SPEEDEX

A Scalable, Parallelizable, and Economically Efficient Decentralized EXchange

Geoff Ramseyer, David Mazières, Ashish Goel

Stanford University

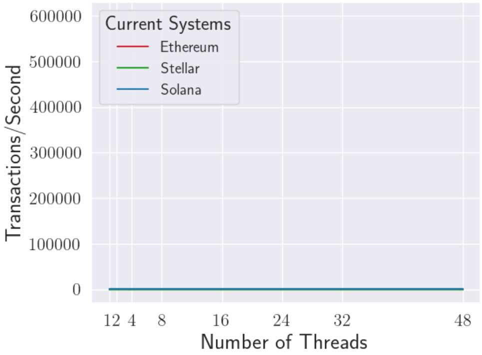
Digital Currency Interoperability

- **Digital currencies are on the horizon**
- **Interoperability will be a crucial challenge**
  - Anyone should be able to pay anyone seamlessly, regardless of currencies
- **Need efficient infrastructure to trade currencies**
- **Shared infrastructure should be jointly operated, not centrally controlled**
  - Replicated state machine with decentralized consensus layer

Is blockchain a good basis for an asset exchange?

Requirements for an Ideal Decentralized Exchange

Computational Performance

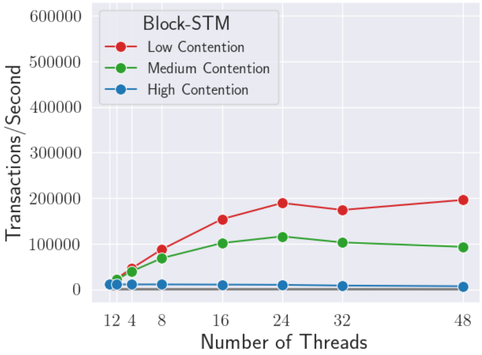


Current Blockchains on Real Traffic
(Source: realtps.net)

- **No! Not if you look at existing blockchains**
  - Dozens of transactions per second

Requirements for an Ideal Decentralized Exchange

Computational Performance

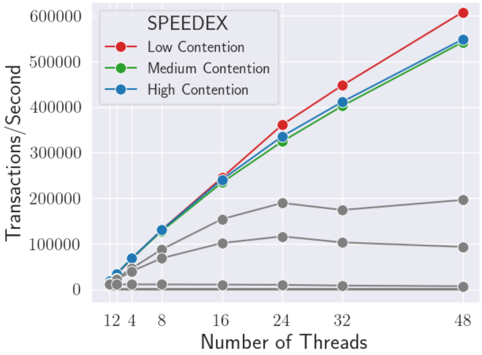


Block-STM [GSXDLM22] on a *payments-only* workload

- **No! Not if you look at existing blockchains**
  - Dozens of transactions per second
- **State of the art systems not sufficiently scalable**
  - Even just for payments

Requirements for an Ideal Decentralized Exchange

Computational Performance

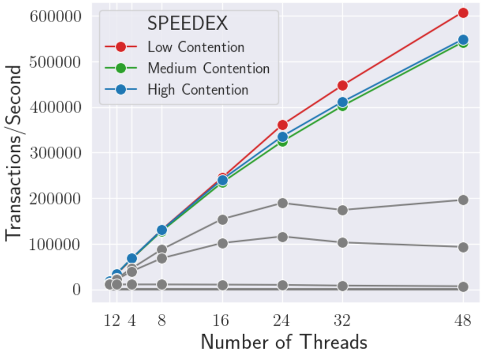


SPEEDEX on a *payments-only* workload

- **No! Not if you look at existing blockchains**
  - Dozens of transactions per second
- **State of the art systems not sufficiently scalable**
  - Even just for payments
- **SPEEDEX gets linear scalability**
- **Exchange is a much harder problem than just payments**

