# 99%容错共识指南

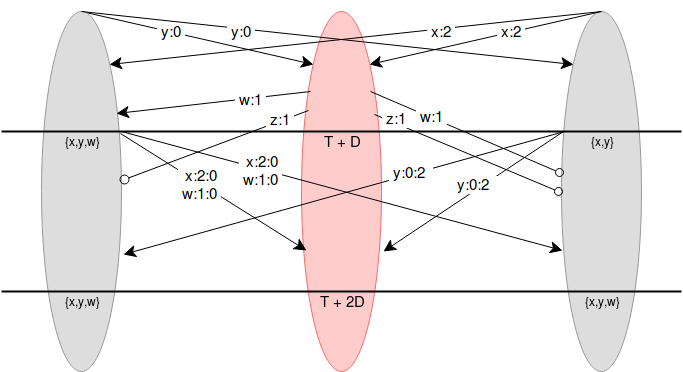
特别感谢Emin Sirer的审阅

我们很早就听到可以实现一个50%容错的共识同步网络，在这个网络中，从一个诚实节点广播的消息一定会在一个确切的时段内被其他的诚实节点接受（如果一个攻击者拥有超过50%的算力，他们可以实施“51%攻击”，有一个可以适用任何算法的模型）。我们也很早就听过，如果你想要放弃网络的同步性假设，并且有一个“在异步下是安全的”算法，最大的可实现容错下降到33% (PBFT, Casper FFG等都属于这一类)。但是你知道吗，如果你添加更多的假设(具体来说，你需要观察者等。比如，那些不积极参与共识但关心其输出的用户，也要积极关注共识，而不仅仅是在事后下载其输出)，你能把容错率一直提高到99%吗?

这其实早就为人所知;莱斯利·兰波特1982年发表的著名论文《拜占庭将军问题》对算法进行了描述。下面我将尝试以一种简化的形式描述和重新构造算法。

假设有N个共识参与节点，每个人都同意这些节点参与(这取决于上下文，它们可能是由受信任的群体选择的，或者，如果强烈需要去中心化，可以通过某种工作量证明或股权证明机制)。我们把这些节点分别标注为0 .... n - 1。假设有一个已知的网络延迟加上时钟差之和的边界D(例如D = 8秒)。每个节点都可以在T时刻发布一个值（恶意节点当然可以在T之前或之后提交值），所有节点等待(N-1) \* D秒，运行以下程序。定义x:i为“值x是由节点i签名的”，x : i : j为“值x由i签名，值x和i的签名一起被节点j签名”，等等。在第一阶段各个节点发出的提议都是以v:i的形式，在这些v:i中包含了提出者的签名。如果一个验证者i收到一些这样的消息v : i[1] : ... : i[k]，其中i[1] : ... : i[k]是一组给这个消息顺序地签名的节点（如果仅仅有值v，则k=0；而对于v:i，k=1），然后验证者检查(1)时间还没到T + k \* D，(2)他们收到一个有效的消息包含值v；如果这两个条件的检查都通过，那么他就发布 v : i[1] : ... : i[k] : i。

当T + (N-1) \* D时，节点停止监听。此时，可以保证诚实的节点都“有效地看到”同一组值。



节点1(红色)是恶意的，节点0和2(灰色)是诚实的。一开始，两个诚实节点分别提出了消息y和x，随后恶意节点提出消息w和z。消息w准时到达节点0而不是节点2，但是消息z不能准时到达任何节点，在T + D时刻，节点0和2重播了他们所看到的尚未在区块链上广播的所有值，而且添加上了他们的签名（消息x和w由节点0签名，消息y由节点2签名）。所有的诚实节点都看到了消息x，y，w。

