Two-agent Approximate Agreement from an Epistemic Logic perspective

Jorge Armenta-Segura[2] Sergio Rajsbaum[1,3]

Instituto de Matemáticas
Universidad Nacional Autónoma de México
Ciudad Universitaria, CDMX 04510, Mexico

Jérémy Ledent[4]

MSP group, Computer and Information Sciences
University of Strathclyde
Glasgow, Scotland

Abstract

We investigate the two agents approximate agreement problem in a dynamic network in which topology may change unpredictably, and where consensus is not solvable. It is known that the number of rounds necessary and sufficient to guarantee that the two agents output values $1/k^3$ away from each other is k. We distil ideas from previous papers to provide a self-contained, elementary introduction, that explains this result from the epistemic logic perspective.

Keywords: Distributed algorithm, approximate consensus, fault-tolerance, epistemic logic.

1 Introduction

Problems of reaching agreement are central to distributed computing. Often *consensus* is needed to agree on the same value, for example, when agents need to agree on whether to commit or abort the results of a distributed database transaction. However, in many situations consensus is impossible to achieve, such as networks where agents may crash and delays are unpredictable [17], or read/write shared memory models [25], or certain dynamic networks [11]. All these impossibility results are due to the same reason: the indistinguishability graph [2] of the global states of a protocol trying to solve consensus is connected, while that of the consensus task is not [5,21,26].

In this paper we are interested in many other situations, where *approximate agreement* is sufficient; for example, when sensors estimate a certain measurement or clock synchronization where agents maintain similar time estimates. Many variants of approximate agreement, and in various message passing and shared memory models have been considered since early on, e.g. [15]. Approximate agreement is an interesting weakening of consensus that can be solved in many more situations. The task is parametrised by a real number $\varepsilon > 0$, and the agents must decide values that are at most ε away from each other. The time to reach a decision however, will be a function of how small ε is. Many algorithms have been proposed to try to minimize the time until decision is reached, e.g. in shared memory models [23], in networks [10], and others.

[1] Partially supported by UNAM-PAPIIT grant IN106520.

[2] Email: jesusarmenta@ciencias.unam.mx

[3] Email: rajsbaum@im.unam.mx

[4] Email: jeremy.ledent@strath.ac.uk

Consensus has been thoroughly studied from the epistemic logic perspective, and the close connection with common knowledge is well-known [16]. Approximate agreement has been less studied from the epistemic logic perspective. It was recently shown that it is closely related to k-iterated knowledge in a certain shared memory model [20], roughly, meaning that the agents must know, that they know, that they know that they know, and so forth (k times), each other's inputs, in order to reach certain degree of approximation of their decisions. This was shown within a general simplicial complex model for multi-agent epistemic logic for task solvability. In this paper we are interested in studying in more detail this result, as well as providing a self-contained, more elementary introduction to the topic.

We focus on one specific distributed computing model, that has received much attention recently, *dynamic networks* (see surveys [6,8,24,27]). There is a fixed set of agents that operate in rounds communicating by sending messages to each other. Rounds are synchronous in the sense that the messages received at some round have been sent at that round. At each round, the communication graph is chosen arbitrarily among a set of directed graphs, $\mathcal{G}$. This set $\mathcal{G}$ determines the network model. Hence the communication graph can change unpredictably from one round to the next. Dynamic networks are interesting for various reasons, in particular because they include as special cases other classic models, such as shared memory models [22], so results in dynamic networks can often be extrapolated to other models.

Approximate agreement is solvable in a dynamic network model $\mathcal{G}$ if and only if each communication graph in $\mathcal{G}$ has a rooted spanning tree [9] (e.g. [5,7]). In this work we consider such a $\mathcal{G}$, where approximate agreement is solvable, but not consensus. Two agents, starting with binary input values, are to decide values in the interval of their inputs, which are at most $1/3^k$ apart from each other [5]. We rephrase this problem epistemically, as requiring that the agents reach k-iterated knowledge about their respective inputs. This knowledge can be achieved, if and only if, the protocol runs for k rounds. We show these results in the dynamic epistemic logic (DEL) [4] framework of [20], instantiated to a dynamic network. We follow recent work, in using the dual of a Kripke graph, a simplicial complex, as a model for multi-agent epistemic logic, that exposes the topological features that determine if a task is solvable [19,13]. While the results we present here have been shown in these papers, we instantiate the results using graphs instead of simplicial complexes, making the development more accesible.

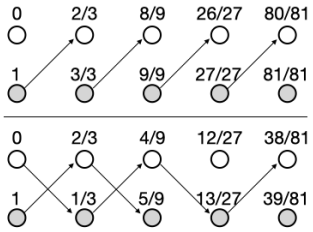
Organization. Before getting into epistemic logic, in Section 2 we present two algorithms that solve approximate agreement. In Section 3 we overview the DEL framework. In Section 4 we give the formal epistemic logic semantics to our dynamic network model, and we do the same for tasks, in Section 5. The solvability results are in Section 6. The conclusions are in Section 7. Additional details and proofs are in the Appendix.

2 Approximate agreement algorithms

We start by presenting algorithms, to guide the reader's intuition before diving into the formal semantics considerations of the subsequent sections.

Two agents, g, w (for gray, white), communicate with each other exchanging messages in a sequence of k rounds. In a *round,* each agent sends a message to the other agent, receives the message sent by the other agent, and then updates its local state. In each round, at most one of the two messages sent may be lost.

Let $N = 3^k$, $k \geq 0$. In the N-*approximate agreement* task, after k rounds, agents decide values d_g, d_w, resp., of the form i/N, for an integer i, $0 \leq i \leq N$. Each agent starts with an input value from the set $\{0, 1\}$. The decisions must be such that if the two input values are equal, then both output values are equal to the input value. Otherwise, the output values d_g, d_w satisfy $|d_g - d_w| \leq 1/N$. The algorithm in Figure 1 solves N-approximate agreement in k rounds (e.g [10,18],[21][Chapter 2]). A simple induction argument on the number of rounds proves its correctness. The examples on the right show two 4-round executions where agents agree on values $1/3^4 = 1/81$ away from each other. The first example (represented in the top two rows) represent an execution where all messages from w are lost. The first column, with labels 0 and 1, corresponds to the initial input values. Then each subsequent column shows the values of the processes after each round. An edge indicates that a message was received. In the second execution (bottom two rows of the picture), both messages arrive in the first two rounds, but in the 3rd round the message from g to w is lost, and in the 4th round the message from w to g.



The algorithm of Figure 1 can be used by the agents to decide values arbitrarily close to each other, by running enough rounds, k. However, the size of messages sent grows with k. Remarkably, there is an algorithm that solves the problem sending 1-bit messages, see Figure 2, which is a reformulation and small adaptation of

[5] Fixing the constant in ε to $1/3^k$ facilitates the presentation, avoiding collateral details, and can be done without loss of generality.

AVERAGING N-APPROXIMATE AGREEMENT (ℓ)

```
1  round r from 1 to k do
2      send(ℓ)
3      m = receive()                          /* receive m ∈ {0, 1, ⊥} */
4      if m ≠ ⊥ then
5          ℓ = ℓ/3 + 2m/3                     /* else ℓ does not change */
6  ouput ℓ
```

Figure 1. Each agent $p \in \{g, w\}$ runs this code. The input is $\ell \in \{0, 1\}$. A message $m = \perp$ indicates that no message was received from the other process.

the algorithm given in [12]. In a 1-bit protocol, each message received from the other process can be either 0, 1 or $\perp$. Thus, in a k-round computation, we call the *view* of a process the sequence $v \in \{0, 1, \perp\}^k$ of messages that were received. For instance with four rounds, $v = (0, 0, \perp, 1)$ indicates that the process received message "0" in the first two rounds, no message in the third round, and message "1" in the fourth round. To reduce notation, we often write such a view as a word, e.g. $v = 00\perp1$. Given a view v and a message $m \in \{0, 1, \perp\}$, we write $v \cdot m$ for the new view where we append m at the end of v.

The algorithm in Figure 2 below is generic, in the sense that both processes simply collect their view during the computation, and at the very end a decision function δ computes the output depending on the view. To choose which bit to send, at each round, the process counts how many messages were received until now, nbMsg $= \#\{m \in \text{view} \mid m \neq \perp\}$, adds its input $\ell \in \{0, 1\}$, and sends the parity of the result. The computation of δ is done locally at the end of the protocol, and does not involve communication between the processes. It is adapted from [12], except that we deal with all four combinations of inputs in $\{0, 1\}$ instead of fixing in advance one input edge. We use the following facts:

- The first time a process p receives a message, it can immediately guess the input of the other process p'. Indeed, since all previous messages were lost by p, we know for sure that all messages were received by p'.
- Process p can then use this information to "decode" the subsequent messages, and behave like in the algorithm of [12].

