

文章编号:1001-1595(2009)04-0348-08

网络地理信息服务中的空间数据缓存算法研究

王 浩^{1,2}, 喻占武³, 曾 武³, 潘少明³

1. 诺基亚(中国)研究中心, 北京 100176; 2. 武汉大学 电子信息工程学院, 湖北 武汉 430079; 3. 武汉大学 测绘遥感信息工程国家重点实验室, 湖北 武汉 430079

The Research on the Algorithm of Spatial Data Cache in Network Geographic Information Service

WANG Hao^{1,2}, YU Zhanwu³, ZENG Wu³, PAN Shaoming³

1. Nokia Research Center, Beijing, 100176, China; 2. School of Electronic Information, Wuhan University, Wuhan 430079, China; 3. State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China

Abstract: Network Geographic Information Service (NGIS) has been widely used in various types of projects, such as government decision making system, business navigation and disaster forecast. While NGIS is gaining in popularity, fast dissemination of voluminous and heterogeneous spatial data becomes a challenge due to limited Internet bandwidth and massive concurrent users. The spatial data should be delivered to the NGIS users within a reasonable time span. In this paper, a cache mechanism for terrain data and image data is proposed in NGIS to improve the interactive service performance. The tile cache index is designed based on the tile cache granularity, which is proved to be efficient. Tile Access average Interval time Longest (TAIL) cache replacement algorithm is proposed and its temporal locality and spatial locality are analyzed. The fast operation method of TAIL is presented. The image data set of LANDSAT7 with 30 meter resolution and the terrain data set of SRTM with 90 meter resolution are used in trace-driven simulations of TAIL. The result shows that TAIL has improved cache hit rate and byte hit rate, and significantly reduces tile-request-response-time and tile-request-number processed by NGIS server compared with currently available typical cache replacement algorithms.

Key words: tile cache replacement; cache index; spatial data; NGIS

摘 要: 网络地理信息服务中, 由于带宽有限以及大规模的并发用户访问, 海量异构的空间数据的快速实时网络传输, 是提高系统性能、缩短服务响应时间的关键因素。提出适合于地形数据和影像数据的缓存机制, 以提高网络地理信息服务的交互性能。以瓦片为缓存粒度, 论述缓存瓦片索引的设计与效率并对缓存索引的性能进行验证。本文提出瓦片访问平均时间间隔最长的缓存置换算法 TAIL (Tile Access average Interval time Longest), 从瓦片访问的时间局部性和空间局部性分析 TAIL 算法并说明提高置换效率的方法。对最高分辨率为 30 m 的全球 LANDSAT7 影像数据集和最高分辨率为 90 m 的全球 SRTM 地形数据集进行日志驱动仿真实验。结果表明, TAIL 与代表性的缓存置换算法相比, 提高了瓦片请求命中率和字节命中率, 同时大幅降低瓦片请求响应时间和服务器端所处理的瓦片请求数。

关键词: 瓦片缓存置换算法; 缓存索引; 空间数据; 网络地理信息服务

中图分类号: P208

文献标识码: A

基金项目: 国家 973 计划(2004CB318206); 国家 863 计划(2007AA12Z214); 测绘遥感信息工程国家重点实验室专项科研经费

高速宽带网络的普及和空间数据获取技术的发展使地理信息服务走向网络化。网络地理信息服务 NGIS (Network Geographic Information Service) 中呈海量特征的空间数据的网络传输成为制约网络地理信息服务质量的关键因素之一, 研究海量空间数据的快速实时传输和可视化对实现网络地形漫游具有重要意义。已有的研究包

括: 采用小波变换实现海量空间数据的多分辨率表示并将 3 维场景做流式、渐进传输^[1-3]; 基于对等网络的地形数据传输与共享机制^[4]; 空间数据缓存^[5-7], 即将空间数据或其中间结果存储在客户端、代理服务器或服务器上, 以便下次有重复请求时直接响应等。缓存技术区别于其他方法主要是通过缓存来服务用户对空间数据的需求, 以快速响

应请求,减轻网络空间数据流量和服务器端负载。

本文针对 NGIS 原型系统 Globesight 的客户端缓存^[8],设计顺序存储缓存索引法并分析其性能和效率,提出瓦片访问时间间隔最长的缓存置换策略 TAIL (Tile Access average Interval time Longest)。TAIL 同时考虑瓦片访问的时间和空间局部性,考虑瓦片访问的短期流行度和长期流行度,能保证瓦片访问的全局优化又可适应瓦片访问模式的变化。

1 相关工作

多分辨率特性的瓦片金字塔模型可以对海量空间数据进行高效的组织和管理,在网络地理信息服务中广泛使用。瓦片金字塔可为空间数据缓存提高良好支持,可以空间数据瓦片作为缓存粒度。针对空间数据缓存,国内研究主要偏向于缓存的部署位置与架构。文献[7]在 WebGIS 中提出空间数据分布式缓存体系,包括客户端缓存、空间数据反向代理缓存与服务端缓存,3 种缓存一起构成分布式缓存架构。反向代理缓存位于 Web 服务器上,为不同类型、应用的空间数据提供单独的缓存,使具有同类型需求的多用户可以共享缓存。文献[9]在构件化 WebGIS 中引入空间缓存框架,包括空间数据库缓存,网络空间缓存和空间数据代理服务器等 3 个层次,其中,空间数据库缓存提高空间数据库读取效率,网络空间缓存提高局域网用户空间数据读取效率,空间数据代理服务器提高广域网用户空间数据读取效率的。

