Synchronisation dans les Réseaux Dynamiques

Bernadette Charron-Bost, Shlomo Moran

Synchronisation dans les Réseaux Dynamiques

Bernadette Charron-Bost[1] Shlomo Moran[2]

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

Abstract

Le problème appelé "firing squad" consiste à synchroniser les nœuds d'un système distribué. Ce problème a été originellement étudié pour une chaîne d'automates communicants à états finis. Au départ, tous les nœuds de la chaîne sont passifs excepté un des deux nœud extrémités chargé d'envoyer un message d'activation aux autres nœuds. Sur réception d'un message d'activation, un nœud devient actif et retransmet ce message à ses voisins. Lorsque tous les nœuds sont devenus actifs, ils doivent "faire feu" (i.e., entrer dans un état particulier) simultanément.

Ce problème se généralise au cas où chaque nœud peut devenir actif lorsqu'il le souhaite, sans devoir impérativement répondre à la sollicitation d'un autre nœud déjà actif. Les nœuds ne sont alors autorisés à faire feu que s'ils sont tous actifs. Nous étudions cette généralisation du "firing squad" lorsque les nœuds correspondent à des machines de Turing (avec un nombre d'état infini) et que la topologie est un graphe dynamique avec des arrêtes qui peuvent changer arbitrairement au cours du temps. Notre principal résultat est une caractérisation des graphes dynamiques pour lesquels le problème du firing squad est résoluble. Nous étudions aussi comment une information sur le diamètre du réseau ou encore la randomisation permettent d'améliorer les solutions algorithmiques à ce problème.

1 Introduction

Many distributed algorithms assume a *synchronous networked system*, in which computation is divided into *synchronized rounds* that are communication closed layers. In this model, it is typically assumed that each run of an algorithm is started by all nodes simultaneously, at the same round (e.g., [8, 4, 5]). In this work, we justify this assumption of synchronous starts for dynamic networks with no central control that monitors the node activities, but with sufficient connectivity assumptions.

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 that may change in each round. All nodes are initially *passive*, and a node becomes *active* upon receiving a *start signal* at an unpredictable time. The goal is then to guarantee that if all nodes received start signals, then the nodes should 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 immediately becomes 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. The firing squad problem (or FS problem, for short) is then formally specified by (FS1) a node fires if and only if all nodes have received start signals and (FS2) all the nodes that fire, fire at the same round.

As a basic synchronization abstraction, the fulfillment of FS1 and FS2 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 both 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. The latter application exemplifies the relevance of the model of non-diffusive start signals as a start signal corresponds there to the termination of a local computation.

The FS problem was originally studied in the context of automata theory (e.g., [6, 7]). This model considers a finite but unknown number n of nodes which are connected in a line. Nodes are identical *finite* state machines whose number of states is independent of n. 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; thus the above model basically 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.

The FS problem has also been studied in the context of fault tolerant distributed computations (e.g., [2, 3]). 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 known. Links are reliable and at most f nodes may be faulty, for various types of faults. 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. Start signals are assumed to be diffusive. Moreover, the problem specification is different: e.g., faulty nodes are allowed not to fire simultaneously. For all these reasons, our results on the dynamic FS problem are incomparable to the ones on the fault-tolerant FS problem.

2 Summary of our results

We consider a networked system with a set V of n nodes, with distinct identifiers. Nodes run identical local algorithms, i.e., their codes do not depend on node identifiers. Moreover, they do not know the network size n.

Computation proceeds in *synchronized rounds*: no node receives messages in round t that are sent in a round different from t. In round t, each node sends messages to all nodes, receives messages from some nodes, and finally goes to its next state and proceeds to round $t+1$. Nodes cannot access the number of the current round. Each node u is initially *passive*: it is part of the network, but sends only heartbeats, and does not change its state. Then it either becomes *active* by receiving a unique *start signal*, or remains passive forever.

Communications that occur at round t, including the heartbeats, are modelled by a digraph $\mathbb{G}(t) = (V, E_t)$ where the set of nodes is fixed while the set of edges may change from round to round. The sequence of digraphs $\mathbb{G} = (\mathbb{G}(t))_{t\in\mathbb{N}}$ is called a *dynamic graph*. Any set of dynamic graphs is called a *network model*. An algorithm A *solves the FS problem for the network model* $\mathcal{G}$ if for each dynamic graph $\mathbb{G}$ in $\mathcal{G}$ the run of A defined by $\mathbb{G}$ and any scheduling of start signals satisfies both FS1 and FS2.

We examine various connectivity properties that hold, not necessary round by round, but globally over finite periods of consecutive rounds. For formalizing these properties, we introduce the digraph $\mathbb{G}(t:t+T) = \mathbb{G}(t) \circ \cdots \circ \mathbb{G}(t+T)$, where $\circ$ denotes the Kronecker product of digraphs. We say that $\mathbb{G}$ is *connected with delay* T if it is strongly connected over each period of T consecutive rounds, i.e., all the digraphs $\mathbb{G}(t:t+T)$ are strongly connected. It is said to be *eventually connected* if for each round t, it is strongly connected over a sufficiently long period of consecutive rounds starting at t, that is to say there exists $s \geqslant t$ such that $\mathbb{G}(t:s)$ is strongly connected. The *dynamic diameter* of $\mathbb{G}$ is the minimum positive integer D, if it exists, such that for every positive integer t, there is a directed path between any two nodes in any period of D consecutive rounds starting at t. We easily check that if $\mathbb{G}$ is connected with delay T, then its diameter is finite and at most equal to $(n-1)T$.

