中国科学技术大学
University of Science and Technology of China

2022秋算法设计与分析
复习与习题课

2022.10.24

考试形式

- **考试形式**

  闭卷考试

- **大致比例**

  分布式算法（40%）+概率算法（40%）+近似算法（20%）

- **题型分布**

  选择题 （10*3）+ 计算与简答题（4*10）+ 算法设计题（2*15）

- **分数计算**

  作业 20% +期末考试 80%（暂定）

- **具体时间** 待薛老师课程结束后商定。

分布式算法部分作业内容与提交方式

- 内容：

- 安全性（safety）板书：Ex
- 分布式算法PPT ch33: P9 Ex
- 分布式算法PPT ch33: P30 EX2.1, Ex 2.2, EX2.3, EX2.4, EX2.5
- 分布式算法PPT ch34 P9 EX3.1, Ex3.2
- 分布式算法PPT ch35 P39 Ex3.9
- 课程邮箱：ustcsealg@163.com

- 提交方式：

- 将所有习题写在一个文档中，写清题号、题目、解答与分析过程，导出为pdf格式，提交课程邮箱，命名为：学号_姓名_分布式算法作业.pdf （注意命名为下划线） 至电子邮箱ustcsealg@163.com。
- 提交时间截止日期：2022年11月1日晚上24时

分布式算法

复习要点提示

分布式系统的概念

Def：一个分布式系统是一个能彼此通信的单个计算装置的集合（计算单元：硬——处理器；软——进程）

包括：紧耦合系统----如共享内存多处理机

松散系统-----cow、Internet

- 分布式系统无处不在，其作用是：
  - ①共享资源
  - ②改善性能：并行地解决问题
  - ③改善可用性：提高可靠性，以防某些成分发生故障

分布式系统面临的困难：
异质性、异步性、局部性

分布式系统的概念

§2.1.1 系统

- **容许的执行**：指无限的执行。
  因为轮的结构，所以
  每个处理器执行无限数目的计算步，
  每个被发送的msg最终被传递

- **同步与异步系统的区别**
  在一个无错的同步系统中，一个算法的执行只取决于初始配置
  但在一个异步系统中，对于相同的初始配置及无错假定，因为处理器步骤间隔及消息延迟均不确定，故同一算法可能有不同的执行。

分布式系统生成树构造

§2.3 构造生成树

- **msg复杂性**

  因为每个结点在任一信道上发送M不会多于1次，所以每个信道上M至多被发送两次(使用该信道的每个处理器各1次)。

  在最坏情况下：M除第1次接收的那些信道外，所有其他信道上M被传送2次。因此，有可能要发送2m-(n-1)个msgs。这里m是系统中信道总数，它可能多达n(n-1)/2。

- **时间复杂性：**O(D)　D—网直径

2.构造生成树

对于flooding稍事修改即可得到求生成树的方法。

分布式系统生成树构造

§2.3 构造生成树

- 为何无环？(无环)

  假设有一环，$p_{i1},\cdots p_{ik}p_{i1}$，若$p_i$是$p_j$的孩子，则$p_i$在$p_j$第1次收到M之后第1次收到M。因每个处理器在该环上是下一处理器的双亲，这就意味着p_{i1}在p_{i1}第1次接收M之前第1次接收M。矛盾！

- 复杂性

  显然，此方法与淹没算法相比，增加了msg复杂性，但只是一个常数因子。在异步通信模型里，易看到在时刻t，消息M到达所有与p_r距离小于等于t的结点。因此有：

Th2.7 对于具有m条边和直径D的网络，给定一特殊结点，存在一个msg复杂性为O(m)，时间复杂性为O(D)的异步算法找到该网络的一棵生成树。

环选举问题：破对称问题

一个算法解决了leader选举问题需满足(根据形式化模型)：

① 终止状态被划分为两类：选中和未选中状态。一旦一个处理器进入选中(或未选中)状态，则该处理器上的转换函数将只会将其变为相同的状态；

② 在每个容许执行里，只有一个处理器进入选中状态，其余处理器进入非选中(non-selected)状态。

本章只讨论系统的拓扑结构是环的情况。

什么是匿名环

- **匿名算法**：若环中处理器没有唯一的标识符，则环选举算法是匿名的。**更形式化的描述**：每个处理器在系统中具有相同的状态机，在这种算法里，msg接收者只能根据信道标号来区别。
- **（一致性的）uniform算法**：若算法不知道处理器数目，则算法称之为uniform，因为该算法对每个n值看上去是相同的。
- **non-uniform算法**：算法已知处理器数目n
- **形式化描述**：在一个匿名、一致性的算法中，所有处理器只有一个状态机；在一个匿名、非一致性的算法中，对每个n值（处理器数目）都有单个状态机，但对不同规模有不同状态机，也就是说n可以在代码中显式表达。