ONE-BIT MESSAGES N-APPROXIMATE AGREEMENT (ℓ) /* input $\ell \in \{0, 1\}$ */

```
1  view = ()                                  /* start with empty view */
2  nbMsg = 0                                  /* total number of messages received until now */
3  round r from 1 to k do
4      send((nbMsg + ℓ) mod 2)                /* send the parity of nbMsg + ℓ */
5      m = receive()                          /* receive m ∈ {0, 1, ⊥} */
6      if m ≠ ⊥ then nbMsg = nbMsg + 1        /* update nbMsg */
7      view = view · m                        /* append m to the view */
8  output δ(ℓ, view)                         /* decision depends on the input and the final view */
```

Figure 2. Approximate agreement with 1-bit messages. Each agent runs this code with input $\ell \in \{0, 1\}$; the output function $\delta(\ell, \text{view})$ is detailed in Figure 3.

The rest of the paper is devoted to answering the question: what do the agents learn about their inputs after running protocols like these ones? For these two specific cases, do they learn something different? We will give a formal semantics based on dynamic epistemic logic to answer such questions. And as an application, we will show that these two algorithms are optimal in the number of rounds: in less than k rounds, it is impossible for the agents to produce outputs closer than $1/3^k$. This result has been proved using combinatorial

LOCAL COMPUTATION OF $\delta(\ell, \text{view})$

```
1  other = ⊥                                  /* input value of the other process */
2  d = 0                                      /* distance to move towards the other process */
3  for i = 1 to k do
4      nbMsg = #{x ∈ view[1...(i − 1)] | x ≠ ⊥}
5      m = view[i]
6      if m ≠ ⊥ then                          /* if m = ⊥, keep the same d */
7          if other = ⊥ then
8              other = (m + i + 1) mod 2      /* find the input of the other process */
9              if other = ℓ then return ℓ
10         m = (m + other) mod 2              /* decode the message */
11         if (m = nbMsg and i mod 2 = 1) or (m ≠ nbMsg and i mod 2 = 0)
12             then d = d + 2/3^i
13             else d = d − 2/3^i
14 if ℓ = 0 then return d
15 else return 1 − d
```

Figure 3. Decision function $\delta(\ell, \text{view})$ used at the end of the one-bit algorithm of Figure 2.

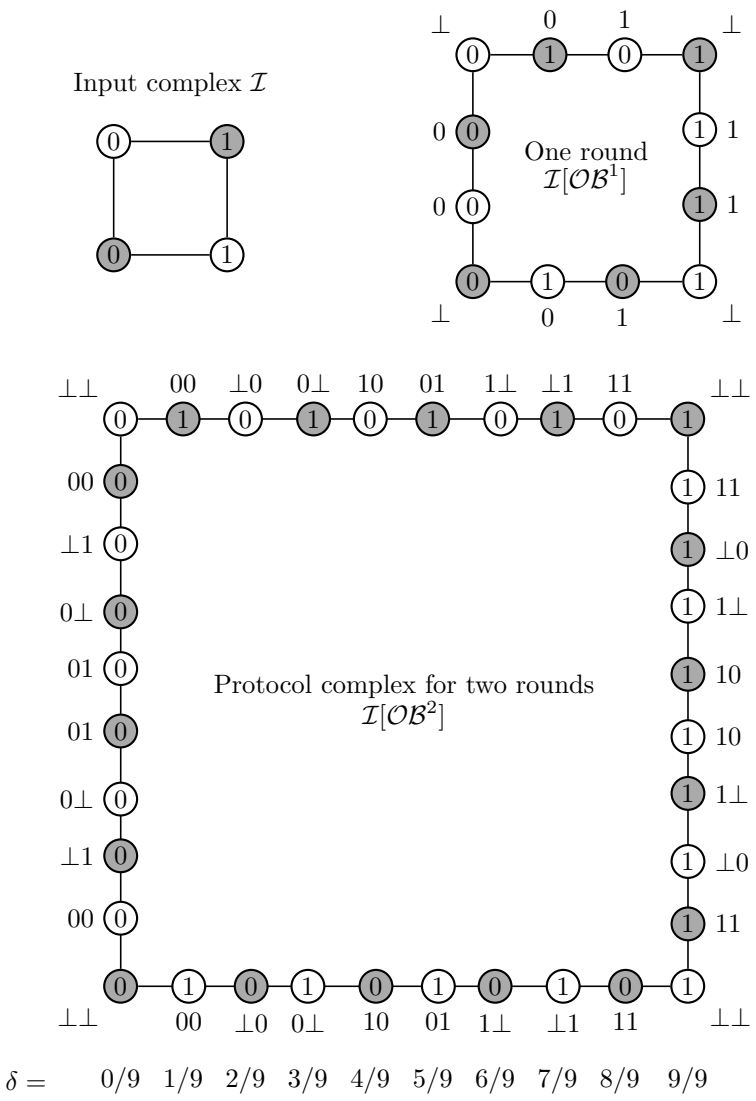


Figure 4. Protocol complex of the 1-bit protocol from Figure 2, for 1 and 2 rounds of computation. Process names are represented by colors; inputs $\ell \in \{0, 1\}$ are written inside the vertices; and views are written next to the vertices. For the two-round protocol, the value of $\delta(\ell, \text{view})$ at each node is given by projecting vertically downwards. These pictures represent protocol complexes in the usual sense of distributed computing, but also in the sense of the *product-update model* construction defined later.

arguments e.g. [18,21], here we will see a different explanation: in less than k rounds, they cannot acquire sufficient knowledge about their inputs. Also, while the two algorithms seem rather different, the agents learn exactly the same information about their inputs in both.

3 Fundamentals about dynamic epistemic logic (DEL)

3.1 A simplicial model for epistemic logic

We describe here the model for epistemic logic, based on chromatic simplicial complexes [20,13]. This reformulates the usual semantics of formulas in Kripke models, in terms of simplicial models. See Appendix B and Theorem B.4.

Syntax

Let AP be a countable set of propositional variables and A a finite set of agents. The language of epistemic logic formulas $\mathcal{L}_{\mathcal{K}}(A, AP)$, or just $\mathcal{L}_{\mathcal{K}}$ if A and AP are implicit, is generated by the following BNF grammar:

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid K_a\varphi \qquad p \in AP,\ a \in A$$

The proof theory of epistemic logics can be found in [14]; in the rest of the paper, we focus on studying models. In the following, the number of agents A is two and we let $A = \{g, w\}$. The agents will be represented as colors, gray and white. We can then use terminology of graphs, a specialization of the case of larger number of agents that requires using simplicial complexes, see Appendix B.

Given a set V, an (undirected) *graph*[6] C over a set of *vertices* V is defined by a non-empty finite set of edges. Each *edge* is a set of two vertices. The set of vertices of C is noted $\mathcal{V}(C)$, and the set of edges $\mathcal{F}(C)$. A *chromatic graph* $\langle C, \chi\rangle$ consists of a graph C and a coloring map $\chi : \mathcal{V}(C) \to A$, such that for all $X \in \mathcal{F}(C)$, the two vertices of X have distinct colors.

Simplicial models.

Let $\mathcal{V}$ be some countable set of values, and $AP = \{p_{a,x} \mid a \in A, x \in \mathcal{V}\}$ be the set of *atomic propositions*. Intuitively, $p_{a,x}$ is true if agent a holds the value x. We write AP_a for the atomic propositions of agent a.

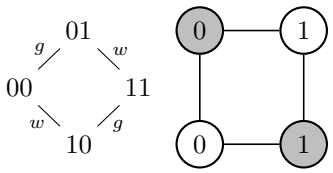
A *simplicial model* $M = \langle C, \chi, \ell\rangle$ consists of a chromatic graph $\langle C, \chi\rangle$, and a labeling $\ell : \mathcal{V}(C) \to \mathcal{P}(AP)$ that associates with each vertex $v \in \mathcal{V}(C)$ a set of atomic propositions concerning agent $\chi(v)$, i.e., such that $\ell(v) \subseteq AP_{\chi(v)}$. Given an edge $X = \{v_0, v_1\} \in C$, we write $\ell(X) = \ell(v_0) \cup \ell(v_1)$.

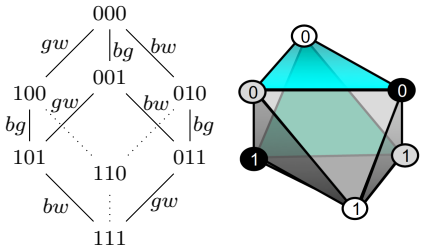
Binary input model.

Each agent gets an input value from the set $\{0, 1\}$. Each agent knows its own input value, but doesn't know which value has been received by the other agents. The figure below from [20] shows the *binary input simplicial model* and its associated Kripke model for two agents, and for comparison also for three agents (although we will not use it in this paper). Notice that every possible combination of 0's and 1's is a possible world.

In the Kripke model, the agents are called b, g, w, and the labeling L of the possible worlds is represented as a sequence of values, e.g., 101, representing the values chosen by the agents b, g, w (in that order). In the 3-agents case, the labels of the dotted edges have been omitted to avoid overloading the picture, as well as other edges that can be deduced by transitivity.

In the simplicial model, agents are represented as colors (black, grey, and white). The labeling ℓ is represented as a single value in a vertex, e.g., "1" in a grey vertex means that agent g has chosen value 1. The possible worlds correspond to edges in the 2-agents case, and triangles in the 3-agents case. Note that the simplicial model depicted on the right, with three agents, is out of the scope of this paper. Indeed, here we only work with two agents, so that the simplicial complex is nothing more than a graph.





