NWEN303 – Assignment 5 – Cakes

Task 1 – Explain given code:

OpenAkka:

The OpenAkka class creates a new Akka system that then starts listening for work to do over the network. Actors and messages get serialized then sent to the listening OpenAkka process where they are deserialized and executed.

AkkaConfig:

The AkkaConfig class contains utility methods to assist with various things including network setup. The class helps find which IP address to use for OpenAkka. The class also provides a simple way to create an ActorSystem using the newSystem method which configures Akka to use TCP connections to communicate with other instances of ActorSystem on different computers/processes.

Actors:

Alice and Bob both produce as much wheat and sugar as they can continuously. They start producing their produce when they receive a message containing the actor to send the produce to, then they repeat this process by sending that same actor to themselves again. In this case, that actor is Charles.

Charles takes wheat and sugar from Alice and Bob, and then when he has both ingredients, he makes a cake. He will continue to make cakes as long as he gets ingredients, and then send the cakes to Tim.

When Tim is created, he gets sent a GiftRequest, he stores the sender for later. When Tim receives a cake, he decrements a counter indicating how hungry he is. When Tim is no longer hungry, he will send a Gift back to the original sender of the GiftRequest.

Cakes:

The Cakes class contains the entry point for the program (not the OpenAkka program). When the program starts, the computeGift method is called. A new ActorSystem is created with IP addresses of OpenAkka systems that are used to run the actors. The actors are initialized here and are sent messages to begin production of their produce, and Tim is sent a GiftRequest using the Akka ask pattern.

Request classes:

Initially the program had one request class called GiftRequest but later GiveOne and MakeOne were added. These classes are sent between actors to make a request for something. The GiftRequest class was used to ask Tim to make and give a gift back to the sender, GiveOne was used to ask a producer to give one item of produce back to the sender, and MakeOne was used to ask a producer to make one item of produce.

Task 3 – Describe performance improvement:

The solution with one bob took around 70s to get the gift, and the solution with 4 bobs took around 35s to get the gift. To measure the speed, I saved the system time when the operation started and then subtracted it from the finish time to get the total time for the gift.

Here are the timing results:

One bob: Took: 69061ms to get gift.

4 Bobs: Took: 34258ms to get gift.

The solution with 4 bobs uses round robin load balancing to ensure the work is distributed evenly between the bobs. This would not be the best way to do it if one actor was on a faster computer since it would finish its work and be waiting for other actors to complete their work before it can do more. A better way might be to have some sort of free/busy state where work is sent to the first free actor.