基于移动 agent 的无结构 P2P 网络拓扑重连方法研究*

谷培影，沈项军，蒋中秋
（江苏大学 计算机科学与通信工程学院，江苏 镇江 212013）

摘　要：为了提高无结构 P2P 网络中资源查找的效率，同时避免在资源查找过程中出现拥塞，提出了一种基于移动 agent 的网络拓扑重连方法。网络节点定期进行拓扑重连，同时通过收集其邻居节点的处理能力以及连通性等信息，指导移动 agent 有目的地在网络迁移，从而使移动 agent 及时发现网络节点上的拥塞，并使用拓扑优化机制降低节点上的负载。实验证明该方法能优化网络的拓扑结构，避免网络查询过程中发生拥塞，提高资源查找的效率，同时适应网络查询负载的动态变化。

关键词：P2P 网络；资源查找；移动 agent；拓扑重连
中图分类号：TP393　　**文献标志码**：A　　**文章编号**：1001-3695(2013)01-0234-04
doi:10.3969/j.issn.1001-3695.2013.01.060

Topology adaptive approach based on mobile agent in unstructured P2P networks

GU Pei-ying, SHEN Xiang-jun, JIANG Zhong-qiu
(*School of Computer Science & Telecommunication Engineering, Jiangsu University, Zhenjiang Jiangsu* 212013, *China*)

Abstract: To enhance the efficiency of resource location and avoid congestion in unstructured P2P (peer to peer) networks, this paper proposed a mobile agent based topology adaptation approach. Peers in the network launched reconnection mechanism periodically, and collected their neighbor peers' information such as processing capacity and connectedness to direct mobile agent to migrate in the network on purpose. Thus mobile agent could find congestions in peers effectively and used the topology optimization mechanism to decrease the query loads in those peers. Experimental results show that the algorithm can improve networks topology structure, avoid congestions, increase the efficiency of resource location, and adapt to the dynamic change of query loads.

Key words: P2P network; resource location; mobile agent; topology adaptation

0　引言

近年来，随着互联网的发展，对等(P2P)网络已逐渐成为一种重要的网络模式，这是因为 P2P 网络具有灵活性、可扩展性、鲁棒性等特点，其与传统的 client/server(C/S)结构存在很大的区别[1]。如传统 C/S 模式下所有的资源都存储在少量的服务器上，而大量的客户机只用来下载服务器上的资源。由于带宽和处理能力的限制，网络的规模受到了很大的限制，而且一旦服务器受到攻击会对整个网络产生严重的影响。P2P 网络可以将每个节点都看做服务器，同时又看做客户机，所有节点共享网络中的数据、存储空间、计算能力、带宽等资源。目前 P2P 网络已被广泛应用于互联网上，出现了文件共享、海量存储、流媒体等应用[2]。

现有的 P2P 网络按照网络的拓扑连接方式主要分为有结构和无结构 P2P 网络[3]。由于无结构的 P2P 网络节点可以自主地选择网络的邻居节点进而形成自由的拓扑连接结构，因此无结构 P2P 网络具有网络节点扩展性强、资源负载能力大等特点，出现了如 Gnutella、KaZaA 等诸多网络协议及对等计算、文件共享等应用[2]。然而由于无结构 P2P 网络的分布式特性，网络中的节点只知道其邻居节点的资源，却并不知道网络中其他节点上资源的分布情况，使得资源发现与搜索等成为无结构 P2P 网络的核心问题[4]。为了解决 P2P 网络中资源定位问题，出现了如洪泛法搜索[5]、random walks 搜索算法[6]、adaptive probabilistic search(APS)[7]等。以上搜索算法可以有效地减少网络中查询信息的转发量，降低了网络的负载。然而由于 P2P 网络节点的异构性，在网络负载较重时，一些处理能力较弱的节点会发生拥塞；处理能力较强的节点却有很多时间都是空闲的，节点的拥塞现象会严重影响到搜索算法的性能。如何平衡网络的负载，充分利用网络节点的处理能力，引起了学术界和工业界广泛、深入的研究。

