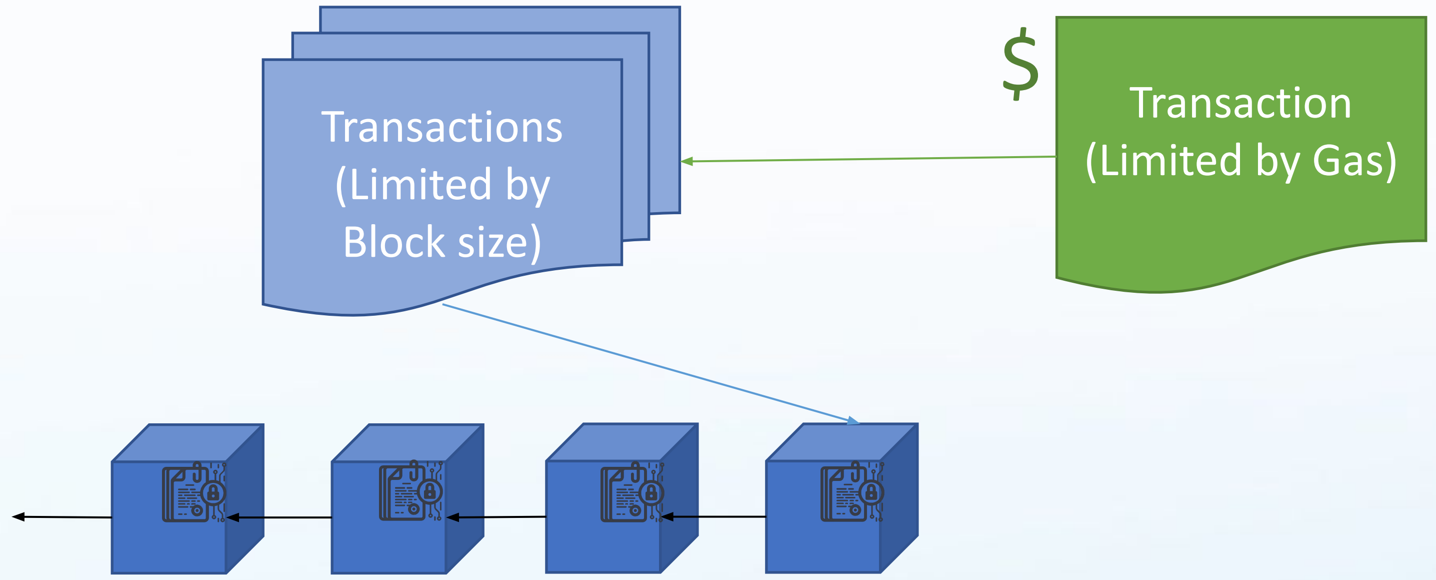
Quantstamp

Leaders in blockchain security and solutions

Constant-Time Updates Using Token Mechanics

Sebastian Banescu
Martin Derka
<u>Jan Gorzny</u>
Sung-Shine Lee
Alex Murashkin

Problem Setting



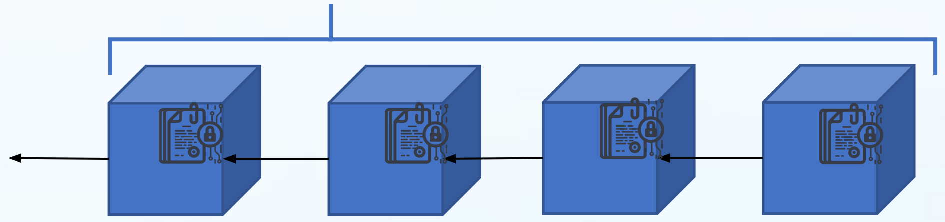
Proof-of-Work blockchain with support for metered smart contracts (e.g., Ethereum)

