# Utility-Based Routing in Payment Channel Networks: A Tradeoff Between Utility And Privacy

Suhan Jiang and Jie Wu

Dept. of Computer and Information Sciences

Temple University

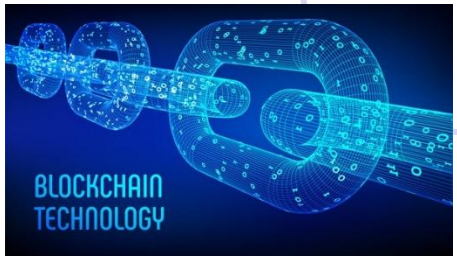
**Fei Zuo**

Department of Computer Science

University of Central Oklahoma

# Outline

1. Payment Channel Networks
2. Utility-Privacy Tradeoff
3. Solution via Utility-based Routing
4. Performance Evaluation
5. Conclusions

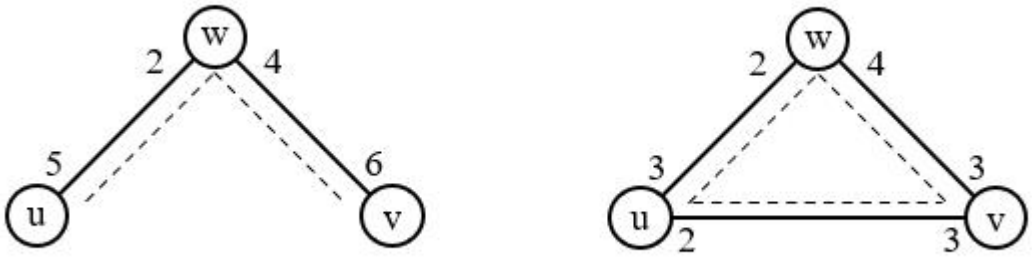


# 1. Payment Channel Networks

- Micropayment channels
  - Quick transactions between trusted neighbors
  - Avoiding block confirmation via offchain payment
- Fund allocations
  - Allocation of node funds to channels
- Bidirectional transactions
  - Fund balance in two directions: *channel capacity*

# Payment Path

- Indirect fund transfer
  - Between two untrusted nodes
- Payment path
  - A sequence of non-repeated trusted neighbors (nodes)
- Types of paths
  - Single-path and multi-path
  - e.g., transfer $4 from u to v



# 2. Utility-Privacy Tradeoff

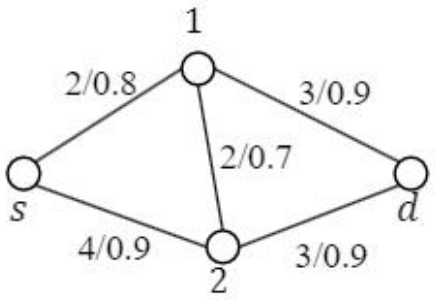
- Transaction fails due to insufficient balance
  - Nodes know the balances of their own channels
  - Only the graph topology, and the sum of balances in either direction on each channel are revealed publicly
  - Some improvements:
    - Split payments, or
    - Make local routing decision
- Tradeoff on not revealing instantaneous balance
  - Prevent observers from inferring others' transaction pattern
  - Guess-and-check routing approach uses unnecessary resources and severely limits the success rate

# 2. Utility-Based Routing (Cont.)

- Optimal route depends not only on the network topology, but also on the benefit value
  - Definition
    - Given a path p, i.e., ⟨s = u0, u1, · · · , uk, d = uk+1⟩, its corresponding expected utility is

$$U = R_p \cdot v - C_p = \left( \prod_{j=0}^{k} r_{j,j+1} \right) \cdot v - \sum_{i=0}^{k} \left( c_{i,i+1} \cdot \prod_{j=0}^{k} r_{j,j+1} \right),$$

  - Example




(a) Network topology.

| Path | $u_d$ | $u_1$ | $u_2$ | $U$ |
|---|---|---|---|---|
| $p_1$ | 20/30 | 15/24 | | 10*/17.2 |
| $p_2$ | 20/30 | | 15/24 | 9.5/17.6* |
| $p_3$ | 20/30 | 8.5/14.8 | 15/24 | 4.8/9.8 |
| $p_4$ | 20/30 | 15/24 | 8.5/14.8 | 3.7/9.3 |

(b) RENUs under different benefit values.

Fig. 1: An example of a simple network.

