中国科学: 信息科学 2011 年 第 41 卷 第 9 期: 1089–1100
www.scichina.com info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS
论 文

P2P 存储系统中基于用户体验的可用性模型与应用

吴俣*, 杨智, 曲直, 肖臻, 代亚非

北京大学计算机科学与技术系, 北京 100871
* 通信作者. E-mail: viewrain@gmail.com

收稿日期: 2009–05–19; 接受日期: 2010–02–10
国家自然科学基金 (批准号: 60873051, 61073015)、国家重点基础研究发展计划 (批准号: 2011CB302305) 和 MoE-Intel 合作研究基金 (批准号: MOE-INTEL-09-06) 资助项目

摘要 保证数据的可用性是 Peer-to-Peer (P2P) 存储系统最重要的属性之一. 可用性分析模型和数据的放置是 P2P 存储系统设计的两个关键问题. 由于用户在 P2P 存储系统中同时作为服务节点和访问节点, 决定了可用性分析必须以用户为中心, 从而提高服务质量, 并降低系统开销. 目前广泛使用的可用性分析模型以及随机放置方法存在以下缺点: 1. 忽视了节点在线时间的模式, 会在不同时间段高估或低估节点的可用性; 2. 忽视了节点对数据的访问规律, 不能准确评估用户体验到的可用性; 3. 忽视了节点可用性差异, 缺乏激励机制. 本文提出了一个新的基于用户体验的可用性模型, 它能够从用户体验的角度, 评价 P2P 存储系统的可用性, 同时它也可以兼容传统的可用性分析模型. 在新模型的基础上, 本文针对两种典型的 P2P 存储应用: 数据共享和个人备份, 提出了相应的分布式数据分发算法. 通过真实日志驱动实验证明, 新的算法在数据共享应用中能大大降低可用性方差, 减少低可用性数据; 同时, 在个人备份应用中能针对不同用户的贡献提供不同层次的服务, 起到有效的激励作用.

关键词 对等网络存储 用户体验 可用性 数据放置算法

1 引言

可用性 (Availability) 是衡量存储系统中数据能被获取的概率. P2P 网络具有高扩展性和自主性, 为搭建存储系统提供了良好的条件. 但由于系统用户高动态性, 如何保证数据的可用性是 P2P 存储系统设计的关键. 在很多应用中, 数据并不需要时刻在线, 而只需在有访问的时候保证数据可用. 因此, 以提高用户感知的可用性为目的来放置数据更能合理分配以及节省资源. 但基于用户的可用性分析面临着以下挑战: (1) 如何准确描述用户的在线行为, (2) 如何从用户的角度来评价数据可用性, (3) 如何选择节点放置副本来提高用户体验到的可用性.

目前的研究 [1−5] 都是从系统角度评价数据可用性, 以随机放置算法为主要部署策略. 这种传统的系统分析模型和放置算法有如下缺点: 第一, 忽视了节点在线时间的模式, 导致节点在不同时间段的可用性被高估或者低估, 不能准确描述用户的在线行为; 第二, 忽视了 P2P 存储应用中节点对数据的

引用格式: 吴俣, 杨智, 曲直, 等. P2P 存储系统中基于用户体验的可用性模型与应用. 中国科学: 信息科学, 2011, 41: 1089–1100

访问规律, 不能从用户角度准确评估数据可用性; 第三, 忽视了节点在线模式和可用性 (即对系统所做的贡献) 的差异, 容易出现资源分配与贡献不对等的情况, 缺乏有效的激励机制.

本文在精确描述节点在线模式和可用性的基础上, 从用户体验的角度建立可用性模型, 并针对典型的 P2P 存储应用提出相应的分布式数据放置算法, 很好地解决了文节开头提到的 3 个主要问题, 最后通过真实日志驱动的实验证明这些算法在提高用户体验方面的有效性以及对系统的积极作用.

2 相关工作

在已有的基于 P2P 的网络存储系统中, 如: TotalRecall[1], Farsite[6] , OceanStore[7], 数据放置问题已有相当多的研究. 但这些数据放置算法都是从系统的角度出发, 忽略了用户的在线模式, 并且不考虑节点对系统贡献的差异. 与之不同, 本文从用户的角度去分析可用性并提出相应的数据放置策略。一些已有的研究考虑到节点和文件的差异性, 文献 [8] 希望把访问频率高的文件放在可用性高的节点上, 以提高系统的服务可用性. 文献 [9] 在假设文件访问频率相同时, 提出一些启发式算法, 包括贪心和分组算法. 然而以上研究未把时间作为分析的因素. 相反, 本文考虑了用户在不同时间段的在线行为及其对可用性的影响, 从另一维度优化用户感知的可用性. 虽然最近的文献 [10, 11] 将时间作为一个重要参数纳入考虑范围, 但与本文关注点不同. 文献 [10] 中提出了基于时间的副本修复 (time-related replication) 算法, 利用节点之前的会话 (session) 时间大概估计下次下线时间. 文献 [11] 中提出了细粒度 (fine-gained) 的随机过程模型, 研究数据可用性随时间的变化. 然而这些模型都没有刻画用户的长期在线模式. 相反, 本文的重点在于找出用户在周期内在线率变化的规律, 从用户角度建立可用性模型, 并利用已有的节点资源合理放置文件, 优化用户体验到的可用性.

