文章编号:1001-2486(2007)06-0059-06

Cocast:一种基于传感器的分布式网络距离预测任意播算法*

王意洁,符永铨,周 婧
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:针对Internet环境中基于网络坐标估计节点间网络距离的问题,提出了一个利用任意播机制分布式的网络坐标预测算法Cocast:所有用户节点均加入一个结构化Peer-to-Peer对等网;对等网中任意的节点均可以发送网络坐标任意播查询消息,该消息被转发到多个负责坐标计算的用户节点,这些用户节点分布式地预测消息发起节点坐标,最终消息发起节点获得一个综合的坐标位置。Cocast基于网络坐标任意播选择提供位置估计的用户节点,利用层次化网络嵌入预测坐标位置,借助位置融合传感器机制过滤恶意节点的影响。实验表明Cocast具有可扩展、快速收敛、渐增精度、抗恶意节点破坏等优点。

关键词:网络距离预测;网络坐标;网络嵌入;P2P对等网
中图分类号:TP393 **文献标识码**:A

Cocast: A Coordinate-fusion Based Distributed Network Distance Prediction Anycast Algorithm

WANG Yi-jie, FU Yong-quan, ZHOU Jing
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: According to the network-coordinate based network distance estimation problem in an Internet-wide environment, an anycast algorithm named Cocast for cooperative network coordinate estimation is presented. All peers join a structured Peer-to-Peer overlay, and any peer in the overlay, i.e., a requestor, may send a network coordinated anycast message, then the anycast message is forwarded to a set of peers in charge of network coordinate estimation, and each of these peers makes responses from a network coordinate to the requestor, finally the requestor gets a synthetical network coordinate. Based on three novel mechanisms, i.e., an anycast service to select peers for network coordinate estimation, a hierarchical network embedding mechanism to compute the network coordinates, and a coordinate-fusion sensor approach to filter malicious peers' fake-coordinate effects, Cocast owns a fast convergence speed in network coordinate estimation and is more stable. Simulation results confirm that Cocast is scalable, fast-convergent, incrementally accurate and resilient.

Key words: network distance estimation; network coordinate; network embedding; Peer-to-Peer overlay

随着内容分发网络(Content Distribution Network, CDN)[1]、P2P(Peer-to-Peer)技术[2-3]等的发展,节点间可扩展的精确网络距离(如Round Trip Time, RTT)测量成为这些Internet网络应用性能的一个关键因素。基于网络坐标的方式预测网络距离(又称为网络嵌入, Network Embedding)得到广泛关注:每个节点在一个坐标空间中获得一个坐标位置,不同节点间的网络距离通过坐标距离来近似[4-6]。

网络嵌入最早由Ng等人[4]提出,他们的方式(称为GNP)是利用一组地标节点作为静态的集中探测点,基于迭代的非线性优化集中计算坐标位置。GNP中节点的坐标更新为集中式的,缺乏扩展性,此外,GNP假定地标节点是可信的。为了消除集中化地标的瓶颈效应,Vivaldi[6]基于弹簧力场的方式让用户节点动态地修正网络坐标。Vivald具有高可扩展性的同时能够及时地反映网络延迟的变化情况;然而,在动态的网络环境下缺乏稳定性,同时没有考虑恶意节点的问题。

此外,已有的基于坐标的网络距离预测[4-6]很少关注实际网络中带有三角不等性违例(Triangle

* **收稿日期**:2007-04-19
基金项目:国家部委基金资助项目;高等学校全国优秀博士学位论文作者专项资金项目(200141)
作者简介:王意洁(1971—),女,教授,博士生导师。

Inequality Violation, TIV)[7]绕道路由* 影响下的坐标计算差错问题。

针对上述已有网络算法的不足,我们提出了轻量级的分布式网络距离预测任意播算法 Cocast (Cooperative Coordinate Anycast),在 Cocast 中,用户以任意播查询的方式获取自己的网络坐标位置。Cocast 随机选择少量用户节点计算自己的位置,这些节点基于力场模拟的方式估计其他节点的位置,这种层次化的网络嵌入能够及时地适应网络延迟空间的变化,缩小恶意节点的破坏范围;Cocast 利用位置融合传感器综合多个用户节点的预测位置,以进一步减弱恶意节点的影响,同时提高坐标位置的稳定性。基于实际网络延迟数据的模拟测试表明 Cocast 具有较高的可扩展性、快速收敛能力、稳定性以及抗恶意节点破坏能力等优点。

