Network Coding for Distributed Storage systems

Alexandros G. Dimakis, P. Brighten Godfrey
Martin J. Wainwright and Kannan Ramchandran

2011/6/3

提纲

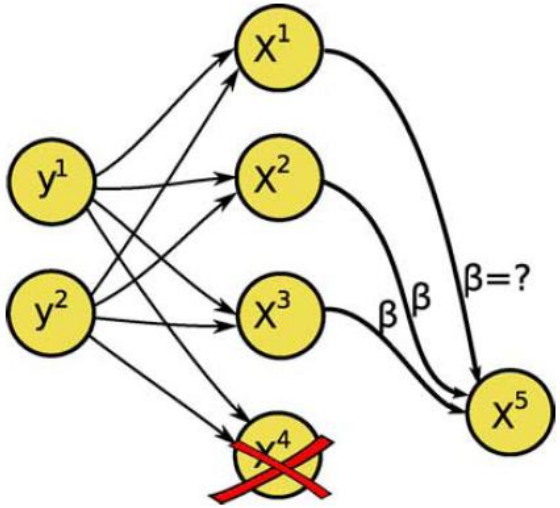
- 背景介绍
  - 分布式存储系统的各方面性能有待改进：
    - 可用性
    - 错误恢复的传输量
    - 存储空间
- 本文的解决方案：
  - OMMDS Code
  - Regenerating Code
- 性能评价

分布式存储系统中的关键问题

- 分布式存储系统的目标：
  - 可用性（Availability）高
  - 错误恢复传输量少
  - 存储空间少
- 分布式存储系统的问题：
  - 各方面性能都不尽如人意

改善分布式存储系统性能的解决方案

- 现有解决方案：
  - 复制R份
  - 使用擦除编码（Erasure Code）：前向纠错编码（FEC）的一种
  - 混合方案：一份复制 + 擦除编码
- 本文解决方案：
  - Optimally Maintained Maximum-Distance Separable (MDS) Code (OMMDS Code)
    - MDS是擦除编码的一种
  - Regenerating Code



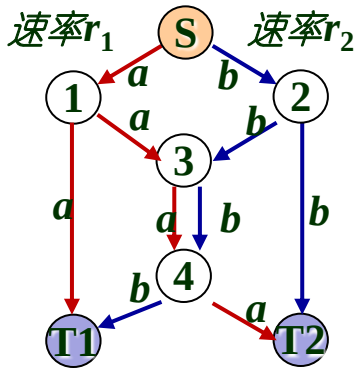
提纲

- 背景介绍
  - 分布式存储系统的各方面性能有待改进：
    - 可用性
    - 错误恢复的传输量
    - 存储空间
- 本文的解决方案：
  - OMMDS Code
  - Regenerating Code
- 性能评价

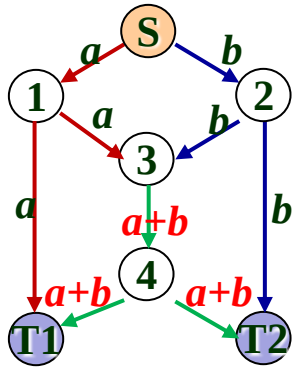
网络编码介绍

- 网络编码(Network Coding, 简称NC):
  - 由 R. Ahlswede等于2000年提出
  - 定义: 网络中的节点除了对收到的信息存储转发外，还可以进行处理；
  - 好处: 可以使组播达到最小割最大流定理所确定的理论最大吞吐率；
  - 研究进展: 线性网络编码就可以实现组播的最大吞吐率；随机线性网络编码；
  - 应用: 降低能耗，减小传输延时，负载均衡等。

