Pricing for Routing and Flow-Control in Payment Channel Networks

Suryanarayana Sankagiri
EPFL, Switzerland
suryanarayana.sankagiri@epfl.ch

Bruce Hajek
University of Illinois, Urbana-Champaign
b-hajek@illinois.edu

1. INTRODUCTION

Blockchains are decentralized digital transaction systems. Most blockchains today suffer from poor transaction throughput, resulting in exorbitant transaction fees and hindering widespread adoption. *Layer-two blockchain mechanisms* are tools that allow transactions to take place outside of the main blockchain system, thereby increasing the system's throughput [2]. A payment channel network (PCN) is one such mechanism that is used in practice. This paper focuses on their long-term transaction processing efficiency.

As the name suggests, a PCN is a network composed of multiple *payment channels*. A payment channel is essentially an escrow fund; in other words, it is a special account in which two parties have partial stake and partial control. Any two blockchain users can establish a payment channel by depositing some funds into such an account. Once a channel is established, the users transact between themselves by exchanging messages between themselves; these transactions are not recorded on the blockchain.

By forming a network of payment channels, users who do not share a channel can transact by routing their transaction through an intermediary. However, the network brings with it the additional challenge of *routing*. An important factor while routing is the capacity of the available paths. Indeed, payment channels are limited in their ability to carry transactions by the amount of funds they hold. There has been extensive prior work on routing in PCNs. In particular, [4] proves that a well-designed routing protocol can perpetually serve a 'circulation demand' (where each user receives as much money as it pays out). However, their protocol cannot handle arbitrary demands. In addition, prior work does not explore the effect of *flow-control*, *i.e.,* dropping certain transaction requests. Flow-control can impact the long-term efficiency of a PCN because serving a transaction alters the feasibility of future transactions.

In this work, we present a joint flow-control and routing protocol for PCNs, which we call the *DEBT (DEtailed Balance Transaction) control protocol*. The goal of the protocol is to maximize the total utility of all users in the network, subject to some constraints imposed by the network. In this protocol, each channel quotes a price to the network; a path's price is then the sum of the channel prices along it. Based on these path prices, transacting node-pairs make flow-control and routing decisions. Channels keep updating their prices over time based on the transactions that flow through them. Under stationary demand conditions, we show that the protocol guides prices and transaction flows toward an optimal operating point.

2. MODEL

A payment channel network consists of a set of nodes V and a set of channels E between pairs of nodes. The nodes are numbered $1, 2, \ldots, |V|$. A channel connecting nodes u and v is denoted (u, v). Each channel has a certain capacity, which refers to the total amount of money escrowed in the channel. Let $c_{u,v}$ denote the capacity of channel (u, v) and let $c \in \mathbb{R}_+^E$ be a vector denoting the capacities of all the channels in the network.

We model the PCN as a discrete-time dynamical system whose state is described by the balances in each of the channels. At any given time t, let the balance of node u in channel (u, v) be $x_{u,v}[t]$. It follows that the balance of v in the same channel is $c_{u,v} - x_{u,v}[t]$. By convention, x contains the balance of the smaller-indexed node of each channel; the balance at the opposite end of the channel is inferred from its capacity. The state vector always satisfies $0 \leq x[t] \leq c$ (the inequalities hold component-wise).

We assume that in each slot, between every transacting node-pair, a single transaction request arrives. Let $a_{i,j}[t]$ denote the monetary value of the transaction request from source i to destination j in slot t. In general, the requested transaction amounts could vary arbitrarily over time. For the sake of simplicity, in this work, we focus on the regime of *constant demands*, *i.e.,* we assume that $a_{i,j}[t] = a_{i,j}$ for all time t and all transacting node-pairs (i, j). The vector $a = (a_{i,j})_{(i,j)\in V\times V}$ is called the *demand vector*.

