Algorand

AlgoNim

Let's play a crypto-Nim on Algorand

by cusma

What's Nim?

Algorand

Nim is a very simple mathematical game of strategy for two players. With a lot of imagination let's name them Alice and Bob. Just to be fair from the very beginning: Nim is a **zero-sum game** and has been **"mathematically solved"**, this means that exists an "easily calculated" perfect strategy to determine which player will win and what winning moves are open to that player.

So if Alice is a computer, Bob better avoid betting on winning the game.

What's AlgoNim?

Algorand

AlgoNim is a **cryptographic version of Nim** that **runs completely on Algorand Layer 1**, on the Pure PoS consensus protocol, so nobody can cheat. The game has been implemented using Algorand **Python SDK** + **Stateless TEAL** (PyTEAL).

On the current development stage, players play AlgoNim using the command line on their nodes: a **Dealer** (Alice) sets up the match and the **Opponent** (Bob) joins the game.

Further development could let players play on a web browser.

TEAL ASC1s programmatically generation

Algorand

Through the **seamless interaction** of Algorand **Python SDK** and **PyTeal**, AlgoNim **automatically writes** and **initializes** a **dedicated set of stateless TEAL Algorand Smart Contracts 1** and **Algorand Standard Assets** for each match, generating a new ASA + ASC1 architecture ready to be played.

The whole match set-up takes few seconds and costs to the Dealer less then 1 ALGO. **This time/cost performance is quite impressive if compared to other blockchains**.

AlgoNim rules

Algorand

AlgoNim is based on Nim's "normal" variant. Rules are trivial:

1. Alice chooses the number n of pieces to put on the game table for the match;
2. Alice chooses the number m of pieces that can be removed at the most from the game table in each turn;
3. Alice and Bob choose who moves first;
4. On each turn each player removes at least 1 and at the most m pieces from the game table;

Who removes the last piece form the table wins the game!

AlgoNim rewards

Alice and Bob may choose betting some ALGO for the match. AlgoNim ASC1 architecture take cares of betting security.

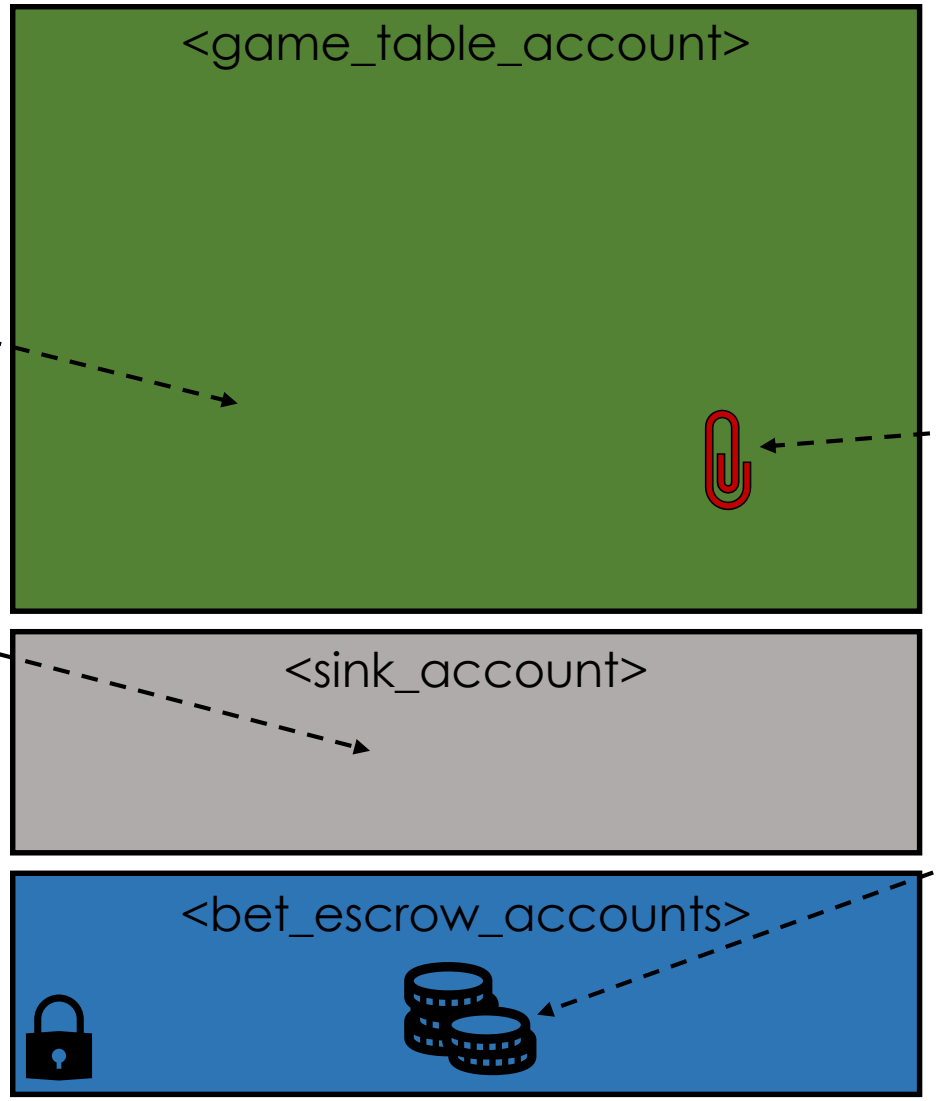
In further developments AlgoNim may reward the winner with **AlgoNim ASA Scores** too, this will enable an AlgoNim global ranking!

