The Distributed Computing Column

Seth Gilbert
National University of Singapore
seth.gilbert@comp.nus.edu.sg

This month, in the Distributed Computing Column, Amitabh Trehan explores a seemingly simple problem: flooding with no memory. Here we have perhaps the simplest possible distributed algorithm, and yet he illustrates surprising depth. Along the way he raises the fundamental question of when is there only one algorithmic solution to a problem (and what does that mean)?

The Distributed Computing Column is particularly interested in contributions that propose interesting new directions and summarize important open problems in areas of interest. We are also interested in understanding the impact of distributed computing, inteprereted broadly. If you would like to write such a column, please contact me.

AMNESIAC FLOODING AND THE CURIOUS CASE OF UNIQUE ALGORITHMS

Amitabh Trehan Durham University
amitabh.trehan@durham.ac.uk

Abstract

Amnesiac Flooding [6, 7, 9] is the stateless/historyless/amnesiac variant of probably the oldest and simplest of distributed algorithms: (classic) flooding. Beginning with a message at a set of initiator(s), the algorithm at every node is simply a single rule: if the message is received from some neighbour(s), immediately forward it to the rest of the neighbours. Note that unlike classic flooding, no copy of the message or history of the flooding is retained to ensure termination/quiescence of the messages. Yet, surprisingly, amnesiac flooding begun from any set of initiators (even in different rounds) terminates on all undirected graphs in the synchronous message passing model. Moreover, it terminates in optimal rounds in bipartite graphs and is at most twice as slow in non-bipartite graphs. A series of results have followed the first discovery, improving our understanding of the process and its variants.

Even more recently Austin, Gadouleau, Mertzios, and Trehan [3, 5] discovered *uniqueness* - under certain reasonable conditions including statelessness, amnesiac flooding is the only algorithm that solves terminating broadcast! As algorithm designers, the study of lower bounds and impossibility results (no algorithm exists) is well established, but we are not aware of results concerning uniqueness or, in general, even a sensible notion of countability of solutions to problems. This article presents some of the fundamental and interesting results around amnesiac flooding from its discovery to the present while not being a comprehensive review of the area that the initial result has spawned.

1 Introduction

Sometimes sloppiness can lead to happy accidents—at least, it did so, in this case. While giving a lecture which introduced the classic flooding algorithm, I neglected to add the standard condition that a node discards a message if it sees it again

(making it necessary for the node to keep copies of messages flooded in the past). The algorithm that remained (Definition 1.1), I named *Amnesiac Flooding* due to its in-built forgetfulness. The classic flooding algorithm is one of the earliest and simplest algorithms in the field of distributed computing. Informally, the classic flooding algorithm is: *if I have the message, I send it immediately to all my neighbours (or to all my neighbours who have not sent me the message in the preceding round); If I get the message again, I ignore the new receipt and do nothing,* Flooding is extremely useful because it is simple, it achieves *broadcast* (i.e. the message is received by every node in the network) and it terminates in optimal time (equivalent to the diameter of the network). Not surprisingly, it is used as a fundamental building block for many other algorithms. For example, in Peleg's time optimal Leader Election algorithm [11], a leader is elected by every node flooding the lowest ID it has seen so far to all its neighbours for diameter number of rounds. As James Aspnes summarises: "Flooding is about the simplest of all distributed algorithms. It is dumb and expensive but easy to implement" [2].

Amnesiac Flooding is thus so: *if I have the message, I send it immediately to all my neighbours who have not just sent me the message.* This variant obviously achieves broadcast, but does it terminate? The immediate answer of every expert I talked to initially when I posed the question was that the process should be non-terminating, i.e., messages would circulate ad-infinitum and all hell would break loose in Network land. Nobody, however, had a proof either way, and the subsequent stubborn search to prove the 'obvious' led to a fruitful path of discovery and even to the creation of a wikipedia page [1].

1.1 Amnesiac Flooding

The algorithm in the fault-free synchronous message passing model is defined as follows. The definition below is for all initiators starting simultaneously (for more general definitions, refer to [3–5]).

