Karl Wüst, Loris Diana, Kari Kostiainen, Ghassan Karame, Sinisa Matetic, Srdjan Capkun

Bitcontracts: Supporting Smart Contracts in Legacy Blockchains

NDSS 2021

Department of Computer Science, ETH Zurich

Motivations

- 传统的平台已有的用户基础
- 重新建立一个新的平台很困难
- 类似以太坊的区块链平台对智能合约可以使用的计算类型有很大限制
- 智能合约只能使用特定语言

GOALS

- 使类似比特币的传统区块链平台支持智能合约
- 使类似以太坊的平台使用智能合约限制更小
- 使开发人员能够使用自己擅长的语言编写智能合约

Method in a Nutshell

- 合约发起者指定一系列服务提供方来执行智能合约
- 合约状态、合约内容、指定服务方身份集合上链
- 需要执行合约时由客户取得以上内容发送给服务提供方
- 合约由服务提供方的子集执行并签名后结果返回给客户
- 客户将执行结果广播，矿工将其上链

Previous Work

Side-chain execution

- Rootstock (RSK) based on RBTC
- 仅有信任同样第三方的用户可以在同一条链上并且可以进行交易
- 主链上的用户需要五次转账才能完成交易

Previous Work

Off-chain execution

- Arbitrum, ACE , and Yoda
- 需要对矿工的行为修改因此无法在传统的区块链系统上部署
- State Channels
- 仅能在已经支持智能合约的区块链系统上执行

Previous Work

Execute-order-validate model

- Hyperledger Fabric
- 增加了验证的步骤因此无法在传统的区块链系统执行

Previous Work

Enclaved execution

- Ekiden
- outsource contract execution into trusted execution environments (TEEs) like SGX enclaves
- TEE compromise

PREVIOUS WORK

Blockchain multiparty computation (MPC)

- 有害节点可能阻止其他节点获取结果
- 仅能用于特定的计算，很难用于商业合约
- 参与者和合约执行时间必须提前给出
- 有些实现需要修改区块链底层

Previous Work

Enclaved multiparty computation

- FastKitten
- 对参与人数和生命周期仍有限制
- 用户共谋

Bitcontracts Overview

Execution and Trust Model

- 离链执行
- 服务提供方为知名公司或非盈利机构
- 需要服务提供方的集合E，包含n个服务提供方
- 每次执行需要t个服务方执行并认证，t<=n
- 需要1/2以上的服务提供方诚实

CHALLEGES

Where to store state?

- Off-chain
- 服务提供方之间需要执行自己的共识协议
- t需要>2/3n
- 执行前需要服务提供方确认自己处于最新状态
- 执行后需要服务提供方保证自己没有与其他分叉

CHALLEGES

Where to store state?

- On-chain
- 利用底层的区块链达成共识功能
- 客户可以各自验证执行的正确性
- 服务提供方可以保持无状态，简化实现

CHALLEGES

How to ensure consistency?

- Fabric: execute-order-validate
- ACE: order-execute-commit
- t>2/3n
- 需要适用execute-order

Overview

- combines off-chain execution of contracts with on-chain storage for contract state
- enables flexible trust models and high availability
- provides transparency towards the contract's clients
- does not require a new consensus protocol

