结构化 P2P 网络热点负载动态迁移策略

孟宪福,陈晓令
(大连理工大学计算机科学与技术学院,辽宁大连 116024)

摘 要: "热点"问题是导致结构化 P2P 网络负载失衡进而影响检索效率的重要因素.在给出节点下载量与被下载量以及"热点"信息与"冷点"信息管理机制的基础上,利用"热点"节点与"冷点"节点的互补特点,提出了推拉结合的结构化 P2P 网络"热点"负载动态迁移策略.由于是由"热点"节点和"冷点"节点的邻居节点来触发负载迁移过程,消除了因 P2P 节点的自私性而带来的对负载迁移不协作问题.模拟实验结果表明,所提出的策略能够有效地解决结构化 P2P 网络的"热点"问题且在工程上是可行的.

关键词: P2P; DHT; "热点" 问题; 负载迁移; 推拉结合
中图分类号: TP301 **文献标识码:** A **文章编号:** 0372-2112 (2011) 10-2407-05

Strategy for Hotspot Load Dynamic Migration in Structured P2P Networks

MENG Xian-fu, CHEN Xiao-ling
(*School of Computer Science and Technology, Dalian University of Technology, Dalian, Liaoning* 116024, *China*)

Abstract: "Hotspot" problem is an important factor of causing the load imbalance and then affecting the retrieval efficiency in structured P2P networks. In this paper, the mechanisms for managing the amount of node's download and upload as well as the information of the "hotspot" nodes and the "cold spot" nodes are described and then the push and pull combined "hotspot" load migration strategy in structured P2P network is proposed by utilizing the complementary nature of "hotspot" nodes and "cold spot" nodes. Since the load migration processes are triggered by their neighbor nodes of "hotspot" or "cold spot", the non-cooperative problem brought by the nodes due to their selfishness for the load migration can be eliminated. The experimental results show that the proposed strategy is effective in handling "hotspot" problem in structured P2P networks and easy to put into practice.

Key words: P2P; DHT; "hotspot" problem; load migration; push-pull

1 引言

DHT(Distributed Hash Table,分布式哈希表)作为一种结构化 P2P 网络,在信息检索的准确性和时间复杂度等方面都明显优于非结构化 P2P 网络,但也具有不可忽视的弱点,如在节点退出或失效时网络恢复代价高和容易产生"热点"问题等[1].其中,"热点"的产生将导致系统负载的严重失衡,进而严重影响系统的检索效率[2,3].DHT 网络环境下产生"热点"问题的原因有很多[4],但归纳起来主要是由于节点所承载的键字空间的差异、共享文件数量的差异以及信息检索的不平衡性等原因造成的.目前,解决 DHT 网络负载平衡问题的策略主要包括缓存的方法[4~6]、ID 空间划分的方法[7]、虚拟服务器的方法[8~11]以及多 Hash 的方法[12,13]等.

文献[4]采用逻辑链路迁移和缓存相结合的方法来解决 DHT 网络节点间的负载不平衡问题.由于需要在请求信息中加入节点自身的负载信息以及每隔周期 T 就需要计算一次自身的各种负载,大大增加了网络负载量和节点的计算量.另外,文献[4]中是将节点应答消息的个数定义为负载,显然不能反映节点的实际负载状况.文献[7]提出一种能够根据节点的能力来有效划分 ID 空间的方法,但这种方法只是一种静态划分方法,不能适应实际 P2P 网络环境下的动态负载变化.文献[9]是由系统指定几个节点作为 Directory 来保存与其联系的节点的负载信息(即每个虚拟服务器的负载),然后每隔 T 秒由 Directory 节点来计算所管理的节点的负载信息,并据此实现虚拟服务器的再分配,难以把握 DHT 网络的整体负载状况,同时也没有给出动态负载的具体管理方法.文献[12]和文献[13]提出了在将关键字分配给节点时采用多哈希的方法,将关键字哈希映射到 $d \geqslant 2$ 个不同的节点上,并从 d 个节点中选择一个负载最小的节点存放对象索引,其余 $d-1$ 个映射节点只存储指

收稿日期:2010-10-20;修回日期:2010-11-23
基金项目:国家自然科学基金(No.60973014)

向存储对象索引的节点的重定向指针.这种方法本身并不能减轻保存共享文件节点的负载量,同时将产生额外的通信开销.

