A Cluster Based System for Analyzing Ethereum Blockchain Transaction Data

Baran Kılıç Can Özturan Alper Sen

Boğaziçi University, Istanbul, Turkey

The Second International Conference on Blockchain Computing and Applications, Nov 2020

Outline

Introduction

Introduction

- Decentralized finance is gaining popularity

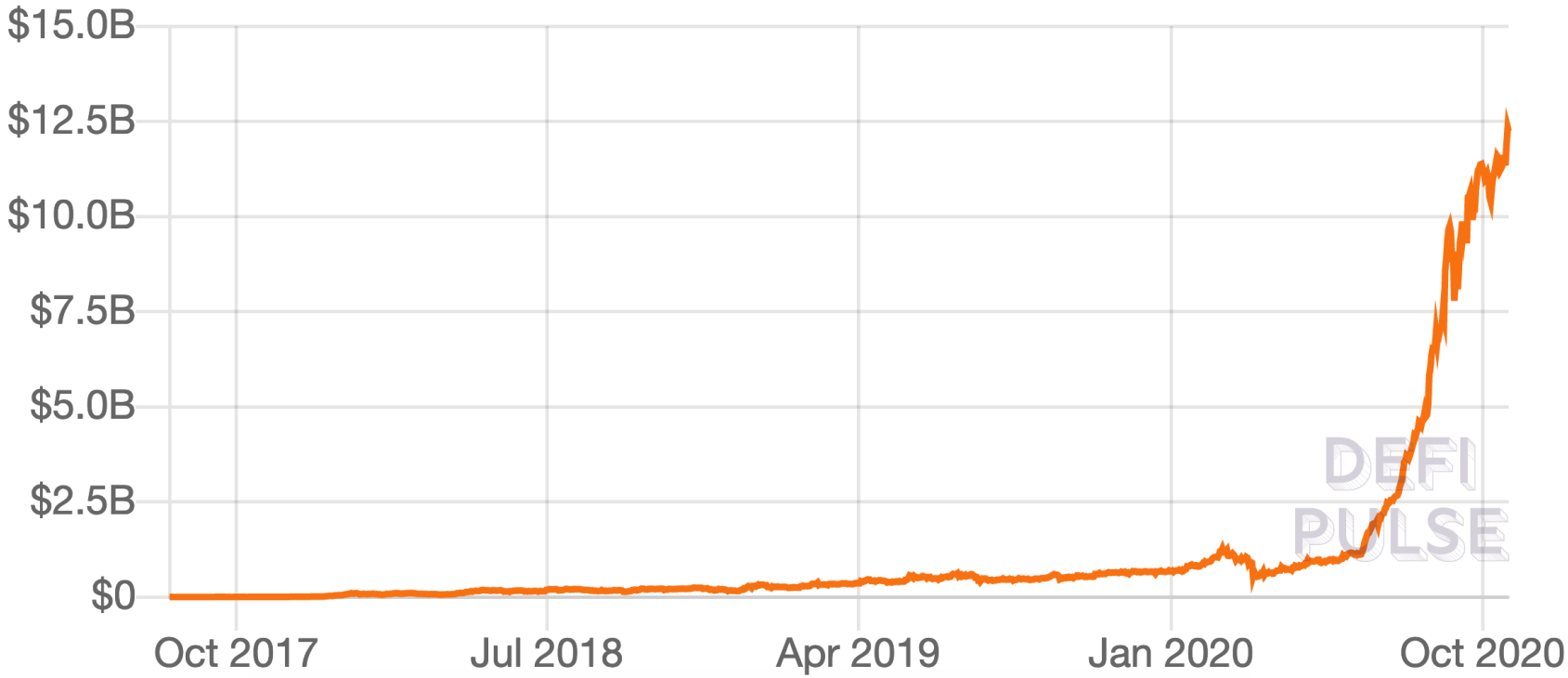
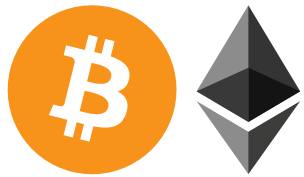


Figure: Total value locked (USD) in DeFi [1]

[1] Source: `https://defipulse.com` (Accessed: 23 Oct 2020)

Decentralized Finance

- Decentralized exchanges
  - bitcoin
  - ether
- ERC20 tokens
- Stablecoins (1 token equal to 1 EUR or 1 USD)
  - Gemini USD (GUSD)
  - Tether USD (USDT)
  - Tether Gold (XAUT)
  - Turkish BiLira (TRYB)
- "Wrapped" bitcoins



Fraudulent Activities

- Blockchain networks are used globally
- Possible to bypass regulations
- Smart contracts act like a bridge between the public Ethereum blockchain ecosystem and the traditional finance ecosystems
- Fraudulently obtained assets
- Assets can go through various transfers
- It is possible that a company that accepts stable coins
- Holding fraudulent stable coins can be risky for the company

Transaction Analysis

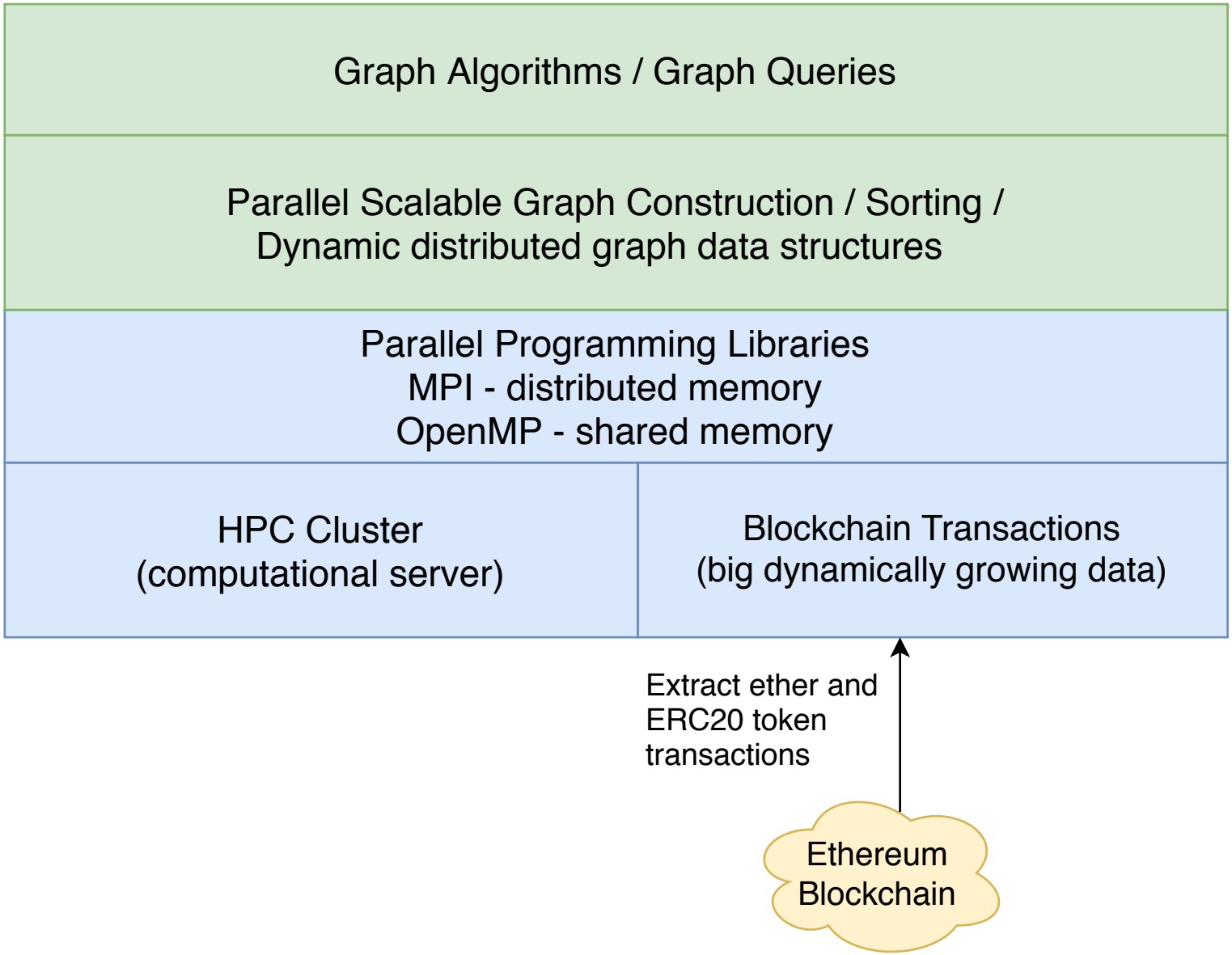
- Massive amount of blockchain transaction data
- Increasing transaction throughputs
  - Sharding and proof of stake in Ethereum 2.0
  - Metastable consensus protocols (e.g. Avalanche)
- The transaction data will not fit into a single node in the future
- We need a scalable distributed transaction graph processing system that will enable fast graph operations to be performed on the blockchain transaction data

Our Contributions

- A cluster based system that constructs a distributed transaction graph in parallel out of raw transaction data that comes from the blockchain.
- A parallel system that offers the advantage of being able to scale by simply increasing the number of nodes in the cluster.
- Performance tests of our system using the whole Ethereum blockchain data.