同步匿名环的不可能性

环选举-异步环选举

§3.3 异步环

- 在非匿名算法中，均匀（一致性）和非均匀（非一致性）的概念稍有不同
  - ① 均匀算法：每个标识符id，均有一个唯一的状态机，但与环大小n无关。而在匿名算法中，均匀则指所有处理器只有同一个状态。（不管环的规模如何，只要处理器分配了对应其标识符的唯一状态机，算法就是正确的。）
  - ② 非均匀算法：每个n和每个id均对应一个状态机，而在匿名非均匀算法中，每个n值对应一个状态机。（对每一个n和给定规模n的任意一个环，当算法中每个处理器具有对应其标识符的环规模的状态机时，算法是正确的。）

下面将讨论msg复杂性：$O(n^2) \rightarrow O(n\log n) \rightarrow \Omega(n\log n)$ 下界

§3.3.1 一个$O(n^2)$算法

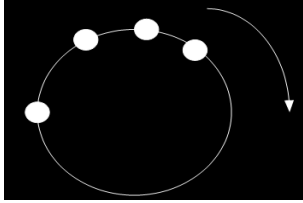
Le Lann、Chang和Roberts给出，LCR算法

- 基本思想
  - ① 每个处理器P_i发送一个msg(自己的标识符)到左邻居，然后等其右邻居的msg
  - ② 当它接收一个msg时，检验收到的id_j，若$id_j > id_i$，则P_i转发id_j给左邻，否则没收id_j(不转发)。

③ 若某处理器收到一个含有自己标识符的msg，则它宣布自己是leader，并发送一个终止msg给左邻，然后终止。

④ 当一处理器收到一个终止msg时，向左邻转发此消息，然后作为non-leader终止。

因为算法不依赖于n，故它是均匀的。



i—表示id 单向

Msg复杂性：O（n^2）

最朴素的方法，

每次都向邻居转发比自己大的信息

否则吞并，直到获得最大的。

->

O(nlogn)->向2个方向各发送2^l个邻居

—>

证明下界O(nlogn)

选ID最大的

环选举-异步环选举

§3.3.2 一个O(*n*lg*n*)算法

- **基本思想**

  算法按阶段执行，在第l阶段一个处理器试图成为其2^l-邻接的临时leader。只有那些在l-th阶段成为临时领袖的处理器才能继续进行到(l+1)th阶段。因此，l越大，剩下的处理器越少。直至最后一个阶段，整个环上只有一个处理器被选为leader。

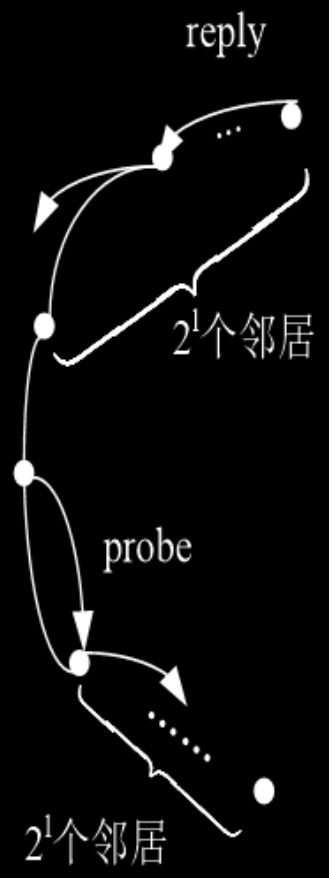
- **具体实现**

  ① **phase0:** 每个结点发送1个probe消息(其中包括自己的id)给两个1-邻居，若接收此msg的邻居的id大于消息中的id，则没收此msg；否则接收者发回一个reply msg。若一个结点从它的两个邻居收到回答msg reply，则该结点成为phase0里它的1-邻居的临时leader，此结点可继续进行phase1。

