

Future of ~~Blockchain~~ DLT and Reputation Consensus

Anton Kolonin

akolonin@aigents.com

Facebook: [akolonin](#)

Telegram: [akolonin](#)

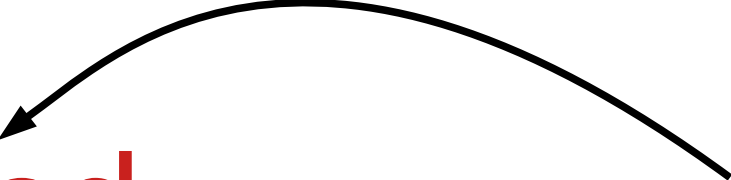


~~Blockchain~~ DLT Fundamental Issue

Speed + Cheapness + Security + Decentralization = CONST

~~Blockchain~~ DLT Fundamental Issue

Optimistic rollup



Speed + Cheapness + Security + Decentralization = CONST

~~Blockchain~~ DLT Fundamental Issue

Zero-Knowledge Succinct Non-Interactive Argument of Knowledge
(ZK-SNARK)


$$\text{Speed} + \text{Cheapness} + \text{Security} + \text{Decentralization} = \text{CONST}$$

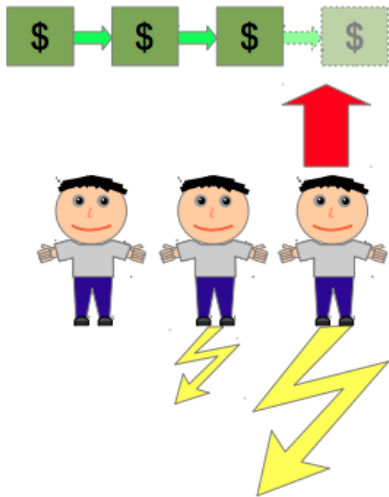
~~Blockchain~~ DLT Fundamental Issue

Side-chains (Level 2)


$$\text{Speed} + \text{Cheapness} + \text{Security} + \text{Decentralization} = \text{CONST}$$

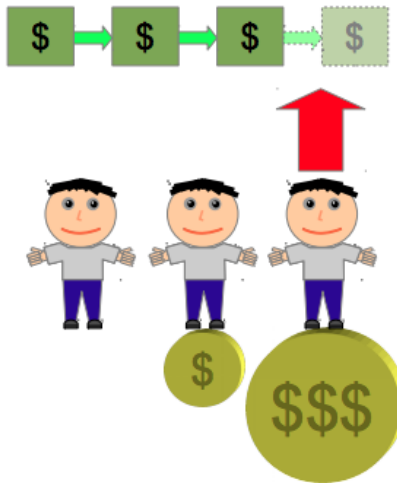
Reputation Consensus as a Liquid Democracy (2017)

Proof-Of-Work



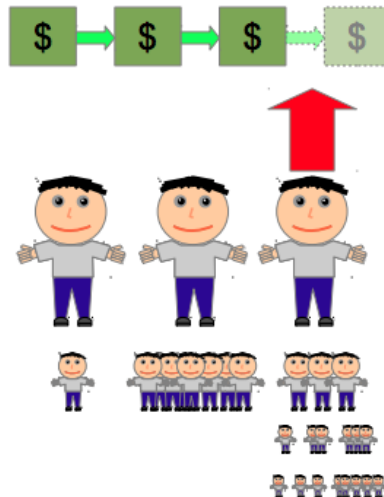
Those who own more computing power govern the network.

Proof-Of-Stake



Those who have more money govern the network.

Proof-Of-Reputation



Those who earn deeper reputation and greater long-term audience base govern the network.

$$R_i = \sum_t \sum_j (R_j * V_{ijt})$$

Reputation Engine

Algorithm 1 Weighted Liquid Rank (simplified version)

Inputs:

- 1) Volume of rated transactions each with financial value of the purchased product or service and rating value evaluating quality of the product/service, covering specified period of time;
- 2) Reputation ranks for every participant at the end of the previous time period.

Parameters: List of parameters, affecting computations - default value, logarithmic ratings, conservatism, decayed value, etc.

Outputs: Reputation ranks for every participant at the end of the previous time period.

```
1: foreach of transactions do
2:   let rater_value be rank of the rater at the end of
     previous period of default value
3:   let rating_value be rating supplied by
     transaction rater (consumer) to ratee (supplier)
4:   let rating_weight be financial value of the
     transaction of its logarithm, if logarithmic ratings
     parameter is set to true
5:   sum rater_value*rating_value*rating_weight for
     every ratee
6: end foreach
```

```
7: do normalization of the sum of the multiplications
   per ratee to range 0.0-1.0, get differential_ranks
8: do blending of the old_ranks known at the end of
   previous period with differential_ranks based on
   parameter of conservatism, so that new_ranks =
   (old_ranks*conservatism+N*(1-differential_ranks)),
   using decayed value if no rating are given to ratee
   during the period
9: do normalization of new_ranks to range 0.0-1.0
10:return new_ranks
```

