Brief Announcement: Grassroots Distributed Systems: Concept, Examples, Implementation and Applications

Ehud Shapiro
Weizmann Institute of Science, Rehovot, Israel

Abstract

Informally, a distributed system is *grassroots* if it is permissionless and can have autonomous, independently-deployed instances – geographically and over time – that may interoperate voluntarily once interconnected. More formally, in a grassroots system the set of all correct behaviors of a set of agents P is strictly included in the set of the correct behaviors of P when they are embedded within a larger set of agents $P' \supset P$.

Grassroots systems are potentially important as they may allow communities to conduct their social, economic, civic, and political lives in the digital realm solely using their members' networked computing devices (e.g., smartphones), free of third-party control, surveillance, manipulation, coercion, or rent seeking (e.g., by global digital platforms such as Facebook or Bitcoin).

Client-server/cloud computing systems are not grassroots, and neither are systems designed to have a single global instance (Bitcoin/Ethereum with hardwired seed miners/bootnodes), and systems that rely on a single global data structure (IPFS, DHTs). An example grassroots system would be a serverless smartphone-based social network supporting multiple independently-budding communities that can merge when a member of one community becomes also a member of another.

Here, we formalize the notion of grassroots distributed systems; describe a grassroots dissemination protocol for the model of asynchrony and argue its safety, liveness, and being grassroots; extend the implementation to mobile (address-changing) devices that communicate via an unreliable network (e.g. smartphones using UDP); and discuss how grassroots dissemination can realize grassroots social networking and grassroots cryptocurrencies. The mathematical construction employs distributed multiagent transition systems to define the notions of grassroots protocols, to specify the grassroots dissemination protocols, and to prove their correctness. The protocols use the blocklace – a distributed, partially-ordered counterpart of the replicated, totally-ordered blockchain.

2012 ACM Subject Classification Computer systems organization → Peer-to-peer architectures; Networks → Network protocol design; Networks → Formal specifications; Software and its engineering → Distributed systems organizing principles

Keywords and phrases Grassroots Distributed Systems, Dissemination Protocol, Multiagent Transition Systems, Blocklace, Cordial Dissemination

Digital Object Identifier 10.4230/LIPIcs.DISC.2023.47

Related Version *Full Version*: `https://arxiv.org/abs/2301.04391`

Acknowledgements I thank Andy Lewis-Pye, Nimrod Talmon and Oded Naor for their comments on an earlier version of the manuscript and Idit Keidar for pointing me to related work. Ehud Shapiro is the Incumbent of The Harry Weinrebe Professorial Chair of Computer Science and Biology at the Weizmann Institute.

1 Introduction

The digital realm is dominated by global digital platforms of two architectures: The first is autocratic (one person – all votes) cloud-based global digital platforms that have adopted surveillance-capitalism [20] as their business model, driving the platforms to monitor, induce, and manipulate their inhabitants for profit. Some regimes also require a "back-door" to the global platform to monitor, censor, control, and even punish the digital behavior of its citizens. The second is blockchain/cryptocurrencies-based systems, which are peer-to-peer and can be "permissionless", open to participation by everyone. These platforms are governed via proof-of-work or proof-of-stake protocols, which are intrinsically plutocratic (one coin – one vote). Despite the promise of openness and distribution of power, the leading cryptocurrency platforms are controlled by a handful of ultra-high-performance server-clusters [8]. See the full paper [13] for a discussion of other server-based systems [7, 3, 1, 6, 17, 4, 18].

Here, we are concerned with providing an alternative architecture for the digital realm, referred to as a *grassroots architecture*, to serve as a foundations for peer-to-peer, smartphone-based, serverless applications [18, 14]. Ultimately, a grassroots architecture may provide a protocol stack to support grassroots digital democracy [15].

In general, a system designed to have a single global instance is not grassroots. Client-server/cloud systems in which two instances cannot co-exist due to competition/conflict on shared resources (e.g., same web address), or cannot interoperate when interconnected, are not grassroots. Neither are peer-to-peer systems that require all-to-all dissemination, including mainstream cryptocurrencies and standard consensus protocols [5, 19, 9], since a community placed in a larger context cannot ignore members of the larger context. Neither are systems that use a global shared data-structure such as pub/sub systems [6], IPFS [3], and distributed hash tables [16], since a community placed in a larger context cannot ignore updates to the shared resource by others.

