capstract:2611.00070v1 [math.CO] 20 Nov 2026

On the Enumeration of Removal Requests in Broadcast Threads

Elara J. Selnor
Department of Discrete Mathematics, Orisel Institute of Mathematical Sciences, Veyport

November 20, 2026

Abstract

We study a broadcast thread in which recipients request removal by replying to all, while removal is deferred until no replies remain. Under a memory-h rule, a recipient replies to a request unless they occur among its h most recent ancestral senders. For a list of n participants, the thread terminates if and only if $h = n$. With full ancestry memory the number of requests is $\lfloor e(n-1)! \rfloor$ for $n \geq 2$, although no participant sends twice on a single ancestry. With memory $n-1$, there are $(n-1)!$ requests at every sufficiently large depth. We derive the depth counts, individual traffic, and the limiting distribution of the number of eligible senders on the ancestry of a uniformly selected request. An atomic shared registry instead admits exactly n requests. The counts are verified by explicit enumeration of sender sequences.

Keywords: enumeration, self-avoiding walks, branching, broadcast communication

1 Introduction

A participant who wishes to leave a group email thread may request removal by replying to all recipients. If other participants also wish to leave, a removal request can give rise to further removal requests. We consider a list on which every recipient wishes to be removed and on which removal is processed only after the thread has become quiet. The messages being counted are the requests for removal themselves.

The rule for suppressing duplicate requests matters. A participant may avoid sending a second request in the ancestry of a message while still responding to separate branches of the thread. We compare this rule with bounded ancestry memory and with a shared registry that permits only one request per participant across the whole thread. In each case, all requests are sent to the original list; a previously sent request does not remove its sender from the list.

The ancestry of a request is a word over the participant set. With full ancestry memory it is a self-avoiding walk on a complete graph, a setting studied by Slade [3] and Deng et al. [1]. The enumeration of the resulting words also follows from standard constructions for labelled sequences [2]. We use this representation to count requests, deliveries, and the number of people who have already requested removal along a typical ancestry.

The distinction between local and shared memory is substantial. With full ancestry memory the thread stops, but its request count grows factorially. For a list of ten, it contains 986,410 requests before removal can be processed. If only the nine most recent senders in each ancestry are remembered, the thread never becomes quiet. A shared registry reduces the count to ten.

2 Model

Let $X = \{1, \dots, n\}$ with $n \geq 2$, and let participant 1 send an initial removal request. Each message is addressed to all other $n-1$ participants. A recipient who is eligible under the chosen memory rule sends exactly one reply to all, also requesting removal. All eligible replies are eventually delivered, and each message retains its own chain of ancestors. Different branches are not merged.

We identify a message at depth t with its sender sequence $(x_0, \dots, x_t)$, where $x_0 = 1$. Its parent is obtained by deleting x_t. Every eligible recipient gives a distinct child, so message counts are counts of such sequences, not counts of distinct message bodies. The initial request has depth zero. A delivery is one message received by one recipient; it is distinct from a sent message.

Under the *memory-h* rule, for $1 \leq h \leq n$, a recipient replies if and only if they do not occur among the last h senders of the message's ancestry, or anywhere in that ancestry if it contains fewer than h senders. Thus the current sender is always ineligible. The rule is applied separately on every branch. In particular, a request on one branch does not suppress a request by the same person on another.

For $h = n$ we call the rule *full ancestry memory*. Since there are only n participants, no sender can recur on a branch. As a comparison, the *shared-registry* rule admits at most one request per participant across all branches, using an atomic check when the request is sent. Each of the other participants receives the initial request and sends one reply, so the total is exactly n.

These are deterministic counting models. They do not require a probability of replying or a distribution of delivery times. In particular, a pending reply is counted even if a different branch has already produced a request from its sender.

3 Counts by depth

Let $M_t(n, h)$ be the number of messages at depth t, and write $(a)_r = a(a-1)\cdots(a-r+1)$ with $(a)_0 = 1$.