本文以 Chord 网络[4]为研究对象,采用"热点"副本迁移策略来解决"热点"问题.通过将"热点"上的文件副本迁移到"冷点"上,以达到减少"热点"被下载量和提高网络整体利用率的目的.

2 相关定义

(1)下载量与被下载量

下载量:如果 A 节点从别的节点上下载了数据量为 x 的文件,则称 x 为节点 A 的下载量,即节点 A 接受别的节点所提供的服务量.

被下载量:如果别的节点从 A 节点上下载了数据量为 x 的文件,则称 x 为节点 A 的被下载量(又称上载量),即节点 A 为别的节点所提供的服务量.

(2)"热点"节点与"冷点"节点

热点节点:如果在一段时间$[t_1, t_2]$内节点 A 的被下载量大于阈值 H,则称节点 A 是这段时间内的"热点"节点.

冷点节点:如果在一段时间$[t_1, t_2]$内节点 A 的被下载量小于阈值 L,则称节点 A 是这段时间内的"冷点"节点.

这里 $H > L$.阈值 H 与 L 的选择取决于特定的网络应用背景,其选择方法将在第 4 节中详细讨论.

(3)活动索引与非活动索引

活动索引:是指在检索过程中能直接利用的索引.现有的 Chord 网络中的索引都属于活动索引.

非活动索引:是指在检索过程中不能直接利用的索引,只有当满足一定条件被激活而变为活动索引之后才能被直接利用.

(4)时间区间 T

时间区间 T(又称为 T 时间段)用于表示距离当前时间的一段时间区间,即:$T = < Tc\text{-}\triangle, Tc >$,其中 Tc 表示当前时间,$\triangle$表示时间长度.

3 "热点"负载的动态迁移策略

在已有的研究中,很多文献都利用单阈值将节点划分为重载节点(heavy nodes)和轻载节点(light nodes),通过轻载节点周期性地查询重载节点并将重载节点的虚拟服务器迁移到轻载节点上来实现负载平衡[8],因此,需要消耗大量的网络带宽,实用性很低.而利用本文的双阈值划分方法,只要将少量的热点负载迁移到满足条件的冷点节点上,就能够为热点节点"降温",进而消除因"热点"的存在而对检索效率产生的影响.

由定义可知,一个节点是"热点"还是"冷点",是由该节点的被下载量确定的.为此,需要首先解决被下载量与下载量的管理问题.

3.1 下载量与被下载量的管理

考虑到 P2P 网络节点的自私特点,节点的下载量与被下载量不能保存在本地节点上.本文利用邻居节点来管理节点的下载量和被下载量.

当节点 A 被下载了共享文件时,就立即触发一个将被下载信息传递给其后继节点 *Successor*(A)的 UpEvent 事件.所传递的被下载信息记录 R 包括:被下载节点 IP、文件 ID、文件长度(*filesize*)、被下载时间(*uptime*)等数据项.

当节点 A 下载了共享文件时,就立即触发一个将下载信息传递给其后继节点 *Successor*(A)的 DownEvent 事件.所传递的下载信息记录 S 包括:被下载节点 IP、下载节点 IP、文件 ID、文件长度(*filesize*)、下载时间(*downtime*)等数据项.

考虑到 P2P 网络的 Churn 特性,在后继节点接收到下载或被下载信息之后,立即将其传递给节点的前趋节点,也就是由节点的后继和前趋两个节点来管理节点的下载与被下载信息.同时,前趋节点和后继节点相互监视对方的存在,一旦发现对方 IP 有变化,就将己方保存的当前节点的下载与被下载信息拷贝到对方节点上.

3.2 UpEvent 事件与 DownEvent 事件的处理

UpEvent 事件是当某节点被下载了共享文件时触发的,此时在将被下载信息传递给后继节点之后,在后继节点上将完成以下工作:

(1)将接收到的当前节点的被下载信息记录 R 保存到本地的 *UpSet* 集合中,同时传递给当前节点的前趋节点.

(2)利用公式(1)计算当前节点的 T 时间段内的被下载量 *UpCount*:

$$UpCount = \sum_{\substack{Ri \in UpSet, \\ Ri.uptime \in T}} Ri.filesize \tag{1}$$

这里,*filesize* 是被下载的文件长度,*uptime* 是被下载的时间.