Semantics.

The definition below mimics the usual semantics of formulas in Kripke models, reformulated here in terms of simplicial models:

Definition 3.1 We define the satisfaction relation $M, X \models \varphi$ determining when a formula φ is *true* in some epistemic state (M, X). Let $M = \langle C, \chi, \ell\rangle$ be a simplicial model, $X \in \mathcal{F}(C)$ an edge of C and $\varphi \in \mathcal{L}_K$.

[6] Our non-traditional notations, C for a graph and $\mathcal{F}(C)$ for the set of edges of C, come from the general setting for n agents where graphs are replaced by simplicial *complexes*, and edges are replaced by *facets*. This is consistent with the notations of previous papers [20,13].

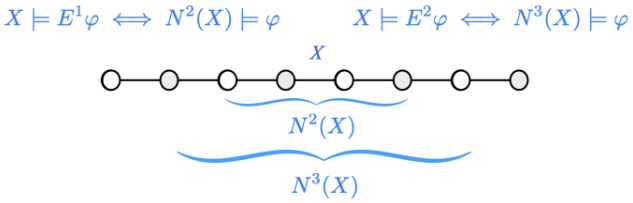
$$
\begin{array}{lll}
M, X \models p & \text{iff} & p \in \ell(X) \\
M, X \models \neg\varphi & \text{iff} & M, X \not\models \varphi \\
M, X \models \varphi \wedge \psi & \text{iff} & M, X \models \varphi \text{ and } M, X \models \psi \\
M, X \models K_a\varphi & \text{iff} & \text{for all } Y \in \mathcal{F}(C), a \in \chi(X \cap Y) \text{ implies } M, Y \models \varphi
\end{array}
$$

Group knowledge.

For a formula φ, we write $E\varphi = K_g\varphi \wedge K_w\varphi$ for the *group knowledge* of φ among the agents $\{g, w\}$. Let E^k denote k nested E operators. An *edge path* is a sequence of (not necessarily distinct) edges, such that each two consecutive edges in the sequence intersect in a vertex. For an edge X let $N^k(X)$ be the set of edges reachable from X by an edge path of at most k edges. Thus, $N^1(X) = \{X\}$, denoted by $N(X)$. Also, $N^2(X)$ is equal to X together with the edges that intersect X, and in general $N^k(X) = N(N^{k-1}(X))$.

Lemma 3.2 *For a simplicial model M, edge X, and any $\varphi \in \mathcal{L}_{\mathcal{K}}$, we have that $M, X \models E^k\varphi$, iff $M, Y \models E\varphi$ for every $Y \in N^k(X)$.*

The figure below is an illustration of the cases $k = 1, 2$. It shows that $M, X \models E\varphi$ states that φ should hold in X and in its two neighboring edges, namely, in the edges that belong to $N^2(X)$, and analogously for $k = 2$.



Morphisms between models

Let C and D be two graphs. A *simplicial map*[7] $f : C \to D$ maps the vertices of C to vertices of D, such that if X is an edge of C, $f(X)$ is an edge of D. A *chromatic simplicial map* between two chromatic graphs is a simplicial map that preserves colors, i.e. $\chi(f(v)) = \chi(v)$ for all v.

Lemma 3.3 *Let $f : C \to D$ be a simplicial map. If C' is a connected subgraph of C then $f(C')$ is a connected subgraph of D.*

A *morphism* of simplicial models $f : M \to M'$ is a chromatic simplicial map that preserves the labeling: $\ell'(f(v)) = \ell(v)$. The next theorem from [20] states that morphisms of simplicial models cannot "gain knowledge".

Lemma 3.4 (knowledge gain) *Consider simplicial models $M = \langle C, \chi, \ell\rangle$ and $M' = \langle C', \chi', \ell'\rangle$, and a morphism $f : M \to M'$. Let $X \in \mathcal{F}(C)$ be an edge of M, a an agent, and $\varphi \in \mathcal{L}_{CK}$ a positive formula, i.e. which does not contain negations except, possibly, in front of atomic propositions. Then, $M', f(X) \models \varphi$ implies $M, X \models \varphi$.*

3.2 Dynamic epistemic logic basic notions

Dynamic epistemic logic (DEL) is the study of modal logics of model change [4,14]. A modal logic studied in DEL is obtained by using action models [3]. An action can be thought of as an announcement made by the environment (not necessarily public). An action model describes all the possible actions that might happen, as well as how they affect the different agents. The product-update operation takes an epistemic model M and an action model $\mathcal{A}$, and creates a new model $M[\mathcal{A}]$ that describes all the new possible worlds after an action from $\mathcal{A}$ has occurred in M. This classic version is described in Appendix D, here we present the simplicial model version [20,13].

[7] A simplicial map is usually called a homomorphism in graph theory.

Action models

A *simplicial action model* $\langle T, \chi, \mathsf{pre}\rangle$ consists of a chromatic graph $\langle T, \chi\rangle$, where the edges $\mathcal{F}(T)$ represent communicative *actions*, and pre assigns to each edge $X \in \mathcal{F}(T)$ a precondition formula $\mathsf{pre}(X)$ in $\mathcal{L}_{\mathcal{K}}$.

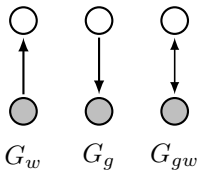
Given two chromatic graphs C and T of dimension n, the Cartesian product $C \times T$ is the following chromatic graph. Its vertices are of the form (u, v) with $u \in \mathcal{V}(C)$ and $v \in \mathcal{V}(T)$ such that $\chi(u) = \chi(v)$; the color of (u, v) is $\chi((u, v)) = \chi(u) = \chi(v)$. Its edges are of the form $X \times Y = \{(u_0, v_0), \ldots, (u_k, v_k)\}$ where $X = \{u_0, \ldots, u_k\} \in C$, $Y = \{v_0, \ldots, v_k\} \in T$ and $\chi(u_i) = \chi(v_i)$.

Let $M = \langle C, \chi, \ell\rangle$ be a simplicial model, and $\mathcal{A} = \langle T, \chi, \mathsf{pre}\rangle$ be a simplicial action model. The *product update simplicial model* $M[\mathcal{A}] = \langle C[\mathcal{A}], \chi[\mathcal{A}], \ell[\mathcal{A}]\rangle$ is a simplicial model whose underlying graph is a sub-graph of the Cartesian product $C \times T$, induced by all the edges of the form $X \times Y$ such that $\mathsf{pre}(Y)$ holds in X, i.e., $M, X \models \mathsf{pre}(Y)$. The valuation $\ell : \mathcal{V}(C[\mathcal{A}]) \to \mathcal{P}(AP)$ at a pair (u, v) is just as it was at u: $\ell[\mathcal{A}]((u, v)) = \ell(u)$.

4 An action model for dynamic networks

Distributed computing model

For two processes, the model $\mathcal{G}$ where approximate agreement is solvable, but not consensus, is unique. It consists of the three digraphs G_g, G_w, G_{gw} on two vertices, g, w, one with the antiparallel arrows, and the other two with an arrow from one vertex to the other, meaning that either both messages arrive, or only one. It is easy to see that in any submodel of $\mathcal{G}$ consensus can be solved (if it has at least one arrow). The model $\mathcal{G}$ is equivalent to several message passing and shared memory models that have been previously considered, see e.g. [1,21].



Each agent has some input value, and they communicate r rounds. In each round an agent sends a message to the other agent, and the message arrives by the end of the round, or not at all. Which messages arrive depends on which graph from $\mathcal{G}$ is selected (it is convenient to assume that each vertex of a graph of $\mathcal{G}$ has a loop; an agent learns its own messages. But we do not draw the loops in the figures). In every round, any of the three graphs in $\mathcal{G}$ can be selected.

Input model

An *input model*, $\mathcal{I}$, represents all the possible input values. To illustrate the ideas, we use the input simplicial model of Section 3.1 where two agents $A = \{g, w\}$ each has an input value from the set $\{0, 1\}$.

Actions

An *action*[8] $t \in \hat{T}$ is given by b_0, b_1, c, where c is a sequence of r digraphs from $\mathcal{G}$ and $b_0, b_1 \in \{0, 1\}$ are binary values, representing the inputs of both processes. Such an action is written $c^{b_0 b_1}$. The precondition $\mathsf{pre}(c^{b_0 b_1})$ of this action is a formula expressing the fact that the inputs of the agents g, w are respectively b_0, b_1. Formally, if input_a^x denotes the atomic proposition expressing that agent a has input value x, then $\mathsf{pre}(c^{b_0 b_1}) = \mathsf{input}_g^{b_0} \wedge \mathsf{input}_w^{b_1}$.

Distributed algorithm

An action $t = c^{b_0 b_1}$ includes all the information about the inputs and message deliveries of an execution, but not about the actual algorithm being executed by the agents. Namely, an algorithm specifies the contents of messages sent in each round, and the state transition of each agent. Each agent starts in an initial state, determined by its input value. At the end of a round, each agent changes to a new state following a deterministic transition function, based on its current state, and on the message received. The new state of the agent determines the message sent in the new round. The local state of agent a at the end of execution t is denoted $\mathsf{view}_a(t)$.