- R_d - default initial reputation rank;
- R_c - decayed reputation in range to be approached by inactive agents eventually;
- C - conservatism as a blending “alpha” factor between the previous reputation rank recorded at the beginning of the observed period and the differential one obtained during the observation period;
- *FullNorm* – when this boolean option is set to *True* the reputation system performs a full-scale normalization of incremental ratings;
- *LogRatings* - when this boolean option is set to *True* the reputation system applies $\log_{10}(1+value)$ to financial values used for weighting explicit ratings;
- *Aggregation* - when this boolean option is set to *True* the reputation system aggregates all explicit ratings between each unique combination of two agents with computes a weighted average of ratings across the observation period;
- *Downrating* - when this boolean option is set to *True* the reputation system translates original explicit rating values in range 0.0-0.25 to negative values in range -1.0 to 0.0 and original values in range 0.25-1.0 to the interval 0.0-1.0.
- *UpdatePeriod* – the number of days to update reputation state, considered as observation period for computing incremental reputations.

A Reputation System for Multi-Agent Marketplaces

Anton Kolonin, Ben Goertzel, Cassio Pennachin, Deborah Duong, Matt Iklé, Nejc Znidar, Marco Argentieri

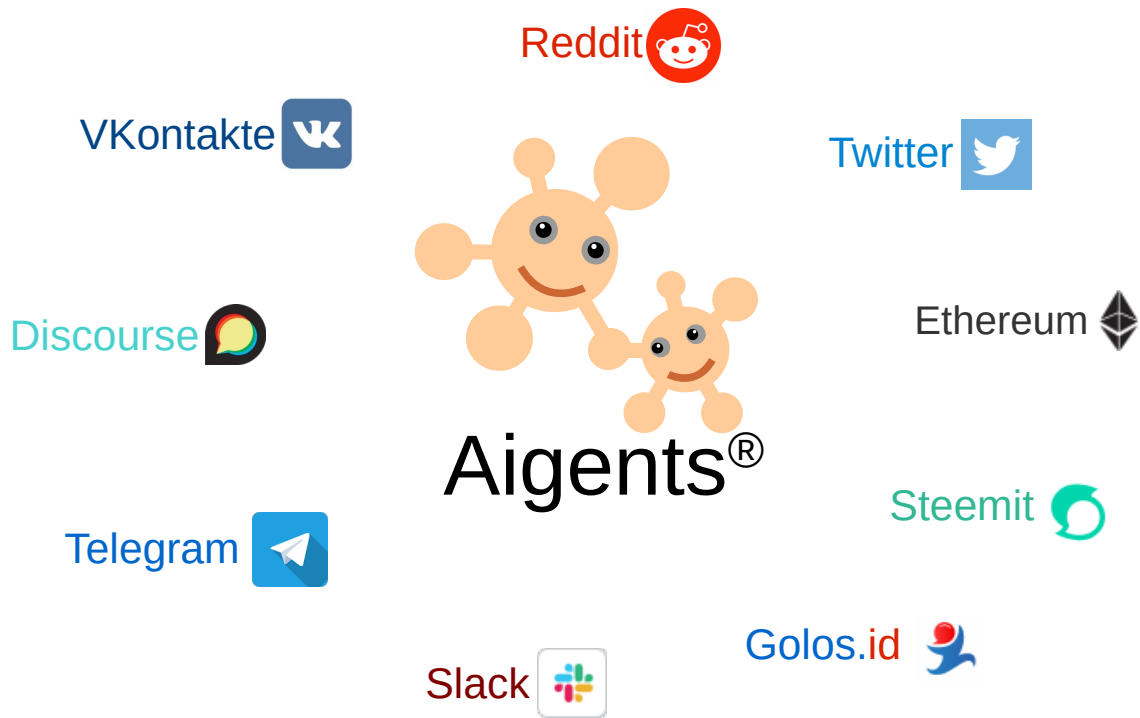
<https://arxiv.org/pdf/1905.08036.pdf>

<https://github.com/singnet/reputation>

<https://github.com/aigents/aigents-java/blob/master/src/main/java/net/webstructor/peer/Reputationer.java>

Reputation System for Social Platforms (2014-2022)

Unified Liquid Rank Reputation computation across diverse social platforms



<https://arxiv.org/abs/1912.00176>

<https://aigents.medium.com/aigents-bot-for-telegram-groups-1dba32140047>

“Weighted Liquid Rank” for Fraud Resistance (2019)

Using Reputation System for protection from scam identifying dishonest suppliers on online marketplaces



Ranks of Suppliers, dishonest Supplier (including alias) in red and honest suppliers in blue

<https://arxiv.org/pdf/1905.08036.pdf>

<https://blog.singularitynet.io/minimizing-recommendation-fraud-7dabbee8fc00>

https://aiforgood2019.github.io/papers/IJCAI19-AI4SG_paper_28.pdf

Reputation Consensus for Distributed Ledger (2021)

International Journal of Network Security & Its Applications (IJNSA) Vol.13, No.4, July 2021

