无结构 P2P 网络搜索方法及其改进

夏启志, 谢高岗
(中国科学院 计算技术研究所, 北京 100080)

摘 要: 对等(P2P)计算是未来网络中的关键技术,对等网络是实现下一代互联网的重要组成部分。如何高效地搜索 P2P 网络上的资源是 P2P 网络实现的最为关键的问题。在讲述对等网络的基本搜索方法的基础上,分析了许多改进的搜索方法,包括基于转发的改进方法、基于缓存的改进方法和基于覆盖网拓扑优化的改进方法。

关键词: 对等计算; 无结构 P2P 网络; 搜索方法
中图法分类号: TP393 **文献标识码**: A **文章编号**: 1001-3695(2005)09-0256-05

Search Methods and Its Improvements for Unstructured P2P Networks

XIA Qi-zhi, XIE Gao-gang
(Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: Peer-to-Peer computing will be a key technology in future networks. P2P networks will be an important part of the next generation Internet. How to search the resources in the P2P networks efficiently is the key problem when the P2P networks are developed. After introducing the basic search methods, this paper presents many improved search approaches. These include: forwarding-based approaches, cache-based approaches and overlay topology optimization.

Key words: Peer-to-Peer Computing; Unstructured P2P Networks; Search Methods

1 引言

对等计算[1]是实现下一代互联网(Next Generation Internet, NGI)的重要技术,也是 Internet 上发展最快的应用领域,特别是基于 P2P 的文件共享系统近年来得到了广泛应用。P2P 网络是建立在 Internet 上的覆盖网(Overlay),P2P 模式不同于传统的客户机/服务器模式,它改变了传统的集中存储和处理资源的方法,P2P 将网络边界上的资源有效地组织起来,P2P 使得资源提供者和资源接收者之间能够直接(Directly)相互交换信息。在理想的 P2P 网络中,节点既是客户端又是服务器,既提供资源又消费资源,P2P 应用在 Internet 上的流量在逐年增加。

P2P 网络像其他计算机系统一样是由应用驱动的,P2P 网络有以下应用目标:①网络的代价共享,对于集中式的网络系统同时服务于很多客户机,系统的代价和开销主要是由服务器来承担,而 P2P 网络把这些 CPU 处理代价和存储代价分散到网络的各个对等体;②更好的可扩展性和可靠性,系统中不再需要一个中央服务器;③资源集聚,P2P 是一个分布式的方案,它将网络上所有节点的计算能力和存储空间汇集到一起,将各个节点的文件汇合到一起,形成一个分布式的系统;另一个目标是自组织(Ad hoc)通信和动态性,P2P 创造了一个自组织的网络通信环境,对等体可以根据自己的意愿加入和离开;其他目标还有匿名能力和隐私能力等。对等网络有着非常广泛的应用,可分为资源共享和协同工作。资源共享包括时下流行的文件共享、分布式搜索和分布式计算。协同工作包括即时通信(Instant Messaging, IM)、协同游戏和协同办公等。

这些对等网络的可用性依赖于对数据高效的查找和提取方法。如何高效地搜索 P2P 网络上的资源是 P2P 网络实现的最关键问题。首先是因为 P2P 网络上的资源不再在一个单一的服务器上,而是分散在各个对等体上;其次,虽然理想的 P2P 网络各个对等体是平等的,但它们在提供资源能力上是各不相同的[2],搜索方法要考虑对这些对等体是否应该给予不同的优先级和处理方法;P2P 网络上的对等体行为差别很大,它们在 P2P 网络上的活跃时间差别很大,这也导致了网络的动态变化,搜索方法应该适应这种变化。

2 对等网络介绍

2.1 P2P 网络的体系结构

所有的对等网络有一个共同点,那就是实际的数据传输是在资源的请求者与接收者间直接进行的。但是,P2P 的控制层面的实现有不同的方式,据此所有的对等网络的应用可以归结为如下三种体系结构(图 1):

