一种结构化P2P网络动态负载均衡算法的研究

陆垂伟[1,2] 李之棠[1] 林怀清[1] 黄庆凤[1] 张冶江[1]
(华中科技大学计算机学院 武汉 430074)[1] (黄石理工学院计算机学院 黄石 435003)[2]

摘 要 负载均衡是P2P网络的研究热点之一,当前负载均衡技术存在负载均衡程度低、假设条件过多等问题。提出一了种增强型负载均衡算法ELB_P2P,它根据节点的承载能力为其分配相应大小的可动态调整的ID地址空间以及合理的载荷,在负载转移时自动选择延迟小带宽高的轻载节点,并引入负载转移流量控制机制。实验表明,相对于Chord等传统P2P协议,ELB_P2P算法有更快的负载均衡速度、更小的负载均衡开销,系统稳定性好,在网络重载情况下也能取得较低的负载不平衡度,并且对节点属性没有苛刻的限制和假定。

关键词 负载均衡,均衡开销,P2P网络,虚拟服务器
中图法分类号 TP393.3 **文献标识码** A

Dynamic Load-balancing Algorithm on Structured P2P Network

LU Chui-wei[1,2] LI Zhi-tang[1] LIN Huai-qin[1] HUANG Qing-feng[1] ZHANG Ye-jiang[1]
(Computer School, Huazhong University of Science and Technology, Wuhan 430074, China)[1]
(Computer School, Huangshi Institute of Technology, Huangshi 435003, China)[2]

Abstract Loading balancing is one of research hotspot of P2P network. There exist many problems such as low load-balancing degree and excess assumption conditions etc. in existing load-balancing technology. The paper proposed an enhanced load-balancing algorithm: ELB_P2P. The algorithm assigns rational load and corresponding ID address space that can be dynamicly regulated to every peer in P2P system. In addition, the algorithm introducesd flux control mechanism, and automatically selected light load peers with low delay and high bandwidth for load diversion. The experiments demonstrate: compared with traditional Chord protocol, the ELB_P2P algorithm has faster velocity of load balancing, less spending of load-diversion, and more excellent stability of P2P system, furthermore, it still obtains high load-balancing degree in the event that P2P network load is very heavy, and without stern limitation and assumption conditions aiming to property of peers.

Keywords Load balancing, Load-balancing spending, P2P network, Virtual servers

基于DHT的结构化P2P技术已成为当前P2P应用的主流技术,它比上一代的非结构化P2P技术在可扩展性、负载均衡、容错能力、资源查找速度等方面都有很大的提高。目前有关结构化P2P技术的研究分支有很多,其中负载均衡是研究的一个焦点,优秀的均衡算法能大幅度提高P2P系统的稳定性和承载能力。引起系统负载不均衡的主要原因是节点上的任务是动态随机分配的,而且任务的完成时间不确定。故只能在运行时测量并计算节点的负载分配状况,然后根据状况在P2P系统内调剂转移负载,提高整个系统的负载均衡程度。目前,负载均衡问题已经有一些研究成果[1-3],但仍存在收敛速度低、负载均衡程度不高、假设条件过多过于苛刻等问题。

本文提出了一种增强型负载均衡(Enhanced Load Balance)算法ELB_P2P,它以环形网络为基础,根据节点的负载能力为其分配连续的IDs,并根据节点负载能力和网络规模的变化动态调整IDs的大小,在负载转移时自动选择延迟小带宽高的轻载节点,并控制负载转移流量。模拟实验表明,该算法在节点能力不均衡的P2P系统中,仍有较快的负载均衡速度、较小的负载均衡开销、较好的系统稳定性,并能在网络重载情况下保持较低的负载不平衡度。

1 相关研究

已有许多工作致力于优化结构化P2P系统的负载均衡性能。Chord[4]和CAN[5]系统在均衡负载时,均假设节点能力和每个资源带来的资源是相同的,这简化了负载均衡算法,但与实际情况不符,因此负载均衡性能并不理想。文献[6]提出了一种基于局部网络信息的负载平衡算法,大大提高了系统负载均衡速度,但该算法是针对同构网络设计的,且假定每个节点的性质和能力都相同。文献[7]提出了一种分布式负载均衡算法,充分考虑了链路延迟并解决了单点失效问题,负载均衡效果良好并减少负载转移开销45%以上,但不适合多种瓶颈资源情况下的P2P系统。文献[8]设计了一种基于综