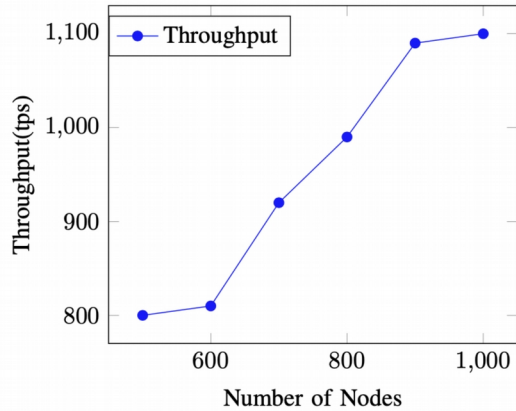


Figure 1: Throughput vs Number of network nodes

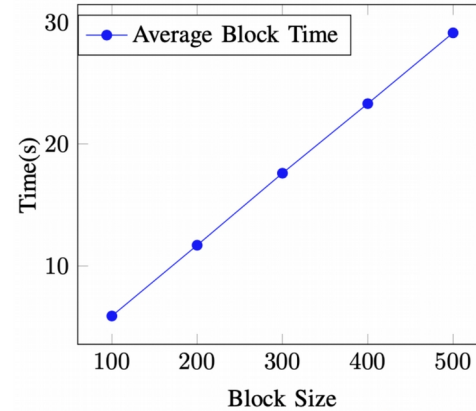


Figure 2: Average Block Time as the number of transactions in a single Block is varied

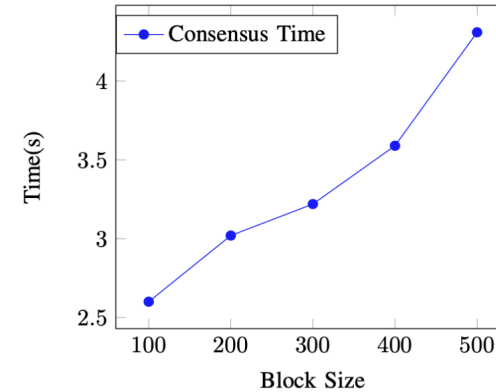


Figure 3: Consensus Time as the number of transactions in a single Block is varied

Proof-of-Reputation: An Alternative Consensus Mechanism for Blockchain Systems

<https://arxiv.org/abs/2108.03542>

<https://aircconline.com/ijnsa/V13N4/13421ijnsa03.pdf>

Table 1 shows the performance of our scheme against other existing consensus mechanisms.

Under certain conditions when the number of participating nodes is increased, our scheme can achieve up to 1,100 transactions per second.

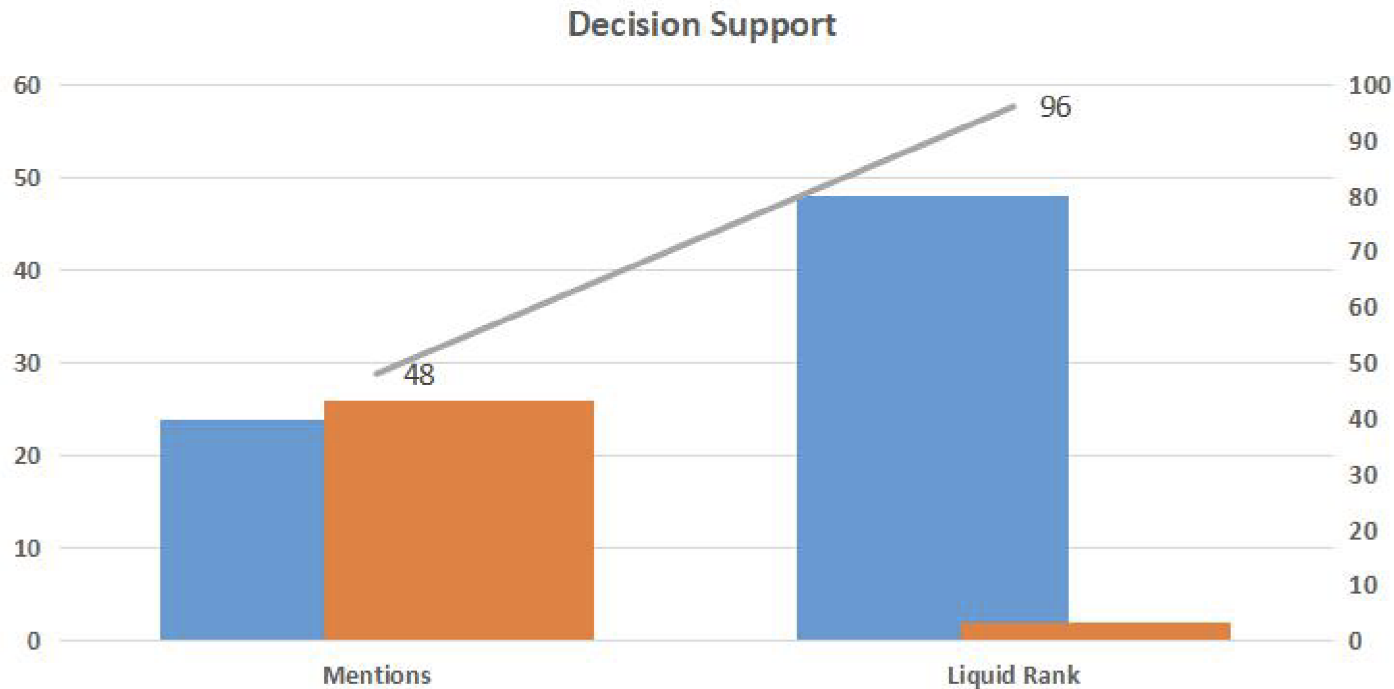
Table 1. Comparison with other consensus mechanisms

