G-HASH[1]

贾新禹

2015 年 5 月 14 日

Contents

1 引言

目前，形如集合，序列，树和图等结构化数据给像高效存储，索引，部分查询 (如子图/超图搜索) 和相似性搜索之类的传统数据管理领域提出了一个巨大的挑战。随着图数据库的迅速发展，在图数据库中的图相似性搜索成为了一个日益重要的研究课题。图相似性搜索已经多个领域进行了应

用，例如化学，分子信息学，传感器网络管理，社交网络管理，XML 文档等等。在化学与制药领域，每天会产生大量的化学分子结构数据。一旦一个新的化学结构被合成后，这个结构的特性就可能通过查询已知的分子结构，通过其特性来预测。大规模图数据中的相似性搜索让科研人员和工程师们可以对图精确建模，认识到图数据间的固有联系，减少大数据库的计算代价。

图查询需要分成两种：(i) 子图查询 (ii) 相似度查询。*子图查询*目的是确定一个包含着查询图的集合。*相似性查询*是根据差异距离来找出数据库中和查询相似的图，如 k-NNs 查询 (最相近 k 查询) 和范围查询。k-NNs 查询是为了得到最相近的 k 个图。范围查询，是为了得到差异距离小于预设值的所有图。在本文中，我们只研究在大规模图数据库中的 k-NNs 相似性搜索。

在图上进行相似性搜索是十分富有挑战的。我们认为一个理想相似性搜索设计应该达到以下三个相关的 (有时是相反的) 目标: 准确度高，时间短，空间小。为了准确度高，我们要求相似性度量方法应该抓住数据的固有相似度。众所周知，一些图上的操作，像子图同构，都是 NP 问题。所以为了时间短，我们需要设计一个高效的算法来尽量避免全图搜索。为了空间小，所以索引结构不能对图数据库开销太大。

许多简单的图相似性度量方法是将图转换为在高维欧几里得空间 (*特征空间*) 中的表示，然后利用空间上的索引技术来进行相似度搜索。现有的特征提取方法大多数都可以归为两类:(i) 每幅图的子结构单独计数（如从一幅图中生成随机的通路），(ii) 图集合中的子结构一起计数（如挖掘频繁子结构）。尽管相似性搜索已经被广泛应用了，但是在特征提取的适配性和特征的索引策略上仍有很多限制。首先，图数据库（尤其是大规模的图数据库）的特征提取是需要大量运算量的。其次，特征提取过程可能产生很多特征值，需要占用大量内存空间。学术界曾提出过许多不同的特征选取方法来确定"有辨识度的"特征。但是，对于数据库搜索表现优异的特征选取方法在图相似性上不一定会很好，所以我们还需要自行去判断与权衡。

在本文中，我们探索了一种新的图相似性搜索方法。我们通过核函数来定义相似度。和通常不同的是，*核函数*没有直接提取特征值，而是将数据映射到一个高维的函数空间，并利用在这个函数空间中数据的内积来计算其相似度。基于核函数的图相似度测量方法的优势在于核函数统计学表现非常好，例如可以高精度的分类。但是将核函数用于数据库搜索也有些问题，主要的难点在于（i）核函数用在图上计算量很大（ii）目前为止，没有一个明确的方法来标引大规模图数据库上的核函数计算。

我们的方法称作 G-hash。我们致力于提出一种新的核函数来高效计算大规模图数据库上的特征。在我们的模型中，图被用核函数直接压缩成一个节点集。传统上，对于复杂的图结构，这样的压缩方法会丢失大量的信息。但是我们将大多数拓扑信息压缩到了每个节点的特征向量上来避免大量信息的丢失。这种方法提供了一个对于信息量丰富的图的简单表示，并且也很好进行比较。我们接着利用压缩集表示法将图节点哈希化。哈希的键值是基于这些表示集的，所以相似的节点会被哈希到同一个格子或者相邻的格子中。一旦我们将数据库中所有图哈希成一个表后，我们可以找到所有相似的节点，然后利用它们计算查询图和数据库中图的距离，并找出查询图的 k-NNs(k 个相近图)。

