RESEARCH-ARTICLE

Grassroots Platforms with Atomic Transactions: Social Graphs, Cryptocurrencies, and Democratic Federations

EHUD SHAPIRO, London School of Economics and Political Science, London, U.K.

Published: 06 January 2026

ICDCN 2026: 27th International Conference on Distributed Computing and Networking
January 6 - 9, 2026
Nara, Japan

ICDCN '26: Proceedings of the 27th International Conference on Distributed Computing and Networking (January 2026)
https://doi.org/10.1145/3772290.3772309
ISBN: 9798400718885

Grassroots Platforms with Atomic Transactions: Social Graphs, Cryptocurrencies, and Democratic Federations

Ehud Shapiro
Weizmann Institute of Science, London School of Economics
Rehovot, Israel
Department of Mathematics and Data Science Institute
London School of Economics
London, United Kingdom
ehud.shapiro@weizmann.ac.il

Abstract

Grassroots platforms aim to offer an egalitarian alternative to global platforms — centralized/autocratic (Facebook etc.) and decentralized/plutocratic (Bitcoin etc.) alike. Whereas global platforms can have only a single instance—one Facebook, one Bitcoin—grassroots platforms can have multiple instances that emerge and operate independently of each other and of any global resource except the network, and can interoperate and coalesce into ever-larger instances once interconnected, potentially (but not necessarily) forming a single instance. Key grassroots platforms include grassroots social graphs, grassroots social networks, grassroots cryptocurrencies, and grassroots federations. Previously, grassroots platforms were defined formally and proven grassroots using unary transition systems, in which each transition is carried out by a single agent. However, grassroots platforms cater for a more abstract specification using transactions carried out atomically by multiple agents, something that cannot be expressed by unary transition systems. As a result, their original specifications and proofs were unnecessarily cumbersome and opaque.

Here, we aim to provide a more suitable formal foundation for grassroots platforms. To do so, we enhance the notion of a multiagent transition system to include atomic transactions and revisit the notion of grassroots platforms within this new foundation. We present concise specifications of key grassroots platforms using atomic transactions: befriending and unfriending for grassroots social graphs, coin swaps for grassroots cryptocurrencies, and communities forming, joining, and leaving a federation for grassroots federations. We prove a general theorem that a platform specified by atomic transactions that are interactive is grassroots; show that the atomic transactions used to specify all three platforms are interactive; and conclude that the platforms thus specified are indeed grassroots. We thus provide a crisp mathematical foundation for grassroots platforms and a solid and clear starting point from which their implementation can commence.

ICDCN 2026, Nara, Japan

ACM ISBN 979-8-4007-1888-5/26/01
https://doi.org/10.1145/3772290.3772309

CCS Concepts

• **Computer systems organization → Peer-to-peer architectures**; • **Networks → Network protocol design**; **Formal specifications**; • **Software and its engineering → Distributed systems organizing principles**.

Keywords

Grassroots Protocols, Multiagent Transition Systems, Atomic Transactions, Social Graphs, Cryptocurrencies, Democratic Federations, Peer-to-peer Systems

ACM Reference Format:
Ehud Shapiro. 2026. Grassroots Platforms with Atomic Transactions: Social Graphs, Cryptocurrencies, and Democratic Federations. In *27th International Conference on Distributed Computing and Networking (ICDCN 2026), January 06–09, 2026, Nara, Japan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3772290.3772309

1 Introduction

Background. The Internet today is dominated by centralised global platforms—social networks, Internet commerce, 'sharing-economy'—with autocratic control [43, 44]. An alternative is emerging—blockchains and cryptocurrencies [16, 17, 28, 39, 40, 42]—that are also global platforms, but with decentralized, plutocratic control [8].

Grassroots platforms [33–35, 37] aim to offer an egalitarian alternative to global platforms, centralized and decentralized alike. Global platforms can only have a single instance–one Facebook, one Bitcoin—as two instances of the platform would clash over the global resources they utilize—domain name, port number, or boot nodes, which are hardwired into their code. Even if it is possible to modify their code ('hard-fork') to create non-conflicting instances—Facebook′, Bitcoin′—such forked instances would ignore, rather than interoperate with, the primary instance. Grassroots platforms, in contrast, can have multiple instances that emerge and operate independently of each other and of any global resource except the network, yet can interoperate and coalesce once interconnected, forming ever-larger platform instances, potentially (but not necessarily) coalescing into a single instance.

Any platform that operates on a shared global resource or employs a single replicated (Blockchain [29]), or distributed (IPFS [4], DHT [31]) shared global data structure, as well as distributed pub/sub systems with a global directory [7, 11, 12], are all not grassroots. BitTorrent [1] and Mastodon [30] are similar in spirit but are peer-to-peer among servers not people; in particular, in such systems a

group of people with smartphones cannot do more if more people with smartphones join the group, unless they are also joined by a server, thus these systems do not fall under the definition of grassroots. An example of a platform that is grassroots in spirit (even if not formally proven as such) is Scuttlebutt [20, 38].

Motivation. Key grassroots platforms include grassroots social graphs and networks [34], grassroots cryptocurrencies [23, 35], and grassroots federations [18, 37]. Previously, grassroots platforms were defined formally using unary multiagent transition systems, in which each transition is carried out by a single agent [33]. However, grassroots platforms cater for a more abstract specification using transactions carried out atomically by multiple agents, something that cannot be expressed by unary transition systems. As a result, their original specifications were more cumbersome and opaque than they should be. Moreover, in grassroots platforms transactions carried out by different sets of participants need not be synchronized with each other, which greatly simplifies their mathematical foundations.

Here, we aim to provide a crisp formal foundation for these key grassroots platforms and beyond. To do so, we enhance the notion of a multiagent transition system to include atomic transactions and revisit the notions of grassroots platforms and their implementation within this enhanced foundation.

Furthermore, previous work [33] provided sufficient conditions for when a platform (formally, a protocol defined via a family of multiagent transition systems) is grassroots. While going through the route presented in previous work is possible, the new foundations offer a simpler, more direct and mathematically preferable way to prove that a platform, specified via atomic transactions, is grassroots. Here, we follow this simpler path.

Contributions. This paper:

(1) Presents atomic transactions and how they induce multiagent transition systems.
(2) Provides crisp specifications of three key grassroots platforms via atomic transactions: Social graphs, cryptocurrencies, and democratic federations.
(3) Provides a simpler and more stringent definition of grassroots platforms.
(4) Provides a sufficient condition for a set of transactions to induce a grassroots protocol.
(5) Shows that the atomic transactions employed in specifying each of the three key platforms satisfy this condition and therefore are indeed grassroots.

Key Grassroots Platforms. Three exemplar grassroots platforms motivate our work: grassroots social graphs, where people maintain their friendship connections through local storage and peer-to-peer relationships without central control; grassroots cryptocurrencies, where individuals mint personal coins and establish mutual credit lines through atomic swaps; and grassroots federations, where communities form and federate democratically without central coordination. Section 2 presents their detailed specifications using atomic transactions.

Paper outline. Section 2 introduces atomic transactions and presents transactions-based specifications of three key grassroots platforms: Grassroots social graphs, grassroots cryptocurrencies, and grassroots federations. Section 3 introduces multiagent transition systems and how they are induced by a set of atomic transactions. Section 4 provides multiagent transition systems of the three platforms as induced by their atomic transactions and proves their safety properties. Section 5 introduces protocols, grassroots protocols, protocols induced by atomic transactions, interactive atomic transactions, and proves the main theorem: That a protocol induced by an interactive set of atomic transactions is grassroots. It then deduces that the three specified grassroots platforms are indeed grassroots. Section 6 discusses, informally, the implementation of grassroots platforms. Section 7 discusses related and future work and concludes.