For example, consider the execution $c = G_w, G_{gw}, G_g$, meaning that in the first round only w receives a message; in the second round both processes receive a message; and in the last round only g receives a message. Picking $b_0 = 0$ and $b_1 = 1$ gives us the action c^{01}, where g starts with value 0 and w starts with value 1. This is independent of the algorithm. In the algorithm of Figure 1, local states are simply the value of variable ℓ, so this action leads to the local states $\mathsf{view}_g(c^{01}) = 4/27$ and $\mathsf{view}_w(c^{01}) = 3/27$. In the one-bit algorithm of Figure 2, the local states are the views, so this action leads to the local states $\mathsf{view}_g(c^{01}) = \bot 01$ and $\mathsf{view}_w(c^{01}) = 00\bot$.

[8] For the moment, an action is simply an element of the set $\hat{T}$; we will later define a simplicial action model $\langle T, \chi, \mathsf{pre}\rangle$ whose set of edges is in bijection with $\hat{T}$.

Indistinguishability

The indistinguishability relation $t \sim_a t'$ is a function of both the actions t, t' and the algorithm. It is defined as $t \sim_a t'$ iff $\mathsf{view}_a(t) = \mathsf{view}_a(t')$.

Action model

For a given algorithm, we can reformulate the indistinguishability relation as a simplicial action model $\langle T, \chi, \mathsf{pre}\rangle$ where $\langle T, \chi\rangle$ is a chromatic graph whose vertices are $\mathcal{V}(T) = \{\langle a, \mathsf{view}_a(c^{b_0 b_1})\rangle \mid a \in A, c^{b_0 b_1} \in \hat{T}\}$ and whose edges are of the form, for each action $c^{b_0 b_1} \in \hat{T}$, $X = \{\langle b, \mathsf{view}_w(c^{b_0 b_1})\rangle, \langle g, \mathsf{view}_g(c^{b_0 b_1})\rangle\}$. The precondition of such a facet is $\mathsf{pre}(X) = \mathsf{input}_w^{b_0} \wedge \mathsf{input}_g^{b_1}$.

We write $\mathcal{A}^r$ for the simplicial action model of the r-round averaging algorithm of Figure 1, and $\mathcal{OB}^r$ for the action model of the r-round one-bit algorithm of Figure 2. We also write $\mathcal{DN}^r$ for a generic r-round dynamic network action model over an unspecified algorithm. The action models $\mathcal{OB}^1$ and $\mathcal{OB}^2$ are depicted in Figure 4 (as we mention below, $\mathcal{DN}^r$ and $\mathcal{I}[\mathcal{DN}^r]$ are always isomorphic). For $r = 1$, each of the four input edges has been subdivided into 3 "smaller" edges: one for each possible graph of $\mathcal{G}$. The colors white, grey, of the vertices correspond respectively to agents w, g. The view of each vertex is written next to it. The precondition of the three top edges is true exactly in the top edge of the input model $\mathcal{I}$ on the left, and similarly for the other sides of the square. For $r = 2$, the situation is similar except that each edge of $\mathcal{I}$ is subdivided into $3^2 = 9$ edges.

Perhaps surprisingly, the product update models $\mathcal{I}[\mathcal{A}^r]$ and $\mathcal{I}[\mathcal{OB}^r]$ are isomorphic, meaning that both algorithms from Figures 1 and 2 acquire the same knowledge. They both subdivide each edge of the input complex into exactly 3^r edges, thus they are both also isomorphic to the well-known *full-information* protocol complex.

Theorem 4.1 *Let $\mathcal{I}$ be the binary input model for two processes. For any number of rounds r, the product update models $\mathcal{I}[\mathcal{A}^r]$ and $\mathcal{I}[\mathcal{OB}^r]$ are isomorphic.*

Properties about the action model

One last thing to notice about this construction is that, when we compute the product update model $\mathcal{I}[\mathcal{DN}^r]$, we obtain a simplicial model whose underlying graph is the same as the one of $\mathcal{DN}^r$. So, starting from the input model $\mathcal{I}$, the effect of applying $\mathcal{DN}^r$ is to subdivide each edge of the input (the same thing happens for any other input model). Remarkably, the topology of the input graph is preserved. And in fact, there is a rate of subdivision speed, determined by the constant $1/3$. These are the two basic properties that we will need in the analysis of approximate agreement (known in several specific contexts e.g. [12,21]). Remarkably, they hold for any algorithm, not only full information algorithms.

Theorem 4.2 *For any algorithm for two agents, if the input model $\mathcal{I}$ is connected, then the product update model $\mathcal{I}[\mathcal{DN}^r]$ is connected. Furthermore, each edge is subdivided into at most 3^r edges.*

We have seen that the two properties mentioned in this theorem hold for a full information algorithm. It is not hard to see that they hold for any algorithm, since the indistinguishability relation of a non-full information algorithm is a coarsening of the full information indistinguishability algorithm.

5 Tasks

The action models of the previous section, $\mathcal{A}^r$ and $\mathcal{OB}^r$, describe how the knowledge of the processes evolve when we execute a specific algorithm. Both of those algorithms are solving the same distributed *task*, namely, approximate agreement. In this section, we introduce the notion of *task*, which is an abstract specification of the goal that we are trying to solve, independently of the particular algorithm used to solve it. Informally, a task specifies for each possible input configuration, what are the possible output values that the agents may decide. This is once again formalized using the notion of action model; however, in the case of tasks, the actions do not correspond to communicative events, but to decisions taken by the processes. Tasks have been studied since early on in distributed computability [5]. The DEL semantics that we use here was first introduced in [20].

Consider a simplicial model $\mathcal{I} = \langle I, \chi, \ell\rangle$ called the *initial simplicial model*. Each edge of I, with its labeling ℓ, represents a possible initial configuration. We fix $\mathcal{I}$, the binary inputs model of Section 3.1.

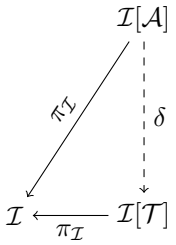
A *task* for $\mathcal{I}$ is a simplicial action model $\mathcal{T} = \langle T, \chi, \mathsf{pre}\rangle$ for agents A, where each edge is of the form $X = \{\langle w, d_w\rangle, \langle g, d_g\rangle\}$. The values d_w, d_g are taken from an arbitrary domain of *output values*. Each such X has a precondition that is true in one or more facets of $\mathcal{I}$, interpreted as "if the input configuration is a facet in which $\mathsf{pre}(X)$ holds, and every agent $a \in A$ decides on the value d_a, then this is a valid execution".

5.1 Semantics of task solvability

Given the simplicial input model $\mathcal{I}$ and a communication model $\mathcal{A}$ such as $\mathcal{DN}^r$, we get the *simplicial protocol model* $\mathcal{I}[\mathcal{A}]$, that represents the knowledge gained by the agents after executing $\mathcal{A}$. To solve a task $\mathcal{T}$, each agent, based on its own knowledge, should produce an output value, such that the edge with the output values corresponds to an edge of $\mathcal{T}$, respecting the preconditions of the task.

The following gives a formal epistemic logic semantics to task solvability. Recall that a morphism δ of simplicial models is a chromatic simplicial map that preserves the labeling: $\ell'(\delta(v)) = \ell(v)$. Also recall that the product update model $\mathcal{I}[\mathcal{A}]$ is a sub-graph of the Cartesian product $\mathcal{I} \times \mathcal{A}$, whose vertices are of the form (i, ac) with i a vertex of $\mathcal{I}$ and ac a vertex of $\mathcal{A}$. We write $\pi_{\mathcal{I}}$ for the first projection on $\mathcal{I}$, which is a morphism of simplicial models.

Definition 5.1 A task $\mathcal{T}$ is *solvable* using the algorithm $\mathcal{A}$ if there exists a morphism $\delta : \mathcal{I}[\mathcal{A}] \to \mathcal{I}[\mathcal{T}]$ such that $\pi_I \circ \delta = \pi_I$, i.e., the diagram of simplicial complexes below commutes.



The justification for this definition is the following. An edge X in $\mathcal{I}[\mathcal{A}]$ corresponds to a pair (i, ac), where i is an edge of $\mathcal{I}$ representing input value assignments to the agents, and ac is an edge of $\mathcal{A}$ codifying the communication exchanges that took place. The morphism δ takes X to an edge $\delta(X) = (i, dec)$ of $\mathcal{I}[\mathcal{T}]$, where dec is the edge of $\mathcal{T}$ defining the set of decision values that the agents will choose in X. Moreover, $\mathsf{pre}(dec)$ holds in i, meaning that dec corresponds to valid decision values for input i. The commutativity of the diagram expresses the fact that both X and $\delta(X)$ correspond to the same input assignment i. Now consider a single vertex $v \in X$ with $\chi(v) = a \in A$. Then, agent a decides its value solely according to its knowledge in $\mathcal{I}[\mathcal{A}]$: if another edge X' contains v, then $\delta(v) \in \delta(X) \cap \delta(X')$, meaning that a has to decide on the same value in both edges.