Overview

- 智能合约账户由服务提供方共同维护
- 合约状态存储在区块链上
- 需要合约执行的输入一定是最新的合约状态，并且没有被作为合约的输入
- 单个交易执行多个合约

Cryptocurrency Properties

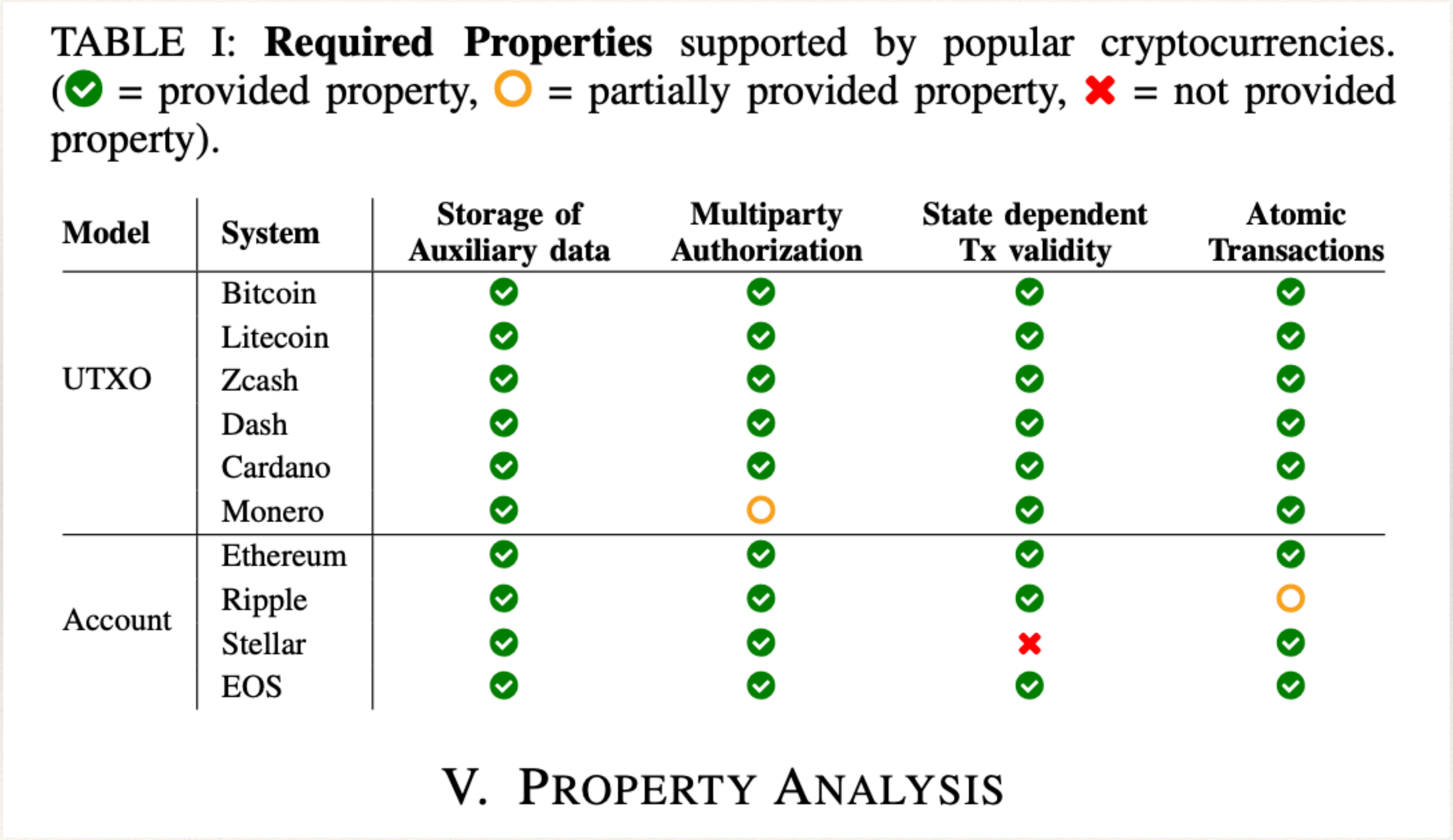

TABLE I: **Required Properties** supported by popular cryptocurrencies. (✓ = provided property, ○ = partially provided property, ✗ = not provided property).

