# Coordinated Consensus in Dynamic Networks

Fabian Kuhn
Faculty of Informatics,
University of Lugano
6904 Lugano,Switzerland
fabian.kuhn@usi.ch

Yoram Moses
Department of Electrical
Engineering, Technion
Haifa, 32000, Israel
moses@ee.technion.ac.il

Rotem Oshman
Computer Science and
Artificial Intelligence Lab, MIT
Cambridge, MA 02139, USA
rotem@csail.mit.edu

## ABSTRACT

We study several variants of coordinated consensus in dynamic networks. We assume a synchronous model, where the communication graph for each round is chosen by a worst-case adversary. The network topology is always connected, but can change completely from one round to the next. The model captures mobile and wireless networks, where communication can be unpredictable.

In this setting we study the fundamental problems of eventual, simultaneous, and $\Delta$-coordinated consensus, as well as their relationship to other distributed problems, such as determining the size of the network. We show that in the absence of a good initial upper bound on the size of the network, eventual consensus is as hard as computing deterministic functions of the input, e.g., the minimum or maximum of inputs to the nodes. We also give an algorithm for computing such functions that is optimal in every execution. Next, we show that simultaneous consensus can never be achieved in less than $n-1$ rounds in any execution, where $n$ is the size of the network; consequently, simultaneous consensus is as hard as computing an upper bound on the number of nodes in the network.

For $\Delta$-coordinated consensus, we show that if the ratio between nodes with input 0 and input 1 is bounded away from 1, it is possible to decide in time $n-\Theta(\sqrt{n\Delta})$, where $\Delta$ bounds the time from the first decision until all nodes decide. If the dynamic graph has diameter $D$, the time to decide is $\min\{O(nD/\Delta), n-\Omega(n\Delta/D)\}$, even if $D$ is not known in advance. Finally, we show that (a) there is a dynamic graph such that for every input, no node can decide before time $n-O(\Delta^{0.28}n^{0.72})$; and (b) for any diameter $D=O(\Delta)$, there is an execution with diameter $D$ where no node can decide before time $\Omega(nD/\Delta)$. To our knowledge, our work constitutes the first study of $\Delta$-coordinated consensus in general graphs.

**Categories and Subject Descriptors:**
F.2.2 [Analysis of Algorithms and Problem Complexity]: Non-numerical Algorithms and Problems—*computations on discrete structures*
G.2.2 [Discrete Mathematics]: Graph Theory—*network problems*
**General Terms:** Algorithms, Theory
**Keywords:** distributed algorithms, dynamic networks, consensus, coordination, common knowledge


*PODC'11,* June 6–8, 2011, San Jose, California, USA.


## 1. INTRODUCTION

Coordinating the actions of distributed computing devices or mobile agents is an essential distributed task. Applications of coordination abound in robotics and swarm protocols, where many mobile agents cooperate to jointly accomplish some global objective. In such examples, nodes must jointly agree to execute some common action (movement, data collection, or even something so simple as resetting their clocks or starting a protocol) at the same or almost the same time.

Global coordination is a challenging task, made all the more difficult in dynamic settings, where the agents move around and the communication links between them can behave unpredictably. In this paper we study the problems of consensus, simultaneous consensus, and $\Delta$-coordinated (i.e., "almost simultaneous") consensus in dynamic networks. Our goal is to characterize the time complexity of these tasks, as well as to investigate the relationship that they bear to higher-level tasks, such as computing functions of inputs to the nodes and determining the number of nodes.

We study the above problems in the dynamic network model of [18]. The model is round-based; in each round, the communication network is an adversarially-chosen graph over a vertex set $V$ of size $n$. The set $V$ of network nodes is assumed to be fixed throughout an execution, although we do not assume that the participants know $V$ (or even $n$, in some cases). The communication graph is assumed to be connected, but it can change completely from one round to the next. Nodes communicate by broadcasting messages to their immediate neighbors. Similar dynamic network models have previously been considered in, e.g. [1, 16, 17, 22], and many others.

Our main objective in this paper is to understand the complexity of consensus and of coordinating actions and decisions in dynamic networks. We begin by studying *eventual consensus*, in which each node receives an initial input, and all nodes must eventually agree on the input to one of the nodes. We show that eventual consensus is closely related to knowing when a node has been causally influenced by all nodes in the graph; namely, in certain settings, no node can decide on an output value until it has been causally influenced by all nodes. Although the decision value in a consensus protocol is not a deterministic function of the inputs, our result implies that in many settings it is equivalent in difficulty to computing a deterministic function such as the minimum or maximum of inputs. We also give an optimal criterion for determining when a node knows it has been causally influenced by all nodes, and so can decide.

Next we turn our attention to the problem of *simultaneous consensus*, where nodes are required to output their decision value simultaneously. Simultaneous coordination is a useful primitive, as many distributed protocols assume that all nodes begin executing the protocol at the same time; without simultaneous coordination, it is not possible to execute such a protocol soon after some other

protocol completes (that is, the protocols cannot be sequentially composed). It is known that achieving simultaneous consensus is tightly related to obtaining common knowledge in a distributed system [8, 15]. Informally, a fact $\varphi$ is common knowledge whenever $\varphi$ is known to all nodes, and everyone knows that everyone knows $\varphi$, and (everyone knows)$^3$ $\varphi$, and so on (a more rigorous definition is presented in Section 2). When nodes must execute an action at the same time, the fact that the action is being performed must be common knowledge. We show that achieving common knowledge is costly in dynamic networks: it always requires $n-1$ rounds. This holds even in executions where the communication graph is well-behaved, e.g., when it is static and has a small diameter. In particular, this result implies that simultaneous consensus can never be achieved before time $n-1$, even if $n$ is known *a priori*. (Compare to eventual consensus, which can be solved in two rounds if the graph is fully-connected.) If the number of nodes is *not* known *a priori*, then $n$ rounds are required. This implies that solving simultaneous consensus in this model is as hard as computing an upper bound on the size of the network: given a protocol for simultaneous consensus, we can obtain an upper bound on $n$ by simply having each node output the round number in which it decides.

In light of the cost of simultaneous consensus, it is desirable to find a trade-off between the time it takes to achieve coordination and the quality of coordination achieved. We show that such a trade-off exists by considering $\Delta$-*coordinated consensus*, a variant of consensus in which all nodes are required to output their decision values within $\Delta$ rounds of each other. In particular, simultaneous consensus is equivalent to 0-coordinated consensus, and eventual consensus to $\infty$-coordinated consensus.

One might initially expect that a protocol for $\Delta$-coordinated consensus would not be able to improve upon the running time of a simultaneous consensus protocol by more than $\Delta$ rounds, and indeed we show that this is true in the worst-case: for some input and some execution, $\Delta$-coordinated consensus requires $n-\Delta-1$ rounds. However, surprisingly, there are many cases in which even 1-coordinated consensus can decide significantly faster than simultaneous consensus. For example, we give a protocol that halts in $n-\Theta(\sqrt{n\Delta})$ rounds if the ratio of the number of zeroes to the number of ones in the input is bounded away from 1, and we give another protocol that halts in $\min\{O(nD/\Delta), n-\Omega(n\Delta/D)\}$ rounds in graphs where each message takes no more than $D$ rounds to traverse the network (we call $D$ the *dynamic diameter* of the network). Hence, for the purpose of achieving coordinated consensus, having a small-diameter network does help significantly, whereas for simultaneous consensus it does not help at all.

On the negative side, we show that there is a static network such that for every $\Delta$-coordinated consensus algorithm and every input assignment, no node decides before time $n-O(\Delta^{0.28}n^{0.72})$. The network we construct in this lower bound has a diameter of $\Theta(n)$, which makes it inherently "difficult". To complete the picture, we also show that for every $D = O(\Delta)$, there is a static network of diameter $D$ such that for all algorithms and inputs, no node decides before time $\Omega(nD/\Delta)$. Both lower bounds use a novel variation on the standard proof technique used in, e.g., [8] to obtain lower bounds on the time to acquire common knowledge. In [8], one freely moves between indistinguishable points (configurations); in contrast, here we pay a cost each time we move to some new indistinguishable point, and our goal is to minimize the total number of points involved in the proof.

In these two lower bounds we exhibit *static* networks in which solving $\Delta$-coordinated consensus is hard (i.e., it requires many rounds). The hardness arises from the *potential* for dynamic behavior: although in practice the network topology does not change during the execution, the nodes do not know in advance that this will be the case, and informally, they must assume the worst-case dynamic behavior. We note also that the three lower bounds we give in this paper are in some sense incomparable with each other.

- The $n-\Delta-1$ lower bound asserts, in a non-constructive manner, the existence of a particular combination of dynamic network and input assignment for which $\Delta$-coordinated consensus is hard.
- In the $n-O(\Delta^{0.28}n^{0.72})$ lower bound we construct a specific network in which *every* input assignment is hard. This network has diameter $\Theta(n)$.
- The $\Omega(nD/\Delta)$ lower bound also gives a specific network in which every input assignment is hard. While the bound is smaller than the previous one, it applies to every diameter $D = O(\Delta)$.

