Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

The firing squad problem revisited[☆]

Bernadette Charron-Bost[a], Shlomo Moran[b,*]

[a] École polytechnique, 91128 Palaiseau, France
[b] Department of Computer Science, Technion, Haifa, 32000, Israel

ARTICLE INFO

Article history:
Received 3 February 2018
Received in revised form 14 June 2019
Accepted 15 July 2019
Available online xxxx
Communicated by K. Censor-Hillel

Keywords:
Distributed algorithms
Firing squad
Dynamic graphs
Asynchronous start
Randomized algorithms

ABSTRACT

In the classical firing squad problem, an unknown number of nodes represented by identical finite states machines is arranged on a line and in each time unit each node may change its state according to its neighbors' states. Initially all nodes are passive, except one specific node located at an end of the line, which issues a fire command. This command needs to be propagated to all other nodes, so that eventually all nodes simultaneously enter some designated "firing" state.

A natural extension of the firing squad problem, introduced in this paper, allows each node to postpone its participation in the squad for an arbitrary time, possibly forever, and firing is allowed only after all nodes decided to participate. This variant is highly relevant in the context of decentralized distributed computing, where processes have to coordinate for initiating various tasks simultaneously.

The main goal of this paper is to study the above variant of the firing squad problem under the assumptions that the nodes are *infinite* state machines, and that the inter-node communication links can be changed arbitrarily in each time unit, i.e., are defined by a *dynamic graph*. In this setting, we study the following fundamental question: what connectivity requirements enable a solution to the firing squad problem?

Our main result is an exact characterization of the dynamic graphs for which the firing squad problem can be solved. When restricted to static directed graphs, this characterization implies that the problem can be solved if and only if the graph is strongly connected. We also discuss how information on the number of nodes or on the diameter of the network, and the use of randomization, can improve the solutions to the problem.

1. Introduction

Many distributed algorithms assume a *synchronous networked system*, in which computation is divided into communication *closed rounds*: any message sent at some round can be received only at that round. In this model it is typically assumed that each run of an algorithm is started by all nodes simultaneously, i.e., at the same round. For instance, most of synchronous consensus algorithms (e.g., [25,12,27]), as well as many distributed algorithms for dynamic networks (e.g., [17,18]) require synchronous starts.

In this paper, we justify this assumption of synchronous starts for dynamic networks with no central control that monitors the node activities, but with sufficient connectivity assumptions. Specifically, we study a generalization of the associated

☆ A preliminary version of this paper was accepted to the 35th International Symposium on Theoretical Aspects of Computer Science (STACS 2018).
* Corresponding author.
E-mail address: moran@cs.technion.ac.il (S. Moran).

https://doi.org/10.1016/j.tcs.2019.07.023
0304-3975/© 2019 Elsevier B.V. All rights reserved.

synchronization problem, classically referred to as the *firing squad problem*. This generalization considers a communication network of unknown size, in which messages are delivered along a set of edges that may change in each round. Each node is initially *passive*: it is part of the network and broadcasts heartbeats, also called *null* messages, without changing its state. It may then become *active* upon receiving a *start signal* at an unpredictable time, and start executing its code. The goal is to guarantee that if all nodes received start signals, then the nodes eventually synchronize by *firing* – i.e., entering a designated state for the first time – simultaneously.

We generalize the original firing squad problem in two ways: First, while the original formulation assumes that the network is defined by a fixed connected undirected graph, we allow for unidirectional communication links that may be added or deleted in each round. Second, in the original formulation, start signals are supposed to be *diffusive*, i.e., a passive node that receives a message from an active node must immediately become active. In our formulation start signals may be non-diffusive – a passive node that receives a message from an active node may remain passive for an unpredictable time. In other words, start signals are not necessarily correlated to the communication topology.

In this model, the firing squad problem is formally specified as follows:

FS1 (Validity): A node fires if and only if all nodes receive start signals.
FS2 (Simultaneity): All the nodes that fire, fire at the same round.

Observe that when start signals are diffusive and the network is strongly connected, if some node receives a start signal then eventually all nodes receive start signals, and hence FS1-2 coincides with the traditional definition of the firing squad problem [20]. Thus our problem specification is a strict generalization of the original one.

As a basic synchronization abstraction, the fulfillment of FS1 and FS2 above can be used in various types of situations to guarantee simultaneity: for distributed initiation (to force nodes to begin some computation in unison), for distributed termination (to guarantee that nodes complete their computation at the same round), or in real-time processing (where nodes have to carry out some external actions simultaneously). Another typical scenario that requires FS1 and FS2 is when some algorithm needs to be executed several times in a row, and the $(i+1)$st run should be started simultaneously, after all nodes terminated the ith run (see e.g. [5]). The latter two applications exemplify the relevance of the model of non-diffusive start signals as start signal may correspond to the termination of a local computation or to the availability of some input value.

In all the above examples, a firing squad algorithm is a synchronization tool used as a subroutine by some parent algorithm. For instance, consider the use of a firing squad algorithm on the top of some algorithm $\mathcal{A}$ for simulating simultaneous termination of $\mathcal{A}$. In this case, a node is passive if it has not yet completed the execution of $\mathcal{A}$, and the start signal on a node corresponds to the completion of $\mathcal{A}$ by this node. In this example a passive node sends null messages, just indicating that it has not completed the execution of $\mathcal{A}$.

It is easy to see that when the communication graph is permanently complete, the firing squad problem can be solved in one round after all nodes are active. In the opposite scenario where some node is permanently isolated, the problem is clearly unsolvable. This demonstrates a strong connection between the solvability and complexity of the firing squad problem, and the connectivity of the network. The primary aim of this paper is to explore this relation in the context of a dynamic topology.

The firing squad problem was originally studied in the context of automata theory (e.g., [20,21]). This model considers a finite but unknown number n of nodes which are connected in a line (or in some other specific topologies in more recent works – see e.g., [23,8]). Nodes are identical *finite* state machines whose number of states is independent of n, and at each time unit each node changes its state according to the states of its neighbors on the line. A start signal is given to a node located at one end of the line – the "general" – and then is propagated to the rest of the nodes so that all nodes have eventually to fire simultaneously. Thus the above model assumes diffusive start signals. The main challenges in this model are to reduce the number of states of the finite state machine and the time required to reach the firing state.

A natural question raised at this point is then the following: If nodes are not restricted to be finite state machines, but possess a full computational power equivalent to that of a Turing machine, what are the connectivity properties that are needed for solving the firing squad problem in this case?

The firing squad problem has also been studied in the context of fault tolerant distributed computations (e.g., [3,12,7]), and more recently in the context of self-stabilization (e.g., see [10,19]). This model also assumes that the nodes have full computational power, but otherwise the setting of the problem is different: The primary topology is a connected bidirectional graph and the number of nodes n is given. Links are reliable and at most f nodes may be faulty, for various types of faults, each of which includes crash failures. Hence the topology is weakly dynamic, in the sense that all the edges from non-faulty nodes are stable, and it may become non-strongly connected (e.g., in the case of crash failures). Moreover, the scope of the study of fault tolerant firing squad algorithms is limited to diffusive start signals. Finally, due to the lack of strong connectivity and the unpredictable behavior of faulty nodes, the simultaneity condition and to a larger extent the validity condition in this model ought to be modified: e.g., faulty nodes in the fault-tolerant firing squad problem are allowed not to fire simultaneously, while all nodes are required to fire simultaneously in FS2. For all these reasons, our results on the dynamic firing squad problem are incomparable to the ones on the fault-tolerant firing squad problem.

Contribution. In this paper we consider a set of an unknown number n of nodes possessing full computational power. Nodes have distinct identifiers, and they run identical codes in some precise sense that is discussed later. The inter-node communication is modelled by a *dynamic graph*, i.e., at each round, nodes communicate along directed edges of an arbitrary communication graph that may change continually and unpredictably from one round to the next. Communication is done by having each node broadcast at each round a message along the unknown set of its outgoing edges in this round. We examine various connectivity properties that hold, not necessary round by round, but globally over finite periods of consecutive rounds. In particular, these properties do not imply any stability of the links, as opposed to the failure model of at most f faulty nodes that guarantees stability of the links from at least $n - f$ nodes or several models of dynamic networks in distributed computing (e.g., see [17,1,26]) that assume the existence of a stable spanning tree in the network over every T consecutive rounds.

