Determining Majority in Networks with Local Interactions and very Small Local Memory*

George B. Mertzios[1], Sotiris E. Nikoletseas[2],
Christoforos L. Raptopoulos[2,3], and Paul G. Spirakis[2,4]

[1] School of Engineering and Computing Sciences, Durham University, UK
[2] Computer Technology Institute (CTI) and University of Patras, Greece
[3] Computer Science Department, University of Geneva, Switzerland
[4] Department of Computer Science, University of Liverpool, UK
george.mertzios@durham.ac.uk, nikole@cti.gr,
raptopox@ceid.upatras.gr, p.spirakis@liverpool.ac.uk

Abstract. We study here the problem of determining the majority type in an arbitrary connected network, each vertex of which has initially two possible types (states). The vertices may have a few additional possible states and can interact in pairs only if they share an edge. Any (population) protocol is required to stabilize in the initial majority, i.e. its output function must interpret the local state of each vertex so that each vertex outputs the initial majority type. We first provide a protocol with 4 states per vertex that *always* computes the initial majority value, under any fair scheduler. Under the uniform probabilistic scheduler of pairwise interactions, we prove that our protocol stabilizes in expected polynomial time for any network and is quite fast on the clique. As we prove, this protocol is optimal, in the sense that there does not exist any population protocol that always computes majority with fewer than 4 states per vertex. However this does not rule out the existence of a protocol with 3 states per vertex that is correct with high probability (whp). To this end, we examine an elegant and very natural majority protocol with 3 states per vertex, introduced in [2] where its performance has been analyzed for the clique graph. In particular, it determines the correct initial majority type in the clique very fast and whp under the uniform probabilistic scheduler. We study the performance of this protocol in arbitrary networks. We prove that, when the two initial states are put uniformly at random on the vertices, the protocol of [2] converges to the initial majority with probability higher than the probability of converging to the initial minority. In contrast, we present an infinite family of graphs, on which the protocol of [2] can fail, i.e. it can converge to the initial minority type whp, even when the difference between the initial majority and the initial minority is $n - \Theta(\ln n)$. We also present another infinite family of graphs in which the protocol of [2] takes an expected exponential time to converge. These two negative results build upon a very positive result concerning the robustness of the protocol of [2] on the clique, namely that if the initial minority is at most $\frac{n}{7}$, the protocol fails

*Partially supported by (i) the MULTIPLEX project – 317532, (ii) the EPSRC Grant EP/K022660/1, and (iii) the SHARPEN project – PE6 (1081).

with exponentially small probability. Surprisingly, the resistance of the clique to failure causes the failure in general graphs. Our techniques use new domination and coupling arguments for suitably defined processes whose dynamics capture the antagonism between the states involved.

1 Introduction

One of the most natural computational problems in many physical systems is to compute the *majority*, i.e. to determine accurately which type of an element of the system appears more frequently. For instance, the majority problem is encountered in various settings such as in voting [8, 10], in epidemiology and interacting particles systems [13], in diagnosis of multiprocessor systems [17], in social networks [14,16] etc. In distributed computing, the majority problem is an important and natural special case of the central problem of reaching *consencus* within a system [6,12], where a number of processes have to agree on any single data value (e.g. leader election [7]). In all these physical systems, some pairs of elements may interact with each other while other pairs may not be able to interact directly. This structure of the possible pairwise interactions between elements of the system can be modeled by a network (i.e. graph), where elements and possible interactions are represented by vertices and edges, respectively.

In order to solve the majority computation problem in a network, we first need to make some assumptions on the underlying model of computation. Much research has been done under the assumption that there exists a central authority, as well as unlimited available memory and full information about the whole network (see e.g. [5, 18]). However, in many real systems we do not have (or we do not wish to have) such a powerful computational model. The weaker the considered model of computation is (e.g. lack of central authority, partial or no information about the system, lack of memory etc.), the more challenging the majority computation becomes.

One of the ways to study distributed systems where agents may interact in pairs and each individual agent is extremely limited (in fact, being equipped only with a finite number of possible states) is by using *population protocols* [1, 3]. Then the complex behavior of the system emerges from the rules governing the possible pairwise interactions of the agents. Population protocols have been defined by analogy to population processes [11] in probability theory and have already been used in various fields, such as in statistical physics, genetics, epidemiology, chemistry and biology [4].