## 1.1 Related work

*Consensus and knowledge.* Consensus is a central topic in distributed computing, initiated by the seminal paper by Pease, Shostak and Lamport [23]. Most of the literature on the subject in the context of message-passing systems assumes that the network is a complete graph, with direct channels connecting every pair of nodes. For more general networks, there has been work on the connectivity requirements for reaching consensus under various failure models (see, e.g., [7]), as well as work on implementing consensus in bounded-degree networks with special properties, such as expanders [14, 9]. We are not aware of a study of the efficiency of consensus protocols in general graphs. The current paper considers an even weaker network model, where the graph can possibly change completely from one round to the next.

While most of the literature on consensus is concerned with tolerating node failures, in the dynamic network model that we consider here the nodes themselves are assumed to be reliable, but the protocol must overcome potentially drastic changes in topology between rounds. Santoro and Widmayer studied consensus in the context of edge failures [24], and showed that it is unsolvable if more than $n-2$ (arbitrarily chosen) edges can be down in every round. The dynamic network model allows a much broader set of executions, since almost all (in fact, all but $n-1$) edges can be down in every round, and their choice is almost arbitrary. The only requirement is that the network in each round be connected.

Some of our results concern cases in which the number of nodes in the network is unknown, or in which there is a rough but inexact bound on the number of nodes. These are unusual assumptions in the context of consensus. A number of standard consensus protocols (e.g., [2]) can easily be modified to handle such assumptions, but this is only due to the fact that the network there is a complete graph, so that a node hears from all correct nodes in every round.

Simultaneous coordination has been shown to be closely related to the notion of common knowledge [15, 10]. Thus, for example, in a simultaneous consensus protocol [8, 21], when the nodes decide on $v$, it must be common knowledge that some initial value is $v$. This is much stronger than for regular consensus, in which a node deciding $v$ must (individually) know that one of the initial values was $v$. It has been shown that deciding in simultaneous consensus (and in a large class of simultaneous coordination tasks) can be reduced to the problem of computing when facts (and which facts) are common knowledge at any given point in an execution. For simultaneous tasks, this enables the design of protocols that are *all-case* optimal: for *every* behavior of the adversary, in the execution of the all-case optimal protocol, nodes decide as fast as they do for

that behavior under any other protocol. (All-case optimality does not exist for eventual consensus, as shown in [21].)

Part of our analysis centers on the problem of $\Delta$-coordinated consensus, in which decisions must be taken at most $\Delta$ rounds apart. In the standard literature, many protocols for eventual agreement are 1-coordinated in this sense: because the network is assumed to be fully-connected, once some correct node $v$ decides, all other correct nodes find out about $v$'s decision in the next round; it is then safe for all correct nodes to decide $v$ as well. For networks that are general graphs, we know of no work developing $\Delta$-coordinated consensus protocols. As in the case of simultaneous coordination, the property of $\Delta$-coordination has a natural counterpart in knowledge theory, called $\Delta$-common knowledge. Very roughly speaking, if $u$ knows that a fact is $\Delta$-common knowledge, then within $\Delta$ rounds everyone will know that this is the case. In order to decide, a node must know that the decision value is $\Delta$-common knowledge [15, 10]; the analysis in Section 6 is the first case in which such coordination is analyzed and nontrivial bounds are obtained as a result.

*Dynamic networks.* In an increasingly networked world, in which various kinds of computing devices of all sizes are connected to form large networks, understanding dynamic networks has become all the more important. It is thus not surprising that in recent years there has been a significant amount of work on dynamic network algorithms, for a large variety of different dynamic network models. Our discussion here is restricted to models similar to the one we consider in the current paper; we refer the interested reader to [19] for a discussion of a few alternative models.

Some initial results on distributed computations in completely adversarial dynamic networks were obtained in [22]. The model as studied in this paper was introduced in [18], where the complexity of basic computation and communication tasks such as determining the size of the network or exchanging information among all the nodes was studied. In [1], Avin et al. study the behavior of random walks in a very similar dynamic network model. Some basic information dissemination tasks, such as globally broadcasting a message, have also been considered in a probabilistic version of the graph model in which edges are independently formed and removed according to a simple random process; e.g., [3, 5, 6]. Another problem related to distributed coordination is clock synchronization. In [16, 17], the problem of clock synchronization was investigated in a partially-synchronous variant of the dynamic graph model we study here. Related dynamic network models were also considered in, e.g., [4, 11, 12, 13], and others.

## 2. MODEL AND DEFINITIONS

We now formally introduce the dynamic graph model, originally introduced in [18]. As explained above, we consider a synchronous-round based model of computation, in which the set of nodes (processes) is not known *a priori*. The set of nodes that participate in a given execution is, however, fixed for the duration of the execution, and each of them has a unique identifier (UID). The nodes share a global clock, which starts at 0 and advances in unit steps.

Communication proceeds in synchronous rounds; we think of round $k$ (for $k = 1, 2, \ldots$) as taking place between time $k-1$ and time $k$. Round $k$ proceeds as follows: first, each node generates a single message to broadcast, based on its local state at time $k-1$. The adversary then selects a communication graph (i.e., a set of edges) for round $k$, and delivers each message to the sender's neighbors in accordance with the edges it chose. The communication graph for each round is assumed to be connected, but this is the only constraint on the adversary.[1] After messages are delivered, each node processes the messages it received, and transitions to a new state (its state at time $k$). Then the next round begins.

The adversary's behavior in a given execution is described by a *dynamic graph* $G = (V, E, \sigma)$, where $|V| > 2$ is a set of nodes (or processes), $E : \mathbb{N}^+ \to \binom{V}{2}$ is a *dynamic edge function* which assigns to each round $r$ a set $E(r)$ of undirected edges over $V$, and $\sigma$ is the *signature* of the execution. The signature is an assignment of a unique identifier (UID) and an input (or initial value) to each node in $V$. If nodes have access to an upper bound on the count $|V|$, this upper bound is also part of the signature $\sigma$. In particular, if $\sigma$ always includes the exact number of nodes, then we say that the count is known *a priori*. We are frequently concerned only with the dynamic network topology; in this case we omit the signature $\sigma$ from our notation.

A dynamic graph $G = (V, E)$ induces a *causal order*, denoted $(u,t) \rightsquigarrow_G (v,t')$, where $(u,t)$ and $(v,t')$ are *time-nodes* representing the states of nodes $u$ and $v$ at times $t$ and $t'$, respectively. Informally, the causal order captures the idea that a time-node $(u,t)$ can only influence another time-node $(v,t')$ in a given execution if there is a chain of messages starting from $u$ at time $t$ and ending at $v$ at time $t'$. Formally, the causal order is defined in the usual way: it is the transitive and reflexive closure of the order $(u,t) \to_G (v,t+1)$, which holds iff either $u = v$ or $\{u,v\} \in E(t+1)$. We omit the subscript $G$ when it is clear from the context.

At time $t$, node $u$ has direct information only about the states of nodes $v$ at time $t'$ such that $(v,t') \rightsquigarrow (u,t)$. This motivates the next definition, which defines all the information a node can possibly acquire about an execution.

DEFINITION 1 (VIEW). *The* view *of node $u$ at time $t$ in dynamic graph $G$, denoted* $\mathsf{view}_{(G,u,t)}$*, is defined as the restriction of $G$ to the time-nodes and edges along paths from time 0 nodes to $(u,t)$ in $G$ (see Fig. 1). In particular,* $\mathsf{view}_{(G,u,t)}$ *includes the states of all nodes $v$ at time $t'$ such that $(v,t') \rightsquigarrow_G (u,t)$.*

In particular, node $u$ cannot know the input value of any node $v$ such that $(v,0) \not\rightsquigarrow (u,t)$. A common strategy in consensus lower bounds and impossibility proofs is to create a situation where $(v,0) \not\rightsquigarrow (u,t)$, and then flip the input value of $v$, without node $u$ being able to tell the difference (at least until time $t$). Thus we are often interested in the set of nodes whose input values $u$ can potentially know at time $t$ (see Fig. 1 for an illustration.)

DEFINITION 2 (PAST SET). *The* past set *of a time-node $(u,t)$ from time $t'$ in graph $G$ is defined by*

$$\mathsf{past}_{(G,u,t)}(t') := \left\{ v \mid (v,t') \rightsquigarrow (u,t) \right\}.$$

If $v \in \mathsf{past}_{(G,u,t)}(0)$ (i.e., if $(v,0) \rightsquigarrow_G (u,t)$), then we say that at time $t$ node $u$ has *heard from* node $v$. As usual, we omit the subscript $G$ from our notation where it is clear from the context.

In static networks, the performance of distributed algorithms often depends on the *diameter* of the network. In a dynamic network, the diameter of the communication graph can change from round to round, and is not a good measure of the amount of time required for information to spread through the network (see [19] for discussion). Thus, we use a more general definition, which explicitly captures the amount of time required for any node to hear from any other node:

DEFINITION 3 (DYNAMIC DIAMETER). *We say that dynamic graph $G = (V, E)$* has a dynamic diameter of $D$ up to time $t$ *if for all $t' \le t$ and $u, v \in V$ we have* $(u, \max\{0, t' - D\}) \rightsquigarrow (v, t')$.

[1] This assumption was called *1-interval connectivity* in [18].

