PROJECT REQUIREMENTS & USE CASES

Andrea Vitaletti

PROJECT OBJECTIVES

- Develop and test state-of-the-art **smart contracts** on Algorand.
- Focus on interesting technical challenges.
- Improve already available smart contracts with enhanced technical solutions.
- New use-cases are very welcome if they can support the development of techically challenging smart contracts.

GROUPS

- Participants will be grouped by the school organizers according to their background and expertise, but also taking into account the time zone to facilitate meetings.
- Groups, composed of 3 students each, will work on the projects offline also interacting with the school organizers and with the lab tutors.

IMPORTANT DATES

- 9/10/2022: Project Idea Proposal and Approval
- 17/10/2022: Project Development and Submission
- 18/10/2022: Project Presentation and Awards

Projects will be continuously evaluated

MATERIAL TO BE SUBMITTED

CRITERIA

Clarity of the specifications of the smart contract (e.g. functionalities, roles etc) including use cases	10
Clarity of the technical challenges in implementing and deploying the smart contract	10
Analysis of the state of the art of “similar” or relevant solutions	10
Proof-of-concept publicly available in a testnet and evidences that your smart contracts work as expected (e.g. reference to relevant transactions, tests)	35

Code on github with a well documented README. Note that code quality will be evaluated in terms of simplicity, clarity and comments as well as good practice in code development (e.g. guidlines).

BONUS

Qualitative discussion on the security and correctness of the developed smart contracts	5
Discussion of potential business benefits	5

A README TEMPLATE

```
# Group

# Goal of the Project
Problem, Solution (Be focused!)

# Smart Contract Specifications
Requirements, Use cases, Functions ...

# State of the Art
Relevant Smart Contracts, Papers
Posts in the developer portal ...

# Technical Challenges
Beyond the state of the art
```

AWARDS

Awards will be granted once projects are submitted to the Algorand Developers Portal

The amounts of the awards per group are the following (before taxes):

- 1st prize: 2500 EUR
- 2nd prize: 1500 EUR
- 3rd prize: 1000 EUR

ALGORAND USE CASES

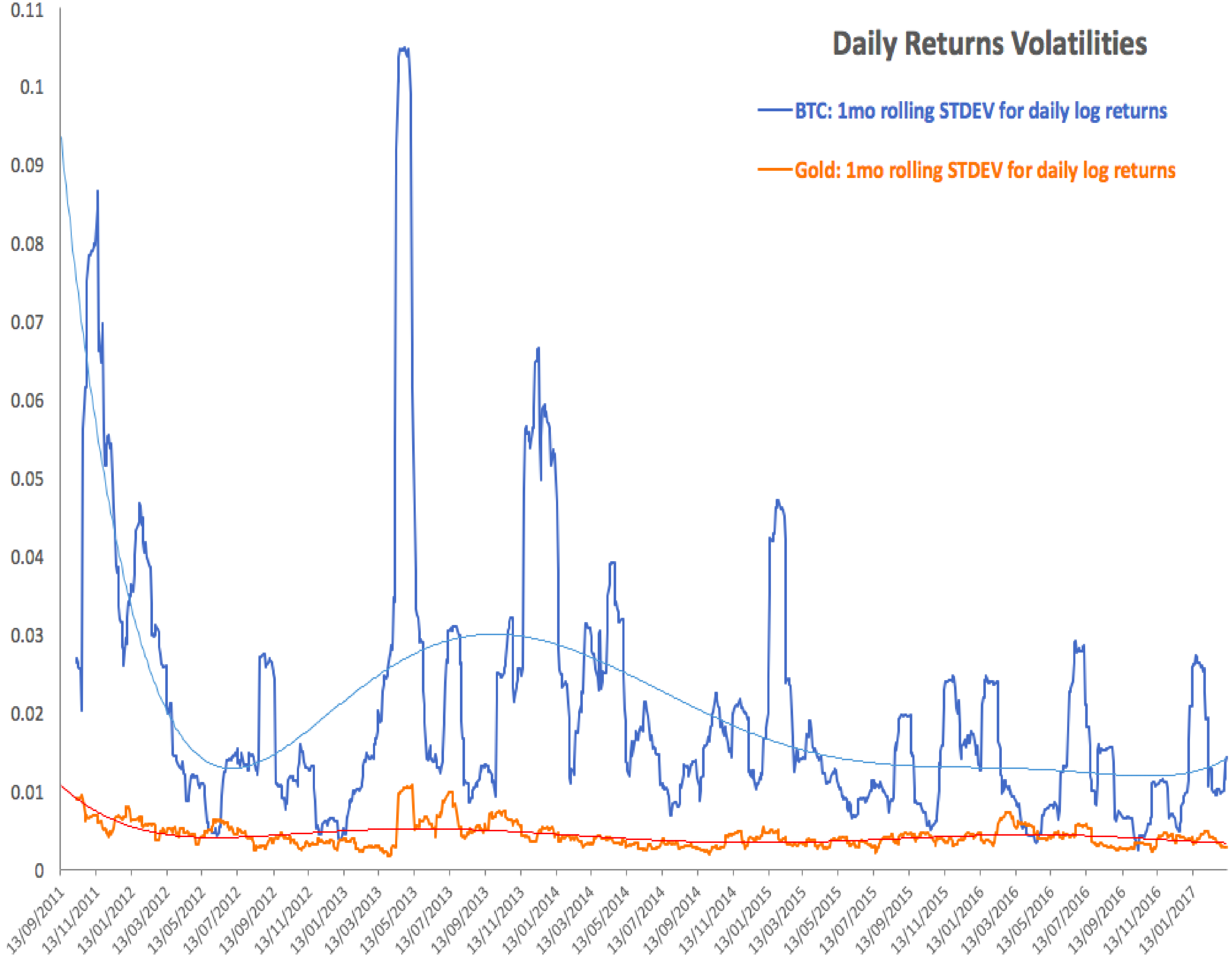
Filter use cases:

Entertainment   Infrastructure   Securities   Supply Chain   Identity   [✓ Stablecoins]

