Willis Allstead
11/8/18
CS 491

Review of Dynamo: Amazon's Highly Available Key-value Store

Amazon is a company built on data management, and it shows when they write a paper on a technology they've invested in. They put thought into every detail of this 13 page paper, and left little to the imagination regarding how Dynamo works. This could explain why, 12 years after the publication of this paper they still implement a system based on these ideas called DynamoDB. The overall presentation of each concept is clear, and it follows a path that makes sense to the reader. Just as they start mentioning a concept the reader may be unfamiliar with such as "consistent hash ring", they give an short explanation of what that is, they might direct the reader to another paper if they have an interest, and they explain why they chose to use it in their implementation over the alternatives. Essentially, Amazon created Dynamo to fill an important and unique roll in their website. They needed a key-value store that was highly available, reliable, and consistent. They also wanted a key-value store that they could customize how it dealt with node failures on per-application basis. For example, if Dynamo was to fail in use on the shopping cart, they knew the data schema for that application so they could make certain decisions on how to recover when node failure did occur.

These problems that Dynamo is built to solve are extremely important to Amazon because they are entirely an internet-based company. If users experience outages, that directly impacts the profits of Amazon. Specifically during highly demanding times such as the holiday season Amazon, even in 2006, was selling up to 3,000,000 products in a single day. Critical node failure without proper correction during these times could mean Amazon losing hundreds of thousands in a day. Implementing failure detection and recovery is fairly complex, especially when you want to keep high efficiency in the system.

Amazon has ideas that they like their internal services to follow to keep their system running smoothly. These include the concept of decentralization, loose coupling, and maintaining a service-oriented system. Amazon had design Dynamo to not only perform well under load, but to also follow these tenants so that in the case of one part of the system experiencing critical failure, it would not take down the entire site.

The related techniques mentioned in the paper were peer-to-peer systems and distributed file systems/databases. They explained the various issues that both pose when wanting to achieve the sorts of results demanded by the SLAs created with the Amazon organization that require certain speeds for services to run at.

One key issue in this paper is the fact that they fail to include numerical evidence that their key-value store is in any way better than these alternatives. The reader can safely assume now, looking back, that the performance of Dynamo was superior to some degree compared to the alternatives based on the mere fact that Amazon would most likely choose the fastest and most reliable technology since it simply would mean more profit. But, the reader should not be expected to jump to these conclusions.

At a high level, Dynamo works by partitioning and replicating data using consistent hashing. It ensures consistency by facilitating object versioning. It also ensures consistency between replicas during updates by employing a quorum-like technique in tandem with a decentralized replica synchronization protocol. Finally, it detects failures through use of a local gossip-based detection protocol.

The authors convinced the readers that Dynamo performs well by noting that it was actually already used in production by the time of writing. It was created for and implemented by the holiday season of 2004 on amazon.com on a number of services on the site. They also provide graphs showing that Dynamo met the requirements for the various SLAs set for it, beating the requirements by up to 33% in the average read/write per second case.

One significant limitation of Dynamo is that it actually performs worse when it comes to fraction of nodes out-of-balance when its request load drops very low. This aspect is one that I think could present opportunity for development in Dynamo or systems that came later. It could be that their attempt to keep Dynamo extremely decentralized has actually hindered the ability of Dynamo to correct fast enough when popular keys in the consistent hash ring start falling off at a high speed. If they considered using the same system they explain in the article, but also include a single node who's sole responsibility was to ensure updates could be passed quicker to each node in the ring, they might be able to fix this issue.

This paper was a pleasure to read, in part because it comes straight from the industry, even if it was 12 or so years ago. You get to see in part why Amazon became such a powerful company, it was because they cared about the small details in every part of the user experience.