

# All Smart Contracts Are Ambiguous

*James Grimmelmann*

Cornell Blockchain

April 12, 2019

# Legal contracts

# An ambiguous contract

“US Fresh Frozen Chicken, Grade A, Government Inspected, Eviscerated, 2½-3 lbs. and 1½-2 lbs. each, all chicken individually wrapped in cryovac, packed in secured fiber cartons or wooden boxes, suitable for export

75,000 lbs. 2½-3 lbs.....@\$33.00

25,000 lbs. 1½-2 lbs.....@\$36.50

per 100 lbs. FAS New York, scheduled May 10, 1957 pursuant to instructions from Penson & Co., New York.”

*Frugaliment Importing Co. v. B.N.S. Int’l Sales Corp.*, 190 F. Supp. 116 (S.D.N.Y. 1960)

# What is “chicken”?

- Buyer (plaintiff): “a young chicken suitable for broiling and frying”
- Seller (defendant): “any bird of that genus”

What if the contract said “young chicken suitable for broiling”?



What’s “suitable”? What’s “broiling”?

# An inescapable problem

- The meaning of natural language is social
- Even “objective” sources like dictionaries depend on how people actually use words
  - This can be hard to ascertain
  - This can vary from person to person
  - This can change over time

# Smart contracts



# An escrow contract

## SAMPLE COPY

### ESCROW AGREEMENT

Agreement made this \_\_\_\_ day of \_\_\_\_\_, 19\_\_\_\_, by and between \_\_\_\_\_ and \_\_\_\_\_ as Escrow Agent (the "Escrow Agent").

### WITNESSETH:

Pursuant to Section 513( ) of the Delaware Insurance Code, \_\_\_\_\_ is required to maintain on deposit in the State of Delaware for the protection of all its policyholders wherever located, except the deposits required by Delaware statute to be maintained solely for the benefit of Delaware policyholders, cash or cash equivalents in an amount and in a manner specified by the said Code and the Insurance Commissioner of the State of Delaware. It is intended that the deposit made and maintained pursuant to this Escrow Agreement satisfy the requirements of said Code and said Insurance Commissioner.

**Now, THEREFORE,** in consideration of the premises and the covenants herein contained, the parties hereto, intending to be legally bound, agree as follows:

(1) \_\_\_\_\_ hereby agrees to deposit with the Escrow Agent cash or cash equivalents of a kind and of a value or amount sufficient to satisfy the applicable deposit requirements of the Delaware Insurance Code. This deposit (hereinafter referred to as the "Escrow Deposit") shall consist initially of the cash or cash equivalents listed on Schedule A attached hereto. Thereafter, the Escrow Deposit shall consist of such other cash or cash equivalents as \_\_\_\_\_ may deliver to the Escrow Agent with written instructions that the same shall be part of the Escrow Deposit. the Escrow Agent shall hold the Escrow Deposit subject to the terms and conditions of this Escrow Agreement. In no event, shall the Escrow Agent have any responsibility to ascertain whether the Escrow Deposit satisfies the applicable deposit requirements of the Delaware Insurance Code.

(2) All cash or cash equivalents delivered by \_\_\_\_\_ as part of the Escrow Deposit shall be registered in the name of \_\_\_\_\_ and shall be accompanied by irrevocable transfer powers executed in blank, sufficient to allow the Escrow Agent to sell or deliver such cash or cash equivalents in accordance with this Escrow Agreement. The Escrow Agent may cause said cash or cash equivalents to be registered in the name of the Escrow Agent or its nominee. The cash or cash equivalents so deposited shall at all times be kept separate and distinct from all other deposits, so that at all times they may be identified as belonging to \_\_\_\_\_.

(3) \_\_\_\_\_ shall receive from time to time payments of any interest payments or other distributions upon any government obligations, corporation obligations included as a part of said