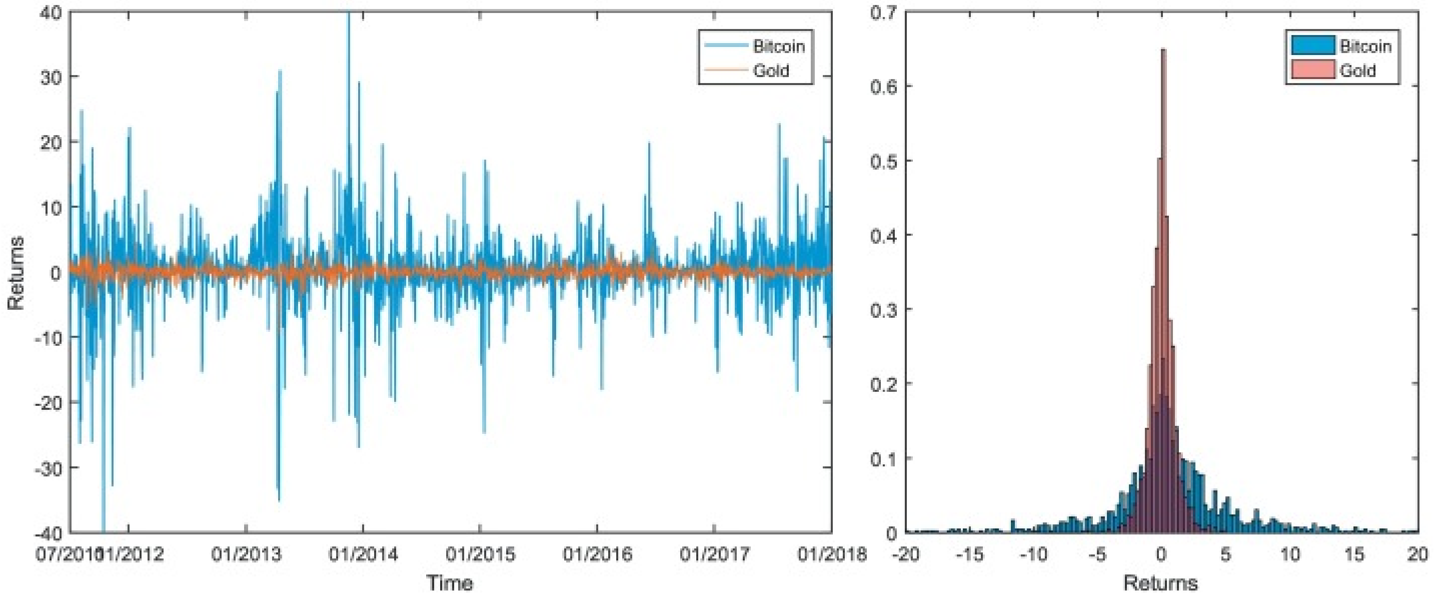
Government / Public Sector   Insurance   Environmental   Gaming   Defi   Financial Institutions

Digital Assets

A DETOUR ON STABLECOINS


Daily Returns Volatilities
BTC: 1mo rolling STDEV for daily log returns
Gold: 1mo rolling STDEV for daily log returns
0.11
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
13/09/2011
13/11/2011
13/01/2012
13/03/2012
13/05/2012
13/07/2012
13/09/2012
13/11/2012
13/01/2013
13/03/2013
13/05/2013
13/07/2013
13/09/2013
13/11/2013
13/01/2014
13/03/2014
13/05/2014
13/07/2014
13/09/2014
13/11/2014
13/01/2015
13/03/2015
13/05/2015
13/07/2015
13/09/2015
13/11/2015
13/01/2016
13/03/2016
13/05/2016
13/07/2016
13/09/2016
13/11/2016
13/01/2017

BITCOIN VS GOLD

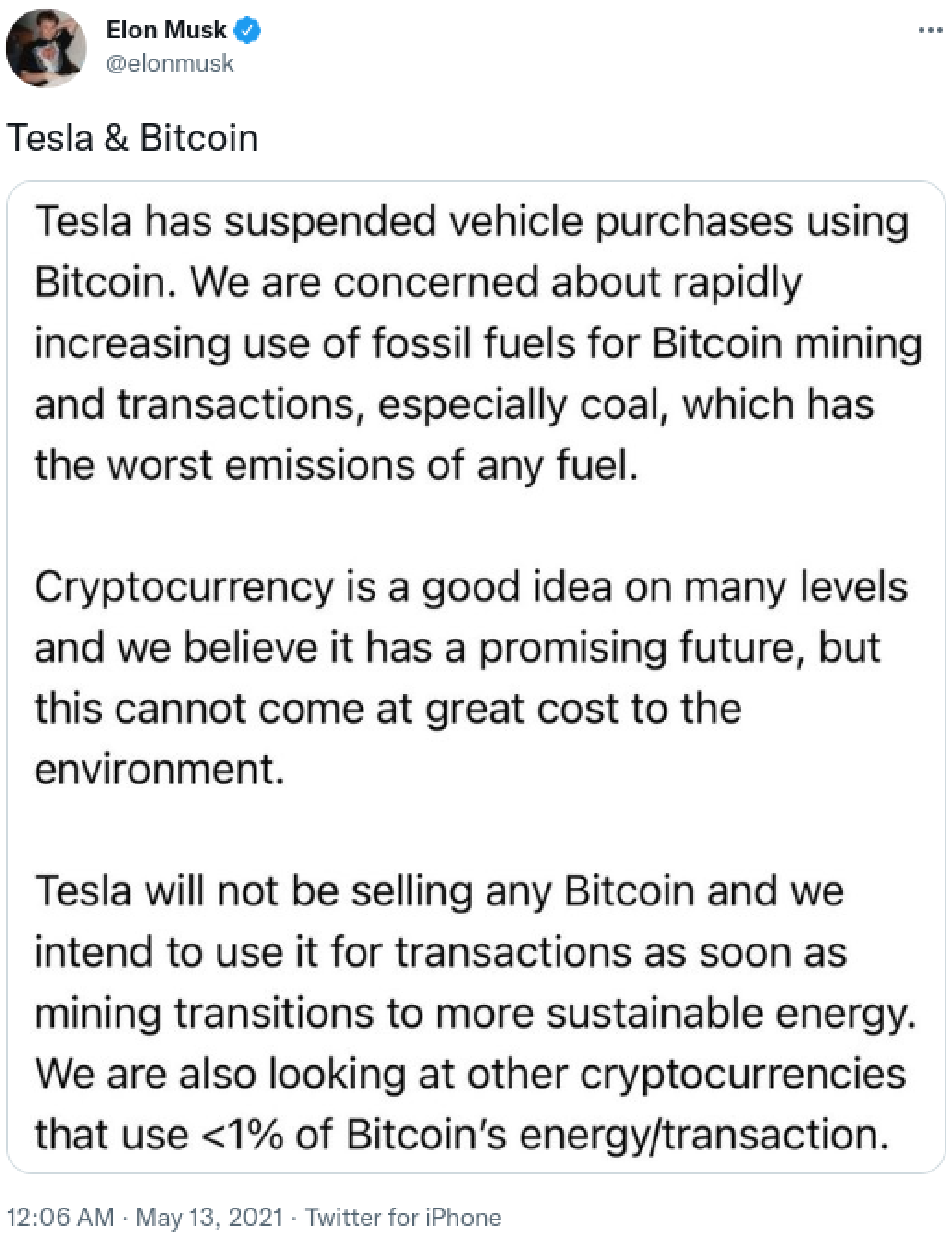


We conclude that Bitcoin and Gold feature fundamentally different properties as assets and linkages to equity markets. Our results hold for the broad cryptocurrency index CRIX. As of now, Bitcoin does not reflect any distinctive properties of Gold other than asymmetric response in variance.

(Klein et al., 2018)

TECHNOLOGY IS DECENTRALIZED

... OPINIONS?


Elon Musk
@elonmusk

Tesla & Bitcoin

Tesla has suspended vehicle purchases using Bitcoin. We are concerned about rapidly increasing use of fossil fuels for Bitcoin mining and transactions, especially coal, which has the worst emissions of any fuel.