Model	**System**	**Storage of Auxiliary data**	**Multiparty Authorization**	**State dependent Tx validity**	**Atomic Transactions**
UTXO	Bitcoin	✓	✓	✓	✓
	Litecoin	✓	✓	✓	✓
	Zcash	✓	✓	✓	✓
	Dash	✓	✓	✓	✓
	Cardano	✓	✓	✓	✓
	Monero	✓	○	✓	✓
Account	Ethereum	✓	✓	✓	✓
	Ripple	✓	✓	✓	○
	Stellar	✓	✓	✗	✓
	EOS	✓	✓	✓	✓

V. Property Analysis

Cryptocurrency Properties

- Multiparty authorization
- `σ = sign(Tx, sk)`
- `verify(Tx, Σ, PK, t)`
- Arbitrary data storage
- `Tx.append_data(d)`
- `d = Tx.read_data(loc, len)`

Cryptocurrency Properties

- State dependent transaction validity
- `Tx.require_previous(id)`
- Atomic transactions
- `Tx.add_transfer(src, dest, val)`

System Model

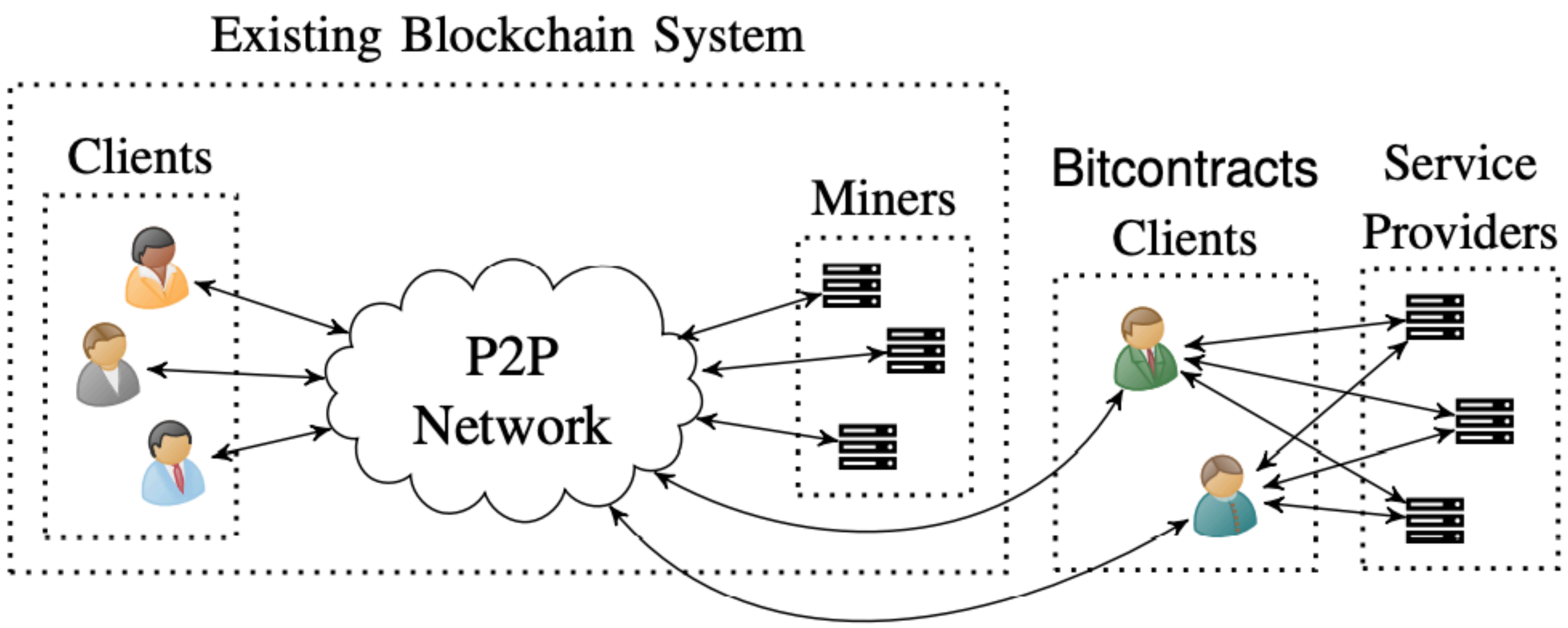

Fig. 1: **Bitcontracts overview.** Bitcontracts extends existing blockchain systems without changing their protocol, i.e. existing nodes such as clients and miners are agnostic to Bitcontracts. Bitcontracts clients interface with the blockchain and Bitcontracts service providers. Service providers are stateless and do not need to interact with the blockchain system.