一般组播

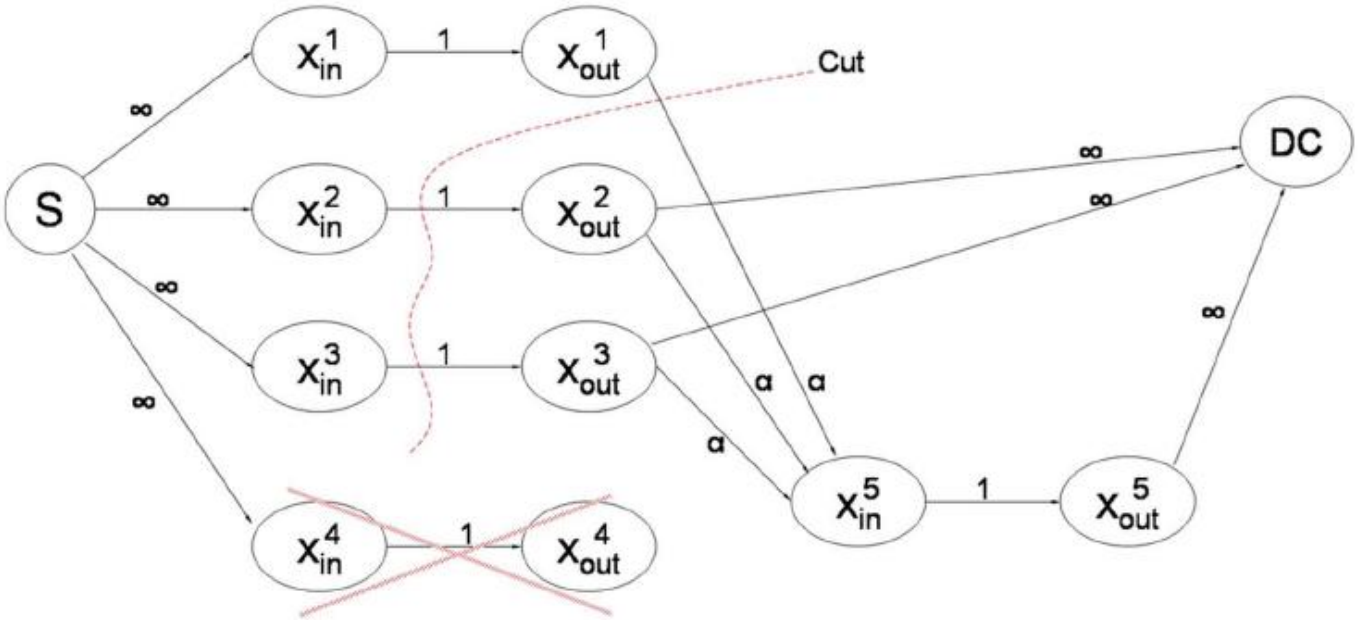


基于网络编码的组播



OMMDS Code-1

- S对应源，Xin和Xout对应节点，DC对应数据请求节点。
- 从源到Data Collector (DC)之间的最小割需大于等于k。



(n,k)-MDS code

Newcomer downloads data from n-1 nodes

$$\Rightarrow \alpha \geq \frac{1}{n-k}$$

提纲

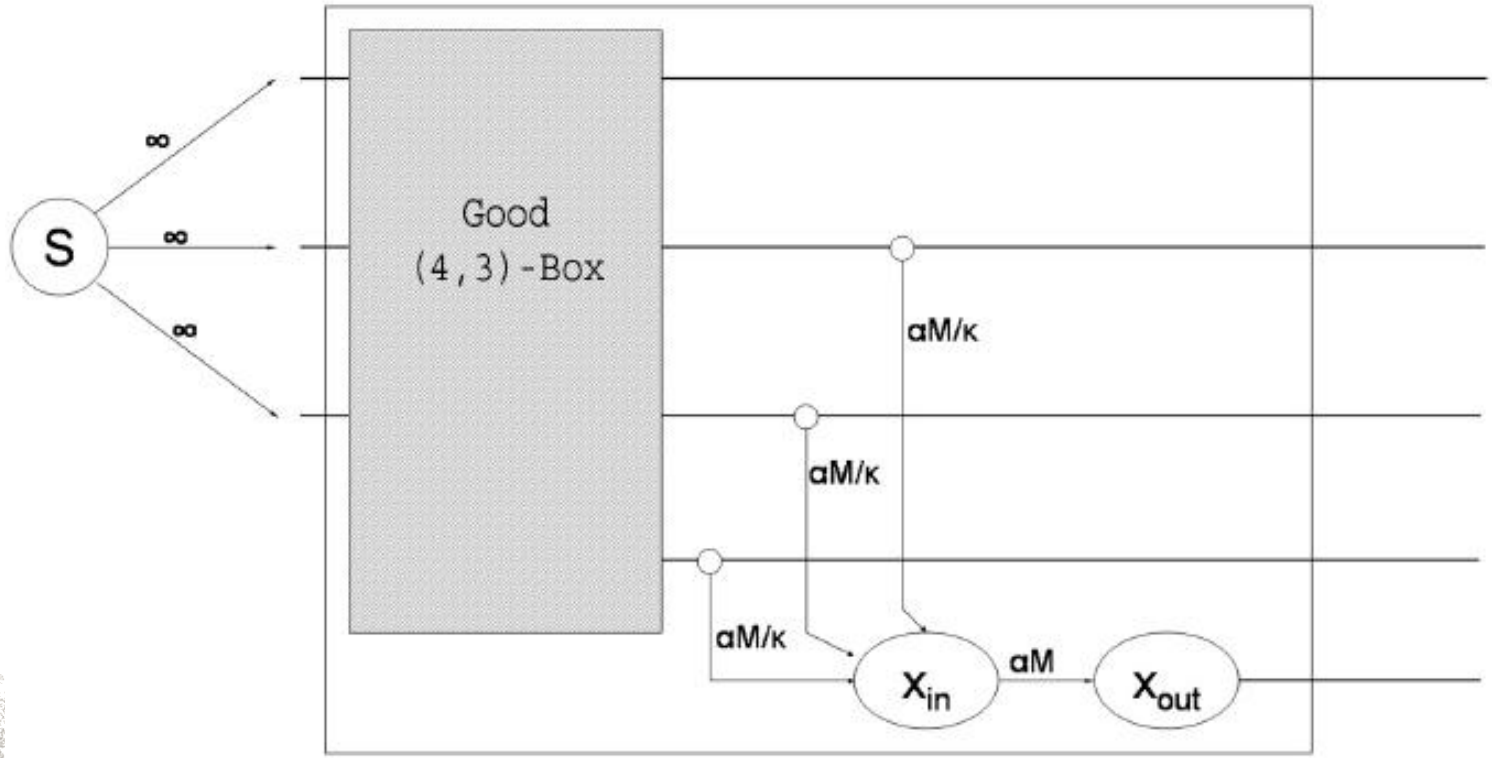
- 背景介绍
  - 分布式存储系统的各方面性能有待改进：
    - 可用性
    - 错误恢复的传输量
    - 存储空间
- 本文的解决方案：
  - OMMDS Code
  - Regenerating Code
- 性能评价

Regenerating Code

- 现有擦除编码方案：
  - 大小为M的文件分为k份，擦除编码将其变为n份，每个节点存一份。
    - 只要一个节点收齐k个就可以解出原始文件
- 现有擦除编码方案的问题：
  - 节点失效备份的最少传输量为OMMDS的结果
  - 理想情况：传输量为一份数据
- Regenerating Code：
  - 每个节点存aM份，newcomer与k个节点连接。
  - 节点失效备份的最少传输量为aM，每个节点传输aM/k比特。
  - 最小a取值：$$\alpha_c = \frac{1}{k} \times \frac{1}{1 - \frac{1}{k} + \frac{1}{k^2}}$$
  - 缺点：每份数据是原来的$\beta_{\mathrm{RC}} = k^2/(k^2 - k + 1)$ 倍，增加了存储量

