前　言

作为一种通用的数据结构，图可以用来表示数据对象之间的各种复杂关系。例如: 图可以表示化合物的分子结构、蛋白质交互网络、社会网络、Web 结构图等。随着科学与工程领域中图数据的大量出现，从图数据库中发现有用的知识已成为数据挖掘领域一项重要的研究课题，图模式挖掘是其中最重要的一个研究分支，因为与图有关的绝大部分应用 (例如: 图查询、图分类、图聚类等) 都需要利用图模式来管理、查询和分析图数据。本书主要对图模式挖掘技术进行深入研究，归纳总结了现有研究成果的主要思想和优缺点，提出了一些新的图模式挖掘问题和解决方法，主要研究成果如下:

第一，提出从图数据库中挖掘代表模式问题及其有效的解决方法。目前的频繁子图挖掘算法通常会产生大量的甚至指数级数量的频繁子图，严重地影响了挖掘结果的可用性。挖掘代表模式既可以极大地减少图模式的输出数量，又能使有意义的图模式保留在挖掘结果中。本书给出了挖掘代表模式问题的形式化定义，并证明了该问题是 NP-hard 问题; 提出了一系列新的概念: δ-覆盖图、跳跃值、δ-跳跃模式等; 发现了 δ-跳跃模式的一个重要性质: δ-跳跃模式一定是代表模式; 利用 δ-跳跃模式的性质，提出了挖掘代表模式的三个算法: RP-FP，RP-GD，RP-Leap。RP-FP 和 RP-GD 可以挖掘完整的代表模式集合，RP-Leap 可以挖掘近似的代表模式集合。RP-FP 从频繁闭图模式中计算代表模式，具有紧的近似比保证。然而，当频繁闭图模式数量较大时，RP-FP 效率较低。RP-GD 采用联机算法的思想，直接从图数据库中挖掘代表模式。算法复杂性分析表明，RP-GD 的效率要远远高于 RP-FP 的效率。RP-Leap 利用图模式搜索空间中大量分枝之间的相似性，能够快速跳过那些几乎不产生代表模式的分枝，从而挖掘一个近似代表模式集合。实验结果表明: (1) RP-FP，RP-GD，RP-Leap 能得到一个小的而有意义的代表模式集合; (2) RP-GD 的挖掘效率远远高于 RP-FP 的挖掘效率，而在结果质量方面，RP-GD 类似于 RP-FP; (3) RP-Leap 以丢失少量代表模式为代价，取得了比 RP-GD 快一个数量级的性能改善。

第二，提出从图数据库中挖掘核心子结构问题及其有效的解决方法。核心子结构在真实的图数据库中大量存在，例如，化合物中的功能团就是一类核心子结构。针对核心子结构的特征，本书给出了核心子结构的形式化定义，称为 Δ-跳跃模式。人们发现了 Δ-跳跃模式的很多重要性质，例如 Δ-跳跃模式是稳定的，它们对噪声和数据的变化不敏感，Δ 值越大，它们的抗干扰能力越强。然而，Δ-跳跃模式不具有反单调性

质，挖掘它们非常具有挑战性。本书通过仔细研究跳跃模式自身的特性，提出了两种新的裁剪技术，即基于内扩展的裁剪和基于外扩展的裁剪。利用这两种裁剪技术，设计了一个高效的跳跃模式挖掘算法 GraphJP。在理论上，严格地证明了这两种裁剪技术的正确性及算法 GraphJP 的正确性。实验结果表明: 这两种新的裁剪技术能有效地裁剪图模式搜索空间，算法 GraphJP 能高效地、可扩展地挖掘频繁跳跃模式，而且挖掘结果中含有图数据库中的核心子结构。

第三，提出基于联合意义度量的 Top-K 图模式挖掘问题及其有效的解决方法。传统 Top-K 挖掘并不考虑图模式之间的相关性，输出的 Top-K 模式在结构上非常相似。如果用户得到其中一个图模式，那么就对其他图模式失去了兴趣。联合意义度量的作用域是图模式集合而不是图模式。因此，基于联合意义度量的 Top-K 挖掘，隐含排斥相关的图模式，可以得到一个多样化且有意义的图模式集合。本书讨论了适用于图模式集合的联合意义度量，并利用信息论中的概念 (联合熵和信息增益) 给出了两个具体的问题定义 MES 和 MIGS，证明了它们是 NP-hard 问题；提出了两个高效的 Top-K 挖掘算法 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 先产生频繁图模式，然后增量贪心地选择 K 个图模式。如果用户给定的意义度量满足 submodular 性质，那么 Greedy-TopK 能提供近似比保证。为了进一步提高 Greedy-TopK 的效率，针对 MES 和 MIGS 这两个具体问题的意义度量又设计了一系列有效的裁剪技术，将其嵌入频繁子图挖掘框架中，从而帮助裁剪图模式搜索空间。然而，当频繁图模式数量较多时，Greedy-TopK 仍然效率低、可扩展性差。为了克服 Greedy-TopK 的缺点，Cluster-TopK 先从图数据库中挖掘所有频繁图模式的一个代表模式集合，然后从代表模式中增量贪心地选择 K 个图模式。Cluster-TopK 最大的优点是无需产生频繁图模式就能快速地从图数据库中挖掘一个代表模式集合。本书从理论上证明了 Cluster-TopK 产生的解和 Greedy-TopK 产生的解非常接近。实验结果表明: 在结果质量和可用性方面，本书提出的 Top-K 挖掘远远优于传统的 Top-K 挖掘。Cluster-TopK 比 Greedy-TopK 要快一到两个数量级，而且 Cluster-TopK 的挖掘结果的质量非常接近于 Greedy-TopK 的挖掘结果的质量。

第四，提出了一种基于频繁闭显露模式的图分类框架 CEP。CEP 包括三个主要步骤: (1) 挖掘频繁闭图模式；(2) 过滤非显露模式；(3) 构造分类规则。第一步，CEP 挖掘所有频繁闭图模式作为候选分类特征。第二步，CEP 保留频繁闭图模式中的显露模式。该步需要计算图模式在不同类别数据库中的支持度，涉及大量子图同构测试。为改善 CEP 的效率，CEP 将频繁闭图模式组织成一个树型结构 T。对数据库中的每个图 G，采用深度优先方式遍历树 T。在遍历过程中，利用 Apriori 性质 (反单调性) 进行裁剪，如果 G 不包含结点 P，那么 G 也不可能包含 P 的孩子结点。通过这种方式，可以极大地减少子图同构测试次数。第三步，CEP 根据剩余的显露模式构造分类规则。在构造分类规则时，提出了一个新的特征选择方法，该方法既考虑显露模式如

何覆盖图数据，又考虑显露模式在图数据中如何分布。实验结果表明: CEP 具有良好的分类性能，而且，可以使领域专家很容易地理解和利用 CEP 产生的分类规则。

本著作共 20.3 万字，其中刘勇老师撰写了 19 万字，刘猛老师撰写了 1.3 万字。

作　者

2022 年 6 月

目　录

第 1 章　绪　　论

1.1　研究的目的和意义

随着计算机硬件与软件技术的飞速发展，尤其是数据库技术与应用的日益普及，人类积累了大量数据。据统计，全球的数据量大约每 20 个月翻一番 [4]。而且，由于计算机应用领域的不断扩大，数据类型变得越来越复杂，结构也越来越多样化。应用系统正面临着处理海量数据和数据多样性的巨大挑战。

数据量的急剧膨胀并没有给使用者在知识获取方面带来理想中的便捷，从而出现了“数据丰富，知识匮乏”的现象。为有效地解决这个问题，从 20 世纪 90 年代开始，数据挖掘技术逐步发展起来。数据挖掘，也称知识发现（Knowledge Discovery from Dataset，简记 KDD），被描述为从数据中抽取出隐含的、具有潜在用途的、人类可理解的模式 [125]。通过数据挖掘发现有用的新概念和新规律，从而辅助人们进行科学分析和决策。自从数据挖掘概念被提出以来，就引起了学术界和工业界的极大关注。在过去 10 多年的时间里，学术界提出了一系列数据挖掘理论、技术和方法，工业界也针对各种应用领域开发了大量的数据挖掘产品。目前，数据挖掘技术已经被广泛地应用到商业、金融、互联网、信息检索等诸多领域，并取得了巨大的成功。有些浏览器就成功应用了 PageRank [23] 技术。PageRank 技术是根据网页之间的链接关系来评价网页的重要程度，与单纯评价网页内容相比，可以获得更高的准确率。该技术已在搜索引擎方面得到了广泛的应用，对该领域影响深远。

数据挖掘技术在这些传统领域的成功应用也激发了人们将数据挖掘技术应用到其他领域的兴趣。人们正试图将数据挖掘技术应用的触角延伸到科学与工程领域，如计算化学、生物信息学、流体动力学等。然而，科学与工程领域中使用的数据与商业领域中使用的数据在本质上有着非常大的差异。科学与工程领域中使用的数据通常具有拓扑结构、几何结构等特点。传统的数据挖掘算法都假定数据被表示成事务集合或者多维向量集合，这使得传统的数据挖掘算法无法直接应用到科学与工程领域。如果在科学与工程领域中使用传统的挖掘算法，需要将数据表示成事务集合或者多维向量集合。在某些情况下，这种转换容易实现，但在绝大多数情况下，这种转换会导致丢失大量有用的信息或根本不可能实现。因此，传统的数据挖掘算法不适合挖掘这些具有拓扑结构的数据。研究者们需要设计新算法来挖掘这些具有拓扑结构的数据，以满足人们逐渐提升的新需求。

图是离散数学和计算机科学中最一般的数据结构，它可以准确地表示科学与工程领域中数据的内部结构和关键特征。例如: 在化学领域，我们可以使用无向标号图表示化合物的拓扑结构。图中的结点表示化合物的原子，结点的标号表示原子的类型 (如 C，N，O 等)。图中的边表示化合物的内部化学键，边的标号表示化学键的类型 (如单键、双键、苯环键等)。在计算生物学领域，我们可以使用图结构表示各种网络 (如蛋白质交互网络、基因相关网络等)。此外，社会关系网络、引用网络、传感器网络、万维网等都可建模为大的图结构数据。

挖掘和管理图数据可以很自然地解决科学与工程领域中的很多关键问题。例如，在药物设计方面，研究人员经常需要在大量的化合物集合中寻找出对病毒具有强烈抑制作用的化学分子结构，以便进一步检测并以此合成新的药物。使用频繁子图挖掘算法可以很容易地解决该问题。首先将实验中对病毒具有强烈抑制作用的化合物集合转化成一个图的集合，然后使用频繁子图挖掘算法挖掘该图集合中的频繁子结构，再结合领域知识对这些频繁子结构进行分析，就可以找出对病毒起抑制作用的化学分子结构。在计算生物学领域，可以利用蛋白质二级结构和蛋白质三级结构预测蛋白质功能。蛋白质的氨基酸序列通过折叠可以形成蛋白质二级结构和蛋白质三级结构。蛋白质二级结构可以用平面标号图来表示，而蛋白质三级结构可以用带有几何信息的空间几何图来表示。通过在蛋白质二级结构和蛋白质三级结构上建立分类预测模型，可以帮助生物学家分析和识别蛋白质的基本功能。在计算机视觉领域，图可以表示图像中实体的组织等复杂关系，子图查询操作可以帮助识别对象和场景。此外，万维网、社会关系网络、传感器网络等领域中的很多问题也都可以利用图挖掘的方法来解决。因此，用图来建模科学与工程领域中具有复杂结构的数据，用基于图的挖掘方法解决一般性的挖掘问题 [38]，将具有广泛的应用背景和重要的现实意义。

尽管与其他数据结构相比，图能够表达更加丰富的语义，在科学研究和工程领域有着更为广泛的应用，但是这种丰富的语义和复杂的内部结构增加了各种挖掘算法的难度，传统的数据挖掘方法难以适用。具体来说，传统数据挖掘算法中的相等、包含等简单操作在图的环境中被映射成图同构、子图同构等复杂操作。图同构问题目前还没有多项式时间算法，子图同构问题已被证明是 NP 完全问题。图的很多计算问题 (例如: 最大公共子图计算、结点最大覆盖等) 也都是 NP 完全问题，具有很高的计算复杂性 [133]。加之目前对图挖掘方面的研究起步不久，相对于广泛的应用前景，取得的成果还很少。因此，设计有效的图挖掘方法是一个值得开展并且具有挑战性的研究课题。

图挖掘所涉及的各方面研究工作目前正在展开，图模式挖掘是其中最基本的也是最重要的一个研究分支。因为图模式几乎在所有图挖掘和图数据管理问题中都有广泛的应用价值。例如:(1) 通过分析挖掘出来的图模式，领域专家可以获取图数据库中有用的信息；(2) 在图查询 [147] 和相似性搜索 [149] 方面，利用图模式作为索引特征建立

有效的索引结构，可以明显地提高查询处理效率；(3) 在图分类 [43] 方面，利用图模式作为分类特征建立有效的分类模型，可以提高分类准确率。此外，在图聚类、复杂网络演化规律预测等方面，挖掘算法也需要利用图模式来进行聚类和预测。

可见，在图挖掘和图数据管理方面，图模式挖掘有着极其重要的价值。然而，该领域的研究目前还存在一些问题和障碍，尚未得到解决。例如: 频繁子图挖掘算法产生的频繁子图模式数量通常太大，得到的挖掘结果很难被利用，甚至当支持度过低时，频繁子图挖掘算法不能在合理的时间内完成挖掘任务。再例如: 目前的图模式挖掘算法基本上只使用支持度作为图模式意义的一种度量，在不同的应用中，用户可能需要不同的意义度量，如何根据不同的意义度量设计高效的图模式挖掘算法也是需要研究的重要问题。本书从这些问题入手，主要针对图模式挖掘算法展开研究。

1.2 国内外研究现状

图挖掘和图数据管理是一个新兴的研究领域 [60,132]，从 2002 年开始引起了学术界和工业界的普遍关注，研究人员开始了相关的研究工作。在短短几年内，该领域的研究工作发展迅速，至今已成为数据挖掘领域的一个热点研究内容。这可以从近年来顶级的国际学术会议 (SIGMOD, VLDB, KDD, ICDE, ICDM, SDM, ICML, AAAI 等) 收录的论文情况反映出来。此外，与图挖掘有关的各种专题研讨会也在陆续开展，例如:International Workshop on Mining and Learning with Graphs(MLG)，ICDM Workshop on Mining Graphs and Complex Structures(MGCS) 等。

国外也有许多专门的研究组从事图挖掘和图数据管理方面的研究。比较著名的研究组有美国 CMU 大学 Christos Faloutsos 教授领导的数据库小组、UIUC 大学 Jiawei Han 教授领导的数据库小组、华盛顿大学 Diane J. Cook 领导的研究小组、IBM 实验室 Philip S. Yu 领导的研究小组等。我国的研究人员也在图挖掘和图数据管理方面开展了相关研究工作，如香港科技大学 Wilfred Ng 教授领导的研究小组、香港中文大学于旭教授领导的研究小组、清华大学周立柱教授领导的研究小组等。

目前，国内外在图挖掘和图数据管理方面的研究工作主要包括以下几个方面。第一，图模式挖掘。第二，图数据的查询。第三，图数据的分类。第四，其他与图挖掘有关的研究工作。下面将详细讨论以上各方面研究的进展情况。

1.2.1 图模式挖掘

1.2.1.1 频繁子图挖掘

频繁子图挖掘作为图挖掘领域中最重要的研究内容很早就被提出了，在国际顶级会议和期刊上都有大量的相关文章。频繁子图挖掘的结果可以用于其他的图挖掘任务。

例如: 作为分类特征建立分类模型，作为索引特征支持查询处理。频繁子图挖掘有两个不同的问题模型，一个是图集合模型，另一个是单图模型。

1.2.1.1.1 图集合模型

在图集合模型中，输入是一个图的集合，该集合中每个图的规模相对较小。频繁子图模式挖掘的目的就是找到该图集合中频繁出现的子图。其具体定义如下: 给定一个图的集合 D 和一个最小支持度 min_sup ($0 \leqslant$min_sup$\leqslant 1$)，要求发现至少在 min_sup$\times|D|$ 个图中出现的所有连通子图。这种模型的应用领域包括化学、生物信息学、XML 数据库等。这种模型又可根据不同的图类别分为下面几种情况:

(1) 标号图

作为一种通用的数据结构，标号图可以用来表示数据内部的各种复杂关系。将一个标号图 G 定义为一个四元组 $G = \{V, E, \Sigma, L\}$，其中，V 代表图中顶点的集合，$E = V \times V$ 代表图中边的集合，Σ 代表顶点标号与边标号的集合，L 是标号函数，用来对顶点和边分配标号，即 $L : V \cup E \to \Sigma$. 标号是一个依赖于具体应用的属性。例如: 在化学领域中，结点的标号可以表示原子类型 (如 C，H，O 等元素)，边的标号可以表示化学键的类型 (如单键、双键、苯环键等)。在标号图中，多个结点或边可以具有相同的标号，例如: 一个化合物的分子中包含多个相同类型的原子和多个相同类型的化学键。目前，所有的研究都假定挖掘出来的图模式必须是连通的。这是因为连通性可以很好地刻画对象 (图) 内实体 (结点) 之间的复杂关系 (边)，而且，将图模式限制为连通图也可极大地减小挖掘问题的复杂性。

针对标号图集合上的频繁子图挖掘问题，已经提出了大量的挖掘算法。按照所采用的技术，这些算法大致可以分成三类:

(a) 贪心搜索方法

贪心搜索方法通过使用各种启发式策略来减小搜索空间，SUBDUE 方法 [61] 是其中最有代表性的方法。SUBDUE 基于最短描述长度压缩 (minimum description length-based compression) 的启发式策略来评价子图的重要性，使用 Beam Search 搜索策略来遍历图模式空间。尽管贪心搜索方法的效率较高，并且适用于单图环境，但该方法不能保证挖掘结果的完整性，可能会丢失某些重要的图模式。

(b) 归纳逻辑编程方法

归纳逻辑编程方法将图数据表示成一系列 Horn 子句，使用归纳逻辑编程系统进行推理 [116]。该类方法的优点是能保证挖掘结果的完整性，并且可以与背景知识相结合。然而，该类方法通常具有极高的计算复杂性，无法应用于数据量较大的环境 [80]。

(c) 图论方法

图论方法目前被广泛研究，近几年，文献中出现的频繁子图挖掘算法都属于此类

方法。图论方法是使用图论中的数学概念来表示和评价子图，通过枚举候选子图和裁剪模式搜索空间等技术来挖掘所有频繁子图。这类方法既可以保证挖掘结果的完整性，又比贪心搜索方法和归纳逻辑编程方法具有更高的挖掘效率。根据图模式搜索空间遍历方式的不同，图论方法又可以分为以下两类:

①基于 Apriori 方法 (也称宽度优先搜索方法)

基于 Apriori 方法利用支持度的反单调性质裁剪候选频繁子图。支持度的反单调性质定义如下: 如果一个子图 P 是频繁的，那么 P 的所有子图也是频繁的；如果一个子图 P 是不频繁的，那么 P 的所有超图也是不频繁的。基于 Apriori 方法宽度优先遍历模式搜索空间，在得到大小为 K 的所有频繁子图之后，连接产生大小为 $K+1$ 的候选频繁子图，然后利用支持度的反单调性质删除候选子图中的非频繁子图，最后扫描数据库，确定剩余候选子图的支持度，得到大小为 $K+1$ 的所有频繁子图。过程迭代进行，直到发现全部频繁子图为止。此类方法的代表算法包括 AGM [68–70]，FSG [83,85]，Path-Join [45] 等。它们之间的区别在于: AGM 采用一次增加一个结点的方式来扩展子图模式，FSG 采用一次增加一条边的方式来扩展子图模式，而 Path-Join 采用路径作为子图模式的扩展单元。基于 Apriori 方法的优点是结构清晰、容易实现。但此类方法由于要逐层枚举候选频繁子图，因此会产生大量的中间结果，消耗过多的存储空间。而且，为确定每个候选子图的支持度需要进行大量子图同构测试，从而造成很多不必要的计算。

②模式增长方法 (也称深度优先搜索方法)

模式增长 (pattern-growth) 方法克服了上述 Apriori 方法的缺点，采用深度优先方式遍历模式搜索空间，此类方法通常的策略是，在某个频繁子图 p 的基础上，扩展产生 p 的孩子 (p 的超图模式) 并计算它们的支持度，对 p 的每个频繁孩子，以深度优先方式继续扩展，直到发现全部频繁子图为止。此类方法的代表算法包括 MoFa [20]，gSpan [145]，FFSM [66]，GASTON [103]，MoSS [21] 等。与基于 Apriori 方法相比，模式增长方法具有如下优点: 由于避免重复产生大量的中间结果，因而有更高的内存利用率；在扩展某个频繁子图 p 时，可以同时计算 p 的孩子的支持度，因而有更高的挖掘效率。

模式增长方法是目前最好的频繁子图挖掘方法。在此类方法的众多挖掘算法中，gSpan [145] 算法在挖掘效率、内存利用率、可扩展性等多个性能指标方面都非常突出，是目前最有效的频繁子图挖掘算法之一 [136]。

为进一步提高挖掘效率，研究人员又提出了许多优化策略，主要包括: 利用并行技术提高挖掘效率 [24]；利用数据库的索引技术改善挖掘大型数据库时的效率 [128]；利用嵌入列表 (embedding list) 技术减少子图同构测试次数 [66,83]；利用图划分的分治策

略，减少子图同构时子图的规模，以及支持数据库动态更新时挖掘频繁子图 [129]；为图模式设计规范化形式，避免了同一个图模式在挖掘过程被多次枚举 [66,69,83,145]；通过减少最右扩展的候选子图数量来提高 gSpan 的挖掘效率 [11]。

此外，研究人员还针对具体的应用背景以及特殊的图类型，提出了一些有针对性的频繁子图挖掘算法。文献 [35] [114] [121] [159] 给出了频繁子树挖掘算法。文献 [76] 给出了一种频繁拓扑子图的挖掘算法。文献 [64] 给出了一种在外平面图上挖掘频繁子图的算法，并给出一个多项式时间内的子图同构算法。

(2) 几何图

如果标号图中的每个结点带有二维或三维空间坐标信息，该图被称为几何图。从几何图数据中挖掘频繁子图，需要给出几何同构的精确定义，几何同构除了要保证拓扑结构相同、边和结点标号匹配，还要保证结点坐标相同。在实际应用中，结点坐标依赖于给定的坐标轴，因此在确定几何同构时还应允许对几何图进行几何转换。常用的三种几何转换分别是旋转、平移、拉伸。针对具体的应用背景，应由用户指定允许的几何转换组合或者由用户提供的新的几何转换。几何转换影响几何同构的定义。例如，在图 **??** 中，如果在几何同构的定义中允许旋转、平移、拉伸这三种几何转换，那么 r_1 几何同构于 r_2，r_3 和 r_4；如果在几何同构的定义中允许旋转、平移这两种几何转换，那么 r_1 几何同构于 r_3 和 r_4；如果在几何同构的定义中只允许平移这一种几何转换，那么 r_1 仅与 r_4 几何同构。

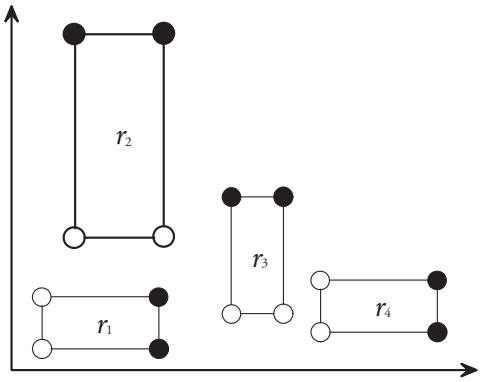


图 1　几何图同构

在实际应用中，由于存在测量误差和计算误差，在几何同构时，不可能保证结点坐标之间完全匹配。因此，几何同构时，我们应允许结点坐标之间存在小的误差。给定两个几何图 G_1 和 G_2，设 Φ 是 G_1 和 G_2 之间满足标号图同构定义的双射函数，T 是允许的几何转换集合，r 是结点坐标之间允许的最大误差，如果对 G_1 中任意结点 v，都有 $|T(c(v)) - c(\Phi(v))| \leqslant r$(其中 $c(v)$ 表示 v 的坐标)，那么称 G_1 r-误差几何同构于 G_2。几何图中频繁子图挖掘问题的定义如下: 给定一个几何图数据库 D，一个最小支持度阀值 min_sup($0 \leqslant$ min_sup $\leqslant 1$)，一个允许的几何转换操作集合 T，以及一个

结点坐标之间允许的最大误差 r，挖掘至少在 min_sup×$|D|$ 个图中出现的所有 r-误差几何连通子图。

文献 [84] 给出了一个从几何图数据中挖掘频繁子图的算法 gFsg。gFsg 允许旋转、平移、拉伸这三种几何转换以及它们的组合。为了提高挖掘效率，gFsg 利用图签名加速几何同构的计算过程。图签名是一种几何转换不变量，例如: 边之间的夹角在几何转换之后仍保持不变。gFsg 在判断两个几何图 G_1 和 G_2 是否存在几何同构之前，先计算 G_1 和 G_2 的图签名，如果它们的图签名不匹配，那么说明 G_1 和 G_2 之间一定不存在几何同构。通过这种方式，gFsg 极大地提高了几何同构的计算效率。

(3) 关系图

关系图本质上是标号图的一种特例。与标号图相比，关系图有三个不同的特征: (a) 关系图中任意两个结点的标号都不相同，因为每个结点表示一个不同的对象；(b) 关系图通常很大，可能包含几千个结点和几百万条边；(c) 为了使挖掘出来的频繁子图有实际意义，通常要求频繁子图还必须满足一些特定的约束，例如，要求频繁子图有高连通性。关系图挖掘的应用领域包括生物网络、社会网络、传感器网络等各种大规模网络。

因为关系图是标号图的一种特例，所以可以使用标号图的频繁子图挖掘算法来挖掘关系图。然而，利用关系图自身的特征，可以设计更加高效的挖掘算法。例如: 由于关系图中任意两个结点的标号都不相同，因此我们可以在多项式时间内解决子图同构问题。文献 [150] 给出了两个从关系图中挖掘高连通性的频繁闭图模式算法 CLOSECUT 和 SPLAT。CLOSECUT 采用模式增长方法，而 SPLAT 采用模式归约方法。

(4) 不确定图

由于测量仪器的局限性、数据噪声等原因，不确定性在实际应用中普遍存在，在很多图数据中也是如此。例如: 计算生物学中的蛋白质交互网络就是一种不确定图，其中的结点表示蛋白质，边表示蛋白质之间的交互 (PPI)。由于实验检测方法的局限性，很多 PPI 是不确定的，经常使用可靠性指数来度量 PPI 真实存在的可能性。目前，从不确定图中挖掘频繁模式的研究还很少。文献 [171] 给出了第一个从不确定图数据中挖掘频繁子图的算法 MUSE。MUSE 首先提出了一种新的数据模型来表示图的不确定性，然后给出了一种新的频繁子图支持度定义，并证明这种支持度定义满足反单调性质。在此基础上，MUSE 采用深度优先搜索方式挖掘全部频繁子图。在挖掘过程中，MUSE 使用了一种高效的支持度计算方法和一系列搜索空间裁剪技术，来尽可能地降低子图同构测试次数。

1.2.1.1.2 单图模型

在单图模型中，输入是一个非常大的图，该图可能包含多达几千万个结点。例如:www 本身就是一个大图。单图模型的应用领域包括 www 研究、社会网络分析等。

在单图模型上进行频繁子图挖掘就是寻找在这个大图中多次出现的子图。但是，单图模型中图模式支持度的定义更复杂。在图集合模型中，一个图模式的支持度是根据该模式在多少个图中出现来确定的，而不是根据该模式在一个图中出现了多少次。在单图模型中，因为一个图模式在一个大图中多次出现可能存在边重叠的情况，所以该模式的支持度有两种不同的定义方式。一种定义方式不允许边重叠，图模式的支持度是边不重叠情况下出现的最大次数。另一种定义方式是允许边重叠，图模式的支持度仅由该模式出现的次数确定。例如，在图 2 中，图 (a) 表示需要计算支持度的图模式 P，图 (b) 表示输入的单图 G。图模式 P 在单图 G 中出现了三次，如图 2(c)(d)(e) 所示。如果使用第一种支持度定义方式，不允许边重叠，那么图模式 P 在 G 中的支持度是 2。如果使用第二种支持度定义方式，允许边重叠，那么图模式 P 在 G 中的支持度是 3。当使用第二种支持度定义方式时，支持度不再具有向下闭包的性质。

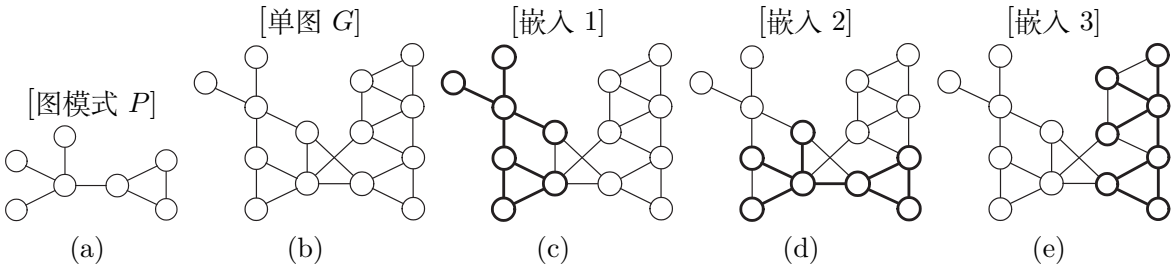


图 2　单图环境中图模式支持度的两种不同定义

与图集合模型相比，在单图模型上挖掘频繁子图的研究工作相对较少。早期的研究工作主要采用贪心算法，如 SUBDUE [61] 和 GBI [157]。贪心算法的缺点是不能保证挖掘结果的完整性。文献 [86] 给出了两种在单图模型上挖掘频繁子图的算法 HSIGRAM 和 VSIGRAM。HSIGRAM 和 VSIGRAM 之间的区别在于，HSIGRAM 采用宽度优先搜索方式遍历图模式空间，而 VSIGRAM 采用深度优先搜索方式遍历图模式空间。为了利用支持度的反单调性质，HSIGRAM 和 VSIGRAM 均采用第一种支持度定义方式，即图模式的支持度定义为边不重叠情况下出现的最大次数。HSIGRAM 和 VSIGRAM 都假定输入的单图中存在大量的不同标号，否则算法不能在可接受时间内终止。

无论是图集合模型还是单图模型，目前频繁子图挖掘存在的主要问题不是算法的挖掘效率，而是输出的频繁子图模式数量太多，冗余太多。在实际应用中，用户需要的是一个冗余小并且有意义的子图模式集合。文献 [75] [127] [141] [143] 给出了频繁项集的一些压缩表示方法。但这些方法最初的设计目标都是用来将频繁项集压缩成一个

小的代表项集集合，而不是将任何类型的频繁模式（如图模式）压缩成一个小的代表模式集合。因此，这些方法或者不能用于图数据库，或者效率不高。目前，频繁子图模式的压缩是一个重要挑战，这方面的研究工作才刚刚开展，取得的成果还非常少。

1.2.1.1.3 频繁子图搜索算法的完整性和策略

对频繁子图的挖掘，可以从搜索算法的完整性和策略这两方面分别讨论，从完整性的角度，当前搜索算法可以分为不精确搜索和精确搜索 [170]。

（1）不精确搜索算法

不精确搜索算法往往用近似计算来衡量两个图的相似程度，即任意两个子图只需要在某种程度上与候选子图近似即可。该类方法难以找到全部的频繁子图，但能够大幅提高计算效率。GREW [87] 作为一类经典的不精确子图搜索算法，致力于利用启发式的方法，寻找单个大规模图上的垂直–不连接子图。FASs [6] 作为基于模糊搜索的频繁子图挖掘方法，一方面引入计算团方法降低了对子图挖掘数量的需求，另一方面尝试在多图集合中挖掘频繁子图，其效果也非常显著。

（2）精确搜索算法

事实上，大部分的频繁子图搜索算法都是精确搜索，该类算法与不精确搜索算法的最大区别在于，其挖掘结果是完整的，换言之，精确搜索算法能够在给定数据中找到全部频繁子图。文献 [131] 中的工作基于 MapReduce，尝试从给定的大规模结点分类图数据集中搜索得到所有的频繁子图。具体而言，该方法分为两个阶段，第一阶段对图中的每个结点挖掘其局部频繁子图，而后对全部结点的子图进行过滤。第二阶段，使用过滤后生成的候选子图集来计算得到的每个图的频率。

1.2.1.1.4 动态图上的频繁子图挖掘

随着对图数据的深入研究，包含时间信息的动态图越发得到研究者们的重视 [5]。现有的频繁子图挖掘方法无论是事务性挖掘还是单图挖掘，都集中在静态图上 [46]。然而，当图数据需要大量动态更新时，对子图的搜索匹配就变得更具挑战性。Cao [49] 等人提出了一种近似分布式的方案，将静态的子图索引改进为动态的子图索引，并扩展到大规模动态图上。Ray [30] 等人提出了 StreamFSM 方法，作为动态图上频繁子图挖掘的近似解。具体而言，在图数据更新后，该方法专注于挖掘更新数据周围的采样区域，但要求图数据是批量添加和更新的。

1.2.1.1.5 流环境下的图模式挖掘

近年来，在流环境下挖掘图模式的研究工作正在逐渐受到关注 [112]。流模型往往假设输入可以在数据的多次传递中按照规定顺序读取，而用于计算的随机存取存储器 (RAM) 的总量与输入的大小呈次线性关系。图模式挖掘在流环境下的任务目标是减少

数据的传递次数，同时使用于存储中间结果的 RAM 量最小化。

Bifet 等人讨论了在不断演变的数据流上对频繁闭合图进行挖掘的可行性，提出了一种增量式的图挖掘方法 [17]。该方法基于这样一个假设，即数据总是以图的形式到达。在这种情况下，当数据按照批次传输时，该方法都会对最新批次的数据进行闭合子图挖掘，同时更新频繁闭合图集。

Angel 等人研究了如何在流式边缘权重更新的情况下，不仅能维护密集子图，还能将实时识别应用于社交媒体流 [12]。该工作构建了一个实体图，其结点对应于现实世界的实体，结点间边的权重对应于它们在帖子中的关联程度。给定用户生成的帖子流，可以获得上述实体之间的相应边缘权重更新流。

1.2.1.1.6 大规模图模式挖掘

随着人类生产、生活方式的不断更新与进步，在此过程中产生的数据量也与日俱增。图数据正朝着大规模、多尺度的方向发展，如何在大规模图数据上进行模式挖掘，已经成为研究者们日益关注的焦点 [13]。

以 MRPF [97] 方法为例，其致力于从复杂的大规模图中进行模式挖掘。该方法主要分为四个步骤，首先对大规模图数据进行分布式存取，而后实现了结点的邻居结点查找与模式初始化，接下来通过基于 MapReduce pass 实现模式扩展和频率计算，最后将任务划分为多个大小相同的子任务，每个子任务又被分配到集群结点上。

Luo 等人提出了一种在 MapReduce 框架下对图数据库进行子图搜索的方法 [98]。该方法首先为数据库中的图构建倒序的边索引，而后通过索引来搜索与查询子图相关的图数据，从而作为查询结果输出。事实上，该方法执行了两个 MapReduce 作业来构建倒序索引。其中，第一个作业负责为图数据库中的每个唯一边构建倒序索引，第二个作业则为这些倒序的索引来构建索引。

此外，也有许多工作关注于全局图模式的挖掘，全局图模式的挖掘的本质与描述性统计中使用的特征度量（如均值、方差和偏斜）非常相似。如 HADI [37] 算法，致力于在 MapReduce 上挖掘海量图中的全局图模式，已被用于分析最大的公共网络图，具有数十亿个结点和边的规模。

1.2.1.2 基于约束的图模式挖掘

基于约束的图模式挖掘就是在挖掘频繁子图时，附加额外的约束条件，要求发现满足约束条件的所有频繁子图。目前，这方面的研究工作都是在图集合模型上进行的。解决这个问题的最简单的方法就是先产生所有频繁子图，然后再逐一检查每个频繁子图是否满足约束条件。显然，这种方法效率太低，而且没有利用约束条件对图模式搜索空间进行有效的裁剪。基于约束的图模式挖掘主要研究如何将约束条件集成到挖掘

过程中以提高挖掘效率。根据约束条件的类型，基于约束的图模式挖掘又可分为:(1) 基于统计约束的图模式挖掘；(2) 基于结构约束的图模式挖掘。

基于统计约束的图模式挖掘就是对频繁子图的统计特性附加额外的约束条件。两个典型的例子为挖掘频繁闭图模式 (如果频繁图模式 p 的所有真超图的支持度都与 p 不相等，那么称 p 为频繁闭图模式) 和挖掘极大频繁图模式 (如果频繁图模式 p 的所有真超图都不是频繁的，那么称 p 为极大频繁图模式)。文献 [146] 提出了一个频繁闭图模式挖掘算法 CloseGraph。CloseGraph 建立在 gSpan [145] 的 DFSCode 枚举框架上。同时，CloseGraph 提出了一些新的概念，如等价出现、早期终止等，来帮助裁剪图模式搜索空间中不含有闭图模式的分枝。挖掘频繁闭图模式可以减少输出图模式的数量。而且，从频繁闭图模式可以恢复所有频繁子图的支持度。然而，频繁闭图模式的数量仍然很大。用户还需要从大量挖掘结果中再次选择真正有意义的图模式。为此，研究人员又提出了两个挖掘极大频繁图模式的算法 SPIN [67] 和 MARGIN [122]。挖掘极大频繁图模式可以极大地减少输出模式的数量。然而，该方法不能恢复所有频繁子图的支持度。而且，该方法会丢失一些真正有意义的图模式。

基于结构约束的图模式挖掘就是对频繁子图的拓扑结构附加额外的约束条件。限定图模式大小、连通度、密度、直径的取值范围是常见的结构约束条件。另外，限定图模式是否包含某个特定子结构也是一种常见的结构约束条件。文献 [168] 提出了一些图数据上特有的结构约束，并对结构约束进行系统的分类。文献 [168] 还给出了合并各种约束条件的一个通用的挖掘框架 gPrune。频繁集团挖掘是最常见的基于结构约束的图模式挖掘。文献 [107] 提出了一个从图集合中挖掘类集团 (quasi-clique) 的算法 Crochet，但是 Crochet 要求集团必须在所有图中都出现。文献 [130] 提出了一个从图数据库中挖掘频繁闭集团的算法 CLAN。后来，文献 [161] 扩展了算法 CLAN，提出了挖掘频繁闭类集团的算法 Cocain。文献 [162] 又扩展了算法 Cocain，从基于磁盘的图数据库中挖掘频繁闭类集团。密度约束是一类重要的结构约束，因为高密度子图通常代表高度相关的对象。文献 [50] 提出了一种以数据流的方式挖掘 www 上稠密子图的算法。文献 [65] 给出了一个在生物网络中挖掘稠密子图的算法。连通度约束也是一类重要的结构约束，因为高连通子图通常代表联系紧密且稳定的群体。文献 [150] 给出了两个从关系图中挖掘高连通性的频繁闭图模式算法 CLOSECUT 和 SPLAT。本质上，频繁子树挖掘 [114,121] 也属于基于结构约束的图模式挖掘。

基于约束的图模式挖掘还存在如下问题:(1) 现有方法通常只考虑如何利用单一约束条件。在实际应用中，用户可能需要利用多个约束条件。如何将多个约束条件最优地集成到挖掘过程中，使整体挖掘效率最大化是基于约束挖掘面临的一个重要挑战。(2) 约束条件的选择依赖于具体的应用。如何为用户推荐合适的约束条件，最大限度地减轻用户的工作量是基于约束挖掘面临的另一个重要挑战。

1.2.1.3 相关图模式挖掘

相关图模式挖掘是数据挖掘领域中的一个重要研究内容。从关系数据库中挖掘相关性已有大量的研究成果，然而，从图数据库中挖掘相关性的研究工作还很少。

文献 [78] 给出了在图数据中挖掘与给定查询子图统计相关的全部子图模式的算法 CGSearch。CGSearch 采用 Pearson 相关系数 (Pearson correlation coefficient) 作为相关性度量指标，挖掘全部与给定查询子图的 Pearson 相关系数不小于某一阈值的子图模式。CGSearch 通过两步得到最终的结果集。第一步，根据相关系数的阀值，导出与给定查询子图统计相关的图模式的支持度范围；然后利用这个支持度范围，在给定查询子图的支持集上挖掘频繁子图，形成候选集。第二步，利用一些启发式规则和 Pearson 相关系数的定义提纯候选集，得到最终的结果集。CGSearch 的作者在文献 [79] 中扩展了 CGSearch 算法，提出了 CGSearch* 算法。CGSearch* 预先存储图数据库中的频繁子图，当给定查询子图具有高支持度时，CGSearch* 不需要挖掘图数据库，通过查询预先存储的频繁子图，就可以形成候选集。文献 [79] 也考虑了如何扩展 CGSearch* 处理通用的相关性度量。

文献 [104] 把相关性度量 h-confidence 的概念扩展到图环境中，如果几个图模式构成的集合使得 h-confidence 度量大于某个用户给定的阀值，那么就认为这几个图模式高度相关，它们构成的集合被称为超集团模式。文献 [104] 还给出了一个算法 HSG 挖掘所有的超集团模式。HSG 利用 h-confidence 度量的上界设计有效的裁剪技术，同时根据子图的特定顺序，采用深度优先 (或宽度优先) 的搜索策略挖掘所有超集团模式。

1.2.1.4 基于意义度量的图模式挖掘

在实际应用中，用户需要的是真正有意义的图模式。目前，有少量研究工作考虑如何度量图模式的意义，并挖掘使意义度量最大的图模式。

文献 [58] 评价了频繁子图的统计意义。频繁子图首先被表示成一个特征向量，向量的基本元素根据领域知识由用户来选择。频繁子图的统计意义由对应向量的统计意义来确定。向量在随机向量数据库中支持度的概率分布根据该向量基本元素的先验概率来计算。向量的统计意义由该向量实际支持度的 p-值来确定。该方法属于模式后处理方法，需要先挖掘所有频繁子图，当频繁子图数量很多时，该方法的效率低下。而且，缺乏领域知识的用户很难确定向量的基本元素及其先验概率。

文献 [144] 研究了如何利用图数据库挖掘使用户给定的意义度量最大化的图模式，并给出了一个通用的挖掘框架 LEAP。LEAP 使用了两种新的技术:(1) 结构跳跃搜索；(2) 频度递减挖掘。文献 [144] 发现结构相似的图模式在意义度量上的差别也不大。结构跳跃搜索利用了这个性质，跳过图模式空间中类似的分枝，以提高挖掘效率。文献 [144]

还发现图模式的意义度量和支持度存在一定的相关性，最有意义的图模式的支持度通常情况下不是很小。频度递减挖掘利用了这个性质，每次固定某个支持度阀值，遍历图模式空间挖掘该阀值下最有意义的图模式。然后，将支持度阀值减小一半，再次遍历图模式空间挖掘最有意义的图模式。这个过程一直迭代下去，直到最有意义的图模式的度量值收敛。而且，LEAP 方法并不限于某个特定的意义度量，而是适用于用户给定的任何意义度量。实验结果显示，LEAP 方法在某些图数据库上比传统的分枝限界算法快一个数量级。

