Collusion-Resistant Worker Set Selection for Transparent and Verifiable Blockchain-Based Voting

Matthieu BETTINGER
Lucas BARBERO
Omar Hasan

Context

- **Schiedermeier's blockchain-based voting protocol (2020)**
  - P participants, and W workers chosen among them
  - Workers compute the referendum's result
    - Using SMPC:
      - Shamir's Secret Sharing scheme
      - Homomorphic encryption
  - All messages are recorded on-chain

Context

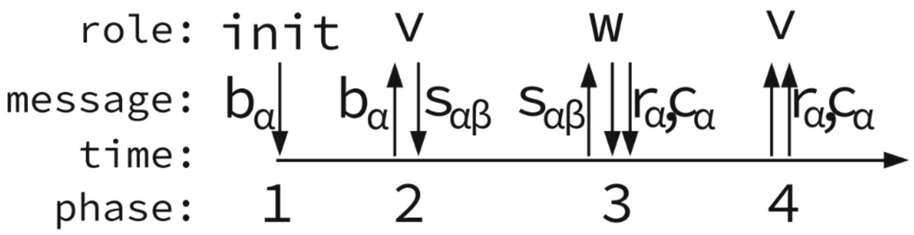
- However: workers are chosen by a single trusted entity

  → Risk of collusion

- Goal:

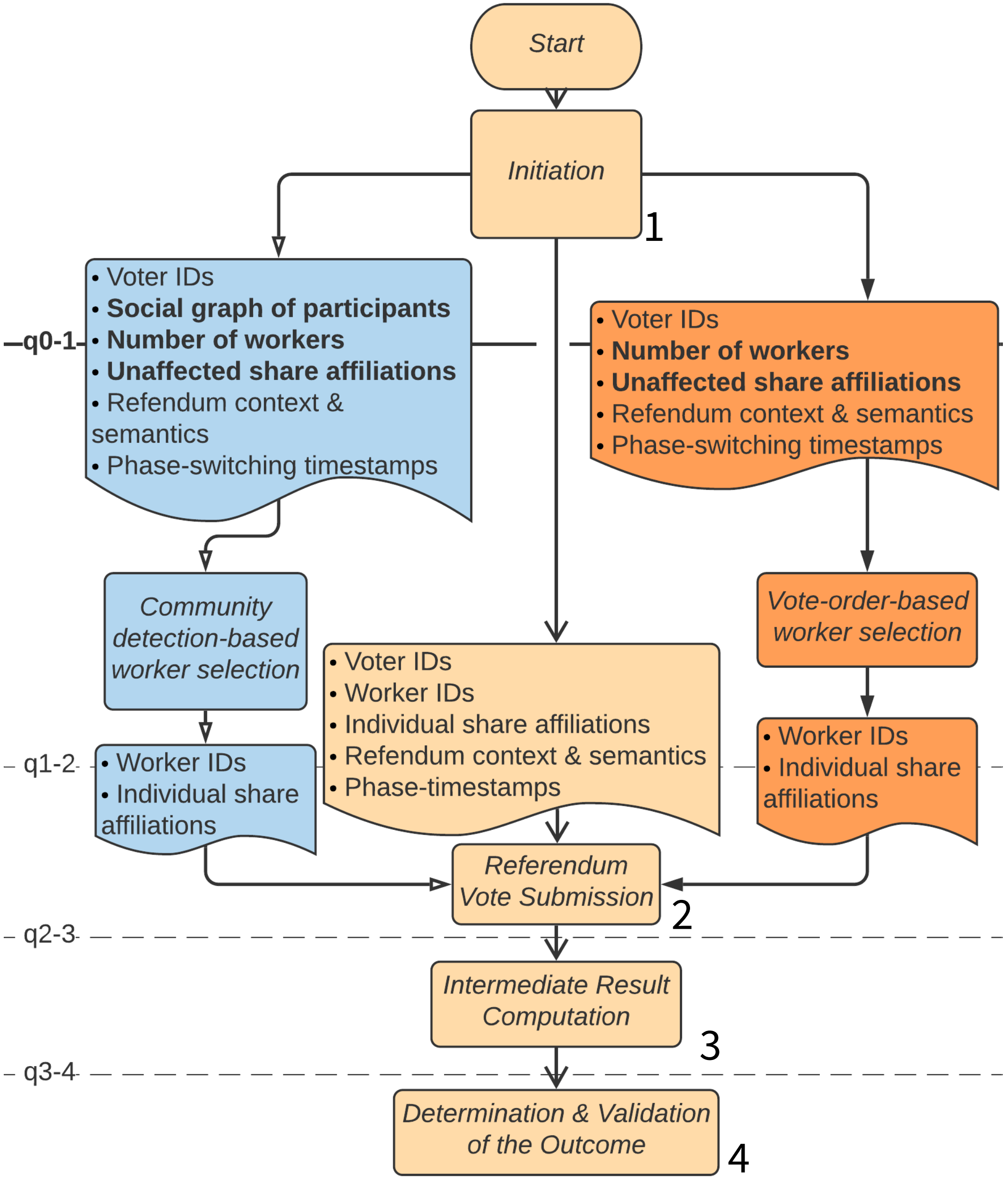
  → Collusion-resistant & verifiable selection of W workers