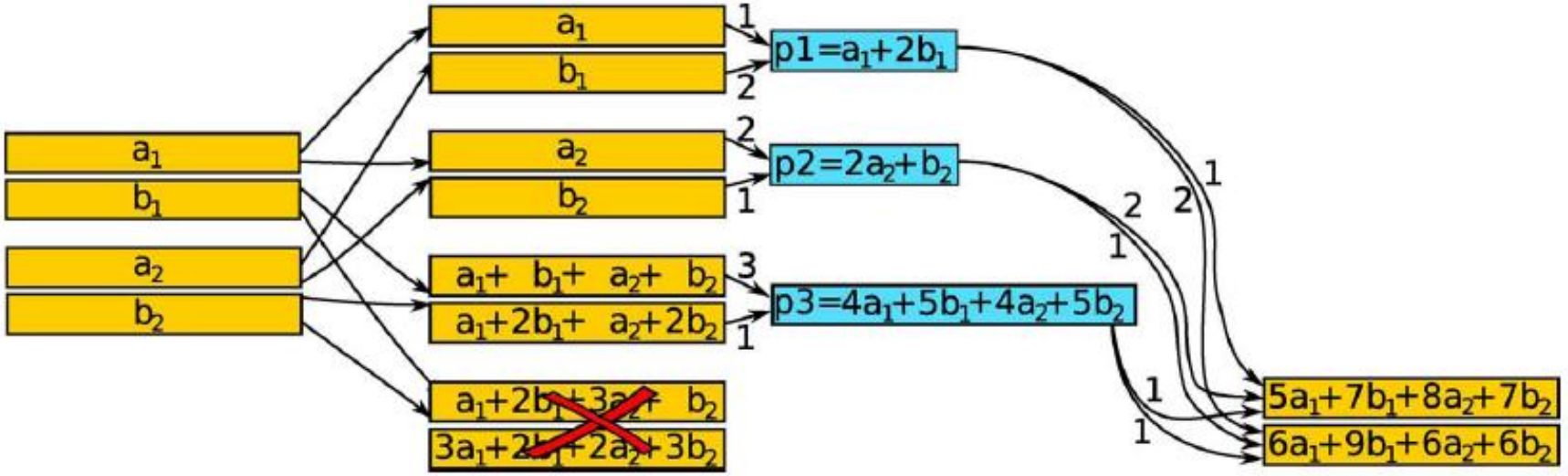
Regenerating Code证明

- 根据网络编码原理，Good box和新的Good box的最小割至少等于M。



Regenerating Code例子

- MDS（4,2）编码，每个包的大小为0.5M，每个节点存两个包1M的数据。
- Newcommer只需要传输1.5M就可以对失效节点进行备份。



提纲

- 背景介绍
  - 分布式存储系统的各方面性能有待改进：
    - 可用性
    - 错误恢复的传输量
    - 存储空间
- 本文的解决方案：
  - OMMDS Code
  - Regenerating Code
- 性能评价

实验中使用的Trace数据

- f：节点失效率
  - 如果一个节点连续一天时间没有回应，则认为该节点失效。
  - 节点失效后需要进行数据备份
- a：节点可用概率

Trace	Length (days)	Start date	Mean # nodes up	f (fraction failed per day)	a
PlanetLab	527	Jan. 2004	303	0.017	0.97
Microsoft PCs	35	Jul. 6, 1999	41970	0.038	0.91
Skype	25	Sept. 12, 2005	710	0.12	0.65
Gnutella	2.5	May, 2001	1846	0.30	0.38

TABLE I: The availability traces used in this paper.