目前的负载均衡研究按照是否需要改变网络拓扑结构分为以下两类：

a)研究通过动态改变网络拓扑结构来实现网络的负载均衡策略。例如，Gia 系统[8]通过引入一个基于节点满意水平的机制进行拓扑调整，节点的满意水平是用来衡量节点的处理能力和所有邻居节点的总处理能力差距的，节点根据这个参数确定其拓扑结构是否需要调整以及调整的频率。DANTE[9]提出了另一种自适应调整网络拓扑结构的方法，即每个节点定期调整自身的拓扑连接。该方法在不同的负载状况下可以形成不同的拓扑结构，但是在形成稳定的拓扑结构后如果网络负载突

收稿日期：2012-05-19；**修回日期**：2012-06-28　**基金项目**：国家自然科学基金资助项目(61005017)；江苏省高校自然科学基础研究项目(10KJB520005)；江苏大学高级人才资助项目(1283000347)

作者简介：谷培影(1989-)，女，河南禹州人，硕士研究生，主要研究方向为 P2P 网络(gupeiying1@126.com)；沈项军(1977-)，男，江苏镇江人，软件工程系副主任，副教授，博士，主要研究方向为模式识别、P2P 网络、分布式多媒体计算；蒋中秋(1979-)，女，河南人，讲师，博士研究生，主要研究方向为 P2P 网络、无线传感器网络.

然加重,在中心节点可能会引发严重的拥塞。

b)研究在不改变网络拓扑结构的基础上来实现负载均衡。该类方法通过把拥塞节点的多余查询消息有目的地转移到有多余处理能力的节点来实现负载均衡。在这类研究中最关键的问题是如何发现拥塞,其中一种方法是基于消息传递模式的,即拥塞节点向周围节点发出消息告知拥塞,这种方式存在通信延迟且会加重网络负载[10]。由于移动 agent 的高灵活性、高效性、低负载、低通信延迟、高异步性等特点,使它逐渐被应用到了拥塞感知方法中。文献[11]提出了一种基于拥塞感知的移动 agent 路由协议,移动 agent 随机地从邻居节点中选择下一个访问节点;Li 等人[12]提出了一种基于移动 agent 的 P2P 网络负载均衡方法,移动 agent 通过轮流访问网络中的所有节点,来寻找拥塞节点。

综上,现有的拓扑重连方法在形成相对稳定的拓扑结构后,如果网络负载突然加重,中心节点容易发生拥塞,进而影响系统的性能。为此本文提出一种基于移动 agent 的无结构 P2P 网络拓扑重连方法。

1　无结构 P2P 网络拓扑调整

无结构 P2P 网络的节点拓扑如图 1(a)所示,网络中所有节点都是随机连接的,这种结构易于扩展且不容易产生拥塞,但是资源搜索的效率较低。为达到既能高效地查找资源,又能有效避免拥塞的目的,本文采用在搜索过程中使拓扑结构逐渐进化的方法,让 P2P 网络由随机的拓扑结构逐渐进化到如图 1(b)所示的结构。该结构的好处是通过大量局部中心节点实现资源查找的高效率,同时大量局部中心节点也能避免网络在查询过程中形成拥塞。为了实现此目的,本文提出一种基于移动 agent 的拓扑重连方法,该方法分两部分进行:a)每个节点都定期发起拓扑重连,使自身尽量与处理能力强的节点连接,从而形成网络局部中心节点;b)利用移动 agent 在局部中心节点上不断迁移,以及时发现拥塞并进行调整,实现拓扑结构的优化。