国外研究中,文献[5]研究了基于瓦片访问模式的瓦片缓存预取和置换算法,基于瓦片间访问的转移概率计算出邻居瓦片的预取概率,预取概率最高的瓦片被预取,转移概率最小的瓦片被置换出缓存。算法针对整个数据集中的每个瓦片做转移概率和预取概率的计算,由于瓦片数量巨大导致算法巨大的计算量无法适用于网络地理信息服务。文献[6]提出根据视角中心位置做缓存置换和预取,文章未给出实验结果;算法未考虑视角中心移动方向,导致在缓存置换时可能删除有用的瓦片同时预取过多无用的瓦片。

由于特定用户具有特定感兴趣的地理信息服务,对特定地理区域范围内的地形数据和影像数据的访问具有空间局部性和位置连续性。用户对与自己生活所在城市相关的地理信息服务兴趣较大,也对风景名胜区或重大赛事所在地(比如,

2008 年奥运会举办地北京)兴趣较大。因此,特定用户地形漫游时会产生大量重复的空间数据请求。这种网络地理信息服务中的用户漫游习惯,为在客户端部署空间数据缓存提供了理论支持。

在距离用户最近的客户端做空间数据缓存,对提高用户空间数据请求响应时间、节省网络带宽、减轻服务器负载具有重要作用。客户端缓存仅仅为单一特定用户服务,属于分布式缓存。客户端缓存会引入一定的负担,然而,从后文第 2.1 节针对缓存索引的设计和性能分析中可见,这种分布式缓存所引入的负担是轻量级、完全可以接受的;从第 3 节对缓存性能的实验结果可见,部署分布式客户端缓存所提升的空间数据服务质量是非常客观的。

网络地理信息服务具有异构性,不同用户所感兴趣的地理信息服务或服务组合存在交集,但交集相对较小。在距离用户很远的 Web 服务器上部署反向代理缓存或在空间应用服务器上部署服务器端缓存^[7],虽可为大量用户所请求的空间数据交集作一定程度的共享,降低了应用服务器读取空间数据库的负载,但是缓存的空间数据仍然需要经过动态网络传输到客户端,无法快速响应用户的瓦片请求和降低网络的空间数据流量。另一方面,反向代理缓存和服务器端缓存,属于集中式缓存,容易成为单点失败点,无法提供可靠的大规模空间数据服务;海量集中式缓存的空间数据索引的建立、查询、更新和删除等操作,代价也很高。

2 空间数据缓存:瓦片的索引与置换算法

以瓦片为缓存粒度,重用空间数据,可以提高网络地理信息服务的服务质量^[10]。本节首先讨论瓦片缓存索引机制,然后提出一种瓦片访问时间间隔最长的缓存置换策略 TAIL。

2.1 基于瓦片金字塔的缓存索引

2.1.1 顺序存储缓存索引表

瓦片金字塔是空间数据按分辨率分级存储与管理的一种有效组织结构,其思想是通过分层与分块操作,将原始数据按不同分辨率重采样产生层,同一层数据按指定个数像素均分为块生成瓦片矩阵。对特定层的瓦片矩阵中均等分的瓦片块,按照从瓦片矩阵左下角开始,从左至右、从下至上的顺序对瓦片进行编号,得到瓦片矩阵的 2 维瓦片坐标系。将特定层中的 2 维瓦片坐标系与层号相结合,则构成可以唯一标识瓦片的 3 维空

间数据瓦片坐标系。图 1 是金字塔模型示例及其对应的 3 维瓦片坐标系,其中瓦片坐标以 $(l, (tx, ty))$ 的形式表示,表示位于第 l 层的瓦片矩阵中第 tx 行、第 ty 列的瓦片块。客户端根据用户当前视角中心位置和经纬度等信息计算出位于视角中心的瓦片坐标 $(l, (tx, ty))$,并以 $URL = http://NGIS_Server_Address/tile.aspx?L=l\&X=tx\&Y=ty\&...$ 的形式向服务器请求瓦片。

