Debt Reduction by Netting in B2B Networks

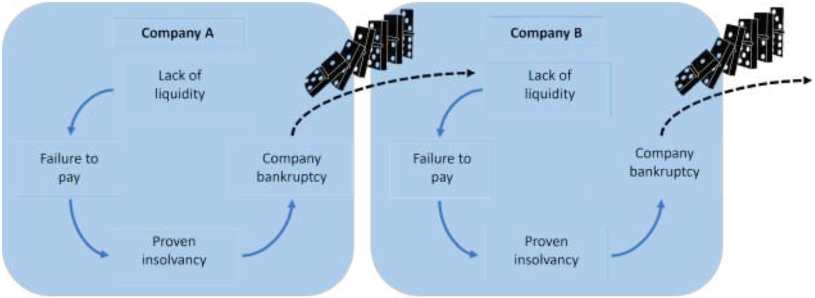
Sylvain Contassot-Vivier, Nazim Fatès, Joannès Guichon

Université de Lorraine, Inria, LORIA, F-54000 Nancy, France

June 13, 2024

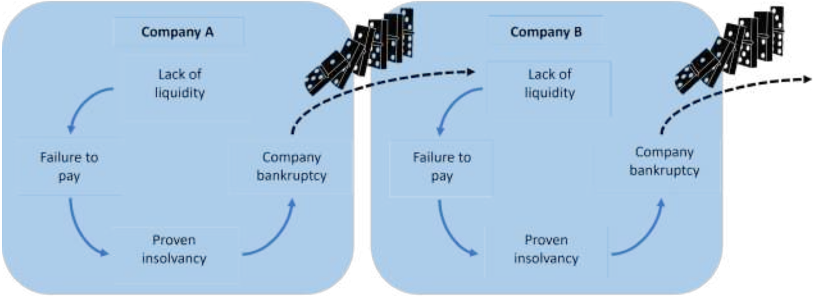
Introduction : Economic fabric (E.F.)

- Companies settle invoices with delays → **intertwined** debts
- Excessive accumulation of debt can cause financial instability



Introduction : Economic fabric (E.F.)

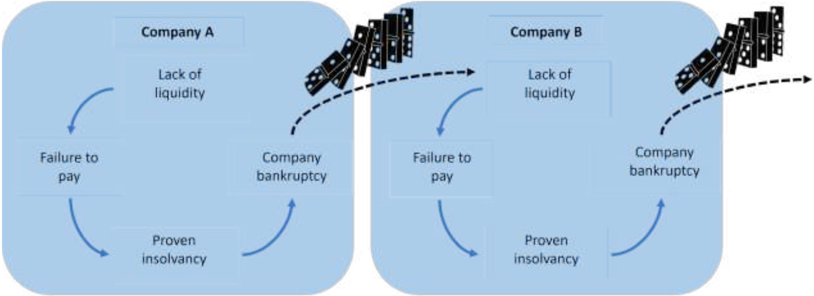
- Companies settle invoices with delays → **intertwined** debts
- Excessive accumulation of debt can cause financial instability



⇒ How can we lower the systemic risk of such a network ?

Introduction : Economic fabric (E.F.)

- Companies settle invoices with delays → **intertwined** debts
- Excessive accumulation of debt can cause financial instability

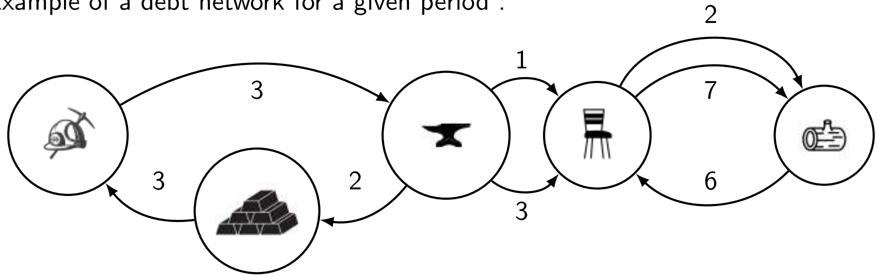


⇒ How can we lower the systemic risk of such a network ?

