论文题目：A Digital Fountain Approach to Reliable Distribution of BulkData

内容出处：JohnW.Byers; Michael Luby; Michael Mitzenmacher; Ashutosh Rege, A Digital Fountain Approach to Reliable Distribution of BulkData, ACM SIGCOMM Computer Communication Review, 28, 4, 56–67, Oct.1998

1.主题：

越来越多的应用需要可靠地向大量的自治客户端分发大量的数据，这就促使了新的组播和广播协议的设计。本文为这些应用描述了一个理想的、完全可扩展的协议，我们称之为数字喷泉。数字喷泉允许任意数量的异构客户端在选择时以最优的效率获取海量数据。而且，即使面对较高的丢包率，也不需要任何反馈渠道来保证可靠的交付。本文用一种新的纠删码-Tornad码来逼近数字喷泉，这种纠删码比标准纠删码快一个数量级。通过性能测试验证了该方法的可行性，并讨论了实验系统的设计、实现和性能。

2.研究现状：

Introduction：

对于计划通过互联网同时向数百万用户有效传播新软件的软件公司来说，一个自然的解决方案是多播或广播传输。这些传输必须完全可靠，网络开销低，并支持大量具有异构特性的接收器。这些应用程序需要的不仅仅是可靠的多播协议，因为用户希望在他们选择的时间访问数据，并且这些访问时间将与其他用户的访问时间重叠。虽然单播协议成功地使用接收者发起的丢失数据重传请求来提供可靠性，但众所周知，该解决方案的多播模拟是不可扩展的。

例如，对于一台将数据分发给数千个客户端的服务器。当客户端丢失数据包时，他们的重传请求可能会在称为反馈导入的过程中迅速使服务器堵塞而不堪重负。即使在服务器可以处理请求的情况下，重传的数据包也通常仅用于客户端的一小部分。在所研究的论文提出时，已经出现了通过使用诸如本地修复，轮询或使用层次结构之类的技术来解决这些限制的更复杂的解决方案。尽管这些解决方案似乎已经足够，但基于自适应重传的解决方案在陆地网络上是不可扩展且效率低下的，同时它们在卫星网络上也无法使用。在卫星网络上，反向信道通常具有较高的延迟和有限的容量。

即在服务器向数以千计（甚至更多）客户端同时发送数据的时候，如果发生丢包问题，则整个传输过程会变得效率低下。

relatedwork：

Nonnenmacher，Biersack和Towsley等人在研究中定义了一种可靠的多播混合方法，即通过重发冗余码字传输重传请求，并在实践中检验了这种方法的好处。他们的工作重点研究基于Reed-Solomon码的纠删码。这种解决方法的基本原理是：源数据以k个数据包序列的形式与附加的冗余数据包一起由发送方发送，并且冗余数据可用于恢复丢失的数据接收方的源数据。接收器一旦接收到足够数量的数据包，便可以重建原始源数据。这种编码的局限性在于：在数据很大的情况下，编码和解码时间会很慢。对此的解决方案有：将源数据分解为小的数据包块，并在这些块上进行编码，未从给定块接收到数据包的客户端接收方请求从该块重新发送。研究证明，当数据包丢失率较低时（通常考虑1％的丢失率），该方法可有效减少重传的次数。但是这种方法并不能够消除对重传，仍然需要不断重传，同时随着客户端接收方的增多和丢包率的增加，重传会更加频繁。

为了消除重传的需要并允许接收者异步访问数据，使用数据轮播或广播磁盘方法可以确保完全的可靠性。在数据轮播方法中，源通过所有数据包的传输反复进行循环。接收方可以随时加入流，然后进行侦听，直到接收到包含传输的所有不同数据包为止。显然，使用这种方法，接收器的接收开销可能会非常高。