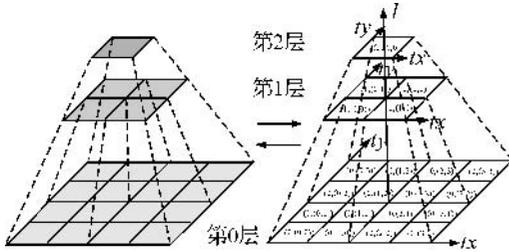


图 1 金字塔模型及其 3 维瓦片坐标系例
Fig. 1 Pyramid model of spatial tile data

根据瓦片金字塔模型,以瓦片为粒度,构造瓦片缓存索引表。为了提高缓存的可扩展性,一般将缓存索引与缓存瓦片数据分开存储,被缓存的瓦片数据有专门的存储区。图 2 是一种顺序存储的缓存索引表,可以用一个 2 维数组 $TileIndex[N][5]$ 来实现,其中 N 为缓存最多能容纳的瓦片个数。第 $i(0 \leq i \leq N-1)$ 个索引对应于缓存中的第 i 个瓦片, $TileIndex[i][0]$ 是该瓦片的瓦片坐标 $(l, (tx, ty))$ 的 MD5 值 $MD5((l, (tx, ty)))$,以此值为键值顺序存储各瓦片的索引,目的是为了快速地搜索缓存中是否存在所请求的瓦片, $TileIndex[i][1]$ 是该瓦片最后一次被访问的时间, $TileIndex[i][2]$ 是该瓦片的总访问次数,即访问频率, $TileIndex[i][3]$ 是该瓦片在缓存中首次存储的时间, $TileIndex[i][4]$ 是该瓦片在缓存存储区的位置偏移量,通过该值可以立刻定位瓦片在缓存存储区中的位置并立刻响应用户请求。

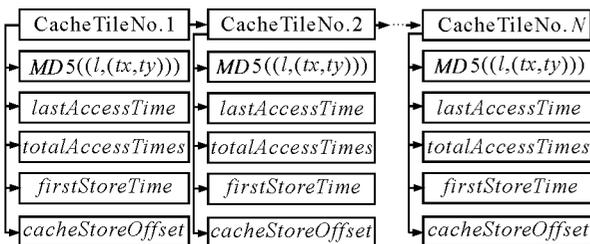


图 2 瓦片缓存顺序索引表

Fig. 2 Tile cache sequential index

2.1.2 缓存索引的操作与维护性能

当用户请求某瓦片 $(l, (tx, ty))$ 时,系统首先生成 $(l, (tx, ty))$ 的 MD5 值,然后在顺序缓存索引表中按二分查找确定该瓦片是否在缓存存储区中,平均查找长度为 $(1+1/n) \log_2(n+1) - 1$, n 为缓存存储区当前瓦片个数, $0 \leq n \leq N$ 。分两种情况讨论:

1. 若缓存命中,则直接读取 $TileIndex[i][4] = cacheStoreOffset$ 确定瓦片在缓存存储区的位置并向用户返回该瓦片,更新索引: $TileIndex[i][1] = lastAccessTime = currentTime$, $TileIndex[i][2] = totalAccessTimes = totalAccessTimes + 1$ 。

2. 若缓存未命中,则系统向 NGIS 服务器发起类似于 $URL = http://NGIS_Server_Address/tile.aspx?L=l\&X=tx\&Y=ty\&...$ 的瓦片请求,并将返回的瓦片在向用户返回后存储在缓存数据区,获取该瓦片在缓存数据区的存储位置偏移量 $cacheStoreOffset$;生成该瓦片的 MD5 值 $MD5((l, (tx, ty)))$,根据 $MD5((l, (tx, ty)))$ 查找其索引在顺序索引数组中应处于的位置,插入、建立该条索引,使得 $TileIndex[i][0] = MD5((l, (tx, ty)))$, $TileIndex[i][1] = TileIndex[i][3] = currentTime$, $TileIndex[i][2] = 1$, $TileIndex[i][4] = cacheStoreOffset$;同时移动位于 $MD5((l, (tx, ty)))$ 后面的索引,移动长度为 $(1+1/n) \log_2(n+1) - 1$, $0 \leq n \leq N$ 。

假设缓存存储区总大小设置 $cache_size$ Mb,瓦片大小为 $tile_size \times tile_size$ 像素,每个像素以 3 字节表示,则 $N = 1024 \times 1024 \times cache_size \div (tile_size \times tile_size \times 3)$ 。若每个索引属性以 16 个字节来存储表示,则索引表大小的上限 max_index_size 为 $5 \times N \times 16$ 字节,即 $max_index_size = 5 \times N \times 16 \div (1024 \times 1024) = 80 \times cache_size \div (3 \times tile_size \times tile_size)$ Mb。举例来说,设缓存大小为 4 096 Mb,如果缓存存储的是最高分辨率为 30 m 的全球 LANDSAT7 影像数据,瓦片大小为 512×512 像素。使用顺序存储索引,则当缓存全部用完时,共计存储 $N = 5460$ 个瓦片,索引表大小约为 0.43 Mb,缓存命中时平均查找长度和缓存未命中时新建索引后已有索引平均移动长度均为 $(1+1/N) \log_2(N+1) - 1$ 即 $(1+1/5460) \log_2 5461 - 1 \approx 11.42$ 。如果 4 096 Mb 缓存用来存储最高分辨率为 90 m 的全球 SRTM 地

形数据,瓦片大小为 128×128 像素,使用顺序存储索引数组,则当缓存全部用完时,共计存储 $N=87\,920$ 个瓦片,索引表大小约为 6.71 Mb ,缓存命中时平均查找长度和缓存未命中时新建索引后已有索引平均移动长度均为 $(1+1/87\,920)\log_2 87\,921-1 \approx 15.42$ 。即使对于当前普通 PC,划分 $4\,096\text{ Mb}$ 的硬盘空间作为瓦片缓存是可行的,而此时瓦片个数、索引表大小和查找/移动效率都是令人满意的。 $4\,096\text{ Mb}$ 缓存空间已足够大,再使用高效的缓存置换策略,缓存的性能和效率可进一步提升。