Blockchain Graph System Architecture

System Architecture



Distributed Transaction Graph Construction

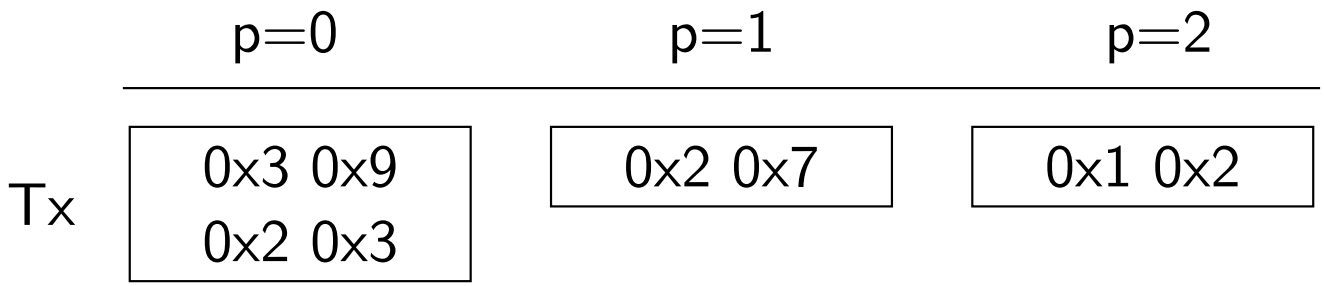
Symbols And Their Meanings

V_c	Set of account addresses
V_s	Set of smart contract addresses
V	All blockchain addresses $V = V_c \cup V_s$
E_{ETH}	All blockchain transactions with zero or more ether payments
T	Set of major ERC20 tokens tracked $T = \{USDT, PAX, \cdots\}$
E_T	ERC20 token t transfer transactions, $t \in T$
E	All transfer transactions $E = E_{ETH} \cup E_T$
$G(V, E)$	Transaction graph

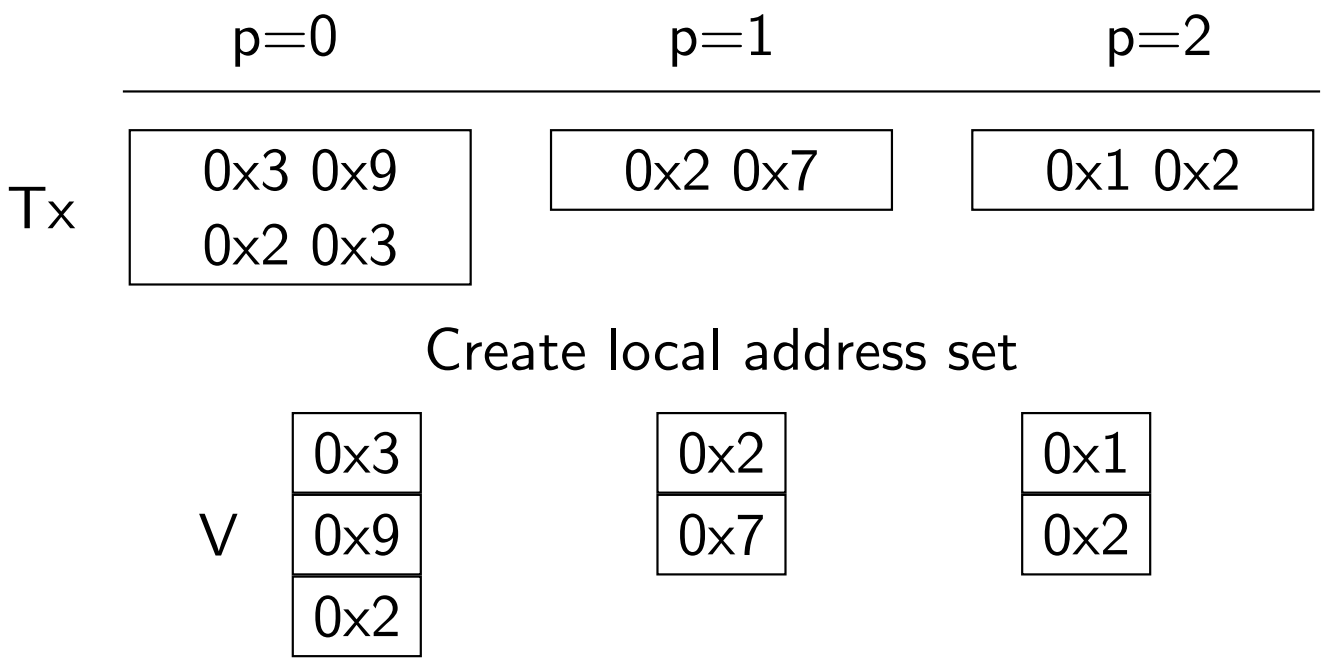
Symbols And Their Meanings

$ID(v)$	Global ID of address v, $ID(v) \in [0, \lvert V \rvert - 1]$
P	Number of processors
p	ID of current processor (0-indexed)
V^p	Blockchain addresses on processor p
E^p	Transfer transactions on processor p
$G^p(V^p, E^p)$	Transaction subgraph on processor p

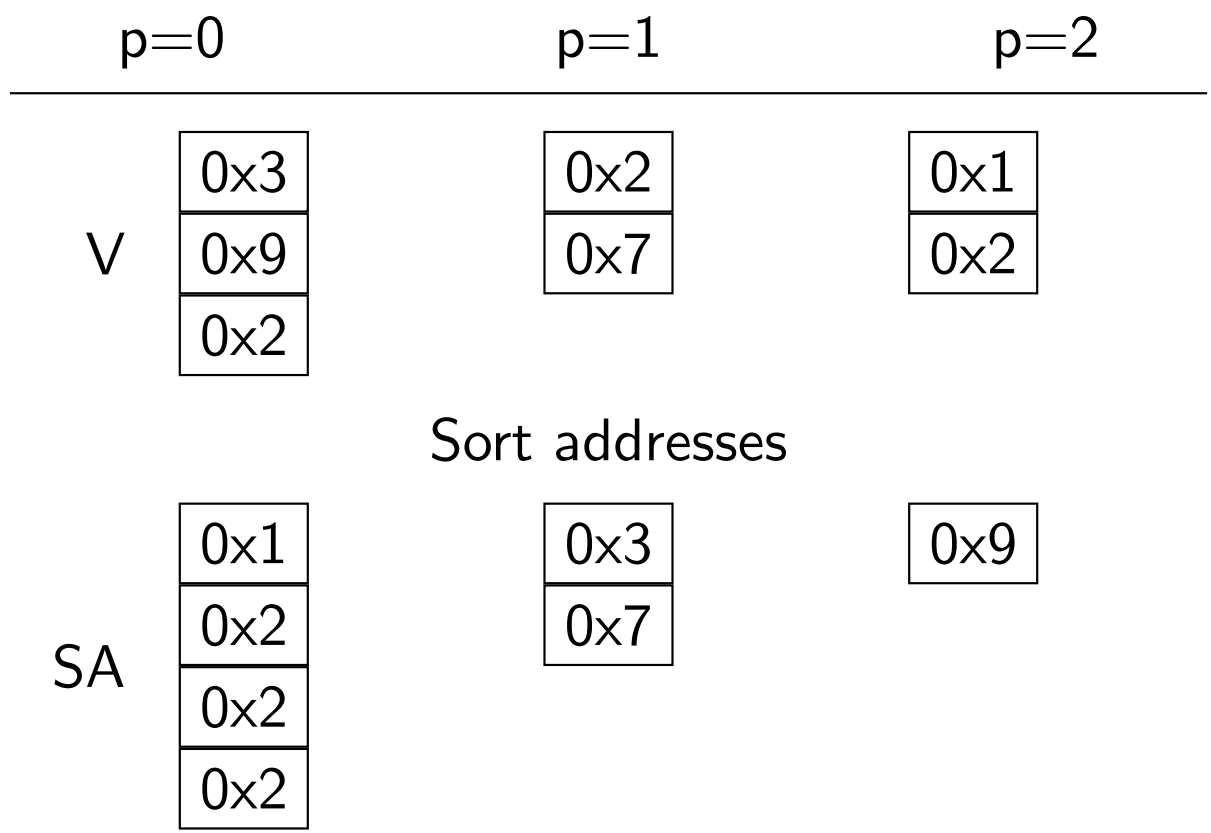
Distributed Transaction Graph Construction