总体而言，我们这篇文章的主要贡献有:

- 提出了一个图核函数和用来进行快速图相似性搜索的索引结构
- 我们的索引只需线性时间去计算（对数据库中的总结点数而言），并且可以通过动态增删来在线构建。
- 我们证明了新的图核函数和相应的索引结构可以更好地在抓取固有图相似性和对于大数据库的快速计算上平衡。

本文组织如下，我们首先在第二章回顾了一些相关领域，如哈希，索引，图核函数等等。在第三章，我们正式定义了图和图相似性搜索。在第四章我们讨论了索引结构和核函数的细节。最后我们对我们的算法进行了广泛的实验，并与现有算法进行对比，并得出了一些结论来指导我们的研究。

2 相关研究

在本章中，我们讨论一些相关的研究。从基本的图数据库索引，子图搜索，模糊子图搜索和图相似性搜索到图核函数

2.1 子图搜索

现在许多的子图搜索都采用了一个相似的框架，先把图分解为一系列小片段，然后将其作为特征，并基于其建立基于特征的索引结构来进行子图查询。属于这类的方法有 *GraphGrep,gIndex,FG-Index,Tree+Delta* 和 *GDIndex*。

在图索引中，最简单的特征就是通路 (walk)。一些情况下，我们可以用路径作为特征，就像这个领域的先驱方法 *GraphGrep*. 路径很好检索特性，并且很好处理。但是，路径的简单性却制约着这种方法的性能。举例而言，即便两图的所有路径都是不同长度的，我们利用路径也不能区分出环和链的拓扑结构。

当意识到路径的局限性后，*gIndex* 和 *FG-Index* 便应运而生了。这两种方法利用子图结构来有效分辨路径和环。但是，基于子图结构的索引也有一些限制，就是子图的枚举和匹配都是运算量非常大的过程。为了克服这些障碍，这些方法采用只提取**频繁子结构**来作为索引特征。还有些相似的方法，像 *Tree+Delta* 和 *TreePi* 则用频繁树结构来代替频繁子图结构来克服这些障碍。

GDIndex 也采用子图结构算法作为基本索引特征，但是并不使其局限于**频繁**子图结构，除此之外，这个算法还采用了哈希表来加速图同构查询。尽管这个算法设计目的是为了子图查询，但也同样支持相似查询。

2.2 模糊子图搜索

除了精确图匹配，也有些其他算法放宽了同构的限时，允许部分匹配或者模糊匹配。这是一个相对较新的领域，因此并没有太多的算法针对这个问题。有一个针对这个问题的算法，*SAGA*，它

被设计用来作为生物路径分析。首先，建立一个基于图片段的索引。当候选图与查询图失配时，我们用一个图距离度量方法来度量差异。

另一个算法叫做 *gApprox* 和 *gIndex* 相似。从思想到名字，包括作者都很相似。这种算法力求从数据库中挖掘频繁模糊模式，并以此作为索引特征。他们还提出一个概念，称为模糊频繁。

TALE 算法也是用来做模糊匹配的一种算法。但是它着重于处理有上千节点的大图。

2.3 图相似性搜索

通常，我们有很多方法去测量图之间的相似度。第一种方法是编辑距离 *(edit distance)*. 编辑距离就是我们将图 G 通过一系列操作 (增删点边，重新标号等) 变换为另一个图 G' 所需的操作数。我们可以通过给不同操作分配不同的费用，然后用费用总和当做距离来调整这个方法的准确度。虽然编辑距离是一种非常直观的图相似性测度方法，但是实际上我们难以计算它 (是个 NP-hard 问题).*C-Tree*[2] 是一种被广泛使用的图索引模型。它没有使用图的片段信息作为特征值，而是把数据图组织在一种内部节点是图闭包 *(graph closures)*, 叶节点是数据图的树形结构中。相比于前两种方法 *GraphGrep* 和 *gIndex*,*C-Tree* 的最大优势在于其支持相似性搜索，而前两个并不支持。