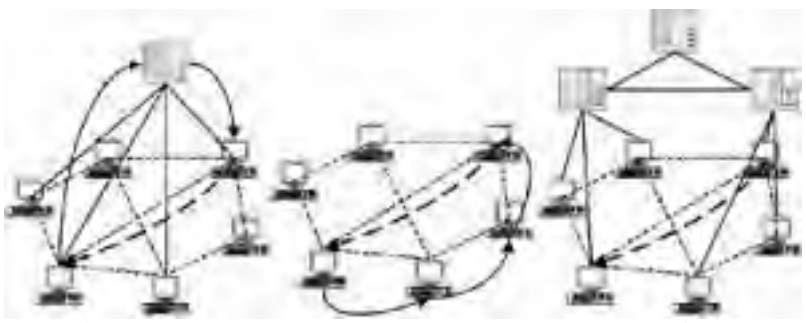
(1) 集中协调式结构。该体系结构使用了一个中央服务器来进行控制操作,所有的客户机登录到中央服务器,服务器负责管理所有客户机的文件和用户数据库。文件的搜索请求发送到服务器,如果找到文件,那么文件就可以从拥有该文件的对等体上直接下载。在这种情况下,服务器维护一个所有对等体的文件的数据库。然而这种结构遇到了文件的版权问题,因为服务器要为传播的所有这些文件的版权负责。这种对等网络结构的特点是有高效的查找过程,查找有较小的开销,查找经过的路径跳数少。

(2) 纯粹分布式结构。纯粹分布式对等网络根本不需要中央服务器,查询以泛洪的方式在网络中传播,收到查询消息的节点再查找本地文件,然后把结果发回给查询节点。因为这

收稿日期: 2004-08-25; 修返日期: 2004-09-28

种结构会产生很多的查询流量, 使得网络有太多的额外开销, 所以该体系结构并不常用。但 FreeNet 仍然使用该结构, 因为这种结构有较好的匿名性, 也消除了单点失败的问题。

(3) 混合式结构。在对等网络的实现上, 混合式结构是最新发展的体系结构, 该结构的目标是实现以上两种结构的优点。通过引入超级对等体(Ultra-peer), 混合结构既有集中协调式结构的特点, 又有纯粹分布式的特点。超级对等体成为一部分对等体的服务器, 就像集中协调式结构一样, 超级节点完成这些对等体的查询工作。而这些超级节点通过纯粹分布式结构连接起来。所以, 混合式的结构在控制层面引入了两层普通对等体通过客户机/服务器模式连接到超级对等体和这些超级节点通过纯粹分布式结构连接到一起。



(a)集中协调式　(b)纯粹分布式　(c)混合式

图 1　P2P 网络体系结构

2.2　P2P 网络的基本搜索方法

P2P 网络系统有三种基本搜索算法:

(1) 集中目录模型(Centralized Directory Model)。Napster[4] 对等网络在 1999 年开始了对等文件共享的革命, 该网络就是采用集中目录模型, 这使得该模型很快流行起来。在这种模型下, 对等体都连到一个目录服务器上, 它们向目录服务器提供共享信息以供其他对等体共享。当目录服务器收到一个查询请求时, 就会从它的目录中匹配该查询, 找出最佳的对等体, 然后文件就在这两个对等体之间进行。这里的最佳是指最小代价、最快和最可用(Available)。这种模型需要一个目录服务器保存所有参与者的信息, 这就导致了可扩展性限制, 因为当请求数目增加时就需要一个更大的服务器, 当用户增加时就需要更大的存储。Napster 的经历表明该模型是非常强壮和高效的。

(2) 泛洪请求模型(Flooded Request Model)。泛洪请求模型是 Gnutella[3] 采用的方法, Gnutella 网络是时下最为流行的对等网络。该方法是在纯粹分布式结构中采用的方法, 不需要向目录服务器报告共享的信息, 而是将请求泛洪到直接相连的邻居, 直到收到响应, 或者达到了最大的泛洪步数。由于这种模型需要很多的网络带宽来进行资源的搜索工作, 因而这种模型的扩展性并不是很好, 但是在像公司这样的小型团体里还是很有效的。为了解决扩展问题, 引入了混合式结构, 把查询请求集中到超级节点, 这样就减少了网络带宽的消耗。

(3) 文件路由模型(Document Routing Model)。FreeNet 网络使用文件路由模型, 这是 P2P 网络最新的搜索方法。这种文件路由模型需要用分布式哈希表(Distributed Hash Tables, DHT), 这是有结构对等网络采用的搜索方法, 这也是有结构和无结构对等网络的根本区别。在这种模型下, 每个对等体都有一个 ID, 每个文件有一个关键字 Key, 当宣告一个关键字为 K_1 的文件时, 先通过哈希映射得到对应的 $K_1 \rightarrow ID_1$, 然后将该文件存到 ID 号为 ID_1 的节点, 文件的存放过程需要将文件路由到该节点 ID_1。反过来, 当查找一个关键字为 K_1 的文件时, 先进行哈希映射得到 $K_1 \rightarrow ID_1$, 然后将该文件从 ID 号为 ID_1 的节点上取到该文件, 从该网络中取文件需要将请求消息路由到 ID_1 节点, 然后文件从 ID_1 节点原路返回。