文献 [111] 给出了一个可扩展的算法 GraphSig，可以从图数据库中挖掘具有统计意义的图模式。GraphSig 需要根据领域知识选择一个有意义的特征集合，特征的先验概率可以通过实验计算获得。GraphSig 进行了如下工作: 首先，GraphSig 把数据库中的每个图转换成一个特征向量集合，每个特征向量对应图中的一个区域。然后，GraphSig 挖掘产生特征向量，识别出有意义的子特征向量。每个有意义的子特征向量可能对应一个特定的有意义子图。对每个有意义的子特征向量，把含有对应子图的区域放在一起构成一个图集合。最后，在每个图集合上，GraphSig 使用高支持度阀值挖掘频繁子图，得到原始图数据库中那些具有低支持度并且具有统计意义的图模式。

1.2.2 图查询

计算机视觉等领域很早就有关于图匹配算法的研究，但这些研究只考虑少量图数据的情况。近年来，随着图数据的大量积累，图数据的查询问题逐渐受到研究者们的关注。在关系数据库中，有效的索引方法可以极大地改善查询处理性能。受这一思想的启发，研究者们提出了一些图数据的索引方法和查询处理技术。

图查询可分为子图查询 (subgraph query)、超图查询 (supergraph query)、子图形似性搜索 (subgraph similarity search) 等。

1.2.2.1 子图查询

子图查询问题定义如下: 输入是一个图数据库 D 和查询图 q，输出是 D 中包含 q 的图集合，即 $\{g|g \in D \wedge q \subseteq g\}$。

从方法论的角度考虑，子图查询处理方法可分为两类。第一类方法是基于特征的查询处理方法。初始阶段，根据一定的选取规则生成索引特征，并创建索引结构。当查询图 q 给定时，根据 q 包含的索引特征，通过某些过滤手段对图数据库进行过滤，形成规模较小的候选集。最后，在候选集上执行子图同构测试并得到最终的结果集。该类方法通常采用如下的过滤规则: 设 $Index$ 是索引特征集合，$F = \{f|f \subseteq q \wedge f \in Index\}$ 是查询图 q 包含的索引特征集合，D_f 是数据库 D 中包含特征 f 的图集合，则查询

图 q 的候选集为 $C_q = \bigcap_{f \subseteq F} D_f$。基于特征的查询处理方法具体涉及的技术包括: (1) 基于路径的图索引方法 GraphGrep [113]; (2) 基于频繁子图的图索引方法 gIndex [147,148] 和 FG-index [34]; (3) 基于频繁子树的图索引方法 TreePi [163]; (4) 基于归纳子图的图索引方法 GDI [134]; (5) 基于频繁子树和鉴别子图 (discriminative subgraph) 的图索引方法 Tree + Delta [167]。第二类方法将图数据组织成一个特殊的索引结构，当查询图到来时，利用这个索引结构进行有效的搜索。这类方法主要包括 Closure-Tree [57] 等。Closure-Tree 借鉴 R-tree 的思想，将图数据组织成一个闭包树 (Closure-Tree)。闭包树中的叶结点代表数据库中的图，中间结点代表该结点所有后裔的一个图闭包 (graph closure)。Closure-Tree 的查询处理过程分为两个阶段。第一个阶段，Closure-Tree 遍历闭包树，如果某个中间结点不含有查询图，则裁剪以该结点为根的分枝，遍历完成之后得到候选结果集 (闭包树中的叶结点)。第二个阶段，在候选集上执行子图同构测试，确定最终的结果集。

此外，在特定的环境中，利用背景知识也可以提高查询处理效率，例如，用于化合物数据的图索引方法 GString [74]。

1.2.2.2 超图查询

超图查询问题定义如下: 输入是一个图数据库 D 和查询图 q，输出是 D 中被 q 包含的图集合，即 $\{g|g \in D \wedge g \subseteq q\}$。目前，超图查询的研究工作还比较少，已知的方法包括 cIndex [28] 和 GPTree [164]。

cIndex 采用基于特征的查询处理方法。初始阶段，cIndex 使用在数据库中频繁出现而在查询日志中很少出现的子图模式作为索引特征，建立索引结构。当查询图 q 给定时，cIndex 仍然采用“过滤 − 验证”机制，根据不被 q 包含的索引特征，通过某些过滤手段对图数据库进行过滤，形成规模较小的候选集。最后，在候选集上执行子图同构测试并得到最终的结果集。cIndex 采用如下的过滤规则: 设 $Index$ 是索引特征集合，D_f 是数据库 D 中包含特征 f 的图集合，则查询图 q 的候选集为 $C_q = D - \bigcup_{f \notin q \wedge f \in Index} D_f$。

GPTree 通过多种方式来提高超图查询处理效率。GPTree 首先提出了一种图编码方法 GVCode，根据这种编码方法，GPTree 设计了一种图数据库的压缩组织方法。然后，给出了生成索引特征的一个精确算法和一个近似算法，并利用索引特征的最优排序来提高查询处理效率。最后，GPTree 基于数据库的压缩组织提出了一种多到一的子图同构检测算法。实验结果表明，GPTree 方法的查询处理时间比 cIndex 方法快一个数量级。

1.2.2.3 子图相似性搜索

在子图查询中，当数据库中不存在查询图的精确匹配时，返回结果为空。如果在图匹配时允许存在小的误差，可以返回近似包含查询图的图集合，那么这样的查询被称为子图相似性搜索。

文献 [149] 和 [152] 研究了图数据库中子图相似性搜索问题，提出了一个结构过滤算法 Grafil。Grafil 通过把查询图的边松弛比转化成最大允许的特征丢失，在执行结构相似性计算之前可以过滤掉数据库中很多不满足条件的图，以提高子图的相似性搜索性能。为了处理满足距离约束的子图查询，文献 [151] 给出了一个基于查询图划分的算法 PIS。Grafil 和 PIS 只支持特定的相似性度量。然而，也有存在多种图的相似性度量，例如: 基于编辑距离的相似性度量，基于最大公共子图的相似性度量等。所有的相似性度量都适用于特定的应用领域。因此，需要研究支持其他相似性度量的查询处理技术以及支持多种相似性度量的查询处理技术。

1.2.3 图分类

图分类是图挖掘领域的一个重要研究分支。它的基本任务就是通过学习已知类别图数据建立分类预测模型，实现对未知类别图数据的自动分类。图分类方法大致可分成两类:(1) 基于特征的分类方法；(2) 基于核函数的分类方法。

1.2.3.1 基于特征的分类方法

基于特征的分类方法是先将图数据转换成特征向量，然后再使用传统分类方法 (例如支持向量机) 构造分类模型。按照特征来源的不同，基于特征的分类方法又可分为: (1) 基于物理化学特性的分类方法；(2) 基于拓扑特征或几何特征的分类方法。

基于物理化学特性的分类方法由领域专家根据应用背景和领域知识事先指定合适的理化特征 (例如: 分子重量、原子电子密度等) 作为分类特征，构造分类模型。其优点是方法简单，避免出现“过拟合”现象。缺点是表达能力不足，会丢失图数据的重要信息，导致分类性能较差。

基于拓扑特征或几何特征的分类方法主要是使用频繁拓扑子图或频繁几何子图作为分类特征进行分类 [42,43]。该类方法的优点是可以获得任意拓扑 (或几何) 结构的特征，具有良好的分类性能和扩展性。缺点是挖掘频繁子图时，最小支持度参数很难确定。如果最小支持度设置过高，只能产生少量极其频繁的图模式，极大地降低了分类器的准确率；如果最小支持度设置过低，产生的图模式数量又太多，分类模型很难构造，甚至频繁子图挖掘算法都不能在合理的时间内完成挖掘任务。

1.2.3.2 基于核函数的分类方法

基于核函数的分类方法在很多应用中已被证明优于其他的方法 [63]。为此，研究人员针对图分类问题，提出了一系列图核函数 [22,48,63,77,99]。本质上，图核函数可以看作两个图之间的相似程度的一种度量。例如: 将图分解成基本成分集合，两个图的核函数可以定义为对应基本成分集合交集的大小。文献 [63] 给出了一种基于环模式的图核方法。该方法根据图中环模式集合和树模式集合定义图核函数，然后利用支持向量机进行分类。然而，该核函数的计算只适用于环个数被某一常数所限制的图。而且这种分类方法只在图中具有天然环结构（例如: 化合物）的情况下才具有较高的分类性能。如果图中没有环结构或者只有少量环结构，那么该方法并不可行。文献 [77] 和 [99] 给出了一种基于随机游走的图核方法。文献 [22] 给出了一种基于最短路径的图核方法。文献 [48] 给出了一种最优局部分配的图核方法。通常基于核函数的分类方法具有良好的分类性能。然而，很多图核函数的计算都是 NP 完全问题。因此，很多图核方法的可扩展性差，只适用于小规模的图数据。

1.2.4 生物信息学上的子图挖掘及潜在医学应用

子图挖掘方法已被广泛地应用于各种生物医学问题，其理论基础在于，许多生物特征和数据集可以表示为一个图或图的集合，这使得子图挖掘方法在生物信息学领域依然有效 [102]。

通常来说，生物信息数据需要从传统格式转换为图表示，这意味着在某些环境下会出现数据信息丢失的问题。然而，即便如此，子图挖掘方法带来的优势也远超其劣势。对于生物信息学上的子图挖掘方法，包括处理多图的问题和处理单图的问题。其中，在多个图中进行子图挖掘的方法往往关注于尽可能地在这些图中找到至少出现过一次的子图。从生物信息学的角度来看，这可能表明它们间的生物学相关性。下面将介绍子图挖掘方法在生物信息学领域的经典应用场景，并阐述其潜在的医学价值。

1.2.4.1 生物分子数据库上的子图挖掘

这是最为经典的一类应用场景，事实上，生物分子数据集也经常被当作开发新的子图挖掘方法的基准 [105]。具体而言，数据集中的每个图代表一个分子，分子的原子被视为结点，原子之间的共价键被视为边 [100]。因此，该类图上的边是无向的，但有时会存在附加信息，如对边进行标记以表示键的类型，或加权以表示键的强度或长度。在该类数据集上进行的子图挖掘往往关注于发现与化学、物理或生物特征相关的常见分子结构（例如: 苯环或二硫键），这对于发现新的药物化合物有很大的潜在应用价值。通过挖掘那些已知对常见模式具有特定药物活性的分子，研究者们可以通过搜索包含

已识别子图的未研究分子来识别潜在的新候选分子。

1.2.4.2 蛋白质结构上的子图挖掘

蛋白质的三维结构通常可以表示图结构，但并非建立在单个原子的水平上，而是在氨基酸残基的水平上。值得注意的是，图表示化的蛋白质结构往往会忽略为构建这些结构而存在的化学键，更多地关注于那些非常接近的氨基酸，无论它们之间是否存在实际键。因此，蛋白质的不同氨基酸残基被表示为图中的结点，如果残基在三维蛋白质结构中的空间接近，那么两个结点用边联结。结点间的距离在生物学上有着实际的意义，假设两个氨基酸足够接近，那么理论上它们就可能会相互作用或以某种方式相互影响。挖掘得到的子图是由在蛋白质数据集中以较高频率出现的氨基酸残基模式组成的。虽然子图挖掘的任务目标可能相当多样化，但最常见的是将找到的子图与蛋白质功能相关联。事实上，这些子图模式通常代表进化保守的三维结构或结构域，这可能表明它们在这些蛋白质中具有生物学功能 [15]，并且可以反过来用作功能预测的特征。

1.2.4.3 系统发育树上的子图挖掘

表示序列之间相似性的系统发育树的构建是生物信息学中常见的内容。由于树是一种特定类型的图，因此子图挖掘的方法可以很容易地应用在系统发育树上以找到相关子树。在这种情况下，结点代表包含在原始数据集中用于比较不同基因、蛋白质、物种或其他感兴趣的生物实体，并辅以表示推断祖先的分枝结点。边代表研究实体与其祖先之间的关系。系统发育子树的挖掘与在更大的系统发育树集合中发现稳健的系统发育关系最为相关。例如，许多用于研究序列之间进化关系的方法都会生成大量不同的树，每棵树都代表一种潜在的系统发育可能性。因此，一种常见的方法是搜索共识树，即包含大多数生成的树都同意的系统发育关系的单一树。通过挖掘不同的解决方案，可以找到那些频繁子树，并组合成共识解决方案。

1.2.5 图数据挖掘与机器学习

随着图数据的规模和应用场景的不断扩大，对图结构数据中信息的抽取与分析也成为越来越多的研究者关注的重点。传统的子图结构挖掘更多地关注于对图结构的抽取，但对于图中信息的传递及附加的特征往往难以深入分析。因而，对图数据进行更深层次的挖掘逐渐成为研究者们日益关注的焦点 [139]。

随着机器学习技术的不断发展，深度神经网络凭借其高效、鲁棒的特性快速地蔓延到不同的研究领域，许多研究者将这种技术应用到图的领域，出现在网络表征学习、

图神经网络等研究方向。这些方向的本质都在于使用先进的机器学习技术实现图中的数据挖掘。因而，当下的数据挖掘研究与机器学习技术逐渐密不可分。

近十年来，涌现了大量的图学习（图挖掘）方法，按照不同的技术基础可主要分为三类: 基于随机游走的方法、基于矩阵分解的方法和基于深度学习的方法，接下来将分别进行介绍。

1.2.5.1 基于随机游走的方法

随机游走是一种方便有效的网络采样方式 [138]。该方法可以生成结点序列，同时保留结点之间的原始关系。基于网络结构，图学习方法可以生成顶点的特征向量，使得下游任务可以在低维空间中挖掘网络信息。作为经典的图学习技术之一，随机游走在对图数据的降维中发挥着重要作用。

图结构数据具有各种数据类型和结构。图中编码的信息与图结构和顶点属性有关，这是影响信息推理与分析的两个关键因素。在实际应用中，很多图只有结构信息，而缺少顶点属性信息。因此，对图结构的学习成为图学习工作一直以来的重点关注领域。

近十年来，图学习的方法不断涌现，它们保留了丰富的网络结构信息。DeepWalk [108] 和 Node2vec [52] 是生成基本网络拓扑信息的网络表示的两种代表性方法。将结点视为单词并将生成的随机顶点序列视为单词序列（句子），模型可以通过这些将序列输入 Word2vec 模型 [93] 来学习顶点的嵌入表示。而图学习的模型往往都基于这样一个基础原理，即两个行为、偏好或特征相似的结点，它们的嵌入向量的投影或距离应当尽可能接近。

1.2.5.2 基于矩阵分解的方法

矩阵本身可以表示结点间的关系，而矩阵分解是一种将矩阵简化为其分量的方法，往往被用来学习图特征和结点嵌入。早期的图学习方法通常使用基于矩阵分解的方法来解决图嵌入问题。矩阵分解的输入是表示为图形的非关系高维数据特征，输出是一组结点嵌入。基于矩阵分解的图学习主要有两种类型，一种是图拉普拉斯矩阵分解，另一种是结点邻接矩阵分解。

1.2.5.2.1 图拉普拉斯矩阵分解

图拉普拉斯矩阵分解有两种，即转导矩阵分解和归纳矩阵分解。前者只嵌入训练集中包含的顶点，后者可以嵌入训练集中不包含的顶点。文献 [29] 中设计了通用框架，文献 [142] 总结了基于图拉普拉斯矩阵分解的图学习方法。此外，图结构可以通过使用局部回归模型和基于局部和全局回归映射的全局回归过程来捕获 [155]，而全局几何可以通过使用局部样条回归 [140] 来保留。

1.2.5.2.2 结点邻接矩阵分解

矩阵分解的另一种方法是直接分解结点邻接矩阵。一般来说，矩阵分解可用于从非关系数据中学习图结构，适用于学习齐次图。基于矩阵分解，可以在低维空间中逼近顶点邻近度。保持顶点邻近度的目的是使误差最小化。工作 [51] 中采用了顶点邻近矩阵的奇异值分解（SVD）。还有其他方法（例如: 正则化高斯矩阵分解 [10]、低秩矩阵分解 [153]）可以用于求解 SVD。

1.2.5.3 基于深度学习的方法

深度学习是过去几年最热门的领域之一。经典的神经网络模型，如递归神经网络（RNN）、卷积神经网络（CNN），难以扩展到图结构的数据上，因此研究者们对这些神经网络进行了扩展。具体来说，可分为五类，即: 图卷积网络、图注意力网络、图自编码器、图生成网络和图时空网络 [137]。

1.2.5.3.1 图卷积网络

图卷积网络 (GCN) 将卷积操作从传统数据（图像或网格）推广到图数据 [81]。其关键在于学习映射函数，通过聚合结点自身的特征和邻居结点特征来生成结点以表示向量。图卷积网络在构建许多其他复杂的图神经网络模型中发挥着核心作用。

1.2.5.3.2 图注意力网络

图注意力网络 (Graph Attention Networks) 与 GCN 略有相似，都是通过构建寻求聚合函数来融合图中的相邻结点、随机游走或候选模块，以学习图中结点的表示向量。其关键区别在于图注意力网络采用注意力机制，将更大的权重分配给更重要的结点、游走路径或模块，再与端到端框架内的神经网络参数一起学习。

1.2.5.3.3 图自编码器

图自动编码器是无监督学习下的框架，旨在通过编码器学习低维结点向量，然后通过解码器重建图数据。图自动编码器是学习图嵌入的一种流行方法，适用于没有属性信息的普通图 [25] 以及属性图 [106]。对普通图来说，许多算法通过构建具有丰富信息的新矩阵（即逐点互信息矩阵）或将邻接矩阵送到自动编码器中捕获一阶和二阶信息。对属性图来说，图自动编码器模型更倾向于使用 GCN 作为编码器的构建块，并通过链路预测解码器重构时的结构信息。

1.2.5.3.4 图生成网络

图生成网络旨在从数据中生成合理的结构。在给定图经验分布的情况下生成图从根本上具有挑战性，主要是因为图是复杂的数据结构。从而，研究者们将生成过程分

解为交替形成结点和边，以采用生成对抗式的训练过程 [18]。化合物合成是图生成网络中的一个有前途的应用领域。在化学图中，原子被视为结点，化学键被视为边缘，其任务目标是发现具有某些化学和物理特性的新的可合成分子。

1.2.5.3.5 图时空网络

图时空网络旨在从时空图中学习潜在图模式或信息，这在交通预测和人类活动预测等许多应用中越来越重要。例如，底层道路交通网络是一个自然图，其中每个关键位置都是一个结点，其交通数据被持续监控。通过开发有效的图时空网络模型，可精确把控整个交通系统的交通状态 [158]，其关键思想在于同时考虑空间依赖和时间依赖。当前可利用很多图时空网络的方法将 GCN 和 RNN 或 CNN 模型相结合，来捕捉时间与空间上的结点依赖关系，从而完成建模。

1.2.6 图挖掘领域的其他研究工作

给定一个大规模复杂网络 (例如: 互联网、社会关系网络、文献引用网络、交通网络等)，为分析结点之间的关系，研究人员提出了连接子图 (connection subgraph) 挖掘问题。连接子图挖掘就是从复杂网络中挖掘能够最好描述给定结点之间关系的连通子图。文献 [47] 给出了挖掘两个结点之间连接子图的算法。文献 [82] 和 [123] 给出了挖掘 $k \geqslant 2$ 个结点之间连接子图的算法。文献 [124] 研究了如何从有向图中挖掘连接子图。

在实际应用中，大规模复杂网络的属性 (如顶点数、边数、直径等)、结构会随时间而变化。研究复杂网络的变化趋势将有重要的指导意义。文献 [92] 发现了复杂网络的直径随时间变小的现象。文献 [14] 发现了社会网络中群体现象的形成规律。文献 [117] 使用信息论的方法发现了动态网络中的社团结构 (community)。文献 [120] 使用组合优化方法发现了动态网络中的社团结构。文献 [118] 提出了动态 tensor 分析技术。文献 [36] 将动态网络上的社团发现描述为 tensor 分解的问题，并提出了一种分解算法。文献 [89] 提出了一种动态网络中未来网络结构的预测算法。文献 [154] 研究了挖掘极大频繁模式 (包括挖掘极大频繁图模式) 问题的复杂性，证明了统计极大频繁模式个数问题是 #-hard。文献 [73] 研究了图的性质空间。文献 [115] 使用最小描述长度 (MDL) 原理研究了单个大图的压缩表示方法。文献 [91] 研究了大图中的采样问题。文献 [59] 在不确定图中计算最可靠子图。

1.3 本书的主要研究工作

1.3.1 本书的主要研究问题

本书主要研究图模式挖掘技术，研究的具体内容包括以下几个方面:

(1) 研究从图数据库中挖掘代表模式问题。目前的频繁子图挖掘算法通常会产生大量的甚至指数级数量的频繁子图模式。大量的输出结果严重地影响了挖掘算法的可用性。为了减少输出的图模式数量，研究人员提出了两种解决方法: (a) 挖掘频繁闭图模式; (b) 挖掘极大频繁图模式。第一种解决方法严格强调图模式的支持度，使得输出的图模式数量仍然很大。第二种解决方法只输出搜索空间中边界上的图模式，使得很多有意义的图模式不能被产生。为此，本书提出了一种新的解决方法——挖掘代表模式。具体研究问题定义如下: 给定一个图数据库、一个最小支持度阀值和一个代表质量阀值，挖掘一个代表模式集合 RS，使得对任意频繁子图模式 P，都存在一个代表模式 $R \in RS$，R 能很好地代表 P。优化目标是最小化代表模式的数量 $|RS|$。

(2) 研究从图数据库中挖掘核心子结构问题。很多高效的频繁子图挖掘算法已经被提出。然而，频繁子图挖掘算法输出的很多图模式都没有实际意义，例如: 只有一条边的图模式没有任何意义。为了得到有意义的图模式，研究人员经常对图模式的拓扑结构施加某种限制，挖掘具有特殊拓扑结构的图模式。例如: 挖掘集团模式、挖掘类集团模式、挖掘高连通性的图模式等。然而，实际应用中很多有意义的图模式并不具有这些特殊的结构约束，而且定义结构约束需要用户具备丰富的领域知识。为了克服现有方法的缺点，本书提出一个新的研究问题: 挖掘图数据库中的核心子结构，例如从化合物中挖掘功能团。针对实际应用中核心子结构的特征，给出了核心子结构的形式化定义，称为 Δ-跳跃模式。如果在数据库 D 上图模式 p 的支持度与 p 的任何真超图的支持度相比都大于一个整数 Δ，那么称 p 为 D 上的 Δ-跳跃模式。具体研究问题定义如下: 给定一个图数据库 D、一个最小支持度阀值和一个跳跃距离阀值 Δ，挖掘 D 中的所有频繁 Δ-跳跃模式。

(3) 研究基于联合意义度量的 Top-K 图模式挖掘问题。目前，与图模式挖掘有关的绝大部分研究主要集中在如何高效地挖掘频繁子图，以及频繁子图的各种简洁表示 (例如: 频繁闭图模式、极大频繁图模式)。然而，很少有研究考虑如何根据用户给定的意义度量来挖掘图模式。本质上，频繁子图挖掘所使用的支持度恰恰是一种特殊的意义度量。在不同的应用中，用户很可能需要不同的意义度量。因此，设计针对通用意义度量的图模式挖掘算法将有广泛的应用背景。给定一个意义度量，一个直观的图模式挖掘方法就是传统 Top-K 挖掘，即对所有可能的图模式根据度量值的大小递减排序，输出前 K 个图模式。然而，传统 Top-K 挖掘并不考虑图模式之间的相关性，输出的 Top-K 模式在结构上非常相似。如果用户得到其中一个图模式，则对其他图模式就失

去了兴趣。在实际应用中，用户需要的是一个多样化而有意义的图模式集合。为了克服传统 Top-K 挖掘方法的缺点，本书提出一个新的研究问题: 基于联合意义度量挖掘 Top-K 图模式。联合意义度量的作用域是图模式集合而不是图模式，因此充分考虑了图模式之间的相关性。具体研究问题定义如下: 给定一个图数据库 D，一个最小支持度阀值 min_sup，一个联合意义度量 M，一个整数 K，从所有频繁图模式中选择 K 个图模式 $\{P_1, P_2, \ldots, P_k\}$，使得 $M(\{P_1, P_2, \ldots, P_k\})$ 最大化。

(4) 研究图的分类问题。图分类与预测是图挖掘的一个重要研究分支，在医学、生物学等领域有着广泛的应用背景。基于频繁模式的分类方法在关系数据上已被证明具有分类准确率高、可扩展性好等优点。随着各种图模式挖掘技术的进步，本书主要研究如何利用图模式对图数据进行分类。

1.3.2 本书的主要研究成果

(1) 提出从图数据库中挖掘代表模式问题及其有效的解决方法。给出了挖掘代表模式问题的形式化定义，并证明了该问题是 NP-hard 问题。提出了一系列新的概念:δ-覆盖图、跳跃值、δ-跳跃模式等。发现了 δ-跳跃模式的一个重要性质:δ-跳跃模式一定是代表模式。提出了挖掘代表模式的三个算法: RP-FP，RP-GD，RP-Leap。RP-FP 和 RP-GD 挖掘覆盖所有频繁图模式的代表模式集合，RP-Leap 挖掘近似覆盖频繁图模式的代表模式集合。RP-FP 先挖掘频繁闭图模式和 δ-跳跃模式，然后搜集完整的覆盖信息，最后通过求解最小集合覆盖问题来计算代表模式。由于利用了 δ-跳跃模式，RP-FP 比挖掘代表项集的算法 RPglobal 有更紧的近似比。然而，当频繁闭图模式数量较多时，RP-FP 效率低。为了克服 RP-FP 的弱点，RP-GD 直接从图数据库中挖掘代表模式。RP-GD 利用了遍历图模式空间时图模式的输出顺序，每当一个图模式 P 输出时，RP-GD 先测试 P 是否是 δ-跳跃模式，如果是，RP-GD 就用 P 创建一个新的代表模式；否则，RP-GD 搜索已有的代表模式集合，测试 P 是否能被覆盖。如果 P 能被覆盖，RP-GD 继续处理下一个输出的图模式；否则，RP-GD 使用一些贪心策略创建一个新的能覆盖 P 的代表模式。为进一步改善 RP-GD 的效率，本书又提出了三个新的启发式策略: 最后成功最先测试，反向路径跟踪，基于侄子代表的覆盖，来加速覆盖测试过程。算法复杂性分析显示，当频繁闭图模式数量较多时，RP-GD 的效率要远远高于 RP-FP 的效率。由于 RP-GD 需要顺序扫描每个频繁闭图模式，但 RP-GD 的效率仍然低于 CloseGraph 的效率。为此，本书又给出了另一个更高效的算法 RP-Leap，来挖掘一个近似的代表模式集合。RP-Leap 利用了图模式搜索空间中大量分枝之间的相似性，快速地跳过那些几乎不产生代表模式的分枝，极大地提高了挖掘效率。实验结果表明:(a) RP-FP，RP-GD，RP-Leap 能得到一个小的覆盖 (或近似

覆盖) 所有频繁图模式的代表模式集合；(b) RP-GD 的挖掘效率远远高于 RP-FP 的挖掘效率，而在结果质量方面，RP-GD 类似于 RP-FP；(c) RP-Leap 以丢失少量代表模式为代价，取得了比 RP-GD 快一个数量级的性能改善。由于图代表了最通用的模式类型，本书给出的三个代表模式挖掘算法也能用来挖掘其他类型的代表模式 (例如: 项集代表模式、序列代表模式、树代表模式等)。

(2) 提出从图数据库中挖掘核心子结构问题及其有效的解决方法。针对实际应用中核心子结构的特征，本书给出了核心子结构的形式化定义，称为 Δ-跳跃模式。并发现了 Δ-跳跃模式的很多重要性质:(a)Δ-跳跃模式是稳定的，它们对噪声和数据的变化不敏感，而且 Δ 值越大，它们的抗干扰能力越强；(b) 挖掘跳跃模式能极大地减少输出模式的数量，使得后续的分析工作量被大大简化，提高了挖掘结果的可用性；(c) 挖掘跳跃模式能使有意义的图模式保留在挖掘结果中。然而，跳跃模式不具有反单调性质，挖掘它们非常具有挑战性。本书通过仔细研究跳跃模式自身的特性，提出了两种新的裁剪技术，基于内扩展的裁剪和基于外扩展的裁剪。利用这两种裁剪技术，设计了一个高效的跳跃模式挖掘算法 GraphJP。在理论上，严格地证明了这两种裁剪技术的正确性以及挖掘算法 GraphJP 的正确性。实验结果表明: 这两种新的裁剪技术能有效地裁剪图模式搜索空间，算法 GraphJP 能高效、可扩展地挖掘频繁跳跃模式，而且挖掘结果中含有图数据库中的核心子结构。本书给出的 Δ-跳跃模式定义，挖掘算法 GraphJP 也容易被扩展来挖掘其他类型的跳跃模式 (例如: 项集跳跃模式、序列跳跃模式、树跳跃模式等)。

(3) 提出基于联合意义度量的 Top-K 图模式挖掘问题及其有效的解决方法。讨论了适用于图模式集合的联合意义度量，并利用信息论中的概念 (联合熵和信息增益) 给出了两个具体的问题定义 MES 和 MIGS，证明了它们是 NP-hard 问题。为了挖掘联合意义度量下的 Top-K 图模式，提出了两个高效的挖掘算法 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 先产生频繁图模式 (或频繁闭图模式)，然后增量贪心地选择 K 个图模式。如果用户给定的意义度量满足 submodular 性质，那么 Greedy-TopK 就能提供近似比保证。为了进一步提高 Greedy-TopK 的效率，又针对 MES 和 MIGS 这两个具体问题的意义度量设计了一系列有效的裁剪技术，将其嵌入频繁子图挖掘框架中帮助裁剪图模式搜索空间。然而，当频繁图模式 (或频繁闭图模式) 的数量较大时，Greedy-TopK 仍然效率低、可扩展性差。为此，又设计了另一个更高效的算法 Cluster-TopK。Cluster-TopK 先从图数据库中挖掘所有频繁图模式的一个代表模式集合，然后从代表模式中增量贪心地选择 K 个图模式。Cluster-TopK 最大的优点是无需产生频繁图模式 (或闭图模式) 就能快速地从图数据库中挖掘一个代表模式集合。因为代表模式的数量通常比闭图模式的数量少很多，所以 Cluster-TopK 的贪心选择也比 Greedy-TopK 的贪心选择快很多。而且，本书从理论上证明了 Cluster-TopK 产

生的解和 Greedy-TopK 产生的解非常接近。实验结果表明: 在结果质量和可用性方面，本书提出的 Top-K 挖掘远远优于传统的 Top-K 挖掘。Cluster-TopK 比 Greedy-TopK 快一到两个数量级。而且，Cluster-TopK 的挖掘结果质量非常接近于 Greedy-TopK 的挖掘结果质量。Greedy-TopK 和 Cluster-TopK 是通用的 Top-K 挖掘算法，也适用于其他的联合意义度量和模式类型 (例如: 项集模式、序列模式、树模式等)。

(4) 提出了一种基于频繁闭显露模式的图分类框架 CEP。CEP 包括三个步骤: 挖掘频繁闭图模式；过滤非显露模式；构造分类规则。第一步，CEP 挖掘所有频繁闭图模式作为候选分类特征。第二步，CEP 保留频繁闭图模式中的显露模式。该步需要计算图模式在不同类别数据库中的支持度，涉及大量子图同构测试。为改善算法效率，CEP 将频繁闭图模式组织成一个树型结构 T。对数据库中的每个图 G，采用深度优先方式遍历树 T。在遍历过程中，利用反单调性质进行裁剪，如果 G 不包含结点 P，那么 G 也不可能包含 P 的孩子结点。通过这种方式，可以极大地减少子图同构测试次数。第三步，CEP 根据剩余的显露模式构造分类规则。在构造分类规则时，提出了一个新的特征选择方法，该方法既考虑显露模式如何覆盖图数据，又考虑显露模式在图数据中如何分布。实验结果显示:CEP 具有良好的分类性能。而且，领域专家也容易理解和利用 CEP 产生的分类规则。

1.4 本书的章节安排

本书其他章节的安排如下: 第 2 章介绍代表模式挖掘算法；第 3 章介绍跳跃模式挖掘算法；第 4 章介绍基于联合意义度量的 Top-K 挖掘算法；第 5 章介绍基于显露模式的图分类方法。

第 2 章　挖掘代表模式

2.1　引言

近年来，科学与工程领域积累了大量可用图来建模的数据，如化合物的分子结构、蛋白质交互网络、社会网络等。设计基于图的挖掘算法已成为数据挖掘领域中一项重要的研究课题。目前，很多高效的频繁子图挖掘算法 [20,66,69,83,103,145] 已经被提出。然而，大多数情况下，这些挖掘算法将产生大量的甚至指数级数量的频繁子图。大量的输出结果严重地影响了挖掘算法的可用性。

挖掘频繁子图的一个最主要的应用就是通过分析发现的图模式来获取图数据库中的有用信息。例如: 在化学信息学领域，化学家们经常需要研究在某类化合物中分子的子结构和化学性质之间的关系。在使用某个频繁子图挖掘算法产生该类化合物的所有频繁子结构之后，化学家们再使用某个可视化工具将产生的频繁子结构及其相关信息（如支持度）显示在屏幕上，然后根据自己的领域知识逐个分析每个频繁子结构，以发现某些子结构和该类化合物的化学性质之间的关系。这种分析的工作量与产生的频繁子结构数量成正比。因此，对于这样的分析，如果挖掘算法能产生一个小的而有意义的子结构集合，那么将极大地减轻化学家们的工作量。不幸的是，目前存在的频繁子图挖掘算法都会产生大量的图模式。例如: 文献 [146] 显示，在一个常用的抗艾滋病病毒的化合物集合 CA 上，即使将支持度设置为 10%，也将输出 15 000 多个频繁子结构。手动地分析这么多的子结构是极其困难的。同样地，在生物学领域，生物学专家在挖掘基因相关网络时也会遇到类似的问题。例如: 在基于中等规模的微阵列数据建立的基因相关网络中，使用图模式挖掘算法 Splat [150] 也将产生几千个网络模式。

为了减少输出图模式的数量，两类主要的解决方法已经被提出:(1) 挖掘频繁闭图模式 [146]；(2) 挖掘极大频繁图模式 [67,122]。如果频繁图模式 p 的所有真超图的支持度都与 p 不相等，那么称 p 为频繁闭图模式。如果频繁图模式 p 的所有真超图都不是频繁的，那么称 p 为极大频繁图模式。第一类解决方法严格强调图模式的支持度，使得输出图模式的数量仍然很大（见 2.5.2 节的实验）。第二类解决方法只考虑了图模式的结构信息，使得很多图模式的支持度信息丢失，很多有意义的图模式不能被产生（见 2.5.7 节的实验）。

实际上，可以用一个小的代表模式集合来概括地表示所有频繁图模式，使得领域专家只分析这个小的代表模式集合就足以了解图数据库中的信息。请看下面的实例，图

1 显示了 5 个化合物的子结构，这 5 个子结构是由从一个真实的化合物集合 NCI/CA 中使用 10% 的支持度挖掘出来的频繁子结构的一个子集。子结构 P_1，P_2，P_3，P_4，P_5 的支持度分别是 27.7%，28.6%，29.1%，15.4%，16.1%。根据闭图模式的定义，这 5 个子结构都将被闭图模式挖掘算法输出。然而，我们注意到，P_1，P_2 和 P_3 在结构上非常相似，而且 P_1，P_2 和 P_3 的支持度也非常接近。同时，P_1 又是 P_2 和 P_3 的超图。因此，P_1 非常适合作为 P_2 和 P_3 的代表模式。同样地，P_4 和 P_5 在结构上非常相似，它们的支持度也非常接近。P_4 又是 P_5 的超图，P_4 非常适合作为 P_5 的代表模式。本质上，对如图 1 所示的 5 个图模式，只产生两个代表模式 P_1 和 P_4，将它们提供给领域专家就足够了。领域专家通过分析这两个代表模式，就可以获得足够多的有用信息。

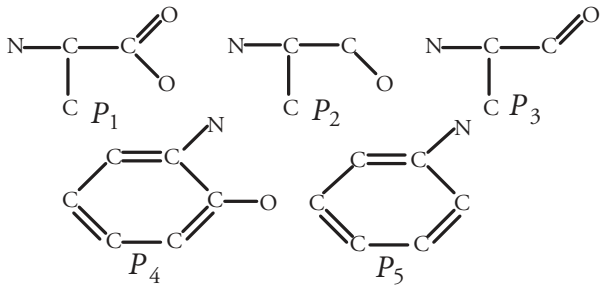


图 1　NCI/CA 中频繁子结构的一个子集

上面的例子说明: 对频繁图模式集合，我们应该产生一个合适的代表模式集合并将它提供给用户。而且，代表模式的数量越少越好。得到代表模式的一个常规做法就是将所有频繁图模式聚类成若干个组，从每组中选择一个代表模式，该代表模式能很好地代表该组中的其他图模式。例如，在图 1 中，子结构 P_1 可以作为 $\{P_1, P_2, P_3\}$ 这一组的代表模式。为了保证结果质量，代表模式和被代表模式之间的某个距离度量值不应大于用户给定的参数 δ。如果一个模式 P 可以代表另一个模式 P'，那么称 P 能 δ-覆盖 P'，或者称 P' 能被 P δ-覆盖（见后面 δ-覆盖的定义）。δ-覆盖有时也简称为覆盖。本章的研究问题定义如下: 给定一个图数据库、一个最小支持度阀值和一个距离阀值，挖掘一个代表模式集合 RS，使得对任何频繁图模式 P，都存在一个代表模式 $R \in RS$，R 能覆盖 P。优化目标是最小化代表模式的数量 $|RS|$。本章严格地证明了该问题是一个 NP-hard 问题。

据我们所知，尽管从数据库中挖掘代表项集问题已有文献 [141] 涉及（见 2.2 节相关工作），然而从图数据库中挖掘代表图模式问题，目前还没有人研究。本章主要研究如何高效地从图数据库中挖掘代表模式。在 δ-覆盖概念 [141](如果 $P \subseteq P'$，并且 P 和 P' 之间的距离小于或等于 δ，那么称 P' δ-覆盖 P) 的基础上，本章提出了三个新的概念:δ-覆盖图、跳跃值、δ-跳跃模式。如果图模式 P 与 P 的任何真超图之间的距离值都大于 δ，那么称 P 为 δ-跳跃模式。本章证明了所有的 δ-跳跃模式一定是代表

模式。利用 δ-跳跃模式的这个性质，本章给出了两个从图数据库中挖掘代表模式的算法:RP-FP 和 RP-GD。

2.4.1.1 节中的引理 2.4.1 显示: 如果一个代表模式集合 RS 能覆盖所有的频繁闭图模式，那么 RS 也一定能够覆盖所有的频繁图模式。因此，我们只需要挖掘覆盖频繁闭图模式的代表模式集合。

RP-FP 从频繁闭图模式中计算代表模式，包含四个步骤:(1) RP-FP 从图数据库中挖掘频繁闭图模式和 δ-跳跃模式；(2) RP-FP 计算不能被 δ-跳跃模式覆盖的频繁闭图模式集合 $\overline{E}$; (3)RP-FP 在 $\overline{E}$ 上搜集完整的覆盖信息，对每个 $P \in \overline{E}$，RP-FP 搜集所有能覆盖 P 的图模式；(4) 在第三步完成之后，问题被转化成了一个最小集合覆盖问题。RP-FP 使用求解最小集合覆盖问题的贪心算法来计算能覆盖 $\overline{E}$ 的最小代表模式集合。由于利用了 δ-跳跃模式，我们证明了 RP-FP 比挖掘代表项集的算法 RPglobal [141] 有更紧的近似比。然而，当频繁闭图模式数量较大时，RP-FP 效率较低，因为 RP-FP 在第三步搜集完整的覆盖信息时，必然涉及大量的子图同构测试。子图同构测试 [39] 已被证明是 NP-hard 问题，因此，我们需要更高效的算法来尽可能地降低子图测试次数。

为了克服 RP-FP 存在的弱点，RP-GD 直接从图数据库中挖掘代表模式。RP-GD 利用了现有的闭图模式挖掘算法 CloseGraph [146] 遍历图模式空间时输出的图模式顺序。每当有一个图模式 P 被挖掘算法输出时，RP-GD 先测试 P 是否是 δ-跳跃模式，如果 P 是 δ-跳跃模式，RP-GD 就用 P 创建一个新的代表模式；否则，RP-GD 搜索已有的代表模式集合，测试 P 是否能被已有的代表模式覆盖。如果 P 能被某个已有的代表模式覆盖，RP-GD 继续处理下一个输出的图模式；否则，RP-GD 使用一些贪心策略创建一个新的能覆盖 P 的代表模式。在 RP-GD 算法中，一个关键的步骤就是测试已有的代表模式是否能覆盖当前输出的图模式。为了加速这个测试过程，本章提出了三个新的启发式策略: 最后成功最先测试，反向路径跟踪，基于侄子代表的覆盖，来提高 RP-GD 的效率。最后，通过算法的复杂性分析，我们在理论上严格地证明了:当频繁闭图模式数量较大时，RP-GD 的效率要远远高于 RP-FP 的效率。

尽管 RP-GD 的效率极大地优于 RP-FP 的效率，RP-GD 仍然要顺序扫描 CloseGraph 输出的每个闭图模式，这使得 RP-GD 的效率仍然低于 CloseGraph 的效率。当支持度阀值过低时，CloseGraph 的挖掘效率仍然很低。在实际应用中，如果能更高效地得到一个近似覆盖所有频繁图模式的代表模式集合，也是用户非常乐意接受的。鉴于上述原因，本章又给出了一个更高效的算法 RP-Leap，来挖掘一个近似的代表模式集合。RP-Leap 利用了图模式搜索空间中大量分枝之间的相似性，快速地跳过那些几乎不含有代表模式的分枝，极大地提高了挖掘效率。而且，RP-Leap 得到的代表模式集合能覆盖绝大部分的频繁图模式。

大量真实数据和合成数据上的实验结果显示:(1)RP-FP，RP-GD，RP-Leap 能得

到一个小的覆盖（或近似覆盖）所有频繁图模式的代表模式集合；(2) 在挖掘效率和结果质量方面，RP-FP 优于挖掘代表项集的算法 RP-global；(3) 当频繁闭图模式数量较大时，RP-GD 的挖掘效率远远高于 RP-FP 的挖掘效率。而且，在结果质量方面，RP-GD 类似于 RP-FP。δ-跳跃模式和新提出的启发式策略是非常有效的，它们能使 RP-GD 的效率提高 5 倍；(4) 尽管 RP-Leap 会丢失少量代表模式，但 RP-Leap 却能比 RP-GD 快一个数量级。而且，RP-Leap 得到的代表模式集合能覆盖所有频繁图模式的 97% 以上；(5) 最后，通过对化合物的分类实验，证明了基于代表模式构造的分类器在分类性能方面不低于基于闭图模式构造的分类器，从而证明了代表模式的可用性。

综上，本章的主要贡献如下: 第一，提出了一个新的研究问题，从图数据库挖掘代表模式，并证明了该问题是 NP-hard 问题。第二，提出了 δ-跳跃模式的概念，并证明了所有 δ-跳跃模式一定是代表模式。第三，利用 δ-跳跃模式的性质和图模式空间中多个分枝之间的相似性，提出了三个挖掘代表模式的算法 RP-FP，RP-GD，RP-Leap。第四，实验结果验证了算法的效率和挖掘结果的可用性。

本章的内容安排如下:2.2 节介绍相关工作；2.3 节介绍图模式挖掘的相关概念，提出一系列新的概念，给出问题的形式化定义，并证明问题的复杂性；2.4 节给出挖掘代表模式的三个算法 RP-FP，RP-GD，RP-Leap，以及算法复杂性分析；2.5 节通过实验验证算法的效率和挖掘结果的质量；2.6 节对本章进行小结。

2.2 相关工作

很多文献中已经提出了频繁子图挖掘算法。它们大致可分为两类，第一类算法根据 Apriori 性质使用逐级搜索策略来枚举所有频繁子图，如 AGM [69] 和 FSG [83]，第二类算法采用深度优先搜索策略来枚举所有频繁子图，如 Mofa [20]，gSpan [145]，FFSM [66]，GASTON [103]。通常第二类算法比第一类算法有更好的内存利用率，因而具有更高的挖掘效率。

因为本章的内容与 gSpan [145]，CloseGraph [146] 密切相关，所以接下来会详细地讨论它们。gSpan 是目前最有效的频繁子图挖掘算法之一 [136]，gSpan 使用一种规范的编码系统，把每个图都映射成唯一对应的深度优先编码 (DFSCode)，所有 DFSCode 定义了一种字典顺序。为了防止同一个频繁子图被多次访问，当扩展图模式时，gSpan 只在最右路径的结点上执行最右扩展。gSpan 以 DFSCode 字典顺序由小到大地输出每个频繁子图。CloseGraph [146] 是著名的闭图模式挖掘算法，它建立在 gSpan 的 DFSCode 枚举框架上。同时，CloseGraph 提出了一些新的概念，如等价出现、早期终止等，来帮助裁剪图模式搜索空间中不含有闭图模式的分枝。正如文献 [146] 中，存在一些情况，简单

地使用早期终止可能会丢失某些闭图模式。为了保证挖掘结果的完整性，CloseGraph 又提出了一个失败检测的方法，来检测早期终止的所有可能失败的情况。然而，在某些情况下 [135]，CloseGraph 给出的失败检测方法不是很有效，而且，文献 [146] 也指出，失败检测方法会使 CloseGraph 的效率下降大约一半。

挖掘代表图模式问题和挖掘代表项集问题密切相关。将频繁项集概括成一个小的代表项集问题 [7,75,127,141,143] 已经被广泛地研究。

自从关联规则挖掘问题 [8] 被 Aggrwal 提出之后，出现了很多高效的频繁项集挖掘算法 [8,9,55,71]。然而，目前频繁项集挖掘算法存在的主要问题不是挖掘算法的效率，而是输出的大量频繁项集使得挖掘算法难以被应用。近年来，数据挖掘工作者开始研究如何用少量代表项集来近似表示所有频繁项集。Afrati 等人 [7] 使用极大频繁项集里面的 K 个项集来近似表示所有频繁项集。Yan 等人 [143] 假定所有 1-项集之间相互独立，并利用这种独立性模型以概要 (profile) 的形式来概括表示所有频繁项集。Xin 等人 [141] 计算能 δ-覆盖所有频繁项集的最小代表项集集合。在文献 [127] 中，一个更复杂的模型——马尔科夫随机域模型被使用，以逐层的方式来概括表示所有频繁项集。在文献 [75] 中，作者基于一个概率模型，提出了一系列规约方法来概括表示所有频繁项集。这些方法绝大部分都属于模式后处理方法，需要先得到所有频繁项集，然后根据提出的模型将所有频繁项集概括成一个小的代表项集集合。而且，这些方法最初的设计目标都是用来将频繁项集概括成一个小的代表项集集合，而不是将任何类型的频繁模式（如图模式）概括成一个小的代表模式集合。因此，这些方法不能用于图数据库，或者用于图数据库时效率很低。

文献 [75] [127] [143] 中的方法不能被扩展并用于图数据库，因为它们使用的模型在图环境中根本不成立。如果将文献 [7] 中的方法用于图数据库，大部分频繁图模式的支持度信息将丢失，因为该方法得到的仅仅是极大频繁模式集合的一个子集合。文献 [141] 提出了两个算法，RPglobal 和 RPlocal，来计算 δ-覆盖频繁项集的最小代表项集集合。RPglobal 和 RPlocal 都需要执行大量的覆盖测试（测试一个项集是否能覆盖另一个项集），覆盖测试的代价和次数是算法性能的决定性因素。为了改善效率，RPglobal 和 RPlocal 使用了类似于 FP-tree [55] 的数据结构来存储代表项集和频繁项集以减少覆盖测试的次数，降低覆盖测试的代价。然而，这些有效的数据结构在图的环境中无法使用。而且，在图环境中，覆盖测试包含子图同构测试，子图同构测试的代价要远远大于项集包含测试的代价。因此，对图数据库中的代表模式挖掘问题，RPglobal 和 RPlocal 算法可扩展性很差，效率也很低。另外，RPlocal 算法本身是基于 FP-Growth 算法 [55] 设计的，因此，RPlocal 并不能直接用于图数据库。

类似于频繁项集挖掘的情况，挖掘频繁子图经常会产生指数级数量的图模式，使得挖掘结果难以被利用。为了解决这个问题，本书提出两个主要的方法:(1) 挖掘频繁

闭图模式 [146]；(2) 挖掘极大频繁图模式 [67,122]。第一类方法严格强调图模式的支持度，使得图模式的输出数量仍然很大（见 2.5.2 节的实验）。第二类方法只考虑搜索空间中边界上的图模式，使得一些重要的图模式丢失（见 2.5.7 节的实验）。文献 [56] 提出从极大频繁图模式集合中选择一个正交的代表子集，然而，该方法得到的只是极大频繁图模式集合的一个子集，因此仍然会丢失一些重要的图模式。

2.3 预备知识

本节介绍图模式挖掘的一些基本概念。本章主要考虑连通的无向标号简单图，通过简单修改，本章提出的方法也适用于有向图、无标号图和不连通图。如无特别说明，本书中的图均指连通的无向标号图。

定义 2.3.1(标号图) 将一个标号图 G 定义为一个四元组 $G = \{V, E, \Sigma, L\}$，其中，V 代表图中顶点的集合，$E = V \times V$ 代表图中边的集合，Σ 代表顶点标号与边标号的集合，L 是标号函数，用来对顶点和边分配标号，即 $L : V \cup E \to \Sigma$.