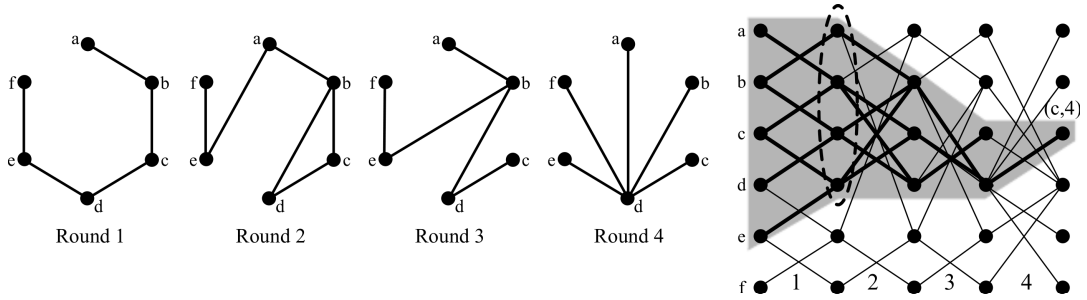


Figure 1: 4 rounds of a 6-node dynamic graph: The nodes and edges in the gray area together form $\mathsf{view}_{(c,4)}$, the 4 nodes inside the dashed ellipse are the nodes in the set $\mathsf{past}_{(c,4)}(1)$.

(In other words, for any time $t' \le t$ and any node $u \in V$, we have $\mathsf{past}_{(u,t')}(t' - D) = V$.)

In our lower bounds and knowledge analysis we assume that nodes execute a *full information protocol*, where the state of each node $u$ at time $t$ is exactly $\mathsf{view}_{(u,t)}$. This state is then broadcast by $u$ at time $t$, allowing $u$'s neighbors $v$ to combine the views they receive and compute $\mathsf{view}_{(v,t+1)}$. Any lower bound on full information protocols extends, of course, to protocols that are not full information. Our upper bounds typically require nodes to send at least the inputs of all the nodes they have heard from, and sometimes more information as well.

## 2.1 Knowledge and Common Knowledge

Knowledge theory, and specifically the notion of common knowledge and its variants, is central to the study of coordinated actions. This connection has been developed and described in [15, 10, 8, 21, 20]. that simultaneous coordination is closely related to common knowledge. In this section we review the basic definitions, focusing on the elements needed for our analysis of dynamic networks. For a more complete exposition of knowledge theory see [10].

Given a distributed protocol $P$, let $\mathcal{R} = \mathcal{R}(P)$ be the set of all runs (executions) of $P$. A *point of* $\mathcal{R}$ is a pair $(G,t)$, representing the global state of the execution at time $t$ when protocol $P$ is executed in dynamic graph $G$.

The fundamental notion underlying the concept of knowledge is *indistinguishablity* among points. We write $(G,t) \sim_u (G',t)$ and say that the two points are *indistinguishable to node* $u$, if node $u$'s view is the same in both runs, that is, $\mathsf{view}_{(G,u,t)} = \mathsf{view}_{(G',u,t)}$. (Recall from Section 2 that a node's view represents all the information it can possibly acquire about the execution.) Notice that two points may be indistinguishable to $u$ even though the runs are quite different; for example, the number of nodes may differ between the two runs, as well as the inputs and UIDs of some of the nodes. The points $(G,t)$ in a system $\mathcal{R}$, and the individual indistinguishability relations $\sim_u$ for all nodes that appear in the runs of $\mathcal{R}$, define an undirected, edge-labelled graph that is called the *similarity graph* for $\mathcal{R}$. We are particularly interested in the connected components of the similarity graph; we denote $(G,t) \sim (G',t)$ if points $(G,t)$ and $(G',t)$ are in the same connected component, that is, there is a sequence $u_0, \ldots, u_k$ such that $(G,t) = (G_0,t) \sim_{u_0} (G_1,t) \sim_{u_1} \ldots \sim_{u_k} (G_{k+1},t) = (G',t)$. (Note that for $i = 0, \ldots, k-1$ we must have $u_i \in V_i \cap V_{i+1}$, where $V_i$ is the node set of $G_i$.)

Intuitively, node $u$ will *know* a fact $\varphi$ at $(G,t)$ exactly if $\varphi$ holds at all points $(G',t)$ such that $(G,t) \sim_u (G',t)$. In other words, a fact is known by $u$ if $u$'s view implies that the fact must be true. ode $u$ knows a given fact iff the node's information implies that the fact is true. We now formalize this intuition with a minimal amount of logical notation.

Given a system $\mathcal{R} = \mathcal{R}(P)$ (that is, a collection of all runs of some protocol $P$), we start out with some set $\Phi$ of *basic facts* of interest; each basic fact is associated with a set of points $(G,t)$ in which it is satisfied. We use $(\mathcal{R},G,t) \vDash \varphi$ to denote the satisfaction of fact $\varphi$ in $(G,t)$.

Now let $K_u\varphi$ stand for the fact that *node $u$ knows that $\varphi$ holds*. We call $K_u\varphi$ a *knowledge formula*, and its satisfaction is formally defined as follows:

$$(\mathcal{R},G,t) \vDash K_u\varphi \qquad \text{iff}$$
$$(\mathcal{R},G',t) \vDash \varphi \text{ holds for all } (G',t) \sim_u (G,t).$$

Note that $\varphi$ does not have to be a basic formula; it can itself be a knowledge formula. For example, the formula $K_uK_v\psi$ asserts that "node $u$ knows that node $v$ knows $\psi$".

It is convenient to define additional knowledge operators $E$ and $C$, which can also be combined and nested. The operator $E$ stands for *everyone knows*, and it is formally defined by

$$(\mathcal{R},G,t) \vDash E\varphi \qquad \text{iff}$$
$$(\mathcal{R},G,t) \vDash K_u\varphi \text{ holds for all } u \in V,$$

where $V$ is the node set of $G$.

The operator $C$ stands for *common knowledge*: a fact $\varphi$ is common knowledge if $\varphi$ holds, and everyone knows that $\varphi$ holds, and everyone knows that everyone knows that $\varphi$ holds, and so on. The $C$ operator can be formally defined as a fixpoint (see [10]), but here we give a more semantic definition, in terms of the similarity graph:

$$(\mathcal{R},G,t) \vDash C\varphi \qquad \text{iff}$$
$$(\mathcal{R},G',t) \vDash \varphi \text{ holds for all } (G',t) \sim (G,t).$$

According to this definition, a fact $\varphi$ is *not* common knowledge at $(G,t)$ whenever there is a graph $(G',t)$ such that $(G',t) \nvDash \varphi$, and a chain $u_0, \ldots, u_k$ such that $(G,t) = (G_0,t) \sim_{u_0} (G_1,t) \sim_{u_1} \ldots \sim_{u_k} (G_{k+1},t) = (G',t)$. Informally this means that $\varphi$ is not common knowledge if some node $u_0$ suspects that some node $u_1$ suspects that... some node $u_k$ suspects that $\varphi$ might not hold. (Here "$u_i$ suspects $\psi$" is to be formally understood as $\neg K_{u_i}\neg\psi$, that is, $u_i$ does not know that $\psi$ is not true.)

Common knowledge is known to be closely related to simultaneous coordination. For example, in simultaneous consensus, a node cannot decide $v$ before it is common knowledge that $v$ is the input to some node. In general, any action $a$ that must be performed simultaneously can only be performed when it is common knowledge that $a$ is being performed. The simultaneity of $a$ implies that whenever $a$ is performed *everyone knows* that $a$ is performed; a

straightforward induction on the length of paths in the similarity graph shows that $a$ is common knowledge, that is, $a$ is performed at all points in the connected component of the similarity graph. We review the argument relating common knowledge and simultaneous consensus in Section 6.1.

# 3. CAUSALITY IN DYNAMIC GRAPHS

As we saw in the previous section, at time $t$ a node $u$ can only know the input of another node $v$ if $v \in \mathsf{past}_{(u,t)}(0)$, i.e., if $(v,0) \rightsquigarrow (u,t)$. Globally-sensitive functions, such as the minimum or maximum of inputs to all nodes, require a node to know when it has *heard from everyone*; node $u$ is only guaranteed that it has the true answer at time $t$ if $\mathsf{past}_{(u,t)}(0) = V$.[2] In this section we give an optimal condition that allows a node to test when it has heard from all nodes in the graph, even if it does not know *a priori* how many nodes there are.

The problem of determining when a node has heard from everyone was already considered in [18], and an $\Theta(n)$-round algorithm was presented. While $\Omega(n)$ is a trivial lower bound on the problem, the algorithm of [18] has the drawback of *always* requiring $\Theta(n)$ rounds, even when the network has small dynamic diameter. Here we give an algorithm which is all-case optimal, and in particular, requires $O(D)$ time in networks with dynamic diameter $D$.

The test is surprisingly simple; it only requires the node to keep track of its past sets from time 0 and from time 1.