2 Grassroots Platforms with Atomic Transactions

Grassroots protocols were defined formally using unary multiagent transition systems [33]. Thus, higher-level protocols that include atomic transactions by multiple agents could not be expressed directly within this formalism, let alone proven grassroots.

Yet, the grassroots platforms we are interested in—grassroots social networks, grassroots cryptocurrencies, and grassroots federations—all call for atomic transactions for their specification. Hence this paper. Here, we use atomic transactions carried out by a set of agents as the basis for specifying grassroots platforms. We revisit the grassroots platforms previously defined using unary transition systems—social graphs and cryptocurrencies—and redefine them more simply and abstractly using binary atomic transactions. In addition, we present a k-ary transactions-based specification of grassroots federations, which where hitherto defined only at the community level.

2.1 Atomic Transactions

Agent, state, configuration. We assume a potentially infinite set of *agents* Π (think of all the agents that are yet to be born), but consider only finite subsets of it, so when we refer to a particular set of agents $P \subset \Pi$ we assume P to be nonempty and finite. We use $\subset$ to denote the strict subset relation and $\subseteq$ when equality is also possible.

In the context of multiagent systems it is common to refer to the state of the system as *configuration*, so as not to confuse it with the *local states* of the agents. As standard, we use S^P to denote the set S indexed by the set P, and if $c \in S^P$ we use c_p to denote the member of c indexed by $p \in P$. Intuitively, think of such a $c \in S^P$ as an array of cells indexed by members of P and with cell values in S.

Atomic transactions. Informally, an atomic transaction:

(1) Can have several active participants (be binary, k-ary) that change their local states atomically as specified.
(2) Specifies explicitly its participants (both active and stationary), thus implicitly defining infinitely-many multiagent transitions where all non-participants remain in their arbitrary state.

In addition to being atomic, the description above implies that transactions are asynchronous [32], in the sense that if a transaction can be carried out by its participants, with non-participants being

in any arbitrary states, it can still be carried out no matter what the non-participants do. In particular, the active participants in a transaction need not synchronize its execution with any non-participant (but they must synchronize, of course, with stationary participants, as changing the local state of any participant would disable the transaction).

Definition 2.1 (Local States, Configuration, Transaction, Active & Stationary Participants, Degree). Given agents $Q \subset \Pi$ and an arbitrary set S of *local states*, a *configuration* over Q and S is a member of $C := S^Q$. An *atomic transaction*, or just *transaction* over Q and S is any pair of configurations $t = c \rightarrow c' \in C^2$ s.t. $c \neq c'$, with $t_p := c_p \rightarrow c'_p$ for any $p \in Q$, and with p being an *active participant* in t if $c_p \neq c'_p$, *stationary participant* otherwise.

A stationary participant in a transaction is redundant if the active participants can ignore its state when performing the transaction. For example, if p and q can befriend regardless of r's state, then including r as a stationary participant would be redundant—it obscures the transaction's essential coordination requirements. Specifications of grassroots platforms using transactions should therefore be concise, namely harbour no redundant participants.

Definition 2.2 (Degree, Concise Set of Transactions). Given agents $P \subset \Pi$, local states S, and a set of transactions T, each $t \in T$ over some $Q \subseteq P$ and S, the *degree* of $t \in T$ (unary, binary,...k-ary) is the number of active participants in t, and the *degree* of T is the maximal degree of any $t \in T$. A stationary participant $q \in Q$ in a transaction $t \in$T over Q is *redundant* in t (given T) if for $\forall s \in S\ \exists t' \in T$ such that $t'_q = s \rightarrow s$ and $t_p = t'_p$ for every $p \neq q \in Q$. A set of transactions is *concise* if it has no transactions with redundant participants.

We now present concise specifications of three grassroots platforms using atomic transactions.

2.2 Grassroots Social Graphs via Befriending and Unfriending

Grassroots social graphs are the foundational infrastructure for grassroots platforms, including grassroots social networks which provide Twitter/X-like feeds and WhatsApp-like groups without subjecting people or their social graph to the control and exploitation. In centralised platforms (Twitter, Facebook, WhatsApp) the social graph is stored, controlled, and commercially exploited [43, 44] by the central authority. In proposed decentralised architectures [15], it resides on the globally-replicated blockchain under the control of miners/validators who are remunerated for their service.

In a grassroots social network, the social graph is stored in a distributed way under the control of the people themselves, with each person storing the local neighbourhood pertaining to them, and no third-party having access unless explicitly granted. The social graph evolves by people forming and breaking friendships, establishing authenticated peer-to-peer connections. The original definition [34] was via a unary multiagent transition system. Here both actions are specified—abstractly and concisely—as binary transactions.

Each agent maintains, as its local state, a set of friends; two agents p and q can atomically become friends, if they were not friends beforehand; and the two friends p and q can atomically cease to be friends. Communication functions of a social network can be added, under the restriction that communication occurs only among friends [34]. For example, feeds with followers, where friends follow each other, and if two friends follow a third person, then the first to obtain an item on the third person's feed disseminates it to the other. Formally, a liveness theorem can be proven for this design [34], stating that if a person p that follows a person q is connected to q via a chain of mutual friends, each of them correct and follows q, then p will eventually receive every item on q's feed. Practically, a true "network celebrity" employing this protocol may achieve efficient large-scale distribution of their feed via the friendship subgraph of their followers. The specification of the grassroots social graphs is the foundation for grassroots social networks with feeds, groups, messaging, explored elsewhere [34, 36].

Definition 2.3 (Grassroots Social Graphs Transactions). For agents $P \subset \Pi$ with local states $S := 2^P$, the social graphs transactions are:

(1) **Befriend**: A binary transaction $c \rightarrow c'$ with participants $\{p, q\} \subseteq P$ where $c'_p := c_p \cup \{q\}$ and $c'_q := c_q \cup \{p\}$, provided $q \notin c_p$ and $p \notin c_q$
(2) **Unfriend**: A binary transaction $c \rightarrow c'$ with participants $\{p, q\} \subseteq P$ where $c'_p := c_p \setminus \{q\}$ and $c'_q := c_q \setminus \{p\}$, provided $q \in c_p$ and $p \in c_q$

2.3 Grassroots Cryptocurrencies via Atomic Swaps

Grassroots Cryptocurrencies. Grassroots cryptocurrencies [23, 35] can provide a foundation for an equitable digital economy, where people price their goods and services in terms of their own personally-minted grassroots coins, and liquidity is achieved through the creation of mutual credit lines among persons (natural and legal) via the swap of personal coins. Grassroots coins issued by different persons form an integrated digital economy via the sole rule in the grassroots cryptocurrencies protocol:

Coin Redemption: p can redeem a q-coin it holds against a p-coin held by q, if there is one, else against any coin held by q, one-for-one. We note that the value of p-coins critically depends on p maintaining computational and economic integrity. In the specification of grassroots cryptocurrencies each agent maintains, as its local state, the set of coins they hold; each agent p may mint additional p-coins ($¢_p^k$ denotes k p-coins); and two agents p and q can atomically swap any coins they hold. The following fragment of the specification employs one unary transaction and one binary atomic transaction:

(1) **Mint**: $c'_p := c_p \cup ¢_p^k$, $k > 0$.
(2) **Swap**: $c'_p := (c_p \cup y) \setminus x$, $c'_q := (c_q \cup x) \setminus y$, provided $x \subseteq c_p$ and $y \subseteq c_q$.