Debt reduction by graph netting

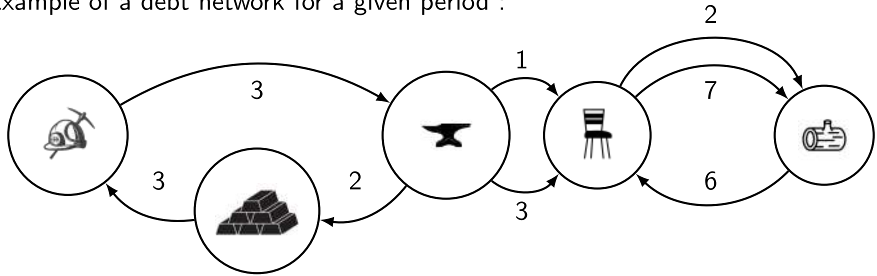
Debt Network Formal Definition

Example of a debt network for a given period :



Debt Network Formal Definition

Example of a debt network for a given period :



Formalisation :

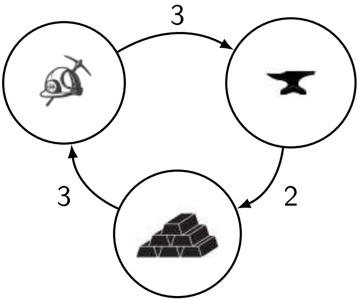
- Debt network as a weighted multi-directed graph $G = (V, E)$
- Edge e_i corresponds to a currency flow w_i
  - from source s_i to destination d_i at date T_i

$\rightarrow$ $e_i = (s_i, d_i, w_i, T_i)$ for each edge

- Debt networks are constructed using 27 million invoices from *Infocert*
- Reduction process on an extracted graph for a given period

Two main types of reduction by netting :

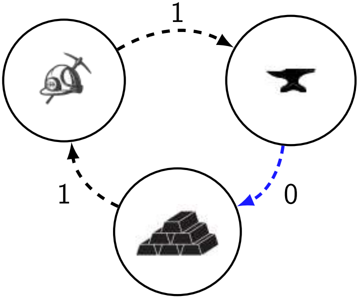
- **Partial** reduction :
  - Possibility to partially reduce an invoice
  - Allows the maximal reduction of debts



B2B Debt Reduction methods

Two main types of reduction by netting :

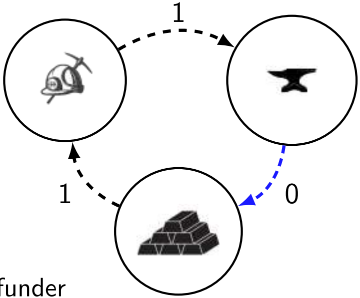
- **Partial** reduction :
  - Possibility to partially reduce an invoice
  - Allows the maximal reduction of debts



B2B Debt Reduction methods

Two main types of reduction by netting :

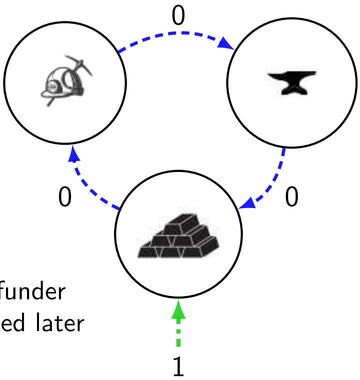
- **Partial** reduction :
  - Possibility to partially reduce an invoice
  - Allows the maximal reduction of debts
- **Integral** reduction :
  - Only full settlement of invoices
  - NP-complete problem with an external funder
  - Funder creates new debt to be reimbursed later



B2B Debt Reduction methods

Two main types of reduction by netting :

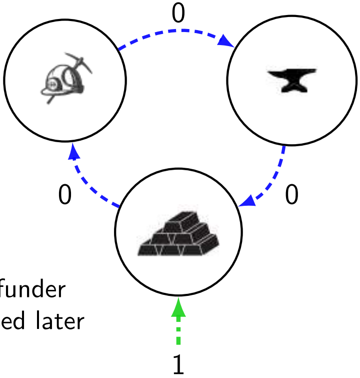
- **Partial** reduction :
  - Possibility to partially reduce an invoice
  - Allows the maximal reduction of debts
- **Integral** reduction :
  - Only full settlement of invoices
  - NP-complete problem with an external funder
  - Funder creates new debt to be reimbursed later



B2B Debt Reduction methods

Two main types of reduction by netting :