而向轮播中添加冗余码字可以大大减少接收开销，这种解决方式主张使用Reed-Solomon码在传输块中添加固定数量的冗余。然后，源反复遍历这组块，依次围绕每个块传输一个数据或冗余数据包，直到所有数据包都用完为止，然后重复该过程。一旦接收器从每个块中接收到足够多的数据包，这种交错方法就可以使接收器重构源数据。在有损网络上使用此方法的局限性在于：接收方可能在等待需要重构的最后几个块中的最后一个数据包时，仍从已重构的块中接收许多不必要的数据包。

这种方法的优点是：不同的接收器可以使用相同的冗余数据从不同的丢失数据包中恢复。原则上，这种想法可以大大减少重传的次数，因为一次冗余数据的重传可以同时使许多接收者受益。将这种方法视为理想解决方案的较弱近似，并将其称为数字喷泉。与以前的方法相比，数字喷泉在概念上更简单，更有效，并且适用于更广泛的网络类别。

3.本文提出方法

**1.依据**

除了保持网络流量到最低限度，一个可扩展的协议分配软件使用的多播应该是：

1. 可靠性：确保完整的发送给所有接收者
2. 高效性：下载文件的时间和任何其他额外处理的时间应该降低到最小
3. 要求：客户端可以启动自行下载，客户端可以偶尔中止下载并且稍后要求继续下载
4. 容忍：各种各样的端到端数据包丢失率

所以丢失的包是至关重要的，而数字喷泉的关键特性是可以从总长度等于源数据的编码数据包的任何子集完整地重建源数据。本文构建了更好的数字喷泉近似值，作为执行可靠分配大型数据的协议的基础。

同时在不同的网络环境中(如Internet、卫星网络和带有移动代理的无线网络)，丢包特性、拥塞控制机制和端到端延迟方面有很大不同。论文致力于开发独立于这些特定于环境的解决方案，并在比较研究中总结了不同环境的丢包率。

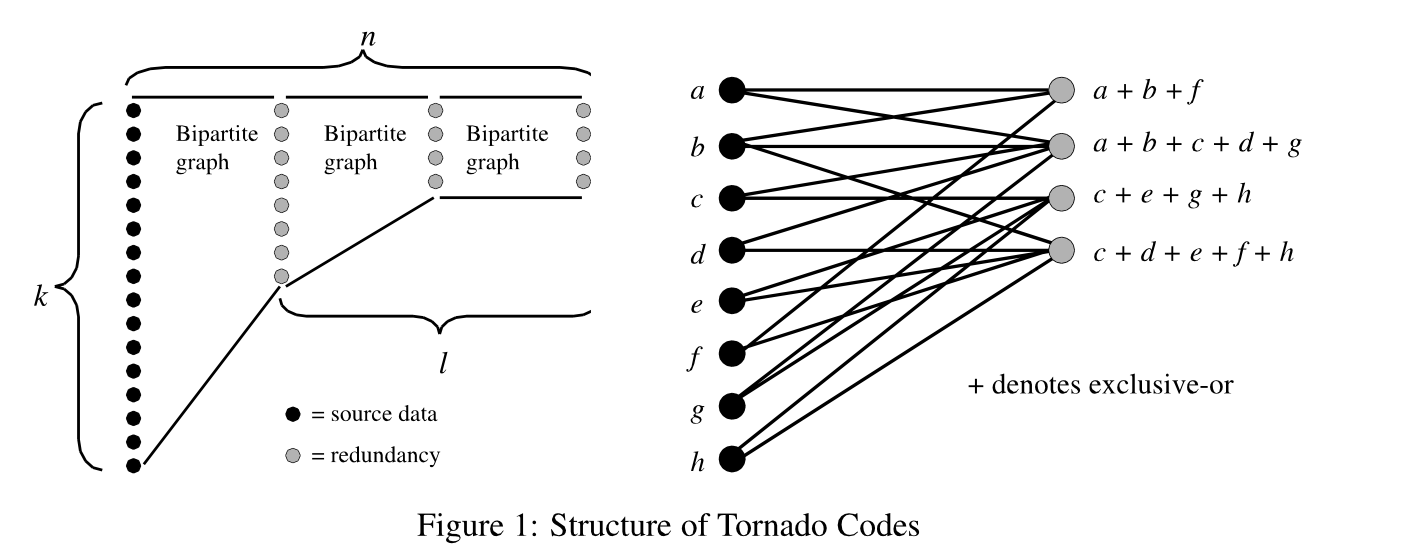
在理想的解决方案中，服务器发送出不同的数据包组成的流，称为编码数据包，它们构成了源数据的编码。每当会话中有任何客户端在监听时，服务器都会发送编码数据包。客户端接受来自通道的编码数据包，直到获得确切的k个数据包为止。在这种理想的解决方案中，无论客户端获得哪个k个编码数据包，都可以重建数据。因此，一旦接收到k个编码数据包，客户端就可以从该信道断开连接。我们假设在这种理想化的解决方案中，服务器只需要很少的处理就可以生成数据包的编码，而客户端只需很少的处理就可以从k个编码的数据包中恢复原始数据。我们将这种理想化解决方案中服务器产生的编码包流以数字喷泉的形式进行比喻。数字喷泉的性质类似于用于止渴的饮水机：饮用一杯水，而不管什么样的水滴充满玻璃杯，都可以为某人止渴。

