



عنوان ارائه:

بررسی ساختار تراکنش‌ها در رمزارز کاردانو و مقایسه آن با بیتکوین و اتریوم

A Review on Cardano Transaction System & Compare it with Bitcoin and Ethereum

توسط: علیرضا صادقی نسب

استاد: دکتر وحید رافع

تاریخ ارائه: 1400/08/24

فهرست مطالب

- مقدمه
- مدل UTXO
- مدل Account/Balance
- مفهوم تراکنش
- مدل E-UTxO

مقدمه

▪ رمزارز کاردانو

★ کاردانو یک بستر زنجیره بلوکی عمومی، متن باز و غیرمتمرکز است

★ با اجماع به دست آمده با استفاده از proof of stake، می تواند تراکنش های p2p را با ارزش داخلی خود (Ada) تسهیل ببخشد

★ این پلتفرم در سال ۲۰۱۵ توسط چارلز هاسکینسون، یکی از هم بنیانگذاران اتریوم تاسیس شد

★ کاردانو بزرگترین رمزارزی است که از proof of stake استفاده می کند. پروتکلی که جایگزین سبتری نسبت به پروتکل های proof of work است



مقدمه

▪ رمز ارز کاردانو – ادامه

★ در پلتفرم کاردانو، Ada در لایه استقرار (SL) قرار دارد. این لایه شبیه به بیتکوین است و تراکنش‌ها را رصد می‌کند. لایه دوم، لایه محاسباتی (CL) است. این لایه به گونه‌ای طراحی شده است که شبیه اتریوم باشد و اجازه دهد تا قراردادهای و برنامه‌های کاربردی هوشمند بر روی آن اجرا شوند.

★ کاردانو در سال ۲۰۲۱، زبان Plutus را معرفی کرد. این زبان یک زبان هوشمند تورینگ کامل است که به زبان Haskell و Marlowe که یک زبان تخصصی قرارداد هوشمند است؛ نوشته شده است. زبان Marlowe زبانی است که برای غیر برنامه‌نویسان در بخش مالی، طراحی شده است



مقدمه

▪ پروتکل های proof of stake

★ این پروتکل ها دسته ای از مکانیزم های اجماع برای زنجیره های بلوکی هستند که با انتخاب اعتباردهنده ها متناسب با تعداد دارایی شان در ارز دیجیتال مرتبط، کار می کند.

★ برخلاف پروتکل های PoW، سیستم های PoS مصرف انرژی زیادی ندارند

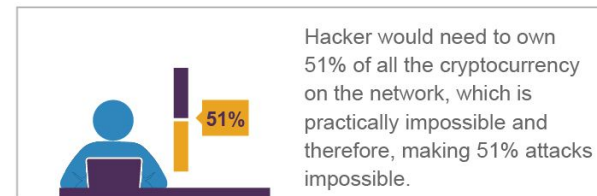
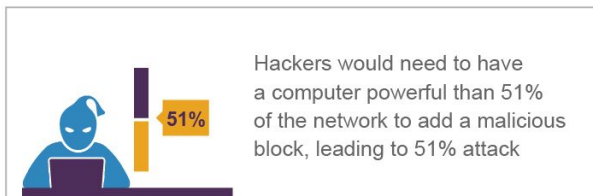
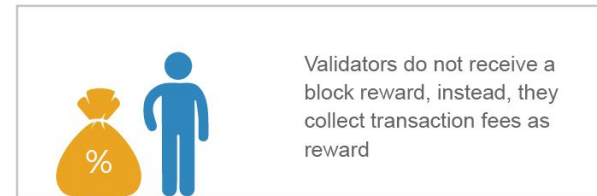
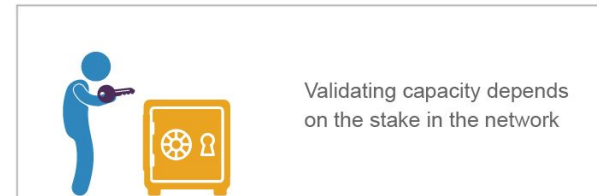
★ اولین استفاده کننده کاربردی از این پروتکل ها، رمزارز PeerCoin در سال ۲۰۱۲ بود. بزرگترین زنجیره بلوکی PoS براساس ارزش بازار، رمزارز کاردانو است