Contract Deployment

- 合约内容包含任意语言编写的代码、资金和键值对存储的合约状态
- 合约发起者选择n个服务提供方以及阈值t
- 合约发起者发起一个交易，对象为服务提供方共同维护的账户
- 交易包含初始资金、代码hash、初始状态hash和初始状态本身
- 代码可以包含在初始交易中也可以在网站发布

Contract Execution

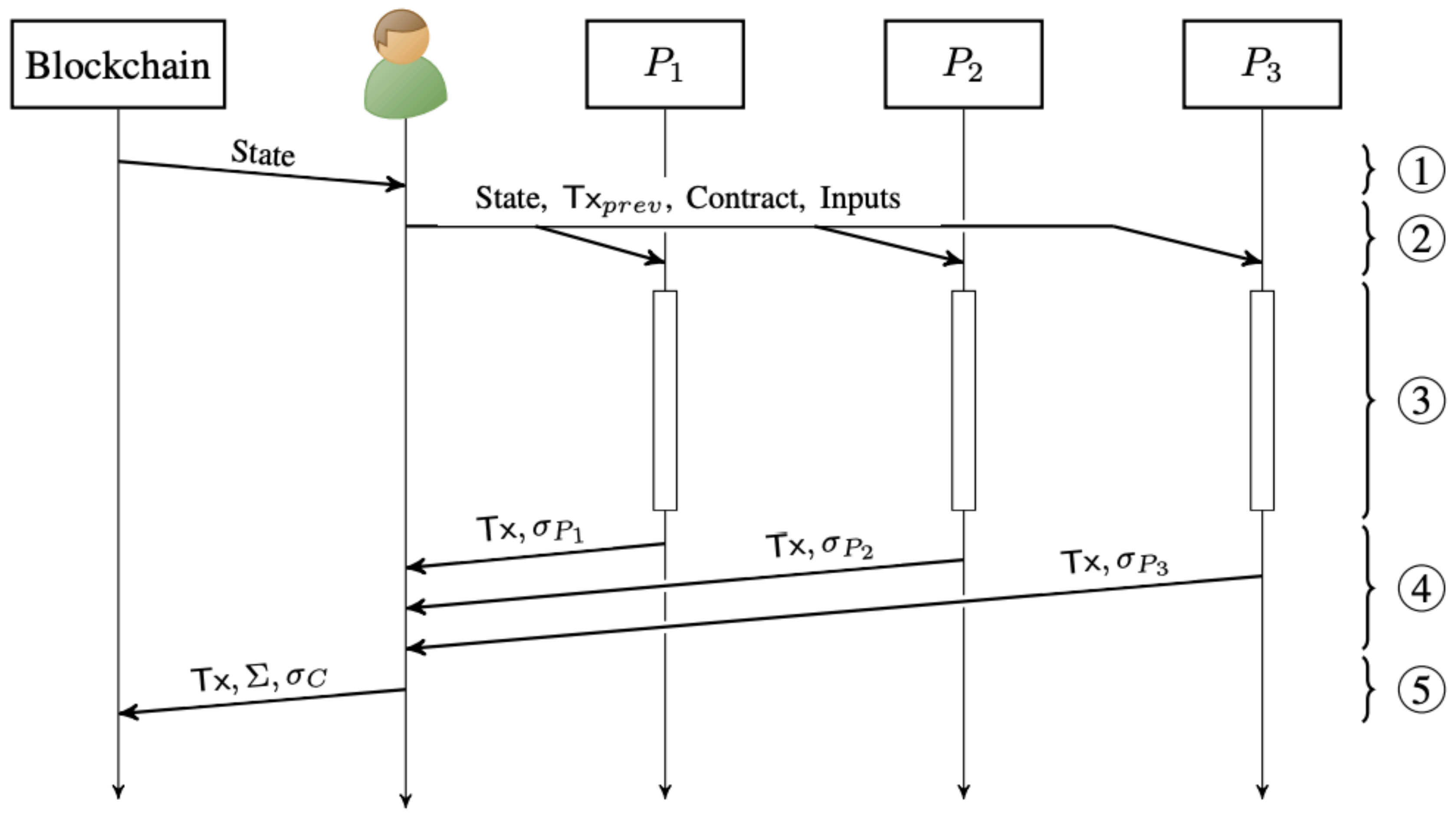
Blockchain
P_1
P_2
P_3
State
State, Tx_{prev}, Contract, Inputs
$\mathsf{Tx}, \sigma_{P_1}$
$\mathsf{Tx}, \sigma_{P_2}$
$\mathsf{Tx}, \sigma_{P_3}$
$\mathsf{Tx}, \Sigma, \sigma_C$
①
②
③
④
⑤