Definition 1.1. ***Amnesiac flooding algorithm (Synchronous).*** *(from [5] (Adapted from [9]) Let $G = (V, E)$ be an undirected graph, with vertices V and edges E (representing a network where the vertices represent the nodes of the network and edges represent the connections between the nodes). Computation proceeds in synchronous 'rounds' where each round consists of nodes receiving messages sent from their neighbours. A receiving node then sends messages to some neighbours in the next round. No messages are lost in transit. The algorithm is defined by the following rules:*

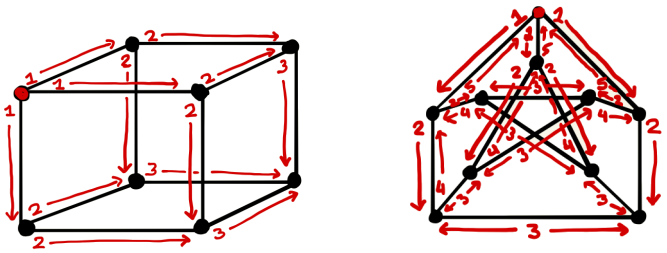
(i) All nodes from a subset of sources or initial nodes $I \subseteq V$ send a message M to all of their neighbours in round 1.

(ii) In subsequent rounds, every node that received M from a neighbour in the previous round, sends M to all, and only, those nodes from which it did not receive M. Flooding terminates *when M is no longer sent to any node in the network.*

In subsequent discussions, we shall restrict ourselves to a single initiator node (i.e. $|I| = 1$), unless otherwise stated. In the next section (Section 1.2), we try out some simple examples by hand (Figures1 and 2).

1.2 Some Illustrative Examples

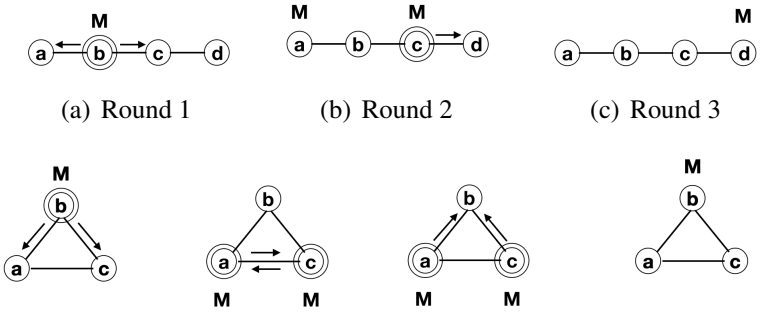
Consider two well known graphs in Figure 1; the hypercube (cube in 3 dimensions) graph and the Petersen graph. These graphs are symmetric so the starting point does not influence the execution of amnesiac flooding with respect to termination time. Consider AF on the hypercube first (Figure 1(a)) - it is easy to see that it stops in round 3 when the node diagonally opposite the origin gets M from all of its neighbours simultaneously. On the Petersen graph (Figure 1(b)), the process terminates in 5 rounds stopping at the origin itself. In terms of diameter, termination on the hypercube takes diameter time but the Petersen graph takes 2* diameter + 1 rounds.



(a) AF on the Hypercube (b) AF on the Petersen Graph

Figure 1: Two well known graph topologies (Hypercube and the Petersen Graph) and the execution of Amnesiac flooding (AF) (from the red coloured node) on them. Arrows point to direction of the transmission with the label giving the round number. Double headed arrows indicate the message crossing in both directions.

To get a better understanding, let's look at some simpler base cases. Figure 2 shows flooding over a line graph and a triangle graph. On the line on 4 nodes, the process beginning with the node b in the figure terminates at the ends of the graph taking only 2 rounds, which is equal to the *eccentricity*(the length of the shortest longest path from a vertex in a graph) of node b in the graph (whose diameter is 3). Note that a line is an example of a bipartite graph. In the triangle graph,



(a) Round 1 (b) Round 2 (c) Round 3

(d) Round 1 (e) Round 2 (f) Round 3 (g) Round 4

Figure 2: AF over line and triangle networks with circled nodes sending M in that round. On the above line, beginning with node b, AF terminates in 2 ($< diameter = 3$) rounds. When AF is executed over the above Triangle graph beginning with node b, both node a and c send M to each other in round 2 and to b in round 3. This is an odd (# nodes) cycle topology (also, a clique) with termination taking $2 * diameter + 1$ rounds.

termination from any initiator (the graph being symmetric) takes only 3 rounds, However, here the diameter is only 1. The triangle is also the smallest clique and the smallest non-trivial cycle with an odd number of nodes (an important subgraph, as it turns out). Potentially, a kind of pattern is emerging: On a symmetric bipartite graph (such as the hypercube) termination takes exactly diameter rounds, on a (non-symmetric) bipartite graph such as a line, it may even take less than diameter time but never more; On symmetric non-bipartite graphs such as a triangle and the Petersen graph, it is taking time that is exactly twice the diameter of the graph plus 1.