Consensus Mechanism	Throughput(TPS)
Proof-of-Work	7
Proof-of-Stake	60
Proof-of-Reputation(Baseline)	800
Proof-of-Burn	854
Proof-of-Reputation	1,100

Reputation System for Recommendation (2022)

Application of Liquid Rank Reputation System for Content Recommendation

RESULTS QUALITATIVE ANALYSIS: DECISION SUPPORT



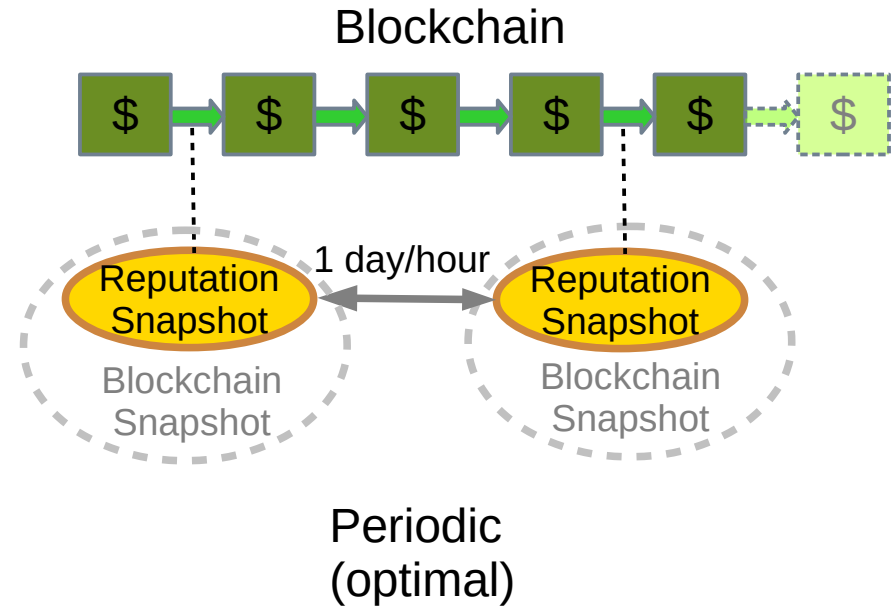
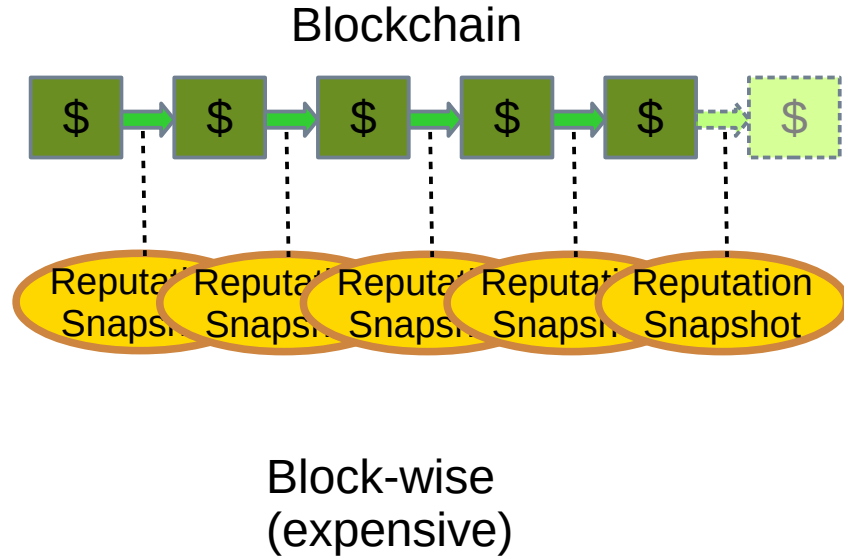
<https://arxiv.org/abs/2209.07641>

