December 2020

Scalable Near Realtime Loss and Duplicate Detection from Received Sequence Numbers

N/A

Scalable Near Realtime Loss and Duplicate Detection from Received Sequence Numbers

ABSTRACT

In a system where a transmitter transmits packets to a number of receivers, a packet may be received one or more times over the set of receivers or may be lost (not received at any of the receivers). The problem is to determine, in a scalable and computationally feasible manner, packets that have been received over the set of receivers and packets that have been lost. For such single-transmitter-multiple-receiver scenarios, this disclosure describes scalable and computationally efficient techniques to detect missing or duplicate packets based on sequence numbers. Each receiver maintains an array of bits corresponding to packets it received. A logical-OR operation across bit arrays reveals the sequence numbers of packets that are lost. A logical-AND between two bit arrays reveals duplicates between those two bit arrays. The result of a pairwise logical-OR operation on two or more bit arrays, when logically ANDed with another bit array, reveals the sequence numbers of packets that have been received in duplicate.

KEYWORDS

- Duplicate packet
- Lost packet
- Packet sequence number
- Load balancing
- Distributed communication

BACKGROUND

In a system where a transmitter transmits packets to a number of receivers, a packet may be received one or more times over the set of receivers or may be lost (not received at any of the receivers). The problem is to determine, in a scalable and computationally feasible manner,

packets that have been received over the set of receivers exactly once, packets received over the set of receivers more than once, and packets that have never been received by any of the receivers.

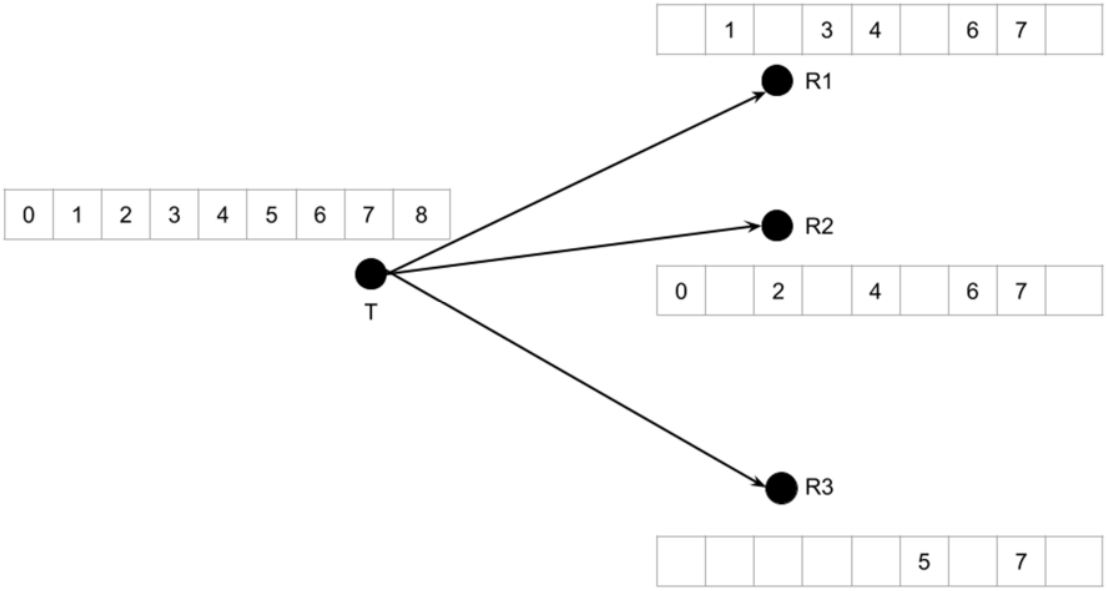

Fig. 1: A single-transmitter-multiple-receiver scenario, where a packet may be received one or more times by a given receiver or may be lost

This is illustrated in Fig. 1, where a transmitter T transmits a sequence of packets numbered strictly monotonically - 0 through 8 - in an order that increases with time.

- The receiver R1 receives packets 1, 3, 4, 6, and 7; it loses packets 0, 2, 5, and 8.
- The receiver R2 receives packets 0, 2, 4, 6, and 7; it loses packets 1, 3, 5, and 8.
- The receiver R3 receives packets 5 and 7; it loses packets 0, 1, 2, 3, 4, 6, and 8.

Between all three receivers, packets 0, 1, 2, 3, and 5 are received once; packets 4 and 6 are received in duplicate; packet 7 is received in triplicate; and packet 8 is lost.

Existing techniques to address packet-loss detection in the described distributed communications scenario are fairly complex. These techniques require an intermediate step of

determining jitter; do not scale well with the number of receivers; require additional transmitter-receiver communications, e.g., a packet send-back; etc.