# 3. Models and Definitions

- Real Channel State: $C_{uv} = (B_{uv}, B_{vu})$
- Noisy Channel State: $\hat{C}_{uv} = (\hat{B}_{uv}, \hat{B}_{vu})$
  - Noise $n_{uv}$ follows a Gaussian distribution: $N(\mu_{uv}, \sigma_{uv}^2)$
  - Revealed balance: $\hat{B}_{uv} = B_{uv} + n_{uv}$
- Routing fee:
  - F $a_i - a_{i+1} = f_i = b_{u_i,u_{i+1}} + r_{u_i,u_{i+1}} \times a_i$ e
    - $b_{u_i, u_{i+1}}$ is the constant base fee charged by the channel from $u_i$ to $u_{i+1}$,
    - $r_{u_i, u_{i+1}}$ is the rate of the fee proportional to the amount to be forwarded

# Utility-based Routing in Noised PCN

- $R_{u,v,a}$: the reliability of channel (u, v) when transferring *a*
  - Taking the variance into consideration

$$R_{u,v,a} = P_{u,v,a} - \sigma^2_{u,v}$$

where $P_{u_{i-1},u_i,a_i} = Prob\left[B_{u_{i-1},u_i} \geq a_i\right]$

$$= Prob\left[\hat{B}_{u_{i-1},u_i} - n \geq a_i\right] = Prob\left[n \leq \hat{B}_{u_{i-1},u_i} - a_i\right]$$

- $npu_i$: node i's NpREN
  - NpREN: the RENU under noised PCN routing
  - Based on the OpRENUs of nodes in i's relay set,

$$npu_i = \sum_{j=i+1}^{i+k} \left( npu_j \cdot r_{i,j} \cdot \prod_{l=i+1}^{j-1} (1 - r_{i,l}) - c_{i,j} \right)$$

# Relay Selection and Prioritization

- Selection
  - Choose neighbors that do not cause loops as downstream neighbors
  - relay selection should consider channel balances
- Ordering

**Theorem 1.** *For any node, once its relays have been determined, the relays should be prioritized in the decreasing order of their corresponding NpRENUs in order to maximize the NpRENU of the node.*

# Relay Selection and Prioritization (Cont.)

**Algorithm 1** NPUTILITY($i$)

**Input:** network topology $G(V, E)$ and node $i$
**Output:** $npu_i$ and an ordered node set

1: Insert downstream neighbor $j$ (s.t.$\hat{B}_{ij} \geq a_j$) to relay set;
2: Sort relays in the decreasing order of npu;
3: Compute corresponding $npu_i$ based on Eq. (4);
4: **return** $npu_i$ and the related relay set with priorities;

- Algorithm 1 uses the downstream neighbors of a node and their corresponding NpRENUs as inputs. However, these inputs are not easy to since this requires the exhaustive exploration of all loop-free paths from source to destination

# Utility-based Routing Optimal Solution

**Algorithm 2** DFSOPTIMAL($i$)

**Input:** network topology $G(V, E)$ and node $i$

1: **if** each $i = d$ **then**
2: **return** $v$ as $npu_i$
3: **for** each neighbor $j$ not in the path from $s$ to $i$ **do**
4: Get $npu_j$, relay set, and order via DFSOPTIMAL($j$);
5: **return** $npu_i$, relay set, and order via NPUTILITY($i$);

- Algorithm 2 recursively explores the current node's downstream neighbors (neighbors not appearing in the path from source to the current node), meaning that the complexity of the optimal solution is very high.

# Utility-based Routing Greedy Solution

**Algorithm 3** DFSGREEDY($i$)

**Input:** network topology $G(V, E)$ and node $i$

1: Initialize $npu_i$ and $u_d$ to $v$
2: **while** $s$'s order is not determined **do**
3: Add $i$ into the set of ordered nodes
4: **if** each $i \neq d$ **then**
5: Call NPUTILITY($i$)
6: **for** each unordered neighbor $j$ **do**
7: Call RELAX($i$, $j$)

**Algorithm 4** RELAX($i$, $j$)

**Input:** node $i$ and node $j$

1: **if** $u_j < u_i \cdot r_{i,j} - c_{i,j}$ **then**
2: Update $u_j$: $u_j \leftarrow (u_i \cdot r_{i,j} - c_{i,j})$
3: Record node $i$ as $j$'s downstream neighbors;

# Utility-based Routing Greedy Solution (Cont.)

- Greedy Solution determines a total order among all nodes
  - For any node, all of its neighbors with higher RENUs are its downstream neighbors
    - not only can the downstream neighbors of any node be uniquely determined
    - but also the NpRENUs of all nodes can be calculated sequentially in the decreasing order of their RENUs.
  - The benefit of the ordering
    - a node's RENU reflects its closeness to the destination in terms of utility
    - with this initial ordering among all nodes, the destination is reachable