3　无结构 P2P 网络搜索方法及其改进机制

无结构 P2P 网络采用集中目录模型或泛洪请求模型搜索方法, 有结构 P2P 网络采用文件路由模型搜索方法。有结构对等网络是基于分布式哈希表 DHT 的, 查找需要 O(log*n*) 步, 像 Gnutella 那样的无结构对等网络的查找需要 O(*n*) 步。有结构 P2P 网络有时被称为第二代对等网络, 因为它有更好的可扩展性, 更快的查找速度, 控制层面上需要更少的额外开销。但是, 当前流行的 P2P 网络应用都是采用无结构的模式。为什么选择无结构的模式而不是有结构的模式? 至少有以下三个原因:

(1) P2P 网络是高度动态的, P2P 用户的在线时间是短暂的, 节点变动很快。通过测量 Gnutella 和 Napster 网络表明, 一个节点的平均在线时间是 60 min[2]。对于一个有 100 000 节点的较大对等网络, 这将使得每分钟有 1 600 个节点离开和加入的变动, 节点变动对无结构的对等网络影响很小。然而, 在有结构的网络中, 为了维持系统的正确性和有效性, 每次节点变动需要 O(log*n*) 个修复操作, 如果某个节点是非正常离开, 那么会导致更多的开销, 如果节点变动过多, 这些修复操作就会使那些低带宽的拨号节点过载。

(2) 基于关键字的查找比精确查找更普遍。有结构网络中的分布式哈希表使用的是精确查找, 给定一个文件名, 然后将文件名转换为一个关键字 Key, 然后执行 Lookup(Key) 操作。分布式哈希表不能很好地支持基于关键字的搜索, 就像在 Google 上的搜索那样, 给出一串关键字, 然后找出匹配这些关键字的文件。当前的 P2P 文件共享系统主要提供音频、视频文件共享, 需要基于关键字的查找方法。

(3) 要查找的目标文件大多数有很多复制品, 而不是单一的。DHT 采取精确查找, 只要提供文件名, 即使系统中只有一个符合该文件名的文件, 那么也会搜索到该文件。相比而言, 像 Gnutella 那样的无结构 P2P 网络并不能可靠地找到符合要求的文件, 除非把查询消息发给所有的节点。但这并不是无结构 P2P 网络的弱点, 因为要查找的目标文件大多数有很多复制品, 用户找到文件后就会在本地留下一个拷贝文件, Gnutella 网络能够很容易地找到这些复制了很多次的文件。

无结构的 P2P 网络的流行还有一个重要原因, 那就是有各种各样的针对无结构 P2P 网络搜索方法的改进机制, 这是本文讨论的核心。有各种方法可以控制和减少在分布式无结构 P2P 网络中基于泛洪搜索方法带来大量不必要流量, 这大致可归为三类: 基于改进转发机制的方法、基于缓存的方法以及拓扑结构的优化。这些改进方法会使得无结构 P2P 网络在将来会有更广阔的应用。

3.1　基于改进转发机制的改进

基于转发的搜索改进方法试图改进泛洪算法, 这些方法不将查询发送到所有的邻居, 而是有选择地发送到一些邻居。目的是要能够减少查询消息、减少被访问的节点, 同时又能够得到满意的查询结果。

改进的 BFS(Modified BFS[7]) : Gnutella 的泛洪搜索机制是 P2P 网络上的宽度优先搜索, 这种机制产生了大量的查询消息, 改进 BFS 方法的目的是减少查询消息。在这种机制中, 对等体不是将查询消息发给所有的邻居, 而是随机地选择一部分邻居发送请求消息。选择的邻居的数目占总邻居数的比例是这种机制的参数。它会比 Gnutella 更加有效地搜索对等网络, 但有一定的随机性。

有指导的 BFS(Directed BFS[6]) : 这种方法将查询消息转发给可能提供更多结果的一部分邻居, 或转发给可能更快提供结果的一部分邻居, 这样做既可以减少查询的代价又可以获得高质量的结果。该方法假定历史成绩好坏能够预示将来成绩的好坏。为了能够智能地选择邻居, 每个节点都维护了其邻居的一些简单统计数据, 如该邻居以前返回的结果数或到该邻居的延迟。从这些统计数据中, 可以实施启发式的选择最好的邻居来发送查询, 例如, 选择返回以前查询的结果最多的邻居; 选择以前返回结果中平均路径跳数最小的邻居, 结果的跳数越小就预示着该邻居到有用数据的距离越小; 选择转发查询消息最多的邻居, 转发查询越多就预示着该邻居节点越稳定, 处理能力越强。