到稿日期:2008-12-16 返修日期:2009-01-05 本文受863国家重点基金项目(2007AA01Z420),国家自然科学基金(60573120)资助。
陆垂伟(1973-),男,博士生,主要研究方向为P2P网络、信息安全,E-mail:Lcwzm@mail.hust.edu.cn;李之棠(1951-),男,教授,博士生导师,主要研究方向为计算机网络、信息系统安全;林怀清(1973-),男,博士生,主要研究方向为P2P网络安全。

合临近测度的负载均衡算法，其平均路径长度为标准 Chord 的 50%以下，延迟减少 65%以上，负载均衡速度快且不平衡度低。文献[9]提出了一种基于虚拟服务器和目录的负载均衡算法，应用于静态异构网络上，文献[10]把这种方法扩展到动态异构网络上，文献[11]提出的负载平衡机制依赖于能快速找到系统中最小和最大的负载节点，但没有给出有效寻找这两种节点的方法。文献[12]在 P2P 网络结构之上再建立了一个结构：k-ary 树，它负责节点信息的收集、发布以及虚拟服务器转移策略的生成，该算法提高了负载均衡效果，但复杂化了网络结构，使均衡速度和容错能力有所降低。

2 动态负载均衡算法

一个优秀的负载平衡算法需要实现两个目标。一是最大的负载平衡程度，即每个节点的利用率接近系统的平均利用率；二是最小的转移开销，即减少转移算法的执行时间和资源占用，降低负载转移量等等，使系统能尽快达到负载平衡的状态，并保持较好的网络稳定性。本文提出的负载平衡算法是基于环形 P2P 网络的，并有如下假设。

(1)把影响节点负载能力的因素，如 CPU 速度、存储容量、延迟、带宽等，统一视为一种资源，用一个参数 C_i 来表示节点 P_i 的最大承载能力，并利用归一化方法将承载能力进行规格化，即 $\sum_i^n C_i = n$，其中 n 表示 P2P 系统中节点的个数，根据该式，节点只有估算出整个网络的平均承载能力后才能得出自己的规格化承载能力，这种估算是通过随机采样其它节点而获得的。

(2)负载均匀分布在 ID 地址空间里。

2.1 相关定义

定义 1(利用率) 本文提出了两种利用率。①节点利用率是指节点 P_i 的负载与其最大承载能力之比，即 $u_i = Load_i/Capacity_i$，它指出了当前节点的负载状况。②系统利用率是指系统中所有的节点的负载与最大承载能力之比，假设系统中共有 n 个节点，即 $U=\frac{\sum_1^n Load_i}{\sum_1^n Capacity_i}$，它代表网络总的负载状况。

定义 2(负载不平衡度) 用来描述整个网络节点负载的不平衡程度，由节点的实际负载与其最大承载能力的方差表示，即 $B=\frac{1}{n}\sum_{i=1}^n [L_i - C_i]^2$，当 $B=0$ 时，系统达到完全负载平衡的理想状态。

2.2 负载分配策略

本文采用虚拟服务器法来提高系统的负载平衡性能，但做了如下改进：①虚拟服务器的 ID 值不再是非连续的随机值，而是一段连续的环地址空间，②节点的最大承载能力与其占有 ID 空间的大小成正比，承载能力越强的节点占有的 ID 空间也越大，另外，只有节点的承载能力有明显变化(大于 5%)时，其占有的 ID 地址空间大小才做相应的改变，这些改进的好处是减少了负载分配与转移算法的复杂性，提高了算法的执行效率和网络系统的稳定性，当然是以浪费部分 ID 资源换来的。假设 A_i 代表节点 P_i 拥有的地址段，那么它承担的负载可表示为 $Share(P_i)=\frac{A_i}{C_i/n}$，从而保证每个节点在进入 P2P 系统之后不久就能承担与其能力相适应的载荷，减少了以后调整的代价。