Requirements for an Ideal Decentralized Exchange

Computational Performance

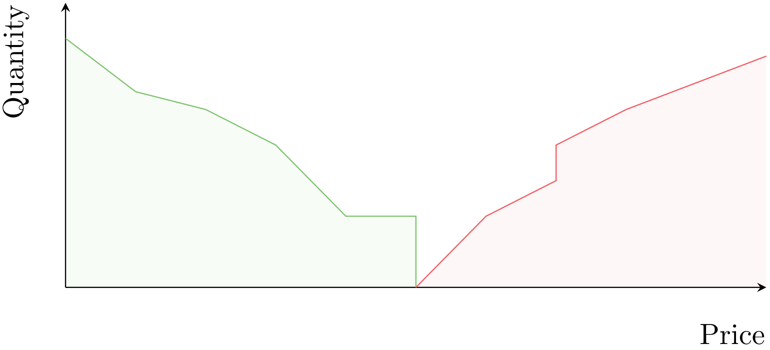


SPEEDEX on a *payments-only* workload

Economic Performance

- **Efficient Liquidity Usage**
- **Fair, Open Access**

Orderbooks



Every offer modifies the orderbook
Every trade can happen at a different price

Decentralized Exchanges Today
Bad in Both Categories

Computational Performance

- **Read-Modify-Update on shared data structures**
- **Worst-case for Optimistic Concurrency Control**

Economic Performance

- **Front-Running**
  - High-Frequency Trading

Sell @ $9

Buy @ $11

- **Suboptimal Liquidity, Cyclic Arbitrage**
  - (next slide)

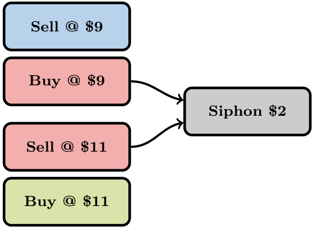
Decentralized Exchanges Today
Bad in Both Categories

Computational Performance

- **Read-Modify-Update on shared data structures**
- **Worst-case for Optimistic Concurrency Control**