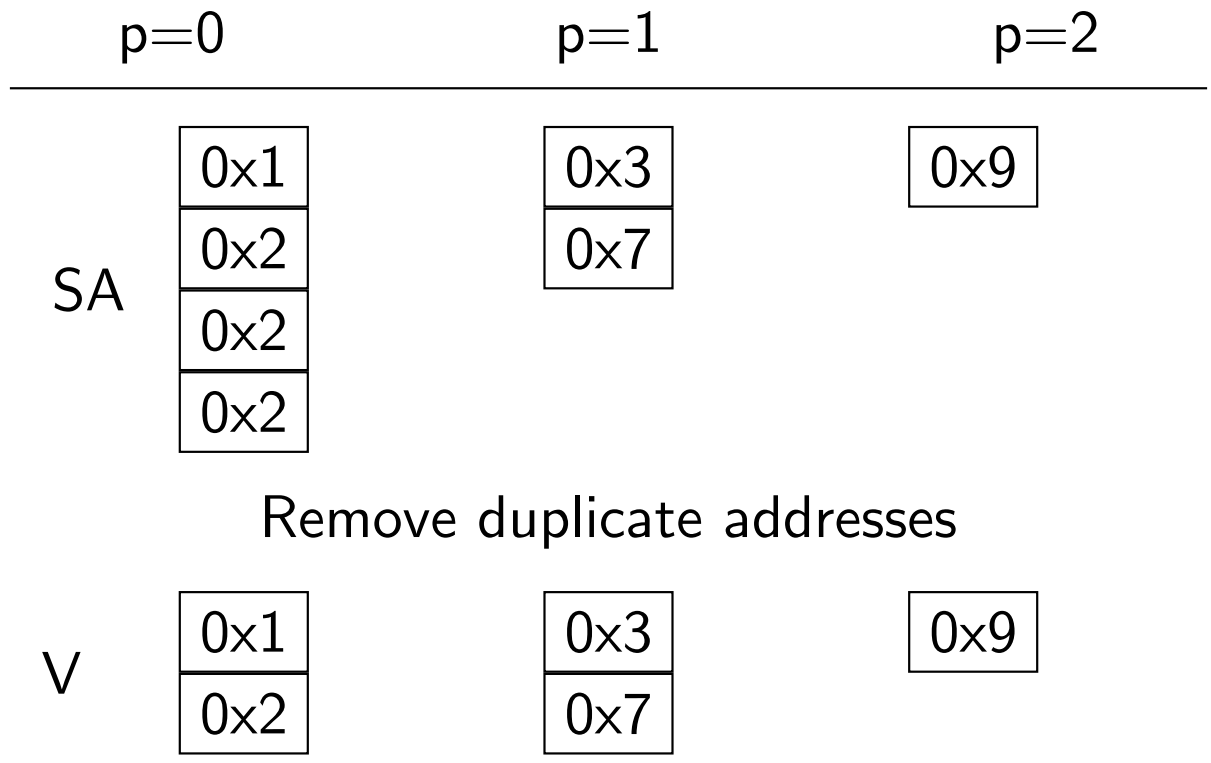
Distributed Transaction Graph Construction



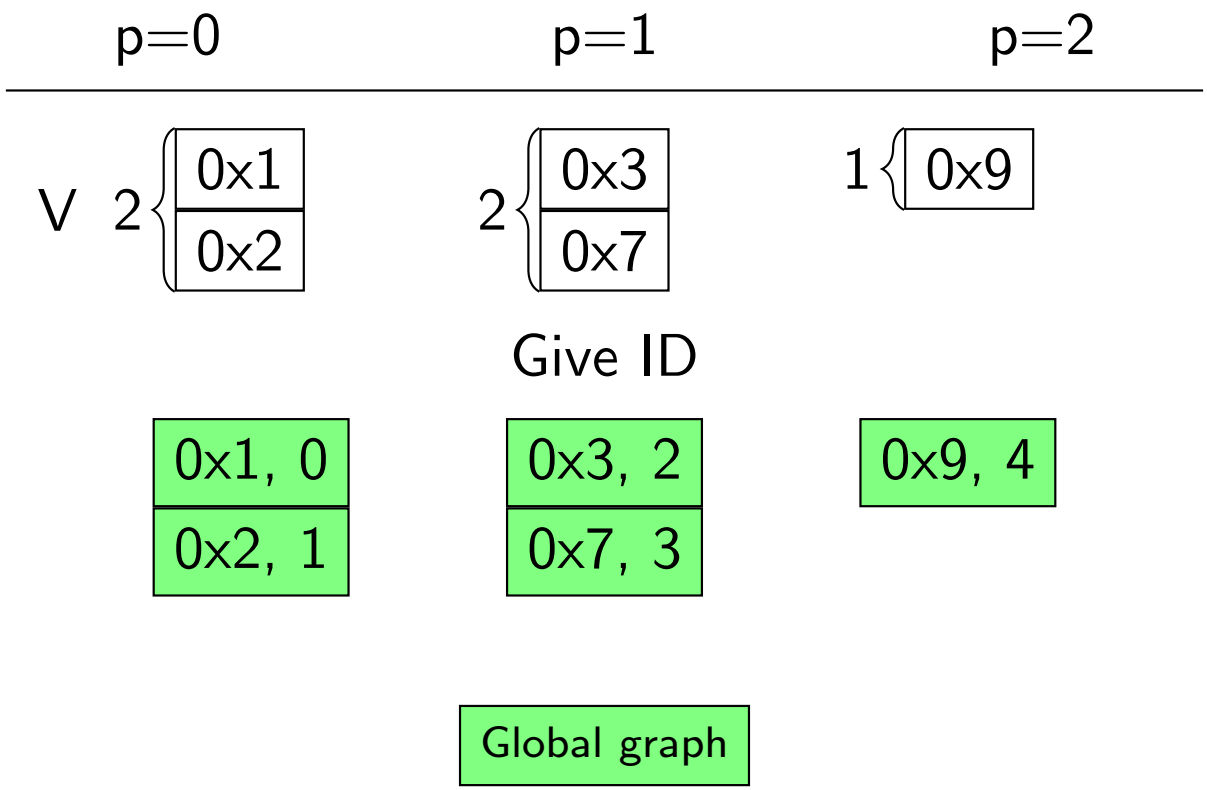
Distributed Transaction Graph Construction