2 Termination

Ultimately, it turns out that this simple stateless process of Amnesiac flooding is terminating and termination time stated in terms of eccentricity of the initiator nodes has a strong co-relation with bipartiteness of the graph.

2.1 The First Proof

The first proof showing that amnesiac flood terminates on any graph appears in [6, 7, 9]. This is a proof by contradiction. It relies on a simple but useful observation:

Observation 2.1. *Any event that occurs infinitely often, observed on a discrete*

time scale, must have at least two occurrences separated by an even interval.

In fact, one can make a stronger claim - any event that occurs 3 or more times must have two occurrences separated by an even interval. Let these three occurrences be at time steps t_1, t_2, and t_3. If all these t's are even, then all the intervals are even (e.g. $t_2 - t_1$). If all these t's are odd, then again all the intervals are even since the difference of two odd numbers is even. For the other cases, either there are two even numbers and their difference is even, or there are two odd numbers, and their difference is again even. Using the observation above, if amnesiac flooding was non-terminating, there must be some node that receives the message infinitely often, in particular, at least 3 times, and therefore, at an even interval. We assume such a node and prove by contradiction that there is no such node showing that amnesiac flooding is terminating. The formal proof follows below (from [7]):

Definition 2.1. *Let G be a graph. The round-sets $R_0, R_1, \ldots$ are defined as:*

R_0 *is the singleton containing an initial node,*
R_i *is the set of nodes which receive a message at round i* $(i \geq 1)$.

Clearly, if $R_j = \emptyset$ for some $j \geq 0$, then $R_i = \emptyset$ for all $i \geq j$. We shall refer to rounds R_i, where $R_i \neq \emptyset$, as *active* rounds.

Theorem 2.2. *Any node $g \in G$ is contained in at most two distinct round-sets.*

Proof. Define $\mathcal{R}$ to be the set of finite sequences of consecutive round-sets of the form:

$$\underline{R} = R_s, \ldots, R_{s+d} \quad \textit{where } s \geq 0,\ d > 0, \textit{ and } R_s \cap R_{s+d} \neq \emptyset\ . \tag{1}$$

In (1), s is the *start-point* $s(\underline{R})$ and d is the *duration* $d(\underline{R})$ of $\underline{R}$. Note that, a node $g \in G$ belonging to R_s and R_{s+d} may also belong to other R_i in (1). If a node $g \in G$ occurs in three different round-sets R_{i_1}, R_{i_2} and R_{i_3}, then the duration between R_{i_1} and R_{i_2}, the duration between R_{i_2} and R_{i_3}, or the duration between R_{i_1} and R_{i_3} will be even. Consider the subset $\mathcal{R}^{EV}$ of $\mathcal{R}$ of sequences of the form (1) where d is even. To prove that no node is in three round-sets, it suffices to prove that $\mathcal{R}^{EV}$ is empty.

We assume that $\mathcal{R}^{EV}$ is non-empty and derive a contradiction.

Let $\mathcal{R}^{EV}_{\hat{d}}$ be the subset of $\mathcal{R}^{EV}$ comprising sequences of minimum (even) duration $\hat{d}$, i.e.

$$\mathcal{R}^{EV}_{\hat{d}} = \{\underline{R} \in \mathcal{R}^{EV} \ \mid\ \forall\ \underline{R}' \in \mathcal{R}^{EV}.\ d(\underline{R}') \geq d(\underline{R}) = \hat{d}\} \tag{2}$$

Clearly, if $\mathcal{R}^{EV}$ is non-empty then so is $\mathcal{R}^{EV}_{\hat{d}}$. Let $\underline{R}^* \in \mathcal{R}^{EV}_{\hat{d}}$ be the sequence with earliest start-point $\hat{s}$, i.e.

$$\underline{R}^* = R_{\hat{s}}, \ldots, R_{\hat{s}+\hat{d}} \tag{3}$$

where

$$\forall\ \underline{R}' \in \mathcal{R}^{EV}_{\hat{d}}\ .\ s(\underline{R}') \geq s(\underline{R}^*) = \hat{s} \tag{4}$$

By (1), there exists $g \in R_{\hat{s}} \cap R_{\hat{s}+\hat{d}}$. Choose node g' which sends a message to g in round $\hat{s} + \hat{d}$. As g' is a neighbour of g, either g' sends a message to g in round $\hat{s}$ or g sends a message to g' in round $\hat{s} + 1$. We show that each of these cases leads to a contradiction.