Problem Statement

```
value1 = 7;
value2 = 8;
.
.
.
valueN = 199;
```

Update all values

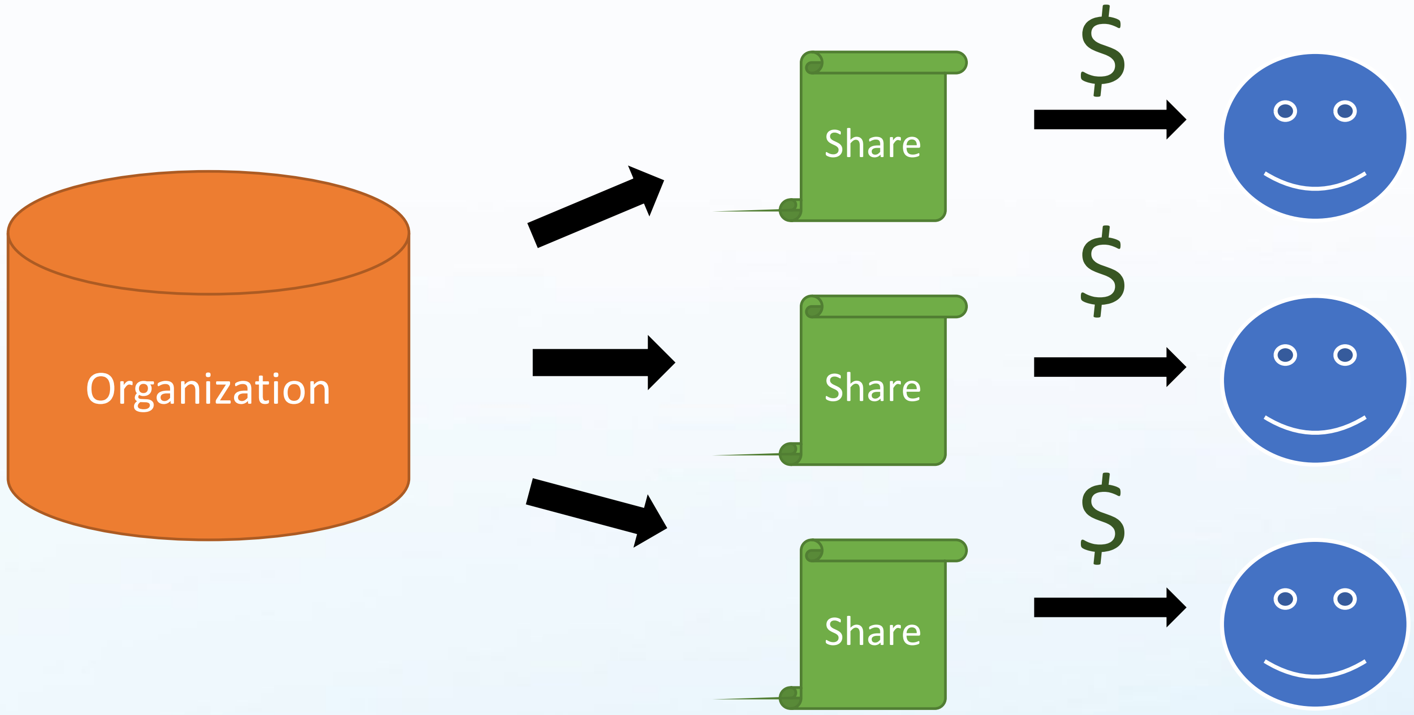
```
value1 = 8;
value2 = 9;
.
.
.
valueN = 200;
```



Proof-of-Work blockchain with support for metered smart contracts (e.g., Ethereum)

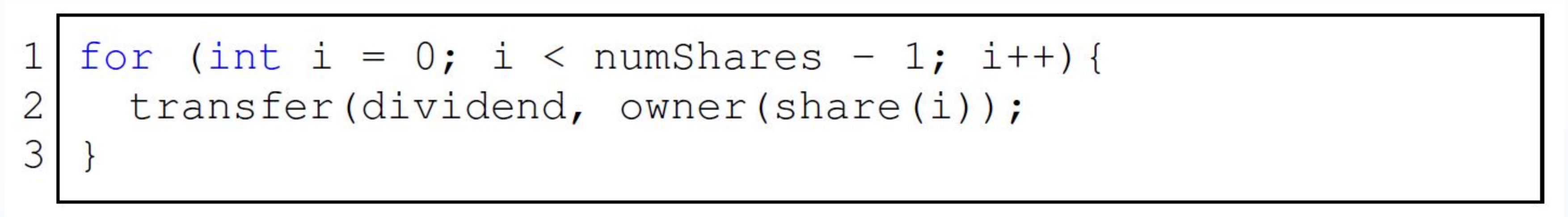
Given that we cannot update all values in a single transaction, and many transactions may be too costly (assuming N is large), how should we store this data?

