Computing Tarski Fixed Points in Financial Networks

STACS 2026

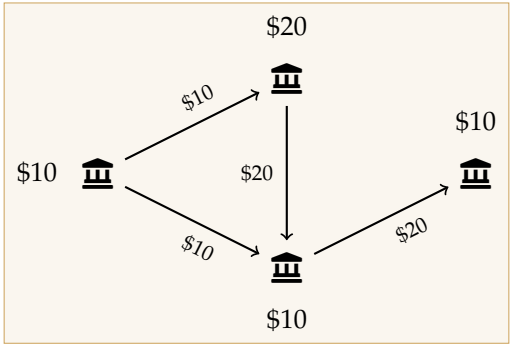
Leander Besting, Martin Hoefer, Lars Huth

Department of Computer Science, RWTH Aachen University

March 12, 2026

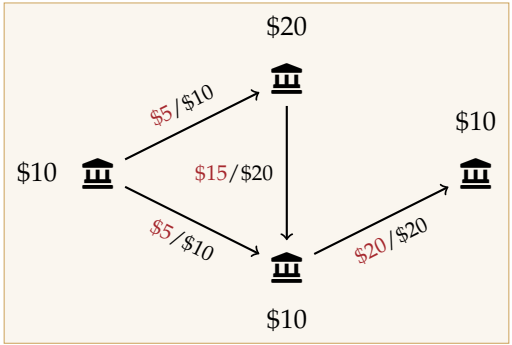
Financial Networks

Financial Networks



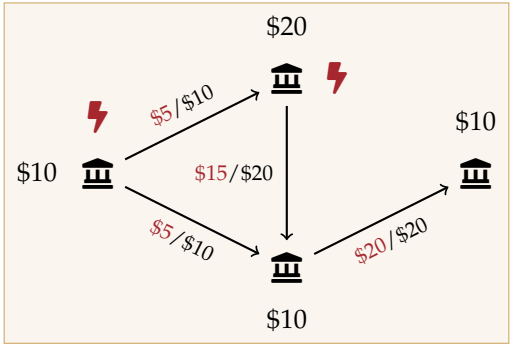
- Banks with Debts and Assets

Financial Networks



- Banks with Debts and Assets
- Flow of Payments

Financial Networks



- Banks with Debts and Assets
- Flow of Payments
- Some Banks are in Default

Clearing States

- Banks V, Debts E
- External Assets a_v^x, Amounts of Debt ℓ_e
- Payment Strategies $p_e(x)$: How much the debtor pays on e if he has x money

Definition

A clearing state is a consistent assignment a_v of gross assets for each bank:

$$a_v = a_v^x + \sum_{(u,v)\in E} p_{(u,v)}(a_u)$$

Payment Strategies

- Valid Amount:
$$p_e(x) \in [0, \ell_e]$$
- Monotonic:
$$x \leq y \Longrightarrow p_e(x) \leq p_e(y)$$
- No Fraud and Limited Liability:
$$\sum_{(u,v)\in E} p_e(x) = \min\{x, \sum_{(u,v)\in E} \ell_{(u,v)}\}$$

Payment Strategies – Proportional

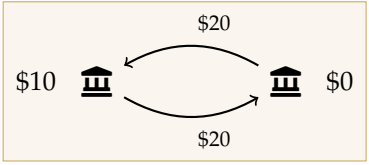
Definition

A bank's assets are divided among its liabilities proportional to their amount of debt

$$p_{(u,v)}(x) = x \cdot \frac{\ell_{(u,v)}}{\sum_{(u,w)\in E} \ell_{(u,w)}}$$

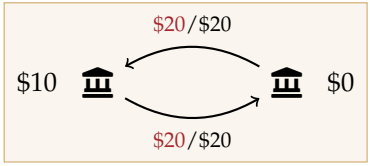
- Used in the original version of the model

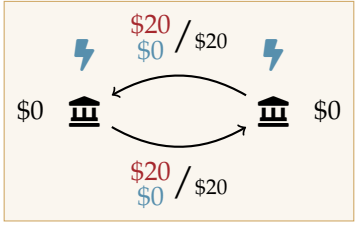
Cycles



- What about cycles?

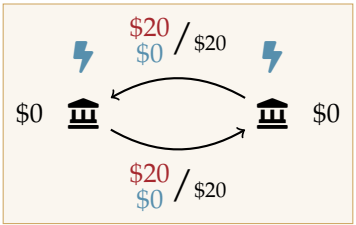
Cycles





- What about cycles?
- Clearing states are fixed points of a monotone operator which locally adjusts payments
- These always exist and form a lattice via the Knaster-Tarski theorem
- Ambiguity: least vs. greatest clearing states

Clearing States – Uniqueness



Lemma

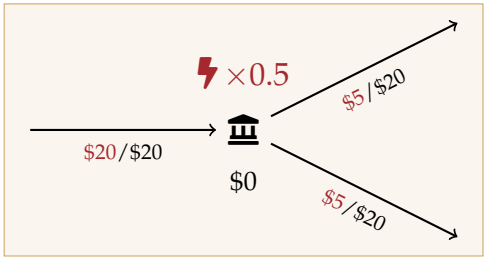
In this model, if the least and greatest Clearing State differ somewhere, the least Clearing State's payments are zero.