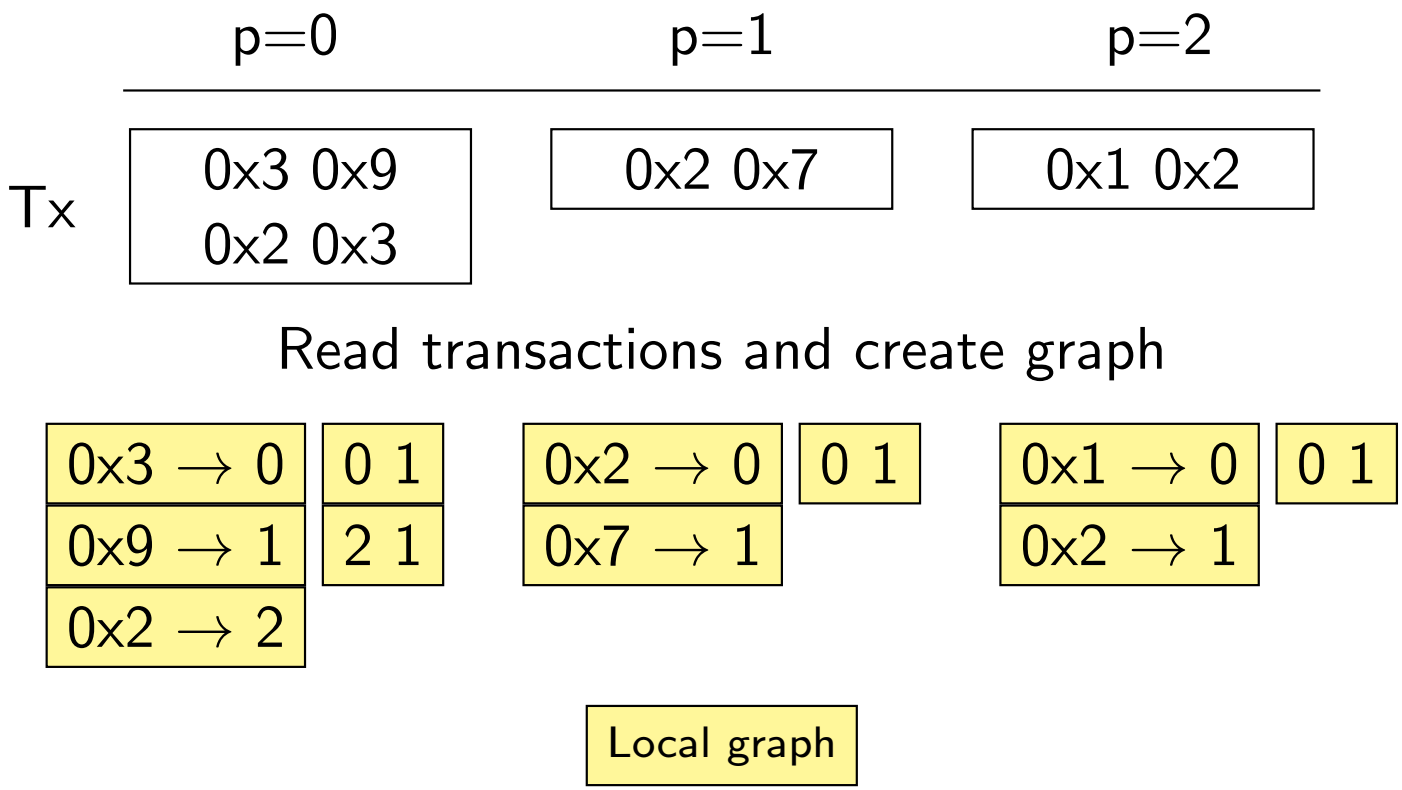
Distributed Transaction Graph Construction



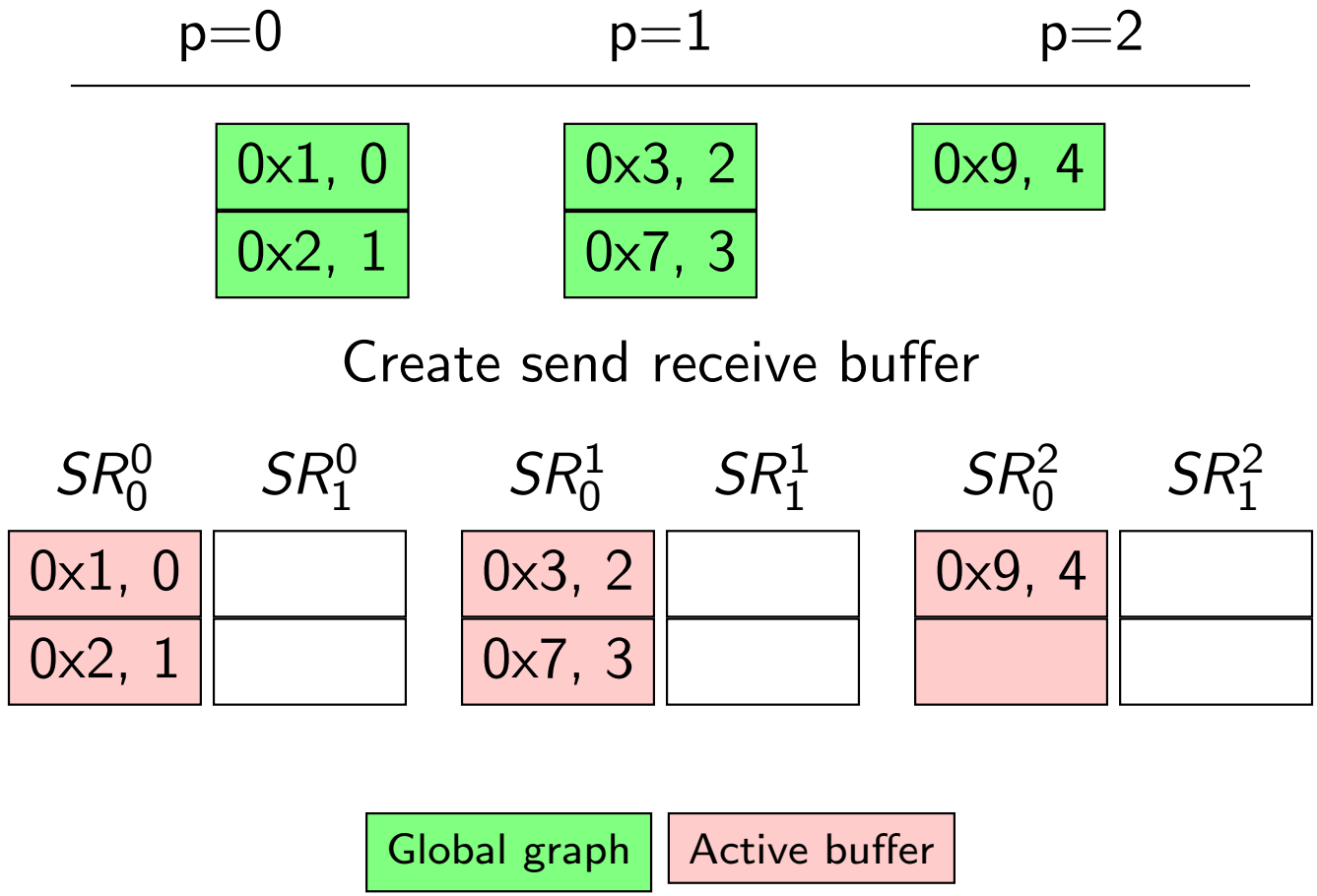
Distributed Transaction Graph Construction



