相似性搜索

作者：徐榕荟

一、 一维空间搜索

1. **二叉搜索树**：非空左子树的所有键值小于其根结点的键值，非空右子树的所有键值大于其根结点的键值。**存在不平衡现象，影响搜索效率。**

(1)二叉搜索树　　(2)不平衡的二叉搜索树

图 1 二叉搜索树

2. **二叉平衡树**。**搜索效率更高**。如：AVL、红黑树。

（1）AVL。左子树和左子树的高度之差的绝对值（平衡因子）不超过 1，且它的左子树和右子树都是一颗平衡二叉树。每当插入或者删除一个节点，为了让它重新维持在一个平衡状态，就需要对其进行旋转处理。学习链接：数据结构之——平衡二叉树（内容详解）-CSDN 博客

（2）红黑树。红黑树是一种接近平衡的二叉树，在每个节点增加了一个存储位来记录节点的颜色（红或黑）。学习链接：【数据结构】史上最好理解的红黑树讲解，让你彻底搞懂红黑树_小七 mod 的博客-CSDN 博客。

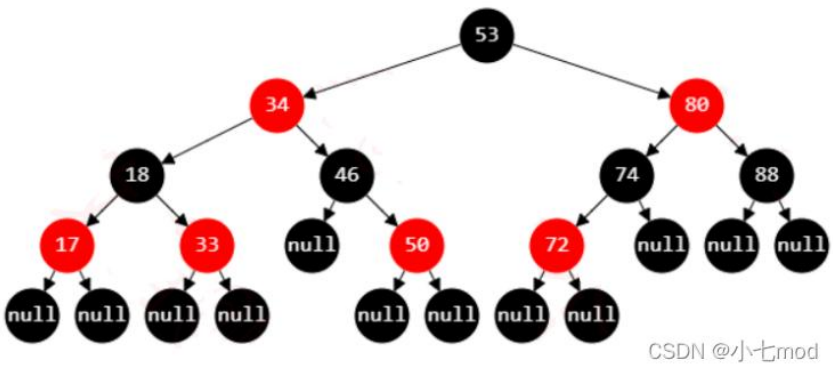


图 2 红黑树

3. **多叉树**。如：B-tree、B+-tree。数据库常用。学习链接：[BTree 和 B+Tree 详解 _b+brtt 的结构图-CSDN 博客]

（1）B-Tree 是为磁盘等外存储设备设计的一种平衡查找树。系统从磁盘读取数据到内存时是以磁盘块为基本单位的，位于同一个磁盘块中的数据会被一次性读取出来。页是其磁盘管理的最小单位，InnoDB 每次申请磁盘空间时都会将若干地址连续磁盘块来达到页的大小 16KB。在查询数据时如果一个页中的每条数据都能有助于定位数据记录的位置，这将会减少磁盘 I/O 次数，提高查询效率。

下图所示为一个 3 阶的 B-Tree：

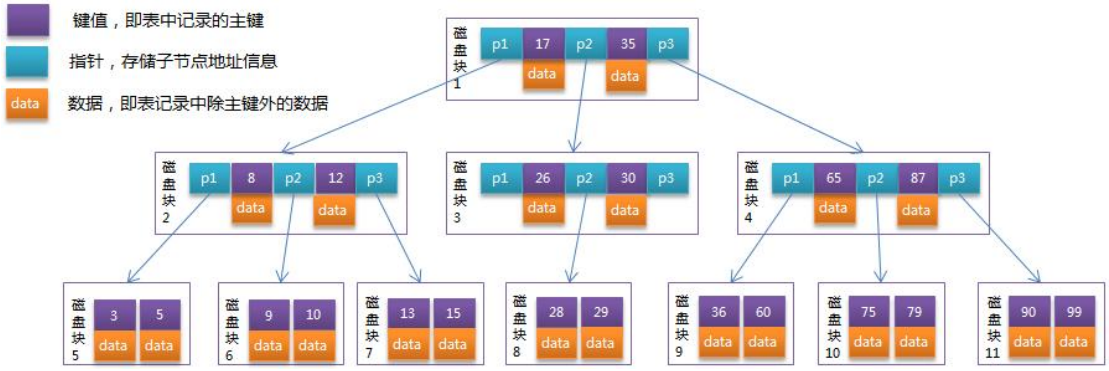


图 3 B-tree

（2）B+-tree。B+Tree 是在 B-Tree 基础上的一种优化，使其更适合实现外存储索引结构，InnoDB 存储引擎就是用 B+Tree 实现其索引结构。