3 基于用户体验的可用性模型

本节将详细描述如何从用户角度分析数据可用性. 首先, 介绍 P2P 存储系统以及相应的测量方法; 第二, 提出精确描述节点活动模式的方法; 第三, 用向量的方式刻画数据可用性; 最后, 把用户访问模式加入到可用性分析模型中, 从而建立基于用户体验的可用性分析模型.

3.1 P2P 存储系统

本文描述的 P2P 存储系统是由大量节点组成的互相协作存储的系统. 节点加入系统时, 会贡献出一部分存储资源为其他节点放置文件. 同时, 该节点也可以分发自己的文件到其他用户存储. 表 1 总结了系统模型的关键参数.

3.2 节点活动模式

准确描述节点活动模式对于基于用户的可用性分析具有重要意义, 一方面, 多个节点的在线活动可决定它们所存储的文件可用性, 另一方面, 节点的访问规律会影响到用户体验到的文件可用性.

由文献 [12] 可知, 在高动态的 P2P 系统中节点活动呈现以天为周期的周期性, 所以我们用在线概率向量 $p_i = \langle p_{i1}, \ldots, p_{in} \rangle$ 代替单一的平均可用性, 其计算方法如下:

通过统计一段时间 t 内一个节点的在线情况, 可以得到该节点在线时间 t_{online}, 那么它在这段时间的可用性 p 可以由 (1) 式得到

表 1 系统参数

Table 1 System parameters

参数名	描述
F_i	节点 i 需要备份的文件集合
F	系统所有文件的集合: $F=\bigcup_i F_i$
P_i	节点 i 可用来存储数据的邻居节点集合
P	系统所有节点集合: $P=\bigcup_i P_i$
N	系统节点总数: $N=\|P\|$
M	系统文件总数: $M=\|F\|$
$p_i=\langle p_{i1},\ldots,p_{in}\rangle$	节点 i 在线概率向量, $p_{it}\in[0,1]$ 表示节点 i 在第 t 时段的在线概率
s_i	节点 i 提供的存储空间
f_j	文件 j 的大小
n_j	文件 j 的副本数, 不包括文件提供者的原始副本
$R=[r_{ij}]$	一种可行的副本放置方法
$p=[r_j]$	放置文件 j 的节点集合所对应的在线概率向量
$A_j=\langle A_{j1},\ldots,A_{jn}\rangle$	文件 j 的可用性向量, $A_{jt}\in[0,1]$ 表示文件 j 在第 t 时段的在线概率
$p[d_j]$	访问文件 j 的节点集合的访问概率向量, 为了简化, 用其在线概率向量代替
$U=[U_{ij}]$	用户体验到的可用性, U_{ij} 表示节点 i 对文件 j 所感受到的可用性

$$p=t_{\text{online}}/t. \tag{1}$$

由于节点呈现出以天为周期的周期性活动, 所以我们考察在一个周期 T(一天) 内不同时间段的在线概率. 假设节点在时间段 Δt 内在线概率不变, 那么节点 i 的在线概率向量为

$$p_i=\langle p_{i1},\ldots,p_{in}\rangle,\quad n=T/\Delta t,\quad p_{it}\in[0,1] \tag{2}$$

p_{it} 表示节点 i 在 Δt_t 上的在线概率, 文献 [13] 已经证明节点每天在同一时间段的在线活动相似, 所以它可以准确刻画节点的在线模式; 另外, 其分量的平均值即为传统的平均可用性, 所以该向量可以同时刻画在线规律和整体可用性. 我们根据 MAZE 系统日志 (见 5.1 小节) 对用户的在线模式进行了聚类分析, 得到用户在线的一些规律: (1) 节点在不同时段在线概率的差异很大, 所以单一均值不能反映节点特征; (2) 即使节点的平均可用性相近, 其在线模式也会有很大不同; (3) 不同类节点的活跃时段之间有 2–7 h 的偏移, 所以在某些时段上可以互补; (4) 最高层可用性节点一直都保持着高可用性, 如果将其分散开, 和低可用节点搭配, 可以发挥更大的效用.

3.3 数据可用性

假设文件 j 存在 k 个节点上, $\{p_i\}$ 代表这 k 个节点的在线向量, 则 j 在一天内的第 t 个时间段的可用性 A_{jt} 如 (3) 式所示:

$$A_{jt}=P(\text{at least one peer is online})=1-P(\text{none of peers is online})=1-\prod_{i=1,\ldots,k}(1-p_{it}). \tag{3}$$

由此可知, 数据可用性向量为

$$A_j=A_j(\{p_i\})=\left\langle 1-\prod_{i=1,\ldots,k}(1-p_{i1}),\ldots,1-\prod_{i=1,\ldots,k}(1-p_{in})\right\rangle. \tag{4}$$

系统中节点数目以及存储容量有限, 所以文件平均冗余度有最大值 $r^* = \sum_{i=1}^{N} s_i / \sum_{j=1}^{M} f_j$, 假定每个文件的副本数 n_j 相同, 则 $n_j \leqslant r^*$.