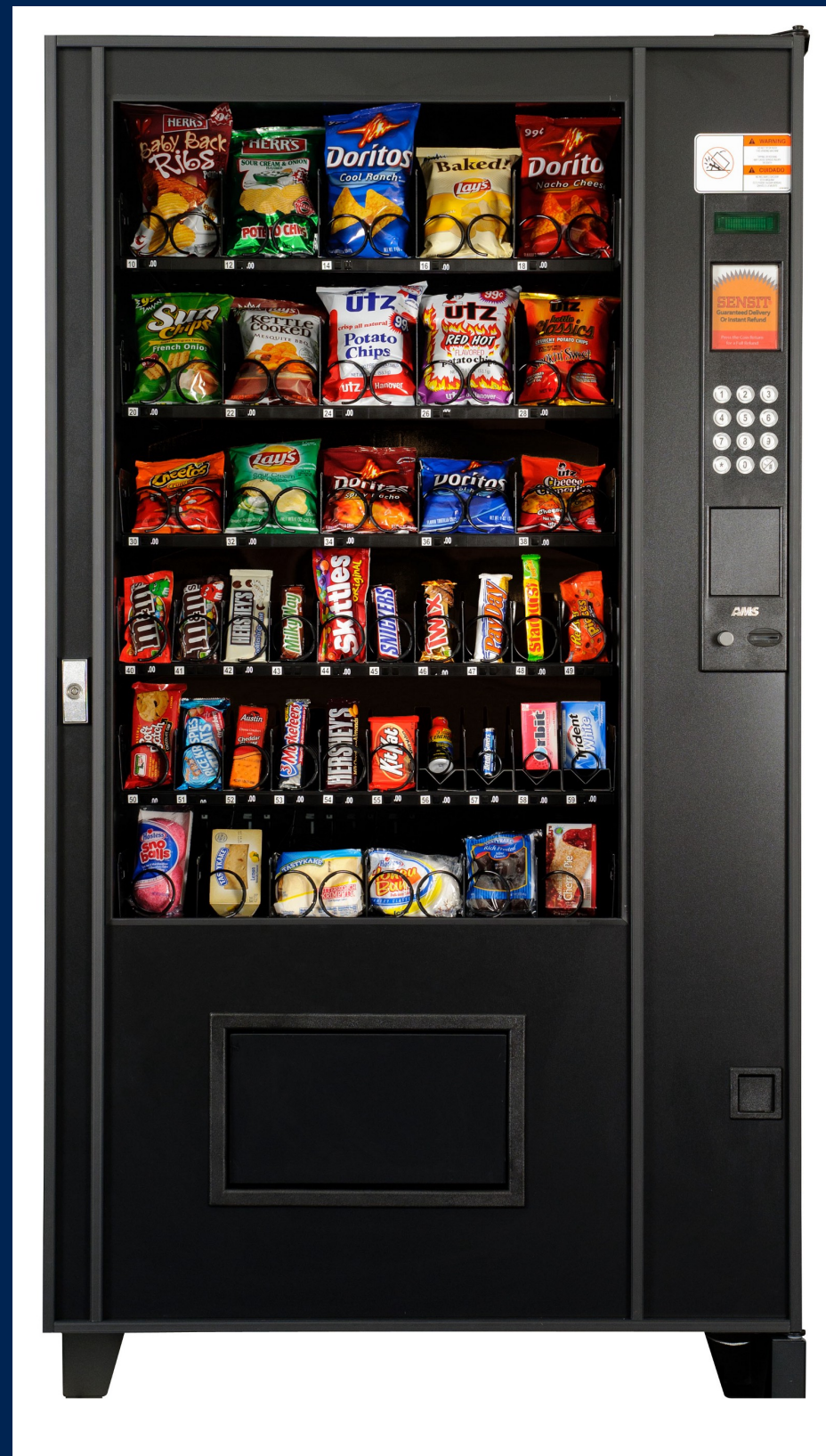
# An escrow “contract”

 [escrow.sol](#)