作为一种通用的数据结构，标号图可以用来表示数据内部的各种复杂关系，在很多应用领域中被广泛使用。例如: 在化学领域，标号图可以表示化合物的拓扑结构，其中顶点对应不同的原子，而边对应不同的化学键。在生物学领域，标号图可以表示蛋白质交互网络，其中顶点对应不同的蛋白质，而边对应蛋白质之间的交互。此外，像社会网络、传感器网络等都可以用标号图来建模。

本书中多次用到图同构和子图同构的概念，下面给出它们的数学定义。

定义 2.3.2(图同构) 给定两个图 $G = \{V, E, \Sigma, L\}$ 和 $G' = \{V', E', \Sigma', L'\}$，若存在一个双射函数 $f : V \leftrightarrow V'$，满足如下条件:
(1) $\forall u \in V, L(u) = L'(f(u))$;
(2) $\forall u, v \in V, ((u, v) \in E) \leftrightarrow ((f(u), f(v)) \in E')$;
(3) $\forall (u, v) \in E, L(u, v) = L'(f(u), f(v))$,
则称 G 与 G' 同构，记为 $G = G'$。

定义 2.3.3(子图同构) 给定两个图 $G = \{V, E, \Sigma, L\}$ 和 $G' = \{V', E', \Sigma', L'\}$，若存在一个单射函数 $f : V \to V'$，满足如下条件:
(1) $\forall u \in V, L(u) = L'(f(u))$;
(2) $\forall u, v \in V, ((u, v) \in E) \Rightarrow ((f(u), f(v)) \in E')$;
(3) $\forall (u, v) \in E, L(u, v) = L'(f(u), f(v))$,
则称 f 是一个从 G 到 G' 的子图同构，也称 G 子图同构于 G'。单射函数 f 也称为 G 在 G' 中的一个**嵌入**。

如果存在一个从 G 到 G' 的子图同构，那么称 G 是 G' 的子图，G' 是 G 的超图，

记为 $G \subseteq G'$。如果 $G \subseteq G'$ 且 $G \neq G'$，那么 G 称为 G' 的真子图，G' 称为 G 的真超图，记为 $G \subset G'$。例如: 在图 1 中，$P_2 \subset P_1$，$P_3 \subset P_1$，$P_5 \subset P_4$。子图同构测试已被证明是一个 NP-完全问题 [39]。如果 $G \subseteq G'$，有时我们也称 G' **包含** G。

定义 2.3.4(支持度)　给定一个图数据库 $D = \{G_1, G_2, \ldots, G_n\}$ 和一个图模式 P, P 在 D 中的**支持集**定义为 D 中包含 P 的图集合, 记为 $D_{\text{supp}}(P) = \{G_i | P \subseteq G_i, G_i \in D\}$。$|D_{\text{supp}}(P)|$ 称为 P 在 D 中的**绝对支持度**，记为 $\text{support}(P; D)$。$|D_{\text{supp}}(P)|/|D|$ 称为 P 在 D 中的**相对支持度**。

当上下文清楚的时候，图模式 P 的绝对支持度和相对支持度都简称为 P 的支持度，都用 $\text{support}(P; D)$ 表示。支持度度量具有**反单调性质**: 如果 $P \subseteq P'$，那么有 $\text{support}(P; D) \geqslant \text{support}(P'; D)$。

定义 2.3.5(频繁图模式)　设 $D = \{G_1, G_2, \ldots, G_n\}$ 是一个图数据库，且和 P 是一个图模式。对于用户给定的一个最小支持度阈值 min_sup，如果 $\text{support}(P; D) \geqslant$ min_sup，那么称 P 是 D 中的频繁图模式。

本书中，min_sup 既可以表示绝对最小支持度阈值，又可以表示相对最小支持度阈值。

定义 2.3.6(闭图模式)　设 $D = \{G_1, G_2, \ldots, G_n\}$ 是一个图数据库，且和 P 是一个图模式。如果在 D 中，P 的任何真超图的支持度都与 P 不相同，那么称 P 是 D 中的闭图模式。

定义 2.3.7(极大频繁图模式)　设 P 是 $D = \{G_1, G_2, \ldots, G_n\}$ 中的一个频繁图模式。如果 P 的任何真超图都不是频繁的，那么称 P 是 D 中的极大频繁图模式。

D 中所有频繁图模式集合记为 $FS = \{P | \text{support}(P; D) \geqslant \text{min\_sup}\}$。$D$ 中所有频繁闭图模式集合记为 $CS = \{P | P \in FS, \neg\exists P' \in FS$，使得 $P \subset P'$ 并且 $\text{support}(P; D) = \text{support}(P'; D)\}$。$D$ 中所有极大频繁图模式集合记为 $MS = \{P | P \in FS, \neg\exists P' \in FS$，使得 $P \subset P'$ 并且 $\text{support}(P'; D) \geqslant \text{min\_sup}\}$。当上下文清楚的时候, 我们有时用 $\text{support}(P)$ 表示 $\text{support}(P; D)$。

采用先深搜索的办法挖掘频繁子图, 会在挖掘的过程中形成一个树状的搜索空间, 这棵树的根结点为空，由根至叶按边个数递增的顺序形成树，非根结点分别代表不同的频繁子图。整个挖掘过程是由根至叶的遍历过程。图 2 显示了这个树状的搜索空间。

我们先定义一些新的概念，然后给出问题形式化定义，最后证明它是 NP-hard 问题。

如果图模式 R 可以作为图模式 P 的代表模式，显然 R 应该是 P 的超图，并且 R 和 P 在数据库中应该经常一起出现。为了度量这两个图模式是否在数据库中经常一起出现，我们使用了 Jaccard 距离 [72]。

定义 2.3.8(Jaccard **距离**)　设 P_1 和 P_2 是两个图模式，D 是一个图数据库。P_1 和

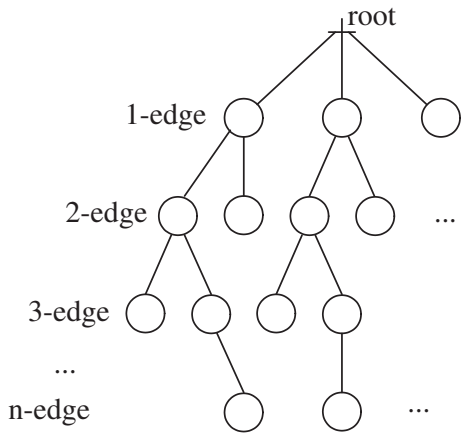


图 2　频繁子图的搜索空间

P_2 之间的 Jaccard 距离定义如下

$$D(P_1, P_2) = 1 - \frac{| T(P_1) \cap T(P_2) |}{| T(P_1) \cup T(P_2) |}$$

其中，$T(P_1)$ 是 D 中包含 P_1 的图集合。

设 $S = \{P_1, P_2, \ldots, P_n\}$ 是一个图模式集合。直觉上，如果一个代表模式 R 能代表 S 中所有的图模式，应有 $\forall P_i \in S$，$P_i \subseteq R$，并且 $D(P_i, R)$ 应很小。为了给出形式化定义，我们使用了 δ-覆盖的概念 [141]。其中，δ $(0 \leqslant \delta \leqslant 1)$ 是两个图模式之间的距离阀值。

定义 2.3.9(δ-**覆盖**)　给定两个图模式 P 和 P'，如果 $P \subseteq P'$，并且 $D(P, P') \leqslant \delta$，那么称 $P'\delta$-覆盖 P（或 P 被 $P'\delta$-覆盖）。

如果图模式 R 能 δ-覆盖图模式 P，那么称 R 是 P 的**代表模式**。被图模式 R δ-覆盖的所有图模式集合称为 R 的 δ-**覆盖集**，记为 $\mathrm{Set}_\delta(R)$。下面将 δ-覆盖简称为覆盖。给定两个图模式 P 和 P_r，只有当 $P \subseteq P_r$ 时，P_r 才有可能成为 P 的代表模式，根据反单调性质，我们可以更容易地计算 P 和 P_r 的 Jaccard 距离: $D(P, P_r) = 1 - \frac{|T(P) \cap T(P_r)|}{|T(P) \cup T(P_r)|} = 1 - \frac{|T(P_r)|}{|T(P)|} = 1 - \frac{\mathrm{support}(P_r)}{\mathrm{support}(P)}$。例如: 在图 1 中，因为 $P_5 \subset P_4$，$D(P_4, P_5) = 1 - \frac{\mathrm{support}(P_4)}{\mathrm{support}(P_5)} = 1 - \frac{0.154}{0.161} \approx 0.043$。

基于 δ-覆盖的定义，还提出了几个新的概念。

定义 2.3.10(δ-**覆盖图**)　设 S 是一个图模式集合，δ 是距离阀值，S 的 δ-覆盖图是一个有向图 $G_\delta(S) = (V, E)$，其中，V 中的顶点与 S 中的图模式一一对应，$(P_i, P_j) \in E$，当且仅当 P_i δ-覆盖 P_j。

例 2.3.1　设 $S = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ 是一个图模式集合，$\mathrm{Set}_\delta(P_4) = \{P_1, P_2, P_4\}$，$\mathrm{Set}_\delta(P_5) = \{P_1, P_2, P_3, P_5\}$，$\mathrm{Set}_\delta(P_6) = \{P_3, P_5, P_6\}$，$S$ 的 δ-覆盖图 $G_\delta(S)$ 如图 3 所示。

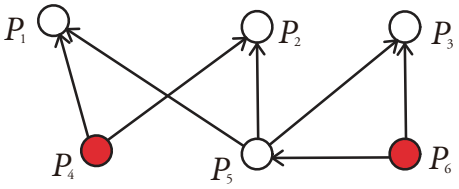


图 3　δ-覆盖图的一个例子

定义 2.3.11(跳跃值)　设 S 是一个图模式集合，P 是 S 中的任意一个图模式。如果 S 中至少存在一个 P 的真超图，那么 P 在 S 中的跳跃值定义为:$JV_S(P) = \min\{D(P,P_i)|\forall P_i \in S, P \subset P_i\}$，否则 $JV_S(P) = +\infty$。

直觉上，图模式 P 在图模式集合 S 中的跳跃值恰恰是 P 与所有在 S 中的真超图的最小距离。

定义 2.3.12(δ-跳跃模式)　设 S 是一个图模式集合，P 是 S 中的任意一个图模式。如果 $JV_S(P) > \delta$，那么称 P 是 S 中的 δ-跳跃模式。

例 2.3.2　图 1 显示了一个图模式集合 S。P_1，P_2，P_3，P_4，P_5 的支持度分别是 27.7%，28.6%，29.1%，15.4%，16.1%。因为在 S 中，P_1 是 P_2 唯一的真超图，我们有 $JV_S(P_2) = D(P_1,P_2) = 1 - \frac{\text{support}(P_1)}{\text{support}(P_2)} = 1 - \frac{0.277}{0.286} \approx 0.031$。因为在 S 中，不存在 P_1 的真超图，我们有 $JV_S(P_1) = +\infty$。类似地，可以计算 $JV_S(P_3)$ ≈0.048，$JV_S(P_4) = +\infty$，$JV_S(P_5)$ ≈0.043。如果 $\delta = 0.1$, 那么 P_1 和 P_4 是 S 中的两个 δ-跳跃模式。

引理 2.3.1　如果 P 是图模式集合 S 中的 δ_1-跳跃模式，并且 $\delta_1 > \delta_2$，那么 P 也是 S 中的 δ_2-跳跃模式。

证明　因为 P 是 S 中的 δ_1-跳跃模式，根据 δ-跳跃模式的定义，P 在 S 中的跳跃值大于 δ_1。又因为 $\delta_1 > \delta_2$，所以 P 在 S 中的跳跃值大于 δ_2。再根据 δ-跳跃模式的定义，可知 P 也是 S 中的 δ_2-跳跃模式。　□

引理 2.3.2　假设 $G_\delta(S)$ 是图模式集合 S 的 δ-覆盖图，那么，$G_\delta(S)$ 中入度为 0 的顶点与 S 中的 δ-跳跃模式一一对应。

证明　设 N 是 $G_\delta(S)$ 中任意一个入度为 0 的顶点。根据 δ-覆盖图的定义，N 代表 S 中的一个图模式 P，并且 S 中任何其他图模式都不能 δ-覆盖 P。如果 P 在 S 中的跳跃值 $JV_S(P)$ 小于或等于 δ，那么 S 中存在另一个图模式 P'，使得 $P \subset P'$ 并且 $D(P,P') \leqslant \delta$。根据 δ-覆盖的定义，P 能被 P' δ-覆盖，产生矛盾。因此，P 在 S 中的跳跃值 $JV_S(P)$ 大于 δ。根据 δ-跳跃模式的定义，P 是 S 中的 δ-跳跃模式。

设 P 是 S 的任意一个 δ-跳跃模式。根据 δ-跳跃模式的定义，P 在 S 中的跳跃值 $JV_S(P)$ 大于 δ。如果 S 中存在另一个图模式 P'，且满足 P' 能 δ-覆盖 P，那么 $P \subset P'$ 并且 $D(P,P') \leqslant \delta$。根据跳跃值的定义，$JV_S(P) \leqslant D(P,P') \leqslant \delta$，产生矛盾。

因此，P 对应 $G_\delta(S)$ 中一个入度为 0 的顶点。 □

根据引理 2.3.2 可知，在例 2.3.1 中的 P_4 和 P_6 是 S 中的两个 δ-跳跃模式。

引理 2.3.3 设 S 是数据库中的所有频繁图模式的集合。那么，P 是 S 中的 0-跳跃模式，当且仅当 P 是一个频繁闭图模式，P 是 S 中的 1-跳跃模式，当且仅当 P 是一个极大频繁图模式。

证明 设 P 是 S 中的任意一个 0-跳跃模式。根据 δ-跳跃模式的定义，可知 $JV_S(P) > 0$。如果 S 中存在 P'，满足 $P \subset P'$ 并且 $\text{support}(P) = \text{support}(P')$，那么 $JV_S(P) \leqslant D(P, P') = 1 - \frac{\text{support}(P')}{\text{support}(P)} = 0$，这与 $JV_S(P) > 0$ 相矛盾。因此，P 是一个频繁闭图模式。

设 P 是 S 中的任意一个频繁闭图模式。根据闭图模式定义，S 中不存在 P' 满足 $P \subset P'$ 并且 $\text{support}(P) = \text{support}(P')$。因此，对 P 的任意真超图 P'，都有 $D(P, P') = 1 - \frac{\text{support}(P')}{\text{support}(P)} > 0$。根据跳跃值的定义，可知 $JV_S(P) > 0$。再根据 δ-跳跃模式的定义，可知 P 是 S 中的一个 0-跳跃模式。

设 P 是 S 中的任意一个 1-跳跃模式。根据 δ-跳跃模式的定义，可知 $JV_S(P) > 1$。如果 S 中存在 P' 满足 $P \subset P'$，那么 $JV_S(P) \leqslant D(P, P') = 1 - \frac{\text{support}(P')}{\text{support}(P)} < 1$，这与 $JV_S(P) > 1$ 相矛盾。因此，P 是一个极大频繁图模式。

设 P 是 S 中的任意一个极大频繁图模式。根据极大频繁图模式的定义，S 中不存在 P' 满足 $P \subset P'$。根据跳跃值的定义，可知 $JV_S(P) = +\infty$。再根据 δ-跳跃模式的定义，可知 P 是 S 中的一个 1-跳跃模式。 □

引理 2.3.3 说明: 频繁闭图模式和极大频繁图模式都是频繁图模式集合中特殊的跳跃模式。因此，本章给出的算法也可以直接用来挖掘频繁闭图模式和极大频繁图模式。

根据上面的定义，从图数据库中挖掘代表模式问题可以定义如下:

输入: 图数据库 DB，最小支持度阀值 min_sup，距离阀值 δ。

输出: 一个代表模式集合 RS。而且，对任意一个频繁图模式 P，RS 中都存在一个代表模式 R，R 能 δ-覆盖 P。

优化目标: 最小化代表模式的数量 $|RS|$。

定理 2.3.1 挖掘代表模式问题是 NP-hard 问题。

证明 通过把最小集合覆盖问题在多项式时间内规约到挖掘代表模式问题，我们来证明该定理。

设 $SCP = (E, S)$ 是最小集合覆盖问题的任意一个实例。其中，$E = \{e_1, e_2, \ldots, e_n\}$，$S = \{S_1, S_2, \ldots, S_K\}$。$SCP$ 能在多项式时间内转化为挖掘代表模式问题的一个实例的步骤如下:

1. 把 E 中的每个元素看作不同的边标号，首先构造一个含有所有边标号 $\{e_1, e_2, \ldots, e_n\}$ 的图 G，放入数据库 DB。图 4 显示了 G 的结构，G 中所有顶点

的标号都相同（未标出），所有边的标号彼此都不相同。

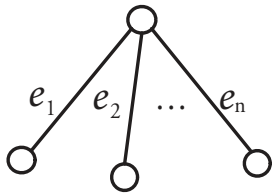


图 4　对应 $\{e_1, e_2, \ldots, e_n\}$ 的图

2. 对 S 中的每个集合 $S_i = (e_i^1, e_i^2, \ldots, e_i^{n_i})$，构造一个图（图 5），放入数据库 DB。

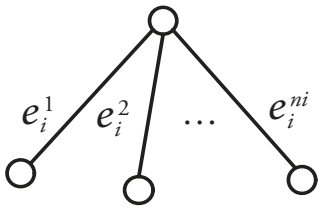


图 5　对应 $\{e_i^1, e_i^2, \ldots, e_i^{n_i}\}$ 的图

3. 对 E 中的每个元素 e_i，如果数据库 DB 中标号为 e_i 的边的出现次数小于 $K+1$，构造仅有一条边（标号为 e_i）的图，放入数据库 DB，直到 DB 中标号为 e_i 的边的出现次数等于 $K+1$。

显然，上面的构造过程可以在多项式时间内完成。设图数据库是 DB，最小支持度阀值是 $\frac{1}{|DB|}$，距离阀值是 $\delta=\frac{K-1}{K}$。我们得到了挖掘代表模式问题的一个实例，用 $GPSP$ 代表该实例。

现在我们证明，$GPSP$ 的最优解能在多项式时间内转化成 SCP 的最优解。

S 中的每个集合都对应一个图模式，而且在数据库 DB 中的支持度至少是 $\frac{2}{|DB|}$。设 RP 是 S 中所有集合对应的图模式集合。E 中每个元素对应 1-边图模式，而且在 DB 中的支持度等于 $\frac{K+1}{|DB|}$。现在，用 MP 表示图 4 所示的图模式。假定 RS^* 是 $GPSP$ 的最优解。显然，MP 一定在 RS^* 中，因为 MP 只能被自己 δ-覆盖。

如果 $|RS^*| = 1$，则 $RS^* = \{MP\}$。因为 MP 能覆盖任意的 1-边图模式 P（P 的支持度是 $\frac{K+1}{|DB|}$），所以 $D(MP, P) = 1 - \frac{\text{support}(MP)}{\text{support}(P)} \leqslant \delta = \frac{K-1}{K}$。容易导出 $\text{support}(MP) \geqslant \frac{2}{|DB|}$。因此，$S$ 中的某个集合 S_i 对应 MP。把 MP 中每条边的标号转换成对应的元素，可得到 S_i。显然，$\{S_i\}$ 一定是 SCP 的最优解。

如果 $|RS^*| > 1$，假设 $RS^* = \{MP, R_1, R_2, \ldots, R_m\}$.

现用反证法证明:$\text{support}(MP) = \frac{1}{|DB|}$。假定 $\text{support}(MP) \geqslant \frac{2}{|DB|}$，对 DB 中的任意图模式 P，有 $\text{support}(P) \leqslant \frac{K+1}{|DB|}$，$D(MP, P) = 1 - \frac{\text{support}(MP)}{\text{support}(P)} \leqslant \delta$。因此，$\{MP\}$

是最优解，这与 $|RS^*| > 1$ 相矛盾。

现用反证法证明: 对 RS^* 中任意一个 R_i，我们有 $\text{support}(R_i) \geqslant \frac{2}{|DB|}$。假定 $\text{support}(R_i) = \frac{1}{|DB|}$，因为 $R_i \subseteq MP$ 并且 $\text{support}(MP) = \frac{1}{|DB|}$，所以任何被 $R_i\delta$-覆盖的图模式都能被 MP δ-覆盖。我们可以从 RS^* 中删除 R_i，得到一个更小的代表模式集合，这与 RS^* 是最优解相矛盾。

现证明: 对任何 1-边图模式 P，RS^* 中都存在一个 R_i，使得 R_i δ-覆盖 P。因为 $\text{support}(P) = \frac{K+1}{|DB|}$，并且 $\text{support}(MP) = \frac{1}{|DB|}$，$D(MP, P) = 1 - \frac{\text{support}(MP)}{\text{support}(P)} > \delta$。因此，$MP$ 不能 δ-覆盖 P。因为 RS^* 能 δ-覆盖所有频繁图模式，所以对任何 1-边图模式 P，RS^* 中都存在一个 R_i，使得 R_i δ-覆盖 P。

现在根据 RS^* 构造 SCP 的一个解，步骤如下:(1) 把 R_i 中每个边标号转换成对应的元素，从 R_i 可以构造一个集合 ES_i。显然，从 $\{R_1, R_2, \ldots, R_m\}$，我们可以得到一个集合族 $ES = \{ES_1, ES_2, \ldots, ES_m\}$，该过程可以在多项式时间内完成。(2) 对 ES 中的每个 ES_i，因为 $\text{support}(R_i) \geqslant \frac{2}{|DB|}$，我们总能发现 S 中的一个集合 S_i'，满足 $ES_i \subseteq S_i'$。对 ES 中的每个 ES_i，扫描 S 中的每个集合 S_i'，如果 $ES_i \subseteq S_i'$，用 S_i' 代替 ES_i。最后，我们从 $ES = \{ES_1, ES_2, \ldots, ES_m\}$ 可以得到集合族 $S' = \{S_1', S_2', \ldots, S_m'\}$，该过程可以在多项式时间内完成。

现证明 $S' = \{S_1', S_2', \ldots, S_m'\}$ 是 SCP 的一个解。因为对任何 1-边图模式 P，RS^* 中都存在某个 R_i 能 δ-覆盖 P，所以 S' 一定能覆盖 E 中所有元素。

现用反证法证明: $S' = \{S_1', S_2', \ldots, S_m'\}$ 是 SCP 的最优解。如果存在一个比 S' 更好的解 S^*，通过把 S^* 中的每个集合转换成对应的图模式，再加上图模式 MP，我们可以得到 $GPSP$ 的一个解，并且这个解优于 RS^*，这与 RS^* 是 $GPSP$ 的最优解相矛盾。

因为最小集合覆盖问题是 NP-hard 问题 [40]，最小集合覆盖问题能在多项式时间内规约到挖掘代表模式问题，挖掘代表模式问题的最优解能在多项式时间内转换成最小集合覆盖问题的最优解，因此，挖掘代表模式问题也是 NP-hard 问题。 □

2.4 挖掘代表模式的算法

本节给出挖掘代表模式的三个算法。第一个算法，RP-FP，从频繁闭图模式集合中计算一个代表模式集合。第二个算法，RP-GD，直接从图数据库中挖掘一个代表模式集合。第三个算法，RP-Leap，挖掘一个近似覆盖所有频繁闭图模式的代表模式集合。

2.4.1　RP-FP 算法

2.4.1.1　算法描述

RP-FP 从频繁闭图模式集合中计算一个代表模式集合，包含四步。第一步，RP-FP 挖掘所有频繁闭图模式和所有 δ-跳跃模式。第二步，RP-FP 删除被 δ-跳跃模式覆盖的频繁闭图模式。第三步，RP-FP 在剩余的图模式上搜集完整的覆盖信息。第四步，RP-FP 使用贪心策略求解能覆盖剩余图模式的代表模式集合。

在第一步中，RP-FP 只需要使用 CloseGraph [146] 挖掘所有的频繁闭图模式，而不需要使用 gSpan [145] 挖掘所有的频繁图模式。这是因为频繁闭图模式的数量可能比频繁图模式的数量少很多，而且，覆盖所有频繁闭图模式的代表模式集合也一定能覆盖所有的频繁图模式，见引理 2.4.1。

引理 2.4.1　如果一个代表模式集合 RS 能覆盖所有的频繁闭图模式，那么 RS 也一定能覆盖所有的频繁图模式。

证明　设 P 是任意一个非闭的频繁图模式，那么，至少存在一个频繁闭图模式 Q 满足 $P \subset Q$，并且 $\text{support}(P) = \text{support}(Q)$。因为 RS 能覆盖所有的频繁闭图模式，所以 RS 中至少存在一个代表模式 R 满足 $Q \subseteq R$ 和 $D(Q,R) = 1 - \frac{\text{support}(R)}{\text{support}(Q)} \leqslant \delta$。我们有 $P \subset R$ 并且 $D(P,R) = 1 - \frac{\text{support}(R)}{\text{support}(P)} \leqslant \delta$。因此，$R$ 能覆盖 P。因为 P 是任意的非闭频繁图模式，所以 RS 能覆盖所有的频繁图模式。　□

现在，一个关键的问题是如何得到所有的 δ-跳跃模式。如果我们根据定义，直接从闭图模式集合中计算 δ-跳跃模式，必然涉及大量的子图同构测试，从而导致效率很低。通过分析 CloseGraph 算法，我们发现 δ-跳跃模式的计算可以嵌入 CloseGraph 算法中，并不需要执行额外的子图同构测试。

在 CloseGraph 算法中，为了测试一个频繁子图 g 是否是闭的，算法需要得到 g 的所有多一条边的超图的支持度。如果存在 g 的多一条边的超图 g' 满足 $\text{support}(g) = \text{support}(g')$，$g$ 被算法判定是非闭的。因此，当 CloseGraph 算法测试一个频繁子图 g 是否是闭的时候，我们可以附带地计算 g 的跳跃值，因为 g 的所有多一条边的超图的支持度已经被得到。如果 g 的跳跃值大于 δ，那么我们就标记 g 为 δ-跳跃模式。通过简单地修改 CloseGraph，我们很容易得到所有的 δ-跳跃模式。与最初的 CloseGraph 算法相比，额外增加的计算代价是可以忽略的。

在第二步中，RP-FP 把每个 δ-跳跃模式作为一个代表模式，删除被 δ-跳跃模式覆盖的图模式，得到不能被 δ-跳跃模式覆盖的图模式集合。这是因为，每个 δ-跳跃模式只能被自己覆盖，如定理 2.4.1 所示。

定理 2.4.1　设 F 是所有频繁闭图模式集合，RS 是能覆盖 F 的任意一个代表模式集合（即 $\forall P \in F$，$\exists R \in RS$ 满足 $P \subseteq R$，并且 $D(P,R) \leqslant \delta$），那么，RS 一定包

括 F 中的所有的 δ-跳跃模式。

证明 设 P 是 F 中的任意一个 δ-跳跃模式。根据 δ-跳跃模式的定义，P 在 F 中的跳跃值大于 δ。也就是说，P 与 F 中的所有真超图的最小 Jaccord 距离大于 δ。根据 δ-覆盖的定义，P 不能被 F 中的任何真超图 δ-覆盖。因此，对任意一个能覆盖 F 的代表模式集合 RS，RS 一定包括 P。因为 P 是 F 中的任意一个 δ-跳跃模式，所以 RS 一定包括 F 中的所有 δ-跳跃模式。 □

根据定理 2.4.1 可知，频繁闭图模式集合 F 中的 δ-跳跃模式必须被选作代表模式。假设 F^δ 是 F 中所有的 δ-跳跃模式集合。因为 F^δ 中的图模式都是代表模式，所以被 F^δ 覆盖的图模式可以被安全地删除（即 $\forall P \in F$，如果 $\exists R \in F^\delta$ 满足 $P \subseteq R$，并且 $D(P,R) \leqslant \delta$，那么从 F 中删除 P）。

设 E 是被 F^δ 覆盖的图模式集合，$\overline{E}$ 是不能被 F^δ 覆盖的剩余图模式集合。我们有 $F = E + \overline{E}$，而且 $F^\delta \subseteq E$。$\overline{E}$ 的大小可能比 F 小很多。在实验中，我们发现 $\overline{E}$ 只是 F 的 20%。

在第三步中，RP-FP 在剩余的图模式集合 $\overline{E}$ 上搜集完整的覆盖信息。对 $\overline{E}$ 里的每个图模式 P，RP-FP 搜集所有能覆盖 P 的图模式。

在第四步中，RP-FP 计算能覆盖 $\overline{E}$ 的最小代表模式集合。把 $\overline{E}$ 里的每个图模式看作一个元素，把 $F - F^\delta$ 中每个图模式的覆盖集看作一个集合，我们可得到一个最小集合覆盖问题。RP-FP 使用求解最小集合覆盖的近似算法 [126] 来解该问题。

算法 1 给出了完整的 RP-FP 算法。简单解释一下算法 RP-FP: 第 1 行，挖掘频繁闭图模式集合 F 和 δ-跳跃模式集合 F^δ。第 2-10 行，得到不能被 F^δ 覆盖的图模式集合 $\overline{E}$。第 11-17 行，在 $\overline{E}$ 上搜集完整的覆盖信息。第 18-24 行，使用贪心算法计算覆盖 $\overline{E}$ 的最小代表模式集合。

2.4.1.2 算法分析

在第一步中，RP-FP 的运行时间等于 CloseGraph 算法的挖掘时间。在第二步中，RP-FP 需要执行 $|F^\delta| \times |F|$ 个子图同构测试。在第三步中，RP-FP 需要执行 $|\overline{E}| \times |F - F^\delta|$ 个子图同构测试。如果 RP-FP 不利用 δ-跳跃模式，RP-FP 需要执行 $|F|^2$ 个子图同构测试。显然，$|F^\delta| \times |F| + |\overline{E}| \times |F - F^\delta| < |F|^2$。在实验中，我们发现 $|F^\delta| \times |F| + |\overline{E}| \times |F - F^\delta|$ 小于 $|F|^2$ 的 $\frac{1}{3}$。在第四步中，RP-FP 的时间复杂性是 $O(\sum_{R \in F - F^\delta} |\mathrm{Set}_\delta(R)|)$ [40]。

根据贪心算法对最小集合覆盖问题的近似比 [40]，容易得到 RP-FP 的近似比。

定理 2.4.2 设 F 是频繁闭图模式集合，F^δ 是 F 中的 δ-跳跃模式集合，RS 是算法 RP-FP 输出的代表模式集合，RP^* 是覆盖所有频繁图模式的最小代表模式集合（最

Algorithm 1: RP-FP 算法

```
Input: 图数据库 D，最小支持度 min_sup，距离阀值 δ，
Output: 一个代表模式集合
使用 CloseGraph 算法产生闭频繁图模式集合 F 和 δ-跳跃模式集合 F^δ(w.r.t. min_sup)；
for F^δ 里的每个 δ-跳跃模式 R do
    输出 R；
    for F 里的每个图模式 P do
        if R 能 δ-覆盖 P then
            把 P 放入集合 E 中；/* E 存放所有能被 F^δ 覆盖的图模式 */
E = F − E；/* E 存放不能被 F^δ 覆盖的图模式 */
for E 里的每个图模式 P do
    for F − F^δ 里的每个候选代表模式 R do
        if R 能 δ-覆盖 P then
            把 P 放入 Set_δ(R)；/* Set_δ(R) 存放能被 R 覆盖的图模式 */
while E ≠ Φ do
    从 F − F^δ 里选择一个代表模式 R，使得 |Set_δ(R)| 被最大化；
    for Set_δ(R) 里的每个图模式 P do
        从 E 和其他的 Set_δ(R')(R' ∈ F − F^δ) 中删除 P；
    输出 R；
```

优解)。那么，$|RP| \leqslant |RP^*| \times \frac{H(\max\limits_{R\in F-F^\delta}|\mathrm{Set}_\delta(R)|)}{1+(H(\max\limits_{R\in F-F^\delta}|\mathrm{Set}_\delta(R)|)-1)\times r}$，其中，$H(n)=\sum\limits_{k=1}^{n}\frac{1}{k}$，$r=\frac{|F^\delta|}{|RP|}$。

证明　设 RP-FP 在第四步执行贪心选择时，输出 s 个代表模式。假定覆盖 $\overline{E}$ 的最小代表模式数是 s^*，根据贪心算法的近似比 [40]，$s \leqslant s^* \times H(\max\limits_{R\in F-F^\delta}|\mathrm{Set}_\delta(R)|)$，显然，$|RP| = |F^\delta| + s$。现在用 $H(m)$ 代表 $H(\max\limits_{R\in F-F^\delta}|\mathrm{Set}_\delta(R)|)$，$|RP^*| = |F^\delta| + s^* \geqslant |F^\delta| + \frac{s}{H(m)} = \frac{H(m)\times|F^\delta|+s}{H(m)} = \frac{H(m)\times r\times|RP|+(1-r)\times|RP|}{H(m)} = \frac{(1+(H(m)-1)\times r)\times|RP|}{H(m)}$。因此，$|RP| \leqslant |RP^*| \times \frac{H(m)}{1+(H(m)-1)\times r}$。　□

如果使用 RPglobal 算法 [141] 挖掘代表模式，近似比是 $H(\max\limits_{R\in F}|\mathrm{Set}_\delta(R)|)$ [141]。因为 $H(\max\limits_{R\in F-F^\delta}|\mathrm{Set}_\delta(R)|) \leqslant H(\max\limits_{R\in F}|\mathrm{Set}_\delta(R)|)$，RP-FP 的近似比优于 RPglobal 的近似比，这个结果已经被实验验证。现在用一个例子说明 RP-FP 优于 RPglobal。对例 2.3.1 给出的代表模式挖掘问题，如果使用 RPglobal，代表模式集合是 $\{P_6, P_4, P_5\}$；如果使用 RP-FP，代表模式集合是 $\{P_4, P_5\}$。可见，RP-FP 的解优于 RPglobal 的解。

2.4.2　RP-GD 算法

当频繁闭图模式数量较多时，RP-FP 算法效率低、可扩展性差。因此，需要设计一个更高效的算法，既能处理频繁闭图模式数量较多的情况，又能保证结果质量（产生的代表模式数量应与 RP-FP 产生的代表模式数量差不多)。本小节给出挖掘代表模式的另一个算法 RP-GD。RP-GD 在挖掘频繁闭图模式的过程中，同时挖掘代表模式。

RP-GD 采用了联机算法的思想。每当某个图模式 P 输出时，RP-GD 测试 P 是

否能被已有的代表模式覆盖。如果 P 不能被覆盖，RP-GD 使用贪心策略创建一个新的能覆盖 P 的代表模式。因此，RP-GD 只需要对所有频繁闭图模式进行一次顺寻扫描，就可产生一个代表模式集合。

采用上面的联机思想，需要解决两个关键的问题:(1) 如果已有的代表模式都不能覆盖当前图模式 P，那么该创建一个什么样的代表模式来覆盖 P？(2) 给定一个频繁闭图模式 P，怎样才能快速地判断 P 是否能被已有的代表模式覆盖？本小节先分别研究了第一个问题和第二个问题，然后在 2.4.2.1 节给出完整的算法，2.4.2.2 节给出算法的复杂性分析。

1. 创建一个新的代表模式。

任何图模式挖掘算法都会以某种特定的顺序输出图模式。这种特定的顺序可以被利用，从而创建新的代表模式。CloseGraph [146] 在图模式空间上执行深度优先搜索，以 DFScode 字典顺序递增地输出频繁闭图模式。因此，CloseGraph 访问每个频繁闭图模式 P 两次，第一次访问是从 P 的父亲到 P；第二次访问是在遍历 P 的所有后裔之后再回到 P。定理 2.4.3 可以保证，在完成 P 的第二次访问之后，所有能覆盖 P 的图模式已经被输出，在这之后输出的图模式不可能覆盖 P。

定理 2.4.3 设 P 是任意一个被 CloseGraph 输出的频繁闭图模式。在第二次访问 P（遍历完以 P 为根的子树后再回到 P）之后，所有能覆盖 P 的频繁闭图模式都已经被输出。

证明 根据文献 [145] 和 [146]，当使用 CloseGraph 挖掘图数据库时，频繁闭图模式按 DFSCode 字典顺序递增地输出: 以前输出图模式的 DSCode 小于以后输出图模式的 DSCode。设 Q 是任意一个能 δ-覆盖 P 的图模式，根据 δ-覆盖的定义，$P \subseteq Q$。设 P 的 DFSCode 是 $(e_0, e_1, \ldots, e_n)$。根据 DFSCode 的定义，Q 的 DFSCode，或者小于 $(e_0, e_1, \ldots, e_n)$，或者是 $(e_0, e_1, \ldots, e_n, e_{n+1}, \ldots, e_m)$ 这样的形式。如果 Q 的 DFSCode 小于 P 的 DFSCode，当 P 在第一次访问时，Q 就已经被输出。如果 Q 的 DFSCode 是 $(e_0, e_1, \ldots, e_n, e_{n+1}, \ldots, e_m)$，显然 Q 是 P 的后裔，在第二次访问 P（遍历完以 P 为根的子树后再回到 P）之后，Q 已经被输出。 □

目前，CloseGraph 算法是在第一次访问频繁闭图模式时就输出它们。我们可以适当地调整 CloseGraph 这种特定的输出顺序，而不影响挖掘结果的完整性和算法效率。我们只在频繁闭图模式第二次被访问时，才输出它们。改变后的输出顺序被称为**覆盖顺序**。

RP-GD 根据覆盖顺序，依次扫描每个频繁闭图模式 P，首先测试 P 是否是 δ-跳跃模式。根据 2.4.1.1 节中的描述可知，在 CloseGraph 算法中 δ-跳跃模式很容易被识别。如果 P 是 δ-跳跃模式，RP-GD 创建一个新的代表模式 P，因为 δ-跳跃模式只能被自己覆盖；否则，RP-GD 扫描已经创建的代表模式，看 P 是否能被它们覆盖。如

