Unified Opinion Dynamic Modeling as Concurrent Set Relations in Rewriting Logic

Carlos Olarte[1], Carlos Ramírez[2], Camilo Rocha[2], and Frank Valencia[3,2]

[1] LIPN, CNRS UMR 7030, Université Sorbonne Paris Nord, Villetaneuse, France
[2] Department of Electronics and Computer Science, Pontificia Universidad Javeriana, Cali, Colombia
[3] CNRS-LIX, École Polytechnique de Paris, France

Abstract. Social media platforms have played a key role in weaponizing the polarization of social, political, and democratic processes. This is, mainly, because they are a medium for opinion formation. Opinion dynamic models are a tool for understanding the role of specific social factors on the acceptance/rejection of opinions because they can be used to analyze certain assumptions on human behaviors. This work presents a framework that uses concurrent set relations as the formal basis to specify, simulate, and analyze social interaction systems with dynamic opinion models. Standard models for social learning are obtained as particular instances of the proposed framework. It has been implemented in the Maude system as a fully executable rewrite theory that can be used to better understand how opinions of a system of agents can be shaped. This paper also reports an initial exploration in Maude on the use of reachability analysis, probabilistic simulation, and statistical model checking of important properties related to opinion dynamic models.

Keywords: Concurrent set relations · opinion dynamic models · social interaction systems · belief revision · rewriting logic · formal verification

1 Introduction

Social media platforms have played a key role in the polarization of social, political, and democratic processes. Social uprisings in the Middle East, Asia, and Central and South America have led to sudden changes in the structure and nature of society during this past decade [20,12,6,15,30,19]. Polarization across the globe has paved the way to the divergence of political attitudes away from the center, towards ideological extremes, sometimes resulting in fractured institutions, erratic policy making, incipient political dialog, and the resurgence of old regimes [13,18,22,20,11,16]. Democracy, viewed as a system of power controlled by the people, has been made vulnerable by severe polarization as opposing sides are seen as adversaries that compete against an enemy needing to be vanquished. As a result, popular election campaigns –including presidential ones– have compromised the basic principles of democratic election in some countries [4,29,28,3]. All these scenarios have a common factor: social media interaction as a medium for *opinion formation* fueling polarization.

Social learning and opinion dynamic models have been developed to understand the role of specific social factors on the acceptance/rejection of opinions, such as the ones communicated via social media (see, e.g., [14,2,17,8]). They are often used to validate how certain assumptions on human behaviors can explain alternative scenarios, such as opinion consensus, polarization and fragmentation. In their micro-level approach, the one followed in the present work, users are considered as agents that can share opinions on a given topic. They update their opinion by interacting with a selected group of users that have some influence on them (e.g., influencers, their family and friends). These dynamics take place at discrete time steps at which (some) agents update their opinion. For instance, an opinion model can deterministically update the opinion of all agents in such a time-step, while another one can non-deterministically update the opinion of a single agent. Depending on the model of choice, which usually defines its own update function for the individual agents, phenomena under different assumptions can be observed. The ultimate goal is to understand how the opinions of the agents, as a social system, are shaped after a certain number of steps.

This work proposes a framework that uses concurrent set relations as the formal basis to specify, simulate, and analyze social interaction systems with dynamic opinion models. The framework uses *influence graphs* to specify the structure of agent interactions in the social system under study: vertices represent agents and a directed weighted edge from a to b represents the weighted influence of agent's a opinion over the opinion of agent b. In the sense of set relations in [25], the framework comprises two main mechanisms that are combined via closures for specifying opinion dynamics over the graphs: namely, an atomic set relation and a strategy. The *atomic set relation* updates the opinion of a single vertex w.r.t. a set of edges (and the corresponding vertices) incident to it. The *strategy* selects the edges that will be used to update in parallel (i.e., synchronously) the opinion associated to the vertices with edges incident to it in the given set. As a consequence, dynamic opinion models can be formalized as a concurrent set relation system, with parametric update function, using the composition of an atomic relation and a strategy via closures. An important observation is that the determinism or non-determinism inherent to a given opinion dynamic model is exactly captured by the deterministic or non-deterministic nature of the corresponding concurrent set relation.

Standard models for social learning are obtained as particular instances of the proposed framework. The classical DeGroot opinion model [9] is obtained as the synchronous closure under the maximal redices strategy of a given atomic set relation. In a similar fashion, gossip-based models that use pairwise interactions to represent the opinion formation process (see, e.g., [10]) are obtained via the asynchronous closure where the strategy selects single edges for the given atomic set relation. Other opinion models can be obtained via the synchronous closure of an atomic set relation, as midpoints between De Groot and gossip-based models.

The proposed framework has been implemented in the Maude system [7]. It is a rewriting logic theory that exploits the reflective capabilities of rewriting logic and that can be particularized to the opinion model of interest. A state is an object-like configuration representing the structure of the system and its opinion values. An object is either an agent u with its opinion o_u, specified as $\langle u : o_u \rangle$, or the influence of agent u over agent v with weight i_{uv}, specified as $\langle (u,v) : i_{uv} \rangle$. The update function μ of each specific model is to be defined equationally. The implementation of both the atomic set relation and the strategy is inspired by the ideas in [24]. The atomic set relation is axiomatized as a (non-executable) rewrite rule that takes as input an agent $\langle u : o_u \rangle$ and a set of edges $A \subseteq E$ in the current state. For a given state, it updates the opinion o_u to a new opinion o'_u using μ, and the opinion and influence of agents adjacent to it w.r.t. A. As a result, each atomic step rewrites a single object $\langle u : o_u \rangle$ to its updated version $\langle u : o'_u \rangle$. The metalevel is used to apply the atomic rewrite rule over the agents in a state according to the edges selected by the given strategy: only agents appearing as targets of the directed edges have their opinion updated. This strategy is defined equationally by the user and computes a collection of subsets of E: a parallel rewrite step under the maximal redices strategy is performed for each subset A of edges. Since the atomic rewrite relation is deterministic, the strategy is the only source of non-determinism in the system and a concurrent step is made for each identified subset A.

The implementation of the proposed framework results in a fully executable top-most object-like rewrite theory in Maude that can be used to better understand how opinions of a system of agents are shaped –and to ultimately understand polarization— using formal methods techniques, such as reachability analysis and temporal model checking.

This work is part of a broader effort to make available computational ideas and approaches for analyzing phenomena in social networks, such as polarization, consensus, and fragmentation. They include concurrency models, modal and probabilistic logics, and formal methods frameworks, techniques, and tools. In this context, the work presented here is a first step towards the use of rewriting logic for such purposes. As it is explained in the sections that follow, one major problem a opinion dynamic model may face is that of state explosion. An initial exploration on the use of probabilistic simulation and statistical analysis is reported in this work. However, the extension of the proposed framework to a fully probabilistic setting, in which –e.g.– the strategy selects the set of edges according to a probability distribution function, falls outside the scope of this work. It needs to be further explored as future work as it may open the door to statistical model checking of novel properties using a new breed of measures and thus pave the way to the analysis of quantitative properties beyond the reach of techniques currently available for opinion dynamic models.

