分布式算法作业

2.1 分析在同步和异步模型下，convergecast 算法的时间复杂性。

解：

（1） 在同步模型中

最坏情况下，算法执行的每一轮中只有一个 msg 传递，而此时生成树汇聚最大值的算法最多执行 n-1 轮（即生成树中除了末端节点每一个节点只有一个子节点），也就是说同步情况下时间复杂度为 O(n-1)

（2） 在异步模中

在异步模型的汇集算法的每个容许执行中，树中每个距离 p_r 为 t 的处理器至多在时刻 t 接收消息 M，因此对于每个节点而言，它到它所有子节点中 t 最大的路径决定了它本身时间花费。因此在最坏情况下，仍应该是同步模型下的最坏情况，即生成树中除了末端节点每一个节点只有一个子节点，此时时间复杂度仍为 O(n-1)

2.2 G 里一结点从 pr 可达当且仅当它曾设置过自己的 parent 变量。

证明：

（1） 证：一结点从 pr 可达，则它曾设置过自己的 parent 变量。

因为图 G 是由 parent 和 children 确定的静态图，任一节点在收到 M 后才会加入到图中。即可达节点收到过 M，执行了算法 2.2 的第五行。由于是容许执行的，所以第 7 行（parent:=j）也会执行。

（2） 证：一节点设置过自己的 parent 变量，则其从 pr 可达。

若算法 2.2 的第 7 行执行过了，因为是容许执行，则必然有第 5 行也执行过了。即节点收到过 M。而 M 又是从 pr 发出的，所以该节点是从 pr 可达的。

2.3 证明 Alg2.3 构造一棵以 Pr 为根的 DFS 树。

证明：

（1） 算法 2.3 构造的图 G 必然是连通的。否则，设 G 存在邻居节点 Pj 和 Pi。Pj 从 Pr 可达，但 Pi 从 Pr 是不可达的。

则：

1） Pi 的 parent 为空；

2） Pi 不为 Pj 的 child.

因为：

G 里一结点从 pr 可达当且仅当它曾设置过自己的 parent 变量。

所以：

1） Pj 的 parent 必然设置过了；

2） Pi 的 parent 为 nill;

3） Pi 属于 Pj 的 unexplored 集合。

而算法的第 11 和 14 行决定了 Pj 会向 Pi 发送 M,使得 Pi 的 parent 成为 Pj,Pi 成为 Pj 的 child。

这与假设的结果矛盾。故 Pi 必然也是从 Pr 可达的。

(2) 算法 2.3 构造的图 G 必然是无环的。否则设 G 中有一个环，P1,P2,…,Pi,P1。令 P1 是该环中最早接收到 M 的节点。则 Pi 是从 P1 可达的，且 P1 的 parent 是 Pi，P1 是 Pi 的 child。

而 Pi 在收到 M 后，向 P1 发送 M。因为 P1 的 parent 已经不为空，所以 P1 收到来自 Pi 的 M 时，根据第 16 行代码，P1 会向 Pi 放回一个<reject>信息，不会将 Pi 设为 parent。而 Pi 未收到 P1 返回的<parent>信息，也不会将 P1 设为 child。与前面的出的结果矛盾。

故 G 是无环的。

(3) 图 G 是一棵 DFS 树。只需证明在有子结点与兄弟结点未访问时，子结点总是先加入树中。

设有节点 P1，P2 和 P3。P2 和 P3 是 P1 的直接相邻节点。P1 在第 12~14 行中先选择向 P2 发送 M，则 P1 当且仅当 P2 向其返回一个<parent>（第 17 行，第 22 行）时才有可能向 P3 发送 M。

而 P2 仅在其向所有的相邻节点发送过 M 后才会向 P1 返回<parent>(第 19~21 行)。所以 P2 的子节点是永远先于 P3 加入树中的，即 G 是 DFS 树。