还有一种名为 *GString* 的子图相似性查询方法也和 *GraphGrep* 一样是用图片段作为特征值的。当然，这种方法与前面两种基于特征值的子图查询还是不同的。在这个方法中，首先我们分解复杂图为节点数较少的连通图，得到的这些连通图每个都是一个特定的图片段。随后，我们用一种标准的编号方式把数据库中的所有图都转化为一个个字符串。并用这些字符串构建一颗后缀树来支持相似性搜索。这种方法融合了子图的数据表达能力 (信息完整) 和用字符串匹配来查询图的速度 (速度快)。

另外, 最大公共子图 *(maximal common subgraph)*[3] 和图配对 *(graph alignment)*[4, 5] 这两种方法也常被用来定义图相似度。虽然有这么多方法，但是不幸的是迄今为止我们仍没有一种简单的方法来索引或者度量大图数据库。

2.4 图核函数 Graph Kernel Functions

目前业界有很多图核函数，而开创性的一个图核函数是 Haussler 在他对 R 卷积核 *(R-convolution kernel)* 的研究中提出的。现在大多数图核函数都遵循它提出的这种框架。R 卷积核是基于把离散的结构 (如图) 分解成一系列的组成元素 (子图) 这个思想的。我们可以定义许多这样的分解, 就像组成结构中的核心一样。R 卷积核保证无论选择怎样的分解方式或者结构核心，都能得要一个对称半正定的函数，或者一个结构间的核函数。这个关键性质将寻找离散结构核函数的问题简化为寻找分解方式和结构间的核函数。R 卷积核可以通过加权分解核来允许组件结构间的加权核。

目前图核函数的研究可以大致分为两类：第一类是考虑图中的所有可能组成结构（例如所有路径），然后以此来计算两图之间的相似度。这一类的算法有 product graph kernel,random walk based kernel 和基于点对间最短路径的核。第二类核函数是尝试通过一组特殊的 (有限的) 结构来

计算局部相似度，并且值在这有限的结构中统计共享的结构。这种方法包含一大类图核算法，叫做 spectrum 核，还有最近频繁子图核。我们发现最有效的核函数是 Vishwanathan 提出的用于全局相似性度量的方法。全局度量需要 $O(n^3)$ 的时间复杂度，n 是图中最大的节点数。众所周知，不同于全局相似度测量，局部相似性度量时间代价是非常昂贵的。因为子结构匹配（如子图同构）是一个 NP-hard 的问题。

我们采用一个最近提出的图小波匹配核，并将其扩展使其能在大数据库运行。

3 背景知识

在我们详细介绍算法细节之前，让我们先了解一下关于图分析计算的所需的基本背景。这章包含 (i) 图核函数，(ii) 图小波分析两部分。

3.1 图

一个**标号图**可以被一个有限的节点集合 V 和一个有限的边集合 $E \in V \times V$ 所描述。在大多数应用中，图都是有标号的。这些标号都是从一个标号集选取的，我们用一个标号函数 $\lambda : V \cup E \to \Sigma$ 来给各个节点和边分配标号。在**标号点图**中只有点有标号，同样，在**标号边图**中只有边有标号。在**全标号图**中，边点都有标号。如果用一种特殊的标号来表示未标号的点和边，那么标号点图和标号边图都可以被看做是全标号图的特殊形式。所以在此文中我们只考虑全标号图来简化问题而又不失一般性。对于标号集 Σ 我们并不指定具体结构，可以是一个字段, 一个向量，也可以是很简单的是个集合。以下我们约定，一幅图用一个四元组 $G = (V, E, \Sigma, \lambda)$ 表示,V, E, Σ, λ 都如上文所述。如果一幅图 $G = (V, E, \Sigma, \lambda)$ 和另一幅图 $G' = (V', E', \Sigma', \lambda')$ 有 1-1 映射的关系 $f : V \to V'$，那么图 G 就是 G' 的**子图**，用 $G \in G'$ 表示。1-1 映射可以有这么几种