2 Grassroots Protocols

We assume a potentially-infinite set of agents Π (think of all the agents yet to be produced), but when referring to a subset of the agents $P \subseteq \Pi$ we assume P to be finite. Each agent is associated with a single and unique key-pair of its own choosing, and is identified by its public key $p \in \Pi$. We refer the reader to the full paper [13] for the necessary definitions and their explanations, and introduce the notion of a grassroots protocol without further ado:

▶ **Definition 1** (Grassroots). *A protocol $\mathcal{F}$ is* ***grassroots*** *if $\emptyset \subset P \subset P' \subseteq \Pi$ implies that $TS(P) \subset TS(P')/P$.*

Informally, a group of agents P with the a set of possible behaviors $TS(P)$ has possible behaviors $TS(P')/P$ when embedded within a larger group P'. A protocol is grassroots if: (i) The behaviors of the agents P on their own, $TS(P)$, are also possible behaviors of these agents when embedded within a larger group P', $TS(P')/P$. In other words, the agents in P may choose to ignore the agents in $P' \setminus P$. Hence the subset relation. (ii) This latter set of behaviors $TS(P')/P$ includes additional possible behaviors of P not in $TS(P)$. Thus, there are possible behaviors of P, when embedded within P', which are not possible when P are on their own. This is presumably due to interactions between members of P and members of $P' \setminus P$. Hence the subset relation is strict.

▶ **Observation 2.** *An all-to-all dissemination protocol cannot be grassroots.*

Intuitively, a group of agents P engaged in a hypothetical grassroots all-to-all dissemination protocol may ignore the additional agents in $P' \setminus P$, in contradiction to the dissemination protocol being all-to-all. The full paper provides a sufficient condition for a protocol to be grassroots, which are useful in proving such a claim:

▶ **Theorem 3** (Grassroots Protocol). *An asynchronous, interactive, and non-interfering protocol is grassroots.*

Informally, a protocol is *asynchronous* if a transition by an agent, once enabled, cannot be disabled by transitions taken by other agents; it is *interactive* if the addition of agents to a group results in additional possible behaviors of the group; and it is *non-interfering* if the possible behaviors of a group of agents are not hampered by the presence of additional stationary agents, namely agents that remain in their initial state.

3 The Social Graph

Paul Baran's original vision of the Internet [2] was of a network of unmanned nodes that would act as switches, routing information from one node to another to their final destinations. Grassroots dissemination is different from packet switching: A block in a blocklace has no "destination", only an author, and dissemination occurs via communication along the edges of the social graph, based on the following social principles.

Social Principles of Cordial Dissemination:
1. **Disclosure**: Tell your friends which blocks you know and which you need
2. **Cordiality**: Send to your friends blocks you know and think they need

In Cordial Dissemination, agents may *follow* other agents, and two agents are *friends* if they follow each other. The basic rule of Cordial Dissemination is that an agent p can receive from agent q a q'-block b if p and q are friends and either $q = q'$ or both p and q follow q'. Note that the friendship relation induces an undirected graph on the agents, referred to as the *social graph*. A *friendship path* is a path in the social graph. The key liveness claim of Cordial Dissemination is that p eventually knows any q-block if there is a friendship path of correct agents from p to q, all of which follow q. The social principles of Cordial Dissemination are realized via the blocklace, introduced next.

4 The Blocklace

The blocklace is a distributed, partially-ordered counterpart of the replicated, totally-ordered blockchain data-structure. It is formally introduced in the full paper [13], and has already been applied to the implementation of grassroots social networking [14], the Flash payment system [11], and the Cordial Miners family of consensus protocols [10]. Here is a concise introduction of its basic concepts.