Theorem 1 (Bounded ancestry memory)**.** *For $1 \leq h < n$ and $t \geq 0$,*

$$M_t(n, h) = (n-1)_{\min(t, h-1)} (n-h)^{\max(0, t-h+1)}. \tag{1}$$

For full ancestry memory,

$$M_t(n, n) = \begin{cases} (n-1)_t, & 0 \leq t \leq n-1, \\ 0, & t \geq n. \end{cases} \tag{2}$$

Proof. At depth t, the last $\min(h, t+1)$ senders are distinct. This is true at the root and remains true when a sender outside that suffix is appended. Every message at depth t therefore has exactly $n - \min(h, t+1)$ children. Multiplying these factors from the root gives Eq. (1). When $h = n$, the factors are $n-1, n-2, \dots, 1, 0$, yielding Eq. (2). ∎

Corollary 2 (Termination threshold)**.** *The thread becomes quiet after finitely many depths if and only if $h = n$. For $h = n-1$, there are $(n-1)!$ requests at every depth $t \geq n-2$.*

Proof. For $h < n$, every sufficiently deep message has $n - h \geq 1$ children. At $h = n-1$, Eq. (1) reduces to $(n-1)!$ for $t \geq n-2$. For $h = n$, Eq. (2) eventually vanishes. ∎

The threshold includes the two-person case: with memory one, the two participants continue to reply to one another; with full ancestry memory, the second request has no eligible recipient. Figure 1 shows the first six depths for a list of six. Counts at depth six and later are zero under full memory, so they are omitted from the logarithmic plot.

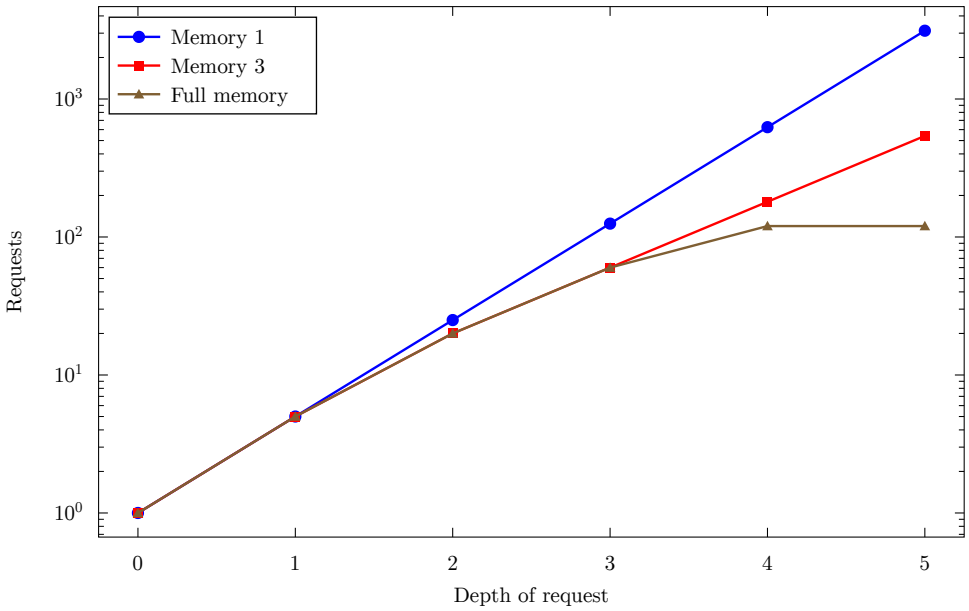


Figure 1: Requests by depth for six participants under three memory rules. Full ancestry memory has no requests after depth five.

4 Full ancestry memory

Denote the total number of requests by $T_n = \sum_t M_t(n, n)$. Every request is delivered to $n-1$ recipients, including recipients who have already appeared on its ancestry. Consequently the total number of deliveries is $(n-1)T_n$.

Theorem 3 (Total requests)**.** *For $n \geq 2$,*

$$T_n = \sum_{t=0}^{n-1} \frac{(n-1)!}{(n-1-t)!} = (n-1)! \sum_{j=0}^{n-1} \frac{1}{j!} = \lfloor e(n-1)! \rfloor \,. \tag{3}$$

Proof. The first expression follows from Eq. (2) and the second by setting $j = n-1-t$. For $m = n-1 \geq 1$, the omitted tail of $em!$ is positive and at most $(m+2)/(m+1)^2 < 1$: after the first omitted term $1/(m+1)$, successive terms decrease by factors at most $1/(m+2)$. Taking the floor proves the result. ∎

Writing $T_1 = 1$ for the one-person case, the exponential generating function in the number of non-initiators is

$$\sum_{m \geq 0} T_{m+1} \frac{z^m}{m!} = \frac{e^z}{1-z}. \tag{4}$$

This is the product of the exponential series and the ordinary sequence generating function; it also gives $T_n \sim e(n-1)!$.