In particular, population protocols are *scalable*, i.e. they work independently of the size n of the underlying network (called the *interaction graph*) and the value of n is not even known to the protocol. Furthermore they are *anonymous*, i.e. there is only one transition function which is common to all entities/agents: the result of an interaction of an agent u at state q_u with an agent v at state q_v is the same regardless of the identity of u and v. The transition function of a population protocol only specifies the result of every possible interaction, without specifying *which* pairs of agents interact or *when* they are chosen to

interact. Usually it is assumed that interactions between agents happen under some kind of a *fairness* condition. For a survey we refer to [3].

In this direction, a very natural and simple population protocol for the majority problem on the clique (i.e. the complete graph), where initially every vertex has one of two possible types (states), has been introduced and analyzed in [2]. In particular, the protocol of [2] assigns only 3 possible states to every agent (i.e. there is a 3×3 transition table capturing all possible interactions) and the interactions between agents are dictated by a *probabilistic scheduler* (i.e. all pairs have the same probability to interact at any step). Every vertex has an identity v, but it is unaware of the identity of any other vertex, as well as of its own identity. Although the underlying interaction graph in [2] is assumed to be a clique, the authors distinguish in their protocol the agents u and v participating in an interaction into an "initiator" and a "responder" of the interaction (when agents u and v interact, each of them becomes initiator or responder with equal probability). Their main result is that, if initially the difference between the initial majority from the initial minority in the complete graph with n vertices is $\omega(\sqrt{n}\log n)$, their protocol converges to the correct initial majority value in $O(n \log n)$ time with high probability.

Most works on population and majority dynamics so far considered only two entity types (e.g. the voter model [8], the Moran process [15]). The analysis of population dynamics with more than two types is challenging. As an example we refer to the model of [2], in which, although agents can have initially one of only two types (red and green), the protocol itself allows every agent to be in one among three different states (red, green and blank) at every subsequent time point. Even though this model is quite simple, it is very hard to be analyzed. Computing the majority with as few states as possible in the more general case, where the interaction graph has an arbitrary structure (as opposed to the complete graph that has been mainly considered so far) remained an open problem.

1.1 Our contribution

In this paper we study the majority problem in an *arbitrary* underlying interaction graph G, where initially every vertex has two possible states (red and green). We consider here the weakest and simplest possible model of computation. In particular, we assume the existence of no central authority and we allow every vertex of G to have only a (small) constant number of available types (or states). Although every vertex of G has a unique identity, no vertex is aware of its own identity or the identity of any other vertex. Furthermore, although only two adjacent vertices can interact, vertices of G do not even know to which other vertices they are adjacent.

First, we focus on the problem of *always* computing the correct majority value in an *arbitrary* (directed or undirected) interaction graph G, regardless of how large the initial difference between the majority and the minority is. In particular, assuming that the interacting pairs of vertices are chosen by an arbitrary fair scheduler, we derive matching lower and upper bounds on the number of available states, for which there exists a population protocol that

always computes the correct majority value. For the lower bound, we prove that there does not exist any population protocol that achieves this with at most 3 different states per vertex. On the other hand, for the matching upper bound we provide a population protocol with 4 states per vertex, which always computes the correct majority value, even if initially the difference between majority and minority is 1. To the best of our knowledge, this is the first 4-state population protocol that correctly computes the majority value in a two type population on an arbitrary interaction graph. In particular, the 4-state majority protocol proposed in [3] only works when the interaction graph is complete. Furthermore we provide polynomial upper bounds on the expected time needed by our new protocol to converge, and we show that in certain cases the running time is $O(n \log n)$, i.e. the same as for the fast protocol of [2].

Second, we provide a detailed analysis of the 3-state protocol of [2] on an arbitrary interaction graph G. Our first result in this direction is that, when the two initial types (red and green) are distributed on the vertices of an arbitrary graph G uniformly at random, the protocol of [2] will converge to the initial majority with higher probability than to the initial minority. The proof of this relies on a well known result in extremal combinatorics (in particular, on Hall's marriage Theorem). Furthermore we present an infinite family of graphs $\{G_n\}_{n\in\mathbb{N}}$ on which the protocol of [2] can fail (i.e. it can converge to the initial minority) with high probability, even when the difference between the initial majority and the initial minority is as large as $n - \Theta(\ln n)$. Then we present another infinite family of graphs $\{G'_n\}_{n\in\mathbb{N}}$ on which the protocol of [2] can take an exponential expected number of steps to converge. In particular, this rules out the possibility to use a Markov chain Monte-Carlo approach to approximate the probability that the protocol of [2] converges to the correct majority value.