- **Partial** reduction :
  - Possibility to partially reduce an invoice
  - Allows the maximal reduction of debts
- **Integral** reduction :
  - Only full settlement of invoices
  - NP-complete problem with an external funder
  - Funder creates new debt to be reimbursed later



We focus on integral reduction because :

- Focus on removing invoices instead of reducing the debt amount
- Greater interest from administrative perspective

Evaluation of Reductions

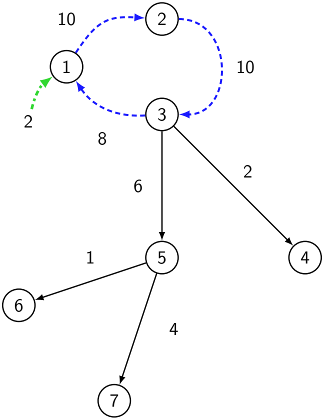
Evaluation of the efficiency of reduction :

For a sub-graph S of G, composed of the debts that will be reduced :

- We define the **amplification factor** :

$$\alpha(S) = \frac{D_S}{F_S}$$

with D_S the total reduced debt in S and F_S the financing needed



Debt cleared: 28, Financing: 2, α= 14

Evaluation of Reductions

Evaluation of the impact on the global debt network :

- In previous work, the **settlement inclusion** factor $I(S)$ was used:

$$I(S) = \frac{D_S}{D_G}$$

  It represents the amount of debt that we reduced compared to the total amount of debt present

- We define the **gain measure** :

$$g(S) = \frac{D_S - F_S}{D_G - F_G}$$

  representing the ratio of effective debt reduced by netting in S compared to the maximum reducible by netting in G

Reduction Algorithm

Reduction algorithm for a graph :

- Step 1: Removal of perfect cycles of length 2

Reduction Algorithm

Reduction algorithm for a graph :

- Step 1: Removal of perfect cycles of length 2
- Step 2: Research of germs, paths with high amplification, using depth-first search algorithm

Reduction Algorithm

Reduction algorithm for a graph :

- Step 1: Removal of perfect cycles of length 2
- Step 2: Research of germs, paths with high amplification, using depth-first search algorithm
- Step 3: Extension of the germs by subsequently selected the edge with the highest potential : Capacity to reduce other edges without the need for more financing.
  We add the said best edge and the ones that it can reduce by potential.

Comparison between using alpha and potential for reduction

Computation of initial germs : paths with a high amplification

Previous reduction method :

- select edges for the reduction one by one by growing the germs
- consider the ratio of debt cleared over investment only
  - $\rightarrow$ Notion of amplification

Comparison between using alpha and potential for reduction

Computation of initial germs : paths with a high amplification

Previous reduction method :

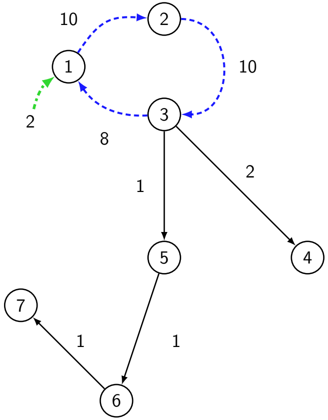
- select edges for the reduction one by one by growing the germs
- consider the ratio of debt cleared over investment only
  - $\rightarrow$ Notion of amplification

New technique :

- select groups of edges that are reduced together (still by growing germs)
- consider the amount cleared when the root edge is selected
  - $\rightarrow$ Notion of potential

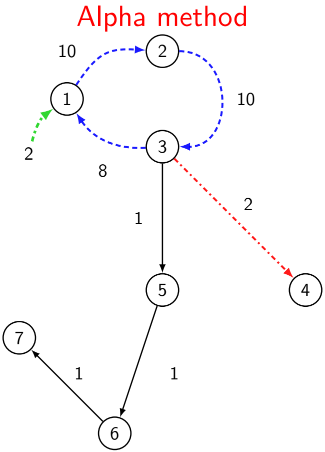
Comparison between using alpha and potential for reduction

Base graph with germ

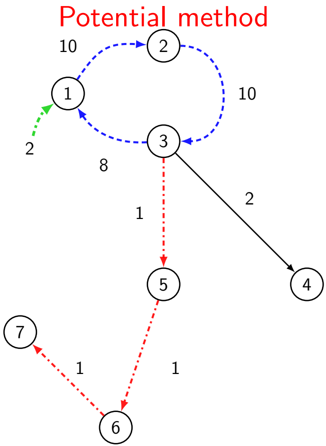


Debt cleared: 28, Financing: 2, Global alpha: 14

Comparison between using alpha and potential for reduction



Debt cleared: 30, Financing: 2, Global alpha: 15, Gain: 0.84



Debt cleared: 31, Financing: 2, Global alpha: 15.5, Gain: 1 !