Cryptocurrency is a good idea on many levels and we believe it has a promising future, but this cannot come at great cost to the environment.

Tesla will not be selling any Bitcoin and we intend to use it for transactions as soon as mining transitions to more sustainable energy. We are also looking at other cryptocurrencies that use <1% of Bitcoin's energy/transaction.

12:06 AM · May 13, 2021 · Twitter for iPhone

THE EFFECT OF OPINION MAKERS

STABLE COIN

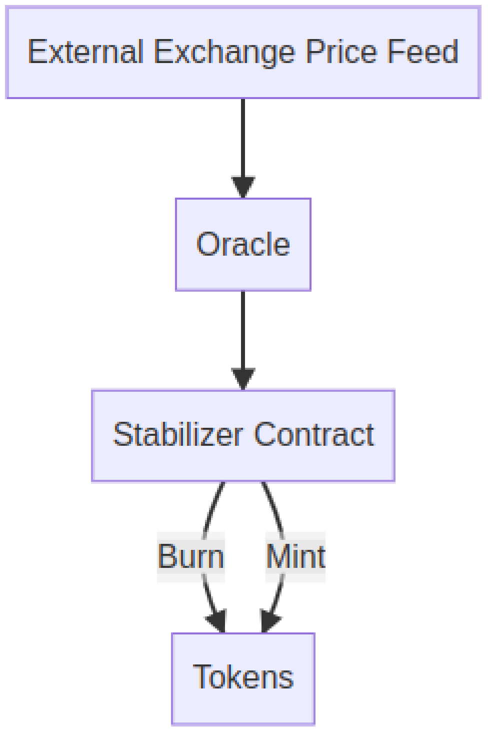
A stablecoin is a digital currency that is pegged to a "stable" reserve asset like the U.S. dollar or gold.

ALGORITHMIC STABLE COIN

Cryptocurrencies – similar to all assets in the market, such as houses or stocks – move up and down in price depending on the market demand and the supply of the asset.

To prevent the price of a stablecoin depegging – moving away from the reference (e.g. $1) – while subject to market conditions, algorithms regulate supply and demand.