环选举-异步环选举

② **phase l**：在l-1阶段中成为临时leader的处理器P_i发送带有自己id的probe消息至它的2^l邻居。若此msg中的id小于左右两个方向上的$2*2^l$个处理器中任一处理器的id，则此msg被没收。若probe消息到达最后一个邻居而未被没收，则最后一个处理器发送reply消息给P_i，若P_i从两个方向均接收到reply消息，则它称为该阶段中2^l邻居的临时leader，继续进入下一阶段。

③ **终止**：接收到自己的probe消息的结点终止算法而成为leader，并发送一个终止msg到环上。



环选举-异步环选举

④ **控制probe msg的转发和应答**

probe消息中有三个域：<prob, id, *l*, hop>

id-标识符

l-阶段数

hop-跳步计数器：初值为0，结点转发probe消息时加1.

若一结点收到的probe消息时，hop值为2^l，则它是2^l邻居中最后一个处理器。若此时msg未被没收也不能向前转发，而应该是向后发回reply消息。

同步环

研究同步环上leader选举问题的上、下界。

- 上界： 提出两个msg复杂性为O(n)的算法，显然，这样的算法的msg复杂性是最优的。但运行时间并非只与环大小n有关，还与算法使用的非普通的id相关。(与id值相关)
- 下界： 讨论

1）只基于标识符之间比较的算法

2）时间受限（即若干轮内终止，轮数只依赖于环大小,不依赖于id值）的算法

当算法受限于上述两个条件时，至少需要$\Omega(n \lg n)$个msgs.

§3.4.2 有限制算法的下界Ω(nlgn)

1. 基于比较的算法的概念和定义

为了得到下界，假定所有处理器在同一轮开始执行

一个环是由结点表按顺时针方向指定的，结点表开始于最小标识符。

在同步模型里，算法的容许执行完全由初始配置定义，这是因为msg延迟及节点步骤的相对次序均无选择的可能。系统的初始配置完全由环决定，即由节点标识符列表按上述规则来决定。当算法的选择可以从上下文判断清楚时，则将由环R确定的容许执行表示为exec(R).

- 匹配：若环R_1上的结点p_i和R_2上的p_j在各自的环里具有相同的位置，则称p_i和p_j是匹配的，即：匹配的结点在各自环上距其最小id的结点的距离相同。

Def3.5 设R1和R2是任意两个大小为n的序等价的环，若每对匹配的结点在exec(R1)和exec(R2)的第1至t轮里有相似的行为，则同步算法A对于环大小为n的标识符集合S是基于t-比较的。

Def 3.3 在一个环R上的一个执行里，某轮r称为是active的，若该执行的第r轮里，某一个结点发送一个msg。当R是从上下文已知时，用r_k表示第k个active轮。

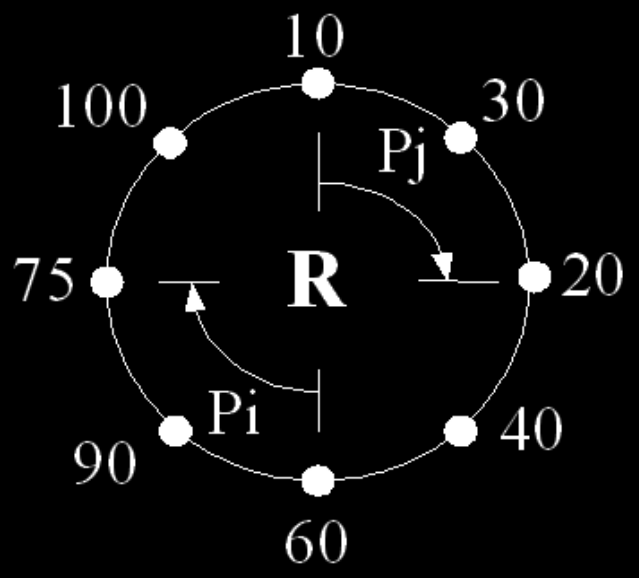
Note： 一旦环R确定，整个容许执行就确定(因为系统同步)