Motivating Example: Distributing Share Dividends On-Chain



Motivating Example

Naïve Solution

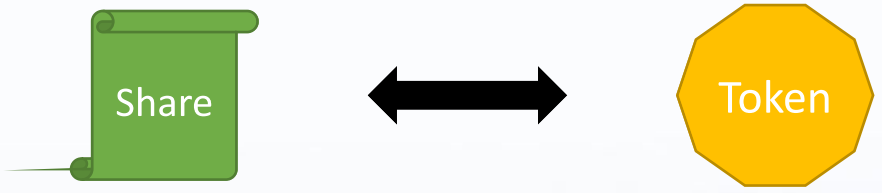


```
for (int i = 0; i < numShares - 1; i++){
  transfer(dividend, owner(share(i));
}
```

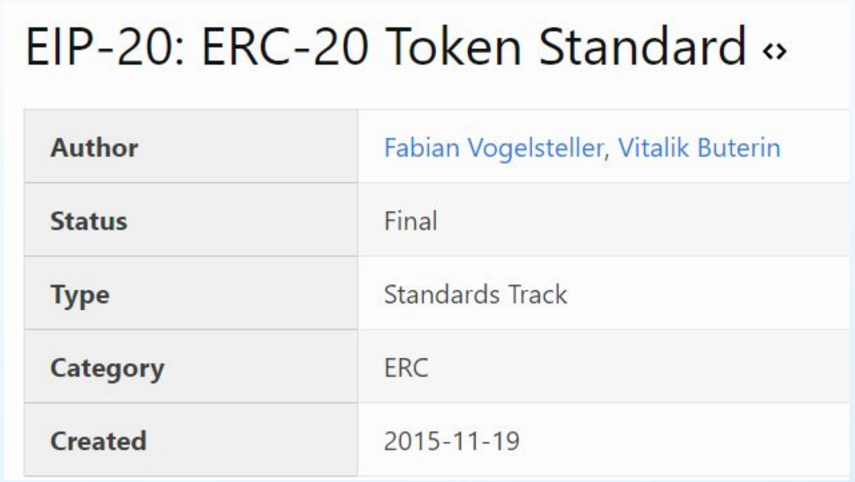
Listing 1. A simple loop to award dividends.

Motivating Example
Token-Based Solution

Associate each share with a token



The value of a share is the value of the corresponding token. Use standard token interfaces for related actions.



