第四章 元数据分布信息缓存管理

集中方式的元数据请求分布管理要求有效机制的支持，以降低对集中决策服务器的依赖，弱化其对系统扩展能力的限制。用户元数据请求的局部性特征决定了活跃元数据被反复访问的概率很大，有效的元数据分布信息缓存能够缓减集中决策服务器的压力，提升其逻辑处理能力，增强系统的可扩展性。

本章结合 BWMMS 讨论集中元数据请求分布管理机制中的元数据分布信息的缓存管理，包括缓存的组织、元数据请求语义带来的缓存项的状态转换、缓存信息的替换策略等问题。最后通过实验评测元数据分布信息缓存对系统的影响。

4.1 元数据分布信息缓存的必要性

由于其简单性和有效性，集中方式的元数据请求分布管理机制具有重要的地位。尽管单个服务器因其有限的物理处理能力，而限制系统的扩展。但是，有效的请求分布管理策略可以降低集中决策服务器的负载。并且，以层叠方式组织和扩展的集中决策服务器系统结构，可以增强对前端服务器规模有效扩展的支持。

缓存技术通过有效的缓存信息缩短请求的处理路径，提高服务器的逻辑处理能力[Levy1990]。计算机系统中存在各种目的的信息缓存，如本地文件系统的目录项缓存、索引节点缓存、数据缓存[Thompson1978][Mckusick1984][Bach1986][Vahalia1996][Lions1996][Bovet2002][Tanenbaum2006]，分布式文件系统客户端的数据和元数据缓存[Dahlin1994-1][Dahlin1994-2]等。

对于文件系统元数据分布信息缓存而言，传统的缓存管理技术，如基于链表的组织结构、通过哈希法的快速查找、以及基于 LRU 的缓存替换算法等仍然适用。但是，它还需要结合元数据请求的语义，更优地组织和管理缓存。

在集中方式的元数据请求分布管理架构中，元数据分布信息缓存将起到至关重要的作用，包括：

1）降低集中决策服务器的负载，提高其逻辑处理能力。

尽管集中决策服务器的物理处理能力有限。但是，如果元数据访问负载鲜有突发、元数据请求不会分散到文件系统很广的范围时，元数据分布发生改变的概率很小，缓存的元数据分布信息将长时间有效，大多数元数据请求的处理不需要集中决策服务器的参与。这能够降低集中决策服务器的负载，提高集中决策服务器的逻辑处理能力。

2）缩短元数据请求处理路径。

在元数据服务器上，元数据请求的处理首先需要明确元数据的分布，以决定请求的

处理权限。文件系统需要根据用户可读的文件名字获得索引节点的标识，再根据索引节点标识获取文件索引节点及相关的元数据 [Bach1986]。元数据分布信息缓存能够支持元数据服务器根据请求处理权限，判断是否需要访问存储设备上的元数据。所以，元数据请求分布信息的缓存不仅可以缩短元数据分布信息的访问路径，还可以控制元数据的读取，消除不必要的存储设备访问，缩短元数据的访问路径。

4.2 元数据分布信息缓存管理的相关定义

用户的访问驱动 MS 请求元数据的分布映射。元数据分布结果将包含两种情况。一种是分布在自己的具有宿主权限的元数据分布信息，另一种是分布在其它 MS 的没有宿主权限的元数据分布信息。

只有在元数据不活跃时，元数据宿主 MS 才请求集中决策服务器解除其分布管理。并且，根据文件系统名字空间结构遍历时，将导致不具备宿主权限的 MS 缓存元数据的分布信息。所以，元数据分布信息缓存不仅需要管理当前所有的具有宿主权限的元数据分布信息，还需要缓存用户访问到、但没有宿主权限的元数据分布信息。

结合第 3.4.1 节，元数据分布信息缓存管理相关的定义为：

定义 4.1 BS 管理的系统当前所有活跃元数据分布信息集合 GMDT:

$$GMDT = \{item \mid item \in AM\}$$

GMDT 是所有活跃元数据的分布信息。所以，$GMDT = AM$ 。

定义 4.2 第 i 个元数据服务器 MSi 的元数据分布信息缓存称为“MDTi”。MDTi 中，分布在自己的具有宿主权限的元数据分布信息集合，称为“LDTi”。 LDTi 中的元数据分布信息项，称为“宿主权限项”:

$$LDTi = \{item \mid item \in MDTi \wedge MSi = HOST(item)\}$$

定义 4.3 MDTi 中，没有分布在自己的不具有宿主权限的元数据分布信息集合，称为“RDTi”。 RDTi 中的元数据分布信息项称为“非宿主权限项”:

$$RDTi = \{item \mid item \in MDTi \wedge MSi \neq HOST(item)\}$$

MDTi、LDTi 和 RDTi 的关系为：

$$MDTi = LDTi \cup RDTi,\ LDTi \cap RDTi = \varnothing$$

综合以上定义，GMDT 和各个 LDT 间存在的关系为：

$$GMDT = \bigcup_{i=1}^{n} LDT_i \ ,\ \bigcap_{i=1}^{n} LDT_i = \varnothing$$。其中n=|S|

4.3 元数据分布信息缓存的组织

元数据分布信息缓存管理首先需要确定标识缓存项的关键字。索引节点号能够唯一标识设备上的索引节点，它是关键字的一部分。在分布式文件系统中，由于文件删除操作不会广播给客户端，被删除文件仍然可能收到元数据请求。同时，已删除文件的索引节点号可能再次分配使用。为正确区别用户访问的目标索引节点，NFS[Sandberg1985]采用“索引节点号+索引节点序列号”的双关键字方式唯一标识用户访问的文件，索引节点序列号在新文件创建时递增。单位存储价格的降低[Henson2006]促进低成本的多版本文件支持。所以，为支持分布式文件系统的扩展，BWMMS 使用“索引节点号+索引节点序列号+索引节点版本号”作为元数据分布信息项关键字。

与传统的缓存管理相同，元数据分布信息缓存使用哈希链表提高分布信息项的查找速度，目标哈希链表通过 $func(ino\#, generation\#, version\#)$ 确定。缓存采用 LRU 管理缓存项的替换。在 LRU 基础上，由于用户访问特征决定了宿主权限项和非宿主权限项具有不同的访问概率，系统应该分开管理这两类信息。所以，缓存采用两个不活跃链表 entry_unused_hosted 和 entry_unused_unhosted 分别管理这两类不活跃项，并基于 LRU 决定分布信息项的替换。表 4.1 是元数据分布信息的结构。

表 4.1 元数据分布信息结构

```
struct mdt_entry {
    struct list_head me_list; /*list */
    struct list_head me_hash; /* hash list*/
    ino_t me_ino; /*  索引节点号*/
    u32 me_generation;/*索引节点序列号*/
    u32 me_version;/*索引节点版本号*/
    u32 me_host; /* MS 标示*/
    spinloct_t me_lock;
    u32 me_state; /* state */
    atomic_t me_refcnt; /* reference count，>0 为活跃元数据*/
    /*用来同步元数据请求*/
    u64 me_timestamp;
    atomic_t me_modicnt;
    atomic_t me_looupcnt;
    atomic_t me_unlinkcnt;
    atomic_t me_linkcnt;
    atomic_t me_renamecnt;
};
```

字段 me_list 和 me_hash 用来组织元数据分布信息缓存。通过这两个字段，MDT 的内存组织如图 4.1 所示。me_refcnt 记录该元数据分布信息项的活跃度。链表 entry_in_use 用来组织引用计数 me_refcnt 大于 0 的活跃元数据项。Entry_unused_hosted 和 entry_unused_unhosted 分别用来组织不活跃的宿主权限和非宿主权限项。Me_ino 记录索引节点号，me_generation 记录索引节点序列号，me_version 记录索引节点版本号。Me_host