★ نحوه ایمن ماندن زنجیره بلوکی از تصاحب اکثریت اعتبارسنجی توسط افراد مخرب در PoS به این صورت است که اعتبارسنج ها باید مقداری توکن زنجیره بلوکی را داشته باشند بنابراین فرد مهاجم می بایست بخش بزرگی از توکن های موجود در زنجیره بلوکی را به دست آورد. این قاعده در PoW توسط قدرت محاسباتی شبکه انجام می شود که این امر نیاز به مصرف مقدار بسیار زیادی انرژی است

مقدمه

PoW vs PoS

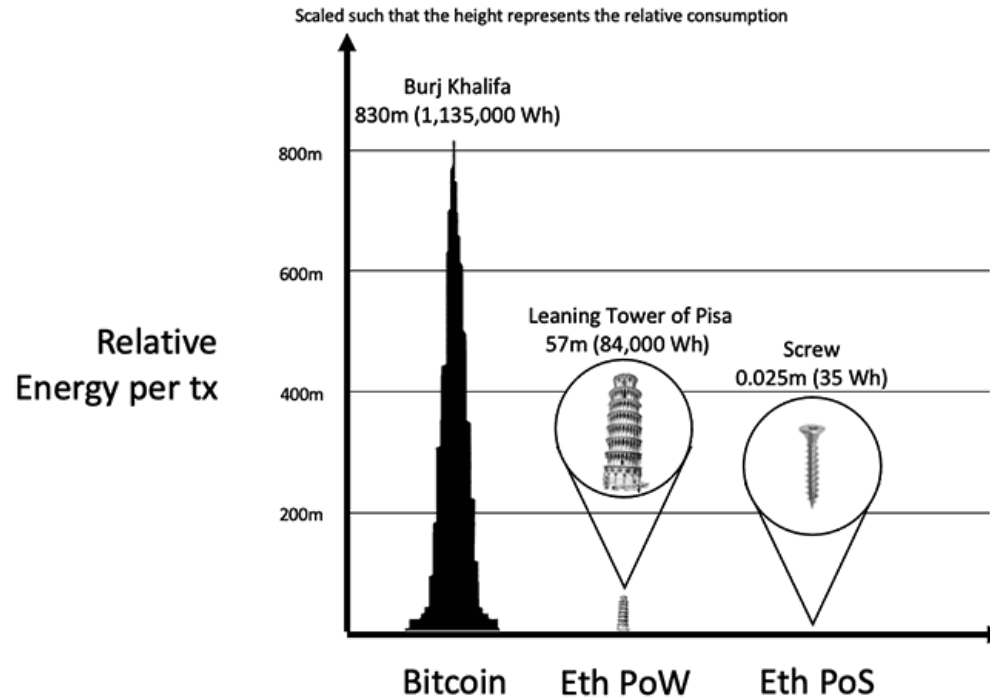
Proof of Work VS Proof of Stake



مقدمه

PoW vs PoS

Relative energy consumption per transaction



مدل UTXO

▪ مدل UTXO

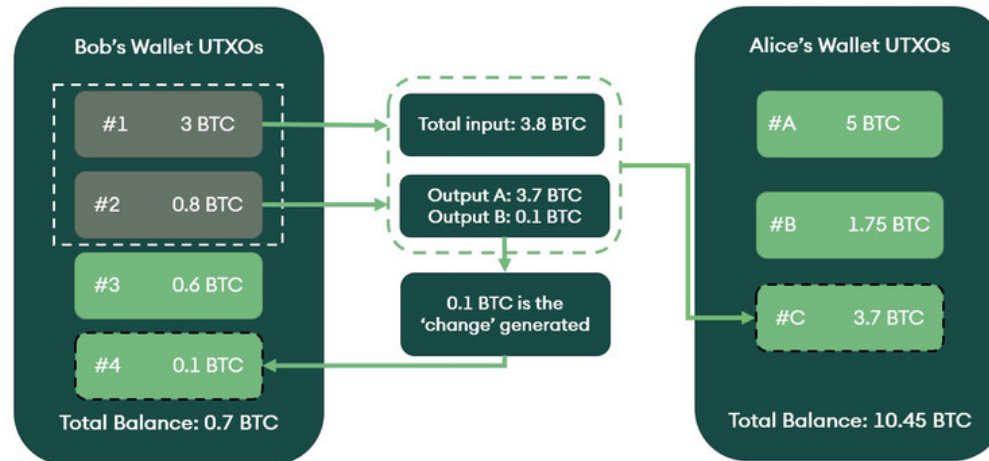
★ در یک مدل UTXO، حرکت دارایی‌ها به شکل یک گراف جهت‌دار غیر چرخه‌ای (DAG) ثبت می‌شود که در آن، گره‌ها تراکنش هستند و یال‌ها، خروجی تراکنش هستند که در آن هر تراکنش اضافی، مقداری از UTXOها را مصرف می‌کند و موارد جدید را اضافه می‌کند