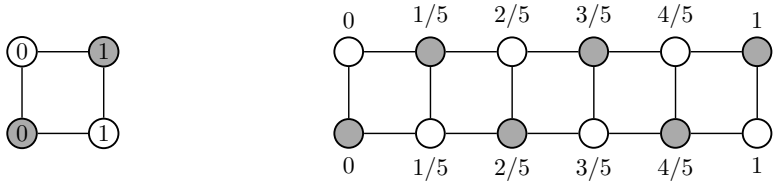
Two observations about the diagram. First, by Lemma 3.4, we know that the knowledge of each agent can only decrease (or stay constant) along the δ arrow. So, any (positive) formula which is known in $\mathcal{I}[\mathcal{T}]$ should already be known in $\mathcal{I}[\mathcal{A}]$. In other words, the goal of the agents is to improve knowledge through communication, by going from $\mathcal{I}$ to $\mathcal{I}[\mathcal{A}]$, in order to match the knowledge required by $\mathcal{I}[\mathcal{T}]$. Second, the possibility of solving a task depends on the existence of a certain simplicial map from the graph of $\mathcal{I}[\mathcal{A}]$ to the graph of $\mathcal{I}[\mathcal{T}]$. We have already seen the appearance of a topological property in Theorem 4.2. Here again topology appears: a simplicial map is the discrete equivalent of a continuous map, thus connectivity is preserved by simplicial maps (Lemma 3.3); hence the topological nature of task solvability.

5.2 Approximate agreement

Recall that in the approximate agreement problem agents are required to decide on values which are close to each other. We have seen in Section 2 that no matter how close to each other one requires the agents to decide, this task is solvable in the $\mathcal{DN}^r$ model, taking a sufficiently large r. Many versions of this task have been considered e.g. [10]. We present here the version of [20], for two agents g and w, which is at the core of previously considered situations.

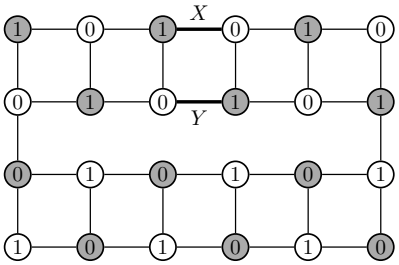
The input complex is the binary input complex for two agents of Section 3.1: so, every possible combination of 0 and 1 can be assigned to the two agents. Their goal will be to output real values in the interval $[0, 1]$, such that: (i) if their input is the same, they both decide the same output, and (ii) if their input is different, they both decide on values d_g and d_w such that $|d_g - d_w| \leq \varepsilon$, for some fixed parameter $\varepsilon \in [0, 1]$. A discrete version of this task is N-*approximate agreement*, where the output values are only allowed to be of the form k/N for $0 \leq k \leq N$. The two decision values should be within $1/N$ of each other: $|d_g - d_w| \leq 1/N$. And to simplify the presentation, let us assume N is odd, without loss of generality, in the sense that to solve ε-agreement, one has to select N large enough, so that $1/N \leq \varepsilon$.

Let $\mathcal{I} = \langle I, \chi, \ell \rangle$ be the input simplicial model for two agents with binary inputs, and $\mathcal{T} = \langle T, \chi, \mathsf{pre} \rangle$ be the following action model. The set of vertices of T is $\mathcal{V}(T) = \{(a, k/N) \mid a \in A \text{ and } 0 \leq k \leq N\}$. The facets of T are edges $X_{k,k'} = \{(g, k/N), (w, k'/N)\}$ with $|k - k'| \leq 1$. The color of a vertex is $\chi(a, k/N) = a$. The precondition $\mathsf{pre}(X_{0,0})$ is true in the worlds 00, 01 and 10 of $\mathcal{I}$; the precondition $\mathsf{pre}(X_{N,N})$ is true in the worlds 11, 01 and 10; and all the other preconditions $\mathsf{pre}(X_{k,k'})$ are true in the worlds 01 and 10. In the figure below are the input model $\mathcal{I}$ (left) and the action model $\mathcal{T}$ (right), for $N = 5$.



$\mathcal{I}$ $\mathcal{T}$

The product update simplicial model $\mathcal{I}[\mathcal{T}]$ is depicted in the next figure, for $N = 5$. The numbers depicted in the nodes are the atomic propositions describing the input values from $\mathcal{I}$. The decision values (of the form $k/5$) are implicit, the first column of nodes corresponds to the decision value 0, the second column is decision value $1/5$, and so on.



$\mathcal{I}[\mathcal{T}]$ for $N = 5$

6 Approximate agreement solvability

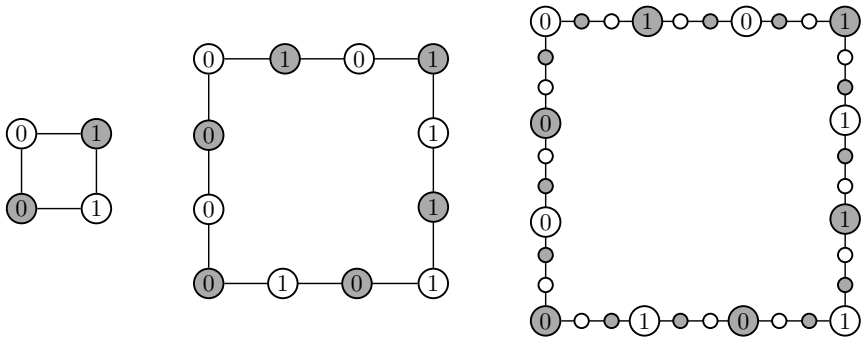
The world X on the figure is the one with the most knowledge in the following sense. We write φ_{01} for the formula expressing that the two inputs are different. Recall (Section 3.1) that $E\varphi = K_g\varphi \wedge K_w\varphi$ for the group knowledge of φ among the agents $\{g, w\}$. Then, we have $\mathcal{I}[\mathcal{T}], X \models E^3\varphi_{01}$. On the other hand, the world Y has less knowledge: we have $\mathcal{I}[\mathcal{T}], Y \models E^2\varphi_{01}$, but $\mathcal{I}[\mathcal{T}], Y \not\models E^3\varphi_{01}$.

Lemma 6.1 *In the simplicial model $\mathcal{I}[\mathcal{T}]$ for the N-approximate agreement task, there are two worlds X, Y which satisfy the formula $E^k\varphi_{01}$, for $k = \lceil N/2 \rceil$.*

Proof We choose the worlds in the "middle" of the model $\mathcal{I}[\mathcal{T}]$, as shown in the picture above. More formally, recall that the vertices of $\mathcal{I}[\mathcal{T}]$ are defined as tuples (a, i, d) where a is an agent, i its input value and d its decision value. The world X is defined as the edge $\{(g, 1, \lfloor N/2 \rfloor/N), (w, 0, (\lfloor N/2 \rfloor + 1)/N)\}$, and $Y = \{(w, 1, \lfloor N/2 \rfloor/N), (g, 0, (\lfloor N/2 \rfloor + 1)/N)\}$. Checking that the formula is satisfied in these worlds simply consists in computing the length of the shortest path to one of the 00 or 11 edges on the sides (Lemma 3.2). □

Solvability of approximate agreement

We now study the solvability of approximate agreement in the r-round dynamic graph model $\mathcal{DN}^r$. Recall that each round subdivides each edge into three edges (Theorem 4.2). The picture below shows the input model $\mathcal{I}$, the model $\mathcal{I}[\mathcal{DN}]$ after one round, and the model $\mathcal{I}[\mathcal{DN}^2]$ after two rounds.



$\mathcal{I}$ $\mathcal{I}[\mathcal{DN}]$ $\mathcal{I}[\mathcal{DN}^2]$

Lemma 6.2 *In the r-round model $\mathcal{I}[\mathcal{DN}^r]$, there is no world X such that $\mathcal{I}[\mathcal{DN}^r], X \models E^k\varphi_{01}$, for $k = \lceil 3^r/2 \rceil$.*

Proof After r rounds each of the four edges of the input model $\mathcal{I}$ has been subdivided into 3^r edges. Thus, every world is at a distance at most $k-1$ from the nearest world with inputs 00 or 11. □

Putting the two lemmas together, we get the following result:

Theorem 6.3 *The N-approximate agreement task is not solvable in the r-round model $\mathcal{I}[\mathcal{DN}^r]$, when $N \geq 3^r + 1$.*

Proof Assume for contradiction that the task is solvable. Then, we would have a map $\delta : \mathcal{I}[\mathcal{DN}^r] \to \mathcal{I}[\mathcal{T}]$. Our goal is to find a contradiction using Lemma 3.4. To achieve this, we should find a formula φ and a world Z of $\mathcal{I}[\mathcal{DN}^r]$, such that φ is false in Z but true in $\delta(Z)$. We choose the formula $\varphi := E^k\varphi_{01}$ for $k = \lceil N/2 \rceil$. Since $N \geq 3^r + 1$ implies $\lceil N/2 \rceil \geq \lceil 3^r/2 \rceil$, we know by Lemma 6.2 that this formula is false in every world Z of $\mathcal{I}[\mathcal{DN}^r]$. All that remains to do is prove that there exists a world of $\mathcal{I}[\mathcal{T}]$, *which is in the image of* δ, and where the formula φ is true.