The main contribution of this paper is a characterization of the connectivity properties that enable to solve the firing squad problem in dynamic (and hence also in static) graphs. Our algorithms are valid for the generalized model assuming arbitrary start signals and arbitrary dynamic graph topology, while our impossibility results hold for previous restricted models which assume only diffusive start signals and bidirectional dynamic graphs.

On the positive side, we show that if the dynamic graph is guaranteed to be strongly connected within each T consecutive rounds, then the problem admits a solution where the local codes of the nodes all depend on T. Our solution requires at most linear time in the network size n, and uses messages of size $O(n \log n)$.

On the negative side, we show that under the sole assumption that such a constant T exists but is unknown, the problem becomes unsolvable. Moreover, the problem remains unsolvable in this case even when the number of nodes in the network is given and even in the restricted model of diffusive start signals. The above results imply that if at each round, the communication graph can be any member of a given set of directed graphs over the same set of nodes, then the firing squad problem is solvable if and only if each member in the set is strongly connected.

Our solution is obtained by combining two basic procedures: the first implements local virtual clocks whose values cannot exceed the *diameter of the dynamic graph* unless all nodes are active, and the second collects the identities of all the active nodes in the network. The idea is then that a node fires when the value of its virtual clock is sufficiently large compared to the number of active nodes it has heard of so far.

We also show that if an upper bound on the diameter of the dynamic graph is given, then the problem is solvable in time that is linear in that upper bound, using messages of size $O(\log n)$. This solution is applicable to anonymous networks where the nodes have no identifiers.

Unfortunately when the upper bound on the diameter is significantly larger than the network size, this solution is not satisfactory regarding its time complexity. In the case a polynomial bound on the network size is given, we conclude by giving a randomized algorithm for the firing squad problem with a linear time in the network size while messages size is in $O(\log n \log\log n)$.

2. The model

2.1. Distributed computations in the dynamic graph model

We consider a networked system with a set V of n nodes. Nodes have unique identifiers which are not mutually known, and the network size n is unknown as well. For complexity analysis it is assumed that the identifiers' length is logarithmic in n. Some of our algorithms do not use node identifiers, i.e., they work in *anonymous networks*. Furthermore, nodes run identical local algorithms, i.e., their codes do not depend on node identifiers (see the discussion on the algorithm $B_{c,T}$ in Section 5).

Computation proceeds in *communication closed rounds*: no node receives messages in round t that are sent in a round different from t. In round t $(t = 1, 2 \dots)$, each node broadcasts a message, receives messages from some nodes which are determined dynamically, and finally goes to its next state and proceeds to round $t + 1$. Importantly, we assume that nodes cannot access the number of the current round.

Each node u is initially *passive*: it is part of the network, and at each round it broadcasts a *null message*, indicating that it is in a passive state. It may, in a later round, become *active* and start running its local algorithm upon receiving a unique *start signal* at the beginning of some round $s_u \geqslant 1$, or may remain passive forever, in which case we let $s_u = \infty$. A run is *active* if all nodes are eventually active.

The coupling of asynchronous starts and null messages is motivated by the examples given in the introduction: when a firing squad algorithm is constructed on the top of a parent algorithm $\mathcal{A}$, start signals correspond to some specific events of $\mathcal{A}$ (e.g., termination of $\mathcal{A}$), and a node is passive if this event has not yet occurred. In the example of synchronized termination, the null messages sent by a passive node are actually “not-yet-terminated” messages.

Upon the receipt of its start signal, a node u sets up its local variables with its initial state and starts executing its code. The value of any local variable x_u at the beginning of round t is denoted by $x_u(t)$. Thus $x_u(t)$ is undefined for $t < s_u$.

Communications that occur at round t, including the null messages, are modelled by a directed graph $\mathbb{G}(t) = (V, E_t)$ where the set of nodes is fixed while the set of edges may change from round to round. We assume a self-loop at each node in all the directed graphs $\mathbb{G}(t)$ since any node can communicate with itself instantaneously. The sequence of directed graphs $\mathbb{G} = (\mathbb{G}(t))_{t \in \mathbb{N}}$ is called a *dynamic graph* [4]. The dynamic graph $\mathbb{G}$ is said to be *bidirectional* if each directed graph $\mathbb{G}(t)$ is bidirectional, i.e., $(u, v) \in E_t$ if and only if $(v, u) \in E_t$.

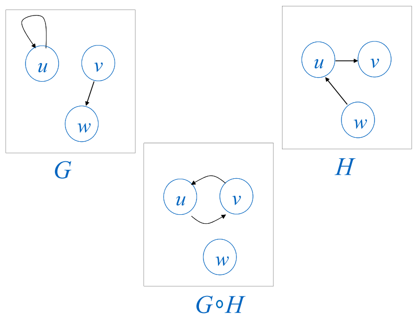

Fig. 1. A graph product $G \circ H$**.** If G or H contain nodes with no self loops, then G and H may contain edges which are not in $G \circ H$.

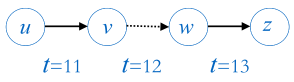

Fig. 2. A broken path. (u, v), (v, w) and (w, z) are edges of $\mathbb{G}(11)$, $\mathbb{G}(12)$ and $\mathbb{G}(13)$ respectively. If $s_u = s_v = s_z = 11$ and $s_w = 13$, then (u, v) is an edge of $\mathbb{G}^*(11)$, (w, z) is an edge of $\mathbb{G}^*(13)$, but (v, w) is not an edge of $\mathbb{G}^*(12)$, and thus the path in the figure is a $u \sim z$ broken path in the round interval [11,13].

The dynamic graph can be given by an online adversary, or endogenously as in *influence systems* [6]. Except in Section 6 which deals with randomized algorithms, our model assumes nothing on the way start signals are generated: a node may receive an *external* start signal coming from outside, or it may receive a start signal relayed by some active node. In particular, there may be more than one external start signal in the network. Start signals are *diffusive* if a message from an active node u that is received by a passive node v at round t is considered as conveying a start signal which will make node v active at round $t + 1$, i.e., $s_v = t + 1$. Diffusive start signals are thus correlated to the dynamic graph.

A run of a firing squad algorithm is entirely determined by the dynamic graph $\mathbb{G} = (\mathbb{G}(t))_{t\in\mathbb{N}}$ and by the list $\mathbb{S} = (s_u)_{u\in V}$ of rounds when nodes become active. We denote by $\mathbb{G}^*(t) = (V, E^*_t)$ the directed graph of edges in E_t connecting two nodes that are active in round t. The sets of u's incoming neighbors (in-neighbors for short) in the directed graphs $\mathbb{G}(t)$ and $\mathbb{G}^*(t)$ are denoted by $\mathrm{In}_u(t)$ and $\mathrm{In}^*_u(t)$, respectively.

Any set of dynamic graphs (possibly with different sets of nodes) is called a *network model*. We say that an algorithm *A solves the firing squad problem for the network model $\mathcal{D}$* if for each dynamic graph $\mathbb{G}$ in $\mathcal{D}$ and each scheduling of start signals $\mathbb{S}$, the run of A defined by $\mathbb{G}$ and $\mathbb{S}$ satisfies FS1 and FS2. The firing squad problem is *solvable for $\mathcal{D}$* if there is an algorithm that solves it for $\mathcal{D}$.

2.2. Paths and broken paths in a dynamic graph

Let us first recall that the *product* of two directed graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, denoted by $G_1 \circ G_2$, is the directed graph $G = (V, E)$, where $E = \{(u, v) : \exists w \in V, (u, w) \in E_1, (w, v) \in E_2\}$; see Fig. 1.

For any dynamic graph $\mathbb{G}$ and any integers $t' > t \geqslant 1$, let $\mathbb{G}(t : t') = \mathbb{G}(t) \circ \cdots \circ \mathbb{G}(t')$. By convention, $\mathbb{G}(t : t) = \mathbb{G}(t)$, and $\mathbb{G}(t : t')$ is the directed graph with only a self-loop at each node when $t' < t$. We also use the notation $\mathbb{G}(I)$ instead of $\mathbb{G}(t : t')$ when I is the integer interval $[t, t']$. Hence if I is the empty interval, then $\mathbb{G}(I)$ is the graph with only a self-loop at each node.

