分布式算法

EX1. 对于s的每一次执行的每一个配置都成立的断言p，则p是不是就是不变式？

解：不是。证明如下：我们假设有一个断言P它不是不变式，但是它可以由不变式Q导出。我们知道对于不变式Q，Q=>P，则P在S的每一次执行的每一个配置中都成立。这样P就符合了以上的条件，但是Ｐ不是不变式，得证。

EX2. 分析在同步和异步模型下，convergecast算法的时间复杂性

1. 在同步模型下，在每个容许执行中，高度为d的生成树的根节点会在d轮收到所有节点的消息。证明如下：

当d=1时，根节点会在第一轮收到它的所有孩子节点的信息，所以结论成立。

假设t<=d-1时，上述定理成立。

则第d轮时，由上述假设我们可知，在第d-1轮中，高度为d-1的子树收到它的所有节点的消息。高度为d-1的子树的根节点是高度为d的根节点的孩子节点，因此它们所收集的信息只需要一轮就可以传送到高度为d的树的根节点，也就是说高度为d的生成树的根节点会在d轮收到所有节点的消息。

综述所述上述定理成立。

1. 在异步模型下，在每个容许执行中，高度为d的生成树的根节点会在至多d时间内收到所有节点的消息。

当d=1时，树由根节点和它的孩子节点组成，由异步模型易知，根节点至多只需要1个单位时间就可以收集到所有节点的信息，所以上述定理成立。

假设t<=d-1时，上述定理成立。

当树的高度为d时，由上述假设可知，当树的高度为d-1时，它收集到所有子节点的消息的时间至多为d-1个单位时间，而能构造高度为d-1的树的节点都是高度为d的根节点的孩子节点。由异步模型易知，从孩子节点到根节点至多只需要1个单位时间，所以从根节点到所有节点至多只需要d时间就可以收集到所有节点的消息。

EX3. 证明Pr可达当且仅当它曾设置过自己的parent变量。

证明：充分性：因为Pr是可达的，所以它收到了消息M，所以它执行了算法2.2的第五行 upon receiving M from neighbor Pj，因此它就会执行第七行将parent变量进行赋值。必要性：因为Pr曾经设置过自己的parent变量，所以它执行了第七行代码，要执行第七行代码必须要执行第五行代码，也就是说它必须要收到消息M,而消息M是从Pr发出的，所以是Pr可达的。

Ex4. 证明Alg2.3构造一棵以Pr为根的DFS树

要证明一棵树是DFS树，只需要证明三点：连通性，无环，一个节点的孩子节点永远先于它的兄弟节点加入到树中。

1. 连通性

如果不连通，那么必有一个节点是Pr不可达的。我们假设有两个节点Pi,Pj，其中Pi和Pj相连，并且Pj是Pr可达的，Pi是Pr不可达的。由Ex3的结论可知，Pj是Pr可达当且仅当它曾设置过自己的parent变量，所以Pi的parent变量是nil。因为Pj是Pr可达的，所以Pj收到了消息M，有Alg2.3算法可知，Pj会向Pi发送M消息，根据算法的第9行可知，Alg2.3会将Pi的parent变量设置为j,这与假设矛盾，所以该算法构造出的树是连通的。

1. 无环

假设在树中存在环P1P2P3.....PnP1,其中Pi是Pj的父节点（i<j）。由Alg2.3可知，如果Pi是Pj的父节点，那么Pi必定会在Pj之前接收到M消息。因为行成了环，那么也就是说P1会在P1之前接收到M消息，这显然矛盾，所以树中不存在环。

1. 一个节点的孩子节点永远先于它的兄弟节点加入到树中

我们假设有这样的三个节点P1,P2,P3,，其中P2，P3是P1的孩子节点，并且P2和P3节点以及他们所有子节点都未加入到树中。根据算法Alg2.3,我们假设P1现将消息M发送给P2,那么P1要想给P3发消息M，那么必须等到P1收到来自于P2的消息<parent>消息，从会向P2发送M消息。而P1要想收到来自于P2的<parent>消息，P2必须要收到来自于它所有孩子节点的<parent>消息。所以P2的孩子节点会比P2的兄弟节点P3率先加入到树中，即一个节点的孩子节点永远先于它的兄弟节点加入到树中。

综述所述，命题得证。

EX5. 证明Alg2.3的时间复杂性为O(m)