In order to prove our results on the classes $\{G_n\}_{n\in\mathbb{N}}$ and $\{G'_n\}_{n\in\mathbb{N}}$, we first proved the intermediate result that for any $\varepsilon > 0$, if the minority has size at most $(\frac{1}{7} - \varepsilon)n$ in the complete graph with n vertices, then the protocol of [2] converges to the initial minority with exponentially small probability. The latter result shows that, although the performance of the protocol of [2] can drop significantly when the interaction graph G is not the complete graph, it is quite robust when G is the complete graph. Our proof concerning the robustness of the protocol of [2] in the complete graph is novel and uses a non-trivial coupling argument which can be of independent interest.

2 The model and notation

A *population protocol* consists of a finite set Q of states/types[5], a finite set of input symbols X, an input function $\iota : X \to Q$, a finite set of output symbols Y, an output function $\gamma : Q \to Y$, and a joint transition function $\delta : Q \times Q \to Q \times Q$. If, for any pair of states $q_1, q_2 \in Q$, $\delta(q_a, q_b) = (q'_a, q'_b)$ implies that $\delta(q_b, q_a) = (q'_b, q'_a)$, then the population protocol is called *symmetric*. A population protocol is executed by a fixed finite *population* of agents with types in Q. We assume

[5] In the original formulation of population protocols these are called *states*, but we chose to also use the term *type* to avoid confusion with the states in a Markov chain.

that each agent has an identity $v \in V$, but agents are oblivious to their own identity and to identities of agents they interact with.

Initially, each agent is assigned a type according to an *input* $x : V \to X$ that maps agent identities to input symbols. In the general population protocol model, agents are identified with the vertices of an *interaction graph*, whose edges indicate the possible agent interactions that may take place. Here, an interaction graph is a simple connected graph G (i.e. without loops or multiple edges), which can be directed or undirected. A function $C : V \to Q$ is called a *configuration*. We will say that the population protocol reaches configuration C at time t if, for every agent v in the population, the state of v at time t is $C(v)$.

In the original model, agents do not send messages or share memory; instead, an interaction between two agents updates both of their types according to a joint transition function (which can be also represented by a table). Interactions between agents are planned by a scheduler under a general "fairness" condition; the actual mechanism for choosing which agents interact is abstracted away. The fairness condition states that for any two configurations C, C', if C occurs infinitely often and C' is reachable from C, then C' also occurs infinitely often.

In this paper, we consider a special case of a fair scheduler, namely the *probabilistic scheduler*, which is defined on directed graphs as follows. During each execution step, a directed edge $(v, u) \in E$ is chosen uniformly at random from E, where v (i.e. the tail of (v, u)) is called the *initiator* and u (i.e. the head of (v, u)) is called the *responder* of the interaction. Then, agents v and u update their types jointly according to δ. In particular, if v is of type q_v and u is of type q_u, the type of v (respectively u) becomes q'_v (respectively q'_u), where $(q'_v, q'_u) = \delta(q_v, q_u)$. The types of all other agents remain unchanged. The probabilistic scheduler is defined on undirected graphs similarly, by replacing every undirected edge $\{v, u\}$ by the two directed edges (v, u) and (u, v). That is, in an undirected graph G, the probabilistic scheduler selects first an undirected edge uniformly at random and then it selects equiprobably one of its endpoint as the initiator. Note that a symmetric protocol does not distinguish between initiators and responders. Thus, if the protocol is symmetric, the probabilistic scheduler on undirected graphs just chooses at each execution step one undirected edge uniformly at random and lets its endpoints interact according to the transition function.

Given the probabilistic scheduler, a population protocol *computes* a (possibly partial) function $g : X^V \to Y$ with error probability at most ϵ, if for all $x \in g^{-1}(Y)$, the population eventually reaches a configuration C that satisfies the following properties with probability at least $1 - \epsilon$: (a) all agents agree on the correct output, i.e. $g(x) = \gamma(C(v))$ for all $v \in V$ and (b) this holds also for every configuration reachable from C. A population protocol *stably computes* a (possibly partial) function $g : X^V \to Y$ if, for *every* fair scheduler, the population eventually reaches a configuration C that satisfies both properties (a) and (b).