We now fix a run of a firing squad algorithm, with the dynamic graph $\mathbb{G}$ and the scheduling of start signals $\mathbb{S}$ which, as above, determine the dynamic graph $\mathbb{G}^*$. The sets of u's in-neighbors in $\mathbb{G}(t : t')$ and in $\mathbb{G}^*(t : t')$ are denoted by $\mathrm{In}_u(t : t')$ and $\mathrm{In}^*_u(t : t')$, respectively, or by $\mathrm{In}_u(I)$ and $\mathrm{In}^*_u(I)$ for short when $I = [t, t']$.

Let t and t' be two positive integers such that $t' \geqslant t$; a *$v{\sim}u$ path in the interval* $[t, t']$ is any sequence $P = (v_0 = v, v_1, \ldots, v_m = u)$ with $m = t' - t + 1$ and (v_k, v_{k+1}) is an edge of $\mathbb{G}(t + k)$ for each $k = 0, \ldots, m - 1$. The path P is said to be *broken* if one of its edges (v_k, v_{k+1}) is not in $\mathbb{G}^*(t + k)$; see Fig. 2. Hence there exists a $v{\sim}u$ path in the interval $[t, t']$ if and only if $v \in \mathrm{In}_u(t : t')$, and there exists a $v{\sim}u$ non-broken path in $[t, t']$ if and only if $v \in \mathrm{In}^*_u(t : t')$.

2.3. Delayed connectivity of a dynamic graph

Let us recall that a directed graph is *strongly connected* if for each pair of nodes u, v there is a directed path from u to v. For $c \geqslant 1$, *c-strong connectivity* is defined by (e.g., see [9]):

Definition 1. Let $G = (V, E)$ be a directed graph and let $c < |V|$ be a positive integer. We say that G is *c-strongly connected* if G remains strongly connected whenever less than c nodes are removed from G.

Note that a directed graph is strongly connected if and only if it is 1-strongly connected.

Definition 2. A dynamic graph $\mathbb{G}$ is *continuously c-strongly connected* if each directed graph $\mathbb{G}(t)$ is c-strongly connected.

Next we extend the above definition to bounded-length intervals of dynamic graphs.

Definition 3. Let c, T be two positive integers. The dynamic graph $\mathbb{G}$ is *c-connected with delay T* if for every positive integer t, the directed graph $\mathbb{G}(t : t + T - 1)$ is c-strongly connected. When $c = 1$, we use the abbreviation *connected with delay T*.

Finally, we present our weakest connectivity assumption for dynamic graphs.

Definition 4. A dynamic graph $\mathbb{G}$ is said to be *eventually connected* if for any positive integer t, there exists $t' \geqslant t$ such that $\mathbb{G}(t : t')$ is strongly connected.

Using the connectivity properties of dynamic graphs defined above, we characterize the connectivity properties that enable solution to the firing squad problem. For a positive integer T, $\mathcal{D}_T$ denotes the network model composed of all dynamic graphs that are connected with delay T. The union $\overline{\mathcal{D}} = \bigcup_{T=1}^{\infty} \mathcal{D}_T$ then consists of all dynamic graphs with *bounded delay connectivity*. Let $\mathcal{D}^{\diamond}$ denote the network model of eventually connected dynamic graphs. The relations among the above sets of dynamic graphs are thus given by the strict inclusions

$$\mathcal{D}_1 \subset \mathcal{D}_2 \subset \cdots \subset \mathcal{D}_T \subset \mathcal{D}_{T+1} \subset \cdots \subset \overline{\mathcal{D}} \subset \mathcal{D}^{\diamond}.$$

In the next sections, we show that the firing squad problem is not solvable for $\overline{\mathcal{D}}$ (and hence also for $\mathcal{D}^{\diamond}$), but for each positive integer T there is an algorithm that solves it for $\mathcal{D}_T$.

3. Bounded delay connectivity is not enough

In this section we show that the firing squad problem is not solvable for the network model $\overline{\mathcal{D}}$ with bounded delay connectivity. This result holds even when using algorithms that depend on the network size, in which case, we say that *the network size, n, is given*. Specifically, we show that for the network model $\overline{\mathcal{D}}$, the validity condition FS1 can be achieved if and only if n is given, and the firing squad problem (i.e., FS1 and FS2) cannot be solved even if n is given. These two impossibility results still hold for the original model of diffusive start signals, and when all communication graphs are bidirectional.

Proposition 5. *For the network model of dynamic graphs with bounded delay connectivity $\overline{\mathcal{D}}$, the validity condition FS1*

1. *can be achieved if the network size n is given;*
2. *cannot be achieved even if it is given that starting signal are diffusive, that the network is bidirectional, and that the network size is either n or $n + 1$ for some given integer n.*

Proof. (1) When the network size n is given, there is a trivial algorithm achieving FS1: At each round, each active node broadcasts the list of all the active nodes it has heard of, and updates its own list by the messages it receives from other nodes. A node fires when it has heard of exactly n nodes.

(2) For the sake of contradiction, let n be fixed, and suppose that there is an algorithm A which fulfills FS1 in each run over a bidirectional dynamic graph with n or $n+1$ nodes that is connected with some bounded delay, and in which starting signals are diffusive. Let W be a set of cardinality $n+1$, let u be a node in W, and let $V = W \setminus \{u\}$. Then by our assumption, A satisfies FS1 in the run $\mathcal{R}$ with the set of nodes V that are all active from the first round, the dynamic graph $\mathbb{G}$ in which each $\mathbb{G}(t)$ is K_V, the complete graph over V, and the start signal are diffusive. Hence, there is some t_0 such that every node in V has fired by round t_0 in $\mathcal{R}$.

Now consider a run $\mathcal{R}'$ over a dynamic graph $\mathbb{H}$ in which $\mathbb{H}(t) = K_V^u$ for $1 \leqslant t \leqslant t_0$, where K_V^u denotes the bidirectional graph over W with the same edges as K_V plus the self-loop (u, u), and $\mathbb{H}(t) = K_W$, the complete directed graph over W, for $t > t_0$. Clearly, $\mathbb{H}$ is connected with delay $t_0 + 1$; hence $\mathbb{H} \in \overline{\mathcal{D}}$. Assume further that in $\mathcal{R}'$, all nodes other than u are active from the first round, and $s_u = t_0 + 1$. Then since nodes in V cannot distinguish between $\mathcal{R}$ and $\mathcal{R}'$ during the first t_0 rounds, they all fire by round t_0 in $\mathcal{R}'$, which violates FS1. □

We now show that there is no algorithm that solves the firing squad problem for $\overline{\mathcal{D}}$, even if the number of nodes in the dynamic graph is given. This demonstrates that adding the simultaneity condition FS2 to the validity condition FS1 makes the problem strictly harder.

Theorem 6. *For each fixed integer $n > 2$, the firing squad problem is not solvable for the network model of bidirectional dynamic graphs with n nodes and with bounded delay connectivity, even if the start signals are assumed to be diffusive.*

Proof. For the sake of contradiction, suppose that for some $n > 2$, there is an algorithm A – that may depend on n – solving the firing squad problem in any bidirectional dynamic graph with n nodes, with bounded delay connectivity and with diffusive start signals.

Let V be a set of n nodes, and let u, v be two distinct nodes in V. For $x \in \{v, u\}$, let G_x be the graph consisting of a complete graph over $V \setminus \{x\}$ and the self loop (x, x). Let further $I = (V, E_I)$ denote the directed graph with only a self-loop at each node, i.e., $E_I = \{(w, w) : w \in V\}$.

We consider the run $\mathcal{R}$ of A in which all nodes are active in the first round, and with the dynamic graph consisting of an alternating sequence of graphs $\mathbb{G} = (G_u, G_v, G_u, G_v, \dots)$. Since $n > 2$, both $G_u \circ G_v \circ G_u$ and $G_v \circ G_u \circ G_v$ are the complete graph on V, and thus $\mathbb{G} \in \mathcal{D}_3 \subset \overline{\mathcal{D}}$. Hence all nodes fire at the same round t_F in the run $\mathcal{R}$.

Now assume that $\mathbb{G}(t_F) = G_u$ (the case $\mathbb{G}(t_F) = G_v$ is symmetric). For every integer $i \in [1, t_F]$, let $\mathbb{H}_i$ be a dynamic graph which is identical to $\mathbb{G}$, except that $\mathbb{H}_i(i) = \mathbb{H}_i(i+1) = \ldots = \mathbb{H}_i(t_F) = I$. In particular, $\mathbb{H}_{t_F} = (G_u, \dots, G_v, I, G_v, G_u, G_v, G_u, \dots)$ and $\mathbb{H}_1 = (I, \dots, I, G_v, G_u, G_v, G_u, \dots)$. Then each dynamic graph $\mathbb{H}_i$ belongs to $\mathcal{D}_{t_F+3} \subset \overline{\mathcal{D}}$.