Since $\mathcal{I}[\mathcal{DN}^r]$ is connected, its image $\delta(\mathcal{I}[\mathcal{DN}^r])$ is connected (by Lemma 3.3). Moreover, the world 00 and the world 11 of $\mathcal{I}[\mathcal{T}]$ must both be in the image of δ, because of the commutative diagram of Definition 5.1. By connectedness, at least one of the middle worlds X, Y of Lemma 6.1 must belong to the image of δ. By Lemma 6.1, this world satisfies φ, which concludes the proof. □

Conversely, we have seen in Section 2 that N-approximate agreement is solvable in r rounds whenever $N = 3^r$. The proof of the above theorem sheds light on the required knowledge to solve approximate agreement: while consensus is about reaching common knowledge, approximate agreement is about reaching some finite level of nested knowledge.

7 Conclusions

We have considered a basic notion of approximate agreement, and a computational model for two agents, where the task is solvable for any desired level of precision. We first presented two algorithms, which solve the task for a given precision level, using the same number of communication rounds. With this concrete setting in mind, we have proceeded to give a formal semantics based on dynamic epistemic logic, both to our computational model, and to the approximate agreement task. We derived a lower bound result showing that these algorithms are optimal in the number of rounds. The lower bound is due to the impossibility of the agents gaining global knowledge about their inputs faster, and a consequence of the connectivity of the computational model's epistemic states. Although much of these results were previously known, we have made an effort to distil the essential ingredients of several previous papers, and combined them into a unified, self-contained way, providing thus a more elementary introduction to the area, based only of graph theory, instead of higher dimensional simplicial complexes.

References

[1] Attiya, H. and S. Rajsbaum, *The combinatorial structure of wait-free solvable tasks*, SIAM J. Comput. **31** (2002), pp. 1286–1313.

[2] Attiya, H. and S. Rajsbaum, *Indistinguishability*, Commun. ACM **63** (2020), p. 9099.
URL https://doi.org/10.1145/3376902

[3] Baltag, A., L. Moss and S. Solecki, *The logic of common knowledge, public announcements, and private suspicions*, in: *TARK VII*, 1998, pp. 43–56.

[4] Baltag, A. and B. Renne, *Dynamic epistemic logic*, in: *The Stanford Encyclopedia of Philosophy, see https://plato.stanford.edu/archives/win2016/entries/dynamic-epistemic/*, Metaphysics Research Lab, Stanford University, 2016 .

[5] Biran, O., S. Moran and S. Zaks, *A Combinatorial Characterization of the Distributed 1-Solvable Tasks*, J. Algorithms **11** (1990), pp. 420–440.

[6] Braud-Santoni, N., S. Dubois, M.-H. Kaaouachi and F. Petit, *The next 700 impossibility results in time-varying graphs*, Int. J. Netw. Comput. **6** (2016), pp. 27–41.

[7] Castañeda, A., P. Fraigniaud, A. Paz, S. Rajsbaum, M. Roy and C. Travers, *A topological perspective on distributed network algorithms*, in: K. Censor-Hillel and M. Flammini, editors, *Structural Information and Communication Complexity - 26th International Colloquium, SIROCCO 2019, L'Aquila, Italy, July 1-4, 2019, Proceedings*, Lecture Notes in Computer Science **11639** (2019), pp. 3–18.

[8] Casteigts, A., P. Flocchini, W. Quattrociocchi and N. Santoro, *Time-varying graphs and dynamic networks*, in: H. Frey, X. Li and S. Ruehrup, editors, *Ad-hoc, Mobile, and Wireless Networks* (2011), pp. 346–359.

[9] Charron-Bost, B., M. Függer and T. Nowak, *Approximate consensus in highly dynamic networks: The role of averaging algorithms*, in: M. M. Halldórsson, K. Iwama, N. Kobayashi and B. Speckmann, editors, *Int. Colloquium on Automata, Languages, and Programming (ICALP)*, number 9135 in Lecture Notes in Computer Science (2015), pp. 528–539.

[10] Charron-Bost, B., M. Függer and T. Nowak, *Fast, Robust, Quantizable Approximate Consensus*, in: I. Chatzigiannakis, M. Mitzenmacher, Y. Rabani and D. Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs) **55** (2016), pp. 137:1–137:14.
URL http://drops.dagstuhl.de/opus/volltexte/2016/6281

[11] Coulouma, É., E. Godard and J. G. Peters, *A characterization of oblivious message adversaries for which consensus is solvable*, Theor. Comput. Sci. **584** (2015), pp. 80–90.
URL https://doi.org/10.1016/j.tcs.2015.01.024

[12] Delporte-Gallet, C., H. Fauconnier and S. Rajsbaum, *Communication complexity of wait-free computability in dynamic networks*, in: A. W. Richa and C. Scheideler, editors, *Structural Information and Communication Complexity (SIROCCO)*, number 12156 in LNCS (2020), pp. 291–309.

[13] van Ditmarsch, H., É. Goubault, J. Ledent and S. Rajsbaum, *Knowledge and simplicial complexes*, arXiv e-prints (2020), to appear in the journal Information and Computation, Elsevier.
URL https://arxiv.org/abs/2002.08863

[14] van Ditmarsch, H., W. van der Hoek and B. Kooi, "Dynamic Epistemic Logic," Springer, 2007.

[15] Dolev, D., N. A. Lynch, S. S. Pinter, E. W. Stark and W. E. Weihl, *Reaching approximate agreement in the presence of faults*, J. ACM **33** (1986), pp. 499–516.
URL https://doi.org/10.1145/5925.5931

[16] Fagin, R., J. Halpern, Y. Moses and M. Vardi, "Reasoning About Knowledge," MIT Press, 1995.

[17] Fischer, M., N. A. Lynch and M. S. Paterson, *Impossibility Of Distributed Commit With One Faulty Process*, Journal of the ACM **32** (1985), pp. 374–382.

[18] Függer, M., T. Nowak and M. Schwarz, *Tight bounds for asymptotic and approximate consensus*, in: C. Newport and I. Keidar, editors, *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018* (2018), pp. 325–334.
URL https://dl.acm.org/citation.cfm?id=3212762

[19] Goubault, É., M. Lazić, J. Ledent and S. Rajsbaum, *A dynamic epistemic logic analysis of the equality negation task*, in: *Dynamic Logic. New Trends and Applications - Second International Workshop, DaLí 2019, Proceedings*, 2019, pp. 53–70.

[20] Goubault, É., J. Ledent and S. Rajsbaum, *A simplicial complex model for dynamic epistemic logic to study distributed task computability*, Information and Computation (2020), p. 104597, in print, a preliminary version appeared in Proc. of GandALF 2018.

[21] Herlihy, M., D. Kozlov and S. Rajsbaum, "Distributed Computing Through Combinatorial Topology," Elsevier-Morgan Kaufmann, 2013.

[22] Herlihy, M., S. Rajsbaum, M. Raynal and J. Stainer, *From wait-free to arbitrary concurrent solo executions in colorless distributed computing*, Theor. Comput. Sci. **683** (2017), pp. 1–21.
URL https://doi.org/10.1016/j.tcs.2017.04.007

[23] Hoest, G. and N. Shavit, *Towards a topological characterization of asynchronous complexity*, SIAM J. Comput. **36** (2006), pp. 457–497.

[24] Kuhn, F. and R. Oshman, *Dynamic networks: models and algorithms*, SIGACT News **42** (2011), pp. 82–96.
URL https://doi.org/10.1145/1959045.1959064

[25] Loui, M. C. and H. H. Abu-Amara, *Memory requirements for agreement among unreliable asynchronous processes*, Advances in Computing Res. **4** (1987), pp. 163–183.

[26] Moses, Y. and S. Rajsbaum, *A layered analysis of consensus*, SIAM J. Comput. **31** (2002), pp. 989–1021.

[27] Winkler, K. and U. Schmid, *An overview of recent results for consensus in directed dynamic networks*, Bull. EATCS **128** (2019).
URL http://bulletin.eatcs.org/index.php/beatcs/article/view/581/585

A Approximate agreement algorithms

Here we provide additional details about the correctness of the algorithms described in Section 2. Recall that $N = 3^k$, agents start with values in $\{0, 1\}$, and they have to decide values at most $1/N$ apart, unless their inputs are equal, in which case their outputs should be equal to their inputs. First we show that the algorithm in Figure 1 solves N-approximate agreement in k rounds.

