Colin McClelland
CPSC 324 324 Final Project

Google Napa relates closely to the concepts that have been discussed throughout the semester. Napa performs automatic scaling to meet the demands of client workload and to ensure that its clients consistently receive sub-second query performance. Client databases at Napa are replicated across data centers such that the underlying data is available despite any level of outage. If an outage were to occur clients are redirected to any one of the available replicas that are guaranteed to be consistent. As a system, Napa must handle vast amounts of data that is continuously generated by Google's products. Napa is the combination of three sub-systems that are built entirely out of the existing Google infrastructure. A table in Napa is maintained as a collection of Colossus files. Like GFS, Colossus employs a shared disk architecture and has similar approaches for coordination and fault tolerance. In addition, Spanner is used to track system state and F1 Query is the query engine. Napa is the backend for both internal and external products at Google and powers many of their dashboards. Napa also employs a unique size-based file page layout. Smaller delta files use the PAX layout while larger files are optimized for lookup queries while larger delta files use columnar-based storage which is optimized for table scans. Napa also maintains a queryable timestamp to denote the time up to which data is available to be queried, which relates to the idea of a low watermark from the Millwheel paper. Napa will automatically adjust its configurations to ensure that the client's data freshness requirements are met. Napa exploits parallelism to improve its query latency. Input queries are partitioned into thousands of sub-queries and executed in parallel in different machines in the data center. Napa also includes optimizations that are used by systems like Hive such as pushing down filters and selects to limit the amount of data that has to be processed upfront. Similar to Tensorflow and MapReduce, Napa has strategies in place to reduce tail latency caused by stragglers, including hedging and expected progress reports.

Microsoft Monaeyball also relates the topics discussed throughout the semester. Like Napa, Moneyball also performs automatic scaling to meet client needs. Napa attempts to expand the use of serverless compute to clients who have time-sensitive applications. These clients traditionally would be more likely to provisioned compute despite it being more computationally expensive, but Moneyball employs a proactive resume which improves query latency. Proactive resumes and pauses also ensure that resources do not remain idle and unused for extended periods of time, and instead are paused using predictions from machine learning algorithms. This allows resource costs to decrease with limited impact on query performance. The unused resources are redirected to other clients' databases which increases throughput for Azure's serverless databases.