智能 BFS(Intelligent BFS[7]) : 为了减少查询消息数、减少查询访问的对等体数, 智能 BFS 将查询发送到最可能得到返回结果的邻居, 这种方法强调更有效地找到目标。当一个对等体收到一个查询消息, 如果接收节点能够提供结果, 那么就将结果文件发送到请求节点, 否则, 接收节点就将查询消息发送到它最可能获得结果的邻居。为了防止查询消息无止境地在 P2P 网络上传播, 查询消息规定了一个最大查询深度。当查询结果返回到请求节点时, 查询结果返回路径上的对等体会记录该结果和提供该结果的对等体。为了决定将请求消息发给哪些邻居对等体, 对等体需要根据查询消息将邻居分级。每个对等需要维护一个邻居描述, 这个邻居描述记录了最近收到的结果, 以及收到该结果的邻居。这种方法的关键是如何选择发送邻居, 如何将邻居分级。为了计算邻居的级别, 查询接收节点将当前的查询和以前的查询进行比较, 找出和以前查询的相似性, 然后把查询继续发送到最可能的节点。该搜索方法的关键是基于相似性的邻居选择方法。下面是一种相似计算方法:

$$P_{simPk}(P_i, q) = \sum_j Q_{sim}(q_j, q)^{\alpha}$$

以上式子描述了请求 q 与对等体 P_i 的相似性, 其中, P_i 是 Pk 的邻居, 求和部分中 q_j 是 P_i 回答的结果, $Q_{sim}(q_j, q)$ 是 q_j 和 q 的相似性。智能 BFS 选择相似性最大邻居来发送请求消息。这种机制要保留以前获得的结果 (q_j, P_i), 当存储结果的空间满载时, 可以使用最近最少使用 LRU 方法更换结果。智能 BFS 搜索机制相对于 Gnutella 会产生更少的查询消息, 更快地找到目标, 查询消息的 TTL 可以比 Gnutella 稍大一些。

反复深入(Iterative Deeping[6]) : 这种方法是逐渐增大 TTL, 分几次来搜索网络, 反复深入搜索过程, 如果搜索到了期望的结果, 那么就中止后续的搜索。这种方法能够自适应地选择 TTL 和搜索深度、能够有效地减少不必要的查询消息。为了实施这种技术, 需要一个系统级的策略来指定反复的深度, 比如反复三次: 第一次搜索到深度 a, 第二次搜索到深度 b, 第三次搜索到深度 c, 那么系统的策略是 $P = \{a, b, c\}$, 而且还需要一个相继搜索的时间间隔 W。在这种策略下, 源节点 S 首先发起一个深度为 a 的宽度优先搜索, 收到搜索请求的节点暂时保存搜索消息, 深度为 a 的所有节点被暂时冻结。同时, 源节点 S 会收到这些节点的响应, 经过时间 W 后, 如果 S 的查询被满足, 那么搜索就结束, 否则, S 就发起下一轮的搜索, 搜索深度为 b。这时, S 发送一个 TTL 为 b 的重发消息, 收到重发消息的节点会向前传递重发消息。如果一个深度为 a 的节点收到重发消息, 那么就解除先前的冻结, 向它的所有邻居发送一个 TTL 为 $b - a$ 的查询消息。每个查询消息都有一个唯一的标志 GUID, 重发消息也包含该 GUID, 使得重发消息和请求消息能够相匹配。当深度为 b 的查询结束后, 就会以以上的方式继续搜索。由于 c 是最后的一轮搜索, 所以深度为 c 的搜索结束后, S 不再发起下一轮搜索, 整个搜索结束。反复深入机制能够减少访问节点和查询消息, 但响应时间会较长。

范围扩张(Expanding Ring[5]) : 这种方法使用多次泛洪, 每次泛洪都比前一次泛洪 TTL 大。范围扩张方法与反复深入方法不同是: 范围扩张中下一轮的泛洪是重新开始的。起初, 查询节点发起一个较小 TTL 的泛洪, 如果查询不成功, 就增大 TTL 再次发起一次泛洪, 重复该过程直到目标被找到。在 P2P 网络上大部分目标都被复制很多次, 范围扩张能够很有效地找到这些被复制了很多次的目标。当目标被复制的比例增加时, 范围扩张机制能够有效地减少需要的 TTL 值, 这种机制解决了 TTL 的适当选择问题, 但增加了重复的查询消息。