1 Cocast 模型

Cocast 是一个基于任意播方式提供网络坐标查询的网络距离预测系统。用户发送网络坐标任意播查询 Cocast 分布式的预测用户坐标位置。Cocast 成员完全由分散化的用户节点构成。Cocast 成员均加入一个 DHT(Distributed Hashtable)分布式哈希表,如 Pastry[3]中,该 DHT 负责作为 Cocast 层次化网络嵌入的底层基础设施。

Cocast 成员按照提供的功能差别分为普通节点(Simple Node, SN)和任意播服务节点(Anycast Service Node, ASN)。Cocast 成员在加入时独立地以概率为 P 的可能成为 ASN 节点,其中 P 为 Cocast 系统参数,P 的取值需要在系统的扩展性和维护开销之间进行权衡。普通节点仅能够自主地查询自己的坐标;任意播服务节点除具有普通节点功能外,还负责响应任意 Cocast 成员(包括 SN 和其他 ASN 节点)的坐标查询请求。

2 Cocast 网络距离预测

Cocast 的主要部件包括基于 DHT 的分布式 ASN 查找子图,基于 ASN 排序启发式的网络坐标任意播服务,层次化网络嵌入和位置融合机制四部分。

2.1 基于 DHT 的分布式 ASN 查找子图

ASN 查找子图作为 Cocast 网络坐标任意播查询的基础设施,主要提供为请求者定位一组优化的 ASN 节点联系信息功能。ASN 查找子图每个节点具有一个本地缓存 Cache,以存放一组 ASN 节点联系信息。ASN 查找子图通过 ASN 发布/响应过程构建。我们选一个特定标识符 Root 作为 ASN 查找子图标识符,记"维护"该 Root 标识符的节点为根节点。过程如下:

(1)任何将成为 ASN 节点的用户 u 基于底层 DHT 路由机制发布一个包含自己联系信息的"publish"消息 M 至根节点,并设定自己为"叶"节点。

(2)消息 M 在 DHT 中沿着朝向根节点 Root 的路径转发,每个中间节点记录上一跳节点信息。

(3)最终 Root 接收到注册消息 M 后,将 M 存到本地的 Cache 中。然后随机选择本地 Cache 中一组 ASN 节点构造响应"notice"消息 R(ASN 节点数目至少为 K),并利用 DHT 路由机制发送至在发布过程中的上一跳节点。

(4)响应路径中的节点随机选择 R 中的部分 ASN 节点信息进行缓存;如果在发布过程具有上一跳节点,则继续转发消息至对应的节点,并且重新设定自己为"内部"节点。

(5)最终 u 接收到响应消息 R,将消息 R 中的 ASN 节点缓存到本地 Cache。

上述 ASN 发布/响应过程构建了一个隐含的分布式 ASN 查找子图,进一步地,由 DHT 基于标识符关键字路由机制知 ASN 查找子图不包含环路。

2.2 基于 ASN 排序启发式的网络坐标任意播查询服务

任意播查询消息包括:用户当前联系信息(IP 地址,端口,当前坐标位置,坐标差错程度等),请求的

* 绕道路由(Detour Routing):两个用户间的网络距离并不代表网络中的最短路径,这类路由由于经济、技术等因素而广泛存在于 Internet 中[7]。

ASN 节点总数 K,当前已存入消息中的 ASN 联系信息。该消息基于 Cocast 的底层 DHT 路由机制转发,中间转发节点利用本地缓存 Cache 存入消息中一定数量的 ASN 节点联系信息。在每个任意播消息转发节点,选取本地缓存 Cache 中与请求者当前坐标距离最远的 $\rho \times size$ 个节点存入任意播消息,其中,ρ 为 Cocast 预先设定的选取百分比,$size$ 为当前 Cache 中的 ASN 节点数量。