Proposition 4 (Individual traffic)**.** *The initiator sends one request and receives $T_n - 1$ deliveries. Every other participant sends $(T_n - 1)/(n-1)$ requests and receives $T_n - (T_n - 1)/(n-1)$ deliveries.*

Proof. The initiator occurs at the start of every ancestry and cannot appear again. All non-initiators are interchangeable under permutations that fix participant 1, and together they send the remaining $T_n - 1$ requests. Every message is delivered to all participants except its sender, giving the stated receipt counts. ∎

The identity $(T_n - 1)/(n-1) = T_{n-1}$ holds for $n \geq 3$. Thus each non-initiator on a ten-person list sends as many requests as the entire full-memory process on nine people. Table 1 gives exact values; the shared-registry benchmark sends n requests in each row.

Table 1: Full ancestry memory. Deliveries count one copy per recipient; each non-initiator's requests are counted across all branches.

List size n	Requests T_n	Requests per non-initiator	Deliveries
3	5	2	10
4	16	5	48
5	65	16	260
6	326	65	1,630
7	1,957	326	11,742
8	13,700	1,957	95,900
9	109,601	13,700	876,808
10	986,410	109,601	8,877,690

5 A uniformly selected request

Choose one of the T_n messages uniformly after the full-memory process has finished. Let U_n be the number of participants who have not yet appeared in its ancestry. For $j = 0, \ldots, n-1$, messages with $U_n = j$ have depth $n - 1 - j$. Hence

$$\Pr(U_n = j) = \frac{1/j!}{\sum_{q=0}^{n-1} 1/q!}. \tag{5}$$

Proposition 5 (Remaining eligible senders)**.** *As $n \to \infty$, U_n converges in distribution to a Poisson random variable of mean one. Its exact expectation is*

$$\mathbb{E}U_n = \frac{\sum_{q=0}^{n-2} 1/q!}{\sum_{q=0}^{n-1} 1/q!}. \tag{6}$$

Proof. The probabilities in Eq. (5) tend to $e^{-1}/j!$, which sums to one. The mean follows by summing $j/j! = 1/(j-1)!$ for $j > 0$. ∎

In particular, the proportion of requests with no eligible sender left on their ancestry tends to e^{-1}. These terminal requests still go to all other participants. They produce no further replies under full memory, but their delivery is part of the traffic.

6 Verification and scope

We checked Eq. (1) and Eq. (2) by explicitly generating every sender sequence for $2 \le n \le 6$, every memory length $1 \le h \le n$, and depths through six. All 20 parameter pairs agreed with the formulas. Table 1 was calculated using integer arithmetic from Eq. (3). The plot uses the exact depth counts; there are no sampled observations.

A shared registry removes duplicates across branches, whereas ancestry memory removes them only along a single branch. This is a difference in the scope of the record, not its retention time. The shared-registry comparison assumes atomic admission and reliable delivery of the initial request. It does not model delayed or lost mail, partial participation, departures during the process, or messages with any purpose other than requesting removal. Under the stated rule, removal occurs only after the last request has been delivered; bounded memory with $h < n$ therefore never reaches that point.

References

[1] Youjin Deng, Timothy M. Garoni, Jens Grimm, Abrahim Nasrawi, and Zongzheng Zhou. The Length of Self-Avoiding Walks on the Complete Graph. *Journal of Statistical Mechanics: Theory and Experiment*, 2019:103206, 2019. doi: 10.1088/1742-5468/ab3da3.

[2] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, Cambridge, 2009. doi: 10.1017/CBO9780511801655.

[3] Gordon Slade. Self-Avoiding Walk on the Complete Graph. *arXiv preprint arXiv:1904.11149*, 2019. doi: 10.48550/arXiv.1904.11149.