1.1　基于移动 agent 的网络拓扑优化

为了避免局部中心节点发生拥塞,本文利用移动 agent 来尽早地发现网络中的拥塞,并进行网络的拓扑优化调整。因此移动 agent 在网络中的迁移策略就显得异常重要。由于一般在中心节点的拥塞对整个网络的性能影响比较大,而这些节点都有一个特点,即处理能力比较强而且连接度比较高,因而本文引入了一个指标,用其指导移动 agent 尽可能地与连通度高且处理能力强的节点迁移。每个节点都维护一个邻居表,如表 1 所示,保存所有邻居节点的连通度和处理能力,移动 agent 根据该表选择下一个访问节点。节点 i 的连通度的计算方法为

$$\chi(i,k_c)=\sum_{h=1}^{k_c}\frac{N(i,h)}{h^{\delta}} \tag{1}$$

其中:$\chi(i,k_c)$表示节点 i 与 k_c 半径内的所有节点的连通程度;$N(i,h)$是与节点 i 相距 h 跳的节点个数;δ 是控制因数;h^{δ} 用来控制不同距离节点对节点连通度影响的比权重。$\chi(i,k_c)$越大节点 i 可能接收到的查询消息越多。

表 1　邻居表

指标	P_1	P_2	P_3	P_4	P_5
处理能力	1	0.1	10	100	100
连通度	6	3	7.5	14	12.5

移动 agent 利用当前节点的邻居节点对它的吸引力来确定迁移到哪个节点。假设节点 i 是移动 agent 当前所在节点的一个邻居节点,则该节点对移动 agent 的吸引力 A_i 计算如下:

$$A_i=\chi(i,k_c)\times C_i \tag{2}$$

其中:$\chi(i,k_c)$是节点 i 的连通度,C_i 是节点 i 的处理能力。

移动 agent 计算邻居表内所有节点的吸引力,然后选择最近没有访问过且对其吸引力最强的节点作为一个访问节点。防止移动 agent 陷入一个小循环,同时保证移动 agent 不断地向连通度高的节点迁移,以及时发现拥塞并进行处理。

图 2 是移动 agent 路由选择的一个示意图。假设移动 agent 在节点 P,节点 P 的邻居表如表 1 所示。移动 agent 根据邻居表通过计算得出节点 P 的五个邻居节点,按照它们对移动 agent 的吸引力从大到小排序依次为(P_4,P_5,P_3,P_1,P_2),移动 agent 最近访问的三个节点为(P_4,P_3,P_6)。显然 P_4 对移动 agent 的吸引力最强,但 P_4 在移动 agent 最近经过节点列表中,根据判断移动 agent 决定迁移到节点 P_5。

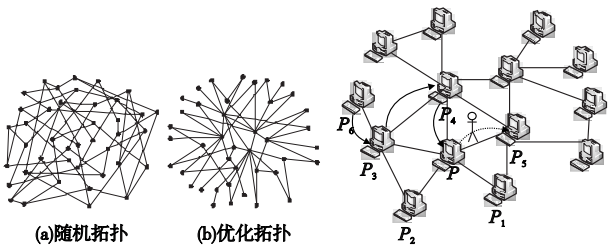

图1　网络拓扑图

图2　移动agent的路由选择示例

移动 agent 在迁移的过程中会判断经过的节点是否发生拥塞。本文利用一个消息在一个节点的等待时间来确定节点的拥塞程度(CL)[13],节点 i 在时刻 t 的拥塞程度为

$$CL_i(t)=\frac{1+Q_i(t)}{C_i} \tag{3}$$

其中:C_i 表示节点 i 在单位时间可以处理的消息数,当一个节点正在处理消息时标记为“busy”,这时转发到这个节点的消息都放在该节点的输入缓存队列中;$Q_i(t)$表示在时刻 t 节点 i 的输入缓存队列中的消息数。式(3)计算的是一个消息转发到节点 i 到处理完毕需要的等待时间。

根据 CL 可以确定节点的负载状态。本文中设置节点有三个负载状态,即轻负载、正常负载和过载。当节点过载时就认为该节点发生了拥塞。

$$\begin{cases}\text{flag}_i=-1 & CL_i\leqslant B_{\text{thred}}\\ \text{flag}_i=0 & B_{\text{thred}}<CL_i<U_{\text{thred}}\\ \text{flag}_i=1 & CL_i\geqslant U_{\text{thred}}\end{cases} \tag{4}$$