ALGORITHMS REGULATE SUPPLY AND DEMAND

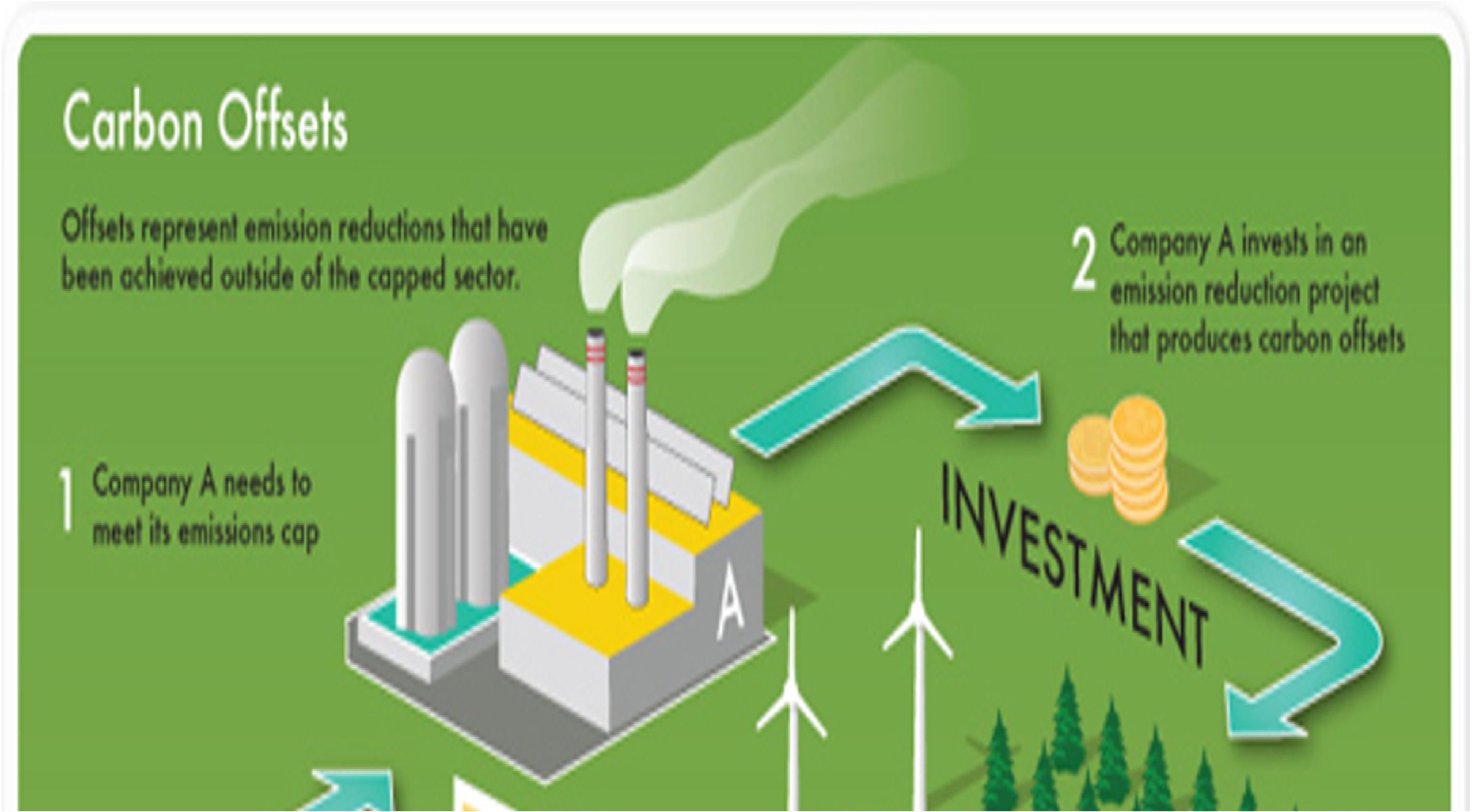


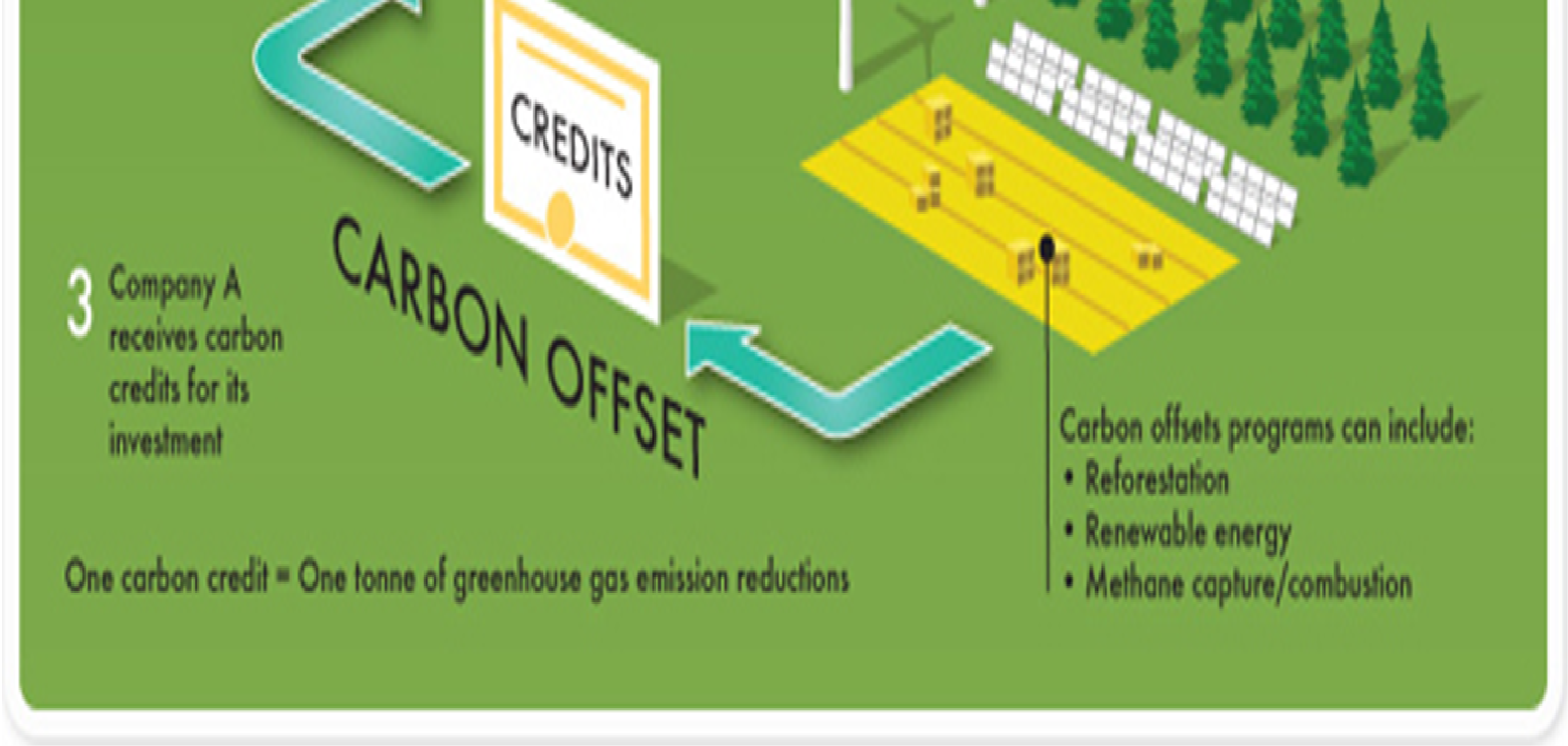
The stabilizer computes the amount of tokens to be burned and minted from every users wallet.

If price of the coin is greater than the fiat currency, it burns tokens, it mints token otherwise.

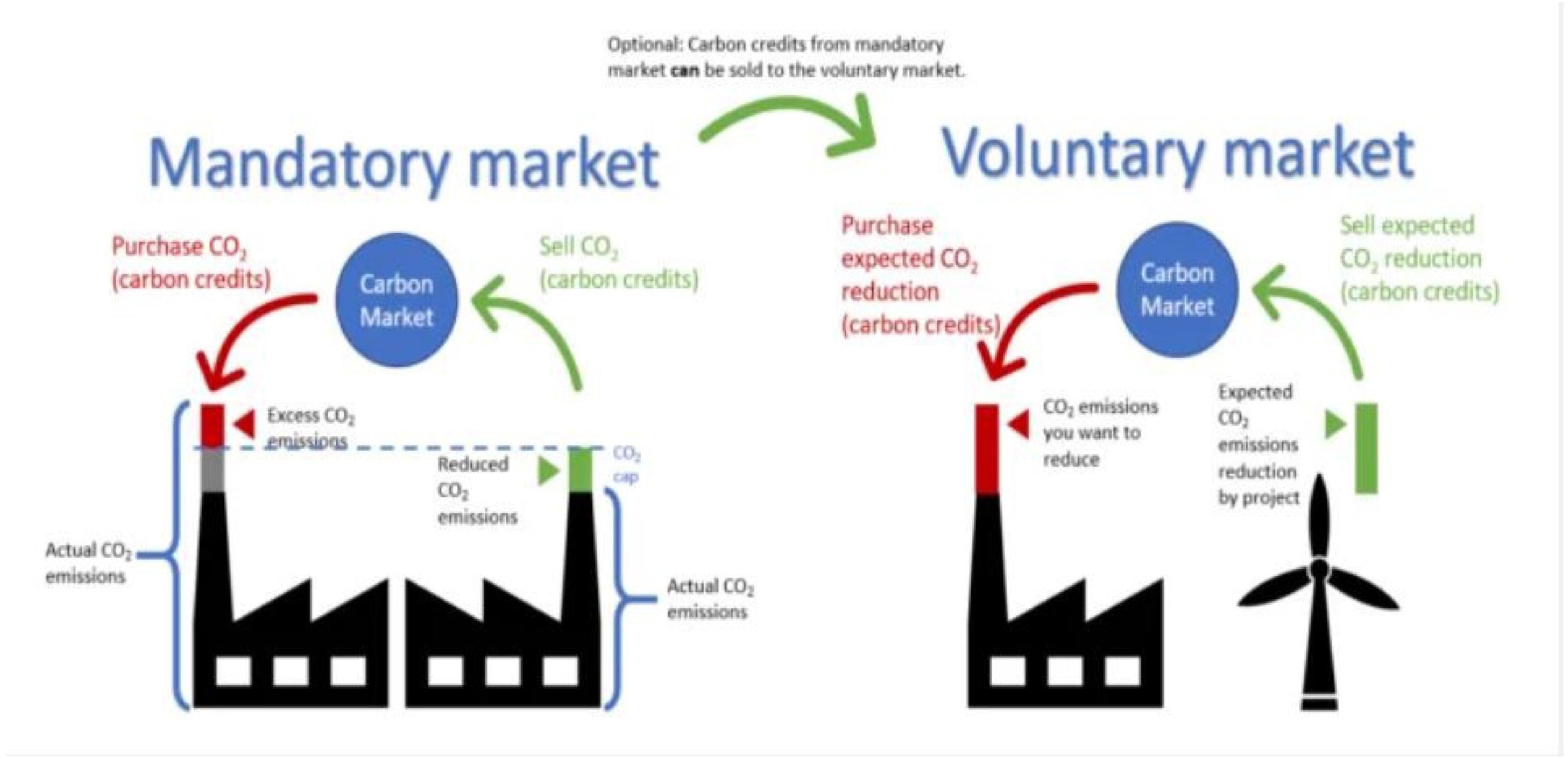
A DETOUR ON CARBON CREDITS

A carbon credit is a tradable certificate representing the right to emit a set amount of pollutant.