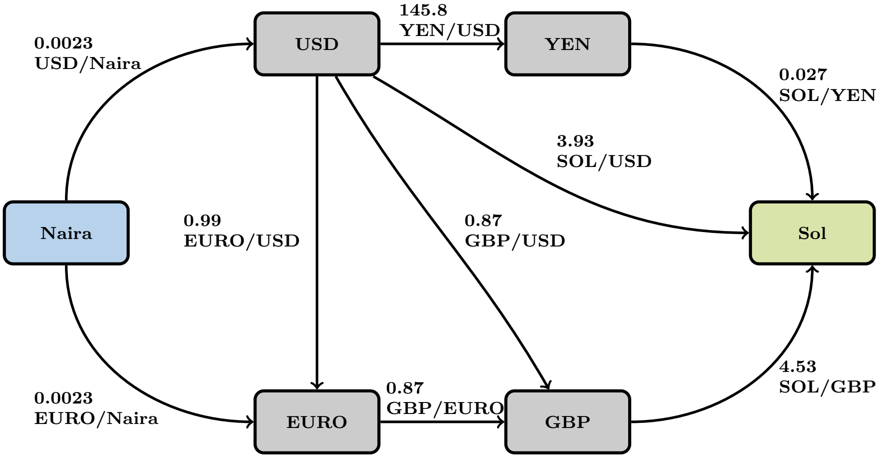
Economic Performance

- **Front-Running**
  - High-Frequency Trading

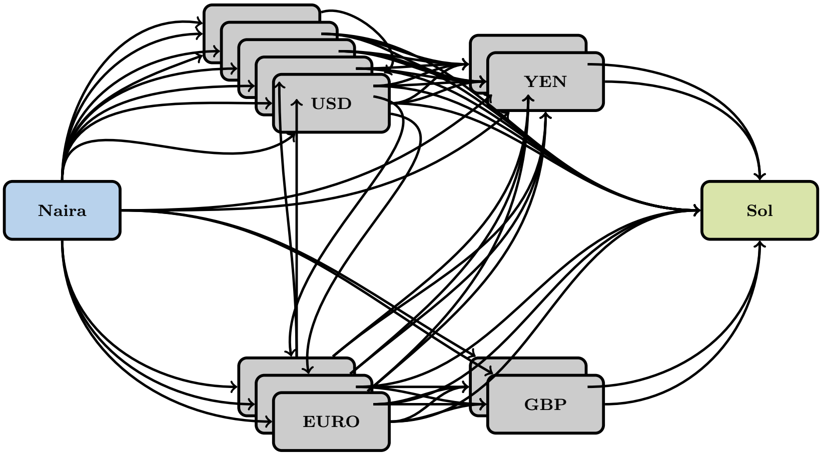


- **Suboptimal Liquidity, Cyclic Arbitrage**
  - (next slide)

Suboptimal Liquidity and Cyclic Arbitrage



Suboptimal Liquidity and Cyclic Arbitrage



SPEEDEX: Batch Trading

- **Input: Block of Offers**

1. **Compute Valuations**
2. **Trade with SPEEDEX at Valuation Quotients**
   - Meaningless units
   - No pairwise matching!

- **"Clearing" if no surplus or debt**

Trade 10 USD for EUR
min $\frac{9}{10} \frac{EUR}{USD}$

Trade 9 EUR for JPY
min $140 \frac{JPY}{EUR}$

Trade 1350 JPY for USD
min $\frac{1}{135} \frac{USD}{JPY}$

Trade 10000 USD for EUR
min $1000 \frac{EUR}{USD}$

SPEEDEX
Pricing Engine

$p_{USD} = 9$
$p_{EUR} = 10$
$p_{JPY} = \frac{1}{15}$

Theorem (Arrow and Debreu, 1954)

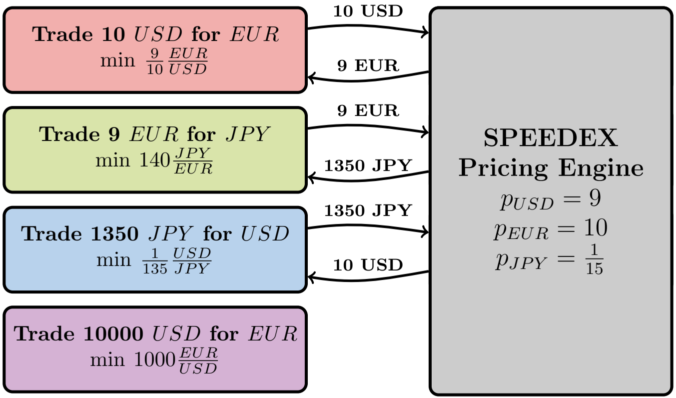
There always exists a unique set of valuations* $\{p_A\}$ *that clears the market.*

SPEEDEX: Batch Trading

- **Input: Block of Offers**

1. **Compute Valuations**
2. **Trade with SPEEDEX at Valuation Quotients**
   - Meaningless units
   - No pairwise matching!

- **“Clearing” if no surplus or debt**



Theorem (Arrow and Debreu, 1954)

There always exists a unique set of valuations $\{p_A\}$ that clears the market.*

Why Uniform Clearing Valuations?

Computational Performance

- **Commutative Trades**
  - Almost entirely hardware atomics
- **Linear Scalability**
  - Scalable Commutativity Rule [CKZMK13]

Economic Performance

- **No In-Batch Front-Running**
  - No need to hyperoptimize limit prices and network latency
  - Everyone gets the same rates
- **No Cyclic Arbitrage / Optimal Liquidity Usage**
  - No need to specify intermediate assets

SPEEDEX In Context

Prior Work

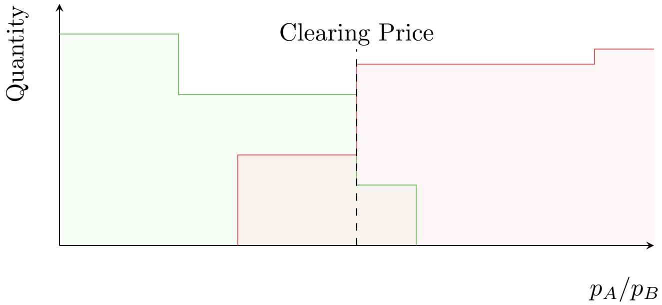
- **2-Asset Batch Trading**
  - Response to High-Frequency Trading [BCS15]
  - NYSE Opening/Closing Auction
- **Many-Asset Batch Trading [CoWSwap]**
  - ✖ Unscalable, Low Throughput