由于一个基于比较的算法在等价环上的行为相似，因此：

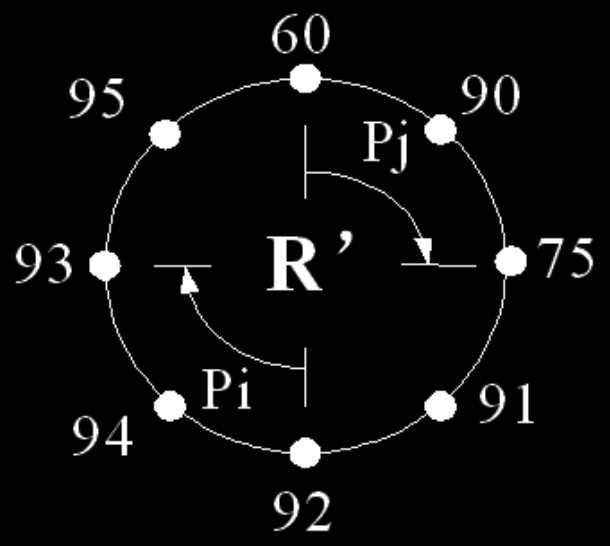
对于序等价的环R_1和R_2，一轮在exec(R_1)里是主动的当且仅当它在exec(R_2)里也是主动的。

因为消息中的信息在k个轮里只能在环上通过k个结点，所以一个结点在k轮之后的状态只依赖于它的k-邻居。

然而一个更强的性质是：一个结点在第k个主动轮之后的状态只依赖于它的k-邻居。这实际上告诉我们：信息只有在主动轮里才能获得。这一点在下面的引理中给出形式证明。



R



R’

① P_i的1-邻居60，90，75

② R’中id唯一

③ R次序：

10,30,20,40,60,90,75,100

P_j与10距离为1，

P_j的1-邻居60，90，75

R’次序：

60,90,75,91,92,94,93,95

P_j与60距离为1

同步环选举算法

§3.4.2 有限制算法的下界Ω(nlgn)

- 同步的下界不可能从异步的下界导出

  因为上节中的算法表明：同步模型中的附加假定是必不可少的。
- 同步的下界对于非均匀和均匀算法均成立，但异步的下界只对均匀算法成立。
- 但是从同步导出的异步结果是正确的，并且提供了一个非均匀算法的异步下界。

异步通信模型中领导者选举问题
所需的消息数下界为Ω(nlgn)且
算法不依赖于比较的或者限时的

分布式系统计算模型的复杂性：
系统中无全局时钟、由并发执行部件构成
对策：因果状态、一致状态、全局状态
（因果序问题）

- 分布式系统计算模型的复杂性
  - 系统由并发执行部件构成
  - 系统中无全局时钟
  - 必须捕捉系统部件可能的失效
- 对策
  - 因果关系（Causality）
  - 一致状态（Consistent states）
  - 全局状态