- 对于所有 $v \in V, \lambda(v) = \lambda'(f(v))$
- 对于所有 $(u, v) \in E, (f(u), f(v)) \in E'$
- 对于所有 $(u, v) \in E, \lambda(u, v) = \lambda'(f(u), f(v))$

换言之，如果一幅图保持着另一幅图的所节点标签，边关系，和边标签，那么这副图就是另一幅图的子图。一个**通路** *(walk)* 是一个节点列表 $v_1, v_2, ..., v_n$, 对于所有 $i \in [1, n-1]$,v_i 和 v_{i+1} 有边连接。一个**路径** *(path)* 是一个不含重复节点的通路, 即对于所有 $i \neq j, v_i \neq v_j$。

3.2 再生 Hilbert 空间

对于大量图数据的分析，核函数是一个很强大的处理工具。核函数的优势在于它无需精确计算对应点对就可以把一组数据放入一个高维的 Hibert 空间。我们用一个叫做核的函数来处理这个过

程。一个二元函数 $K : X \times X \to \mathbb{R}$ 如果符合以下公式，则它是一个半正定函数。

$$\sum_{i,j=1}^{m} c_i c_j K(x_i, x_j) \geq 0 \tag{1}$$

上式中 $m \in \mathbb{N}$, 例子 $x_i \in X (i = [1, n])$, 系数集 $c_i \in \mathbb{R} (i = [1, n])$。另外，如果 $x, y \in X, K(x, y) = K(y, x)$ 那么这个二元函数就是对称的。一个对称半正定函数保证存在一个 Hilbert 空间 $\mathcal{H}$ 和一个映射 $\Phi : X \to \mathcal{H}$, 例如

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle \tag{2}$$

对于所有的 $x, x' \in X$。$\langle x, y \rangle$ 表示 x 与 y 的内积。这个结果就是我们所熟悉的 Mercer 定理。一个对称半正定函数又称为 Mercer 核函数简称为核函数。通过将数据空间变为 Hilbert 空间，核函数提供了一种对包括图在内的不同数据的统一分析环境。即使一开始的数据空间根本不像一个向量空间，也可以统一化。我们称这种归一化方法为" 核诀窍"，并将其应用在许多不同的数据分析任务上，包括分类，回归，通过准则分析的特征提取等。

3.3 图小波分析

小波函数常被当做一种用于将一个函数或者信号通过不同的变换分解和表示为它的子结构的方法。小波常用在数字化的数据上，例如通信信号或者数学函数，也可以用在一些规律的数值结构上，像矩阵和图像。但是图是一种随意的结构，并且可能表示非数值化的关系，在数据元素之间也可能存在拓扑关系。近期的一些研究证实了利用小波函数对图做多解析度分析的可行性。两个小波函数的典型就是 $Haar$ 和 $Mexicanhat$。Crovella 和其他人开发了一种对于交通网络数据分析的多尺度方法。在这个应用中，他们尝试去确定一个交通现象发生的规模。他们将交通网络表示为一个标号图，并且用一些测量方法像每个单位时间的比特携带数来做标号。Maggioni 和其同事证实了一个通用的双正小波分析可用于图分析。在他们的方法中，他们将利用扩散操作的二元性来激发多解析度分析。他们的方法主要适用于大空间的分类，例如流形和图, 对于带属性的分子结构的适应性尚不清楚。在这里主要的技术问题就是如何在多解析度分析中包含节点标签。