<https://ieeexplore.ieee.org/document/9923352>

■ Relevant ■ Irrelevant — Precision %

Copyright © 2023 Anton Kolonin, Aigents®

Reputation Consensus – Synchronization Options

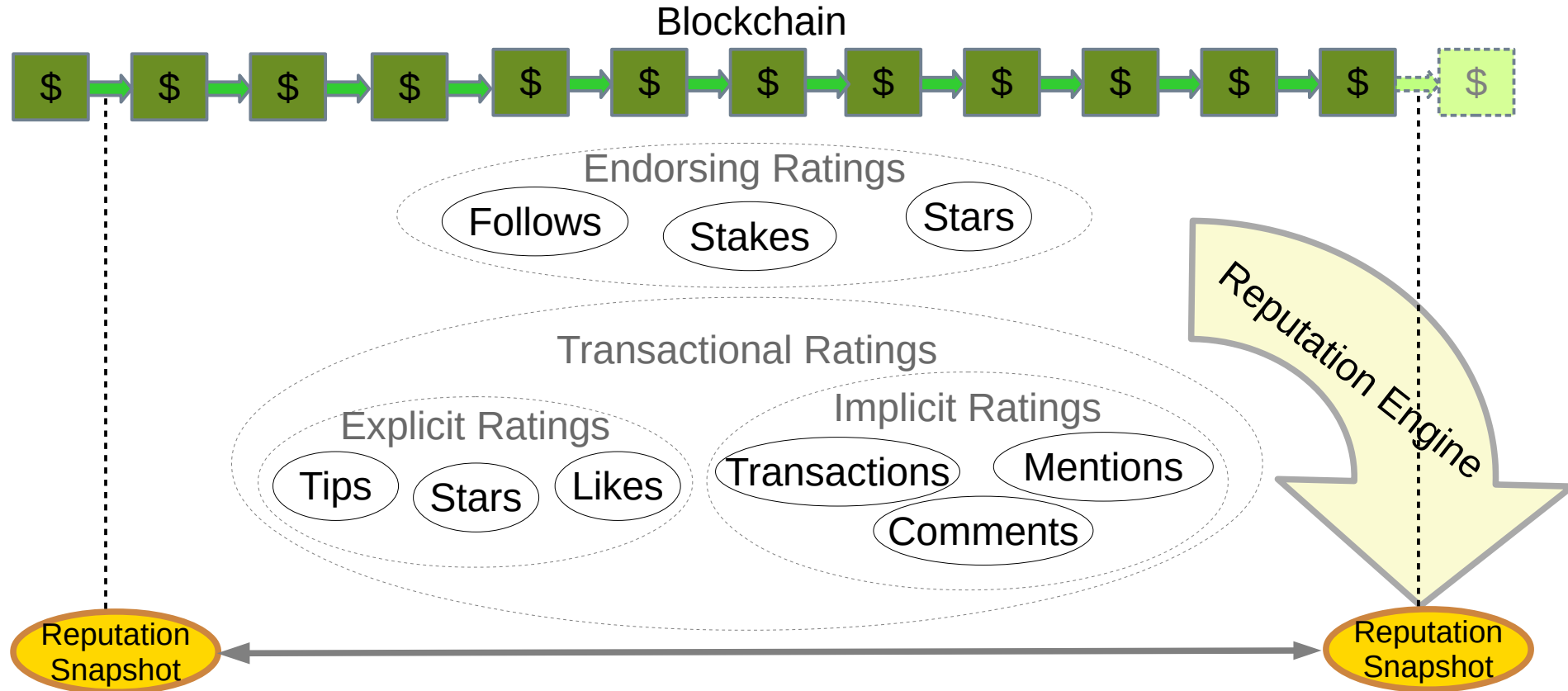


A Reputation System for Artificial Societies

Anton Kolonin, Ben Goertzel, Deborah Duong, Matt Ikle

<https://arxiv.org/pdf/1806.07342.pdf>

Reputation Consensus – Rating Sources

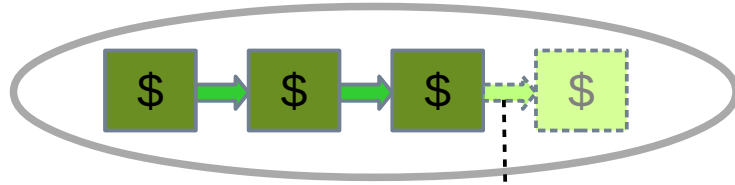


A Reputation System for Artificial Societies

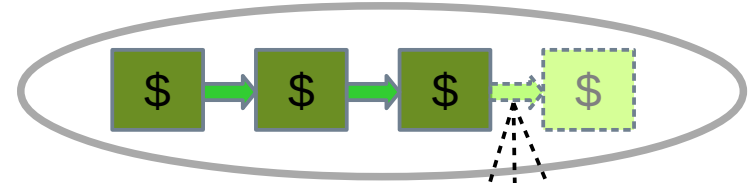
Anton Kolonin, Ben Goertzel, Deborah Duong, Matt Ikle