向量时间戳

§4.2.1 Lamport时间戳

- 系统有序性的重要性

  若分布式系统中存在全局时钟，则系统中的事件均可安排为全序。例如，可以更公平地分配系统资源。

- 全序对事件的影响和由H关系确定的偏序对事件的影响是一致的

- 如何通过H关系确定的偏序关系来建立一个“一致”的全序关系？

  - 在$<_H$的DAG上拓扑排序
  - On the fly：Lamport提出了动态即时地建立全序算法

16

§4.2.2 向量时间戳

- Lamport时戳缺点

  若$e_1 <_H e_2$，则 $e_1.TS < e_2.TS$；反之不然。

  例如：1.3<2.1，但是$e_6 < e_4$不成立

  原因：并发事件之间的次序是任意的

  不能通过事件的时戳判定两事件之间是否是因果相关

- 判定事件间因果关系的重要性

  例子：违反因果关系检测

  在一个分布式对象系统中，为了负载平衡，对象是可移动的，对象在处理器之间迁移是为了获得所需的调用的进程或资源。如下图：

21

- 并发事件
- 因果相关事件

因果关系

分布式系统为何缺乏全局的系统状态？

1、非即时通信

2、相对性影响

3、中断

§4.2 因果关系

分布式系统为何缺乏全局的系统状态？

1.非即时通信

A和B同时向对方喊话

他们都认为是自己先喊话

C听到两人是同时喊话

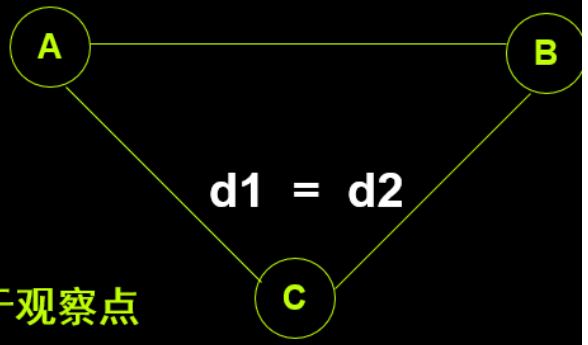
结论：系统的全局状态依赖于观察点

原因：

传播延迟

网络资源的竞争

丢失msg重发



8

e1.VT=(5,4,1,3)　e2.VT=(3,6,4,2)　e3.VT=(0,0,1,3)

e1、e2 是并发的（有大有小），e3 在因果序上先于 e1（e1 所有值都大于 e3）

作业相关问题

- **EX 1. convergecast （汇集算法）时间复杂度分析**
  - 时间复杂度要带符号 “O()”；
  - 题目的答案 O(d)，d表示树的深度；
  - 易写成O(n)，n 是节点的个数（X）。
- **EX 2. Pr可达←→其parent变量赋值**
  - 这里要证的是当且仅当，需要从充分性和必要性两个方面来证明。

1. 分析在同步和异步模型下汇集算法的复杂性。

解：与广播算法分析时间复杂性的步骤一致，一两句的说明不是分析。

<1> 同步模型

引理：在汇集算法的每个容许执行里，树中每个高为 **t** 子树根结点在第 **t** 轮里收到所有孩子的**msg**。

归纳证明。。。

定理：当生成树高为 **d** 时，存在一个时间复杂度为**O(d)**的同步汇集算法。

<2> 异步模型

引理：在汇集算法的每个容许的执行里，树中每个高为 **t** 的子树根结点在时刻 **t** 收到所有孩子的**msg**。

归纳证明。。。

定理：当生成树高为 **d** 时，存在一个时间复杂度为**O(d)**的异步汇集算法。

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

解：

引理2.6：在异步模型的每个容许执行中，算法2.2构造一棵以Pr为根的生成树。

两个方向证明题目：依据是算法2.2和题目条件（异步模型的每个容许执行中），不是空口讨论。方法不一，原则是有理有据，逻辑清楚。

<1> => pr可达，（因为图G是由parent与children确定的静止图）收到m才会加入图中，所以可达结点收到过m，执行了Alg2.2第5行。由于是容许执行，第7行，即parent：=j也会执行。也就是被赋值。

<2> <=当第7行执行过，由于是容许执行，第5行也执行过，即收到过m，而m是由pr发出的，所以可达。

➢EX 3. 证明 Alg2.3 →以 pr 为根的DFS树

□应该先证明构造了一棵树（从连通性和无环性说明）；然后再证明该生成树是 DFS 树。

可以证明：在有子结点与兄弟结点未访问时，子结点总是先加入树中。根据Alg2.3证明这一点。

➢EX 4. 证明Alg 2.3的时间复杂度为O(m)

□ 注意这里的证明需要分模型（同步和异步）讨论

Ex2.4 证明 Alg2.3 的时间复杂性为 O(m)。

解：

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)

综上分析，该算法的时间复杂度为 O(m)。

5. 修改Alg2.3，使其时间复杂度为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通知其余的未访问过的邻居，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)。

6. 证明同步环上不存在匿名的、一致性的Leader选举算法。

解：由Lemma3.1可得。

假设R是大小为n>1的环（非均匀），A是其上的一个匿名算法，它选中某处理器为leader。因为环是同步的且只有一种初始配置，故在R上A只有唯一的合法执行。

Lemma3.1： 在环R上算法A的容许执行里，对于每轮k，所有处理器的状态在第k轮结束时是相同的。

- Note：每个处理器同时宣布自己是Leader！

7. 证明异步环系统中不存在匿名的Leader选举算法。

解：每个处理器的初始状态相同，状态机相同，接收的消息序列也相同（只有接收消息的时间可能不同），故最终处理器的状态一致。由于处理一条消息的至多需要1时间单位，若某时刻某个处理器宣布自己是Leader （接收到m条消息），则在有限时间内（m时间单位）其他处理器也会宣布自己是Leader。

- Note：每个处理器陆续宣布自己是Leader！

祝大家考试顺利！