Observation 1 *If a symmetric population protocol stably computes a function on an undirected interaction graph G, then it also stably computes the same function on a directed interaction graph G' that comes from G by assigning to every edge of G one or two directions.*

2.1 Majority with high probability on the clique

Angluin et al. [2] proposed a population protocol for computing majority with high probability (whp) in the case where the interaction graph is a clique. Their protocol uses just 3 types $Q = \{b, g, r\}$. For convenience, we will sometimes refer to these types as the *blank, green* and *red* type respectively. The joint transition function δ is given by:

$$\delta(x, y) = \begin{cases} (x, x), & \text{if } x = y \text{ or } y = b \\ (x, b), & \text{if } (x, y) \in \{(g, r), (r, g)\}. \end{cases} \tag{1}$$

One of the main results in [2] is that if the underlying interaction graph is a clique K_n (i.e. the complete graph on n vertices) and interactions are planned according to the above probabilistic scheduler, then with high probability $1 - o(1)$, the above 3-type majority protocol converges to the initial majority value if the initial difference between the majority and the minority is $\omega(\sqrt{n} \log n)$.

2.2 Representation

Using the probabilistic scheduler to plan agent interactions has the advantage that we can describe evolution by using a discrete time Markov chain $\mathcal{M}$. For general interaction graphs, the state space $\mathcal{S}$ of $\mathcal{M}$ can have up to $|V|^{|Q|}$ states, namely one for each configuration $C : V \to Q$. Specifically for the model of [2], we denote by W_t (respectively R_t and G_t) the set of agents in state b (respectively r and g) at time t. Note that if the interaction graph has a high degree of symmetry, then $\mathcal{S}$ can be reduced significantly. One such example is the clique K_n, in which case we can describe a state of $\mathcal{M}$ by the tuple $(|R_t|, |G_t|)$ (where we have also used the fact that $|W_t| = n - |G_t| - |R_t|$).

3 At least 4 types are needed for majority

Definition 1 (rank). *For any population protocol P, denote by $Q(P)$ the set of types used by P. Let $\mathcal{P}_g$ be a class of population protocols that stably compute the function g. The* rank *of $\mathcal{P}_g$ is*

$$R(\mathcal{P}_g) \stackrel{def}{=} \min_{P \in \mathcal{P}_g} |Q(P)|. \tag{2}$$

In this section, we prove that 3 states are not sufficient to stably compute the majority function in any 2-type population and for any interaction graph.

Theorem 1. *Let $\mathcal{P}_{majority}$ be the class of population protocols that stably compute the majority function in any 2-type population of agents and for any interaction graph. Then $R(\mathcal{P}_{majority}) > 3$.*

4 Computing majority in arbitrary interaction graphs

In this section we introduce a symmetric population protocol with 4 states (called the *ambassador protocol*) which, given an *arbitrary* undirected graph $G = (V, E)$ as the underlying interaction graph of the population, *stably computes* the majority of the types of the vertices of G (even if the majority differs only by one from the minority). Assuming that the input symbols are g (for green) and r (for red), the set of states in the ambassador protocol is $Q = \{(g, 0), (g, 1), (r, 0), (r, 1)\}$. The input function ι is such that $\iota(g) = (g, 1)$ and $\iota(r) = (r, 1)$. The output function γ is such that $\gamma((g, i)) = g$ and $\gamma((r, i)) = r$, where $i \in \{0, 1\}$. Finally, for simplicity of the presentation, we present the transition function δ in the form of a table in Figure 1.

$u \setminus v$	$(g,0)$	$(g,1)$	$(r,0)$	$(r,1)$
$(g,0)$	$-$	$((g,1),(g,0))$	$-$	$((r,1),(r,0))$
$(g,1)$	$((g,0),(g,1))$	$-$	$((g,0),(g,1))$	$((g,0),(r,0))$
$(r,0)$	$-$	$((g,1),(g,0))$	$-$	$((r,1),(r,0))$
$(r,1)$	$((r,0),(r,1))$	$((r,0),(g,0))$	$((r,0),(r,1))$	$-$

Fig. 1. The transition matrix of the ambassador model.

The transition matrix of Figure 1 can be interpreted as follows. Every row and every column is labeled by a state of Q. The cell that belongs to the row labeled with state $q_1 \in Q$ and to the column labeled with state $q_2 \in Q$ has either the symbol "$-$" or an ordered pair of states $(q_1', q_2') \in Q \times Q$. In the cases where this cell has the ordered pair (q_1', q_2'), then $\delta(q_1, q_2) = (q_1', q_2')$, otherwise $\delta(q_1, q_2) = (q_1, q_2)$. Note that the ambassador protocol is defined on undirected interaction graphs, i.e. in every interaction there is no initiator or responder. That is, whenever $\delta(q_1, q_2) = (q_1', q_2')$ then $\delta(q_2, q_1) = (q_2', q_1')$.