其中:B_{thred}和 U_{thred}分别是确定节点状态的两个阈值,flag = -1,0,1 分别对应轻负载、正常负载和过载三个状态。当 flag = -1 时节点接受新的连接请求;flag = 0 时,如果该节点的上一个状态是 flag = 1,则不接受连接请求,否则接受连接请求;flag = 1 时节点不接受新的连接请求。

当移动 agent 发现一个节点拥塞时,指导该节点主动断开与一些节点的连接(称这些节点为待重连节点),移动 agent 派遣出一个子 agent 负责把这些待重连的节点重连到有富余处理能力的节点。子 agent 以广度优先遍历的方式,寻找有多余处理能力的节点(即轻负载的节点),并将待重连节点与之建立连接,等所有的待重连节点都重连完毕,子移动 agent 主动退出。

应用上述方法,基于移动 agent 的拓扑优化的流程如图 3 所示。

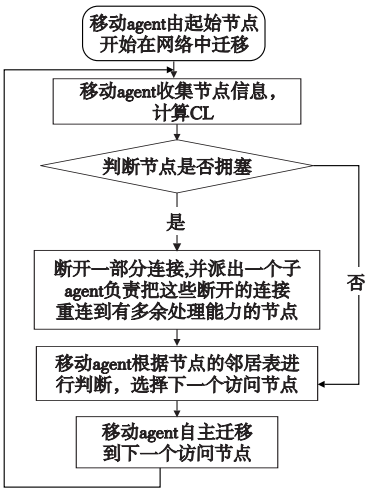

图3 移动agent拓扑优化流程

1.2 节点定期发起的拓扑重连

为了充分利用处理能力强的节点的处理能力,避免在处理能力弱的节点发生拥塞,每个节点都定期进行拓扑重连。为了实现节点的定期拓扑重连,每个节点定期发出一个节点收集消息,该消息以 random walk 的方式在网络中收集参加拓扑重连的候选节点集合 C,用 TTL 限制消息传送路径的长度。节点搜集完毕后,将候选节点返回给消息发起节点,开始进行拓扑重连。

假设发起拓扑重连的节点为 p,节点 p 从候选节点集中选择节点进行重连,直到节点 p 已经与 m 个候选节点建立连接或者剩余候选节点的处理能力小于所有邻居节点的处理能力时,节点 p 的一次拓扑重连过程结束。节点 p 进行拓扑重连的过程如算法 1 所示。在算法 1 中,节点 p 与其中一个候选节点的连接过程如下:节点 p 从邻居节点中选择一个连接度大于 1 且处理能力最弱的节点(假设为 n),从候选节点集中选择处理能力最强的节点(假设为 c),如果节点 c 的处理能力大于节点 n 的处理能力,则节点 p 断开与 n 的连接并与节点 c 建立连接;否则用同样的方法选择下一个候选节点进行拓扑重连。

算法 1 节点得到候选节点后的拓扑重连过程

```
//C 是候选节点集
//Neigh 是邻居节点集
//Capa[ ]是节点的处理能力
//m 是最多重连的节点数
1 i←1
2 while i < k://最多与 k 个候选节点重连
3   n←处理能力最小且连接度大于 1 的邻居节点
4   while 1://选择候选节点
5     c←处理能力最强的候选节点
6     C. remove(c)
7     if Capa[n] > = Capa[c]:
8       break
9     else:
10      节点向 c 发出重连请求
11      if c 接受重连请求:
12        N. remove(n) //与节点 n 断开连接
13        N. append(c) //与节点 c 建立连接
14        i←i + 1
15        break
16    if Capa[n] > = Capa[c] or |C| =0:
17      break
18  重连结束
```