Case (i): g' sends a message to g in round $\hat{s}$

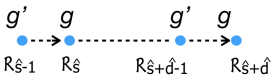


Figure 3: Node g' sends a message to node g in round $\hat{s}$: the first round of the minimum even length sequence (of length $\hat{d}$) in which g repeats

Refer to Figure 3. In this case, there must be a round $\hat{s} - 1$ which is either round 0 and g' is the initial node, or g' received a message in round $\hat{s} - 1$. Thus, the sequence

$$\underline{R}^{*'} = R_{\hat{s}-1}, R_{\hat{s}}, \ldots, R_{\hat{s}+\hat{d}-1} \quad \textit{where } g' \in R_{\hat{s}-1} \cap R_{\hat{s}+\hat{d}-1} \tag{5}$$

has $d(\underline{R}^{*'}) = (\hat{s} + \hat{d} - 1) - (\hat{s} - 1) = \hat{d}$ which is even and so $\underline{R}^{*'} \in \mathcal{R}^{EV}_{\hat{d}}$. As $\underline{R}^{*'} \in \mathcal{R}^{EV}_{\hat{d}}$, by (4)

$$s(\underline{R}^{*'}) \geq s(\underline{R}^*) \tag{6}$$

But, from (5), $s(\underline{R}^{*'}) = \hat{s} - 1$ and, from (4), $s(\underline{R}^*) = \hat{s}$. Thus, by (6),

$$\hat{s} - 1 = s(\underline{R}^{*'}) \geq s(\underline{R}^*) = \hat{s}$$

which is a contradiction.

Case (ii): g sends a message to g' in round $\hat{s} + 1$

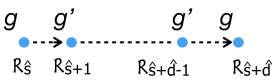


Figure 4: Node g sends a message to node g' in round $\hat{s} + 1$: round $\hat{s}$ is the first round of the minimum even length sequence (of length $\hat{d}$) in which g repeats

Refer to Figure 4. By the definition of $\mathcal{R}^{EV}$, the smallest possible value of $\hat{d}$ is 2. However, it is not possible to have $\hat{d} = 2$ in this case as then $\underline{R}^* = R_{\hat{s}}, R_{\hat{s}+1}, R_{\hat{s}+2}$. This would mean that g sends a message to g' in round $\hat{s}+1$. But, we chose g' to be such that g' sends a message to g in round $\hat{s}+\hat{d} = \hat{s}+2$. This cannot happen as g cannot send a message to g' and g' to g in consecutive rounds by the definition of rounds. Thus,

$$\underline{R}^* = R_{\hat{s}}, R_{\hat{s}+1}, \ldots, R_{\hat{s}+\hat{d}-1}, R_{\hat{s}+\hat{d}} \text{ where } \hat{s}+1 < \hat{s}+\hat{d}-1$$

Consider the sequence

$$\underline{R}^{*''} = R_{\hat{s}+1}, \ldots, R_{\hat{s}+\hat{d}-1} \tag{7}$$

As g' receives a message from g in round $\hat{s}+1$ and g' sends a message to g in round $\hat{s}+\hat{d}$, it is clear that $g' \in R_{\hat{s}+1} \cap R_{\hat{s}+\hat{d}-1}$. Thus, $\underline{R}^{*''} \in \mathcal{R}$. As $\hat{d}$ is even, so is $(\hat{s}+\hat{d}-1)-(\hat{s}+1) = \hat{d}-2$ and therefore $\underline{R}^{*''} \in \mathcal{R}^{EV}$. Now, $\underline{R}^* \in \mathcal{R}^{EV}_{\hat{d}}$ and so, as $\underline{R}^{*''} \in \mathcal{R}^{EV}$, we have, by (2),

$$d(\underline{R}^{*''}) \geq d(\underline{R}^*) \tag{8}$$

As $d(\underline{R}^{*''}) = \hat{d}-2$ from (7) and $d(\underline{R}^*) = \hat{d}$ from (3), we have, by (8),

$$\hat{d}-2 = d(\underline{R}^{*''}) \geq d(\underline{R}^*) = \hat{d}$$

This contradiction completes the proof.