The main idea of the ambassador protocol can be described as follows. The first component of the state of a vertex u denotes the color of u. That is, whenever u is at state (q, i) (resp. (r, i)), where $i \in \{0, 1\}$, this is interpreted as "u is currently colored green (resp. red)". Furthermore, if the second component of the state of u is 1 (resp. 0), this is interpreted as "u has an ambassador" (resp. "u has no ambassador"). Whenever two vertices u and v interact during the execution of the protocol, a "battle" takes place between the colors of u and v, where the "ambassadors" of u and v (whenever they exist) try to spread their own color to the other vertex, in the following sense:

- Assume that u and v have different colors. If both u and v have an ambassador, then they both keep their own colors in the next step, but both their ambassadors disappear. If u has an ambassador and v has no ambassador, then v takes the color of u in the next step and the ambassador now moves from u to v. If both u and v have no ambassador then their state remains the same at the next step.

– Assume that both u and v have the same color; then they both maintain their color in the next step. If they both have (or if they both do not have) an ambassador, then their state remains the same at the next step. If one of them (say u) has an ambassador and the other one (say v) does not have one, then the ambassador moves from u to v.

The correctness of the ambassador protocol under the assumption of an *arbitrary* fair scheduler and its efficiency under the probabilistic scheduler is proved in the next two theorems.

Theorem 2. *Given any (un)directed graph, if there exists initially a majority, then the* 4*-state ambassador protocol stably computes the initial majority value.*

Theorem 3. *Let G be an arbitrary connected undirected interaction graph with n vertices. Assuming the probabilistic scheduler, if initially there are k red vertices and $\ell \neq k$ green vertices, then the expected time until the* 4*-state ambassador protocol converges is $O(n^6)$. If, additionally, the interaction graph is the complete graph K_n, then the expected time until the protocol converges is $O\left(\frac{\ln n}{|k-\ell|} n^2\right)$.*

From Theorem 3, we can see that the running time of our 4-state protocol in the case where the difference between the majority and the minority is $\Theta(n)$ is $O(n \ln n)$, which is comparable to the running time of the fast 3-state protocol of [2].

5 The model of Angluin et al. on arbitrary graphs

In this section, we provide a detailed analysis of the 3-state protocol of [2] on arbitrary interaction graphs G. In particular, in Subsection 5.1, we present our result concerning the random initial placement of individuals on the vertices of the interaction graph. In Subsection 5.2, we prove our auxiliary result that, when the minority is sufficiently small, the probability that the protocol of Angluin et al. fails in computing the majority value is exponentially small. Although this result shows the robustness of the protocol of [2] in the clique, we use it as an intermediate step in proving in Subsection 5.3 that there exists a family of graphs in which the protocol can fail with high probability. Finally, in Subsection 5.4, we prove the existence of a family of graphs in which the protocol of [2] can take an exponential expected number of steps to reach consensus.

5.1 Random initial placement

In the next theorem we provide a sufficient condition under which the majority protocol described in [2] correctly identifies the initial majority with probability at least $\frac{1}{2}$. This result is in wide contrast to the negative result of Subsection 5.3 (cf. Theorem 6), in which we highlight a case where the majority protocol of [2] fails with high probability. The proof of the next theorem is based on Hall's marriage Theorem (cf. chapter 5, [9]):

Theorem 4. *For any strongly connected directed graph G, if the initial assignment of individuals to the vertices of G is random, then the majority protocol described in [2] correctly identifies the initial majority with probability at least $\frac{1}{2}$.*

5.2 Clique

By the discussion in Subsection 2.2, the state space of the Markov chain $\mathcal{M}$ describing the evolution of the protocol at time t contains tuples of the form $(|R_t|, |G_t|)$, where R_t (resp. G_t) is the set of vertices of type r (resp. g) at time t. In particular, in this subsection we are interested in upper bounding $\Pr\{\text{absorption at } (n,0) | \text{initially at } (\epsilon n, n-\epsilon n)\}$. The core of our proof lies in the definition of two discrete time processes $\mathcal{W}$ and $\mathcal{C}$ that "filter" the information from the original Markov chain $\mathcal{M}$.