DESCRIPTION

For single-transmitter-multiple-receiver scenarios, this disclosure describes scalable and computationally efficient techniques to detect missing or duplicate sequence numbers. A receiver maintains an array of bits corresponding to packets it received or didn't. The index to the array is the packet number. An array entry is 1 if the packet is received successfully and 0 otherwise.

A logical-OR operation across bit arrays reveals the sequence numbers of packets that are lost. A logical-AND between two bit arrays reveals duplicates between those two bit arrays. Alternatively, the result of a pairwise logical-OR operation on two or more bit arrays, when logically ANDed with another bit array, reveals the sequence numbers of packets that have been received in duplicate. The logical-OR and logical-AND operations can be carried out by one or more aggregator modules or threads. The described techniques can scalably handle a large number of receivers, since the receivers only communicate bit-statuses upwards with aggregator thread(s). The length of the bit array, also referred to as the queue, can be adjusted to capture the longest expected delay across the communications channel between the transmitter and the receivers.

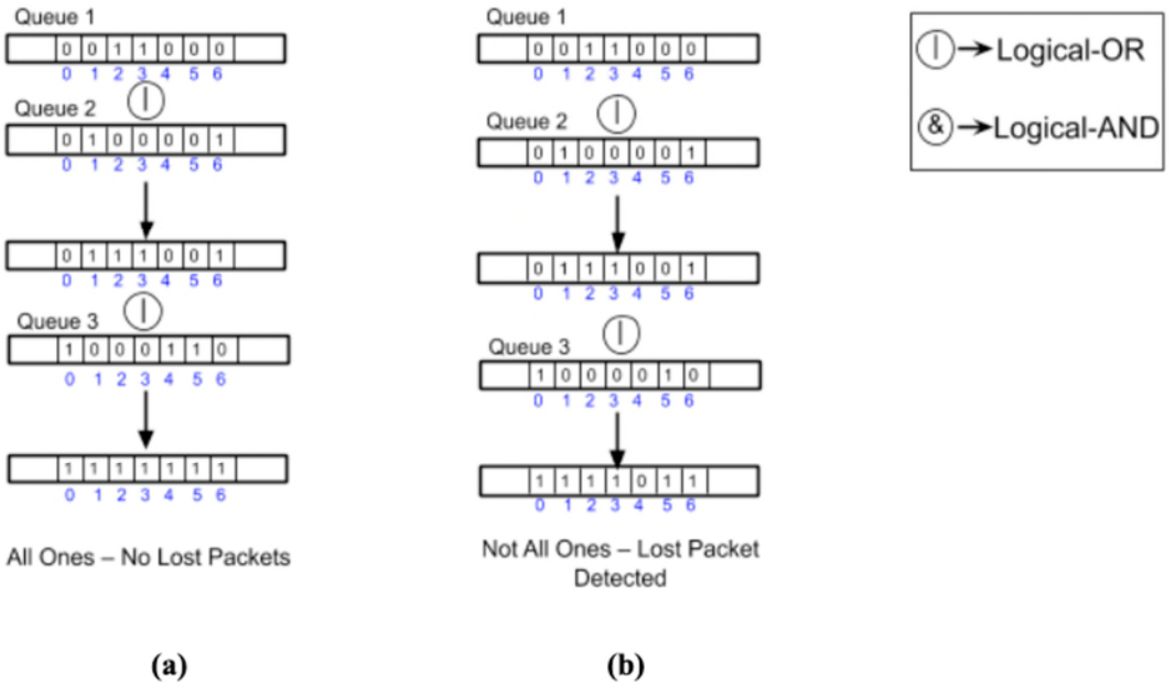

Fig. 2: Detecting lost packets by performing a logical OR on the bit-arrays of the receivers: (a) no packet loss; (b) packet loss detected

Fig. 2 illustrates detecting lost packets by performing a logical OR on the bit-arrays of the receivers. As mentioned above, each receiver maintains a bit-array, also known as a queue, indicating the status of the packets received by it. The numbers in blue font in Fig. 2 are the packet sequence numbers.

In Fig. 2(a), receiver 1 has received packets 2 and 3, and hence marks the corresponding entries of queue-1 as 1 and marks the remaining entries of queue-1 as 0. Similarly, receiver 2 has received packets 1 and 6, and hence marks the corresponding array entries of queue-2 as 1 and marks the remaining entries of queue-2 as 0. Receiver 3 has received packets 0, 4, and 5, and hence marks the corresponding entries of queue-3 as 1 and marks the remaining entries of queue-3 as 0. A logical OR of the three queues reveals that each packet has been received by at least one receiver and no packets are lost. Similarly, in the example of Fig. 2(b), a logical OR all the three queues reveals that packet 4 has not been received by any of the receivers - it has been lost.