For a positive integer T, let $\mathcal{G}_T$ denote the network model composed of all dynamic graphs that are connected with delay T. The union $\overline{\mathcal{G}} = \bigcup_{T=1}^{\infty} \mathcal{G}_T$ then consists of all dynamic graphs with *bounded delay connectivity*. Let $\mathcal{G}^{\diamond}$ denote the network model of eventually connected dynamic graphs. The relations among the above snetwork models are thus given by the strict inclusions:

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

The main contribution of this paper is a characterization of the connectivity properties that enable to solve the FS problem, with respect to this hierarchy.

2.1 Firing with a bounded diameter

As a preliminary step, we present an algorithm A_D in the case that a finite bound D on the diameter of the dynamic graph is given. Our algorithm is based on local virtual clocks whose values may reach D only if all nodes are active. Basically, u's local clock, denoted r_u, is the length of the shortest path of active nodes ending at u. The remarkable 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. Hence the firing test for the node u in A_D is just " $r_u \geqslant D$".

The algorithm A_D does not use node identifiers; its time complexity is in $O(D)$ and it uses only $O(\log D)$ bits per message.

2.2 Firing with T-Delayed Connectivity

We then present the algorithm B_T and show that it solves the FS problem in linear time for dynamic graphs that are connected with delay T while no bound on the diameter is given.

Each node maintains the same virtual clocks r_u as in A_D in order to compute the network size $|V|$, and $T|V|$ is then used as a bound on the network diameter. For that, each node u collects the identifiers of the active nodes that u has heard of in a variable HO_u. Then the strategy for firing is the same as in the algorithm A_D: The node u fires when $r_u \geqslant T|V|$, which is enforced by the condition $|HO_u| \leqslant \lceil (r_u + 2)/T \rceil$. This technique requires distinct node identifiers and long messages since each node u broadcasts HO_u in each round.

Algorithm B_T, firing with T-delayed connectivity

Initialization: % upon the receipt of the start signal
  $r_u \in \mathbb{N}$, initially 0 ; $HO_u \subseteq V$, initially $\{u\}$
In each round do:
  send $\langle r_u, HO_u \rangle$ to all and receive messages (from in-neighbors)
  if at least one received message is a heartbeat **then** $r_u \leftarrow 0$
  else $r_u \leftarrow 1 + \min\{r : \langle r, HO \rangle \text{ is received }\}$
  $HO_u \leftarrow \bigcup_{\langle r, HO \rangle \text{ is received}} HO$
  if $|HO_u| \leqslant \lceil (r_u + 2)/T \rceil$ **then** Fire

Theorem 1. *The algorithm B_T solves the FS problem for the network model $\mathcal{G}_T$. Moreover, in any active run all nodes fire in less than nT rounds after all nodes have become active and messages are of size $O(n \log n)$.*

It should be noted that as a byproduct, the algorithm B_T thus solves the problem of counting the network size despite asynchronous starts in any model of dynamic graphs that are connected with delay T, and in particular in the model of continuously strongly connected dynamic graphs.

2.3 Bounded delayed connectivity is not sufficient

On the negative side, we show that under the sole assumption that the network is connected with some unknown delay T, the FS problem is unsolvable.

Theorem 2. *The FS problem is not solvable for the network model $\overline{\mathcal{G}}$ of dynamic graphs with bounded delay connectivity, even if the start signals are diffusive and the network size is known.*

As an immediate corollary of Theorems 1 and 2, we obtain that if at each round, the communication graph may be any member of a given set of digraphs over the same set of nodes, then the FS problem is solvable if and only if each member in the set is strongly connected.

2.4 Bound on the network size and randomization

Finally, we show that if a polynomial bound N on the network size is given, then randomization may reduce the message size in B_T without degrading its linear time complexity. We present a Monte Carlo algorithm that may produce runs in which FS1 or FS2 are violated with arbitrary small probability. The algorithm assumes that the dynamic graph is generated by an oblivious adversary, which determines the complete sequence of digraphs before the run begins.

For the sake of simplicity, we present the algorithm in the case $T = 1$. 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 $[\,0, 1/2\,)$, and works as follows: upon becoming active, each node u generates ℓ independent random numbers $Y_u^1, \ldots, Y_u^\ell$ 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 $r_u > 1.5\, n_u$ (instead of $r_u > n_u$ as in B_1).

The asymptotic Cramér-Chernoff bounds (e.g., see [1]) show that for large enough ℓ, the value of n_u provides with high probability a good approximation of the number of active nodes that u has heard of so far. Moreover, if $r_u > 1.5\, n_u$, then with high probability node u has heard of all nodes. More precisely, $\ell = \left\lceil 243 \cdot (\ln 4N^2 - \ln \eta) \right\rceil$ enforces a final probability of at least $1 - \eta$ for these successful active and non-active runs.

Sending exact values for Y_u^i would require nodes to send real numbers, which cannot be represented using a bounded number of bits. Instead nodes send rounded and range-restricted approximations of Y_u^i, each of them using $O\left(\log\log(N/\eta)\right)$ bits.

Theorem 3. *The algorithm $R_{N,\eta}$ solves the FS problem with probability at least $1 - \eta$ for the network model $\mathcal{G}_1$. 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 and messages are of size $O\left(\log(N/\eta) \cdot \log\log(N/\eta)\right)$.*

References

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

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

[3] 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.

[4] Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, STOC'10, pages 513–522, New York, NY, USA, 2010. ACM.

[5] 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.

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

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

[8] Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, April 1980.