Definition 2 (The Blank Process $\mathcal{W}$). *This process keeps track of the number of blank vertices over time, i.e. $\mathcal{W}(t) \stackrel{def}{=} \langle \#$ vertices of type b at time $t \rangle$.*

For convenience, we will use the following notation to describe transitions of $\mathcal{M}$: We will write $g \to r$ to describe a transition of the form $(x,y) \to (x-1,y)$, for some $x, y \in \{1, 2, \ldots, n\}$. More specifically, $g \to r$ is used to describe a transition where a directed edge (v,u) is chosen by the scheduler, v is of type g, and u is of type r. Similarly, we will use $r \to g$ for transitions of the form $(x,y) \to (x,y-1)$, $g \to b$ for transitions of the form $(x,y) \to (x,y+1)$ and $r \to b$ for transitions of the form $(x,y) \to (x+1,y)$. We note that the state of $\mathcal{M}$ at any time t can be fully described by the initial state and by a sequence of transitions among $\{g \to r, r \to g, g \to b, r \to b\}$.

Definition 3 (The Contest Process $\mathcal{C}$). *Transitions of $\mathcal{M}$ are paired recursively, starting from time 0 as follows: Every transition that increases the number of blanks is paired with the earliest subsequent transition that decreases the number of blanks and is not paired yet.* [6] *For an arbitrary time t, we denote by $\tau(t)$ (or just τ for short) the number of pairs until time t. The Contest Process $\mathcal{C}$ is defined over the time scale τ, where $\mathcal{C}(0) = |R_0|$ and for $\tau = 1, 2, \ldots,$*

$$\mathcal{C}(\tau) = \begin{cases} \mathcal{C}(\tau-1)+1, & \text{if the } \tau\text{-th pair is } (r \to g, r \to b), \\ \mathcal{C}(\tau-1)-1, & \text{if the } \tau\text{-th pair is } (g \to r, g \to b) \text{ and} \\ \mathcal{C}(\tau-1), & \text{otherwise.} \end{cases} \tag{3}$$

Notice that the processes $\mathcal{W}$ and $\mathcal{C}$ are dependent. As a matter of fact, $\mathcal{C}$ is not even defined using the same time scale as $\mathcal{W}$ and $\mathcal{M}$ (to indicate this, we have used the convention that t is the time variable for processes $\mathcal{M}, \mathcal{W}$, while τ is the time variable for process $\mathcal{C}$). However, observe that if we initially begin with no blanks (i.e. $|R_0| + |G_0| = n$ hence $\mathcal{W}(0) = 0$), then whenever $\mathcal{W}$ decreases its value, we have a transition step of $\mathcal{C}$.

[6] We assume that the pairing concerns only transitions that change the state of $\mathcal{M}$. In particular, transitions of the form $b \to r, b \to g, b \to b, g \to g$ and $r \to r$ are ignored in this pairing as irrelevant.

Lemma 1 (Relating $\mathcal{C}$ and $\mathcal{M}$). *For any $T \in \mathbb{N}$, denote by $\mathcal{C}_{|T}$ the value of $\mathcal{C}$ given only states $\mathcal{M}(t), t = 0, 1, \ldots, T$ (i.e. given the history of $\mathcal{M}$ up to time T). Then, $\mathcal{C}_{|T} \geq |R_T|$ for any $T \in \mathbb{N}$. Furthermore, if $\mathcal{C}_{|T} = 0$, then all vertices are of type g.*

Lemmas 2 and 3 below concern the domination of processes $\mathcal{W}$ and $\mathcal{C}$ by appropriate birth-death processes and are used in the proof of Theorem 5.

Lemma 2 (Domination of $\mathcal{W}$). *Let $\alpha, \beta, \kappa \in \{1, \ldots, n-1\}$, with $\alpha < \beta$. Let also $\mathcal{B}_{\mathcal{W}}$ be a birth-death process, which has state space $\mathcal{S}_{\mathcal{B}_{\mathcal{W}}} = \{S_0, \ldots, S_n\}$, with S_n an absorbing state and transition probability matrix P, with $P(S_i, S_{i+1}) = 1$ for all $i \in \{0, \ldots, \alpha\} \cup \{\beta, \ldots, n-1\}$, $\frac{P(S_i, S_{i+1})}{P(S_i, S_{i-1})} = \frac{2\kappa}{\alpha}$ for all $i \in \{\alpha+1, \ldots, \beta-1\}$ and $P(S_i, S_i) = \Pr(\mathcal{W}(t) = i | \mathcal{W}(t-1) = i)$, for all $t \geq 1$ and for all $i \in \{\alpha + 1, \ldots, \beta-1\}$. Then, given that the vertices of type r are at most κ, the process $\mathcal{W}$ is stochastically dominated by $\mathcal{B}_{\mathcal{W}}$ in the following sense: $\Pr(\mathcal{W}(t) > x | \mathcal{W}(0) = 0) \leq \Pr(\mathcal{B}_{\mathcal{W}}(t) \in \cup_{y>x} S_y | \mathcal{B}_{\mathcal{W}}(0) = S_0)$, for any time t and $x \in \{0, \ldots, n\}$.*