A second assumption we make is that the transaction demand arriving to the payment channel network is elastic. In other words, node-pairs prefer to have the entire transaction be served by the PCN, but it is acceptable that the request is dropped or partially served. We model elastic transaction requests by means of a utility function. We assume that the node-pair (i, j) gains a utility of $U_{i,j}(q_{i,j}[t])$ upon being served a transaction of amount $q_{i,j}[t] \in [0, a_{i,j}]$ by the network. We assume that $U_{i,j}(\cdot)$ is a concave, differentiable, and nondecreasing function over $[0, a_{i,j}]$. We also assume that $U_{i,j}(0) = 0$ and $U'_{i,j}(0) < \infty$.

We assume that every node pair has a fixed set of paths between them that they consider for routing transactions. Denote the k^{th} path by $p_{i,j,k}$ and the set of such paths by $P_{i,j}$. Note that a path from i to j is different from a path from j to i. Let P denote the set of all paths ($P = \cup_{i,j} P_{i,j}$). Let R denote the $P \times E$ routing matrix with entries in $\{-1, 0, 1\}$

constructed using the convention given in [3]. In R, each row corresponds to a particular path and each column corresponds to a particular channel.

With every path in the network, we associate a flow, which represents the amount of money sent along that path over a period of time. Let $f_{i,j,k}[t]$ denote the amount of money being sent on path $p_{i,j,k}$ in slot t. The amounts of money sent from node i to node j along all possible paths in slot t is denoted by $f_{i,j}[t]$, *i.e.*, $f_{i,j}[t] \triangleq (f_{i,j,1}[t], \ldots, f_{i,j,k}[t])$. The total amount of money sent from i to j in slot t is denoted by $q_{i,j}[t]$. Thus $q_{i,j}[t] \triangleq \Sigma_k f_{i,j,k}[t]$. Finally, let $f[t] \in \mathbb{R}^P$ denote the set of all the flows in the network in slot t.

With this notation in place, we see that at the end of each slot, the balances are updated as:

$$x[t+1] = x[t] - Rf[t]. \tag{1}$$

If a requested flow vector is not feasible (*i.e.*, $x[t]$ becomes negative or exceeds the capacity), a channel redistributes its balances evenly by performing an expensive *rebalancing operation* (see [2]). Perpetual rebalancing is not viable for a payment channel network. In particular, the network should aim to achieve a *detailed balance flow* in steady state. A flow vector f is said to be a detailed balance flow if $Rf = 0$, which means that the amount of money flowing through each channel is equal in the two opposite directions.

3. THE DEBT CONTROL PROTOCOL

3.1 A Network Utility Maximization Problem

To formulate the protocol's objective, we introduce some notation. Let $\mathcal{N}$ denote the set of transacting node-pairs, *i.e.*, $\mathcal{N} = \{(i,j) : a_{i,j} > 0\}$. Let $U(f) = \sum_{(i,j)\in\mathcal{N}} U_{i,j}(q_{i,j})$ denote the total utility of all transacting node-pairs as a function of a stationary flow f. Let A denote the set of non-negative flows satisfying the demand constraints:

$$A \triangleq \{f : f \geq 0,\ f_{i,j} \leq a_{i,j}\ \forall\ (i,j) \in \mathcal{N}\}. \tag{2}$$

Define a *feasible flow* to be any flow that meets both the demand constraints and the detailed balance constraints (*i.e.*, the condition $Rf = 0$).

For certain technical reasons, we add a quadratic regularizer term, $H(f)$, to the utility function, where

$$H(f) \triangleq -\sum\nolimits_{(i,j)\in\mathcal{N}} \eta_{i,j} \sum\nolimits_{k=1}^{|P_{i,j}|} (f_{i,j,k})^2. \tag{3}$$

Here, $\eta_{i,j} \geq 0$ controls the weight of the regularizer.

The protocol's goal is to find a feasible stationary flow that maximizes the net utility of the payment channel network. In mathematical terms, this can be expressed as obtaining a solution to the following optimization problem:

$$\max_{f\in A} \quad U(f) + H(f) \quad \text{such that} \quad Rf = 0 \tag{\textbf{P}}$$

The symbol (**P**) denotes that the optimization problem presented above is the *primal* (or original) problem. Let f^* denote any solution to this problem.