Time consideration necessity

Results of our reduction algorithm are interesting but :

- Still lacking the time component
- This is essential considering :
  - The dynamic aspect of invoices and liquidity management
  - The possibility to use the funder as a liquidity buffer

Time consideration necessity

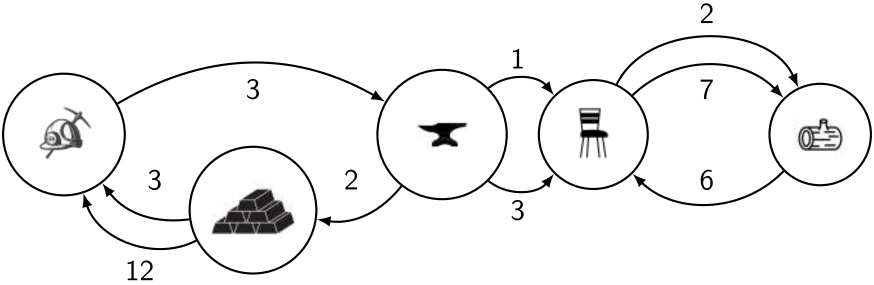
Results of our reduction algorithm are interesting but :

- Still lacking the time component
- This is essential considering :
  - The dynamic aspect of invoices and liquidity management
  - The possibility to use the funder as a liquidity buffer

Our proposition is to apply our reduction algorithm using :

- A sliding timeframe $\mathcal{T}$
- Invoices leaving the system after $\mathcal{D} = 28$ days

$t = 0$, before reduction :



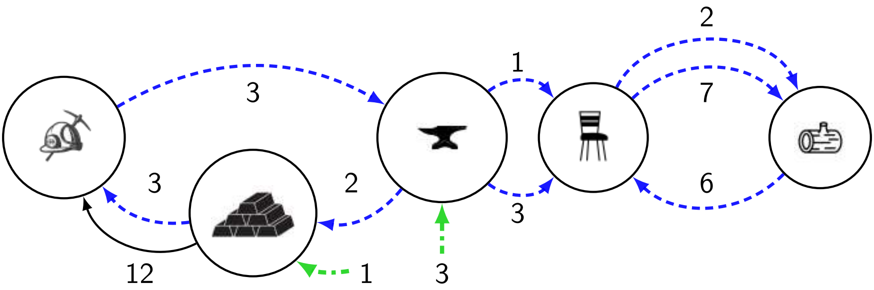
Debt reduced : 0, financing : 0, $\alpha = 0$

Total debt : 39, total financing : 16, $\alpha_{global} = 2.43$

Gain : 0, inclusion : 0

Time-based Algorithm

$t = 0$, during reduction :

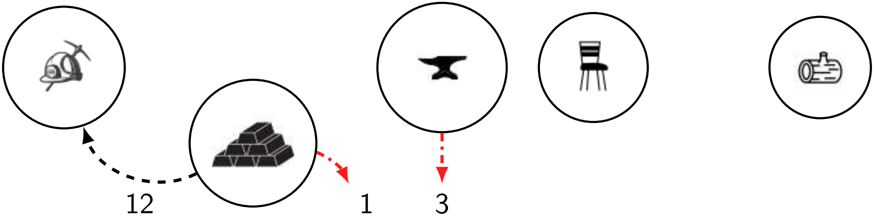


Debt reduced : 27, financing : 4, $\alpha = 6.75$
Total debt : 39, total financing : 16, $\alpha_{global} = 2.43$