★ کیف پول کاربران، لیستی از خروجی‌های مصرف نشده مرتبط با تمام آدرس‌های متعلق به کاربر را پیگیری می‌کند و موجودی کاربران را محاسبه می‌کند. این مدل از بسیاری جهات شبیه پول نقد است. هر موجودی که در کیف پول زنجیره بلوکی هست را می‌توان با ترکیبات مختلف UTXO براساس تراکنش‌های قبلی ایجاد کرد اما مقدار موجودی ثابت باقی می‌ماند؛ به عبارت دیگر، موجودی موجود در یک آدرس کیف پول معین، مجموع تمام UTXOهای خرج نشده از تراکنش‌های قبلی است

مدل UTXO

مفهوم تغییر در مدل های UTXO

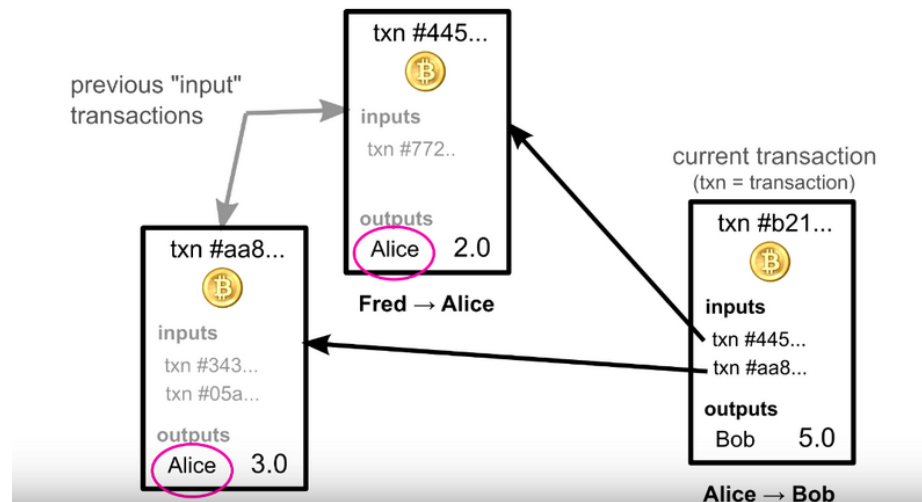
* مانند خرید نقدی در یک فروشگاه که نمی توان برای مثال یک ۵۰ دلاری را به قطعات کوچکتر تبدیل کرد، همین رویه برای UTXO نیز صادق است. نمی توان UTXO را به بیت های کوچکتر شکافت. تمام UTXO ها استفاده می شوند و باقی مانده آن به صورت یک UTXO کوچکتر دیگر، وارد کیف پول می شود.



مدل UTXO

مزیت مدل های UTXO

* با امکان بررسی و ردیابی اندازه، سن و مقدار UTXOهایی که انتقال می یابند، می توان معیارهای دقیقی در مورد نحوه استفاده زنجیره بلوکی و فعالیت مالی زنجیره استخراج کرد. مقیاس پذیری و حریم خصوصی از دیگر مزایای این مدل هستند. همچنین، منطق تراکنش های نیز ساده شده است زیرا هر UTXO فقط یک بار و به صورت کلی می تواند مصرف شود. این مهم موجب می شود تا تایید تراکنش بسیار ساده تر شود.




مدل Account/Balance

▪ مدل Account/Balance

- ★ مدل های زنجیره بلوکی که بر روی یک مدل A/B استقرار می یابند، از یک حساب (که می تواند توسط یک کلید خصوصی یا یک قرارداد هوشمند کنترل شود) برای نگهداری موجودی سکه استفاده می کنند. در این مدل، دارایی های به عنوان موجودی در حساب های کاربران نشان داده می شوند و مانده ها به عنوان یک وضعیت کلی از حساب ها ذخیره می شوند. مانده ها توسط گره ها ذخیره می شوند و با هر تراکنش به روز می شوند
- ★ از بسیاری جهات، زنجیره های A/B به روشی مشابه حساب های بانکی سنتی عمل می کنند. موجودی کیف پول هنگام واریز سکه ها افزایش می یابد و زمانی که سکه ها به جای دیگری منتقل می شوند، کاهش می یابد. تفاوت اساسی در اینجا این است که برخلاف UTXO ها، می توان قسمتی از مانده را استفاده کرد