Organization. After recalling the notion of set relations in Section 2, Section 3 shows how different models for social learning can be seen as particular instances (atomic set relation and strategy) of this framework. The implementation in Maude is described in Section 4, while different analyses performed on the proposed rewrite theory are introduced in Section 5. Section 6 concludes the paper. The full Maude specification supporting the set relations framework is available at [23], as companion tool to the paper.

2 Set Relations

This section introduces set relations and their notation, as used in this paper. It defines the asynchronous, parallel, and synchronous set relations as closures of an atomic set relation. This section is based, mainly, on [25].

Let $\mathcal{U}$ be a set whose elements are denoted $A, B, \dots$ and let $\rightarrow$ be a binary relation on $\mathcal{U}$. An element A of $\mathcal{U}$ is called a $\rightarrow$*-redex* iff there exists $B \in \mathcal{U}$ such that the *pair* $\langle A; B\rangle \in \rightarrow$. The expressions $A \rightarrow B$ and $A \not\rightarrow B$ denote $\langle A; B\rangle \in \rightarrow$ and $\langle A; B\rangle \notin \rightarrow$, respectively. The *identity* and *reflexive-transitive* closures of $\rightarrow$ are defined as usual and denoted $\xrightarrow{0}$ and $\xrightarrow{*}$, respectively.

It is assumed that $\mathcal{U}$ is the family of all *nonempty* finite subsets of an abstract and possibly infinite set T whose members are called *elements* (i.e., $\mathcal{U} \subseteq \mathcal{P}(T)$, $\emptyset \notin \mathcal{U}$, and if $A \in \mathcal{U}$, then $\mathrm{card}(A) \in \mathbb{N}$). Therefore, $\rightarrow$ is a binary relation on finite subsets of elements in T. When it is clear from the context, curly brackets are omitted from set notation; e.g., $a, b \rightarrow b$ denotes $\{a, b\} \rightarrow \{b\}$. Because this convention, the symbol ‘,’ is overloaded to denote set union. For example, if A denotes the set $\{a, b\}$, B the set $\{c, d\}$, and D the set $\{d, e\}$, the expression $A, B \rightarrow B, D$ denotes the pair $a, b, c, d \rightarrow c, d, e$.

Given a set of elements, in the asynchronous set relation exactly one redex is selected to be updated.

Definition 1 (Asynchronous Set Relation). *The* asynchronous relation $\xrightarrow{\square}$ *is defined as the asynchronous closure of* $\rightarrow$*, i.e., the set of pairs* $\langle A; B\rangle \in \mathcal{U} \times \mathcal{U}$ *such that* $A \xrightarrow{\square} B$ *iff there exists a* $\rightarrow$*-redex* $A' \subseteq A$ *and an element* $B' \in \mathcal{U}$ *such that* $A' \rightarrow B'$ *and* $B = (A \setminus A') \cup B'$*.*

In the parallel set relation, a non-empty collection of redices is identified to be updated in parallel (i.e., without interleaving).

Definition 2 (Parallel Set Relation). *The* parallel relation $\xrightarrow{\parallel}$ *is defined as the parallel closure of* $\rightarrow$*, i.e., the set of pairs* $\langle A; B\rangle \in \mathcal{U} \times \mathcal{U}$ *such that* $A \xrightarrow{\parallel} B$ *iff there exist* $\rightarrow$*-redices* $A_1, \dots, A_n \subseteq A$ *(nonempty) pairwise disjoint and elements* $B_1, \dots, B_n$ *in* $\mathcal{U}$ *such that* $A_i \rightarrow B_i$*, for* $1 \leq i \leq n$*, and* $B = \left(A \setminus \bigcup_{1\leq i\leq n} A_i\right) \cup \left(\bigcup_{1\leq i\leq n} B_i\right)$*.*

The synchronous set relation $\xrightarrow{s}$ applies as many atomic reductions as possible, in parallel. However, in contrast to the previous two closures, the redices are selected with the help of a strategy s, namely, a function that identifies a non-empty subset of redices. As a consequence, the synchronous set relation is a subset of the parallel set relation. It is important to note that the notion of strategy used for defining the synchronous closure of the atomic set relation is different to the one introduced in Section 1 for the framework; the name used in this section is kept from [25].

Definition 3 ($\rightarrow$-strategy). *A* $\rightarrow$-strategy *is a function* s *that maps any element* $A \in \mathcal{U}$ *into a set* $s(A) \subseteq \mathcal{P}(\rightarrow)$ *such that if* $s(A) = \{\langle A_1; B_1\rangle, \dots \langle A_n; B_n\rangle\}$*, then* $A_i \subseteq A$ *and* $A_i \rightarrow B_i$*, for* $1 \leq i \leq n$*, and* $A_1, \dots, A_n$ *are pairwise disjoint.*

Definition 4 (Synchronous Relation). *Let s be a $\rightarrow$-strategy. The* synchronous relation $\xrightarrow{s}$ *is defined as the synchronous closure of $\rightarrow$ w.r.t. s, i.e., the set of pairs $\langle A; B\rangle \in \mathcal{U} \times \mathcal{U}$ such that $A \xrightarrow{s} B$ iff $B = \left(A \setminus \bigcup_{1\leq i\leq n} A_i\right) \cup \left(\bigcup_{1\leq i\leq n} B_i\right)$ where $s(A) = \{\langle A_1; B_1\rangle, \ldots \langle A_n; B_n\rangle\}$.*

This section is concluded with an example that illustrates the notions introduced so far.

Vaccine Example. Consider the directed weighted graph $G = (V, E, i)$ in Figure 1. It represents a social system with six agents $V = \{a, b, c, d, e, f\}$ and twelve opinion influences. The label $i(u, v)$ associated to each edge (u, v) from agent u to agent v denotes the opinion influence $i_{uv} = i(u, v)$ of agent u over the opinion of agent v (about a given topic): these values are in the real interval $[0, 1]$ (i.e., $i : E \rightarrow [0, 1]$); the higher the value, the stronger the influence. In this example, the influence of f over a is the strongest possible. Notice that agents may also have *self-influence*, representing agents whose opinion need not be completely influenced by the opinion of the others.

The initial opinions (or beliefs) of the agents are depicted within the box below each node. They are specified by a function $o : V \rightarrow [0, 1]$, which is assumed to represent the opinion value $o_u = o(u)$ of each agent u on the given topic. The greater the value, the stronger (weaker) the agreement (disagreement) with the proposition, and 0 represents total disagreement. In this example such a proposition is *vaccines are safe*. Intuitively, the agents a, b, and c are in strong disagreement with vaccines being safe (the anti-vaxxers) and the rest are in strong agreement (the pro-vaxxers).

Notice that although a is the most extreme anti-vaxxer, the most extreme pro-vaxxer f has a strong influence over a. Hence, it is expected that the evolution of a's opinion will be highly influenced by the opinion of f. In general, an agent's opinion evolution takes into account a subset of its influences, as will be explained shortly.

