Oliver McLane & Jason Miner

November 28, 2021

Nathan Williams

Operating Systems

Dining Philosophers

Diagram

Description automatically generated

Our program created 5 philosophers around table which are each handed a left and right stick that are share by their neighbors. The sticks are created in the main methods and are synchronized throughout the run method in the Philosopher class to make sure only one person had a stick at a time. When both sticks are grabs by a philosopher will switch states and begin to eat and after a random time based on the state it will switch its state and thinking for a certain time until the begins to grab stick. After 60 seconds the program will end.

While making the program we ran into several road blockers that were hard to solve. If you look over our code, you’ll notice that the stick has states that are unused, these were used for testing purposes. We figured that you want to store a stick for each philosopher a left and a right. These would be handed into through the constructor, stored in a list and then synchronized across the philosophers. The first roadblock we ran into was the list sizes were all weird, as actually creating the list and then using them throughout the main method. We would create the philosopher list, but you need the sticks to be passed in, which meant the stick list needed to be created first, which confused things because we assumed we would use the length of the philosophers list. After this we ran into the issue of deadlocks because everyone could grab their right stick all together and thus no sticks were left on the table, creating a circular wait. To fix this we simply made the last created philosopher pick up the left stick instead of the right, breaking the circular wait.