2.2 瓦片缓存置换算法

当缓存存储区满时,需要选择并删除部分瓦片,以使得从服务器端响应请求所返回的瓦片得以缓存。好的缓存置换算法对于缓存效率有重大影响。缓存置换策略在 Web 和多媒体视讯等应用中得到广泛研究,发现基于缓存对象大小的 SIZE 置换策略在命中率上要好于最近最少使用 LRU(Least Recently Used),最不经常使用 LFU(Least Frequently Used)和先进先出 FIFO(First In First Out)策略^[11]。但针对空间数据金字塔模型的瓦片矩阵,瓦片是等大小的,以瓦片为缓存粒度时,无法实施 SIZE 置换策略,需要研究新的适用于这一特点的置换策略。

2.2.1 代表性缓存置换算法的瓦片时空关联分析

缓存置换算法中所涉及的空间数据的时空关联,主要指在缓存置换过程中,需要重点考虑瓦片访问的时间局部性和空间局部性。时间局部性指最近被访问的瓦片在不久将来有更高的概率再次被访问。当前时刻与瓦片被用户访问时刻的时间间隔,是瓦片访问时间局部性的体现。LRU 是利用时间局部性的代表性置换算法,认为缓存中瓦片的再次访问概率 p 反比于瓦片被用户访问时刻与当前时刻的时间间隔,于是(\sim 为正比符号)
 $p \sim 1/\Delta t$, $\Delta t = \text{currentSystemTime} - \text{lastAccessTime}$
 置换时,将 p 最小(即 lastAccessTime 最小)的瓦片置换出缓存。LRU 依据瓦片最近一次访问时间,体现了瓦片短期流行度,能较好地适应地形漫游时用户对访问模式的变化。但 LRU 只是瓦片访问时间上的局部优化,没有考虑瓦片的长期访问特性,无法体现瓦片长期流行度。

瓦片访问的空间局部性是指,在空间距离上相邻的瓦片总是倾向于在被访问时间上也相邻,

即地形漫游时,某瓦片在某时刻被访问,则下一刻该瓦片附近的瓦片(包括该瓦片本身)有更高的概率再次被访问。缓存中瓦片与当前瓦片的距离,是瓦片访问空间局部性的体现。通常,瓦片数据以 3 维金字塔模型进行组织和管理,由于地形漫游时处于视角中心的当前瓦片总是在快速地变化,实时计算缓存中已有瓦片与当前瓦片的距离困难较大而且计算量、复杂度较高,且“距离”没有统一的定义(下一步工作方向)。但这并不说明瓦片访问的空间局部性就不能被利用开发。文献[12]指出,对空间对象的访问收敛于特定的工作区,即空间实体访问的空间局部性表现为如果特定区域被频繁访问,那么该区域内的空间对象也将以高概率被访问。考虑瓦片对象的空间局部性的特例,即考虑与缓存中瓦片零距离的瓦片(缓存中的瓦片本身)的空间局部性,文献[13]指出可将瓦片访问的空间局部性体现在瓦片的长期流行度上。文献[14]和[15]显示,空间数据瓦片的访问呈现集中访问的特性,频繁请求的瓦片,其长期访问概率符合幂率(power-law)关系的齐普夫定律(Zipf's law),即长期流行度排在第 i 位的瓦片的长期访问概率为

$$p_i = \Omega / i^\alpha$$

其中, $\Omega = (\sum_{i=1}^Z \frac{1}{i^\alpha})^{-1}$, Z 为瓦片金字塔中的瓦片总数, $\alpha \in [0.5, 1]$ 为 Zipf 参数, $1 \leq i \leq Z$ 。因此,地形漫游时对缓存中瓦片的再次访问概率 p ,也可以简化地看作正比于该瓦片的长期流行度,即瓦片的访问频率 totalAccessTimes 。于是有

$$p \sim \text{totalAccessTimes}$$

LFU 置换缓存中访问频率 totalAccessTimes 最小的瓦片,就是瓦片访问空间局部性的一个应用。LFU 本质上依据瓦片访问的 Zipf 定律,利用了瓦片访问的长期流行度,认为瓦片的访问频率越高,未来再访问的可能性也越高。利用长期流行度,LFU 考虑了瓦片访问的空间局部性和全局特性,有利于瓦片的总体优化访问。但 LFU 无法适应瓦片访问模式变化和突发性访问,即会造成缓存污染^[16],这与 LRU 对访问方式的变化特别敏感刚好相反。

可见,利用瓦片访问的时间局部性原理的 LRU 与利用瓦片访问的空间局部性原理的 LFU 具有一定的互补性。此外,先进先出 FIFO 算法认为,缓存中瓦片再次访问概率 p 反比于当前时

刻与瓦片首次存储于缓存中的时刻的时间间隔,有

$p \sim 1/\Delta t, \Delta t = \text{currentSystemTime} - \text{firstStoreTime}$
FIFO 置换 p 最小(即 firstStoreTime 最小)的瓦片。