Recall the object-like notation in Section 1. The set of elements T is made of pairs of the form $\langle u : r\rangle$ or $\langle (u, v) : r\rangle$, with $u, v \in V$, $(u, v) \in E$, and $r \in [0, 1]$. The graph in Figure 1 can be specified as the set of elements Γ:

$$\begin{aligned}\Gamma = \{\, & \langle a : 0.0\rangle, \langle b : 0.1\rangle, \langle c : 0.15\rangle, \langle d : 0.82\rangle, \langle e : 0.89\rangle, \langle f : 0.92\rangle, \\ & \langle (a, b) : 0.6\rangle, \langle (a, c) : 0.4\rangle, \langle (b, d) : 0.6\rangle, \langle (c, e) : 0.6\rangle, \langle (d, c) : 0.2\rangle, \langle (d, f) : 0.4\rangle, \\ & \langle (e, f) : 0.6\rangle, \langle (f, a) : 1.0\rangle, \langle (b, b) : 0.4\rangle, \langle (c, c) : 0.4\rangle, \langle (d, d) : 0.4\rangle, \langle (e, e) : 0.4\rangle\}.\end{aligned}$$

The atomic relation $\rightarrow_A$ is defined over elements representing agents and is parametric on a set A of elements representing edges in Γ. In this example, it follows the pattern

$$\langle u : o_u\rangle \rightarrow_A \left\langle u : \sum_{\langle (x,u) : i_{xu}\rangle \in A} o_x \cdot \frac{i_{xu}}{\sum_{\langle (y,u) : i_{yu}\rangle \in A} i_{yu}} \right\rangle, \qquad (1)$$

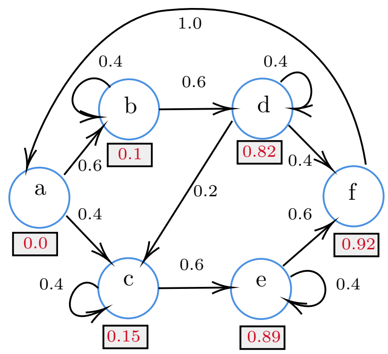

Fig. 1: Graph representing opinion and influence interaction in a social system. Initial opinions are given within the box below each node. The labels on each edge (u, v) represent the influence value of agent u over agent v.

where the summation in the denominator is assumed to be non-zero. The opinion o_u of an agent u w.r.t. to A is updated to be the weighted average of the opinion values of those agents adjacent to u and whose influence is present in A. For instance, let $A = \{\langle (a,b) : 0.6\rangle, \langle (b,b) : 0.4\rangle, \langle (c,e) : 0.6\rangle\}$. Then, the atomic set relation $\to_A$ has the following two pairs:

$$\langle b : 0.1\rangle \to_A \langle b : 0.04\rangle \qquad\qquad \langle e : 0.89\rangle \to_A \langle e : 0.15\rangle .$$

In the case of agent b, its opinion is updated to $0.04 = 0.0 \cdot \frac{0.6}{1.0} + 0.1 \cdot \frac{0.4}{1.0}$ because, w.r.t. A, it is influenced both by itself and by agent a, whose opinion value is 0.0 and influence over b is 0.6. In the case of agent e, its opinion is influenced only by agent c. The value is updated to $0.15 = 0.15 \cdot \frac{0.6}{0.6}$. It can be said that, w.r.t. A, agent e acts like a *puppet* whose own opinion is not taken into account when it is updated.

The asynchronous closure of $\to_A$ has exactly two pairs, one for each redex determined by $\to_A$ (i.e., one for agent b and another for agent e):

$$\Gamma \xrightarrow{\square}_A (\Gamma \setminus \{\langle b : 0.1\rangle\}) \cup \{\langle b : 0.04\rangle\} \qquad \Gamma \xrightarrow{\square}_A (\Gamma \setminus \{\langle e : 0.89\rangle\}) \cup \{\langle e : 0.15\rangle\}.$$

The parallel closure $\xrightarrow{\|}$ has three pairs: one in which the opinions of both b and e are updated, in addition to the same two pairs present in the asynchronous closure:

$$\Gamma \xrightarrow{\|}_A (\Gamma \setminus \{\langle b : 0.1\rangle\}) \cup \{\langle b : 0.04\rangle\} \qquad \Gamma \xrightarrow{\|}_A (\Gamma \setminus \{\langle e : 0.89\rangle\}) \cup \{\langle e : 0.15\rangle\}$$
$$\Gamma \xrightarrow{\|}_A (\Gamma \setminus \{\langle b : 0.1\rangle, \langle e : 0.89\rangle\}) \cup \{\langle b : 0.04\rangle, \langle e : 0.15\rangle\}.$$

Finally, to illustrate the synchronous closure of $\to$, let $s = A$ be the strategy. That is, all redices in $\to_A$ are identified to be reduced. Therefore, this relation has the only pair in which the opinions of both b and e are updated in parallel:

$$\Gamma \xrightarrow{s}_A (\Gamma \setminus \{\langle b : 0.1\rangle, \langle e : 0.89\rangle\}) \cup \{\langle b : 0.04\rangle, \langle e : 0.15\rangle\}.$$

3 Opinion Dynamic Models

This section shows how opinion dynamic models can be specified as set relations (see Section 2). In particular, a gossip-based and the classical De Groot opinion models are introduced, as well as a generalization of De Groot and gossip (under some conditions), here called the *hybrid opinion model*.

The three above-mentioned models are defined, as stated in Section 2, over a directed weighted graph $G = (V, E, i)$ representing a social system, with agents V, directed opinion influences $E \subseteq V \times V$, and influence values $i : E \to [0, 1]$. A given topic (i.e., proposition) is fixed. The weight $i_{uv} = i(u, v)$ associated to each edge $(u, v) \in E$ from agent u to agent v denotes the opinion influence value of agent u over the opinion value of agent v on the given topic. The opinion value $o_u = o(u) \in [0, 1]$ associated to each agent $u \in V$ in the given topic is assumed to be known by all agents in the system. As in Section 2, the higher the value of a opinion (resp. influence), the stronger the agreement (resp. influence).

The set of elements T in the set relations framework (see Section 2) is made of pairs of the form $\langle u : r\rangle$ or $\langle (u, v) : r\rangle$, with $u, v \in V$, $(u, v) \in E$, and $r \in [0, 1]$. A *G-configuration* (or *configuration*) is the set of elements in T that exactly represent the structure of G, and the values of opinions and interactions. Therefore, in the rest of this section, it is assumed that any configuration Γ can be partitioned in two sets Γ_o and Γ_i, respectively containing elements of the form $\langle u : o_u\rangle$ specifying opinions and $\langle (u, v) : i_{uv}\rangle$ specifying influences.

A model specifies how opinions (associated to agents) can be updated. Each model definition comprises three pieces; namely, an atomic relation, a strategy, and an update function for opinions. Therefore, a model specifies how a G-configuration $\Gamma = \Gamma_o \cup \Gamma_i$ can change to another G-configuration $\Gamma' = \Gamma_{o'} \cup \Gamma_i$, where only opinions are updated. It is important to note that the notion of strategy introduced in this section generalizes the notion of strategy introduced in Section 2, as will be explained later.

The atomic relation is defined in Section 3.1 for the three models. Each model is introduced by identifying a specific strategy and a specific update function in subsequent sections.