AlgoNim ASA + ASC1 architecture

Algorand

The **game table** is represented by a **TEAL account**.

A pieces' **sink** is represented by a different **TEAL account**.

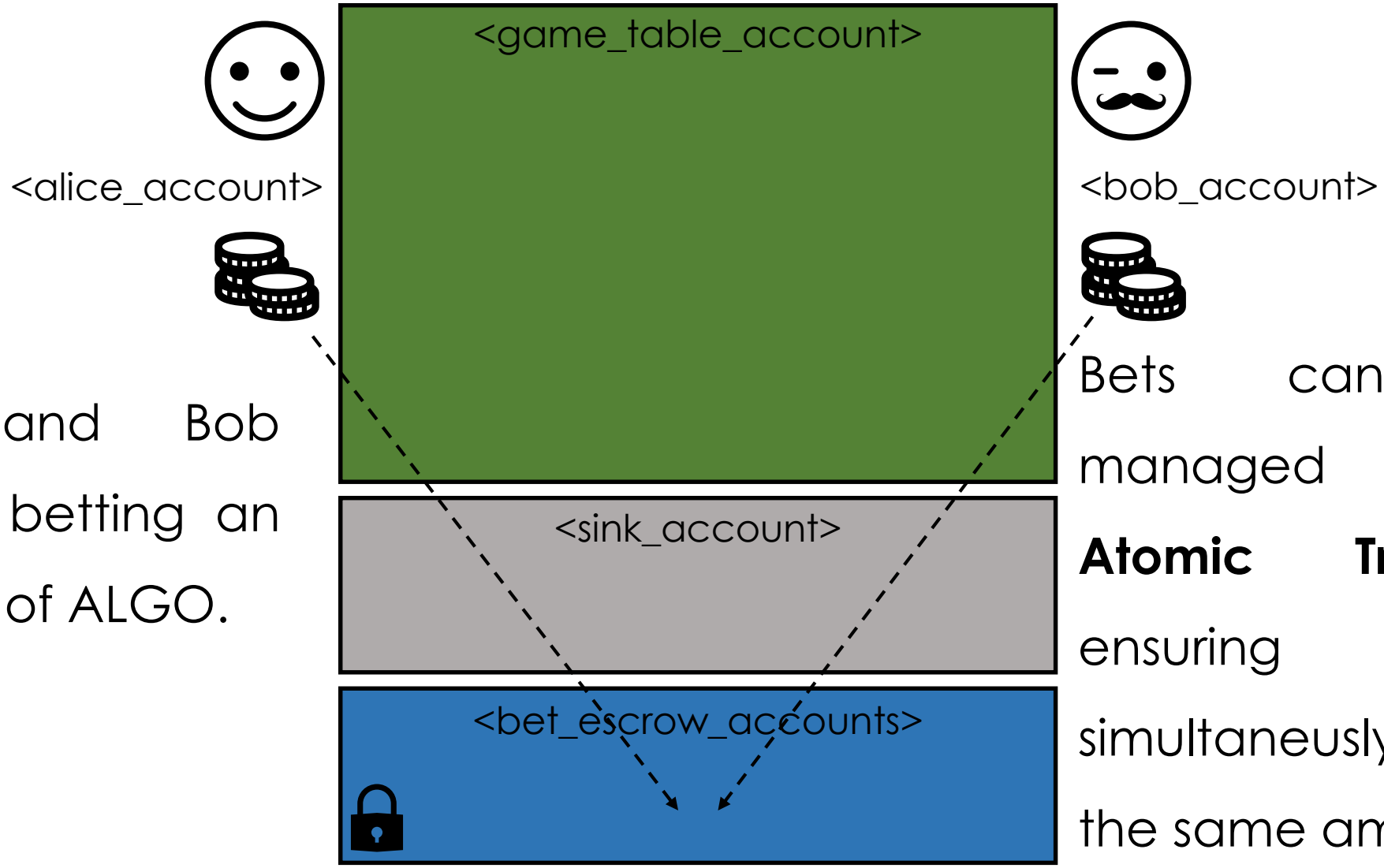


The **pices «on the table»** are represented by **ASAs guarded in the game table TEAL account**.