The swap transaction can realize the key economic functions of grassroots cryptocurrencies:

(1) **Payment**: Paying q with q-coins, namely $x = ¢_q^j$, $j > 0$, and $y = \emptyset$ (also paying with other coins q accepts as payment, e.g. community-bank coins). A payment could be made for love or, more typically, expecting in exchange (or after having received) an 'off-chain' product or service from q. In a practical application y could include a payment receipt, a confirmed purchase order, etc.

(2) **Mutual Credit Lines**: Establish mutual credit lines via the swap of personal (self-minted) coins, namely $x = ¢_p^j$ and $y = ¢_q^k$ for some $j, k > 0$. Typically $j = k$, but mutual credit with a premium for one of the agents, namely with $j \neq k$, is also possible. A digital economy based on grassroots cryptocurrencies achieves liquidity through persons (natural and legal) establishing mutual credit lines among them.
(3) **Redemption**: The obligatory 1:1 swap of q-coins held by p against an equal number of coins held by q, namely $x = ¢_q^j$, $|y| = j$.

The original specification of grassroots cryptocurrencies was via unary multiagent transition system and hence was quite involved, and so was the proof of them being grassroots [35]. The definition in turn led to an implementation via a unary payment system [23]. Here we provide an alternative specification—abstract and concise—via binary transactions. We will later prove the specification to be grassroots.

Informally, we associate with each agent $p \in P$ a unique 'colour' and an infinite multiset of identical p-coloured coins. When applied to multisets of coins, $\cup$ and $\setminus$ denote multiset union and multiset difference, respectively. Formally:

Definition 2.4 (Coins). We assume an infinite multiset C of identical *coins* ¢. Given set of agents $P \subset \Pi$, we refer to $¢_p$, a p-indexed element of the indexed set C^P, as a p-coin, with $¢_p^k$ being a multiset of k p-coins. A *set of P-coins* is a member of 2^{C^P}, namely a multiset of P-indexed coins.

The transactions of grassroots cryptocurrencies include the unary minting of p-coins by p, and the atomic swap between p and q of a set of coins held by p in exchange for a set of coins held by q.

Definition 2.5 (Grassroots Cryptocurrencies Transactions). For agents $P \subset \Pi$ with local states $S := 2^{C^P}$, the grassroots cryptocurrencies transactions are:

(1) **Mint**: A unary transaction $c \to c'$ with participant $p \in P$ where $c'_p := c_p \cup ¢_p^k$ for $k > 0$
(2) **Swap**: A binary transaction $c \to c'$ with participants $\{p, q\} \subseteq P$ where $c'_p := (c_p \cup y) \setminus x$ and $c'_q := (c_q \cup x) \setminus y$, provided $x \subseteq c_p$ and $y \subseteq c_q$

The swap transaction can realize several different economic functions, including payments, opening of mutual credit lines, and coin redemption, as described above.

The present specification does not distinguish between voluntary swap transactions, namely payments and forming mutual credit, and obligatory swaps, namely coin redemption. This issue is further discussed below.

2.4 Grassroots Federations via Joining and Leaving

Grassroots Federations. Grassroots Federations [37] aim to address the democratic governance of large-scale decentralized digital communities, e.g., of the size of the EU, the US, existing social networks, and even humanity at large. A grassroots federations evolves via the grassroots formation and federation of digital communities, each governed by an assembly selected by sortition from its members. The approach assumes digital freedom of assembly—that people can freely form digital communities that can federate into ever-larger communities as well as spawn child communities, based on geography, relations, interests, or causes. Small communities (say < 100) are governed by their members. Larger communities—no matter how large—are each governed democratically by a small (say ≈ 100) assembly elected by sortition among its members.

A specification of grassroots federations via unary multiagent transition systems has yet to be attempted. It would require a grassroots consensus protocol by which members of two communities decide, atomically, to join or leave. Here, we define such atomic transactions abstractly: All members of a community carry its state locally; and all of them change its state atomically to realize joining or leaving another community. The details are shown below.

Background. Grassroots Federations [18, 37] is a process by which communities evolve and federate with other communities. Key steps include a community joining a federation, by the mutual consent of the two, and a community leaving a federation, by the unilateral decision of one of the two.

In the following $G = (V, E)$ is a federations graph with communities as nodes and directed edges indicating community relation, specifically $f \to v$ indicates that v is a child community of f, and G' is the graph resulting from applying any of the following transitions to G:

(1) **Federate** $v \in V$**:** Add a new node $f \notin V$ to V and the edge $f \to v$ to E.
(2) **Join** $f \to v$**:** Add $f \to v$ to E provided that $f \in V$ and G' is acyclic.
(3) **Leave** $f \to v$**:** If $f \to v \in E$ then remove $f \to v$ from E.

Challenges. The grassroots solution would be to realize each community as a digital social contract [9], which also requires a grassroots consensus protocol among the participants, namely one that is executed by the participants themselves [19], not by third-parties (miners, validators) operating a global platform. In contrast to the undirected grassroots social graphs, grassroots federations graphs are directed. As a federation is intended to offer a hierarchical democratic governance structure, its directed graph must be acyclic. To allow local transactions to evolve the federation without creating cycles, a partial order $\succ$ on communities can be devised, with conditioning the joining of g as a child of f on $f \succ g$. Such a partial order can be defined, for example, based on height in the graph, geographic containment, or topic generality. In a real deployment of grassroots federations we expect the partial order to be a superposition of geography, topics, and maybe other issues, so that the federation of dog lovers of a village can join both the general village federation and the regional federation of dog lovers, which in turn can join the general regional federation. The resulting federations structure, termed *laminar*, was investigated in [18]. With this constraint we can address the requirement that the federations graph always be acyclic.

A related technical issue that arises in a grassroots setting is to ensure, without central coordination, that each community has a globally-unique identifier. This is required so that a grassroots federations can scale indefinitely, integrating communities that have emerged independently of each other. To achieve that, any agent p may create one singleton community with itself as a member,

identified by p. A community with a unique identifier c may create any number of parent communities using the **Federate** transition, and endows the i^{th} parent community it creates with the unique identifier (c, i). Here, we employ community identifiers P^*, where each $c \in P^*$ consists of an agent $p \in P$ followed by a finite list of integers, and define a total (not partial) order $\succ$ on P^*, where $v \succ v'$ for $v, v' \in P^*$ if the list v is longer than the list v', with ties resolved lexicographically.

The fundamental difference between grassroots social graphs and grassroots federations is that the actors in the former are individual agents, whereas the actors in the latter are communities of agents. To address it, we provide an agent-based specification of grassroots federations: All members of a community hold an identical copy of its state, and a transaction between two communities, say g joining f, is realized by the corresponding k-ary atomic transaction among the members of the two communities, with k being the size of their union, in which the states of all members of both communities change atomically to reflect this federation-level transaction.

Naturally, this results in many aspects of the transaction being abstracted-away, including that:

(1) Joining requires mutual consent among the two communities, and leaving is unilateral by one community.
(2) Decisions on behalf of a community are taken by the assembly of a community.
(3) The decision process is constitutional and democratic.

Specification. A *federations graph* $G = (V, E)$ over P is a labelled directed acyclic graph with nodes labelled by community identifiers; we do not distinguish between nodes and their labels. The *initial federations graph* over P is $G_0(P) = (P, \emptyset)$. Let $\mathcal{G}(P)$ be the set of federations graphs over P.

Definition 2.6 (Community and Personal Subgraph). Given a federations graph $G = (V, E) \in \mathcal{G}(P)$, for a community $v \in V$ the *community subgraph* $G_v = (V_v, E_v)$ of v is the subgraph of G where V_v includes v and all nodes adjacent to v and E_v includes all edges incident with v. An agent $p \in P$ is a *member* of community $v \in G$ if there is a path in G from v to p, and the *personal subgraph* G_p of p is the union of the subgraphs of the communities p is a member of, namely $G_p := \bigcup_{v:p \in v} G_v$

Thus, G_p includes any community v that p is a member of and any edge incident with v.

