Wyatt Blair

EMT-678-WS: Big Data Technologies

10/11/2024

**Class 4 - Gilbert - MIT CAP theorem Brewer**

The CAP theorem describes a fundamental limitation of distributed systems: “In a network subject to communication failures, it is impossible for any web service to implement an atomic read/write shared memory that guarantees a response to every request”. In essence, there is a tradeoff between consistency (the system always returns the correct answer) and availability (the system always responds).

The tolerable amount of consistency deficiency or availability deficiency will be application dependent. For example, the paper points to Google’s *Chubby Lock* *Service* as a system which guarantees consistency at the detriment of availability. The reliability and timeliness of the network across which Google hosts such a service are safe bets and therefore partitions and network anomalies are rare. This allows engineers to create a “best effort availability” system which delivers consistent data as often as possible—perfect for a simple file system like *Chubby Lock*. Alternatively, an engineer could create a system which guarantees availability at the detriment of consistency. The paper uses web caching as an example of such a system. A user can quickly and reliably pull content from nearby servers, but there is not a strong guarantee that the data is up to date (violating consistency) since it can take a while for a web-page’s changes to propagate. A third provided by the paper is that of airline reservation software. In this circumstance, different states of airplane capacity demand different levels of consistency and availability; as such, these systems use a dynamic amount of both. By defining the desired amount of consistency, the rest of the system can scale for availability dynamically using the TACT toolkit. Finally, the paper proposes several system architecture strategies to handle macro demands for dynamic consistency and availability. Among these strategies are: data partitioning, operation partitioning, functional partitioning, user partitioning, and hierarchical partitioning.