Bets are locked in **TEAL escrow accounts**.

How and what do we bet?

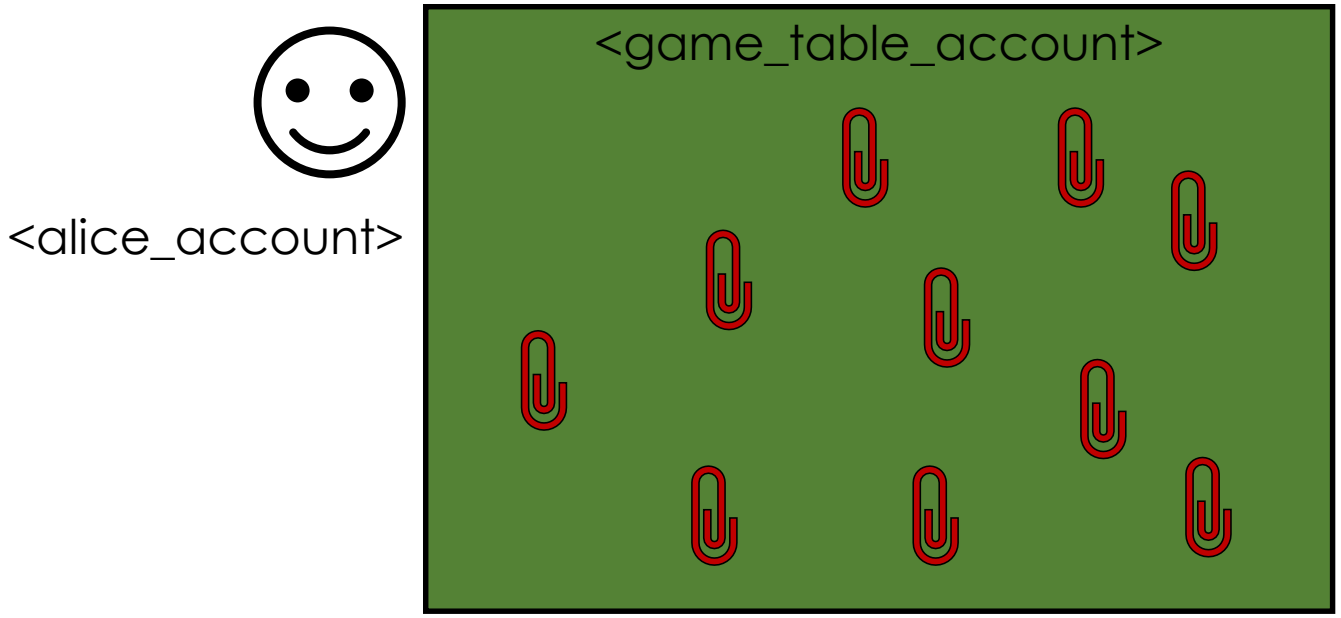
Algorand



Alice and Bob choose betting an amount of ALGO.

Bets can be managed by an **Atomic Transfers** ensuring players simultaneusly bet the same amount.