果 P 能被覆盖，RP-GD 继续处理下一个输出的图模式；否则，RP-GD 使用贪心策略创建一个新的覆盖 P 的代表模式。

如果当前图模式 P 既不是 δ-跳跃模式，又不能被已经创建的代表模式覆盖，那么称 P 是一个**贪心模式**，因为 RP-GD 需要使用贪心策略为 P 创建一个新的代表模式。根据定理 2.4.3，所有能覆盖 P 的模式都已经输出，因此，我们需要考虑这样的问题：当遇到一个贪心模式 P 时，如何从以前输出的图模式中为 P 选择一个代表模式？直觉上，我们应该选择一个既能覆盖 P 又能使后续覆盖最大化的代表模式 R，即最大化 $|\mathrm{Set}_\delta(R) - \mathrm{Set}_\delta(R_1) \cup \mathrm{Set}_\delta(R_2) \cup \ldots \cup \mathrm{Set}_\delta(R_n)|$，其中，$R_1, R_2, \ldots, R_n$ 是已经产生的代表模式。采用这种贪心策略，可以得出下面的近似比，见定理 2.4.4。

定理 2.4.4　设 F 是频繁闭图模式集合，RS 是根据上面描述的贪心策略而得到的代表模式集合，RS^* 是能覆盖 F 的最小代表模式集合（最优覆盖）。那么，$|RS| \leqslant |RS^*| \times \frac{1}{2} \times (\max\limits_{P \in F} |\mathrm{Set}_\delta(P)| + 1)$。

证明　假定根据上面描述的贪心策略而输出的代表模式依次是 R_1，R_2，…，R_n，其中，$n = |RS|$。现在，我们为每个代表模式分配代价 1，并且让第一次被覆盖的图模式平摊这个代价。

如果 P 第一次被 R_i 覆盖，那么 $\mathrm{cost}(P) = \frac{1}{|\mathrm{Set}_\delta(R_i) - \mathrm{Set}_\delta(R_1) \cup \mathrm{Set}_\delta(R_2) \cup \ldots \cup \mathrm{Set}_\delta(R_{i-1})|}$，而且，$|RS| = \sum \mathrm{cost}(P)$。分配到最优覆盖 RS^* 的代价是 $\sum\limits_{R \in RS^*} \sum\limits_{P \in \mathrm{Set}_\delta(R)} \mathrm{cost}(P)$。因为 RS^* 能覆盖所有频繁闭图模式，所以 $|RS| = \sum \mathrm{cost}(P) \leqslant \sum\limits_{R \in RS^*} \sum\limits_{P \in \mathrm{Set}_\delta(R)} \mathrm{cost}(P)$。

设 R 是 RS^* 中任意一个代表模式，$\mathrm{Set}_\delta(R)$ 中有 x 个贪心模式，y 个非贪心模式，那么，$x + y = |\mathrm{Set}_\delta(R)|$。设 x 个贪心模式根据输出顺序分别是 $P_1^i, P_2^i, \ldots, P_x^i$，再假设使用上面的贪心策略为 P_1^i 创建的代表模式是 $R_1^i (i = 1, 2, \ldots, x)$。因为有 x 个图模式第一次被 R 覆盖，根据贪心策略，至少有 x 个图模式第一次被 R_1^i 覆盖，因此，$\mathrm{cost}(P_1^i) \leqslant \frac{1}{x}$。类似地，有 $\mathrm{cost}(P_2^i) \leqslant \frac{1}{(x-1)}$，…，$\mathrm{cost}(P_x^i) \leqslant 1$ 和 $\mathrm{cost}(P_1^i) + \mathrm{cost}(P_2^i) + \ldots + \mathrm{cost}(P_x^i) \leqslant \sum\limits_{k=1}^{x} \frac{1}{k}$。对 $\mathrm{Set}_\delta(R)$ 中任意一个非贪心模式 P，因为至少存在一个图模式与 P 平摊代价，所以 $\mathrm{cost}(P) \leqslant \frac{1}{2}$，我们有 $\sum\limits_{P \in \mathrm{Set}_\delta(R)} \mathrm{cost}(P) \leqslant \sum\limits_{k=1}^{x} \frac{1}{k} + \frac{y}{2} \leqslant 1 + \frac{1}{2} \times (|\mathrm{Set}_\delta(R)| - 1) = \frac{1}{2} \times (|\mathrm{Set}_\delta(R)| + 1)$。因此，$|RS| = \sum \mathrm{cost}(P) \leqslant \sum\limits_{R \in RS^*} \sum\limits_{P \in \mathrm{Set}_\delta(R)} \mathrm{cost}(P) \leqslant |RS^*| \times \frac{1}{2} \times (\max\limits_{P \in F} |\mathrm{Set}_\delta(P)| + 1)$。　□

然而，如果 RP-GD 使用上面的贪心策略，当为贪心模式 P 选择新的代表模式时，RP-GD 必然涉及在 P 之后输出的图模式的覆盖测试。为使 RP-GD 高效可扩展，我们不是最大化后续覆盖，而是采用了更加松弛的贪心策略。

实验中，我们发现在大多数情况下，当为贪心模式 P 选择代表模式时，最大化后续覆盖的代表模式出现在以 P 为根的子树中。因此，我们根据下面三个贪心策略进行

贪心选择。**贪心选择策略 1:** 在以 P 为根的子树中选择能覆盖 P 的最大图模式。**贪心选择策略 2:** 在应用贪心选择策略 1 之后，如果存在多个候选代表模式，选择覆盖 P 的超图最多的代表模式。例如，在图 6 中，假定 P 是贪心模式，R_1 和 R_2 都能覆盖 P。尽管 R_1 和 R_2 大小相等，但根据组合原理，R_1 包含更多 P 的超图，因此，R_1 能覆盖更多 P 的超图，R_1 应该被选择。**贪心选择策略 3:** 在应用贪心选择策略 1 和 2 之后，如果还存在多个候选代表模式，选择与 P 距离最近的代表模式。每当遇到贪心模式 P 时，RP-GD 就使用上面的三个贪心策略为 P 选择一个代表模式。

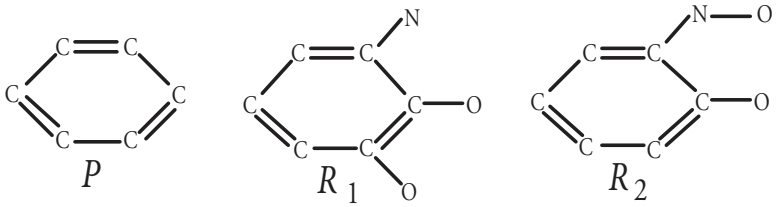


图 6　说明贪心选择策略 2 的例子

2. 搜索已有的代表模式。

在 RP-GD 算法中，一个关键步骤就是搜索已经产生的代表模式，测试当前模式是否能被已经产生的代表模式覆盖。检测一个代表模式是否能覆盖当前模式，可能涉及一个子图同构测试。因而，通过顺序扫描已有的代表模式而实现这个关键步骤效率会很低。本节给出三个启发式策略，尽可能地减少子图同构次数，提高这个关键步骤的效率。

给定当前图模式 P，如果我们能快速地判断某个代表模式能覆盖 P，就可以避免大量子图同构测试。实验中，我们发现: 由于相邻输出的图模式在结构和支持度上都比较接近，在上次搜索中成功发现的代表模式经常能覆盖下一个输出的图模式。启发式 2.1 基于这个发现而设计。

启发式 2.1　设 R 是上次搜索中成功发现的代表模式（或者是没发现而新创建的代表模式)。那么，当下一个图模式 P 输出时，先测试 R 是否能覆盖 P。

根据启发式 2.1，当搜索已经产生的代表模式时，我们采用了“最后成功最先测试”策略。每当成功发现一个代表模式或创建一个新的代表模式 R 时，就将 R 放在已有的代表模式集合的最前面。当下个频繁闭图模式 P 输出时，就可以先测试 R 是否能覆盖 P。如果 R 能覆盖 P，就可以避免大量子图同构测试。

当一个新的代表模式 R 创建时，如果能定位被 R 覆盖的将来输出的图模式，就可节省为那些将来输出的图模式而搜索代表模式的代价。为此，我们提出了下面两个启发式策略。在这之前，先给出引理 2.4.2，它说明在图模式搜索空间中路径的一个重要性质。

引理 2.4.2　设 $(P_1, P_2, \ldots, P_n)$ 是图模式搜索空间中以 P_1 为根的一条路径，假

定 R 能覆盖 P_n，那么，在这条路径上存在一个结点 P_k $(1 \leqslant k \leqslant n)$ 满足 R 能覆盖 $P_i(k \leqslant i \leqslant n)$，并且 R 不能覆盖 P_j $(1 \leqslant j < k)$。

证明　因为 $(P_1, P_2, \ldots, P_n)$ 是图模式搜索空间中以 P_1 为根的一条路径，所以 $P_1 \subset P_2 \subset \ldots \subset P_n$。因为 R 能覆盖 P_n，所以 $P_n \subseteq R$，因此，$P_1 \subset P_2 \subset \ldots \subset P_n \subseteq R$。根据 δ-覆盖定义，一定存在 $k(1 \leqslant k \leqslant n)$ 满足 R 能覆盖 $P_i(k \leqslant i \leqslant n)$，并且 R 不能覆盖 $P_j(1 \leqslant j < k)$。　□

当为图模式 P 成功发现一个旧代表模式或者创建一个新代表模式时，根据引理 2.4.2，对图模式搜索空间中从 P 到根路径上的图模式来说，就可以使用启发式 2.2 避免大量子图同构测试。

启发式 2.2　设 P 是当前输出的频繁闭图模式，当覆盖 P 的旧代表模式 R 被成功发现时（或者覆盖 P 的新代表模式被创建时），顺序检查从 P 到根路径上的每个图模式，直到发现一个不能被 R 覆盖的图模式。

如果 R 是当前图模式 P 的代表模式，那么，$P \subseteq R$，同时 R 也是从 P 到根路径上所有图模式的超图。因此，当使用启发式 2.2 沿着从 P 到根的路径反向检测每个图模式时，我们只需要比较它们的支持度，而不需要执行子图同构测试。启发式 2.2 也被称为“反向路径跟踪”策略。

假设 R 是当前图模式 P 创建的新的代表模式。除了从 P 到根路径上的图模式，R 还可能覆盖其他路径上的图模式（例如: 以后输出的频繁闭图模式）。是否可以枚举 R 的子结构并标记它们，当那些子结构输出时，测试 R 是否覆盖它们就可以不需要执行子图同构测试? 然而，这种方法并不可行。一方面，R 的某些子结构已经输出并被覆盖，枚举它们没有意义。另一方面，为了判断两个图模式是否相同，R 的子结构需要被转换成规范编码（例如 DFSCode），这等价于图同构。

然而，在某些情况下，我们可以保证 R 的某些子结构一定出现在将来的某个分枝中。而且，很容易导出它们的规范编码（DFSCode）。图 7 显然是图模式搜索空间中的一个分枝。假设 P_3 只能被自己覆盖。当一个新的代表模式 P_3 被创建时，P_2 还没有被遍历。然而，P_3 是 P_2 的超图，P_3 很可能覆盖 P_2。而且，从 P_3 的 DFSCode 很容易得到 P_2 的 DFSCode。因此，当 P_2 输出时，我们应首先测试 P_3 能否覆盖 P_2，如果 P_3 能覆盖 P_2，就可避免大量子图同构测试。启发式 2.3 就是为此而设计的。

启发式 2.3　如果图模式 $(e_0, e_1, \ldots, e_n, e_x, e_y)$ 是一个代表模式，$(e_0, e_1, \ldots, e_n, e_y)$ 是一个有效的 DFSCode，那么，$(e_0, e_1, \ldots, e_n, e_x, e_y)$ 很可能覆盖 $(e_0, e_1, \ldots, e_n, e_y)$。

根据启发式 2.3，当一个代表模式 (e_0 , e_1 , . . . , e_n , e_x , e_y) 被创建时，如果 (e_0, e_1, . . . , e_n, e_y) 是一个有效的 DFS Code，我们就把边 e_y 和连同 (e_0, e_1, . . . , e_n, e_x, e_y) 的支持度放入一个散列表中。当 (e_0, e_1, . . . , e_n) 被扩展成 (e_0, e_1, . . . , e_n, e_y) 时，首先测试边 e_y 是否在这个散列表中，如果在，根据它们的支持度不需要子图同构测试，

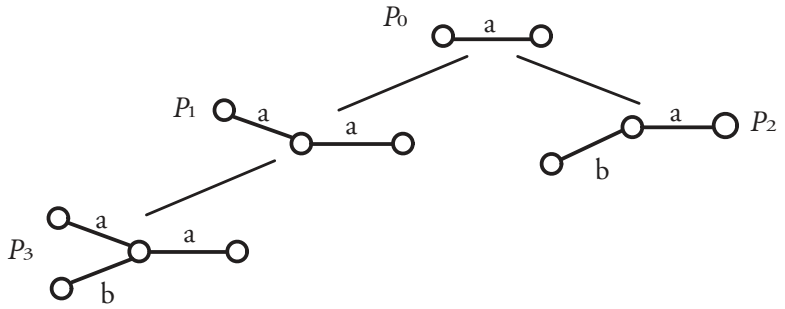


图 7　说明启发式策略 2.3 的例子

就可以判断 $(e_0, e_1, \ldots, e_n, e_y)$ 能否被 $(e_0, e_1, \ldots, e_n, e_x, e_y)$ 覆盖。启发式 2.3 也被称为“基于侄子代表覆盖”策略。

2.4.2.1　算法描述

本小节将 2.4.2 节的贪心策略和启发式策略集成到频繁子图挖掘过程中，给出了一个挖掘代表图模式的算法，RP-GD，如算法 2 所示。

Algorithm 2: RP-GD 算法

Input: 图数据库 D，最小支持度 min_sup，距离阀值 δ
Output: 一个代表模式集合 RS

```
1 从 D 中删除不频繁的边和结点;
2 S^1 = 所有频繁 1-边子图的 DFS 编码;
3 GS = ∅; /* 全局堆栈，用来跟踪图模式搜索空间中从根结点到当前结点路径上的所有频繁子图 */
4 RS = ∅;
5 for S^1 里的每个 DFS 编码 s do
6     调用 RP-GD-GetRepresentative(s, NULL, D, min_sup, δ, RS);
```

在初始化阶段，RP-GD 删除图数据库中不频繁的边和顶点，得到频繁边集合。对每个频繁 1-边子图（用 DFSCode 表示），RP-GD 调用算法 3 深度优先遍历图模式空间，产生代表模式。

现解释算法 3 的主要步骤。在第 1 行，测试当前的 DFS 编码 s 是否是 s 对应图模式的最小 DFS 编码。如果不是，裁剪以 s 为根的分枝。根据文献 [145]，只在最小 DFS 编码上执行最右扩展就可保证挖掘结果的完整性。在第 4 行，如果早期终止条件被满足 [146]，s 的所有后裔都不是闭图模式，可以安全地裁剪以 s 为根的分枝。后面，还将对早期终止做进一步解释。在第 8 行，扫描数据库 D，枚举 s 的所有多一条边的超图 $s \diamond_x e$。在第 9 行，计算 s 在频繁闭图模式集合 F 里的跳跃值，因为 s 的所有多一条边的超图的支持度在第 8 行就能获得。在第 10 行，把当前 DFS 编码 s 的最后一条边放入全局堆栈 GS。全局堆栈 GS 跟踪从根结点到当前结点路径上的所有频繁图模式。GS 里的每个入口对应一个图模式，含有如下信息:(1)DFSCode 的最后一条边;

Algorithm 3: RP-GD-GetRepresentative 算法

Input: 一个 DFSCode s，s 的父亲 p，图数据库 D，最小支持度 min_sup，距离阀值 δ，代表模式集合 RS

Output: 代表模式集合 RS

```
if s ≠ min(s) then
    返回;
if ∃cp，cp 是 p 的等价出现的孩子，并且根据字典 DFS 顺序 s > cp then
    返回; /* 早期终止 */
C = ∅;
扫描 D 一次，发现所有边 e 使得 s 能被 e 扩展成频繁子图 s ⋄_x e，把所有的 s ⋄_x e 放入一个临时集合 C;
计算 s 在闭频繁图模式集合 F 中的跳跃值 JV_F(s);
把 s 的最后一条边放入全局堆栈 GS; /* GS 跟踪图模式搜索空间中从根结点到当前结点路径上的所有频繁子图 */
if JV_F(s) = 0 then
    GS[top].covered = true; /* 标记非闭子图已经被覆盖 */
if JV_F(s) > 0 then
    for GS 里的每个入口 P(频繁子图 P) do
        if GS[P].covered = false，P 能被 s 覆盖，并且根据贪心选择策略 s 优于 GS[P].R_cand then
            GS[P].R_cand = s;
从 C 中删除不能从 s 最右扩展而得到的子图 s ⋄_x e;
根据字典 DFS 顺序，对 C 中的子图进行排序;
for s 的每个频繁最右扩展孩子 s ⋄_r e do
    调用 RP-GD-GetRepresentative(s ⋄_r e, s, D, min_sup, δ, RS);
if JV_F(s) > δ then
    创建一个新的代表模式 R_new = s，把 R_new 放入 RS;
    使用“反向路径跟踪”策略，用 R_new 试图覆盖 GS 中的每个图模式;
if GS[top].covered = false then
    使用“最后成功最先测试”和“基于侄子代表覆盖”策略在 RS 中搜索一个覆盖 s 的代表模式 R;
    if 不存在这样的代表模式 R then
        创建一个新的代表模式 R_new = GS[top].R_cand，把 R_new 放入 RS;
        使用“反向路径跟踪”策略，用 R_new 试图覆盖 GS 中的每个图模式;
    else
        使用“反向路径跟踪”策略，用 R 试图覆盖 GS 中的每个图模式;
从 GS 中弹出栈顶 GS[top];
```

(2) 支持度；(3) 覆盖标记 (covered)；(4) 候选代表模式 R_{cand}；(5) 跳跃值。全局堆栈 GS 的栈顶对应当前的图模式 s。如果 s 的跳跃值等于 0(在第 11 行)，说明 s 是一个非闭的图模式，标记 s 已经被覆盖。在第 15-19 行，扫描 GS 里的每个入口 P(图模式 P)，测试当前图模式 s 是否能覆盖 P；如果 s 能覆盖 P，并且根据 2.4.2 节的贪心选择策略，s 优于 P 的候选代表模式 $GS[P].R_{\text{cand}}$，那么用 s 替代 $GS[P].R_{\text{cand}}$。在第 23-25 行，对 s 的每个最右扩展孩子，递归调用算法 3，继续深度优先遍历过程。在第 26 行，在遍历完当前图模式 s 的后裔之后，测试 s 是否是 δ-跳跃模式。如果 s 是 δ-跳跃模式，创建一个新的代表模式 R_{new}=s，把 R_{new} 放入代表模式集合 RS；然后，应用“反向路径跟踪”策略，扫描 GS 里的每个入口 P(图模式 P)，如果 P 能被 R_{new} 覆盖，标记 P 被覆盖。在第 30 行，测试当前图模式 s 是否已经被覆盖。如果 s 还没有被覆盖，使用“最后成功最先测试”和“基于侄子代表覆盖”策略搜索已经存在的代表模式集合 RS，试图发现一个能覆盖 s 的代表模式 R。如果不存在这样的代表模式 R，说明 s 是一个贪心模式。在第 33 行，用 s 的候选代表模式 $s.R_{\text{cand}}$ 创建一个新的代表模式 $R_{\text{new}} = s.R_{\text{cand}}$，把 R_{new} 放入代表模式集合 RS。在第 34 行，再次应用“反向路径跟踪”策略。在第 39 行，在遍历完以 s 为根的分枝之后，从全局堆栈中弹出栈顶（s 的最后一条边)。

对所有频繁 1-边子图调用算法 3 之后，RS 里就含有覆盖所有频繁子图的代表模式。

在算法 3 的第 4 行应用早期终止时，我们使用了基于桥的裁剪方法 [135]。该方法的优点是不需要执行失败检测就能保证挖掘结果的完整性，该方法要求等价出现的前边必须是图中的桥，只有当扩展的前边是桥时，我们才记录扩展出现的次数。因此，在遍历图模式空间之前，我们需要扫描图数据库，发现并标记数据库中所有图的桥。这个预处理过程可以在多项式时间内完成，与挖掘代价相比，增加的额外代价可以忽略不计。

2.4.2.2 算法复杂性分析

本节分析算法 RP-GD 的时间复杂性。除了挖掘图数据库固有的代价，RP-GD 最主要的计算代价在于算法 3 的第 31 行，测试当前模式是否能被已经创建的代表模式覆盖。在下面的分析中，我们假定 RP-GD 不使用 2.4.2 节的启发式策略。启发式策略对 RP-GD 效率的改善在实验中给出。

给定最小支持度 min_sup 和距离阀值 δ，假定有 $|F|$ 个频繁闭图模式，有 $|F^{\delta}|$ 个 δ-跳跃模式，算法 RP-GD 输出 $|RS|$ 个代表模式。显然，贪心模式数量是 $|RS| - |F^{\delta}|$，在所有频繁闭图模式中，$|F|-|RS|$ 个图模式能被已有的代表模式覆盖。

假定在 RP-GD 执行过程中的某个时刻，RS 中有 m 个代表模式。如果当前模式能被 RS 中的代表模式覆盖，在算法 3 的第 31 行大约执行 $\frac{1}{2}m$ 个子图同构测试；否则，第 31 行需要执行 m 个子图同构测试。

假定平均每隔 $\frac{|F|-|RS|}{|RS|}$ 个频繁闭图模式创建一个新的代表模式放入 RS。对 $|F|-|RS|$ 个能被已有代表模式覆盖的频繁闭图模式来说，算法 3 的第 31 行大约执行 $\frac{|F|-|RS|}{|RS|}\times\frac{1}{2}\times(1+2+\ldots+|RS|)=\frac{|F|-|RS|}{|RS|}\times\frac{1}{2}\times\sum_{k=1}^{|RS|}k$ 个子图同构测试。

假定平均每隔 $\frac{|F^\delta|}{|RS|-|F^\delta|}$ 个 δ-跳跃模式创建一个覆盖贪心模式的新代表模式放入 RS。对第一个贪心模式，第 31 行需要执行 $\frac{|F^\delta|}{|RS|-|F^\delta|}$ 个子图同构测试。对第二个贪心模式，第 31 行需要执行 $2\times\frac{|F^\delta|}{|RS|-|F^\delta|}+1$ 个子图同构测试。对最后一个贪心模式，第 31 行需要执行 $(|RS|-|F^\delta|)\times\frac{|F^\delta|}{|RS|-|F^\delta|}+(|RS|-|F^\delta|-1)$ 个子图同构测试。因此，对所有贪心模式来说，第 31 行总共需要执行 $\frac{|F^\delta|}{|RS|-|F^\delta|}\times(1+2+\ldots+(|RS|-|F^\delta|))+(1+2+\ldots+(|RS|-|F^\delta|-1))=\frac{|F^\delta|}{|RS|-|F^\delta|}\times\sum_{k=1}^{|RS|-|F^\delta|}k+\sum_{k=1}^{|RS|-|F^\delta|-1}k$ 个子图同构测试。

因此，在算法 3 的第 31 行，总共需要执行大约 $\frac{|F|-|RS|}{|RS|}\times\frac{1}{2}\times\sum_{k=1}^{|RS|}k+\frac{|F^\delta|}{|RS|-|F^\delta|}\times\sum_{k=1}^{|RS|-|F^\delta|}k+\sum_{k=1}^{|RS|-|F^\delta|-1}k$ 个子图同构测试。

设 δ-跳跃模式和代表模式的比值是 $r=\frac{|F^\delta|}{|RS|}$，代表模式和频繁闭图模式的比值是 $\lambda=\frac{|RS|}{|F|}$。我们有 $\frac{|F|-|RS|}{|RS|}\times\frac{1}{2}\times\sum_{k=1}^{|RS|}k+\frac{|F^\delta|}{|RS|-|F^\delta|}\times\sum_{k=1}^{|RS|-|F^\delta|}k+\sum_{k=1}^{|RS|-|F^\delta|-1}k=(\frac{(1-\lambda)\lambda}{4}+\frac{r(1-r)\lambda^2}{2})\times|F|^2+\frac{1+2r\lambda}{4}\times|F|\approx(\frac{(1-\lambda)\lambda}{4}+\frac{r(1-r)\lambda^2}{2})\times|F|^2$。

类似地，如果不使用 δ-跳跃模式，可以推导出在算法 3 的第 31 行大约执行 $(\frac{(1-\lambda)\lambda}{4}+\frac{\lambda^2}{2})\times|F|^2$ 个子图同构测试。如果 $r=\frac{9}{10}$，$\lambda=\frac{1}{3}$（在实验中观察到的比值），$(\frac{(1-\lambda)\lambda}{4}+\frac{r(1-r)\lambda^2}{2})\times|F|^2$ 大约是 $(\frac{(1-\lambda)\lambda}{4}+\frac{\lambda^2}{2})\times|F|^2$ 的一半。因此，δ-跳跃模式能使算法 RP-GD 加快至少一倍。而且，随着 λ 值的增加，δ-跳跃模式对性能的改善作用会更加明显。

在 2.4.2.1 节，我们分析了 RP-FP 需要执行 $|F^\delta|\times|F|+|\overline{E}|\times|F-F^\delta|$ 个子图同构测试。因为 $|\overline{E}|\geqslant|RS|-|F^\delta|$，可以证明 $|F^\delta|\times|F|+|\overline{E}|\times|F-F^\delta|\geqslant(\lambda r+\lambda(1-r)(1-\lambda r))\times|F|^2$。如果 $r=\frac{9}{10}$，$\lambda=\frac{1}{3}$，$(\frac{(1-\lambda)\lambda}{4}+\frac{r(1-r)\lambda^2}{2})\times|F|^2$ 不到 $(\lambda r+\lambda(1-r)(1-\lambda r))\times|F|^2$ 的 $\frac{1}{5}$。根据 RP-FP 和 RP-GD 的复杂性分析可知，在效率方面，RP-GD 极大地优于 RP-FP。

2.4.3　RP-Leap 算法

尽管 RP-GD 极大地优于 RP-FP，RP-GD 仍然依赖于 CloseGraph 算法。为了保证覆盖所有的频繁闭图模式，RP-GD 需要顺序地检查每个频繁闭图模式，显然这使得

RP-GD 的效率要低于 CloseGraph 的效率。由于 CloseGraph 算法中等价出现的裁剪条件过于严格，图模式搜索空间中的很多分枝即使在不产生 (或极少产生) 代表模式的情况下也不能被裁剪掉。

在实际应用中，如果能从图数据库中更高效地挖掘一个代表模式集合 RS，而且 RS 能覆盖绝大部分频繁图模式，也是非常有意义的。基于这个想法，本节给出了一个更高效的算法——RP-Leap，来挖掘一个近似覆盖频繁图模式的代表模式集合。RP-Leap 通过尽可能地跳过图模式搜索空间中那些不产生代表的分枝，来提高挖掘效率。

先介绍 RP-Leap 的主要思想。图 8 显示了图模式搜索空间（模式搜索树）中的一个分枝。该分枝中的每个结点都代表一个图模式。当向图模式 g 增加新边时，g 分别被扩展成 $g \diamond e_1, g \diamond e_2, \ldots, g \diamond e_n$。在 $g \diamond e_1$ 为根的分枝中，含有所有 $g \diamond e_1$ 的超图。在以 $g \diamond e_2$ 为根的分枝中，含有 $g \diamond e_2$ 的超图，但不含有 $g \diamond e_1$ 的超图。同理，在以 $g \diamond e_i$ 为根的分枝中，含有 $g \diamond e_i$ 的超图，但不含有 $g \diamond e_j$ $(j < i)$ 的超图。

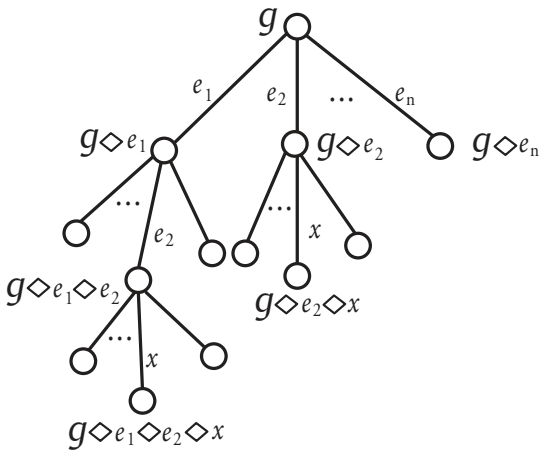


图 8　图模式搜索空间中的一个分枝

对以 $g \diamond e_2$ 为根的分枝中的任意一个图模式 $g \diamond e_2 \diamond x$，在 $g \diamond e_1$ 为根的分枝中很可能存在 $g \diamond e_2 \diamond x$ 的超图 $g \diamond e_1 \diamond e_2 \diamond x$。如果 g 和 $g \diamond e_1$ 在数据库中经常一起出现，$g \diamond e_2 \diamond x$ 和 $g \diamond e_1 \diamond e_2 \diamond x$ 也会经常一起出现，这就意味着 $g \diamond e_1 \diamond e_2 \diamond x$ 的支持度接近于 $g \diamond e_2 \diamond x$ 的支持度。

在 $g \diamond e_1$ 为根的分枝被遍历完成之后，就会有一个代表模式 R 能覆盖 $g \diamond e_1 \diamond e_2 \diamond x$。根据覆盖的定义，$R$ 是 $g \diamond e_1 \diamond e_2 \diamond x$ 的超图，因此，R 也是 $g \diamond e_2 \diamond x$ 的超图。如果 g 和 $g \diamond e_1$ 在数据库中经常一起出现，根据上面的分析可知，$\text{support}(g \diamond e_2 \diamond x) \approx \text{support}(g \diamond e_1 \diamond e_2 \diamond x)$。因为 $D(R, g \diamond e_2 \diamond x) = 1 - \frac{\text{support}(R)}{\text{support}(g \diamond e_2 \diamond x)} \approx 1 - \frac{\text{support}(R)}{\text{support}(g \diamond e_1 \diamond e_2 \diamond x)}$，$R$ 有很大的可能性覆盖 $g \diamond e_2 \diamond x$。因此，如果 g 和 $g \diamond e_1$ 在数据库中经常一起出现，当 $g \diamond e_1$ 为根的分枝被遍历完成时，我们就可以跳过以 $g \diamond e_2$ 为根的分枝。同理，我们也可以跳过以 $g \diamond e_1$ 的任何右兄弟结点为根的分枝。类似地，如果存在一条边 e_i，使得 g 和 $g \diamond e_i$ 经常一起

出现，当 $g \diamond e_i$ 为根的分枝被遍历完成时，我们也可以跳过以 $g \diamond e_j$ （$j > i$）为根的分枝。

现在需要解决的一个问题是如何度量 g 和 $g \diamond e_1$ 经常一起出现? 前面定义的 Jaccard 距离可以使用,现在称 Jaccard 距离为基于支持度的距离,用 D_{supp} 表示。为了更精确地度量这个条件,下面定义了基于嵌入的距离,用 D_{emb} 表示。如果 $D_{\text{supp}}(g, g \diamond e_1) \leqslant \epsilon$（或 $D_{\text{emb}}(g, g \diamond e_1) \leqslant \epsilon$），就表示 g 和 $g \diamond e_1$ 经常一起出现。δ 是用户给定的参数，本章称 δ 为跳跃阀值。

定义2.4.1 (基于嵌入的距离) 设 P_1 是 P_2 的真超图，P_1 和 P_2 基于嵌入的距离定义为 $D_{\text{emb}}(P_1, P_2) = 1 - \frac{\text{embedding}(P_1)}{\text{embedding}(P_2)}$。其中，embedding($P_1$) 是 P_1 在数据库中的嵌入次数（在数据库所有图中的所有子图同构次数之和）。

算法 4 给出了完整的 RP-Leap 算法。

Algorithm 4: RP-Leap 算法

Input: 图数据库 D，最小支持度 min_sup，距离阀值 δ，跳跃阀值 ϵ
Output: 一个代表模式集合 RS

```
1 从 D 中删除不频繁的边和结点;
2 S^1 = 所有频繁 1-边子图的 DFS 编码;
3 GS = ∅; /* 全局堆栈，用来跟踪图模式搜索空间中从根结点到当前结点路径上的所有频繁子图 */
4 RS = ∅;
5 for S^1 里的每个 DFS 编码 s do
6 |   s.min_distance = 1; /* 初始化最大值 */
7 |   调用 RP-Leap-GetRepresentative(s, NULL, D, min_sup, δ, ϵ, RS);
```

RP-Leap 的框架类似于 RP-GD。然而，通过使用前面描述的跳跃搜索技术，RP-Leap 以丢失少量代表模式的代价，换取了在性能方面的极大改善。

在算法 RP-Leap-GetRepresentative(算法 5) 的第 1 行，在遍历以当前图模式 s 为根的分枝前,先使用跳跃搜索技术测试该分枝是否能被跳过。我们使用 p.min_distance 表示 p 和 p 的已经被枚举的孩子之间的最小距离 D_{supp}（或者 D_{emb}）。若 p.min_distance< ϵ，则说明存在 p 的一个已经被枚举的孩子 c 满足 c 和 p 经常一起出现。根据上面描述的跳跃思想，覆盖以 c 为根的分枝中图模式的代表模式也具有很大的可能性覆盖以 s 为根的分枝中的图模式，遍历以 s 为根的分枝中的图模式几乎不产生新的代表模式。当从 p 扩展到 s 时，覆盖以 c 为根的分枝中图模式的代表模式已经被得到，因此，可以跳过以 s 为根的分枝的遍历。在第 7 行，根据 p 和它的孩子 s 之间的距离 $D_{\text{emb}}(p, s)$（或 $D_{\text{supp}}(p, s)$）更新 p.min_distance。在第 13 行，对 s 的最右扩展孩子递归调用算法之前，初始化 s 和 s 的被枚举的孩子之间的最小距离 s.min_distance 等于 1。随着对 s 的孩子的遍历，s.min_distance 的值将会逐渐地减小。当 s.min_distance $< \epsilon$ 时，我们就可以跳过对 s 的剩余孩子分枝的遍历。在算法

Algorithm 5: RP-Leap-GetRepresentative 算法

Input: 一个 DFSCode s，s 的父亲 p，图数据库 D，最小支持度 min_sup，距离阀值 δ，跳跃阀值 ϵ，跳跃阀值 ϵ，代表模式集合 RS
Output: 代表模式集合 RS

```
if p! = NULL，并且 p.min_distance < ε then
    返回;
if s ≠ min(s) then
    返回;
p.min_distance = min(p.min_distance, D_supp(p, s));
把 s 的最后一条边放入全局堆栈 GS; /* GS 跟踪图模式搜索空间中从根结点到当前结点路径上的所有频繁子图 */
for GS 里的每个入口 P(频繁子图 P) do
    根据 RP-GD 的贪心选择策略，更新覆盖 P 的候选代表模式;
扫描 D 一次，发现 s 的所有频繁最右扩展孩子;
s.min_distance = 1; /* 初始化最大值 */
for s 的每个频繁最右扩展孩子 s ◇_r e do
    调用 RP-Leap-GetRepresentative(s ◇_r e, s, D, min_sup, δ, ε, RS);
if GS[top].covered = false then
    使用 "最后成功最先测试" 和 "基于侄子代表覆盖" 策略在 RS 中搜索一个覆盖 s 的代表模式 R;
    if 不存在这样的代表模式 R then
        创建一个新的代表模式 R_new = GS[top].R_cand，把 R_new 放入 RS;
        使用 "反向路径跟踪" 策略，用 R_new 试图覆盖 GS 中的每个图模式;
    else
        使用 "反向路径跟踪" 策略，用 R 试图覆盖 GS 中的每个图模式;
从 GS 中弹出栈顶 GS[top];
```

RP-GD 中提出的贪心选择策略（见 2.4.2 节）和启发式策略（见 2.4.2 节）也都能应用到 RP-Leap 算法中。

2.5 实验结果及分析

本节通过大量实验来考查算法的挖掘结果质量、执行效率、可扩展性、不同参数的作用，以及挖掘结果的可用性。

2.5.1 实验设置

我们从两个真实的化合物集合中导出了几个图集合。第一个化合物集合 (PTE) 用来评价化合物的毒性，含有 340 个化合物。第二个化合物集合 (AIDS) 用来测试化合物对艾滋病病毒的抑制作用，含有大约 44 000 个化合物。根据实验结果，AIDS 化合物可以被分成三类:CA (confirmed active)，CM(confirmed moderately) 和 CI(confirmed inactive)，其中，CA 含有 422 个化合物，CM 含有 1 081 个化合物，CI 含有剩余的化合物。

除了真实的图数据，我们也使用文献 [83] 中的图数据生成器生成了具有各种不同特征的图集合。有 6 个参数控制合成的图集合: (1)$|D|$ (合成图的数量)；(2)$|T|$ (合

成图的平均大小)；(3)$|L|$ (潜在频繁子图模式的数量)；(4)$|I|$ (潜在频繁子图模式的平均大小)；(5)$|V|$ (结点标号的数量)；(6)$|E|$(边标号的数量)。例如，合成的图集合用类似 “D1kV4E2I5T20L20” 的记号表示，D1kV4E2I5T20L20 代表参数设置为 $|D|$=1k，$|T|$=20，$|L|$=20，$|I|$=5，$|V|$=4，$|E|$=2 的一个合成图集合。

本章算法使用 C++ 语言实现，用带有-O3 优化选项的 g++ 编译。用于实验的计算机具有 PIV-4 3.2GHz CPU 和 512M 内存，运行 RedHat Linux 8.0 操作系统。我们使用文献 [135] 中的方法实现了 CloseGraph。为了与挖掘代表项集算法 RPglobal [141] 比较，我们实现了挖掘代表图模式的 RPglobal 算法。

2.5.2　RP-FP 和 RP-GD 的结果质量和效率

本节比较 CloseGraph，RPglobal，RP-FP，RP-GD 四个算法的输出图模式数量和运行时间。在 RP-GD 中将应用 δ-跳跃模式和 2.4.2 节给出的所有启发式策略。2.5.3 节将评价 δ-跳跃模式和启发式策略对 RP-GD 算法效率的改善。挖掘代表模式算法（RPglobal，RP-FP，RP-GD）总的运行时间包括两部分: (1) 挖掘频繁闭图模式的时间，这部分时间恰恰是 CloseGraph 的运行时间；(2) 产生代表模式的时间。对每个图集合，我们固定距离阀值 $\delta = 0.1$，变化最小支持度 min_sup。

这里，我们只给出有代表性的部分数据集合上的实验结果。图 9 显示了在 PTE 数据集上不同算法所输出的图模式数量和运行时间。图 10 显示了在合成数据集 D1kV4E2I5T20L40 上的实验结果。在这些实验中，如果一个算法不能在 30 分钟内完成它的任务，就终止算法的运行。因此，在某些图上，RPglobal 和 RP-FP 的实验结果是不完整的。在图 9(a) 和 10(a) 中，我们也显示了 δ-跳跃模式的数量。根据定理 2.4.1 可知，δ-跳跃模式数量是本章优化问题最优解的下界（即 δ-跳跃模式数量是覆盖所有频繁图模式的最小代表模式数量的下界）。

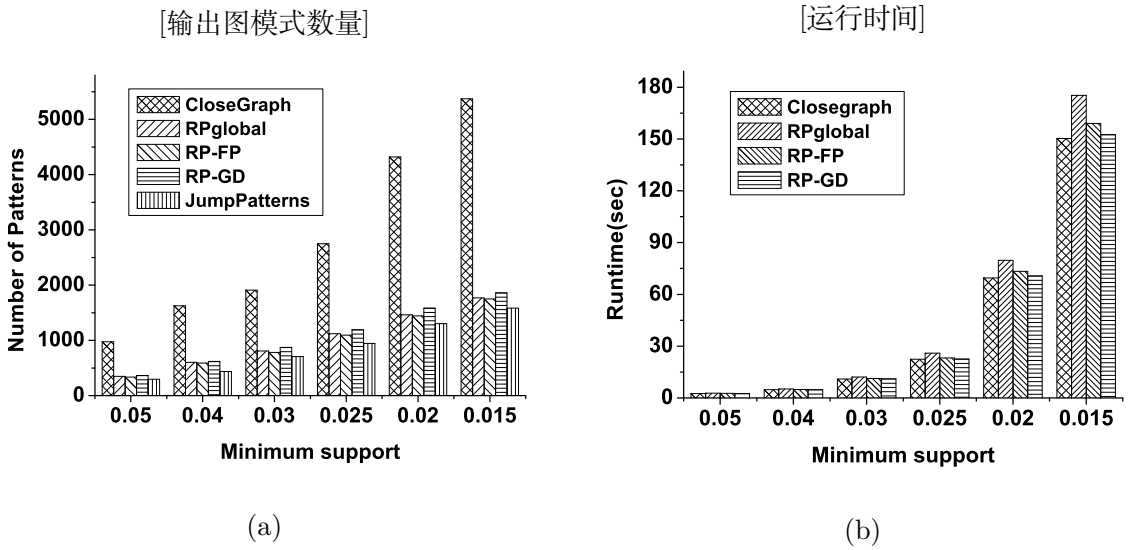


图 9　PTE 上的实验结果

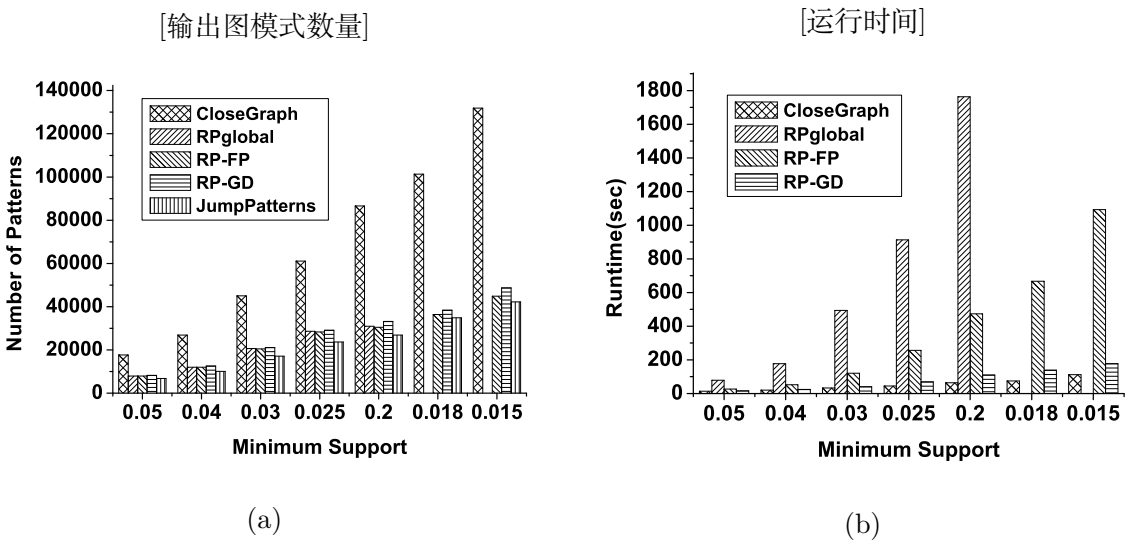


图 10　D1kV4E2I5T20L40 上的实验结果

从图 9(a) 和 10(a) 可以看出，代表模式数量只是在频繁闭图模式数量的 2/7 到 2/5 之间。RP-FP 产生的代表模式数量要稍微小于 RPglobal 产生的代表模式数量，这是因为 RP-FP 比 RPglobal 有更紧的近似比（见定理 2.4.2）。RP-GD 产生的代表模式数量要稍微多于 RP-FP 和 RPglobal 产生的代表模式数量，因为 RP-GD 没有近似比保证。然而，在所有情况下，RP-GD 产生的代表模式数量最多是 RP-FP 产生的代表模式数量的 120%。