From the viewpoint of the node u, the run $\mathcal{R}$ is indistinguishable up to round t_F from the run similar to $\mathcal{R}$ except that $\mathbb{G}$ is replaced by $\mathbb{H}_{t_F}$. Hence the node u must also fire at round t_F, and since $\mathbb{H}_{t_F} \in \overline{\mathcal{D}}$, all other nodes also fire at round t_F in this run.

Using a similar argument, we get that from the viewpoint of v, the run of A on $\mathbb{H}_{t_F}$ in which all nodes are active at the first round is indistinguishable up to round t_F from the similar run on $\mathbb{H}_{t_F-1}$. Hence in this latter run on $\mathbb{H}_{t_F-1}$, all nodes fire at round t_F as well. By repeating this argument for the dynamic graphs $\mathbb{H}_{t_F-2}, \mathbb{H}_{t_F-3}, \dots, \mathbb{H}_1$, we finally get that all nodes in V fire at round t_F in the run $\mathcal{R}'$ of A defined by the dynamic graph $\mathbb{H}_1$ and start signals all received in the first round.

From the viewpoint of the node v, the run $\mathcal{R}'$ is indistinguishable up to round t_F from the run $\mathcal{R}''$ of A identical to $\mathcal{R}'$ except that u is passive forever. Hence the node v fires at round t_F of the run $\mathcal{R}''$, violating FS1 – a contradiction. □

Using a similar proof technique, it can be shown that if the dynamic graph is not restricted to be bidirectional, then the problem is not solvable for the network model of eventually connected dynamic graphs that contains two nodes. The problem is solvable for bidirectional dynamic graphs with two nodes by having each node fire immediately if and after it has received a non-null message from the other node.

4. Firing with a bounded diameter

As a first step towards our main positive result – an algorithm that solves the firing squad problem in dynamic graphs that are c-connected with delay T – we present a solution in the case that a finite bound on the *diameter* of the dynamic graph is given. We start with some definitions.

Let $\mathbb{G} = (\mathbb{G}(t))_{t\in\mathbb{N}}$ be a dynamic graph. The *dynamic distance at the index* t from node v to node w, denoted $d_t(v, w)$, is defined as the minimum positive integer d such that there is a $v \sim w$ path in the interval $[t, t+d-1]$. If for all integers $t' \geqslant t$ there is no $v \sim w$ path in the interval $[t, t']$, then $d_t(v, w) = \infty$.

The *dynamic diameter*, or in short *diameter of the dynamic graph* $\mathbb{G}$ is the minimum positive integer D such that for every positive integer t, the directed graph $\mathbb{G}(t : t + D - 1)$ is complete, or infinity if there is no such integer:

$$\operatorname{diam}(\mathbb{G}) = \sup_{t\geqslant 1,\ (v,w)\in V^2} d_t(v, w).$$

Let $\mathcal{D}$ be any network model composed of dynamic graphs with diameters all bounded by D. Then we solve the firing squad problem for $\mathcal{D}$ by using local virtual clocks whose values may reach D only if all nodes are active. The key property of these virtual clocks is that if some node sets its clock to D, then all nodes set their clocks to D at the same round.

Basically, the value of u's local clock is the length of the shortest non-broken path ending at u. Our virtual clocks thus follow the evolution rule of the global virtual time (GVT) of a distributed discrete-event simulator [16].

The corresponding algorithm for $\mathcal{D}$, denoted A_D, does not use identifiers, and the computation and storage requirements of the nodes do not grow with the network size (but they grow with the parameter D). More precisely, its time complexity is in $O(D)$ and it uses only $O(\log D)$ bits per message.

Notation: In the pseudo-codes of all our algorithms, M_u denotes the set of non-null messages received by u in the current round. Thus M_u is the set of non-null messages sent to u by its active in-neighbors in the current topology. If non-null messages are vectors of some size, then ${M_u}^{(i)}$ denotes the set of the i-th entries of the messages in M_u.

We begin the correctness proof of the algorithm A_D by two useful lemmas about the way the virtual clocks r_u's evolve, whatever the connectivity properties of dynamic graphs are.

Lemma 7. *For every node u that is active from round s_u, if $t < t'$ and $s_u \leqslant t'$, then $r_u(t')$ is defined and*

1. *if there exists a broken path ending at u in the interval $[t, t'-1]$, then $r_u(t') \leqslant t' - t - 1$;*
2. *otherwise, for every $v \in \mathrm{In}_u(t : t'-1)$ it holds that $r_v(t)$ is defined and $r_u(t') \leqslant r_v(t) + t' - t$.*

Algorithm 1 Algorithm A_D, firing with diameter at most D.

```
Initialization:          % upon the receipt of the start signal
 1: $r_u \in \mathbb{N}$, initially 0
In each round do:
 2: send $\langle r_u \rangle$ to all and receive messages (from in-neighbors)
 3: if at least one received message is null then
 4:     $r_u \leftarrow 0$
 5: else
 6:     $r_u \leftarrow 1 + \min(M_u)$
 7: end if
 8: if $r_u \geqslant D$ then
 9:     Fire
10: end if
```

Proof. 1. Let $P = (v_t = v, v_{t+1}, \ldots, v_{t'} = u)$ be the assumed broken path, and let i be the maximal integer in $[t+1, t']$ such that the edge (v_{i-1}, v_i) of P is not in $\mathbb{G}^*(i-1)$. Then v_i is active at round i, and by line 4 of the algorithm A_D, $r_{v_i}(i) = 0$. By induction, for $k = i+1, \ldots, t'$, we show that $r_{v_k}(k) \leqslant k - i \leqslant k - t - 1$. Substituting $k = t'$, we obtain that $r_u(t') \leqslant t' - t - 1$.
2. If there is no such broken path, then for each $v \sim u$ path P as above, every node v_k is active at round k for each $k = t, \ldots, t'$. By line 6 of the algorithm A_D and by induction on k, $k = t+1, \ldots, t'$, it holds that $r_{v_k}(k) \leqslant r_v(t) + k - t$. Substituting $k = t'$, we obtain that $r_u(t') \leqslant r_v(t) + t' - t$. □

Lemma 8. *For every node u and at every round $t \geqslant s_{\max} = \max_{v \in V}(s_v)$ of an active run, we have $r_u(t) \geqslant t - s_{\max}$. Moreover, if $t \geqslant s_{\max} + 1$ and $\mathrm{In}_u(s_{\max} : t-1)$ contains a node v such that $s_v = s_{\max}$, then $r_u(t) = t - s_{\max}$.*

Proof. By the definition of $s_{\max}$, for each node u we have that $r_u(s_{\max})$ is defined and $r_u(s_{\max}) \geqslant 0$. An easy induction on $t \geqslant s_{\max}$ shows that $r_u(t) \geqslant t - s_{\max}$.

If $\mathrm{In}_u(s_{\max} : t-1)$ contains a node v such that $s_v = s_{\max}$, then there is a $v \sim u$ path in the interval $[s_{\max}, t-1]$. Since all nodes are active from round $s_{\max}$ onward, none of the $v \sim u$ paths in $[s_{\max}, t-1]$ are broken. The opposite inequality $r_u(t) \leqslant t - s_{\max}$ now follows from Lemma 7.2 and $r_v(s_{\max}) = 0$. □

Theorem 9. *The algorithm A_D solves the firing squad problem for any network model composed of dynamic graphs with diameters all bounded by D. Moreover, all nodes in an active run of the algorithm fire exactly D rounds after all nodes have become active and use messages of size $O(\log D)$.*

Proof. Let us first consider a run of the algorithm A_D in which there is a node v that is never active. Let u be any node in V and assume that $s_u < \infty$. We will show that u's virtual clock remains forever less than D. Consider any round $t \geqslant s_u$.

1. If $t \leqslant D$ then, since $r_u(t) \leqslant t - 1$, we have that $r_u(t) \leqslant D - 1$.
2. Otherwise $t \geqslant D + 1$ and there is a broken path $v \sim u$ in the interval $[t - D, t - 1]$. Using Lemma 7.1, we also get $r_u(t) \leqslant D - 1$ in this case.

This shows that no node ever fires in any run that is not active.