2.2.2 瓦片访问平均时间间隔最长置换算法

基于第 2.1 节的瓦片缓存索引,可以非常方便地实现 LRU,LFU 和 FIFO。然而,LRU 只考虑了瓦片访问的时间局部性,利用瓦片的短期流行度使得 LRU 只能对瓦片访问进行局部优化,无法区分缓存中经常访问和很少访问的瓦片;只考虑了瓦片访问空间局部性的 LFU 可能造成缓存存储区污染^[16],即之前获得高访问频率的瓦片将长期存储在缓存中;FIFO 忽视了用户对特定地理区域的信息感兴趣这一现象,删除了符合用户初始兴趣的瓦片。

综合考虑上述几种代表性的置换策略优缺点,利用瓦片顺序缓存索引,提出一种瓦片访问时间间隔最长置换算法 TAIL (Tile Access Interval time Longest replacement)。假设某瓦片 j 在缓存置换时已被访问 $\text{totalAccessTimes}(j)$ 次,访问时刻从旧到新依次记为 $T_1^j, T_2^j, \dots, T_{\text{totalAccessTimes}(j)}^j$,瓦片 j 的平均访问时间间隔记为 $\text{tail}(j)$,则有

$$\text{tail}(j) = \frac{\sum_{i=1}^{\text{totalAccessTimes}(j)-1} (T_{i+1}^j - T_i^j)}{\text{totalAccessTimes}(j)} = \frac{\text{lastAccessTime}(j) - \text{firstStoreTime}(j)}{\text{totalAccessTimes}(j)}$$

TAIL 算法认为缓存中瓦片的再次访问概率 p 反比于平均访问时间间隔,有

$$p \sim 1/\text{tail}(j)$$

在置换缓存瓦片时,将再次访问概率 p 最小(即平均访问时间间隔最长)的瓦片 K 删除。

2.2.3 TAIL 算法的瓦片时空关联分析

实际上,将 currentSystemTime 引入 $\text{tail}(j)$,有

$$\text{tail}(j) = \frac{\text{lastAccessTime}(j) - \text{firstStoreTime}(j)}{\text{totalAccessTimes}(j)} = \frac{\text{currentSystemTime} - \text{firstStoreTime}(j)}{\text{totalAccessTimes}(j)} + \frac{-(\text{currentSystemTime} - \text{lastAccessTime}(j))}{\text{totalAccessTimes}(j)}$$

由第 2.2.1 节的分析可见,TAIL 等效于 (FIFO+LRU)/LFU,即 $\text{TAIL} \sim (\text{FIFO} + \text{LRU})/\text{LFU}$ 。

TAIL 具有简单明确的物理意义,表示缓存中瓦片访问的平均时间间隔。而本质上,TAIL

置换算法综合了 FIFO,LRU,LFU 3 种算法,巧妙地利用瓦片访问的时间局部性和空间局部性,考虑了缓存中瓦片与当前瓦片的距离、瓦片被用户访问时刻与当前时刻的时间间隔是两个不可忽略的重要因素。TAIL 可以在瓦片访问的长期流行度和短期流行度间取得一个平衡,可以平滑或屏蔽 3 种算法单独使用时的缺点,即有利与瓦片访问的整体优化,又可以适应于不同用户地形漫游时对瓦片访问的模式变化和突发性访问。

2.2.4 TAIL 的置换效率

TAIL 仅仅针对缓存中的所有瓦片进行访问时间间隔的计算,区别于文献[5]中针对整个空间数据集中的所有瓦片进行转移概率计算。缓存中的瓦片数量相对于整个空间数据集中的瓦片数量,可以忽略不计,因此 TAIL 算法相比文献[5]是容易实施的。另一方面,进行 TAIL 算法的计算量并不大,为进一步加快置换过程,可以扩展第 2.1 节的缓存索引,在原缓存索引基础上再增加一个属性 tail ,使得 $\text{TileIndex}[j][5] = \text{tail}(j)$ 。每当有瓦片缓存命中时,首先更新该瓦片的索引的第 2、第 3 项属性 lastAccessTime 和 totalAccessTimes ,然后计算 $\text{tail}(j) = \frac{\text{lastAccessTime}(j) - \text{firstStoreTime}(j)}{\text{totalAccessTimes}(j)}$ 并将之赋值于 $\text{TileIndex}[j][5]$ 。此情况下, tail 的计算可以在后台计算,不影响当前进程。当需要置换时,在索引中的所有 N 个 $\text{TileIndex}[j][5]$ 中寻找具有最大值的索引并将该索引中的 $\text{TileIndex}[i][4] = \text{cacheStoreOffset}$ 记录后,将该索引删除并根据 cacheStoreOffset 将缓存存储区对应的瓦片删除。