备份的时候, 我们为每个文件选出至多 n_j 个可用节点放置数据, 从而得到数据放置矩阵 $R = [r_{i,j}]_{N\times M}$, 其中 $r_{i,j}$ 表示文件 j 是否被分配给节点 i 存储:

$$r_{i,j} = \begin{cases} 1, & \text{当节点 } i \text{ 存储文件 } j \text{ 时,} \\ 0, & \text{其他.} \end{cases} \qquad i = 1, 2, \ldots, N, j = 1, 2, \ldots, M,$$

对每个节点来讲, 所存数据大小不能超过其提供的容量, 所以有 (5) 式; 对每个文件来讲, 副本数可以通过 (6) 式进行计算.

$$\forall i, \sum_{j=1}^{M} r_{i,j} f_j \leqslant s_i, \tag{5}$$

$$\forall j, \sum_{i=1}^{N} r_{i,j} = n_j \leqslant \sum_{i=1}^{N} s_i \Big/ \sum_{j=1}^{M} f_j. \tag{6}$$

r_j 代表 R 的第 j 列向量, 由此可知存储文件 j 副本的节点集合, 我们把这些集合的在线模式向量表示为 $p[r_j]$, 则 f_j 的数据可用性向量 $A_j = A_j(p[r_j])$ 可由 (3) 和 (4) 式计算得出.

3.4 用户体验的可用性分析

3.3 小节从数据的角度准确分析了其在不同时间段的可用性, 但对用户而言, 不同的访问模式会导致差异很大的用户体验. 下面我们将从用户的角度来建立可用性分析模型.

$U = [U_{ij}]$, U_{ij} 表示节点 i 对于文件 j 所感受到的可用性. 假设文件 j 的可用性向量为 $\langle A_{j1}, \ldots, A_{jn}\rangle$. 节点 i 的在线概率向量为 $\langle p_{i1}, \ldots, p_{in}\rangle$, 为了简化, 我们用节点的在线向量来代替节点的访问向量, 表示当节点在线时随时都有可能访问该文件.

则节点 i 对于文件 j 所感受到的可用性

$$U_{ij} = \sum_{t=1}^{n} p_{it} A_{jt} \Big/ \sum_{t=1}^{n} p_{it}. \tag{7}$$

对每个节点来讲, 公式 (7) 的分母是由节点自身决定, 因此为了提高 U_{ij}, 放置算法需要, 在现有的节点资源限制下, 最大化 $\sum_{t=1}^{n} p_{it} A_{jt}$. 由 (7) 式可知, 文件被访问的概率与访问节点集合的在线向量 $p[d_j]$ 相关. 如果文件 j 只被一个用户 i 访问, 则 (7) 式中的 p_{it}, 只与这一个节点的在线向量相关. 如果文件 j 被 k 个用户访问, 则公式 (7) 中的 p_{it} 等于 k 个访问用户的在线向量对应分量的均值.

4 启发式数据分发算法

本章将要介绍 3 种去中心化的启发式数据分发算法, 分别是现有的随机分发算法, 和本文提出的分层模式互补和同层模式相似算法.

4.1 准备知识

1) 信息收集.

在文献 [9] 中已经详细介绍了: i) 可写节点 (writable peers) 集合 P_i, 也就是节点 i 可用来存储数据的邻居节点集合; ii) 最大冗余度的估计: 首先通过同步相位 (locking phase) 的策略保证节点资源不被多个节点重复计算, 然后利用公式 (8)–(10) 算出节点 i 能分发的最大副本数 n_i:

$$S_i = \sum_{i' \in P_i} s_{i'}, \tag{8}$$

$$Z_i = \sum_{i' \in P_i} \sum_{j \in F_{i'}} f_j, \tag{9}$$

$$n_i = S_i / Z_i. \tag{10}$$

2) 节点信息传递: 在协议消息中附加节点 i' 的存储空间 $s_{i'}$, 需要存储的文件总大小, 以及本身的在线可用性信息. 本节将沿用这些概念, 但不同的是文献 [9] 中传递的可用性信息是节点平均可用性, 而我们将传递该节点的在线模式信息.

3) 节点在线模式信息及数据可用性信息: 在实际系统设计中, 我们采用在线位串的方式来表达传递节点在线模式信息. 每个节点 i 记录其最近一段时间的在线位串 b_i, 比如, 统计之前一周每个小时 [1)] 的在线情况: 在线为 1, 反之为 0, b_i 形如 $10111100\cdots1010$, 用 $\mathrm{len}(b_i)$ 表示位串的总位数, 用 $\{b_i\}$ 表示位中 1 的个数, 则节点平均可用性为

$$A_i = \{b_i\}/\mathrm{len}(b_i). \tag{11}$$

我们还可以通过简单的与或运算得出不同节点共现的情况和总体覆盖时间段的情况: 假设节点 A, B 的在线位串分别是 b_A 和 b_B, 那么共现的时段数为 $C_{AB} = \{\mathrm{len}(b_A \& b_B)\}$, 总体覆盖的时段数为 $O_{AB} = \{\mathrm{len}(b_A | b_B)\}$, 共现系数为 $\rho = C_{AB}/O_{AB}$, 覆盖系数为 $\sigma = O_{AB}/\mathrm{len}(\{b_A\} + \{b_B\})$.