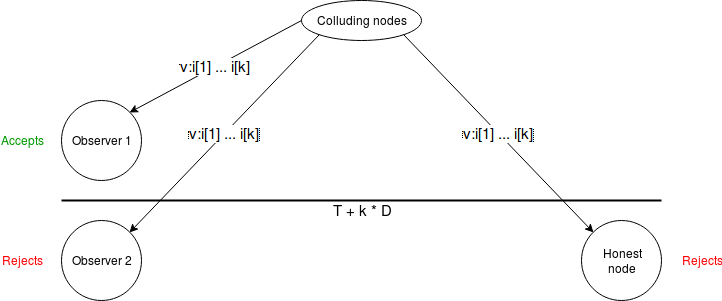
如果问题要求选择一个值，他们可以使用一些“选择”函数从他们看到的值中选择一个值（例如，他们取哈希值最低的那个）。节点可以同意这个值。

现在，让我们来探究一下为什么这是可行的。我们需要证明的是,如果一个诚实节点看到了特定值(有效),然后其他诚实节点也看到这个值(如果我们证明了这一点,那么我们知道所有诚实节点都看到一组相同的值,因此如果所有诚实节点都运行相同的选择功能,他们会选择相同的值)。假设任何诚实的节点都接收到他们认为有效的消息(即消息在T + k \* D时刻之前到来) 假设x是另一个诚实节点的索引，这样的话x要么是{i[1] ... i[k]}中的一个，要么不是。

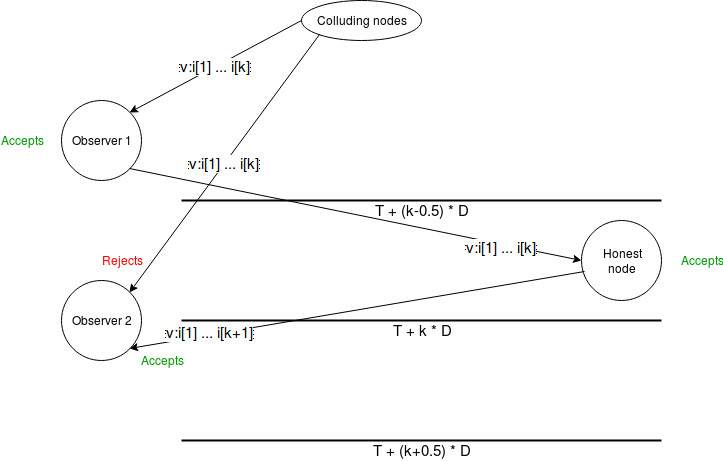
* 在第一种情况下(比如消息x =i[j]),我们知道诚实节点x已经广播了该消息,他们这样做是为了回应他们在时刻T + (j-1) \* D之前收到的消息，这个消息带有签名i[j-1]。所以他们广播这条消息，而且这条消息一定会在T + j \* D时刻之前被其他所有的诚实节点接收到。（注：这种情况是节点x在{i[1] ... i[k]}中）
* 在第二种情况下，由于诚实的节点在T + k \* D时间之前看到消息，因此它们将使用自己的签名广播消息，并保证每个人（包括x）都将在T + (k+1) \* D时间之前看到消息。（注：这种情况是节点x不在{i[1] ... i[k]}中）

注意到该算法运行的方式是在消息超时之际添加某个节点的签名作为一种“碰撞”，正式这种特性包装如果一个诚实节点收到了这个消息，他们就能确定其他人也准时收到了这个消息，在这里“准时”的定义，随着签名的增加带来的网络延迟增加，可以看做很短的时间内的准时。

在一个节点是诚实的情况下，我们能保证消极观察者（即关注结果的非共识参与节点）也可以看到结果吗，即使我们要求它们一直监视这个过程。按照以上的计划，有一个问题。假设一个恶意指挥者和一个包含k个恶意验证者的集合发出了消息v : i[1] : .... : i[k]，然后在T + k \* D时刻之前广播给一些“受害者”节点。这些“受害者”节点认为这个消息是准时的，但是当他们再次广播这个消息时，这个消息只能在T + k \* D时刻之后到达所有参与共识的节点，所以其他所有的诚实参与共识的节点会拒绝这个消息。



但我们可以堵住这个洞。我们要求边界值D达到两倍的网络延迟加上时钟差。我们让观察者的超时时间不同：有一个观察者在T + (k - 0.5) \* D时刻前收到了消息v : i[1] : .... : i[k]。现在，假设又有一个观察者看到这个消息并接受它。他们能在T + k \* D时刻之前广播到一个诚实节点，然后诚实节点给这个消息签名，这个带了签名的消息会在T + (k + 0.5) \* D时刻之前到达所有的观察者，这个消息的在带有第k+1个签名时过期。