3.1 The Atomic Relation

The atomic relation $\to_A$ is parametric on a subset $A \subseteq \Gamma_i$ and defines how the opinion of a single agent may evolve. The set of influences A directly identifies the influences (and indirectly the opinions) to update the opinion of each agent in the configuration Γ (i.e., in Γ_o). For each one of the three models, the atomic relation $\to_A$ follows the pattern:

$$\langle u : o_u\rangle \to_A \langle u : \mu(\Gamma, A, u)\rangle \,, \tag{2}$$

where μ is the update function specific to each model. This function takes as input a G-configuration (e.g., Γ), a subset of its influences (e.g., A), and the agent whose opinion is to be updated (e.g., u), and outputs the new opinion for agent u w.r.t. Γ and A in the corresponding model.

3.2 Gossip-based Models

In a gossip-based model, single peer-to-peer interactions are used to update the opinion of a single user at each time-step. In general, a strategy in the proposed framework identifies a collection of subsets of interactions in Γ_i. In particular, the strategy ρ_{gossip} maps a G-configuration to the collection of singletons made from the influences in Γ_i:

$$\rho_{\text{gossip}}(\Gamma) = \{\{x\} \mid x \in \Gamma_i\}.$$

This means that, at each time-step, the opinion value of agent v can be updated w.r.t. the opinion value of agent u for each singleton $\{\langle (u,v) : i_{uv} \rangle\}$ computed by the strategy $\rho_{\text{gossip}}(\Gamma)$.

The update function μ_{gossip} is defined for any $u \in V$ and $A = \{\langle (v,u) : i_{vu} \rangle\} \in \rho_{\text{gossip}}(\Gamma)$ as:

$$\mu_{\text{gossip}}(\Gamma, A, u) = o_u + (o_v - o_u) \cdot i_{vu}.$$

Each singleton $A \in \rho_{\text{gossip}}(\Gamma)$ determines an atomic relation that updates exactly one agent's opinion in the given configuration. Recall, from Section 3.1, that each pair in the atomic set relation $\rightarrow_A$ has the form:

$$\langle u : o_u \rangle \rightarrow_A \langle u : \mu_{\text{gossip}}(\Gamma, A, u) \rangle \,.$$

Hence, in this model, the opinion of an agent u is updated by identifying an edge from other agent v with influence i_{vu} over u and by adding to its current opinion o_u the weighted difference of opinion $(o_v - o_u) \cdot i_{vu}$ of v over u.

A gossip-based model is identified as a binary set relation on G-configurations in terms of the asynchronous closure of $\rightarrow_A$, for each singleton $A \in \rho_{\text{gossip}}(\Gamma)$.

Definition 5. *The $\rightarrow_{gossip}$ set relation is the set of pairs $\langle \Gamma; \Gamma' \rangle$ of G-configurations such that:*

$$\Gamma \rightarrow_{gossip} \Gamma' \quad \textit{iff} \quad (\exists A \in \rho_{gossip}(\Gamma))\ \Gamma \overset{\square}{\rightarrow}_A \Gamma'.$$

From the viewpoint of concurrency, the gossip-based opinion dynamic model captured by $\rightarrow_{\text{gossip}}$ is non-deterministic in the sense that at each state (i.e., G-configuration) exactly $|\Gamma_i|$ transitions are possible, one per edge in E.

3.3 De Groot

In the De Groot model, the opinion value of every agent in the network is updated at each time-step. All influences are considered at the same time.

The strategy for De Groot in the proposed framework identifies the whole set of interactions in the network, i.e., Γ_i. In particular, the strategy ρ_{DeGroot} maps a G-configuration to the singleton whose only element is Γ_i:

$$\rho_{\text{DeGroot}}(\Gamma) = \{\Gamma_i\}.$$

The update function μ_{DeGroot} is defined for any $u \in V$ and $A \in \rho_{\text{DeGroot}}(\Gamma)$ (i.e., $A = \Gamma_i$) as:

$$\mu_{\text{DeGroot}}(\Gamma, A, u) = o_u + \sum_{\langle (v,u):i_{vu} \rangle \in A} (o_v - o_u) \cdot \frac{i_{vu}}{\sum_{\langle (x,u):i_{xu} \rangle \in A} i_{xu}},$$

where the summation in the denominator is assumed to be non-zero. Otherwise, the value of this function is assumed to be o_u (i.e., the opinion of agent u does not change).

The De Groot model is identified as a binary set relation on G-configurations in terms of the synchronous closure of $\to_{\Gamma_i}$ under the maximal redices strategy for $s = \Gamma_i$.

Definition 6. *The $\to_{DeGroot}$ set relation is the set of pairs $\langle \Gamma; \Gamma' \rangle$ of G-configurations such that:*

$$\Gamma \to_{DeGroot} \Gamma' \quad \textit{iff} \quad \Gamma \overset{\Gamma_i}{\to}_{\Gamma_i} \Gamma'.$$

From the viewpoint of concurrency, the De Groot opinion dynamic model captured by $\to_{\text{DeGroot}}$ is deterministic in the sense that, at each state, there is exactly only one possible transition where all influences are taken into account to update each agent's opinion without interleaving.

3.4 The Hybrid Model

The hybrid model considers every possible influence scenario in the network, i.e., any possible combination of influences are used to update the opinion of agents that may be affected by them at each time-step. Therefore, the strategy in the proposed framework identifies all non-empty subsets of interactions in Γ_i. In particular, the strategy ρ_{hybrid} maps a G-configuration to the collection of non-empty subsets made from the influences in Γ_i:

$$\rho_{\text{hybrid}}(\Gamma) = \{A \mid A \subseteq \Gamma_i \text{ and } A \neq \emptyset\}.$$

This means that, at each time-step, the opinion value of an agent v can be updated with a subset of its influencers.

The update function μ_{hybrid} is the same as function ρ_{DeGroot}. That is, it is defined for any $u \in V$ and $A \in \rho_{\text{hybrid}}(\Gamma)$ as:

$$\mu_{\text{hybrid}}(\Gamma, A, u) = o_u + \sum_{\langle (v,u):i_{vu} \rangle \in A} (o_v - o_u) \cdot \frac{i_{vu}}{\sum_{\langle (x,u):i_{xu} \rangle \in A} i_{xu}},$$

where the summation in the denominator is assumed to be non-zero. Otherwise, the value of this function is assumed to be o_u (i.e., the opinion of agent u does not change). Each subset $A \in \rho_{\text{hybrid}}(\Gamma)$ determines an atomic relation that may update more that one agent's opinion. Hence, in this model, the opinion of an agent is updated by identifying some edges that may have influence over it.

The hybrid model is identified as a binary set relation on G-configurations in terms of the synchronous closure of $\to_A$, for each subset $A \in \rho_{\text{hybrid}}(\Gamma)$.

Definition 7. *The $\rightarrow_{hybrid}$ set relation is the set of pairs $\langle \Gamma; \Gamma' \rangle$ of G-configurations such that:*

$$\Gamma \rightarrow_{hybrid} \Gamma' \quad \textit{iff} \quad (\exists A \in \rho_{hybrid}(\Gamma))\ \Gamma \xrightarrow{A}_{A} \Gamma'.$$

From the viewpoint of concurrency, the hybrid opinion dynamic model has the maximum degree of non-determinism possible. Moreover, this model is more general than the De Groot model.

Theorem 1. $\rightarrow_{DeGroot} \subseteq \rightarrow_{hybrid}$.

Proof. It follows by noting that $\Gamma_i \in \rho_{\text{hybrid}}(\Gamma)$ and, for each vertex $u \in V$, the equality $\mu_{\text{DeGroot}}(\Gamma, \Gamma_i, u) = \mu_{\text{hybrid}}(\Gamma, \Gamma_i, u)$ holds.