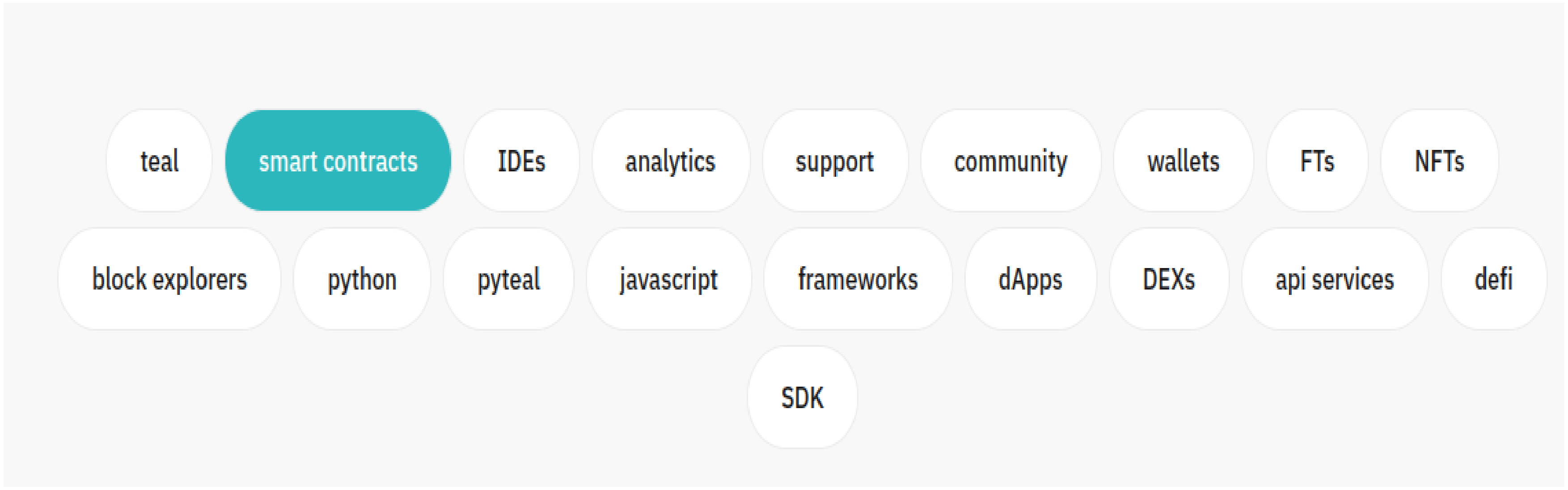

Carbon Offsets
Offsets represent emission reductions that have been achieved outside of the capped sector.
1 Company A needs to meet its emissions cap
2 Company A invests in an emission reduction project that produces carbon offsets
A
INVESTMENT


CREDITS
CARBON OFFSET
3 Company A receives carbon credits for its investment
One carbon credit = One tonne of greenhouse gas emission reductions
Carbon offsets programs can include:
• Reforestation
• Renewable energy
• Methane capture/combustion

CARBON CREDITS MARKET


Optional: Carbon credits from mandatory market **can** be sold to the voluntary market.
Mandatory market
Purchase CO_2 (carbon credits)
Carbon Market
Sell CO_2 (carbon credits)
Excess CO_2 emissions
CO_2 cap
Reduced CO_2 emissions
Actual CO_2 emissions
Actual CO_2 emissions
Voluntary market
Purchase expected CO_2 reduction (carbon credits)
Carbon Market
Sell expected CO_2 reduction (carbon credits)
CO_2 emissions you want to reduce
Expected CO_2 emissions reduction by project

ECOSYSTEM TOOLS & PROJECTS



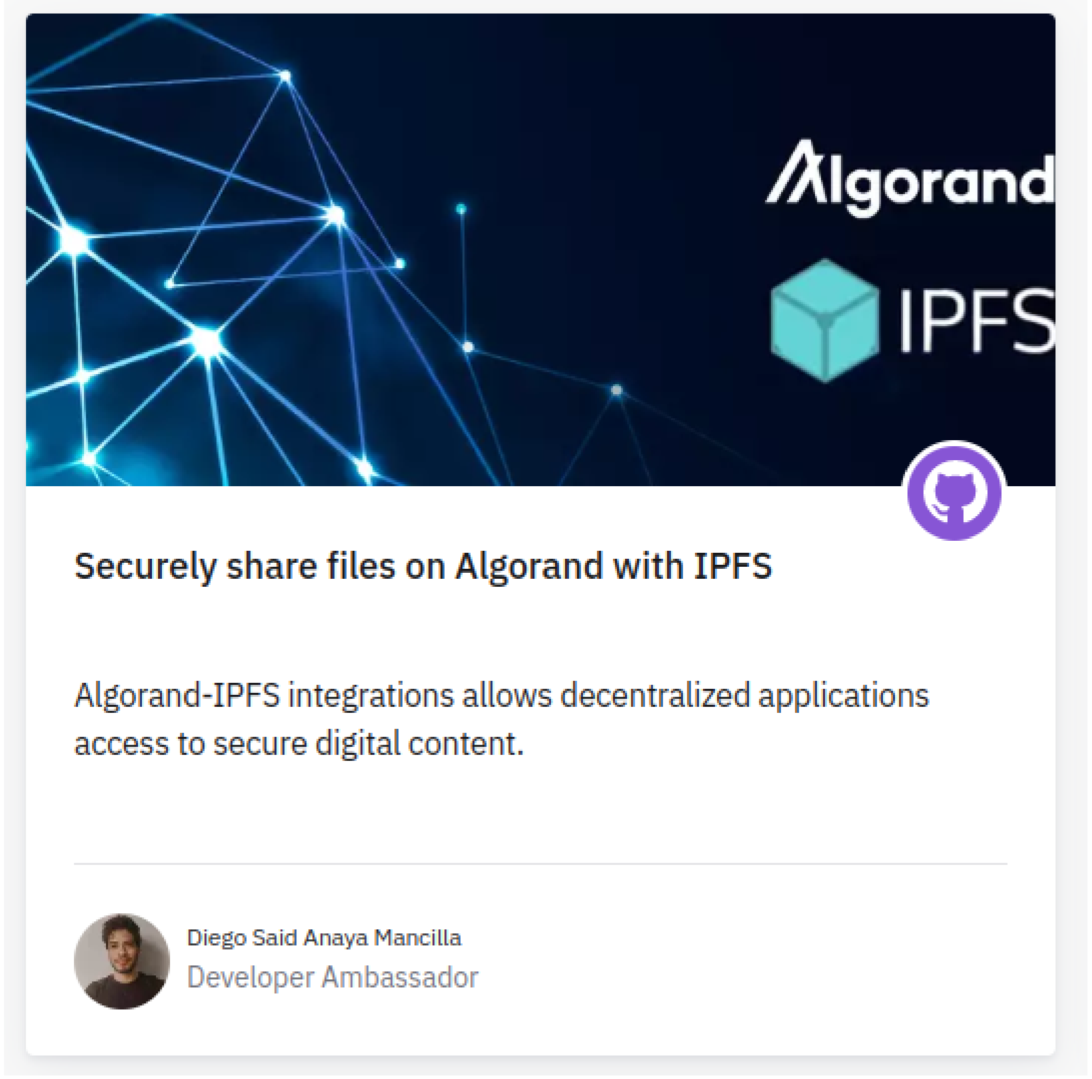
OTHER LAYER-1 CAPABILITIES

- Standard Assets.
    - Stablecoins, loyalty points, system credits, in-game points …
- Atomic Transfers: either all transactions succeed or all fail
    - Circular trades - Alice pays Bob if and only if Bob pays Claire if and only if Claire pays Alice.
    - Group payments - Everyone pays or no one pays.
    - Decentralized exchanges - Trade one asset for another without going through a centralized exchange.
    - Distributed payments - Payments to multiple recipients
    - Pooled transaction fees - One transaction pays the fees of others.
- Rekiing: Change Private Spending key without the need to change the Public Address