显然, 共现系数越大, A 和 B 的可用性以及在线模式越相似; 覆盖系数越大, A 和 B 的可用性或/与在线模式差异越大, 就越互补. 数据可用性信息可以通过 “或” 操作来计算. 比如, f_j 只有 3 个副本, 放在节点 A, B, C 上, 假设其在线位串分别是 b_A, b_B, b_C, 那么 f_j 所获得的可用性位串为 $A_j^b = b_A | b_B | b_C$. 通过该位串, 可算出数据可用性向量及平均可用性.

4.2 随机分发算法

随机算法分为两步, 首先通过收集到的可写节点集合信息使用公式 (8)–(10) 计算出副本数 n_i, 然后从可写节点集合中随机选择 n_i 个节点存储. 这是已有的方法 [9], 本文不再赘述.

4.3 分层模式互补算法

分层模式互补专门针对热门共享数据, 算法目的是: (1) 尽可能使每个文件都能公平使用资源, (2) 存储节点的可用时间能尽可能覆盖所有时段, 使得这些热门文件能在各个时段被用户访问到. 算法分为两个步骤:

1) 副本数计算: 与随机分发算法相同.

1) 时间粒度可以根据实际系统情况调整, 原则上是小于用户每次上线的平均时间

2) 数据分配: 统计 P_i 中每个节点的平均可用性, 并按递增顺序排序, 将排序好的节点分成 L 组: $g = g_1, g_2, \ldots, g_L$, 根据每个组内的最小可用性 $\min_i$ 可以得到 L 个区间: $[\min_1, \min_2), [\min_2, \min_3), \ldots, [\min_L, +\infty)$[2], 对任意一个可用性值, 如果它落在第 l 个区间, 则属于第 l 层. 对节点 i 要存储的每一个文件 j, 首先从 P_i 中随机选择一个节点 (有足够可用空间) 分配第一个副本, 记录当前文件的可用性位串; 然后对每个文件循环分配剩余副本: i) 根据算出的当前每个文件的平均可用性, 按递增顺序排序, 得到 $\langle A_1^b, A_2^b, \ldots, A_k^b \rangle$(假设节点 i 有 k 个文件要存储); ii) 按照顺序依次对每个文件 j 评估其可用性在 g 中所属的层次 l, 然后在 g_{L+1-l} 中选择与覆盖系数最大的节点 (有足够可用空间) 来存储下一个副本. 若 g_{L+1-l} 中无可用节点, 则从临近层次的节点中选择. 忽略此情况; iii) 更新 A_j^b. 具体算法描述如下:

分层模式互补算法

可写节点集合估计:

1. 节点 i 选择可写节点集合 P_i.
2. 锁住 (lock) $x \in P_i$.
3. 估计 $S_i = \sum_{i' \in P_i} s_{i'}$ 和 $Z_i = \sum_{i' \in P_i} \sum_{j \in F_{i'}} f_j$, 计算 $n_i = S_i / Z_i$.

副本放置:

4. for each $x \in P_i$: 计算平均可用性 A_x.
5. sort$\{A_x\}$, $x \in P_i$, 按序分成 L 组: $g = \{g_1, g_2, \ldots, g_L\}$.
6. for each $f_j \in F_i$: 随机选择 $x \in P_i$ 放置第一个副本, initialize(A_j^b).
7. $r = 2$ to n_i: // 放置剩余 $n_i - 1$ 个副本.
   - 7.1 for each f_j in F_i: 计算 A_j by A_j^b.
   - 7.2 $F_i' = \text{sort}\{f_j\}$ by A_j, $f_j \in F_i$.
   - 7.3 for each $f_j \in F_i'$:
     - 7.3.1 计算 A_j 所属区间 l.
     - 7.3.2 选择 $x \in g_{L+1-l}$(空间足够) 分配第 r 个副本, maximize $\sigma(b_x, A_j^b)$.
       - i) if $x = \phi$, 从 g_{L+1-l} 的临近层次中选择 x, maximize $\sigma(b_x, A_j^b)$.
       - ii) if $x = \phi$, 跳过.
     - 7.3.3 $A_j^b = A_j^b \& b_x$.
8. 释放 P_i 中的节点.

4.4 同层模式相似算法

同层模式相似算法针对个人备份系统, 算法目的是: (1) 尽量保证用户在线时能访问到数据, 所以把数据存在那些在线模式与其相似的邻居节点上. (2) 激励用户在线贡献资源, 使节点感受到的数据可用性与其本身的节点可用性正相关. 算法分为两个步骤:

1) 副本数计算: 与随机算法相同.