LEMMA 3.1. *Node $u$ knows at time $t$ that $\mathsf{past}_{(u,t)}(0) = V$ iff $\mathsf{past}_{(u,t)}(0) = \mathsf{past}_{(u,t)}(1)$.*

PROOF. First, suppose that $\mathsf{past}_{(u,t)}(0) = \mathsf{past}_{(u,t)}(1)$. This implies that $\mathsf{past}_{(u,t)}(1) = V$: if $V \setminus \mathsf{past}_{(u,t)}(1)$ is non-empty, connectivity in round 1 implies that there is some edge $\{v,w\} \in E(1)$ such that $v \in \mathsf{past}_{(u,t)}(1)$ and $w \notin \mathsf{past}_{(u,t)}(1)$. But this means that $(w,0) \rightsquigarrow (v,1) \rightsquigarrow (u,t)$, and hence $w \in \mathsf{past}_{(u,t)}(0)$ and $\mathsf{past}_{(u,t)}(0) \neq \mathsf{past}_{(u,t)}(1)$. Thus, $\mathsf{past}_{(u,t)}(1) = V$, which also implies that $\mathsf{past}_{(u,t)}(0) = V$.

For the other direction, suppose $\mathsf{past}_{(u,t)}(0) \neq \mathsf{past}_{(u,t)}(1)$. This does not necessarily imply that $\mathsf{past}_{(u,t)}(0) \neq V$; however, there is some node $v \in \mathsf{past}_{(u,t)}(0) \setminus \mathsf{past}_{(u,t)}(1)$ that $u$ has not heard from since time 0. At time 0, no communication rounds have occurred yet, so $v$ does not yet know who its neighbors will be. The adversary can conceal arbitrarily many nodes from $(u,t)$ by connecting them only to node $v$ throughout the execution. Since $u$ never hears from $v$ from time 1 onwards, it cannot distinguish (for example, in the graph from Fig. 1, node (c) cannot tell whether node (f) is part of the network or not). Therefore node $u$ cannot *know* it has heard from everyone (even if in fact it has). □

We remark that if $\mathsf{past}_{(u,t)}(0) \neq V$ and $u$ has no *a priori* upper bound on the count, then $u$ has no upper bound on $|V|$ at time $t$: as we saw above, any node $v$ from which $u$ has not heard could be "concealing" arbitrarily many other nodes that are connected to the rest of the graph only through $v$. Thus, if $\mathsf{past}_{(u,t)}(0) \neq V$, then at time $t$ node $u$ cannot know the value of a wide class of functions, including majority, minimum or maximum with unbounded inputs, and in general any function $f : (\bigcup_{n=1}^{\infty} D^n) \to D$ (where $D$ is the data domain) satisfying the following condition: for any input assignment $I \in D^n$ there exists a size $n' > n$ and an extension $I' \in D^{n'}$ of $I$, such that $f(I) \neq f(I')$. For each such function, Lemma 3.1 yields an all-case optimal algorithm: by forwarding all input values (or sufficient information about them to allow $f$ to be computed), and stopping as soon as $\mathsf{past}_{(u,t)}(0) = \mathsf{past}_{(u,t)}(1)$, we obtain an algorithm that cannot be beaten by any other algorithm in any execution.

In fact, it turns out that knowing when $\mathsf{past}_{(u,t)}(0) = V$ is crucial not only for computing deterministic functions of the input but also for eventual consensus, as we show below.

[2]This assumes that inputs are unbounded. If inputs are bounded from above or below, then a node knows it has the true minimum or maximum if it has heard the smallest or largest possible value (respectively). However, if this smallest or largest value is not present then the node cannot halt until it hears from everyone.

# 4. CONSENSUS AND CAUSALITY

In this section we show that when nodes do not have an initial upper bound on the count that is tight to within a factor of 2 of the true count, eventual consensus is in some sense equivalent to knowing when $\mathsf{past}_{(u,t)}(0) = V$. Specifically, for either the all-zero or all-one input assignment (or both), no node can decide until it hears from all the other nodes.

For simplicity, the statement we include here applies only to *comparison-based* algorithms, in which nodes can only compare UIDs to each other (but they cannot, e.g., execute a different program based on the UID they are assigned).

For $i \in \{0,1\}$, let $\sigma_{V,i}$ denote the signature where all nodes in $V$ receive $i$ as their input, and the upper bound on the count is $2n$.

THEOREM 4.1. *If nodes are given an upper bound on the count that is loose to a factor of at least 2, then for any comparison-based algorithm there is an $i \in \{0,1\}$ such that in any execution $G = (V,E,\sigma_{V,i})$, no node $u$ can decide at time $t$ if $\mathsf{past}_{(G,u,t)}(0) \neq V$.*

PROOF SKETCH. Suppose not. Then there exist executions $G_i = (V_i, E_i, \sigma_i)$ for $i = 0,1$, such that $\sigma_i$ assigns to all nodes of $V_i$ input $i$, the sets of UIDs used in $\sigma_0$ and $\sigma_1$ are disjoint, and there exist nodes $u_i, v_i$ and times $t_i$ such that $u_i$ decides at time $t_i$, even though $v_i \notin \mathsf{past}_{(G_i,u_i,t_i)}(0)$ (that is, $u_i$ does not hear from $v_i$ before it decides).

Because $u_0$ and $u_1$ do not hear from all the nodes in their respective executions, we can "stitch together" $G_0$ and $G_1$ without these nodes noticing. Consider the execution $H = (V_0 \cup V_1, E_H, \sigma_0 \cup \sigma_1)$, where for all $s \geq 1$ we set $E_H(s) := E_0(s) \cup E_1(s) \cup \{v_0, v_1\}$. In $H$, nodes are provided $2n$ as an upper bound on the count. Because $u_0$ and $u_1$ do not hear from $v_0$ and $v_1$ respectively, they cannot distinguish $H$ from $G_0$ and $G_1$ respectively, and they each decide the same as they would in the original execution. But in $G_0$ all nodes must decide 0, including $u_0$, and in $G_1$ all nodes must decide 1, including $u_1$; therefore agreement is violated in $H$. □

The assumption that the upper bound provided to the nodes is loose to within a factor of 2 is nearly tight: if nodes have access to an upper bound $N < 2(n-1)$, then the claim no longer holds, and nodes can halt without being causally influenced by everyone on both the all-zeroes and all-ones input assignments. One simple protocol illustrating this is the one where nodes decide on the majority input. To know that it has the true majority value $v$, it is sufficient for a node to hear of $\lfloor N/2 \rfloor + 1$ copies of $v$ in the input; when $N < 2(n-1)$ we have $\lfloor N/2 \rfloor + 1 < n$, so a node can sometimes decide before it has heard from all the nodes, even in the case of the all-zeroes or the all-ones input assignment.

# 5. COMPUTING COMMON KNOWLEDGE

As noted in Section 2, simultaneous coordination is closely related to common knowledge: a simultaneous action can only be performed when it is common knowledge that it is being performed. To understand simultaneous consensus in dynamic networks, we characterize the time required to achieve common knowledge.

The results in this section hold for common knowledge in general; see Section 6.1 for a discussion of how they apply to simultaneous consensus. Roughly speaking, we prove the following:

- Even if $n$ is known *a priori*, it takes $n-1$ rounds to acquire common knowledge of any fact that is not "trivially common knowledge".[3]
- If $n$ is not known *a priori*, then it takes $n$ rounds to acquire common knowledge of any fact about time 0 that is not initially common knowledge (such as $n$ itself).

For simplicity, we focus here on facts pertaining to time 0, such as the inputs to consensus. However, the result holds for other times as well; any fact about time $t$ cannot become common knowledge until time $t+n-1$.

Recall that a fact is common knowledge in $(G,t)$ iff it holds at all points $(H,t) \sim (G,t)$ in the similarity component of $(G,t)$. To prove the result above, we show that we can change any aspect of the dynamic graph $G$ at times $0,\ldots,n-2$ (and in particular, in rounds $1,\ldots,n-2$) while still remaining inside the similarity component of $(G,t)$. Formally, we show the following.

THEOREM 5.1. *For any full-information protocol,*

1. $(G,t) \sim (H,t)$ *for all* $t \le n-2$*; and*
2. *If* $n$ *is not known a priori then in addition,* $(G,n-1) \sim (H,n-1)$.

PROOF SKETCH. The main concept in the proof is *hiding*: given a set $X \subseteq V$, times $t' \le t$ and a node $u \in V$, we say that $X$ *at time* $t'$ *can be hidden from* $(u,t)$ if there is a point $(G',t) \sim (G,t)$ such that $\mathsf{past}_{(G',u,t)}(t') \cap X = \emptyset$. Hiding $X$ at time $t'$ means that we move inside the similarity component of $(G,t)$ to a point $(G',t) \sim (G,t)$ where node $u$ knows nothing about the states of the nodes in $X$ from time $t'$ onwards. Once we have done this, we can add or remove any edges adjacent only to nodes in $X$ in round $t'$, while still remaining in the similarity component of $(G,t)$, because $u$ does not learn of these changes by time $t$.

To prove the theorem, we show by induction on $k \le n-2$ that for any set $X$ of size at most $n-k-1$ and for any node $u \notin X$, set $X$ at time $t-k$ can be hidden from $(u,t)$, without altering any round preceding time $t-k-1$. We hide sets of *decreasing size* as we go back in time; essentially, we "use up" one node for each round we go back. The case where $k = n-2$ yields the theorem, since it shows that we can hide any single node at time $t-(n-2)$, and then change its state. In particular, we can hide any node at time 0 from any other node at time $n-2$, so the state of no node is common knowledge at time $n-2$. Moreover, if $n$ is not known *a priori*, then we can hide any node $w$ at time 1 from some node $u \neq w$ at time $n-1$ and proceed to add more nodes to the network, as in the proof of Lemma 3.1. By adding more nodes we can increase the dynamic diameter of the network to more than $n-1$, which again shows that the state of no node is common knowledge (nor is the size of the network common knowledge).