We assume a given cryptographic hash function *hash*. A *block* created by agent $p \in \Pi$, also referred to as a *p-block*, is a triple $b = (h, x, H)$, with h being a hash pointer $\mathit{hash}((x, H))$ signed by p, also referred to as a *p-pointer*; x being the *payload* of b; and H a finite set of signed hash pointers. If $H = \emptyset$ then b is a *genesis block*; an hash pointer $h' \in H$ points to the block $(h', x', H)'$, if such a block exists; and if h' is a p-pointer it is also referred to as a *self-pointer*.

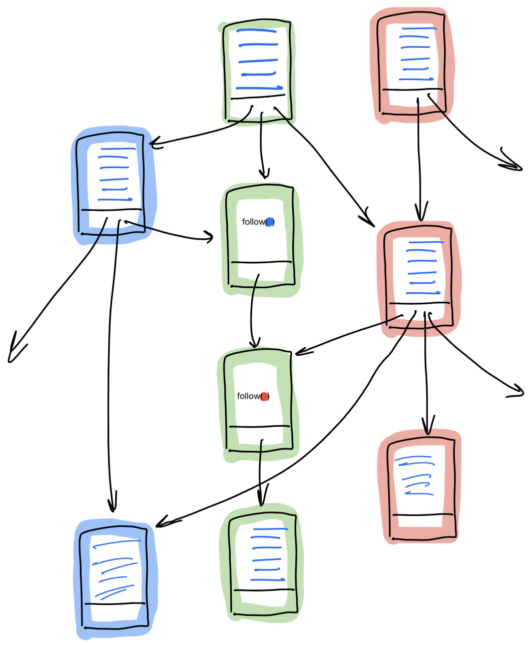

Figure 1 A self-closed blocklace: Blocks are color-coded by agent, thus pointers among blocks of the same color are self-pointers, others are non-self pointers, which may be dangling.

A *blocklace* is a set of blocks, which is *closed* if every pointer in every block in B points to a block in B. A pointer that does not point to a block in B is *dangling* in B, hence a closed blocklace has no dangling pointers. A blocklace is *self-closed* if it has no dangling self-pointers. Note that the notion of a cryptographic hash function implies that it is computationally-infeasible to create hash pointers that form a cycle. We are concerned only with computationally-feasible blocklaces, and any such blocklace B induces a DAG, with the blocks of B as vertices and with directed edges $b \to b'$ for every $b, b' \in B$ for which b includes a pointer to b'. A block b *observes* a block b' in B if there is a path $b = b_1 \to b_2 \ldots \to b_n = b'$ in B, $n \geq 1$. We note that the "observes" relation is a partial order on B. Two p-blocks that do not observe each other are referred to as an *equivocation* by p, and if their payloads include conflicting financial transactions by p, they are also called a *double-spend* by p. If B includes an equivocation by p we say that p is an *equivocator* in B.

5 Cordial Dissemination

Here we present the blocklace-based Cordial Dissemination Protocol $\mathcal{CD}$; claim it to be live (Prop. 4) and grassroots (Prop. 5); present pseudocode realizing $\mathcal{CD}$ for the model of Asynchrony (Alg. 1); and discuss an implementation for mobile agents communicating over an unreliable network, namely smartphones communicating via UDP.

We employ the blocklace as follows. The *local state* of each agent p is a blocklace, consisting of blocks produced by p and blocks by agents that p follows and that were received by p. A correct agent maintains a self-closed blocklace, buffering received out-of-order blocks. A *configuration* c consists of a set of local states, one for each agent. The local state of agent p in c is denoted by c_p. Given agents $p, q \in \Pi$, then p *follows* q in c if c_p includes a p-block with payload (FOLLOW, q); p *needs* the q-block b in c if p follows q in c_p and $b \in c_q \setminus c_p$; and p and q are *friends* in c if p and q follow each other in c. The *social graph* $(P, E(c))$ induced by a configuration c has an edge $(p, q) \in E(c)$ if p and q are friends in c.