单纯依靠节点定期的拓扑重连,最终会形成以几个处理能力最强的节点为中心的集中式网络,这样资源搜索的效率虽然很高,但是中心节点可能会发生严重的拥塞现象。为了避免出现这种情况,在节点拓扑重连的过程中加入了前面介绍的基于移动 agent 的拓扑优化方法。当发现拥塞时,移动 agent 就指导节点进行进一步的拓扑重连,从而避免拥塞的发生,提高系统的性能。

2 模拟实验

本实验在 Windows 平台下利用 Python 2.6 进行程序的编写及运行,根据 Gnutella 协议设计网络,该网络包含 10 000 个节点,移动 agent 由加入网络的第一个节点生成,并伴随着整个网络一直存在。网络中每个节点的初始邻居数都大致相同,约为 10 个,节点的处理能力用 c_i 表示,假设节点 i 中有 n 个数据,节点 i 处理一个查询信息需要的时间 $t_{proc} = n/C_i$,采用从 Gnutella 网络测量[14]中获得的节点能力分布情况(见表 2)来模拟节点能力的异构性。系统的负载主要是由节点对网络资源的查询引起的,为每个节点分配一个查询产生速率 q_i,用来表示节点 i 单位时间发起的查询个数,假设所有节点的初始查询产生速率相同,为每秒 0.2 个,则在本系统中每秒会发起 2 000 个查询消息。实验中确定节点状态的两个阈值 B_{thred} 和 U_{thred} 分别设置为 0.05 和 0.1。

实验中有 5 000 个不同的数据,每个数据在网络中有相同数目的副本,一个节点上的每个数据都是不同的。用回应率来表示一个查询可以得到网络中多少节点的回应,假设回应率为 0.01,那么在 10 000 个节点的系统中,有 100 个节点可能会回应查询。本实验中回应率设置为 0.01,即每个数据随机分布在 100 个不同的节点上;采用 random walks 进行搜索,每个查询产生 5 个 walkers。为了保证查询的高成功率,TTL 设置为 50,节点收集消息的 TTL 设置为 30,一个节点每次最多与 5 个节点重连。计算连通度的两个参数 k_c 和 δ 分别为 2 和 1。

表 2 节点处理能力分布表

节点比例/%	节点能力 C_i	节点比例/%	节点能力 C_i
20	0.1	4.9	100
45	1	0.1	1 000
30	10		

为了了解拓扑结构的改变,采用聚集因数[15]来衡量网络的聚集程度。定义如下:$G = (V,E)$ 是一个无向图,顶点 V 代表网络中的节点,边 $E \subseteq V \times V$ 是节点间的连接。用 N_i 表示节点 $i(i \in V)$ 的邻居节点集,$k_i = |N_i|$。节点 i 的聚集因数 CC_i 被定义为节点 i 的邻居节点间的连接数除以最大可能连接数:

$$CC_i = \frac{|(N_i \times N_i) \cap E|}{k_i(k_i - 1)} \tag{5}$$

整个网络的聚集因数为

$$CC = \frac{1}{|V|}\sum_{i \in V} CC_i \tag{6}$$

显然 $0 \leqslant CC \leqslant 1$,$CC$ 越接近 1,网络的拓扑结构越集中化;反之,CC 越接近 0,网络的拓扑结构越随机化。图 4 是随着网络拓扑结构的改变聚集因数的变化情况,可见初始的 CC 值是非常低的;随着时间推移,网络拓扑结构改变,CC 值越来越大,说明随着拓扑结构的改变网络拓扑结构越来越集中,但 CC 值没有变得更大,说明生成的是有多个局部中心的拓扑结构。

图 5 给出了随着时间的推移网络中不同处理能力节点平均节点度(邻居节点的数目)的变化情况。实验中以 100 s 为一个时间段来观察节点度的变化,可以看出开始有大量的节点连接到处理能力最强的节点,但很快就达到一个稳定的状态,且处理能力越强的节点连通度越高,说明处理能力越强的节点可以分担更多的查询负载,能够充分利用网络节点的处理能力。

