Name: Charitha Kammari
Ucid: ck369

CS 643, Cloud Computing – Homework 2

1. (3 points) What is the typical application structure in Windows Azure? What type of communication is used to exchange data between application components and why?

Answer: In Windows Azure, the foundational structure for applications is referred to as a service. Users are required to provide definition and configuration details, including at least one role. Roles represent code with an entry point that operates within its own dedicated virtual machine. Microsoft primarily offers two roles: Web roles, typically accessible from the Internet, and Worker roles, responsible for executing arbitrary code and often handling tasks assigned by Web roles. Microsoft has also introduced VM roles, where a virtual hard disk is uploaded.

Communication between Web and Worker roles is facilitated through the fabric, involving agents in both roles, and can incorporate queues for either asynchronous or synchronous interactions.

2. (3 points) What is the most important consideration when assigning map tasks to workers in MapReduce? Why? How does the Master/JobTracker deal with it?

Answer: When assigning map tasks to workers in MapReduce, the critical factor is ensuring the proximity of data to the worker, enhancing task efficiency as workers read input directly from local disks. After writing intermediate key/value pairs to local disks, these are divided into regions corresponding to the number of reduce tasks, with their locations relayed to the Master.

The Master/Job Tracker subsequently allocates each reduce task to an accessible worker. The worker reads intermediate key/value pairs from map workers and applies the user's reduce operation, generating output stored in the Google File System (GFS).

3. (4 points) Assume this simple Dryad topology: A -> B. If task B crashes, will both tasks have to be re-executed? Justify your answer.

Answer: Different channels in use introduce various possibilities.

If the channel is a file, only task B needs re-execution, as the file processed by task A remains unaffected by B's crash.

In the case of a TCP pipe, both tasks A and B must be re-executed because the vertices at both ends need to be scheduled concurrently, requiring the rebuilding of the TCP link and data exchange through the new link.

If the channel is a shared-memory FIFO, both tasks A and B need re-execution since both endpoint vertices must run within the same process, and the crash of task B would lead to the entire process crashing.