Observation 1. *Let $G \in \mathcal{G}(P)$ for some set of agents $P \subset \Pi$. Then $G = \bigcup_{p \in P} G_p$.*

Proof of Observation 1. Let $G = (V, E)$ as above. We prove the two directions of equality:

(1) $\subseteq$: Let $v \in V$. Then some $p \in P$ is a member of v by construction. Hence $v \in V_p$. Let $f \to g \in E$, with some $p \in V_f$. Then by definition $p \in V_g$, and hence $f \to g \in E_p$.
(2) $\supseteq$ Let $p \in P$ and $v \in V_p$. Then p is a member of v in G, and therefore $v \in V$. Let $e \in E_p$. By construction, this can be the case only if $e \in E$.

□

Namely, a federations graph is fully-defined by its members' subgraphs. Hence, in the following grassroots federations transition system, the evolving federations graph G created during a run is stored in a distributed way, with each agent $p \in P$ maintaining G_p. The result is that each agent stores the state of all communities they are a member of; equivalently, the state of each community is stored by all its members. Formally:

Definition 2.7 (Federations Configuration, Valid). A *federations configuration* c over $P \subset \Pi$ has local states $\{G_p : G \in \mathcal{G}(P), p \in P\}$. The *initial federations* are $G0 := (V0, \emptyset) \in \mathcal{G}(P)$, where $V0$ has a p-labelled node for every $p \in P$, and the *initial federations configuration* $c0$ is defined by $c0_p := G0_p := (\{p\}, \emptyset)$. A federations configuration is *valid* if there is a graph $G \in \mathcal{G}(P)$ such that $c_p = G_p$ for every $p \in P$.

Note that the initial federations configuration is valid. The atomic transactions of grassroots federations ensure that when the state of a community v changes in a graph G through the addition or removal of an edge incident to v in G, this change is carried out atomically by all $p \in P_v$, each updating its local state G_p.

This formulation abstracts away the decision process. In practice, communities may employ assemblies selected by sortition [18, 37], which in turn may be responsible for storing the community social graph and employ a consensus protocol among them, rather than requiring storage and consensus from all members. Still, we use the direct-democracy model here to simplify the exposition.

Definition 2.8 (Grassroots Federations Transactions). For agents $P \subset \Pi$ with local states being federations subgraphs from $\mathcal{G}(P)$, the grassroots federations transactions for every $G = (V, E) \in \mathcal{G}(P)$ are:

(1) **Federate** v: c, c' are configurations over $Q := P_v$, $v \in V$, $f = (v, i+1)$, where i is the maximal index of a parent of $v \in G$, if any, $i = 0$ if none, and
$\forall p \in Q.(c_p = G_p \wedge c'_p := (V \cup \{f\}, E \cup \{f \to v\})_p)$.
(2) **Join** $f \to g$: c, c' are configurations over $Q := P_f \cup P_g$, $f, g \in V$, $f \succ g$, and
$\forall p \in Q.(c_p = G_p \wedge c'_p := (V, E \cup \{f \to g\})_p)$.
(3) **Leave** $f \to g$: c, c' are configurations over $Q := P_f$, $f, g \in V$, $f \to g \in E$, and $\forall p \in Q.(c_p = G_p \wedge c'_p := (V, E \setminus \{f \to g\})_p)$.

This completes the transactions-based specification of the three key grassroots platforms. It does not distinguish between transactions carried out by mutual consent (befriend, open a credit line, join), and transactions that are obligatory once requested (unfriend, redeem, leave). We discuss adding liveness requirements in Section 7.

3 Multiagent Transition Systems and Grassroots Protocols

In this section we define transition systems, multiagent transition systems, and grassroots protocols.

3.1 Transition systems

The following is a simplified variation, sufficient for the purpose of this work, on the foundations introduced in [32]. In the following, $a \neq b \in X$ is a shorthand for $a \neq b \wedge a \in X \wedge b \in X$.

Definition 3.1 (Transition system, Computation, Run). A *transition system* is a tuple $TS = (S, s0, T)$, where:

(1) S is an arbitrary non-empty set, referred to as the set of *states*.
(2) Some $s0 \in S$ is the designated the *initial state*.
(3) $T \subset S^2$ is a set of *transitions over* S, where each transition $t \in T$ is a pair (s, s') of non-identical states $s \neq s' \in S$, also written as $t = s \rightarrow s'$.

A *computation* of TS is a (nonempty, potentially infinite) sequence of states $r = s_1, s_2, \cdots$ such that for every two consecutive states $s_i, s_{i+1} \in r$, $s_i \rightarrow s_{i+1} \in T$. If $s_1 = s0$ then the computation is called a *run* of TS.

Given a computation $r = s_1, s_2, \ldots$, we use $r \subseteq T$ to mean $(s_i \rightarrow s_{i+1}) \in T$ for every $(s_i \rightarrow s_{i+1}) \in r$.

We note that the notion of transition systems originally employed to define grassroots systems [32, 33] included a liveness condition. The abstract atomic transactions employed here are volitional and have no associated liveness conditions; adding these is discussed in Section 7. Also, the original work considered faulty computations and fault-tolerant implementations. These could be re-introduced in follow-on work that considers live and fault-tolerance implementations of the specifications presented here.

3.2 Multiagent Transition Systems with Atomic Transactions

Definition 3.2 (Multiagent Transition System). Given agents $P \subset \Pi$ and an arbitrary set S of *local states* with a designated *initial local state* $s0 \in S$, a *multiagent transition system* over P and S is a transition system $TS = (C, c0, T)$ with *configurations* $C := S^P$, *initial configuration* $c0 := \{s0\}^P$, and *transitions* $T \subseteq C^2$ being a set of transactions over P and S, with the *degree* of TS being the degree of T.

Unary multiagent transition systems were introduced in [32] and were employed to define the notion of grassroots protocols [33] and to provide unary specifications for various grassroots platforms [23, 34, 35]. Here, we employ k-ary transition systems, for any $k \leq |P|$, in which several agents can change their state simultaneously.

However, rather than specifying a multiagent transition system over a set of agents P directly, we specify it via the concise atomic transactions (Definitions 2.1, 2.2) the transition system intends to realize, which are typically of bounded degree, or at least a degree smaller than P.

A transaction over $Q \subseteq P$ defines a set of multiagent transition over P in which all members of $P \setminus Q$ are stationary:

Definition 3.3 (Transaction Closure). Let $P \subset \Pi$, S a set of local states, and $C := S^P$. For a transaction $t = (c \rightarrow c')$ over local states S with participants $Q \subseteq P$, the *P-closure of t*, $t{\uparrow}P$, is the set of transitions over P and S defined by:

$$t{\uparrow}P := \{t' \in C^2 : \forall q \in Q.(t_q = t'_q) \wedge \forall p \in P \backslash Q.(p \text{ is stationary in } t')\}$$

If R is a set of transactions, each $t \in R$ over some $Q \subseteq P$ and S, then the *P-closure of R*, $R{\uparrow}P$, is the set of transitions over P and S defined by:

$$R{\uparrow}P := \bigcup_{t \in R} t{\uparrow}P$$

Namely, the closure over $P \supseteq Q$ of a transaction t over Q includes all transitions t' over P in which members of Q do the same in t and in t', and the rest remain in their current (arbitrary) state.