To hide a set $X$ at time $t-k$ we must remove all edges from nodes in $X$ to node $u$ and to other nodes that $u$ is causally influenced by in rounds $t-k+1, t-k+2,\ldots,t$. ("Removing" here means that we move to a point $(G',t) \sim (G,t)$ where these edges do not exist, by first hiding both endpoints of the edge at time $t-k+1$ from some node at time $t$.) To ensure that connectivity is preserved, before we remove edges we choose some node $w \notin X \cup \{u\}$ and add edges between $w$ and all nodes in the graph. Then we remove all edges from nodes in $X$ to all nodes except $X \cup \{w\}$. In the resulting graph, only nodes in $X \cup \{w\}$ hear from nodes in $X$ in round $t-k+1$. Our final step is to use the induction hypothesis to hide $X \cup \{w\}$ at time $t-k+1$ from $(u,t)$, so that we have $\mathsf{past}_{(u,t)}(t-k) \cap X = \emptyset$. □

An immediate consequence of Theorem 5.1 is that all initial values become common knowledge precisely at time $n-1$ if $n$ is known *a priori*. Thus, a simultaneous consensus protocol that is all-case optimal can be designed. It decides at time $n-1$ in all executions, and no protocol for this task can ever decide earlier. In fact, Theorem 5.1 implies that simultaneously acting based on any nontrivial function of the initial values can be done at time $n-1$ (when $n$ is known), and this is all-case optimal.

We also note that Theorem 5.1 implies that solving simultaneous consensus is as hard as computing an upper bound on the count, because a simultaneous consensus protocol can only decide at time $t$ if $t \ge n$ represents an upper bound on the count.

# 6. $\Delta$-COORDINATED CONSENSUS

Since simultaneous consensus is expensive and requires $n-1$ rounds even in very well-behaved executions, it is interesting to consider a trade-off between the performance of the consensus algorithm and the degree of coordination it achieves. To this end we consider the following problem:

DEFINITION 4 ($\Delta$-COORDINATED CONSENSUS). *A protocol solves* $\Delta$-coordinated consensus *if it solves consensus, and in addition, all nodes decide no later than* $\Delta$ *rounds after the first node decides.*

In the sequel we assume, unless stated otherwise, that the count is initially known. (An upper bound on the count can be used instead, or one can combine the algorithm in this section with the criterion from Section 3.)

One might expect that $\Delta$-coordinated consensus should not be much easier than simultaneous consensus. For example, when $\Delta = 1$, we require all nodes to decide within one round of each other; it seems that if we can achieve this, then simultaneous coordination can be achieved at not much extra cost (a cost of $\Delta$ additional rounds, perhaps). Indeed, in the worst case this expectation is borne out by the following theorem.

THEOREM 6.1. *For any* $\Delta$*-coordinated consensus algorithm, there exists an execution in which no node decides before round* $n-\Delta-1$*, even when* $n$ *is known* a priori.

PROOF. Suppose that there exists a $\Delta$-coordinated consensus algorithm $\mathcal{A}$, such that in every execution some node decides before time $R < n-\Delta-1$. Then in $\mathcal{A}$, *all* nodes decide no later than time $R+\Delta < n-1$ in every execution. We can obtain an algorithm for simultaneous consensus in fewer than $n-1$ rounds by simply having each node run $\mathcal{A}$ and output $\mathcal{A}$'s decision value at time $R+\Delta < n-1$, contradicting the lower bound from Section 5. □

This result shows the existence of only one "bad" execution where no node can decide until time $n-\Delta-1$. Given the general similarity between $\Delta$-coordinated consensus and simultaneous consensus, one might expect that a $\Delta$-coordinated consensus protocol would *never* be able to decide before time $n-\Delta-1$ (just as simultaneous consensus can never decide before time $n-1$). However, we now show that even in 1-coordinated consensus, nodes can sometimes decide significantly earlier than time $n-\Delta-1$. Consider the following protocol.

[3]For example, the current round number is trivially common knowledge.

*Clear-Majority Protocol.* Fix some integer $k_{\max}$, and for each $k = 1, \ldots, k_{\max}$, let $t_k := n - k \cdot \Delta - 1$. In each round the nodes forward the set of all node UIDs they have heard from so far, along with the input to each node. At time $t_k$, an undecided node decides $v$ iff it has heard of at least $\lfloor n/2 \rfloor + 1 + \binom{k}{2}\Delta$ inputs equal to $v$. Finally, at time $n-1$, all the nodes know all the inputs; at this point any undecided node decides on the majority input (breaking ties in some consistent way if there is no majority).

LEMMA 6.2. *The clear-majority protocol solves $\Delta$-coordinated consensus. Furthermore, when the fraction of identical inputs is at least $(1/2 + \epsilon)n$ for some constant $\epsilon$, and if $\Delta \leq (\epsilon n - 1)/2$, all nodes can decide after $n - \Theta(\sqrt{n\Delta})$ rounds.*

PROOF. Agreement and validity follow immediately from the fact that nodes always decide on the majority value (or, if there is no majority value, all nodes reach time $n-1$ and decide in some consistent way). To show that the protocol is $\Delta$-coordinated, suppose that in some execution, the earliest node $u$ decides on value $v$ at time $t_k$. We must show that all nodes decide no later than time $t_k + \Delta = t_{k-1}$.

Because the communication graph in every round is connected, for all $s \leq n-1$, at time $n - s - 1$ in the execution each node has heard all but at most $s$ of the inputs. In particular, by time $t_{k-1} = n - (k-1)\Delta - 1$ each node has heard all but $(k-1)\Delta$ of the inputs. Since $u$ decides $v$ at time $t_k$, the input assignment contains at least $\lfloor n/2 \rfloor + 1 + \binom{k}{2}\Delta$ values equal to $v$, and hence by time $t_{k-1}$ each node hears at least $\lfloor n/2 \rfloor + 1 + \binom{k}{2}\Delta - (k-1)\Delta = \lfloor n/2 \rfloor + 1 + \binom{k-1}{2}\Delta$ inputs equal to $v$. Thus, all nodes that do not decide at time $t_k$ decide $v$ at time $t_{k-1} = t_k + \Delta$, as required.

Now suppose that for some constant $\epsilon$, the input assignment contains at least $(1/2 + \epsilon)n$ copies of some value $v$. By time $t_k = n - k \cdot \Delta - 1$ each node hears all but $k \cdot \Delta$ of the input values, i.e., at least $(1/2+\epsilon)n - k \cdot \Delta$ copies of $v$. If $\Delta \leq (2\epsilon n - 1)/4$, we set $k_{\max} = \lfloor \sqrt{(2\epsilon n - 1)/\Delta} - 1 \rfloor$, and then simple algebra shows that $(1/2+\epsilon)n - k_{\max} \cdot \Delta \geq \binom{k_{\max}}{2} + \lfloor n/2 \rfloor + 1$; thus, by time $t_{k_{\max}}$, each node hears sufficiently many copies of $v$ to decide. For this value of $k_{\max}$ we have $t_{k_{\max}} = n - \Theta(\sqrt{n\Delta})$. □

The clear-majority protocol can be viewed as an instance of a more general scheme, in which nodes decide as soon as they know that everyone else will decide the same value within $\Delta$ rounds. Using this abstract scheme, any eventual consensus algorithm can be transformed into a $\Delta$-coordinated consensus protocol as follows. Let "$\mathcal{A} = v$" stand for the formula that asserts that $(G, t)$ is $v$-valent with respect to algorithm $\mathcal{A}$ (that is, in any possible extension of the first $t$ rounds of $G$, all nodes decide $v$). Let $K_u^{@t}\varphi$ denote the formula that means "node $u$ knows that at time $t$ fact $\varphi$ will hold", and let $E^{@t}\varphi := \bigwedge_{u \in V} K_u^{@t}\varphi$. Now we can state the protocol:

*The $\Delta$-Ladder.* Given an eventual consensus algorithm $\mathcal{A}$ in which all nodes decide no later than round $n-1$, we first transform $\mathcal{A}$ into a full-information algorithm $\mathcal{A}'$. Nodes execute $\mathcal{A}'$, but do not immediately output its decisions. Instead, each undecided node $u$ evaluates the following decision rules at each decision point $t_k = n - \Delta \cdot k - 1$ (the rules are given here in reverse order w.r.t. the time each rule is evaluated):

- Decide $v$ at time $n-1$ if $(\mathcal{R}(\mathcal{A}'), G, n-1) \vDash K_u(\mathcal{A}' = v)$, that is, if it is known that the run is $v$-valent for $\mathcal{A}'$.
- Decide $v$ at time $n - \Delta - 1$ if
$$(\mathcal{R}(\mathcal{A}'), G, n - \Delta - 1) \vDash K_u E^{@(n-1)}\left(\mathcal{A}' = v\right),$$
that is, if it is known that everyone will decide $v$ no later than time $n-1$.
- Decide $v$ at time $n - 2\Delta - 1$ if
$$(\mathcal{R}(\mathcal{A}'), G, n - 2\Delta - 1) \vDash K_u E^{@(n-\Delta-1)} E^{@(n-1)}\left(\mathcal{A}' = v\right),$$
that is, if it is known that everyone will know at time $n - \Delta - 1$ that everyone will decide $v$ no later than time $n-1$.
- ...