2) 数据分配: 首先, 统计 P_i 中每个节点的总体可用性, 按递增顺序排序, 将排好序的节点分成 L 组: $g = g_1, g_2, \ldots, g_L$. 确定节点 i 的可用性层次 l, 然后对于节点 i 要存储的每一个文件 j, 在同层次或相邻层次中选择有足够空间且共现系数最大的 n_i 个节点存储副本.

2) 这里没有区间 $(-\infty, \min_1)$, 因为在这些节点中无论怎么分配, 可用性都不会小于 $\min_1$

同层模式相似算法

可写节点集合估计:

1. 节点 i 选择可写节点集合 P_i.
2. 锁住 (lock) $x \in P_i$.
3. 估计 $S_i = \sum_{i' \in P_i} s_{i'}$ 和 $Z_i = \sum_{i' \in P_i} \sum_{j \in F_{i'}} f_j$, 计算 $n_i = S_i/Z_i$.

副本放置:

4. for each $x \in P_i$: 计算平均可用性 A_x.
5. sort$\{A_x\}$, $x \in P_i$, 按序分成 L 组: $g = \{g_1, g_2, \ldots, g_L\}$.
6. for each $f_j \in F_i'$:.
   - 6.1 if: $f_j \in g_l, |g_l| \geqslant n_i$, 选择 n_i 个节点 X 存储副本, maximize ρ.
   - 6.2 else:
     - 6.2.1 $X = g_l$, 存储 $|g_l|$ 个副本.
     - 6.2.2 在 g_l 临近的层次中选择节点, maximize ρ.
8. 释放 P_i 中的节点.

5 模拟实验

本节将通过真实日志驱动的模拟实验来研究: (1) 传统随机放置算法和新算法 (见 4.2 和 4.4 小节) 所获得基于用户体验的可用性差异; (2) 比较上述 3 种算法在不同连通度的 P2P 系统中以及不同副本数下的性能差异.

5.1 日志数据

本文中的实验都是基于 MAZE[3)] 系统的在线日志. MAZE 是由本研究组建立的在中国教育网 (CERNET: China education and research network) 内最大的 P2P 文件共享系统, 拥有超过两百万注册用户以及 2 万的同时在线人数. 如文献 [14] 所述, 该系统节点的动态性很高, 平均可用性较低 (0.26), 对最高带宽有限制. 我们之前的研究 [14−16] 已经证明了 MAZE 系统中的用户行为与其他流行的 P2P 文件共享系统用户的行为类似. 所以本文的分析具有广泛的适用性.

MAZE 系统每隔 3 min 就对所有在线用户信息记录一个快照 (snapshot), 这些快照组成了用户的在线日志. 通过它, 可以获得节点每次上线和下线的时间. 如果节点在两周内没有再上线, 则认为该用户永久离开系统. 我们收集了连续一个月的在线日志: 从 2005 年 3 月 1 日到 2005 年 3 月 30 日, 平均每天在线活跃用户为 11 K (平均可用性大于 0.2), 这些节点平均持续在线时间为 5.7 h, 平均持续下线时间为 9.5 h. 由于本文暂不考虑用户永久离开后的修复问题, 所以随机选出 344 个在此期间一直存活的节点进行模拟分析.

5.2 评测方法

5.2.1 模拟环境建立

344 个被选出的节点组成一个 P2P 存储系统池, 节点之间的链接是随机的, 假设网络中任两个节点的连通概率为 $m \in [0, 1]$, 当 $m < 1$ 时, 是非集中式的 P2P 系统; 当 $m = 1$ 时, 可以看成集中式的

3) http://maze.pku.edu.cn

P2P 系统.

节点的上下线行为由日志驱动, 真实体现了用户的在线模式和可用性. 在模拟中, 我们假设每个节点可存储 20 个大小相同的文件. 存储总空间和文件总大小的比值为冗余度 $r^* = \sum_{i \in P} s_i / \sum_{j \in F} f_j$, 该值也是可控制的实验参数. 为了减少随机的偶然性, 我们将每个实验运行 20 次, 给出平均结果.

5.2.2 模拟器设计

1) 共享热门文件存储系统模拟器: 按照 5.2.1 小节所述建立模拟环境, 每个节点根据前两周日志获得在线位串, 分别使用随机算法和分层模式互补算法进行数据分发. 然后根据剩余 16 天的用户在线日志查看每个小时文件是否可用, 从而计算以下评测指标: (1) 所有用户体验的平均可用性 $\overline{A}$ 及其分布, (2) 用户体验的可用性方差 var. 值得注意的是, 文献 [9] 中提出的分组算法只是分层模式互补算法的特例, 即节点的在线概率向量长度为 1 的特殊情况, 因此不在实验中单独比较.

2) 用户文件备份系统模拟器: 与前者类似, 但数据分发策略分别是随机算法和同层模式相似算法. 其评测指标有: (1) 所有用户体验的平均可用性 $\overline{A}$ 及其分布; (2) 节点贡献 (自身可用性) 和其体验的相关度.

5.3 实验结果

本章中, 我们比较了随机算法和本文的两种新算法在提高用户体验的性能差异.