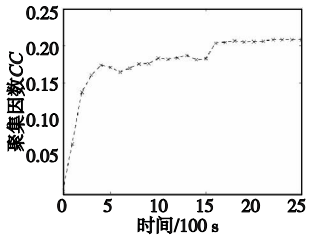

图4　网络聚集因数随时间的变化

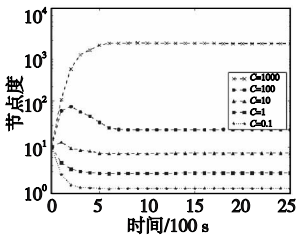

图5　不同处理能力节点的平均连接度变化

图 6 给出了网络中查询成功需要的平均跳数的变化情况。查询平均跳数反映了网络的资源查找效率,查询所需的跳数越少,资源查找效率越高。实验中对每 100 s 时间段上网络中的成功查询所需要的跳数求均值得到如图 6 所示结果,随着对网络拓扑结构的优化,查询需要的平均跳数逐渐减小。从图 6 中可以看出,拓扑优化后需要的跳数比开始时降低了很多,可见基于移动 agent 的拓扑优化方法大大提高了无结构网络的搜索性能。

通过统计移动 agent 的每 20 000 次访问中对不同能力节点的访问次数,并对其计算平均值,图 7 给出移动 agent 对不同能力网络节点的平均访问次数。开始时所有节点的节点度都是大致相同的,因而移动 agent 对各节点的平均访问次数相差不大。经过拓扑重连处理后,能力强的节点的连通度越来越高,移动 agent 对这些节点的访问频率也明显增高了。说明移动 agent 根据本文设计的指标可以在中心节点间连续迁移,这样移动 agent 就能及时地发现拥塞并进行相应的处理。

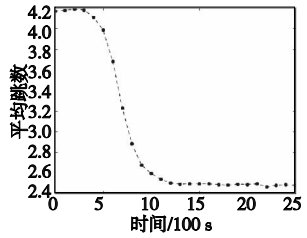

图6　查询成功需要的平均跳数

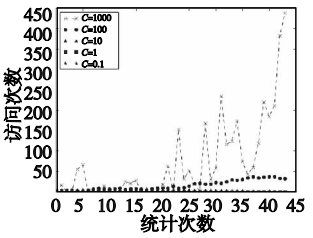

图7　移动agent对不同能力节点的平均访问次数

图 4 ~ 7 的实验证明,应用本文方法,整个网络形成了如图 1(b)所示的局部中心网络,可以有效提高无结构 P2P 网络资源查找的效率。同时,为验证本文方法对网络负载动态变化的适应性,将本文方法与 DANTE[9] 系统在查询速率增大的情况下,对网络拓扑结构及资源查找时间的变化进行了对比分析,如图 8 和 9 所示。

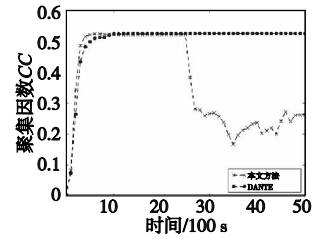

图8　节点的查询发起速率改变时对拓扑结构的影响

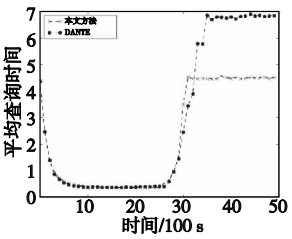

图9　节点查询发起速率改变对资源查找时间的影响

刚开始时所有节点的初始查询发起速率为 0.1(即每个节点每隔 10 s 发起一个查询请求),在图中所示的虚拟时间为 25 个单位时查询发起速率突然增加到 1.0(即节点每 1 s 发起一个查询请求)。由图 8 可以看出,经过一段时间两种方法都形成相对稳定的拓扑结构。节点的查询发起速率增加后在 DANTE 方法下 *CC* 值没有变化,而在本文提出的方法中 *CC* 值明显减小。这是因为在节点查询发起速率突然增大时,由于处理能力强的节点拥有过多的邻居,因而接收到过多的查询请求而产生了拥塞,移动 agent 很快发现这些节点上的拥塞并采用了相应的拓扑优化方法减低负载。由图 9 可以看出,在网络负载加重时两种方法查找资源需要的平均时间都增大了,但本文方法需要的时间明显低于 DANTE,说明本文方法可以适应网络负载的动态变化。