随机走动(Random Walks[5]) : 该方法每次将查询消息随机地转发给一个邻居, 直到目标被找到。每个这样的查询消息是一个行走器(Walker) , 这种机制能够将查询额外开销减少一个数量级, 但是响应时间也会增加一个数量级。为了减少响应时间, 可以增加行走器, 一个请求节点同时发出 k 个查询消息, 每个行走器沿着自己的道路走, k 个行走器会大概将延迟减至 $1/k$ 倍。多个行走器的随机走动需要一种中止机制来中止走动, 可以采用两种方法, 即使用 TTL 或核查。使用 TTL 就是当行走器走到一定的步数后就自动中止, 使用核查就是每个行走器间歇地询问查询节点是继续转发查询还是中止当前的走动。核查是一种较好的自适应的中止机制, 需要当前节点和请求节点间交换信息, 因为使用的是固定数目的行走器, 核查不会带来过多的额外开销。进一步的改进是让行走器每走四步核查一次, 这样使得核查的开销和核查的效益达到适当的平衡。随机走动方法的缺点是不稳定性, 成功率依赖于网络的拓扑结构和随机的邻居选择。随机走动也不能从过去的成功和失败中学习到什么, 表现出较高的可变性。

自适应概率搜索(Adaptive Probabilistic Search, APS[8]) : APS 类似于以上的随机走动机制, 要使用 k 个行走器, 只是转发时不是采用随机选择邻居, 而是根据邻居的历史有选择的概率性选择下一个转发节点。在这种机制中, 对等体能够根据以前搜索的反馈有效的指导行走器的搜索, 同时只是保留了邻居的一些信息。APS 搜索有较高的准确性、较低的带宽消耗、能够发现较多的目标, 在动态环境中有较好的健壮性, 这些特点来源于 APS 的学习机制。在 APS 中, 每个节点为每个邻居维护了一些信息, 这些信息包括该节点发出的每个请求和转发该请求的指标, 如果某个节点有 R 个邻居, 对于 N 个目标那么就会需要 $O(R \times N)$ 的存储空间。这些指标标志了从该邻居获得某个目标的响应的可能性, 节点会根据这些指标概率性地将某

个目标的请求转发相应的邻居, APS 会动态地维护这些指标。当某个行走器能够成功地找到目标, 那么在该行走器路径上找到该目标的可能性会增加, 如果行走器不能找到目标, 那么会相应地减少。APS 对这些指标的更新有两种策略, 即乐观的策略和悲观的策略。对于乐观的策略, 行走器搜索时会增加沿途的指标, 如果失败, 那么沿着相反的路径减少沿途的指标; 对于悲观的策略, 行走器搜索时会减少沿途的指标, 如果成功, 那么沿着相反的路径增加沿途的指标。

进一步改进 APS 有两种方法。显然地, 如果有超过 $k/2$ 的行走器是成功的, 我们应该采用乐观的更新策略; 如果有超过 $k/2$ 的行走器是失败的, 我们应该采用悲观的更新策略。一种方法是交换 APS(swapping-APS, s-APS) , 根本的目标是减少更新消息来进一步地减少带宽消耗。该方法中, 可以记录行走器查找某个目标的成功率, 根据成功率来动态地选择更新策略。另一种方法是加权的 APS(weighted-APS, w-APS) , 其想法是对离目标近的节点给予优先选择。该方法根据目标离节点的距离来更新指标和查找的可能性, 离目标越近的节点, 指标的变化越大, 可能性的改变越大。节点离目标的距离可以根据搜索消息得知。

指标路由(Routing Indices[11]) : 在这种方法中, 每个节点维护了邻居的一些指标, 节点根据这些指标将查询请求转发给最好的邻居。这些指标是一个数据结构, 当收到一个查询时, 能够计算出邻居节点的好坏, 得到一些较好的节点。这些指标可以是邻居能够返回的文件总数, 同时将文件分类, 指标也可以是邻居能够返回某类文件的数目。这种方法不适合动态变化很快的对等网络。