The Model

- Proposed by Eisenberg & Noe in 2001
- Used by the ECB for analyzing the Banking Sector

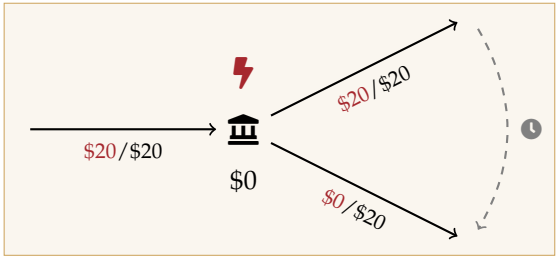
Model Extensions

Extensions – Default Rates



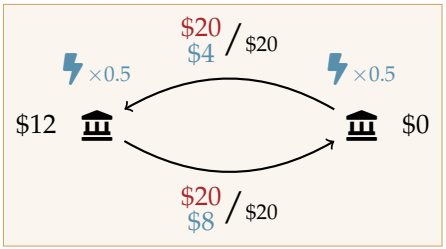
- Proposed by Rogers and Veraart in 2011
- Assumes that a defaulting bank calling in its loan will only realize some fraction immediately
- Modeled by multiplying each defaulting banks assets with some factor

Extensions – Priorities



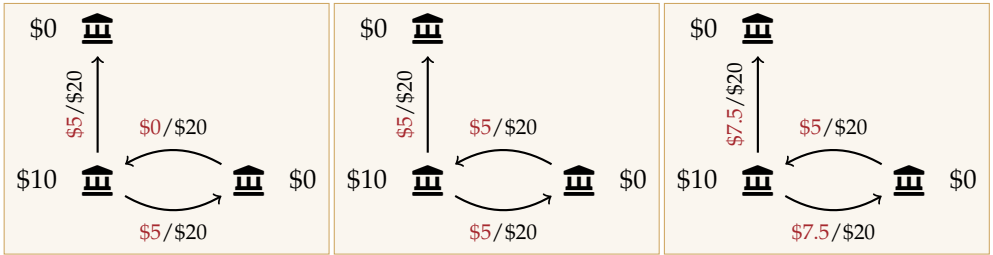
- Models the different maturity dates of debts
- Debt with higher priority is paid of fully before debt with lower priority starts to get paid

Least vs. Greatest Clearing States



- With these extensions the least clearing state can differ a lot from the greatest one

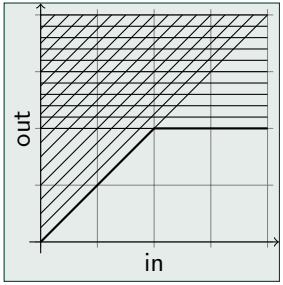
Decentralized Clearing

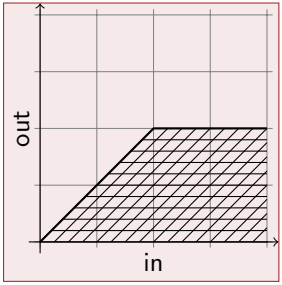


- The least clearing state represents the outcome of a decentralized clearing process

Computing Clearing States

- Greatest clearing state can be found with a LP

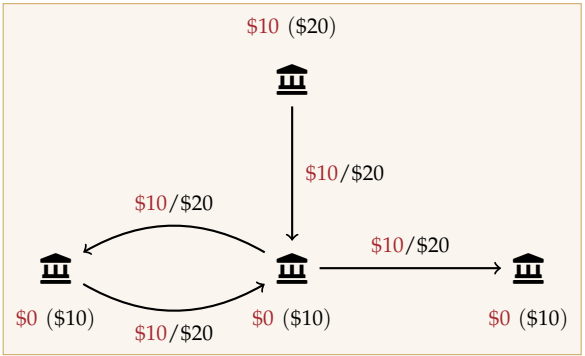




- This does not work for the least clearing state

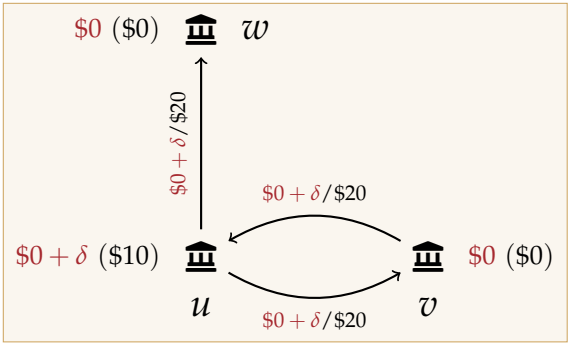
Our Algorithm

Reduced Assets



- Each step of the algorithm gives the banks a reduced amount of external assets b_v^x
- We track the clearing state w.r.t. these reduced assets
- Each step increases these reduced assets until they match the actual assets