4 基于哈希的快速图相似性搜索

正如之前所说，现在的图查询算法查询时间很快但是没有好的相似性度量方法。核函数可以提供一个好的相似性度量方法，但是核函数的矩阵运算需要大量的时间，所以我们很难直接利用其来建立索引。为了解决这个问题，我们提出了一种新的算法，G-Hash。现在的算法常常关注速度或精度的一种, 而 G-Hash 在两者效果均很好。我们利用小波图匹配核 (WA) 来定义相似性，并用 Hash 表作为索引结构来加速图相似性查询。下面我们先介绍下 WA 方法。

4.1 小波图匹配核简介

WA 方法的思想是先通过压缩每个节点周围邻接节点的属性信息，然后应用非递归线性核去计算图之间的相似度。这个方法包含两个重要的概念：*h-hop* 邻域和离散小波变换。用 $N_h(v)$ 标记一个节点 v 的 h 跳邻域，代表一个距离节点 v 最短距离是 h 跳 (跳过 h 个节点, 也就是 h+1 的曼哈顿距离) 的节点集合。离散小波变换涉及到一个小波函数的定义，见下文公式(3) , 也用到了 h 跳邻域。

$$\psi_{j,k} = \frac{1}{h+1} \int_{j/(k+1)}^{(j+1)/(k+1)} \varphi(x)dx \tag{3}$$

$\varphi(x)$ 代表 *Haar* 或者 *Mexican Hat* 小波函数，h 是在将 $\varphi(x)$ 在 $[0,1)$ 区间上分成 $h+1$ 个间隔后的第 h 个间隔，$j \in [0,h]$。基于以上两个定义，我们现在可以将小波分析用在图上了。我们用小波函数来计算每个节点的局部拓扑和。公式(4)展示了一个小波度量方法，记做 $\Gamma_h(v)$，以图 G 中的一个节点 v 为例。

$$\Gamma_h(v) = C_{h,v} \times \sum_{j=0}^{k} \psi_{j,k} \times \bar{f}_j(v) \tag{4}$$

其中,$C_{h,v}$ 是一个归一化因子

$$C_{h,v} = (\sum_{j=0}^{h} \frac{\psi_{j,h}^2}{|N_h(v)|})^{-1/2}, \tag{5}$$

$\bar{f}_j(v)$ 是离节点 v 最远距离为 j 的原子特征向量的平均值

$$\bar{f}_j(v) = \frac{1}{|N_j(v)|} \sum_{u \in N_j(v)} f_u \tag{6}$$

f_u 表示节点 v 的特征向量值。这样的特征向量值只会是下面四种中的一种：定类，定序，定距，定比。对于定比和定距特征值，直接在上述的小波分析时代入其值即可得到局部特征值。对于定类和定序节点特征，我们首先建立一个直方图，然后用小波分析提取出特征值。在节点 v 分析完成后，我们可以得到一个节点列表 $\Gamma^h(v) = \{\Gamma_1(v), \Gamma_2(v), ..., \Gamma_h(v)\}$, 我们称其为小波测度矩阵。用此方法我们可以将一个图转换为一个节点向量集合。因为小波变换有明确的正负区域，所以这些小波压缩特征可以表示出局部的邻接节点和距离较远的邻接节点的差异。因此，通过小波变换，一幅图的结构化信息可以压缩成节点特征。从而我们可以忽略拓扑结构来专心于节点匹配。核函数就是建立在这些集合上的，我们以图 G 和 G' 为例，图匹配核函数是这样的

$$k_m(G, G') = \sum_{(u,v) \in V(G) \times V(G')} K(\Gamma^h(u), \Gamma^h(v)), \tag{7}$$

$$K(X, Y) = e^{\frac{-\|X-Y\|_2^2}{2}}. \tag{8}$$

就像在后文实验部分我们验证的一样，WA 方法展示了一种很好的利用核函数进行图相似度定义的方法。但是这个方法有一个问题，就是小波匹配核的总时间复杂度是 $O(m^2)$，核矩阵的是