Note that while the set of multiagent transitions $R{\uparrow}P$ in general may be over a larger set of agents than R, T and R are of the same degree, by construction. Also, note that while all transitions of a multiagent transition system over P are also over P, each transaction in a set of transactions is typically over a different set of participants. Thus, a set of transactions R over S, each with participants $Q \subseteq P$, defines a multiagent transition system over S and P as follows:

Definition 3.4 (Transactions-Based Multiagent Transition System). Given agents $P \subset \Pi$, local states S with initial local state $s0 \in S$, and a set of transactions R, each $t \in R$ over some $Q \subseteq P$ and S, the *transactions-based multiagent transition system* over P, S, and R is the multiagent transition system $TS = (S^P, \{s0\}^P, R{\uparrow}P)$.

In other words, one can fully specify a multiagent transition system over S and P simply by providing a concise set of atomic transactions over S, each with participants $Q \subseteq P$. This is what we do next for grassroots social networks, grassroots cryptocurrencies, and grassroots federations.

4 Atomic Transactions-Based Grassroots Platforms and their Safety Properties

Having specified the transactions of grassroots platforms in Section 2 and introduced the multiagent transition system framework in Section 3, we now define the grassroots platforms as the multiagent transition systems induced by the sets of transactions presented above, and prove their safety properties.

4.1 Grassroots Social Graph

Definition 4.1 (Grassroots Social Graphs). Given agents $P \subset \Pi$, the *grassroots social graphs SG* is the transactions-based multiagent transition system over agents P and local states $S := 2^P$, with the befriend and unfriend transactions of Definition 2.3.

Lemma 4.2 (Friendship Safety). *Given a run r of SG, a configuration $c \in r$, and agents $p, q \in P$, then $p \in c_q$ iff $q \in c_p$.*

Proof. The proof is by induction on the length of the run $r = c0, c_1, \ldots, c_n$. Assume $|r| = 1$, namely r consists of the initial configuration $c0$. Then all local states are empty, satisfying the Lemma. Assume the lemma holds for runs of length $|r| = n$, that c_n satisfies the Lemma, and consider the transition $c_n \rightarrow c_{n+1}$. It can be either Befriend or Unfriend; both satisfy for every $p \in P$ the condition $p \in c_q$ iff $q \in c_p$ for c_{n+1} if c_n does. □

We note that each configuration c in a run r of SG induces a graph with agents as vertices and an edge $p \leftrightarrow q$ if $p \in c_q$ and $q \in c_p$, and that the graphs induced by two consecutive configurations in r differ by exactly one added or removed edge.

4.2 Grassroots Cryptocurrencies

Definition 4.3 (Grassroots Cryptocurrencies). Given agents $P \subset \Pi$, the *grassroots cryptocurrencies GC* is the transactions-based multiagent transition system over agents P and local states $S := 2^{C^P}$, with the Mint and Swap transactions of Definition 2.5.

Lemma 4.4 (Safety: Conservation of Money). *Given a run r of GC, a configuration $c \in r$ and an agent $p \in P$, the number of p-coins in c is the number of p-coins minted by p in the prefix of r ending in c.*

Proof. The proof is by induction on the length of the run $r = c0, c_1, \ldots, c_n$. Assume $|r| = 1$, namely r consists of the initial configuration $c0$. Then all local states are empty, satisfying the Lemma. Assume the lemma holds for runs of length $|r| = n$, that c_n satisfies the Lemma, and consider the transition $c_n \rightarrow c_{n+1}$. If the transition is Mint and the claim holds for c_n, then after minting it still holds by adding the minted coin. If it is Swap, it still holds as Swap does not add new coins to the configuration or remove coins from it. □

4.3 Grassroots Federations

Definition 4.5 (Grassroots Federations). Given agents $P \subset \Pi$, the *grassroots federations GF* is the transactions-based multiagent transition system over agents P and local states being federations subgraphs from $\mathcal{G}(P)$ (Definition 2.7), with the Federate, Join, and Leave transactions of Definition 2.8.

Lemma 4.6 (Grassroots Federations Safety). *In a run r of GF, any configuration $c \in r$ is valid.*

Proof. The proof is by induction on the length of the run. Initially, $G = G0$ and the initial configuration $c0$ satisfies $G0 = \bigcup_{p \in P} c0_p$. Assume that configuration c is valid and encodes $G = (V, E)$, and consider a transition $c \rightarrow c'$ with c' encoding G', namely $G' = \bigcup_{p \in P} c'_p$. We consider the various cases:

(1) **Federate** v: Each $p \in P_v$ changes their state to $c'_p = (V \cup \{f\}, E \cup \{f \rightarrow v\})_p)$.
(2) **Join** $f \rightarrow g$: Each $p \in (P_f \cup P_g)$ changes their state to $c'_p = (V, E \cup \{f \rightarrow g\})_p)$.
(3) **Leave** $f \rightarrow g$: Each $p \in P_f$ changes their state to $c'_p = (V, E \setminus \{f \rightarrow g\})_p$.

Examining these changes shows that they all preserve validity, in that also in the new configuration each agent p records, by construction, exactly the p-projection of the updated federations graph G'. □

5 Grassroots Protocols

Here we define what is a protocol; define when a protocol is grassroots; show how to define a protocol via a set of transactions; present conditions under which a protocol defined via transactions is grassroots; argue that the three protocols defined via transactions above satisfy these conditions; and conclude that these three protocols are grassroots.

5.1 Protocols and Grassroots Protocols

A protocol is a family of multiagent transition systems, one for each set of agents $P \subset \Pi$, which share an underlying set of local states $\mathcal{S}$ with a designated initial state $s0$. A *local-states function* $S : P \mapsto 2^{\mathcal{S}}$ maps every set of agents $P \subset \Pi$ to an arbitrary set of local states $S(P) \subset \mathcal{S}$ that includes $s0$ and satisfies $P \subset P' \subset \Pi \implies S(P) \subset S(P')$. Given a local-states function S, we use C to denote configurations over S, with $C(P) := S(P)^P$ and $c0(P) := \{s0\}^P$. Next, we define the notions of a protocol and a grassroots protocol.

Definition 5.1 (Protocol). A *protocol* $\mathcal{F}$ over a local-states function S is a family of multiagent transition systems that has exactly one transition system $\mathcal{F}(P) = (C(P), c0(P), T(P))$ for every $P \subset \Pi$.

Informally, in a grassroots protocol a set of agents P, if embedded within a larger set P', can still behave as if it is on its own, but has additional behaviours at its disposal due to possible interactions with members of $P' \setminus P$. To define the notion formally we employ the notion of projection.

Definition 5.2 (Projection). Let $\emptyset \subset P \subset P' \subset \Pi$. If c' is a configuration over P' then c'/P, the *projection of c' over P*, is the configuration c over P defined by $c_p := c'_p$ for every $p \in P$.

Note that in the definition above, c_p, the state of p in c, is in $S(P')$, not in $S(P)$, and hence may include elements "alien" to P, e.g., friendship with or a coin of $q \in P' \setminus P$.

We use the notions of projection and closure (Definition 3.3) to define the key notion of this paper, a grassroots protocol:[1]

Definition 5.3 (Oblivious, Interactive, Grassroots). A protocol $\mathcal{F}$ is:

(1) *oblivious* if for every $\emptyset \subset P \subset P' \subseteq \Pi$, $T(P)\uparrow P' \subseteq T(P')$
(2) *interactive* if for every $\emptyset \subset P \subset P' \subseteq \Pi$ and every configuration $c \in C(P')$ such that $c/P \in C(P)$, there is a computation $c \xrightarrow{*} c'$ of $\mathcal{F}(P')$ for which $c'/P \notin C(P)$.
(3) *grassroots* if it is oblivious and interactive.