Theorem A.1 *The algorithm Averaging Approximate Agreement is correct.*

Proof Let me_p be the initial value of agent p.
For $k = 1$, suppose w.l.o.g. that the message from g arrives, then the final value of agent w is $d_w = (me_w + 2me_g)/3$. If both messages arrives then $d_w = (me_g + 2me_w)/3$ so $|d_w - d_g| = |(2me_g - me_g + me_w - 2me_w)/3| = |(me_g - me_w)/3| \leq 1/3$. If just one message arrives then $d_g = me_g$, so $|d_w - d_g| = |(me_w + 2me_g - 3me_g)/3| = |(me_w - me_g)/3| \leq 1/3$.
Let d_p^k be the final value of agent p after k rounds, and suppose that $|d_w^k - d_g^k| \leq 1/3^k$ for some k, then, if message from g arrives in round $k + 1$ (w.l.o.g.), $d_w^{k+1} = (d_w^k + 2d_g^k)/3$. The rest of the proof is analogous to induction base, analysing both cases whether message from w gets lost or not. □

To prove the correctness of the One-Bit Messages algorithm of Figure 2, we rely on Theorem 4.1. So let us prove it first:

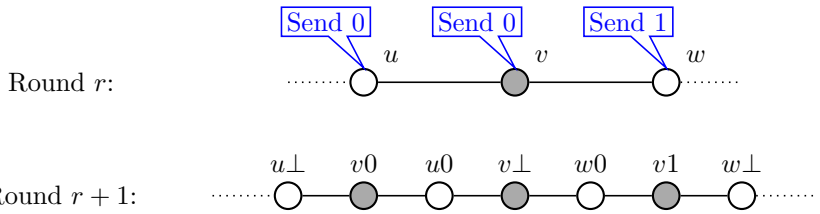
Theorem 4.1. *Let $\mathcal{I}$ be the binary input model for two processes. For any number of rounds r, the product update models $\mathcal{I}[\mathcal{A}^r]$ and $\mathcal{I}[\mathcal{OB}^r]$ are isomorphic.*

Proof Recall that $\mathcal{I}[\mathcal{A}^r]$ is the product update model for r-rounds of the Averaging algorithm in Figure 1, and $\mathcal{I}[\mathcal{OB}^r]$ is the product update model for r-rounds of the One-Bit algorithm of Figure 2. It is well known that $\mathcal{I}[\mathcal{A}^r]$ is isomorphic to the full-information protocol complex, which subdivides each edge of the input into 3^k edges, preserving the topology. Thus, we show that $\mathcal{I}[\mathcal{OB}^r]$ does the same thing.

We proceed by induction on r. For the one-round algorithm, $\mathcal{I}[\mathcal{OB}^1]$ is depicted in Figure 4, and indeed each edge of the input complex has been subdivided into three.

Let us now assume that $\mathcal{I}[\mathcal{OB}^r]$ is indeed isomorphic to $\mathcal{I}[\mathcal{A}^r]$, i.e., that each edge of $\mathcal{I}$ has been subdivided in 3^k small edges. Consider an arbitrary vertex of $\mathcal{I}[\mathcal{OB}^r]$ (a grey vertex w.l.o.g.), of the form $\langle g, \mathsf{view}_g(t)\rangle$ for some action $t = c^{b_0,b_1}$. Let us write $v = \mathsf{view}_g(t)$ for the view of this vertex. Moreover, let us assume that the two neighbors of this vertex, with views u and w, are about to send two distinct messages, 0 and 1 respectively. (One can check that this property is true everywhere in $\mathcal{I}[\mathcal{OB}^1]$: every vertex has neighbors that will send different messages in the next round.)

After one more round of the one-bit algorithm, we obtain six edges in $\mathcal{I}[\mathcal{OB}^{r+1}]$ as depicted below (we assume w.l.o.g. that the grey vertex sends the message "0"; the same picture can be drawn with a 1 instead). Moreover, we can check that the five vertices in the middle at round $r + 1$ still have the inductive property that both neighbors will send a different message at the next round.



This local reasoning can be done around each node of $\mathcal{I}[\mathcal{OB}^r]$. Thus, every edge is subdivided into 3,which concludes our proof. □

Now we can show that the algorithm in Figure 2 solves N-approximate agreement in k rounds.

Theorem A.2 *The algorithm One-Bit Messages Approximate Agreement is correct.*

Proof Since $\mathcal{I}[\mathcal{OB}^r]$ is isomorphic to $\mathcal{I}[\mathcal{A}^r]$, and we have shown in Theorem A.1 that $\mathcal{I}[\mathcal{A}^r]$ solves 3^r-approximate agreement, we know that the One-Bit algorithm can solve approximate agreement as long as we choose the right decision function δ.

This is precisely what the algorithm from Figure 3 does: $\delta(\ell, \text{view})$ simulates the Average Approximate Agreement algorithm via the isomorphism exhibited in Theorem 4.1. A detailed proof of correctness can be found in [12]. □

B Simplicial models and Epistemic Logic

Here we include generalized notions for any number of agents from section 3.

Generalizing graphs to complexes

Given a set V, a *simplicial complex* C is a family of non-empty finite subsets of V such that for all $X \in C$, $Y \subseteq X$ implies $Y \in C$. We say Y is a *face* of X. Elements of V (identified with singletons) are called *vertices*. Elements of C are *simplexes*, and those which are maximal w.r.t. inclusion are *facets*. The set of vertices of C is noted $\mathcal{V}(C)$, and the set of facets $\mathcal{F}(C)$. The *dimension* of a simplex $X \in C$ is $|X| - 1$, and a simplex of dimension n is called an *n-simplex*. A simplicial complex C is *pure* if all its facets are of the same dimension, n. In this case, we say C is of dimension n. A graph without isolated vertices is a pure sumplicial complex of dimension 1. Given the set A of agents (that we will represent as colors), a *chromatic simplicial complex* $\langle C, \chi \rangle$ consists of a simplicial complex C and a coloring map $\chi : \mathcal{V}(C) \to A$, such that for all $X \in C$, all the vertices of X have distinct colors.

Simplicial maps

Let C and D be two simplicial complexes. A *simplicial map* $f : C \to D$ maps the vertices of C to vertices of D, such that if X is a simplex of C, $f(X)$ is a simplex of D. A *chromatic simplicial map* between two chromatic simplicial complexes is a simplicial map that preserves colors. Let $\mathcal{S}_A$ be the category of pure chromatic simplicial complexes on A, with chromatic simplicial maps for morphisms.

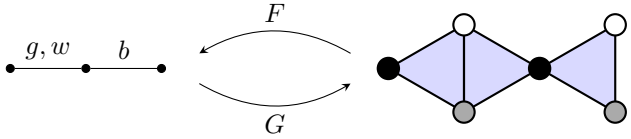
Equivalence with Epistemic Logic

A *Kripke frame* $M = \langle S, \sim \rangle$ over a set A of agents consists of a set of states S and a family of equivalence relations on S, written $\sim_a$ for every $a \in A$. Two states $u, v \in S$ such that $u \sim_a v$ are said to be *indistinguishable* by a. A Kripke frame is *proper* if any two states can be distinguished by at least one agent. Notice that being proper means that the intersection of all equivalence relations $\sim_a$ is the identity; this may reveal interesting parallels with distributed knowledge (a formula that is true in all states in the intersection relation), see e.g. [16]. Let $M = \langle S, \sim \rangle$ and $N = \langle T, \sim' \rangle$ be two Kripke frames. A *morphism* from M to N is a function f from S to T such that for all $u, v \in S$, for all $a \in A$, $u \sim_a v$ implies $f(u) \sim'_a f(v)$. We write $\mathcal{K}_A$ for the category of proper Kripke frames, with morphisms of Kripke frames as arrows.

The following theorem states that we can canonically associate a proper Kripke frame with a pure chromatic simplicial complex, and vice versa. In fact, this correspondence extends to morphisms, and thus we have an equivalence of categories, meaning that the two structures contain the same information.

Theorem B.1 ([20]) *$\mathcal{S}_A$ and $\mathcal{K}_A$ are equivalent categories.*

Example B.2 [[20]] The picture below shows a Kripke frame (left) and its associated chromatic simplicial complex (right). The three agents, named b, g, w, are represented as colors black, grey and white on the vertices of the simplicial complex. The three worlds of the Kripke frame correspond to the three triangles (i.e., 2-dimensional simplexes) of the simplicial complex. The two worlds indistinguishable by agent b, are glued along their black vertex; the two worlds indistinguishable by agents g and w are glued along the grey-and-white edge.



Simplicial models

Let $\mathcal{V}$ be some countable set of values, A be a finite set of agents and $AP = \{p_{a,x} | a \in A, x \in \mathcal{V}\}$ be the set of *atomic propositions*. Intuitively, $p_{a,x}$ is true if agent a holds the value x. We write AP_a for the atomic propositions concerning agent a. A *simplicial model* $M = \langle C, \chi, \ell \rangle$ consists of a chromatic simplicial complex $\langle C, \chi \rangle$, and a labeling $\ell : \mathcal{V}(C) \to \mathscr{P}(AP)$ that associates with each vertex $v \in \mathcal{V}(C)$ a set of atomic propositions concerning agent $\chi(v)$, i.e., such that $\ell(v) \subseteq AP_{\chi(v)}$. Given a simplex $X \in C$, we write $\ell(X) = \bigcup_{i=0}^{n} \ell(v_i)$. A *morphism* of simplicial models $f : M \to M'$ is a chromatic simplicial map that preserves the labeling: $\ell'(f(v)) = \ell(v)$. We denote by $\mathcal{SM}_{A,AP}$ the category of simplicial models over the set of agents A and atomic propositions AP.