Raw

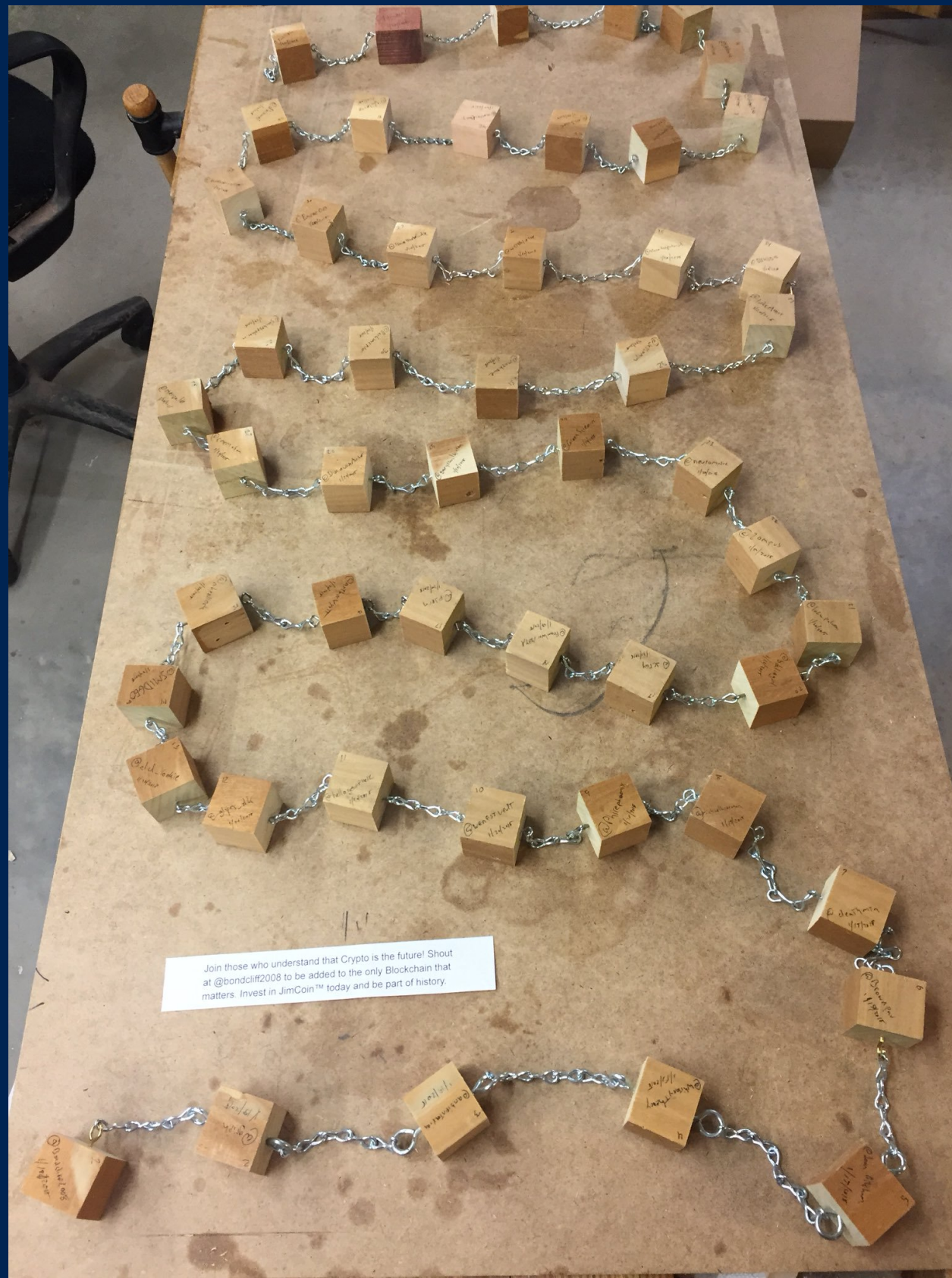
```
1  pragma solidity ^0.4.21;
2
3  import "../node_modules/zeppelin-solidity/contracts/token/ERC20/ERC20.sol";
4  import "../node_modules/zeppelin-solidity/contracts/ownership/Ownable.sol";
5  import "../webshop/Webshop.sol";
6
7  contract Escrow is Ownable {
8      enum PaymentStatus { Pending, Completed, Refunded }
9
10     event PaymentCreation(uint indexed orderId, address indexed customer, uint value);
11     event PaymentCompletion(uint indexed orderId, address indexed customer, uint value, PaymentStatus status);
12
13     struct Payment {
14         address customer;
15         uint value;
16         PaymentStatus status;
17         bool refundApproved;
18     }
19
20     mapping(uint => Payment) public payments;
21     ERC20 public currency;
22     address public collectionAddress;
23     Webshop public webshop;
24
25     function Escrow(ERC20 _currency, address _collectionAddress) public {
26         currency = _currency;
27         collectionAddress = _collectionAddress;
28         webshop = Webshop(msg.sender);
29     }
```

# Standard example





# Blockchain?



# Distributed ledgers

- Every participant has copy of a shared ledger
- New transactions accepted only if:
  - Cryptographically signed by the sender
  - Consistent with previous transactions

# Blockchain consensus

- Participants must agree on transactions
- Penalty for disagreement is incompatibility
- Strong incentive to accept valid blocks

# Smart contracts on a blockchain

- Specify a virtual machine (VM)
  - Transactions run programs on the VM
  - These programs can send and receive funds
- Blockchain protocol forces consistency

