The Firing Squad Problem Revisited

Bernadette Charron-Bost
École polytechnique, CNRS, 91128 Palaiseau, France
charron@lix.polytechnique.fr

Shlomo Moran
Department of Computer Science, Technion, Haifa, Israel 32000
moran@cs.technion.ac.il

Abstract

In the classical firing squad problem, an unknown number of nodes represented by identical finite state 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.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms, Theory of computation → Dynamic graph algorithms, Theory of computation → Distributed algorithms

Keywords and phrases Synchronization, Detection, Simultaneity, Dynamic Networks

Digital Object Identifier 10.4230/LIPIcs.STACS.2018.20

1 Introduction

Many distributed algorithms assume a *synchronous networked system*, in which computation is divided into *synchronized rounds* that are communication closed layers: any message sent at some round can be received only at that round. In this model it is typically assumed that each execution of an algorithm is started by all nodes simultaneously, i.e., at the same round. For instance, most of synchronous consensus algorithms (eg., [21, 12, 23]), as well as many distributed algorithms for dynamic networks (eg., [16, 17]) 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 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 which may change in each round. All nodes are initially *passive*, and a node becomes *active* upon receiving a *start signal* at an unpredictable time. We stress that receiving a message from an active node is not necessarily considered as a start signal. The goal is then to guarantee that the nodes synchronize by *firing* - i.e., entering a designated state for the first time - simultaneously if and only if *all* nodes are eventually active. Formally, the following must be satisfied:

FS1 (Validity): A node fires if and only if all nodes have received start signals.

FS2 (Simultaneity): All the nodes that fire, fire at the same round.

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), in real-time processing (where nodes have to carry out some external actions simultaneously), or for distributed termination (to guarantee that nodes complete their computation at the same round). 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 execution should be started simultaneously, after all nodes terminated the i-th execution (see e.g., [5]).

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. At the opposite scenario, the problem is clearly unsolvable if some node is permanently isolated. This demonstrates a strong correlation 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.

The firing squad problem was originally studied in the context of automata theory (eg., [18, 19]). 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 eg., [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. It should be noted that the above model assumes *diffusive start signals*, for which the timing of start signals is not arbitrary: upon the receipt of a message from an active node, a passive node becomes active, i.e., receiving such a message is considered as a start signal. 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: considering that nodes are no longer 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 to solve the firing squad problem?

It should be noted that the firing squad problem has also been studied in the context of fault tolerant distributed computations (eg., [3, 12, 7]), and more recently in the context of self-stabilization (eg., see [10]). This model also assumes that each node has a full computational power, but otherwise the setting of the problem is different: Nodes are connected by a complete graph, and thus the number of nodes n is given. At most f nodes may be faulty, for various types of Byzantine faults. This implies a permanent complete connectivity between the non-faulty nodes, and arbitrary connectivity of all other links. Besides, due to the unpredictable behavior of faulty nodes, the simultaneity condition and to a larger extent the validity condition in this model ought to be drastically weakened: eg., it is only required that all non-faulty nodes eventually fire simultaneously. Finally, the study of the problem is strictly limited to diffusive start signals.

Contribution. In this paper we consider a set of an unknown number n of nodes possessing full computational power. Nodes have distinct identities which are not mutually known, but otherwise they run identical codes (in some precise sense that is discussed later). The inter-node communication is modeled by a *dynamic graph*, i.e., at each round, nodes communicate along directed edges of an arbitrary communication graph which 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 a stable clique of size $n - f$, or several models of dynamic networks in distributed computing (eg., see [16, 1, 22]) 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. On the positive side, we show that if the dynamic graph is guaranteed to be connected within each period of T consecutive rounds, where the constant T is given, then the problem is solvable in time which is at most linear in the (unknown) network size. 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 the firing squad problem is solved for a *static* directed graph if and only if it 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 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 D on the diameter of the dynamic graph is given, then the problem is solvable in time linear in D. This solution is applicable to anonymous networks, where nodes have no identities, and it uses much shorter messages. We conclude by showing that when a polynomial bound on the network size is given, the use of randomization can substantially reduce messages size while preserving a linear time complexity.