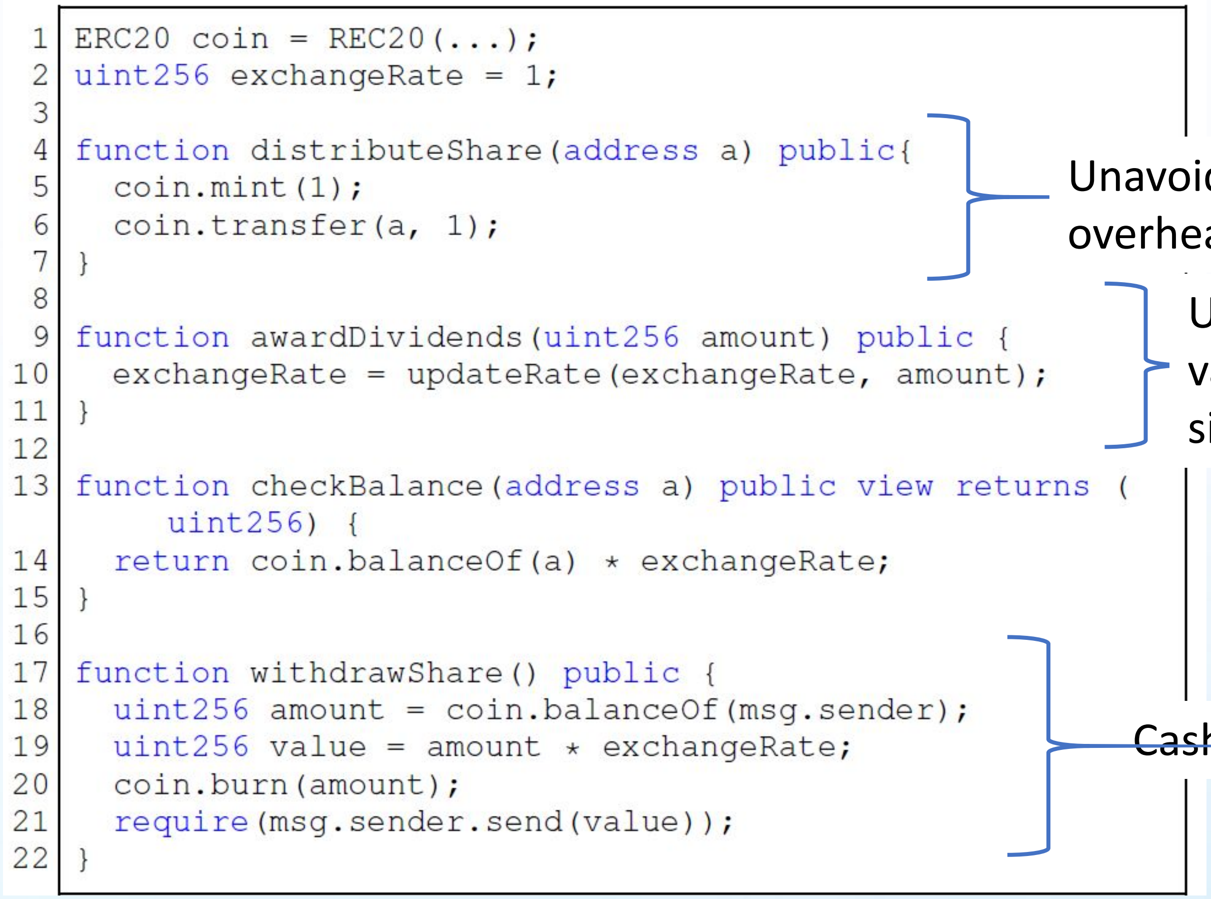
EIP-20: ERC-20 Token Standard

Author	Fabian Vogelsteller, Vitalik Buterin
Status	Final
Type	Standards Track
Category	ERC
Created	2015-11-19

https://eips.ethereum.org/EIPS/eip-20

Motivating Example

Token-Based Solution



```
ERC20 coin = REC20(...);
uint256 exchangeRate = 1;

function distributeShare(address a) public{
  coin.mint(1);
  coin.transfer(a, 1);
}

function awardDividends(uint256 amount) public {
  exchangeRate = updateRate(exchangeRate, amount);
}

function checkBalance(address a) public view returns (
    uint256) {
  return coin.balanceOf(a) * exchangeRate;
}

function withdrawShare() public {
  uint256 amount = coin.balanceOf(msg.sender);
  uint256 value = amount * exchangeRate;
  coin.burn(amount);
  require(msg.sender.send(value));
}
```

Unavoidable overhead

Updates all values simultaneously

Convenience

Cashing out

Motivating Example

Token-Based Solution

Share Action	Contract Function
Distributing Shares	distributeShare() *using* **mint()**
Awarding Dividends	updateExchangeRate()
Checking Balance	checkBalance() *using* **balanceOf()**
Selling Shares	withdraw() *using* **burn()**
Transferring Shares	transfer()

Common ERC20 functions

Motivating Example
Token-Based Solution

- Updating the exchange rate changes the value, awarding dividends:

$$rate' = \frac{(rate \times supply) + amount}{supply}$$

- Can award new shares similarly:

$$rate' = \frac{(rate \times supply)}{supply + x}$$

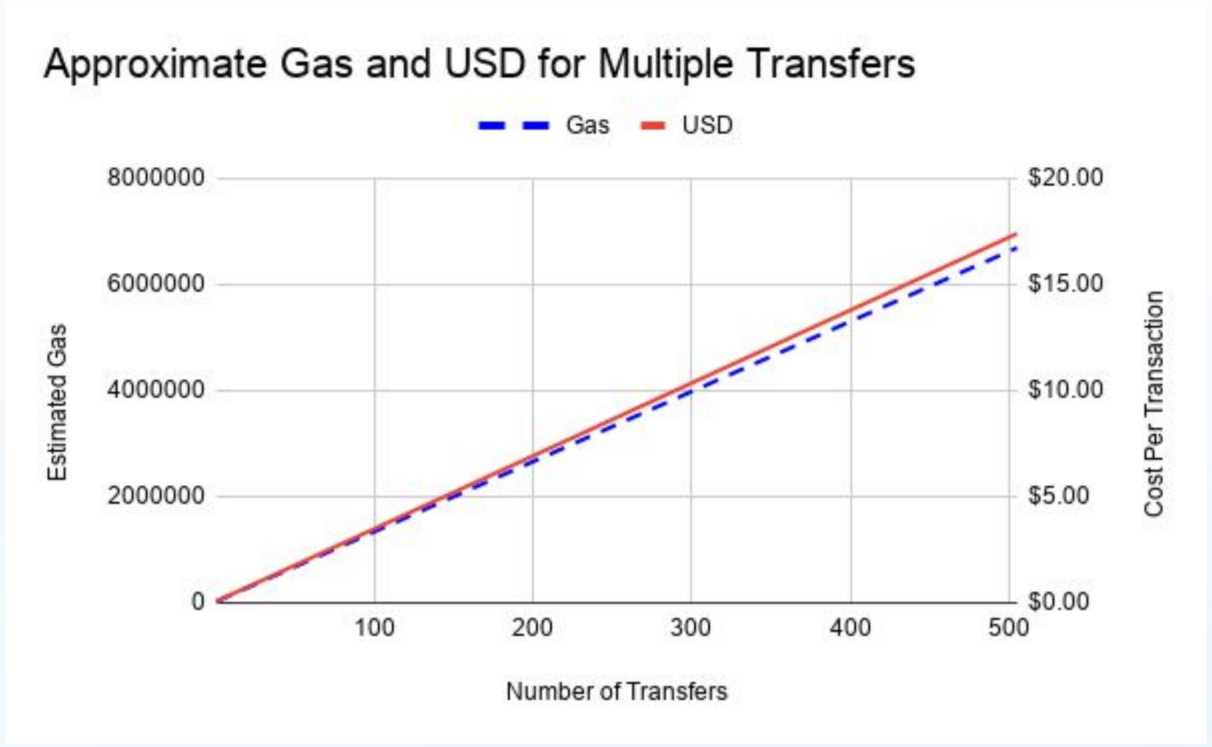
Motivating Example
Token-Based Solution

- Other enhancements are possible:
  - Asymmetric cost functions:
    $f(x) = k/y$ for some constant $k > 0$ and y is the circulating balance

    $$f(x) \to \infty \text{ as } y \to 0 \quad \text{and} \quad f(x) \to 0 \text{ as } y \to \infty$$

  - Binary data

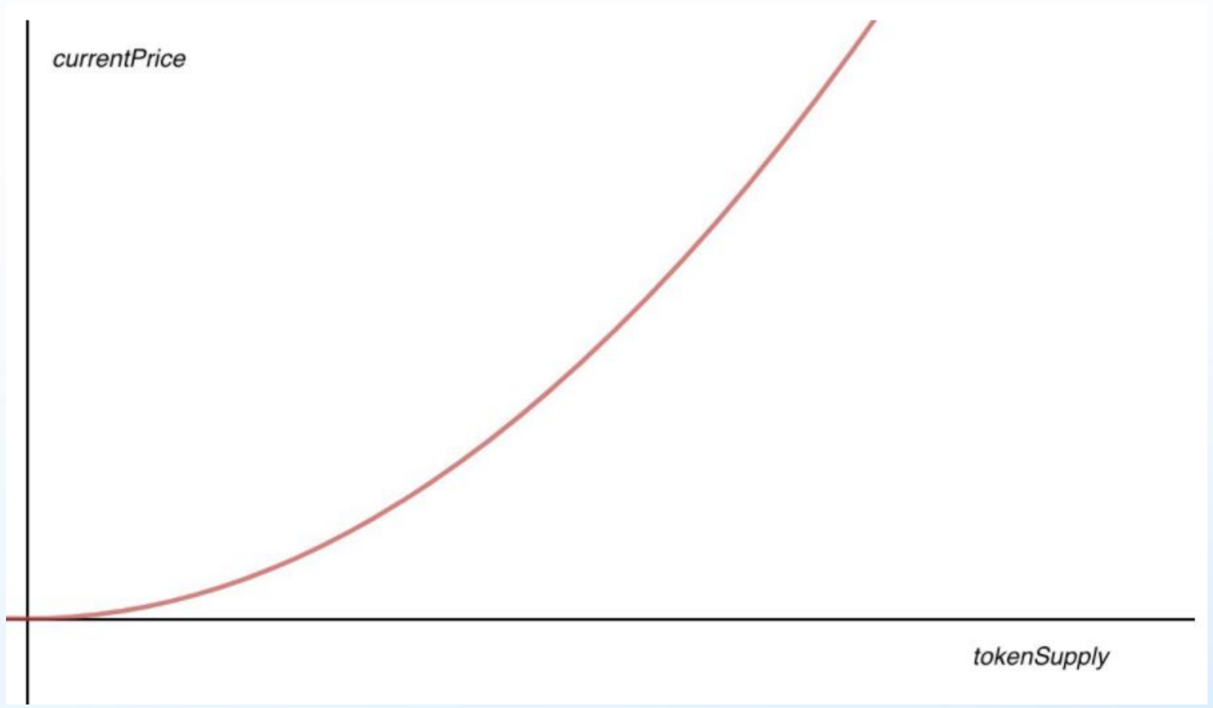
Analysis



Cost to update in this solution: constant 26414 gas, or about $0.07 at time of writing

Related Work

- Bonding Curves
- Prediction Markets
- “Traditional” Data Structure Research



https://yos.io/2018/11/10/bonding-curves/

Conclusion

Thank you!

Questions? Comments?

jan@quantstamp.com