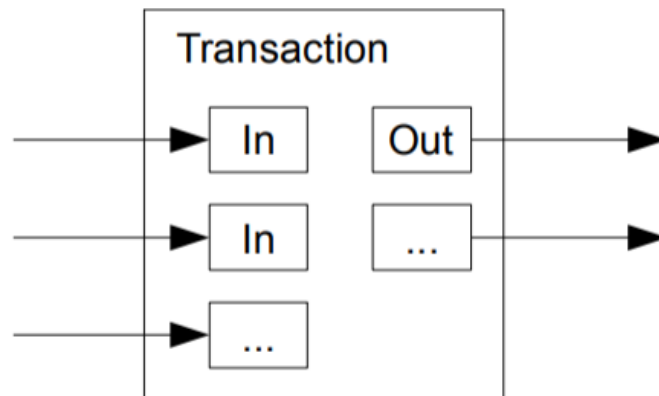
Account-based blockchain transaction

Alice's balance	Transaction	Bob's balance
\$10	From: Alice	\$3
-\$4	To: Bob	+\$4
	Amount: \$4	
	Signature: 	

مفهوم تراکنش

تراکنش‌ها؛ ورودی‌ها و خروجی‌ها

* در یک محیط زنجیره بلوکی، هر تراکنش می‌تواند یک یا چند ورودی و یک یا چند خروجی داشته باشد. ابتدا برای درک اینکه یک تراکنش چگونه کار می‌کند و چگونه با UTXOها ارتباط دارد، باید مفاهیم ورودی و خروجی را درک کرد. به صورت انتزاعی می‌توان گفت که یک تراکنش، عملیاتی است که در آن، خروجی‌های قبلی را باز (unlock) می‌کند و خروجی‌های جدیدی را ایجاد می‌کند



مفهوم تراکنش

▪ خروجی تراکنش‌ها

- * خروجی تراکنش شامل یک آدرس (می‌توان در نظر گرفت که یک قفل است) و یک مقدار است. مطابق با این قیاس انجام شده، امضایی که به این آدرس تعلق دارد، کلیدی است که جهت باز کردن قفل خروجی به کار گرفته می‌شود. پس از باز شدن قفل، یک خروجی می‌تواند به عنوان یک ورودی استفاده شود. تراکنش‌های جدید خروجی‌های تراکنش‌های قبلی را خرج می‌کنند و خروجی‌های جدیدی را تولید می‌کنند که می‌تواند توسط معاملات آینده، مصرف شود
- * هر خروجی در مدل UTXO شامل مقدار وجه ارسال شده و همچنین آدرس مقصد است که فقط یک هَش BLAKE2b-256 از کلید عمومی آدرس‌هاست.