# Smart contracts and ambiguity



# The claim for unambiguity

- The meaning of “chicken” is a *social fact*
  - Its meaning can vary and be misunderstood
- The meaning of  $2+2$  in Python is a *technical fact*
  - This expression will always evaluate to 4
  - Its meaning never changes, and if you think it evaluates to 5 you are wrong

# Where does program meaning come from?

- Why *doesn't*  $2+2$  in Python evaluate to 5?
- Not because that's what “ $2+2$ ” inherently means
  - Any more than “chicken” inherently means any *gallus gallus domesticus*, even one that is wholly unsuitable for cooking
- In 1991, GvR picked  $+$  as the addition operator
  - He could have picked  $++$  instead

# Program meaning is a social fact, too

- Yes,  $2+2$  in Python is unambiguously 4
- But that's only because Python users have already agreed on what "Python" is
- If they agreed differently, "Python" would be different, and so might  $2+2$
- This happens *every time* there's a new version
- Technical facts depend on social facts!

# Examples

# Example 1: Oracles

- How does a smart contract observe the world?
  - An *oracle* has to tell it what happened
  - E.g., a trusted party or a fixed data feed
- This is also a problem of ambiguity
  - The world is complex
  - Contract terms map ambiguously onto the world
  - The oracle resolves the ambiguity

# Example 2: Upgrades

- In 2017, Bitcoin upgraded to implement “segregated witness”
- Some data moved out of the blockchain, effectively allowing more transactions
- The blockchain before the upgrade and the blockchain after have *different semantics*
- Some transactions that were valid under the old rules are invalid under the new ones

# Example 3: Bitcoin Cash

- A long-running dispute over Bitcoin block size caused some users to fork Bitcoin Cash
  - Bitcoin has  $\sim 1\text{MB}$  blocks
  - Bitcoin Cash had 8MB blocks (now 32MB)
- The two blockchains have different semantics
  - Is a block valid? The question can't be answered without specifying *by whom*



# Forks and ambiguity

- Forks are consensus failures
  - Each blockchain by itself achieves local consensus, but there is no global consensus
- Forks create explicit ambiguity
  - Each blockchain by itself is “unambiguous” but the *choice of blockchains* creates ambiguity
- These are inextricably linked

# Example 4: 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.”

# The DAO

- April 2016: The DAO begins crowdfunding for a democratic online venture capital fund
- May 2016: 11,000+ investors put \$150M+ of assets into The DAO
- June 2016: An anonymous hacker drains \$50M of the assets into their own account

# Ethereum Classic

- Following the DAO hack, Ethereum upgraded to a new version that *specifically unwound the DAO transactions*
- Some users objected enough to fork Ethereum Classic, which didn't have this “upgrade”
- The two blockchains have different semantics

# The DAO (legal) contract

- The English phrase “the smart contract code existing on the Ethereum blockchain at `0xbb9bc244d798123fde783fcc1c72d3bb8c18941`” is ambiguous
- “the Ethereum blockchain” does not uniquely refer: do you mean ETH or ETC?
- It uniquely referred when the contract was drafted, but no longer

Where to go from here?





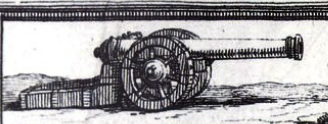

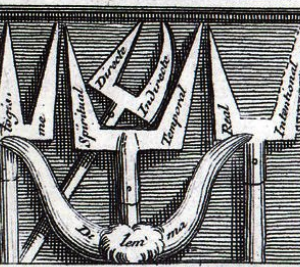
# All is not lost

- Smart contracts are based on social facts
  - Social facts are empirically contingent: they are always open to contestation and change
- Legal contracts are based on social facts, too
  - And a lot of the time, they work just fine!
- Smart contracts *cannot* be perfectly unambiguous
  - But perhaps they can be unambiguous enough



*Non est potestas Super Terram quae Comparetur ei Iob. 41. 24.*



|  |   |   |
|--|---|---|
|  | <b>LEVIATHAN</b>  |  |
|  | Or  |  |
|  | THE MATTER, FORME   |  |
|  | and POWER of A COMMON   |  |
|  | WEALTH ECCLESIASTICALL  |  |
|  | and CIVIL.  |   |
|  | By THOMAS HOBBS   |   |
|  | of MALMESBVRY.  |   |
|  |  |   |



# Blockchains are made out of people



Questions