Process


role: init V W V
message: b_α b_α $s_{\alpha\beta}$ $s_{\alpha\beta}$ r_α,c_α r_α,c_α
time:
phase: 1 2 3 4

Schiedermeier et al. (2019)

Process integration diagram of proposed solutions


Start

Initiation 1

–q0-1–

- Voter IDs
- **Social graph of participants**
- **Number of workers**
- **Unaffected share affiliations**
- Refendum context & semantics
- Phase-switching timestamps

- Voter IDs
- **Number of workers**
- **Unaffected share affiliations**
- Refendum context & semantics
- Phase-switching timestamps

Community detection-based worker selection

Vote-order-based worker selection

– q1-2–

- Worker IDs
- Individual share affiliations

- Voter IDs
- Worker IDs
- Individual share affiliations
- Refendum context & semantics
- Phase-timestamps

- Worker IDs
- Individual share affiliations

Referendum Vote Submission 2

– q2-3–

Intermediate Result Computation 3

– q3-4–

Determination & Validation of the Outcome 4

Bettinger et al. (2021)

Proposal 1:
Verifiable random number generation

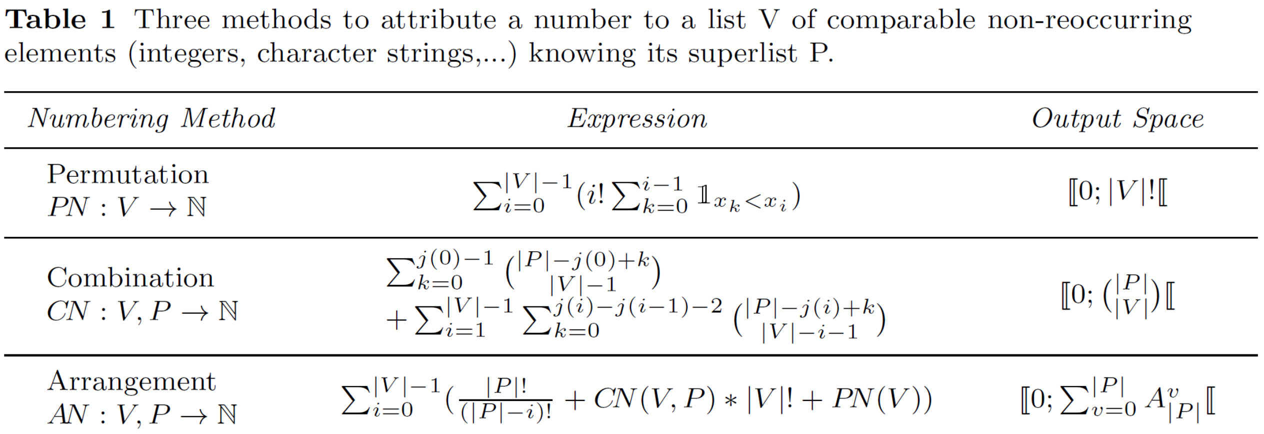


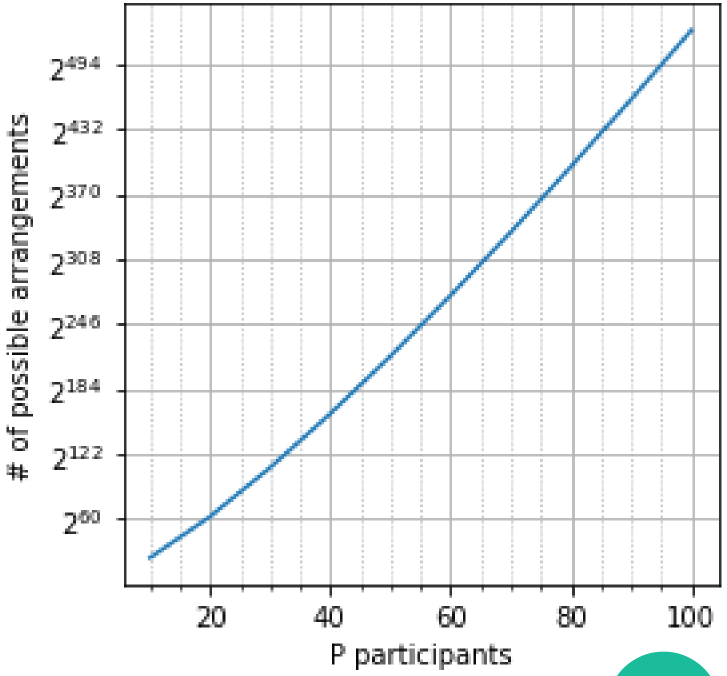
Table 1 Three methods to attribute a number to a list V of comparable non-reoccurring elements (integers, character strings,...) knowing its superlist P.

Numbering Method	*Expression*	*Output Space*
Permutation $PN : V \to \mathbb{N}$	$\sum_{i=0}^{\lvert V\rvert-1}(i!\sum_{k=0}^{i-1} \mathbb{1}_{x_k<x_i})$	$[\![0; \lvert V\rvert ! [\![$
Combination $CN : V, P \to \mathbb{N}$	$\sum_{k=0}^{j(0)-1} \binom{\lvert P\rvert-j(0)+k}{\lvert V\rvert-1}$ $+\sum_{i=1}^{\lvert V\rvert-1} \sum_{k=0}^{j(i)-j(i-1)-2} \binom{\lvert P\rvert-j(i)+k}{\lvert V\rvert-i-1}$	$[\![0; \binom{\lvert P\rvert}{\lvert V\rvert} [\![$
Arrangement $AN : V, P \to \mathbb{N}$	$\sum_{i=0}^{\lvert V\rvert-1}(\frac{\lvert P\rvert!}{(\lvert P\rvert-i)!} + CN(V,P) * \lvert V\rvert! + PN(V))$	$[\![0; \sum_{v=0}^{\lvert P\rvert} A_{\lvert P\rvert}^{v} [\![$

- **PN : Ordering of *V* voters**
- **CN : Which subset of *V* participants voted**
- **AN : PN & CN + Sum(AN with fewer voters)**