当频繁闭图模式数量较少时，挖掘频繁闭图模式的时间支配总的运行时间，产生代表模式的时间只占总的运行时间很小的比例。如图 9 所示，当频繁闭图模式数量较少时，所有算法的运行时间相差不大。

当频繁闭图模式数量较多时，产生代表模式的时间支配总的运行时间。如图 10 中所示，当频繁闭图模式数量较多时，RP-FP，RPglobal，RP-GD 三个算法的运行时间差别很大。例如: 当最小支持度等于 2% 时，RPglobal 需要大约 1 800 秒才能产生代表模式，RP-FP 需要 500 多秒产生代表模式，而 RP-GD 只需要 100 多秒就能完成相同的任务。这些实验结果与 2.4.2.2 节给出的算法复杂性分析相一致。

2.5.3　δ-跳跃模式和启发式策略的作用

本节评价 δ-跳跃模式和 2.4.2 节给出的启发式策略的作用。根据 δ-跳跃模式和启发式策略的不同组合，我们实现了 RP-GD 算法的几个变体。表 1 显示了进行各种组合而得到的算法变体。

表 1　RP-GD 的变体

名称	描述
RP-GD-NJP	不使用 δ-跳跃模式和任何启发式策略
RP-GD-JP	在 RP-GD-NJP 中使用 δ-跳跃模式
RP-GD-H1	在 RP-GD-JP 中使用启发式 2.1
RP-GD-H2	在 RP-GD-JP 中使用启发式 2.2
RP-GD-H3	在 RP-GD-JP 中使用启发式 2.3
RP-GD	在 RP-GD-JP 中使用所有启发式策略

图 11(a) 显示了 RP-GD 算法的不同变体的运行时间（δ=0.1）。为了更清楚地显示 δ-跳跃模式和启发式策略的作用，图 11 (b) 只给出了产生代表模式的时间。

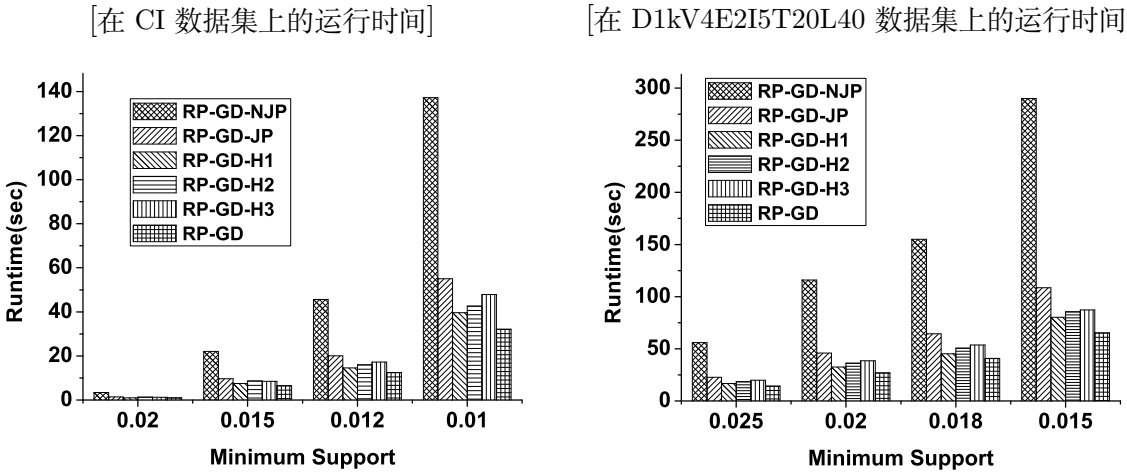


图 11　测试 RP-GD 的变体

在 2.4.2.2 节的算法复杂性分析中，我们显示了 δ-跳跃模式能极大地减少子图同

构次数，提高 RP-GD 算法的效率。实验结果与分析结果相一致。如图 11 所示，δ-跳跃模式能使 RP-GD 的性能提高 2-3 倍。

从图 11 也可以看出，每种启发式策略都能进一步改善 RP-GD 的性能。在所有启发式策略中，启发式 2.1（“最后成功最先测试” 策略）最有效，对 RP-GD 性能的改善最明显。当所有启发式策略一起使用时，可使 RP-GD 的性能得到最大的改善。因为在算法执行过程中某些频繁闭图模式可以根据多个启发式策略中的任何一个来确定是否已被覆盖，所以当所有启发式策略一起使用时，总的性能改善小于每种启发式策略单独使用时性能改善之和。最后，当 δ-跳跃模式和所有启发式策略一起使用时，RP-GD 的性能被提高 4-5 倍。

2.5.4 RP-Leap 的结果质量和效率

本节评价 RP-Leap 算法的结果质量和运行效率。因为 RP-Leap 不能保证覆盖所有频繁闭图模式，为了评价 RP-Leap 的结果质量，我们定义了下面的覆盖率。将一个挖掘代表模式算法的**覆盖率**定义为被代表模式覆盖的频繁闭图模式数量和所有频繁闭图模式数量的比值。显然，RP-FP 和 RP-GD 算法的覆盖率都是 100%。为了让 RP-Leap 与覆盖率为 100% 的算法比较，本节使用效率最高的 RP-GD 算法。现用 RP-Leap-support 表示在 RP-Leap 算法中使用了基于支持度的距离度量 D_{supp}，用 RP-Leap-Embedding 表示在 RP-Leap 算法中使用了基于嵌入的距离度量 D_{emb}。在下面的实验中，如无特别说明，固定 $\delta = 0.1$，ϵ=0.01。

图 12 显示了当最小支持度 min_sup 变化时，RP-GD，RP-Leap-Support，RP-Leap-Embedding 算法在 PTE 数据集上的覆盖率和运行时间。图 13 显示了在 CA 数据集上的覆盖率和运行时间。从图 12 和 13 可以看出，RP-Leap-Support 比 RP-GD 快一个数量级，RP-Leap-Embedding 比 RP-GD 快 2-3 倍。同时，RP-Leap-Embedding 的覆盖率超过 97%，RP-Leap-Support 的覆盖率超过 84%。我们也注意到，随着最小支持度的降低，RP-Leap-Embedding 的覆盖率保持相对稳定，而 RP-Leap-Support 的覆盖率在逐渐减小。原因如下: 在 RP-Leap 算法中，随着最小支持度的降低，基于支持度距离 D_{supp} 的裁剪条件比基于嵌入距离 D_{emb} 的裁剪条件更容易被满足。因此，当最小支持度降低时，RP-Leap-support 能跳过模式搜索空间中更多的分枝，从而更多的频繁闭图模式不能被覆盖，导致 RP-Leap-Support 的覆盖率逐渐减小。然而，即使当最小支持度非常低时，RP-Leap-Support 仍然具有很高的覆盖率。例如: 在 PTE 数据集上，当最小支持度等于 0.015 时，RP-Leap-Support 的覆盖率超过了 84%，而 RP-Leap-Support 比 RP-GD 快一个数量级还多。一般情况下，覆盖率越高，代表模式数量越多。因为 RP-GD 的覆盖率高于 RP-Leap-Embedding 的覆盖率，RP-Leap-

Embedding 的覆盖率高于 RP-Leap-Support 的覆盖率，所以 RP-GD 产生的代表模式数量多于 RP-Leap-Embedding 产生的代表模式数量，而 RP-Leap-Embedding 产生的代表模式数量多于 RP-Leap-Support 产生的代表模式数量。

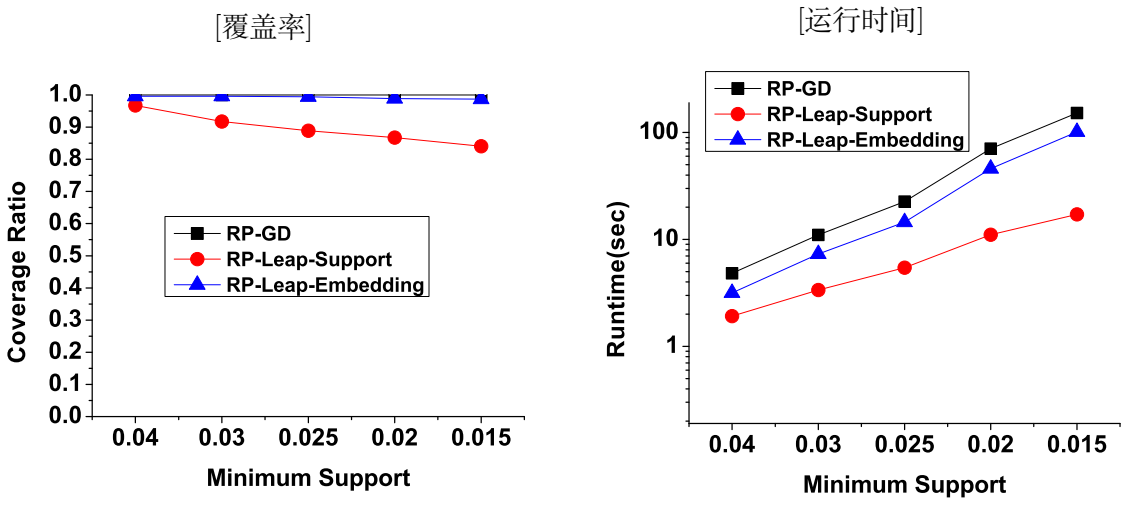


图 12　在 PTE 数据集上的实验结果

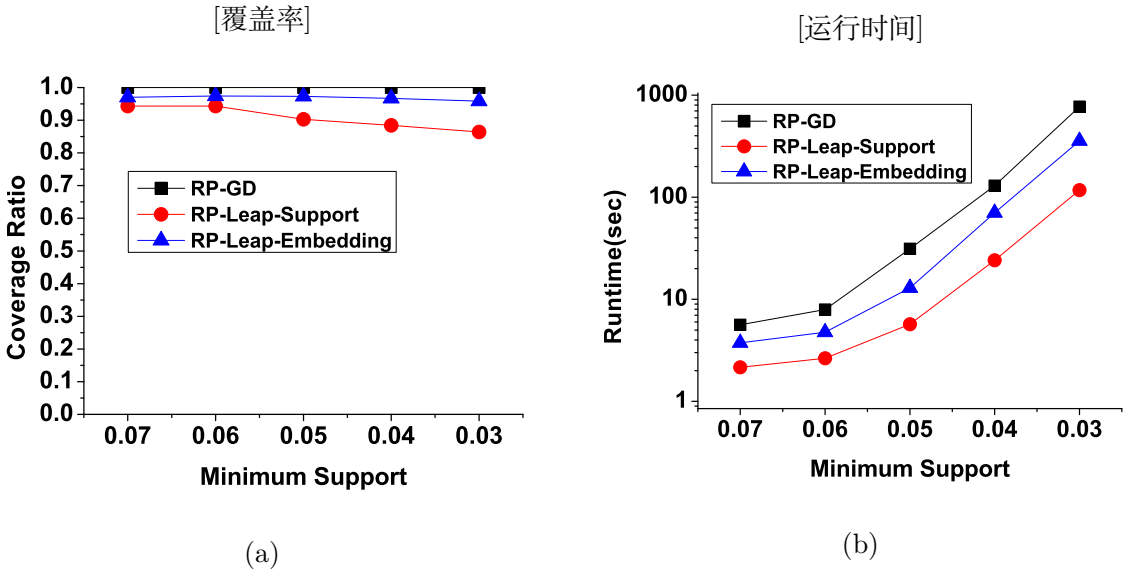


图 13　在 CA 数据集上的实验结果

下面测试跳跃阀值 ϵ 对结果质量和效率的影响。图 14 显示了当跳跃阀值 ϵ 变化时（固定 min_sup= 0.05），RP-Leap-Support 和 RP-Leap-Embedding 在 CA 数据集

上的覆盖率和运行时间。从图 14 可以看出，随着跳跃阀值 ϵ 的增加，算法的覆盖率在下降，而运行效率在改善。在实际应用中，ϵ 值的选择需要折中考虑。如果用户强调高覆盖率，那么选择低的 ϵ 值（例如 0.01）。反之，如果用户强调挖掘效率，那么选择高的 ϵ 值（例如 0.1）。

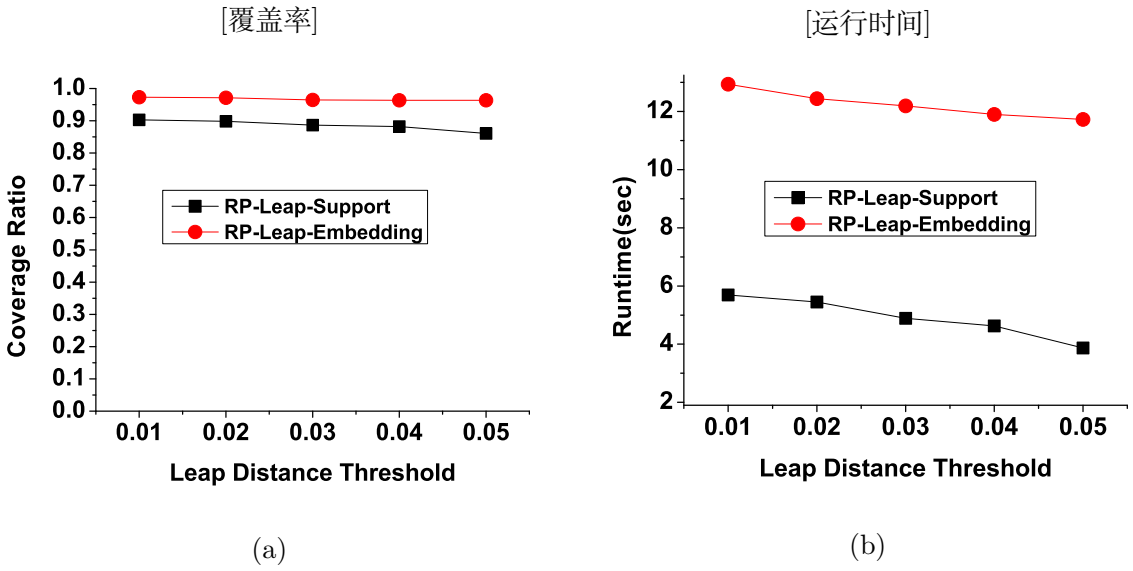


图 14　在 CA 数据集上变化跳跃阀值 ϵ

2.5.5　影响压缩比的因素

频繁闭图模式数量和代表模式数量的比值可以看作一个压缩比。压缩比越高，算法在实际应用中越有效。本节研究影响压缩比的因素。当距离阀值 δ 固定时，压缩比依赖于数据库的特征。对不同特征的数据库，压缩比可能变化很大。

当距离阀值 δ 和数据库中图的特征都固定时，一般说来，数据库越大，压缩比越高。下面的实验说明了这种情况。我们固定图的特征（$|V|$=4，$|E|$=2，$|I|$=15，$|T|$=30，$|L|$=20），变化参数 $|D|$，产生几个大小不同的数据库。然后，固定 min_sup=0.2，δ=0.1，在这些数据库上运行 RP-GD。实验结果如图 15 所示。可以看出，当图特征固定而数据库变大时，压缩比也在增加。其原因如下：当图特征固定而数据库变大时，频繁闭图模式数量也在增加，而代表模式数量保持相对稳定。2.5.2 节和 2.5.4 节显示了，RP-GD 产生的代表模式数量少于 RP-FP 和 RP-Leap 产生的代表模式数量。因此，如果使用 RP-Leap 和 RP-GD，我们可以得到更高的压缩比。因为在实际应用中，数据库通常都很大，因此挖掘代表模式的算法在实际应用中会很有效。

一般说来，数据库中图越多样化，压缩比越高。下面的实验说明了这种情况。我们固定参数（$|D|$=10K，$|V|$=4，$|E|$=2，$|I|$=12，$|L|$=20），变化图的平均大小参数 $|T|$，

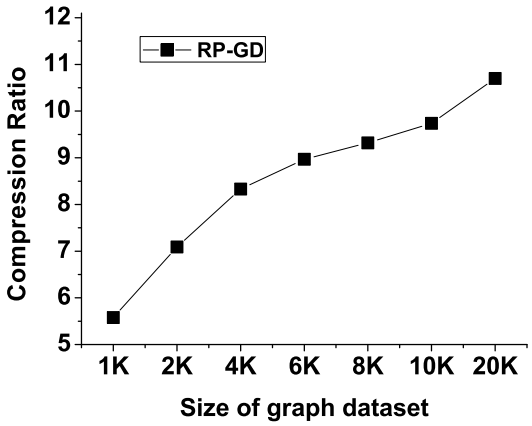


图 15　在 D?V4E2I15T30L20 的压缩比

产生几个数据库。然后，固定 min_sup= 0.25，δ=0.1，在这些数据库上运行 RP-GD。实验结果如图 16 所示。可以看出，当图的平均大小增加时，压缩比也在增加，其原因如下: 当图的平均大小增加时，图变得更加多样化，这使得频繁闭图模式的增长率要高于代表模式的增长率。

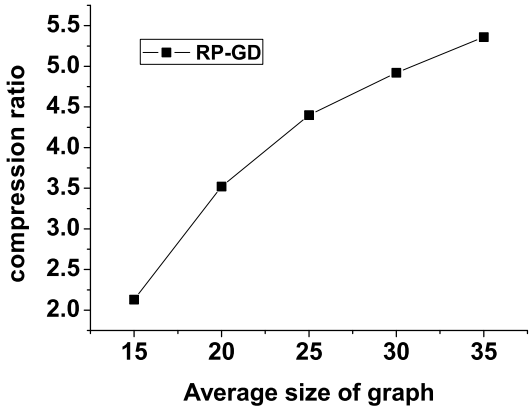


图 16　在 D10KV4E2I12T?L20 的压缩比

最后，研究距离阀值 δ 对压缩比的影响。图 17 显示了 δ 值变化时，PTE 和 D1KV4E2I15T30L20 数据集上的压缩比。显然，当 δ 值增加时，代表模式减少，从而压缩比增加。根据引理 2.3.3，当 δ=1 时，产生的代表模式集合恰恰是极大频繁图模式集合。从图 17 可以看出，在一般情况下，当频繁闭图模式数量和极大频繁图模式数量的比值大时，压缩比也高。

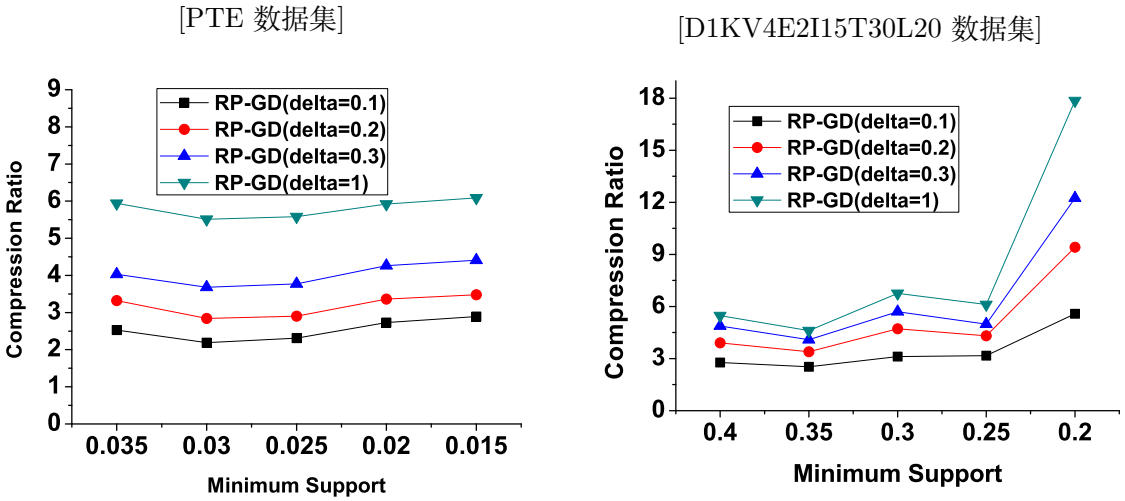


图 17　不同 δ 值对压缩比的作用

2.5.6　RP-GD 和 RP-Leap 的可扩展性

本小节研究算法 RP-GD 和 RP-Leap 的可扩展性。我们使用参数 V4E2I5T20L40 生成大小分别为 10K，20K，40K，60K，80K，100K 的 6 个图数据库。然后，固定 min_sup= 0.05，$\delta = 0.1$，在这些数据库上运行 RP-GD 和 RP-Leap，实验结果如图 18 所示。可以看出，算法 RP-GD 和 RP-Leap 有很好的可扩展性，随着数据量的增加, 它们的运行时间也在线性地增加。

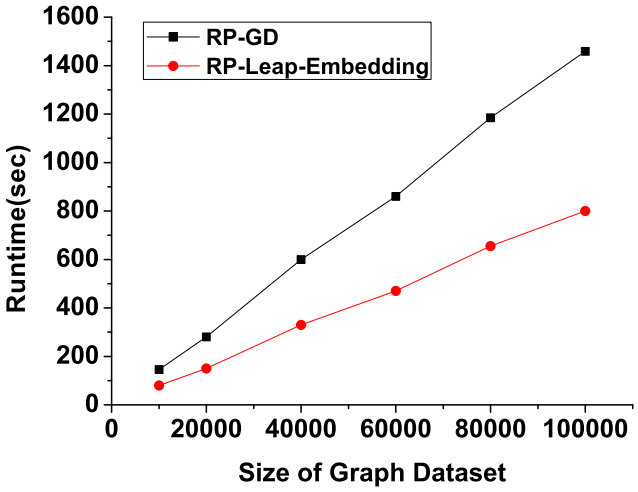


图 18　RP-GD 和 RP-Leap 的可扩展性

2.5.7　代表模式的可用性

本小节考查代表模式的可用性。实验中，我们发现很多有意义的图模式都具有高跳跃值。例如: 图 19 显示了 GraphRank 算法 [58] 发现的 CA 数据集里最有意义的图模式。它的 (绝对) 支持度是 64，跳跃值是 0.203 15。根据实验中的参数设置，它显然是一个 δ-跳跃模式。根据定理 2.4.1，代表模式集合包含所有 δ-跳跃模式，因此，它将作为一个代表模式被发现。

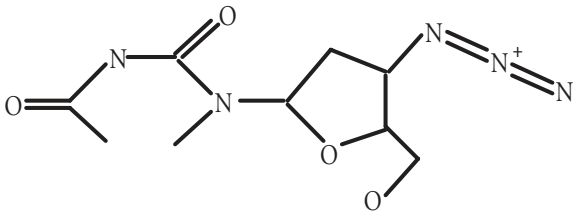


图 19　CA 数据集里最有意义的子结构

下面，我们给出代表模式中含有强分类的特征。在 AIDS 数据集合上，我们构造两个分类任务:(1) 对 CA(confirmed active) 类和 CM(confirmed moderately) 类中的化合物进行分类；(2) 对 CA 类和 CI(confirmed inactive) 类中的化合物进行分类。我们分别根据频繁闭图模式、极大频繁图模式、代表模式构造分类器。使用固定参数 $\delta = 0.1$ 和 $\epsilon = 0.01$ 分别运行算法 RP-FP，RP-GD 和 RP-Leap-Support，得到几个代表模式集合。分类准确率使用 5 次交叉验证进行评价，在每次交叉验证的训练集上，再使用另一个 5 次交叉验证对分类模型进行参数选择。所有分类器都使用相同的支持向量机 LIBSVM [26]，参数 C 在 $[2^{-5},2^{5}]$ 之间选择。所有分类器都使用线性核。我们使用 ROC 曲线下的面积 (AUC) 来度量分类性能，AUC 越大，表示分类性能越好。

在第一个分类实验中，对所有分类器使用相同数量的特征。首先，应用著名的特征选择方法——SVM RFE [53]，从频繁闭图模式集合和几个代表模式集合中选择一些特征，使得被选择的特征数等于极大频繁图模式数。然后，根据被选择的特征构造分类器。表 2 显示了第一个实验的分类性能 (AUC)。在第二个分类实验中，我们仍用 SVM RFE 从频繁闭图模式集合中选择一些特征，使得被选择的特征数等于 RP-FP 产生的代表模式数。对那些代表模式集合，我们保留所有的特征。表 3 显示了第二个实验的分类性能 (AUC)。

从表 2 可以看出，基于频繁闭图模式和代表模式构造的分类器的分类性能明显优于基于极大频繁图模式构造的分类器的分类性能。这说明极大频繁图模式集合中丢失了一些重要的强分类特征。从表 2 和表 3 可以看出，基于代表模式的分类器在分类性能方面几乎与基于频繁闭图模式的分类器相同。在表 3 中，尽管基于频繁闭图模式的

分类器使用了特征选择 SVM RFE，而基于代表模式的分类器没使用特征选择，它们的分类性能的差别仍然很小，这说明代表模式集合中保留着强分类特征。

表 2　分类性能比较（特征数 = 极大频繁图模式数）

min_sup	CA vs CM					CA vs CI				
	Closed	RP-FP	RP-GD	RP-Leap	Maximal	Closed	RP-FP	RP-GD	RP-Leap	Maximal
20%	0.797 29	**0.808 93**	0.806 16	0.805 54	0.725 46	0.903 19	0.910 66	**0.913 36**	0.91112	0.886 92
18%	0.807 94	0.812 33	0.818 65	**0.820 87**	0.767 15	0.900 98	0.909 57	0.911 62	**0.915 9**	0.880 09
16%	0.804 01	**0.831 6**	0.822 45	0.829 45	0.770 59	0.915 61	0.925 88	0.929 63	**0.930 73**	0.856 34
14%	0.821 84	0.835 08	0.836 85	**0.838 34**	0.780 12	0.934 21	**0.937 01**	0.934 78	0.933 6	0.858 98
12%	0.825 14	0.833 91	**0.839 92**	0.835 84	0.805 05	0.937 8	0.946 78	**0.948 15**	0.941 07	0.891 73
10%	0.830 87	0.841 46	**0.845 73**	0.840 26	0.812 36	0.941 49	0.941 66	0.940 21	**0.942 1**	0.919 25

表 3　分类性能比较（从闭频繁子图中选择 #RP − FP 个特征）

min_sup	CA vs CM				CA vs CI			
	Closed	RP-FP	RP-GD	RP-Leap	Closed	RP-FP	RP-GD	RP-Leap
20%	**0.816 63**	0.803 08	0.799 42	0.807 49	**0.921 96**	0.917 13	0.918 88	0.918 12
18%	**0.826 52**	0.817 08	0.817 93	0.817 04	**0.925 88**	0.917 96	0.918 98	0.919 96
16%	0.831 37	**0.832 47**	0.828 27	0.828 31	0.932 62	0.929 49	0.930 78	**0.933 49**
14%	**0.832 75**	0.828 49	0.827 87	0.831 06	**0.943 7**	0.938 92	0.936 04	0.936 34
12%	**0.835 17**	0.835 16	0.834 7	0.832 98	0.943 55	0.944 1	**0.945 06**	0.942 3
10%	0.833 48	0.836 01	**0.838 92**	0.838 2	0.940 87	**0.945 33**	0.944 06	0.942 28

在 2.5.2 节，2.5.4 节和 2.5.5 节中，我们显示了代表模式数量比频繁闭图模式数量少很多。因此，对代表模式进行特征选择比对频繁闭图模式进行特征选择更容易。因为图分类必然涉及子图同构测试，图模式数量越少，子图同构次数测试越少，因此基于代表模式的分类器在实际应用中比基于频繁闭图模式的分类器更有效。而且，挖掘代表模式（使用 RP-Leap）也比挖掘频繁闭图模式更高效。这些都说明了挖掘代表模式的优点。

2.6　本章小结

本章研究了如何从图数据库中高效地挖掘代表模式。首先，提出了几个新的概念：δ-覆盖图、跳跃值、δ-跳跃模式。然后，利用 δ-跳跃模式的性质，提出了挖掘代表模式的两个高效算法：RP-FP 和 RP-GD。利用图模式搜索空间中多个分枝之间的相似性，提出了挖掘近似代表模式的更高效的算法 RP-Leap。实验结果表明：RP-FP，RP-GD 和 RP-Leap 都能得到一个小的代表模式集合。当频繁闭图模式数量较大时，RP-GD 的挖掘效率远远高于 RP-FP 的挖掘效率。RP-Leap 以丢失少量代表模式为代价，取得

了比 RP-GD 多一个数量级的性能改善。最后，对化合物的分类结果证明了代表模式的可用性。因为图代表了最通用的模式类型，所以本章提出的 δ-跳跃模式，RP-GD 算法的启发式策略，RP-Leap 算法的跳跃策略也可以用来挖掘其他类型的代表模式（例如: 项集代表模式、序列代表模式、树代表模式等）。

第 3 章　挖掘跳跃模式

3.1　引言

作为一种通用的数据结构，图可以用来表示数据对象之间的各种复杂关系，例如: 图可以表示化合物的分子结构 [3]、蛋白质交互网络 [16]、社会网络 [1]、Web 结构图 [110] 等。随着科学与工程领域中图数据的大量出现，从图数据库中发现有用的知识已成为数据挖掘领域一项重要的研究课题。

目前，很多高效的频繁子图挖掘算法 [20,66,69,83,103,145] 已经被提出。为了进一步提高挖掘效率，研究人员又利用并行技术 [24]、索引技术 [128] 等来优化现有的频繁子图挖掘算法。然而，频繁子图挖掘算法输出的很多图模式都没有实际意义，例如: 只有一条边的图模式没有任何意义。为了得到有意义的图模式，研究人员又对图模式的拓扑结构施加某种限制，挖掘具有特殊结构的图模式，例如: 文献 [130] 提出了挖掘频繁闭集团的算法 CLAN；文献 [161] 提出了挖掘频繁闭类集团的算法 Cocain；文献 [162] 扩展了算法 Cocain，从基于磁盘的图数据库中挖掘频繁闭类集团；文献 [150] 研究了从关系图数据库中挖掘带有连接约束的频繁闭图模式。

然而，实际应用中很多有意义的图模式并不具有这些特殊的结构约束 (例如: 集团、类集团、高连通性、高密度等)，而且定义结构约束需要用户有丰富的领域知识。为了克服现有方法的缺点，本章提出一个新的研究问题: 挖掘图数据库中的核心子结构。下面的例子展示了在实际应用中核心子结构的特征。

美国国家癌症研究所 (NCI) 给出了对 HIV 病毒具有强抵抗作用的一个化合物集合 (CA 数据集)，CA 数据集中含有 422 个化合物，其中的化合物被分成如下几个化学类别 [2]:Azido Pyrimidines，Dyes Polyanions，Pyrimidine Nucleosides，Heavy Metal Compounds，Purine Nucleosides。我们发现每类中的核心子结构 (最大公共子结构) 都具有这样一个特性: 当核心子结构被扩展成任何一个多一条边的新的子结构时，新的子结构的支持度都快速地下降。例如: 图 1 显示了 Azido Pyrimidines 类的核心子结构，它在 CA 中的 (绝对) 支持度是 64。该子结构的任何真超图在 CA 中的支持度都小于或等于 51。实际上，每类中的核心子结构在化学上都代表一个重要的功能团。各种新的化合物主要以这些功能团为基础来创建。某个功能团 G 在一些化合物中可能被扩展成 G_1，而在另一些化合物中又可能被扩展成 G_2。因此，这些功能团的任何超结构的支持度都会明显地减少。

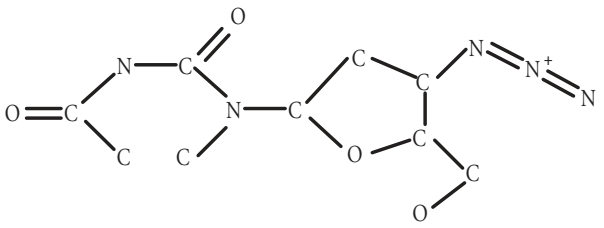


图 1　在 Azido Pyrimidines 类的核心子结构

针对实际应用中核心子结构的特征，本章提出了一种新的意义度量，称为 Δ-跳跃模式。如果在数据库 D 上图模式 p 的支持度比 p 的任何真超图的支持度都大于一个整数 Δ，那么 p 被称为 D 上的 Δ-跳跃模式。显然，我们对跳跃模式的定义能很好地刻画核心子结构的特征。

跳跃模式具有很多重要的特性，挖掘这些特性具有重要的实际意义。首先，跳跃模式是稳定的 (见 3.5.1 节)，它们对噪声和数据的变化不敏感。这一特性非常重要，因为在实际应用中噪声是普遍存在的，跳跃模式的这种抗干扰能力使得它们能被更广泛地应用。其次，挖掘跳跃模式能极大地减少挖掘结果的数量 (见 3.5.2 节)，这使得后续的分析工作量被大大简化，提高了挖掘结果的可用性。最后，也是最重要的，挖掘跳跃模式仍然能使有意义的图模式 (核心子结构) 保留在挖掘结果中 (见 3.5.6 节)。

本章研究如何从图数据库中高效地挖掘频繁跳跃模式。各种图挖掘算法主要利用支持度的反单调性质 (Apriori 性质) 对搜索空间进行裁剪。然而，跳跃模式不具有这样的性质，这使得挖掘跳跃模式非常具有挑战性。通过仔细研究跳跃模式自身的特性，本章提出了两种新的裁剪技术，基于内扩展的裁剪和基于外扩展的裁剪。通过将这两种新的裁剪技术集成到著名的 DFS 编码枚举框架中 [145]，我们设计了一个高效的挖掘算法 GraphJP。在理论上，我们严格地证明了这两种裁剪技术的正确性以及算法 GraphJP 的正确性。

大量真实和合成数据上的实验结果表明这两种新的裁剪技术能有效地裁剪图模式搜索空间，算法 GraphJP 能高效、可扩展地挖掘频繁跳跃模式。此外，一个重要的发现是当 Δ 值稍微大于 0 时 (例如:Δ=2)，频繁 Δ-跳跃模式的数量比频繁闭图模式的数量少很多。而且，在化合物上的实验结果表明算法 GraphJP 的挖掘结果中包含了化合物的核心子结构 (化合物的功能团)。

综上，本章的主要贡献如下: 第一，提出了一个新的研究问题，从图数据库挖掘跳跃模式；第二，研究了跳跃模式的性质，发现了跳跃模式具有稳定性、抗噪声干扰能力强等优点；第三，提出了两种新的裁剪技术，即基于内扩展的裁剪和基于外扩展的裁剪；第四，提出了一个高效的频繁跳跃模式挖掘算法 GraphJP；第五，实验结果验证了算法的效率和挖掘结果的可用性。

本章的内容安排如下:3.2 节介绍相关工作；3.3 节给出问题定义，并研究跳跃模式

的性质；3.4 节介绍新的裁剪技术，并给出完整的算法，以及证明裁剪技术和算法的正确性；3.5 节通过实验验证裁剪技术的有效性，挖掘算法的效率和扩展性以及挖掘结果的可用性；3.6 节对本章进行小结。

3.2　相关工作

频繁子图挖掘是最基本的图模式挖掘问题。很多高效的频繁子图挖掘算法已经被提出，例如:AGM [69]，FSG [83]，Mofa [20]，gSpan [145]，FFSM [66]，GASTON [103] 等。此外，研究人员还利用各种计算机技术来进一步提高挖掘效率。例如: 文献 [24] 提出了一个图模式并行挖掘算法；文献 [128] 利用数据库的索引技术来挖掘基于磁盘的图数据库；文献 [129] 提出了一个基于划分的方法 PartMiner，在数据库动态更新的情况下，挖掘频繁子图；文献 [11] 在 gSpan [145] 框架的基础上，提出了一些优化策略，通过减少最右扩展候选子图数量来提高挖掘效率。

除了挖掘通用的频繁子图，还有很多研究工作考虑挖掘具有特殊结构的图模式，例如: 挖掘频繁子树、挖掘频繁集团或类集团等。作为一种特殊的图模式，频繁子树 [35,114,121,159] 挖掘已经被广泛地研究。在很多应用领域中，集团是一种有着特殊意义的图模式。目前，已经提出了很多挖掘集团模式的算法。文献 [107] 提出了一个从图集合中挖掘类集团的算法 Crochet ，但是 Crochet 要求集团必须在所有图中都出现。文献 [130] 提出了一个从图数据库中挖掘频繁闭集团的算法 CLAN。后来，文献 [161] 又扩展了 CLAN，提出了挖掘频繁闭类集团的算法 Cocain。文献 [162] 扩展了算法 Cocain，从基于磁盘的图数据库中挖掘频繁闭类集团。文献 [96] 利用新的裁剪技术从图数据库中挖掘最大类集团。文献 [150] 研究了从关系图数据库中挖掘带有连接约束的频繁闭图模式，提出了一个模式增长方法 CloseCut 和一个模式规约方法 Splat。

挖掘频繁子图经常会产生指数级数量的图模式，使得挖掘结果难以被利用。为了减少图模式的输出数量，研究人员提出了挖掘频繁闭图模式 [146]、挖掘极大频繁图模式 [67,122] 和挖掘频繁图产生器 [160]。挖掘极大频繁图模式只输出搜索空间中边界上的图模式，一些重要的图模式将丢失。如文献 [160] 所示，挖掘频繁闭图模式和挖掘频繁图产生器输出的图模式数量大致相当。如 3.5 节中的实验所示，频繁闭图模式的数量仍然很多，用户难以利用。在第 2 章，我们提出了挖掘频繁闭图模式和挖掘极大频繁图模式的一个折中方法，挖掘一个代表模式集合，使得任何频繁图模式都能被一个代表模式 δ-覆盖。

实际应用中的图数据库可能存在很多核心子结构，比如化合物集合中的功能团。这些核心子结构是用户真正需要的重要图模式。为了能从图数据库中挖掘这些核心子结构，本章提出了一个新的研究问题: 挖掘频繁 Δ-跳跃模式。Δ-跳跃模式能很好地描

述核心子结构的特征。同时，Δ-跳跃模式扩展了闭图模式的定义，使得闭图模式成为了一种特殊的跳跃模式 (0-跳跃模式)。如 3.5 节中的实验所示，当跳跃距离阀值 Δ 稍微大于 0 时，频繁 Δ-跳跃模式的数量比频繁闭图模式的数量少很多。而且，挖掘频繁 Δ-跳跃模式 (Δ >0) 比挖掘频繁闭图模式快很多。更重要的是，通过挖掘频繁 Δ-跳跃模式，我们仍然可以得到数据库中重要的图模式。

Δ-跳跃模式的定义和项集挖掘中 δ-容错闭项集 [33] 和 Δ-闭项集 [19] 的定义有些类似，但又不完全相同。而且，文献 [19] 和 [33] 中的方法只能用来从事物数据库中挖掘项集模式，不能扩展到图数据库中挖掘图模式。

3.3 问题定义

本节给出跳跃模式的精确定义，并研究了跳跃模式的性质。本章定义的跳跃模式和第 2 章定义的跳跃模式的区别如下: 本章跳跃模式的作用域是数据库，而第 2 章跳跃模式的作用域是频繁图模式集合。因此，极大频繁图模式一定是第 2 章定义的跳跃模式，但是极大频繁图模式不一定是本章给出的跳跃模式。更重要的是，本章给出的算法直接挖掘本章定义的跳跃模式，而第 2 章的跳跃模式是通过现有频繁图模式挖掘算法得到的。

3.3.1 跳跃模式

本质上, 闭图模式是根据一个图模式与其超图模式之间的某种特殊关系来定义的。下面, 我们扩展闭图模式的定义来更精确地刻画一个图模式与其超图模式之间的关系。

定义 3.3.1(绝对跳跃值) 设 P 是数据库 D 中的一个图模式，P 在 D 中的绝对跳跃值定义为 $JV_{\text{abs}}(P, D) = \min\{\text{support}(P) - \text{support}(P') \mid \forall P', P \subset P'\}$。

定义 3.3.2(绝对跳跃模式) 设 P 是数据库 D 中的一个图模式，给定一个整数 $\Delta(\Delta \geqslant 0)$，如果 $JV_{\text{abs}}(P, D) > \Delta$，那么称 P 是 D 中的绝对 Δ-跳跃模式。

定义 3.3.3(相对跳跃值) 设 P 是数据库 D 中的一个图模式，P 在 D 中的相对跳跃值定义为 $JV_{\text{rel}}(P, D) = \min\{\frac{\text{support}(P)-\text{support}(P')}{\text{support}(P)} \mid \forall P', P \subset P'\}$。

定义 3.3.4(相对跳跃模式) 设 P 是数据库 D 中的一个图模式，给定一个小数 $\delta(0 \leqslant \delta \leqslant 1)$，如果 $JV_{\text{rel}}(P, D) > \delta$，那么称 P 是 D 中的相对 δ-跳跃模式。

直觉上, 图模式 P 在数据库 D 中的绝对（相对）跳跃值是 P 与 P 的所有真超图之间最小的绝对（相对）支持度差异。

例 3.3.1 图 2 显示了一个图数据库 D。图 3 显示了 D 中图模式的一个子集。显然，P_1 在 D 中的支持度是 4，P_1 的任何超图在 D 中的支持度都小于或等于 3。因此，P_1 在 D 中的绝对跳跃值 $JV_{\text{abs}}(P_1, D)$=1，P_1 在 D 中的相对跳跃值 $JV_{\text{rel}}(P_1, D)$=1/4。

类似地，可以得到 $JV_{\text{abs}}(P_2, D)$=2，$JV_{\text{rel}}(P_2, D)$=2/3，$JV_{\text{abs}}(P_3, D)$=0，$JV_{\text{rel}}(P_3, D)$=0。如果 Δ=1，那么图 3 显示的图模式集合中只有 P_2 是 D 中的绝对 Δ-跳跃模式。

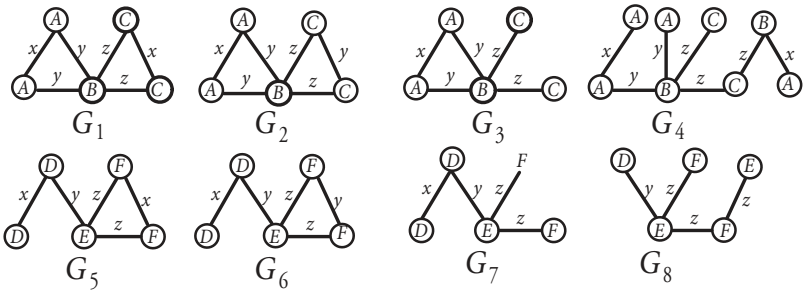


图 2　图数据库 D

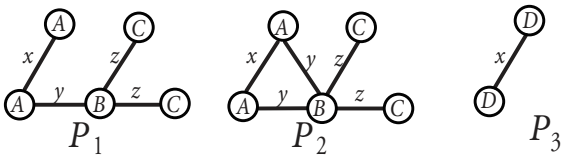


图 3　图模式子集

根据上面的定义，从图数据库中挖掘频繁跳跃模式问题可以定义如下:

输入: 图数据库 $D = \{G_1, G_2, \ldots, G_n\}$，最小支持度阈值 min_sup，最小绝对跳跃阀值 Δ（或最小相对跳跃阀值 δ）。

输出: D 中所有频繁的绝对 Δ-跳跃模式（或相对 δ-跳跃模式）的集合，即 $\{P|\text{support}(P; D) \geqslant \text{min\_sup}, JV_{\text{abs}}(P; D) > \Delta\}$（或者 $\{P|\text{support}(P; D) \geqslant \text{min\_sup}, JV_{\text{rel}}(P; D) > \delta\}$）。