For space consideration, some proofs are omitted in this version.

2 The Model

2.1 Distributed computations in the dynamic graphs model

We consider a networked system with a *fixed* set of n nodes. Nodes have unique identifiers, and the set of identifiers is denoted by V. The identities of the nodes are not mutually known, and the network size n is unknown as well. Nodes may also ignore their own identities, in which case the network is said to be *anonymous*. Furthermore, nodes run identical programs, i.e., programs do not depend on node identities (see the discussion in Section 5).

Computation proceeds in *synchronized rounds*, which are communication closed in the sense that 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 attempts to send messages to all nodes, receives messages from some nodes, and finally goes to its next state and proceeds to round $t + 1$. The round number t is used for a reference, but is unknown to the nodes.

In every *run* of an algorithm, each node u is initially *passive*: it is part of the network, but sends only heartbeats – that we call *null* messages – and does not change its state. Then it either becomes *active* by receiving a unique *start signal* at the beginning of some round $s_u \geqslant 1$, or remains passive forever – in which case we let $s_u = +\infty$. A run is *active* if all nodes are eventually active.

Upon the receipt of its start signal, node u sets up its local variables (with its initial state) and starts executing its program. For any of its local variables x_u, the value of 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 are modeled by a directed graph $\mathbb{G}(t) = (V, E_t)$ that may change from round to round. We assume a self-loop at each node in all the 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]. It can be decided ahead of time, by an online adversary, or endogenously as in *influence systems* [6]. Similarly, the way start signals are generated is left totally arbitrary: 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, and start signals may be not 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 at which nodes become active. We denote by $\mathbb{G}^*(t) = (V, E_t^*)$ the directed graph of edges in E_t connecting two nodes which 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.

Let $\mathcal{D}$ be a set of dynamic graphs. We say that an algorithm A *solves the firing squad problem for* $\mathcal{D}$ if for each $\mathbb{G} \in \mathcal{D}$ 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 $G_1 \circ G_2$, is the directed graph with the set of nodes V and with an edge (u, v) if there exists $w \in V$ such that $(u, w) \in E_1$ and $(w, v) \in E_2$.

For any dynamic graph $\mathbb{G}$ and any integers $t' > t \geqslant 1$, we 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 $[t, t']$.

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 $k = 0, \ldots, m - 1$. Hence there exists a $v\sim u$ path in the interval $[t, t']$ if and only if $v \in \mathrm{In}_u(t : t')$. The path P is said to be *broken* if one of its edges (v_k, v_{k+1}) is not in $\mathbb{G}^*(t + k)$.

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 then defined by (see, e.g., [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 then characterize the connectivity properties that enable solutions to the firing squad problem. For a positive integer T, $\mathcal{D}_T$ denotes the set of dynamic graphs which are connected with delay T. The union $\mathcal{D}_B = \bigcup_{T=1}^{\infty} \mathcal{D}_T$ is the set of dynamic graphs with *bounded delay connectivity* and $\mathcal{D}_E$ denotes the set of eventually connected dynamic graphs. The relations among the above sets of dynamic graphs are thus given by the strict inclusions

$$\mathcal{D}_1 \subset \mathcal{D}_2 \subset \cdots \subset \mathcal{D}_T \subset \mathcal{D}_{T+1} \subset \cdots \subset \mathcal{D}_B \subset \mathcal{D}_E.$$

In the next sections, we show that the firing squad problem is not solvable for $\mathcal{D}_B$ (and hence also for $\mathcal{D}_E$), but for each positive integer T, it is solvable 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 set $\mathcal{D}_B$ of the dynamic graphs with bounded delay connectivity, even if the network size, n, is given. Specifically, we show that for this set of dynamic graphs, the validity condition FS1 can be achieved if and only if n is given, and the firing squad problem (i.e., FS1 plus FS2) cannot be solved even if n is given.