10 participants : 10^6 possibilities

100 participants : 10^{158} possibilities

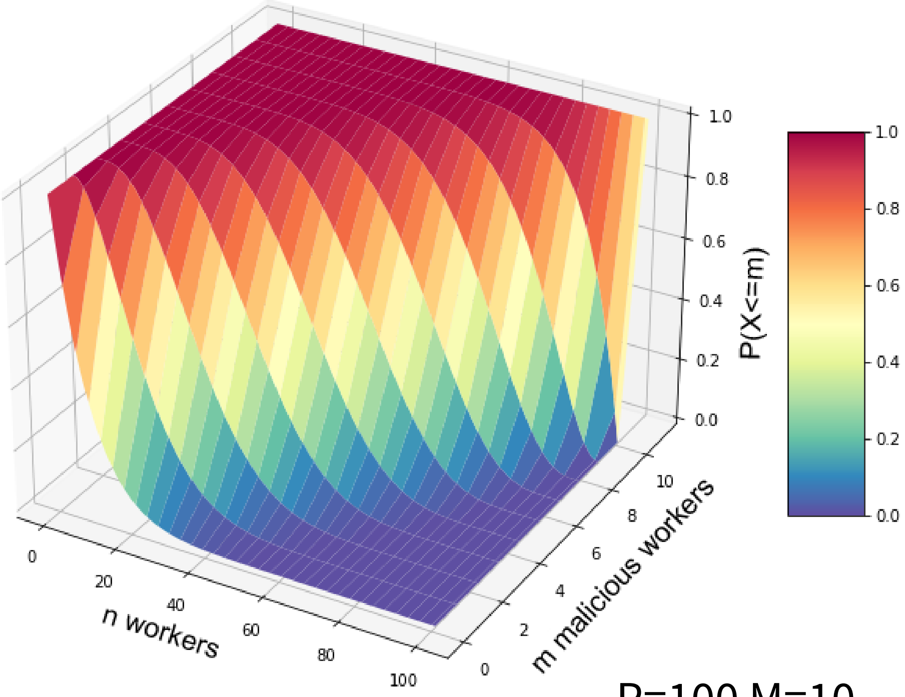


Proposal 1:
Probability Distribution