5.3.1 共享热门文件存储系统

在共享热门文件存储系统模拟器中我们比较了随机选择算法 (简写为 Rand) 和分层模式互补算法 (简写为 DL-PC: different levels and pattern complemented) 在不同连通度及不同冗余度下的性能, 评测指标如 5.2.2 小节所述. 实验结果表明采用新算法能够获得更高的用户平均可用性, 同时减少不同用户体验的差异性, 防止过低可用性的出现. 图 1 显示的是, 在不同副本数 (r^*) 条件下, 所有用户体验的平均可用性 A^* 随着连通度 (m) 递增的变化情况. 由此可知: (1) 使用分层互补算法得到的用户平均可用性可提高约 9%; (2) 随着副本数增多, 随机选择算法的结果趋近分层互补算法的结果, 因为副本越多, 随机选择就越容易达到分层互补的效果; (3) 连通度越小, 用户感知的平均可用性越小, 随机算法尤其明显, 当连通度大于 0.1 时基本保持稳定. 图 2 给出在不同副本数下, 用户感知的可用性方差 var 随连通性递增的变化, 可知: (1) 分层互补所得到的方差远低于随机选择造成的方差, 且在不同副本数下方差基本不变. 说明该算法能够更加平衡的分配资源; (2) 随着副本数增多, 随机算法的方差会减小 (但仍高于分层互补算法). 图 3 显示的是用户感受的可用性在不同连通度和不同副本数下的 CDF 图 (由于篇幅所限, 只展示了 $m = 0.5, r^* = 3$ 的情况). 显然, 分层互补的曲线比随机选择的曲线整体向右偏移, 绝大部分的点都集中在中高可用性上; 且消除了超低可用性的情况 (无左部的长尾).

5.3.2 用户文件备份系统

在用户文件备份系统模拟器中我们比较了随机选择算法 (简写为 Rand) 和同层模式相似算法 (same level and pattern similar, 简写为 SL-PS) 在不同连通性和不同冗余度条件下的性能, 评测指标如 5.2.2 小节所述. 结果显示, 采用同层相似算法可以保证用户平均可用性, 并为每个用户提供与其贡献成正比的服务.

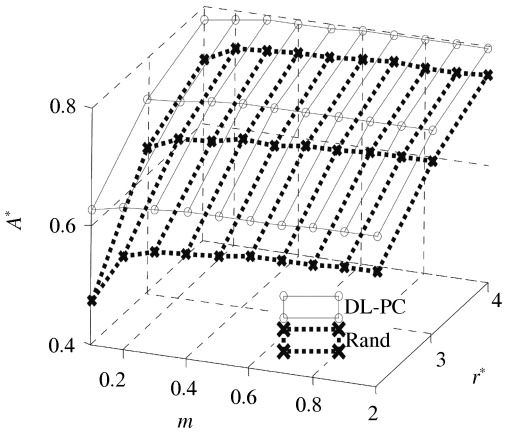

图 1 用户体验平均可用性

Figure 1 Mean user-experienced availability

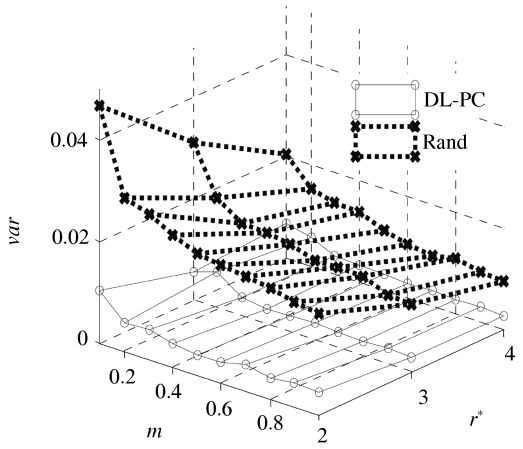

图 2 用户体验平均可用性方差

Figure 2 Variance of user-experienced availabilities

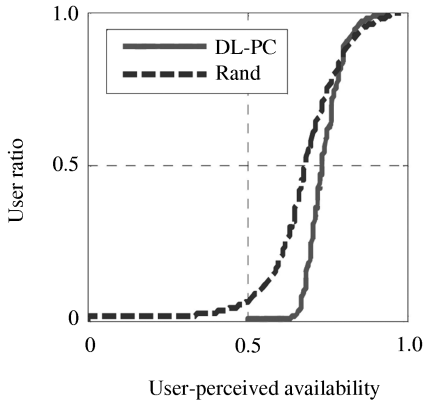

图 3 用户感受的可用性的 CDF 图 ($m = 0.5, r^* = 3$)

Figure 3 CDF of user-experienced availabilities ($m = 0.5, r^* = 3$)