Interestingly, 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 set of dynamic graphs with bounded delay connectivity $\mathcal{D}_B$, the validity condition FS1 can be achieved if the network size n is given, but cannot be achieved if it is given that the network size is either n or $n + 1$.*

Next we show that there is no algorithm that solves the firing squad problem for $\mathcal{D}_B$, 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 and that the knowledge of the size of the network does not help in the sole context of bounded delay connectivity.

▶ **Theorem 6.** *The firing squad problem is not solvable for the set $\mathcal{D}_B$ of dynamic graphs with bounded delay connectivity, even if the size of the network n is given.*

Proof. By contradiction, suppose that there is an algorithm A solving the firing squad problem in any dynamic graph with n nodes and with bounded delay connectivity, and let V be a set of $n > 1$ nodes.

Let u, v be two distinct nodes in V, and for $x \in \{v, u\}$ let G_x be the graph consisting of a complete graph over $V \setminus \{x\}$ plus 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., $E_I = \{(v, v) : v \in V\}$.

We consider the run of A in which all nodes are active in the first round, and with the dynamic graph consisting of alternating sequence of directed graphs $\mathbb{G} = (G_u, G_v, G_u, G_v, \dots)$. Clearly, $\mathbb{G} \in \mathcal{D}_B$, and thus by FS1-2, all nodes fire at the same round t_F.

Now assume that $\mathbb{G}(t_F) = G_u$ (the case $\mathbb{G}(t_F) = G_v$ is similar). From the viewpoint of u, $\mathbb{G}$ is indistinguishable up to round t_F from the dynamic graph $\mathbb{G}^1$ that is similar to $\mathbb{G}$ except at round t_F where $\mathbb{G}^1(t_F) = I$. Hence u also fires at round t_F with the dynamic graph $\mathbb{G}^1$. Since $\mathbb{G}^1 \in \mathcal{D}_B$, all other nodes also fire at round t_F with $\mathbb{G}^1$. Using a similar argument, we get that from the viewpoint of v, $\mathbb{G}^1$ is indistinguishable up to round t_F from the dynamic graph $\mathbb{G}^2$ that is similar to $\mathbb{G}^1$ except at round $t_F - 1$, in which $\mathbb{G}^2(t_F - 1) = I$. Hence with $\mathbb{G}^2$, all nodes fire at round t_F as well.

By repeating this argument t_F times, we show that all nodes fire at round t_F in the run of A with start signals all received in the first round, and the dynamic graph $\mathbb{G}^{t_F} = (I, \dots, I, G_v, G_u, G_v, G_u, \dots)$. From the viewpoint of any node $v \neq u$, the latter run is indistinguishable up to round t_F from the run with the same dynamic graph $\mathbb{G}^{t_F}$ and where all nodes are active from round one except node u which is passive forever. All nodes other than u fire at round t_F, violating FS1 - a contradiction. ◀

4 Firing with a Bounded Diameter

As a first step towards our main positive result, which 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 *distance from node v to node w at time t*, denoted $d_t(v, w)$, is defined as the minimum positive integer δ such that there is a $v \sim w$ path in the interval $[t, t + \delta - 1]$. If for any $t' \geqslant t$ there is no $v \sim w$ path in the interval $[t, t']$, then conventionally $d_t(v, w) = +\infty$.

The *diameter* of the dynamic graph $\mathbb{G}$ is then defined as the minimum positive integer d such that for any positive integer t, the directed graph $\mathbb{G}(t : t + d - 1)$ is complete, or infinity if there is no such integer, namely $\operatorname{diam}(\mathbb{G}) = \sup_{t \geqslant 1,\, v,w \in V^2} d_t(v, w)$.

Let $\mathcal{D}$ be a set of dynamic graphs, and assume that a finite bound D on the diameters of the dynamic graphs in $\mathcal{D}$ is given. Then a solution to the firing squad problem is enabled by using local virtual clocks whose values may reach D only if all nodes are active. Moreover, if some virtual clock is set to D, then all virtual clocks are set to D at the same round. The corresponding algorithm, denoted A_D, does not use identifiers, and the computation and storage capabilities of the nodes do not grow with the network size. 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 multiset of non-null messages received by u in the current round. Thus M_u^* at round t is the multiset of messages sent to u by the nodes in $\mathrm{In}_u^*(t)$. If non-null messages are vectors of some size, then $M_u^{*(i)}$ denotes the multiset of the i-th entries of the messages in M_u^*.

