ALLIANCE UNIVERSITY

Private University established in Karnataka State by Act No.34 of year 2010
Recognized by the University Grants Commission (UGC), New Delhi

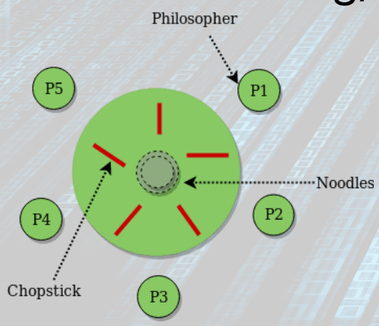
OPERATING SYSTEM AND PARALLEL PROGRAMMING

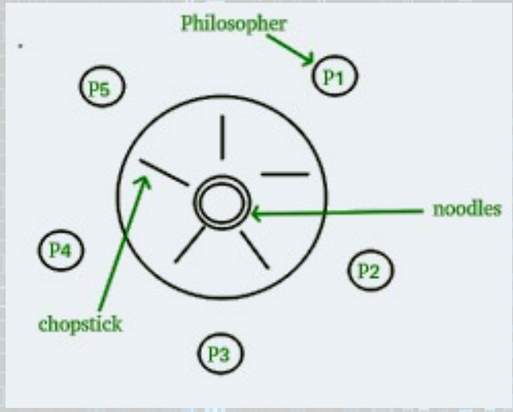
TOPIC: THE DINING PHILOSOPHERS PROBLEM

PRESENTED BY: PAVAN V,CHANDANA MN

PROBLEM STATEMENT?

- The Dining Philosopher Problem states that N philosophers seated around a circular table with one chopstick between each pair of philosophers.
- There is one chopstick between each philosopher. A philosopher may eat if he can pick up the two chopsticks adjacent to him.
- One chopstick may be picked up by any one of its adjacent followers but not both.
- Each philosopher can only alternately think and eat.
- A philosopher can only eat their spaghetti when they have both a left and right fork.
- Thus two sticks will only be available when their two nearest neighbors are thinking, not eating.





The problem is to design a algorithm such that no philosopher will starve; i.e., each can f continue to alternate between eating and thinking, assuming that no philosopher can know when others may want to eat or think.

SOLUTION

- It may lead to dead lock if a all the philosopher pick the fork together or if the philosophers adjacent to each other pick the stick at the same time.
- A solution of the Dining Philosophers Problem is to use a semaphore to represent a chopstick. A chopstick can be picked up by executing a wait operation on the semaphore and released by executing a signal semaphore.

The structure of the chopstick is shown below –

sem_t chopstick [5];

CODE SNIPPET

```c
int main()
{
    int i,a[5];
    pthread_t tid[5];
    sem_init(&room,0,5);

    for(i=0;i<5;i++)
        sem_init(&chopstick[i],0,1);

    for(i=0;i<5;i++)
    {
        a[i]=i;
        pthread_create(&tid[i],NULL,philosopher,(void *)&a[i]);
    }
    for(i=0;i<5;i++)
        pthread_join(tid[i],NULL);
}
```

OUTPUT

```
Philosopher 4 is hungry
Philosopher 4 is eating
Philosopher 3 is hungry
Philosopher 2 is hungry
Philosopher 1 is hungry
Philosopher 0 is hungry
Philosopher 4 has finished eating
total time taken by the philosopher 4 is 2 msecs
Philosopher 3 is eating
Philosopher 3 has finished eating
total time taken by the philosopher 3 is 10 msecs
Philosopher 2 is eating
Philosopher 2 has finished eating
total time taken by the philosopher 2 is 9 msecs
Philosopher 1 is eating
Philosopher 1 has finished eating
total time taken by the philosopher 1 is 16 msecs
Philosopher 0 is eating
Philosopher 0 has finished eating
total time taken by the philosopher 0 is 52 msecs
```