MS&E 339 Fall 2022-2023
Week 2: Algorithms for Decentralized Finance*

Ashish Goel, Bryan Cai, Rebecca Jia, Thanawat Sornwanee

October 24, 2022

1 Types of Currency Systems

In modern economies, money is used as a common medium for exchanging goods and services. When individuals and businesses then use this currency to facilitate transactions, they are implicitly trusting the central bank. On the other end of the spectrum, bartering systems directly trade goods and services without the need for common currency, and require no trust or currency. Today we study a third model of credit, known as **credit networks**, which is a decentralized model where individuals create their own currency and use it to trade. Note that not every pair of individuals will be willing to trade with each other, but as long as a chain of credit exists that connects the two, a transaction can be made. In summary:

1. Barter: If I need a goat from you, I had better have the blanket that you are looking for. This problem is intuitive, but it has low liquidity. The transaction can occur only when the products do exists and the change in the utility of each person getting a new product and losing one's old product has to be (weakly) positive.

2. Centralized banks: Issue currencies, which are essentially IOU's from the bank. This system has a very high liquidity, since it allows strangers to trade freely. However, this system will require a high trust on the centralized banks. In this system, a trade occurs with currencies as the mediums. The seller does not have to get a better utility immediately after the trade, since he can later use the currencies he receives as the payment to purchase a desirable good from other people, who also have trust in the centralized banks.

3. Credit Networks: Bilateral exchange of IOU's among friends.

*For details see `https://web.stanford.edu/\protect\unhbox\voidb@x\protect\penalty\@M\{}ashishg/msande339/`

2 Credit Networks

2.1 What is a Credit Network?

Credit networks are typically presented as graphs $G = (V, E)$ where each node in V represents an agent that can produce its own currency[1], and edges represent trust between agents. An edge (u, v) has an associated *credit limit* $c_{uv} \geq 0$ that represents how much u is willing to trust v. If $c_{uv} = 0$, v will not be trustworthy to u in the sense that u will not accept any payment from v currency anymore. However, note that v can buy some product from u if there exists other trustworthy medium.

When a direct transaction occurs, that amount is deducted from the appropriate edge and added to the reverse edge. For example, if u receives 10 IOU's from v, then c_{uv} decreases by 10 and c_{vu} increases by 10. This credit c_{uv} is replenished when payments are made in the other direction. Again, payments can occur between nodes that do not have edges between them as long as a chain of trust exists between the nodes.

Every node in the network is vulnerable to go default only from its own neighbors, and only for the amount it directly trusts them for.

2.2 Liquidity

We next study the question of liquidity: after a period of transactions, does the network remain in a healthy, connected state, or do nodes quickly become isolated? We sill use the following model:

- Each edge (u, v) has integer directional trust capacities $c_{uv}, c_{vu} \geq 0$ (summing up both directions).
- Transaction rate matrix $\Lambda = \{\lambda_{uv} : u, v \in V, \lambda_{uu} = 0\}$. Note that the transaction only occurs within the system so there is no sink or source.
- At an infinitesimal time step δt, a unit payment transaction of (u, v) occurs with probability $\lambda_{uv}\delta t$.
- A unit transaction of (u, v) fails if and only if there is no sequence of nodes $u = x_1, x_2, \ldots, x_k = v$ such that

  $$c_{x_i x_{i+1}} \geq 1 \text{ for all } i \in \{1, 2, \ldots k-1\}.$$

  If a sequence satisfying the above inequalities exists, we call the corresponding path as a feasible path.

For simplicity, we can further assume that all payments are routed along the shortest feasible path. What happens when there are multiple shortest paths? Could there potentially be more optimal routing strategies? To further analyze credit networks, the following definition is useful.

[1] assume all currency exhange ratios to be unity

Definition 2.1 Cycle-Reachability

Let $\mathcal{S}$ and $\mathcal{S}'$ be two states of the network. We say that $\mathcal{S}'$ is cycle-reachable from $\mathcal{S}$ if the network can be transformed from state $\mathcal{S}$ to state $\mathcal{S}'$ by routing a sequence of payments along feasible cycles (i.e. from a node to itself along a feasible path).

With the concept of cycle-reachability, we can prove the following theorem that will answer our previous question.

Theorem 2.2 (Path equivalence) *Let $(s_1, t_1), (s_2, t_2), ..., (s_T, t_T)$ be the set of transactions of value $v_1, v_2, ..., v_T$ respectively that succeed when the payment is routed along the shortest feasible path from s_i to t_i. Then the same set of transactions succeed when the payment is routed along any feasible path from s_i to t_i.*

Proof: Sending a unit of flow along two different paths from the same source to the same destination leads to two states that are cycle-reachable. ■

However, two different paths may not lead to the same realization of the network. This suggests that the choice of feasible chain for a transaction to be conducted is not negligible.