Fig. 2: **Contract call.** To call a smart contract, the client first assembles the state from the blockchain and then sends the state, the previous transaction, and his inputs to the service providers. The service providers then execute the contract call and send the resulting transaction as well as their signatures to the client, who finalizes the transaction and broadcasts it.

Contract Execution

- 客户首先联系t个服务提供方用于执行合约
- 客户取得合约的最新状态
- 客户取得最近的交易Txprev、合约代码、合约输入和其他用于资金转移的信息
- 客户将所有的信息发送给t个服务提供方

Contract Execution

- 服务提供方首先通过`read_data`获得合约代码以及状态hash，并计算客户提供的两者hash进行对比，如果一致进入下一步
- 服务提供方通过客户输入执行合约，创建新的交易`Tx`并通过`Tx.require_previous(Txprev.id)`使其依赖前一笔交易
- 服务提供方将合约新状态的hash、代码hash以及交易hash通过`append_data`放入交易的存储空间

Contract Execution

- 服务提供方将合约合约状态变化的列表通过`append_data`放入交易中
- 如果发生资金变动，服务提供方还需要通过`add_transfer`将转移记录放入交易中
- 最后服务提供方通过`σPk = sign(Tx)`将交易签名并返回给客户

Contract Execution

- 客户从所有服务提供方出取得交易内容Tx和签名σPk
- 客户汇总所有的签名P1 , . . . , σPt 为Σ
- 如果产生资金转移，客户还需在Tx上提供自身的签名σC
- 客户使用(Tx, Σ, σC) 对交易进行签名并广播

Contract Dependencies

- 整个执行的过程是原子的
- 每个合约在其信任模型下执行的可信
- 客户首先在本地决定所有要执行的合约
- 客户将所有这些合约和所需信息发送给服务提供方之后服务方执行按照之前描述执行

Use of Oracles

- Service providers can natively act as oracles for many use cases since the contract code can directly connect to external websites or data feeds.

Incentives

- 理想场景：服务提供方只在可以保证得到服务费后才提供服务，客服只有在得到服务后才付费
- 客户在请求服务前向服务提供方共同控制的账户提供t份的服务费用
- 在一定的时间间隔后，服务提供方根据其在合约中记录的执行数获得费用