记录该文件索引节点当前的宿主 MS 信息。其他字段主要用作元数据请求的同步控制，在第六章将具体阐述。

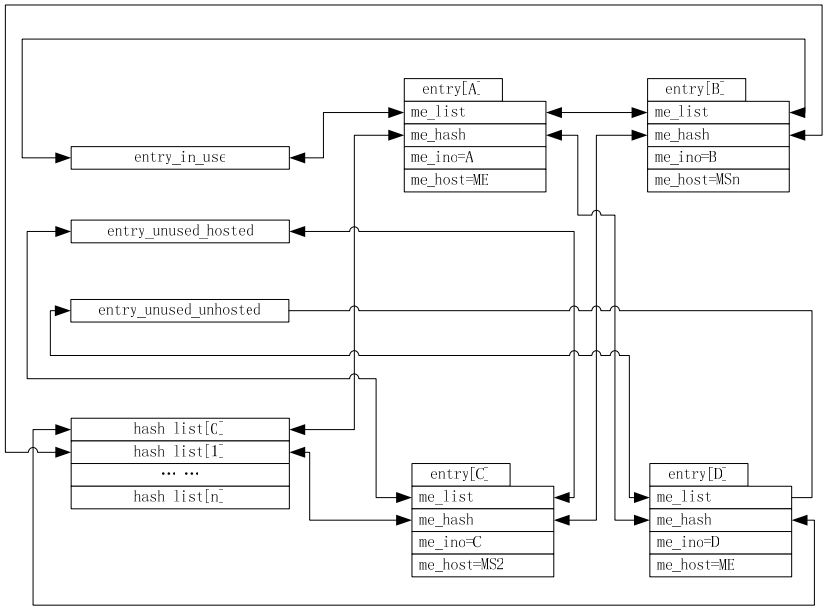

图 4.1 元数据分布信息缓存内存组织

4.4 元数据分布信息的状态转换

元数据请求语义将导致元数据分布信息的状态转换。根据元数据活跃性管理元数据请求分布的策略决定了元数据在第一次访问前和最后一次访问后没有相应的缓存项存在。并且，不同类型的元数据分布信息项将支持不同的请求处理流程。所以，在用户访问驱动下，两类元数据分布信息项具有不同的状态转换。从整体而言，元数据分布信息缓存项的生存周期从用户第一次请求时生成内存结构开始，到不活跃时释放内存结构结束。在生存周期过程中，它将随着元数据请求发生状态的转换。其整体的状态转换 Petri Net 表示如图 4.2 所示，起始状态仅位置 Pno-entry 具有 1 个标记。

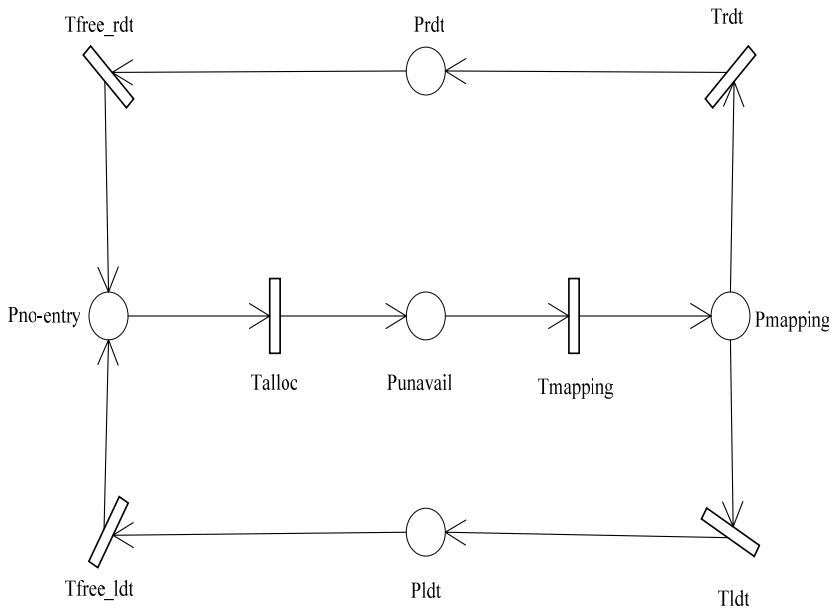

图 4.2 元数据分布信息的整体状态转换图

其中各个位置的含义为：

表 4.2 整体状态转换图的位置含义

位置名	含义
Pno-entry	没有对应缓存项存在，需要分配内存结构
Punavail	具有内存结构，但其中信息还不可用，需要请求 BS 决策其请求分布
Pmapping	正在请求 BS 决策过程中
Pldt	具有宿主权限
Prdt	没有宿主权限

各个变迁的含义为：

表 4.3 整体状态转换图的变迁含义

变迁名	含义
Talloc	为分布信息项分配内存结构
Tmapping	与 BS 通信，请求分布决策
Tldt	请求结果，具有宿主权限
Tfree_ldt	释放宿主权限的分布信息项
Trdt	请求结果，没有宿主权限
Tfree_rdt	释放非宿主权限的分布信息项

4.4.1 宿主权限项的状态转换

根据宿主权限项的请求语义，扩展图 4.2 的 Pldt 部分，得到图 4.3 的宿主权限项的状态转换的 Petri Net 描述，其起始状态是位置 Pno-entry 具有 1 个标记。