LET’S START WITH FUNDAMENTAL CHALLENGES

A way for developers to learn fundamental and advanced skills by actually coding. You’ll modify incomplete code snippets that perform key Algorand tasks …

CHECK THE DEVELOPER PORTAL FOR INETRESTING CHALLENGES/REQUESTS



Securely share files on Algorand with IPFS

Algorand-IPFS integrations allows decentralized applications access to secure digital content.

Diego Said Anaya Mancilla
Developer Ambassador

…even because at the end you have to publish there!

SMART CONTRACT TEMPLATES

- https://github.com/algorand/smart-contracts/tree/master/devrel
- https://github.com/scale-it/algo-builder/tree/master/examples

GUIDELINES

BEST PRACTICES @ ULAM LABS

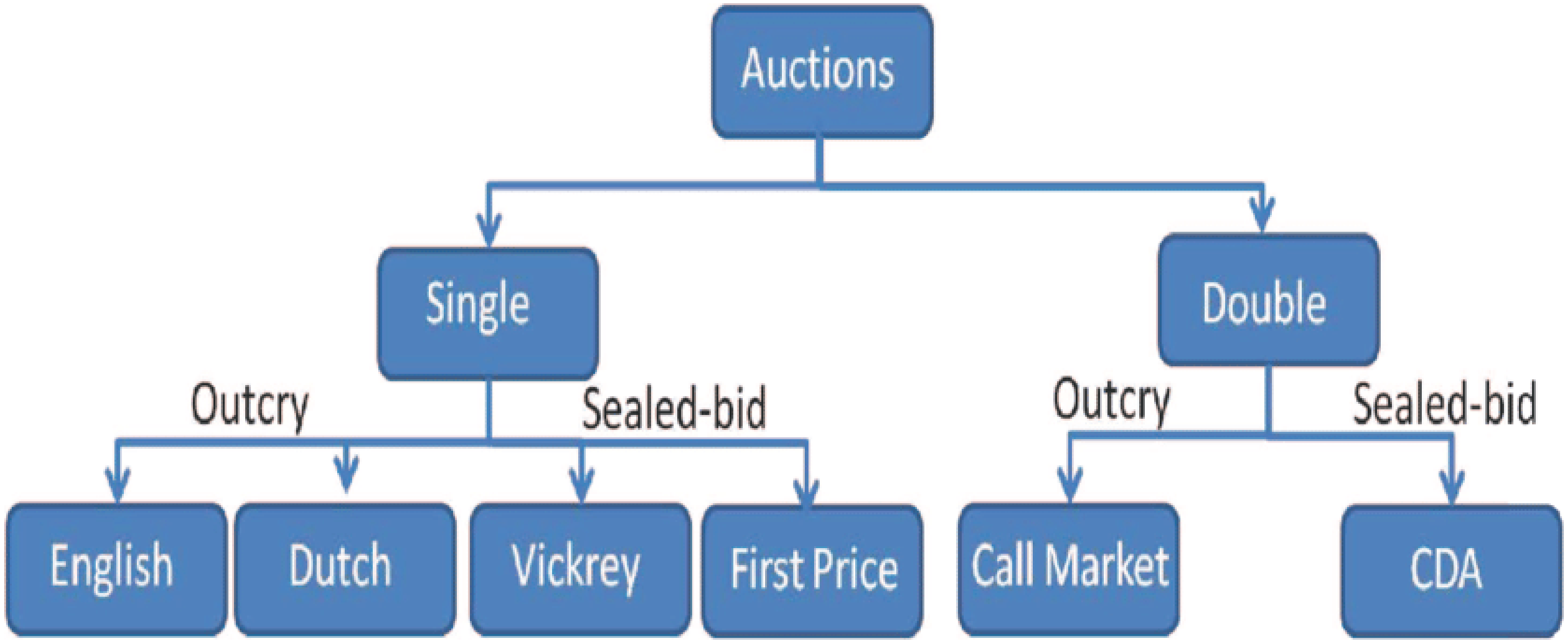
FEW PERSONAL SUGGESTIONS

1. COMPARE ALGORAND SMART CONTRACTS WITH OTHER PLATFORMS (E.G. ETHEREUM)

Smart signatures are limited, current smart contracts are very powerful

Consider auctions for example (Mogavero et al., 2021)

A POSSIBLE TAXONOMY OF AUCTIONS



Can we implement all these?

EXAMPLES OF ACUTIONS: NOTARIZED BIDDING

Bids are simply written (i.e., notarized) on the blockchain as soon as they are issued.

- nothing forces the auctioneer to assign the asset to the winning bidder;
- nothing forces the winning bidder to pay the bid;
- bidders might want instead to keep confidential their bids avoiding leaks that can be exploited by their competitors.

EXAMPLES OF ACUTIONS: DEPOSITED BIDDING

The winning bidder will obtain the asset in exchange for the amount of cryptocurrency deposited at bidding phase.

- confidentiality is still an issue

EXAMPLES OF ACUTIONS: COMMITTED BIDDING

This auction allows private bidding for sealed-bids auctions.