It is not necessarily the case that $\rightarrow_{\text{gossip}} \subseteq \rightarrow_{\text{hybrid}}$. This is because the update functions do not always agree when the collection of selected influences A is a singleton. In particular, for each singleton $A = \{\langle (v, u) : i_{vu} \rangle\}$, $\mu_{\text{hybrid}}(\Gamma, A, u) = o_v$, meaning that agent u in the hybrid model behaves always like a puppet when $u \neq v$. Note that this is not (necessarily) the case in $\rightarrow_{\text{gossip}}$. Nevertheless, there is a class of graphs for which this inclusion holds.

Theorem 2. *If G is such that each vertex has a self-loop and is influenced at most by another vertex, and the summation of its incoming influences is 1, then $\rightarrow_{gossip} \subseteq \rightarrow_{hybrid}$.*

Proof. If $\Gamma \rightarrow_{\text{gossip}} \Gamma'$, there is a singleton $A \in \rho_{\text{gossip}}(\Gamma)$ such that $\Gamma \xrightarrow{\square}_{A} \Gamma'$. Let $A = \{\langle (v, u) : i_{vu} \rangle\}$. If u has exactly one incoming edge, then $v = u$ (by the initial assumption) and $\rho_{\text{gossip}}(\Gamma, A, u) = o_u = \rho_{\text{hybrid}}(\Gamma, A, u)$. Since $A \in \rho_{\text{hybrid}}(\Gamma)$, it follows that $\Gamma \rightarrow_{\text{hybrid}} \Gamma'$. If u has two edges, and the self-loop is taken, the case $v = u$ is as above. Otherwise, if $u \neq v$, the same transition is obtained in the hybrid model by taking $A' \in \rho_{\text{hybrid}}(\Gamma)$ where $A' = A \cup \{\langle (u, u) : 1 - i_{vu} \rangle\}$ (an noticing that the denominator in μ_{hybrid} becomes 1).

4 The Framework in Rewriting Logic

This section presents a rewrite theory that implements the set relations framework in Section 2. Off-the-shelf definitions are provided to instantiate the framework with opinion dynamic models, such as the ones introduced in Section 3. This section assumes familiarity with rewriting logic [21] and Maude [7]; Appendix A presents some preliminaries on these two subjects. The full Maude specification supporting the set relations framework is available at [23].

A rewrite theory $\mathcal{R}$ (using Maude's notation) is defined to represent networks of agents and their opinions. The atomic relation (Equation (2)) is defined as a non-executable rewrite rule, and the set relation framework is implemented using the meta-programming facilities in Maude. The framework is parametric on an update function (μ) and a strategy (ρ), as explained in Section 3. The rewrite theory $\mathcal{R}$ must be extended equationally to instantiate such parameters.

4.1 Influences, Opinions, and State

An agent a and its opinion o_a, and the influence of agent a over agent b with weight i_{ab}, are specified with the help of the following sorts and function symbols:

```
sorts Agent Opinion Edge .
op  <_:_>    :      Agent       Float -> Opinion [ctor] .
op  <`(_,_`):_> : Agent Agent Float -> Edge    [ctor] .
```

The user is expected to provide appropriate constructors for the sort `Agent`, e.g., by extending $\mathcal{R}$ with the subsort relation `subsort Nat < Agent` to use natural numbers as identifiers for agents.

Sets of agents, opinions, and edges (sorts `SetAgent`, `SetOpinion`, and `SetEdge` respectively) are defined as "`,`"-separated sets of elements in the usual way. A G-configuration $\Gamma = \Gamma_o \cup \Gamma_i$ is represented by a term of sort `Network`, defining the set of agents' opinions (Γ_o) and influences (Γ_i) with the following sorts and function symbols:

```
sort Network .
op < nodes:_ ; edges:_ > : SetOpinion SetEdge -> Network [ctor] .
```

Analyzing opinion dynamics usually requires determining the number of interactions between agents and the time needed to reach a given state. A term of the form "N `in step:` t `comm:` nc" of sort `State` represents the state of a network N at the current time-unit t, when a number of interactions/communications nc have taken place:

```
 sort State .
 op _ in step:_ comm:_ : Network Nat Nat -> State [ctor] .
```

4.2 Strategies and the Atomic Relation

The atomic relation $\rightarrow_A$ is parametric on a non-empty subset $A \subseteq \Gamma_i$. A strategy identifies each one of such subsets at each time-step. A `SetSetEdge` is a "`;`"-separated set of set of edges.

```
sort SetSetEdge . subsort NeSetEdge < SetSetEdge .
op mt : -> SetSetEdge [ctor] .
op _;_ : SetSetEdge SetSetEdge -> SetSetEdge [ctor assoc comm id: mt] .
```

Some distinguished `SetSetEdge`s include the singleton with all the edges in the network (De Groot model), the set containing only singletons (Gossip model) and the set of non-empty subsets of edges (Hybrid model).

```
var SE : SetEdge .   var E : Edge .
op deGroot  : SetEdge     -> SetSetEdge .
eq deGroot(SE) = SE .

op gossip  : SetEdge       -> SetSetEdge .
eq gossip(empty) = mt .
eq gossip((E, SE)) = E ; gossip(SE) .

op hybrid : SetEdge -> SetSetEdge .
```

```
eq hybrid(SE) = power-set(SE) \ empty .
```

```
op strategy :  -> SetSetEdge . --- user defined strategy
```

The operator `strategy` must be defined by the user to identify the subsets $A \subseteq \Gamma_i$ available in each transition. This can be done, e.g., by adding the equation

```
eq strategy =  gossip(edges) .
```

where `edges` is the set of edges in the network currently being modeled.

The atomic rewrite relation is captured by a non-executable rewrite rule that updates the belief of a given `AGENT` (u in Equation (2)) when a set of `EDGES` (A) is selected and the current state of the system is `STATE` (Γ):

```
var AGENT : Agent . vars BELIEF BELIEF' : Float . var STATE : State .
vars SETEDGE EDGES : SetEdge .

op update : State SetEdge Agent -> Float .  --- user defined μ

crl [atomic] : < AGENT : BELIEF >  =>  < AGENT :  BELIEF' >
    if BELIEF' := update(STATE, SETEDGE, AGENT) [nonexec] .
```

The function `update` (μ in Equation (2)) must be specified by the user. The framework provides instances of this function for the models presented in Section 3.