The social principles of Cordial Dissemination are realized by the blocklace as follows: Disclosure is realized by the Create transition, with any new p-block serving as a *multichannel ack/nack message*, informing whether p follows another agent q, and if so also of the latest q-block known to p, for every $q \in \Pi$. Cordiality is realized by the Cordially-Send-b-to-q transition. The liveness condition requires an agent to disclose every so often the blocks they know and to receive any block sent to it. The Follow transition allows p to offer friendship to any agent q; but there is no liveness requirement on p to do so. Agents p and q are friends if they follow each other, namely both have produced a FOLLOW block for the other. Agent p of course knows if it follows q; but p can know that q follows it only if it receives a (FOLLOW, p) block b. However, b may include q-pointers to blocks p does not know yet, hence p may have to "peek" into its received but not-yet-incorporated blocks and look for a (FOLLOW, p) q-block, in order to know whether it is friends with q.

The full paper describes the cordial dissemination protocol as a family of distributed multiagent transition systems. Here we capture the essence of the protocol informally.

The $\mathcal{CD}$ Cordial Dissemination Protocol
The local state of each agent p consists of a blocklace B, which initially includes a genesis p-block that has no payload and no pointers, as well as received blocks not yet incorporated in B.
The protocol proceeds by any agent p taking any of the following transitions:

1. **Create/Follow**: Create a new p-block b that points to the tips of B, as well as to the previous p-block, and add b to B. If the payload of b is (FOLLOW, q) then send b to q.
2. **Cordially-Send-*b*-to-*q***: If B has a block b such that (i) p knows that q is a friend, (ii) p knows that q follows the creator of b, and (iii) p does not know that q knows b, then send b to q.
3. **Receive-*b***: If a received q-block b is not in B then add b to B, provided p follows q and any q-block b points to is already in B.

The liveness condition of the protocol requires that every message sent is eventually received, and every agent every so often creates a new block and cordially sends blocks to their friends.

The safety assurance of the protocol is that the local blocklace of any correct agent p is self-closed and has no equivocations by p (but may include equivocation by faulty agents). The following liveness proposition holds for the model of asynchrony:

▶ **Proposition 4** ($\mathcal{CD}$ Liveness). *Let r be a run of $\mathcal{CD}$, $p, q \in P$. If in some configuration $c \in r$, p and q are connected via a friendship path, all of its members follow q in c and are correct in r, then for every q-block b in r there is a configuration $c' \in r$ for which $b \in c'_p$.*

▶ **Proposition 5.** *The Cordial Dissemination protocol $\mathcal{CD}$ is grassroots.*

6 Pseudocode Implementation

Algorithm 1 presents pseudocode implementation of the Cordial Dissemination protocol $\mathcal{CD}$, for an single agent p for the model of Asynchrony. We assume that the agent p can specify the payload for a new block, including it being a friendship offer. As according to the model of asynchrony each message sent is eventually received, we assume the **reliably_send** construct to keep a record of sent messages so as not to send the same message to the same agent twice.

Algorithm 1 Grassroots Cordial Dissemination for Asynchrony

Code for agent p.

```
    Local variables:
 1: $B \leftarrow \{create\_block(\bot, \emptyset)\}$                              ▷ The local blocklace of agent $p$

 2: upon decision to create block with payload $x$ do
 3:     $b \leftarrow create\_block(x, tips(B))$                                        ▷ 1. Create
 4:     $B \leftarrow B \cup \{b\}$
 5:     if $x = (\text{FOLLOW}, q)$ then reliably_send $b$ to $q$                            ▷ 1. Follow

 6: upon a new block in $B$ do                                          ▷ 2. Cordially-Send-$b$-to-$q$
 7:     for all $b \in B, q \in P : friend(q) \wedge follows(q, b.creator) \wedge \neg agentObserves(q, b)$ do
 8:         reliably_send $b$ to $q$

 9: upon receive $b$ s.t. $B \cup \{b\}$ is self-closed and $p$ follows $b.creator$ do          ▷ 3. Receive-$b$
10:     $B \leftarrow B \cup \{b\}$
```