<https://arxiv.org/pdf/1806.07342.pdf>

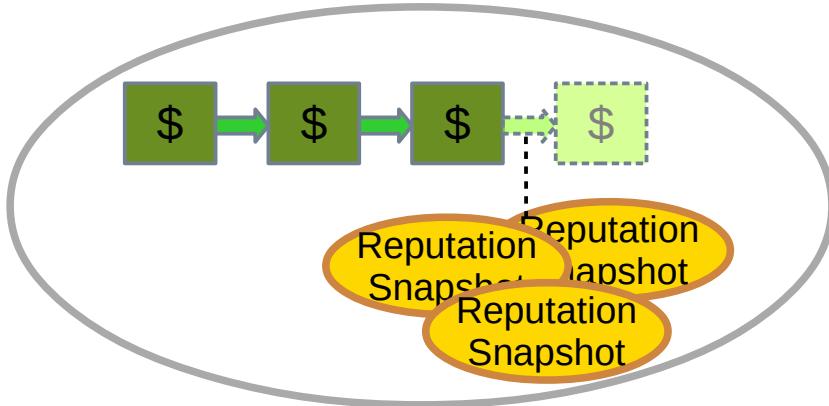
Reputation Consensus – Engine Design Options



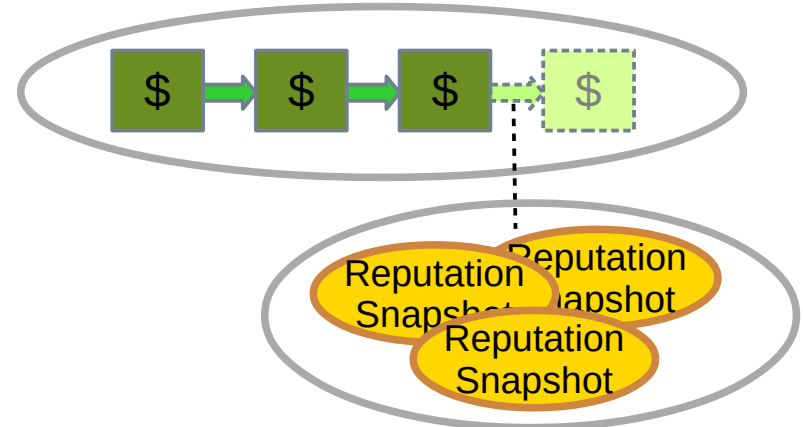
Centralized off-chain



Decentralized off-chain



Decentralized on-chain (reputation mining)



Decentralized side-chain (reputation consensus)

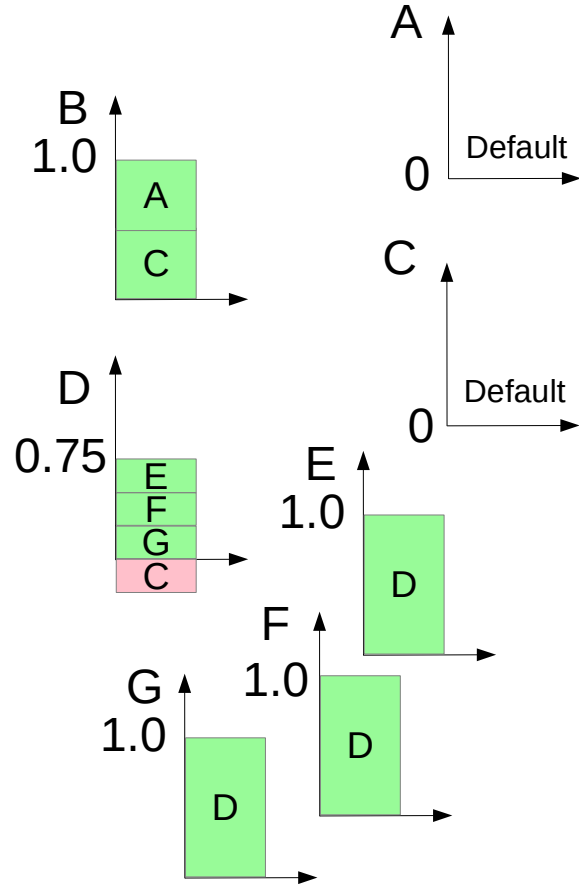
A Reputation System for Artificial Societies