Our Contribution

1. **Compute Valuations at Scale**
2. **Feasibility of Many-Asset Batch Trading**
3. **High-Performance System Design**

Equilibria Computation

- **2-asset case**
  - Binary search
- **Many-asset case**
  - Find simultaneous intersection point of many high-dimensional manifolds over a high-dimensional simplex



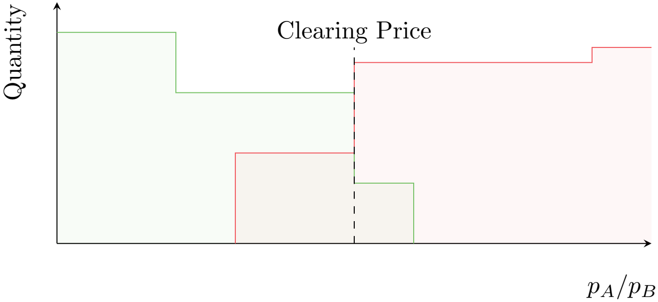
Fast Equilibria Computation
Use Market Structure to Simplify Problem

All Prior Known Algorithms

- **Runtime** *poly*(#*offers*)

Our Design

- **Iterative (based on Tâtonnement [CMV05])**
  - Economics 101 Price Adjustment
- **Iteration runtime** $O(\#assets^2 \lg(\#offers))$
  - Incrementally sort offers by limit price

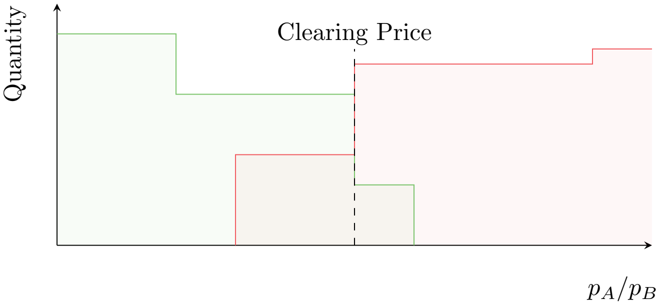


Empirical Results

$\sim 100\mu s$ per iteration, ~ 1000 iterations

Approximate Equilibria Computation
Two Types of Acceptable Approximation

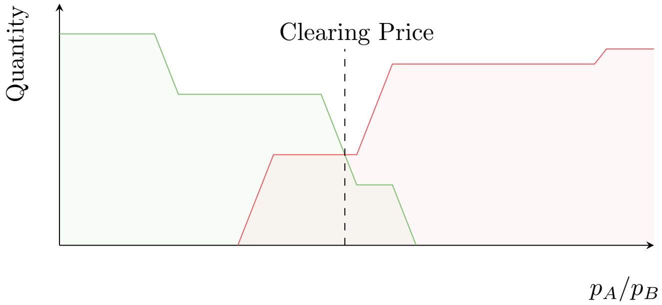
- **Most literature algorithms are approximate**
  - Too many bits to even write down exact equilibria
- **These approximations will do nasty things!**
  - ✖ Mint money
  - ✖ Invalid trades
  - ✖ ...