Let us now consider an active run of the algorithm. By the first claim in Lemma 8, the condition in line 8, namely $r_u \geqslant D$, eventually holds at each node u. Moreover, r_u may increase by at most 1 in every round, and thus there exists at least one round at which r_u is equal to D.

Let t_0 be the first round at which some node u sets its clock r_u to D. Then $r_u(t_0+1) = D$ and

$$\forall v \in V, \quad r_v(t_0+1) \leqslant D. \tag{1}$$

Note also that $t_0 \geqslant r_u(t_0)$, and hence $t_0 - D + 1 \geqslant 1$.

The assumption on the diameter implies that $\mathrm{In}_u(t_0 - D + 1 : t_0) = V$. Since $r_u(t_0+1) = D$, Lemma 7.1 implies that there is no broken path ending at node u in the interval $[t_0 - D + 1, t_0]$. Hence, $s_{\max} = \max_{v \in V}(s_v) \leqslant t_0 - D + 1$ and $\mathrm{In}_u^*(t_0 - D + 1 : t_0) = \mathrm{In}_u(t_0 - D + 1 : t_0) = V$.

In particular, $\mathrm{In}_u^*(t_0 - D + 1 : t_0)$ contains the latest activated nodes. Hence, by Lemma 8, $D = r_u(t_0+1) = t_0 - s_{\max} + 1$ and for every node $v \in V$, $r_v(t_0+1) \geqslant r_u(t_0+1) = D$. Using (1), we get that $r_v(t_0+1) = D$ for every node $v \in V$. Moreover, by the definition of t_0, we also have that $r_v(t') < D$ for every round $t' \leqslant t_0$. Thus all nodes simultaneously fire at the end of round t_0. □

Observe that the diameter of any connected dynamic graph with n nodes is at most $n - 1$. Thus one immediate spinoff of Theorem 9 is the following corollary, which when an upper bound N on the network size is given, provides a solution to the firing squad problem which uses messages of size $O(\log N)$.

Corollary 10. *If nodes have an upper bound N of the network size, the firing squad problem can be solved in any continuously strongly connected dynamic graph in N rounds after all nodes have become active using only $O(\log N)$ bits per message.*

5. Firing with T-delayed connectivity

We now present the algorithm $B_{c,T}$ and show that it solves the firing squad problem in linear time for dynamic graphs that are c-connected with delay T while no bound on the diameter or the number of nodes of the dynamic graph is given.

The algorithm $B_{c,T}$ uses the same virtual clocks r_u as the previous algorithm A_D. However since no bound on the diameter is given, the network size $|V|$ is computed by each node, and then $\lceil T|V|/c \rceil$ is used as a bound on the network diameter.

In order to compute $|V|$, each node u collects the identifiers of the active nodes that u has heard of in a variable HO_u. The node u terminates this computation and fires when its virtual clock r_u is large enough compared to the size of its HO_u set.

A similar idea was first used in [14], and also later in *early stopping consensus algorithms* [11,13], and in the counting algorithm of [17] but with synchronous starts. This technique requires distinct node identifiers and long messages since each node u broadcasts HO_u in each round.

Algorithm 2 Algorithm $B_{c,T}$, firing with T-delayed connectivity.

```
Initialization:          % upon the receipt of the start signal
 1: $r_u \in \mathbb{N}$, initially 0
 2: $HO_u \subseteq V$, initially $\{u\}$
In each round do:
 3: send $\langle r_u, HO_u \rangle$ to all and receive messages (from in-neighbors)
 4: if at least one received message is null then
 5:     $r_u \leftarrow 0$
 6: else
 7:     $r_u \leftarrow 1 + \min\left({M_u}^{(1)}\right)$
 8: end if
 9: $HO_u \leftarrow \cup_{HO \in {M_u}^{(2)}} HO$
10: if $|HO_u| \leqslant \left\lceil \frac{c}{T}(r_u + 2) \right\rceil - 2c$ then
11:     Fire
12: end if
```

The following lemma is needed for the analysis of the algorithm $B_{c,T}$.

Lemma 11. *If $G = (V, E)$ is c-strongly connected, then for any non-empty subset S of nodes, the following holds:*

$$|\Gamma_{\text{in}}(S) \setminus S| \geqslant \min(c, |\overline{S}|) \tag{2}$$

where $\Gamma_{\text{in}}(S)$ denotes the set of in-neighbors of S in G, and $\overline{S} = V \setminus S$.

Proof. We assume that there is a non-empty subset $S \subseteq V$ which violates (2), and show that G is not c-strongly connected. Let $R = \Gamma_{\text{in}}(S) \setminus S$. Condition (2) implies that $|R| < |\overline{S}|$, and thus the set $U = \overline{R \cup S}$ is non-empty. By definition, R intersects any path from a node in U to a node in S. Since $|R| < c$, the directed graph G is not c-strongly connected. □

The correctness proof of the algorithm $B_{c,T}$ then relies on the following inequality.

Lemma 12. *In each run of the algorithm $B_{c,T}$ on a dynamic graph $\mathbb{G}$ that is c-connected with delay T, for each node u and each round $t \geqslant s_u$, it holds that $r_u(t)$ and $HO_u(t)$ are defined and*

$$|HO_u(t)| \geqslant \min\left(1 - 2c + \frac{c}{T}(r_u(t) + 2), n\right). \tag{3}$$

Proof. Let us consider any node u and round $t \geqslant s_u$. The lemma clearly holds if $HO_u(t) = V$. So we now assume that $|HO_u(t)| < |V|$, and consider the two following cases:

1. If $t = 1$, then $s_u = 1$, $HO_u(t) = \{u\}$, and $r_u(t) = 0$. The lemma holds in this case.
2. Otherwise $t \geqslant 2$, and let $a, b \in \mathbb{N}$ satisfy $t = aT + b$ with $1 \leqslant b \leqslant T$.
   We split the interval $[1, t-1]$ into $a + 1$ subintervals $I_0, I_1, \ldots, I_a$ defined as follows:
   (a) $I_0 = [t - b + 1, t - 1]$; in particular, I_0 is the empty interval when $b = 1$.
   (b) for every index i, $0 < i \leqslant a$, $I_i = [t - b - iT + 1, t - b - (i-1)T]$.

Observe that $|I_0| = b - 1 < T$, and $|I_i| = T$ for $i > 0$. For every index $i \in [0, a]$, we let

$$J_i = I_i \cup I_{i-1} \cup \ldots \cup I_0.$$

Let further $S_{-1} = S^*_{-1} = \{u\}$, and for $i \in [0, a]$ let $S_i = \mathrm{In}_u(J_i)$ and $S^*_i = \mathrm{In}^*_u(J_i)$. Then we easily check that for every index i, $-1 \leqslant i \leqslant a - 1$, we have

$$S_{i+1} = S_i \cup \{v \in V : (v, u) \in E(\mathbb{G}(I_{i+1})) \text{ for some } u \in S_i\},$$

and

$$S^*_{i+1} = S^*_i \cup \{v \in V : (v, u) \in E(\mathbb{G}^*(I_{i+1})) \text{ for some } u \in S^*_i\}.$$

Clearly for every index i, $0 \leqslant i \leqslant a$, it holds that $S_{i-1} \subseteq S_i$. Moreover, since $|I_i| = T$ for $i \geqslant 1$ and $\mathbb{G}$ is c-connected with delay T, the graph $\mathbb{G}(I_i)$ is c-strongly connected. Using Lemma 11 with $G = \mathbb{G}(I_i)$ and $S = S_{i-1}$, we get that if $S_i \neq V$ for $i \geqslant 1$ then $|S_i| - |S_{i-1}| \geqslant c$. Since $|S_0| \geqslant |S_{-1}| = 1$, a simple induction on i shows that if $S_i \neq V$, then $|S_i| \geqslant ci + 1$. Let k be the maximal integer in $[-1, a]$ such that $S_k = S^*_k$ (such an integer exists since $S_{-1} = S^*_{-1}$). Then we get that $S_k = S^*_k \subseteq HO_u(t)$, and hence by our assumption $|S_k| < |V|$.
We now consider the two cases:
(a) If $k = a$ then since $r_u(t) \leqslant t - 1 = aT + b - 1$, we obtain

$$1 - 2c + \frac{c}{T}(r_u(t) + 2) \leqslant ac + 1 + \frac{c}{T}(b + 1 - 2T) \leqslant ac + 1 \leqslant |S^*_a| = |HO_u(t)|.$$