(3)if ($UpCount > H$)\{

"推"方式副本迁移处理,详见 3.3.2 节\}

DownEvent 事件是当某节点下载了共享文件时触发的,此时在将下载信息传递给后继节点之后,在后继节点上将完成以下工作:

(ⅰ)将接收到的当前节点的下载信息记录 S 保存到本地的 *DownSet* 集合中,同时传递给当前节点的前趋节点.

(ⅱ)利用式(1)和式(2)分别计算当前节点的 T 时间段内的被下载量 *UpCount* 和下载量 *DownCount*:

$$DownCount = \sum_{\substack{Si \in DownSet,\\ Si.downtime \in T}} Si.filesize \quad (2)$$

这里,*filesize* 是下载的文件长度,*downtime* 是下载的时间.

(ⅲ)if (*UpCount* < *L* and *DownCount* > δ){

"拉"方式副本迁移处理,详见 3.3.3 节}

当某节点接受了其他节点的服务(下载量)之后,就应该在满足"冷点"条件时接受来自"热点"的负载,否则就成为 Free rider 节点.将 *DownCount* > δ 作为"拉"方式判断条件就是基于这一考虑的.

3.3 推、拉结合的"热点"负载迁移策略

"推"方式是由"热点"节点的邻居节点发起迁移过程,将"热点"节点上的文件副本"推"到"冷点"节点上;"拉"方式是由"冷点"节点的邻居节点发起迁移过程,将"热点"节点上的文件副本"拉"到"冷点"节点上.

无论是"推"方式还是"拉"方式,都需要在网上寻找"冷点"信息或"热点"信息,因此,如何有效地管理和寻找"冷点"信息和"热点"信息是需要研究的重要课题.

3.3.1 "热点"信息与"冷点"信息的管理

对于"热点"信息和"冷点"信息的管理与寻找需要遵循以下两个基本原则:一是不能遍历所有节点寻找,以免代价太高;二是寻找方法应与 DHT 网络特点结合起来.鉴于此,本文将所有"热点"信息都保存在 *nodeID* = *Hash*(*H*)节点上,所有"冷点"信息都保存在 *nodeID* = *Hash*(*L*)节点上.由于 P2P 网络中的文件请求可能满足"重尾"分布[10]特点,利用一个节点来管理"冷点"信息可能影响 P2P 网络的可扩展性,为此,在第 4 节将讨论"冷点"信息管理的调整策略.

3.3.2 "推"方式文件副本迁移策略

"推"方式的副本迁移是由"热点"的后继节点来完成的.基本步骤如下:

(1)从 *Hash*(*L*)节点上获取"冷点"集合 *Sc*.

(2)if (| *Sc* | = =0) {

将"热点"节点的相关信息(节点 ID、节点 IP、*T* 时间段及其对应的被下载量)保存到 *nodeID* = *Hash*(*H*)的节点上.结束处理.}

(3)if (| *Sc* | ! =0) {

求解迁移方案并保存到 *Mig* 集合中,具体过程将在本节后半部分叙述.}

(4)根据 *Mig* 中的迁移方案进行副本迁移.

(5)调整相关索引,参见 3.3.4 节.结束.

步骤(3)中的求解是一个优化问题,即:将"热点"上的多个文件副本如何迁移到 | *Sc* | 个"冷点"上,使得各节点的预期被下载量基本相同.这一问题可以利用遗传算法等优化方法来获取优的解,但考虑到实用性,本文将采用如下的方法进行求解,以利于在后续的模拟实验中应用.基本步骤如下:

(1)将"热点"上的共享文件按 *T* 时间段内的被下载量降序排列,生成队列 *FQueue*.

(2)将所有"冷点"节点按 *T* 时间段内的被下载量升序排列,生成队列 *NQueue*.

(3)执行如下算法:

for each $f \in$ *FQueue* do

for each $n \in$ *NQueue* do

if (f 的被下载量 + n 的被下载量 < *H*-β) {

将 f 副本迁移到 n 上的方案保存到集合 *Mig* 中;n 被下载量 + = f 被下载量;

if ("热点"与 *Mig* 的被下载量之差小于 *H*-β)

求解过程结束;}

if (| *Mig* | = =0) {

将 *FQueue* 队首的文件副本迁移到 *NQueue* 队首节点上的方案保存到集合 *Mig* 中;}

在本算法中,β 是调节因子,用于防止在副本迁移后"冷点"马上变为"热点"的问题.