AlgoNim, match set up

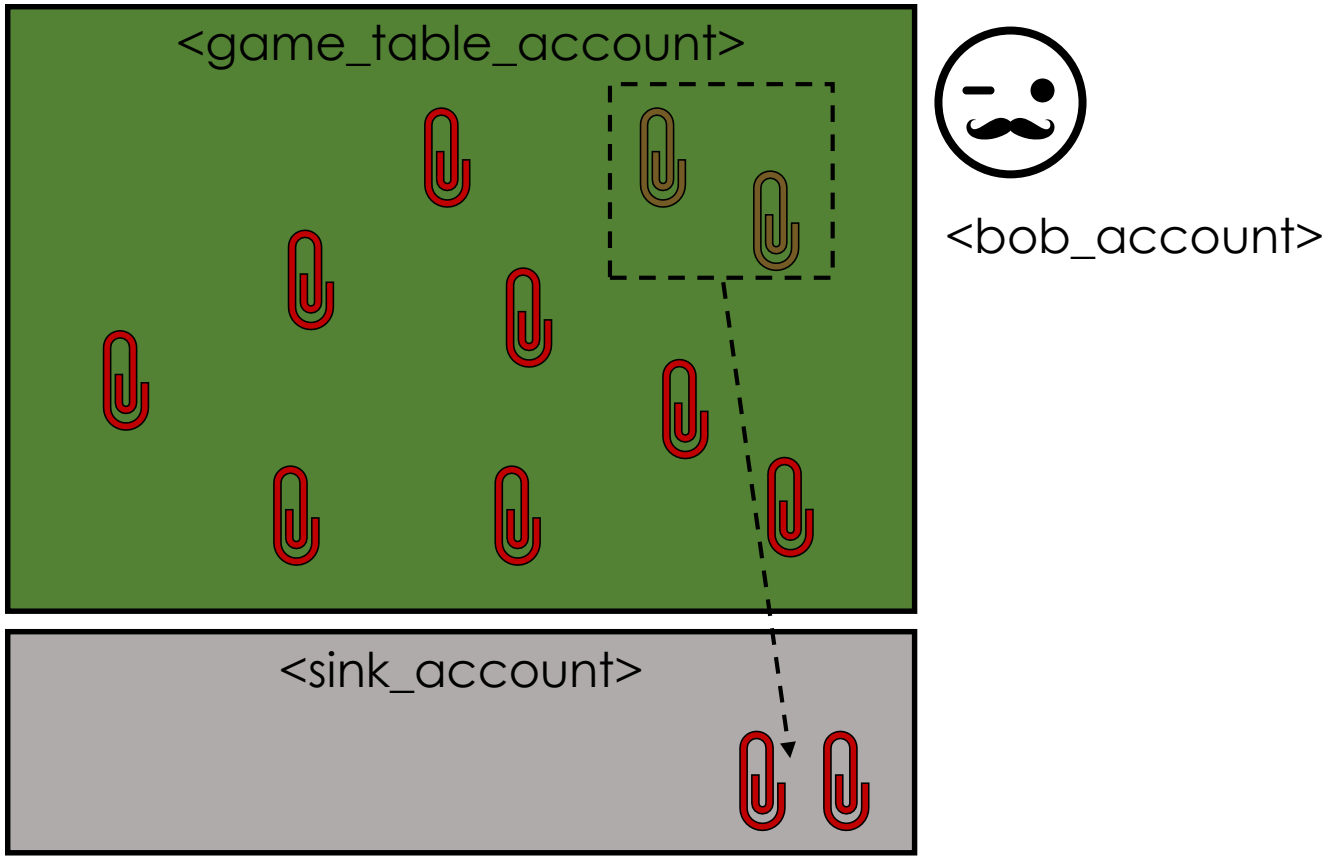


The Dealer (Alice) decides to play this match with n pieces on the game table, so the **game initialization AT** generates n **ASA pieces** and transfer them to the **game table TEAL account**.