由于缓存索引数组不是以 $\text{TileIndex}[j][5]$ 为键值的顺序数组,缓存索引对 $\text{TileIndex}[j][5]$ 的查找效率不高。而且,由于用户地形漫游时,对瓦片的访问具有位置连续性,当某个瓦片在缓存中未命中时,用户接下来漫游所需要的瓦片将以较大可能仍然不能命中。因此这种置换方法每次操作只针对一个瓦片,当有连续的瓦片请求未命中而缓存又满时,需要多次频繁执行删除操作,影响了新瓦片缓存时操作的效率。为进一步地提升效率,可以设置一个缓存大小下限 $\text{cache_size_lowerbound}$ MB。当缓存满需要置换时,一次性删除 M 个平均访问时间间隔最长的瓦片,使得缓存存储区已用缓存的大小小于 cache_size_

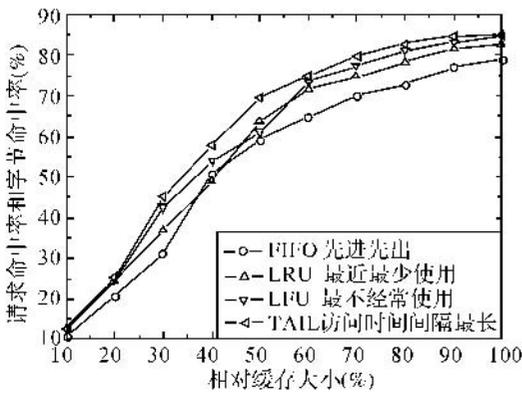
lowerbound, 即 $M \times (tile_size \times tile_size \times 3) > 1\,024 \times 1\,024 \times (cache_size - cache_size_lowerbound)$ 。

3 实验结果

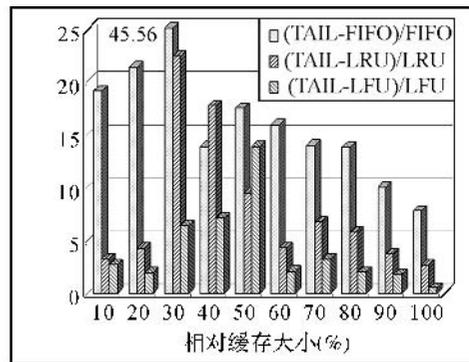
采用日志驱动仿真实验,考察 TAIL 在瓦片请求命中率、请求响应时间及服务器处理瓦片请求数 3 个方面的性能。实验数据集为最高分辨率为 30 m 的全球 LANDSAT7 影像数据,瓦片大小为 512×512 像素,以及最高分辨率为 90 m 的全球 SRTM 地形数据集,瓦片大小为 128×128 像素。日志文件由 Globesight 客户端产生,记录了某时段高峰访问时,在 35 378 s 内多个客户端针对影像数据和地形数据分别发出的 36 000 个瓦片请求信息。通过分析日志,发现这 36 000 个瓦片请求所对应的影像数据总量为 $2.831\,16e+10$ 字节,约 26.37 GB,不同请求所对应影像数据总量为 $4.199\,55e+09$ 字节,约 3.92 GB;36 000 个瓦片请求对应的地形数据总量为 $1.769\,9e+09$ 字节,约 1.65 GB,不同请求对应地形数据总量为 $2.630\,96e+08$ 字节,约 0.25 GB。从实验结果发现,针对影像数据和地形数据,TAIL 的性能均好

于 FIFO,LRU 和 LFU,且表现出相似的优越性。为节约篇幅,下面仅对影像数据的实验结果进行讨论。

图 3(a)是 FIFO,LRU,LFU 和 TAIL 4 种算法的请求命中率和字节命中率与缓存大小的关系图,图 3(b)是 TAIL 针对 FIFO,LRU,LFU 的命中率提高百分比,其中缓存大小采用百分比表示,仿真中采用的缓存大小为所有不同请求的数据总量的百分数。需注意的是,在实验中,由于单一数据集的瓦片矩阵中瓦片大小一致,在以瓦片为缓存粒度的缓存算法中,请求命中率和字节命中率是等同的。此外,实验中未对基于瓦片大小作置换操作的 SIZE 算法进行实验。由图 3 可见,FIFO 算法表现出最差的缓存命中效率;LRU 与 LFU 算法比 FIFO 好,LFU 在大多数情况下稍好于 LRU;TAIL 在所仿真的 4 种置换算法中表现出最好的命中率,这是由于 TAIL 综合考虑了 FIFO,LRU 和 LFU 的优缺点,最大限度地利用了瓦片缓存索引中所有可用信息,对地形漫游时瓦片访问的时间局部性和空间局部性都进行了考察。



(a) 4 种算法的命中率
(a) Cache Hit Rate of TAIL.