当上下文清楚的时候，我们就使用 Δ-跳跃模式表示绝对 Δ-跳跃模式，使用 δ-跳跃模式表示相对 δ-跳跃模式。

3.3.2　跳跃模式的性质

引理 3.3.1　设 P 是数据库 D 中的一个 Δ_1-跳跃模式。若 $\Delta_1 > \Delta_2$, 则 P 也是数据库 D 中的一个 Δ_2-跳跃模式。

证明　根据绝对 Δ-跳跃模式的定义，$JV_{\text{abs}}(P, D) > \Delta_1$。因为 $\Delta_1 > \Delta_2$，所以 $JV_{\text{abs}}(P, D) > \Delta_2$。因此，$P$ 也是数据库 D 中的一个 Δ_2-跳跃模式。　□

引理 3.3.2　设 P 是数据库 D 中的一个 δ_1-跳跃模式。若 $\delta_1 > \delta_2$, 则 P 也是数据库 D 中的一个 δ_2-跳跃模式。

证明　类似于引理 3.3.1 的证明。　□

引理 3.3.3 P 是数据库 D 中的一个绝对 0-跳跃模式，当且仅当 P 是数据库 D 中的闭图模式。

证明 设 P 是数据库 D 中的一个绝对 0-跳跃模式。根据绝对 Δ-跳跃模式的定义，$JV_{\text{abs}}(P,D) > 0$。如果存在 P 的真超图 P'，使得 $\text{support}(P) = \text{support}(P')$。根据绝对跳跃值的定义，$JV_{\text{abs}}(P;D) \leqslant \text{support}(P;D) - \text{support}(P';D) = 0$。这与 $JV_{\text{abs}}(P;D) > 0$ 相矛盾。因此，P 是数据库 D 中的一个闭图模式。

设 P 是数据库 D 中的一个闭图模式。根据闭图模式的定义，不存在 P 的真超图 P'，使得 $\text{support}(P;D) = \text{support}(P';D)$。因此，对 P 的任何真超图 P' 都有 $\text{support}(P;D) - \text{support}(P';D) > 0$。根据绝对跳跃值的定义，可得 $JV_{\text{abs}}(P;D) > 0$。再根据绝对 Δ-跳跃模式的定义，可知 P 是数据库 D 中的一个绝对 0-跳跃模式。 □

引理 3.3.4 P 是数据库 D 中的一个相对 0-跳跃模式，当且仅当 P 是数据库 D 中的闭图模式。

证明 类似于引理 3.3.3 的证明。 □

引理 3.3.3 和引理 3.3.4 说明: 闭图模式实际上是一种特殊的跳跃模式 (0-跳跃模式)。因此，本章给出的算法也可以直接用来挖掘频繁闭图模式。

跳跃模式的一个重要特征是它们具有一定程度的稳定性。例如: 当数据库中的数据发生变化时，闭图模式集合中的 Δ-跳跃模式（$\Delta > 0$）仍然可能是闭图模式。并且，Δ 值越大，这种可能性越大。见下面的引理和推论。

引理 3.3.5 设 P 是数据库 D 中的一个 Δ_1-跳跃模式。D' 是另一个数据库，并且满足 $|D - D'| \leqslant \Delta_2$（$0 \leqslant \Delta_2 \leqslant \Delta_1$），则 P 是数据库 D' 中的一个 $(\Delta_1 - \Delta_2)$-跳跃模式。

证明 用反证法证明该引理。假定 P 不是数据库 D' 中的 $(\Delta_1-\Delta_2)$-跳跃模式。根据绝对跳跃模式的定义，存在 P 的真超图 P'，使得 $\text{support}(P;D')-\text{support}(P';D') \leqslant \Delta_1-\Delta_2$。因为 $\text{support}(P;D')-\text{support}(P';D')=(\text{support}(P;D'-D)+\text{support}(P;D'\cap D))-(\text{support}(P';D'-D)+\text{support}(P';D'\cap D))=(\text{support}(P;D'-D)-\text{support}(P';D'-D))+(\text{support}(P;D'\cap D)-\text{support}(P';D'\cap D)) \geqslant \text{support}(P;D'\cap D)-\text{support}(P';D'\cap D)$，所以 $\text{support}(P;D'\cap D)-\text{support}(P';D'\cap D) \leqslant \Delta_1-\Delta_2$。因为 $|D-D'| \leqslant \Delta_2$，所以 $\text{support}(P;D-D') \leqslant \Delta_2$。因此，$\text{support}(P;D)-\text{support}(P';D)=(\text{support}(P;D-D')+\text{support}(P;D\cap D')) - (\text{support}(P';D-D')+\text{support}(P';D\cap D')) = (\text{support}(P;D-D')-\text{support}(P';D-D'))+(\text{support}(P;D\cap D')-\text{support}(P';D\cap D')) \leqslant \text{support}(P;D-D') + (\Delta_1 - \Delta_2) \leqslant \Delta_2 + (\Delta_1 - \Delta_2) = \Delta_1$。根据绝对跳跃模式的定义，可知 P 不是数据库 D 中的 Δ_1-跳跃模式，这与已知条件相矛盾。因此，P 一定是数据库 D' 中的一个 $(\Delta_1 - \Delta_2)$-跳跃模式。 □

推论 3.3.1 设 P 是数据库 D 中的一个 Δ-跳跃模式，D' 是另一个数据库，并且

满足 $|D-D'| \leqslant \Delta$，则 P 是数据库 D' 中的一个闭图模式。

证明　根据引理 3.3.3 和引理 3.3.5 容易证明该推论。　□

推论 3.3.1 说明: 在删除或改变数据库 D 中任意 Δ 个数据图之后，D 中原有的 Δ-跳跃模式仍旧是 D 中的闭图模式。对于 δ-跳跃模式，我们也有如下类似的引理和推论。

引理 3.3.6　设 P 是数据库 D 中的一个 δ_1-跳跃模式, D' 是另一个数据库，并且满足 $|D-D'| \leqslant \text{support}(P;D) \times \delta_2$ $(0 \leqslant \delta_2 \leqslant \delta_1)$，则 P 是数据库 D' 中的一个 $(\delta_1-\delta_2)$-跳跃模式。

证明　类似于引理 3.3.5 的证明。　□

推论 3.3.2　设 P 是数据库 D 中的一个 δ-跳跃模式，D' 是另一个数据库，并且满足 $|D-D'| \leqslant \text{support}(P;D) \times \delta$，则 P 是数据库 D' 中的一个闭图模式。

证明　根据引理 3.3.4 和引理 3.3.6 容易证明该推论。　□

下面通过实验证明: 随着 Δ 值的增加，Δ-跳跃模式对噪声的抗干扰能力增强。也就是说，当向数据库中加入噪声而使数据库发生改变时，原来的 Δ-跳跃模式仍然有很大的可能性是变化之后数据库中的 Δ-跳跃模式。Δ 值越大，这种可能性越大。

实验中使用的 CA 数据集将在实验部分详细描述。对 CA 数据集中的每个图，应用下面的过程，向该数据集中加入噪声。对 CA 数据集中的每个图 G 的每个结点 v，扔一枚不公平的硬币，正面出现的概率为 p（噪声比），反面出现的概率为 1-p。如果正面出现，将 v 的标号改为任意的一个结点标号（从所有结点标号中均匀随机选择）。如果反面出现，不对 v 进行任何改动，继续处理 G 的下一个结点。

使用上面的过程向数据集中加入噪声, 可以由旧的数据集 D 创建一个新的数据集 D'。现在研究如下问题: D 中的频繁 Δ-跳跃模式有多大的可能性仍旧是 D' 中的频繁 Δ-跳跃模式? 为了度量这种可能性的大小，我们使用如下定义的**稳定率**。设 FJS 是 D 中的频繁 Δ-跳跃模式集合，FJS' 是 D' 中的频繁 Δ-跳跃模式集合。将频繁 Δ-跳跃模式的稳定率定义为 $|FJS \cap FJS'|/|FJS|$。

使用最小相对支持度 min_sup = 10%，对上面的实验重复 10 次取平均值，结果如图 4 所示。为了比较，我们也在图中给出了闭图模式 (0-跳跃模式) 的稳定率。

从图 4 可以看出，随着 Δ 值的增加，Δ-跳跃模式的稳定率也在增加。Δ-跳跃模式的这一特性非常重要，因为有意义的图模式经常是具有高跳跃值的跳跃模式 (见 3.5.6 节)。跳跃模式的跳跃值越高，它的稳定性越强。因此，即使在数据中存在大量噪声的情况下，通过挖掘 Δ-跳跃模式，我们仍能得到有意义的图模式。同样，δ-跳跃模式也具有这一特性，随着 δ 值的增加，δ-跳跃模式的抗噪声干扰能力也在增强。

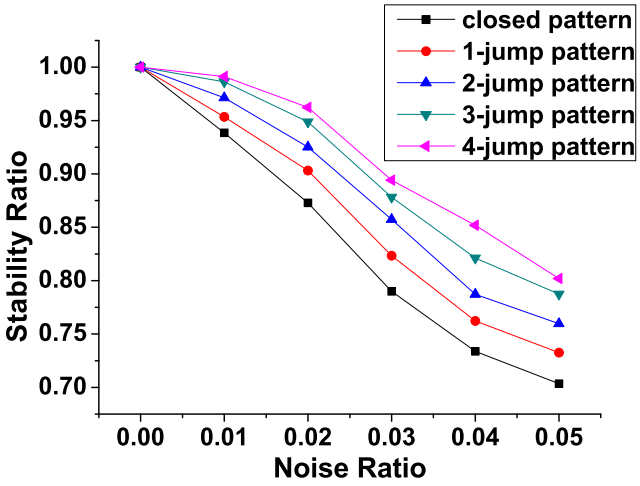


图 4　Δ-跳跃模式的稳定率

3.4　挖掘频繁跳跃模式

3.4.1　DFS 编码搜索树

本章的算法利用了著名的 DFS 编码搜索树 [145] 来挖掘频繁跳跃模式。本小节简单地介绍了有关的基本概念。

最小 DFS 编码。

当在图 G 上执行深度优先搜索（DFS）时，一个对应的 DFS 树被创建。不同的 DFS 搜索可以创建不同的 DFS 树。例如: 对图 5(a) 中的 G，图 5(b) 和图 5(c) 显示了两棵不同的 DFS 树。在 DFS 树中的边称为**前边** (在图 5(b) 和图 5(c) 中加粗显示)，否则称为**后边**。如图 5(b) 和图 5(c) 所示，根据 DFS 访问结点的顺序，我们可以对每个结点进行编号。边 e 可以表示成 (i, j)，其中 i 和 j 是与 e 相邻的结点标识 (编号)。如果 $i < j$，那么 $e = (i, j)$ 表示一条前边，否则 $e = (i, j)$ 表示一条后边。

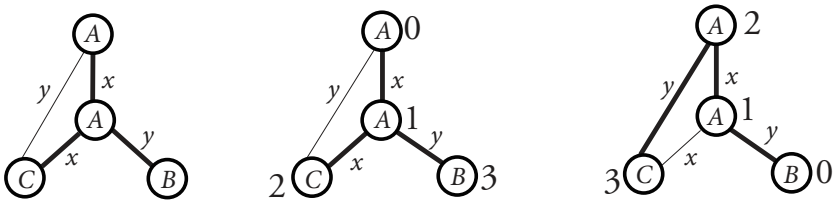


图 5　图的 DFS 树

给定图 G 的一个 DFS 树 T，可以在 G 的所有边上定义一种线性顺序如下: 设 $e_1 = (i_1, j_1)$ 和 $e_2 = (i_2, j_2)$ 是 G 的任意两条边。若 e_1 和 e_2 都是前边，$j_1 < j_2$(或者

$i_1 > i_2 \wedge j_1 = j_2$)，则 $e_1 \prec e_2$；若 e_1 和 e_2 都是后边，$i_1 < i_2$(或者 $i_1 = i_2 \wedge j_1 < j_2$)，则 $e_1 \prec e_2$；若 e_1 是前边，e_2 是后边，$j_1 < i_2$，则 $e_1 \prec e_2$；若 e_1 是后边，e_2 是前边，$i_1 < j_2$，则 $e_1 \prec e_2$。

现在将边 $e = (i, j)$ 表示为五元组 (i, j, l_i, l_e, l_j)，其中，l_i 和 l_j 为对应结点的标号，l_e 为边标号。给定图 G 的一个 DFS 树 T，根据上面定义的线性顺序，可以为 G 定义一个 DFS 编码 $\text{code}(G, T)$。例如: 如果图 5(b) 中的 T_1 是图 5(a) 中 G 的 DFS 树，那么有 $\text{code}(G, T_1) = \{(0, 1, A, x, A) - (1, 2, A, x, C) - (2, 0, C, y, A) - (1, 3, A, y, B)\}$。

假定在图 G 的标号集合上也存在一种线性顺序。那么，根据边上的线性顺序和标号集合上的线性顺序，可以构造一种 DFS 字典顺序。根据 DFS 字典顺序，任意两个 DFS 编码都可以比较大小。例如: 在图 5 中，$\text{code}(G, T_1) < \text{code}(G, T_2)$。在图 G 的所有 DFS 编码中，最小 DFS 编码 $\min(G)$ 称为 G 的规范编码。在图 5 中，$\min(G) = \text{code}(G, T_1)$。最小 DFS 编码的一个重要性质是: 两个图 G_1 和 G_2 同构，当且仅当 $\min(G_1) = \min(G_2)$。

3.4.1.1　最右扩展

给定图 G 的一个 DFS 树 T，第一个被访问的结点称为**根**，最后一个被访问的结点称为**最右结点**，从根到最右结点的路径称为**最右路径**。例如: 在图 5(b) 中最右路径是 $0 \to 1 \to 3$，在图 5(c) 中最右路径是 $0 \to 1 \to 2 \to 3$。

为了避免产生重复的频繁子图，当通过增加新边扩展频繁子图时，文献 [145] 只执行如下受限的扩展:(1) **向后扩展**，即通过联结最右结点和最右路径上的另一个结点，增加一条新边。(2) **向前扩展**，即引进一个新结点，通过联结最右路径上的一个结点和该新结点，增加一条新边。这两种受限的扩展被称为**最右扩展**。

3.4.1.2　DFS 编码搜索树

文献 [145] 采用先深搜索的办法挖掘频繁子图，只在最小 DFS 编码上进行最右扩展，而且可保证挖掘结果的完整性。采用此办法挖掘频繁子图时，会在搜索的过程中形成一个树状的搜索空间（称为 DFS 编码搜索树），其中的每个结点代表一个频繁子图。图 6 显示了从图 2 中的数据库挖掘频繁子图时（$\text{min\_sup} = 2$）形成的 DFS 编码搜索树。在该编码搜索树的每个结点的旁边都有一对数 $(a : b)$，其中 a 表示该结点被枚举的顺序编号，b 表示该结点代表的频繁子图的支持度。在本章中，我们有时使用结点的顺序编号表示该结点代表的频繁子图。例如，图 6 中结点 1 就表示频繁子图 $\{(0, 1, A, x, A) - (1, 2, A, y, B)\}$。

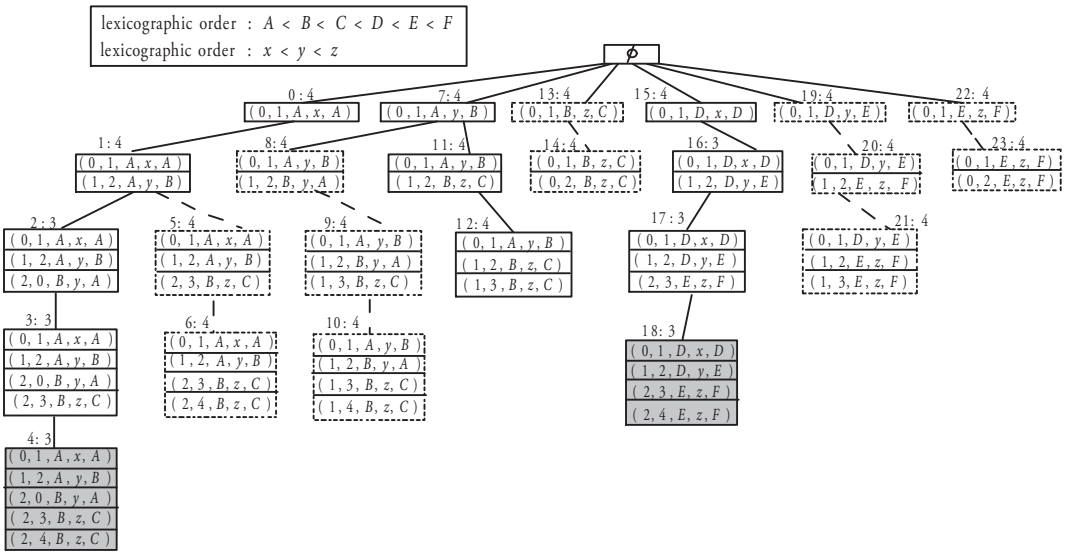


图 6　用 min_sup = 2 挖掘图 2 中的数据库时形成的 DFS 编码搜索树

3.4.2　裁剪搜索空间的基本思想

利用 DFS 编码搜索树，可以设计一个直接的算法来挖掘所有的频繁跳跃模式。具体地说，每当发现一个频繁子图 p 时，我们搜索 p 的所有多一条边的超图并计算它们的支持度，然后根据跳跃模式的定义判断 p 是否是跳跃模式。然而，这种算法效率太低，因为它需要枚举所有的频繁子图。

实际上，当挖掘频繁跳跃模式时，DFS 编码搜索树中的很多分枝不需要枚举，因为那些分枝不含有任何频繁跳跃模式。例如，当使用 min_sup = 2 和 Δ=1 挖掘图 2 中的数据库时，形成的 DFS 编码搜索树如图 6 所示。该搜索树中的两个灰色结点对应仅有的两个频繁 Δ-跳跃模式，其他结点都不是频繁 Δ-跳跃模式。原则上，该搜索树中不包含频繁 Δ-跳跃模式的分枝都应该被裁剪掉。本章给出的裁剪技术能裁剪掉该搜索树中所有用虚线连接的分枝。

在介绍具体的裁剪技术之前，先定义几个在后面的讨论中将使用的记号。$p \diamond e$ 表示在图模式 p 中增加一条新边 e 而得到的一个图模式。注意: $p \diamond e$ 表示 e 可能是也可能不是 p 的最右扩展。$p < q$ 表示 p 的最小 DFS 编码小于 q 的最小 DFS 编码，即 $\min(p) < \min(q)$。根据最小 DFS 编码的定义容易证明下面两个引理，它们将在后面的证明中被使用。

引理 3.4.1　设 p 是 DFS 编码搜索树中的一个结点，q 是 p 的任意一个后裔，则 $p \subset q$，$p < q$。

证明　因为 q 是 p 的一个后裔，显然 q 是 p 的一个真超图，我们有 $p \subset q$。设 p 的最小 DFSCode 是 $(e_0, e_1, \ldots, e_n)$，其中 e_i 是 p 的边编码 [145]。根据 DFSCode 的定

义，q 的最小 DFSCode 将是这样的形式 $(e_0, e_1, \ldots, e_n, e_{n+1} \ldots, e_m)$。又根据字典 DFS 顺序 [145]，有 $(e_0, e_1, \ldots, e_n) < (e_0, e_1, \ldots, e_n, e_{n+1} \ldots, e_m)$。因此，$p < q$。 □

引理 3.4.2　如果 $p \diamond e < p$，$p \diamond e \subseteq q$，则 $q < p$。

证明　因为 $p \diamond e < p$，根据 DFS 编码的定义 [145]，$p \diamond e$ 的任何超图的最小 DFSCode 都小于 p 的最小 DFSCode。又因为 $p \diamond e \subseteq q$，所以 q 的最小 DFSCode 小于 p 的最小 DFSCode，即 $q < p$。 □

下面给出裁剪技术的基本思想。设 p 是 DFS 编码搜索树中的一个结点，如果 p 的所有后裔都不是跳跃模式，那么我们可以安全地裁剪以 p 为根的搜索分枝。具体地说，如果对 p 的任意后裔 q，都存在扩展边 e 满足 $\text{support}(q) - \text{support}(q \diamond e) \leqslant \Delta$ ($\frac{\text{support}(q)-\text{support}(q \diamond e)}{\text{support}(q)} \leqslant \delta$)，说明以 p 为根的搜索分枝不含有 Δ-跳跃模式 (δ-跳跃模式)，我们可以安全地裁剪该分枝。基于这个思想，本章提出了两种新的裁剪技术，即基于内扩展的裁剪和基于外扩展的裁剪，将在下两小节详细介绍它们。

3.4.3　基于内扩展的裁剪

先定义几个新概念。在图模式 p 中增加一条新边 e，可得到图模式 $p \diamond e$。如果 e 引进一个新结点，称 e 是 p 的**外扩展边**，否则称 e 是 p 的**内扩展边**。接下来，我们用 $e = (i, j, e_l)$ 表示 e 是 p 的内扩展边，其中 i 和 j 是 p 的结点标识，e_l 是新边的标号。我们用 $e = (i, e_l, v_l)$ 表示 e 是 p 的外扩展边，其中 i 是 p 的结点标识，e_l 是新边的标号，v_l 是新结点的标号。注意: 如果 e 是 p 的内扩展边，e 可能是也可能不是 p 的向后扩展边。例如: 在图 6 中 $e = (2, 0, y)$ 是图模式 $\{(0, 1, A, x, A) - (1, 2, A, y, B)\}$(结点 1)的内扩展边和向后扩展边。然而，$e = (2, 0, y)$ 只是图模式 $\{(0, 1, A, x, A) - (1, 2, A, y, B) - (2, 3, B, z, C)\}$ (结点 5) 的内扩展边，但不是它的向后扩展边。

定义 3.4.1(内关联)　设 G 是数据库中的一个图，图模式 p 是 G 的一个子图，$e = (i, j, e_l)$ 是图模式 p 的内扩展边。如果对 p 到 G 的每一个子图同构 f，G 都有一条标号为 e_l 的边 $(f(i), f(j))$，那么称在 G 上 (i, j, e_l) 内关联于 p。

如果在图 G 上 $e = (i, j, e_l)$ 内关联于 p，意味着 p 在 G 上的每次出现都蕴含着 $p \diamond e$ 的出现。

定义 3.4.2(内关联数)　设 D 是一个图数据库，$e = (i, j, e_l)$ 是图模式 p 的内扩展边。在 D 上 e 相对于 p 的内关联数的定义为 $\text{Ass\_Internal}_D(e, p) = |\{G_i | G_i \in D,\ e \text{ 在 } G_i \text{ 上内关联于 } p\}|$。

例 3.4.1　边 $e = (2, 0, y)$ 是图模式 $p = \{(0, 1, A, x, A) - (1, 2, A, y, B)\}$ 的内扩展边。对于图 2 中的数据库 D，容易验证在 G_1，G_2 和 G_3 上，e 内关联于 p。因此，在 D 上 e 相对于 p 的内关联数 $\text{Ass\_Internal}_D(e, p) = 3$。

引理 3.4.3 设 D 是一个图数据库，$e=(i,j,e_l)$ 是图模式 p 的内扩展边。如果 $\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p)-\Delta, p \subset q, p\diamond e \not\subseteq q$，那么 $\text{support}(q)-\text{support}(q\diamond e) \leqslant \Delta$。

证明 把 p 的支持集划分成不相交的两个子集 S_1 和 S_2，其中 S_1=$\{G_i|G_i \in D$，在 G_i 上 e 内关联于 $p\}$，S_2=$\{G_i|G_i \in D$，在 G_i 上 e 不与 p 内关联$\}$。显然，$\text{support}(p) = |S_1|+|S_2|$，$\text{Ass_Internal}(e,p)$=$|S_1|$。因为 $p \subset q$，我们可以假定在 S_1 中有 x 个图含有 q，在 S_2 中有 y 个图含有 q。因此，$\text{support}(q) = x+y$。设 G 是 S_1 中含有 q 的任意一个图，I_q 是 q 在 G 中的任意一个实例。因为 $p \subset q$，I_q 也含有 p 在 G 中的一个实例 I_p。因为在 G 上 e 内关联于 p，所以 I_p 能被扩展成 $I_p \diamond e$。因为 $p\diamond e \not\subseteq q$，所以我们有 $I_p\diamond e \not\subseteq I_q$。因为 e 是 p 的内扩展边，所以使用 e 扩展 I_q 不会引进新的结点。因此，I_q 能被 e 扩展成 $I_q \diamond e$。G 含有 $q\diamond e$, 因为 G 是 S_1 中含有 q 的任意一个图，我们有 $\text{support}(q\diamond e) \geqslant x$，因此，$\text{support}(q)-\text{support}(q\diamond e) \leqslant (x+y)-x = y \leqslant |S_2| = \text{support}(p)-|S_1| = \text{support}(p)-\text{Ass\_Internal}(e,p) \leqslant \Delta$。 □

根据引理 3.4.1，引理 3.4.2 和引理 3.4.3，我们为 Δ-跳跃模式导出如下基于内扩展的裁剪条件。

定理 3.4.1 设 D 是一个图数据库，$e=(i,j,e_l)$ 是图模式 p 的内扩展边。如果 $\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p)-\Delta$，$p\diamond e < p$，那么在 DFS 编码树上 p 的所有后裔都不是 Δ-跳跃模式。

证明 设在 DFS 编码树上，q 是 p 的任意一个后裔。根据引理 3.4.1，我们有 $p \subset q$，$p<q$。现在用反证法证明 $p\diamond e \not\subseteq q$。如果 $p\diamond e \subseteq q$，根据引理 3.4.2，我们有 $q<p$。这与 $p<q$ 相矛盾，因此，$p\diamond e \not\subseteq q$。因为 $\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p)-\Delta$，$p \subset q$，$p\diamond e \not\subseteq q$，根据引理 3.4.3，我们有 $\text{support}(q)-\text{support}(q\diamond e) \leqslant \Delta$。根据 Δ-跳跃模式的定义，q 不是 Δ-跳跃模式。 □

对于 DFS 编码树中的一个图模式 p，如果定理 3.4.1 中的条件成立，那么 p 的所有后裔都不是 Δ-跳跃模式。而且，因为 $\text{support}(p)-\text{support}(p\diamond e) \leqslant \text{support}(p)-\text{Ass\_Internal}(e,p) \leqslant \Delta$，所以 p 本身也不是 Δ-跳跃模式。因此，可以安全地裁剪以 p 为根的分枝。这种裁剪技术称为**基于内扩展的裁剪**。

例 3.4.2 假定用 min_sup=2 和 Δ=1 从图 2 所示的数据库中挖掘频繁 Δ-跳跃模式。生成的 DFS 编码搜索树如图 6 所示。该搜索树中结点 5 对应的图模式为 $p=\{(0,1,A,x,A)-(1,2,A,y,B)-(2,3,B,z,C)\}$，边 $e=(2,0,y)$ 是 p 的内扩展边。根据最小 DFS 编码的定义，$p\diamond e < p$。容易计算 $\text{support}(p)$=4，$\text{Ass\_Internal}_D(e,p)$=3。因此，$\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p)-\Delta$。利用基于内扩展的裁剪技术，以结点 5 为根的分枝可以安全地被裁剪掉。同理，以结点 8 为根的分枝也可以利用基于内扩展的裁剪技术安全地被裁剪掉。

类似地，我们也为 δ-跳跃模式导出了如下基于内扩展的裁剪条件。

定理 3.4.2　设 D 是一个图数据库，min_sup 是最小支持度，$e=(i,j,e_l)$ 是图模式 p 的内扩展边。如果 $\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p) - \text{min\_sup} \times \delta$，$p \diamond e < p$，那么在 DFS 编码树上 p 的所有后裔都不是频繁 δ-跳跃模式。

证明　设在 DFS 编码树上 q 是 p 的任意一个后裔。根据引理 3.4.1，我们有 $p \subset q$，$p < q$。现在用反证法证明 $p \diamond e \not\subseteq q$。如果 $p \diamond e \subseteq q$，根据引理 3.4.2，我们有 $q < p$。这与 $p < q$ 相矛盾，因此，$p \diamond e \not\subseteq q$。因为 $\text{Ass\_Internal}_D(e,p) \geqslant \text{support}(p) - \text{min\_sup} \times \delta$，$p \subset q, p \diamond e \not\subseteq q$，根据引理 3.4.3，我们有 $\text{support}(q) - \text{support}(q \diamond e) \leqslant \text{min\_sup} \times \delta$。根据 q 是否频繁分两种情况讨论: (1) 如果 q 不是一个频繁模式，结论显然成立。(2) 如果 q 是一个频繁模式，那么 $\text{support}(q) \geqslant \text{min\_sup}$。因此，$\frac{\text{support}(q)-\text{support}(q \diamond e)}{\text{support}(q)} \leqslant \frac{\text{min\_sup} \times \delta}{\text{min\_sup}} = \delta$。根据 δ-跳跃模式的定义，q 不是 δ-跳跃模式，结论成立。　□

3.4.4　基于外扩展的裁剪

除了利用内扩展边导出有效的剪枝条件，我们还可以利用外扩展边设计有效的裁剪技术。先给出外关联和外关联数的定义。

定义 3.4.3(外关联)　设 G 是数据库中的一个图，图模式 p 是 G 的一个子图，(i, e_l, v_l) 是图模式 p 的外扩展边。如果对 p 到 G 的每一个子图同构 f，G 都有一个结点 v，并且同时满足如下条件:

(1) 结点 v 的标号是 v_l;

(2)$(f(i), v)$ 是一条标号为 e_l 的边;

(3) 不存在 p 中的结点 j 使得 $f(j) = v$,

那么称在 G 上 $e=(i,e_l,v_l)$ 外关联于 p。

定义 3.4.4(外关联数)　设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。在 D 上 e 相对于 p 的外关联数的定义为 $\text{Ass\_External}_D(e,p)=|\{G_i | G_i \in D$，$e$ 在 G_i 上外关联于 $p\}|$。

例 3.4.3　边 $e=(0,x,D)$ 是图模式 $p=\{(0,1,D,y,E)\}$ 的外扩展边。对于图 2 中的数据库 D，容易验证在 G_5，G_6 和 G_7 上，e 外关联于 p。因此，在 D 上 e 相对于 p 的外关联数 $\text{Ass\_External}_D(e,p)=3$。

如果在图 G 上 $e=(i,e_l,v_l)$ 外关联于 p，意味着 p 在 G 上的每次出现都蕴含着 $p \diamond e$ 的出现。注意: 如果 e 是 p 的外扩展边，e 可能是也可能不是 p 的向前扩展边。例如: 边 $e=(2,z,C)$ 是图模式 $\{(0,1,A,x,A)-(1,2,A,y,B)\}$ 的外扩展边和向前扩展边。然而 $e=(2,x,A)$ 只是图模式 $\{(0,1,A,y,B)-(1,2,B,y,A)-(1,3,B,z,C)\}$ 的外扩展边，但不是它的向前扩展边。

利用引理 3.4.3 的思想，我们很可能做出这样的假设: 设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。如果 $\text{Ass\_External}_D(e,p)\geqslant \text{support}(p)-\Delta$，$p\subset q$，$p\diamond e\not\subseteq q$，那么 $\text{support}(q)-\text{support}(q\diamond e)\leqslant\Delta$。不幸的是，这样的假设并不成立。请看下面的反例。

例 3.4.4 考虑图 7 中的数据库和图模式。边 $e=(1,y,C)$ 是图模式 $p=\{(0,1,A,x,B)\}$ 的外扩展边。容易计算得到 $\text{support}(p)$=3，$\text{Ass\_External}_D(e,p)$= 3。设 Δ=1，显然 $\text{Ass\_External}_D(e,p)\geqslant\text{support}(p)-\Delta$。对于图 7 中的模式 q，我们有 $p\subset q$，$p\diamond e\not\subseteq q$，上面假设的条件成立。然而，$\text{support}(q)$=3，$\text{support}(q\diamond e)=1$。因此，$\text{support}(q)-\text{support}(q\diamond e)>\Delta$，上面假设的结论并不成立。

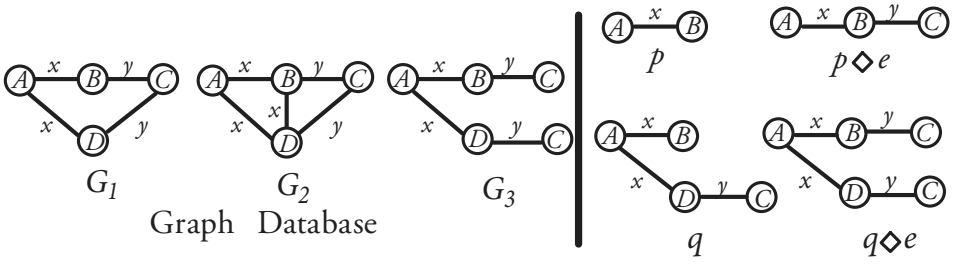


图 7 基于外扩展裁剪的反例

在上面的例子中，尽管在 G_1，G_2 和 G_3 上边 $e=(1,y,C)$ 外关联于图模式 $p=\{(0,1,A,x,B)\}$。然而，在 G_1 和 G_2 上边 e 中标号为 C 的端点只能出现在 q 的实例中。因此，在 G_1 和 G_2 上 q 不能被扩展成 $q\diamond e$，从而使得上面假设的结论不成立。对于内关联，这种情况从来不会发生，因为使用内扩展边扩展图模式时不会引进任何新结点。对于一般的外关联，我们无法保证例 3.4.4 中的情况不会发生。因此，我们不能像内关联那样简单地利用外关联来设计裁剪技术。

幸运的是，当外关联满足一些额外的约束时，我们可以保证例 3.4.4 中的情况不会发生。在介绍额外约束之前，再定义几个新概念。

在一个图中，不在任何环中出现的边称为**桥**。例如，在图 7 所示的 G_3 中，标号为 y 的两条边就是桥。

定义 3.4.5(桥关联) 设 G 是数据库中的一个图，图模式 p 是 G 的一个子图，(i,e_l,v_l) 是图模式 p 的外扩展边。如果对 p 到 G 的每一个子图同构 f，G 都有一个结点 v，并且同时满足如下条件:

(1) 结点 v 的标号是 v_l;

(2)$(f(i),v)$ 是一条标号为 e_l 的边;

(3) 不存在 p 中的结点 j 使得 $f(j)=v$;

(4)$(f(i),v)$ 是 G 中的桥，

那么称在 G 上 $e=(i,e_l,v_l)$ 桥关联于 p。

定义 3.4.6(桥关联数)　设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。在 D 上 e 相对于 p 的桥关联数的定义为 $\text{Ass\_Bridge}_D(e,p)=|\{G_i|G_i\in D$，在 G_i 上 e 桥关联于 $p\}|$。

例 3.4.5　边 $e=(1,y,C)$ 是图模式 $p=\{(0,1,A,x,B)\}$ 的外扩展边。对于图 7 中的数据库 D，可以验证在 G_1，G_2 和 G_3 上 e 外关联于 p，$\text{Ass\_External}_D(e,p)$=3。然而，只在 G_3 上，e 桥关联于 p。因此，$\text{Ass\_Bridge}_D(e,p)$=1。

根据外关联和桥关联的定义可以看出桥关联是一种特殊的外关联。因此，桥关联数应小于或等于外关联数，即 $\text{Ass\_Bridge}_D(e,p)\leqslant\text{Ass\_External}_D(e,p)$。使用桥关联，我们有下面的引理。

引理 3.4.4　设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。若 $\text{Ass\_Bridge}_D(e,p)\geqslant\text{support}(p)-\Delta, p\subset q, p\diamond e\not\subseteq q$，则 $\text{support}(q)-\text{support}(q\diamond e)\leqslant\Delta$。

证明　类似于引理 3.4.3 的证明。　□

根据引理 3.4.1，引理 3.4.2 和引理 3.4.4，我们可以为 Δ-跳跃模式导出如下基于外扩展的剪枝条件。

定理 3.4.3　设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。如果 $\text{Ass\_Bridge}_D(e,p)\geqslant\text{support}(p)-\Delta$，$p\diamond e<p$，那么在 DFS 编码树上 p 的所有后裔都不是 Δ-跳跃模式。

证明　参照定理 3.4.1 的证明过程，根据引理 3.4.1，引理 3.4.2 和引理 3.4.4 容易证明该定理。　□

例 3.4.6　演示定理 3.4.3 的作用。假定用 min_sup = 2 和 Δ=1 从图 2 所示的数据库 D 中挖掘频繁 Δ-跳跃模式。生成的 DFS 编码搜索树如图 6 所示。该搜索树中结点 19 对应的图模式为 $p=\{(0,1,D,y,E)\}$，边 $e=(0,x,D)$ 是 p 的外扩展边。根据最小 DFS 编码的定义，有 $p\diamond e<p$。容易计算 $\text{support}(p)$=4，$\text{Ass\_Bridge}_D(e,p)$=3。因此，$\text{Ass\_Bridge}_D(e,p)\geqslant\text{support}(p)-\Delta$。根据定理 3.4.3，以结点 19 为根的分枝不含有 Δ-跳跃模式，所以该分枝可以安全地被裁剪掉。

除了使用桥关联，在某些特殊的情况下仅仅使用外关联也可以得到有效的剪枝条件，如定理 3.4.4 所示。

定理 3.4.4　设 D 是一个图数据库，$e=(i,e_l,v_l)$ 是图模式 p 的外扩展边。如果 $\text{Ass\_External}_D(e,p)\geqslant\text{support}(p)-\Delta$，并且根据标号集上定义的顺序 v_l 小于 p 中所有结点的标号，则在 DFS 编码树上 p 的所有后裔都不是 Δ-跳跃模式。

证明　设在 DFS 编码树上 q 是 p 的任意一个后裔。根据引理 3.4.1，我们有 $p\subset q$，$p<q$。现在用反证法证明 q 中不含有标号为 v_l 的结点。若 q 中含有标号为 v_l 的结点，根据最小 DFS 编码的定义，我们有 $q<p$。这与 $p<q$ 相矛盾。

把 p 的支持集分成不相交的两个子集 S_1 和 S_2，其中 $S_1=\{G_i|G_i\in D$，在 G_i 上

e 外关联于 p}，S_2={$G_i|G_i \in D$，在 G_i 上 e 不与 p 外关联}。显然，support(p) = $|S_1|+|S_2|$，Ass_Internal(e,p)=$|S_1|$。因为 $p \subset q$，我们可以假定在 S_1 中有 x 个图含有 q，在 S_2 中有 y 个图含有 q。因此，support(q) = $x+y$。设 G 是 S_1 中含有 q 的任意一个图，I_q 是 q 在 G 中的任意一个实例。因为 $p \subset q$，I_q 也含有 p 在 G 中的一个实例 I_p。因为在 G 上 e 外关联于 p，I_p 能被扩展成 $I_p \diamond e$，$I_p \diamond e$ 中引进了一个标号为 v_l 的新结点 v。因为 q 中不含有标号为 v_l 的结点，所以 I_q 中不含有这个新结点 v。因此，I_q 能被扩展得到 $I_q \diamond e$，G 含有 $q \diamond e$。因为 G 是 S_1 中含有 q 的任意一个图，我们有 support($q \diamond e$) $\geqslant x$。因此，support(q) $-$ support($q \diamond e$) $\leqslant (x+y)-x = y \leqslant |S_2|$ = support(p) $-$ $|S_1|$ = support(p) $-$ Ass_External(e,p) $\leqslant \Delta$。根据 Δ-跳跃模式的定义，q 不是 Δ-跳跃模式。 □

例 3.4.7 演示定理 3.4.4 的作用。假定用 min_sup = 2 和 $\Delta = 1$ 从图 2 所示的数据库 D 中挖掘频繁 Δ-跳跃模式。生成的 DFS 编码搜索树如图 6 所示。该搜索树中结点 13 对应的图模式为 $p = \{(0,1,B,z,C)\}$，边 $e = (0,y,A)$ 是 p 的外扩展边。容易计算 support(p)=4，Ass_External$_D$(e,p)= 3。因此，Ass_External$_D$(e,p) $\geqslant$ support(p) $-\Delta$。而且，标号 A 小于 p 中所有结点的标号。定理 3.4.4 的条件被满足。因此，以结点 13 为根的分枝可以被安全地裁剪掉。

利用外扩展边，定理 3.4.3 和定理 3.4.4 给出了两种不同的裁剪技术。这两种裁剪技术被称为**基于外扩展的裁剪**。类似地，我们也为 δ-跳跃模式设计了如下基于外扩展的裁剪技术。

定理 3.4.5 设 D 是一个图数据库，min_sup 是最小支持度，$e = (i, e_l, v_l)$ 是图模式 p 的外扩展边。如果 Ass_Bridge$_D$(e,p) $\geqslant$ support(p) $-$ min_sup $\times\, \delta$，$p \diamond e < p$，那么在 DFS 编码树上 p 的所有后裔都不是频繁 δ-跳跃模式。

证明 参考定理 3.4.2 和定理 3.4.3 的证明过程，很容易证明该定理。 □

定理 3.4.6 设 D 是一个图数据库，min_sup 是最小支持度，$e = (i, e_l, v_l)$ 是图模式 p 的外扩展边。如果 Ass_External$_D$(e,p) $\geqslant$ support(p) $-$ min_sup $\times\, \delta$，并且根据标号集上定义的顺序 v_l 小于 p 中所有结点的标号，那么在 DFS 编码树上 p 的所有后裔都不是频繁 δ-跳跃模式。

证明 参考定理 3.4.2 和定理 3.4.4 的证明过程，很容易证明该定理。 □

3.4.5 GraphJP 算法

使用 3.4.3 节和 3.4.4 节给出的裁剪技术，可以裁剪掉大量的非频繁跳跃模式。然而，剩余的频繁图模式也不全是跳跃模式。我们需要从剩余的频繁图模式中选出跳跃模式。幸运的是，由 3.4.3 节和 3.4.4 节的裁剪技术，这很容易实现。为了尽可能早地

裁剪图模式搜索空间，对每一个频繁图模式 p，我们检查 p 的所有可能的扩展边，以判断 3.4.3 节和 3.4.4 节中的剪枝条件是否能被满足。因此，对每一个频繁图模式 p，我们得到了 p 的所有多一条边的超图及其支持度，根据跳跃模式的定义，我们可以直接判断 p 是否是跳跃模式。

本小节将 3.4.3 节和 3.4.4 节的裁剪技术集成到 DFS 编码搜索框架中，给出挖掘频繁 Δ-跳跃模式的完整算法 GraphJP，如算法 6 所示。

Algorithm 6: GraphJP 算法

Input: 图数据库 D，最小支持度 min_sup，跳跃距离阈值 Δ
Output: D 中所有频繁 Δ-跳跃模式集合 JS

```
1 扫描 D 一次，得到所有频繁边;
2 删除 D 中不频繁的结点和边;
3 检索并标记 D 中每个图的所有桥;
4 S^1 = D 中所有频繁 1-边子图的最小 DFS 编码
5 JS = ∅;
6 for S^1 里的每个 DFS 编码 p do
7     调用 MiningJumpPatterns(p, D, min_sup, Δ, JS);
8 输出 JS;
```

算法 GraphJP 有如下工作: 初始化时，算法先扫描数据库，得到频繁边集合，然后删除数据库中不频繁的边和结点，最后检测并标记数据库中每个图的所有桥（定理 3.4.3 的裁剪技术需要利用这些信息）。在这之后，对每个 1-边频繁子图，算法调用子过程 MiningJumpPatterns(如算法 7 所示) 进行深度优先搜索，发现所有频繁 Δ-跳跃模式。