构建数字喷泉实施数字喷泉的理想方法是直接使用纠删码，该纠删码机制接收由k个数据包组成的源数据，并产生足够多的编码数据包以满足用户需求。实际上，诸如Reed-Solomon码之类的标准纠删码具有理想的特性，即只要客户端接收到任何k个传输数据包，客户端的解码器就可以重建原始源数据。但是纠删码通常用于将由k个数据包组成的文件扩展为n个编码数据包，其中k和n均为输入参数。将比率n/k称为纠删码的拉伸因子。此有限的拉伸因子限制了纠删码可以近似于数字喷泉的程度，合理近似值（例如18、22、23、25）会将n设置为k的倍数，然后反复循环传输n个编码数据包。对于任何n的预定值，在足够高的58%丢包率下，客户端可能不会在一个周期内接收到n个数据包中的k个。因此在有损环境中，客户端在重建源数据之前可能会收到使用较少的重复传输，从而降低了信道效率。但在实践中，论文的实验结果表明，即使在非常高的丢包率下，这种效率低下的数据来源也不大。

一个更严重的限制是关于编码和解码操作的效率。即使对于中等的k和n值，标准Reed-Solomon纠删码的编码和解码处理时间也是极大的。

**2.方法核心与实验验证**

本论文提出了替代方案：通过使用更快的Tornado码来避免这种极大的编码、解码成本。使用龙卷风代码的主要缺点是，解码器需要略多于k的传输数据包才能重建源数据。论文对Tornado码的构建以及Tornado码与Reed-Solomon码的对比进行了深入的描述。