不管是同步模型还是异步模型中，每个节点在其邻接边上至多发送一次M，每个节点至多生成一个msg作为对每个邻接边上收到的M的相应。因此Alg2.3至多发送4m个消息，即算法的消息复杂度是O(m)

1. 在同步模型中，每个节点必须要等到收到M消息才会被唤醒，并且每一轮只会发送一个消息，也就是说每一轮只有一个节点在发送消息，所以时间的复杂度和消息的复杂度是一样的，为O(m)
2. 在异步模型中，一个时刻至多有一个消息在传输，所以时间的复杂性和消息的复杂性相同，都是O(m)

EX6. 修改Alg2.3获得一新算法，使构造DFS树的时间复杂性为O(n),并证明

新算法：每个节点在确定了自己的父节点之后，就向自己的邻节点中除了双亲节点外广播一个消息Z，说明自己已经有父节点了，那么相应的邻节点就将该节点从自己的邻接节点表中删去。

算法复杂度分析：根据以上的算法可知，每个节点只需要三个消息（消息M，消息Z，消息<parent>）,就完成了自己的工作，所以总的消息复杂度是O(n)。根据Ex5的证明可知，Alg2.3算法的时间复杂度和消息复杂度是一样的，修改Alg2.3后的算法并没有改变这一性质，所以该新算法的时间复杂度是O(n)。

EX7. 证明同步环中不存在匿名的，一致性的领导者选举算法。

证明：假设在同步环上有一个匿名的，一致性的领导者选举算法A。那么在同步环上算法A的容许执行里，对于每一轮k,所有处理器的状态在第k轮结束时是相同的。

以上的假设论断证明如下：当k=0时，因为算法A是匿名的，一致性算法，所以所有的处理器具有相同的初始状态，论断成立。

我们假设该论断对k-1轮也成立。那么在第k-1轮结束后所有的处理器都具有相同的状态，所以每个处理器都会想左边发现Ml消息，向右边发送Mr消息。所以在第k轮中每个处理器都会受到来自于左边处理器的Mr消息和来自右边处理器的Ml消息。因为算法A是匿名的一致性算法，所以所有的处理器具有相同的局部算法，所以处理Mr和Ml消息后得到的结果是一致的。也就是说在第k轮，所有处理器的状态都是相同的。

因为在每一轮中每个处理器的状态都是相同的，所以如果我们选择了某一个处理器作为leader,那么根据领导者选举算法的原理也必须选择其他所有处理器作为leader，这样就违背了领导者选举算法只有一个leader的定义，产生矛盾。所以同步环中不存在匿名的，一致性的领导者选举算法。

EX8. 证明异步环系统中不存在匿名的领导者选举算法。

假设在异步环系统中存在匿名的领导者选举算法，那么说明每个处理的初始状态是相同的，状态机是相同的，收到的消息序列是相同的，只是收到消息的时间不同，也就是说所有的处理器最终都将达到相同的状态。一个消息在相邻节点的传送时间最多是一个单位时间。假设某一个处理器宣布自己成为leader，那么必将在有限的时间内所有其他的处理器也都宣布自己成为了leader，这与领导者选举算法只能有一个leader的定义产生矛盾，所以在异步环中不存在匿名的领导者选举算法。

EX9. 若将环Rrevn划分为长度为j(j是2的方幂)的连续片段，则所有这些片段是次序等价的

证明：对一个整数P(0≤P≤n−1),可以表示为：

其中m=lg n,则有rev(P)=。

设P、Q在同一个片段上，P1、Q1在同一片段上，且设这两个片段时相邻的，由模运算的加法可得：P1=P+ l；Q1=Q + l。l表示片段的长度，l=2k。

又因为：

且P、Q在同一个片段上，有

|P-Q|<l=2k

所以存在r(0≤r≤k),满足 ar ≠ br。否则，|P−Q|≥l。这与P、Q在同一个片段上矛盾。

设s=min⁡{r},则根据rev(P),rev(Q)的表示方法可得：

sign(rev(P)-rev(Q))=sign (as-bs)

而

显然，P与P1的前k位相同，Q与Q1的前k位相同。由0≤s≤k得

sign(rev(P1)-rev(Q1))=sign (as-bs)

这两个相邻片段是序等价的，根据等价的传递关系，可得所有的片段都是次序等价