Evalution

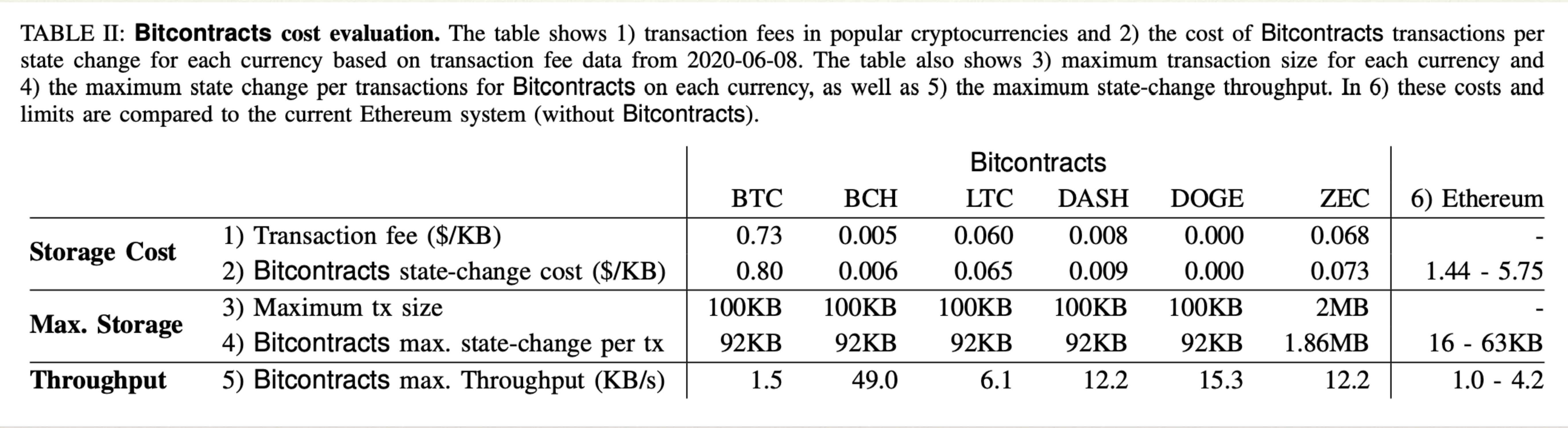

TABLE II: **Bitcontracts cost evaluation.** The table shows 1) transaction fees in popular cryptocurrencies and 2) the cost of Bitcontracts transactions per state change for each currency based on transaction fee data from 2020-06-08. The table also shows 3) maximum transaction size for each currency and 4) the maximum state change per transactions for Bitcontracts on each currency, as well as 5) the maximum state-change throughput. In 6) these costs and limits are compared to the current Ethereum system (without Bitcontracts).

		Bitcontracts						
		BTC	BCH	LTC	DASH	DOGE	ZEC	6) Ethereum
Storage Cost	1) Transaction fee ($/KB)	0.73	0.005	0.060	0.008	0.000	0.068	-
	2) Bitcontracts state-change cost ($/KB)	0.80	0.006	0.065	0.009	0.000	0.073	1.44 - 5.75
Max. Storage	3) Maximum tx size	100KB	100KB	100KB	100KB	100KB	2MB	-
	4) Bitcontracts max. state-change per tx	92KB	92KB	92KB	92KB	92KB	1.86MB	16 - 63KB
Throughput	5) Bitcontracts max. Throughput (KB/s)	1.5	49.0	6.1	12.2	15.3	12.2	1.0 - 4.2