每个 ID 都可以构成一个虚拟服务器，虚拟服务器相当于 DHT 中的节点，直接对外提供数据服务，因而一个物理节点可能对应多个虚拟服务器，当物理节点过载时，负载就从它的虚拟服务器转移到其它轻载的物理节点的虚拟服务器上去，这种转移过程可以在任意两个节点之间进行，而不局限于相邻节点之间，从而提高了整个 P2P 系统的负载平衡程度。

本算法假定 DHT 管理着一个分段均匀的环形 ID 地址空间，即[0,1)，并以 1 为模，节点 P_i 获取一个 ID：$ID(P_i) \in [0,1)$，并被指派给地址段的拥有者$(ID(P_j), ID(P_i)]$，这里 P_j 是节点 P_i 最近的前继节点。为了定量分析节点负载分配情况，令 $\alpha=\alpha(n)$代表每单位节点负载能力的虚拟服务器数量。当 $\alpha=1$ 时，每个节点只对应一个虚拟服务器，但是最大的共享是 O (log$n$)，当 $\alpha=$ O (log$n$)时，则实现 O (1)的最大共享，但是节点度数以 $\alpha=$ O(log$n$)方式增加。这样就会出现一个问题：如果 α 较小，性能非常低的节点在维持一个虚拟服务器的情况下都会发生过载，如果 α 较大，就会导致高的节点度数，使中等能力的节点都必须维持 α 个虚拟服务器和过多的网络连接。为保证最小承载能力(C_{min})的节点接近于它们的公平负载，必须使得 $1/(n\alpha)=$ O(C_{min}/n)，但是由于 C_{min} 可能是任意小，而且是不可靠和难于估计的，因此 α 又必须是任意的大。若设定一个最低承载能力门限 γ_d 来解决这个问题，即某节点承载能力低于 γ_d，那么它就不产生虚拟服务器实例，也不参加标准 DHT 路由协议，亦即被负载均衡算法所抛弃。被抛弃的节点不再是环结构的组成部分并且不再提供路由信息，但可以作为数据存储点，继续为系统执行数据查询操作，方法是让这些节点在系统中随机选择一个父节点，并由父节点为它们分配数据存储。根据前面的论述，设计出负载分配伪代码，如图 1 所示。

```
ID_ Pi= Random[0,1)
n′,Ci′← estimates of n and Ci
ID_Fraction= 1/ n′
ID_Begin = ID_ Pi/ ID_Fraction
If Ci′<γd then
  m =0
  Exit
Else m = 0.6 + Ci′ · α(n′)
End if
Select m virtual servers with IDs r1…rm at ID_Begin−(i+ri)
ID_Fraction and ri∈[0,1)
End the Initialization process of Pi
Estimate new n″,Ci″with setting cycle
If (|n′− n″|/ n′) >0.05 then
  n′= n″
  If (|Ci′− Ci″|/ Ci′) >0.05 then
    Ci′= Ci″
  End if
End if
If n′or Ci′changed then
Regulate IDs address and virtual servers of Pi as above
End if
```

图 1 负载分配算法

伪代码的前半部分用于节点的初始化，包括自身承载能力估算，ID 地址的选择以及是否有资格加入负载均衡算法，

后半部分用于节点周期性地收集网络状况和重新估算自身承载能力，并据此调整 ID 地址段和虚拟服务器数量。

2.3 负载转移策略

每个节点周期性检查其负载情况，当载荷达到某种程度时，就按以图 2 所示的伪代码来执行负载转移。

```
While (Pi is active)
Begin
  If Li>εCi Then
    State(Pi) = overload
    {P1,…Pn}= RefreshNeiborsList(Pi)
    For k= 1 to n do
      If (TTLk-i> t) or ((Ci-Li)>(Ck-Lk)/2) Then
        Remove Pk from {P1,…Pn}
      End if
    N = SortbyUC({P1,…Pn})
    FlowLimit = f
    Repeat Li --transload--> N Until Li<εCi
  Else
    Wait next inspection cycle coming
  End if
End
```