$O(n^2 \times m^2)$，n 是数据库图个数，m 是平均节点个数。这意味着，当数据库尺寸增加时，计算时间将大幅度增加。

4.2 Hash 函数实现快速图相似性搜索

根据将图中的每个节点利用函数映射到一个特征空间上的这一思想，我们可以设计出一种核函数来进行快速相似性搜索。具体的说，我们有了以下两条发现。

- 当图 G 中节点 u 向量和图 G' 中的节点 v 完全不同，节点 u 和 v 的核值很小基本不影响图核值。所以我们如果仅计算相似点对的核值，得到的核矩阵将和 WA 方法中的相似性度量类似，但耗时很少。
- 如果基于节点向量来构造哈希表的话，相似的对象的位置上将也很靠近。所以我们可以很快速得利用哈希表来寻找相似点对。另外，如果数据库中所有图都存入哈希表中的话，一个位置可能对应不同图中的多个相似节点。因为这些节点都是相似的，所以只要利用其中两个节点进行一次 RBF 核运算就可以代表所有的结果了。节点覆盖给了我们另一个省时间的契机。

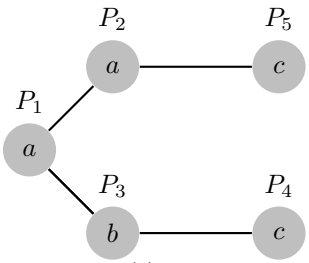
基于以上两条发现，我们引入我们的方法:k-NNs 查询 (G-hash).G-Hash 是一种利用 WA 方法进行精确相似度度量，并利用哈希来降低时间复杂度。具体算法见下文。

4.3 索引构建

像 WA 方法一样，我们首先需要利用小波变换将数据库中的所有图拆解成点向量的形式。因为 WA 方法对距离参数的选取并不敏感，用不同的小波算法造成的差异也并不大 [6] ，所以我们设定 h 为 2，并才用 *Haar* 小波算法来简化算法

当图完全拆解成节点向量后，我们就可以利用这些向量构建哈希表。这时，我们需要一个哈希函数来把相似的节点哈希到相同的位置。这就意味着我们要基于节点向量构建哈希键。我们可以用和构造节点向量同样的思想构建哈希键，即利用节点标号和邻接信息来构造。我们将每一个节点向量中的特征信息离散成一个整数，并将每个节点标签直接编码为一个 n 位的字符串（除去一个代表本身标号的 1，其他全 0)。其他特征就近似到最近的整数上。当节点向量变成了一个整数列表后，我们再把节点向量变为一个字符串 (用二进制表示每个整数，并用下划线连接)。这样的一个字符串就是对于节点的哈希键。我们以图表 1 的图 P 来举例说明。

例子 4.1. 我们选取节点标签和邻接节点各个标号的个数作为度量特征。所以我们一共有四个特征。这个图的特征仅属于一种类型: 定类。我们首先写出节点特征的柱形图 *(每个数字就是一个高度)* 如图表 *1* 中的 *(B)*。然后用小波函数提取出实际特征值。举例而言，对于 P_3 在用 $h = 0$ 的小波分析提取后，局部特征是 $[b, 2, 0, 1]$。然后生成的哈希表如图表 *1(C)* 所示。