In other words, the transition probability between two realizations of a network will not be well defined, since the choice of feasible path is not specified in the model.

Even if we enforce least distance path, there can still be an implication of tie-breaking algorithm. Even if we have also specified a tie-breaking rule, another problem will be that we will lose the symmetric in the transition matrix between different realizations of a network. In other words, we want to use different types of state in order to preserve the symmetric property from the symmetry of the transaction rate matrix.

Because $\mathcal{S}'$ is cycle-reachable from $\mathcal{S}$ if and only if $\mathcal{S}$ is cycle-reachable from $\mathcal{S}'$, *cycle-reachability* partitions all possible states of the credit network into equivalence classes.

Theorem 2.3 (Steady-state) *If the transaction rates are symmetric, then the network has a uniform steady-state distribution over all reachable equivalence classes.*

Note that this theorem relies on the fact there can be only finite equivalence classes, since all capacities are non-negative integers.

Another important question to ask oneself is whether the initial state matters for the steady state if we already know the network structures and dynamics, which are specified by the transition matrix and the observation of the trust capacity of each edge (the sum of both directional trust capacities, which is invariant across time).

Although one may think that it does not matter due to the ergodicity, there is still a case where the initial state is important. The main implication come from that the uniform distribution in the steady state is upon all reachable equivalence classes. Although it is obvious that equivalence classes are highly related to the trust capacity of each edge , which we assumed to be known, the actual initial allocation of the directional trust capacity.

For example, let's consider a network of 2 nodes called u and v. Let specify $c_{uv} + c_{vu} = 1$, and the transition matrix to be a zero matrix. Thus, we can see that, if $c_{uv} = 1, c_{vu} = 0$, the steady state will be that the state of $c_{uv} = 1, c_{vu} = 0$ will occur with probability 1, and,

if $c_{uv} = 0, c_{vu} = 1$, the steady state will be that the state of $c_{uv} = 0, c_{vu} = 1$ will occur with probability 1.

Although this may raise a concern, it can be seen that, for any network, with non-zero transition component for each edge with non-zero trust capacity, the reachable equivalence classes will be independent from the initialization of the directional trust capacity.

2.2.1 Examples

Two node network Assume the capacity along the edge is c. Then we have $c + 1$ states; each in a different equivalence class. A transaction is successful if the credit limit in that direction is not empty, so the success probability for a transaction is $\frac{c}{c+1}$.

Tree networks Trees have no cycles, and so each state is its own equivalence class. Hence, all states are equally likely. Let $c_1, c_2, ..., c_L$ be the capacities along the path from s to t in the tree. A transaction is successful if none of the credit limits along the path are 0, so the success probability is

$$\prod_{i=1}^{L} \frac{c_i}{c_i + 1}.$$

General graph Assume capacity $c = 1$ on each edge, and that the Markov chain is ergodic. Let d_v denote the degree of node v. Then, the stationary probability that v is bankrupt is at most $\frac{1}{1+d_v}$.

3 Strategic Formation of Credit Networks

In the previous section, we took the "trust values" or "credit capacities" on the edges in the network to be given by an exogenous process, i.e., the nodes take these values as given. It is possible that these credit capacities are misapportioned, i.e. a lot of credit capacity is allocated along edges that are not crucial for transactions to succeed, while a lot of transactions fail because of inadequate credit capacities.

However, in reality, the network structure, especially in a lightning network, is not given exogenously but rather is endogenously created by the stakeholders (i.e. the nodes) within the system. This leads to the issue of strategic network formation – how do nodes decide how much initial "trust" to establish along each edge? And does that lead to a credit network that is a good match for the underlying transaction rates?

In a vanilla credit network, trust can be initiated by different mechanisms, which can also include personal relationships, which may be difficult to model. However, there exist some types of credit networks where the allocation of trust can be easily quantified, such as the Lightning network.

3.1 Lightning Network

In a typical cryptocurrecy transaction, the finalization has to wait for the addition of at least one block into the corresponding blockchain. This introduces delay; in addition, writing a transaction onto a blockchain is often quite expensive, both in monetary terms and in terms of energy and resource consumption. The Lightning network is an attempt to circumvent this problem. It is an example of what is called a Layer 2 system – where the bulk of the recording and computation required for transactions happens off-chain.