- Hypergeometric Law

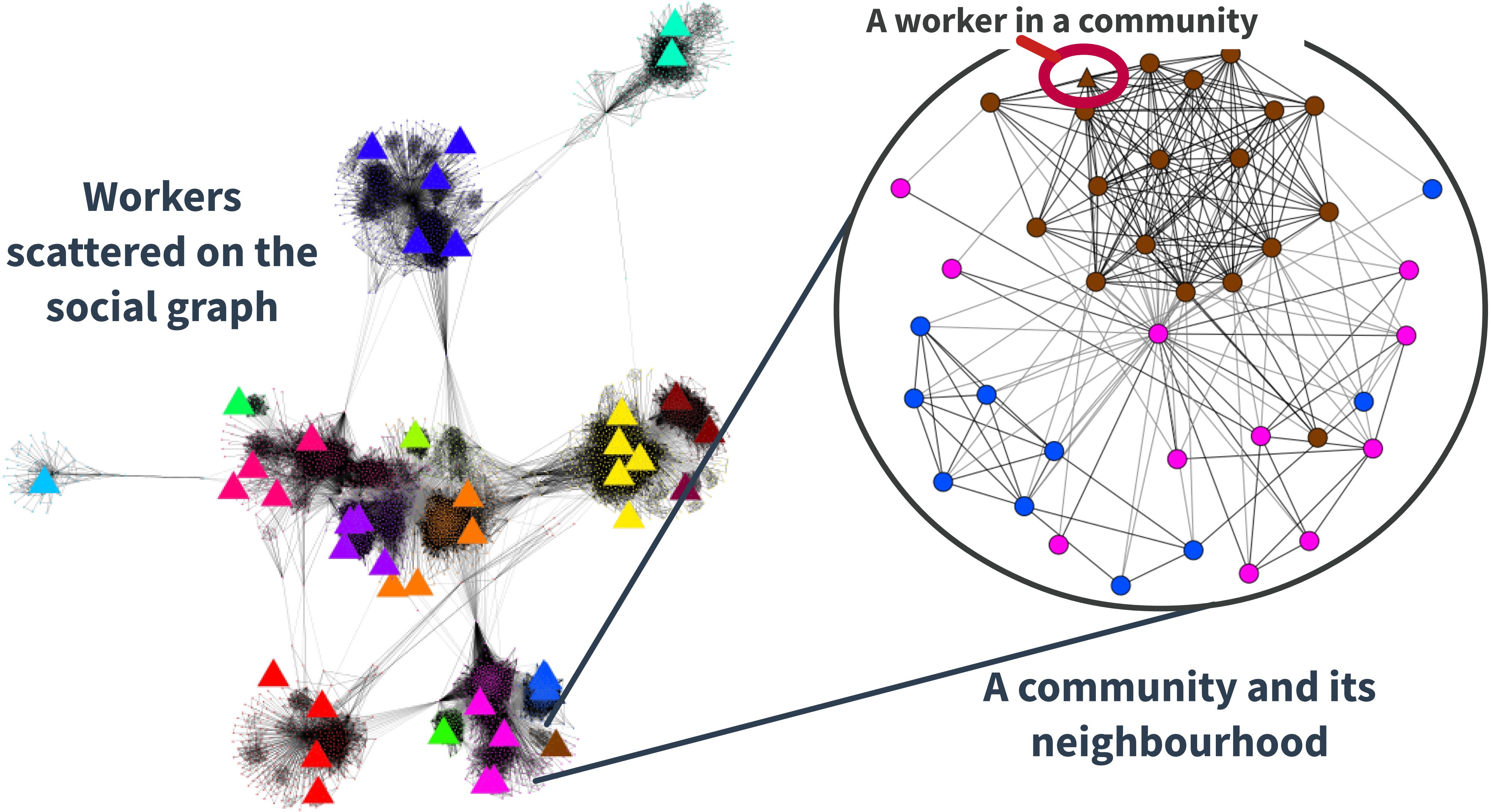
  H(n, M/P, P)

  - P participants
  - M ≤ P malicious participants
  - n workers
  - m malicious workers