五种方案的模型化结论

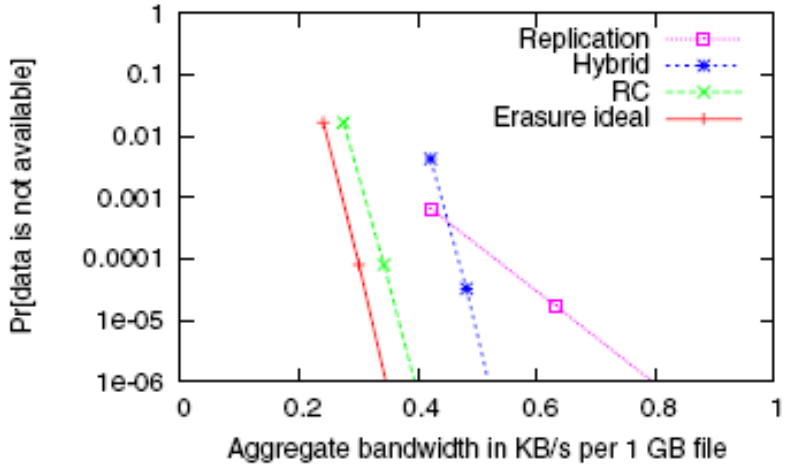
- 复制R份
  - 存储量： $\mathcal{R} \cdot \mathcal{M}$
  - 失效率： $(1-a)^{\mathcal{R}}$
  - 错误恢复传输量： $f \cdot \mathcal{R} \cdot \mathcal{M}$
- 理想的擦除编码
  - 存储量：
  - 失效率： $U_{\text{ideal}}(n,k) := \sum_{i=0}^{k-1} \binom{n}{i} a^i (1-a)^{n-i}$
  - 错误恢复传输量： $f \cdot \mathcal{R} \cdot \mathcal{M}$
- 混合
  - 存储量： $\mathcal{R} \cdot \mathcal{M}$
  - 失效率： $(1-a) \cdot U_{\text{ideal}}(n,k)$
  - 错误恢复传输量： $f \cdot \mathcal{R} \cdot \mathcal{M}$

五种方案的模型化结论

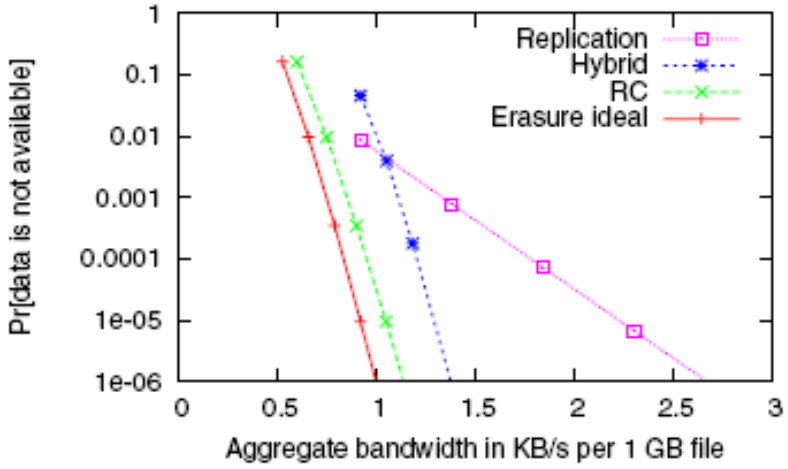
- OMMDS编码
  - 存储量： $\mathcal{R} \cdot \mathcal{M}$
  - 失效率： $U_{\text{ideal}}(n,k)$
  - 错误恢复传输量： $f \cdot \mathcal{R} \cdot \mathcal{M} \cdot \beta_{\text{OMMDS}}$
- Regenerating Code
  - 存储量： $\mathcal{M} \cdot n \cdot \beta_{\text{RC}}$
  - 失效率： $U_{\text{ideal}}(n,k)$
  - 错误恢复传输量： $f \cdot \mathcal{M} \cdot n \cdot \beta_{\text{RC}}$