Oblivious. Being oblivious implies that if a run of $\mathcal{F}(P')$ reaches some configuration c', then anything P could do on their own in the configuration c'/P (with a transition from $T(P)$), they can still do in the larger configuration c' (with a transition from $T(P')$), effectively being oblivious to members of $P' \setminus P$. The reason is that if $t = (c'/P \rightarrow d) \in T(P)$, then by the definition of closure, $t' = (c' \rightarrow d') \in t\uparrow P' \subseteq T(P')$, where $d'_p = d_p$ if $p \in P$ else $d'_p = c'_p$. Inductively, this implies that if agents P employ only transitions in $T(P)$ from the start, with their local states remaining in $S(P)$, they could continue doing so indefinitely, effectively ignoring any members in $P' \setminus P$. (Proving the corresponding claim within the original definition [32] was more difficult, and required the notion of interleaving of computations of two multiagent transition systems.)

We note that any protocol that uses a global data structure, whether replicated (Blockchain) or distributed (DHT [31], IPFS [4]), is not oblivious as members of P cannot ignore changes to the global data structure made by members of $P' \setminus P$; and hence is not grassroots.

Interactive. A slight complication in the definition of interactive is the use of a finite computation $\xrightarrow{*}$ rather than just a single transition $\rightarrow$. It is required as in some grassroots platforms agents need to make some preparatory steps before they may interact with each other. For example, in grassroots cryptocurrencies agents need first to mint some coins before they can swap them.

Being interactive is a weak liveness requirement. Informally, a standard liveness requirement has the form "something good must eventually happen". Here, the requirement is "it must be the case that something good (P interacting with non-P) may eventually

[1] We change earlier terminology and use the more precise "interactive" instead of the bland "interactive" [33].

happen". Namely, no matter what members of P do, if they run within a larger set of agents it is always the case that they can eventually interact with non-P's. Moreover, being interactive not only requires that P can always eventually interact with $P' \setminus P$, but that they do so in a way that leaves "alien traces" in the local states of P, so that the resulting configuration c'/P could not have been produced by P running on their own. For example, forming friendships with agents outside of P results in their non-P names entering c/P; receiving coins from, agents outside of P results in their non-P coins entering c/P. We note that while federated systems [1, 30] are oblivious, as one server may choose to ignore all other servers and just serve its clients, they are not interactive, as a set of clients P without a server cannot do more if embedded within a larger set of clients P', still without a server.

5.2 Transactions-Based Grassroots Protocols

The original grassroots paper [33] provided sufficient conditions for a protocol to be grassroots—asynchrony, interference-freedom, and interactivity. They should still hold under the new definition, with some adaptation, mostly simplification (and with interactivity renamed interactivity). However, here we are interested in transactions-based transition systems, therefore we will follow a different route: We first show how a local-states function can be used to define a set of transactions, and then show how a set of such transactions can be used to define a protocol, termed transactions-based protocol. We then prove that a single condition on such transactions—interactivity—is sufficient for a transactions-based protocol to be grassroots.

Definition 5.4 (Transactions Over a Local-State Function). Let S be a local-states function. A set of transactions R is *over* S if every transaction $t \in R$ is a multiagent transition over Q and $S(Q')$ for some $Q \subseteq Q' \subset \Pi$. Given such a set R and $P \subset \Pi$, $R(P) := \{t \in R : t \text{ is over } Q \text{ and } S(Q'), Q \subseteq Q' \subseteq P\}$.

Definition 5.5 (Transactions-Based Protocol). Let S be a local-states function and R a set of transactions over S. Then a *protocol* $\mathcal{F}$ *over* R *and* S includes for each set of agents $P \subset \Pi$ the transactions-based multiagent transition system $\mathcal{F}(P)$ over P, $S(P)$, and $R(P)$, $\mathcal{F}(P) := (S(P)^P, \{s0\}^P, R(P)\uparrow P)$.

Next we aim to find conditions under which a transactions-based protocol is grassroots. In the original paper [33], fulfilling three conditions were deemed sufficient: Asynchrony, interference-freedom, and interactivity. Intuitively speaking, transactions-based multiagent transition systems are asynchronous by construction, as the essence of a transaction is that it can be taken no matter what the states of non-participants are. They are also non-interfering for the same reason. The following Proposition captures this:

Proposition 5.6. *A transactions-based protocol is oblivious.*

To prove it, we need the following Lemma:

Lemma 5.7 (Closure Transitivity). *Let $\emptyset \subset P \subset P' \subset \Pi$ and R a set of transactions. Then*

$$(R(P)\uparrow P)\uparrow P' = R(P)\uparrow P'.$$

Proof. We argue both directions of the equality:

(1) $\subseteq$: Let $t \in R(P)$ and consider any $t' \in (t\uparrow P)\uparrow P'$. By definition, $t_p = t'_p$ if $p \in P$, p is stationary in t' if $p \in P' \setminus P$, which, by the definition of closure, $t' \in t\uparrow P'$, namely $t' \in R(P)\uparrow P'$.
(2) $\supseteq$: Let $t' \in R(P)\uparrow P'$. Then there is a transaction $\hat{t} \in R(P)$ over some $Q \subseteq P$, for which $t' \in \hat{t}\uparrow P'$. By construction, $\hat{t}_p = t'_p$ if $p \in Q$ and p is stationary in t' if $p \in P' \setminus Q$. By definition of closure, $t = \hat{t}\uparrow P$ satisfies $t' \in t\uparrow P'$, thus $t' \in (\hat{t}\uparrow P)\uparrow P'$, namely $t' \in (R(P)\uparrow P)\uparrow P'$.

□

We can now prove the Proposition:

Proof of Proposition 5.6. Let $\mathcal{F}$ be a protocol over the state function S, R a set of transactions over S, and $\emptyset \subset P \subset P' \subseteq \Pi$. We have to show that $T(P)\uparrow P' \subseteq T(P')$. By definition $T(P) = R(P)\uparrow P$ and $T(P') = R(P')\uparrow P'$, thus:

$$T(P)\uparrow P' = (R(P)\uparrow P)\uparrow P' = R(P)\uparrow P' \subseteq R(P')\uparrow P' = T(P').$$

by definition of $T(P)$, Lemma 5.7, and since $P \subset P'$ and therefore $R(P) \subseteq R(P')$. □

The remaining condition is being interactive, which we aim to capture as follows:

Definition 5.8 (Interactive Transactions). A set of transactions R over a local-states function S is *interactive* if for every $\emptyset \subset P \subset P' \subset \Pi$ and every configuration $c \in C(P')$ such that $c/P \in C(P)$, there is a computation $(c \xrightarrow{*} c') \subseteq R(P')\uparrow P'$ for which $c'/P \notin C(P)$.

In other words, with an interactive set of transactions, any group of agents that have been so far self-contained will always have a computation with non-members that will take the group outside of its "comfort zone", resulting in members of the group having a local state with "alien traces" that could have been produced only by interacting with non-members.