非转发机制(GUESS[9]) : 转发机制有一个根本的缺点, 那就是搜索时产生大量的查询消息转发的额外开销, 缺少对搜索过程的集中控制, 非转发机制 GUESS 是资源定为的另一种可行的选择, 这比转发机制的效率提高了一个数量级[9] 。在采用 GUESS 协议的 P2P 网络中, 对等体直接向被查询的节点发送查询消息, 而不是依赖其他节点将查询消息路由到被查询的节点。网络中的对等体要维护两个缓存, 连接缓存和查询缓存。缓存的每一项有一个指向其他对等体的指针。连接缓存相当于 Gnutella 网络中的邻居对等体组成的列表, 所有连接缓存中的对等体可以被看着是该对等体的邻居。对等体并不和它的邻居用 TCP 建立连接, 而是通过 UDP 与邻居通信, 所以此时的邻居关系是单向的, 如果 Q 在 P 的连接缓存中, 那么 P 不一定在 Q 连接缓存中。由于 UDP 不在主机间建立连接, 有可能某个对等体没有被告知而它的邻居就离开了。查询缓存是一个临时申请的空间, 保存着指向其他对等体的指针, 用来提高查询的性能。某个对等体 P 的连接缓存和查询缓存有以下数据格式:

{ IP address of Q, TS, NumFiles, NumRes}

以上条目指向节点 Q, TS 是节点 P 最后一次和 Q 交互的时间戳; NumFiles 是节点 Q 拥有的文件数目, NumRes 是 Q 最后返回给 P 的结果数目。

采用 GUESS 的 P2P 网络中的对等体也要维护节点的状态。首先因为 P2P 网络中的节点的活跃时间很短, 对等体必须定期刷新它的连接缓存。对等体通过定期地向它的邻居发送 Ping 消息来维护其连接缓存, 如果邻居没有回应, 那么对等体就将该邻居从连接缓存中清除; 如果收到邻居的回应, 那么就更新对应条目的 TS 值。当收到一个 Ping 消息, 对等体就回应一个 Pong 消息, 该消息中包含一些从自己连接缓存中提取的 IP 地址, 这样做是让对等体能够共享连接缓存中的条目, 得以发现更加多产的对等体。当一个对等体首次加入对等网络时, 需要使用一个介绍协议将自己的 IP 地址加入其他节点的连接缓存中, 当节点 P 首次与节点 Q 交互时, Q 会以概率 p 将 P 的 IP 加入自己的连接缓存。

GUESS 的根本特征是: 查询消息不是通过泛洪机制广播到其他对等体, 对等体重复地向连接缓存中的对等体发送单播查询消息。对等体会探测(Probe) 足够多的节点来获得足够多的结果, 探测会以规定的方式和顺序发出。对等体发出探测后, 要么收到回答, 要么等待一段时间后超时。很多情况下, 对等体需要探测很多的节点, 为此, 当一个节点被探测后, 它在返回结果的同时也返回一个 Pong 消息, 请求节点收到这个 Pong 消息后就把其中的条目放入自己的查询缓存中, 要探测的对等体要么在连接缓存中, 要么在查询缓存。查询缓存是临时性的, 查询结束后就被清除。

3.2　基于缓存方法的改进

缓存方法包括索引缓存和内容缓存。通过在 ISP 边界收集分析 P2P 数据报, 发现 P2P 网络的通信在 ISP 边缘是相当重复的, 其缓存高达 67% 字节命中率, 而 Web 缓存在 ISP 边界只有 30% ～60% 字节命中率。

本地索引(Local Indices[6]) : 在本地索引的方法中, 每个节点维护了离自己的距离不超过 r 步的节点的数据索引, 当该节点收到查询消息时, 它可以为离自己半径为 r 范围内的所有节点处理查询, 这里的 r 是索引半径, 是系统范围内的参数。通过这种方法, 可以将对查询的处理放到很少的节点上进行, 既能返回需要的结果, 又将代价大大降低了。当 r 很小时, 每个节点需要索引的元数据(Metadata) 很小, 所以一个较小的 r 能够被像 Gnutella 那样松散控制的 P2P 网络接收。