P=100 M=10

Proposal 2: Worker distancing

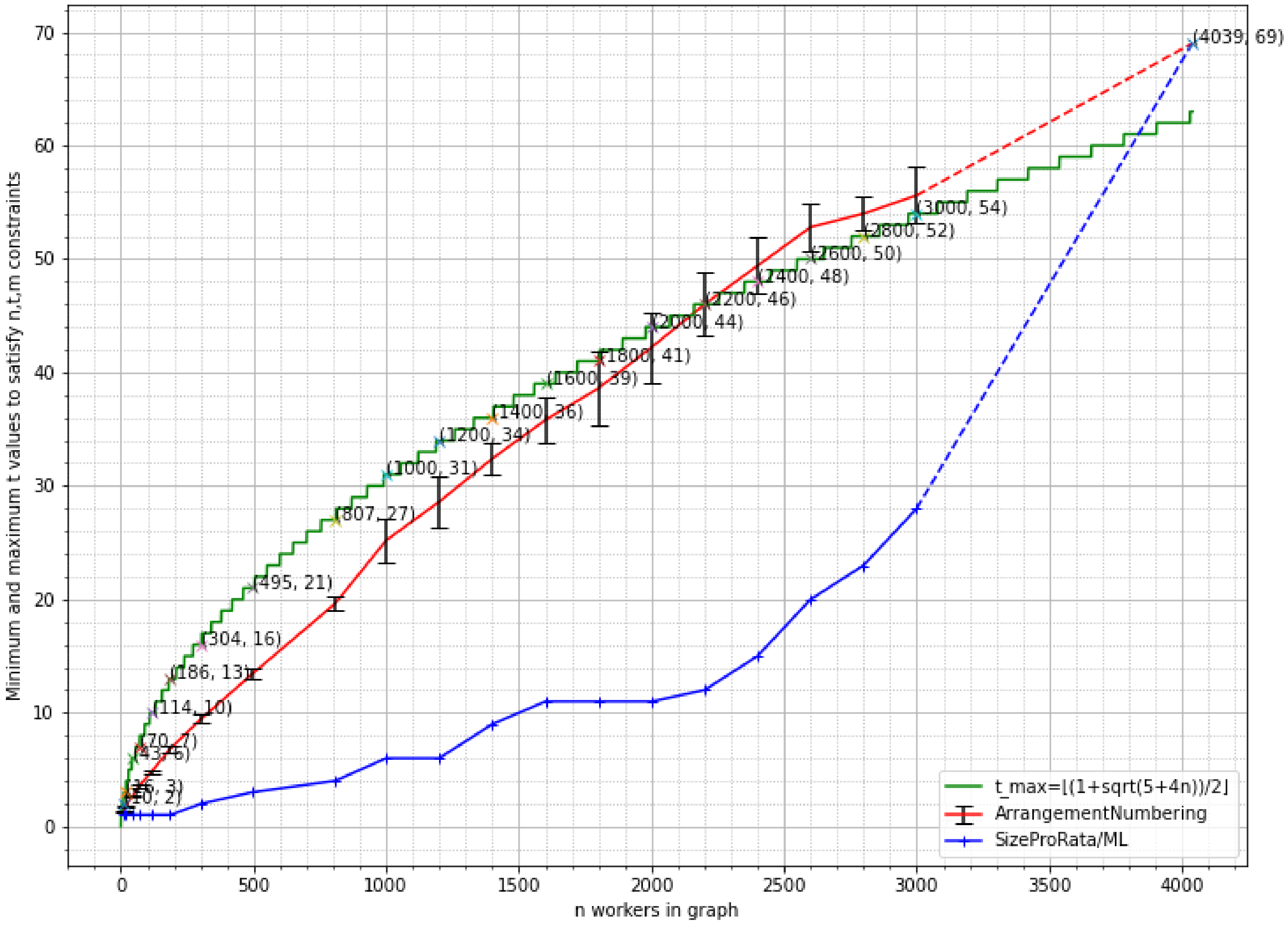


Workers scattered on the social graph

A worker in a community

A community and its neighbourhood

Results:
Cliques of adjacent workers



Application to iExec's worker selection

- Scheduler: trusted to distribute work fairly

  → Proposed solutions can be used

- Difference with Schiedermeier's:
  - iExec workers perform multiple tasks
  - versus anonymized participants for each vote

Application to iExec’s worker selection

- Aral et al. (2020)
  - Use of workers’ task execution history
  - Clustering of workers that fail together

  → **Algorithm to maximize success probability**