In general, at time $n - k \cdot \Delta - 1$, a node decides $v$ if it has not decided already and

$$(\mathcal{R}(\mathcal{A}'), G, n - k \cdot \Delta - 1) \vDash K_u E^{@(n-(k-1)\Delta-1)} \ldots E^{@(n-\Delta-1)} E^{@(n-1)}\left(\mathcal{A}' = v\right).$$

It is easy to see that any instantiation of this scheme satisfies $\Delta$-coordinated consensus; this is in some sense the optimal strategy. However, it requires nodes to keep track of information about the full dynamic graph, and to evaluate complex knowledge criteria; the clear-majority protocol uses less precise rules, but they are simpler and easier to evaluate. In general, any approximation for the knowledge criteria above can be used, as long as the same approximation is applied consistently at each decision point $n - k \cdot \Delta - 1$.

*Approximate $\Delta$-Ladder.* Let $\mathcal{A}$ be an eventual consensus algorithm with round complexity at most $n-1$, let $k_{\max} \in \mathbb{N}$, and fix a collection $\left\{\Phi_u^{k,v}\right\}_{u \in V, k \in [k_{\max}], v \in \{0,1\}}$ of *local* knowledge formulas, such that $u$ can evaluate the satisfaction of $\Phi_u^{k,v}$ based on its local state. These formulas represent the decision rules, and they must satisfy:

(a) **Consistency:** for all $u$,
$$\mathcal{R}(\mathcal{A}) \vDash \Phi_u^{0,0} \rightarrow (\mathcal{A} = 0) \text{ and } \mathcal{R}(\mathcal{A}) \vDash \Phi_u^{0,1} \rightarrow (\mathcal{A} = 1).$$

(b) **Timeliness:** for all executions $G$,
$$(\mathcal{R}(\mathcal{A}), G, n-1) \vDash \Phi_u^{0,0} \vee \Phi_u^{0,1}.$$

(c) **Coordination:** for all $1 \leq k \leq k_{\max}$ and $v \in \{0,1\}$, if $(\mathcal{R}(\mathcal{A}), G, n - k \cdot \Delta - 1) \vDash K_u \Phi_u^{k,v}$, then
$$(\mathcal{R}(\mathcal{A}), G, n - (k-1)\Delta - 1) \vDash \bigwedge_{w \in V} K_w \Phi_w^{k-1,v}.$$

Then a protocol for $\Delta$-coordinated consensus is given by the following: the nodes simulate algorithm $\mathcal{A}$ with their local inputs, but do not output $\mathcal{A}$'s decisions immediately. Instead, for each $k = k_{\max}, \ldots, 1$, a node $u$ (which has not decided already) decides $v$ at time $n - k \cdot \Delta - 1$ if $(\mathcal{R}, n - k \cdot \Delta - 1) \vDash K_u \Phi_u^{k,v}$.

LEMMA 6.3. *Any instantiation of the $\Delta$-ladder protocol solves $\Delta$-coordinated consensus.*

Finally, let us give another instantiation of the approximate $\Delta$-ladder, which decides quickly in graphs where all nodes hear from everyone quickly.

*Dynamic Diameter-Based Protocol.* Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be any function that satisfies $f(0^n) = 0$ and $f(1^n) = 1$. Nodes always forward their full view of the execution so far. At time $n - k \cdot \Delta - 1$, a node decides $f(\bar{x})$ if it knows that the input assignment is $\bar{x}$, and it knows that there exists some $D$ such that the dynamic graph had a diameter of at most $D$ until time $(k-1)D$ (where $(k-1)D \leq n - k \cdot \Delta - 1$).

To see that this decision rule is consistent with the requirements, suppose that the rule for deciding at time $n - k \cdot \Delta - 1$ holds at node $u$, i.e., $u$ knows the input assignment and dynamic diameter of the graph is at most $D$ until time $(k-1)D$. If $k \geq 2$, then for any two nodes $w, w'$ we have $(w, (k-2)D) \rightsquigarrow (w', (k-1)D)$; consequently at time $(k-1)D$, all nodes know that the dynamic graph had diameter at most $D$ up to time $(k-2)D$ and all nodes know the input assignment. When time $n-(k-1)\Delta-1$ arrives the decision rule for $k-1$ is satisfied. If $k = 1$, then the decision rule for time $n-(k-1)\Delta-1 = n-1$ holds trivially, because it only requires nodes to know the input assignment.

The value we choose for $k_{\max}$ should satisfy $(k_{\max}-1)D < n - k_{\max} \cdot \Delta - 1$, otherwise the decision rule for time $t_{k_{\max}}$ would be unsatisfiable. If we choose $k_{\max} \geq \lfloor n/(D+\Delta) \rfloor$, nodes can stop as early as time $n - k_{\max} \cdot \Delta - 1 < n(1-\Delta/(D+\Delta)) + \Delta = nD/(D+\Delta) + \Delta$. For example, if the communication graph is always a clique, then the running time is slashed by a factor of $\Delta$. Note that the algorithm does not commit in advance to some diameter $D$; nodes always evaluate the stopping condition with respect to all $D$, and check if some bound $D$ satisfies the requirement.

## 6.1 Lower Bounds

In the following, we prove two lower bounds that complement the upper bounds from the previous section.

In Section 5 we proved a lower bound on common knowledge. Viewed through the lens of simultaneous consensus, we can interpret our strategy as follows: to show that simultaneous consensus cannot decide in $(G,t)$, we showed that there exist two points $(G_0,t)$ and $(G_1,t)$ such that

(a) In $G_0$ the input to all nodes is 0, and in $G_1$ the input to all nodes in 1; and

(b) $(G_0,t) \sim (G,t) \sim (G_1,t)$.

To briefly review the argument, suppose that some node decides $v$ in $(G,t)$. Consider the path between $(G,t)$ and $(G_{1-v},t)$ in the similarity graph; denote this path by

$$(G,t) = (H_0,t) \sim_{u_1} (H_1,t) \sim_{u_2} \ldots \sim_{u_\ell} (H_\ell,t) = (G_{1-v},t).$$

We show that some node decides $v$ in $(G_{1-v},t)$, violating validity, by employing the following argument at each step $i = 1,\ldots,\ell$ along the path:

1. Some node $w$ decides $v$ in $(H_i,t)$; therefore,
2. From simultaneity and agreement, node $u_i$ decides $v$ in $(H_i,t)$; therefore,
3. Node $u_i$ also decides $v$ in $(H_{i+1},t)$, because it cannot distinguish $(H_{i+1},t)$ from $(H_i,t)$.

This argument hinges on simultaneity, and we cannot employ it as-is to prove lower bounds on $\Delta$-coordinated consensus. However, $\Delta$-coordination allows us to make the following weaker argument:

1. Some node $w$ decides $v$ in $(H_i,t)$; therefore,
2. **From $\Delta$-coordination and agreement, node $u_i$ decides $v$ in $(H_i, t+\Delta)$;**[4] therefore,
3. If $(H_i,t+\Delta) \sim_{u_i} (H_{i+1},t+\Delta)$, node $u_i$ also decides $v$ in $(H_{i+1},t+\Delta)$, because it cannot distinguish $(H_{i+1},t+\Delta)$ from $(H_i,t+\Delta)$.

The key difference is that unlike before, now we have to pay for each step we take in the similarity graph; our lower bound is weakened by $\Delta$ rounds at each step, as we move forward in time.

[4]Technically, there exists some $t' \leq t+\Delta$ such that $u_i$ decides $v$ in $(H_i,t')$. The essential property is that by time $t+\Delta$ node $u_i$ has already decided $v$ in $H_i$.

This reasoning, applied repeatedly, yields the following lemma.

LEMMA 6.4. *Let $G, G_0, G_1$ be runs, where in $G_0$ and $G_1$ all nodes receive input 0 and 1, respectively. Assume that for some $\ell \geq 1$ and time $t$, the following two sequences of steps (i.e., edges) exist in the similarity graph:*

$$\begin{aligned}(G,t+\Delta) &\sim_{u_1} (H_1,t+\Delta),\\ (H_1,t+2\Delta) &\sim_{u_2} (H_2,t+2\Delta),\\ &\ldots\\ (H_{\ell-1},t+\ell\Delta) &\sim_{u_\ell} (G_0,t+\ell\Delta)\end{aligned}$$

*and*

$$\begin{aligned}(G,t+\Delta) &\sim_{u'_1} (H'_1,t+\Delta),\\ (H'_1,t+2\Delta) &\sim_{u'_2} (H'_2,t+2\Delta),\\ &\ldots\\ (H'_{\ell-1},t+\ell\Delta) &\sim_{u'_\ell} (G_1,t+\ell\Delta);\end{aligned}$$

*Then in any $\Delta$-coordinated consensus algorithm, no node can decide by time $t$ in $G$.*

PROOF SKETCH. As outlined above, if some node decides $v$ in $(G,t)$, we show by induction on the path length ($\ell$) that some node decides $v$ in $(G_{1-v},t+\ell\Delta)$, violating validity. □

The condition of Lemma 6.4 involves many different times, $t+\Delta, t+2\Delta, \ldots, t+\ell\Delta$. A simpler condition that implies the lemma can be obtained by replacing all times with the last time, $t+\ell\Delta$ (to still obtain a lower bound for time $t$). We show the existence of the following two walks in the similarity graph:

$$\begin{aligned}(G,t+\ell\Delta) = (H_0,t+\ell\Delta) &\sim_{u_1} (H_1,t+\ell\Delta) \sim_{u_2} \ldots\\ &\sim_{u_\ell} (H_\ell,t+\ell\Delta) = (G_0,t+\ell\Delta), \text{ and}\\ (G,t+\ell\Delta) = (H'_0,t+\ell\Delta) &\sim_{u'_1} (H'_1,t+\ell\Delta) \sim_{u'_2} \ldots\\ &\sim_{u'_\ell} (H'_\ell,t+\ell\Delta) = (G_1,t+\ell\Delta).\end{aligned}$$

This only strengthens the condition, since $(G,t) \sim_u (G',t)$ implies $(G,t') \sim_u (G,t')$ for all $t' \leq t$. Thus the existence of these walks is sufficient to apply Lemma 6.4.