在子过程 MiningJumpPatterns 的第 1 行，扫描数据库 D，发现当前图模式 p 的所有可能扩展边（包括非最右扩展边）。所有非最右扩展边记录在 NRE 中，所有最右扩展边记录在 RE 中。根据 DFS 编码搜索树的构造，如果边 e 不是 p 的最右扩展，$p \diamond e$ 将在 p 之前被发现，而且 $p \diamond e < p$。因此，对 NRE 中的每条边 e 都有 $p \diamond e < p$。第 2-4 行对 NRE 中的内扩展边应用基于内扩展的裁剪技术。如果 NRE 中存在内扩展边 e，使得 $\text{Ass\_Internal}(e,p) \geqslant \text{support}(p) - \Delta$，根据定理 3.4.1，说明以 p 为根的分枝不含有 Δ-跳跃模式。因此，可以安全地裁剪以 p 为根的分枝，从子过程 MiningJumpPatterns 返回。类似地，第 5-10 行对 NRE 中的外扩展边应用基于外扩展的裁剪技术 (定理 3.4.3 和定理 3.4.4)。第 11 行计算当前图模式 p 的绝对跳跃值 $JV_{\text{abs}}(p)$，因为在第 1 行已经枚举了 p 的所有扩展边，从而得到了 p 的所有多一条边的超图的支持度。如果 $JV_{\text{abs}}(p)$ 大于 Δ，说明 p 是频繁 Δ-跳跃模式，将 p 放入结果集 JS 中。注意: 尽管算法检查 p 的所有可能扩展边，然而像 gSpan [145] 算法那样，只对 p 进行最右扩展。对 RE 中的每个最右扩展边 e（第 16 行），如果 $p \diamond e$ 是频繁的，

Algorithm 7: MiningJumpPatterns 算法

Input: 一个最小的 DFSCode p，图数据库 D，最小支持度 min_sup，跳跃距离阀值 Δ，频繁 Δ-跳跃模式集合 JS

Output: 频繁 Δ-跳跃模式集合 JS

扫描 D 一次，发现所有边 e 使得 p 能被 e 扩展成子图 $p \diamond_x e$；RE 记录 p 的所有最右扩展，NRE 记录 p 的所有非最右扩展；

if $\exists e \in NRE$ 满足 $Ass\_Internal(e,p) \geqslant \text{support}(p) - \Delta$ **then**

    返回；/* 定理 5.4.1 的裁剪技术 */

if $\exists e \in NRE$ 满足 $Ass\_Bridge(e,p) \geqslant \text{support}(p) - \Delta$ **then**

    返回；/* 定理 5.4.3 的裁剪技术 */

if $\exists e \in NRE$ 满足 $Ass\_External(e,p) \geqslant \text{support}(p) - \Delta$，并且 e 的一个结点标号小于 p 中所有结点的标号 **then**

    返回；/* 定理 5.4.4 的裁剪技术 */

根据 RE 和 NRE 计算 p 的绝对跳跃值 $JV_{\text{abs}}(p)$

if $JV_{\text{abs}}(p) > \Delta$ **then**

    把 p 放入 JS；

根据字典 DFS 顺序，对 RE 中的所有最右扩展进行排序；

for RE 里的每个最右扩展边 e **do**

    if $p \diamond e$ 是频繁的 *(w.r.t.* min_sup*)*，并且 $p \diamond e = \min(p \diamond e)$ **then**

        调用 MiningJumpPatterns $(p \diamond e, D, \text{min\_sup}, \Delta, JS)$；

    if $Ass\_Internal(e,p) \geqslant \text{support}(p) - \Delta$ **then**

        跳出 for 循环；/* 定理 5.4.1 的裁剪技术 */

    if $Ass\_Bridge(e,p) \geqslant \text{support}(p) - \Delta$ **then**

        跳出 for 循环；/* 定理 5.4.3 的裁剪技术 */

    if $Ass\_External(e,p) \geqslant \text{support}(p) - \Delta$，并且 e 的一个结点标号小于 p 中所有结点的标号 **then**

        跳出 for 循环；/*
        定理 5.4.4 的裁剪技术 */

并且 $p \diamond e$ 是对应图模式的最小 DFS 编码，那么算法对 p 的孩子 $p \diamond e$ 递归调用子过程 MiningJumpPatterns 继续深度优先搜索（第 18 行）。在处理完 p 的一条最右扩展边 e 之后（完成对 $p \diamond e$ 的递归调用），算法可以利用边 e 再次应用裁剪技术。第 20-22 行再次使用基于内扩展的裁剪技术，设 e' 是剩余的任意一条最右扩展边，$q{=}p \diamond e'$ 是 p 的一个剩余孩子，显然，$q \diamond e < q$。如果 $\text{Ass\_Internal}(e, p) \geqslant \text{support}(p) - \Delta$，容易证明 $\text{Ass\_Internal}(e, q) \geqslant \text{support}(q) - \Delta$。根据定理 3.4.1，可知以 q 为根的分枝不含有 Δ-跳跃模式。因此，如果 $\text{Ass\_Internal}(e, p) \geqslant \text{support}(p) - \Delta$，说明在以 p 的剩余孩子为根的分枝中都不含有 Δ-跳跃模式，那么算法可以扔掉 p 的剩余的最右扩展边，从子过程 MiningJumpPatterns 返回。类似地，第 23-28 行再次应用基于外扩展的裁剪技术（定理 3.4.3 和定理 3.4.4）。算法 GraphJP 完成之后，JS 中包含所有频繁 Δ-跳跃模式。

下面给出算法 GraphJP 正确性的证明，算法的效率将在实验中得到验证。

定理 3.4.7　算法 GraphJP 产生的结果是正确而完备的。

证明　正确性证明。设 p 是结果集 JS 中的任意一个图模式。根据算法的执行步骤可知，p 只能在子过程 MiningJumpPatterns 的第 13 行被加入 JS。根据第 12 行的测试条件和 Δ-跳跃模式的定义，可知 p 是 Δ-跳跃模式。又因为进入子过程 MiningJumpPatterns（被算法 GraphJP 第 7 行和子过程 MiningJumpPatterns 第 18 行调用）的任何图模式一定是频繁的，因此 p 是频繁 Δ-跳跃模式。

完备性证明。设 p 是数据库中的任意一个频繁图模式。如果 p 不在结果集 JS 中，那么说明 p 可能在子过程 MiningJumpPatterns 中的第 3 行，第 6 行，第 9 行，第 21 行，第 24 行，第 27 行被裁剪掉，也可能在第 12 行被过滤掉。如果 p 在第 3 行或第 21 行被裁剪掉，根据定理 3.4.1，可知 p 不是 Δ-跳跃模式。如果 p 在第 6 行或第 24 行被裁剪掉，根据定理 3.4.3，可知 p 不是 Δ-跳跃模式。如果 p 在第 9 行或第 27 行被裁剪掉，根据定理 3.4.4，可知 p 不是 Δ-跳跃模式。如果 p 在第 12 行被过滤掉，根据 Δ-跳跃模式的定义，可知 p 不是 Δ-跳跃模式。因此，如果一个频繁图模式 p 不在结果集 JS 中，说明 p 一定不是 Δ-跳跃模式。　□

类似于上面给出的 GraphJP 算法，根据定理 3.4.2，定理 3.4.5 和定理 3.4.6 给出的裁剪技术，我们也很容易设计一个挖掘频繁 δ-跳跃模式的算法。

3.5　实验结果及分析

我们进行了大量的实验来考查算法的执行效率、可扩展性、跳跃阀值的作用、裁剪技术的有效性以及挖掘结果的可用性。

3.5.1 实验设置

在实验中，我们使用化合物数据 PTE 和 AIDS。PTE 数据含有 340 个化合物。文献 [83] 显示: 尽管 PTE 数据集只含有少量化合物，然而该数据集中大部分化合物的尺寸都很大，而且可以共享大的化学结构。这些大的化学结构很可能是化合物的功能团。AIDS 数据含有大约 44 000 个化合物。AIDS 化合物根据对艾滋病病毒的抑制作用可以被分成三类:CA(confirmed active)，CM(confirmed moderately) 和 CI(confirmed inactive)。其中，CA 含有 422 个化合物，CM 含有 1 081 个化合物，CI 含有剩余的化合物。文献 [146] 显示:CA 中的化合物可以共享大的化学片段 (chemical fragment)，这些大的化学片段很可能是化合物的功能团。而 CM 和 CI 中的化合物更加多样化，不以某些特定的化学片段为中心。与 CM 和 CI 相比，CA 数据集更有意义，因为从 CA 数据集中可以选择或合成新的抗艾滋病药物。因为我们的算法适合挖掘化合物的核心子结构 (功能团)，所以我们的算法更适合挖掘 CA 数据集。此外，为了测试算法在不同特征数据库中的效率，我们也使用文献 [83] 中的图数据生成器生成了具有各种不同特征的图集合。生成器的参数含义在文献 [83] 中有详细的描述。化合物主要以树结构为主，为了测试内关联裁剪技术的有效性，我们让合成数据含有更多的环结构。

本章中的算法使用 C++ 语言实现，用带有-O3 优化选项的 g++ 编译。用于实验的计算机具有 PIV 3.0GHz CPU 和 1GB 内存，运行 RedHat Linux 8.0 操作系统。

据我们所知，GraphJP 是第一个从图数据库中挖掘频繁跳跃模式的算法，不借助其他算法也能完成同样的任务。因此，我们可以从各种角度来考查算法 GraphJP。

3.5.2 跳跃距离阀值的作用

本小节考查不同的跳跃距离阀值对输出的跳跃模式数量和挖掘效率的影响。我们选择了 PTE，CA 和 D1kV4E2I5T20L20 三个图集合。对每个图集合，固定一个尽可能小的支持度，同时变化跳跃距离阀值 Δ 的大小。图 8 显示了在 PTE 数据集上当 Δ 值变化时，输出的频繁 Δ-跳跃模式数量和算法 GraphJP 的挖掘时间。图 9 显示了在 CA 数据集上的实验结果，图 10 显示了在 D1kV4E2I5T20L20 数据集上的实验结果。

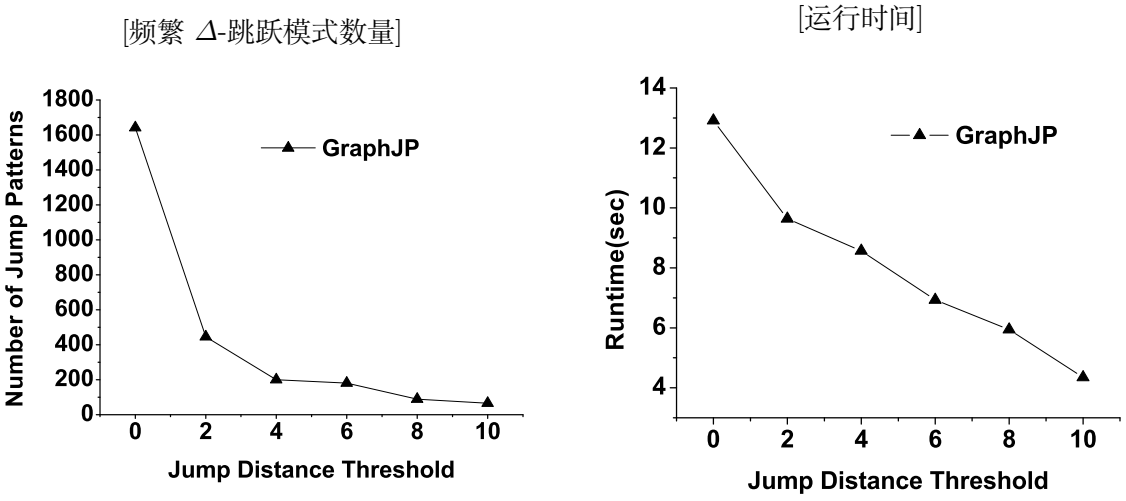


图 8　在 PTE 数据集上的实验结果

从图 8 ∼ 图 10 可以看出，当 Δ 值增加时，频繁 Δ-跳跃模式的数量下降。根据 Δ-跳跃模式的定义可知，当 Δ 值增加时，意味着更多的频繁图模式能被过滤掉，因而频繁 Δ-跳跃模式的数量减少。在实验中，我们发现了一个非常有意思的现象: 当 Δ 值从 0 开始稍微增加时，频繁 Δ-跳跃模式的数量就急剧下降。例如: 在 PTE 数据集上，当 Δ 值从 0 变化到 2 时，频繁 Δ-跳跃模式的数量从 1 641 急剧下降到 445。因为 0-跳跃模式等价于闭图模式 (见引理 3.3.3)，所以，当 Δ 稍微大于 0 时，频繁 Δ-跳跃模式的数量就比频繁闭图模式的数量少很多。在实际应用中，用户往往希望得到一个小的而有意义的图模式集合，而有意义的图模式又是那些具有高跳跃值的跳跃模式 (见 3.5.6 节)。因此，在实际应用中，我们可以使用大的 Δ 值来挖掘图数据库，快速地得到一个小的而有意义的图模式集合。

从图 8 ∼ 图 10 可以看出，随着 Δ 值的增加，算法的运行时间也在迅速地减少，这很容易理解。根据定理 3.4.1，定理 3.4.3 和定理 3.4.4 可知，当 Δ 值增加时，裁剪条件更容易被满足，更多不含有 Δ-跳跃模式的分枝可以被安全地裁剪掉，因而算法的挖掘效率也被大大地改善。例如: 在 CA 数据集上，当 Δ=0 时，GraphJP 需要大约 150 秒来完成挖掘任务。然而，当 Δ=10 时，GraphJP 需要大约 50 秒就能完成挖掘任务。

我们也考查了相对的跳跃距离阀值 δ 对输出的频繁 δ-跳跃模式数量和挖掘效率的影响。图 11 显示了在 CA 数据集上，当 δ 值变化时，输出的频繁 δ-跳跃模式的数量和挖掘算法所用的时间。类似于 Δ 的作用，当 δ 值增加时，频繁 δ-跳跃的数量迅速地减少，而算法的运行效率被快速地改善。

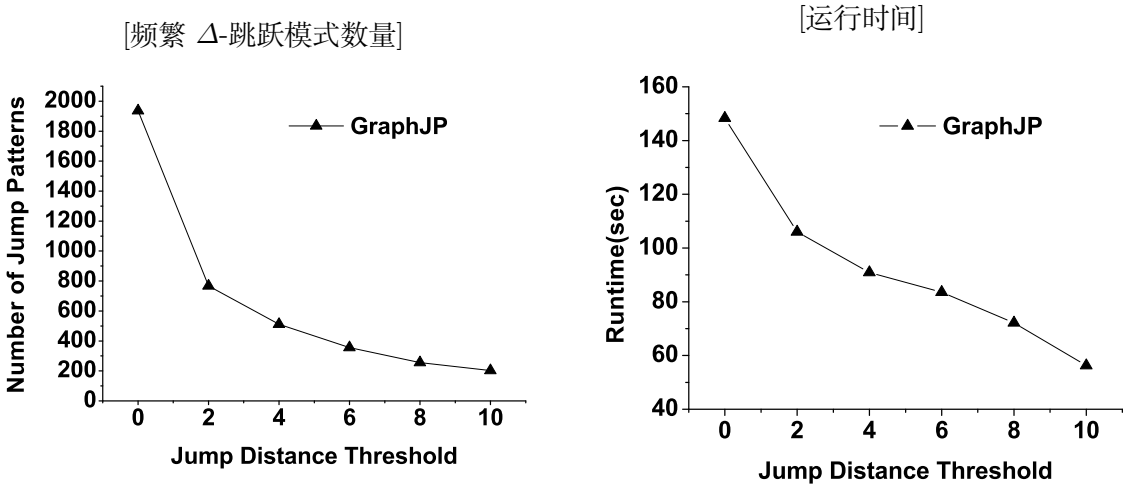


图 9　在 CA 数据集上的实验结果

3.5.3　裁剪技术的有效性

本小节评价算法 GraphJP 中使用的裁剪技术的有效性。表 1 显示了各种裁剪技术组合而得到的算法的变体。

表 1　GraphJP 算法的变体

算法的变体	含义
GraphJP-NP	在 GraphJP 不使用任何裁剪技术
GraphJP-IP	在 GraphJP 只使用基于内扩展的裁剪技术
GraphJP-EP	在 GraphJP 只使用基于外扩展的裁剪技术
GraphJP	在 GraphJP 使用所有裁剪技术

我们选择 PTE，CA 和 D1kV4E2I5T20L20 三个图集合。对每个图集合，固定跳跃距离阀值 Δ 等于 2，变化最小支持度 min_sup 的大小。图 12 显示了在 PTE 数据集上当支持度变化时，不同算法的运行时间和枚举的子图数量。图 13 显示了在 CA 数据集上的实验结果，图 14 显示了在 D1kV4E2I5T20L20 数据集上的实验结果。从图 12 ~ 图 14 可以看出，基于内扩展的裁剪技术和基于外扩展的裁剪技术能极大地改善算法 GraphJP 的效率。而且，支持度越低，这两种裁剪技术越有效。例如: 在 CA 数据集上，当支持度是 5% 时，GraphJP 比 GraphJP-NP 快 1 个数量级还多。

此外，我们也观察到在化合物数据集上，基于外扩展的裁剪技术比基于内扩展的裁剪技术更有效。这是因为化合物中主要以稀疏图为主，频繁图模式集合中的大部分都是树结构。因此，外扩展边的数量要明显多于内扩展边的数量，从而使得基于外扩展的裁剪条件更容易被满足。相反的，合成图集合 D1kV4E2I5T20L20 中含有更多的

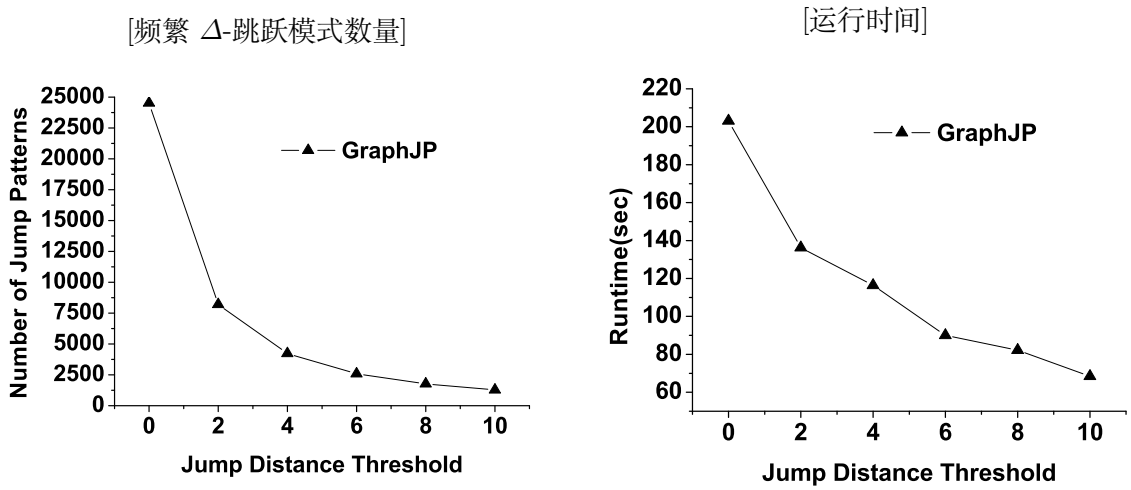


图 10　在 D1kV4E2I5T20L20 数据集上的实验结果

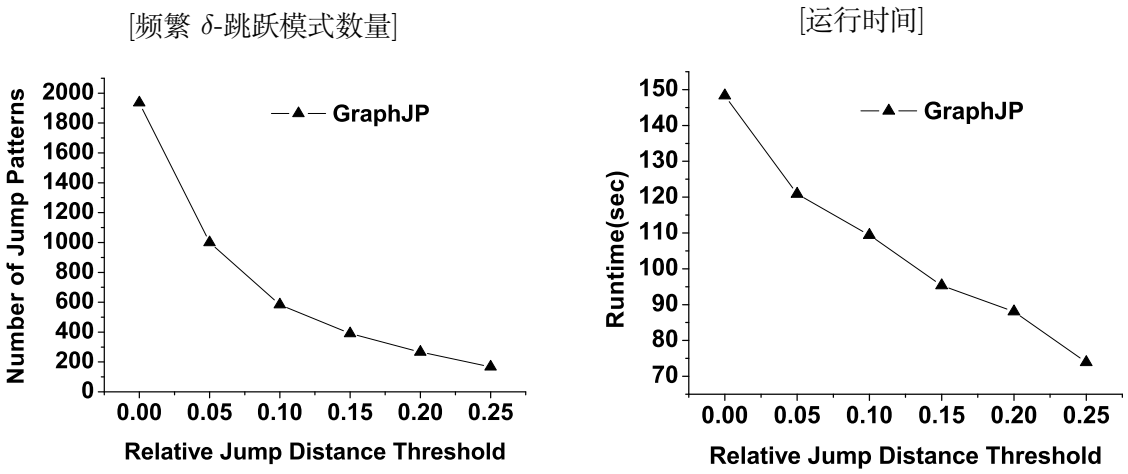
环，这使得内扩展边的数量多于外扩展边的数量，因此在这个数据集上基于内扩展的裁剪技术比基于外扩展的裁剪技术更有效。

在实验中，我们也改变了图的密度，测试不同的图密度对裁剪技术的影响。我们发现，图密度越大，基于内扩展的裁剪技术就越有效。反之，图密度越小，基于外扩展的裁剪技术就越有效。

3.5.4　与 CloseGraph 比较

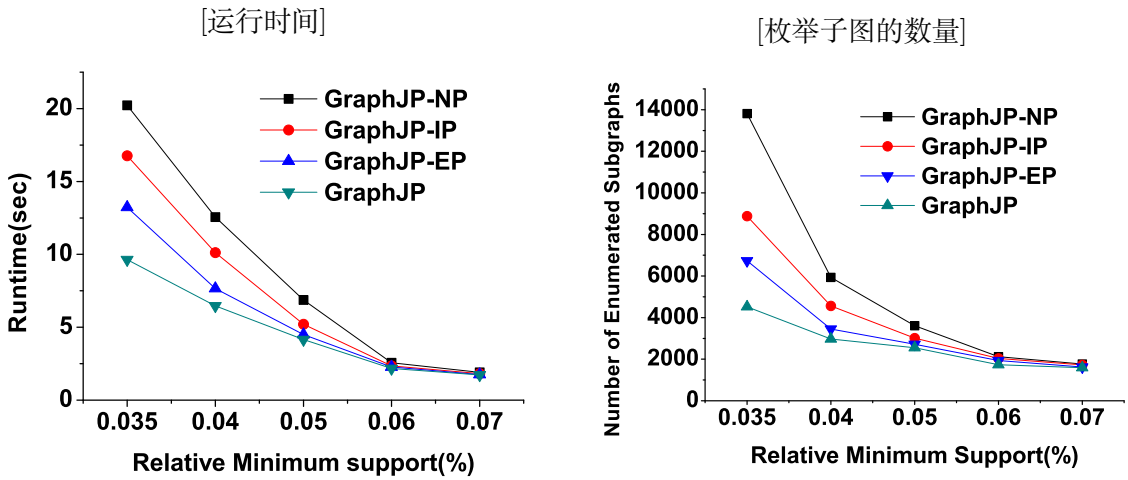
根据引理 3.3.3，闭图模式等价于 0-跳跃模式。因此，算法 GraphJP 也可以直接用来挖掘频繁闭图模式。我们将著名的闭图模式挖掘算法 CloseGraph [146] 与 GraphJP 进行比较。图 15 显示了在 PTE 数据集和 CM 数据集上变化支持度大小时，GraphJP(Δ=0) 和 CloseGraph 的运行时间的比较。

可以看出，GraphJP 优于 CloseGraph。总的来说，挖掘效率能提高 20% 左右。这是因为，CloseGraph 为了保证挖掘结果的完整性使用了失败检测机制。然而，正如文献 [146] 所指出的那样，失败检测有时侯能使 CloseGraph 的性能下降大约一半。而我们的算法 GraphJP 利用桥约束 (定理 3.4.3) 来保证不丢失任何挖掘结果，从而完全避免了失败检测，提高了算法的挖掘效率。更重要的是，当 Δ 大于 0 时，CloseGraph 并不能挖掘频繁 Δ-跳跃模式，而 GraphJP 随着 Δ 值的增加可以更高效地挖掘频繁 Δ-跳跃模式 (见 3.5.2 节)。



(a) (b)

图 11　在 CA 数据集上的实验结果，测试 δ 的作用



(a) (b)

图 12　在 PTE 数据集上裁剪技术的有效性，Δ=2

3.5.5　算法可扩展性

本小节研究算法 GraphJP 的可扩展性。我们使用参数 V4E2I5T20L20 生成了从 10K 到 100K 的合成数据，固定 min_sup = 6%，Δ=2，运行 GraphJP，实验结果如图 16 所示。从图 16 可以看出，算法 GraphJP 有很好的可扩展性。随着数据量的增加，GraphJP 的运行时间也在线性地增加。

3.5.6　跳跃模式的可用性

本小节考查算法 GraphJP 挖掘结果的可用性。我们显示 CA 数据集上有意义的图模式是那些具有高跳跃值的跳跃模式。CA 数据集中的化合物被分成如下几个化学

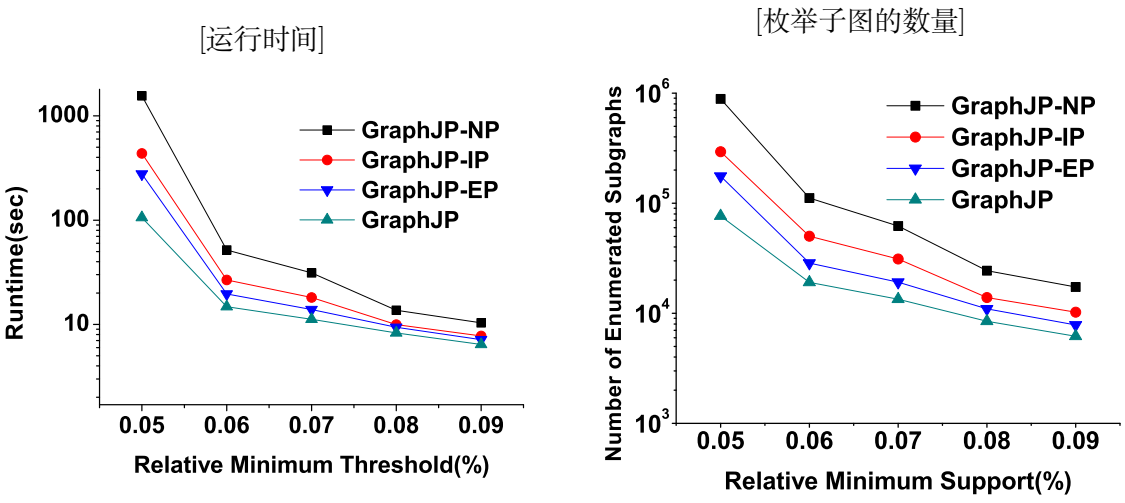
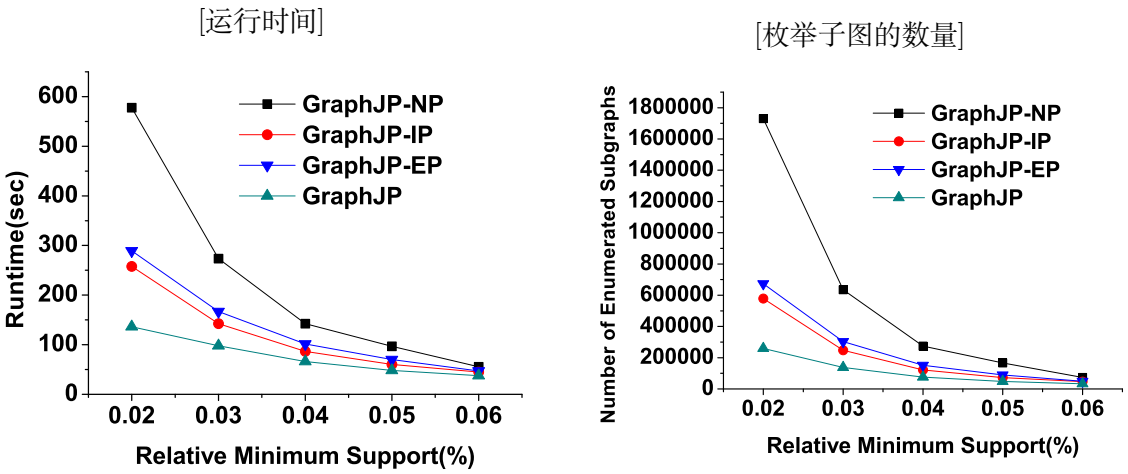


图 13　在 CA 数据集上裁剪技术的有效性，Δ=2

类别 [2]:Azido Pyrimidines，Dyes Polyanions，Pyrimidine Nucleosides，Heavy Metal Compounds，Purine Nucleosides。我们发现每类中的核心子结构 (最大公共子结构) 都是 CA 数据集上具有高跳跃值的跳跃模式。例如: 图 17 显示了 Azido Pyrimidines 类的核心子结构，它的支持度是 64，绝对跳跃值是 13，相对跳跃值是 0.203 15。Azido Pyrimidines 类中的化合物 AZT 目前是一种使用最为广泛的抗艾滋病药物 [2]，而且 AZT 只比该类的核心子结构多一条边。图 18 显示了另一类化合物 Dyes Polyanions 的核心子结构，它的支持度是 52，绝对跳跃值是 16，相对跳跃值是 0.307 962。其他类化合物的核心子结构也都有类似的特点，即具有高跳跃值。显然，从化学角度讲每类中的核心子结构可以代表该类的一个功能团。这些功能团可以为新药物的设计提供重要的指导信息，因而具有重要的实际意义。因为这些功能团 (核心子结构) 都是具有高跳跃值的跳跃模式，根据前面实验中的参数设置，它们都将被算法 GraphJP 发现。

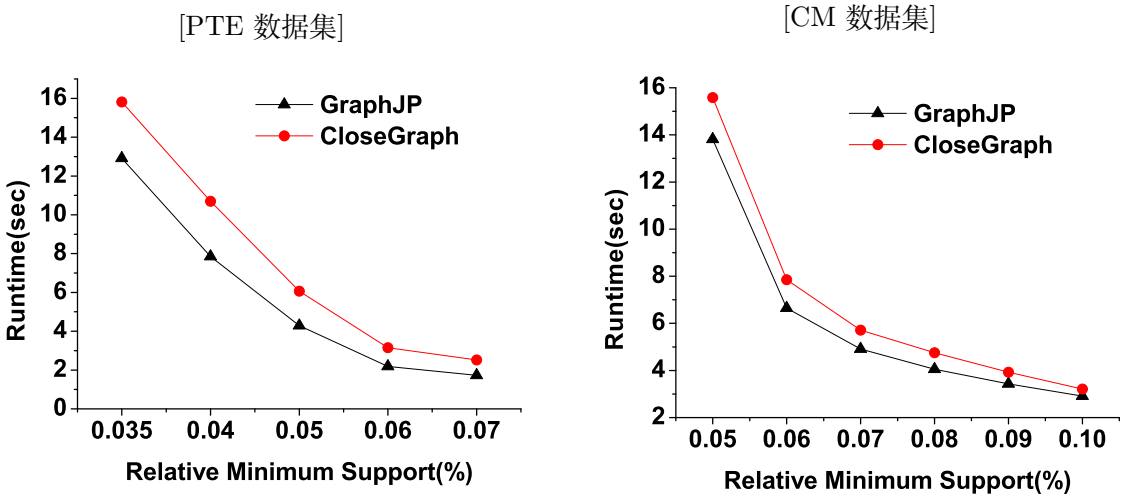
3.6　本章小结

本章提出了从图数据库中挖掘频繁跳跃模式的问题，并给出了解决该问题的一个高效算法 GraphJP。为了有效地裁剪图模式搜索空间，本章提出了两种新的裁剪技术，即基于内扩展的裁剪和基于外扩展的裁剪。实验结果表明，这些裁剪技术能使算法的性能提高一个数量级，GraphJP 算法具有线性扩展性和高挖掘效率。在化合物上的实验结果显示 GraphJP 能高效地挖掘图数据库中的核心子结构 (化合物中的功能团)。因为图代表了通用的模式类型，本章给出的度量定义和挖掘方法也容易被扩展来挖掘其他的模式类型，例如: 项集模式、序列模式、树模式等。



(a) (b)

图 14　在 D1kV4E2I5T20L20 数据集上裁剪技术的有效性，Δ=2



(a) (b)

图 15　比较 GraphJP(Δ=0) 和 CloseGraph

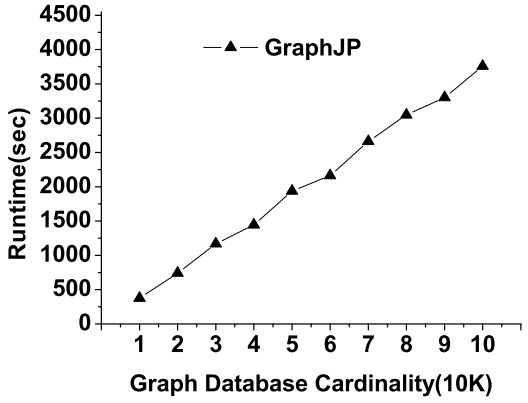


图 16　在 V4E2I5T20L20 上的可扩展性，min_sup = 6%，$\Delta = 2$

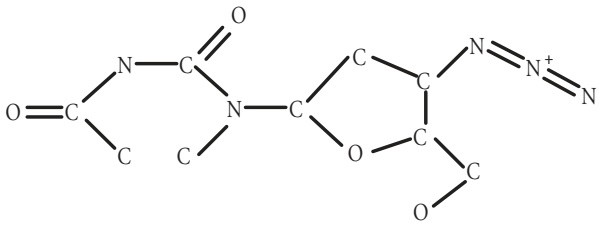


图 17　在 Azido Pyrimidines 类的核心子结构

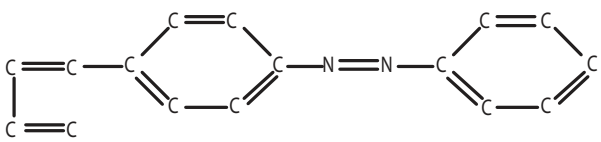


图 18　在 Dyes Polyanions 类的核心子结构

第 4 章　基于联合意义度量的 Top-K 图模式挖掘

4.1　引言

近年来，科学与工程领域积累了大量可以用图来建模的数据，如化合物分子结构、蛋白质交互网络、基因相关网络、社会网络、传感器网络等。目前，很多与图有关的应用都需要利用图模式来管理、查询和分析图数据。例如: 在图查询 [34,147] 中，利用图模式可以建立有效的索引，提高查询处理效率；在图分类 [43] 中，利用图模式可以建立有效的分类模型，提高分类准确率。因此，从图数据库中发现有用的图模式已经成为数据挖掘领域一项重要的研究课题。

目前，与图模式挖掘有关的绝大部分研究主要集中在如何高效地挖掘频繁子图 [20,66,69,83,103,145]，及频繁子图的各种简洁表示 (频繁闭图模式 [146] 和极大频繁图模式 [67,122])。然而，很少有研究考虑如何根据用户给定的意义度量来挖掘图模式。本质上，频繁子图挖掘所使用的支持度恰恰是一种特殊的意义度量。在不同的应用中，用户很可能需要不同的意义度量。因此，设计针对通用意义度量的图模式挖掘算法将有广泛的应用背景。

给定一个意义度量，一个直观的图模式挖掘方法就是传统 Top-K 挖掘，即对所有可能的图模式根据度量值的大小递减排序，输出前 K 个图模式。然而，传统 Top-K 挖掘并不考虑图模式之间的相关性，输出的 Top-K 模式可能非常相似。图 1 显示了信息增益作为意义度量时，传统 Top-K 挖掘方法从一个化合物集合中挖掘的 Top-5 图模式。这 5 个图模式在结构上非常相似。如果用户得到其中一个图模式，则对其他图模式就失去了兴趣，因为在实际应用中用户希望得到的是一个多样化的图模式集合。

为了克服传统 Top-K 挖掘方法的缺点，本章研究基于联合意义度量的 Top-K 图模式挖掘方法。联合意义度量的作用域是图模式集合而不是图模式，因此充分考虑了图模式之间的相关性。给定一个联合意义度量，本章要研究的问题就是挖掘 Top-K 图模式，联合起来使该意义度量最大化。本章的目标是设计一个适用于任何联合意义度量的通用 Top-K 挖掘算法。

本章首先讨论了适用于图模式集合的联合意义度量，并利用信息论中的概念 (联合熵和信息增益) 给出了两个具体的问题定义 MES 和 MIGS，然后证明了它们是 NP-hard 问题。为了高效地挖掘联合意义度量下的 Top-K 图模式，本章还给出了两个算法 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 先产生频繁图模式 (或频繁闭图模

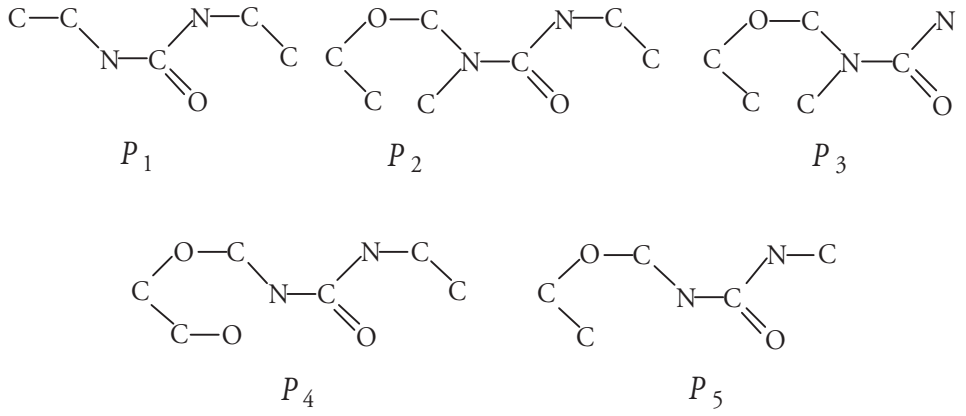


图 1　传统 Top-K 方法产生的 Top-5 图模式

式)，然后增量贪心地选择 K 个图模式。我们证明了如果用户给定的意义度量满足 submodular 性质，Greedy-TopK 能提供近似比保证。为了进一步提高 Greedy-TopK 的效率，我们针对 MES 和 MIGS 这两个具体问题的意义度量设计了有效的裁剪技术，将其嵌入频繁子图挖掘框架中以帮助裁剪图模式搜索空间。然而，当频繁图模式 (或频繁闭图模式) 数量较大时，Greedy-TopK 效率低、可扩展性差。为此，我们又提出了一个更高效的算法 Cluster-TopK。Cluster-TopK 先从图数据库中挖掘所有频繁图模式的一个代表模式集合，然后从代表模式中增量贪心选择 K 个图模式。Cluster-TopK 最大的优点是无需产生频繁图模式 (或闭图模式) 就能快速地从图数据库中挖掘一个代表模式集合。而且，因为代表模式的数量比闭图模式的数量少很多，所以 Cluster-TopK 的贪心选择也比 Greedy-TopK 的贪心选择快很多。此外，我们还严格地从理论上证明了 Cluster-TopK 产生的解和 Greedy-TopK 产生的解非常接近。

大量真实数据上的实验结果表明本章定义的 Top-K 挖掘在结果质量和可用性方面要远远优于传统 Top-K 挖掘。Cluster-TopK 和 Greedy-TopK 挖掘的结果在质量和可用性方面都非常接近。然而，Cluster-TopK 能比 Greedy-TopK 要快 1 到 2 个数量级。

综上，本章的主要贡献如下: 第一，提出了一个新的研究问题，即挖掘基于联合意义度量的 Top-K 图模式。第二，分析了适用于图模式集合的联合意义度量，给出了两个具体的问题定义 MES 和 MIGS，并证明它们是 NP-hard 问题。第三，提出了两个高效算法 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 对满足 Submodular 性质的意义度量有近似比保证，但是当频繁子图数量较多时挖掘效率较低。理论分析表明 Cluster-TopK 产生的解和 Greedy-TopK 产生的解非常接近，而且 Cluster-TopK 的挖掘效率远远高于 Greedy-TopK 的挖掘效率。第四，进行了大量广泛的实验来考查 Cluster-TopK 和 Greedy-TopK 的效率、可扩展性以及挖掘结果的可用性。

本章的内容安排如下:4.2 节介绍相关工作；4.3 节介绍预备知识；4.4 节给出问题

定义和 NP-hard 问题证明；4.5 节给出完整的算法描述和算法分析；4.6 节通过实验验证算法的效率及挖掘结果的可用性；4.7 节对本章进行小结。

4.2　相关工作

频繁子图挖掘在数据挖掘领域中已经被广泛研究。很多高效的频繁子图挖掘算法已经被提出，大致可分为两类:(1) 基于 Apriori 的方法；(2) 模式增长 (pattern-growth) 方法。基于 Apriori 方法的宽度优先遍历模式搜索空间，在得到大小为 K 的所有频繁子图之后，先连接产生大小为 $K+1$ 的候选频繁子图，然后利用支持度的反单调性质删除候选子图中的非频繁子图，最后扫描数据库，确定剩余候选子图的支持度，得到大小为 $K+1$ 的所有频繁子图。基于 Apriori 方法的代表算法包括 AGM [68–70]，FSG [83,85]，Path-Join [45]，它们之间的区别在于 AGM，FSG，Path-Join 分别采用顶点、边、路径作为子图模式扩展的单元。模式增长方法采用深度优先方式遍历模式搜索空间，此方法通常的策略是，在某个频繁子图 p 的基础上，扩展产生 p 的孩子 (p 的超图模式) 并计算它们的支持度，对 p 的每个频繁孩子，以深度优先方式继续扩展，直到发现全部频繁子图为止。模式增长方法的代表算法包括 MoFa [20]，MoSS [21]，FFSM [66]，GASTON [103]，gSpan [145]。通常模式增长方法比基于 Apriori 方法有更好的内存利用率，因而具有更高的挖掘效率。上面所有这些算法都是挖掘完整的频繁子图集合，而实际应用中用户只需要其中一部分有意义的图模式。

当用户分析大的图数据库时，一个小的而有意义的图模式集合可以帮助用户快速地发现图数据库中的重要信息。除了利用图模式直观分析之外，图模式在图查询和图分类领域中也有广泛的应用。文献 [147] 显示用图模式作为索引特征可以明显地提高查询处理效率。文献 [31] 和 [43] 显示基于图模式的分类方法在保证高分类准确率的同时还具有高扩展性等优点。在这些与图模式有关的应用中，使用的图模式只是所有频繁子图中一部分有意义的图模式，如果使用所有频繁子图将会引起索引规模过大及“过适应” 问题。因此，如何高效地挖掘有意义的图模式将具有广泛的应用背景和重要的现实意义。

目前，只有少量研究工作考虑如何直接挖掘有意义的图模式。文献 [58] 评价了频繁子图的统计意义，该方法首先挖掘所有频繁子图，然后把每个频繁子图表示成一个特征向量，向量的基本元素根据领域知识由用户来选择。频繁子图的统计意义由对应向量支持度的 p-值来确定。该方法属于模式后处理方法，需要先挖掘所有频繁子图。当频繁子图很多时，该方法的效率低下。而且，缺乏领域知识的用户很难确定向量的基本元素及其先验概率。文献 [111] 提出了一个可扩展的算法 GraphSig，直接从图数据库中挖掘具有统计意义的图模式。GraphSig 需要根据领域知识选择一个有意义的特

