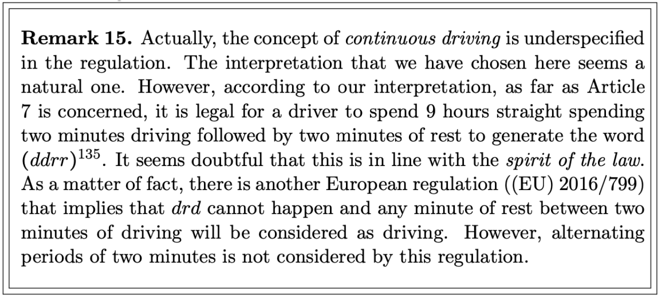


Remark 16. An important problem with the formal ontology of *continuous driving* is that it is not a physical observable like speed. Neither does it seem to be defined in an unambiguous way in terms of physical observables. Consequently we run into troubles as, for example, the one mentioned in Remark 5.1

Violation of the spirit of the law

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes, unless he takes a rest period.

Article 7 (2nd part): This break may be replaced by a break of at least 15 minutes followed by a break of at least 30 minutes each distributed over the period in such a way as to comply with the provisions of the first paragraph.

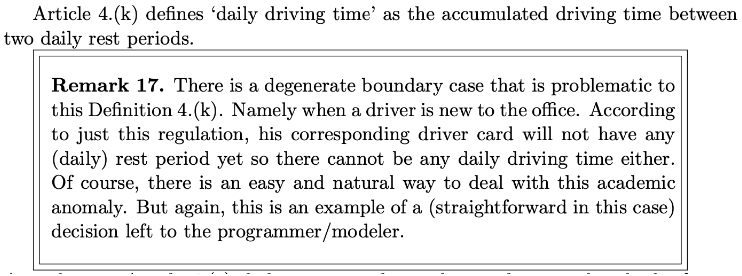


Remark 15. Actually, the concept of *continuous driving* is underspecified in the regulation. The interpretation that we have chosen here seems a natural one. However, according to our interpretation, as far as Article 7 is concerned, it is legal for a driver to spend 9 hours straight spending two minutes driving followed by two minutes of rest to generate the word $(ddrr)^{135}$. It seems doubtful that this is in line with the *spirit of the law*. As a matter of fact, there is another European regulation ((EU) 2016/799) that implies that *drd* cannot happen and any minute of rest between two minutes of driving will be considered as driving. However, alternating periods of two minutes is not considered by this regulation.

Degenerate case

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes, unless he takes a rest period.

Article 7 (2nd part): This break may be replaced by a break of at least 15 minutes followed by a break of at least 30 minutes each distributed over the period in such a way as to comply with the provisions of the first paragraph.



Article 4.(k) defines 'daily driving time' as the accumulated driving time between two daily rest periods.

Remark 17. There is a degenerate boundary case that is problematic to this Definition 4.(k). Namely when a driver is new to the office. According to just this regulation, his corresponding driver card will not have any (daily) rest period yet so there cannot be any daily driving time either. Of course, there is an easy and natural way to deal with this academic anomaly. But again, this is an example of a (straightforward in this case) decision left to the programmer/modeler.

CANTIERE
BUONGOVERNO

Pledge for humanities

- Cost and gdp argument

Pledge for humanities

- Cost and gdp argument
- Incorporate qualitative values into the cost argument

Preferred humanity

Those who know who and what they are do not need to ask what they should do. And those who must ask will not be able to stop asking until they begin to look inside themselves.

Joseph Weizenbaum