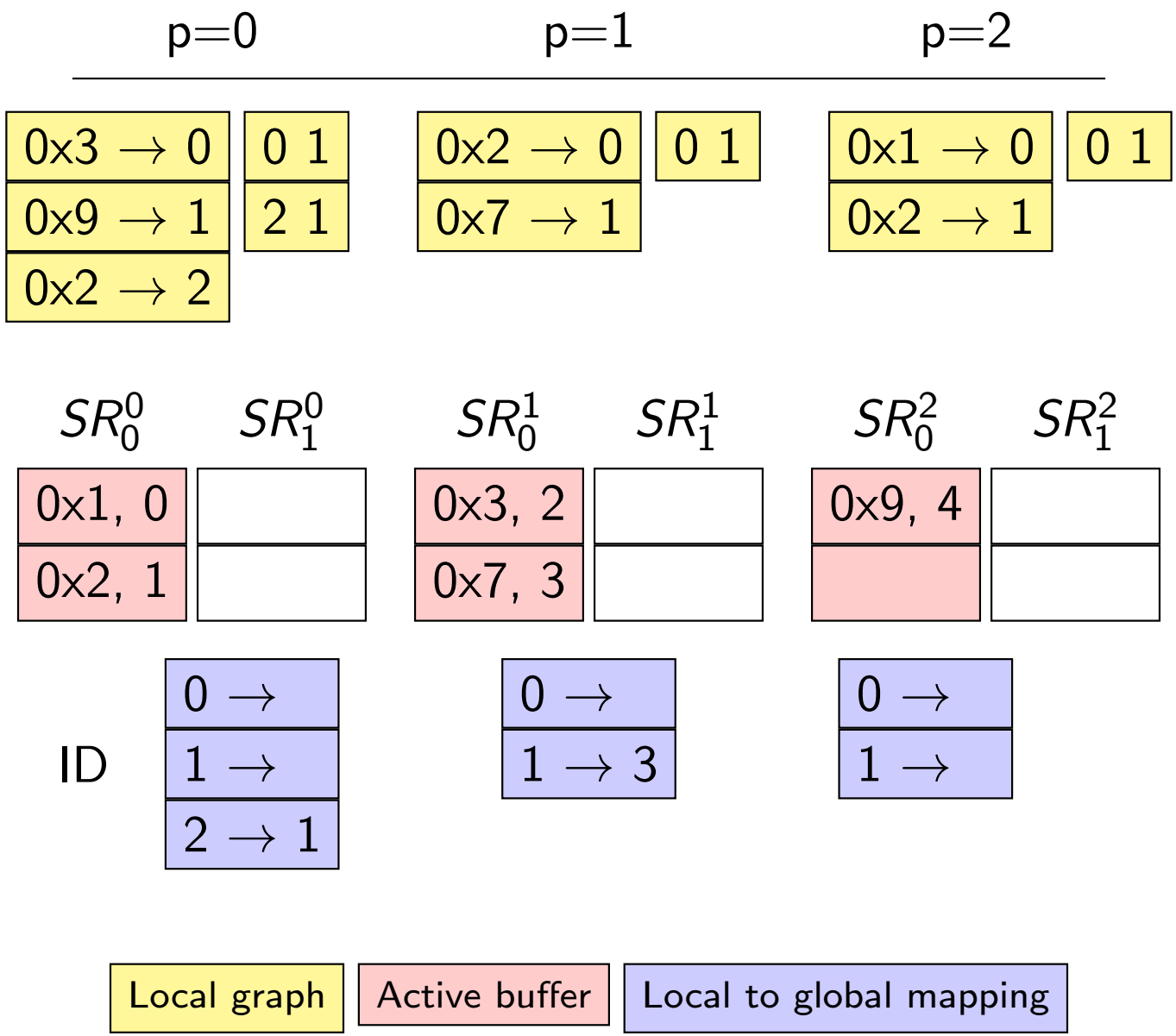
Distributed Transaction Graph Construction



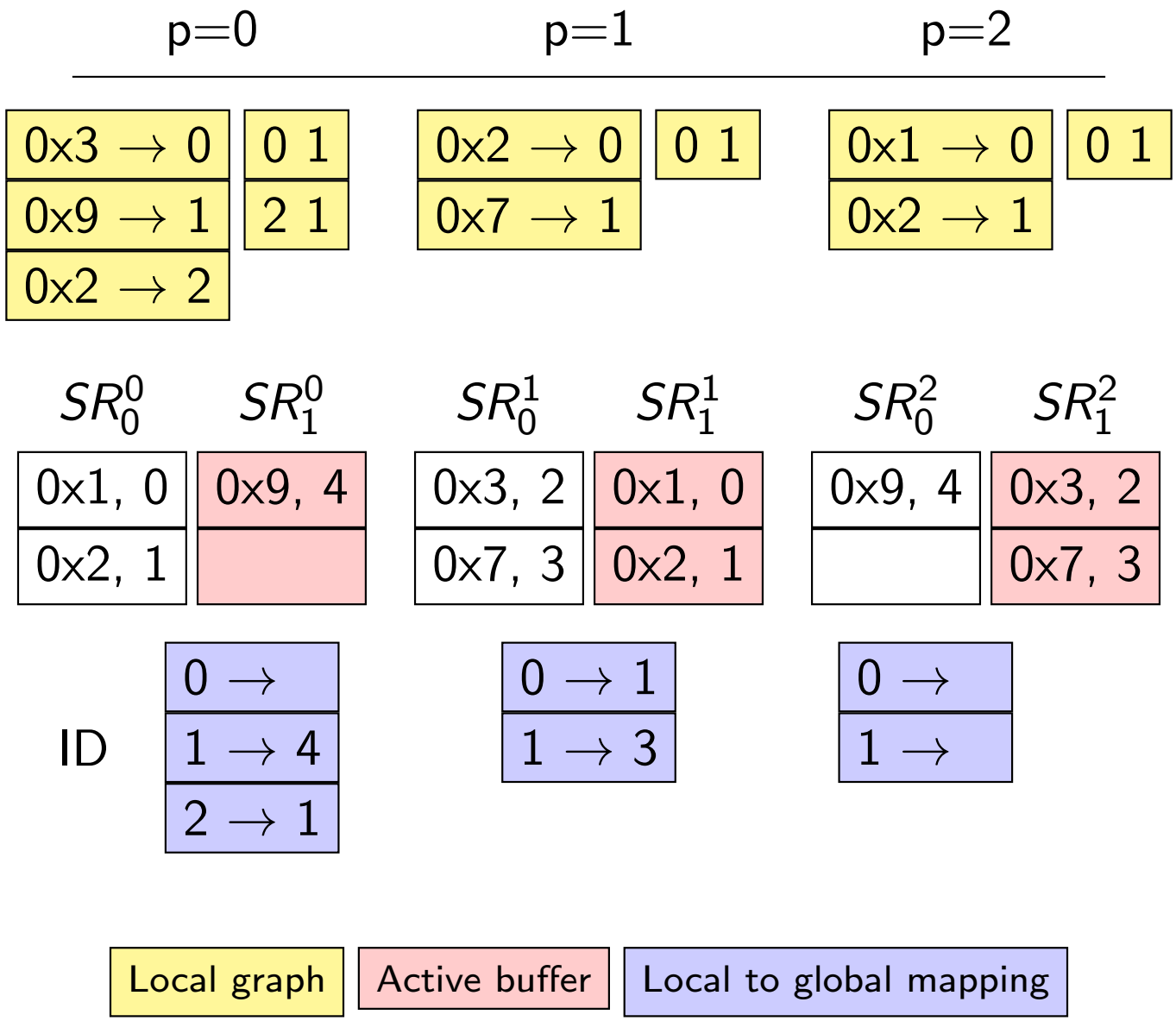
Distributed Transaction Graph Construction