Observe that the set of feasible flows is a compact, convex set. Moreover, it is nonempty for any problem parameters, since the empty flow ($f = 0$) is a feasible flow. Therefore, a solution to (**P**) always exists. Also note that (**P**) is a convex optimization problem, since both $U(f)$ and $H(f)$ are concave and the constraint set is convex. Lastly, if all $\eta_{i,j}$ are strictly positive, then the objective function is strongly concave. This ensures that f^* is unique.

3.2 The Dual Problem

The primal problem (**P**) is hard to solve because of the detailed balance constraints. With the aid of Lagrange multipliers, we derive its *dual problem* that does not explicitly have these constraints. Let $\lambda_{u,v}$ denote the Lagrange multiplier for the constraint $(Rf)_{u,v} = 0$; let $\lambda \in \mathbb{R}^E$ denote the vector of all such terms. Define the Lagrangian of the problem (**P**) by

$$L(f,\lambda) \triangleq U(f) + H(f) - \lambda^T Rf. \tag{4}$$

Using the Lagrangian, we can formulate an equivalent form of the problem (**P**) as follows:

$$\max_{f\in A} \inf_{\lambda\in\mathbb{R}^E} \quad L(f,\lambda) \tag{5}$$

The dual of the optimization problem (**P**) is obtained by changing the order of minimization and maximization in (5):

$$\inf_{\lambda\in\mathbb{R}^E} \max_{f\in A} \quad L(f,\lambda) = \inf_{\lambda\in\mathbb{R}^E} D(\lambda), \tag{\textbf{D}}$$

where $D(\lambda)$, called the *dual function*, is defined as follows:

$$D(\lambda) \triangleq \max_{f\in A} U(f) + H(f) - \lambda^T Rf \tag{6}$$

For any λ, $L(f,\lambda)$ is finite for all $f \in A$, because A is a bounded set. Therefore, $D(\lambda)$ is well-defined for all $\lambda \in \mathbb{R}^E$. The dual function is a convex function of λ [1].

3.3 A Dual Algorithm

Since the dual problem is an unconstrained convex optimization problem, it is easy to solve using the gradient descent method. To do so, we need to establish conditions under which $D(\lambda)$ is differentiable and also obtain an expression for the gradient of $D(\lambda)$. Lemma 1 gives us an expression for the subdifferential of $D(\lambda)$. Because $D(\lambda)$ is a convex function, the subdifferential set is nonempty at all points. $D(\lambda)$ is differentiable precisely at those points where the subdifferential set has a unique element. The lemma follows immediately from Danskin's theorem, also known as the envelope theorem. (See Appendix B of [1]).

LEMMA 1. *Let $D(\lambda)$ be the function as defined in (6). The subdifferential set of $D(\lambda)$ is given by*

$$\partial D(\lambda) = \{\nabla_\lambda L(f,\lambda) : f \in F(\lambda)\} = \{-Rf : f \in F(\lambda)\}$$

where $F(\lambda) \triangleq \arg\max_{f\in A} L(f,\lambda)$ is the set of all flow vectors that maximize the Lagrangian, given λ.

The gradient descent algorithm to solve the dual problem is presented below. Initialize the algorithm by setting $\lambda[0]$ to be the zero vector. For every $t \in \mathbb{N}$, set

$$\begin{aligned} f[t] &= \arg\max_{f\in A} L(f,\lambda[t]) \\ \lambda[t+1] &= \lambda[t] + \gamma Rf[t] \end{aligned} \tag{\textbf{A}}$$

Here, γ is a strictly positive stepsize parameter in the algorithm that remains constant for all time. In each iteration t, the flows $f[t]$ are set so as to maximize the Lagrangian, given the current values of $\lambda[t]$, while the Lagrange multipliers $\lambda[t+1]$ are updated in a direction opposite to the gradient of $D(\lambda[t])$. In case there is more than one value of f that maximizes $L(f,\lambda[t])$, we can set $f[t]$ to any such value. In this case, algorithm (**A**) is equivalent to the subgradient method applied to $D(\lambda)$.