图 4 所示的是, 采用 SL-PS 算法后, 分别按照基于用户体验的可用性分析模型 (简写为 UB) 和传统的系统可用性分析模型 (简写为 TS) 对其可用性进行评估的结果, 并且显示其在不同副本数 (r^*) 和连通度 (m) 条件下的变化情况, 图 5 是对随机算法的相应评估. 显然, 传统系统可用性与用户感受的可用性存在巨大差异, 相差高达 0.3–0.5. 分层相似算法比相似算法所得到的用户平均体验略高, 且提供有差异的服务, 好处是: (1) 使得用户贡献与其体验成正比, 从而起到良好的激励作用; (2) 绝大部分活跃用户的体验得到了保证. 图 6 是按照用户的可用性递增顺序显示该用户体验到的的可用性 (由于篇幅所限, 只展示了 $m = 0.5; r^* = 1$ 的情况). 采用同层相似算法后, 节点随着自身可用性的增大将获得更好的用户体验, 而在随机算法下, 每个用户感受的可用性也是随机的, 所以每个用户都有搭便车 (free-rider) 的动机, 最终导致整个系统的瘫痪.

6 讨论

为了能够更好地应用本文提出的可用性模型和数据分发算法, 本节我们讨论一些算法在应用到

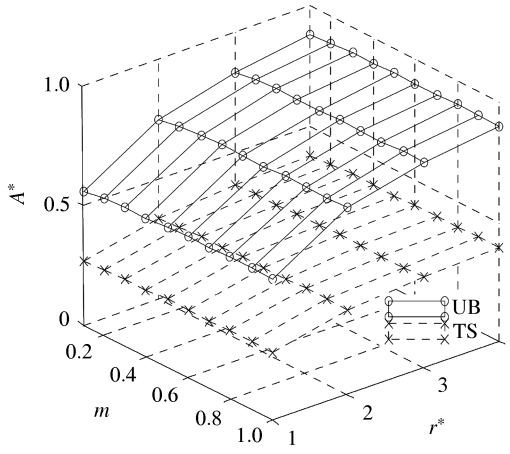

图 4　不同模型评估的可用性 (采用 SL-PS 算法)

Figure 4　Availabilities evaluated by different models (using SL-PS algorithm)

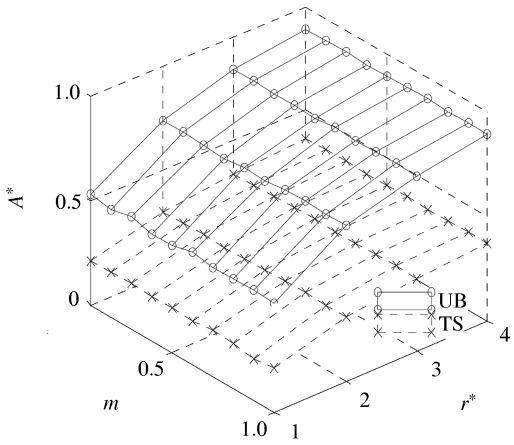

图 5　不同模型评估的可用性 (采用 Rand 算法)

Figure 5　Availabilities evaluated by different models (using Rand algorithm)

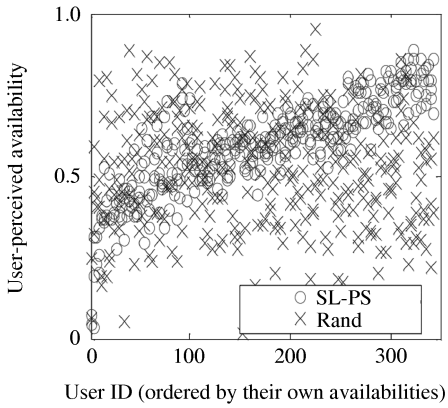

图 6　用户感知的可用性分布图 ($m = 0.5, r^* = 1$)

Figure 6　Distribution of user-experienced availabilities ($m = 0.5, r^* = 1$)

实际系统中可能遇到的一些问题, 以及如何扩展算法来解决这些问题. 在分层模式互补算法中, 假设共享的文件的热度是相近的. 对于文件热度不等的情况, 系统可以结合文件热度这个维度来进一步来优化设计数据放置机制. 针对文件热度这个维度上的优化已有很多工作, 其核心思想是把热度高的文件分配给在线率也高的节点来存储. 为了兼容这个维度, 可以将文件的可用性向量乘以一个权重 (归一化后为文件热度). 假设对于 N 个文件, 其任一文件 i 的访问频率是 r_i, 则其可用性向量扩展为 $\langle A_{i1}, A_{i2}, \ldots, A_{in}\rangle r_i/\Sigma_{i=1}^{N} r_i$. 这样, 分层模式互补算法的目标是就转化为在文件热度不等的条件下, 让系统整体的加权可用性最高. 算法将会考虑到文件的热度因素, 把热度高的文件分配给在线率也高的节点来存储. 在同层模式相似算法中, 我们默认在用户对每个文件访问的模式与其自身模式相近. 但是不同文件由于性质不同, 用户可能期望更为灵活的访问方式. 为此, 系统可以允许用户更改每个文件的访问向量. 但是, 为了保证贡献多的用户能够得到较高的服务质量, 用户仍然在同层选取在线模式与文件访问模式相近的节点.