3.3.3 "拉"方式文件副本迁移策略

"拉"方式的副本迁移是由"冷点"的后继节点来完成的.基本步骤如下:

(1)从 *Hash*(*H*)节点上获取"热点"集合 *Sh*.

(2)if (| *Sh* | = =0) {

将"冷点"节点的相关信息(节点 ID、节点 IP、*T* 时间段及其对应的被下载量)保存到 *nodeID* = *Hash*(*L*)的节点上.结束处理.}

其余处理步骤与 3.3.2 节的"推"方式相似.

3.3.4 索引的更改与添加

对于"热点"节点来讲,对所有其副本满足迁移条件的原始文件的索引都更改为非活动索引;当没有一个副本满足迁移条件时,则将副本被迁移的原始文件的索引仍然保持为活动索引,以便"热点"和相应"冷点"共同承担对同一个文件的下载服务.

对于"冷点"来讲,对被迁移过来的所有共享文件的副本都要添加为活动索引.

4 相关设置与讨论

4.1 阈值 *H* 和 *L* 的设置

H 与 *L* 的初值可根据网络实际应用背景来设置,由第一个入网节点确定.

当网络中长期只存在"热点"节点时,可适当增大 *L* 值;当网络中长期只存在"冷点"节点时,可适当减小 *H* 值.一旦某节点对 *H* 值或 *L* 值进行修正之后,可通过广播的方式通知其他节点.

4.2 "冷点"信息和"热点"信息的管理

由前述可知,本文分别利用 *nodeID* 为 *Hash*(*L*)和

Hash(*H*)节点来保存“冷点”信息和“热点”信息.考虑到P2P检索请求的“重尾”分布特性,如果仅利用 *Hash*(*L*)节点来管理“冷点”信息,则有可能影响网络的可扩展性.为此,可利用多节点管理策略.具体策略为:根据“冷点”ID的低 *m* 个二进制位,来计算应将“冷点”信息保存到的节点ID.比如,当此 *m* 个二进制位为 *x* 时,将“冷点”信息保存到 *Hash*(*L* + *x*)的节点上,其中 *m* 可根据网络规模等进行设置,一般很小即可.在需要时,对“热点”信息的管理也可以采用同样的策略.

4.3 非活动索引的激活

当某“热点”需要迁移副本时,可首先选择被迁移过来的副本(如果存在的话),如果该副本的原始文件所在节点已不再是“热点”,则直接删除该文件副本和索引,这样,在检索时,如果只存在该文件的非活动索引,就直接将其激活,使其变为活动索引参与检索过程.

5 模拟实验与结果分析

为了验证本文策略的有效性,我们利用开源的peerSim系统进行模拟实验.Chord网络[14]中共存在1000种共享文件,其文件ID为1~1000,每个文件的大小为1~20之间.实验以 *zipf*(C/i^{α})分布来生成文件查询消息,其中 *C* 设置为0.1,α 设置为1.

将本文策略中的 *H*、*L* 和 β 设置为动态值,实验中每隔△时间(即 *T* 时间段长度)对这三个值进行调整,同时确保 *H*、*L*>0 且 *H*>*L*,最后设置 $\beta = 0.1 \times (H\text{-}L)$.

实验1 负载分布与被访问节点个数统计

设置网络节点个数为1000,将本文策略中的△参数设置为600单位时间,在查询服从 *zipf* 分布情况下,实验进行300000单位时间.图1是采用本文策略的最终负载分布图.所有节点的负载均在350000至460000之间,最大节点负载与最小节点负载之比为1.17,说明本文策略能有效地平衡网络节点的负载量,达到“热点”负载向“冷点”迁移的目的.如果不采用本文策略,则负载分布也基本满足“重尾”分布特性.