图表 1Tornado编码的结构

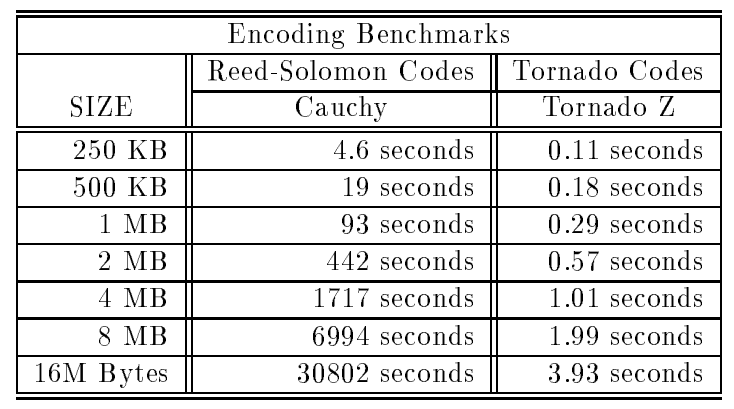
Reed-Solomon码的解码时间较长，是由于所使用的线性方程组致密。Tornado码使用稀疏的随机方程构建，即每个方程式的平均变量数很小。这种稀疏性允许实质上更有效的编码和解码。然而为更快的编码和解码付出的代价是：k个数据包不再足以重建源数据。取而代之的是需要略多于k个数据包。对于龙卷风代码，方程式的形式为Y3=X1⊕X4⊕X7，其中⊕是按位异或。龙卷风代码还使用以下形式的方程式：Y53=Y3⊕Y7⊕Y13，即冗余分组可以从其他冗余分组导出。编码时间取决于等式系统中的异或运算次数，Tornado码的解码过程使用两个基本操作。第一个操作是用出现的方程式中的值替换接收到的变量。第二个操作是一个简单的替换规则。可以应用替换规则来恢复出现在方程式中的任何缺失变量，在该方程式中，该变量是唯一的缺失变量。例如：再次考虑方程式Y3＝X1⊕X4⊕X7。假设已经接收到包含Y3的冗余数据包以及包含X1和X7的源数据包，但是尚未接收到X4。然后，我们可以再次使用异或运算，使用上述方程式求解X4。反复使用此替换规则，单个数据包的到达可能使我们能够重构几个其他数据包。然后，通常单次到达会产生替换“龙卷风”，从而允许恢复所有剩余的源数据包。因此称这种编码为龙卷风编码。

只要有足够的数据包到达，解码就可以停止，以便可以重建源数据。龙卷风代码仅使用异或运算，并避免了在对Reed-Solomon码进行解码时固有的场运算和矩阵求逆。用于解码的异或运算的总数最多是用于编码的数目，并且通常更少。使用龙卷风代码重建源数据，只需恢复略大于k的n个数据包。因此如果需要（1+f）k个编码数据包来重建源数据，则解码效率为1+f。

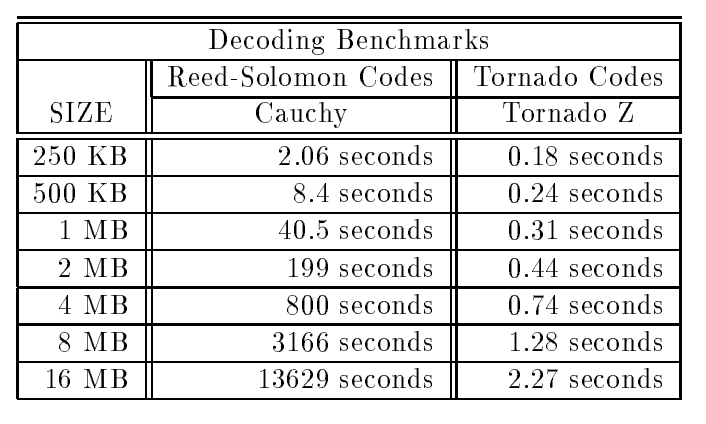
龙卷风编码相对于标准代码的优势在于，它们在解码效率的少量增加与编码和解码时间的显著减少之间进行了权衡。龙卷风代码的编码和解码时间比Reed-Solomon码快几个数量级。

**Tornado码与标准Reed-Solomon码的传输效率比较**

性能在实践中，k和£的值大约为数万的Tornado代码可以在短短几秒钟内进行编码和解码。论文将Tornado代码的效率与先前为网络应用建议的标准代码进行比较（如下表）。表中Cauchy纠删码是基于Cauchy矩阵的Reed-Solomon纠删码的标准实现。