Lemma 3 (Domination of $\mathcal{C}$). *Let β, κ be positive integers, with $\beta + \kappa < n$. Let also $\mathcal{B}_{\mathcal{C}}$ be a birth-death process, which has state space $\mathcal{S}_{\mathcal{B}_{\mathcal{C}}} = \{T_0, \ldots, T_n\}$, with T_0, T_n absorbing states and transition probability matrix Q, with $Q(T_i, T_{i+1}) = 1$ for all $i \in \{\kappa, \ldots, n-1\}$, $\frac{Q(T_i, T_{i+1})}{Q(T_i, T_{i-1})} = \frac{\kappa}{n-\beta-\kappa}$ for all $i \in \{1, \ldots, \kappa - 1\}$ and $Q(T_i, T_i) = \Pr(\mathcal{C}(\tau) = i | \mathcal{C}(\tau - 1) = i)$, for all $\tau \geq 1$ and for all $i \in \{1, \ldots, \kappa - 1\}$. Then, given that the vertices of type b are at most β, the process $\mathcal{C}$ is stochastically dominated by $\mathcal{B}_{\mathcal{C}}$ in the following sense: $\Pr(\mathcal{C}(\tau) > x | \mathcal{C}(0) = |R_0|) \leq \Pr(\mathcal{B}_{\mathcal{C}}(\tau) \in \cup_{y>x} T_y | \mathcal{B}_{\mathcal{C}}(0) = |R_0|)$, for any τ and $x \in \{0, \ldots, n\}$.*

The main Theorem of this subsection is stated below.

Theorem 5. *Let $\epsilon < \frac{1}{7}$. For large enough n, starting from ϵn agents of type r and $(1 - \epsilon)n$ agents of type g on the clique K_n, the probability that the clique eventually contains only agents of type r is at most $e^{-\Theta(n)}$.*

We note here that the upper bound on ϵ in the statement of Theorem 5 is only used to facilitate exposition of our arguments in the proof. We claim that this upper bound can be increased further by using the same proof ideas, but that we cannot reach arbitrarily close to $\frac{1}{2}$. However, we conjecture that the constant ϵ in Theorem 5 can be as as large as $\frac{1}{2} - \epsilon'$, where $\epsilon' > 0$ is a small constant. We also conjecture that starting from a single agent of type r on the clique K_n, the probability that the clique eventually contains only agents of type r is at least $e^{-\Theta(n)}$.

5.3 Minority domination

Consider the lollipop graph, which consists of a complete graph K_{n_1} on n_1 vertices, among which vertex v is connected to the leftmost vertex u of a line graph L_{n_2} on n_2 vertices, with $n_1 + n_2 = n$. In this subsection we prove that the protocol of [2] may fail with high probability in the lollipop graph.

Theorem 6. *Consider a lollipop graph on n vertices, which consists of a complete graph K_{n_1} on $n_1 \geq 100 \ln n$ vertices, among which vertex v is connected to the leftmost vertex u of a line graph L_{n_2} on $n_2 = n - n_1$ vertices. Suppose that initially vertex v and all vertices in L_{n_2} are of type r, while all vertices in $K_{n_1} \backslash \{v\}$ are of type g. Then with high probability, eventually only vertices of type g will remain in the graph.*

The proof builds upon the proof of Theorem 5 and also uses the following fact: given a line graph L_m, in which the leftmost vertex is of type g and all other $m-1$ vertices are of type r, the probability that the protocol of [2] eventually reaches a configuration in which all vertices are of type g is $\frac{1}{2(m-1)}$. For the proof of Theorem 6, we define stochastic processes $\mathcal{W}'$ and $\mathcal{C}'$ as the processes $\mathcal{W}$ and $\mathcal{C}$ from Subsection 5.2, but concerning the clique K_{n_1}. These processes take into account only transitions involving either edges that belong to the clique or the directed edge (u, v). We note that Lemma 1 continues to hold for the modified process $\mathcal{C}'$ with only one modification (because of the existence of the edge $\{v, u\}$) concerning its second part: T must satisfy ${\mathcal{C}'}_{|T} = 0$ and ${\mathcal{C}'}_{|T-1} > 0$ in order to be able to deduce that K_{n_1} has only vertices of type g.