Mobile Agents Communicating via UDP. A refinement of the Cordial Dissemination protocol for mobile (address-hopping) agents communicating over an unreliable network, namely smartphones communicating via UDP, is presented in the full paper, and a more concrete variant of it is preserted in the context of grassroots social networking [14]. Cordial dissemination over UDP exploits the ack/nak information of blocklace blocks to its fullest, by p retransmitting to every friend q every block b that p knows (not only p-blocks) and believes that q needs, until q acknowledges knowing b. In this protocol, every block includes also the IP address of its creator at the time of creation, and a new p-block is created whenever p changes its IP address. Retransmission is initiated by timeout, and assuming that timeouts are separated by seconds and mobile address changes are independent and are separated by hours, the probability of two friends hopping together without one successfully informing the other of its new IP address is around 10^{-7}. If the two hopping friends have a stationary joint friend, then it is enough that one of the hoppers successfully informs the stationary friend of the address change, for the other hopper to soon know this new address from their stationary common friend. Under the same assumptions, the probability of a clique of n friends loosing a member due to all hopping simultaneously is around $10^{-3.6*n}$. Note that such a loss is not terminal – assuming that friends have redundant ways to communicate (physical meetings, email, SMS, global social media), new addresses can be communicated and the digital friendship restored.

7 Applications of Cordial Dissemination

Applications of blocklace-based cordial dissemination include grassroots social networking [14], grassroots cryptocurrencies[12], as well as non-grassroots applications: The Flash payment system [11] and the Cordial Miners family of consensus protocol [10].

References

1 Ittai Abraham, Kartik Nayak, Ling Ren, and Zhuolun Xiang. Good-case latency of byzantine broadcast: A complete categorization. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC'21, pages 331–341, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3465084.3467899.

2 Paul Baran. On distributed communications networks. *IEEE transactions on Communications Systems*, 12(1):1–9, 1964.

3 Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv:1407.3561*, 2014.

4 Sonja Buchegger, Doris Schiöberg, Le-Hung Vu, and Anwitaman Datta. Peerson: P2p social networking: early experiences and insights. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pages 46–52, 2009.

5 Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OsDI*, volume 99, pages 173–186, 1999.

6 Gregory Chockler, Roie Melamed, Yoav Tock, and Roman Vitenberg. Constructing scalable overlays for pub-sub with many topics. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, pages 109–118, 2007.

7 Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, pages 68–72. Berkeley, CA, USA, 2003.

8 Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. In *International Conference on Financial Cryptography and Data Security*, pages 439–457. Springer, 2018.

9 Idit Keidar, Eleftherios Kokoris-Kogias, Oded Naor, and Alexander Spiegelman. All you need is dag, 2021. arXiv:2102.08325.

10 Idit Keidar, Oded Naor, and Ehud Shapiro. Cordial miners: A family of simple and efficient consensus protocols for every eventuality. *arXiv preprint arXiv:2205.09174*, 2022.

11 Andrew Lewis-Pye, Oded Naor, and Ehud Shapiro. Flash: An asynchronous payment system with good-case linear communication complexity. *arXiv preprint arXiv:2305.03567*, 2023.

12 Ehud Shapiro. Grassroots cryptocurrencies: A foundation for a grassroots digital economy. *arXiv preprint arXiv:2202.05619*, 2022.

13 Ehud Shapiro. Grassroots distributed systems: Concept, examples, implementation and applications. *arXiv preprint arXiv:2301.04391*, 2023.

14 Ehud Shapiro. Grassroots social networking: Serverless, permissionless protocols for twitter-/linkedin/whatsapp. *arXiv preprint arXiv:2306.13941*, 2023.

15 Ehud Shapiro and Nimrod Talmon. Foundations for grassroots democratic metaverse. In *AAMAS '22: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 1814–1818, 2022.

16 Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM computer communication review*, 31(4):149–160, 2001.

17 Robert Strom, Guruduth Banavar, Tushar Chandra, Marc Kaplan, Kevan Miller, Bodhi Mukherjee, Daniel Sturman, and Michael Ward. Gryphon: An information flow based approach to message brokering. *arXiv preprint cs/9810019*, 1998.

18 Dominic Tarr, Erick Lavoie, Aljoscha Meyer, and Christian Tschudin. Secure scuttlebutt: An identity-centric protocol for subjective and decentralized applications. In *Proceedings of the 6th ACM conference on information-centric networking*, pages 1–11, 2019.

19 Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pages 347–356, 2019.

20 Shoshana Zuboff. *The age of surveillance capitalism: The fight for a human future at the new frontier of power: Barack Obama's books of 2019*. Profile books, 2019.