上述 ASN 节点选择策略主要是为了降低网络距离较近节点间 Internet 三角不等性违例[5]的影响。其中 ASN 排序启发式基于当前坐标距离,按升序对 ASN 节点排序。随着节点的坐标精确度升高,上述启发式的可信程度逐渐升高。

2.3 层次化网络嵌入

Cocast 利用层次化网络嵌入来解决坐标位置更新,主要包括 ASN-ASN 网络嵌入和 SN-ASN 网络嵌入两个过程,均采用基于"推力"模拟的 CoPush 算法,以"推"的方式预测坐标位置。

- 在高层,每个 ASN 节点 i 计算自己坐标时,发送网络坐标任意播查询获取一组 ASN 节点集合 S,然后集合 S 中的 ASN 利用 CoPush 算法预测 i 的坐标,形成了 ASN-ASN 层次的网络嵌入;
- 在底层,每个 SN 节点基于同样方式获得自己的坐标位置,从而形成 SN-ASN 层次的网络嵌入。

CoPush 过程如图 1。CoPush 算法中,c 为移动因子,我们采用 Vivaldi[6]推荐的 0.25。层次化网络嵌入的有效性依赖于 ASN 的坐标位置精确度,利用 CoLandmark 方式计算 ASN 节点的初始化坐标以提供相对较高的初始坐标位置。

CoPush(*remote*)

/* *remote* 包含三个域:节点联系信息;当前坐标 $remote.\overrightarrow{Coordinate}$;当前坐标相对差错 $remote.e$ */

/* ASN 节点利用 CoPush 估计请求者的新坐标位置 */

1.获取请求者的当前坐标位置 $remote.\overrightarrow{Coordinate}$。

2.计算本节点坐标与请求者坐标距离 $CoordinateDis = \| this.\overrightarrow{Coordinate} - remote.\overrightarrow{Coordinate} \|$。

3.计算当前移动方向的单位向量 $unitVector = this.\overrightarrow{Coordinate} - remote.\overrightarrow{Coordinate}$。

4.测量距用户的网络延迟 *latency*。

5.$\delta = c \times \frac{remote.e}{this.e + remote.e}$ //其中 c 为常数,$this.e$ 代表本节点的当前坐标相对差错

6.计算请求者的新坐标位置

$$remote.\overrightarrow{Coordinate} = remote.\overrightarrow{Coordinate} + \delta * unitVector * (CoordinateDis - latency)$$

7.计算估计坐标位置的差错 *diff*。

$$diff = \frac{|NewCoordinateDis - latency|}{\min(NewCoordinateDis, latency)}$$

其中,*NewCoordinateDis* 为 ASN 与查询者新坐标的距离。

图 1 CoPush 算法

Fig.1 The algorithm of CoPush

CoLandmark:ASN 查找子图根结点调用 CoLandmark 过程计算部分的 ASN 初始坐标,该调用过程仅执行一次。过程如下:

如果 Root 检查自己未执行过 CoLandmark 过程且当前缓存中 ASN 节点数量超过 K,则启动 CoLandmark 计算(K 取值要大于当前维数)。步骤如下:

(1) Root 从本地 ASN 缓存中随机选择数量为 S 的 ASN 节点集合 M,通知 M 中的节点互相探测,其中 S 应当大于当前维数。

(2) M 中节点将探测网络延迟信息直接发送至 Root,Root 基于单纯形下山法[8]计算 M 中节点坐标,具体为最小化 $\sum_{i,j\in\{1,\cdots,|M|\},i>j}\left(\frac{d_{i,j}-\hat{d}_{i,j}}{d_{i,j}}\right)^2$。其中,$d_{i,j}$为节点 i 和 j 之间的网络距离,$\hat{d}_{i,j}$为节点 i 和 j 的坐标空间距离;并且 Root 在本地缓存中标记 M 中节点为"magic",并沿 ASN 查找子图广播 M 中节点的初始位置信息。