B+Tree 相对于 B-Tree 有几点不同：

（a）非叶子节点只存储键值信息。

（b）所有叶子节点之间都有一个链指针。

（c）数据记录都存放在叶子节点中。

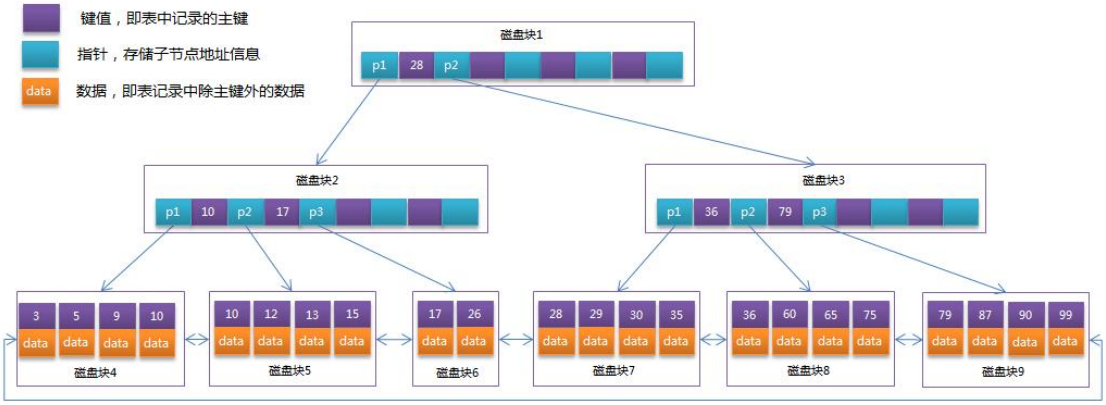


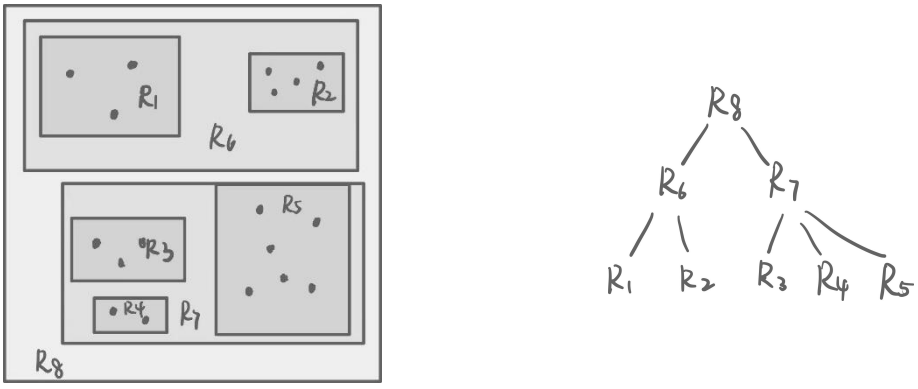
图 4 B+-tree

二、 多维空间搜索

1. R-tree

核心思想：聚合距离相近的节点并在树结构的上一层将其表示为这些节点的最小外接矩形，这个最小外接矩形就成为上一层的一个节点。叶子节点上的每个矩形都代表一个对象，节点都是对象的聚合，并且越往上层聚合的对象就越多。

