临界区问题 Lamport 面包房算法（N 进程）

Lamport 面包房算法

- n 个进程的临界区问题

  - 每当进程申请进入临界区，它被分配一个数
  - 拥有最小数的进程，被选中进入其临界区
  - 如果进程 P_i 和进程 P_j 分配了相同的数，并且 $i < j$ ，那么，进程 P_i 首先进入临界区。进程编号 i 、 j 不会重复。
  - 数的分配器以递增的顺序产生 1,2,3,3,3,3,4,5...

面包房算法（续）

- 定义操作符号 ‘ <’
  - $(a,b) < (c,d)$ ， if $a < c$ ， or if $a = c$ and $b < d$
  - 定义操作函数 max()
  - $\max(a_0,\ldots, a_{n-1})$ 是 $a_0,\ldots, a_{n-1}$ 中的一个数 k ，使得 $k \geq a_i$ ， for $i = 0, \ldots, n - 1$
- 定义共享数据

  boolean choosing[n];

  int number[n];

  初始值分别是 **false** 和 **0**

面包房算法（续）

```
do {
    choosing[i] = true;
    number[i] = max(number[0], number[1], …, number [n – 1])+1;
    choosing[i] = false;
    for (j = 0; j < n; j++) {
      while (choosing[j]) ;
      while ((number[j] != 0)&&((number[j], j) < (number[i],
    i))) ;
    }
      critical section
    number[i] = 0;
      remainder section
} while (1);
```

讨论

- 产生的数，为什么会重复
  - **number[i] = max()+1**
  - 1,2,3,3,3,3,4,5...
  - choosing 数组的必要性
  - 假设没有它，会怎么样?
  - 看一个 Special Case“ 面包房算法 choosing 的必要性 .doc”


End