# The Structure and Legal Interpretation of Computer Programs

*James Grimmelmann*

Duke Law School

February 26, 2018

# In this talk

- Draw out the similarities and differences in:
  - … how legal actors interpret legal texts
  - … how computers interpret programs
- Motivating example: unauthorized access
- [Bracket questions of hardware vs. software]

# Three vignettes

# Video poker


GAME KING
MULTI-GAME
MULTI-GAME
Earn & Win!
WEDNESDAYS IN JUNE
INSERT TICKETS
GAME KING
MULTI-GAME
EARN
1000 POINTS
$20 FREE SLOT PLAY
WIN
ROYAL FLUSH NO DEUCES
4 DEUCES
1800
1200
2100
1400
2400
1600
2700
1800
3000
2000

# Website scraping

```
User-agent: *
Disallow: /private/
```

```
User-agent: *
Dsallow: /private/
```

# The DAO

"The terms of The DAO Creation are set forth in the smart contract code existing on the Ethereum blockchain at `0xbb9bc244d798123fde783fcc1c72d3bb8c189413`. Nothing in this explanation of terms or in any other document or communication may modify or add any additional obligations or guarantees beyond those set forth in The DAO's code."

# Software with legal effects

- Software can convey permission (to use it)
- Obvious analogies: statutes, licenses, etc.
- These have their own legal interpretive rules
- *What are the interpretive rules for software?*

# Naive functional meaning

# Who is the interpreter?

- Legal texts are addressed to *people*: citizens, counterparties, guests, and especially judges
  - So we care about their meaning to people
- But software is addressed to *computers*: it consists of a series of commands to execute
  - I.e., the functional effects of a program derive from its meaning to a computer

# Proposition:
# *functional meaning ≠ legal meaning*

- Interpretive strategy: *naive functional meaning*
  - Let the computer interpret the code for you
    - What the code allows is what the law allows
- This is obviously insufficient as a theory
  - Computers malfunction; software is buggy

# Proposition:
# *functional meaning ~ legal meaning*

- Not anything goes!
  - Video poker is not video backgammon
    - There really is "smart contract code existing on the Ethereum blockchain at `0xbb9b`..."
- Legal meaning is based on functional meaning

# Literal functional meaning

# Specification and semantics

- What does `2**2` mean in a programming language?
- Three answers:
  - Use a program: a *reference implementation* whose behavior is by stipulation treated as correct
  - Use natural language: a *specification* that defines the behavior of a correct implementation
  - Use mathematics: a *formal semantics* that identifies programs with abstract entities

# Two questions

- Where do specifications and semantics come from?
  - Some people got together to define them
- What language are we running?
  - “Python” 2.7 is different from “Python” 3.6
- These questions can be answered only by reference to a community of programmers and users

# Fixing functional meaning

- A technical community agrees on a process for deriving a functional meaning from texts
- Developers implement that process on different computers, with different tools, etc.
- Most of the time, running a program on most implementations yields the same result
- A program's *literal functional meaning* is what a standardized implementation would do with it

# Ordinary functional meaning

# The price we pay

- Running a program produces *a* result, but not necessarily the right result
- Specifying up front the resolution of all ambiguities means getting many wrong
- The concept of “bug” assumes a distinction between actual and intended behavior

# Ordinary meaning

- The ordinary legal meaning of a text is the meaning a reasonable audience would give it
  - A program's audience consists of its users
    - Users expect that programs contain bugs
- A program's *ordinary functional meaning* is what reasonable people in the position of its users would expect it to do, if it were free of bugs

# Three vignettes, redux

# Video poker

- Reasonable video poker players understand:

  (1) They are allowed to play skillfully

  (2) Quitting and returning to a game probably wasn't intended to change the payout multiplier

- The case looks hard because of the conflict between (1) and (2). But ordinary functional meaning controls: the payout trick is a bug.

# Robots.txt

- Reasonable web scrapers understand:

  (1) `Dsallow` isn't a valid keyword

  (2) The standard is written for bots to process

- (2) means that web scrapers and web hosts have selected into literal functional meaning, i.e., (1) is not "corrected" to `Disallow`

# The DAO

- Reasonable blockchain investors understand:

  (1) The DAO contract was buggy

  (2) The DAO's legal instruments purported to make the contract judicially unreviewable

  (3) The DAO depends on Ethereum

- Whether (2) successfully selects literal functional meaning is a question of offline contract law. But (3) makes even literal functional meaning ambiguous!

# Questions?