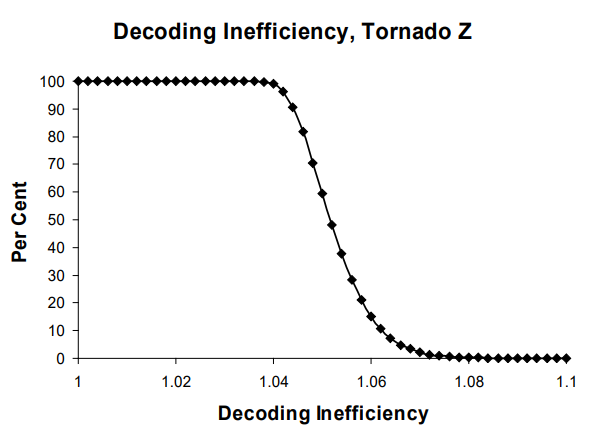
图表 2 编码次数的比较



图表 3 解码次数的比较

所有实验均在Sun167MHzUltraSPARC1上进行了基准测试，UltraSPARC1具有64MB运行Solaris2.5.1的RAM。所有运行的数据包长度均为P=1KB。对于所有运行，将包含k个数据包的文件编码为n=2k个数据包，即扩展因子为2。对于柯西码的解码，假设使用k/2个原始文件数据包和k/2个冗余数据包来恢复原始文件。当使用具有拉伸因子2的轮播编码时，该假设近似成立，因此接收到的数据包中大约有一半是冗余数据包。

TornadoZ的平均解码效率为1.054，因此平均为1.054。k/2个原始文件数据包和1.054。k/2个冗余数据包用于恢复原始文件。实验结果表明，即使对于较小的文件，Tornado码的编码和解码速度也比Reed-Solomon码快得多。在下图中，论文显示了针对解码效率低下的特定值，接收器无法重建源数据的试验百分比。例如，在每个节点代表一个数据包的情况下使用TornadoZ码，解码效率为1.064对应于接收17,024=1.064·16,000个数据包。超过90％的客户端可以在接收到这么多数据包之前重建源数据。



图表 4 解码效率

实施数字喷泉的主要障碍是标准Reed-Solomon代码在这些应用中的运行时间过高。因此，论文介绍了Tornado码，这种编码有极快的编码和解码算法。在实验中与实际使用的Reed-Solomon码相比，这些代码通常比数字喷泉具有更好的近似性。最后，论文描述了基于IP组播构建的基于Tornado代码的批量数据分发工作原型系统的设计和性能，并探讨了喷泉方法的其他研究方向。

**3.实验原型多播系统设计**

对于使用Tornado码实现可靠的分发协议，论文描述了一种用于将批量数据分发给可能异步访问数据的大量异构接收器的实验系统。该种系统是使用基于IP多播的协议为Internet设计的。开发此系统的目的是演示在实际系统中使用Tornado码的可行性，而不是创建用于部署的功能完备的多播协议。在实际的使用当中，不同的信道特性应当使用不同的拥塞控制和容忍接收器异质性的方法。

在多播协议的设计中，两个主要问题是分层多播的使用以及客户端对消息进行解码的方法。

**跨多个组播组分层**

对于跨多个组播组分层论文采用拥塞控制方法遵循其他提倡分层组播的研究者的提倡。此方法的主要思想是使源数据能够跨多个多播组传输数据，从而允许接收方订阅这些层的适当子集。当然，出于实际考虑，必须保证与给定源关联的多播组的数量最少。接收者的订阅级别基于诸如到源的瓶颈链路的宽度和网络拥塞之类的因素。提议的分层方案共有的基本思想是：

•服务器跨多个层传输数据，其中这些层通过传输速率的高低进行排序。

•这些层是累积的，因为订阅第i层的接收器还订阅了其下的所有层。我们说，当接收者订阅第0到第i层时，订阅者订阅了第i层。

•拥塞控制是通过使用同步点（SP）来实现的，同步点是在流中经过特别标记的数据包。接收者只能在SP之后立即尝试加入更高的层，并且仅跟踪来自最后SP的事件历史。流中发送SP的速率与带宽成反比：较低带宽的接收器有更多的机会上移到更高的级别。

•服务器将生成周期性的突发，而不是客户端进行显式的加入尝试，在此期间，数据包以每层正常速率的两倍发送。这具有创建类似于拥塞者在明确加入后将经历的网络拥塞状况的效果。接收者使用丢包事件来指示拥塞。因此，如果接收方在突发期间未发现任何数据包丢失，则它将在下一个SP处添加一个层。相反，每当分组丢失发生在同步点之前的脉冲串之外时，接收机就会降至较低的订阅级别。