本地索引技术工作过程如下: 一个系统范围内的策略规定了查询消息的处理在哪些深度上进行, 通常在深度为 $2r+1$ 的节点进行。深度不在策略中的节点只是简单地将查询消息转发给邻居。例如, 如果搜索策略 $P=\{1,5\}$, 那么离查询节点距离为 1 和 5 的节点会处理查询消息, 其他节点只是简单地把查询转发给下一深度的节点。为了在每个节点上维护索引, 需要在节点加入、离开和更新时采取一些额外的步骤。当 X 节点加入网络时, 它会广播发送一个 TTL 为 r 的加入消息, 加入消息中包含有节点 X 拥有的数据。当某个节点收到 X 发送的加入消息, 它也会直接返回一个加入消息给 X。这样, 双方都把对方元数据加到自己的索引里。当一个节点离开时, 其他索引该节点的对等体会把该节点的元数据删除。当一个用户更新数据时, 它会把更新发到半径为 r 的范围内, 收到更新的节点会更新相应的索引。

DRLP 协议(Distributed Resource Location Protocol [12]) : 这种方法是一种索引缓存方法。如果节点不知道要查询的数据位置, 那么就将查询消息以一定的概率发送到各个邻居, 如果得到某个查询的结果, 那么结果会以查询相反的方向返回给请求节点, 并在返回路径的每个节点上记录目标数据的位置, 在以后的查询中, 保留了其他节点的目标数据的索引节点可以直接与请求节点联系, 通告目标数据的位置。

内容缓存(Content Caching[13]) : 它是将文件的内容直接复制到节点上, 这样做会减少某些受欢迎的节点的负担, 减少响应时间。P2P 网络的数据流量占互联网上流量的很大部分, 仅 Kazaa 对等网络的流量就占了总流量的 36.9% , 而 WWW 不过占总流量的 14.3% 。通过模拟, 文献[13] 指出, 如果在 Kazaa 对等网络上实施内容缓存, 入口方向可以获得 35% 的字节命中率, 出口方向可以获得 85% 的字节命中率。P2P 网络上一些非常大的目标占了流量的很大部分, 前 300 个目标总共有 180GB, 请求这些目标的流量高达 5.64TB, 对这些较大目标进行内容缓存会使得通信代价得到进一步的共享。

3.3 基于拓扑结构的优化

GIA[14] 中的拓扑自适应: GIA(Gianduia) 网络采用混合式体系结构, 考虑到对等网络上各个对等体能力的不同, GIA 动态地选择超级节点, 动态地自适应地形成拓扑结构。GIA 网络比 Gnutella 网络性能提高了三到五倍, 在保持网络简单性的同时, 更好地提高了可扩展性。GIA 建立在以前的工作之上, 采用了以下改进方法: ①动态的拓扑适应, 使得大部分节点离高能力的节点很近, 使得连接很多邻居的节点的能力的确很强; ②主动的流控制, 根据处理能力分配令牌进行流控制, 能够区分不同能力; ③一步复制, 每个节点都保存了其邻居的数据指针, 这样使得高能力的节点能够回答更多的查询; ④随机走动搜索方法, 这种方法避免了泛洪搜索带来的过多查询。

实现 GIA 的关键是拓扑自适应。在下面的算法中, 节点的能力指的是节点每秒能够处理的查询数目, 系统规定了节点能够建立连接的最大数目 max_nbrs, 每个节点都有一个备用邻居节点缓存。每个节点有一个满意度 S, $0 \le S \le 1$, 只要节点不是完全满意($S = 1$) , 它就会试图与备用节点缓存中的节点建立连接。设节点 Y 是节点 X 的备用节点, X 试图与 Y 建立连接, 那么建立连接的过程采用如下算法:

```
Let Ci represent capacity of node i
if num_nbrsX + 1 ≤ max_nbrs then { we have room}
   ACCEPT Y; return
{ we need to drop a neighbor}
subset← i, i∈ nbrsX such that Ci≤ CY
if no such neighbors exist then
   REJECT Y; return
cansidate Z ← highest-degree neighbor from subset
if ( CY > max( Ci, i∈ nbrsX) ) { Y has higher capacity}
or ( num_nbrsZ > num_nbrsY + H) { Y has fewer nbrs}
then DROP Z; ACCEPT Y
else
REJECT Y
{ H represents a hysteresis factor. }
```

以上拓扑自适应算法一直继续, 并试图达到一个稳定的状态。

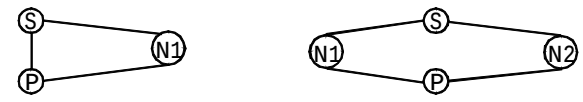
位置相关拓扑匹配(Location-aware Topology Matching, LTM[15]) : 对等网络中对等体任意选择邻居的做法带来了严重的问题, 那就是物理网络与逻辑网络拓扑结构的不一致, 这使得网络上的消息要穿越不必要的路径, LTM 方法试图动态地建立与物理网络一致的逻辑覆盖网, 消除这种逻辑拓扑的不一致。模拟实验结果[15] 显示, LTM 可以减少 75% 的流量开销和 65% 的查询响应时间。