□

Theorem 2.2 implies that $R_i = \emptyset$ for $i \geq 2n$, where n is the number of vertices of G. Therefore amnesiac flooding always terminates. The corollary below gives a loose bound on termination time (tighter ones follow later):

Corollary 2.2.1. *Synchronous amnesiac flooding always terminates in fewer than* $2n+1$ *rounds.*

3 Termination Time

We have previously seen that Amnesiac Flooding on any synchronous network will visit every node at most twice and thus, terminate, in at most $2n+1$ rounds. However, we can find much better termination times. This maybe a good time to remind us of classic flooding and its termination time. The classic stateful flooding algorithm maybe summarised as follows:

From a source s, send a message to all its neighbours. Every receiving node forwards the message to all its neighbours, or all neighbours having not just received the message from. If the message is received again, ignore the message and do not forward.
This algorithm will achieve broadcast with termination time in optimal time i.e. in $eccentricity(s)$ (plus potentially one additional ruound) time where s is the initiator of the message. Time optimality is easy to see since the farthest node cannot receive the message before $eccentricity(s)$ rounds.

3.1 Almost Trivial Termination on Bipartite Graphs

As it turns out, proving termination and showing that this time matches the optimal is almost trivial for bipartite graphs. It is easy to see that amnesiac flooding is a bit like doing BFS exploration and one can now think of this exploration in terms of a traversal tree (Figure 5). The exploration happens level wise where each level is explored simultaneously. However, since the graph is bipartite, there are no cross-edges and by the amnesiac flooding property, a message cannot go back up to the previous level. Thus, the messages can only go down to the next level with that whole level getting explored simultaneously. Thus, amnesiac flooding will terminate in the number of levels, which is the same as eccentricity of the initiator node. Thus, amnesiac flooding not only terminates on bipartite graphs, it does so in optimal time. Note that the smallest eccentricity in a graph is known as the *radius* and the largest is the *diameter* and diameter is never more than twice the radius. The following Theorem follows:

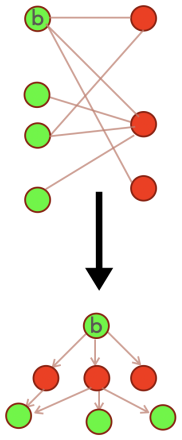


Figure 5: Amnesiac Flooding in a bipartite graph starting from a node b.

Theorem 3.1. *In a bipartite graph G, amnesiac flooding initiated from a node g_0 terminates in rounds = $e(g_0)$, where $e(g_0)$ is the eccentricity of the vertex g_0 in G.*

3.2 Termination on Non-Bipartite Graphs

The situation is a lot more complicated for non-bipartite graphs. Thinking in terms of the traversal, there are now cross edges and since there's no memory, messages can easily traverse back to the parent level from an already traversed level. Hussak and Trehan [7, 9] present and prove the termination bounds (Theorem 3.2). They do so by direct arguments about amnesiac flooding and by defining *ec nodes* (equidistantly connected nodes). *ec nodes* are two neighbouring nodes equidistant from an initiator node and they exist if and only if the graph is non-bipartite. Turau [12] proves similar bounds by constructing a bipartite *auxillary graph* for every non-bipartite graph (by connecting two copies of the original graph that have had their cross edges removed).

Theorem 3.2. *([7], Theorem 12) Let G be a non-bipartite graph with diameter d and let $g_0 \in G$ be an initial node of eccentricity e. Then, amnesiac flooding terminates in j rounds where j is in the range $e < j \leq e + d + 1$.*

Combining Theorems 3.1 and 3.2, and from their proofs, we get the following powerful general theorem:

Theorem 3.3. *Let G be a graph with diameter d and let $g_0 \in G$ be an initial node of eccentricity e. Then, amnesiac flooding terminates in rounds $= e(g_0)$, where $e(g_0)$ is the eccentricity of the vertex g_0 in G, if and only if G is bipartite, otherwise (if and only if the graph is non-bipartite), amnesiac flooding terminates in j rounds where j is in the range $e < j \leq e + d + 1$.*

Given that eccentricity is upper bounded by the diameter of the graph, Corollary 3.3.1 follows:

Corollary 3.3.1. *Let G be a graph with diameter d. Amnesiac flooding initiated from any initiator node on G terminates in at most d rounds if G is bipartite, and in at most $2d + 1$ rounds, if G is non-bipartite.*