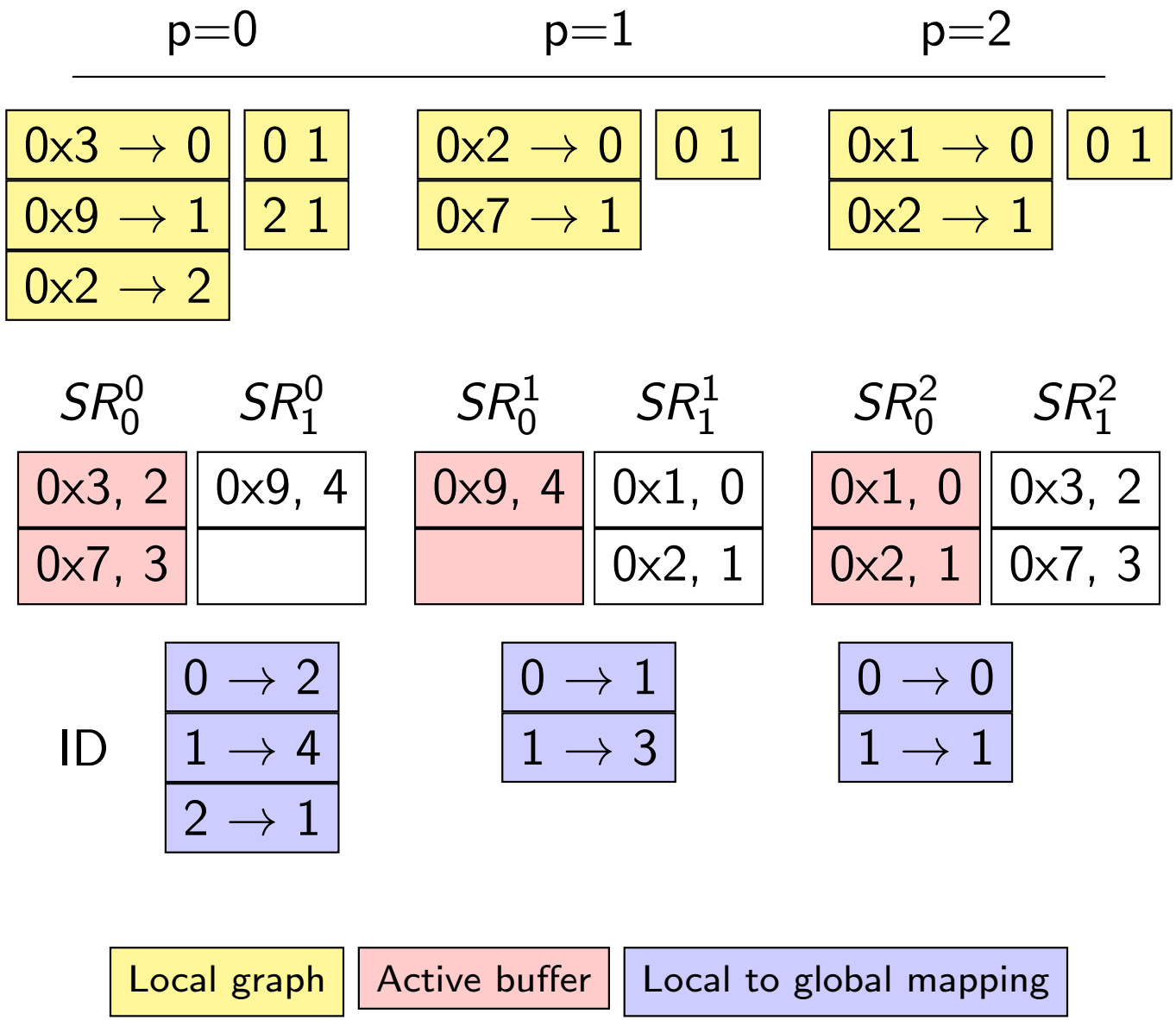
Distributed Transaction Graph Construction



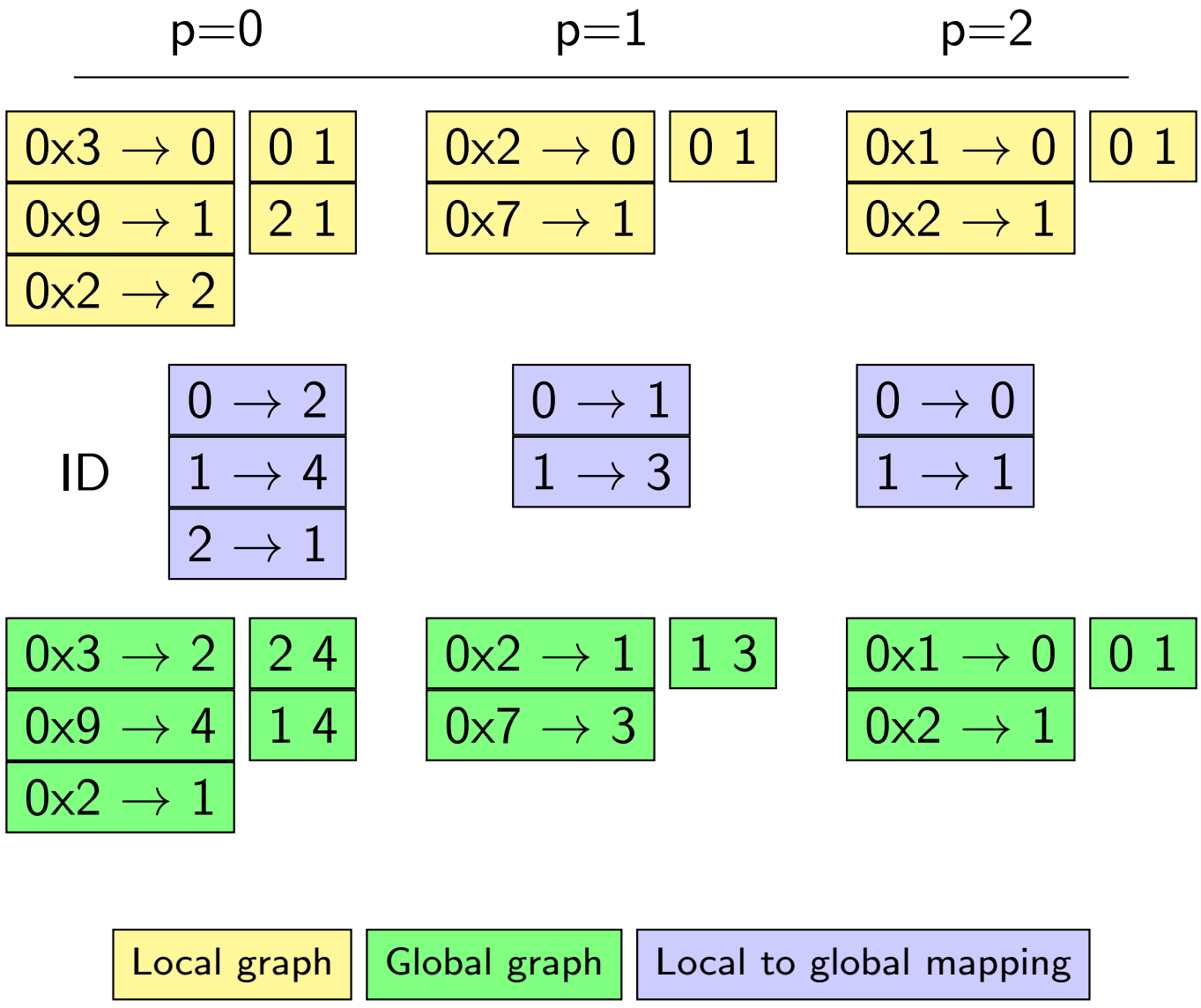
Distributed Transaction Graph Construction



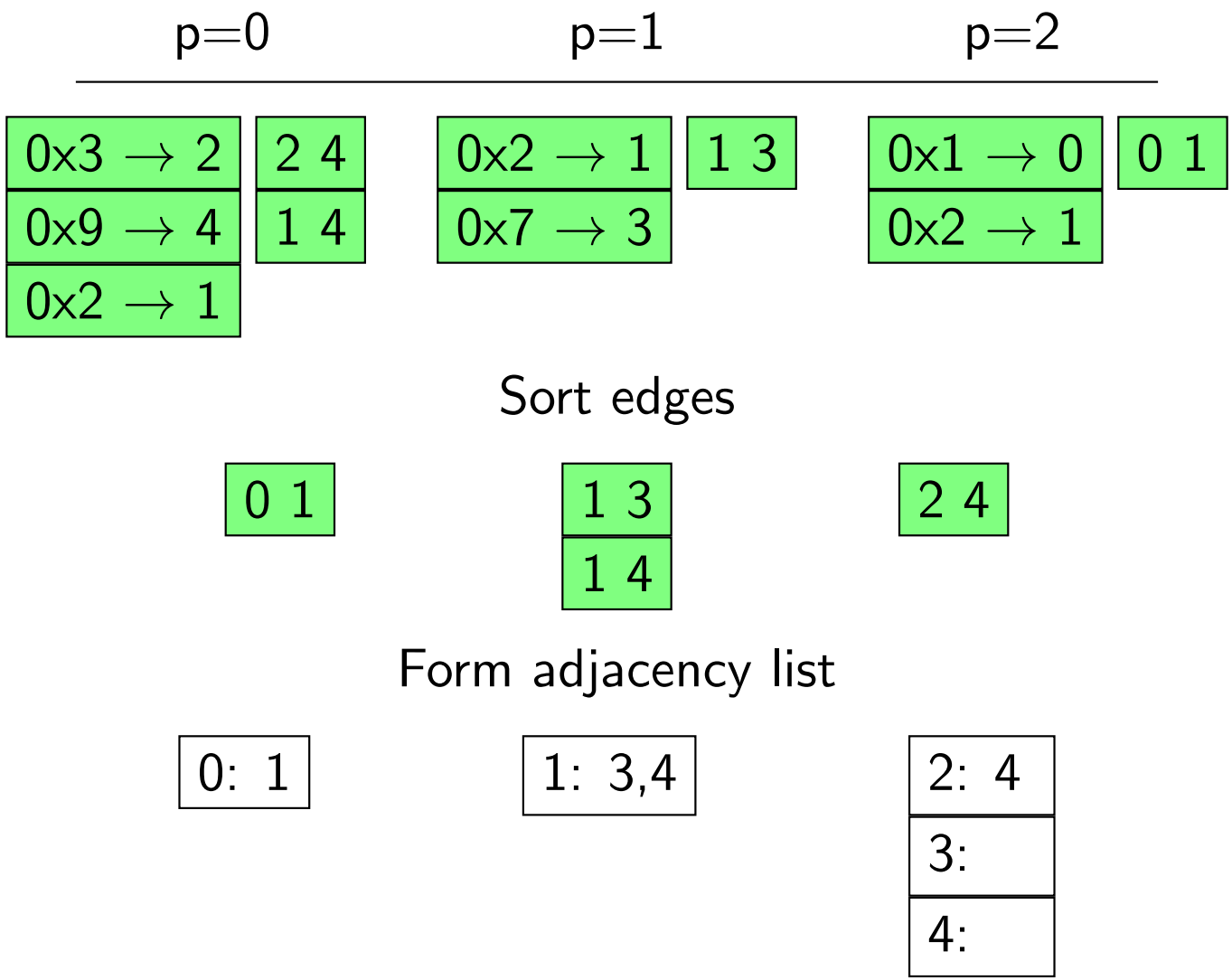
Distributed Transaction Graph Construction