Moving pieces from the table to the sink

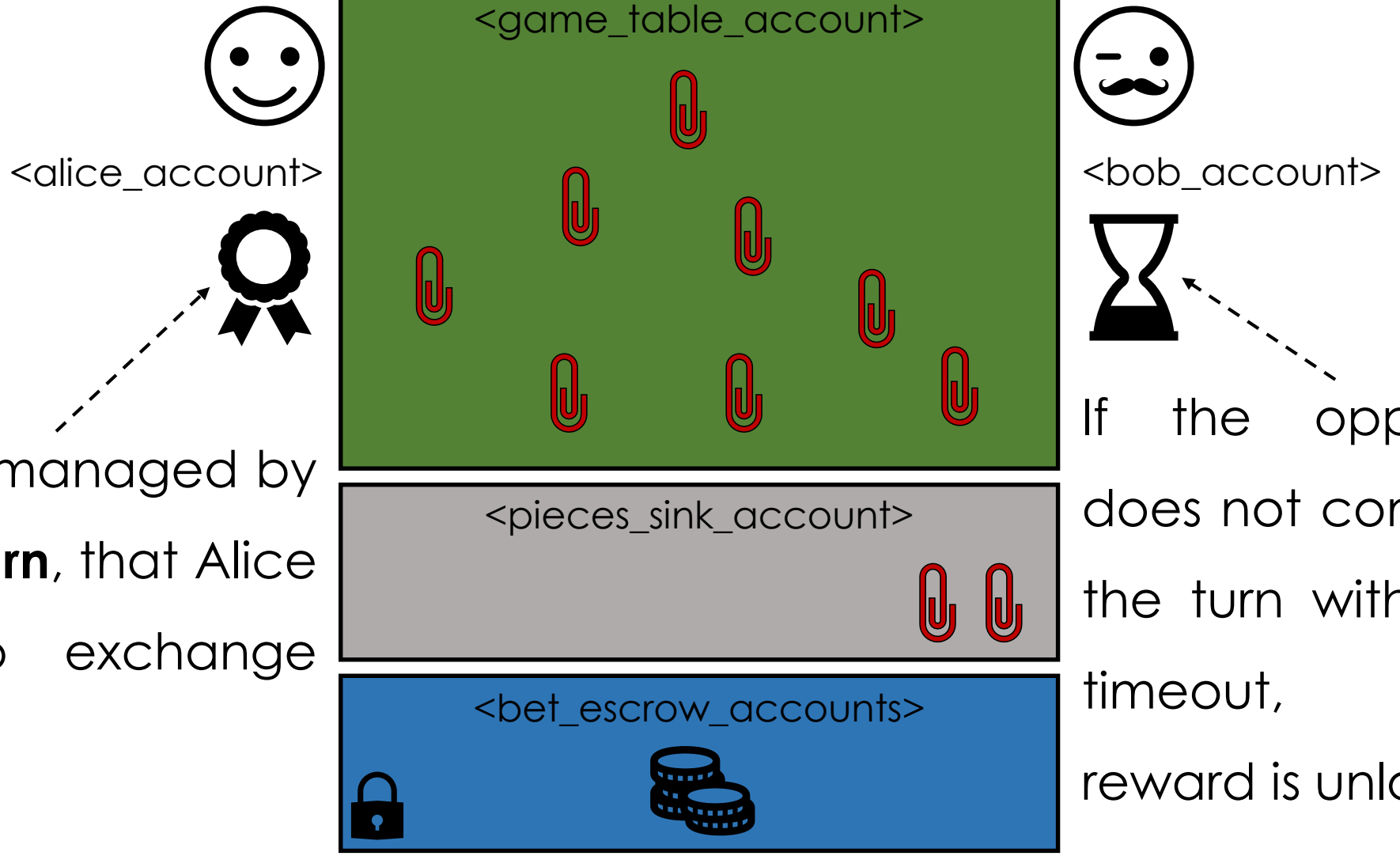
Algorand

The Dealer decides that at the most m pieces can be removed from the game table in each turn for this match.



The **game table TEAL account** will limit the amount of ASA pieces to be transferred to **TEAL sink account** to m.

