中国科学: 信息科学 2012 年 第 42 卷 第 5 期: 561 577
www.scichina.com info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS
论 文

DKNNS: 面向延迟敏感型应用的可扩展精确分布式 K 近邻搜索算法研究

符永铨*, 王意洁

国防科技大学计算机学院并行与分布处理国家重点实验室, 长沙 410073
* 通信作者. E-mail: quanyongf@gmail.com

收稿日期: 2010 12 31; 接受日期: 2011 03 08
国家重点基础研究发展计划 (批准号: 2011CB302601)、国家自然科学基金 (批准号: 60873215)、湖南省自然科学杰出青年基金 (批准号: S2010J5050) 和高等学校博士学科点专项科研基金 (批准号: 200899980003) 资助项目

摘要 为了降低用户访问延迟, 延迟敏感型网络应用需要选择合适的邻近服务节点响应用户访问请求. 分布式 K 近邻搜索通过可扩展的选择距任意用户节点邻近的 K 个服务节点, 可以有效满足网络应用延迟优化的目的. 已有工作在精确度以及可扩展性等方面存在不足. 针对可扩展精确的 K 近邻搜索问题, 文中提出了分布式 K 近邻搜索方法 DKNNS (distributed K nearest neighbor search). DKNNS 将大量的服务节点组织为邻近性感知的多级环, 通过最远节点搜索机制选择优化的 K 近邻搜索初始化节点, 然后基于回退方式快速的在目标节点邻近区域发现 K 个近邻. 基于理论分析, 模拟测试以及真实环境下的部署实验发现, 在不同规模的节点集合下, DKNNS 算法能够确定近似最优的 K 个服务节点. 且 DKNNS 的查询延迟, 查询开销均显著低于 Meridian 算法. 最后, DKNNS 的返回结果相对于 Meridian 具有较高的稳定性.

关键词 延迟敏感型网络应用 K 近邻搜索 网络坐标 网络测量 分布式系统

1 引言

随着 Internet 技术的发展, 延迟敏感型网络应用近几年得到广泛关注, 例如语音 IP(VoIP)、内容分发网络 (CDN)、在线商务交易、在线网络游戏等. 这类应用的典型特征是强调信息的实时性, 信息传输过程遇到的高延迟严重影响实时信息的时效性 [1]. 例如, Google 声称增加 500 ms 的搜索结果返回延迟, 其利润下降 20%; 而 Amazon 报告称增加 100 ms 页面响应延迟, 其销售额下降 1%[2]. 在线网络游戏同样对网络延迟较为敏感, 研究人员发现在端到端延迟超过 200 ms 后游戏质量严重下降 [3]. 此外, 语音 IP 对网络延迟较敏感, 高延迟或延迟波动严重影响语音通话质量, 因此 Cisco 建议语音 IP 和视频会议端到端单向延迟不超过 150 ms, 延迟波动不超过 30 ms[4]. 为了优化延迟敏感型应用的服务质量, 有必要可扩展的测量延迟敏感型应用中服务器以及用户间的网络延迟距离, 选择距用户节点延迟距离最近的服务器提供访问服务. 由于用户节点可能辅助服务器节点提供延迟敏感型应用访问服务 (例如 Skype 应用), 以降低服务器负载以及流量开销, 因此下文将提供网络应用访问服务的服务器或用户节点统一称为服务节点.

引用格式: 符永铨, 王意洁. DKNNS: 面向延迟敏感型应用的可扩展精确分布式 K 近邻搜索算法研究. 中国科学: 信息科学, 2012, 42: 561 577

网络距离测量包括面向距离绝对值探测的绝对距离测量, 和面向邻近节点搜索的相对距离测量 [5,6]. (1) 绝对延迟距离测量提供端到端延迟绝对值, 有助于监测延迟距离波动, 但无法直接实现最优服务节点选择. 为了实现最优服务节点选择, 绝对距离测量需要探测每个服务节点与用户间延迟距离, 导致可扩展性不高. 为了降低探测开销, 网络坐标 [1,7~12] 通过坐标距离预测服务节点与用户节点间网络延迟距离, 然而其精确度受网络延迟三角不等性违例制约. (2) 相对距离测量以直接定位距指定目标节点较近的服务节点为目的, 通过设计可扩展的邻近节点搜索算法快速求解邻近服务节点, 只需要少量的绝对距离探测即可完成邻近节点搜索 [5,6], 故其响应时间和探测开销较绝对距离探测显著降低, 因此相对距离测量更适用于延迟敏感型应用最优服务节点选择需求.

针对相对网络距离测量问题, 本文将该问题一般化为分布式系统中 K 近邻搜索问题, 下文简称为 K 近邻搜索. 给定一个具有动态性的服务节点集合, K 近邻搜索从服务节点集合中可扩展的寻找距任意用户节点网络延迟最近的 K 个节点集合. 其中 K 为系统参数, K 为 1 即为最近服务节点搜索. 而 $K>1$ 时, 搜索过程返回距用户节点最近的 K 个服务节点, 有利于上层应用设计灵活的服务节点筛选策略, 从而选择综合性能最好的邻近服务节点.

与相对距离测量相关的研究按服务节点选择过程可分为集中式排序 [13~17] 和分布式搜索 [18~22] 两类. 集中式排序需要预先测量服务节点与用户节点间邻近度, 导致测量开销较大, 且存在单点失效问题, 故可扩展性不高; 同时, 在服务节点集合动态的加入/退出时, 集中式排序需要频繁的更新服务节点信息, 容易产生网络通信热点地区, 影响了系统的健壮性. 而分布式搜索通过服务节点节点协作的方式发现任意目标节点的邻近服务节点, 避免了网络通信瓶颈, 降低了测量开销. 然而由于网络延迟空间存在显著的分簇现象, 已有的分布式搜索算法容易陷入局部最优值 [5]; 其次, 由于网络测量过程涉及节点范围广, 导致探测测量开销仍然较大, 而且搜索过程历时较长, 从而降低了其适用范围.

针对已有研究的不足, 本文提出了一个分布式 K 近邻搜索算法 DKNNS(distributed K nearest neighbor search). DKNNS 算法将服务节点组织为轻量级的无结构化覆盖网, 每个服务节点均可以响应任意用户节点提出的 K 近邻搜索请求. 为了适应网络延迟空间分簇现象, 每个服务节点通过低开销的流言 (gossip) 通信和邻近节点搜索过程, 快速的发现位于网络延迟空间各个距离尺度的邻居节点集合. 因此在 K 近邻搜索时, 每个服务节点均能够利用邻居集合分布式的发现距目标节点最近的邻居. 为了快速的完成 K 个近邻搜索, 在 K 近邻搜索时, 首先, 利用最远节点搜索机制选择 K 近邻搜索的发起节点, 避免距目标过近导致的搜索过早中止问题; 其次, DKNNS 利用分布式搜索结合快速回退的方式在目标节点邻近区域迅速确定 K 个近邻节点.

理论分析发现 DKNNS 能够以 $O(K\log\Delta)$ 跳步发现 K 个近似最近邻居, 其中 Δ 为服务节点间最大延迟距离. 基于模拟测试发现, DKNNS 算法能够确定近似最优的 K 个服务节点, 在 K=10 时, 与真实 K 近邻的重合率超过 80%. 同时 DKNNS 引发的查询延迟和查询开销并不随着系统规模增大显著提高. 基于 PlanetLab 部署实验证实了 DKNNS 得到的近邻集合接近最优结果, 超过 50% 比例的查询时间不到 10 s, 平均查询带宽开销不到 5 KB, 且 K 近邻信息具有稳定性, 能够满足延迟敏感型应用的服务节点选择需求.

2 相关工作

Intenet 网络环境下 K 近邻搜索相关的研究按照搜索过程的消息通信方式可以分为集中式和分布式两类. 集中式 K 近邻搜索需要所有服务节点与目标节点的延迟信息, 利用集中式排序的方式选

择距目标节点最近的 1 个或多个服务节点. 然而, 集中式过程可扩展性低, 在服务节点/用户节点规模上升时, 容易引发性能瓶颈. 另一方面, 分布式 K 近邻搜索通过服务节点协作的发现距目标节点邻近的服务节点. 分布式搜索过程避免了网络通信热点地区, 提高了可扩展性, 然而, 由于网络延迟空间不满足三角不等性且存在分簇现象 [23], 分布式搜索过程容易陷入局部最优.

(1) 集中式邻近性搜索研究. iPlane 系统构建近似 Internet 结构的拓扑模型 [13,14], 然后通过拓扑模型预测节点集合之间的邻近性. Netvigator 系统利用低维度的坐标表达节点集合之间的邻近性, 降低了邻近信息存储开销 [15,16]. iPlane 和 Netvigator 面向通用的邻近性节点选择服务, 而 Binning[17] 面向拓扑感知的覆盖网构建应用, 通过多个逻辑桶 (Bin) 的方式将邻近的节点映射到相同的桶内.