Distributed Transaction Graph Construction



Distributed Transaction Graph Construction



Distributed Transaction Graph Construction

1: **procedure** GENERATEGRAPH(E^p, p)
2: $V^p \leftarrow \varnothing$
3: **for each** $(s,t) \in E^p$ **do**
4: $V^p \leftarrow \{s,t\} \cup V^p$
5: **end for**
6: $SA^p \leftarrow$ parallelSampleSort(V^p) ▷ sorted array
7: $V^p \leftarrow \{SA^p_i | i = 1 \vee (i \in [2..|SA^p|] \wedge SA^p_i \neq SA^p_{i-1})\}$
8: $IdStart \leftarrow \sum_{j=0}^{p-1} |V^j|$ ▷ by parallel scan
9: $ID(v_j) = IdStart + j \quad \forall j \in [0..V^p - 1]$
10: $SR^p_0 \leftarrow \{(v, ID(v)) | v \in V^p\}$ ▷ send/recv buffer
11: $SR^p_1 \leftarrow \varnothing$ ▷ send/recv buffer
12: **for** $i \in [0..P-1]$ **do**
13: $j \leftarrow i \bmod 2$ ▷ index of current SR
14: $k \leftarrow i + 1 \bmod 2$ ▷ index of SR to recv.
15: **for each** $(s,t) \in E^p$ **do**
16: $sID \leftarrow$ binarySearch(SR^p_j, s)

Distributed Transaction Graph Construction

17: **if** $sID \neq null$ **then**
18: $ID(s) \leftarrow sID$
19: **end if**
20: $tID \leftarrow \text{binarySearch}(SR_j^p, t)$
21: **if** $tID \neq null$ **then**
22: $ID(t) \leftarrow tID$
23: **end if**
24: **end for**
25: $n \leftarrow (i+1) \bmod P$ ▷ ID of next proc.
26: $m \leftarrow (i-1+P) \bmod P$ ▷ ID of prev. proc.
27: $SR_k^n \leftarrow SR_j^p$ ▷ send SR to next proc.
28: $SR_k^p \leftarrow SR_j^m$ ▷ recv. SR from prev. proc.
29: **end for**
30: $IDE^p \leftarrow \{(ID(s), ID(t)) | (s,t) \in E^p\}$
31: $SIDE^p \leftarrow \text{parallelSampleSort}(IDE^p)$
32: formAdjacencyListofGraph($SIDE^p$)
33: **return** $G^p(V^p, E^p)$
34: **end procedure**

Tests

Ethereum Blockchain Data Statistics Used In Tests

(a)	Blocks	0 - 9 499 999
(b)	Time Coverage of Blocks	30.07.2015-17.02.2020
(c)	No. of Transactions	633 762 485
(d)	No. of Addresses	69 223 762
(e)	No. of 31 Major ERC20 Token Transfer Transactions	24 646 152
(f)	List of symbols of 31 Major ERC20 Tokens	USDT PAX EURS GUSD TRYB BAT CEO LNK HT HEDG MKR CRO VEN INO INB SNX MOF ZRX SXP OKB XIN SAI HOT DAI HPT BUSD XAUT USDC SUSD HOG QCAD

Description of Tests

Test	Description
T1	Transaction Graph Construction
T2	Graph Partitioning using ParMetis
T3	Page Ranking on transaction graph
T4	Degree distributions of transaction graph nodes
T5	No. of transfer transactions of 31 major ERC20 tokens

- All these tests were programmed using the MPI libraries

Setup

- Amazon Web Services
- 16 c5.4xlarge EC2 machine instances
  - 16 virtual CPUs per instance
  - 32 GiB memory per instance
- Cluster node count: 4, 8, 12, 16
- MPI process per node: 1, 2, 4, 8, 12, 16

Timings

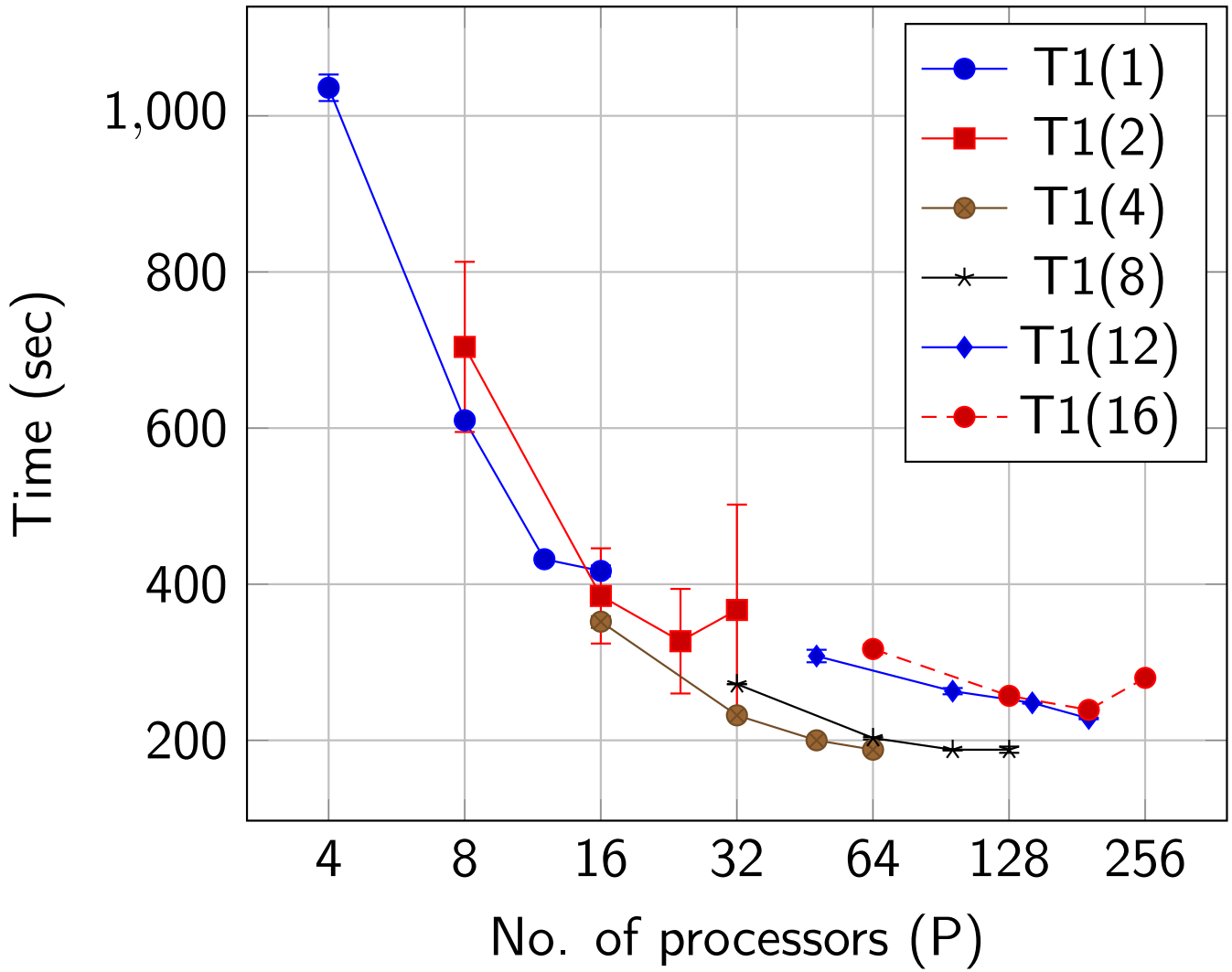


Figure: Test T1 distributed transaction graph construction timings. The integers i in parentheses $T1(i)$ indicate number of MPI processes per cluster node