该方法的吸引人的特征在于，接收器不需要向源提供拥塞控制反馈，并且接收器不需要协调加入尝试以防止对其他接收器的干扰。这些特征在不希望接收器到源和接收器间通信的数字喷泉情况下特别重要。

跨多个组播组调度分组传输如前所述，级别为i的接收器订阅了所有第0层到第i层。因此，重要的是仔细地计划在多个层之间的分组传输，以最小化客户端接收到的重复分组的数量。拉伸因子c限制了可以传输的不同数据包的数量，因此，特别是在存在高丢包率的情况下，也对客户端收到的重复数据的数量产生了很大的影响。当然，使用较大的拉伸因子可提供更大的灵活性，但会减慢解码时间并增加解码的空间要求。由于这些原因，论文中提议通常选择使用的拉伸因子为c=2，当然在Tornado码中使用更大的拉伸因子也是可行的。

**在客户端进行重构**

客户端负责观察SP并修改其订阅级别（作为拥塞保证）。客户端必须执行的另一项活动是源数据的重建。有两种方法可以实现客户端解码协议：

•增量方法：客户端在每个数据包到达后执行初步解码操作。这种方法导致一些冗余的计算：重建的源数据以后可能会完整无缺地到达。此外，在到达目的地后立即处理各个分组可能会有相当大的开销。

•第二种方法：根据统计观察结果，等待直到固定数量的数据包到达，很有可能可以从中重建源。如果此时无法完成解码，则可以单独或成组处理其他数据包。虽然增量方法的优点是可以使某些解码计算与数据包接收重叠，但经过论文实验发现增量方法在实践中更易于实现，解码速度损失很小。

4.本文提出方法的优势

在本论文中，对于在服务器向数以千计（甚至更多）客户端同时发送数据的时候，如果发生丢包问题，则整个传输过程会变得效率低下的问题，论文作者在已有的一些关于数字喷泉的研究的基础上设计了Tornado码，优化了数字喷泉协议，极大地增强了协议编码、解码的效率。

同时作者设计了基于Tornado码的数字喷泉原型数据分发系统，提出分层多播的使用以及客户端对消息进行解码的方法。在实验室环境下验证了这种协议的实用性与高性能，展望了数字喷泉与Tornado码在路由问题中的极大作用。

5.本文提出方法有何不足

Tornado编码解决方案的问题在于:如果拉伸因子c较小，则经常会收到重复的数据包，造成数据的冗余；如果拉伸因子c较大，则解码的空间和时间要求变得过高又会影响效率。未来的解决方向是适当地设置编码的参数。

同时，在查阅其他文献的过程中我发现，Tornado码采用级联二分图对分割后的数据块进行编码，虽然可以较快地编码，但经过编码后的数据存储到网络时，还是面临着数据被窃取、篡改的危险。