مفهوم تراکنش

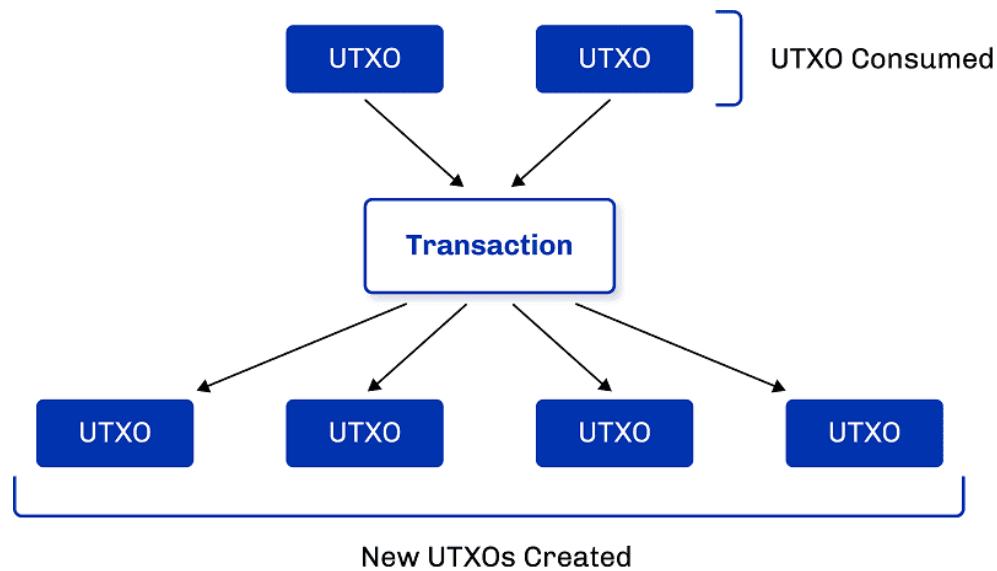
▪ ورودی تراکنش‌ها

- * ورودی تراکنش خروجی تراکنش قبلی است. ورودی‌های تراکنش شامل یک اشاره‌گر و یک امضای رمزنگاری است که به عنوان کلید باز کردن قفل عمل می‌کند. اشاره‌گر به خروجی تراکنش قبلی اشاره می‌کند و کلیدی است که این خروجی را می‌گشاید. هنگامی که یک خروجی توسط یک ورودی باز می‌شود، زنجیره بلوکی خروجی قفل نشده را به عنوان “خرج شده” علامت‌گذاری می‌کند. خروجی‌های جدید ایجاد شده توسط یک تراکنش را می‌توان با ورودی‌های جدید نشان داد و با همین رویه، زنجیره توالی می‌یابد. این خروجی‌های جدید که هنوز قفل نشده‌اند (خرج شده‌اند) همان UTXO ها هستند
- * هر ورودی در مدل UTXO، یک شناسه تراکنش دارد که هَش BLAKE2b-256 تراکنش و یک index خروجی‌هایی است که از تراکنش استفاده می‌کنند.

مفهوم تراکنش

▪ نحوه کار کردن مدل UTXO

* در یک مدل UTXO، تراکنش‌ها خروجی‌های خرج نشده‌ای را مصرف می‌کنند که از تراکنش‌های قبلی دریافت کرده‌اند و در نهایت، خروجی‌های جدیدی را به وجود می‌آورند که به عنوان ورودی تراکنش‌های آتی می‌توانند مورد استفاده قرار بگیرند



مدل E-UTxO

▪ نیاز به گسترش مدل UTxO

✓ مدل UTxO فقط پرداخت‌ها را مدیریت می‌کند (انتقال وجوه). اتریوم این مشکل را با افزودن حساب‌های قراردادی حل کرد. این راه‌حل خود پیچیدگی‌ها و آسیب‌پذیری‌های بالقوه‌ای را به وجود آورد

🔧 تغییرات مورد نیاز در مدل UTxO:

✓ امکان نگهداری از حالت قرارداد

✓ امکان اجرای کد قرارداد در توالی تراکنش‌ها

★ در مدل گسترش‌یافته، علاوه بر مقدار (value) می‌توان داده دلخواه نیز جابه‌جا کرد

★ در مدل گسترش‌یافته، به جای استفاده از کلیدهای عمومی برای قفل‌ها و کلیدهای خصوصی برای امضا، از منطق‌های دلخواهی که در اسکریپت‌ها تعریف می‌شود؛ استفاده می‌شود. این منطق کنترل می‌کند که آیا تراکنش (و داده آن) مجاز است یک ورودی باشد یا خیر

مدل E-UTxO

▪ قابلیت‌های افزوده شده مدل E-UTxO

★ به خروجی‌های تراکنش اجازه داده شده است تا یک مقدار داده را به همراه اعتباردهنده حمل کنند که به عنوان یک آرگومان اضافی در طول فرآیند اعتبارسنجی ارسال می‌شود. این به قرارداد این امکان را می‌دهد که بدون تغییر کد (اعتباردهنده)، به همراه خود برخی از حالت‌ها (داده‌ها) را حمل کند

★ اعتباردهنده اطلاعاتی را در مورد تراکنشی که در حال تایید است دریافت می‌کند. این اطلاعات به عنوان یک آرگومان اضافی از نوع TxInfo ارسال می‌شود. این اطلاعات اضافی، اعتباردهنده را قادر می‌سازد تا شرایط بسیار قوی‌تری را نسبت به مدل پیشین اعمال کند؛ به ویژه، می‌تواند خروجی‌های تراکنش فعلی را بررسی کند

★ به تراکنش‌ها، مفهوم فاصله اعتبار افزوده شده است. هر اسکریپتی که در طول اعتبارسنجی اجرا می‌شود، می‌تواند فرض کند که تیک فعلی در آن بازه است اما مقدار دقیق تیک فعلی را نمی‌داند

★ همه‌ی آرگومان‌های اعتباردهنده (رهایی‌بخش، مقدار داده و TxInfo) به عنوان مقادیری Data بیان می‌شوند. تمامی کلاینت‌ها ملزم به انکود و دیکود این داده‌ها در اسکریپت اعتباردهنده هستند