$$\beta_{\text{RC}} = k^2/(k^2 - k + 1)$$

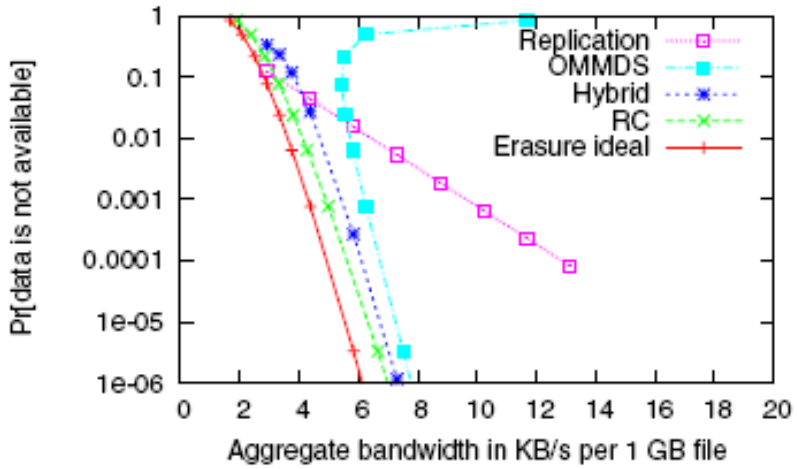
模型对Trace文件的分析结果—k=7



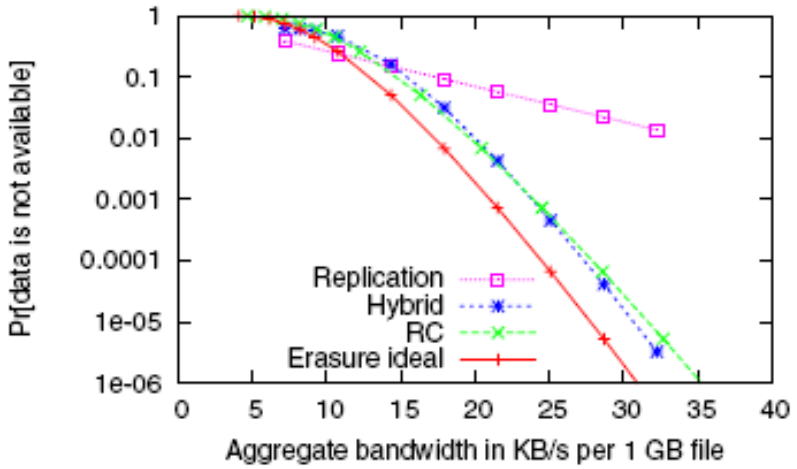
(a) PlanetLab trace



(b) Microsoft PCs trace

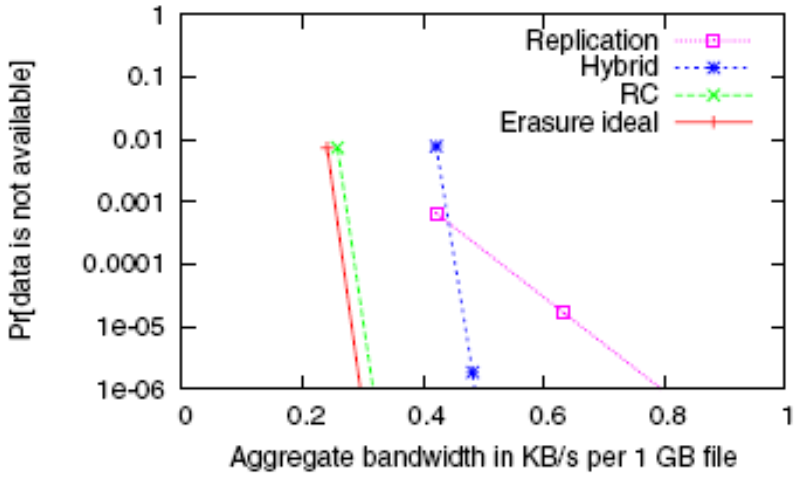


(c) Skype superpeers trace

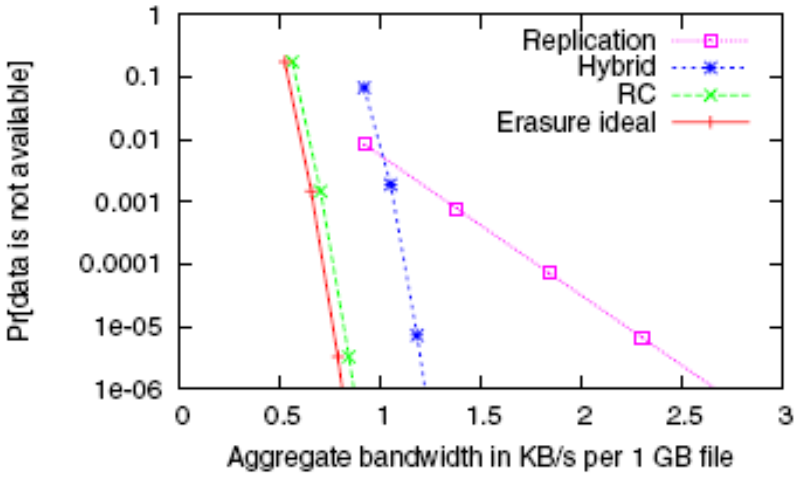


(d) Gnutella peers trace

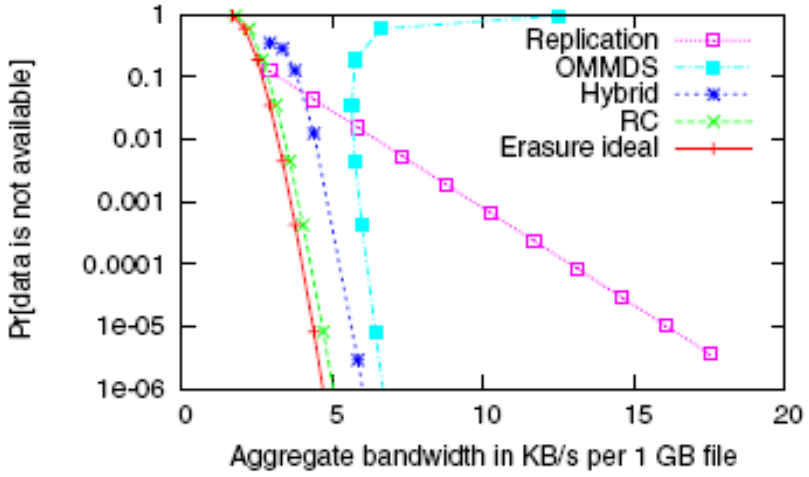