- the sealed envelopes can be opened only when necessary

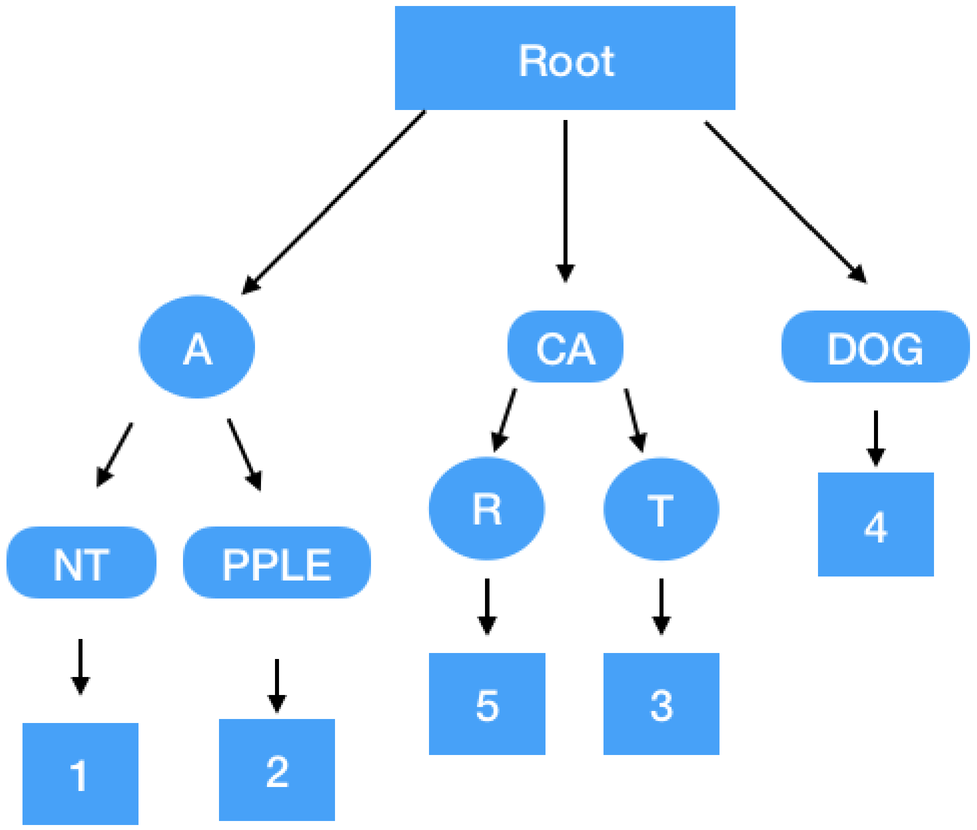
EXAMPLES OF AUCTIONS: CONFIDENTIAL BIDDING

In this auction, the winner is privately computed without revealing the offers of the other bidders. Each bidder reveals his/her bid only to the auctioneer, who exploits this information to compute and publicly declare the winner.

KEY INGREDIENTS

2. IMPLEMENT INTERESTING DATA STRUCTURES ON SMART CONTRACTS

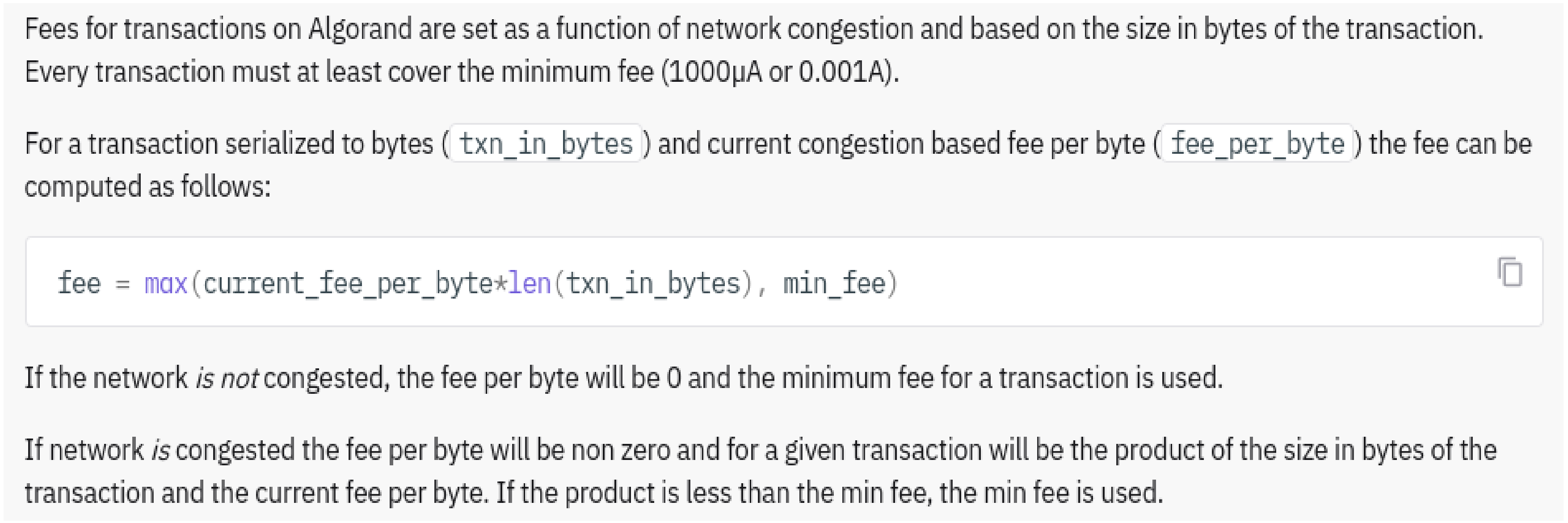
MERKLE PATRICIA TRIE



- Get, Put, Del, Verify Data Integrity
- They allow quickly calculating the new tree root after an update without recomputing the entire tree
- Verify the inclusion of a key-value pair without the access to the entire key-value pairs.

Nice Article on MPT

TRANSACTION FEES



Fees for transactions on Algorand are set as a function of network congestion and based on the size in bytes of the transaction. Every transaction must at least cover the minimum fee (1000µA or 0.001A).

For a transaction serialized to bytes (`txn_in_bytes`) and current congestion based fee per byte (`fee_per_byte`) the fee can be computed as follows:

```
fee = max(current_fee_per_byte*len(txn_in_bytes), min_fee)
```

If the network *is not* congested, the fee per byte will be 0 and the minimum fee for a transaction is used.

If network *is* congested the fee per byte will be non zero and for a given transaction will be the product of the size in bytes of the transaction and the current fee per byte. If the product is less than the min fee, the min fee is used.