Proposition 5.9. *A protocol over an interactive set of transactions is interactive.*

Proof. Let S be a local-states function, R an interactive set of transactions over S, and $\mathcal{F}$ a transactions-based protocol over R and S, $\emptyset \subset P \subset P' \subseteq \Pi$, and $c \in C(P')$ a configuration such that $c/P \in C(P)$. By R being interactive (Definition 5.8), there is a computation $c \xrightarrow{*} c' \subseteq R(P')\uparrow P'$ for which $c'/P \notin C(P)$. By the definition of $\mathcal{F}$ as a transactions-based protocol over R and S, $T(P') = R(P')\uparrow P'$, hence this computation is of $T(P')$, establishing that $\mathcal{F}$ is interactive. □

Our main result follows from the definitions and results above:

Theorem 5.10. *A protocol over an interactive set of transactions is grassroots.*

Proof. Let $\mathcal{F}$ be a protocol over a set of transactions R (Definition 5.5), where R is interactive (Definition 5.8). Since $\mathcal{F}$ is a transactions-based protocol then, according to Proposition 5.6, $\mathcal{F}$ is oblivious. And since $\mathcal{F}$ is over an interactive set of transactions then, according to Proposition 5.9, $\mathcal{F}$ is interactive. Therefore, by Definition 5.3, $\mathcal{F}$ is grassroots. □

5.3 The Three Platforms are Grassroots as Specified

We argue briefly that the transactions-based specification of our three grassroots platforms of interest—social graphs, cryptocurrencies, and democratic federations—are all interactive, and therefore according to Theorem 5.10 the protocols that they specify are all grassroots.

Corollary 5.11. *Grassroots Social Graphs are grassroots.*

Proof. Consider $P \subset P'$ and a configuration $c \in C(P')$ such that $c/P \in C(P)$. This means that all friendships of members of P in c are with other members in P. Hence the transaction in $R(P')$ in which a member of P establishes friendship with a member of $P' \setminus P$ satisfies the definition of interactivity. □

Corollary 5.12. *Grassroots Cryptocurrencies are grassroots.*

Proof. Consider $P \subset P'$ and a configuration $c \in C(P')$ such that $c/P \in C(P)$. This means that all coins held by members of P in c are P-coins. Hence the transaction in $R(P')$ in which a member of P swaps coins with a member of $P' \setminus P$ (possibly preceded by transactions in which the two members mint coins, if they have not done so already) satisfies the definition of interactivity. □

Corollary 5.13. *Grassroots Federations are grassroots.*

Proof. Consider $P \subset P'$ and a configuration $c \in C(P')$ such that $c/P \in C(P)$. This means that the subgraph of G projected by members of P is a connected component of G. Hence the Join transaction in $R(P')$ in which a member of P forms an edge with a member of $P' \setminus P$, the direction of which depends on the order $\succ$ of their identifiers, satisfies the definition of interactivity. □

6 Implementation

The specifications presented here provide the formal foundation for grassroots platforms. Recent work [36] explores implementing such platforms through Grassroots Logic Programs (GLP), a secure, multiagent, concurrent logic programming language. GLP provides binary communication primitives through paired reader/writer variables, where k-ary atomic transactions can be realized via consensus protocols such as Constitutional Consensus [19]. The notion of implementations among transition systems, as well as their fault-tolerance, has been studied extensively [2, 21, 26, 27, 32, 41]. Here, we discuss this notion briefly and informally, and plan follow-on work to do so formally.

Binary transactions. A standard way to realize binary transactions using unary transition systems is for one agent, say p, to OFFER the transaction to q, who may respond with ACCEPT, upon which p may respond with COMMIT, upon which the offered transaction is deemed to have been executed, or ABORT. Agent p may also issue ABORT before or after receiving any response from q to its offer, provided p has not previously issued COMMIT.

A challenge in this implementation is that a faulty p may fail to either COMMIT or ABORT following an ACCEPT by q, leaving q in limbo, at least in regards to this transaction. Solutions to this are a subject of future work.

For now, we note that, worst case, a friendship offer by p accepted by q, or an offer by community p to join community q, would remain in limbo. If it is committed by p at some later point, which is not convenient to q, then q can promptly unfriend p or remove p, as the case may be, with little or no harm done. In case of grassroots cryptocurrencies, a swap transaction in limbo may tie coins offered by q, which may or may not be harmful to q (not harmful if these are q-coins, which q may mint as it pleases; or p-coins that q tries to redeem, and if p is non-responsive it might indicate that p-coins are not worth much anyhow).

k-ary transactions. The k-ary transactions of grassroots federations require consensus among community members with support for reconfiguration as membership changes. Constitutional Consensus [19] provides such participant-controlled reconfiguration through constitutional amendments. More generally, grassroots platforms entail multiple dynamically-changing, partially-overlapping sets of agents engaged in running partially-dependent consensus protocols.

7 Related and Future Work

Atomic transactions have been investigated early in distributed computing, mostly in the context of database systems [22, 24, 25]. Most research since and until today focuses on their efficient and robust implementation [6, 10]. The integration of atomic transactions in programming languages has also been explored [5]. In terms of formal models of concurrency, the extension of CCS with atomic transactions has been investigated in the past [3, 13, 14], but without follow-on research, so it seems. While transition systems have been the bedrock of abstract models of computation since the Turing machine, we are not aware of previous attempts to explore atomic transactions within their context.

Previous work on formal implementations of grassroots platforms employed unary transition systems [23, 32, 34, 35]. While this new definition of a grassroots protocol tries to capture the same intuition as the original one [33], the new definition is crisper. It is also more restrictive in two senses, and more lax in a third:

(1) It is specified in terms of configurations and transitions not runs, so its restriction applies to all configurations, not only those reachable via a run.
(2) A technical limitation of comparing sets of runs, as done in the original definition, is that doing so does not capture the internal/hidden nondeterminism of intermediate configurations. So, according to the original definition, the smaller group P may have a run that leads to a configuration with multiple choices, while the larger group P' has a set of runs, each leading separately to only one of these choices of P, but no run of P' leads to a configuration in which the members of P have the same choices they would have if they were to run on their own. The current definition in terms of configurations and transitions, rather than sets of runs, eliminates this deficiency.
(3) On the other hand, the original definition in terms of runs also addressed liveness. Within the original definition, an all-to-all dissemination protocol $\mathcal{F}$ in which $\mathcal{F}(P)$ satisfies the liveness requirement that every message sent by an agent in P eventually reaches every agent in P, is not grassroots. The reason is that a run of P, which is live when member of P run on their own, being oblivious of members outside P, would

not be live within the context of P', as it would not provide dissemination between members of P and members of $P' \setminus P$, indefinitely. We consider this limitation of the new definition, as it relates to all-to-all dissemination, hypothetical, as it seems that such a protocol could not be realized without global directory (e.g., a DHT) for agents to find each other, which would not be oblivious, and hence not grassroots also under the new definition.

While liveness is well-understood in the context of unary transition systems, it is opaque in the more abstract case of atomic transactions, as it may not be possible to identify the culprit in case an enabled transaction is never taken. On the other hand, a unary implementation of atomic transactions clearly has liveness requirements. Hence, we opted not to incorporate liveness in the current context of atomic transactions, and leave it as a subject of future research.

The direction we envision is viewing every agent as a pair: a nondeterministic *user* and a deterministic *app*, with a *user interface* providing bidirectional communication transactions between the user and the app [36]. As the nondeterministic user models a volitional person, liveness requirements in such a system may take the form "a message sent by a person p to their friend q is eventually received by q" [34], or "a swap request from p to q that includes q-coins is eventually responded to by q" [23].

Acknowledgments

I thank Idit Keidar, Andy Lewis, and Nimrod Talmon for our discussions and their feedback on this and related topics. Useful discussions with Claude and Chat-GPT are acknowledged.