3　结束语

本文提出了一种基于移动 agent 的无结构 P2P 网络拓扑重连方法,节点定期进行拓扑重连,以形成中心式的拓扑,同时移动 agent 在网络中根据所在节点的邻居表的指引在连通度高且处理能力强的节点间连续迁移,寻找拥塞节点进行拓扑结构的进一步优化。该方法通过改变无结构 P2P 网络的拓扑结构,实现了负载均衡,减小了 random walks 搜索算法搜索无结构网络所需路径的长度,提高了整个网络的资源搜索的性能。

参考文献:

[1] 方启明,杨广文,武永卫,等. 基于 P2P 的 Web 搜索技术综述[J]. 软件学报,2008,19(10):2706-2719.

[2] 王学龙,张璟. P2P 关键技术研究综述[J]. 计算机应用研究,2010,27(3):801-805.

[3] POUREBRAHIMI B, BERTELS K, VASSILIADIS S. A survey of peer-to-peer networks[C]//Proc of the 16th Annual Workshop on Circuits, Systems and Signal Processing. 2005.

[4] THAMPI S M, SEKARAN K C. Survey of search and replication schemes in unstructured P2P networks [J]. Network Protocols and Algorithms,2010,2(1):93-131.

[5] THEOTOKIS S. A survey of peer-to-peer file sharing technologies [EB/OL]. http://www. eltrun. aueb. gr/whitepapers/p2p _2002. pdf.

[6] LV Qin, CAO Pei, COHEN E, *et al*. Search and replication in unstructured peer-to-peer networks [C]//Proc of the 16th International Conference on Supercomputing. New York:ACM Press,2002:84-95.

[7] TSOUMAKOS D, ROUSSOPOULOS N. Adaptive probabilistic search for peer-to-peer networks [C]//Proc of the 3rd IEEE International Conference on P2P Computing. Washington DC:IEEE Computer Society,2003:102-109.

[8] CHAWATHE Y, RATNASAMY S, LANHAM N, *et al*. Making Gnutella-like P2P systems scalable [C]//Proc of ACM SIGCOMM. New York:ACM Press,2003:407-418.

[9] MERINO L R, ANTA A F, LÓPZE L, *et al*. Self-managed topologies in P2P networks[J]. Computer Networks, 2009, 53(10):1722-1736.

[10] CARDELLINI V, COLAJANNI M, YU P S. Dynamic load balancing on Web-server systems[J]. IEEE Internet Computing,1999,3(3):28-39.

[11] SHEKHAR H M P, RAMANATHA K S. Mobile agents based congestion aware routing in mobile Ad hoc networks[C]//Proc of the 6th IEEE International Conference on 3G and Beyond. 2005:47-53.

[12] LI Hui, SHAO Fei. An improved load balancing algorithm for P2P system based on mobile agent[C]//Proc of the 2nd International Conference on Artifical Intelligence, Management Science and Electronic Commerce. 2011:2791-2794.

[13] KWONG K W, TSANG D H K. A congestion-aware search protocol for unstructured peer-to-peer networks[C]//Proc of the 2nd International Conference on Parallel and Distributed Processing and Applications. Berlin:Springer-Verlag,2004:319-329.

[14] SAROIU S, GUMMADI P K, GRIBBLE S D. A measurement study of peer-to-peer file sharing systems [C]//Proc of Multimedia Computing and Networking. 2002:156-170.

[15] STROGATZ S H, WATTS D J. Collective dynamics of 'small-world' networks[J]. Nature,1998,393:440-442.