Turns

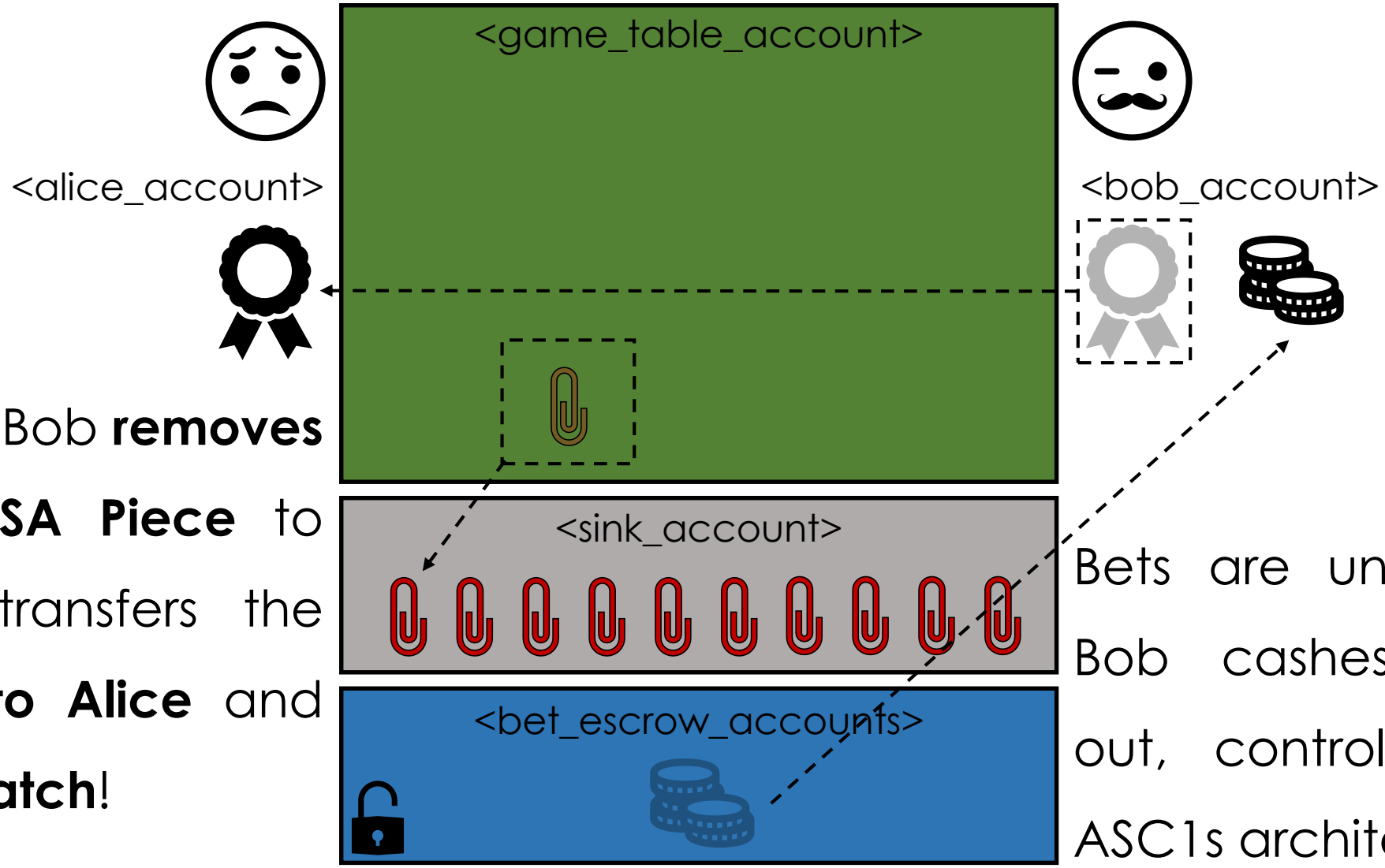


Turns are managed by an **ASA Turn**, that Alice and Bob exchange over time.

If the opponent does not complete the turn within the timeout, the reward is unlocked.

Winning the match!