One can easily verify that the bounds in Theorem 3.3 are tight: AF in the symmetric bipartite hypercube graph (Figure 1) terminates from any initiator vertex in diameter (equal to eccentricity) rounds, and in the symmetric non-bipartite graphs Triangle (Figure 2) and Petersen graph (Figure 1) in twice the diameter ((equal to eccentricity)) plus 1 rounds. However, when node eccentricity is less than the diameter, on the bipartite line graph (Figure 2, with node b as initiator), AF terminates in eccentricity (less than diameter) time. For non-bipartite graphs, one can construct graphs and choose initiator nodes such that amnesiac flooding can terminate in, for any chosen j, in the range $e < j \leq e + d + 1$. For example, in Figure 6, AF terminates in only $e(= 5) + 1$ rounds when begun from node a

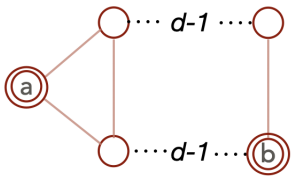


Figure 6: Amnesiac Flooding on graph G begun from node a terminates in $e + 1$ rounds, whereas from node b it terminates in $e+d+1$ rounds where $e = e(a) = e(b)$ is the eccentricity of the nodes and d is the diameter of G.

but in $e(= 5) + d(= 5) + 1$ rounds when begun from node b. The tight nature of the if and only if result of Theorem 3.3 raises the intriguing question of whether there maybe stateless/semi-stateless algorithms that can be used for testing bipartiteness/diameter approximation using a run of amnesiac flooding. However, the dependence on eccentricity (rather than diameter) suggests this maybe challenging.

3.3 Multiple Starters and Multiple Floodings

Though the results in [7] are stated for a single starter, it also states that these can be extended to multiple starters (due to the nature of the proofs) and this is done in the journal version of the paper [9] by an elegant adaptation and restatement of the property of bipartiteness in the context of multiple starters. The proofs and results then follow in a straightforward manner. Extending the definitions of *ec nodes* and *eccentricity* to a set I of starters (from a singleton node), we proposed the notion of $I - Bipartiteness$:

Definition 3.1. *The graph (G, E) is I-bipartite iff (G, E) has no ec-nodes. Equivalently, (G, E) is I-bipartite iff the quotient graph of (G, E), in which the nodes of I are contracted to a single node, is bipartite.*

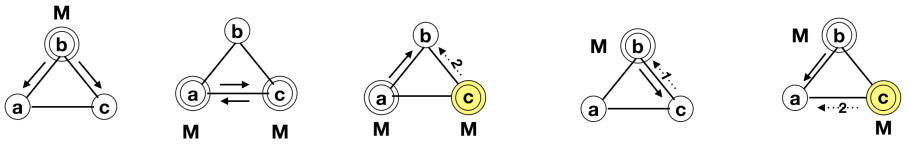
This leads to a restatement of Theorem 3.3 as follows (Theorem 3.4).

Theorem 3.4. *Let $G(V, E)$ be a graph with diameter d and let $I \subseteq V$ be a set of initial nodes. Then, amnesiac flooding terminates in rounds $= e(I)$ if and only if G is bipartite, and otherwise in j rounds where j is in the range $e(I) < j \leq e(I)+d+1$, where $e(I)$ is the eccentricity of the set I (defined as $e(I) = max\{distance(I, g) : g \in V\}$) and d is the diameter of G.*

This result is also tight, and a general condition for meeting the $e(I) + 1$ lower bound on non-bipartite graphs is shown in [9].

Potentially, from a more practical consideration, the behaviour of amnesiac flooding and other related floodings when there are multiple messages of interest in the network, is of interest. This is more so if we are in the $CONGEST$ model where messages are limited to (poly)logarithmic size. Assuming each message is of logarithmic size, one can easily execute a constant number of amnesiac floodings in a parallel, stateless manner with all the nice properties. In general, there could be different rules and floodings (distinct from amnesiac flooding) which could be executed. Hussak and Trehan discuss some variants and their termination in [8]. Maybe the latest results on uniqueness (Section 6) could have some bearing on that discussion.

4 Asynchronous Amnesiac Flooding



(a) Round 1 (b) Round 2 (c) Round 3 (d) Round 4 (e) Round 5

Figure 7: Asynchronous AF over a Triangle with an adaptive scheduler. Both node a and c send M to each other in round 2. In round 3, a sends M to b but the adversary makes c hold the message for one round (shaded node). In the next round, we have a round analogous to round 2 and so on.

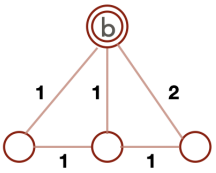


Figure 8: Non-termination of AF under a fixed delay on a single edge. The numbers on the edges give the number of rounds required for delivery.