An asynchronous, parallel, or synchronous rewrite step, depending on the underlying strategy, is captured by the rewrite rule `step` below:

```
var SETNODE : SetNode . vars STEPS COMM : Nat .
op moduleName : -> Qid . --- Name of the module with the user's network

crl [step] : STATE => STATE'
    if  EDGES ; SSE := strategy /\
        STATE'      := step([moduleName], STATE, EDGES) .
```

In this rule, the current `STATE` is updated to `STATE'` by non-deterministically selecting a set of `EDGES` from the set of set of edges available according to the `strategy`. The function `step` below takes as parameters the meta-representation of the user's module defining the network (`moduleName`), the current state, and the selected set of edges.

```
var SETAG : SetAgent . var SETOP : SetOpinion . var OP : Opinion .
op step : Module State SetAgent SetOpinion SetEdge -> State .
op step : Module State                     SetEdge -> State .
eq step(M, STATE, EDGES) = step(M, STATE, incidents(EDGES), empty, EDGES) .
eq step(M, STATE, empty, SETOP, EDGES) =
   < nodes: (nodes(STATE) / SETOP) ; edges: edges(STATE) >
   in step: (steps(STATE) + 1) comm: (comm(STATE) + | non-self(EDGES) |) .
eq step(M, STATE, (AGENT, SETAG), SETOP, EDGES) =
   step(M, STATE, SETAG, (SETOP, next(M, AGENT, EDGES, STATE)), EDGES) .
```

The function `step` recursively computes the beliefs of the agents `incident` to `EDGES`. The updated beliefs are accumulated in the set of opinions `SETOP`. The opinions of the other agents remain as in `STATE` (operator `/`), and the number of steps and the number of communications are updated accordingly. The expression | `non-self(.)` | returns the number of edges that are not self-loops, and `nodes(.)` returns the opinions (Γ_o) in a state.

The function `next` computes the outcome of the transition $\langle u : o_u \rangle \rightarrow_A \langle u : o'_u \rangle$ by applying (`metaApply`) the rule `atomic` with the needed substitutions to make this rule executable (and deterministic). Namely, it fixes the opinion to be updated (`AGENT` and `BELIEF`), the current `STATE` and the set of `EDGES` to be considered during the update.

```
op next : Module Agent SetEdge State -> Opinion .
ceq  next(M, AGENT, EDGES, STATE) = OP
 if  SUBS := 'AGENT:Agent   <- upTerm(AGENT) ;
            'BELIEF:Float  <- upTerm(opinion(AGENT, STATE)) ;
            'STATE:State   <- upTerm(STATE) ;
            'EDGES:SetEdge <- upTerm(EDGES) /\
     RES? := metaApply(M, upTerm(< AGENT :  opinion(AGENT, STATE) >),
            'atomic, SUBS, 0) /\
     OP   := if RES? == failure then error
            else downTerm(getTerm(RES?), error) fi .
```

The `opinion` function returns the opinion of an agent in a given state.

5 Experimentation

This section shows how Maude and some of its tools can be used to analyze instantiated versions of the rewrite theory $\mathcal{R}$ (see Section 4) to better understand the evolution of opinions in networks of agents. Of special interest is checking the (im)possibility of reaching a consensus (i.e., agent's opinions converge to a given value) or stability of the systems, computing the number of steps to reach consensus, computing an optimal strategy to reach consensus, measuring the polarization of the system at each time-step, among others. It is noticed that for De Groot and Gossip-like models, there are theoretical results identifying topological conditions that guarantee consensus. In particular, in these models, the agents reach consensus if the graph is strongly connected and aperiodic (i.e., the greatest common divisor of the lengths of its cycles is one) [14].

5.1 Finding Consensus

Let `Example-DG` be the module/theory extending $\mathcal{R}$ with the following operators and equations:

```
op init : -> Network .                  --- Initial state (as in Fig 1)
eq init = < nodes: ... ; edges: ... > in step: 0 comm: 0 .
eq moduleName = 'Example-DG .           --- Name of the theory
--- Predefined μ for De Groot
eq update(STATE, SETEDGE, AGENT)  = deGrootUpdate(STATE, SETEDGE, AGENT) .
eq strategy   = deGroot(edges(init)) . --- De Groot strategy
```

The following command answers the question of whether it is possible to reach a consensus from the `initial` state. (Function `consensus(.)` checks if all opinions o_i and o_j in a given state satisfy $|o_i - o_j| < \epsilon$, where ϵ is an error bound).

```
Maude> search [1]  init =>* STATE such that consensus(STATE) .

Solution 1 (state 34)
STATE --> < nodes: < 0 : 4.80e-1 >, < 1 : 4.79e-1 >, < 2 : 4.79e-1 >, ...
            edges: <(0,1): 5.99e-1 >, <(0,2): 4.00e-1 >, ... >
            in step: 34 comm: 272
```

The consensus about the given proposition is approximately 0.48 and it is reached in 34 steps. Since in the De Groot model all the 12 edges are considered in each interaction, there is a total of $272 = 34 \times 8$ communications (the interactions on the self-loops are not considered in that counting). Note that an application of rule `step` in this case is completely deterministic (the strategy considers only one possible outcome, including all the edges of the network).

Let `Example-H` be as `Example-DG`, but considering the strategy and update functions for the hybrid model. As explained in Section 3.4, the Hybrid model exhibits the maximum degree of non-determinism. Using `search` to check the existence of a reachable state satisfying consensus for the system in Figure 1 (12 edges) becomes unfeasible: a state may have up to 4095 (non-empty subsets of Γ_i) successor states. Certainly, for this network, a solution must exist due to the above output of the `search` command and the fact that $\rightarrow_{\mathrm{DeGroot}} \subseteq \rightarrow_{\mathrm{hybrid}}$.

Consider the following rewrite rule and expression in Maude's strategy language:

```
crl [step'] :   STATE  => STATE'
if  STATE'  := step([moduleName], STATE, EDGES) [nonexec] .

var STR : SetSetEdge .
strat round :  SetSetEdge @ State .
sd round(EDGES ; STR) := (match STATE s.t. consensus(STATE))
                          or-else step'[EDGES <- EDGES] ; round(STR) .
```

Unlike `step`, rule `step'` does not use the model strategy to select the set of `EDGES` that will be used to compute the next state (and hence, it is non executable). The Maude's strategy `round` checks whether the current state satisfies consensus and stops. Otherwise, it non-deterministically chooses a set `EDGES`, applies the rule `step'` instantiating the set of edges with that particular set, and it is recursively called without `EDGES`. In other words, `round` starts with a set of possible interactions and it allows for these interactions to happen only once. This is certainly one of the possible behaviors that can be observed with the Hybrid model. Using this strategy, it is possible to find some states that satisfy consensus and answer the question whether by selecting some groups of agents (non necessarily disjoint) that, interacting only once, may lead to a consensus. (Function `filter>=(n,STR)` returns the sets in `STR` with cardinality at least n).

```
Maude> dsrew [1] init using round(hybrid(edges)) .
Solution 1
result State: < nodes: < 0 : 0.0 >, ... edges: ... > in step: 8 comm: 13 .

Maude> dsrew [1] init using round(filter>=(6, hybrid(edges))) .
Solution 1
result State: < nodes: < 0 : 1.50e-1 >, ... > in step: 21 comm: 88 .
```

As expected, because of the non-deterministic nature of the Hybrid model, the value of consensus (and the number of steps to reach such a state) can heavily depend on the choice of edges at each step. In the first output returned by `dsrew` in the first command, all the sets considered by `round` included edges where a acts as an influencer and the edge $f \longrightarrow a$ is never selected. This explains the value of the consensus, where the opinion of a was propagated to her neighbors. In the send command, larger groups are chosen to interact, and the edge $f \longrightarrow a$ is selected in 4 out of the 21 interactions. Hence, a eventually changes her opinion.