图 2 负载转移算法

代码中 L_i 代表节点 P_i 的负载，C_i 代表 P_i 的最大承载能力，常数 ε 为调节因子，且 $0<\varepsilon<1$，这样节点负载还未达到其最大承载能力时就启动了负载均衡算法，从而使节点还有剩余能力用于处理负载转移，避免节点因重载而崩溃，另外，当 ε 增大时，负载均衡质量降低但负载转移开销减少，反之亦然。函数 RefreshNeiborsList 获取节点 P_i 的邻居节点集合，TTL_{k-i} 代表节点 P_k 与 P_i 之间的链路延迟，参数 t 为延迟阈值，延迟大于 t 的节点或者剩余负载能力不足的节点都被清除出邻居节点集合，函数 SortbyUC 根据节点剩余负载能力从高到低对邻居节点集合进行排队，得到新集合 N，这里采用节点剩余负载能力，即利用率(U)与最大承载能力(C)之乘积作为排队依据而不是仅仅以节点的利用率为依据，这更符合负载均衡的要求。参数 f 是负载转移的流量阈值，对负载转移流量作一个限制可以减少对网络的冲击，提高系统稳定性，当然以少许降低负载均衡速度为代价。负载优先向排在集合 N 前面的节点转移，直到节点 P_i 的负载降到合适的值为止。

3 仿真实验

实验使用 PlanetSim3.2 作为仿真软件，它是基于 Java 和 LGPL-Like 许可证的开源工具，专门针对 Chord 之类的环形 P2P 网络的仿真。实验中采用临近路由选择(PNS)的改进型 Chord 作为比较算法之一，该算法拥有比标准 Chord 更优的路由与数据搜索性能，负载平衡性能较好。实验在 CPU 为 AMD5000+，2G 内存的 PC 机上进行，并以 Tomcat 作为 Java 虚拟机，模拟的节点数量为 10^5 个，参数设置如表 1 所列。

表 1 实验参数配置

实验参数	设置值
节点数量	1×10^5
ID 空间分配	与节点承载能力成正比并连续分配
资源分布	环形空间上均匀分布
算法执行周期 T	200ms
调节因子 ε	0.925
延迟阈值 t	850ms
负载转移的流量阈值 f	500kB/s
最低承载能力门限 γ_d	0.65

3.1 负载不平衡度

从图 3 中可以看出 ELB_P2P 算法的负载不平衡度收敛速度较快，仅 10 个周期 B 的值就降至 50 以下，而 PNS_Chord 算法和 Chord 算法在第 17 个周期 B 值才分别降至 51 和 68，可见 ELB_P2P 算法能明显加快系统负载的平衡。

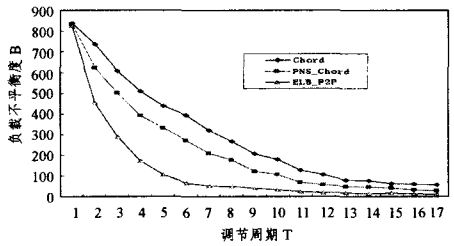


图 3 负载不平衡度的变化

3.2 系统利用率对负载不平衡度的影响

当系统利用率等于 10%，20%，30%，…，90%时，截取了此刻 3 种算法的负载不平衡度的值，并形成图 4 的柱形图。从图 4 中可以看出，随着时间和系统负载的增长，3 种算法的负载不平衡度 B 都呈逐渐下降趋势，但 ELB_P2P 算法的负载不平衡度比其它两种算法下降得更快，当系统负载达到 60%时，B 值就降到理想区间(<50)，此后继续平稳下降，这说明 ELB_P2P 算法在重载时也能保持较好的负载均衡性能和系统稳定性。