(2) 分布式 K 近邻搜索. Mithos[18] 通过一个覆盖网组织节点集合, 并利用梯度下降的方式搜索距目标邻近的节点, 搜索过程可能陷入局部最优值. PIC[19] 基于欧氏网络坐标的方式近似节点之间的距离, 然后通过贪心的方式搜索最近的 K 个邻居节点作为 K 近邻. 由于搜索结果受到网络坐标的近似性影响 [7], PIC 难以保证搜索质量. Meridian[20] 利用基于多级环状覆盖网结构搜索 1 近邻信息, 受到网络延迟空间分簇的影响, Meridian 难以搜索到真正最近邻 [23]. OASIS[21] 利用地理坐标距离将用户访问请求转发到最近的服务节点, 降低了测量开销, 然而由于地理距离并不严格对应延迟信息, 因此 OASIS 难以保证降低用户与服务节点的交互延迟.

已有的邻近节点搜索研究没有考虑额外的约束条件, 只是选择近似最近的一个或者多个服务节点. 针对多约束条件下分布式服务节点选择问题, DONAR[22] 提出部署独立的映射节点集合提供综合多种约束条件的服务节点选择功能. DONAR 具有灵活的约束条件表达接口. 然而 DONAR 映射节点基于地理位置近似用户距服务节点的延迟, 容易因地理位置与实际延迟的不匹配引发较高的用户访问延迟.

另一方面, 数据库领域和计算机理论科学领域 K 近邻搜索主要关注于度量空间背景下 K 近邻搜索算法设计与分析, 相关研究可参考综述 [24~27]. 度量空间背景下的研究与本文研究具有互补关系, 然而由于网络延迟参数并不满足度量空间的三角不等性假设条件, 因此度量空间背景下研究结论并不直接适用于网络延迟空间.

3 问题定义

由于延迟敏感型网络应用中服务节点集合和目标节点集合具有大规模、动态性特点, 分布式实现 K 近邻搜索, 避免网络性能瓶颈, 成为一个迫切的研究问题. 分布式 K 近邻搜索可以通过如下方式定义:

定义 1(分布式 K 近邻搜索) 假设一个分布式系统中存在一个动态的服务节点集合 S, 给定 Internet 中任意的节点 T, 目标是基于服务节点集合分布式协作, 从 S 中寻找距 T 网络延迟最小的 K 个节点集合 S_1, K 为系统参数.

由定义 1 可知, 目标 T 的真实 K 近邻可能并不唯一. 在分布式环境下, 由于每个节点只有少量的邻居节点集合, 难以完全收集所有服务节点距目标节点的距离信息, 导致分布式 K 近邻搜索过程得到的结果容易陷入局部最优值. 为了量化搜索到的 K 个近邻距目标节点的邻近性程度, 我们通过近似度来量化分布式 K 近邻搜索的精确度.

定义 2(Θ 近似) 假设基于分布式 K 近邻搜索机制得到 K 个近邻节点集合 S_1, 目标 T 的真实 K 近邻节点集合为 S_2, 若 S_1 距 T 的距离和与 S_2 距 T 的距离和比值不大于 Θ, 即 $\frac{\sum_{i\in S_1} d_{iT}}{\sum_{i\in S_2} d_{iT}} \leqslant \Theta$ 则

称该分布式 K 近邻搜索结果为 Θ 近似, $\Theta \geqslant 1$.

在 Θ=1 时, 搜索到的 K 个近邻 S_1 就是目标 T 的一组真实 K 近邻. 可以利用反证法证明若将 S_1 中每个节点按距目标 T 的距离升序排序, 则序列中第 i 个节点距目标节点的距离与第 i 个真实近邻距目标节点的距离相等. 否则, 可以得到一个距目标节点距离和更小的 K 近邻序列, 这与 S_2 为真实的 K 近邻前提矛盾.

4 DKNNS 算法

针对网络延迟空间的三角不等性违例, 以及分簇现象, 为了支持灵活的 K 近邻搜索服务, 本文提出了分布式 K 近邻搜索算法 DKNNS (distributed K nearest neighbor search). DKNNS 算法不依赖于三角不等性假设, 而是基于低度量模型 (inframetric model)[28], 构建反映节点间分簇特征的邻居列表多级环, 然后, 递归的搜索多级环上距目标节点最近的服务节点, 查询 1 个近邻, 然后通过快速的搜索回退过程, 查询剩余的 $K-1$ 个近邻, 从而实现面向任意目标的 K 个最近服务节点搜索服务.

4.1 多级环维护

(1) 发现新服务节点. 由于分布式搜索过程中当前服务节点逐步靠近目标节点, 因此服务节点不仅需要维护网络延迟空间典型区域的服务节点集合, 而且需要维护与本节点邻近的服务节点集合. 为了区分各个距离范围, 将每个节点维护的逻辑邻居组织为一个以本节点为圆心, 包含固定数目环的多级环. 多级环按距本节点延迟指数级增加的方式将逻辑邻居放入不同的环内. 第 i 个环存放距本节点延迟位于 $(2^{(i-1)}, 2^i]$ 区间的邻居节点, 第 1 个环维护 $(0, 1]$ 范围内的逻辑邻居. 每个节点的存储信息包括物理地址, 距本节点实际 RTT 延迟以及该节点坐标等. 每个环上维护的节点数目具有一个上限 m.

由于单纯利用 gossip 方式难以快速发现距服务节点邻近的节点集合, 我们引入邻近节点搜索过程选择靠近本节点的逻辑邻居作为环上节点. 发现新节点包括: 1) gossip 过程: 每个节点均匀随机的从环上选择一组逻辑邻居, 构建包含其地址信息的 gossip 消息, 然后从环上选择一个邻居, 将该消息发送给该邻居; 2) 邻近节点覆盖: 每个节点定期的发送以本节点为目标节点的 K 近邻搜索消息, 并将 K 近邻搜索到的近邻节点放入本节点环上. 默认 K 值为 20.

(2) 维护多级环. 为避免因为多级环过大而产生的较高的网络探测开销, 每个环具有上限 m, 如果超过了该上限, 则执行环维护过程. 为了避免在环达到 m 后, 频繁环维护过程引发高计算开销, 我们设定一个环维护阈值 m_1. 即只有在环的大小大于 $(m+m_1)$ 后执行环维护, 并且每次环维护删除掉对应环上多余的 m_1 个邻居. 针对最小化环维护延迟以及提高环上节点覆盖率的问题, 我们基于待维护环上节点的坐标向量信息, 利用 Meridian 算法的贪心式最大化超体积多面体 (maximal hypervolume polytope) 算法, 选择节点间距离最大的节点子集作为多级环上节点. 该算法可以在线性时间完成多余的邻居删除.

由于最大化超体积多面体算法需要环上节点之间的网络延迟信息, 为了避免环上节点集合相互探测导致的等待延迟, 我们采取网络坐标的方式利用坐标距离近似节点间延迟. 每个节点为自己计算一个低维度的欧氏空间坐标向量, 并通过 gossip 通信过程请求逻辑邻居的坐标信息, 并在得到坐标信息后利用 gossip 过程的往返时间近似 RTT, 触发本节点的坐标更新过程.

坐标更新时, 当前节点利用当前缓存的每个逻辑邻居节点坐标位置, 采取弹簧力场方式 [8] 计算坐标更新的方向以及更新的幅度, 更新本节点的坐标位置 (如算法 1 TIV-Vivaldi 所示). 由于网络延迟

空间的三角不等性违例影响了 Vivaldi 精度 [1,10], 因此 TIV-Vivaldi 增加了对三角不等性违例的适应能力. 1) 绕开容易引发三角不等性违例的延迟. 我们基于 Wang 等 [10] 提出的三角不等性预警机制, 在通信双方节点的坐标精确度较高而当前网络延迟与预测距离差错较大时, 不更新坐标位置. 2) 降低三角不等性违例影响幅度. 在坐标收敛过程中, 发生三角不等性违例时, 如果坐标相对差错 ϵ 大于预设的阈值, 则将 ϵ 降低为一个默认差错值, 从而降低三角不等性违例对于坐标精度参数 e_i 的影响.

算法 1: 坐标向量计算过程

TIV-Vivaldi(rtt, x_j, e_j):

Input: rtt, x_j, e_j

* node i (coordinate x_i, coordinate errore_i) neighbor j's rtt, neighbor j's coordinate x_j and the coordinate error e_j, threshold is used to determine whether a TIV occurs, with default value as 0.3, Ω denotes the di erence between the real latency and the coordinate distance that are used for detecting TIVs, with default value as 100 ms, MAXERROR is the maximal relative error of the coordinate distances, with default value as 2, COORDERROR and COORDCONTROL are the constants for controlling the update magnitude of the coordinate update, with default values both as 0.25.