Algorand

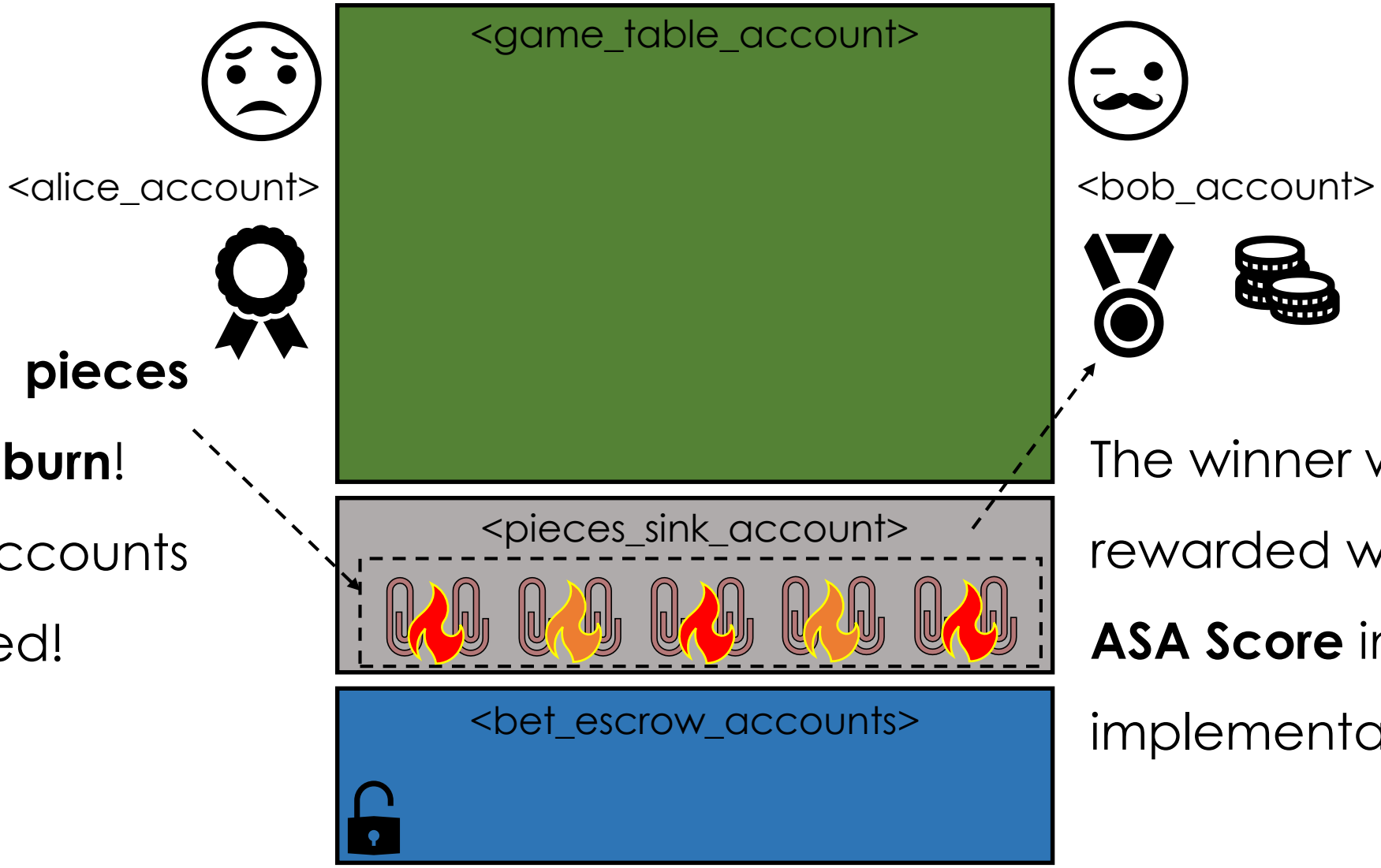


With an **AT** Bob **removes the last ASA Piece** to the sink, transfers the **ASA Turn to Alice** and **wins the match**!

Bets are unlocked, Bob cashes them out, controlled by ASC1s architecture.

Destroying useless ASAs

Algorand



Useless **pieces**

ASA are burn!

Empty accounts are closed!

The winner will be rewarded with an **ASA Score** in further implementations.

AlgoNim implementation

Algorand

Step 1: the Dealer chooses match's set-up

The Dealer (Alice) chooses the number of ASA Pieces to play with, the maximum ASA Pieces' to be removed on each turn from the Game Table account to the Sink account as well as the Match Duration that determines the Bet Escrows expirys. The Dealer declares Opponent's (Bob) public address for this match. The Opponent (Bob) must then join the Match accepting the bet proposal and Opt-In both for ASA Pieces and ASA Turn.