(3) ASN查找子图中节点缓存消息中"magic"节点,以满足后来加入的ASN节点初始化需求。

2.4 位置融合机制

Cocast采用位置融合传感器来提高网络嵌入过程的容错性和稳定性。每个ASN节点 i 位置估计的响应消息包括:ASN节点当前坐标 pos_i,用户的新坐标 $npos_i$,当前坐标距离与网络距离的偏差度 $diff_i$。其中,$diff_i$ 表示新坐标与底层物理网络匹配的程度,计算过程如图1步骤6。

我们具体考虑三类位置融合传感器:

(1)最小偏差度(Min Sensor)。选取响应消息中偏差度最小的坐标位置。

$$npos_i \wedge \min(diff_i), \quad i \in [1, K]$$

(2)中值(Median Sensor)。选取所有坐标的中间位置。

$$median(\{npos_i^1 : i \in [1, K]\}) \wedge \cdots \wedge median(\{npos_i^{\dim} : i \in [1, K]\})$$

其中,$npos_i^m$ 为 $npos_i$ 坐标的第 i 维。

(3)算术平均(Average Sensor)。选取所有坐标的平均位置,即 $\sum_{i=1}^{K} npos_i / K$。

3 性能测试

利用p2psim[9]项目获取的Internet中约2000个DNS服务器之间的RTT延迟矩阵作为网络延迟数据。我们在freePastry[11]上实现了Cocast。具体考察三类系统:(1)GNP,随机选取Landmark(10~15),取5次实验的最优值(实验中为5次);(2)Vivaldi,每个节点随机选取其他节点进行坐标更新;(3)Cocast,随机选取50%的节点作为ASN节点,每个节点定期的发送查询和坐标更新等消息。Cocast主要参数设置如下:$P = 50\%$, $K = 10\rho = 0.1$, $size = 50$。

考察性能指标为网络坐标的相对差错(Relative Error)和相对排序损失(Relative Rank Loss,RRL)。

- 相对差错定义为:$e = \| d_{xy} - \hat{d}_{xy} \| / \min(d_{xy}, \hat{d}_{xy})$,其中,$d_{xy}$ 为节点 x 和 y 之间网络坐标的距离,而 $\hat{d}_{xy}$ 为节点 x 与 y 之间的网络距离。
- 相对排序损失为:每个节点与其他所有节点在物理网络中距离排序与在坐标空间中排序的差错比例。

我们记录每个节点到其他所有节点的相对差错以及相对排序损失。此外坐标空间的维数取值需要在精度和计算复杂度间进行权衡。Zhang等[5]发现坐标空间的维数超过2以后,典型的网络嵌入方法相对差错降低很小,因此在实验中设定坐标空间维数为2。

3.1 ASN查找子图可扩展性

设定每个ASN节点平均每隔50s发送一次在线消息,平均每隔100s发送一次任意播查询消息,统计3天时间范围内Cocast所有节点接收的消息总数,结果如图2所示。

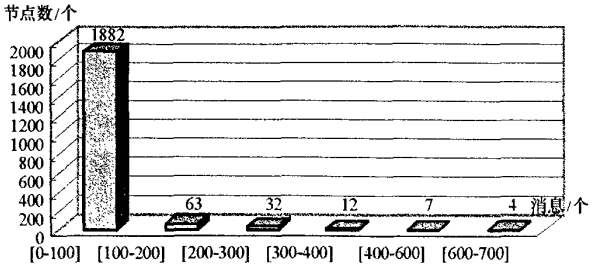


图2 Cocast节点接收消息数目分布情况

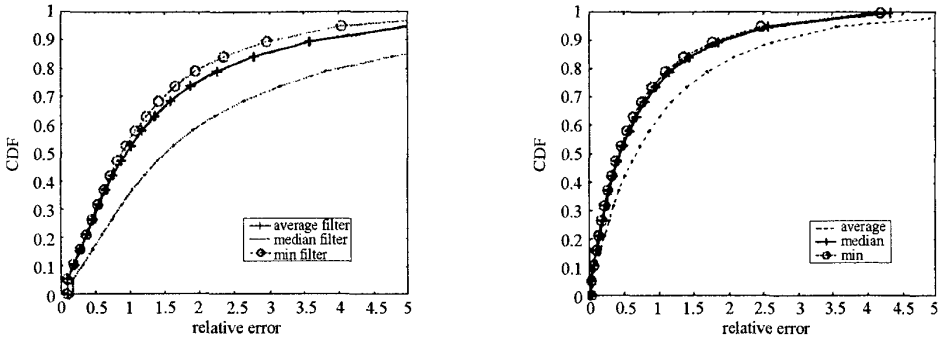
Fig.2 The received-message distribution of Cocast nodes