Test Timings

T1 Graph Const.
T2 ParMetis
T3 PageRank
T4 Deg. Dist.
T5 No. ERC20 Tx

Test	1 MPI process per node			
	P=4	P=8	P=12	P=16
T1	1036	610	432	417
T2	4782	4217	4146	3755
T3	471	261	182	134
T4	30	13	9	8
T5	137	50	35	30

	8 MPI processes per node			
	P=32	P=64	P=96	P=128
T1	272	203	188	188
T3	94	49	44	41
T4	10	7.2	6.1	5.4
T5	92	8.7	6.9	5.5

Table: Timings of tests in seconds

Pagerank

1. Binance
2. Shapeshift
3. Bitfinex
4. Plus Token
5. Kraken

Degree Distribution

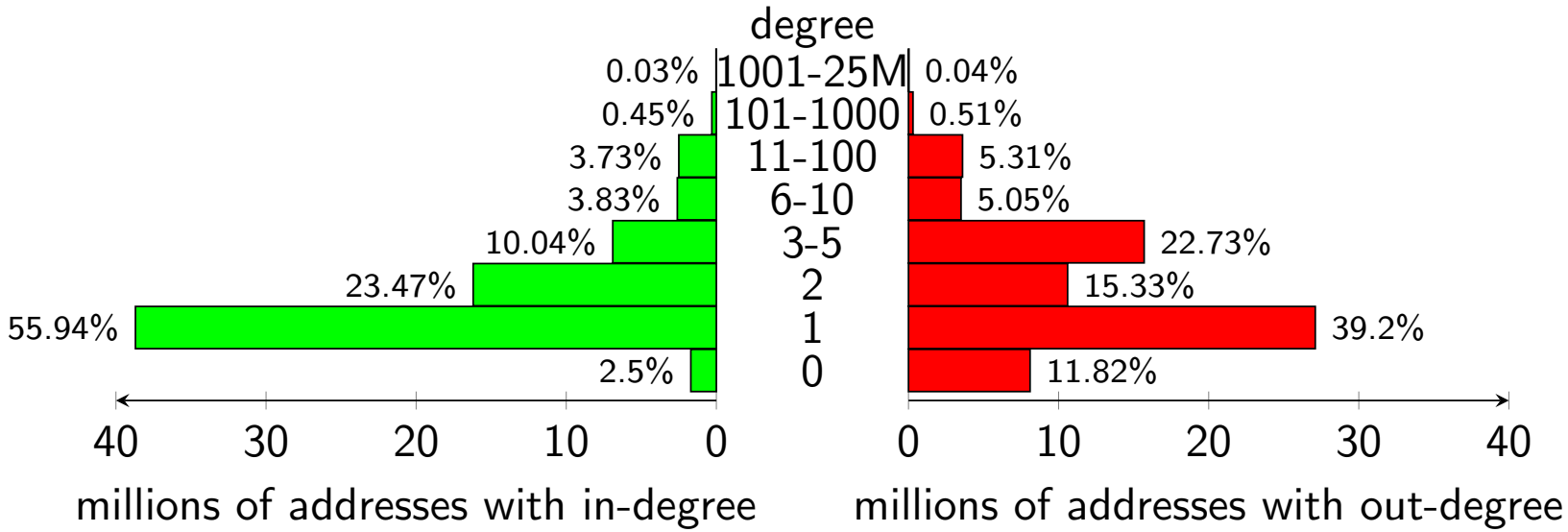


Figure: Degree distributions of transaction graph nodes

Tests

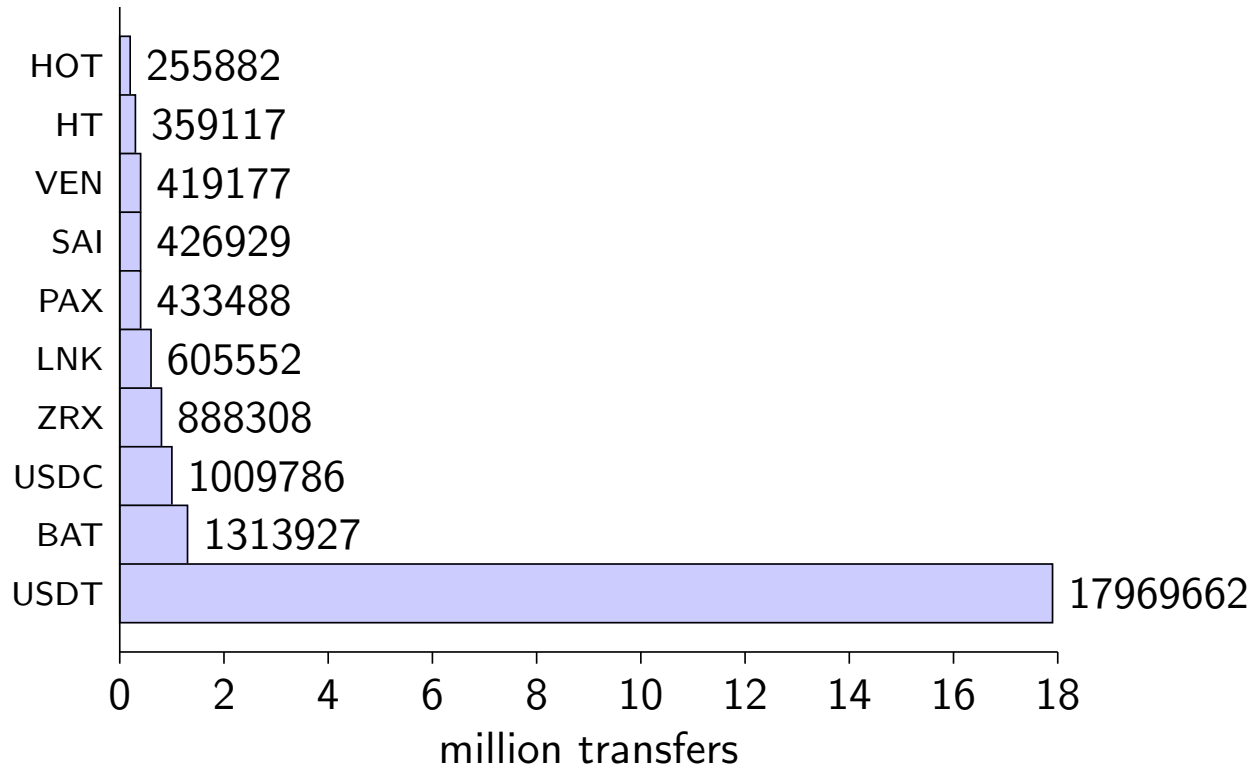


Figure: Numbers of transfer transactions of ten major ERC20 tokens

Conclusion

Applications

- Graph traversal
- Tracing fraudulent activities
- Fast feature extraction of the nodes in transaction graph

Conclusion

- Current Ethereum transaction throughput: 14 tps (transaction per second)
- If tps reaches 1000, there will be 31.5 billion yearly transactions
- To analyze such a huge and dynamically growing graph, a parallel cluster based system is needed
- Our system was able to load and construct adjacency distributed graph representation in about 3 minutes on 16 node Amazon cluster from 70 GB transaction data

Thanks for listening!

Any questions?