Output: x_i

sampleWeight=$e_i/(e_i + e_j)$;

if ($e_i <$ threshold) AND ($e_j <$ threshold) **then**

  if $|(\text{rtt} - \|x_i - x_j\|)| > \Omega$ **then**

    return;

sampleError=rtt$-\|x_i - x_j\|$;

ε=|sampleError|/rtt;

if $\epsilon >$MAXERROR **then**

  ϵ=MAXERROR

α=sampleWeight*COORDERROR;

e_i=ϵ*α+$(1-\alpha)$*e_i;

x_i=x_i+COORDCONTROL*sampleWeight*sampleError*$u(x_i - x_j)$;

4.2 K 近邻搜索过程

为了快速靠近目标节点, 我们基于贪心搜索的思想进行分布式的迭代搜索: 当前节点根据多级环上逻辑邻居距目标节点的延迟信息, 每次选择的下一跳步节点距目标需要比当前节点近 β 倍, 从而以指数级速度逼近目标节点; 在搜索到 1 个近邻后, 由于两个相同发起节点的贪心搜索过程中间经历的节点集合序列相同, 仅区别于最后跳步的节点, 故在目标节点的邻近区域, 继续发现剩余的近邻节点.DKNNS 算法描述如下:

(1) 目标节点坐标计算: 为了快速发现距目标节点的邻近节点集合, 降低搜索过程延迟, 我们基于坐标距离预测的方式进行选择. 为了得到准确的目标节点坐标, K 近邻搜索的最初请求节点 P 初始化目标节点坐标位置, 然后每个 K 近邻搜索的中间节点更新目标坐标位置. K 近邻搜索的发起节点 P 首先向目标节点请求坐标位置, 如果目标节点维护了坐标位置, 则通过请求得到目标节点的初始坐标位置, 否则, 发起节点 P 随机选择 L 个邻居节点探测距目标节点的延迟, 然后利用该邻居节点集合的坐标位置信息, 基于算法 1 迭代的计算目标节点的初始化坐标, 迭代轮数设定为 15. 每个 K 近邻搜索的中间查询节点 I 在接收到 K 近邻搜索消息 M 后, 从 M 中提取目标节点的坐标位置以及差错

值. 然后 I 探测距目标节点的延迟, 并基于本节点的坐标位置, 基于算法 1 更新目标节点的坐标位置. 转入步骤 (2).

(2) 最远发起节点搜索: 为了避免因为搜索发起节点距目标节点较近导致无法搜索足够数目的近邻, 首先发起一个 K 近邻初始化节点选择过程, 选择距目标节点最远的节点发起 K 近邻搜索过程. 由于一个包含很少节点信息的逻辑邻居难以提供距目标节点的邻近信息, 因此需要被排除在 K 邻近性搜索过程之外. K 近邻搜索的最初请求节点 P 首先选择非空环数目不小于 R 的候选节点子集, 然后从中选择距目标坐标距离最远的 L 个节点直接探测距目标节点的 RTT 延迟, 然后判断返回的 RTT 值是否存在大于本节点距目标节点距离阈值 $\beta_f(\beta_f$ 默认为 1.2) 倍的节点 (设为 P_2): 若存在, 则将最远搜索请求发给 P_2, 然后 P_2 继续递归的搜索; 否则, 当前节点 P 停止最远发起节点搜索过程, 并选择当前距目标最远的节点 (设为 P_F) 作为 K 近邻搜索的发起节点. P_F 开始执行 K 近邻搜索过程. 转入步骤 (3).

(3) 最近候选节点选择: 假设当前节点 P_F 距目标节点的坐标距离为 d_T, 当前节点从距本节点坐标距离范围在 $[0, \rho \times d_T]$ 的逻辑邻居集合中 (ρ 为低度量模型 (inframetric model) 参数, 默认为 2.5, 低度量模型介绍详见 5.1 小节), 选择不在已经搜索近邻节点集合且非空环数目不小于 R 的候选节点子集, 然后选择 L 个最靠近目标节点的候选节点. 实验发现 R 为 4, L 为 4 时精确度较高. 接着, 每个候选节点主动的探测距目标节点的距离, 然后节点 P_F 根据直接探测距离从候选节点中选择距目标节点最近的节点 (设为 P_C). 转入步骤 (4).

(4) 最近搜索判断: 首先, 判断如果 P_C 距目标节点的距离小于当前节点距目标节点距离乘以参数 $\beta(\beta$ 小于 1, 称为下一跳步选择阈值), 则当前节点将 K 近邻搜索消息发送给 P_C, P_C 继续进行 K 近邻搜索, 当前节点成为 P_C 的 K 近邻搜索父节点, 转入步骤 (3); 否则当前节点选择距目标节点最近的邻居 (P_C 或者 P_F) 为距目标节点最近的一个近邻, 并将该节点缓存到 K 近邻搜索消息. K 近邻搜索消息中缓存的每个近邻节点标记为已经搜索近邻节点, 已搜索过节点不再参与当前的 K 近邻搜索过程. 转入步骤 (5).

(5) K 近邻搜索完成判断: 若已经得到 K 个近邻, 则搜索过程结束; 否则若当前节点 P_F 没有父节点, 则进入步骤 (7) 搜索重启过程, 否则, 进入步骤 (6) 进行回退搜索.

(6) 回退搜索: 当前节点将 K 近邻搜索消息发送至上一跳步父节点, 父节点从候选节点集合中排除当前已搜索节点, 进入步骤 (3) 进行继续搜索.

(7) 搜索重启: 当前节点从多级环上选择距目标节点最远且不在已搜索过节点集合的逻辑邻居 P_N, 然后将 K 近邻搜索请求发送给 P_N, P_N 从候选节点集合中排除当前已搜索节点, 执行步骤 (3).

5 DKNNS 算法性能分析

(1) 预备知识. 我们利用 d_{uv} 代表 u 与 v 之间的延迟. $B_u(r)$ 为以节点 u 为圆心, r 为半径包含节点构成的闭球, 定义 $B_{ui} = B_u(2^i)$. 每个节点 u 维护 $\log(\Delta)$ 个环, 其中 Δ 为最大距离, 每个环 $S_{ui} \subset B_{ui} \setminus B_{(u,i-1)}$, 且环的大小为 m.

定义 3(良好)　称一个环 S_{ui} 为良好 (well-formed), 如果 S_{ui} 作为 $B_{ui} \setminus B_{(u,i-1)}$ 的 m 个随机节点构成的节点子集.

定义 4(ϵ-nice[20])　称多级环为 ϵ-nice, 如果对于任意的节点对 u, v, 节点 u 的多级环具有一个邻居节点 $w \subset S_{ui}$, 使得 $d_{wv} \leqslant \epsilon * d_{uv}$, $d_{uv}(1+\epsilon) \leqslant 2^i$.

本文我们采取低度量模型 (inframetric model)[28] 对网络延迟空间进行分析. 一个距离函数 d: $V \times V \longrightarrow \mathbb{R}$ 为 ρ- 低度量 (inframetric) (其中 $\rho > 1$), 如果 d 满足下述关系: 1) $d(u,v) = 0$ 当且仅当 $u = v$; 2) $d(u,v) = d(v,u)$; 3) 对于任意的三元组 u, v, w, $d(u,v) \leqslant \rho \max\{d(u,w), d(w,v)\}$. 增量维度以及倍增维度是描述网络延迟空间的常见工具, 二者用于描述一个空间是否可以有效的通过少量小半径的球覆盖大范围的空间, 这可以用于判断是否可以利用少量的采样点来发现与目标节点位于相同小半径球问题, 若可行, 则通过这种采样方式可以递归的求解距指定目标节点临近的节点, 从而用于指导邻近搜索算法的设计.

低度量松弛了度量空间对于网络延迟空间三角不等性的约束, 而是利用 ρ 来描述任意的三元组的边之间的关系, 因此更能适应不满足三角不等性的网络延迟空间. 在低度量模型下, 增量维度以及倍增维度可以被归纳表述 [28]: 1) 增量维度指的是如果对于任意的 $r \geqslant 0$, 以及 $u \in V$, 满足 $|B_u(\rho r)| \leqslant \gamma |B_u(r)|$, 称一个 ρ- 低度量 d 具有增量 (growth)$\gamma \geqslant 1$. 2) 倍增 (doubling) 维度为 γ- 倍增的, 如果对于任意的 $r \geqslant 0$, 以及 $u \in V$, 存在 $B_u(\rho r) \subseteq \bigcup_{i \in I} B_{v_i}(r)$, $v_i \in V$, $i \in I$,$|I| \leqslant \gamma$.

因此, 增量维度与倍增维度都有一个最大下界, 即满足上述不等式的最小数值; 但是二者都没有最大上界. 从实用性角度出发, 我们更关心最大下界的数值. 因此, 下文在涉及增量维度与倍增维度时, 均指的是对应的最大下界.

与增量维度、倍增维度相类似, 网格维度 [16] 表达了不同半径球之间的大小关系. 下文性质 1 和性质 2 显示, 网格维度可以统一的对低度量模型下的增量维度和倍增维度进行描述.

定义 5(网格维度 [16]) 给定一个 n 维网格, 任意的球覆盖的节点数目至多小于以同样的球心及 x 倍的半径的球覆盖节点数目的 x^α 倍. 一个网格维度 (grid dimension) 为上述属性成立的最小的 α.

本文逻辑邻居维护质量分析利用了下述引理. 该引理基于网格维度进行定义, 通过综合 Wong 等 [16] 推导的定理 4.1 可得.

引理 1 给定一个节点集合, 由节点间距离组成的距离空间具有网格维度 α, 设 $\delta \in (0,1)$, $\epsilon < 1$, γ 为常数, N 为节点集合规模, 每个环的大小为 $O(1/\epsilon)\log(N/\delta)$, 任意选择两个节点 u, v, 设 $r = \epsilon d_{uv}$, 选择最小的 i 使得 $d_{uv} + r \leqslant 2^i$, 如果存在 $|B_{ui}| \leqslant \gamma^\alpha |B_v(r)|$, 那么按照切尔诺夫界 (Cherno bound), 从环 S_{ul},$l \leqslant i$ 选择一个点, 落在 $B_v(r)$ 外面的失效概率小于 δ/N^2.