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

Initialization:

$r_u \in \mathbb{N}$, initially 0

In each round t do:

send $\langle r_u \rangle$ to all processes and receive one message from each in-neighbor
if at least one received message is null **then**
  $r_u \leftarrow 0$
else
  $r_u \leftarrow 1 + \min_{r \in M_u^*}(r)$
end if
if $r_u \geqslant D$ **then**
  Fire
end if

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.** *Assume that $t < t'$ 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 $v \in In_u(t : t'-1)$ it holds that $r_v(t)$ is defined and $r_u(t') \leqslant r_v(t) + t' - t$.*

▶ **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 $r_u(t) = t - s_{\max}$.*

From the two above lemmas, we can prove the correctness of the algorithm A_D:

▶ **Theorem 9.** *The algorithm A_D solves the firing squad problem for any set of dynamic graphs with diameters at most 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 $O(\log D)$.*

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 that 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 $B_{c,T}$ that 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 size of the network is given.

The algorithm $B_{c,T}$ uses the same virtual clocks r_u as the previous algorithm A_D. Moreover, each node u collects the identities of the active nodes which u had heard of in a variable HO_u. Then node u fires when its virtual clock r_u is large enough compared to the size of its HO_u set.

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

Initialization:
1: $r_u \in \mathbb{N}$, initially 0
2: $HO_u \subseteq V$, initially $\{u\}$
In each round t **do:**
3: send $\langle r_u, HO_u \rangle$ to all processes and receive one message from each in-neighbor
4: **if** at least one received message is null **then**
5: $\quad r_u \leftarrow 0$
6: **else**
7: $\quad r_u \leftarrow 1 + \min_{r \in M_u^{*(1)}}(r)$
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: $\quad$ Fire
12: **end if**

A similar idea was first used in [14], and also later in *early stopping consensus algorithms* [11, 13] and in the counting algorithm of [16], but with different virtual clocks. This technique requires distinct node identifiers and long messages since each node u broadcasts HO_u in each round.

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 $S \subseteq V$, the following holds:*

$$|\Gamma_{\text{in}}(S) \setminus S| \geqslant \min(c, |\overline{S}|) \tag{1}$$

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

It can be shown that the converse of Lemma 11 also holds. Moreover, the set $\Gamma_{\text{out}}(S)$ of out-neighbors of S can be substituted for $\Gamma_{\text{in}}(S)$ in Lemma 11 since any directed graph G is c strongly connected if and only if its transpose G^T is. Using this out-variant of Lemma 11 and an easy induction, we check that the diameter of a dynamic graph that is c connected with delay T is bounded by $T\left\lfloor 1 + \frac{n-2}{c} \right\rfloor$.

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

▶ **Lemma 12.** *In each run of the algorithm $B_{c,T}$ on a dynamic graph $\mathbb{G}$ which 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*

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

Proof. If $t = 1$, then $s_u = 1$, $HO_u(t) = \{u\}$, $r_u(t) = 0$, and the lemma holds.

So assume now that $t \geqslant 2$, 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 $a+1$ sub-intervals $I_a, I_{a-1}, \ldots, I_1, I_0$ as follows:
1. if $b = 1$, then I_0 is the empty interval, else $I_0 = [t - b + 1, t - 1]$;
2. for $0 < i \leqslant a$, we set $I_i = [t - b - iT + 1, t - b - (i-1)T]$.
We check that $|I_0| = b - 1 < T$, and $|I_i| = T$ for $i > 0$. All the intervals I_i are thus non-empty, except I_0 that is empty if and only if $b = 1$.