3.4 Dual Algorithm to a Network Protocol

The first step towards a decentralized implementation of (**A**) is to observe that the Lagrangian is a sum of terms, each concerning one transacting node-pair. The Lagrangian depends on λ only through the term $R^T\lambda$ (see (4)). Define $\mu \triangleq R^T\lambda$. Then μ is a vector indexed by the paths:

$$\mu_{i,j,k} = \sum\nolimits_{u\to v\in p_{i,j,k}} \lambda_{u,v} - \sum\nolimits_{v\to u\in p_{i,j,k}} \lambda_{u,v}. \quad (7)$$

Similar to the notation $f_{i,j}$, define $\mu_{i,j} \triangleq (\mu_{i,j,1},\ldots,\mu_{i,j,k})$. Further, define $\tilde{L}(f,\mu)$ to be

$$\tilde{L}(f,\mu) \triangleq \sum\nolimits_{(i,j)\in\mathcal{N}} L_{i,j}(f_{i,j},\mu_{i,j}), \text{ where} \quad (8)$$

$$L_{i,j}(f_{i,j},\mu_{i,j}) \triangleq U_{i,j}(q_{i,j}) - \sum_{k=1}^{|P_{i,j}|} f_{i,j,k}\mu_{i,j,k} + \eta_{i,j}(f_{i,j,k})^2$$

Next, observe that in (**A**), the flows are chosen by solving:

$$f[t] = \arg\max_{f\in A} L(f,\lambda[t]) = \tilde{L}(f,\mu[t]); \quad \mu[t] = R^T\lambda[t].$$

The expression of $\tilde{L}(f,\mu)$ given in (8) shows that given μ, the flows between each $(i,j)\in\mathcal{N}$ can be determined independently for each node-pair by solving:

$$f_{i,j}[t] = \arg\max_{\{f_{i,j}:\, f_{i,j,k}\geq 0\,\forall\, k,\ q_{i,j}\leq a_{i,j}\}} L_{i,j}(f_{i,j},\mu_{i,j}[t]) \quad (9)$$

Next, we interpret $\lambda_{u,v}$ as the *channel price*, i.e., the cost of routing one unit of flow in the direction $u\to v$ through the channel (u,v). Let the price for routing flows in the opposite direction of the same channel be $-\lambda_{u,v}$. Then $\mu_{i,j,k}$ is the *path price*, *i.e.*, the cost that the node pair (i,j) needs to pay to send a unit flow along the path $p_{i,j,k}$. It follows that $f_{i,j,k}\mu_{i,j,k}$ is the cost of sending a flow of amount $f_{i,j,k}$ along the path $p_{i,j,k}$. Thus, $\sum_k f_{i,j,k}\mu_{i,j,k}$ is the total cost incurred by the node-pair (i,j) for splitting the total flow amount $q_{i,j}$ along different paths.

In $L_{i,j}(f_{i,j},\mu_{i,j})$, this cost is subtracted from the utility gained by the node-pair (i,j) in executing a transaction of amount $q_{i,j}$ (see (8)). The quadratic term $\eta_{i,j}\sum_k(f_{i,j,k})^2$, coming from the regularizer, can be interpreted as a penalty for concentrating all flows along a single path or as an incentive to split the total flow along different paths. This is because for any fixed value of $q_{i,j}$, the sum $\sum_k(f_{i,j,k})^2$ is minimized by splitting $q_{i,j}$ equally among all $f_{i,j,k}$. Thus, the interpretation of (9) is that each node-pair tries to maximize its net utility, i.e., the utility of executing a transaction with the cost of execution subtracted from it.

We now show how solving (9) can be interpreted as simultaneously making routing and flow-control decisions. First, consider the case when $\eta_{i,j}$ is zero. In this case, it is optimal to route the transaction only along the path with the minimum price; all other paths from i to j carry zero flow. Let $\mu^*_{i,j}[t]$ denote the minimum path price. The amount of flow carried by this path, $q_{i,j}[t]$, is given by:

$$q_{i,j}[t] = \arg\max_{q\in[0,a_{i,j}]} U_{i,j}(q) - q\mu^*_{i,j}[t]. \quad (10)$$

The choice of the total amount of flow is interpreted as a flow-control action and the choice of the path to carry the flow is interpreted as a routing decision.