引理 1 给出了网格维度下多级环满足 ϵ-nice 的一个充分条件. 引理 1 说明在网格维度下对于系统中任意两个节点 u, v, 若以 u, v 为球心, 半径分别为 2^i 和 r 的两个球包含节点的个数比例不大于一个常数 γ^α, 那么, 从节点 u 的多级环上选择一个邻居满足距 v 的距离降低至不大于 $\epsilon \times d_{uv} (\epsilon < 1)$ 成功概率接近 1, 即多级环为 ϵ -nice 的. 为了将引理 1 结论推广到低度量模型, 下文我们建立网格维度与低度量模型下增量维度以及倍增维度与网格维度之间的关系 (性质 1 和 2), 然后利用引理 1 分析低度量模型下多级环的质量 (定理 1 和 2).

(2) 多级环质量分析. 为了分析低度量模型下多级环的质量, 首先利用性质 1 与 2 建立了增量维度、倍增维度与网格维度的关系, 然后定理 1 与 2 根据性质 1 与 2 的结论, 建立了低度量模型下多级环质量与网格维度的联系, 并根据引理 1 网格维度下多级环为 ϵ-nice 的充分条件, 推导出低度量模型下多级环为 ϵ-nice 的充分条件. 我们首先给出增量维度与网格维度之间的关系.

性质 1 如果一个 ρ- 低度量具有增量 (growth)$\gamma_g \geqslant 1$, 对于任意的 $x \geqslant \rho$, 任意的球覆盖的节点数目至多小于以同样的球心及 x 倍的半径的球覆盖节点数目的 x^α 倍, 其中 ρ- 低度量的网格维度 (grid dimension) 为 $\log_\rho \gamma_g \leqslant \alpha \leqslant 2\log_\rho \gamma_g$.

性质 1 证明过程及下文证明可见附录. 性质 1 说明低度量空间的网格维度与增量维度呈对数关

系. 我们通过实验发现实际网络延迟空间的低度量参数 ρ 绝大多数小于 3, 增量维度一般小于 10, 因此低度量空间的网格维度并不大. 其次, 我们给出倍增维度与网格维度之间的关系如下.

性质 2 设定一个 ρ- 低度量为 γ_d- 倍增的, 对于任意的 $x \geqslant \rho$, 任意的球覆盖的节点数目至多小于以同样的球心及 x 倍的半径的球覆盖节点数目的 x^{α} 倍, 其中 ρ- 低度量的网格维度 (grid dimension) 的可行区间为 $\frac{1}{4}\log_{\rho}\gamma_d \leqslant \alpha \leqslant \log_{\rho} N$, N 为节点总数.

性质 2 说明在倍增维度下网格维度的可行解是一个区间. 这是由于倍增维度是增量维度的一般化推广 [28], 覆盖整个球的子球球心位置是任意的, 而网格维度和增量维度的整个球与子球的球心位置是同一个, 导致基于倍增维度下的网格维度求解存在不确定性.

基于性质 1 与 2 推导的低度量模型与网格维度之间的关系, 我们接下来分析增量维度 (定理 1) 以及倍增维度 (定理 2) 下多级环满足 ϵ-nice 的所需条件.

定理 1 (增量维度下的多级环质量) 在 ρ- 低度量具有增量 (growth)$\gamma_g \geqslant 1$, 固定 $\delta \in (0,1), \varepsilon < 1$, 设定多级环的每个环的大小为 $O(\frac{1}{\varepsilon})^{\log\gamma_g}\log(N/\delta)$, 如果每个环为良好的, 则多级环以 $1-\delta$ 的概率为 ϵ-nice.

定理 1 说明, 如果多级环的每个环为均匀随机的从对应距离范围选择逻辑邻居, 那么, 对于任意的目标节点, 多级环中均存在一个逻辑邻居, 距目标节点的距离小于当前节点距目标节点的 ϵ 倍, $\epsilon < 1$. 因此, 多级环结构能够支持基于贪心搜索策略的 DKNNS 算法中 K 近邻搜索过程.

定理 2 (倍增维度下的多级环质量) 若一个 ρ- 低度量为 γ_d- 倍增, 固定 $\delta \in (0,1)$, $\varepsilon < 1$, 设定多级环的每个环的大小为 $O(\frac{1}{\varepsilon})^{\log\gamma_d}\log(N/\delta)$, 如果环为良好的, 则多级环以 $1-\delta$ 的概率为 ϵ-nice.

通过定理 1 和 2, 我们证明了在增量维度和倍增维度下, 只要多级环中每个环保持一定数目的随机节点集合, 那么整个多级环可以实现 ϵ-nice, 即对于任意的目标节点, 均可以从多级环中发现距目标节点距离降低至 ϵ 倍以内的邻居, 从而有效的支持最近邻搜索过程.

(3) 候选节点选择范围. 另一方面, 由于 DKNNS 算法要求搜索过程中下一跳步节点距目标节点的距离要降低至 β 倍以内, 我们需要进一步限定搜索过程中的候选节点集合. 为了不遗漏距目标节点距离降低 β 倍的逻辑邻居, 根据低度量模型定义, 我们证明 (定理 3) 当前节点需要选择距本节点距离低于 ρd 的节点集合.

定理 3 DKNNS 算法中, 如果当前节点 A 希望选择距目标节点距离上限为 βd 的节点集合 H, 则 H 中每个节点距 A 的距离 x 必满足如下条件: 如果 $\rho\beta > 1$, 则 $x < \rho d$, 而如果 $\rho\beta \leqslant 1$, 则 $d/\rho < x < \rho d$.

(4) DKNNS 搜索精度分析. 由于均匀随机的选择 $O(\log N)$ 数目的节点放入多级环的每个环中, 即可实现多级环为 ϵ-nice 的. 而我们通过实验验证, DKNNS 算法通过结合 gossip 方式和 K 近邻搜索的方式可以快速的填充多级环中的每个环. 接下来我们分析在多级环为 ϵ-nice 时, DKNNS 算法的 K 近邻搜索的精确程度. 首先, 在 K=1 时, 我们给出 DKNNS 算法寻找到的 1 近邻的近似度以及搜索效率.

定理 4 (1 近邻) 设多级环为 ϵ-nice,$\epsilon < \beta$, 则 DKNNS 算法通过 $O(\log\Delta)$ 跳步, 确定的 1 近邻为目标节点的 $1/\beta$ 近似 1 近邻, Δ 为系统中任意节点间最大距离.

定理 4 说明 DKNNS 搜索到的 1 近邻为近似最近邻居, 近似度 $\Theta = 1/\beta$. 因此提高 β 可以使得近似度接近为 1. 由于 DKNNS 算法选择候选节点集合规模与 β 无关, 因此, DKNNS 算法设置较高的 β 值并不会增大候选节点规模. 进一步地, 我们可以将定理 4 进行推广, 在多级环保持 ϵ-nice 下, DKNNS 算法寻找的 K 个近邻为近似最优的.

定理 5 ($\boldsymbol{K}$ 近邻) 假设多级环为 ϵ-nice($\epsilon < \beta$), 基于 DKNNS 的 K 近邻搜索确定的 K 个近邻

节点集合 S_1 相对于真实的 K 近邻节点集合 S_2 是 $1/\beta$ 近似, 并且搜索过程在 $O(K\log\Delta)$ 跳步完成, K 为近邻数目, Δ 为系统中任意节点间最大距离.

定理 5 说明基于回退方式的 K 近邻搜索过程可以快速的找到较为精确的近邻节点集合, 并且随着 β 接近 1, K 近邻的近似度逐渐接近为 1.

6 模拟测试

为客观对比算法性能, 我们采取邻近性搜索研究通用的模拟方式: 首先, 利用 3997 个 DNS 服务器间 RTT 延迟矩阵 [29] 作为底层物理网络静态延迟; 其次, 选择一组节点作为执行 K 近邻搜索的服务器, 其余节点为用户, 每次 K 近邻搜索随机的从系统中服务器与用户中选择 K 近邻搜索的目标节点; 模拟测试共执行 10 万次, K 近邻搜索的近邻数目在 1, 5, 10, 15, 20, 25 中进行选择. 限于篇幅限制, 下面只给出 10 近邻模拟测试结果, 其余近邻数目下的模拟测试结果与 10 近邻模拟测试结论一致.

目前公认先进的邻近性搜索算法包括 Cornel 大学提出的 Meridian 算法 (通过分布式的搜索过程发现 1 个近邻), 以及 HP 实验室提出的 Netvigator 算法 (集中式的发现距一个目标的 K 近邻). 因此针对二者进行对比测试: (1) Meridian 算法只返回 1 个近邻, 然而由于发起节点通过重复的执行 Meridian 算法就能够发现多个近邻, 因此本文对比这种重复利用 Meridian 算法进行 k 次 1 近邻搜索的邻近性搜索方法. Meridian 算法采取开发者推荐的试验配置, 同心环的每个环上节点数目上限为 10, 近邻搜索的下一跳步选择阈值 β 为 0.5. (2) Netvigator 方法需要全局范围的地标节点信息以及查询节点与全局地标节点集合路由路径上路由器作为额外的地标节点, 而本文延迟矩阵不包含路由器信息, 作为替代, 我们在设置 30 个地标节点之外, 还为每个节点随机选择 30 个节点作为额外的地标节点, 这种方法称为 quasi-Netvigator. (3) 本文 DKNNS 算法下一跳步选择阈值 β 为 0.9, 最远截断阈值 β_f 为 1.2, 多级环的每个环上最多允许 10 个邻居. 坐标维度设定为 5. 设定低度量参数 ρ 为 2.5(基于 3997×3997 延迟矩阵发现超过 98% 的低度量 ρ 小于 2.5).