When a full-information protocol is used, all information about the input becomes common knowledge at time $n-1$; therefore we cannot hope to have $t+\ell\Delta > n-1$ when we apply (the simplified version of) Lemma 6.4. In order to maximize $t$ and obtain the strongest possible lower bound, we must minimize $\ell$; that is, we must find short walks in the similarity graph. Since our ultimate goal is to span between $G$ and two runs where the inputs are 0 and 1 (respectively), the walk should allow us to flip the inputs of as many nodes as possible in each step.

*Lower bound for static paths.* We now apply the strategy described above to obtain an $n - O(\Delta^{0.28} n^{0.72})$ lower bound in static paths of length $n$. A path is a natural candidate for proving strong lower bounds: we can flip the inputs of nodes at one end of the path, and the nodes at the other end do not find out for a long time. However, to use Lemma 6.4, we must be able to flip the inputs of *all* the nodes in the network, not just the nodes at the ends of the path. Thus, we start with some path $u_1,\ldots,u_n$, and flip the inputs in some prefix $u_1,\ldots,u_\beta$ of the path; node $u_n$ cannot distinguish the two cases until time roughly $n-\beta$. Then we find a short walk in the similarity graph from our original path $u_1,\ldots,u_n$ to a new path, $u_{\beta+1},\ldots,u_{n+\beta}$ (i.e., we preserve the order of nodes, but we rotate the path so that now it starts at $u_{\beta+1}$;

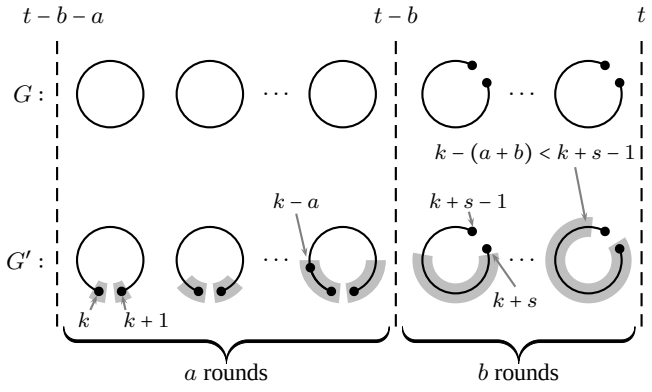


Figure 2: Illustration of Lemma 6.5. The shaded arcs indicate which nodes can distinguish $G$ from $G'$. Switching from $P_k$ to $P_{k+s}$ "cuts off" the spread of information about the missing edge, and prevents it from reaching node $k+s-1$.

note that here and in the sequel, node indices are given modulo $n$. The parameter $\beta$ will be fixed later). Now we can flip the inputs of nodes $u_{\beta+1},\ldots,u_{2\beta}$; node $u_\beta = u_{n+\beta}$, located at the end of the new path, cannot distinguish until time roughly $n-\beta$. We require at most $\lceil n/\beta\rceil$ such steps to flip any input assignment into either the all-zero or the all-one input assignment.

The strength of the lower bound is determined by the length of the walk from one path, $u_{i\cdot\beta+1},\ldots,u_{n+i\cdot\beta}$, to the next path, $u_{(i+1)\cdot\beta+1},\ldots,u_{n+(i+1)\cdot\beta}$. To construct the walk we use an intricate recursion. During the walk between paths we do not change any input values; in the sequel we focus on the dynamic graph and assume some fixed signature for all the executions we consider.

Let $P_k := u_{k+1},\ldots,u_{n+k}$ denote the path starting at node $u_{k+1}$, and let $C$ denote the cycle $u_1,\ldots,u_n,u_1$. It is convenient to use the cycle $C$ to bridge between paths: we cannot remove any edge of a path without violating connectivity, but a cycle is 2-vertex connected, so we can drop any of its edges. Intuitively, to move from a path $P_k$ to a different path $P_{k+s}$ (for $s\neq 0$), we first close $P_k$ to form the cycle $C$, then drop edge $\{v_{k+s},v_{k+s+1}\}$ to obtain $P_{k+s}$. The following lemma shows how we can move from a path to the cycle while ensuring that some node cannot distinguish the two executions; it represents an intermediate step which will be used later to move between two paths.

LEMMA 6.5. *Let $k\in\{0,\ldots,n-1\}$, $s\in\mathbb{Z}$, and let $0\le a\le|s|$ and $b\ge 0$ satisfy $a+b<n-s$. Fix a time $b<t\le n-1$. Consider two graphs $G,G'$ that agree on the first $\max\{0,t-b-a\}$ rounds, such that*

- *In rounds $r\in[t-b-a+1,t-b]$, $G(r)=C$ and $G'(r)=P_k$;*
- *In rounds $r\in[t-b+1,t]$, $G(r)=G'(r)=P_{k+s}$.*

*Then $(G,t)\sim_{u_{k+s-1}}(G',t)$.*

PROOF SKETCH. Assume that $s\ge 0$ (the other case is symmetric). At each time $r=(t-b-a+1)+i$ for $i=1,\ldots,a$, only nodes $u_{k-i},\ldots,u_{k+i+1}$ might have learned of the missing edge, $\{u_k,u_{k+1}\}$. Thus, at time $t-b$, only nodes $u_{k-a},\ldots,u_{k+a+1}$ can distinguish $G$ from $G'$. Next, both graphs switch to $P_{k+s}$ where the distance between any node $u_{k-a},\ldots,u_{k+a+1}$ and node $u_{k+s-1}=u_{k-(n-s+1)}$ is at least $n-s-a$. Since $b<n-s-a$, node $u_{k+s-1}$ does not learn of the difference by time $t$ (see Fig. 2). □

Next, we show how to use Lemma 6.5 to recursively transform a suffix of the execution from one path $P_k$ to a different path $P_{k+s}$. (Our eventual goal is to transform the entire execution from one path to another.) Let $d((G,t),(G',t))$ denote the distance between $(G,t)$ and $(G',t)$ in the similarity graph.

LEMMA 6.6. *Fix a time $0\le t\le n-1$ and a value $1\le\beta\le n-1$. Let $G,G'$ be dynamic graphs that agree up to time $t-(n-1-\beta)$, such that in rounds $r\in[t-(n-1-\beta)+1,t]$, $G(r)=P_k$ and $G'(r)=P_{k'}$ (for some $k,k'$). Then $d((G,t),(G',t))\le 9(n/\beta)^{\log_2 3}$.*

PROOF SKETCH. Define $\ell_\beta := \lceil\log_2(n/\beta)\rceil$. We show by induction on $\ell_\beta$ that $d((G,t),(G',t))\le 3^{\ell_\beta+1}-1$. The claim then follows, because

$$3^{\ell_\beta+1}-1\le 3^{\log_2(n/\beta)+2}=9\left(\frac{n}{\beta}\right)^{\log_2 3}.$$

Let us denote $d_\beta := 3^{\ell_\beta}+1$. Note that we are transforming the suffix $[t-(n-1-\beta)+1,t]$ of the execution; hence, smaller values of $\beta$ (or equivalently, larger values of $\ell_\beta$) are "harder" because they require us to transform a longer suffix.