Evalution

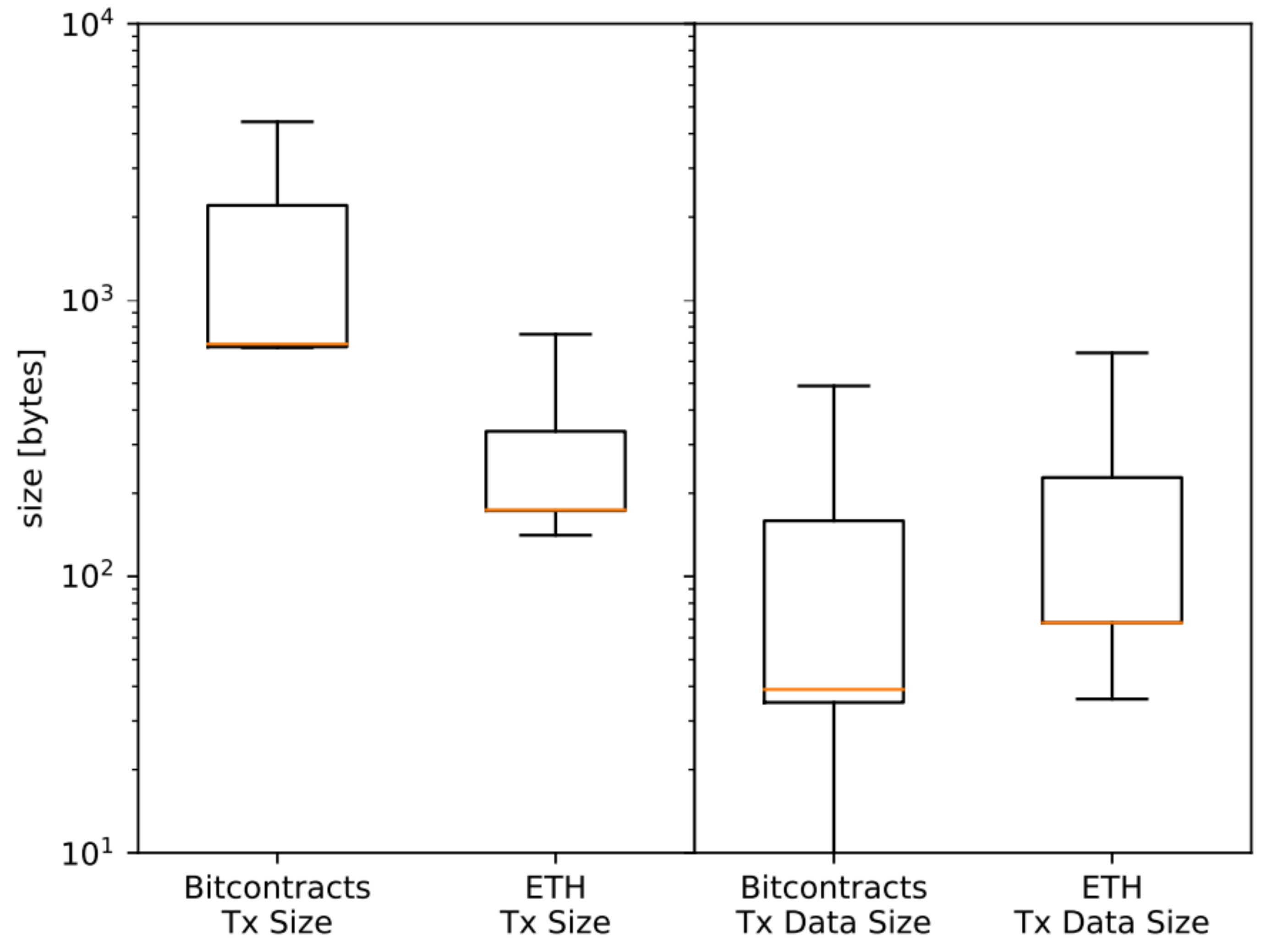

Fig. 4: Full transaction size (left) and transaction data size (right) comparison between Bitcontracts and Ethereum. In Ethereum, the transaction data size includes all transaction inputs, such as function arguments. In Bitcontracts, the transaction data size is the size of all state outputs of the transaction. In the box plots, the whiskers show the 2nd and 98th percentile respectively. The orange line (coinciding with the bottom of the boxes in all but the third box) shows the median value and the bottom and top of the boxes show the 25th and 75th percentile, respectively.