Then by induction, we construct a sequence of at most $a + 2$ sets of nodes $S_{-1}, S_0, \ldots, S_k$ as follows:

1. $S_{-1} = \{u\}$.
2. Suppose that $S_{-1}, \ldots, S_i$, $-1 \leqslant i \leqslant a$, are constructed.
   - **a.** If $i = a$, then the construction stops.
   - **b.** Otherwise, $-1 \leqslant i \leqslant a-1$. We let $H_{i+1} = \mathbb{G}\big(I_{i+1}\big)$ and we distinguish three cases.
     - **i.** $i \geqslant 0$ and H_{i+1} contains no edge (w, v) such that $w \notin S_i$ and $v \in S_i$. Then the construction stops.
     - **ii.** H_{i+1} contains an edge (w, v) such that $w \notin S_i$ and $v \in S_i$, and there exists a $w{\sim}v$ broken path in I_{i+1}. Then the construction stops.
     - **iii.** Otherwise, we let $S_{i+1} = In_u(t-b-(i+1)T+1 : t-1) = In_u(I_{i+1} \cup \cdots \cup I_0)$, which is the union of S_i and of the set of S_i's in-neighbors in the directed graph H_{i+1}. In particular, if u has no proper in-neighbor in $H_0 = \mathbb{G}(I_0)$ (eg., if $b = 1$), then $S_0 = \{u\}$.

Let us observe that $S_{-1} \subseteq S_0$, and the sequence $\big(S_i\big)_{0 \leqslant i \leqslant k}$ is increasing. More precisely, using the T delayed c connectivity of $\mathbb{G}$ and Lemma 11, we obtain that for every index i, $1 \leqslant i \leqslant k$, if $S_i \neq V$, then $|S_i| - |S_{i-1}| \geqslant c$. By an easy induction, then we obtain the following lower bound on $|S_k|$.

▶ **Claim 13.** *If $S_k \neq V$, then the cardinality of S_k is at least $c\,k + 1$.*

Because of the way HO_u is updated (line 9 of the algorithm), we check the following claim by induction.

▶ **Claim 14.** *$HO_u(t)$ contains every set S_i for $-1 \leqslant i \leqslant k$, and in particular $S_k \subseteq HO_u(t)$.*

We now distinguish the following three exhaustive cases:

Construction terminated by (a): Since clearly $r_u(t) \leqslant t - 1 = aT + b - 1$, we have

$$(1-2c) + \frac{c}{T}\,(r_u(t)+2) \leqslant (ac+1) + \frac{c}{T}\,(b+1-2T) \leqslant ac+1 \ .$$

Moreover by Claims 13 and 14, it holds that $|HO_u(t)| \geqslant |S_k| \geqslant ac+1$. Hence

$$|HO_u(t)| \geqslant (1-2c) + \frac{c}{T}\,(r_u(t)+2) \ ,$$

which shows the lemma in this case.

Construction terminated by (b.i): In this case, $0 \leqslant k \leqslant a-1$ and so the interval I_{k+1} is defined and is of length T. Since $\mathbb{G}$ is connected with delay T, this implies that $S_k = V$. It follows that $HO_u(t) = V$ and the lemma trivially follows.

Construction terminated by (b.ii): We first observe that $S_k \neq V$. Thus by Claims 13 and 14, $|HO_u(t)| \geqslant |S_k| \geqslant ck+1$. Also, observe that the assumed $w{\sim}v$ broken path in I_{k+1} can be extended to a $w{\sim}u$ broken path in the interval $I_{k+1} \cup \cdots \cup I_0 = [t-b-(k+1)T+1, t-1]$. Since $b \leqslant T$, this implies by Lemma 7.1 that $r_u(t) \leqslant (k+2)T - 2$ or equivalently that $k \geqslant \frac{r_u(t)+2}{T} - 2$. Thus we get

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

which proves the lemma in this case. ◀

▶ **Theorem 15.** *The algorithm $B_{c,T}$ solves the firing squad problem for every set 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 they use messages 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 $|HO_u(t)| > \left\lceil \frac{c}{T}\left(r_u(t)+2\right)\right\rceil - 2c$, 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 10 eventually holds at each node u.