The induction base is straightforward; it is omitted here. For the step we use Lemma 6.5. Set $a=\beta$ and $b=n-1-2\beta$. Given static graphs $H_1$, $H_2$, let $G[H_1,H_2]$ be the dynamic graph defined by

$$G[H_1,H_2](r) := \begin{cases} G(r) & r\le t-(n-1-\beta),\\ H_1 & t-(n-1-\beta)<r\le t-b,\\ H_2 & t-b<r\le t.\end{cases}$$

Since $b=n-1-2\beta$ and $\ell_{2\beta}=\ell_\beta-1$, the induction hypothesis shows that for any graph $H$ and for any two paths $P_q,P_{q'}$ we have $d((G[H,P_q],t),(G[H,P_{q'}],t))\le d(2\beta)$. Further, Lemma 6.5 shows that $d((G[P_q,P_{q+\beta}],t),(G[C,P_{q+\beta}],t))=1$ for any $q$ (because these points are indistinguishable to some node). Thus, we construct the following walk (see Fig. 3):

$$\begin{aligned}(G,t)=(G[P_k,P_k],t)&\overset{d(2\beta)}{\underset{\text{I.H.}}{\to}}(G[P_k,P_{k+\beta}],t)\overset{1}{\underset{\text{Lem. 6.5}}{\to}}\\ (G[C,P_{k+\beta}],t)&\overset{d(2\beta)}{\underset{\text{I.H.}}{\to}}(G[C,P_{k'+\beta}],t)\overset{1}{\underset{\text{Lem. 6.5}}{\to}}\\ (G[P_{k'},P_{k'+\beta}],t)&\overset{d(2\beta)}{\underset{\text{I.H.}}{\to}}(G[P_{k'},P_{k'}],t)=(G',t).\end{aligned}$$

The length of the walk is at most $3d(2\beta)+2=3(3^{\ell_\beta}-1)+2=3^{\ell_\beta+1}-1$. □

THEOREM 6.7. *In the static line graph, for any input assignment, no $\Delta$-coordinated consensus algorithm can decide by time $n-O(\Delta^{\frac{1}{2+\log_2 3}}n^{1-\frac{1}{2+\log_2 3}})\approx n-O(\Delta^{0.28}n^{0.72})$.*

PROOF SKETCH. Let $\sigma$ be any signature, and let $\sigma_0,\sigma_1$ be the corresponding signatures where all nodes receive input 0 or 1, respectively. Let $P^{k,\tau}$ denote the dynamic graph defined by $P^{k,\tau}(r)=P_k$ for all $r$, using signature $\tau\in\{\sigma,\sigma_0,\sigma_1\}$. Also set $t:=n-2\beta-1$.

For $v\in\{0,1\}$, we span between $(P^{1,\sigma},t)$ and $(P^{n-\beta,\sigma_v},t)$ by repeating the following steps $O(n/\beta)$ times:

1. Flip the inputs of the leftmost $\beta$ nodes on the path to $v$ in one move (the endpoint of the line cannot distinguish),
2. Applying Lemma 6.6, move from our current point $(P^{k,\tau},t)$ to $(P^{k+\beta},t)$ in $O((n/\beta)^{\log_2 e})$ steps.

The total length of the walk is $\ell=O((n/\beta)\cdot(n/\beta)^{\log_2 e})=O((n/\beta)^{1+\log_2 3})$. Now, fix $\beta$ such that $\beta\ge c\cdot\Delta\cdot(n/\beta)^{1+\log_2 e}$. For this setting of the parameters, Lemma 6.4 shows that no node decides by time $t-\ell\Delta=n-O(\Delta^{\frac{1}{2+\log_2 3}}n^{1-\frac{1}{2+\log_2 3}})$. □

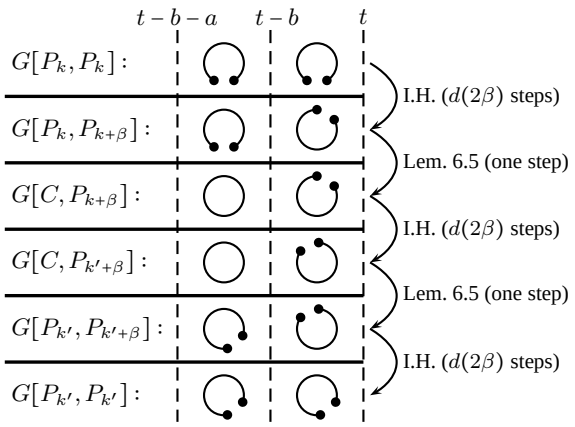


Figure 3: The recursion from Lemma 6.6. The two graphs shown for each step represent the communication graph for rounds $t-b-a, \ldots, t-b$ and for rounds $t-b+1, \ldots, t$.

The final theorem states that the dynamic diameter-based protocol is asymptotically optimal for diameters $D = O(\Delta)$, even for static executions. The proof uses Lemma 6.4 with $t + \ell\Delta \approx n/2$, that is, we construct a short walk in the indistinguishability graph for time roughly $n/2$ (recall that $t$ is the time for which we wish to show the lower bound, and $\ell$ is the length of the walk we construct in the similarity graph). We start with an execution whose first $t$ rounds are a static graph with diameter $D$, and the remaining $n/2-t$ are a static path. This $(n/2-t)$-round suffix means that the nodes at the end of the path require roughly $n$ rounds to learn what the communication graph was in each of the first $n/2-t$ rounds. In $\ell = O(t/D)$ steps in the similarity graph, we move from this execution to a static path. Because we need only maintain indistinguishability until time roughly $n/2$, once we have reached the static path we can flip the inputs of nodes $1, \ldots, n/2$ on the path in one step; node $n$ does not find out by time $n/2$. By repeating this process twice we can flip all the inputs. Since we have $\ell = O(t/D)$ and we are constrained by $t + \ell\Delta \leq n/2$ (as indistinguishability is only maintained until time $n/2$), we can apply Lemma 6.4 to obtain the lower bound at time $t = \Omega(nD/\Delta)$.

THEOREM 6.8. *For every $D = O(\Delta)$, there is a static graph $H = (V, E_H)$ with diameter at most $D$ such that for every $\Delta$-coordinated consensus algorithm, every input $\sigma$ and every dynamic graph $G = (V, E, \sigma)$ with $E(r) = E_H$ for all rounds $r$ until the first node decides, no node can decide at a time before $\Omega(nD/\Delta)$.*

## 7. REFERENCES


[1] C. Avin, M. Koucky, and Z. Lotker. How to explore a fast-changing world (cover time of a simple random walk on evloving graphs). In *Proc. 35th Coll. on Automata, Lang. and Programming (ICALP)*, pages 121–132, 2008.

[2] A. Bar-Noy and D. Dolev. Consensus algorithms with one-bit messages. *Distributed Computing*, 4:105–110, 1991.

[3] H. Baumann, P. Crescenzi, and P. Fraigniaud. Parsimonious flooding in dynamic graphs. In *Proc. 28th Symp. on Principles of Distributed Computing (PODC)*, pages 260–269, 2009.

[4] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *CoRR*, abs/1012.0009, 2010.

[5] A. Clementi, C. Macci, A. Monti, F. Pasquale, and R. Silvestri. Flooding time in edge-markovian dynamic graphs. In *Proc. 27th Symp. on Principles of Distributed Computing (PODC)*, pages 213–222, 2008.

[6] A. E. F. Clementi, A. Monti, F. Pasquale, and R. Silvestri. Broadcasting in dynamic radio networks. *J. Comput. Syst. Sci.*, 75(4):213–230, 2009.

[7] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. ACM*, 40(1):17–47, 1993.

[8] C. Dwork and Y. Moses. Knowledge and common knowledge in a Byzantine environment: crash failures. *Information and Computation*, 88(2):156–186, 1990.

[9] C. Dwork, D. Peleg, N. N Pippenger, and E. Upfal. Fault tolerance in networks of bounded degree. In *Proc. 8th Symp. on Theory of Computing (STOC)*, pages 370–379, 1986.

[10] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge*. MIT Press, Cambridge, MA, 2003.

[11] A. Fernández Anta, A. Milani, M. A. Mosteiro, and S. Zaks. Opportunistic information dissemination in mobile ad-hoc networks. In *Proc. 24th Conference on Distributed Computing (DISC)*, pages 374–388, 2010.

[12] A. Ferreira. Building a reference combinatorial model for MANETs. *IEEE Network Magazine*, 18(5):24–29, 2004.

[13] A. Ferreira, A. Goldman, and J. Monteiro. Performance evaluation of routing protocols for MANETs with known connectivity patterns using evolving graphs. *Wireless Networks*, 16(3):627–640, 2010.

[14] J. A. Garay and R. Ostrovsky. Almost-everywhere secure computation. In *Proc. 27th EUROCRYPT*, pages 307–323, 2008.

[15] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *J. ACM*, 37(3):549–587, 1990.

[16] F. Kuhn, C. Lenzen, T. Locher, and R. Oshman. Optimal gradient clock synchronization in dynamic networks. In *Proc. 29th Symp. on Principles of Distributed Computing (PODC)*, pages 430–439, 2010.

[17] F. Kuhn, T. Locher, and R. Oshman. Gradient clock synchronization in dynamic networks. In *Proc. 21st Symp. Paralellism in Algorithms and Architectures (SPAA)*, pages 270–279, 2009.

[18] F. Kuhn, N. Lynch, and R. Oshman. Distributed computation in dynamic networks. In *Proc. 42nd Symp. on Theory of Computing (STOC)*, pages 513–522, 2010.

[19] F. Kuhn and R. Oshman. Dynamic networks: models and algorithms. *SIGACT News*, 42:82–96, March 2011.

[20] T. Mizrahi and Y. Moses. Continuous consensus via common knowledge. *Distributed Computing*, 20(5):305–321, 2008.

[21] Y. Moses and M. R. Tuttle. Programming simultaneous actions using common knowledge. *Algorithmica*, 3:121–169, 1988.

[22] R. O'Dell and R. Wattenhofer. Information dissemination in highly dynamic graphs. In *Proc. 9th Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pages 104–110, 2005.

[23] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[24] N. Santoro and P. Widmayer. Time is not a healer. In *Proc. 6th Symp. on Theoretical Aspects of Computer Science (STACS)*, pages 304–313. Springer-Verlag, 1989.