**Project17 MPT研究报告**

Merkle Patricia Tree(简称MPT树，实际上是一种trie前缀树)是以太坊中的一种用来组织管理账户数据、生成交易集合哈希的数据结构，可以用来存储所有的(key，value)对。

Trie树也叫作Radix树，为了提高效率，以太坊在实现上对其做了一些改进。在一般的radix树中，key是从树根到对应value得真实的路径。即从根节点开始，key中的每个字符会标识走那个子节点从而到达相应value。Value被存储在叶子节点，是每条路径的终止。假如key来自一个包含N个字符的字母表，那么树中的每个节点都可能会有多达N个孩子，树的最大深度是key的最大长度。与二叉查找树不同，键不是直接保存在节点中，而是由节点在树中的位置决定 。一个节点的所有子孙都有相同的前缀，也就是这个节点对应的字符串，而根节点对应空字符串。

一般情况下，不是所有的节点都有对应的值，只有叶子节点和部分内部节点所对应的键才有相关的值。实际上trie每个节点是一个确定长度的数组，数组中每个节点的值是一个指向子节点的指针，最后有个标志域，标识这个位置为止是否是一个完整的字符串。

MPT树的特点如下:

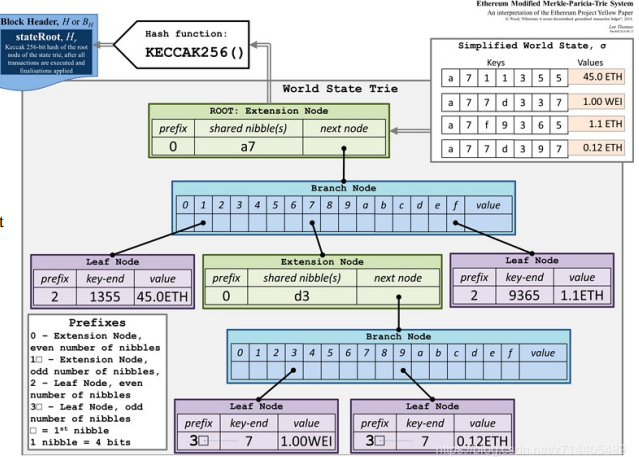
叶子节点和分支节点可以保存value, 扩展节点保存key；

没有公共的key就成为2个叶子节点；key1=[1,2,3] key2=[2,2,3]

有公共的key需要提取为一个扩展节点；key1=[1,2,3] key2=[1,3,3] => ex-node=[1],下一级分支node的key

如果公共的key也是一个完整的key，数据保存到下一级的分支节点中；key1=[1,2] key2=[1,2,3] =>ex-node=[1,2],下一级分支node的key; 下一级分支=[3],上一级key对应的value。

结构图如下：



通过hash可以还原出节点上的数据，因此这样只需要保存一个root(hash)，即可还原出完整的树结构，同时还可以按需展开节点数据。

以下为几种基本操作：

Get操作：

将需要查找Key的Raw编码转换成Hex编码，得到的内容称之为搜索路径；从根节点开始搜寻与搜索路径内容一致的路径；

1.若当前节点为叶子节点，存储的内容是数据项的内容，且搜索路径的内容与叶子节点的key一致，则表示找到该节点；反之则表示该节点在树中不存在。

2.若当前节点为扩展节点，且存储的内容是哈希索引，则利用哈希索引从数据库中加载该节点，再将搜索路径作为参数，对新解析出来的节点递归地调用查找函数。

3.若当前节点为扩展节点，存储的内容是另外一个节点的引用，且当前节点的key是搜索路径的前缀，则将搜索路径减去当前节点的key，将剩余的搜索路径作为参数，对其子节点递归地调用查找函数；若当前节点的key不是搜索路径的前缀，表示该节点在树中不存在。

4.若当前节点为分支节点，若搜索路径为空，则返回分支节点的存储内容；反之利用搜索路径的第一个字节选择分支节点的孩子节点，将剩余的搜索路径作为参数递归地调用查找函数。

Insert操作：

插入操作也是基于查找过程完成的，一个插入过程为：

根据上述描述的查找步骤，首先找到与新插入节点拥有最长相同路径前缀的节点，记为Node；

1.若该Node为分支节点：

2.剩余的搜索路径不为空，则将新节点作为一个叶子节点插入到对应的孩子列表中；

剩余的搜索路径为空（完全匹配），则将新节点的内容存储在分支节点的第17个孩子节点项中（Value）；

3.若该节点为叶子／扩展节点：

剩余的搜索路径与当前节点的key一致，则把当前节点Val更新即可；

剩余的搜索路径与当前节点的key不完全一致，则将叶子／扩展节点的孩子节点替换成分支节点，将新节点与当前节点key的共同前缀作为当前节点的key，将新节点与当前节点的孩子节点作为两个孩子插入到分支节点的孩子列表中，同时当前节点转换成了一个扩展节点（若新节点与当前节点没有共同前缀，则直接用生成的分支节点替换当前节点）；

4.若插入成功，则将被修改节点的dirty标志置为true，hash标志置空（之前的结果已经不可能用），且将节点的诞生标记更新为现在；

Delete操作：

删除操作与插入操作类似，都需要借助查找过程完成，一次删除过程为：

根据4.1中描述的查找步骤，找到与需要插入的节点拥有最长相同路径前缀的节点，记为Node；

1.若Node为叶子／扩展节点：

若剩余的搜索路径与node的Key完全一致，则将整个node删除；

若剩余的搜索路径与node的key不匹配，则表示需要删除的节点不存于树中，删除失败；

若node的key是剩余搜索路径的前缀，则对该节点的Val做递归的删除调用；

2.若Node为分支节点：

删除孩子列表中相应下标标志的节点；

删除结束，若Node的孩子个数只剩下一个，那么将分支节点替换成一个叶子／扩展节点；

3.若删除成功，则将被修改节点的dirty标志置为true，hash标志置空（之前的结果已经不可能用），且将节点的诞生标记更新为现在；