## 改造到其他共识算法上

从理论上讲，上述算法可以作为一个独立的共识算法使用，甚至可以使用到一个POS区块链。达成共识的第N轮+1的验证者集本身可以在第N轮中共识决定（例如，每一轮协商一致也可以接受“存款”和“取款”交易，如果被接受并且正确签名，这些交易将增加或删除下一轮的验证者）。需要额外添加的主要成分是决定谁被允许出块的机制(例如，每一轮可以有一个指定的出块者)。还可以将其修改为可用的pow区块链，方法是允许参与共识的节点通过发布一个工作量证明的解决消息并用他们的公钥签名来实时“声明自己”。

然而，同步假设是非常强大的，因此我们希望在不需要超过33%或50%容错的情况下，能够在不需要同步也可以工作。有一种方法可以做到这一点。假设我们有一些其他的共识算法（例如，PBFT, Casper FFG，基于链的PoS），它的输出可以被偶尔在线的观察者看到（我们称之为阈值相关共识算法，与上面的算法不同，上面的算法我们称之为延迟相关共识算法）。假设依赖于阈值的共识算法连续运行，在一种模式中，它不断地“确定”新的块到一个链(即，每个最终确定的值都指向作为“父”块的一些先前确定的值;如果有一个指针序列A ->…-> B，我们称A为B的后代。

我们可以将延迟依赖算法改进到这个结构上，让始终在线的观察者在检查点上访问一种“强终结性”，容错率为95%（通过添加更多的验证器并要求过程花费更长时间，您可以任意地将其推到接近100%）。

每次当时间达到4096秒的倍数时，我们运行延时相关算法，选择512个随机节点参与算法。有效的建议是由与阈值相关的算法确定的任何有效且有价值的链。如果一个节点在时间T + k \* D (D = 8秒)之前看到某个带有k个签名的最终值，它会将链接收到它的已知链集合中，并添加自己的签名重新广播它；观察者使用T + (k - 0.5) \* D作为阈值。

最后使用的“选择”函数很简单:

* 如果最后确定的值不是上一轮中已经商定的最后确定值的后代，则将被忽略
* 最后确定的值如果无效将被忽略
* 要在两个有效的最终值之间进行选择，请选择具有较低哈希值的值

如果5%的验证器是诚实,只有约一千亿分之一的概率在随机选择出来的512个节点中没有一个是诚实的，所以只要网络延迟+时钟差异小于D / 2，上面的算法将正常工作，能够正确协调节点在一些单一的确定值,即使因为与阈值相关的算法的容错坏了多个相互冲突的最终值提出了。

如果符合阈值相关共识算法的容错性(通常为50%或67%诚实)，那么阈值相关共识算法要么不会确定任何新的检查点，要么会确定彼此兼容的新检查点(例如:一系列的检查点,每个前面的检查点作为父母), 因此，即使网络延迟超过D/2(甚至D)，并且由于参与延迟依赖算法的节点不同意他们接收的那个值，但它们接收的值仍然保证是同一个链的一部分，因此没有实际的分歧。在未来的某个回合中，一旦延迟恢复正常，依赖于延迟的共识将“同步”恢复。

如果与阈值相关和与延迟相关的共识算法的假设同时被打破(或在连续的回合中)，那么算法就会崩溃。例如，假设在第一轮中，与阈值相关的共识最终确定了Z -> Y -> X，而与延迟相关的共识在Y和X之间不一致，在下一轮中，与阈值相关的共识最终确定了X的后代W，它不是Y的后代;在延迟依赖共识中，同意Y的节点不接受W，而同意X的节点接受W。然而，这是不可避免的;在拜占庭容错理论中，不可能存在超过1/3的异步一致性的安全共识，这是一个众所周知的结果，就像不可能存在超过1/2的容错，甚至允许同步假设，但假设存在离线观察者则不一样。