模拟测试对比性能指标包括: (1) 平均绝对差错, 定义为预测的 K 近邻与真实最近邻延迟差异绝对值与 K 的比值; (2) 平均消息负载, 定义为 K 近邻搜索产生消息的规模, 模拟测试中设定每个数据包为 50 Byte; (3) 发现真实 K 近邻的比例, 定义为预测的 K 近邻中包含真实 K 近邻所占的比值; (4) 平均查询时间, 定义为 K 近邻搜索所需的搜索时间, 其中 Meridian 与 DKNNS 的搜索时间为分布式搜索过程的延迟, 而 quasi-Netvigator 为探测到 30 个全局地标节点的延迟 (由于额外地标节点模拟路由器节点集合, 假定已经在探测距全局地标节点延迟时稍带测量).

(1) 搜索时间对比. 我们对比不同方法的查询延迟, 结果如图 1 所示. 在 K 为 10 时, DKNNS 的查询延迟约为 3 s, 而 Meridian 算法的平均完成时间约为 25 s. Meridian 算法由于需要所有的候选节点探测距目标节点的延迟, 并且假定延迟空间符合三角不等性并不完全成立, 导致 K 近邻搜索过程延长, 搜索延迟要远大于 quasi-Netvigator 算法和 DKNNS 算法.

(2) 搜索精确度对比. 其次, 对比不同方法的精确性, 结果如图 2 所示. DKNNS 算法具有最高的精确度, DKNNS 算法平均绝对差错值不高于 7 ms, 而基于 Meridian 思想的 K 近邻搜索平均绝对差错要高于 DKNNS, 并且在不同节点规模下的稳定性低于 DKNNS 算法, 基于相对坐标思想的 quasi-Netvigator 算法的绝对差错高于 30 ms, 并且在不同节点规模下的稳定性低于 DKNNS 算法.

(3) 搜索带宽开销对比. 接着我们对比不同方法的网络通信负载, 结果如图 3 所示. DKNNS 算法的平均负载约为 2 KB, 且 DKNNS 算法随着节点规模的变化网络通信负载增加幅度不明显, 显示了

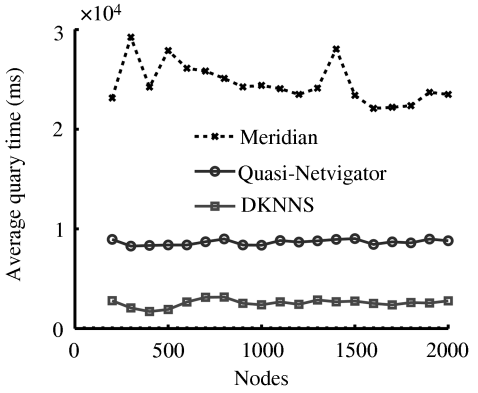


图 1 搜索时间

Figure 1 Query time

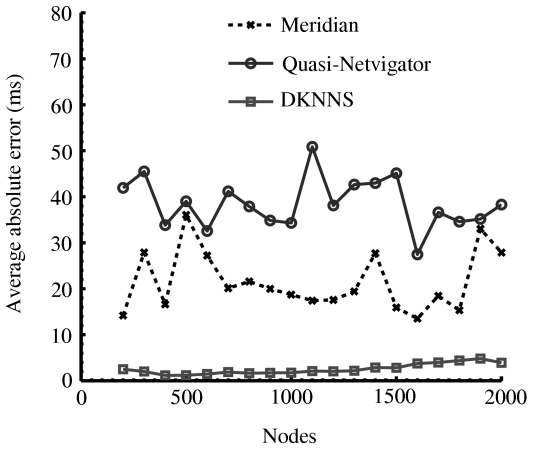


图 2 搜索精度

Figure 2 Query accuracy

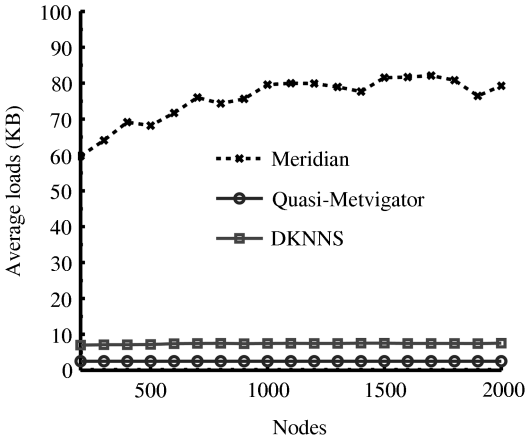


图 3 搜索带宽

Figure 3 Query overhead

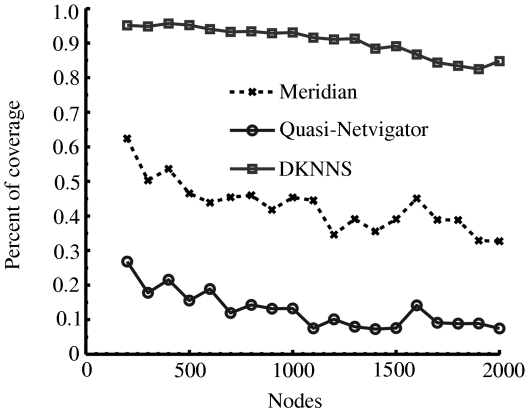


图 4 真实最近邻比例

Figure 4 Percent of coverage

DKNNS 算法能够更好的适应不同规模的节点集合环境. quasi-Netvigator 由于利用固定数目的地标节点集合, 因此网络平均负载保持恒定, 而 Meridian 算法由于需要选择延迟范围的所有节点探测距目标节点的距离, 因此网络负载随着节点集合的增大而升高.

(4) 发现真实 K 近邻的比例. 最后, 对比不同方法的真实 K 近邻覆盖比例, 结果如图 4 所示. DKNNS 算法的覆盖率较高, Meridian 算法的覆盖率低于 DKNNS 算法, quasi-Netvigator 算法的覆盖率最低, 保持在 0.1 左右. 同时, 我们通过模拟测试还发现 DKNNS 算法的覆盖率在 K 值增大时, 稳定性增大, 显示了 DKNNS 算法可以灵活的支持不同邻近节点数目的 K 近邻搜索.

7 PlanetLab 实验

为了检验真实环境下 DKNNS 算法与相关 K 近邻搜索方法的性能, 我们基于 Java 语言设计实现了 DKNNS 算法, 以及基于重复执行 Meridian 思想的 K 近邻搜索算法. 选择 PlanetLab 上分布在全

球的 173 个机器作为服务器节点. 并随机选择 PlanetLab 上 412 台服务器作为目标节点执行 K 近邻搜索.

实验过程对比算法包括: (1) DKNNS, DKNNS 的 K 近邻搜索参数 β 为 0.9, 最远节点搜索的 β_f 为 1.2, 每个节点多级环上每个环大小上限为 10. 坐标维度设定为 5. 设定低度量参数 ρ 为 2.5. 每隔 30 分钟执行 1 次 K 近邻搜索, 发现服务节点 (K 为 20). (2) 基于重复方式进行 K 近邻搜索的 Meridian 算法, K 近邻搜索的邻近性搜索阈值 β 为 0.5; 每个节点同心环上每个环大小上限为 10. (3) 基于直接探测的集中式延迟测量机制, 标记为 Direct: 为了判断 K 近邻搜索精确度, 每个服务器节点直接探测距离各个目标结点的延迟, 多次测量到达同一个节点的延迟时, 利用滑动平均的方式进行计算. 通过收集到各个目标节点距各个服务器的延迟后, 集中式的统计距目标节点延迟最近的 K 个近邻.

为了公平对比 DKNNS 以及 Meridian 算法的性能, 对于 DKNNS 以及 Meridian 算法的逻辑邻居集合进行统一配置. 设定每个节点每隔 30 s 执行一次逻辑邻居集合更新操作, 且每次集合更新时, 与 10 个邻居进行消息通信. 每个节点每隔 2 min 执行一次 K 近邻搜索操作. K 值在 1, 10, 25, 30 之间进行随机选择.

(1) K 近邻搜索时间开销对比. 其次, 对比不同方法的 K 近邻搜索时间开销, 结果如图 5 所示. 实验发现, PlanetLab 试验床中机器的高负载对 K 近邻搜索延迟具有显著影响, DKNNS 与 Meridian 的 K 近邻搜索中从程序发出消息命令到消息实际从物理网卡中发出的等待延迟在数百毫秒到数秒不等. 因此, DKNNS 与 Meridian 的查询延迟均比模拟测试有所增大. 在 K 值为 10 时 DKNNS 查询延迟要显著优于 Meridian 算法. 超过 85% 比例的 DKNNS 查询在 30 s 以内完成, 而此时仅有 60% 比例的 Meridian 查询在 30 s 内完成.

(2) K 近邻搜索精确度对比. 由于难以实时测量 DKNNS 与 Meridian 相对于真实近邻的偏差程度, 为了对比不同方法的 K 近邻搜索精度, 我们转而采取近邻搜索结果延迟分布函数的形式, 量化 K 近邻搜索过程返回近邻距目标节点的邻近性分布. 显然, 分布函数位于 Direct 方式结果右侧, 且差异越大, 则搜索到的 K 近邻有效性越低. 不同方法 K 近邻搜索结果的延迟分布如图 6 所示. DKNNS 的测量延迟分布接近直接探测对应的近邻延迟分布, 显示了 K 近邻搜索过程具有较高的精确度. 而 Meridian 算法的 K 近邻搜索结果要低于 Direct 算法的精确度.