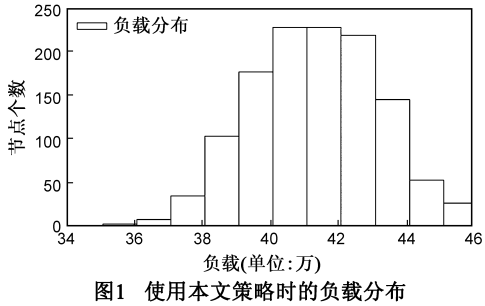

图1 使用本文策略时的负载分布

图2是从30000周期开始统计的、每隔1000单位时间被访问节点数量比较图.从图2可以看出,在不采用本文策略时,平均被访问节点数只有500左右,而在利用本文策略后,系统中每个时间段内被访问的节点数量显著增加,平均被访问节点数在800至950之间浮动.本文策略正是通过提高网络中小负载节点的利用率使网络负载更加平衡.

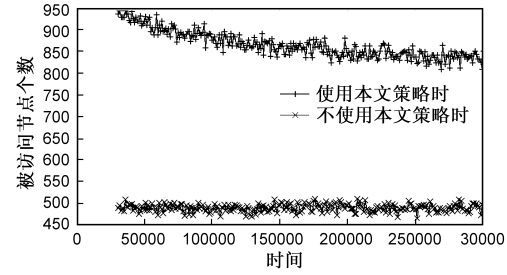

图2 被访问节点个数统计

实验2 查询分布变化时的负载分布统计

表1给出了在改变 *zipf* 分布的 α 值时所观察到的系统负载分布情况.可以看出,尽管 α 值由0.5变化到2.5,但运行300000单位时间后的节点负载比仍然可以保持在1~2之间,这说明本文策略对文件查询服从 *zipf* 分布的P2P网络有较强适应性.

表1 α 取不同值的实验结果

α	最大负载	最小负载	负载比
0.5	1904.51	1075.92	1.77
1.0	9284.08	7478.21	1.24
1.5	9428.53	6809.01	1.38
2.0	9639.31	7095.59	1.35
2.5	9932.85	7046.64	1.40

实验3 本文策略与文献[4]和文献[9]的比较

实验中将文件查询分布 *zipf* 的 α 值设为1.0,文件数量设为1000个,每个文件长度为1~10之间,网络节点数为50个,实验共进行 3×10^6 单位时间.

由图3可以看出,采用本文策略后所有节点的负载都集中在350上下,且偏移量不大.

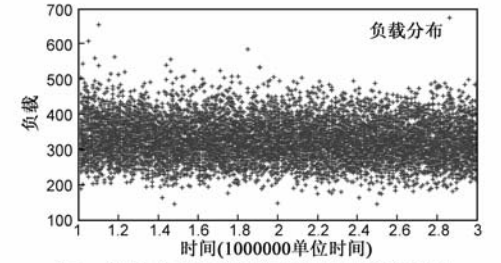

图3 基于本文策略的节点实时负载分布图

从图4可知,采用文献[9]策略的负载量呈现出不平衡性,多数节点负载较低而一部分节点负载较高.采用文献[4]的负载分布与文献[9]基本相同.

图5给出了3种策略在运行过程中的文件副本迁移量对比图.显然,迁移量越大对网络带宽的浪费就越大.可以看出,文献[4]的副本迁移量波动较大,而本文策略的副本迁移量低于文献[9]的策略.

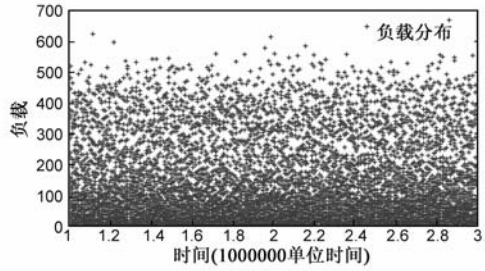

图4　基于文献[9]策略的节点实时负载分布图

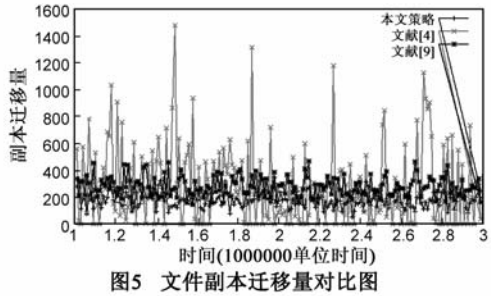

图5　文件副本迁移量对比图

6　结论与今后的工作

本文以节点实际的下载量与被下载量为基础,研究应对 DHT 网络"热点"问题的策略.从节点的下载量和被下载量管理到"热点"负载迁移,给出了适合于 DHT 网络环境的完整的解决方案.在今后的研究中,将考虑将节点的处理能力和网络传输能力作为"热节"节点和"冷节"节点的评价参数,并以此来优化本文提出的副本迁移策略.