(b) Otherwise $k < a$ and the maximality of k implies that $S_{k+1} \neq S^*_{k+1}$. From the above descriptions of S_{k+1} and S^*_{k+1}, we derive that for some node $v \in S_k = S^*_k$, there is a $v\sim u$ path in the interval J_{k+1} that is broken in I_{k+1}. Since $b \leqslant T$, Lemma 7.1 implies that $r_u(t) \leqslant (k + 2)T - 2$ or equivalently that $k \geqslant \frac{r_u(t)+2}{T} - 2$. Thus we get

$$|HO_u(t)| \geqslant ck + 1 \geqslant c\left(\frac{r_u(t) + 2}{T} - 2\right) + 1 = 1 - 2c + \frac{c}{T}(r_u(t) + 2).$$

Thus the lemma is also proved in the case $t \geqslant 2$. □

Theorem 13. *The algorithm $B_{c,T}$ solves the firing squad problem for any network model composed of dynamic graphs that are c-connected with delay T. Moreover, in any active run all nodes fire in less than $\left\lceil \frac{T}{c}(n-1) \right\rceil + T$ rounds after all nodes have become active and messages are of size $O(n \log n)$.*

Proof. Let us first consider a run of the algorithm in which there is a node v that is never active. Then no node ever receives a non-null message from v, and so for any node u that is active at round t, we have $|HO_u(t)| \leqslant n - 1$. This implies by Lemma 12 that $|HO_u(t)| > \left\lceil \frac{c}{T}(r_u(t) + 2) \right\rceil - 2c$, and hence u does not fire at round t. We conclude that no node ever fires in this run.

Let us now consider an active run of the algorithm. First, observe that by the first claim in Lemma 8 and the fact that the cardinality of each set HO_u is at most n, the condition in line 11 eventually holds at each node u.

Moreover, because of the initialization and update rules for the HO variables (lines 2 and 9), a node v different from u is in $HO_u(t + 1)$ if and only if there exists a non-broken $v\sim u$ path in some non-empty interval $[s, t]$. Since $u \in \mathrm{In}^*_u(s_u)$, this shows that

$$HO_u(t + 1) = \bigcup_{s \geqslant s_u} \mathrm{In}^*_u(s : t). \tag{4}$$

Let t_0 be the first round at which the condition in line 10 holds at some node, and let u denote one such node, i.e.,

$$|HO_u(t_0 + 1)| \leqslant \left\lceil \frac{c}{T}(r_u(t_0 + 1) + 2) \right\rceil - 2c. \tag{5}$$

Inequality (3) of Lemma 12 implies that $HO_u(t_0 + 1) = V$. In particular, $HO_u(t_0 + 1)$ contains the latest activated nodes. Let v denote one such node, i.e., $s_v = s_{\max}$. By (4), there is a non-broken path $v\sim u$ in some interval $[s, t_0]$ with $s \geqslant s_u$. It follows that $s \geqslant s_v$. Thereby $t_0 \geqslant s_{\max}$ and $v \in \mathrm{In}^*_u(s_{\max} : t_0)$. This implies, by Lemma 8, that

$$r_u(t_0 + 1) = r_v(t_0 + 1) = t_0 + 1 - s_{\max} = \min_{w \in V} r_w(t_0 + 1).$$

Using Lemma 12 again, we get that for all $w \in V$, $HO_w(t_0 + 1) = V$. Therefore the inequality (5) holds for all nodes in round $t_0 + 1$, and by the definition of t_0 this is the first round in which this inequality holds for all nodes. Hence all nodes fire simultaneously at the end of round t_0. The bound on messages size is implied by the fact that an HO set may contain at most n identities whose lengths are logarithmic in n. □

The only operations in the algorithm $B_{c,T}$ that involve node identifiers are performing the union and extracting the cardinalities of the sets HO_u. Since the decisions to fire depend only on the cardinalities of the sets HO_u and not on the actual values of the identifiers in these sets, the sequences of operations performed by each node in a specific run are independent of these values.

A close examination of the proof of Theorem 13, shows that each node actually computes the set V, and so its cardinality. As a byproduct, the algorithm $B_{c,T}$ thus solves the problem of counting the network size despite asynchronous starts in any model of dynamic graphs that are c-connected with delay T, and in particular in the model of continuously strongly connected dynamic graphs. This should be compared with the impossibility result by Wattenhofer [28] which states that if passive nodes do not emit any message, then counting is impossible with asynchronous starts.

A comparison of Algorithms A_D and $B_{c,T}$ demonstrates that the availability of information about the diameter (or size) of the dynamic graph dramatically improves our solutions to the firing squad problem: For instance, let us consider any dynamic graph $\mathbb{G}$ with n nodes and diameter D. *A priori*, the sole guarantee about $\mathbb{G}$'s connectivity we get is then that $\mathbb{G}$ is $(n-1)$-connected with delay D. Then Algorithm A_D solves the problem on $\mathbb{G}$ within D rounds using messages of size $\log D$ while Algorithm $B_{n-1,D}$ needs distinct identifiers and solves the problem in approximately $2D$ rounds and messages of size $\Omega(n \log n)$.

6. Bound on the network size and randomization

In this section we show that if a polynomial bound N on the network size n is given, then randomization may reduce the message sizes in our firing squad algorithm $B_{c,T}$ without degrading its linear time complexity. Similarly to $B_{c,T}$, our randomized algorithm for the firing squad problem actually estimates the size of the network, and thus as a byproduct, provides a solution to the approximate counting problem for the case of asynchronous starts. In this sense, it generalizes the randomized approximate counting algorithm of [17], which assumes that all nodes start simultaneously.

When considering randomized solutions to the firing squad problem, we restrict our discussion to runs in which dynamic graphs and start signals are generated by an *oblivious* adversary, whose decisions are independent of the random choices made by the algorithm. An oblivious adversary is *restricted to a network model* $\mathcal{D}$ if the dynamic graphs it produces belong to $\mathcal{D}$.

A randomized algorithm R is said to *solve the firing squad problem for a network model* $\mathcal{D}$ *with probability at least* $1-\eta$, if the following holds: for each oblivious adversary that is restricted to $\mathcal{D}$, a run of R in which the dynamic graph and start signals are generated by this adversary satisfies FS1 and FS2 with probability at least $1-\eta$. In other words, we consider Monte Carlo algorithms, which may produce erroneous results with arbitrary small probability.

For the sake of simplicity, we present our randomized firing squad algorithm in the case $c = T = 1$, i.e., for dynamic graphs that are continuously strongly connected, but the generalization to the case of c-connectivity with delay T is straightforward. First observe that by Corollary 10, if we use N as an upper bound on the diameter of the network, then the A_N algorithm in Section 4 solves the firing squad problem within $O(N)$ rounds using messages of size $O(\log N)$. When N is significantly larger than the network size n, the time complexity of this solution is thus not satisfactory.

The algorithm, denoted $R_{N,\eta}$, depends on two parameters N and η, where N is a positive integer and η is any real number in $[0, 1/2)$.

The algorithm works as follows: upon becoming active, each node u generates ℓ independent random numbers $Y_u^{(1)}, \dots, Y_u^{(\ell)}$, where ℓ depends on N and η, and the distribution of each $Y_u^{(i)}$ is exponential with rate 1. At each round, any active node u first broadcasts the smallest value of the $Y_v^{(i)}$'s it has heard of for each index $i \in [1 \dots \ell]$, and then computes from the minimum values it received so far an estimation n_u of the number of nodes it heard of. Node u fires when the value of its clock r_u is sufficiently large compared to n_u.

The analysis of the algorithm $R_{N,\eta}$ relies on the following lemma in [22] which is an application of the Cramér-Chernoff method (see for instance [2], sections 2.2 and 2.4).

Lemma 14. *Let* $I = \left\{ \left(Y_1^{(1)}, \dots, Y_1^{(\ell)}\right), \dots, \left(Y_{|I|}^{(1)}, \dots, Y_{|I|}^{(\ell)}\right) \right\}$ *be a finite set of ℓ-tuples of independent exponential variables all with rate 1, and let* $W := \frac{\ell}{\sum_{i=1}^{\ell} \min_{1 \leqslant j \leqslant |I|} Y_j^{(i)}}$. *Then we have*

$$\Pr[\, |W - |I|| > 2|I|/9 \,] \leqslant 2 \exp(-\ell/243)\,. \tag{6}$$

Lemma 14 is used to show that for large enough ℓ, the value of n_u at the end of round t provides with high probability a good approximation of the number of active nodes that u has heard of so far. This implies, via Lemma 12, that if $n_u < 2r_u/3$ then with high probability node u has heard of all other nodes (yielding the condition $n_u < 2r_u/3$ for node u to fire in line 14). As for the algorithm $B_{1,1}$, we will conclude that with high probability, no node ever fires in a non-active run, and all the nodes fire at the same round of any active run. More precisely, we choose $\ell = \lceil 243 \cdot (\ln 4N^2 - \ln \eta) \rceil$ to guarantee a final probability of at least $1-\eta$ for these successful active and non-active runs.