# 4. Performance Evaluation

- Custom network topology
  - 100 nodes and 340 edges
  - Based on the BA model

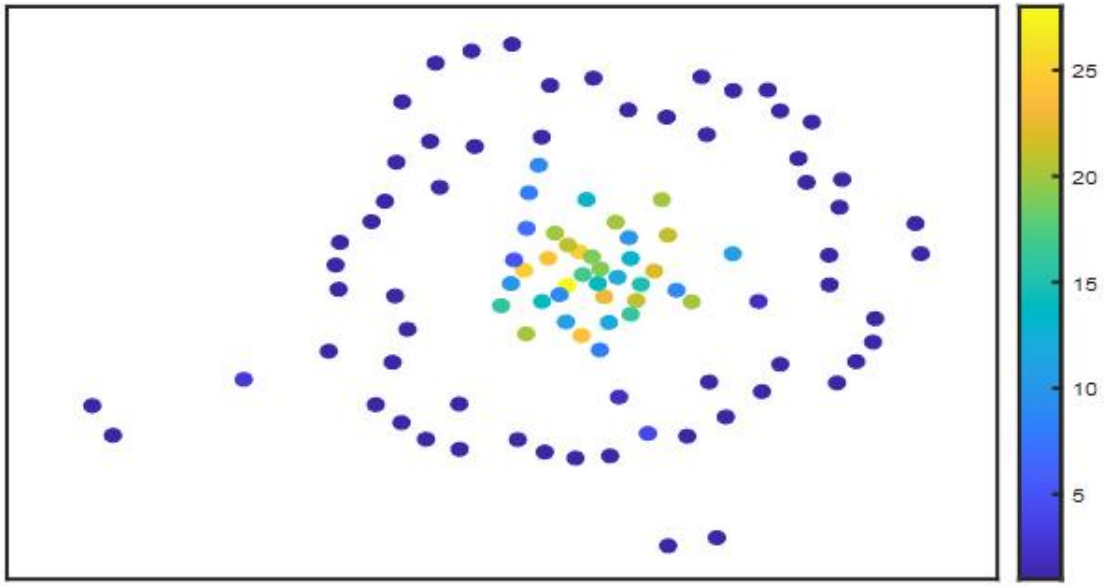
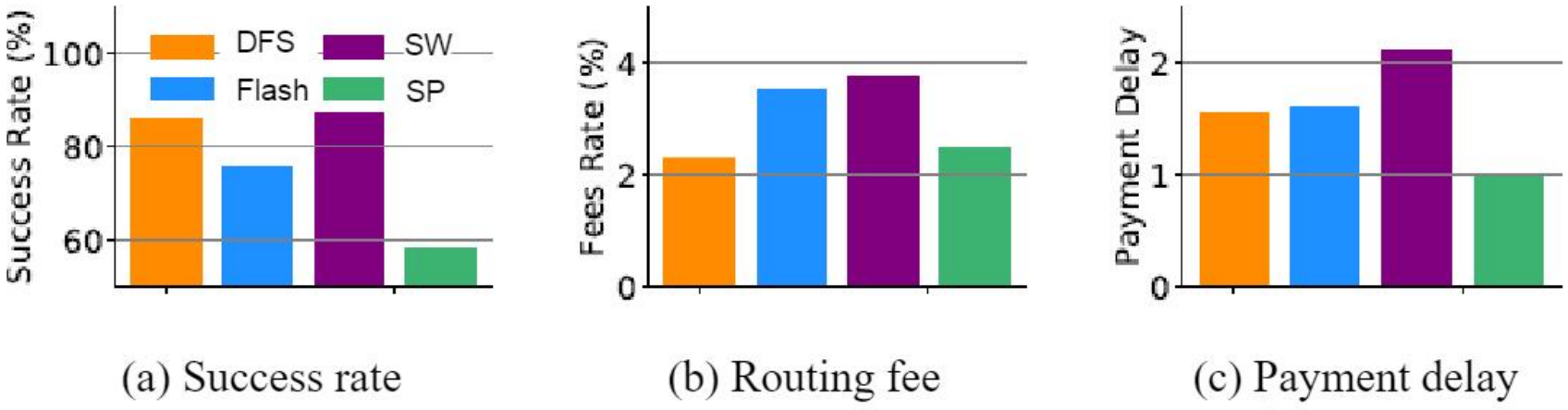



Fig. 3: A custom network of 100 nodes and 340 edges (not shown), where node degrees are represented in colors (yellow: high degree, blue: low degree).

# Network Setup

- Channel capacity
  - channel's capacity is set randomly in [50000, 75000)
- Channel initla balance
  - Randomly balanced
- Transaction size
  - Heterogeneous

# Simulation Summary




(a) Success rate (b) Routing fee (c) Payment delay

Fig. 4: Performance results with routing algorithms.

Success ratio: DFSGREEDY has almost the same payment success rate as SilentWhispers, they are both much higher than the other algorithms

Routing fee: DFSGREEDY has the lowest

Payment delay: that the DFSGREEDY's payment delay is 4% and 27% lower than Flash and SilentWhispers, respectively.

# 6. Conclusions

- We introduce noise mechanisms to the existing PCN design, and define the concept of channel reliability
- We identify the relation between privacy loss and channel noise distribution, which leads to a new binding with routing fee
- We propose a utility-based routing problem in the proposed PCNs, where payment routing aims at utility maximization rather than cost minimization
- Both optimal and heuristic solutions are provided and implemented in a distributed way
- We conduct experiments on the LN simulator CLoTH to evaluate our algorithm