5.2 Statistical Analysis

An alternative approach to deal with the inherent state explosion problem when analyzing $\mathcal{R}$ is to perform statistical model checking. In the following, the tool `umaudemc` [26] is used for such a purpose. The `umaudemc` command `scheck` enables Monte-Carlo simulations of a rewrite theory extended with probabilities; it estimates the value of a quantitative temporal expression written in the query language QuaTEx [1].

Consider the following QuaTEx expression that computes the probability of reaching a consensus before N communications:

```
Prob-C(N) = if (s.rval("consensus(S)")) then 1.0 else
             if (s.rval("comm(S)") <= N) then # Prob-C(N) else 0.0 fi fi;
```

The two commands below estimate the probability of reaching consensus before 30 (QuaTEx formula `E[Prob(30)]`) and 20 communications, respectively, in the running example when the gossip-based model is considered. The confidence level of these analyses is 95% and the same probability is assigned to every successor state (`uniform`).

```
umaudemc  scheck ex-gossip init formula -a 0.05 -d 0.01 --assign uniform
  (μ = 0.587)
umaudemc scheck ex-gossip init formula -a 0.05  -d 0.01 --assign uniform
  (μ = 0.348)
```

As expected, reducing the maximum number of communications decreases the changes of reaching a consensus state.

The authors in [5] hypothesize that the less dispersed opinion becomes, the easier it will be to reach consensus. In fact the variance, a standard measure of dispersion, is used as a measure of opinion polarization in social networks [5]. The following commands aim at testing such a hypothesis in the running example when considering the hybrid model:

```
umaudemc  scheck example-H init ... --assign uniform
  (μ = 0.901)
umaudemc scheck example-H init  ... --assign "term(variance(L,R))"
  (μ = 1.0)
umaudemc scheck example-H init ...  --assign "term(distance(L,R))"
  (μ = 0.987)
```

These commands estimate the probability of reaching consensus before 300 communications (`E[Prob(300)]`). In the first case, all the successor states are assigned the same probability. In the second, successor states whose set of chosen agents has higher `variance` are assigned higher probabilities. In the third command, successor states whose set of chosen agents are more *polarized*, in the sense that the `distance` between the maximal and the minimal opinions is bigger, are assigned higher probabilities. These results confirm the hypothesis that it is more likely (1.0 vs 0.9) to reach consensus sooner when communications of agents with more distant opinions is encouraged to reduce dispersion of opinions.

6 Concluding Remarks

This paper presented a unified framework for dynamic opinion models. Such models are tools to analyze the evolution of opinion values, about a given topic, in a network of agents whose opinion may be influenced by other agents. Set relations, which are used for specifying and analyzing concurrent behavior in collections of agents, are the formalism used to unify the modeling of these systems. This framework relies on two mechanisms, namely, an atomic relation that updates the opinion of single agents based on a collection of interactions and a strategy defining the collections of interactions to be considered. The framework is formally specified as a rewrite theory, which is expected to be instantiated for the opinion dynamic model of interest. Three different dynamic opinion models (De Groot, goossip-like, and hybrid) are shown to be instances of this framework. Experiments on these models show that statistical model checking is a promising alternative to tackle the state explosion problem when analyzing models with a high degree of non-determinism, such is the case of the hybrid model. To the best of the authors' knowledge, this is the first documented effort to make available concurrency theory, techniques, and tools for the specification and analysis of opinion dynamics models and properties such as polarization and consensus.

The ultimate goal of making available computational ideas and approaches for analyzing phenomena in social networks requires (significant) additional work. First, a more in-depth exploration of properties related to these phenomena in social networks is required. This may lead to the proposal of new temporal and probabilistic properties that cannot be handled with current techniques and approaches supporting the opinion dynamic modeling community, but that may be highly supported by the developments in concurrency and computational logics. Second, extensions to the current framework in terms of more general dynamic networks (i.e., the value of influences can change), temporal networks (i.e., nodes and edges can appear and disappear), and the inclusion of several topics/propositions that may share causal relations are in order. Third, more experimental validation is required, ideally with data gathered from real social networks. Fourth, building on the abstract relations proposed here, techniques from concurrency theory become available for the analysis of social systems. It is worth exploring standard concurrency techniques such as bisimulation and testing equivalences to answer questions such as whether two social systems ought to be equivalent and whether there is a social context, represented as a social system, that can tell the difference between two other social systems.

References

1. Agha, G., Meseguer, J., Sen, K.: PMaude: Rewrite-based Specification Language for Probabilistic Object Systems. ENTCS **153**(2), 213–239 (May 2006)
2. Alvim, M.S., Amorim, B., Knight, S., Quintero, S., Valencia, F.: A formal model for polarization under confirmation bias in social networks. Log. Methods Comput. Sci. **19**(1) (2023)
3. Ballard, A.O., DeTamble, R., Dorsey, S., Heseltine, M., Johnson, M.: Dynamics of polarizing rhetoric in congressional tweets. Legislative Studies Quarterly **48**(1), 105–144 (2023)
4. Beaufort, M.: Digital media, political polarization and challenges to democracy. Information, Communication & Society **21**(7), 915–920 (2018)
5. Bramson, A., Grim, P., Singer, D.J., Berger, W.J., Sack, G., Fisher, S., Flocken, C., Holman, B.: Understanding polarization: Meanings, measures, and model evaluation. Philosophy of Science **84**(1), 115–159 (2017)
6. Center for Strategic and International Studies: The #MilkTeaAlliance in Southeast Asia: Digital revolution and repression in Myanmar and Thailand (April 2021), https://www.csis.org/blogs/new-perspectives-asia/milkteaalliance-southeast-asia-digital-revolution-and-repression, visited 12-30-2023
7. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.: All About Maude - A High-Performance Logical Framework, LNCS, vol. 4350. Springer (2007)
8. Das, A., Gollapudi, S., Munagala, K.: Modeling opinion dynamics in social networks. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining. p. 403–412. Association for Computing Machinery, New York, NY, USA (2014)
9. Degroot, M.H.: Reaching a consensus. Journal of the American Statistical Association **69**(345), 118–121 (1974)
10. Fagnani, F., Zampieri, S.: Randomized consensus algorithms over large scale networks. In: 2007 Information Theory and Applications Workshop. pp. 150–159 (2007)
11. Fitriani, Habib, M.: Social media and the fight for political influence in southeast asia (August 2023), https://thediplomat.com/2023/08/social-media-and-the-fight-for-political-influence-in-southeast-asia, visited 12-30-2023
12. Foundation, T.A.: Violent Conflict, Tech Companies, and Social Media in Southeast Asia: Key Dynamics and Responses. The Asia Foundation, San Francisco, USA (2020)
13. Garrett, R.K.: The "echo chamber" distraction: Disinformation campaigns are the problem, not audience fragmentation. Journal of Applied Research in Memory and Cognition **6** (2017)
14. Golub, B., Sadler, E.: Learning in social networks. Social Science Research Network (SSRN) (02 2017), http://dx.doi.org/10.2139/ssrn.2919146
15. Gordon-Zolov, T.: Chile's estallido social and the art of protest. Sociologica **17**(1), 41–55 (January 2023)
16. Gupta, S., Chauhan, V.: Understanding the role of social networking sites in political marketing. Jindal Journal of Business Research **12**(1), 58–72 (2023)
17. Haoxiang Xia, Huili Wang, Z.X.: Opinion dynamics: A multidisciplinary review and perspective on future research. International Journal of Knowledge and Systems Science **2**(4), 72–91 (2023)