(b) TAIL 对命中率提高百分比
(b) Percent Improvement in Hit Rate of TAIL

图 3 全球 LANDSAT7 影像数据集的缓存命中率与缓存大小关系

Fig. 3 Cache hit rate vs. cache size in LANDSAT7 image data

图 4 是并发用户数为 256,没有使用缓存(以相对缓存大小为 0 表示)、使用 TAIL 缓存并在不同相对缓存大小情况时,仿真实验达到稳定状态后,用户瓦片请求响应时间的对比图。可见,当 Globesight 客户端被屏蔽缓存功能时,用户响应时间非常长,这可能导致地形漫游系统不流畅。而开启 TAIL 缓存功能后,用户瓦片请求响应时

间得到大幅的缩短,缓存越大,TAIL 瓦片命中率越高,瓦片请求响应时间越短。一个较为重要的现象是:Globesight 客户端仅仅开启较小的缓存空间,即可较大幅度地缩短响应时间。图 4 中,当相对缓存大小为 10%(约 400 MB)时,瓦片请求响应时间相对于无缓存时,响应时间缩短了 17.8%。256 为单一服务器情况下能够保证空

间数据服务质量(瓦片响应时间小于 4 s)时所能支持的最大并发用户数^[10]。当并发用户数超过 256 时,瓦片请求响应时间在无缓存时超过 10 s,而执行缓存策略则可以降低至 3 s 以内;由于有、无缓存时瓦片响应时间差别较大,不便于绘制于同一图中,本文未给出其具体图示,但实验结果显示,TAIL 缓存均可大幅缩短响应时间。

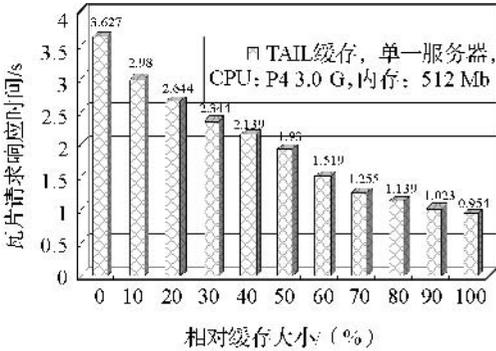


图 4 全球 LANDSAT7 的瓦片请求响应时间
Fig. 4 Tile request response time in LANDSAT7 image data

图 5 是仿真实验达到稳定状态后,应用服务器每秒处理的瓦片请求数的性能图。该图反映了无缓存时以及采用 TAIL 缓存在不同缓存大小时,服务器端处理空间数据的负载情况。当无缓存时(以相对缓存大小为 0 表示),服务器每秒处理的瓦片请求个数平均约为 0.8 个,而在客户端的实际日志中,每秒钟平均发送 1 个瓦片请求(36 000 个请求/35 378 s),服务器处理的瓦片请求数小于客户端实际发送的瓦片请求数。原因在于,客户端在作快速地形漫游时,部分瓦片请求被系统取消以减少无用的瓦片请求。由图 5 可见,使用缓存后,服务器每秒钟处理的瓦片请求个数得到较大幅度的减少,特别是当缓存相对大小为 30%(对应缓存命中率达到 46%)时,服务器处理瓦片请求数降低到无缓存时的一半左右;再增加缓存空间时,瓦片请求数降幅更大,当相对缓存大小为 100%(TAIL 命中率达到 86%)时,服务器的负载仅仅为无缓存时负载的 11.3%。因此,使用客户端缓存,结合较好的缓存算法,能够明显的降低服务器端负载,以支持更多的并发用户数。

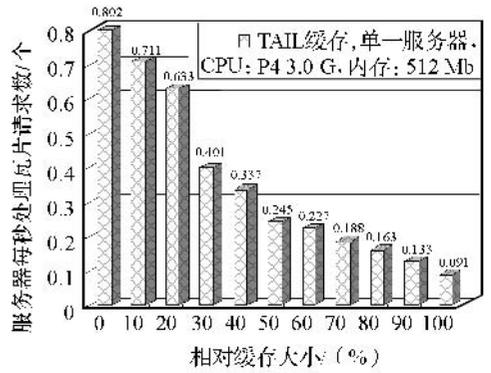


图 5 服务器处理 LANDSAT7 瓦片请求数
Fig. 5 Tile request number processed by NGIS server in LANDSAT7 image data

4 结 语

网络地理信息服务中的海量空间数据的高效传输与可视化是影响地形漫游服务质量的重要因素。针对地形数据和影像数据,以瓦片为缓存粒度,在客户端部署空间数据缓存,并详细讨论缓存索引和基于瓦片访问平均时间间隔最长的置换算法 TAIL。实验结果表明,与 LRU, LFU, FIFO 等算法相比,TAIL 拥有更高的瓦片请求命中率和字节命中率,能大幅降低用户访问瓦片的响应时间和服务器端所处理的瓦片请求数。下一步工作包括,将本文的缓存算法同时部署到文献[7]中的客户端缓存、反向代理缓存和服务端缓存中,以较清晰地考察不同等级的分布式缓存的聚集合作效应,以及研究如何在 3 维瓦片金字塔模型中,定义任意 2 个瓦片间的距离以更深入地研究瓦片访问的空间局部性,为空间数据缓存算法提供更好的理论支持。

参考文献:

[1] ZHANG Liqing, ZHANG Yan, YANG Chongjun, et al. Effective Solutions to a Global 3D Visual System in Networking Environments[J]. Science in China: D Earth Sciences, 2005, 48(11): 2032-2039.

[2] SAHM J, SOETEBIER I, BIRTHELMER H. Efficient Representation and Streaming of 3D Scenes[J]. Computers & Graphics, 2004, 28: 15-24.