Now consider the general case. The solution to (9) can be expressed in terms of the classical *waterfilling scheme*. To illustrate this, we invoke the idea of Lagrange multipliers once again to deal with the constraints in (9). Let $\nu_{i,j}$ denote the Lagrange multiplier for the demand constraint $q_{i,j}\leq a_{i,j}$. By the KKT conditions [1], the optimal solution to (9) must satisfy:

$$f_{i,j,k} = \left(\frac{U'_{i,j}(q_{i,j}) - \nu_{i,j} - \mu_{i,j,k}}{2\eta_{i,j}}\right)^+ \ \forall\ k. \quad (11)$$

The total flow $q_{i,j}$ must also satisfy the demand constraint $(q_{i,j}\leq a_{i,j})$ and the complementary slackness condition: $(q_{i,j}-a_{i,j})\nu_{i,j}=0$. Lastly, each $\nu_{i,j}$ must be nonnegative.

4. CONVERGENCE ANALYSIS

We conclude this paper by showing that, with sufficiently small step sizes, the flows under the DEBT control protocol converge to the optimal flow for the network (Proposition 1). The proof of this result follows from standard results on gradient descent on smooth convex functions as well as Lemmas 2 and 3. The first lemma establishes that the dual problem always has a (finite) solution, and the optimal flow in response to such a solution is primal optimal. The lemma follows from Proposition 3.4.2 of [1], which states sufficient conditions for strong duality to hold. The second lemma establishes conditions under which the dual function is smooth. This result follows from standard properties of the Fenchel conjugate of convex functions.

LEMMA 2. *For any instance of the primal problem* (**P**)*, the corresponding dual problem* (**D**) *has a solution, i.e., there exists* $\lambda^*\in\mathbb{R}^E$ *such that* $D(\lambda^*) = D^* \triangleq \inf_{\lambda\in\mathbb{R}^E} D(\lambda)$*. Further, the set* $F(\lambda^*) = \arg\max_{f\in A} L(f,\lambda^*)$ *contains a solution to* (**P**)*.*

ASSUMPTION 1. $\eta_{i,j}\geq\eta>0\ \forall\ (i,j)\in\mathcal{N}$.

LEMMA 3. *Under Assumption 1:*

- $F(\lambda) = \arg\max_{f\in A} L(f,\lambda)$ *is a single-valued, continuous function for all* λ*.*
- $D(\lambda)$ *is smooth with parameter* $\|R\|^2_{op}/\eta$*, where* $\|\cdot\|_{op}$ *denotes the operator norm of a matrix.*

ASSUMPTION 2. *The stepsize of* (**A**) *satisfies* $\gamma<\eta/\|R\|^2_{op}$.

PROPOSITION 1. *Under Assumptions 1 and 2:*

- $D(\lambda[t]) - D(\lambda^*) \leq \frac{\|\lambda^*\|}{2\gamma t}\ \forall\ t\geq 1$.
- $\lambda[t]\to\lambda^{**}$ *for* $\lambda^{**}\in\arg\min_{\lambda\in\mathbb{R}^E} D(\lambda)$ *as* $t\to\infty$.
- $f[t]\xrightarrow{t\to\infty} f^*$*, where* f^* *is the unique solution to* (**P**)*.*

5. REFERENCES

[1] BERTSEKAS, D. *Nonlinear Programming.* Athena Scientific, 1999.

[2] GUDGEON, L., ET AL. Sok: Layer-two blockchain protocols. In *International Conference on Financial Cryptography and Data Security* (2020), Springer, pp. 201–226.

[3] SIVARAMAN, V., ET AL. The effect of network topology on credit network throughput. *Performance Evaluation 151* (2021).

[4] VARMA, S. M., AND MAGULURI, S. T. Throughput optimal routing in blockchain based payment systems. *IEEE Transactions on Control of Network Systems 8*, 4 (2021).