算法分析与设计第二次作业

姓名：郭昊　学号：SA14011008

分布式算法

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

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，也就是说同步情况下的时间复杂度为 $O(n-1)$ 。

异步模型中：

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 $O(n-1)$ 。

EX 2.2 证明在引理 2.6 中，一个处理器在图 G 中是从 p_r 可达的，当且仅当它的 parent 变量曾被赋过值。

$(\leftarrow)$：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。

$(\rightarrow)$：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 p_r 发出的，所以该节点是从 p_r 可达的。

EX 2.3 证明 Alg2.3 构造一棵以 p_r 为根的 DFS 树。

证明：连通性：假设构造的图 G 存在邻居节点 p_j 和 p_i 。 p_j 从 p_r 可达，但 p_i 从 p_r 是不可达的。则 p_i 的parent为nil或者 p_i 不为 p_j 的child。由于 G 里一结点从 p_r 可达当且仅当它曾设置过自己的parent变量。所以：

1） p_j 的parent必然设置过了；

2） p_i 的parent为nil或者 p_i 属于 p_j 的unexplored集合。

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

这与假设的结果矛盾。故 p_i 必然也是从 p_r 可达的。

无环： 假设 G 中存在一个环， $p_1, p_2, ..., p_i$ 。令 p_1 是该环中最早接收到 M 的节点。则 p_i 是从 p_1 可达的，且 p_1 的parent是 p_i ， p_1 是 p_i 的child。而 p_i 在收到 M 后，向 p_1 发送 M 。因为 p_1 的parent已经不为空，所以 p_1 收到来自 p_i 的 M 时，根据第16行代码， p_1 会向 p_i 放回一个<reject>信息，不会将 p_i 设为parent。而 p_i 未收到 p_1 返回的<parent>信息，也不会将 p_1 设为child。与前面的出的结果矛盾。故 G 是无环的。

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 p_1， p_2 和 p_3 。 p_2 和 p_3 是 p_1 的直接相邻节点。 p_1 在第12~14行中先选择向 p_2 发送M，则 p_1 当且仅当 p_2 向其返回一个<parent>（第17行，第22行）时才有可能向 p_3 发送 M 。而 p_2 仅在其向所有的相邻节点发送过 M 后才会向 p_1 返回<parent>。所以 p_2 的子节点是永远先于 p_3 加入树中的，即 G 是DFS树。

EX 2.4 证明 Alg2.3 的时间复杂性为 $O(m)$ 。

证明：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。
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 $O(m)$ 。

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

在每个处理器中维护一个本地变量，同时添加一个消息类型，在处理器 p_i 转发 M 时，发送消息 N 通知其余的未访问过的邻居，这样其邻居在转发 M 时便不会向 p_i 转发。

在消息 M 和<parent>中维护一个发送数组，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 $O(n)$ 。

EX 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。

EX 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。

EX 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 $P(0 \le P \le n-1)$, 可以表示为：

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

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

设 P 、Q 在同一个片段上， P_l 、Q_l 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： $P_1 = P + l$ ； $Q_1 = Q + l$ 。l 表示片段的长度， $l = 2^k$ 。

又因为： $Q=\sum_{i=1}^{m} b_i \cdot 2^{i-1}$

且 P 、Q 在同一个片段上，有 $\left|P-Q\right| < l = 2^k$

所以存在 $r\left(0 \le r \le k\right)$, 满足 $a_r \ne b_r$ 。否则，$|P-Q| \ge l$ 。这与 P 、Q 在同一个片段上矛盾。

设 $s = min\left\{r\right\}$, 则根据 $rev\left(P\right), rev\left(Q\right)$ 的表示方法可得：

$$sign\Big(rev\big(P\big)-rev\big(Q\big)\Big)=sign\ \left(a_s-b_s\right)$$

而 $P_1 = P + l = \sum_{i=1}^{m} a_i \cdot 2^{i-1} + 2^k \qquad Q_1 = Q + l = \sum_{i=1}^{m} b_i \cdot 2^{i-1} + 2^k$

显然， P 与 P_l 的前 k 位相同，Q 与 Q_l 的前 k 位相同。由 $0 \le s \le k$ 得

$$sign\Big(rev\big(P_l\big)-rev\big(Q_l\big)\Big)=sign\ \left(a_s-b_s\right)$$

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。