[3] YANG Bisheng, LI Qingquan. An Algorithm for Progressive Transmission Vector Map Data over the WWW[J]. Acta Geodaetica et Cartographica Sinica, 2005, 34(4): 355-360. (杨必胜, 李清泉. World Wide Web(WWW)上矢量地图数据的多分辨率传输算法 [J]. 测绘学报, 2005, 34

- (4): 355-360.)
- [4] YU Zhanwu, ZHENG Sheng, LI Zhongmin. A Large-scale Terrain Transmission Mechanism Based on Hybrid P2P[J]. Acta Geodaetica et Cartographica Sinica, 2008, 37(2): 243-249. (喻占武, 郑胜, 李忠民. 一种混合式 P2P 下的大规模地形数据传输机制 [J]. 测绘学报, 2008, 37(2): 243-249.)
- [5] KANG Y K, KIM K C, KIM Y S. Probability-based Tile Pre-fetching and Cache Replacement Algorithms for Web Geographical Information Systems[C]//5th East European Conf on Advances in Databases and Information Systems. Vilnius: Springer Press, 2001: 127-140.
- [6] TU S R, HR X, LI X, RATCLIFF J J. A Systematic Approach to Reduction of User Perceived Response Time for GIS Web Services[C]//The Ninth ACM Intern Symp on Advances in Geographic Information System. Washington: ACM Press, 2001: 47-52.
- [7] LI Haosong, ZHU Xinyan, LI Jingwei, et al. Research on Spatial Data Distributed Cache Technology in WebGIS[J]. Geomatics and Information Science of Wuhan University, 2005, 30(12): 1092-1095. (李浩松, 朱欣焰, 李京伟, 等. WebGIS 空间数据分布式缓存技术研究 [J]. 武汉大学学报: 信息科学版, 2005, 30(12): 1092-1095.)
- [8] YU Zhanwu, LI Zhongmin, ZHENG Sheng. Network Geographic Information System Architecture Based on Object-based Storage[J]. Geomatics and Information Science of Wuhan University, 2008, 33(3): 285-288. (喻占武, 李忠民, 郑胜. 基于对象存储的新型网络 GIS 体系结构研究 [J]. 武汉大学学报: 信息科学版, 2008, 33(3): 285-288.)
- [9] LUO Yingwei, WANG Xiaolin, XU Zhuoqun. Design and Implementation of a Hierarchical Component-based WebGIS[J]. Chinese Journal of Computers, 2004, 27(2): 177-185. (罗英伟, 汪小林, 许卓群. 层次化 WebGIS 构件系统的设计与实现 [J]. 计算机学报, 2004, 27(2): 177-185.)
- [10] WU Huayi, ZHANG Hanwu. QoGIS: Concept and Research Framework[J]. Geomatics and Information Science of Wuhan University, 2007, 32(5): 385-388. (吴华意, 章汉武. 地理信息服务质量 (QoGIS): 概念和研究框架 [J]. 武汉大学学报: 信息科学版, 2007, 32(5): 385-388.)
- [11] ROLAND P W. Optimizing Response Time Rather than Hit Rates of WWW Proxy Caches[D]. Blacksburg: Virginia Polytechnic Institute and State University, 1996.
- [12] MIN J K. Beast: A Buffer Replacement Algorithm Using Spatial and Temporal Locality [C]//IET International Conference on Computational Science and Its Applications. Glasgow: Springer Press, 2006: 67-76.
- [13] TALAGALA N, ASAMI S, PATTERSON D, et al. The Art of Massive Storage: a Web Image Archive[J]. IEEE Journal of Computer, 2000, 33(11): 22-28.
- [14] FISHER D. Hotmap: Looking at Geographic Attention [J]. IEEE Transactions on Visualization and Computer Graphics, 2007, 13(6): 1184-1191.
- [15] TALAGALA N, ASAMI S, PATTERSON D. Usage Patterns of a Web-based Image Collection [C]//16th IEEE Symposium on Mass Storage Systems. San Diego: IEEE Press, 1999: 203-214.
- [16] TOLY C. Obtaining the Optimal Cache Document Replacement Policy for the Caching System of an EC Website [J]. European Journal of Operational Research, 2007, 181(2): 828-841.

(责任编辑: 丛树平)

收稿日期: 2008-07-02

修回日期: 2009-02-10

第一作者简介: 王浩(1981—), 男, 博士生, 研究方向为多媒体网络。E-mail: alexwhu@163.com

First author: WANG Hao(1981—), male, PhD candidate, majors in multimeida networks.

E-mail: alexwhu@163.com

《测绘学报》统计数据

《中国学术期刊综合引证报告 2008》是以 CNKI“中国知识资源总库”中最大的文献信息资源——“中国期刊全文数据库”, 所收录的各类学术期刊 6 631 种的引文数据为基础编制而成的一部大型综合性科学文献计量年报。

根据科学出版社出版的《中国学术期刊综合引证报告》(CAJCCR) 2008 版(总第 7 卷)统计,《测绘学报》影响因子 1.403, 5 年影响因子 1.769, 在其所收录的“遥感与测绘”类 25 种科技期刊中名列第一; 总被引频次 1 245 次, 名列第 3; 基金论文比 89%, 他刊总引比 91%, 被引期刊数 319 种, 中国知网(<http://www.cnki.net>)Web 即年下载率 53%。