哲学家就餐问题

1. 问题

在 wikipedia 上,哲学家就餐问题的描述如下:

Five silent philosophers sit at a table around a bowl of spaghetti.
A fork is placed between each pair of adjacent philosophers.
(An alternative problem formulation uses rice and chopsticks
instead of spaghetti and forks.)
Each philosopher must alternately think and eat. However, a philosopher
can only eat spaghetti when he has both left and right forks. Each fork
can be held by only one philosopher and so a philosopher can use the
fork only if it's not being used by another philosopher. After he finishes
eating, he needs to put down both forks so they become available to others.
A philosopher can grab the fork on his right or the one on his left as they
become available, but can't start eating before getting both of them.
Eating is not limited by the amount of spaghetti left: assume an infinite
supply.
The problem is how to design a discipline of behavior (a concurrent algorithm)
such that each philosopher won't starve, i.e. can forever continue to alternate
between eating and thinking, assuming that any philosopher cannot know when
others may want to eat or think.

2. 算法设计

基本思路：五个哲学家分别代表五个线程，有thinking、waiting、eating这五种状态，然后各个线程不断调用-(void)changePhilosopherStatus:(philosopher *)phi 方法来改变哲学家状态，从而实现题目要求。

1)线程安全

由于五个哲学家分别代表五个线程，而筷子则相当于线程间共同访问的资源。因此，为了实现线程间的通信，则要保证线程安全，所以选择使用临界区。（obj-c的NSLock类可以构造出一个临界区）

2)防止死锁

为了防止线程进入死锁（即无限等待），则设定条件，当某个线程进入临界区时，只有在左右两个筷子都可用的时候才会选择吃饭。

3)防止饿死

为了防止饿死，则对于每个处于waiting状态的哲学家在尝试拿起筷子的时候，会先去检测该哲学家左右两个哲学家的状态，假如左右两人的等待时间比自己长（濒临饿死的时候），则不拿起筷子吃饭，否则，拿起筷子吃饭。

3. 实验心得

通过这次实验，对临界区、死锁和饿死有了进一步的了解，不单单是理论上的理解，还有在实际编程时如何运用,同时也返现了多线程编程的困难性。