6.改进方案或者你的观点

在查阅文献的过程中我发现了一些研究者对基于Tornado的复制算法的改进，但是很少，这里列举一个方法：即通过数据指纹本地存储。

步骤１数据分割：将数据Ｄ分割为ｍ个大小为ｂ的数据块，即Ｄ＝［d1,d2,…,dm］

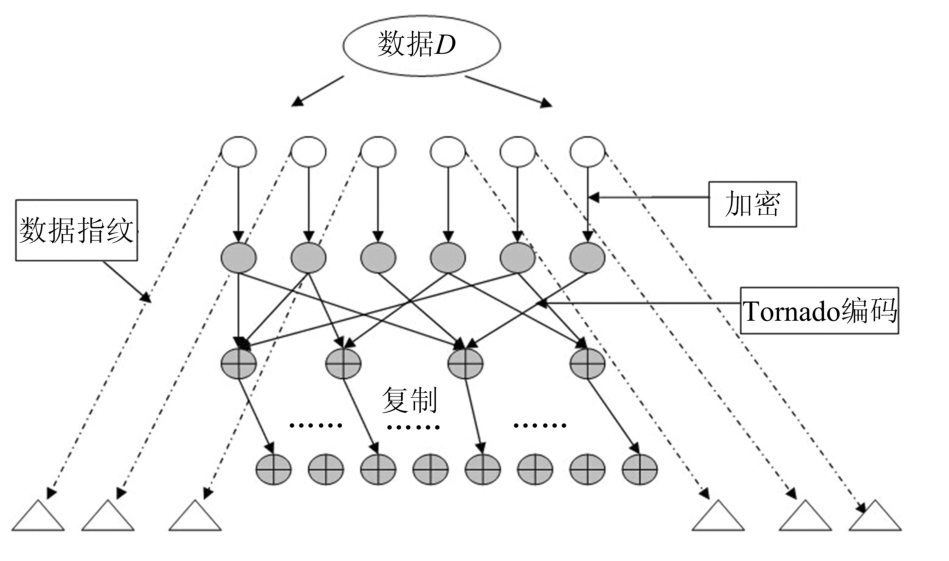
步骤２数字指纹：计算ｍ个数据块的数字指纹并存于本地

步骤３加密：对ｍ个数据块进行加密，产生ｍ个加密数据块

步骤４编码：对ｍ个加密数据块进行编码，产生p个校验块

步骤５冗余：把ｐ个校验块复制成kp个校验块

步骤６存储：将ｍ个加密数据块和kp个校验块分散存储在网络的多个结点上,如图5所示：



图表 5 算法改进数据存储

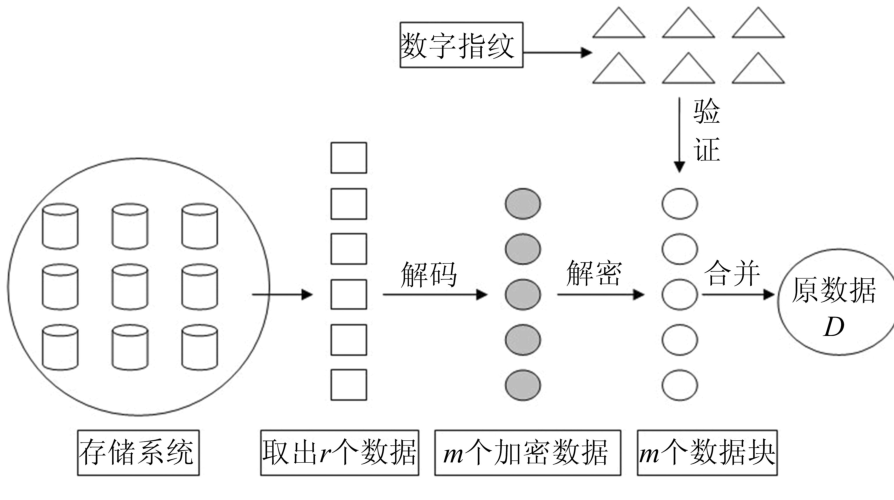
数据恢复算法为该算法的逆过程，具体步骤为：

步骤１解码：从当前可用的结点上获取r个数据块，对它们进行解码，得到原来的ｍ个加密数据块

步骤２解密：对ｍ个加密数据块进行解密，得到原ｍ个未加密的数据

步骤３验证：将解密出的数据计算数字指纹，与存于本地的数字指纹进行对比，若数字指纹相同，则数据没有被篡改

步骤４恢复数据：由解密得到的ｍ个数据块恢复数据D．如图6所示：



图表 6恢复数据

该方法主要改进了基于Tornado编码的算法的可用性，持久性，安全性，同时也解决了部分数据冗余的问题，有效解决了在网络存储中数据的安全性和鲁棒性问题，并且应用本算法在云端存储手机通信录时也取得了很好的效果。