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

$$HO_u(t+1) \subseteq \bigcup_{s \geqslant s_u} \mathrm{In}^*_u(s:t) \ . \tag{3}$$

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.,

$$|HO_u(t_0+1)| \leqslant \left\lceil \frac{c}{T}\left(r_u(t_0+1)+2\right)\right\rceil - 2c \ . \tag{4}$$

From Lemma 12, we deduce 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 (3), there is a $v{\sim}u$ non-broken path 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 every node $w \in V$, $HO_w(t_0+1) = V$. Therefore the inequality (4) 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 only operations in the algorithm $B_{c,T}$ that involve the node identities are performing the union and extracting the cardinalities of the sets HO_u. Since the decisions made by the algorithm are determined only by the cardinalities of the sets HO_u and not by the actual values of the identities in these sets, it is clear that 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 15, 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 [24] which states that if passive nodes do not transmit any signal, then counting is impossible with asynchronous starts.

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 size 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 [16], which assumes that all nodes start simultaneously.

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, this solution is thus not satisfactory regarding its time complexity.

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. 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 $[\,0, 1/2\,)$. For this algorithm, it is assumed that the dynamic graph, and the start signals s_v, are managed by an *oblivious* adversary, which has no access to the outcomes of the random choices made by the algorithm.

The algorithm works as follows: upon becoming active, each node u generates ℓ independent random numbers $Y_u^{(1)}, \ldots, Y_u^{(\ell)}$, 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 $i \in \{1, \ldots, \ell\}$, 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.

Using Cramér-Chernoff's bounds [2], we show that with high probability, the value of n_u at the end of round t provides 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 $B_{c,T}$, we conclude that with high probability, no node ever fires in a non-active run, and all 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, by using a rounded and range-restricted calculations as in [20]. Specifically, we round down each $Y_u^{(i)}$ to the next smaller integer power of 13/12, denoted $\overline{Y}_u^{(i)}$. Then the resulted approximate value $\overline{n_u}$ of n_u satisfies $n_u \leqslant \overline{n_u} \leqslant \frac{13}{12} n_u$, which guarantees that with high probability, $\overline{n_u}$ is also a good approximation of the number of active nodes that u has heard of so far.

By the 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

$$\Pr\left[\forall u \in V,\ \forall i,\ Y_u^{(i)} \in [\eta/(4\ell N), \ln(4\ell N/\eta)]\right] > 1 - \eta/2 \ , \tag{5}$$

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 $O\left(\log\log(N/\eta)\right)$ bits. This leads to messages length in $O\left(\log(N/\eta) \cdot \log\log(N/\eta)\right)$ bits.

We note, however, that the implied calculations require exponentially higher storage capacities: computing $\overline{n_u}$ (line 14 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 may require $\Omega(\ell N \eta^{-1})$ bits.

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

Initialization:

$r_u \in \mathbb{N}$, initially 0

$\overline{Y}_u = \left(\overline{Y}_u^{(1)}, \ldots, \overline{Y}_u^{(\ell)}\right) \in \mathbb{R}^\ell$ with $\ell = \left\lceil 243 \cdot (\ln 4N^2 - \ln \eta)\right\rceil$, initially rounded and range-restricted approximation of independent random numbers with exponential distribution of rate 1.

$n_u \in \mathbb{N}$, initially 0

In each round t do:

send $\langle r_u, \overline{Y}_u \rangle$ to all processes and receive one message from each in-neighbor

if at least one received message is null **then**

  $r_u \leftarrow 0$

else

  $r_u \leftarrow 1 + \min_{r \in M_u^{*(1)}}(r)$

end if

for $i = 1, \ldots, \ell$ **do**

  $\overline{Y}_u^{(i)} \leftarrow \min_{\overline{Y}^{(i)} \in M_u^{*(i+1)}} \left(\overline{Y}^{(i)}\right)$

end for

$\overline{n_u} \leftarrow \ell / \sum_{i=1}^{\ell} \overline{Y}_u^{(i)}$

if $\overline{n_u} < 2\, r_u/3$ **then**

  fire

end if

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 (5), and this range restriction is violated with probability of at most $\eta/2$.