(3) K 近邻搜索带宽开销对比. 对比 K 近邻搜索带宽开销, 结果如图 7 所示. DKNNS 搜索带宽开销较 Meridian 算法显著降低. DKNNS 超过 90% 比例的查询开销不到 6 KB, 而 Meridian 算法超过 90% 比例的 10 近邻搜索开销大于 40 KB.

(4) K 近邻搜索控制开销对比. 为了度量 K 近邻搜索算法的效率, 我们收集 DKNNS 与 Meridian 中每个参与节点每隔 2 min 的转发协议控制开销. Meridian 算法的控制开销包括 gossip 消息及环维护开销, DKNNS 算法的控制开销包括 gossip 消息, 及用于多级环维护的 K 近邻搜索. 结果如图 8 所示. DKNNS 平均约为 2 KB, 而 Meridian 的平均开销超过 20 KB, 并且受网络状况的影响较大. 由于 Meridian 算法和 DKNNS 算法的 gossip 间隔相同, 说明 Meridian 算法的环探测开销占据控制开销的主要比例. 而 DKNNS 算法通过分布式的 K 近邻搜索, 将控制开销分散到各个节点, 且执行频率较低, 因此协议控制开销较低.

(5) K 近邻搜索稳定性对比. 为避免频繁的 K 近邻搜索造成的等待时间延长的问题, 在实际网络应用中可能需要将 K 近邻结果缓存. 在节点集合频繁的加入、退出的环境下, 缓存结果可能很快失效. 因此, 我们考察在相对稳定的服务器集合环境下 K 近邻搜索的稳定性. 例如内容分发网络或者地

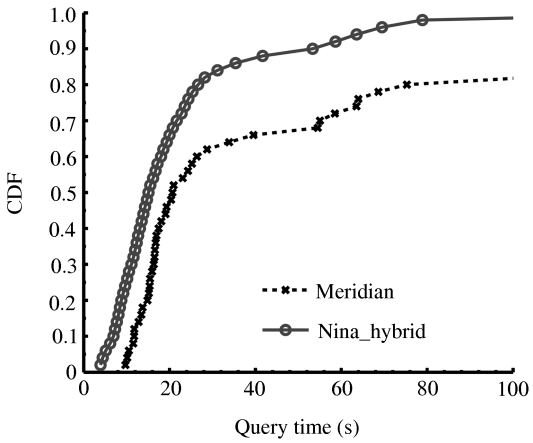


图 5　查询延迟

Figure 5　Query time

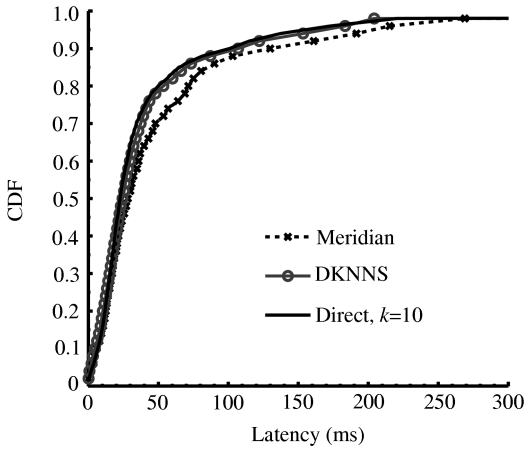


图 6　最近邻延迟

Figure 6　RTT distribution

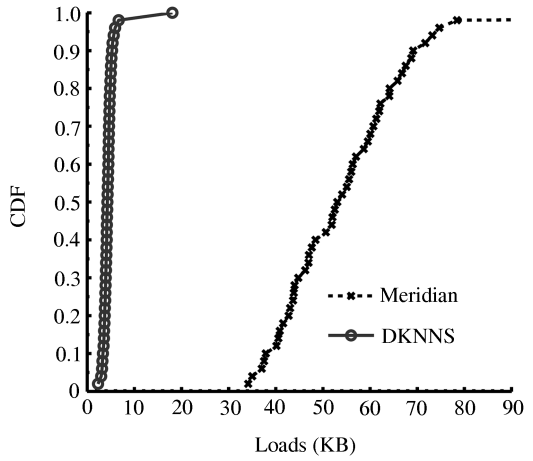


图 7　消息开销

Figure 7　Query loads

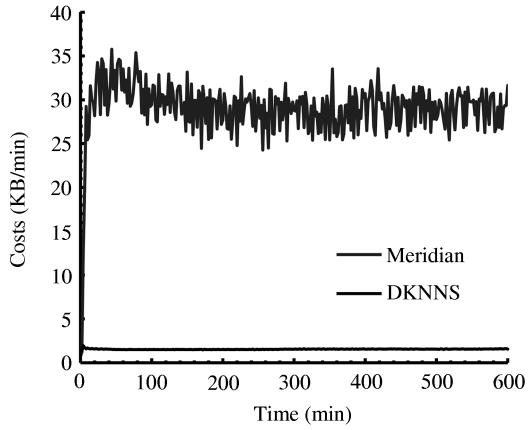


图 8　维护开销

Figure 8　Maintenance costs

理分布的数据中心网络满足这种稳定性. 为了检验 K 近邻搜索的稳定性, 在所有节点加入后, 我们对比 Meridian 与 DKNNS 算法的 1 近邻搜索和 10 近邻搜索结果稳定性, 稳定性通过对比同一个目标节点的时间邻近的 K 近邻搜索结果包含相同近邻节点的比例进行度量. 结果如图 9 所示. 结果发现 DKNNS 算法具有较高的搜索结果稳定性. 在 K 为 1 时, 约 70% 时间邻近的查询返回相同的节点. 在 K 为 10 时, 不到 10% 的查询返回的相同近邻数低于 0.6. 而 Meridian 算法在 K 为 1 时不到 50% 的查询返回相同的结果. 而 K 为 10 时, 超过 20% 的查询返回相同近邻比例低于 0.6.

8　结论

针对延迟敏感型应用中, 可扩展的寻找任意目标节点的 K 个服务节点的问题, 本文提出了分布式 K 近邻搜索算法 DKNNS. 每个节点维护一个邻近性感知的多级环, 利用改进的 Vivaldi 算法快速的预测多级环上邻居距目标的延迟, 在执行 K 近邻搜索时, 利用贪心搜索策略以指数级速度逼近目标节点

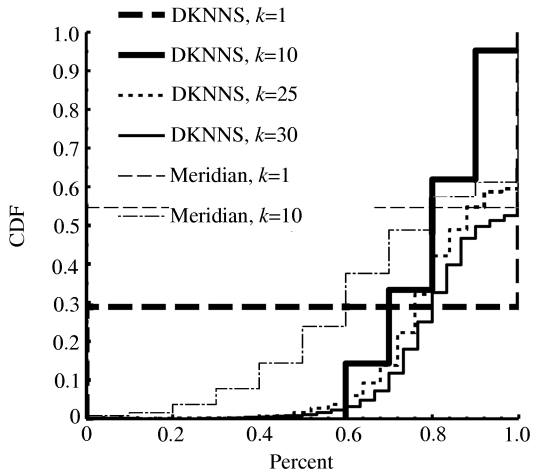


图 9 搜索稳定性

Figure 9 Search stability

邻近区域, 然后基于回退的方式快速的确定 K 个近邻. 基于理论分析, 模拟测试, 以及 PlanetLab 平台实际部署发现 DKNNS 算法搜索精确度高, 查询开销低, 且查询延迟短. 在下一步的工作中, 我们将针对 PlanetLab 虚拟机环境上快速精确网络测量问题, 进一步优化 DKNNS 查询延迟, 实现多目标约束的延迟敏感型网络应用服务器选择.