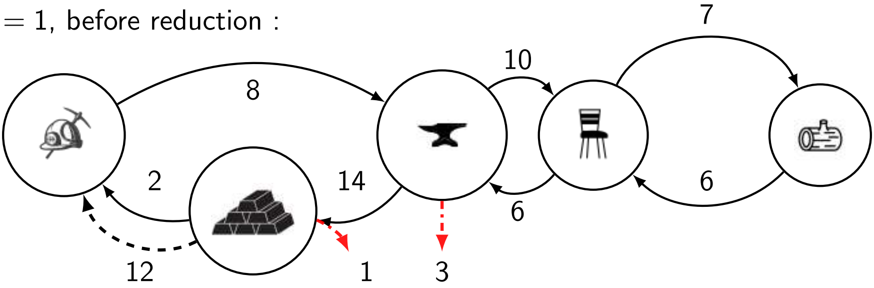
Gain : $\frac{27-4}{39-16} = 1$, inclusion : $\frac{27}{39} = 0.69$

Time-based Algorithm

$t = 0$, after reduction :

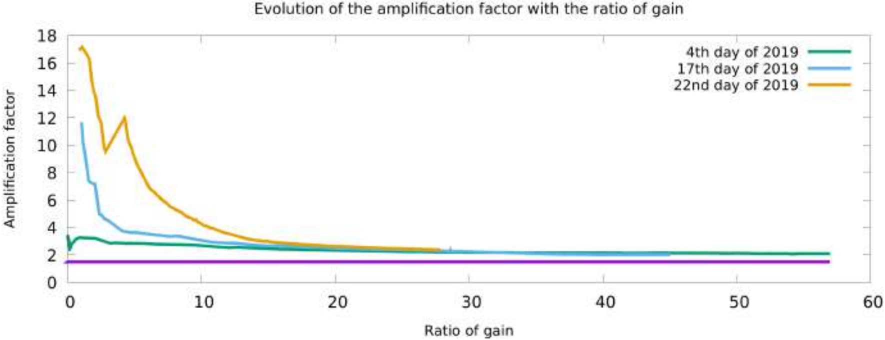


$t = 1$, before reduction :



Experimental Results using daily processing

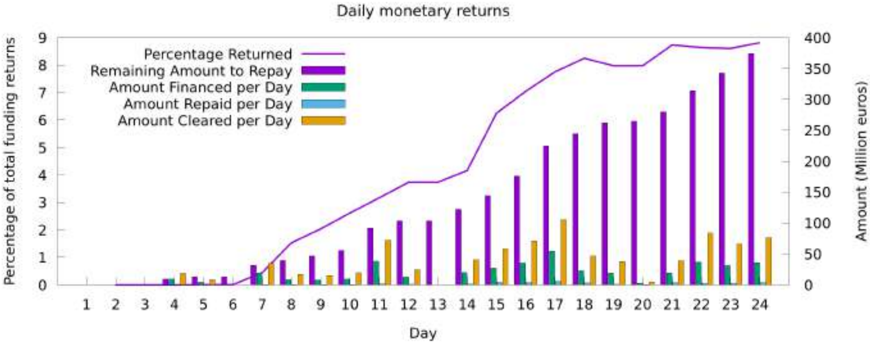
- Depending on the period, the algorithm might stop early to maintain an amplification factor above the user-defined threshold (here 1.5)
- Lowering amplification expectations could clear more debt at the cost of efficiency



Results on returns to the funder

- On the first 24 days of the year :
  - 9% of investment is recovered through the reduction process
  - The remaining is recovered through classical means (max delays)

  → Returns are expected to increase over the course of time



Conclusion and directions

Key Findings

- **Reduction Techniques:** Implementation of successful debt reduction strategies, including time-based one.
- **Algorithm Performance:** Promising results in reducing debts by systematically targeting high-impact transactions.

Conclusion and directions

Key Findings

- **Reduction Techniques:** Implementation of successful debt reduction strategies, including time-based one.
- **Algorithm Performance:** Promising results in reducing debts by systematically targeting high-impact transactions.

Implications

- **Financial Stability:** Reducing intertwined debts and potential bankruptcies.
- **Risk management:** Returns assured by the algorithm decrease the funder's risk.

Conclusion and directions

Future Work

- **Algorithm Refinement:** Enhancement of the performance and scalability of algorithms to handle larger datasets (Python to C++).
- **Long term studies:** Analyze the results for large spans of time in terms of reduction and returns.
- **Reading:** Read and research more into chain failures, economic possibilities. Need to dive more into literature in general.
- **Risk and stability of the system:** measure the robustness of our system to random and characterized attacks.
- **Integration with Financial Tools:** Explore integration possibilities with existing financial tools ?

Questions ?

Thanks for listening,
if you have any question feel free to ask.