▶ **Theorem 16.** *In any dynamic graph that is continuously strongly connected and with at most N nodes, the algorithm $R_{N,\eta}$ solves the firing squad problem with probability at least $1-\eta$. 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.*

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, i.e., are no more supposed to be propagated by the nodes in the network.

We modeled 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: does the existence of a leader could be useful for achieving or improving solutions to the firing squad problem?

References

1 Sebastian Abshoff, Markus Benter, Andreas Cord-Landwehr, Manuel Malatyali, and 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*, pages 22–34, 2013.

2 Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities. A nonasymptotic theory of independence*. Oxford University Press, Oxford, 2013.

3 James E. Burns and Nancy Lynch. The byzantine firing squad problem. *Advances in Computing Research*, 4:147–161, 1987.

4 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. In Hannes Frey, Xu Li, and Stefan Rührup, editors, *ADHOC-NOW*, volume 6811 of *Lecture Notes in Computer Science*, pages 346–359. Springer, 2011.

5 Bernadette Charron-Bost and André Schiper. The Heard-Of model: computing in distributed systems with benign faults. *Distributed Computing*, 22(1):49–71, 2009.

6 Bernard Chazelle. Natural algorithms and influence systems. *Communications of the ACM*, 55(12):101–110, 2012.

7 Brian A. Coan, Danny Dolev, Cynthia Dwork, and Larry Stockmeyer. The distributed firing squad problem. In *ACM Symposium on Theory of Computing Conference, STOC'85*, pages 335–345, 1985.

8 Thiago Correa, Breno Gustavo, Lucas Lemos, and Amber Settle. An overview of recent solutions to and lower bounds for the firing synchronization problem. *arXiv preprint arXiv:1701.01045*, 2017.

9 Reinhard Diestel. *Graph Theory*. Springer-Verlag Berlin Heidelberg, 2017.

10 Danny Dolev, Ezra N. Hoch, and Yoram Moses. An optimal self-stabilizing firing squad. *SIAM Journal on Computing*, 41(2):415–435, 2012.

11 Danny Dolev, Rüdiger Reischuk, and H. Raymond Strong. Early stopping in Byzantine agreement. *jacm*, 37(4):720–741, 1990.

12 Danny Dolev and H. Raymond Strong. Authenticated algorithms for Byzantine agreement. 12(4):656–666, 1983.

13 Cynthia Dwork and Yoram Moses. Knowledge and common knowledge in a Byzantine environment: Crash failures. *Information and Computation*, 88(2):156–186, oct 1990.

14 Steven Finn. Resynch procedures and a fail-safe network protocol. *IEEE Transactions on Communications*, 27(6):840–845, 1979.

15 Julien M. Hendrickx, Alexander Olshevsky, and John N. Tsitsiklis. Distributed anonymous discrete function computation. *IEEE Trans. Automat. Contr.*, 56(10):2276–2289, 2011. doi:10.1109/TAC.2011.2163874.

16 Fabian Kuhn, Nancy A. Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on*

Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010, pages 513–522. ACM, 2010. doi:10.1145/1806689.1806760.

17 Fabian Kuhn, Yoram Moses, and Rotem Oshman. Coordinated consensus in dynamic networks. In *Proceedings of the 30th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–10. ACM, 2011.

18 Edward F. Moore. The firing squad synchronization problem. *Sequential Machines, Selected papers*, pages 213–214, 1964.

19 F. R. Moore and G. G. Langdon. A generalized firing squad problem. *Information and Control*, 12(3):212–220, 1968.

20 Rotem Oshman. *Distributed Computation in Wireless and Dynamic Networks*. PhD thesis, Massachusetts Institute of Technology, 2012.

21 Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. 27(2):228–234, 1980.

22 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*, pages 3:1–3:14, 2015.

23 T. K. Srikanth and Sam Toueg. Simulating authenticated broadcasts to derive simple fault-tolerant algorithms. *Distributed Computing*, 2(2):80–94, 1987.

24 Roger Wattenhofer. Principles of distributed computing. Unpublished, 2014.