APPROACH

1. Get Suggested Params
2. Recover Sender Account
3. Construct Transaction with Fee
4. Sign and Submit Transaction

more

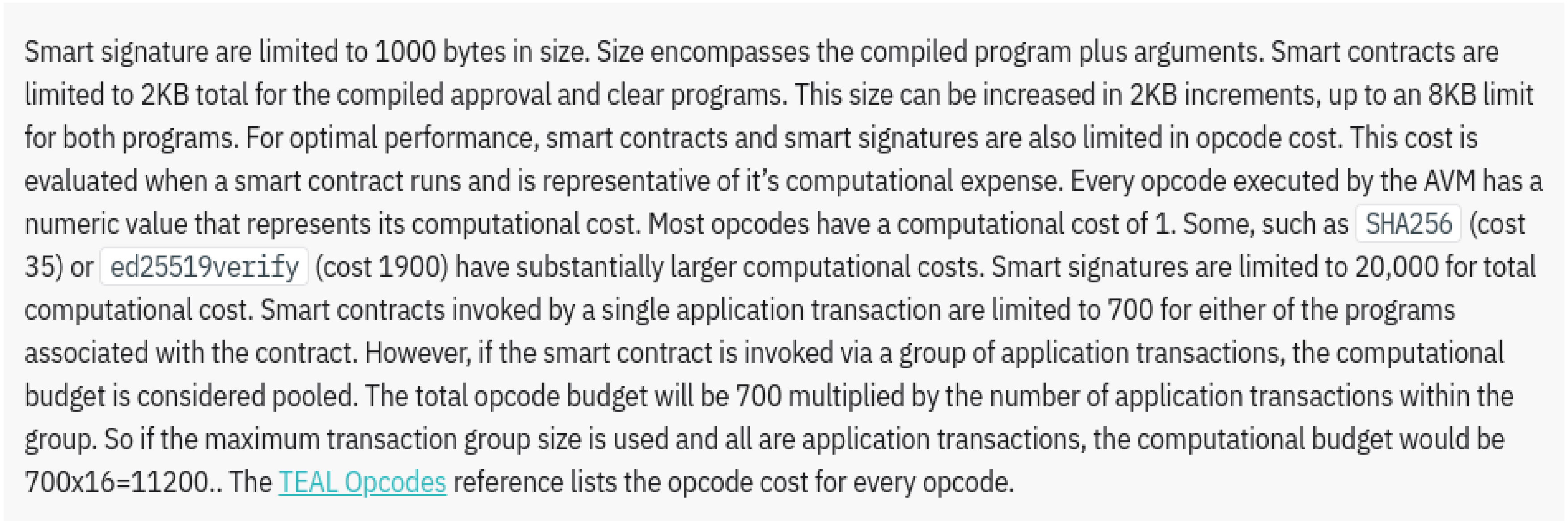
GET SUGGESTED PARAMS

AlgoExplorer
Algorand Blockchain Explorer

```
curl -X 'GET' \
  'https://node.algoexplorerapi.io/v2/transactions/params' \
  -H 'accept: application/json'
```

```
{
  "consensus-version": "https://github.com/algorandfoundation/
  "fee": 0,
  "genesis-hash": "wGHE2Pwdvd7S12BL5FaOP20EGYesN73ktiC1qzkkit8
  "genesis-id": "mainnet-v1.0",
  "last-round": 23632437,
  "min-fee": 1000
}
```

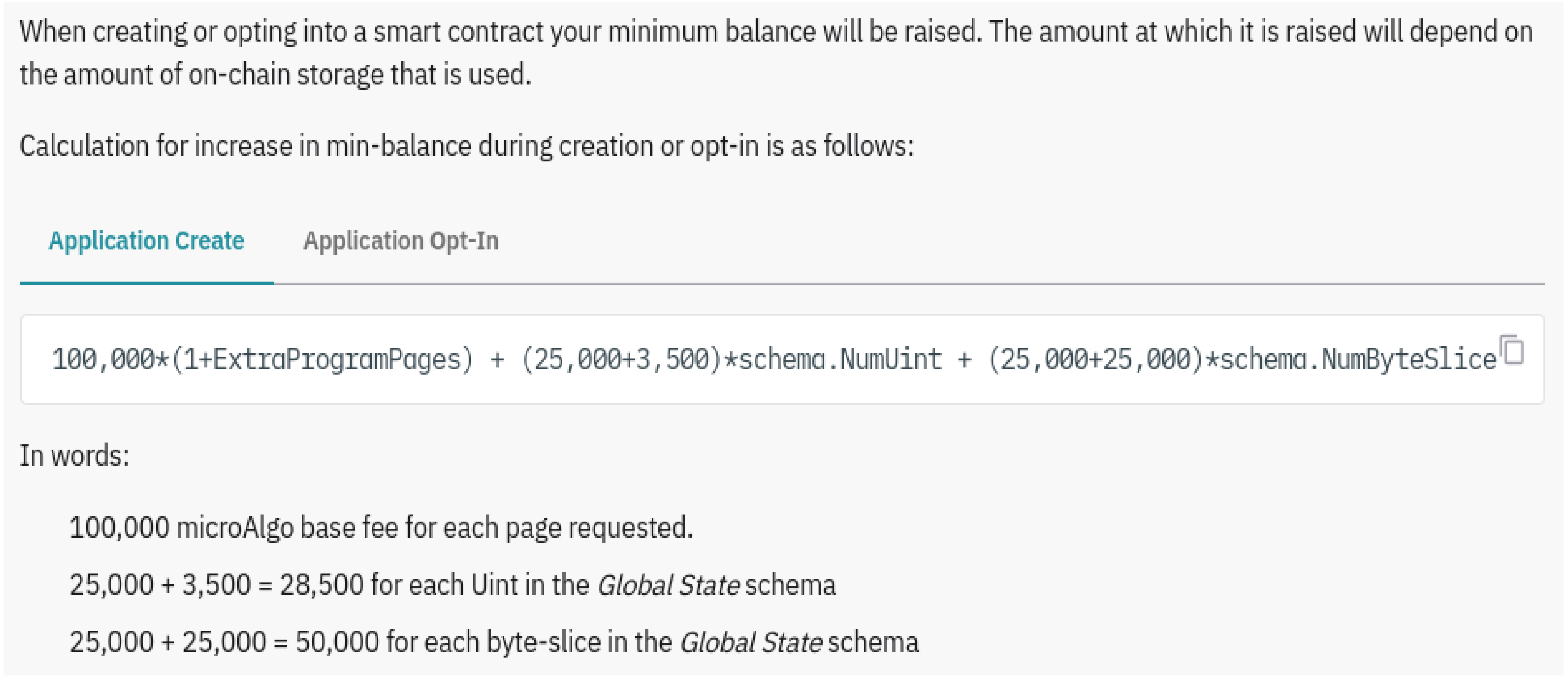
COST OF TEAL OPCODES



Smart signature are limited to 1000 bytes in size. Size encompasses the compiled program plus arguments. Smart contracts are limited to 2KB total for the compiled approval and clear programs. This size can be increased in 2KB increments, up to an 8KB limit for both programs. For optimal performance, smart contracts and smart signatures are also limited in opcode cost. This cost is evaluated when a smart contract runs and is representative of it's computational expense. Every opcode executed by the AVM has a numeric value that represents its computational cost. Most opcodes have a computational cost of 1. Some, such as `SHA256` (cost 35) or `ed25519verify` (cost 1900) have substantially larger computational costs. Smart signatures are limited to 20,000 for total computational cost. Smart contracts invoked by a single application transaction are limited to 700 for either of the programs associated with the contract. However, if the smart contract is invoked via a group of application transactions, the computational budget is considered pooled. The total opcode budget will be 700 multiplied by the number of application transactions within the group. So if the maximum transaction group size is used and all are application transactions, the computational budget would be 700x16=11200.. The TEAL Opcodes reference lists the opcode cost for every opcode.

Opcode Costs.

MINIMUM BALANCE REQUIREMENT FOR A SMART CONTRACT



When creating or opting into a smart contract your minimum balance will be raised. The amount at which it is raised will depend on the amount of on-chain storage that is used.

Calculation for increase in min-balance during creation or opt-in is as follows:

```
100,000*(1+ExtraProgramPages) + (25,000+3,500)*schema.NumUint + (25,000+25,000)*schema.NumByteSlice
```

In words:

- 100,000 microAlgo base fee for each page requested.
- 25,000 + 3,500 = 28,500 for each Uint in the *Global State* schema
- 25,000 + 25,000 = 50,000 for each byte-slice in the *Global State* schema

EVIDENCES THAT YOUR SMART CONTRACTS WORK AS EXPECTED

- Reference to relevant transactions AlgoExplorer Algorand Blockchain Explorer
- Automated testing for Algorand smart contracts

THANKS

QUESTIONS?