**偏序定义：**

定义→关系需要满足一下三个条件：

1. 如果事件a、b在一个进程中，a发生于b之前，那么a→b；
2. 如果a是发送一条消息的事件，而b是接收这条消息的事件，那么a→b；
3. 假如a→b成立，且b→c成立，那么a→c成立。如果两个事件a、b不满足a→b或者b→a，那么认为a、b两个事件是并发的。

Clock Condition: For any events a, b: if a---> b then C(a) < C(b).

满足以下两个条件，那么就满足Clock Condition：

1. 如果a、b是同一个进程中的事件，且a发生在b之前，那么C<a> < C<b>
2. 如果a、b是不同进程中的事件，且a是发送一条消息的事件，而b是接收这条消息的事件，那么C<a> < C<b>

Implementation Rule (to satisfy the Clock Condition):

IR1：任何一个进程Pi在两个成功事件之间递增Ci

IR2：

(a) 如果事件a表示Pi进程发送消息m，那么m包含一个时间戳Tm，Tm=Ci<a>。

(b)进程Pj收到消息m，Pj需要将Cj设置为等于或大于当前值，且大于Tm的值。

**全序定义：**

定义一个关系"=>"：对于Pi中的事件a和Pj中的事件b，如果(i)C<a> < C<b>或(ii)C<a> = C<b>，且Pi < Pj, 那么a=>b。

define a relation => as follows:

if a is an event in process Pi and b is an event in process Pj, then a => b if and only if either **(i)** Ci<a> < Cj<b> or **(ii)** Ci<a> = Cj<b> and Pi < Pj

一个由多个进程组成，且贡献一个资源的系统, 一个时刻只能有一个线程使用资源，所以线程间需要同步来避免冲突。寻找一个满足如下三个条件的算法来将资源分配给线程：(假设初始状态资源只会被分配给一个进程。)

1. 资源在分配给其他进程前，需要获取了资源的进程进行释放资源的操作；

2. 资源的请求必须按照请求的发生顺序进行分配；

3. 如果每个被授予资源的进程最终都释放了资源，那么每个请求最终都会被授予资源。

**算法**由以下5个**规则**定义：（假设每个规则定义的行为为一个独立的事件）

1. 为了申请资源，Pi发送资源申请请求Tm:Pi给所有其他的进程，并将消息放入自己的请求队列。Tm表示消息的时间。
2. 当Pj收到Tm:pi的请求，将其放入请求队列并发送一个带有时间戳的ACK给Pi。
3. 释放资源时，Pi将Tm:Pi从请求队列中移除，并发送一个带有时间戳的Pi释放资源的消息给所有的其他进程。
4. 当Pj接收到Pi释放资源的消息时，它将Tm:Pi请求资源的消息从请求队列中移除。
5. 当以下两个条件被满足时Pi获得资源：(a) 按=>的顺序，Tm:Pi的消息在请求队列的最前面；(b) Pi从其他每个进程至少收到了一条时间戳大于Tm的消息