The following description are from https://lightning.network/.

> The Lightning Network is dependent upon the underlying technology of the blockchain. By using real Bitcoin/blockchain transactions and using its native smart-contract scripting language, it is possible to create a secure network of participants which are able to transact at high volume and high speed.
>
> **Bidirectional Payment Channels.** Two participants create a ledger entry on the blockchain which requires both participants to sign off on any spending of funds. Both parties create transactions which refund the ledger entry to their individual allocation, but do not broadcast them to the blockchain. They can update their individual allocations for the ledger entry by creating many transactions spending from the current ledger entry output. Only the most recent version is valid, which is enforced by blockchain-parsable smart-contract scripting. This entry can be closed out at any time by either party without any trust or custodianship by broadcasting the most recent version to the blockchain.
>
> **Lightning Network.** By creating a network of these two-party ledger entries, it is possible to find a path across the network similar to routing packets on the internet. The nodes along the path are not trusted, as the payment is enforced using a script which enforces the atomicity (either the entire payment succeeds or fails) via decrementing time-locks.
>
> **Blockchain as Arbiter.** As a result, it is possible to conduct transactions off-blockchain without limitations. Transactions can be made off-chain with confidence of on-blockchain enforceability. This is similar to how one makes many legal contracts with others, but one does not go to court every time a contract is made. By making the transactions and scripts parsable, the smart-contract can be enforced on-blockchain. Only in the event of non-cooperation is the court involved – but with the blockchain, the result is deterministic. [1]

Interestingly, and somewhat surprisingly, once the bidirectional payment channels are established, the Lightning Network is just another credit network. Of course in the Lightning Network, nodes have to tie up some funds into a smart-contract to establish credit capacity along an edge. Thus, the credit-establishment mechanism is very different from our original model of a credit network (and hence, the underlying strategic network formation problem is quite rich), though the liquidity analysis remains the same (making the network formation problem tractable to study in some simple scenarios).

The paper "The evolving topology of the Lightning Network: Centralization, efficiency, robustness, synchronization, and anonymity", Martinazzi and Flori [2], has shown an empirical analysis on the Bitcoin Lightning Network during its first year. In summary, the Lightning network involves a tension, where efficiency is positively affected by all nodes having a relationship with a central node, but robustness, decentralization, and anonymity are negatively affected by this relationship.

As we can see that the interaction with others is important for understanding the model behavior, it is quite obvious that other agents' decision should affect our utility as well as our decision.

For simplicity, we limit our scope of interest into the case of static setting, where every agent decides at the same time and has to commit to his own decision. For future exploration, it is useful to also consider the case of a new agent joining the network later, seeing the existing structure of the network, or when an agent is allowed to change his trust allocation at a later time. For this static case of interest, one useful tool will be the game theoretic notion of Nash equilibrium.

3.2 Nash Equilibrium

Assume that there are N agents. Each of them has the option to determine how much they trust each of the other agents. Let a_i denote an agent i, and let S_i denote that agent's strategy space, or the space of all feasible trust vectors for a_i. Each agent has a cost for each for each of their trust vectors/strategies.

Assume a_i has S_i. Consider a network topology G with initial credit values, which are dependent on S_v for all agents a_v in the system. The agent a_i receives some utility $U_i(G)$ for this. Furthermore, we can add a cost of actions for a_i to choose to do S_i as $C_i(S_i)$.

A **Nash Equilibrium** is a strategy for each agent such that no agent will have the incentive to unilaterally deviate from their strategy.

A pure strategy is a strategy playing only one action. This type of strategy is usually specified by an action.

A mixed strategy is a randomized strategy when the corresponding agent will choose from a set of actions with some non-deterministic probabilistic distribution. This type of strategy is usually specified by a probabilistic distribution among each action.

Note that in addition to individual utilities, there can also be some overarching system objective; we will call this the "welfare function", denoted $W(G)$. Often, $W(G)$ may just be the sum of individual utilities. We would like the Nash Equilibrium to correspond to high welfare. In the case of mixed strategy Nash equilibrium, the final state of the world, G, may be randomized, making the individual utilities as well as the welfare random variables. In such case, we can consider the expected utilities and the expected welfare instead.

The next lecture will analyze the Nash equilibrium of a credit network formation game under simple assumptions.

References

[1] Lightning Network. How it works.

[2] Stefano Martinazzi and Andrea Flori. The evolving topology of the lightning network: Centralization, efficiency, robustness, synchronization, and anonymity. *PLOS ONE*, 15(1):1–18, 01 2020.