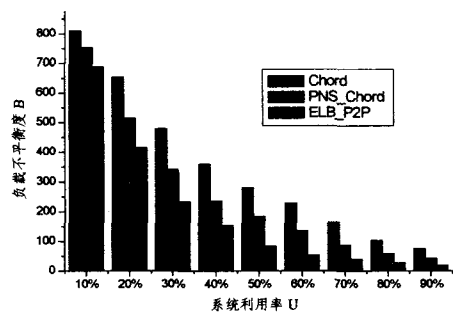


图 4 系统利用率逐渐增长时负载不平衡度的变化

3.3 负载转移量的分布

从图 5 可以看出，在 ELB_P2P 算法中考虑延迟因素之后，大部分负载转移都在低延迟的节点之间进行，以 90%的负载累积转移量为比较标准，ELB_P2P 算法的负载转移主要在延迟小于 300ms 的节点之间进行，PNS_Chord 算法的负载转移主要在延迟小于 600ms 的节点之间进行，而 Chord 算法的负载转移主要在延迟小于 800ms 的节点之间进行。这表明 ELB_P2P 算法加快了负载转移速度，大幅度减少了负载转移开销。

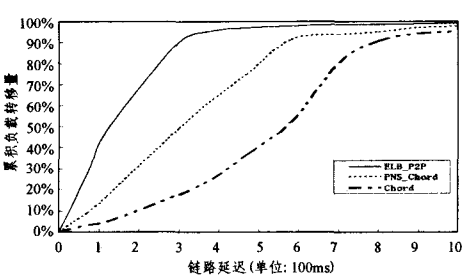


图 5 系统负载转移累积量

结束语 本文针对结构化 P2P 网络提出了一种增强型负载均衡算法 ELB_P2P，根据该算法，节点的承载能力与其占有 ID 空间的大小成正比，并且 ID 地址连续分配，从而减少了算法的复杂性，负载转移时考虑了节点之间的延迟并引入流量控制机制，提高了系统负载均衡性能。仿真实验表明该算法具有负载均衡速度快，负载均衡开销小，系统稳定性高，负载不平衡度低等优点。

参 考 文 献

[1] Giakkoupis G, Hadzilacos V. A scheme for load balancing in heterogeneous distributed hash tables[C]// Proc. PODC. July 2005

[2] Karger D, Ruhl M. Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems[C]// Proc. SPAA. 2004

[3] Godfrey P B, Stoica I. Heterogeneity and load balance in distributed hash tables[C]// Proc. IEEE INFOCOM. 2005

[4] Ratnasamy S, Francis P. A scalable content addressable network [C]// SIGOMM 2001. San Diego, CA, USA, 2001

[5] Stoica I, Morris R. Chord: A scalable peer-to-peer lookup service for internet applications[C]// SIGOMM 2001. San Diego, CA USA, 2001

[6] 姚磊，戴冠中，等. 基于局部网络信息的 P2P 系统负载平衡算法[J]. 计算机应用，2007(05)：1080-1083

[7] 李振宇，谢高岗. 基于 DHT 的 P2P 系统的负载均衡算法[J]. 计算机研究与发展，2006(09)：1579-1585

[8] 张三峰，吴国新. P2P 网络非对称 DHT 方法及负载均衡技术研究[J]. 通信学报，2007(09)：60-67

[9] Rao A, Lakshminarayanan K, Surana S, et al. Load Balancing in Structured P2P Systems[C]// Proc. IPTPS. Feb. 2003

[10] Godfrey P B, Lakshminarayanan K, et al. Load balancing in dynamic structured P2P systems[C]// Proc. IEEE INFOCOM. 2004

[11] Bawam G P. Distributed balanced tables: not making a hash of it all[R]. Stanford University, Database Group, 2003

[12] Zhu Y, Hu Y. Efficient proximity - aware load balancing for DHT-based P2P systems [J]. IEEE Trans on parallel and distributed systems, 2005(04)

(上接第 31 页)

是 G_{XTC} 被分割为两个连通子图；当网络中 26 个节点失效时 $G_{M\text{-}XTC}$ 被分割为 3 个连通子图，而 G_{XTC} 却被分成 11 个连通子图。由图 5、图 6 可以看出，拓扑 $G_{M\text{-}XTC}$ 在保证 MAC 协议效率的前提下，相比 G_{XTC} 来说具有更好的连通性和鲁棒性。