LTM 使用两个节点间的延迟作为代价, 通过泛洪 TTL = 2 的消息, 每个节点可以计算离自己半径为 2 的范围内的节点的代价。LTM 方法的第一步是使用 TTL-2 来发现自己半径为 1 或 2 的节点代价。第二步是剪掉高代价的连接, 设 $d(i, S, v)$, 表示收到由节点 S 发出的 ID 为 i, TTL 为 v 的探测器, 分以下两种情况讨论操作过程(图 2) 。

(1) 如果存在图 2(a) 那样的子图, 而且 SP 或 N1P 代价最大, 那么就将连接 SP 或 N1P 放入节点 P 的将要断开的列表中; 如果 SN1 最大, 那么 SN1 放入 $N1$ 的将要断开的节点列表中;

(2) 如果存在图 2(b) 那样的子图, N1P 或 N2P 最大, 那么就将 N1P 或 N2P 放入 P 的将要断开的节点列表中;

(3) 如果节点 P 只收到一个 $d(i, S, 0)$, 那么当 SP 最大时 P 不进行任何操作, 否则, 就创建新的连接 SP。



(a)SP>SN1 或 N1P>SN1　　(b)N1P 或 N2P 是最大代价

图 2 低效的拓扑

4 结论

对等网络是近年来研究的热点, 是未来 Internet 的关键技术。对等网络的可用性依赖于对数据的高效查找和提取方法, 如何高效地搜索 P2P 网络上的资源是 P2P 网络实现的最关键的问题。本文介绍了对等网络上资源定位、资源搜索的各种方法以及方法的改进。这些改进方法大大减少了查询需要发送的消息, 减少了需要访问的节点数。本文介绍的 P2P 网络的搜索方法及其改进方法会对 P2P 网络的研究提供有用的参考和启发, 会对 P2P 网络实现提供指导。

参考文献:

[1] ejan S Milojicic, *et al*. Peer-to-Peer Computing[R] . Hewlett-Packard Company, 2002.

[2] S Saroiu, P K Gummadi, S D Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems[C] . Proceedings of Multimedia Computing and Networking, 2002.

[3] Gnutella, Gnutella Protocol[EB/OL] . http: / /gnutella. wego. com/.

[4] Napster Website[EB/OL] . http: / /www. napster. com/.

[5] Q Lv, P Cao, F Cohen, *et al*. Search and Replication in Unstructured Peer-to-Peer Networks[C] . ICS, 2002.

[6] B Yang, H Garcia-Molina. Improving Search in Peer-to-Peer Networks[C] . ICDCS, 2002.

[7] V Kalogeraki, D Gunopulos, D Zeinalipour-Yazti. A Local Search Mechanism for Peer-to-Peer Networks[C] . CIKM, 2002.

[8] D Tsoumakos, N Roussopoulos. Adaptive Probabilistic Search (APS) for Peer-to-Peer Networks[R] . Technical Report CS-TR-4451, Un. of Maryland, 2003.

[9] B Yang, Patrick Vinograd. Evaluating GUESS and Non-Forwarding Peer-to-Peer Search[Z] . 2003

[10] Gnutella2 Website[EB/OL] . http: / /www. gnutella2. com/.

[11] A Crespo, H Garcia-Molina. Routing Indices for Peer-to-Peer Systems [C] . ICDCS, 2002.

[12] D Tsoumakos, *et al*. A Comparison of Peer-to-Peer Search Methods [J] . WebDB, 2002.

[13] Stefan Saroiu, Krishna P Gummadi. An Analysis of Internet Content Delivery Systems[Z] . 2002.

[14] Y Chawathe, S Rantnasamy. Making Gnutella-like P2P System Scalable[C] . SIGCOMM, 2003.

[15] Yunhao Liu, *et al*. Location-Aware Topology Matching in P2P Systems [C] . Hong Kong: Proceedings of IEEE INFOCOM, 2004.

作者简介:

夏启志(1978-) , 男, 湖北黄冈人, 硕士研究生, 主要研究方向为下一代互联网、网络协议测试; 谢高岗(1974-) , 男, 浙江衢州人, 副研究员, 博士, 主要研究方向为网络测试、监控与管理。