Assuming a completely asynchronous model with completely local independent clocks and arbitrary delivery times, a question arises right off the bat: *For a node, when can one say that messages from two distinct neighbours have been received simultaneously?*. Note that this question is central to executing AF - since, in AF, the node floods to the other neighbours. This and other considerations led

us to propose the *Round Asynchronous model* as one simplified model to posit amnesiac flooding in [6, 7, 9].

Definition 4.1. ***Round-Asynchronous Model.*** *Computation proceeds in rounds where each round consists of nodes receiving messages scheduled to be delivered to it in that round, does local processing, and outputs (possibly different) messages to its neighbours. For every message, a* scheduling adversary *decides in which future (but unknown to the nodes) round the message is delivered to its destination. The adversary could be* adaptive *i.e. it can decide a delivery time for every message, or* fixed *i.e. for every edge, it decides on a fixed delay for that edge which does not change during the execution of the algorithm.*

For the stronger adaptive model, Figure 7 shows an execution of AF under an adversary strategy that leads to non-terminating broadcast. Remarkably, under the fixed adversary, we found that a single fixed delay on a single edge can lead to non-terminating broadcast as shown in Figure 8.

5 Dynamism and Fault-Sensitivity

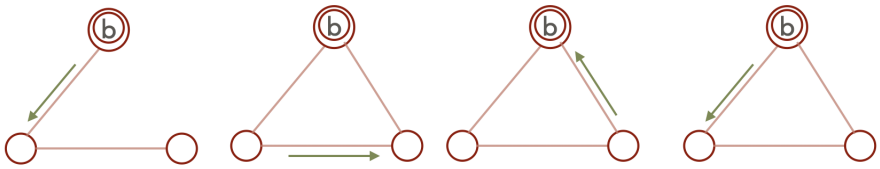


Figure 9: Amnesiac Flooding not terminating on insertion of an edge

In the previous section (Section 4), we saw (in the asynchronous setting), the delay of a single time step on a single edge can lead to non-termination. In a similar flavour, in [3–5], Austin, Gadouleau, Mertzios, and Trehan show that dropping even a single message on a single edge can lead to non-termination in certain cases. This work tries to delve deeper into the structural properties of amnesiac flooding in a formal manner and discovers the sensitivity of amnesiac flooding. Besides the just mentioned single message failure non-termination, the paper also shows potential non-termination in case of a single uni-directional link failure, or a weak byzantine failure (as defined in the paper). Combined with the easy observation that on the family of directed graphs, a uni-directional ring is an obvious topology for non-termination, one can see that amnesiac flooding can be very sensitive. This may not be the full story, however, since one can conjecture that another or a larger number of faults may actually restore terminating broadcast.

Earlier versions [8] have also studied dynamism in the form of node and edge deletion and insertion and conclude that Amnesiac Flooding is stable i.e. continues to do terminating broadcast in case of node and edge deletions (potentially requiring connectivity for broadcast) but cannot guarantee the same in case of edge or node insertions (Figure 9).

6 Amnesiac Flooding is Unique

Uniqueness may be a lightly used English word but Austin, Gadouleau, Mertzios, and Trehan [3–5] show that amnesiac flooding is unique in a formal technical sense: *Amnesiac Flooding is the only algorithm that can solve terminating broadcast under certain reasonable conditions*. Moreover, relaxing any of these conditions allows more algorithms to solve the problem. We believe that even this statement is unique - at least, we are not aware of results in the field of Algorithms design that can state that a single (or even a finite number of algorithms) can solve a particular problem. In fact, one would need to devise a reasonable definition of uniqueness to make such a statement - we do have an easier way to do so in our setting as will be seen shortly.

The formal statement is as follows (from [3–5]):

Theorem 6.1 (Uniqueness of Amnesiac Flooding)**.** *Any terminating broadcast algorithm possessing all of* Strict Statelessness*,* Obliviousness*,* Determinism *and* Unit Bandwidth *behaves identically to Amnesiac Flooding on all graphs under all valid labellings for all source nodes.*

Elaborating, the four conditions are as follows:

1. *Strict Statelessness*: Nodes maintain no information other than their port labellings between rounds. This includes whether or not they were in the initiator set.

2. *Obliviousness*: Routing decisions may not depend on the contents of received messages.

3. *Determinism*: All decisions made by a node must be deterministic.

4. *Unit Bandwidth*: Each node may send at most one message per edge per round.