مدل E-UTxO

توصیف رسمی مدل E-UTxO

LEDGER PRIMITIVES

Quantity	an amount of currency
Tick	a tick
Address	an “address” in the blockchain
Data	a type of structured data
DataHash	the hash of a value of type Data
TxId	the identifier of a transaction
$txId : Tx \rightarrow TxId$	a function computing the identifier of a transaction
Script	the (opaque) type of scripts
$scriptAddr : Script \rightarrow Address$	the address of a script
$dataHash : Data \rightarrow DataHash$	the hash of a data value
$[[_]] : Script \rightarrow Data \times Data \times Data \rightarrow \mathbb{B}$	applying a script to its arguments

DEFINED TYPES

Output = (*value* : Quantity,
addr : Address,
dataHash : DataHash)

OutputRef = (*id* : TxId, *index* : \mathbb{N})

Input = (*outputRef* : OutputRef,
validator : Script,
data : Data,
redeemer : Data)

Tx = (*inputs* : Set[Input],
outputs : List[Output],
validityInterval : Interval[Tick])

Ledger = List[Tx]

مدل E-UTxO

▪ نوع TxInfo

OutputInfo = (*value* : Quantity,
validatorHash : Address,
dataHash : DataHash)

InputInfo = (*outputRef* : OutputRef,
validatorHash : Address,
dataVal : Data,
redeemer : Data,
value : Quantity)

TxInfo = (*inputInfo* : Set[InputInfo],
outputInfo : List[OutputInfo],
validityInterval : Interval[Tick],
thisInput : ℕ)

مدل E-UTxO

- توابع کمکی جهت اعتبارسنجی مدل گسترش یافته

$\text{lookupTx} : \text{Ledger} \times \text{TxId} \rightarrow \text{Tx}$

$\text{lookupTx}(l, id)$ = the unique transaction in l whose id is id

$\text{unspentTxOutputs} : \text{Tx} \rightarrow \text{Set}[\text{OutputRef}]$

$\text{unspentTxOutputs}(t) = \{(\text{txId}(t), 1), \dots, (\text{txId}(id), |t.outputs|)\}$

$\text{unspentOutputs} : \text{Ledger} \rightarrow \text{Set}[\text{OutputRef}]$

$\text{unspentOutputs}([]) = \{\}$

$\text{unspentOutputs}(t :: l) = (\text{unspentOutputs}(l) \setminus t.inputs) \cup \text{unspentTxOutputs}(t)$

$\text{getSpentOutput} : \text{Input} \times \text{Ledger} \rightarrow \text{Output}$

$\text{getSpentOutput}(i, l) = \text{lookupTx}(l, i.outputRef.id).outputs[i.outputRef.index]$

مدل E-UTxO

■ اعتبارسنجی یک مدل فرض در مدل گسترش یافته

1. The current tick is within the validity interval

$$\text{currentTick} \in t.\text{validityInterval}$$

2. All outputs have non-negative values

$$\text{For all } o \in t.\text{outputs}, o.\text{value} \geq 0$$

3. All inputs refer to unspent outputs

$$\{i.\text{outputRef} : i \in t.\text{inputs}\} \subseteq \text{unspentOutputs}(l).$$

4. Value is preserved

$$\text{Unless } l \text{ is empty, } \sum_{i \in t.\text{inputs}} \text{getSpentOutput}(i, l).\text{value} = \sum_{o \in t.\text{outputs}} o.\text{value}$$

5. No output is double spent

$$\text{If } i_1, i_2 \in t.\text{inputs} \text{ and } i_1.\text{outputRef} = i_2.\text{outputRef} \text{ then } i_1 = i_2.$$

6. All inputs validate

$$\text{For all } i \in t.\text{inputs}, \llbracket i.\text{validator} \rrbracket(i.\text{data}, i.\text{redeemer}, \text{toData}(\text{toTxInfo}(t, i, l))) = \text{true}.$$

7. Validator scripts match output addresses

$$\text{For all } i \in t.\text{inputs}, \text{scriptAddr}(i.\text{validator}) = \text{getSpentOutput}(i, l).\text{addr}$$

8. Data values match output hashes

$$\text{For all } i \in t.\text{inputs}, \text{dataHash}(i.\text{data}) = \text{getSpentOutput}(i, l).\text{dataHash}$$

با تشکر از توجه شما