**Given the relatively 'old' age of this obviously classic paper, I thought it'd be prudent to view a presentation that provides more context about its background and effects, so I came across this one by Jason Brown (worked at Apple during this talk): https://youtu.be/UiPeZrDeoBI*

1 Summary

For context, this is an industry research paper by Xerox published in the late-80s discussing the problems they ran into and some solutions to those problems related to their directory lookup service, Clearinghouse, that was a 'productization' of their earlier Grapevine research paper from the early-80s. In addition to introducing novel algorithms to maintain mutual consistency among (database) sites during updates, this paper was also the first to discuss usage of epidemic algorithms (gossip) in practice. Among the algorithms in the paper were (1) Direct Mail (each new update triggers updates to all other sites from its entry), (2) Anti-Entropy (updates come in the form of each site randomly and semi-regularly picking another site with which to exchange database contents and resolve differences accordingly, but this is obviously expensive), and (3) Rumor Mongering, the basis of gossip, based off of the following concepts of epidemics: infective (site with update that it is willing to share), susceptible (site that hasn't received that update), and removed (site that received the update but is no longer willing to share it).

2 Strengths of the paper

Given that this paper was written by Xerox to offer solutions to problems that were plaguing their (and probably everyone else's) databases at the time, the key contribution of this paper is the discussion on epidemic-like processes, i.e., *anti-entropy* and *rumor mongering*. In order to increase reliability and improve performance within the new process(es) to maintain mutual consistency between sites during updates, Xerox aimed to make their implementation agnostic of underlying communication protocols (TCP/IP) and they also knew to implement randomization in terms of communication between nodes within the network. These were two novel and important concepts that also aid today's distributed systems. Another interesting aspect of the paper is the discussion on handling distributed deletes, since handling deletes is a relatively complex endeavor in a distributed system. "Death certificates" (modern systems like Cassandra and Riak call this "tombstone") are a 'logical delete' in that they affix some metadata to a file signifying that it has been deleted instead of physically deleting it right away, thus preventing other nodes from re-adding the deleted file to the newer node during reconciliation (anti-entropy). These certificates are retained for a fixed time length, and then the relevant files are eventually physically deleted by garbage collection processes running locally on each node.

3 Major weakness of the paper

I do wish the authors would divulge more information about how the database administrators assessed whether the 'jobs' ran to completion with the rumor mongering process. It wasn't quite clear to me, although I am fairly certain that this was a non-issue given that had there been issues regarding completion, the rumor mongering protocol would likely not have gained the traction that it did.

4 Future work opportunities

Although I am not familiar with it myself, Jason Brown claims that upon deep inspection, the processes outlined in this paper seems quite similar to modern NoSQL databases. He also highlights two modern gossip use cases in particular: (1) Cassandra which stems from Professor van Renesse's Scuttlebutt paper, and (2) Riak which stems from the Plumtree paper. Gossip lends itself to being an attractive process to be implemented in distributed systems that prioritize minimal resource use and aim to be scalable. Apache Cassandra uses the Gossip protocol for failure detection of cluster nodes, as well as transmitting the cluster nodes' metadata.