The size of the messages used by the algorithm can be limited, at the price of higher storage capacity at the nodes as explained below, by using rounded and range-restricted calculations as in [24]. Specifically, we round down each $Y_u^{(i)}$ to the next smaller integer power of 12/13, denoted $\overline{Y}^{(i)}$. Then the resulting approximate value $\overline{n_v}$ of n_v satisfies $n_v \leqslant \overline{n_v} \leqslant \frac{13}{12} n_v$, and Equation (6) provides sufficiently good bounds on the difference $|\overline{n_v} - \nu_v|$, where ν_v is the number of nodes v had heard of.

By definition of the exponential distribution, it is not hard to see that the random variables $Y_u^{(i)}$ are all within the range $[\eta/(4\ell N), \ln(4\ell N/\eta)]$ with high probability, namely

$$\Pr\left[\forall u \in V,\ \forall i,\ Y_u^{(i)} \in [\eta/(4\ell N), \ln(4\ell N/\eta)]\right] > 1 - \eta/2, \tag{7}$$

which allows us to ignore runs in which the randomized variables $Y_u^{(i)}$ are not in the above range. The number of distinct variables $\overline{Y}_u^{(i)}$ in that range is $O(\log(N\eta^{-1}))$, hence each such variable can be represented using $O(\log\log(N/\eta))$ bits. This leads to message length of $O(\log(N/\eta) \cdot \log\log(N/\eta))$ bits.

We note, however, that the implied calculations require exponentially higher storage capacity: computing $\overline{n_u}$ (line 13 of algorithm $R_{N,\eta}$) must be done with the ℓ *exact* values of the variables $\overline{Y}_u^{(i)}$, and exact representation of numbers occurring in the implied calculations could require $\Omega(\ell N \eta^{-1})$ bits.

The correctness proof of the algorithm with the approximate random variables $\overline{Y}_u^{(i)}$ is valid for all runs in which the exact random variables are in the range $[\eta/(4\ell N), \ln(4\ell N/\eta)]$ of (7), and this range restriction is violated with probability of at most $\eta/2$.

Algorithm 3 The randomized algorithm $R_{N,\eta}$, firing with continuous strong connectivity.

```
Initialization:          % upon the receipt of the start signal
 1: $r_u \in \mathbb{N}$, initially 0
 2: $\overline{Y}_u = \left(\overline{Y}_u^{(1)}, \ldots, \overline{Y}_u^{(\ell)}\right) \in \mathbb{R}^\ell$ (where $\ell = \lceil 243 \cdot (\ln 4N^2 - \ln \eta) \rceil$), initially rounded and range-restricted approximations of independent
    random numbers with exponential distribution of rate 1.
 3: $n_u \in \mathbb{N}$, initially 0
In each round do:
 4: send $\langle r_u, \overline{Y}_u \rangle$ to all and receive messages (from in-neighbors)
 5: if at least one received message is null then
 6:     $r_u \leftarrow 0$
 7: else
 8:     $r_u \leftarrow 1 + \min\left({M_u}^{(1)}\right)$
 9: end if
10: for $i = 1, \ldots, \ell$ do
11:     $\overline{Y}_u^{(i)} \leftarrow \min\left({M_u}^{(i+1)}\right)$
12: end for
13: $\overline{n_u} \leftarrow \ell / \sum_{i=1}^{\ell} \overline{Y}_u^{(i)}$
14: if $r_u > 1.5\ \overline{n_u}$ then
15:     fire
16: end if
```

Theorem 15. *The algorithm $R_{N,\eta}$ solves the firing squad problem with probability at least $1-\eta$ for any network model composed of dynamic graphs that are continuously strongly connected. Moreover, in any active run, with probability at least $1-\eta$ all nodes fire simultaneously in less than $2n$ rounds after the last nodes have become active.*

Proof. Let us fix any real number $\eta \in (0, 1/2]$, and let ℓ be as defined in line 2 of the algorithm. By the paragraph preceding Theorem 15, it suffices to prove that if the values of the exact random variables $Y_u^{(i)}$ are within the range $[\eta/(4\ell N), \ln(4\ell N/\eta)]$, then the statement of the theorem holds with probability $1 - \eta/2$. Consider now a run of $R_{N,\eta}$ which is managed by an oblivious adversary that is restricted to continuously strongly connected dynamic graphs.

For any node u that becomes active at round s_u and at each round $t \geqslant s_u + 1$, let $\nu_u(t)$ denote the number of nodes that u has heard of at the beginning of round t.[1] For a continuously strongly connected graph, applying Lemma 12 with $c = T = 1$ implies:

$$\nu_u(t) \geqslant \min\left(r_u(t) + 1, n\right). \tag{8}$$

Let us first consider a run of the algorithm in which some node v remains passive forever. Clearly, no node can hear of v, and hence for any node u that is active at round t, we have $\nu_u(t) \leqslant n - 1$. With (8), it follows that $\nu_u(t) \geqslant r_u(t) + 1$.

[1] In other words, $\nu_u(t)$ is equal to the cardinality of the set $HO_u(t)$ (not computed here) in the algorithm $B_{c,T}$.

Let n_u be the value defined by replacing the variables $\overline{Y}_u^{(i)}$ in line 13 of the algorithm by their exact values $Y_u^{(i)}$. Then since $Y_u^{(i)} \geqslant \overline{Y}_u^{(i)}$ for all i, we get that $n_u \leqslant \overline{n_u}$. Hence at every round t, $t \geqslant s_u + 1$, we have

$$\Pr[\overline{n_u}(t) \geqslant 2r_u(t)/3] \;>\; \Pr[n_u(t) \geqslant 7\nu_u(t)/9] \;\geqslant\; 1 - 2\exp(-\ell/243), \tag{9}$$

where the rightmost inequality is by Lemma 14 and the fact that the adversary is oblivious. Thus with probability at least $1 - 2\exp(-\ell/243)$ the condition in line 14 of the algorithm is not fulfilled for node u at time t. Node u makes a true update to n_u (line 13) at most $n-2$ times (when the set of nodes it heard of strictly increases), and if in all these updates the condition in line 14 does not hold, then u never fires. Hence by the union bound, the probability that node u never fires is at least $1 - 2(n-2)\exp(-\ell/243)$. Using the union bound again for all n nodes in the network, and the inequality $N \geqslant n$, we obtain that the probability that no node ever fires is at least

$$1 - 2N(N-2)\exp(-\ell/243) > 1 - \eta/2. \tag{10}$$

Let us now consider an active run of the algorithm. First we observe by the discussion above that Equation (10) actually bounds the probability that no node v fires before its virtual clock r_v is set to n (the true network size). So in the following discussion we assume active runs in which no node fires before round n.

Next we show that in any active run, each node eventually fires. Assume for contradiction that node v never fires in some run. By applying Lemma 8, we get that the virtual clock r_v grows indefinitely. On the other hand the value of the variable n_v is updated at most $n-1$ times in a run, and hence is bounded. This implies that the condition in line 14 eventually holds, and hence v eventually fires – a contradiction.

We now prove that all nodes fire at the same round. Let t_0 be the first round at which some virtual clock, say r_u, is set to n. Using (8), we deduce that $\nu_u(t_0) = n$, i.e., by the end of round t_0 node u has heard of all nodes in the network, which are all active. Then the same argument as in the proof of Theorem 13 shows that at the end of round t_0 all the virtual clocks have synchronized and each node has heard of all other nodes, namely

$$\forall v \in V,\ \forall t \geqslant t_0 + 1,\ \ r_v(t) = t - s_{\max} \quad \text{and} \quad \nu_v(t) = n. \tag{11}$$

Once the above holds, all nodes have the same list of ℓ minimal values $(\overline{Y}_u^{(1)}, \ldots, \overline{Y}_u^{(\ell)})$. Hence each node v sets the value of n_v at line 13 to the *same* estimate $\tilde{n}$ of n, which is not changed thereafter:

$$\forall v \in V,\ \forall t \geqslant t_0 + 1,\ \ n_v(t) = \tilde{n}. \tag{12}$$

It follows that if the condition at line 14 holds at some node in round $t \geqslant t_0$, then it holds at all nodes in that round. This implies, under our assumption that no node fires before round n, that all nodes fire at the same round, say round θ, as claimed. It remains to prove that with high probability, $\theta \leqslant 2n + s_{\max}$.