7 总结

为了在高动态环境以及有限的用户资源条件下搭建 P2P 存储系统, 本文提出了基于用户体验的可用性分析模型, 将用户的在线模式、可用性差异以及访问模式作为参数, 比传统的可用性模型更加准确地描述了 P2P 存储系统的用户体验. 基于该模型, 我们对典型的 P2P 存储应用提出了相应的分布式算法, 并通过真实日志驱动的模拟实验证明了新算法可以在共享存储应用中提高用户体验到的可用性, 同时减少超低可用性文件的存在, 并使得整个文件可用性分布比较均衡; 在用户备份应用中保证绝大部分活跃用户的体验, 并且达到多劳多得的目的, 有效建立激励机制.

参考文献

1 Bhagwan R, Tati K, Cheng Y, et al. Total recall: system support for automated availability management. In: Proceedings of the 1st ACM/Usenix Symposium on Networked Systems Design and Implementation. San Francisco, 2004. 337–350

2 Bhagwan R, Savage S, Voelker G. Replication strategies for highly available peer-to-peer storage systems. In: Proceedings of FuDiCo: Future Directions in Distributed Computing. Bertinoro, 2002

3 Weatherspoon H, Kubiatowicz J. Erasure coding vs. replication: a quantitative comparison. In: Proceedings of IPTPS. Cambridge, 2002

4 Blake C, Rodrigues R. High availability, scalable storage, dynamic peer networks: pick two. In: Proceedings of the 9th Workshop on Hot Topics in Operating Systems. Lihue, Hawaii, 2003

5 Chun B G, Dabek F, Haeberlen A, et al. Efficient replica maintenance for distributed storage systems. In: Proceedings of the 3rd USENIX Symposium on Networked Systems Design and Implementation. San Jose, 2006

6 Adya A, Bolosky W J, Castro M, et al. FARSITE: federated, available, and reliable storage for an incompletely trusted environment. In: Proceedings of OSDI. Boston, 2002

7 Kubiatowicz J, Bindel D, Chen Y, et al. OceanStore: an architecture for global-scale persistent storage. In: Proceedings of ASPLOS. Cambridge, 2000

8 Ramanathan M. Increasing object availability in peer-to-peer systems. In: Proceedings of the 18th International Parallel and Distributed Processing Symposium. Santa Fe, New Mexico, 2004

9 Lin W K, Ye C, Chiu D M. Decentralized replication algorithms for improving file availability in P2P networks. In: Quality of Service, 15th IEEE International Workshop. Evanston, 2007. 29–37

10 Kim K. Time-related replication for P2P storage system. In: Proceedings of the 7th International Conference on Networking. Cancun, 2008

11 Tian J, Yang Z, Dai Y F. A data placement scheme with time-related model for P2P storages. In: Proceedings of the 7th IEEE International Conference on Peer-to-Peer Computing. Galway, 2007

12 Liu H Y. Feature analysis of the resources and use behaviors in P2P file-sharing system Maze. Dissertation for Master's Degree. Beijing: Peking University, 2005

13 Gao Q, Yang Z, Tian J, et al. A hierarchically differential P2P storage architecture. J Softw, 2007, 18: 2481–2494

14 Tian J, Dai Y. Understanding the dynamic of peer-to-peer systems. In: Proceedings of the 6th International Workshop on Peer-to-Peer Systems. Bellevue, 2007

15 Yang M, Zhang Z, Li X, et al. An empirical study of free-riding behavior in the Maze P2P file-sharing system. In: Proceedings of IPTPS. Ithaca, 2005

16 Lian Q, Peng Y, Yang M, et al. Robust incentives via multi-level Tit-for-Tat. Int J Concurr Comp, 2008, 20: 167–178

User-experience-based availability analysis model and its application in P2P storage systems

WU Yu*, YANG Zhi, QU Zhi, XIAO Zhen & DAI YaFei

Department of Computer Science, Peking University, Beijing 100871*, China*
*E-mail: viewrain@gmail.com

Abstract Data availability is one of the most important properties of peer-to-peer (P2P) storage systems. Availability analysis model and data placement are two key design choices. Users in P2P storage system are both providers and customers. This characteristic determines that the availability analysis must be user-centric, and thereby enhance the quality of service and decrease the system cost. The popular approach in recent studies is simple random placement with steady-state model, which has the following drawbacks: 1) It ignores the up/down patterns of nodes, whose availability is over-estimated or under-estimated at different periods of time. 2) It ignores the access patterns of users, so the availability perceived by users is hard to evaluate precisely. 3) It ignores the huge difference of nodes' availability, thus leading to the absence of incentive. This paper proposes a novel user-experience-based availability model, which evaluates the availability of P2P storage system in terms of user experience, which can degenerate to traditional availability analysis model. Based on the new model, this paper proposes decentralized data placement algorithms for two typical P2P storage applications: "data sharing" and "personal backup". By the trace-driven simulation, we prove that our methods can enhance the availability perceived by users greatly, reduce the variance of the availability dramatically and eliminate the nodes with low availability in data-sharing applications; meanwhile, it can provide different-level service to encourage users according to their contributions.

Keywords P2P storage, user experience, availability, data replacement