18. Iversen, T., Soskice, D.: Information, inequality, and mass polarization: Ideology in advanced democracies. Comparative Political Studies **48**(13), 1781–1813 (2015)
19. Kirby, E.: The city getting rich from fake news. BBC News Documentary (05 2017), https://www.bbc.com/news/magazine-38168281
20. Lynch, M.: After the arab spring: How the media trashed the transitions. Journal of Democracy **26**(4), 90–99 (2015)
21. Meseguer, J.: Conditional rewriting logic as a unified model of concurrency. Theoretical Computer Science **96**(1), 73–155 (1992)
22. Neverov, K., Budko, D.: Social networks and public policy: Place for public dialogue? In: Proceedings of the International Conference IMS-2017. p. 189–194. Association for Computing Machinery, New York, NY, USA (2017)
23. Olarte, C., Ramírez, C., Rocha, C., Valencia, F.: Opinion dynamic modeling as concurrent set relations in rewriting logic, https://github.com/promueva/maude-opinion-model
24. Rocha, C., Muñoz, C.A.: Synchronous set relations in rewriting logic. Science of Computer Programming **92**, 211–228 (2014)
25. Rocha, C., Muñoz, C.A., Dowek, G.: A formal library of set relations and its application to synchronous languages. Theoretical Computer Science **412**(37), 4853–4866 (2011)
26. Rubio, R., Martí-Oliet, N., Pita, I., Verdejo, A.: Strategies, model checking and branching-time properties in maude. J. Log. Algebraic Methods Program. **123**, 100700 (2021)
27. Rubio, R., Martí-Oliet, N., Pita, I., Verdejo, A.: Qmaude: Quantitative specification and verification in rewriting logic. In: Chechik, M., Katoen, J., Leucker, M. (eds.) FM 2023. LNCS, vol. 14000, pp. 240–259. Springer (2023)
28. Sarma, P., Hazarika, T.: Social media and election campaigns: An analysis of the usage of twitter during the 2021 assam assembly elections. International Journal of Social Science Research and Review **6**(2), 96–117 (January 2023)
29. Suresh, V.P., Nogara, G., Cardoso, F., Cresci, S., Giordano, S., Luceri, L.: Tracking fringe and coordinated activity on twitter leading up to the us capitol attack (2023)
30. Wikipedia Foundation: 2021 Colombian protests, https://en.wikipedia.org/wiki/2021_Colombian_protests, visited 12-30-2023

A Appendix: Overview of Maude

A *rewrite theory* [21] is a tuple $\mathcal{R} = (\Sigma, E, L, R)$ such that: (Σ, E) is an equational theory where Σ is a signature that declares sorts, subsorts, and function symbols; E is a set of (conditional) equations of the form $t = t'$ **if** ψ, where t and t' are terms of the same sort, and ψ is a conjunction of equations; L is a set of *labels*; and R is a set of labeled (conditional) rewrite rules of the form $l : q \longrightarrow r$ **if** ψ, where $l \in L$ is a label, q and r are terms of the same sort, and ψ is a conjunction of equations. Condition ψ in equations and rewrite rules can be more general than conjunction of equations, but this extra expressiveness is not needed in this paper.

$T_{\Sigma,s}$ denotes the set of ground terms of sort s, and $T_\Sigma(X)_s$ denotes the set of terms of sort s over a set of sorted variables X. $T_\Sigma(X)$ and T_Σ denote all terms and ground terms, respectively. A substitution $\sigma : X \to T_\Sigma(X)$ maps each variable to a term of the same sort, and $t\sigma$ denotes the term obtained by simultaneously replacing each variable x in a term t with $\sigma(x)$.

A *one-step rewrite* $t \longrightarrow_{\mathcal{R}} t'$ holds if there is a rule $l : q \longrightarrow r$ **if** ψ, a subterm u of t, and a substitution σ such that $u = q\sigma$ (modulo equations), t' is the term obtained from t by replacing u with $r\sigma$, and $v\sigma = v'\sigma$ holds for each $v = v'$ in ψ. The reflexive-transitive closure of $\longrightarrow_{\mathcal{R}}$ is denoted as $\longrightarrow^*_{\mathcal{R}}$.

A rewrite theory $\mathcal{R}$ is called *topmost* iff there is a sort *State* at the top of one of the connected components of the subsort partial order such that for each rule $l : q \longrightarrow r$ **if** ψ, both q and r have the top sort *State*, and no operator has sort *State* or any of its subsorts as an argument sort.

Maude [7] is a language and tool supporting the specification and analysis of rewrite theories. A Maude module (`mod` M `is` ... `endm`) specifies a rewrite theory $\mathcal{R}$. Sorts and subsort relations are declared by the keywords `sort` and `subsort`; function symbols, or *operators*, are introduced with the `op` keyword: `op` f `:` s_1 ... s_n `->` s, where $s_1, \ldots, s_n$ are the sorts of its arguments, and s is its (value) sort. Operators can have user-definable syntax, with underbars '`_`' marking each of the argument positions (e.g., `_+_`). Some operators can have equational attributes, such as `assoc`, `comm`, and `id:` t, stating that the operator is, respectively, associative, commutative, and/or has identity element t. Equations are specified with the syntax `eq` t `=` t' or `ceq` t `=` t' `if` ψ; and rewrite rules as `rl` `[`l`]` `:` u `=>` v or `crl` `[`l`]` `:` u `=>` t' `if` ψ. The mathematical variables in such statements are declared with the keywords `var` and `vars`.

Maude provides a large set of analysis methods, including computing the normal form of a term t (command `red` t), simulation by rewriting (`rew` t), reachability analysis (`search` t `=>*` t' `such that` ψ), and rewriting according to a given rewrite strategy (`srew` t `using` *str*). Basic such rewrite strategies include $r[\sigma]$ (apply rule with label r once with the optional ground substitution σ), `idle` (identity), `fail` (empty set), and `match` P `s.t.` C, which checks whether the current term matches the pattern P subject to the constraint C. Compound strategies can be defined using concatenation $(\alpha\,;\,\beta)$, disjunction $(\alpha \,|\, \beta)$, iteration $(\alpha\,*)$, α `or-else` β (execute β if α fails), etc.

The Unified Maude model-checking tool [26] (`umaudemc`) allows for the use of different model checkers to analyze Maude specifications. Besides being an interface for the standard LTL model checker of Maude, it also offers the possibility of interfacing external CTL and probabilistic model checkers. For the purpose of this paper, the command `scheck` [27] is used to assign probabilities to the transition system generated by an initial term t, and perform statistical model checking to estimate quantitive expressions written in the QuaTEx language. Hence, it is possible to compute, e.g., the expected value of the number of communication or interactions needed to reach a consensus in a network.

Meta-programming. Maude supports *meta-programming*, where a Maude module M (resp., a term t) can be (meta-)represented as a Maude *term* $\overline{M}$ of sort `Module` (resp. as a Maude term $\bar{t}$ of sort `Term`) in Maude's `META-LEVEL` module. Maude provides built-in functions such as `metaRewrite`, and `metaSearch`, which are the "meta-level" functions corresponding to "user-level" commands to perform rewriting and search, respectively.