参考文献

[1] King V, Saia J. Choosing a random peer[A]. Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing (PODC2004)[C]. New York: ACM Press, 2004. 125 – 130.

[2] Karger D, Ruhl M. Simple efficient load balancing algorithms for peer-to-peer systems[A]. Voelkerm GM, Shenker S, eds. Lecture Notes in Computer Science (LNCS)[C]. Berlin: Springer-Verlag, 2005. 131 – 140.

[3] Lua EK, Crowcroft J, Pias M, Sharma R, Lim S. A survey and comparison of peer-to-peer overlay network schemes[J]. Journal of IEEE Communications Survey and Tutorial, 2005, 7(2): 72 – 93.

[4] 熊伟,谢冬青,焦炳旺,刘洁.一种结构化 P2P 协议中的自适应负载均衡方法[J].软件学报,2009,20(3):660 – 670.
Xiong w, Xie DQ, Jiao BW, Liu J. Self-adaptive load balancing method in structured P2P protocol[J]. Journal of Software, 2009, 20(3): 660 – 670. (in Chinese)

[5] 冯国珍,张金城,顾庆,陆桑璐,陈道蓄.一种基于覆盖网络拓扑的无结构 P2P 主动复制策略[J].软件学报,2007, 18(9):2226 – 2234.
Feng GF, Zhang JC, Gu Q, Lu SL, Chen DX. An overlay topology based proactive replication in unstructured P2P systems[J]. Journal of Software, 2007, 18(9): 2226 – 2234. (in Chinese)

[6] Anwitaman D, Roman S, Karl A. Query-load balancing in structured overlays[A]. Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid[C]. USA: IEEE Press, 2007, 5. 14 – 17.

[7] Colin M. The distribution of sums of certain i.i.d. pareto variates[J]. Communications in Statistics-Theory and Methods, 2006, 35(1): 395 – 405.

[8] Rao A, Lakshminarayanan K, Surana S, Karp R, Stoica I. Load balancing in structured P2P systems[A]. Proceedings of Second International Workshop on Peer-to-Peer Systems (IPTPS)[C]. Berkeley, CA, USA: IPTPS Press, 2003, 2. 68 – 79.

[9] Sonesh S, Brighten G, Karthik L, Richard K, Ion S. Load balancing in dynamic structured peer-to-peer systems[J]. Performance Evaluation, 2006, 63(3): 217 – 240.

[10] Chen C, Tsai KC. The server reassignment problem for load balancing in structured P2P system[J]. IEEE Transactions on Parallel and Distributed Systems, 2008, 19(2): 234 – 246.

[11] 李振宁,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,2006,43(9):1579 – 1585.
Li ZN, Xie GG. A load balancing algorithm for DHT-based P2P systems[J]. Journal of Computer Research and Development, 2006, 43(9): 1579 – 1585. (in Chinese)

[12] Byers J, Considine J, Mitzenmacher M. Simple load balancing for distributed hash tables[A]. Kaashoek MF, Stoica I, eds. Lecture Notes in Computer Science (LNCS)[C]. Berlin: Springer-Verlag, 2003. 80 – 87.

[13] Ledlie J, Seltzer M. Distributed, secure load balancing with skew, heterogeneity, and churn[A]. Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)[C]. Washington: IEEE Computer Society, 2005. 1419 – 1430.

[14] 洪锋,李明禄. SChord:优化解决 Chord 覆盖网的扰动问题[J].电子学报,2005,33(12A):2361 – 2365.
Hong F, Li ML. SChord: Handling churn in chord[J]. Acta Electronica Sinica, 2005, 33(12A): 2361 – 2365.

作者简介

孟宪福　男,1960 年 2 月生于辽宁大连.副教授.研究方向为对等计算等.
E-mail: xfmeng@dlut.edu.cn