优势：可剪枝。



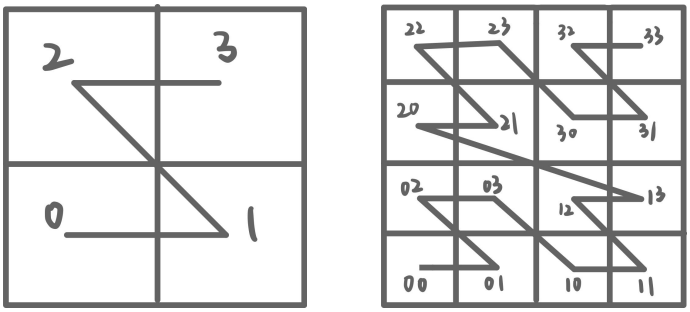
(2)点在空间上的结构　　(2)树结构

图 5 R-tree 示意图

R 树有很多变种，例如 R+-tree，以及 R*-tree 等，这些树的主要改进是在数据集合变化的过程中，矩形框不断分割或者合并的过程。

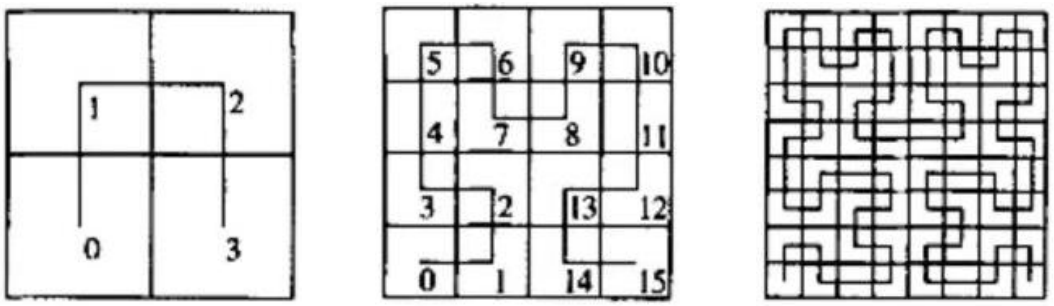
2. 空间填充曲线。如：Z-曲线，H-曲线。

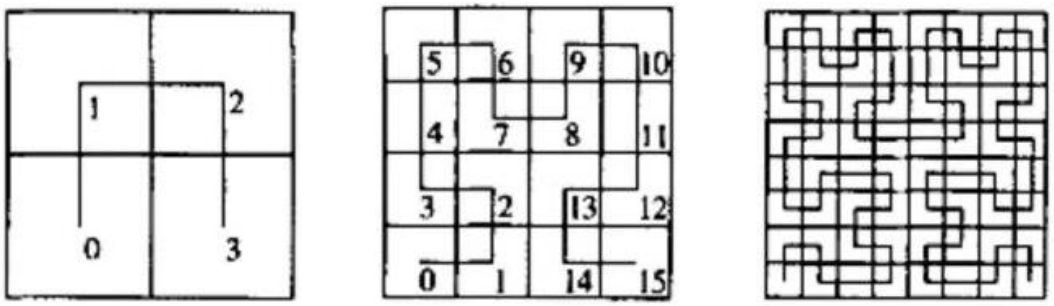
定义：空间填充曲线是一种降低空间维度的技术，可以将**高维空间数据映射到一维空间，并利用转换后的索引值存储和查询数据**。从数学的角度上看，可以将空间填充曲线看成是一种把 D 维空间数据转换到 1 维连续空间上的映射函数。空间填充曲线通过有限次的递归操作将多维空间划分为众多的网格，再通过一条连续的曲线经过所有的网格。



(1) 1 阶　　（2）2 阶

图 6 Z-曲线示意图





(2) 1 阶　　（2）2 阶　　（3）3 阶

图 7 H-曲线示意图

使用了空间填充曲线之后，再结合一维索引技术，就能够支持最近多维空间的相似性搜索。

3. VA-File （原论文：[PDF] A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces | Semantic Scholar）

核心思想：先查询每个格子中所有位置到查询点的距离范围，并按最近距离从小到大排序记录最近格子和与最近格子距离范围有交集的格子。然后从近到远依次查询这些格子内的点到查询点的距离，删去队列中最近位置大于上述距离的格子，直到队列为空。（先找格子再找点）

优势：搜索维度高时，VA-File 的搜索效率远远优于其他所有 tree 算法。

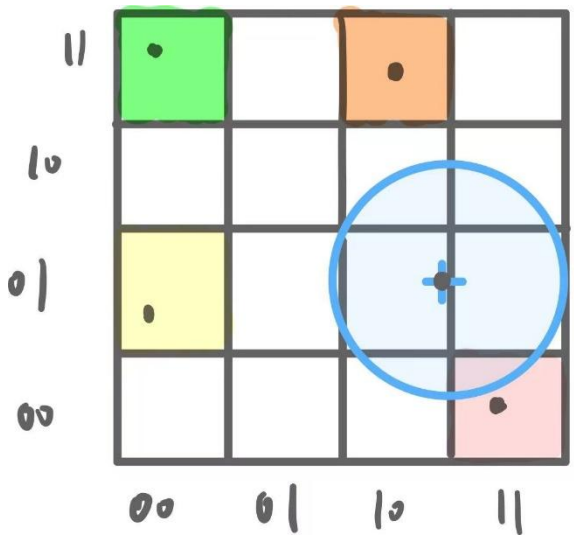


图 8 VA-File 示意图

示例：根据查询点位置得到最近点的候选格队列【1.粉格 2.橙格】（绿格和黄格因为最近点大于粉格的最远点而被淘汰），粉格出队列，然后计算粉格中点到查询点的距离并找到距离最小点，发现其距离小于到橙格的距离，删除队列中的橙格，队列为空，找到最相似的点。

4. 其他空间搜索方法

(1) 网格索引：**思想**是将研究区域用横竖线划分大小相等和不等的网格，每个网格可视为一个桶(bucket)，构建时记录落入每一个网格区域内的空间实体编号。进行空间查询时，先计算出查询对象所在网格，再在该网格中快速查询所选空间实体。**优点**：简单，易于实现，具有良好的可扩展性；**缺点**：网格大小影响网格索引检索性能。

(2) 四叉树索引是在网格索引的思想基础上，为了实现要素真正被网格分割，同时保证桶内要素不超过一个量而提出的一种空间索引方法。**构造方法**：首先将整个数据空间分割成为四个相等的矩阵，分别对应西北(NW)，东北(NE)，西南(SW)，东南(SE)四个象限；若每个象限内包含的要素不超过给定的桶量则停止，否则对超过桶量的矩形再按照同样的方法进行划分，直到桶量满足要求或者不再减少为止，最终形成一颗有层次的四叉树。**优点**：一定程度上实现了地理要素真正被网格分割，保证了桶内要素不超过某个量，提高了检索效率；**缺点**：对于海量数据，四叉树的深度会很深，影响查询效率可扩展性不如网格索引：当扩大区域时，需要重新划分空间区域，重建四叉树，当增加或删除一个对象，可能导致深度加一或减一，叶节点也有可能重新定位。

(3) 哈希索引…

课程内容参考链接：
https://www.researchgate.net/profile/Stephen-Blott/publication/220538823_What%27s_wrong_with_high-dimensional_similarity_search/links/553e01f10cf2fbfe509b81f6/Whats-wrong-with-high-dimensional-similarity-search.pdf