Linear Increase

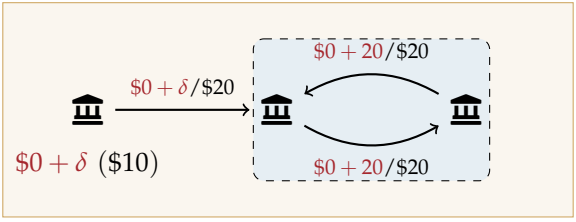


This increase works for $\delta \leq 10$

- Perform a small increase of reduced assets that doesn't cross any nonlinearities

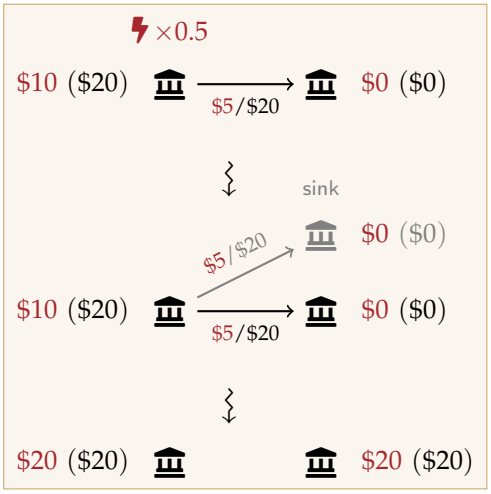
$$
\begin{aligned}
\text{max.} \quad & \delta \\
\text{s.t.} \quad & d_u = \delta + d_v \\
& d_i = \tfrac{d_u}{2} && i \in \{v, w\} \\
& d_u \leq 40 \\
& d_i \leq 20 && i \in \{v, w\} \\
& \delta \leq 10 \\
& d_i \geq 0 && i \in V
\end{aligned}
$$

Flooding



- Happens when payments enter a strongly connected component with no outgoing edges
- Payments keep flowing until some edges are saturated
- Leads to non-linear (constant) increase of clearing state

Default Costs



- Preprocessing: Replace default costs with sink banks
- On Solvency: Transform network by deleting out-edges of solvent bank and replacing payments with assets

Our Clearing Algorithm

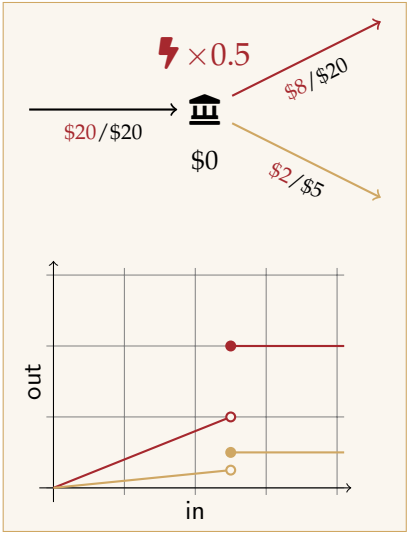
1. Choose bank v with reduced assets
2. Compute reachable SCCs from v
3. Use LP for flow increase in flooded SCCs
4. Use LP for linear effects of increasing v's assets
5. Deal with new solvencies
6. Repeat

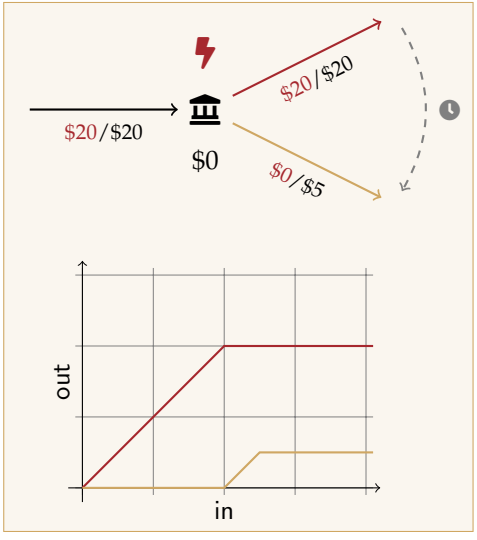
Runtime

- The algorithm has runtime $\mathcal{O}((n+k) \cdot (n+m)^3)$ (where k is the total number of priority levels)

Further Results

Piecewise Linear Payment Functions





Further Results

- This clearing algorithm works for all piecewise linear payment strategies whose pieces are $[a, b)$ intervals
- We can compute not just the least but any clearing state
- We adapted some results about claims trading from the greatest clearing state to the least clearing state (Claims trading: Preprocessing of the network to improve the clearing state)

Thank you for your attention!