Evalution

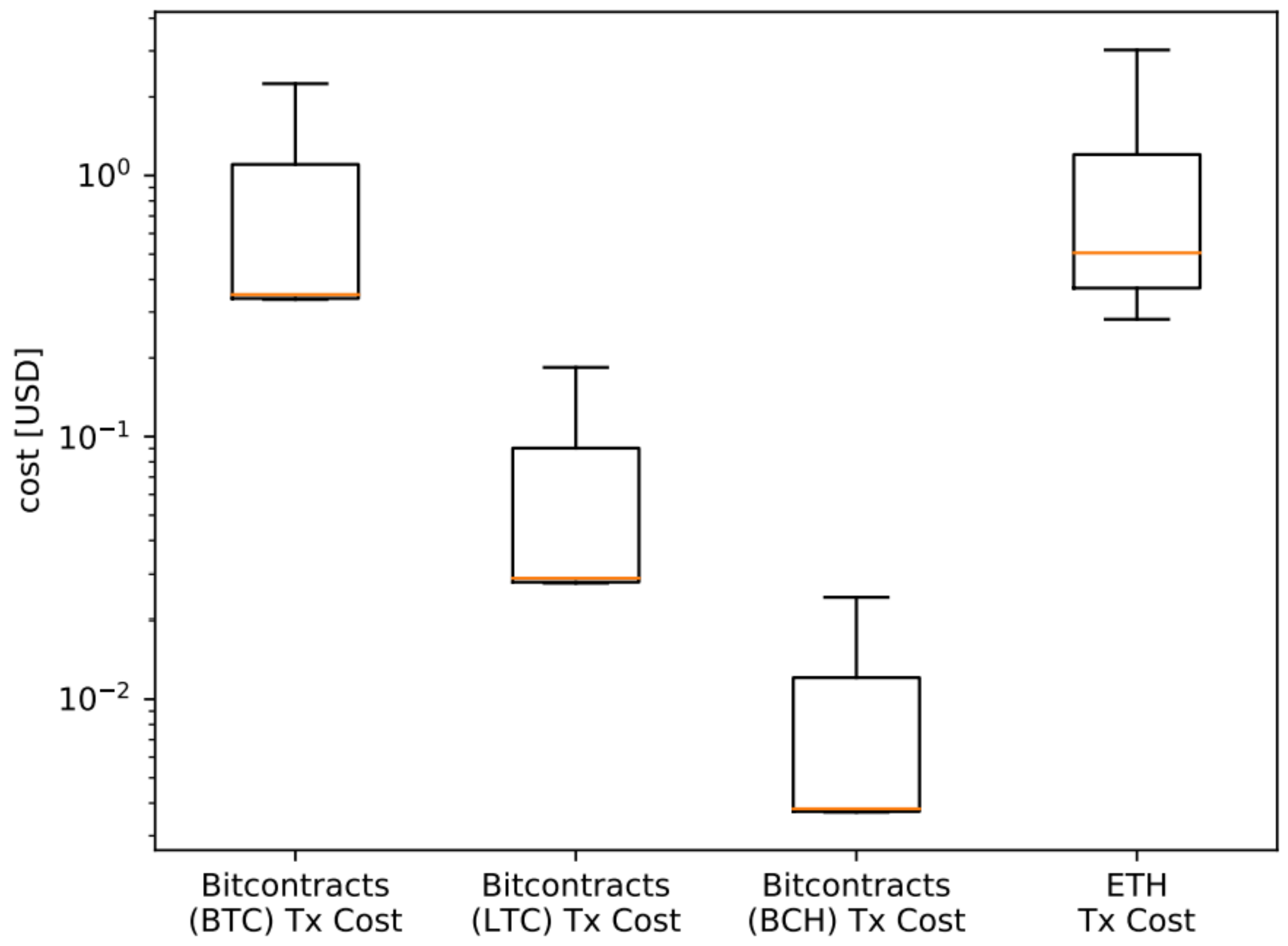

Fig. 5: Transaction cost comparison between Bitcontracts and Ethereumas box plots with the whiskers showing the 2nd and 98th percentile respectively. The orange line (coinciding with the bottom of the boxes in all but the last box) shows the median value and the bottom and top of the boxes show the 25th and 75th percentile, respectively.