Equivalence with epistemic logic

A *Kripke model* $M = \langle S, \sim, L\rangle$ consists of a Kripke frame $\langle S, \sim\rangle$ and a function $L : S \to \mathscr{P}(AP)$. Intuitively, $L(s)$ is the set of atomic propositions that are true in the state s. A Kripke model is *proper* if the underlying Kripke frame is proper. A Kripke model is *local* if for every agent $a \in A$, $s \sim_a s'$ implies $L(s) \cap AP_a = L(s') \cap AP_a$, i.e., an agent always knows its own values. Let $M = \langle S, \sim, L\rangle$ and $M' = \langle S', \sim', L'\rangle$ be two Kripke models on the same set AP. A *morphism of Kripke models* $f : M \to M'$ is a morphism of the underlying Kripke frames such that $L'(f(s)) = L(s)$ for every state s in S. We write $\mathcal{KM}_{A,AP}$ for the category of local proper Kripke models.

Proposition B.3 ([20]) *Given a simplicial model M and a facet X, $M, X \models \varphi$ iff $F(M), X \models_{\mathcal{K}} \varphi$. Conversely, given a local proper Kripke model N and state s, $N, s \models_{\mathcal{K}} \varphi$ iff $G(N), G(s) \models \varphi$, where $G(s)$ is the facet $\{v_0^s, \ldots, v_n^s\}$ of $G(N)$.*

We can now extend Theorem B.1 to an equivalence between simplicial models and Kripke models.

Theorem B.4 ([20]) *$\mathcal{SM}_{A,AP}$ and $\mathcal{KM}_{A,AP}$ are equivalent categories.*

C Knowledge in simplicial models

Here we include additional details about group knowledge and knowledge gain from subsection 3.1.

Lemma 3.2 For a simplicial model M and edge X, we have that $M, X \models E^k\varphi$, iff $M, Y \models E\varphi$ for every $Y \in N^k(X)$.

Proof For $k = 1$ the proof is immediate.
Let k such that $M, X \models E^k\varphi$ iff $M, Y \models E\varphi$ for every $Y \in N^k(X)$. Then $M, X \models E^k(E\varphi)$ iff $M, Z \models E^2\varphi$ for every $Z \in N^k(X)$. Finally $M, Y \models E\varphi$ for every $Y \in N^2(Z)$ such that $Z \in N^k(X)$, i.e. $Y \in N^{k+1}(X)$ (since $N^{k+1}(X) = N(N^k(X))$). □

Lemma 3.4 Consider simplicial models $M = \langle C, \chi, \ell\rangle$ and $M' = \langle C', \chi', \ell'\rangle$, and a morphism $f : M \to M'$. Let $X \in \mathcal{F}(C)$ be an edge of M, a an agent, and $\varphi \in \mathcal{L}_{CK}$ a positive formula, i.e. which does not contain negations except, possibly, in front of atomic propositions. Then, $M', f(X) \models \varphi$ implies $M, X \models \varphi$.

Proof Suppose that φ is atomic, then $M', f(X) \models \varphi$ iff $\varphi \in l(f(X)) = l(X)$, so $M, X \models \varphi$.
If $\varphi = \neg p$ for some atomic p then $M', f(X) \not\models p$, so $p \notin l(f(X)) = l(X)$, therefore $M, X \not\models p$.
If $\varphi = \psi \wedge \theta$ for some formulas ψ and θ as depicted above, then $M', f(X) \models \psi$ and $M', f(X) \models \theta$. Therefore $M, X \models \psi \wedge \theta$.
If $\varphi = K_a(\psi)$ with ψ being a formula like depicted above, let $Y \in \mathcal{F}(M)$ such that $a \in \chi(Y \cap X)$, then $a \in \chi(f(Y) \cap f(X))$ so $M', f(Y) \models \psi$ and therefore $M, Y \models \psi$, implying that $M, X \models \varphi$.
Finally, every possitive formula can be seen as combinations of formulas like depicted above but linked with $\wedge$. Hence, if $\varphi = \bigwedge_{i\in\mathbb{N}} \psi_i$ then for all $i \in \mathbb{N}$, $M', f(X) \models \psi_i$ so that $M, X \models \psi_i$. Finally, $M, X \models \bigwedge_{i\in\mathbb{N}} \psi_i$. □

D Dynamic epistemic logic

Here we present the classic version of DEL.

An *action model* is a structure $\langle T, \sim, \mathsf{pre}\rangle$,where T is a domain of *action points*, such that for each $a \in A$, $\sim_a$ is an equivalence relation on T, and $\mathsf{pre} : T \to \mathcal{L}_{\mathcal{K}}$ is a function that assigns a *precondition* formula $\mathsf{pre}(t)$ to each $t \in T$. Let $M = \langle S, \sim, L\rangle$ be a Kripke model and $\mathcal{A} = \langle T, \sim, \mathsf{pre}\rangle$ be an action model. The *product update model* is $M[\mathcal{A}] = \langle S[\mathcal{A}], \sim^{[\mathcal{A}]}, L[\mathcal{A}]\rangle$, where each world of $S[\mathcal{A}]$ is a pair (s, t) with $s \in S$, $t \in T$ such that $\mathsf{pre}(t)$ holds in s. Then, $(s, t) \sim_a^{[\mathcal{A}]} (s', t')$ whenever it holds that $s \sim_a s'$ and $t \sim_a t'$. The valuation $L[\mathcal{A}]$ at a pair (s, t) is just as it was at s, i.e., $L[\mathcal{A}]((s, t)) = L(s)$. For an initial Kripke model M, the effect of action model $\mathcal{A}$ is a Kripke model $M[\mathcal{A}]$. Notice that if M is a local proper Kripke model and $\mathcal{A} = \langle T, \sim, \mathsf{pre}\rangle$ is a proper action model, then $M[\mathcal{A}]$ is proper and local.

Equivalence with simplicial action models

Recall from Theorem B.4 the two functors F and G that define an equivalence of categories between simplicial models and Kripke models. We have a similar correspondence between action models and simplicial action models. On the underlying Kripke frame and simplicial complex they are the same as before; and the precondition of an action point is just copied to the corresponding facet. The following proposition says that the "classic" product update agrees with the "fully simplicial" one.

Proposition D.1 *Consider a simplicial model M and simplicial action model $\mathcal{A}$, and their corresponding Kripke model $F(M)$ and action model $F(\mathcal{A})$. Then, the Kripke models $F(M[\mathcal{A}])$ and $F(M)[F(\mathcal{A})]$ are isomorphic. The same is true for G, starting with a Kripke model M and action model $\mathcal{A}$.*

Proof Particular case of theorem B.4, since $F(M[\mathcal{A}])$ is the image of $F(M)[F(\mathcal{A})]$ under the functor who states equivalence. □

E Dynamic networks and Tasks

Here we include additional details about Section 4 and 5.

Theorem 4.2 For any algorithm for two agents, the product update model $M[\mathcal{DN}^r]$ is a graph which is connected (assuming the input model is connected). Furthermore, each edge is subdivided into at most 3^r edges.

Proof Since underlying simplicial complex of $M[\mathcal{DN}^r]$ is the same as the one of $G(\mathcal{DN}^r)$, we just need to prove that $G(\mathcal{DN}^r)$ is connected.
For $r = 1$, let $\mathcal{I}$ be the input model, then the function who carries any edge X of $\mathcal{I}$ into the set $\{X \in \mathcal{F}(G(\mathcal{DN}^r))|M, \{(b, b_0), (g, b_1)\} \models pre(X)\}$ is bijective. Moreover $f[\mathcal{I}] = \mathcal{F}(G(\mathcal{DN}))$ and since any edge $X \in \mathcal{F}(\mathcal{I})$ is such that $|f(X)| = 3$, X splits into 3 edges in $G(\mathcal{DN})$.
Now consider arbitrary edges $X, Y \in \mathcal{F}(\mathcal{I})$ and let $X_i, Y_j \in \mathcal{F}(G(\mathcal{DN}))$ subdivisions of X, Y respectively. Since $\mathcal{I}$ is connected, exists $(X, Z_1, \ldots, Z_n, Y)$ a sequence of connected edges in $\mathcal{I}$. Hence, in $G(\mathcal{DN})$ we have another edge path who links X_i with any subdivision of Z_1, which is linked with any subdivision of Z_2 and so on, until reach any subdivision of Z_n, and finally Y_j.
For r such that $G(\mathcal{DN}^r)$ is connected, we can consider $G(\mathcal{DN}^r)$ as an input model, and since $G(\mathcal{DN}^{r+1})$ is also the underlying graph of $(M[\mathcal{DN}^r])[\mathcal{DN}]$, the rest of the proof is analogous to induction base. Finally, since each edge is already subdivided into 3^r edges in $G(\mathcal{DN}^r)$, they are subdivided into $3(3^r) = 3^{r+1}$ edges in $G(\mathcal{DN}^{r+1})$.

□