Anton Kolonin, Ben Goertzel, Deborah Duong, Matt Ikle

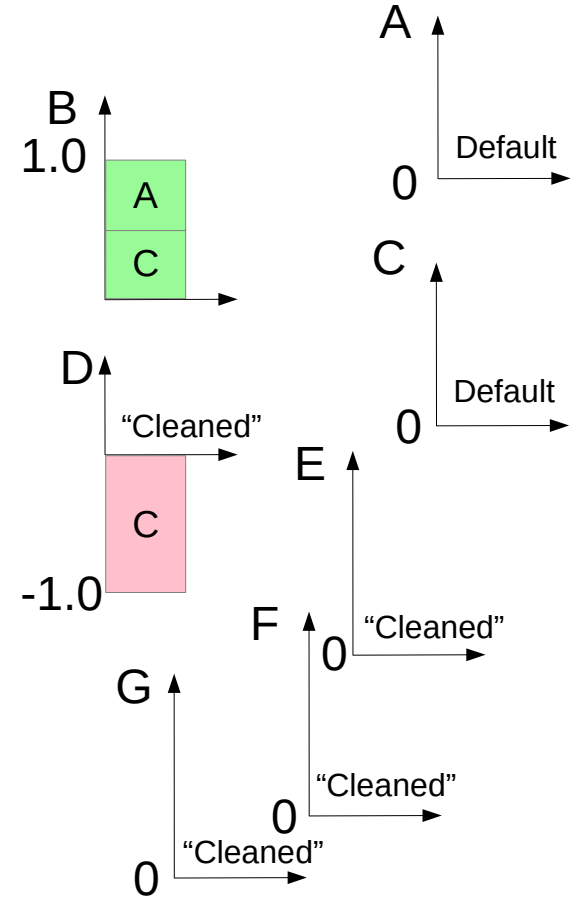
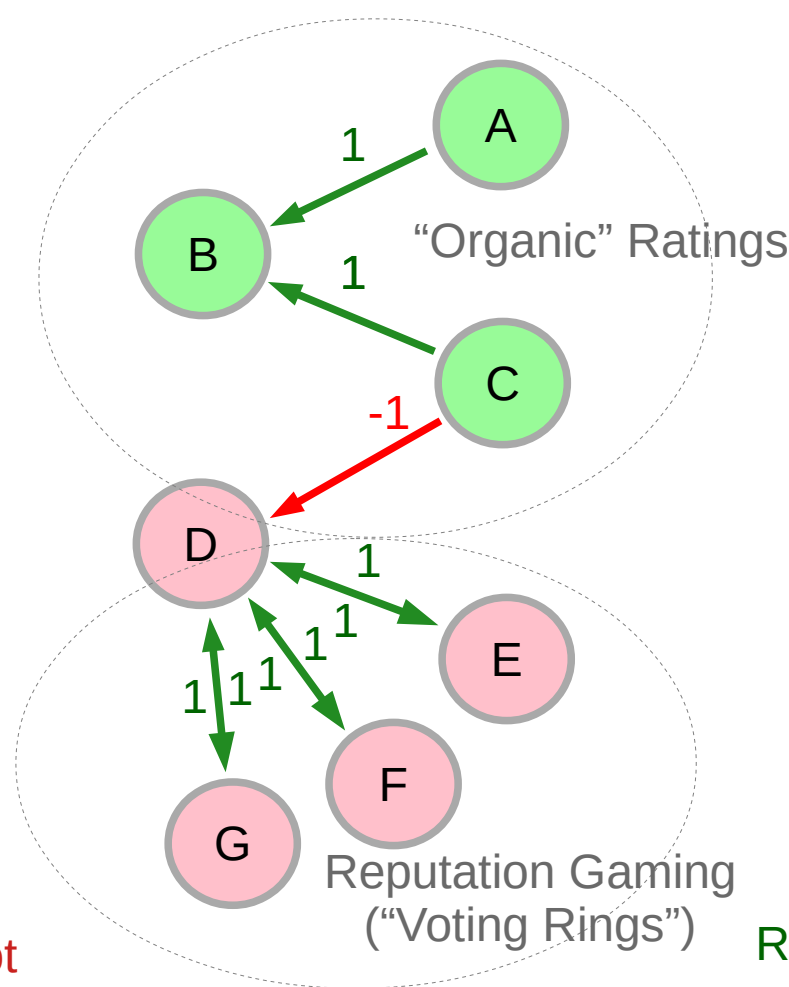
<https://arxiv.org/pdf/1806.07342.pdf>

Copyright © 2023 Anton Kolonin, Aigents®

Next: Resisting Reputation Gaming (Churning) [1.0..-1.0]