结果显示,约95%的节点发起以及转发的消息数目小于100,说明Cocast绝大部分节点负载较轻;然而另一方面,结果显示极少量的ASN查找子图节点的消息处理数目呈重尾分布,存在极少数节点的

消息处理数目为600~700,平均为其他节点的3~5倍,实验显示这些节点位置对应ASN查找子图的根节点周围。实验结果证明:ASN查找子图能够高效地分散化消息处理开销,具有较高的扩展能力。

3.2 不同传感器的网络坐标预测精度对比

考察三类位置融合传感器(Median Sensor, Min Sensor, Average Sensor)在没有恶意节点情况下的性能。每个节点平均进行10次探测。不同传感器的相对差错CDF(Cumulative Distribution Function,累积分布函数)分布如图3所示。



(a) 没有采用CoLandmark的Cocast相对差错分布 (b) 采用CoLandmark的Cocast相对差错分布

图3 不同位置融合传感器下Cocast的相对差错分布

Fig.3 The relative error distributions of Cocast with different coordinate-fusion sensors

结果说明:(1)Min Sensor和Median Sensor相对差错均低于Average Sensor,平均低20%以上;(2)采用CoLandmark计算ASN节点初始坐标可以有效降低节点的相对差错程度,图3(b)中每类传感器相对于图3(a)中的该类传感器平均相对差错降低约30%。

节点的相对排序损失如表1所示,结果表明:(1)采用不同的传感器对RRL影响不大;(2)在利用CoLandmark下,三类传感器的RRL有较大幅度降低,降幅将近达到1倍。说明采用CoLandmark计算ASN节点初始坐标可以降低节点的RRL。

表1 采用不同传感器下Cocast的相对排序损失(50%,90%百分位数)

Tab.1 Relative rank losses (RRLs) of Cocast with different sensors

传感器类型	不采用CoLandmark RRL(50%百分位数)	不采用CoLandmark RRL(90%百分位数)	采用CoLandmark RRL(50%百分位数)	采用CoLandmark RRL(90%百分位数)
Median	0.48	0.54	0.17	0.34
Min	0.47	0.53	0.16	0.32
Average	0.49	0.55	0.18	0.36

3.3 Cocast和Vivaldi的安全性比较

选择20%和40%比例的ASN节点作为恶意节点,恶意节点返回与实际计算偏差尽可能高的虚假位置。假定用户可以计算真正的坐标差错。设定每个用户一次请求的ASN数目为10,节点平均探测100次。

实验结果如图4所示,图中百分比代表当前恶意节点比例:(1)证实了Vivaldi的安全性较低;(2)基于传感器(Min和Median)的Cocast在40%和20%比例的恶意用户节点下其相对差错远低于Vivaldi,这是因为Cocast采用传感器有效地消除了恶意用户制造的"噪音";(3)Min传感器比Median传感器更能消除恶意节点的影响,这是因为Min传感器考虑的是选择相对差错最小的坐标,而Median传感器则选择所有节点坐标位置的"中心"。