Violation of even one of the above conditions allows us to propose other terminating broadcast algorithms. For example, a randomised algorithm which chooses between amnesiac flooding and flooding to every neighbour at every occasion of flooding, will eventually achieve terminating broadcast.

In a general setting, potentially one of the first hurdles one can imagine is how to rule out trivial extensions of any algorithm such as delaying execution of a command by a single wait statement, for example. Fortunately, in our setting of true statelessness, this does not seem possible, because delaying execution would necessitate use of memory and state to remember the command that is delayed. Beyond that, proving the result was technically challenging and we hope, this is a result and a direction which the research community will find interesting.

7 Conclusions

Amnesiac Flooding is a stateless version of flooding which was introduced in 2019. The synchronous version of this was shown to terminate in optimal number of rounds in bipartite graphs and in at most twice longer time in non-bipartite graphs. In fact, it was subsequently shown to terminate from any number of starters (simultaneous or non-simultaneous) with the termination time partitioning the graphs on the basis of an extension of bipartiteness called I-bipartiteness. A number of other interesting results, variants and related questions have been proposed in recent times including broader results on memory requirements for broadcast algorithms [10]. A most recent and exciting result, in our opinion, is that amnesiac flooding is the only stateless deterministic oblivious (to message content) algorithm that can achieve terminating broadcast. Amnesiac flooding is thus, technically, a unique algorithm. This poses the question of quantification of algorithmic solutions to particular problems.

References

[1] Amnesiac flooding, October 2025. https://en.wikipedia.org/wiki/Amnesiac_flooding.

[2] James Aspnes. Flooding, February 2019. http://www.cs.yale.edu/homes/aspnes/pinewiki/Flooding.html.

[3] Henry Austin, Maximilien Gadouleau, George B. Mertzios, and Amitabh Trehan. Amnesiac Flooding: Easy to Break, Hard to Escape. In Dariusz R. Kowalski, editor, *39th International Symposium on Distributed Computing (DISC 2025)*, volume 356 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:23, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.DISC.2025.10, doi:10.4230/LIPIcs.DISC.2025.10.

[4] Henry Austin, Maximilien Gadouleau, George B. Mertzios, and Amitabh Trehan. Amnesiac flooding: Easy to break, hard to escape. *CoRR*, abs/2502.06001, 2025. URL: https://doi.org/10.48550/arXiv.2502.06001, arXiv:2502.06001, doi:10.48550/ARXIV.2502.06001.

[5] Henry Austin, Maximilien Gadouleau, George B. Mertzios, and Amitabh Trehan. Brief announcement: Amnesiac flooding: Easy to break, difficult to escape. In Alkida Balliu and Fabian Kuhn, editors, *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2025, Hotel Las Brisas Huatulco, Huatulco, Mexico, June 16-20, 2025*, pages 541–544. ACM, 2025. doi:10.1145/3732772.3733523.

[6] Walter Hussak and Amitabh Trehan. On termination of a flooding process. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019.*, pages 153–155. ACM, 2019. doi:10.1145/3293611.3331586.

[7] Walter Hussak and Amitabh Trehan. On the termination of flooding. In Christophe Paul and Markus Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 17:1–17:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.STACS.2020.17.

[8] Walter Hussak and Amitabh Trehan. Terminating cases of flooding. *CoRR*, abs/2009.05776, 2020. URL: https://arxiv.org/abs/2009.05776, arXiv:2009.05776.

[9] Walter Hussak and Amitabh Trehan. Termination of amnesiac flooding. *Distributed Comput.*, 36(2):193–207, 2023. URL: https://doi.org/10.1007/s00446-023-00448-y, doi:10.1007/S00446-023-00448-Y.

[10] Garrett Parzych and Joshua J. Daymude. Memory Lower Bounds and Impossibility Results for Anonymous Dynamic Broadcast. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing (DISC 2024)*, volume 319 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:18, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.DISC.2024.35, doi:10.4230/LIPIcs.DISC.2024.35.

[11] David Peleg. Time-optimal leader election in general networks. *J. Parallel Distrib. Comput.*, 8(1):96–99, January 1990. doi:10.1016/0743-7315(90)90074-Y.

[12] Volker Turau. Amnesiac flooding: synchronous stateless information dissemination. In *SOFSEM 2021: Theory and Practice of Computer Science: 47th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2021, Bolzano-Bozen, Italy, January 25–29, 2021, Proceedings 47*, pages 59–73. Springer, 2021.