参考文献

1 Agarwal S, Lorch J. Matchmaking for online games and other latency-sensitive P2P systems. In: SIGCOMM'09. Barcelona: ACM, 2009. 315 326

2 Greenberg A, Hamilton J, Maltz D, et al. The cost of a cloud: research problems in data center networks. Comput Commun Rev, 2009, 39: 68 73

3 Beigbeder T, Coughlan R, Lusher C, et al. The e ects of loss and latency on user performance in unreal tournament 2003. In: NetGames'04. Portland: ACM, 2004

4 Szigeti T, Hattingh C. Quality of service design overview. Http://www.ciscopress.com/articles/article.asp?p=357102&seq Num=2. 2004

5 Wang Y J, Li X Y. Network distance prediction technology research. J Softw, 2009, 20: 1574 1590 [王意洁, 李小勇. 网络距离预测技术研究. 软件学报, 2009, 20: 1574 1590]

6 Xing C Y, Chen M. Techniques of network distance prediction. J Softw, 2009, 20: 2470 2482 [邢长友, 陈鸣. 网络距离预测技术. 软件学报, 2009, 20: 2470 2482]

7 Zhang R, Tang C, Hu Y, et al. Impact of the inaccuracy of distance prediction algorithms on Internet applications an analytical and comparative study. In: INFOCOM'06. Barcelona: IEEE, 2006. 1 12

8 Cox R, Dabek F, Kaashoek F, et al. Practical, distributed network coordinates. ACM SIGCOMM Comput Commun Rev, 2004, 34: 113 118

9 Cho nes D, Sanchez M, Bustamante F. Network positioning from the edge an empirical study of the e ectiveness of network positioning in P2P systems. In: INFOCOM'10. San Diego: IEEE, 2010

10 Wang G, Zhang B, Ng E. Towards network triangle inequality violation aware distributed systems. In: Internet Measurement Comference. San Diego: ACM, 2007. 175 188

11 Chen Y, Wang X, Song X, et al. Phoenix: Towards an accurate, practical and decentralized network coordinate system. In: Proceedings of IFIP/TC6 Networking 2009 (Networking'09). Aachen, 2009

12 Xing C Y, Chen M. A hierarchical network distance prediction mechanism. Chin J Comput, 2010, 33: 356 364 [邢长友, 陈鸣. 一种层次化网络距离预测机制. 计算机学报, 2010, 33: 356 364]
13 Madhyastha H, Isdal T, Piatek M, et al. iPlane: an information plane for distributed services. In: OSDI. Seattle: USENIX Association, 2006. 367 380
14 Madhyastha H, Katz-Bassett E, Anderson T, et al. iPlane Nano: path prediction for peer-to-peer applications. In: Proceedings of the Usenix Conference on Networked Systems Design and Implementation (NSDI). Boston: USENIX Association, 2009. 137 152
15 Xu Z, Sharma P, Lee S. Netvigator: scalable network proximity estimation. HP Laboratories Technical Report, HPL-2004-28. 2004
16 Sharma P, Xu Z, Banerjee S, et al. Estimating network proximity and latency. ACM SIGCOMM Comput Commun Rev, 2006, 36: 39 50
17 Ratnasamy S, Handley M, Karp R, et al. Topologically-aware overlay construction and server selection. In: INFOCOM. New York: IEEE, 2002
18 Waldvogel M, Rinaldi R. E cient topology-aware overlay network. SIGCOMM Comput Commun Rev, 2003, 33: 101 106
19 Costa M, Castro M, Rowstron A, et al. PIC: practical Internet coordinates for distance estimation. In: Proceedings of the International Conference on Distributed Computing Systems. Hachioji: IEEE, 2004. 178 187
20 Wong B, Slivkins A, Sirer E. Meridian: a lightweight network location service without virtual coordinates. In: SIGCOMM. Philadelphia: ACM, 2005. 85 96
21 Freedman M, Lakshminarayanan K, Mazi'eres D. OASIS: anycast for any service. In: NSDI. San Jose: USENIX Association, 2006
22 Wendell P, Jiang W, Freedman M, et al. DONAR: decentralized server selection for cloud services. In: SIGCOMM. New Delhi: ACM, 2010. 231 242
23 Vishnumurthy V, Francis P. On the di culty of finding the nearest peer in P2P systems. In: Internet Measurement Conference. Vouliagmeni: ACM, 2008. 9 14
24 Chávez E, Navarro G, Baeza-Yates R, et al. Searching in metric spaces. ACM Comput Surv, 2001, 33: 273 321
25 Hjaltason G, Samet H. Index-driven similarity search in metric spaces (survey article). ACM Trans Database Syst, 2003, 28: 517 580
26 Indyk P. Nearest neighbors in high-dimensional spaces. In: Handbook of Discrete and Computational Geometry. 2nd ed. Atlanta: CRC Press LLC, 2004
27 Clarkson K. Nearest-neighbor searching and metric space dimensions. In: Shakhnarovich G, Darrell T, Indyk P, eds. Nearest-Neighbor Methods for Learning and Vision: Theory and Practice. Cambridge: MIT Press, 2006. 15 59
28 Fraigniaud P, Lebhar E, Viennot L. The inframetric model for the internet. In: INFOCOM. Phoenix: IEEE, 2008. 1085 1093
29 Zhang B, Ng E, Nandi A, et al. Measurement-based analysis, modelling, and synthesis of the internet delay space. IEEE/ACM Trans Netw, 2010, 18: 229 242

附录

性质 1 证明 首先, 根据增量维度定义, 可得

$$|B_P(xr)| \leqslant \gamma_g \left| B_P\left(\frac{x}{\rho} r\right) \right|.$$

然后递归调用 $\lceil \log_\rho x \rceil$ 次增量维度定义, 直至 $\frac{x}{\rho^{\lceil \log_\rho x \rceil}} < 1$, 可得

$$|B_P(xr)| \leqslant \gamma_g^{\lceil \log_\rho x \rceil} |B_P(r)| = x^{\log_x \gamma_g \lceil \log_\rho x \rceil} |B_P(r)| = x^\alpha |B_P(r)|, \alpha = \log_x \gamma_g \times \lceil \log_\rho x \rceil.$$

因此, 根据上限函数 $\lceil \rceil$ 定义, 我们可以计算 α 的下界:

$$\alpha \geqslant \log_x \gamma_g \times \log_\rho x = \log_\rho \gamma_g.$$

另一方面, 由于 $x \geqslant \rho$, $\gamma_g > 1$, 因此可得

$$\log_\rho \gamma_g = \frac{\log \gamma_g}{\log \rho} \geqslant \frac{\log \gamma_g}{\log x} = \log_x \gamma_g.$$

接着我们可以求解 α 的上界:

$$\alpha \leqslant \log_x \gamma_g \times \left(\log_\rho x + 1\right) = \log_\rho \gamma_g + \log_x \gamma_g \leqslant \log_\rho \gamma_g + \log_\rho \gamma_g = 2\log_\rho \gamma_g.$$

结论得证.

性质 2 证明 (1) 假设 ρ- 低度量网格度量为未知数 α, 对于任意的 u, v, v 满足 $v \in B_u(\rho r)$, 其中 $r > 0$. 对于任意满足 $x \in B_u(\rho^2 r)$ 的节点 x, 根据低度量定义,x 距 v 的距离 d_{vx} 满足

$$d_{vx} \leqslant \rho \max\left\{d_{Pv}, \rho^2 r\right\} = \rho^3 r.$$

因此 $x \in B_v(\rho^3 r)$, 故

$$B_P(\rho^2 r) \subseteq B_v(\rho^3 r).$$

故根据网格维度的定义, 可知

$$|B_P(\rho^2 r)| \leqslant |B_v(\rho^3 r)| \leqslant (\rho^3)^\alpha |B_v(r)| \leqslant (\rho^4)^\alpha |B_v(r/\rho)|.$$

(2) 我们采取贪心的方式 [28] 利用半径为 r 的球构建球 $B_u(\rho r)$ 的覆盖 F 如下: 设 $m \leftarrow 0$, 若球 $B_u(\rho r)$ 存在没有被包含的节点子集, 即

$$B_P(\rho r) \backslash \underset{1 \leqslant i \leqslant m}{\cup} B_{v_i}(r) \neq \text{NULL},$$

则任意选择一个新的球心节点 v_{m+1}, 满足

$$v_{m+1} \in B_P(\rho r) \backslash \underset{1 \leqslant i \leqslant m}{\cup} B_{v_i}(r).$$

然后更新 $m \leftarrow m + 1$, 递归执行上述过程.

对于上述基于贪心方式构建的一个覆盖 F:$B_u(\rho r) = \cup_{1 \leqslant i \leqslant \gamma_d} B_{v_i}(r)$, 标记 C 为 F 中所有球 $B_{v_i}(r)$ 的球心组成的集合. 根据球心的选择方式可知, 球心集合 C 中任意的节点 v_i 位于球 $B_u(\rho r)$ 内部, 即 $d_{v_i u} \leqslant \rho r$, 其次, 任意两个球心 v_i, v_j 之间的距离 d_{v_i, v_j} 大于 r.

其次, 我们基于反证法证明以上述集合 C 为球心, 以 r/ρ 为半径的球两两不相交. 任意选择两个球心 $v_i \in C$, $v_j \in C$, 假设存在节点 w 满足

$$w \in B_{v_i}(r/\rho) \cap B_{v_j}(r/\rho).$$

则根据低度量定义可知

$$d_{v_i, v_j} \leqslant \rho \max\{d(v_i, w), d(v_j, w)\} \leqslant r,$$

而这与已知 $d_{v_i, v_j} > r$ 矛盾, 故以 C 中节点为球心, 半径为 r/ρ 的任意两个球 $B_{v_i}(r/\rho)$ 与 $B_{v_j}(r/\rho)$, 互不相交的.

(3) 综合 (1) 与 (2), 对于以球心集合 C 中节点为球心, r/ρ 为半径的球的并集, 可得

$$|\cup_{1 \leqslant i \leqslant \gamma} B_{v_i}(r/\rho)| \geqslant \gamma |B_P(\rho^2 r)|/(\rho^4)^\alpha.$$

由于 (2) 贪心覆盖构造过程可知每个球心 v_i 位于 $B_u(\rho r)$ 内部, 即 $v_i \in B_u(\rho r)$, 根据低度量定义, 可知

$$B_{v_i}(r/\rho) \subseteq B_P(\rho^2 r).$$

因此

$$(\rho^4)^\alpha \geqslant \gamma,$$

即

$$\alpha \geqslant \frac{\log \gamma}{4 \log \rho} = \frac{1}{4} \log_\rho \gamma.$$

故 ρ- 低度量为 γ_d- 倍增时, 网格度量 $\alpha \geqslant \frac{1}{4} \log_\rho \gamma_d$.