References

[1] [n. d.].
[2] Martin Abadi and Leslie Lamport. 1993. Composing specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 15, 1 (1993), 73–132.
[3] Lucia Acciai, Michele Boreale, and Silvano Dal Zilio. 2007. A concurrent calculus with atomic transactions. In *European Symposium on Programming*. Springer, 48–63.
[4] Juan Benet. 2014. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561* (2014).
[5] Johannes Borgström, Karthikeyan Bhargavan, and Andrew D Gordon. 2009. A compositional theory for STM Haskell. In *Proceedings of the 2nd ACM SIGPLAN Symposium on Haskell*. 69–80.
[6] Manuel Bravo and Alexey Gotsman. 2019. Reconfigurable atomic transaction commit. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*. 399–408.
[7] Sonja Buchegger, Doris Schiöberg, Le-Hung Vu, and Anwitaman Datta. 2009. PeerSoN: P2P social networking: early experiences and insights. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*. 46–52.
[8] Vitalik Buterin. 2018. Governance, Part 2: Plutocracy Is Still Bad. Available at https://vitalik.eth.limo/general/2018/03/28/plutocracy.html.
[9] Luca Cardelli, Liav Orgad, Gal Shahaf, Ehud Shapiro, and Nimrod Talmon. 2020. Digital social contracts: A foundation for an egalitarian and just digital society. In *CEUR Proceedings of the First International Forum on Digital and Democracy*, Vol. 2781. CEUR-WS, 51–60.
[10] Gregory Chockler and Alexey Gotsman. 2021. Multi-shot distributed transaction commit. *Distributed Computing* 34 (2021), 301–318.
[11] Gregory Chockler, Roie Melamed, Yoav Tock, and Roman Vitenberg. 2007. Constructing scalable overlays for pub-sub with many topics. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*. 109–118.
[12] Gregory Chockler, Roie Melamed, Yoav Tock, and Roman Vitenberg. 2007. Spidercast: a scalable interest-aware overlay for topic-based pub/sub communication. In *Proceedings of the 2007 inaugural international conference on Distributed event-based systems*. 14–25.
[13] Edsko de Vries, Vasileios Koutavas, and Matthew Hennessy. 2010. Communicating transactions. In *International Conference on Concurrency Theory*. Springer, 569–583.
[14] Edsko De Vries, Vasileios Koutavas, and Matthew Hennessy. 2010. Liveness of communicating transactions. In *Asian Symposium on Programming Languages and Systems*. Springer, 392–407.
[15] DSNP. 2022, dsnp.org. Decentralized Social Networking Protocol.
[16] Ethereum. 2021, https://ethereum.org/en/dao. Decentralized autonomous organizations (DAOs) | ethereum.org. https://ethereum.org/en/dao/
[17] Youssef Faqir-Rhazoui, Javier Arroyo, and Samer Hassan. 2021. A comparative analysis of the platforms for decentralized autonomous organizations in the Ethereum blockchain. *Journal of Internet Services and Applications* 12 (2021), 1–20.
[18] Daniel Halpern, Ariel D Procaccia, Ehud Shapiro, and Nimrod Talmon. 2024. Federated Assemblies.
[19] Idit Keidar, Andrew Lewis-Pye, and Ehud Shapiro. 2025. Constitutional Consensus.
[20] Anne-Marie Kermarrec, Erick Lavoie, and Christian Tschudin. 2020. Gossiping with append-only logs in secure-Scuttlebutt. In *Proceedings of the 1st international workshop on distributed infrastructure for common good*. 19–24.
[21] Leslie Lamport. 1999. Specifying Concurrent Systems with TLA$^+$. *NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES* 173 (1999), 183–250.
[22] Butler W Lampson. 1981. Chapter 11. atomic transactions. In *Distributed Systems—Architecture and Implementation: an Advanced Course*. Springer, 246–265.
[23] Andrew Lewis-Pye, Oded Naor, and Ehud Shapiro. 2023. Grassroots Flash: A Payment System for Grassroots Cryptocurrencies. *arXiv preprint arXiv:2309.13191* (2023).
[24] Nancy Lynch, Michael Merritt, William Weihl, and Alan Fekete. 1988. A theory of atomic transactions. In *ICDT'88: 2nd International Conference on Database Theory Bruges, Belgium, August 31–September 2, 1988 Proceedings 2*. Springer, 41–71.
[25] Nancy A Lynch and Michael Merritt. 1993. *Atomic transactions: in concurrent and distributed systems*. Morgan Kaufmann Publishers Inc.
[26] Panagiotis Manolios. 2003. A compositional theory of refinement for branching time. In *Advanced Research Working Conference on Correct Hardware Design and Verification Methods*. Springer, 304–318.
[27] P Blauth Menezes, J Félix Costa, and Amílcar Sernadas. 1995. Refinement mapping for general (discrete event) systems theory. In *International Conference on Computer Aided Systems Theory*. Springer, 103–116.
[28] Satoshi Nakamoto and A Bitcoin. 2008. A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf* 4 (2008).
[29] Satoshi Nakamoto and A Bitcoin. 2008. A peer-to-peer electronic cash system.
[30] Aravindh Raman, Sagar Joglekar, Emiliano De Cristofaro, Nishanth Sastry, and Gareth Tyson. 2019. Challenges in the decentralised web: The mastodon case. In *Proceedings of the internet measurement conference*. 217–229.
[31] Sean Rhea, Brighten Godfrey, Brad Karp, John Kubiatowicz, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Harlan Yu. 2005. OpenDHT: a public DHT service and its uses. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*. 73–84.
[32] Ehud Shapiro. 2021. Multiagent Transition Systems: Protocol-Stack Mathematics for Distributed Computing. *arXiv preprint arXiv:2112.13650* (2021).
[33] Ehud Shapiro. 2023. Grassroots Distributed Systems: Concept, Examples, Implementation and Applications (Brief Announcement).
[34] Ehud Shapiro. 2023. Grassroots Social Networking: Serverless, Permissionless Protocols for Twitter/LinkedIn/WhatsApp. In *OASIS '23* (Rome, Italy). Association for Computing Machinery. doi:10.1145/3599696.3612898
[35] Ehud Shapiro. 2024. Grassroots Currencies: Foundations for Grassroots Digital Economies. *arXiv preprint arXiv:2202.05619* (2024).
[36] Ehud Shapiro. 2025. Grassroots Logic Programs: A Secure, Multiagent, Concurrent, Logic Programming Language. arXiv:2510.15747 [cs.PL] https://arxiv.org/abs/2510.15747
[37] Ehud Shapiro and Nimrod Talmon. 2025. Grassroots Federation: Fair Governance of Large-Scale, Decentralized, Sovereign Digital Communities. *arXiv preprint arXiv:2505.02208* (2025).
[38] Dominic Tarr, Erick Lavoie, Aljoscha Meyer, and Christian Tschudin. 2019. Secure scuttlebutt: An identity-centric protocol for subjective and decentralized applications. In *Proceedings of the 6th ACM conference on information-centric networking*. 1–11.
[39] Shuai Wang, Yong Yuan, Xiao Wang, Juanjuan Li, Rui Qin, and Fei-Yue Wang. 2018. An overview of smart contract: architecture, applications, and future trends. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 108–113.
[40] Zeli Wang, Hai Jin, Weiqi Dai, Kim-Kwang Raymond Choo, and Deqing Zou. 2021. Ethereum smart contract security research: survey and future research opportunities. *Frontiers of Computer Science* 15, 2 (2021), 1–18.
[41] James R Wilcox, Doug Woos, Pavel Panchekha, Zachary Tatlock, Xi Wang, Michael D Ernst, and Thomas Anderson. 2015. Verdi: a framework for implementing and formally verifying distributed systems. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*.

357–368.
[42] Gavin Wood et al. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151, 2014 (2014), 1–32.
[43] Shoshana Zuboff. 2019. *The age of surveillance capitalism: The fight for a human future at the new frontier of power.* Public Affairs, US.
[44] Shoshana Zuboff. 2022. Surveillance capitalism or democracy? The death match of institutional orders and the politics of knowledge in our information civilization. *Organization Theory* 3, 3 (2022), 26317877221129290.