5.4 Expected exponential time to absorption

In this subsection we prove that the protocol of [2] can take an expected exponential time of steps to reach consensus in the case where the underlying graph consists of two disjoint cliques on n_1 and n_2 vertices each with a single edge between them (see Figure 2(a)). The main part of the proof is dedicated in proving an upper bound on the expected number of steps needed for the protocol to reach consensus in the case of a directed graph H, consisting of a single clique K_{n_1} and a vertex v outside the clique, which is connected to a vertex u of the clique with a directed edge (v, u) (see Figure 2(b)). In particular, similarly to Subsection 5.3, we define stochastic processes $\mathcal{W}''$ and $\mathcal{C}''$ as the processes $\mathcal{W}$ and $\mathcal{C}$ from Subsection 5.2, but concerning the clique K_{n_1}. We then prove suitably modified versions of Lemmas 2 and 3 and we apply results on birth-death processes.

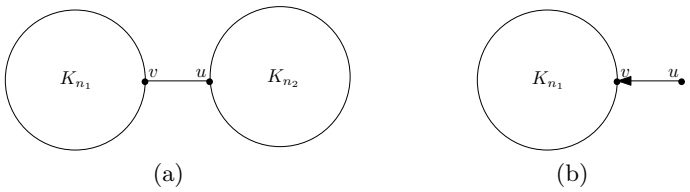

Fig. 2. (a) An interaction graph G with 2 disjoint K_{n_1} and K_{n_2}, connected with a single edge $\{u, v\}$. (b) The graph H consisting of a clique K_{n_1} with an arc (u, v).

Theorem 7. *Let G be an interaction graph on n vertices, which consists of 2 disjoint cliques on n_1 and n_2 vertices each and a single edge between them (see Figure 2(a)). Suppose that initially, all vertices in the K_{n_1} clique are of type g, while all vertices in the K_{n_2} clique are of type r. Then the majority protocol of [2] needs an expected exponential number of steps to reach consensus.*

References

1. D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, 18:235–253, 2006.
2. D. Angluin, J. Aspnes, and D. Eisenstat. A simple population protocol for fast robust approximate majority. *Distributed Computing*, 21(2):87–102, 2008.
3. J. Aspnes and E. Ruppert. An introduction to population protocols. In B. Garbinato, H. Miranda, and L. Rodrigues, editors, *Middleware for Network Eccentric and Mobile Applications*, pages 97–120. Springer-Verlag, 2009.
4. M. Cook, D. Soloveichik, E. Winfree, and J. Bruck. Programmability of chemical reaction networks. In A. Condon, D. Harel, J. N. Kok, A. Salomaa, and E. Winfree, editors, *Algor. Bioprocesses*, Nat. Comp. Series, pages 543–584. Springer, 2009.
5. G. De Marco and A. Pelc. Randomized algorithms for determining the majority on graphs. *Combinatorics, Probability and Computing*, 15(6):823–834, 2006.
6. M. H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.
7. M. Fischer and H. Jiang. Self-stabilizing leader election in networks of finite-state anonymous agents. In *Proceedings of the 10th International Conference on Principles of Distributed Systems (OPODIS)*, 2006.
8. R. A. Holley and T. M. Liggett. Ergodic theorems for weakly interacting infinite systems and the voter model. *The Annals of Probability*, 3:643–663, 1975.
9. S. Jukna. *Extremal Combinatorics with Applications to Computer Science*. Springer, 2011.
10. M. Kearns and J. Tan. Biased voting and the democratic primary problem. In *Proceedings of the 4th Workshop on Internet and Network Economics (WINE)*, pages 639–652. 2008.
11. T. G. Kurtz. *Approximation of Population Processes*. Society for Industrial and Applied Mathematics, 1987.
12. L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.
13. T. M. Liggett. *Interacting Particle Systems*. Springer-Verlag, 2004.
14. A. Mizrachi. Majority vote and monopolies in social networks. Master's thesis, Department of Communication Systems Engineering, Faculty of Engineering, Ben-Gurion University of the Negev, 2013.
15. P. A. P. Moran. Random processes in genetics. *Mathematical Proceedings of the Cambridge Philosophical Society*, 54(1):60–71, 1958.
16. E. Mossel, J. Neeman, and O. Tamuz. Majority dynamics and aggregation of information in social networks. *Autonomous Agents and Multi-Agent Systems*, 28(3):408–429, 2014.
17. F. P. Preparata, G. Metze, and R. T. Chien. On the connection assignment problem of diagnosable systems. *IEEE Trans. on Electr. Computers*, 16:848–854, 1967.
18. M. Saks and M. Werman. On computing majority by comparisons. *Combinatorica*, 11(4):383–387, 1991.