(a) P

Nodes	Label	#a	#b	#c
P_1	a	1	1	0
P_2	a	1	1	1
P_3	b	2	0	1
P_4	c	0	1	0
P_5	c	1	0	0

(b) 节点特征矩阵

哈希键	哈希值
a,1,1,0	P_1
a,1,1,1	P_2
b,2,0,1	P_3
c,0,1,0	P_4
a,1,1,1	P_5

(c) 图 P 的哈希表

图 1: 一幅简单图

注意: 在这个算法框架中，我们在构建哈希索引时没有对查询图做任何假设。

4.4 *K-NNs* 查询过程

为了获得对于给定图的 *K-NNs*, 我们需要计算它和数据库中其他图的距离。以下是我们定义的利用核函数进行两图距离度量的函数

$$\begin{aligned} d(G, G') &= \sqrt{\|\phi(G) - \phi(G')\|_2^2} \\ &= \sqrt{\langle \phi(G) - \phi(G'), \phi(G) - \phi(G') \rangle} \\ &= \sqrt{\langle \phi(G), \phi(G) \rangle + \langle \phi(G'), \phi(G') \rangle - 2\langle \phi(G), \phi(G') \rangle} \\ &= \sqrt{k_m(G, G) + k_m(G', G') - 2k_m(G, G')} \end{aligned} \tag{9}$$

公式中 $k_m(G, G)$ 代表图 G 和其本身的核函数值, $k_m(G', G')$ 是图 G' 及其本身的值, $k_m(G, G')$ 就是图 G 和 G' 的。在后文中，我们会介绍该如何计算这些值。

尽管在将查询图节点哈希到哈希表时，我们可以得到核函数

$$k_m(G, G') = \sum_{v \in G', u \in simi(v)} K(\Gamma^h(u), \Gamma^h(v)) \tag{10}$$

$simi(v)$ 是一个包含着图 G 和节点 v 哈希到同一个位置的节点集合。我们用以下的解码方式来获取包含这些节点的图号和节点号。

显然，仅利用相似点对而非所有点对来计算两图相似度可以节约很多运算时间。因此为了增加准确度，相似的节点应该被哈希到相邻的位置。在图很大 (如大于 40) 时，我们也要计算相邻位置的节点。

因此在核函数计算时我们只考虑相似点对，在使用 RBF 核的情况下，$K(\Gamma^h(u), \Gamma^h(v)) \approx 1$，所以公式 (2) 可以写成

$$K(G, G') \approx \sum_{v \in G', u \in simi(v)} 1 = \sum_{v \in G'} |simi(v)| \tag{11}$$

$|simi(v)|$ 是在 $simi(v)$ 中的节点数目。这意味着我们只需要统计图 G 中和查询 G' 相似的点个数，其和就是我们要求的核。同理，我们可以这样计算每个图与其自己的核。

在完成上述操作后，我们会得到一个距离向量，其每一个值都对应这一个数据库中的图与查询图的距离，通过对这个距离向量排序，我们就可以得到对于给定的查询的 $K - NNs$。

4.5 动态插入与删除

如果要向数据库中插入一副图，我们只需将新图的节点进行哈希，然后加入节点哈希表。这样插入完成后，只有图中所有的节点的位置会发生变化。而且，因为一幅图的节点是有限的，所以插入操作的时间和构建索引的内存也是有限的。删除操作和插入类似，我们只需找到图中节点在哈希表中的对应哈希值，然后删掉即可。

参考文献

[1] Xiaohong Wang, Aaron Smalter, Jun Huan, and Gerald H. Lushington. G-hash: Towards fast kernel-based similarity search in large graph databases. *EDBT 2009, March 24–26*, pages 472–480, 2009.

[2] H. He and A. K. Singh. Closure-tree: an index structure for graph queries. *Proc. International Conference on Data Engineering'06 (ICDE)*, 2006.

[3] T. Jiang Y. Cao and T. Girke. A maximum common substructure-based algorithm for searching and predicting drug-like compounds. *Bioinformatics, 24(13)*, 2008.

[4] H. Fro¨hlich, J. K. Wegner, F. Sieker, and A. Zell. Optimal assignment kernels for attributed molecular graphs. *Proceedings of the 22nd international conference on Machine learning*, 2005.

[5] J.-P. Vert. The optimal assignment kernel is not positive definite. *Technical Report HAL-00218278, French Center for Computational Biology*, 2008.

[6] A. Smalter, J. Huan, and G. Lushington. Graph wavelet alignment kernels for drug virtual screening. *Proceedings of the 7th Annual International Conference on Computational Systems Bioinformatics*, 2008.