Approximate Equilibria Computation

Two Types of Acceptable Approximation

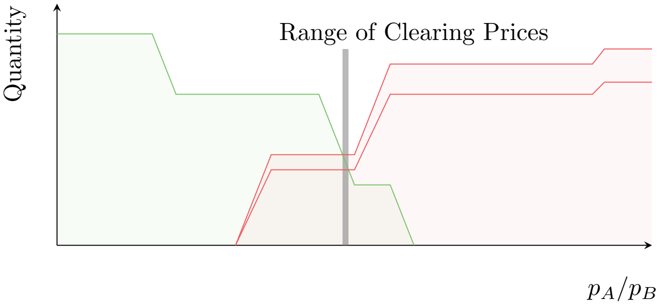
- **Smooth limit price thresholds**
  - Reduces iterative oscillation
- **Trade partially if limit price is *close to* (and below) market rate**
  - Experiments use $2^{-10} \approx 0.1\%$



Approximate Equilibria Computation

Two Types of Acceptable Approximation

- **Charge small, fixed fee**
  - Relaxes "market clearing" to allow surplus, not debt
  - Experiments use $2^{-15} \approx 0.003\%$
- **Range of approximate clearing prices**
  - For computational efficiency, rather than for profit



Accurate Equilibria Computation

- **Linear program turns prices into trades**
  - Efficient to solve (using market structure decomposition)
- **Guarantees approximations take acceptable forms**

$$\max \sum_{A,B} p_A \, x_{AB}$$

$$s.t. \; p_A L_{AB}\left(\frac{p_A}{p_B}\right) \leq p_A x_{AB} \leq p_A U_{AB}\left(\frac{p_A}{p_B}\right) \qquad \forall A, B$$

$$p_A \sum_{B \in [N]} x_{AB} \geq (1 - \text{fee}) \sum_{B \in [N]} p_B x_{BA} \qquad \forall A$$

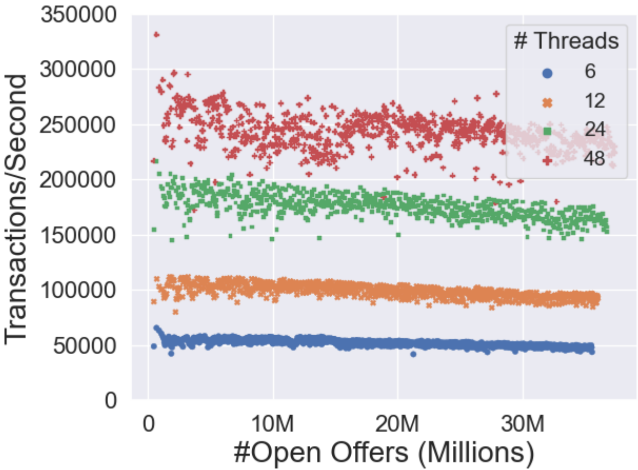
Accurate Equilibria Computation

- **Linear program turns prices into trades**
  - Efficient to solve (using market structure decomposition)
- **Guarantees approximations take acceptable forms**

$$\begin{aligned} \max \ & \text{Trade Volume} \\ s.t. \ & \text{Correct (Smoothed) Offer Execution} \\ & \text{Assets Conserved (after fees)} \end{aligned}$$

Overall Performance

- **48-CPU replicas**
- **Linear Scalability**
  - Contention with background work (logging)
- **Scalability enables adding features but maintaining throughput**
  - 50 assets, 10M accounts, Hashing, Logging,...
- **Log Dependence on** *#offers*



Implementation

- **Self-contained SPEEDEX**
  - https://github.com/scslab/speedex
  - $\sim$ 30,000 LOC C++
- **Prototyped in Stellar (Layer-1 Blockchain)**
  - https://github.com/gramseyer/stellar-core
  - Adds $\sim$ 2000 LOC C++ to Stellar
  - Commutative semantics, not parallel performance
  - This is the only piece that needs a “hard fork”
    - Parallelization does not require coordinated upgrades

SPEEDEX

- **Is blockchain a good basis for an asset exchange?**
  - Not for traditional order matching
- **Yes, if you solve more problems at the same time**
  - Linear scalability via commutativity
  - Eliminate (a common type of) front-running, improve liquidity
- **Scalability can require letting go of traditional semantics, and this can be a good thing in many other ways!**
- **SPEEDEX handles more than 200,000 trades/second on 48-core commodity hardware, with 10s of millions of open offers**

Thank You!