3.4 Cocast与GNP,Vivaldi坐标系统的精确度比较

设定Cocast和Vivaldi平均探测次数为10次和100次,而GNP则静态地选择多次计算的最优值。统

计三者90%百分位数下的平均相对差错以及对应的坐标位置。实验结果如图5所示。

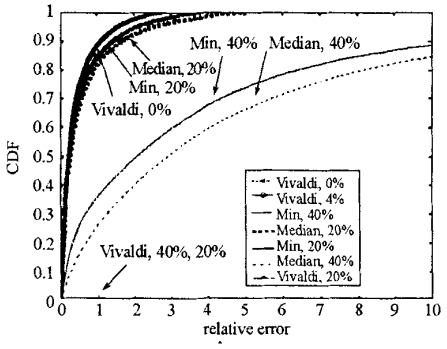


图4 恶意节点下Cocast与Vivaldi相对差错CDF分布

Fig.4 Relative error distributions with malicious nodes

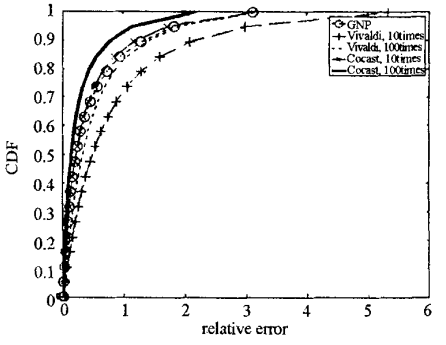


图5 Cocast, GNP以及Vivaldi相对差错CDF分布

Fig.5 Relative error distributions of Cocast, GNP and Vivaldi

结果显示:(1)在10次探测时,Cocast 90%百分位数的相对误差低于对应Vivaldi约40%,这是因为Cocast采用了CoLandmark计算ASN初始坐标;(2)而在平均探测次数为100时,Cocast 90%百分位数的相对差错低于对应的Vivaldi约20%,同时低于GNP约20%,说明采用Min Sensor可以有效地降低节点的相对差错。

4 结论

本文提出了一个轻量级的分布式网络坐标任意播服务Cocast。Cocast具有以下特点:

(1) 利用分布式的层次化网络嵌入提高了网络坐标计算的收敛速度以及健壮性;

(2) 在坐标更新时利用网络坐标任意播查询的方式,提高了坐标更新的透明性,而且中间转发节点可以对网络坐标预测节点选择进行优化;

(3) Cocast利用位置融合传感器对用户的位置估计进行综合,在很大程度上消除了部分恶意节点的虚假坐标影响,完全消除恶意节点的影响仍然需要进一步的研究。

基于实际网络延迟数据集的实验表明我们的算法具有可扩展性、快速收敛、渐增精度和抗恶意节点破坏等优点。下一步的工作包括检验更有效的位置融合传感器以及ASN节点选择策略等。

参考文献:

[1] Su A, Choffnes D, Kuzmanovic A, et al. Drafting behind Akamai (Travelocity-based Detouring)[C]//Proc. ACM SIGCOMM, 2006.

[2] Ramasubramanian V, Sirer E G. The Design and Implementation of a Next Generation Name Service for the Internet[C]//Proc. ACM SIGCOMM, 2004.

[3] Rowstron A I T, Druschel P. Pastry: Scalable Decentralized Object Location, and Routing for Large-scale Peer-to-peer Systems[C]//IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany, 2001.

[4] Ng T S E, Zhang H. Predicting Internet Networking Distance with Coordinates-based Approaches[C]//Proceedings of IEEE INFOCOM, June 2002.

[5] Lee S, Zhang Z L, Sahu S, et al. On Suitability of Euclidean Embedding of Internet Hosts[C]//Proceedings of ACM SigMetrics, 2006.

[6] Dabek F, Cox R, Kaashoek F, et al. Vivaldi: A Decentralized Network Coordinate System[C]//Proceedings of ACM SIGCOMM 2004, Portland, OR, August 2004.

[7] Zheng H, Lua E K, Pias M, et al. Internet Routing Policies and Round-trip-times[C]//The 6th Anuual Passive and Active Measurement Workshop, Boston, MA, March 2005.

[8] Nelder J A, Mead R. A Simplex Method for Function Minimization[J]. Computer Journal, 1965, 7:308-313.

[9] Gil T, Li J, Kaashoek F, et al. Peer-to-peer Simulator[CP]. http://pdos.lcs.mit.edu/p2psim, 2003.

[10] Medina A, Lakhina A, Matta I, et al. BRITE: An Approach to Universal Topology Generation[C]//Proceedings of MASCOTS 2001, Cincinnati, OH, August 2001.

[11] Druschel P, Rowstron A, Hu Y, et al. FreePastry[CP]. http://freepastry.rice.edu/FreePastry/, 2007.