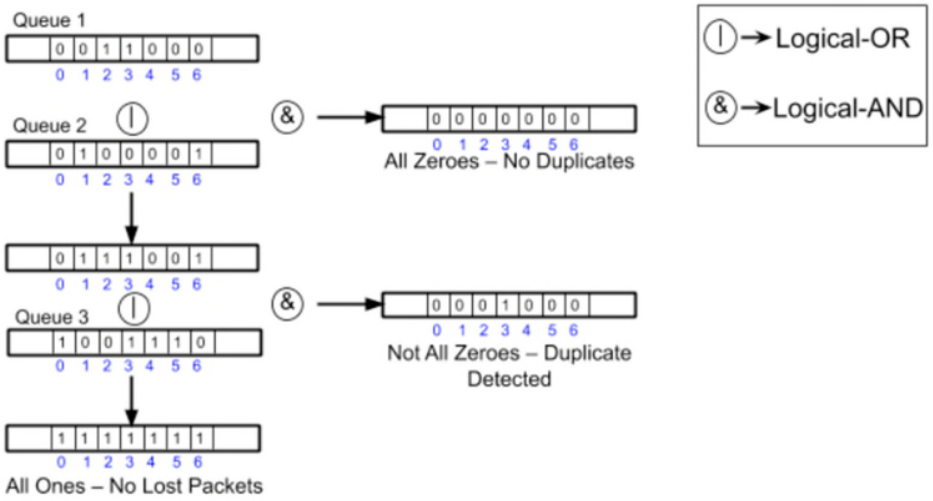

Fig. 3: Detecting duplicate packets by performing a pairwise logical-OR followed by a logical-AND

Fig. 3 illustrates the detection of duplicate packets amongst the set of receivers, per the techniques of this disclosure. A logical-AND between two bit arrays reveals duplicates between those two bit arrays. A logical-AND of queues 1 and 2 results in an all-zero array since there are no duplicates between receivers 1 and 2. To detect duplicates across three queues, e.g., queues 1, 2, and 3, a procedure can be as follows. A logical-AND between queue-3 and the result of a pairwise logical-OR of queues 1 and 2 reveals a 1 in packet 3, indicating that packet 3 has been received in duplicate between receiver 3 and one of receivers 1 and 2. (Fig. 3 also illustrates the carrying out of the logical-OR of all three queues, resulting in the conclusion that no packets are lost.) The same result can be obtained by performing logical multiplication between all pairs of queues. This approach allows identifying the exact set of all receivers that received a given packet without any ambiguity. Testing for duplicates received by other receivers, e.g., a receiver 4, can similarly be carried out by logically ANDing queue-4 with the result of the logical-OR of queues 1, 2, and 3.

In practice, the described bit-arrays or queues at each receiver can be implemented as ring buffers, such that bit-statuses of the most recently received packets (the head of the ring

buffer) overwrite the bit-statuses of the earliest received packets (the tail of the ring buffer). The length of the ring buffer can be set to a conservatively large number, e.g., a multiple of the largest expected delay between the transmitter and the receivers. Slices of each ring are periodically sent to aggregator thread(s) that perform the above-described logical operations to detect packet losses and duplicates. When a configurable interval passes, e.g., the number of dirty aggregated slices crosses a selected value, the oldest slice is checked for any zero bits, which are then declared as lost packets.

Duplicate packets can be communicated in a channel separate from lost packets. As mentioned above, the number of aggregation stages can be one, e.g., all threads sending slices to one aggregation thread, or many, e.g., multiple aggregation stages can be used to split the load. One way to do this is to maintain an aggregation thread for even packet numbers and another aggregation thread for odd packet numbers. Another way is to build a (binary or other) tree that resembles the hierarchy of aggregation threads. Maintaining multiple aggregation stages reduces bottlenecks in evaluating the sequence of received packet numbers and further enhances scalability.

CONCLUSION

For single-transmitter-multiple-receiver scenarios, this disclosure describes scalable and computationally efficient techniques to detect missing or duplicate packet sequence numbers. A receiver maintains an array of bits corresponding to packets it received. A logical-OR operation across bit arrays reveals the sequence numbers of packets that are lost. A logical-AND between two bit arrays reveals duplicates between those two bit arrays. The result of a pairwise logical-OR operation on two or more bit arrays, when logically ANDed with another bit array, reveals the sequence numbers of packets that have been received in duplicate.