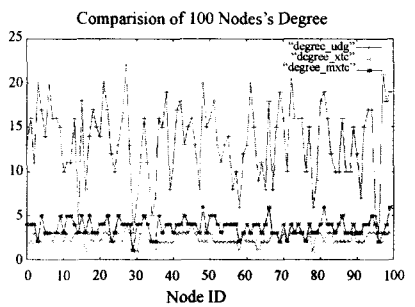
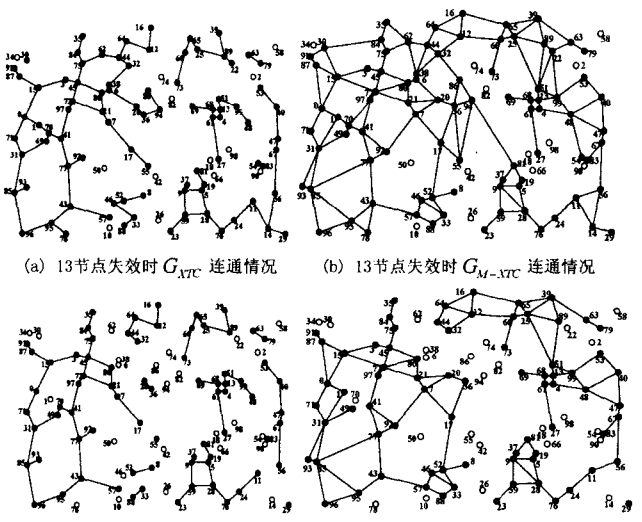


图 5 节点度分析



(a) 13节点失效时 G_{XTC} 连通情况 (b) 13节点失效时 $G_{M\text{-}XTC}$ 连通情况

(c) 25 节点失效时 G_{XTC} 连通情况 (d) 25 节点失效时 $G_{M\text{-}XTC}$ 连通情况

图 6 拓扑连通性鲁棒性分析示意图

结束语 本文对无线传感器网络拓扑控制的相关算法进行了介绍，在分析 XTC 算法的基础上，提出了一种基于局部网络信息的分布式拓扑控制算法 M-XTC，该算法保持了 XTC 算法简单、实用，不需要节点位置信息，适用于普通节点、异构网络和三维空间等优点，并且经过仿真验证，该算法更有利于延长网络的生存时间，具有更好的实时性和鲁棒性。

参 考 文 献

[1] Poduri S, Pattem S, Krishnamachari B, et al. A unifying framework for tunable topology control in sensor networks [R]. CRES-05-004. University of Southern California, 2005: 1-15

[2] Narayanaswamy S, Kawadia V, Sreenivas RS, et al. Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol[C]// Proc. of the European Wireless Conf. Florence, 2002: 156-162

[3] Kubisch M, Karl H, Wolisz A, et al. Distributed algorithms for transmission power control in wireless sensor networks[C]// Yanikomeroglu H, ed. Proc. of the IEEE Wireless Communications and Networking Conf. (WCNC). New York: IEEE Press, 2003: 16-20

[4] Li L, Halpern JY, Bahl P, et al. A cone-based distributed topology control algorithm for wireless multi-hop networks[J]. IEEE/ACM Trans. on Networking, 2005, 13(1): 147-159

[5] Li N, Hou J C. Topology control in heterogeneous wireless networks: Problems and solutions[C]// Proc. of the IEEE Conf. on Computer Communications (INFOCOM). New York: IEEE Press, 2004: 232-243

[6] Heinzelman W R, Chandrakasan A P, Balakrishnan H. Energy-Efficient communication protocol for wireless microsensor networks[C]// Nunamaker J, Sprague R, eds. Proc. of the Hawaaian Int'l Conf. on System Science (HICSS). Washington: IEEE Press, 2000: 3005-3014

[7] Deb B, Bhatnagar S, Nath B. A topology discovery algorithm for sensor networks with applications to network management[R]. DCS-TR-441. Rutgers University, 2001

[8] Younis O, fahmy S. HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks[J]. IEEE Trans. on Mobile Computing, 2004, 3(4): 660-669