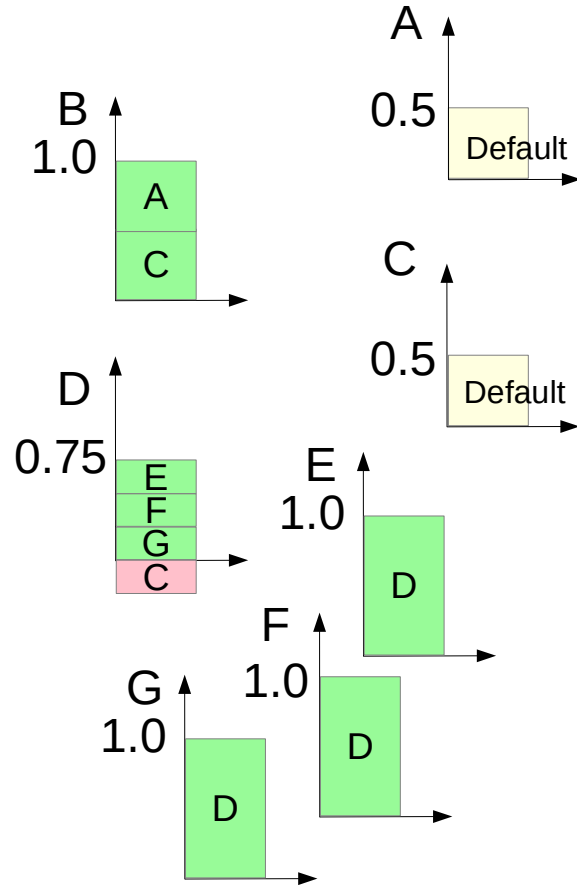


Abused Reputation Snapshot

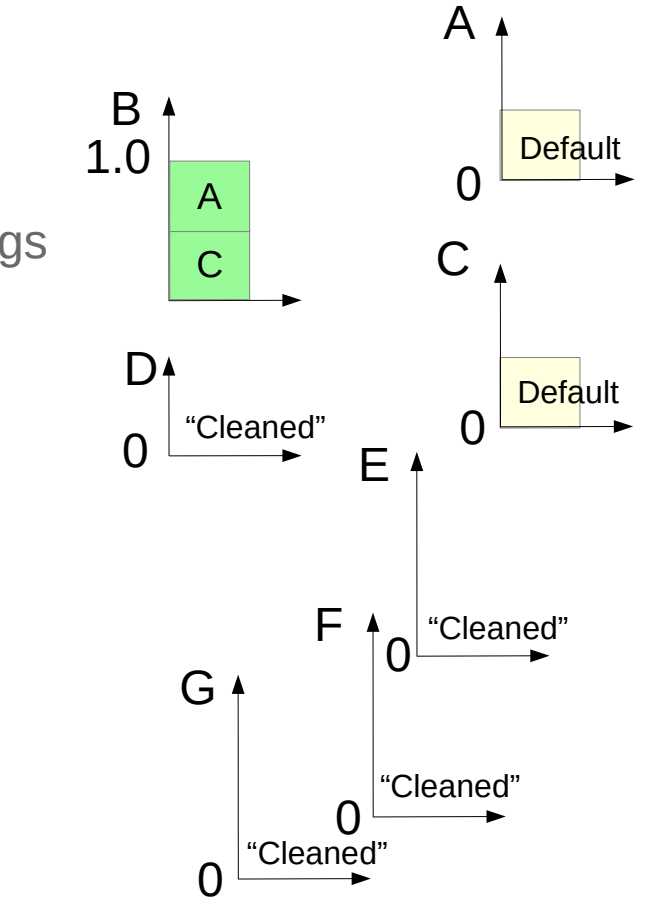
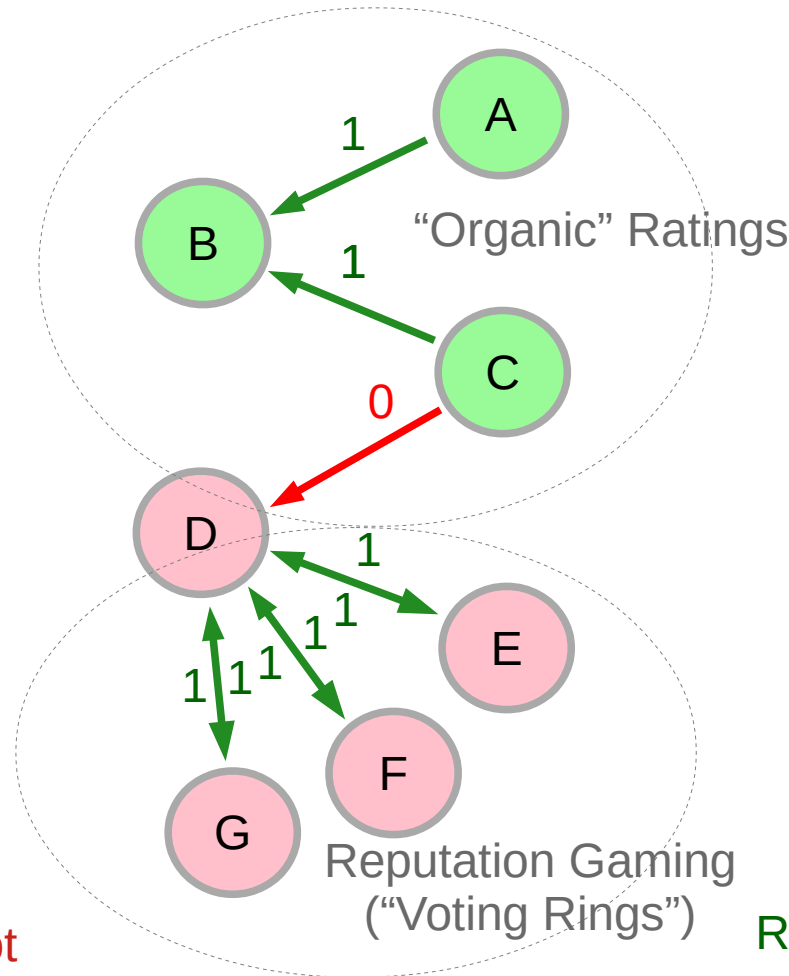


Rescued Reputation Snapshot

Next: Resisting Reputation Gaming (Churning)



Abused Reputation Snapshot



Rescued Reputation Snapshot

Thank You and Welcome!

Anton Kolonin

akolonin@aigents.com

Facebook: [akolonin](#)

Telegram: [akolonin](#)