模型对Trace文件的分析结果—k=14



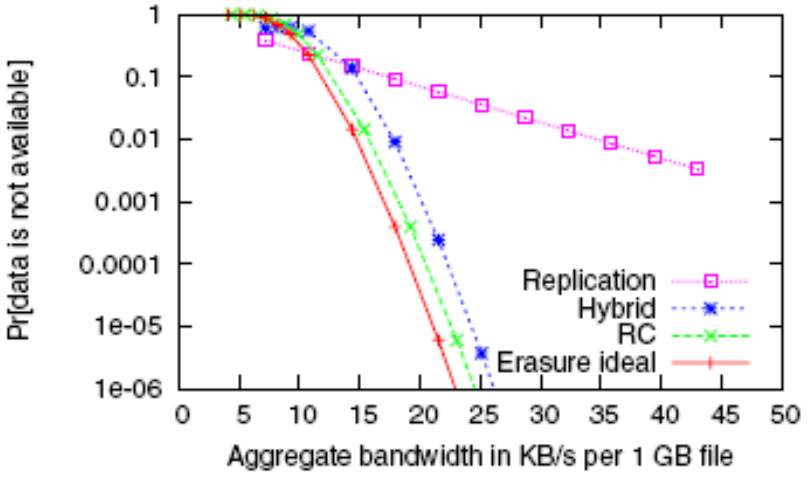
(a) PlanetLab trace



(b) Microsoft PCs trace



(c) Skype superpeers trace



(d) Gnutella peers trace

结论

- Regenerating Code的优点（与混合相比）：
  - 降低了维护开销
  - 简化了系统结构
- Regenerating Code的缺点（与混合相比）：
  - 需要与k个节点通行才能恢复原有数据
  - 增加了数据读取所需的带宽 $\beta_{\mathrm{RC}} = k^2/(k^2 - k + 1)$
- Regenerating Code适用场景：
  - 不经常进行读取操作的大文件存储系统