(4) 另一方面, 由于 $|x^\alpha| \leqslant N$, 因此在 $x \geqslant \rho$ 时, $\alpha \leqslant \log_\rho(N)$, N 为系统中节点总数. 综上, 定理得证.

定理 1 证明 任意选择两个不同的节点 u, v,$d_{uv} > 0$, 设 $r = \varepsilon d_{uv}$, 选择最小的整数 i 满足 $d_{uv} + r \leqslant 2^i$, 因此 $r + d_{uv} > 2^{i-1}$, 否则与 i 是最小的整数矛盾.

根据低度量模型定义, 对于球 B_{ui} 中的任意节点 x, 节点 x 距节点 v 的距离 d_{xv} 满足

$$d_{vx} \leqslant \rho \max\left\{d_{uv}, d_{ux}\right\} \leqslant \rho \times 2^i \leqslant \rho \times 2\left(r + d_{uv}\right) = \rho \times 2\left(1 + \frac{1}{\varepsilon}\right) r = \rho\gamma r,$$

其中 $\gamma = 2(1 + 1/\varepsilon)$. 因此, 球 B_{ui} 满足如下条件:

$$B_{ui} \subseteq B_v(\rho\gamma r).$$

接着利用性质 1 结论, 可得

$$|B_{ui}| \leqslant |B_v(\rho\gamma r)| \leqslant (\rho\gamma)^{\alpha}|B_v t(r)|,$$

其中 α 为倍增维度的网格维度. 基于引理 1, 位于 S_{ul}, $l \leqslant i$ 环上的某个节点, 位于 $B_v(r)$ 的失效概率小于 δ/N^2. 定理得证.

定理 2 证明 与增量维度下多级环质量证明类似, 首先选择两个不同的节点 $u, v, d_{uv} > 0$, 设 $r = \varepsilon d_{uv}$, 选择最小的整数 i 满足 $d_{uv} + r \leqslant 2^i$. 因此与定理 1 相似, 根据低度量模型定义, 对于球 B_{ui} 中的任意节点 x, 节点 x 距节点 v 的距离 d_{xv} 满足

$$d_{vx} \leqslant \rho\gamma r,$$

其中 $\gamma = 2(1 + 1/\varepsilon)$. 因此, 球 B_{ui} 满足如下条件:

$$B_{ui} \subseteq B_v(\rho\gamma r).$$

接着, 直接利用性质 2 结论, 可得

$$|B_{ui}| \leqslant |B_v(\rho\gamma r)| \leqslant (\rho\gamma)^{\alpha}|B_v t(r)|,$$

其中 α 为倍增维度的网格维度. 基于引理 1, 可知定理成立.

定理 3 证明 由于 DKNNS 算法步骤 (4) 最近搜索判断要求下一跳步节点距目标的距离要降低至不大于当前节点距目标距离的 β 倍, 根据低度量模型定义, 下一跳步的候选节点集合 H 需要满足下述不等式组

$$\begin{cases} \beta d < \rho \max\{x, d\}, \\ x < \rho \max\{\beta d, d\}, \\ d < \rho \max\{\beta d, x\}. \end{cases}$$

根据 DKNNS 搜索策略的贪心特征可知 $\beta < 1$, 而对于任意的三元组, 根据低度量定义可知 $\rho > 1$.

为了对上述不等式组化简, 我们分情况进行讨论: (1) 若 $x > d$, 将上述不等式组化简可得,$x < \rho d \wedge x > d$ (2) 若 $x \leqslant d$, 上述不等式组可化为 $d < \rho \max\{\beta d, x\}$. 综合 (1),(2) 可得

- 在 $\rho\beta \geqslant 1$ 时, 则 $x \leqslant \beta d \vee \beta d < x \leqslant d$. 即 $x \leqslant d$.
- 在 $0 < \rho\beta < 1$ 时, 此时 $d/\rho < x \leqslant d$.

综上, 可知若 $\rho\beta \geqslant 1$, 则我们需要选择满足 $x < \rho d$ 的节点集合, 而若 $\rho\beta \leqslant 1$, 则我们需要选择 $d/\rho < x < \rho d$ 范围的节点集合. 定理得证.

定理 4 证明 根据 DKNNS 步骤 (4) 最近搜索判断步骤, 下一跳步转发节点 V 距离目标 T 的距离至少降低为当前节点到目标节点距离的 β 倍 (β 小于 1), 因此 DKNNS 算法以指数级速度降低距目标 T 的距离, 故 1 近邻搜索过程在 $O(\log\Delta)$ 跳步内结束. 另一方面, 假设节点 U 接收到 DKNNS 近邻搜索消息, 并寻找到下一跳步节点为 V, 因此 $d_{VT} \leqslant \beta d_{UT}$, 且真实 1 近邻节点 V^* 距 T 的距离满足

$$\frac{d_{UT}}{d_{V^*T}} > \frac{1}{\beta},$$

因此若当前节点 U 无法搜索到距目标距离不高于 βd_{UT} 的下一跳步近邻, 搜索过程终止, 此时当前环上距目标节点最近的节点 V' 距目标的距离满足 $\frac{d_{V'T}}{d_{V^*T}} < \frac{1}{\beta}$. 否则, 因为环满足 ϵ-nice, 且 $\epsilon < \beta$, 我们能够以接近 1 的概率寻找到距目标节点距离不大于 βd_{UT} 的下一跳步转发节点, 这与搜索终止矛盾. 定理得证.

定理 5 证明 在 DKNNS 算法寻找每个近邻节点过程中, 由假设条件多级环保持 ϵ-nice, 类似定理 4 证明过程, 每个近邻节点 V_i 距目标节点 T 的距离相对于真实最近邻节点 V^* 距目标节点 T 的距离满足

$$\frac{d_{V_iT}}{d_{V^*T}} < \frac{1}{\beta},$$

否则, 根据多级环保持 ϵ-nice, 我们可以从 V_i 的多级环上选择距目标节点距离不高于 βd_{V_iT} 的下一跳步节点, 满足 DKNNS 步骤 (4) 最近搜索判断规则, 这与 V_i 节点是一个近邻发生矛盾.

因此, 综合 DKNNS 算法发现的 K 个近邻节点集合 S_1 可得,

$$\sum_{v_i \in S_1} d_{V_iT} < \frac{1}{\beta} \times K \times d_{V^*T} \leqslant \frac{1}{\beta} \sum_{i \in S_2} d_{iT},$$

故由第 3 节定义 2 可知, 基于 DKNNS 的 K 近邻搜索确定的 K 个近邻节点集合为 $1/\beta$ 近似.

另一方面, 由于每个近邻节点的搜索过程均是指数级降低距目标节点的距离, 因此, DKNNS 算法的每个近邻节点搜索过程均可以在 $O(\log \Delta)$ 跳步完成, 故 K 个近邻搜索总跳步数为 $O(K \log \Delta)$. 定理得证.

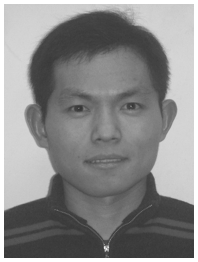
DKNNS: Scalable and accurate distributed K nearest neighbor search for latency-sensitive applications

FU YongQuan* & WANG YiJie

National Key Laboratory for Parallel and Distributed Processing, School of Computer Science, National University of Defense Technology, Changsha 410073*, China*
*E-mail: quanyongf@gmail.com

Abstract To reduce the access latencies of end hosts, latency-sensitive applications need to choose suitably close service machines to answer the access requests from end hosts. Distributed K nearest neighbor search locates K service machines closest to end hosts, which can e ciently optimize the access latencies for end hosts. Existing work has weakness in terms of the accuracy and scalability. According to the scalable and accurate K nearest neighbor search problem, we propose a distributed K nearest neighbor search method called DKNNS in this paper. Service machines are organized into a locality-aware multilevel ring. DKNNS first locates a service machine that starts the search process based on a farthest neighbor search scheme, then discovers K nearest service machines based on a backtracking approach within the proximity region containing the target in the latency space. Theoretical analysis, simulation results and deployment experiments on the PlanetLab show that, DKNNS can determine K approximately optimal service machines, with modest completion time and query loads. Finally, DKNNS is also quite stable that can be used for reducing frequent searches by caching found nearest neighbors.

Keywords latency sensitive network applications, K nearest neighbor search, network coordinate, network measurement, distributed system



Fu YongQuan received the B.S. degree in computer science and technology from School of Computer of Shandong University in 2005, and received the M.S. in computer science and technology in School of Computer Science of National University of Defense Technology in 2008. He is currently a Ph.D. candidate in School of Computer Science of National University of Defense Technology. He is a student member of CCF and ACM. His current research interests lie in the areas of network measurement, peer-to-peer network and distributed system.

Wang YiJie received the Ph.D. degree from National University of Defense Technology in 1998. She was a recipient of the National Excellent Doctoral Dissertation (2001), a recipient of Fok Ying Tong Education Foundation Award for Young Teachers (2006) and a recipient of the Natural Science Foundation for Distinguished Young Scholars of Hunan Province (2010). Now she is a Professor in the National Key Laboratory for Parallel and Distributed Processing, National University of Defense Technology. Her research interests include network computing, massive data processing, parallel and distributed processing.