AlgoNim implementation

Algorand

Step 2: AlgoNim ASA Pieces creation

ASSET NAME: AlgoNim Piece

UNIT NAME: ALGONIMP

DECIMALS: 0

TOTAL SUPPLY: n (choose by the Dealer)

URL: https://github.com/cusma/algonim

CREATOR: <dealer_account>

ASSET ID: ALGONIMP_ID

DEFAULT FREEZE: no

AlgoNim implementation

Algorand

Step 2: AlgoNim ASA Turn creation

ASSET NAME:	AlgoNim Turn
UNIT NAME:	ALGONIMT
DECIMALS:	0
TOTAL SUPPLY:	1
URL:	https://github.com/cusma/algonim
CREATOR:	<dealer_account>
ASSET ID:	ALGONIMT_ID
DEFAULT FREEZE:	No (or all except Alice and Bob in future)

AlgoNim implementation

Algorand

Step 3: Sink TEAL Account creation

LOGIC: <sink_account> will accept only ALGONIMP deposits as part of a specific **Play Turn Atomic Transfer** and for specific ASA Asset ID, ensuring that for this match the Sink will only accpet the ASA Pieces just created by the Dealer in the match's set up.

AlgoNim implementation

Algorand

Step 4: Game Table TEAL Account creation

LOGIC: <game_table_account> can be funded once with the entire supply of the specific ASA Pieces created by the Dealer in the match's set up. The <game_table_account> allows only <dealer_account> or <opponent_account> to transfer its ALGONIMP to <sink_account>, whitin a specific **Play Turn Atomic Transfer**, ensuring that no more than m ASA Pieces are removed each turn and that the players exchange the ASA Turn to make their moves.

AlgoNim implementation

Algorand

Step 5: Bet Escrow TEAL Accounts creation

LOGIC: the <dealer_account> sets up Bet Escrow TEAL Accounts for betting rewards. The Escrows can be unlocked only by the winner with **Play Last Turn Atomic Transfer** or by their respective owners in case of countdown expiring. Bets are managed by the **Bet Atomic Transfer** ensuring each player bets the same amount proposed by the Dealer at the same time.

AlgoNim implementation

Algorand

Play Turn Atomic Transfer (by Dealer or Opponent)

The player that owns the **ASA Turn** moves. On each turn the players issue a **Play Turn Atomic Transfer**:

Play Turn Atomic Transfer structure

1. Asset Send of **ASA Turn** to other player;
2. Asset Send of an amount P $(1 \leq P \leq m)$ of **ASA Pieces** from the **Game Table Account** to **Sink Account**;

AlgoNim implementation

Algorand

Play Last Turn Atomic Transfer (by Dealer or Opponent)

Once a player can transfer the **last ASA Piece** from the **Game Table Account** to the **Sink Account** he/she can try submitting a **Play Last Turn Atomic Transfer**, claiming the victory**.** In case the victory claim AT is authorized the winner gains the betting rewards locked in the opponent's escrow. In future implementations the ASA Pices could be automatically burn and the empty Game Table and Sink Accounts colsed.

AlgoNim implementation

Algorand

Play Last Turn Atomic Transfer structure

1. Last Asset Send of an amount P $(1 \leq P \leq m)$ of **ASA Pieces** from the **Game Table Account** to **Sink Account**;
2. Asset Send of **ASA Turn** to **Opponent Account**;
3. Asset Send of **ASA Pieces total supply** from Sink Account **to winner account**;
4. Close **Bet Escrow Accounts** claiming the betting rewards;

AlgoNim implementation

Play Last Turn Atomic Transfer structure

In future implementations could be included

5. Asset Destory of **ASA pieces**;

6. Colse **game table account** and **sink account**;

AlgoNim implementation

Algorand

Bet Escrows countdown expires

If one of the player does not act for a long time the Bet Escrows countdown condition is triggered and both Alice and Bob can close their Bet Escrow Accounts and claim their own bets back.

AlgoNim future implementations

Algorand

1. Improve the robustness of Bet Escrows (preventing players to stop the game in the middle);
2. Freeze the match's ASAs for anyone but the players;
3. Automatically destroy the match's ASAs;
4. Introduce ASA AlgoNim Score in the Sink as reward for the winner.