2.4 证明 Alg2.3 的时间复杂性为 O(m)。

证明：

(1)：同步模型：每一轮中，根据算法，有且只有一个消息(M or Parent or Reject)在传输，从算法的第 6 、14、16、20、25 行发送消息的语句中可以发现：消息只发往一个处理器结点，除根结点外，所有的处理器都是收到消息后才被激活，所以，不存在多个处理器在同一轮发送消息的情况，所以时间复杂度与消息复杂度一致。

(2) 异步模型：在一个时刻内至多有一个消息在传输，因此，时间复杂度也与消息复杂度一致。消息复杂度：对任一边，可能传输的消息最多有 4 个，即 2 个 M ，2 个相应 M 的消息（Parent or Reject），所以消息复杂度为 O(m)

综上，该算法的时间复杂度为 O(m)

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

解：

两种考虑方式：

(1) 在每个处理器中维护一本地变量，同时添加一消息类型，在处理器 Pi 转发 M 时，发送消息 N 通知其余的为访问过的邻居，这样其邻居在转发 M 时便不会向 Pi 转发。

(2) 在消息 M 和<parent>中维护一发送数组，记录已经转发过 M 的处理器名称。

两种方式都是避免向已转发过 M 的处理器发送消息 M，这样 DFS 树外的边不再耗时，时间。

复杂度也降为 O(n)

3.1 证明同步环系统中不存在匿名的、一致性的领导者选举算法。

证明：

在匿名系统中，每个处理器在系统中具有相同的状态机。由 Lemma3.1 可知，设算法 A 是使环上某个处理器为 leader 的算法。因为环是同步的，且只有一种初始配置。在每轮里，各处理器均发出同样的 message，所以在各轮里各个处理器接收到相同的 message，则状态改变也相同。所以所有处理要么同为 leader，要么同时不为 leader。
故同步环系统中匿名的、一致性的领导者选举算法的算法是不存在的。

3.2 证明异步环系统中不存在匿名的领导者选举算法。

证明：
每个处理器的初始状态和状态机相同，除了接收消息的时间可能不同外，接收到的消息序列也相同。
所以最终处理器的状态也是一致的。由于处理器处理一条消息至多需要 1 单位时间，若某时刻某个处理器宣布自己是 leader，则在有限时间内，其它处理器也会宣布自己是 leader。
故异步环系统中匿名的领导者选举算法是不存在的。

3.9 若将环 R^{rev} 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($0\leq P \leq n-1$),可以表示为：

$$P = \sum_{i=1}^{m} a_i . 2^{i-1}$$

其中 m=lg n
则有$\mathrm{rev}(P) = \sum_{i=1}^{m} a_i . 2^{m-1}$。
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：

$$P1=P+l$$

$$Q1=Q+l$$

式中 l 表示片段的长度，$l=2^k$ 。
又

$$P = \sum_{i=1}^{m} a_i . 2^{i-1}$$

$$Q = \sum_{i=1}^{m} b_i . 2^{i-1}$$

且 P、Q 在同一个片段上，有

$$|P-Q|<l=2^k$$

所以存在 r($0\leq r \leq k$),满足$a_r \neq b_r$。否则，$|P-Q| \geq l$。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：

$$\mathrm{sign}(\mathrm{rev}(P) - \mathrm{rev}(Q)) = \mathrm{sign}\,(a_s - b_s)$$

而

$$\mathrm{P1} = \mathrm{P} + \mathrm{l} = \sum_{i=1}^{m} a_i . 2^{i-1} + 2^k$$

$$\mathrm{Q1} = \mathrm{Q} + \mathrm{l} = \sum_{i=1}^{m} b_i . 2^{i-1} + 2^k$$

显然，P 与 P1 的前 k 位相同，Q 与 Q1 的前 k 位相同。由$0 \leq \mathrm{s} \leq \mathrm{k}$得

$$\mathrm{sign}(\mathrm{rev}(P1) - \mathrm{rev}(Q1)) = \mathrm{sign}\ (a_s - b_s)$$

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的。