Recall that $\overline{n_u}(t) \leqslant \frac{13}{12} n_u(t)$. Hence the probability that the firing condition $\overline{n_u} < 2r_u(t)/3$ (line 14) holds at node u when $r_u(t) = 2n$ and $\nu_u(t) = n$ is bounded by

$$\Pr[\overline{n_u}(t) < 4n/3] \;\geqslant\; \Pr[n_u(t) < 16n/13] \;\geqslant\; \Pr[n_u(t) < 11n/9] \;\geqslant\; 1 - 2\exp(-\ell/243),$$

where the rightmost inequality is again by Lemma 14. Clearly, with at least this probability, the firing condition is satisfied (and all nodes fire simultaneously) no later than round $s_{\max} + 2n$.

To sum up, the probability that in an active run all nodes fire simultaneously at most $2n$ rounds after $s_{\max}$ is at least

$$1 - (2N(N-2) + 2)\exp(-\ell/243) > 1 - \eta/2$$

which completes the proof. □

7. Conclusion and further research

In this paper we studied the firing squad problem in a network of an unknown number of nodes with full computational power, thus extending the original model which assumes that nodes are finite state machines. We focused on a natural extension of the problem in which start signals are left arbitrary; in particular, they are not assumed to be propagated by the nodes in the network as in the case of diffusive start signals.

We modelled the inter-node communication by a dynamic graph, and presented a tight relation between the solvability of the firing squad problem and the connectivity of the dynamic graph. Specifically, we introduced the notion of delayed connectivity, and showed that the firing squad problem is solvable if and only if the dynamic graph is connected with delay T, for some *given* constant T. Our solution uses messages of super-linear size, and we showed that additional information on the diameter or on the size of the network can substantially reduce the message size.

Combining our positive and negative results, we get that when nodes are infinite state machines, the firing squad problem is solvable for arbitrary timing of start signals if and only if it is solvable when restricted to diffusive start signals. An interesting question is whether this equivalence in terms of solvability is still valid in the original model of the firing squad problem, where nodes are finite state machines. It can be shown that this is the case when the topology is a line or a circuit, but it is not clear whether this holds for other topologies.

Possible extensions of this work involve other variations of the model of computation. For instance, it is interesting to determine under what conditions the firing squad problem is solvable in an anonymous network where nodes have limited storage capabilities and communicate through finite bandwith channels as in [15]. Our randomized algorithm provides an efficient Monte Carlo solution for this problem, in the case of a continuously strongly connected network and a polynomial upper bound on the size of the network. Another open question concerns the role of leaders in a dynamic network: could the existence of a leader be useful for achieving or improving solutions to the firing squad problem?

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank two anonymous referees for their many helpful comments.

References

[1] Sebastian Abshoff, Markus Benter, Andreas Cord-Landwehr, Manuel Malatyali, Friedhelm Meyer auf der Heide, Token dissemination in geometric dynamic networks, in: Proceedings of the 9th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics, ALGOSENSORS, 2013, pp. 22–34.
[2] Stéphane Boucheron, Gábor Lugosi, Pascal Massart, Concentration Inequalities. A Nonasymptotic Theory of Independence, Oxford University Press, Oxford, 2013.
[3] James E. Burns, Nancy Lynch, The byzantine firing squad problem, Adv. Comput. Res. 4 (1987) 147–161.
[4] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, Nicola Santoro, Time-varying graphs and dynamic networks, in: Hannes Frey, Xu Li, Stefan Rührup (Eds.), ADHOC-NOW, in: Lecture Notes in Computer Science, vol. 6811, Springer, 2011, pp. 346–359.
[5] Bernadette Charron-Bost, André Schiper, The heard-of model: computing in distributed systems with benign faults, Distrib. Comput. 22 (1) (2009) 49–71.
[6] Bernard Chazelle, Natural algorithms and influence systems, Commun. ACM 55 (12) (2012) 101–110.
[7] Brian A. Coan, Danny Dolev, Cynthia Dwork, Larry Stockmeyer, The distributed firing squad problem, in: ACM Symposium on Theory of Computing Conference, STOC'85, 1985, pp. 335–345.
[8] Thiago Correa, Breno Gustavo, Lucas Lemos, Amber Settle, An overview of recent solutions to and lower bounds for the firing synchronization problem, preprint, arXiv:1701.01045, 2017.
[9] Reinhard Diestel, Graph Theory, Springer-Verlag, Berlin Heidelberg, 2017.
[10] Danny Dolev, Ezra N. Hoch, Yoram Moses, An optimal self-stabilizing firing squad, SIAM J. Comput. 41 (2) (2012) 415–435.
[11] Danny Dolev, Rüdiger Reischuk, H. Raymond Strong, Early stopping in Byzantine agreement, J. ACM 37 (4) (October 1990) 720–741.
[12] Danny Dolev, H. Raymond Strong, Authenticated algorithms for Byzantine agreement, SIAM J. Comput. 12 (4) (November 1983) 656–666.
[13] Cynthia Dwork, Yoram Moses, Knowledge and common knowledge in a Byzantine environment: crash failures, Inf. Comput. 88 (2) (October 1990) 156–186.
[14] Steven Finn, Resynch procedures and a fail-safe network protocol, IEEE Trans. Commun. 27 (6) (1979) 840–845.
[15] Julien M. Hendrickx, Alex Olshevsky, John N. Tsitsiklis, Distributed anonymous discrete function computation, IEEE Trans. Autom. Control 56 (10) (2011) 2276–2289.
[16] R. Jefferson David, Virtual time, ACM Trans. Program. Lang. Syst. 7 (3) (July 1985) 404–425.
[17] Fabian Kuhn, Nancy Lynch, Rotem Oshman, Distributed computation in dynamic networks, in: Proceedings of the Forty-second ACM Symposium on Theory of Computing, STOC'10, ACM, New York, NY, USA, 2010, pp. 513–522.
[18] Fabian Kuhn, Yoram Moses, Rotem Oshman, Coordinated consensus in dynamic networks, in: Proceedings of the 30th ACM Symposium on Principles of Distributed Computing (PODC), ACM, 2011, pp. 1–10.
[19] Christoph Lenzen, Joel Rybicki, Near-optimal self-stabilising counting and firing squads, in: Proceedings of the 19th International Symposium on Self-Stabilizing Systems (SSS), in: LNCS, vol. 10083, Springer, Heidelberg, 2016, pp. 263–280.
[20] Edward F. Moore, The firing squad synchronization problem, in: Sequential Machines. Selected papers, 1964, pp. 213–214.
[21] F.R. Moore, G.G. Langdon, A generalized firing squad problem, Inf. Control 12 (3) (1968) 212–220.
[22] Damon Mosk-Aoyama, Devavrat Shah, Computing separable functions via gossip, in: Proceedings of the 25th ACM Symposium on Principles of Distributed Computing (PODC), ACM, 2006, pp. 113–122.
[23] Yasuaki Nishitani, Namio Honda, The firing squad synchronization problem for graphs, Theor. Comput. Sci. 14 (1) (1981) 39–61.
[24] Rotem Oshman, Distributed Computation in Wireless and Dynamic Networks, PhD thesis, Massachusetts Institute of Technology, 2012.
[25] Marshall Pease, Robert Shostak, Leslie Lamport, Reaching agreement in the presence of faults, J. ACM 27 (2) (April 1980) 228–234.
[26] Nicola Santoro, Time to change: on distributed computing in dynamic networks (keynote), in: 19th International Conference on Principles of Distributed Systems, OPODIS 2015, December 14-17, 2015, Rennes, France, 2015, pp. 3:1–3:14.
[27] T.K. Srikanth, Sam Toueg, Simulating authenticated broadcasts to derive simple fault-tolerant algorithms, Distrib. Comput. 2 (2) (1987) 80–94.
[28] Roger Wattenhofer, The Science of the Blockchain, CreateSpace Independent Publishing Platform, 2016.