征集合，特征的先验概率通过实验计算获得。GraphSig 把数据库中的每个图转换成一个特征向量集合，通过识别有意义的子特征向量来发现原始数据库中那些具有低支持度并且具有统计意义的图模式。该方法的弱点是缺乏领域知识的用户很难确定有意义的特征集合及其先验概率。文献 [144] 研究如何从图数据库挖掘使用户给定的意义度量最大化的图模式，并给出了一个通用的挖掘框架 LEAP。然而，该框架只输出一个意义度量最大的图模式。在实际应用中，用户经常需要多个有意义的图模式来管理和分析图数据。

给定一个意义度量，传统的 Top-K 方法根据给定的度量对图模式排序，输出前 K 个图模式。尽管该方法可以输出多个图模式，但没有考虑模式之间的相关性。如 4.6.2 节实验所示，该方法会输出结构上很相似的图模式，无法满足用户对图模式集合多样化的需求。而且，由于输出的图模式存在很大相关性，得到的 Top-K 图模式的整体作用会被严重削弱 (见 4.6.2 节)。

与上述工作不同，本章研究如何根据联合意义度量来挖掘 Top-K 图模式。联合意义度量隐含地考虑了图模式之间的相关性。因此，根据联合意义度量挖掘 Top-K 图模式将得到一个多样化而有意义的图模式集合。

4.3 预备知识

因为本章后面介绍的意义度量将使用信息论的一些概念 [41]，这里先回顾一下有关的基本定义。

定义 4.3.1(熵) 随机变量 x 的熵定义为 $H(x) = -\sum_{v_x \in \mathrm{dom}(x)} p(v_x)\log(p(v_x))$，其中 $\mathrm{dom}(x)$ 是 x 的定义域，$p(v_x)$ 是 x 等于 v_x 时的概率。

定义 4.3.2(条件熵) 在给定随机变量 x 的条件下，随机变量 y 的条件熵定义为 $H(y|x) = -\sum_{v_x \in \mathrm{dom}(x)} \sum_{v_y \in \mathrm{dom}(y)} p(v_x, v_y)\log(p(v_y|v_x))$，其中 $\mathrm{dom}(x)$ 是 x 的定义域，$\mathrm{dom}(y)$ 是 y 的定义域，$p(v_x, v_y)$ 是 x 等于 v_x，y 等于 v_y 时的联合概率，$p(v_y|v_x)$ 是在 x 等于 v_x 的条件下，y 等于 v_y 时的条件概率。

定义 4.3.3(联合熵) 两个随机变量 x 和 y 的联合熵定义为 $H(x,y) = -\sum_{v_x \in \mathrm{dom}(x)} \sum_{v_y \in \mathrm{dom}(y)} p(v_x, v_y)\log(p(v_x, v_y))$，其中 $\mathrm{dom}(x)$ 是 x 的定义域，$\mathrm{dom}(y)$ 是 y 的定义域，$p(v_x, v_y)$ 是 x 等于 v_x，y 等于 v_y 时的联合概率。

4.4 问题定义

本节首先讨论用于图模式集合的意义度量，然后给出两个具体的问题定义 MES 和 MIGS，最后证明它们是 NP-hard 问题。

文献中已有大量的用于单一模式 (包括项集模式、序列模式和图模式) 的意义度量。例如: 在统计学领域、x^2 检验和 Pearson 相关可以用来度量模式的统计意义。在数据挖掘和机器学习领域，信息增益和交叉熵可以用来度量模式是否适合作为分类特征。文献 [119] 总结了 21 个常用的意义度量。

尽管文献中给出的大部分意义度量都能直接用来度量单一图模式的意义，然而它们中的绝大部分并不能用来量化一个图模式集合的意义。本章算法需要用户提供一个意义度量 M 使得 M 能量化一个图模式集合的意义。信息论中的联合熵和信息增益可以用来量化一个图模式集合的意义。在给出具体的问题定义之前，我们先给出本章要研究的一般问题。

假设 S 是给定的图模式集合 (例如:S 可以表示某个图数据库中所有频繁图模式集合)，T 是 S 的子集合，$M(T)$ 是 T 的意义度量。本章要解决的问题就是发现 S 的一个大小为 K 的子集合 T^*，使得 $M(T^*)$ 被最大化，即

$$T^* = \arg\max_{T\subseteq S,|T|=k} M(T) \tag{4.1}$$

如果给定所有候选的图模式，上面定义的问题显然是一个组合优化问题。本章的目标是设计求解上述问题的通用算法，算法不受任何具体意义度量的限制。为了方便描述算法，我们利用信息论中的联合熵和信息增益给出两个具体的问题定义。

定义 4.4.1(基于熵的意义度量最大化)　给定图数据库 D，D 的一个图模式集合 S(或者最小支持度 min_sup) 和一个整数 K。最大化基于熵的意义度量问题 (**MES**) 就是从 S(或者 D 的所有频繁图模式集合) 中发现一个大小为 K 的子集合 T，使得 $H(T)$ 被最大化，其中 $H(T)$ 是 T 中随机变量 (图模式) 的联合熵。

定义 4.4.2(基于信息增益的意义度量最大化)　给定图数据库 D，D 的一个图模式集合 S(或者最小支持度 min_sup) 和一个整数 K。假设 D 中每个图都有一个类标记，用 C 表示类标记的随机变量。最大化基于信息增益的意义度量问题 (**MIGS**) 就是从 S(或者 D 的所有频繁图模式集合) 中发现一个大小为 K 的子集合 T，使得 $IG(T) = H(C) - H(C|T)$ 被最大化，其中 $H(C)$ 是 C 中随机变量的联合熵（无条件熵)，$H(C|T)$ 是 T 中随机变量 (图模式) 给定的条件下 C 的条件熵。

基于熵的意义度量经常用来在无监督的环境下度量不确定性，而基于信息增益的意义度量经常用来在有监督的环境下选择强分类特征。在上面的形式化定义中，我们把每个图模式 p 看成了一个随机变量 v_p，如果数据库中的某个图 G 含有 p，则在 G 上 v_p 等于 1，否则 v_p 等于 0。

通常，当某个具体的意义度量给定时，等式 (4.1) 定义的问题是 NP-hard 问题。下面，我们证明 MIGS 问题 (定义 4.4.2) 是 NP-hard。类似地，也可以证明 MES 问题

(定义 4.4.1) 是 NP-hard。

定理 4.4.1 最大化基于信息增益的意义度量问题 (MIGS) 是 NP-hard 问题。

证明 通过把最大覆盖问题规约到 MIGS 问题，我们来证明该定理。

设 $MC = (E, S, K)$ 是最大覆盖问题的任一实例，其中 $E = \{e_1, e_2, \ldots, e_N\}$ 是元素集合，$S = \{S_1, S_2, \ldots, S_L\}$ 是 E 中元素的集合族。求解 MC 要求从 S 中选择 K 个集合，使得这 K 个集合覆盖的元素数量最大化。我们可以在多项式时间内把 MC 转化成 MIGS 问题的一个实例。

1. 把 E 中的每个元素看成一个边的标号，那么 S 中的某个集合 $S_i = \{e_{i1}, e_{i2}, \ldots, e_{it}\}$ 可以转换成图 2 所示的图模式，该图模式用 $P(S_i)$ 表示。$P(S_i)$ 中所有结点的标号都相同，但是边的标号各不相同。以相同的方式，我们可以把 S 中的所有集合都转换成对应的图模式。所有这些图模式构成的集合用 PS 表示。

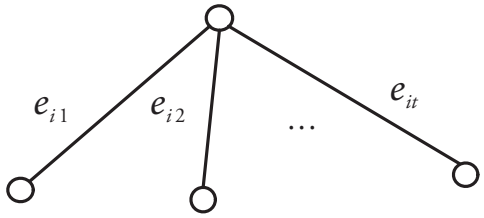


图 2 对应 $S_i = \{e_{i1}, e_{i2}, \ldots, e_{it}\}$ 的图模式 $P(S_i)$

2. 我们为 E 中的每个元素构造一个图。如果一个元素 e 能被 $\{S_{i1}, S_{i2}, \ldots, S_{im}\}$ 中的集合覆盖，我们为 e 构造一个图，如图 3 所示，其中 $P(S_i)$ 对应集合 S_i 的图模式，连接各个图模式 $P(S_i)$ 的边标号都相同。显然，每个元素都对应一个图，我们把所有这样的图标记为正类，放入数据库 DB，此时，$|DB| = n$。

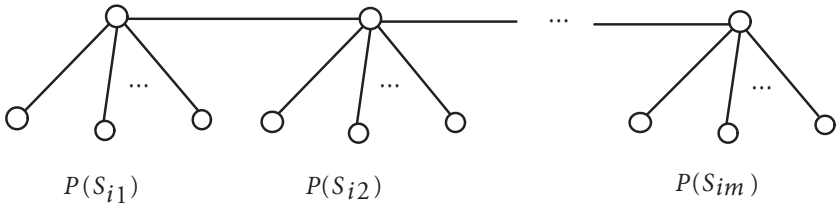


图 3 对应 $\{S_{i1}, S_{i2}, \ldots, S_{im}\}$ 的数据库图

3. 我们随意构造 n 个图，使得这 n 个图中不含有 PS 中的任何图模式。我们把这 n 个图标记为负类，放入数据库 DB，此时，$|DB| = 2n$。

给定上面构造的图数据库 DB,DB 中的一个图模式集合 PS 以及 $MC = (E, S, K)$ 中的正整数 K，我们有 MIGS 问题的一个实例 IMIG。

因为 DB 中正类图个数等于负类图个数，因此 $H(C) = 1/2$，其中 C 表示 DB 中类标记的随机变量。假设 T 是 PS 中任意一个大小为 K 的子集合。因为 T 中每个图

模式都对应 S 中的一个集合，因此由 T 可得 S 的一个大小为 K 的子集合 S'。假设被 S' 覆盖的元素数是 x，对 T 中的任何图模式 p，DB 中的任何负类图都不含有 p。根据条件熵定义，容易证明

$$\begin{aligned} H(C|T) &= -\frac{(n-x)+n}{2n}\cdot\left(\frac{n-x}{(n-x)+n}\log\frac{n-x}{(n-x)+n}+\frac{n}{(n-x)+n}\log\frac{n}{(n-x)+n}\right) \\ &= -\left(\frac{n-x}{2n}\log\frac{n-x}{2n-x}+\frac{1}{2}\log\frac{n}{2n-x}\right) \end{aligned}$$

因此

$$IG(T) = H(C) - H(C|T) = \frac{1}{2}+\frac{n-x}{2n}\log\frac{n-x}{2n-x}+\frac{1}{2}\log\frac{n}{2n-x}$$

设 $f(x)=\frac{n-x}{2n}\log\frac{n-x}{2n-x}+\frac{1}{2}\log\frac{n}{2n-x}$，可以证明 $f(x)$ 的一阶导数 $f'(x)$ 大于 0，$f(x)$ 是一个单调递增函数。当 x 的值增大时，$IG(T)$ 也增大。因此，IMIG 的最优解能被用来导出 MC 的一个最优解。 □

4.5　挖掘 Top-K 图模式

MES 和 MIGS 都是 NP-hard 问题，因此需要设计近似算法或启发式算法求解它们。本章提出了两个挖掘 Top-K 图模式的高效算法，即 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 从频繁图模式集合中贪心选择 Top-K 模式，具有近似比保证。Cluster-TopK 从图数据库中挖掘代表图模式集合，然后再从中贪心选择 Top-K 模式。尽管 Cluster-TopK 没有近似比保证，但 Cluster-TopK 却具有极高的挖掘效率。实验结果显示，Cluster-TopK 比 Greedy-TopK 要快 1 到 2 个数量级。而且，Cluster-TopK 的挖掘结果质量非常类似于 Greedy-TopK 的挖掘结果质量。

4.5.1　Greedy-TopK 算法

Greedy-TopK 算法采用著名的贪心策略，从所有频繁图模式集合中增量地选择 K 个图模式。为了应用贪心策略，我们定义了一个收益函数 b。Greedy-TopK 每次选择使收益函数 b 最大化的图模式。假定 T 是已经选择的图模式集合，对 MES 和 MIGS 问题，图模式 p 的收益函数 b 定义如下

$$b(p) = \begin{cases} H(T,p) - H(T) & \textbf{(MES)} \\ H(C|T) - H(C|T,p) & \textbf{(MIGS)} \end{cases} \tag{4.2}$$

根据式 (4.2) 给出的贪心规则，我们设计了一个贪心算法 Greedy-TopK，如算法 8 所示。初始化时，使用某个图挖掘算法 (例如:gSpan [145]) 挖掘所有频繁模式集合 F，

同时置结果集 T 为空。然后算法选择最有意义的图模式 p^* 放入 T 中。此后，算法进入一个循环过程，每次从剩余的图模式集合 $F-T$ 中选择一个使收益函数 b 最大化的图模式 p，将其放入 T 中。当结果集 T 中图模式个数等于 K 时，退出循环，算法结束。

Algorithm 8: Greedy-TopK 算法

Input: 图数据库 D，最小支持度 min_sup，意义度量 M，输出图模式个数 K
Output: 使 M 最大化的 K 个图模式

```
挖掘 D 中所有频繁图模式集合 F (w.r.t min_sup);
从 F 中选择一个图模式 p* 使 M(p*) 最大化;
T = {p*};
while (|T| < K) do
    从 F − T 中选择图模式 p 使得 b(p) 最大化;
    T = T ∪ {p};
输出 T;
```

下面，我们分析算法 Greedy-TopK 的近似比。当意义度量 M 满足 Submodular 性质 (见下面的定义) 时，贪心算法将给出近似解 [88]。

定义 4.5.1(Submodular **性质**) 设 M 是定义在一个集合 S 上的度量函数。如果对任何 $T \subset T' \subseteq S$ 和任何 $p \in S$，都有 $M(T \cup \{p\}) - M(T) \geqslant M(T' \cup \{p\}) - M(T')$，则称 M 是一个 Submodular 的集合函数。

仔细分析本章提出的两个意义度量，我们发现基于熵的意义度量满足 Submodular 性质，因此算法 Greedy-TopK 对 MES 问题能给出如下的近似比。

定理 4.5.1 设 T 是算法 Greedy-TopK 选择的 K 个图模式集合，T^* 是使联合熵最大化的 K 个图模式集合，那么，$\frac{H(T^*)}{H(T)} \leqslant \frac{e}{e-1}$。

证明 文献 [88] 给出了完整的证明过程。 □

我们发现基于信息增益的意义度量并不满足 Submodular 性质。因此，对 MIGS 问题，算法 Greedy-TopK 不能给出像定理 4.5.1 那样的 (脱机) 近似比。然而，我们注意到，对 MIGS 问题，无条件熵 $H(C)$ 是任何模式集合信息增益的上界。因此，算法 Greedy-TopK 对 MIGS 问题能给出如下的联机近似比。

定理 4.5.2 设 T 是算法 Greedy-TopK 选择的 K 个图模式集合，那么，算法 Greedy-TopK 对 MIGS 问题给出的联机近似比至多是 $H(C)/IG(T)$。

证明 设 $IG(T^*)$ 是 MIGS 问题的最优解，则 $IG(T) \leqslant IG(T^*)$。根据 MIGS 问题定义，$IG(T^*) = H(C) - H(C|T^*) \leqslant H(C)$，我们有 $IG(T) \leqslant H(C)$。因此，算法 Greedy-TopK 对 MIGS 问题给出的联机近似比至多是 $H(C)/IG(T)$。 □

下面，我们研究如何根据给定的意义度量设计有效的裁剪技术，并将其集成到图模式挖掘算法的框架中裁剪图模式搜索空间。

我们知道，所有频繁图模式挖掘算法都利用了支持度的反单调性质裁剪搜索空间，即一个图模式 p 的支持度是 p 的所有超图支持度的上界。然而，通常情况下，给定的意义度量并不具有反单调性质。因此，我们不能像利用支持度那样简单地利用给定的意义度量。

幸运的是，给定一个图模式 p 的意义度量值 $M(p)$，我们可以导出 p 的所有超图意义度量值的上界，从而根据这个上界来裁剪图模式搜索空间。

我们仍然采用 Greedy-TopK 算法的框架来增量地选择 Top-K 图模式。因此，我们分两种情况介绍裁剪技术: (1) 当结果集为空时，选择第一个图模式；(2) 当某些图模式已经被选择放入结果集时，选择下一个图模式。

4.5.1.1　选择第一个图模式的裁剪技术

本小节研究结果集为空时如何设计有效的裁剪技术。请注意，我们在本小节和 4.5.1.2 节给出的裁剪技术可以应用到任何频繁子图挖掘算法的框架中。因此，我们未给出具体的挖掘算法。定理 4.5.3 给出了基于熵的意义度量的裁剪条件。定理 4.5.4 给出了基于信息增益的意义度量的裁剪条件。

定理 4.5.3　设 D 是一个给定的图数据库，q 是 p 的任意超图。如果 $\text{support}(p;D) \leqslant 1/2$，则 $H(q) \leqslant H(p)$。

证明　根据文献 [41] 可知，$H(x)$ 是凹函数，当 $x = 1/2$ 时，$H(x)$ 取最大值，当 $x < 1/2$ 时，$H(x)$ 随着 x 的增加而单调递增。因为 q 是 p 的任意超图，根据反单调性质，$\text{support}(q;D) \leqslant \text{support}(p;D)$。又因为 $\text{support}(p;D) \leqslant 1/2$，所以 $\text{support}(q;D) \leqslant \text{support}(p;D) \leqslant 1/2$。因此，$H(q) \leqslant H(p)$。　□

定理 4.5.4　给定图数据库 $D = PD + ND$，其中 PD 和 ND 分别是正类和反类图数据集合。设 q 是 p 的任意超图，D 中有 n 个图含有 p，PD 中有 m 个图含有 p，那么

$$IG(q) \leqslant \max \begin{cases} H(\frac{|PD|}{|D|}) - \frac{|D|-m}{|D|} H(\frac{|PD|-m}{|D|-m}) \\ H(\frac{|PD|}{|D|}) - \frac{|D|-(n-m)}{|D|} H(\frac{|PD|}{|D|-(n-m)}) \\ IG(p) \end{cases} \tag{4.3}$$

证明　设 D 中有 x 个图含有 q，PD 中有 y 个图含有 q。根据信息增益的定义，$IG(q) = H(\frac{|PD|}{|D|}) - \frac{x}{|D|} H(\frac{y}{x}) - \frac{|D|-x}{|D|} H(\frac{|PD|-y}{|D|-x})$。显然，$IG(q)$ 是 x 和 y 的函数。设 $f(x,y) = H(\frac{|PD|}{|D|}) - \frac{x}{|D|} H(\frac{y}{x}) - \frac{|D|-x}{|D|} H(\frac{|PD|-y}{|D|-x})$。根据文献 [101] 可知，$f(x,y)$ 是一个凸函数。现在分析 x 和 y 的取值范围。因为 q 是 p 的超图，根据反单调性质，可得 $y \leqslant x$，$y \leqslant m$，$x \leqslant n$，$x - y \leqslant n - m$。现在我们把每一对 (x,y) 看作平面上的一个点，所有满足取值范围的点对 (x,y) 将构成平面上的一个四边形，四边形的四个顶

点分别是 $(0,0)$，(m,m)，(n,m)，$(n-m,0)$。根据文献 [62] 可知，凸函数将在凸多边形的顶点上取得最大值。因此，函数 $f(x,y)$ 将在 $(0,0)$，(m,m)，(n,m)，$(n-m,0)$ 这几个点上取得最大值。显然，在点 $(0,0)$ 上，$f(x,y) = H(\frac{|PD|}{|D|})$ 不可能取得最大值，因此，$f(x,y)$ 只能在 (m,m)，$(n-m,0)$，(n,m) 这几个点上取得最大值。如果 $(x,y)=(m,m)$，那么 $f(x,y) = H(\frac{|PD|}{|D|}) - \frac{|D|-m}{|D|}H(\frac{|PD|-m}{|D|-m})$。如果 $(x,y)=(n-m,0)$，那么 $f(x,y) = H(\frac{|PD|}{|D|}) - \frac{|D|-(n-m)}{|D|}H(\frac{|PD|}{|D|-(n-m)})$。如果 $(x,y)=(n,m)$，那么 $f(x,y) = H(\frac{|PD|}{|D|}) - \frac{n}{|D|}H(\frac{m}{n}) - \frac{|D|-n}{|D|}H(\frac{|PD|-m}{|D|-n}) = IG(p)$。定理证毕。 □

4.5.1.2 选择下一个图模式的裁剪技术

本小节研究结果集不空 (已经有图模式被选择) 时如何设计有效的裁剪技术。类似 4.5.1.1 节的定理，我们也可以针对结果集为非空的情况导出一个图模式 p 的所有超图意义度量值的上界。在给出具体的裁剪技术之前，我们先定义一个新的概念——等价类。

定义 4.5.2(等价类) 设 D 是一个给定的图数据库，G 是 D 中的一个图，T 是已经选择的图模式集合。G 相对于 T 的等价类被定义为 $\{G'|G' \in D, \forall p \in T, I(p \subseteq G) = I(p \subseteq G')\}$，其中 $I(\cdot)$ 是指示函数。

根据等价类的定义，一个图模式集合 T 可以将数据库 D 划分成若干个等价类 (块) 集合 $D_T=\{B_i|1 \leqslant i \leqslant l\}$，并且 $D = \bigcup\limits_{1 \leqslant i \leqslant l} B_i$。利用等价类的定义，我们可以在结果集非空的情况下导出有效的裁剪条件。定理 4.5.5 给出了基于熵的意义度量的裁剪条件。定理 4.5.6 给出了基于信息增益的意义度量的裁剪条件。

定理 4.5.5 设 D 是一个给定图数据库，T 是已经选择的图模式集合，$D_T=\{B_i|1 \leqslant i \leqslant l\}$ 是用 T 划分 D 得到的等价类集合，q 是 p 的任意超图 $(p \subseteq q)$，B_i 中有 n_i 个图含有 p，并且 $p \notin T$，$q \notin T$，那么

$$H(T \cup \{q\}) \leqslant \sum_{i=1}^{l} \begin{cases} -\frac{|B_i|}{|D|}\log\frac{|B_i|}{2|D|} & (n_i \geqslant \frac{1}{2}|B_i|) \\ -(\frac{n_i}{|D|}\log\frac{n_i}{|D|} + \frac{|B_i|-n_i}{|D|}\log\frac{|B_i|-n_i}{|D|}) & (n_i < \frac{1}{2}|B_i|) \end{cases} \tag{4.4}$$

证明 我们假设 B_i 中有 x_i 个图含有 q。根据联合熵的定义，有 $H(T \cup \{q\}) = \sum\limits_{i=1}^{l}(-(\frac{x_i}{|D|}\log\frac{x_i}{|D|} + \frac{|B_i|-x_i}{|D|}\log\frac{|B_i|-x_i}{|D|}))$。设函数 $f(x_i) = -(\frac{x_i}{|D|}\log\frac{x_i}{|D|} + \frac{|B_i|-x_i}{|D|}\log\frac{|B_i|-x_i}{|D|})$，容易证明 $f(x_i)$ 是一个凹函数，当 $x_i = \frac{1}{2}|B_i|$ 时，$f(x_i)$ 取得最大值；当 $0 \leqslant x_i < \frac{1}{2}|B_i|$ 时，$f(x_i)$ 单调递增；当 $\frac{1}{2}|B_i| \leqslant x_i \leqslant |B_i|$ 时，$f(x_i)$ 单调递减。因为 q 是 p 的超图，根据反单调性质，我们有 $x_i \leqslant n_i$。因此，当 $n_i \geqslant \frac{1}{2}|B_i|$ 时，$f(x_i)$ 可能取得的最大值为 $-(\frac{\frac{1}{2}|B_i|}{|D|}\log\frac{\frac{1}{2}|B_i|}{|D|} + \frac{B_i-\frac{1}{2}B_i}{|D|}\log\frac{B_i-\frac{1}{2}B_i}{|D|}) = -\frac{|B_i|}{|D|}\log\frac{|B_i|}{2|D|}$；当 $n_i < \frac{1}{2}|B_i|$ 时，$f(x_i)$ 可能取得的最大值为 $-(\frac{n_i}{|D|}\log\frac{n_i}{|D|} + \frac{|B_i|-n_i}{|D|}\log\frac{|B_i|-n_i}{|D|})$。在每个等价类 B_i 上都取函数 $f(x_i)$

的最大值，可得 $H(T\cup\{q\})$ 的最大值。定理证毕。 □

定理 4.5.6　给定图数据库 $D=PD+ND$，其中 PD 和 ND 分别是正类和反类图数据集合。设 T 是已经选择的图模式集合，D_T=$\{B_i|1\leqslant i\leqslant l\}$ 是用 T 划分 D 得到的等价类集合，q 是 p 的任意超图 ($p\subseteq q$)，B_i 中有 $|PB_i|$ 个正类图，B_i 中有 n_i 个图含有 p，B_i 中有 m_i 个正类图含有 p，并且 $p\notin T$，$q\notin T$，那么

$$IG(T\cup\{q\})\leqslant H(\frac{|PD|}{|D|})+\sum_{i=1}^{l}\max\begin{cases}-\frac{|B_i|-m_i}{|D|}H(\frac{|PB_i|-m_i}{|B_i|-m_i})\\-\frac{|B_i|-(n_i-m_i)}{|D|}H(\frac{|PB_i|}{|B_i|-(n_i-m_i)})\\-(\frac{n_i}{|D|}H(\frac{m_i}{n_i})+\frac{|B_i|-n_i}{|D|}H(\frac{|PB_i|-m_i}{|B_i|-n_i}))\end{cases}\tag{4.5}$$

证明　假设 B_i 中有 x_i 个图含有 q，B_i 中有 y_i 个正类图含有 q。根据信息增益的定义，$IG(T\cup\{q\})=H(\frac{|PD|}{|D|})-\sum_{i=1}^{l}(\frac{x_i}{|D|}H(\frac{y_i}{x_i})+\frac{|B_i|-x_i}{|D|}H(\frac{|PB_i|-y_i}{|B_i|-x_i}))$。定义函数 $f(x_i,y_i)=-(\frac{x_i}{|D|}H(\frac{y_i}{x_i})+\frac{|B_i|-x_i}{|D|}H(\frac{|PB_i|-y_i}{|B_i|-x_i}))$。类似于文献 [101] 给出的证明过程，容易证明 $f(x_i,y_i)$ 是一个凸函数。现在分析 x_i 和 y_i 的取值范围，因为 q 是 p 的超图，根据反单调性质，可得 $y_i\leqslant x_i$，$y_i\leqslant m_i$，$x_i\leqslant n_i$，$x_i-y_i\leqslant n_i-m_i$。我们把每一对 (x_i,y_i) 看作平面上的一个点，所有满足取值范围的点对 (x_i,y_i) 将构成平面上的一个四边形，四边形的四个顶点分别是 $(0,0)$，(m_i,m_i)，(n_i,m_i)，$(n_i-m_i,0)$。根据文献 [62] 可知，凸函数将在凸多边形的顶点上取得最大值。因此，函数 $f(x_i,y_i)$ 将在 $(0,0)$，(m_i,m_i)，(n_i,m_i)，$(n_i-m_i,0)$ 这几个点上取得最大值。容易证明，在点 $(0,0)$ 上，$f(0,0)=-\frac{|B_i|}{|D|}H(\frac{|PB_i|}{|B_i|})$ 不可能取得最大值，因此，$f(x_i,y_i)$ 只能在 (m_i,m_i)，$(n_i-m_i,0)$，(n_i,m_i) 这几个点上取最大值。如果 $(x_i,y_i)=(m_i,m_i)$，那么 $f(x_i,y_i)=-\frac{|B_i|-m_i}{|D|}H(\frac{|PB_i|-m_i}{|B_i|-m_i})$。如果 $(x_i,y_i)=(n_i-m_i,0)$，那么 $f(x_i,y_i)=-\frac{|B_i|-(n_i-m_i)}{|D|}H(\frac{|PB_i|}{|B_i|-(n_i-m_i)})$。如果 $(x_i,y_i)=(n_i,m_i)$，那么 $f(x_i,y_i)=-(\frac{n_i}{|D|}H(\frac{m_i}{n_i})+\frac{|B_i|-n_i}{|D|}H(\frac{|PB_i|-m_i}{|B_i|-n_i}))$。在每个等价类 B_i 上都取函数 $f(x_i,y_i)$ 的最大值，可得 $IG(T\cup\{q\})$ 的最大值。定理证毕。 □

4.5.2　Cluster-TopK 算法

因为 Greedy-TopK 算法需要先产生所有的频繁图模式，所以在频繁图模式数量较少的情况下，Greedy-TopK 算法具有较高的效率。然而，在实际应用中，当数据库中图较稠密或者用户给定的支持度阀值较低时，频繁图模式挖掘算法通常产生大量的甚至指数级的频繁子图，这使得 Greedy-TopK 算法不能在合理的时间内完成任务，限制了该算法的可用性。即使使用 4.5.1.1 节和 4.5.1.2 节中的裁剪技术，Greedy-TopK 也需要遍历图模式空间中的大量图模式。因此，本小节给出了另一个更高效的算法 Cluster-TopK 来挖掘 Top-K 图模式。实验结果显示，Cluster-TopK 算法比 Greedy-TopK 算

法快 1 到 2 个数量级，而且 Cluster-TopK 算法的挖掘结果质量与 Greedy-TopK 算法非常接近。

Cluster-TopK 算法的基本思想是将所有频繁图模式聚类成若干个簇，选择每个簇的中心作为代表模式构成一个候选集合 T，然后再使用贪心策略从 T 中增量地选择 K 个图模式。请注意，Cluster-TopK 算法并不是先产生所有的频繁图模式，而是从图数据库中快速地挖掘那些代表模式 (簇中心) 得到候选集，具体内容见 4.5.2.2 节。

假定 $C = \{p_1, p_2, \ldots, p_n\}$ 是由若干个图模式构成的一个簇，p_i 是该簇中心。因为 p_i 作为代表模式被选择，p_i 应具有这样的特点:$\forall p_j \in C$，$p_j \subseteq p_i$，并且 p_j 和 p_i 在数据库图中经常一起出现，即 p_j 和 p_i 有类似的支持度。下面，我们给出这种类型簇的两个形式化定义，δ-簇和 Δ-簇。现用 support(p) 表示图模式 p 的绝对支持度。

定义 4.5.3(δ-**覆盖**) 设 $\delta(0 \leqslant \delta \leqslant 1)$ 是用户给定的一个参数，p 和 q 是任意两个图模式。如果满足 $q \subseteq p$，$1 - \frac{\text{support}(p)}{\text{support}(q)} \leqslant \delta$，则称 q 被 p δ-覆盖。

定义 4.5.4(δ-**簇**) 设 $\delta(0 \leqslant \delta \leqslant 1)$ 是用户给定的一个参数，$C = \{p_1, p_2, \ldots, p_n\}$ 是一个图模式集合。如果 C 中存在图模式 p_i 满足 $\forall p_j \in C$，p_j 都被 p_i δ-覆盖，则称 C 是一个 δ-簇，称 p_i 为该 δ-簇的代表模式 (中心点)。

定义 4.5.5(Δ-**覆盖**) 设 $\Delta(\Delta$ 是大于 0 的整数) 是用户给定的一个参数，p 和 q 是任意两个图模式。如果满足 $q \subseteq p$，$\text{support}(q) - \text{support}(p) \leqslant \Delta$，则称 q 被 p Δ-覆盖。

定义 4.5.6(Δ-**簇**) 设 $\Delta(\Delta$ 是大于 0 的整数) 是用户给定的一个参数，$C = \{p_1, p_2, \ldots, p_n\}$ 是一个图模式集合。如果 C 中存在图模式 p_i 满足 $\forall p_j \in C$，p_j 都被 p_i Δ-覆盖，则称 C 是一个 Δ-簇，称 p_i 为该 Δ-簇的代表模式 (中心点)。

Cluster-TopK 算法就是根据用户给定的参数 δ(或 Δ)，将频繁图模式集合划分成若干个 δ-簇 (或 Δ-簇)，选择每个 δ-簇 (或 Δ-簇) 的代表模式构成候选集，然后从候选集中贪心地选择 K 个图模式。4.5.2.1 节分析了 Cluster-TopK 算法采用这种聚类策略的优点。4.5.2.2 节研究了如何在不产生所有频繁模式的情况下快速地挖掘每个 δ-簇 (或 Δ-簇) 的代表模式。4.5.2.3 给出了具体的算法实现。

4.5.2.1 Cluster-TopK 聚类策略的优点

本小节讨论 Cluster-TopK 算法为什么采用这种先聚类后贪心的策略。我们先给出几个引理，后面定理的证明中将利用它们。设 N 是给定图数据库 D 的大小，即 $|D| = N$。定义函数 $f(n) = \frac{n}{N} \log \frac{N}{n} (0 \leqslant n \leqslant N)$。

引理 4.5.1 当 $1 \leqslant m \leqslant n \leqslant N$ 时，$-f(N-m) \leqslant f(n) - f(n-m) \leqslant f(m)$。

证明 见文献 [166] 中的 PROPOSITION 6.1。 □

引理 4.5.2　当 $1 \leqslant n \leqslant N$ 时，$f(n) + f(N-n) \leqslant n(f(1) + f(N-1))$。

证明　见文献 [166] 中的 PROPOSITION 6.2。 □

利用引理 4.5.1 和引理 4.5.2 的结果，我们可以证明下面的引理 4.5.3。

引理 4.5.3　设 D 是一个给定的图数据库，$|D| = N$，p_1 和 p_2 是两个图模式，d 是 p_1 和 p_2 之间的**海明距离**，即 $d = \sum_{G_i \in D} |I(p_1 \subseteq G_i) - I(p_2 \subseteq G_i)|$，其中 I 是指示函数，如果 $p_1 \subseteq G_i$，$I(p_1 \subseteq G_i)$=1，否则 $I(p_1 \subseteq G_i)$=0，那么，$0 \leqslant H(p_1, p_2) - H(p_1) \leqslant d(f(1) + f(N-1))$。

证明　设 $P(p_1 = 1)$ 表示数据库 D 中含有 p_1 的概率 (即 p_1 在 D 中的相对支持度)，$P(p_1 = 0)$ 表示数据库 D 中不含有 p_1 的概率，则 $P(p_1 = 1) + P(p_1 = 0)$=1。设 $P(p_1 = 1, p_2 = 1)$ 表示数据库 D 中既含有 p_1 又含有 p_2 的概率，即 $P(p_1 = 1, p_2 = 1) = \frac{\sum_{G_i \in D} I(p_1 \subseteq G_i, p_2 \subseteq G_i)}{|D|}$。$P(p_1 = 1, p_2 = 0)$，$P(p_1 = 0, p_2 = 1)$ 和 $P(p_1 = 0, p_2 = 0)$ 可以类似定义。

设 $A = -P(p_1 = 0)\log P(p_1 = 0)$，$B = -P(p_1 = 1)\log P(p_1 = 1)$，$W = -P(p_1 = 0, p_2 = 0)\log P(p_1 = 0, p_2 = 0)$，$X = -P(p_1 = 1, p_2 = 1)\log P(p_1 = 1, p_2 = 1)$，$Y = -P(p_1 = 1, p_2 = 0)\log P(p_1 = 1, p_2 = 0)$，$Z = -P(p_1 = 0, p_2 = 1)\log P(p_1 = 0, p_2 = 1)$。根据熵的定义，可得 $H(p_1, p_2) = W + X + Y + Z$, $H(p_1) = A + B$。

因为 $|D| = N$，我们假设 $P(p_1 = 0) = t/N$，$P(p_1 = 0, p_2 = 1) = s/N$，则 $P(p_1 = 0, p_2 = 0) = P(p_1 = 0) - P(p_1 = 0, p_2 = 1) = (t-s)/N$。因此，$W + Z - A = f(t-s) + f(s) - f(t) = f(s) - (f(t) - f(t-s))$。根据引理 4.5.1，可得 $0 \leqslant W + Z - A \leqslant f(s) + f(N-s)$。再根据引理 4.5.2，可得 $0 \leqslant W + Z - A \leqslant s(f(1) + f(N-1))$。类似地，假设 $P(p_1 = 1) = u/N$，$P(p_1 = 1, p_2 = 0) = v/N$，则 $P(p_1 = 1, p_2 = 1) = (u-v)/N$。再根据引理 4.5.1 和引理 4.5.2，可得 $0 \leqslant X + Y - B \leqslant f(v) + f(N-v) \leqslant v(f(1) + f(N-1))$。

$H(p_1, p_2) - H(p_1) = (W + X + Y + Z) - (A + B) = (W + Z - A) + (X + Y - B)$。因此，$0 \leqslant H(p_1, p_2) - H(P_1) \leqslant (s + v)(f(1) + f(N-1))$。又因为 p_1 和 p_2 之间的海明距离 $d = s + v$，可得 $0 \leqslant H(p_1, p_2) - H(p_1) \leqslant d(f(1) + f(N-1))$。 □

根据引理 4.5.3，我们容易得到下面的引理 4.5.4.

引理 4.5.4　设 D 是一个给定的图数据库，$|D| = N$，$F = \{p_1, p_2, \ldots, p_n\}$ 是任意一个图模式集合，p_{n+1} 是任意一个图模式，如果 p_n 和 p_{n+1} 之间的海明距离是 d，则 $H(p_1, p_2, \ldots, p_n) - H(p_1, \ldots, p_{n-1}, p_{n+1}) \leqslant d(f(1) + f(N-1))$。

证明

$$\begin{aligned} H(p_1, p_2, \ldots, p_n) - H(p_1, \ldots, p_{n-1}, p_{n+1}) &\leqslant H(p_1, \ldots, p_n, p_{n+1}) - H(p_1, \ldots, p_{n-1}, p_{n+1}) \\ &= H(p_n | p_1, \ldots, p_{n-1}, p_{n+1}) \leqslant H(p_n | p_{n+1}) \end{aligned}$$

$$= H(p_n, p_{n+1}) - H(p_{n+1})$$

因为 p_n 和 p_{n+1} 之间的海明距离是 d，再根据引理 4.5.3，可得 $H(p_1, p_2, \ldots, p_n) - H(p_1, \ldots, p_{n-1}, p_{n+1}) \leqslant H(p_n, p_{n+1}) - H(p_{n+1}) \leqslant d(f(1) + f(N-1))$。 □

因为频繁图模式的数量通常很大，直接对频繁图模式进行贪心选择，时间复杂性太高。对频繁图模式先聚类，只提取每个类的代表模式 (中心点) 构成候选集合，可以将频繁模式的数量降低 2 到 3 个数量级。然后对候选集合进行贪心选择，可使算法具有极高的效率。而且，我们可以证明代表模式集合中的最优解和频繁图模式集合中的最优解差别很小。定理 4.5.7 针对 MES 问题，分析了最优解之间的差异。定理 4.5.8 针对 MIGS 问题，分析了最优解之间的差异。定理 4.5.7 和定理 4.5.8 给出了利用 Δ-簇的分析结果。类似地，我们也很容易给出利用 δ-簇的分析结果。

定理 4.5.7 设 D 是一个给定图数据库，$|D| = N$，$F = \{p_1, p_2, \ldots, p_n\}$ 是 D 中频繁图模式集合，F 被划分成 m 个 Δ-簇的集合 $CS = \{C_1, C_2, \ldots, C_m\}$，其中每个 C_i $(1 \leqslant i \leqslant m)$ 都是一个 Δ-簇，并且 $F = C_1 \cup C_2 \cup \ldots \cup C_m$。$CS$ 中每个 Δ-簇的代表模式被选择构成了一个集合 $RS = \{r_1, r_2, \ldots, r_m\}$，其中 r_i $(1 \leqslant i \leqslant m)$ 是 C_i 的代表模式。再假设 T^* 是 F 中的一个大小为 K 的子集合，并且使 $H(T^*)$ 最大化。S^* 是 RS 的一个大小为 K 的子集合，并且使 $H(S^*)$ 最大化。那么，$H(T^*) - H(S^*) \leqslant \Delta K(f(1) + f(N-1))$。

证明 设 $T^* = \{q_1, q_2, \ldots, q_K\}$。因为 F 被划分成 m 个 Δ-簇的集合 CS，$RS = \{r_1, r_2, \ldots, r_m\}$ 是 CS 中 Δ-簇的代表模式集合，并且 $T^* \subset F$，根据 Δ-簇的定义，对任意 $q_i \in T^*$，都存在 RS 中的一个代表模式 r，使得 r 能 Δ-覆盖 q_i。不失一般性，假定 q_1 被 $r_1\Delta$-覆盖。根据 Δ-覆盖的定义，可知 $q_1 \subseteq r_1$，$\text{support}(q_1) - \text{support}(r_1) \leqslant \Delta$。因此，$q_1$ 与 r_1 之间的海明距离 $d = \text{support}(q_1) - \text{support}(r_1) \leqslant \Delta$，我们用 r_1 替换 q_1，令 $T = T^* - \{q_1\} \cup \{r_1\}$。根据引理 4.5.4，可得 $H(T^*) - H(T) \leqslant d(f(1) + f(N-1)) \leqslant \Delta(f(1) + f(N-1))$。以此类推，如果 T^* 中每个图模式都被 RS 中对应的代表模式所替换，可得到一个新的集合 T'，并且 $H(T^*) - H(T') \leqslant \Delta K(f(1) + f(N-1))$。显然，$T'$ 是 RS 的一个大小为 K 的子集合。因为 S^* 是 RS 的一个大小为 K 的子集合，并且使 $H(S^*)$ 最大化，我们有 $H(T') \leqslant H(S^*)$。因此，$H(T^*) - H(S^*) \leqslant H(T^*) - H(T') \leqslant \Delta K(f(1) + f(N-1))$。 □

定理 4.5.8 设 D 是一个给定的图数据库，$|D| = N$，$F = \{p_1, p_2, \ldots, p_n\}$ 是 D 中频繁图模式集合，F 被划分成 m 个 Δ-簇的集合 $CS = \{C_1, C_2, \ldots, C_m\}$，其中每个 C_i $(1 \leqslant i \leqslant m)$ 都是一个 Δ-簇，并且 $F = C_1 \cup C_2 \cup \ldots \cup C_m$。$CS$ 中每个 Δ-簇的代表模式被选择构成了一个集合 $RS = \{r_1, r_2, \ldots, r_m\}$，其中 r_i $(1 \leqslant i \leqslant m)$ 是 C_i 的代表模式。再假设 T^* 是 F 中的一个大小为 K 的子集合，并且使 $IG(T^*)$ 最大化。S^* 是 RS 的一个大小为 K 的子集合，并且使 $IG(S^*)$ 最大化。那么，$IG(T^*) - IG(S^*) \leqslant$

$2\Delta K(f(1)+f(N-1))$。

证明　假设 C 是表示数据库 D 中图类别的随机变量。类似于定理 4.5.7 中的证明过程，可得 $|H(T^*)-H(S^*)| \leqslant \Delta K(f(1)+f(N-1))$，$|H(C,T^*)-H(C,S^*)| \leqslant \Delta K(f(1)+f(N-1))$。因为 $IG(T^*)-IG(S^*)=(H(C)-H(C|T^*))-(H(C)-H(C|S^*))=H(C|S^*)-H(C|T^*)=(H(C,S^*)-H(S^*))-(H(C,T^*)-H(T^*))=(H(T^*)-H(S^*))-(H(C,T^*)-H(C,S^*))$，所以 $IG(T^*)-IG(S^*) \leqslant 2\Delta K(f(1)+f(N-1))$。　□

以 MES 问题为例，我们分析具体的差异。假设定理 4.5.7 中的 $N=10\,000, K=10, \Delta=5$，则有 $H(T^*)-H(S^*) \leqslant \Delta K(f(1)+f(N-1)) \approx 0.073$。这是理论上的最大可能差异，而实验中的结果显示: 从聚类结果中选择的解和从频繁图模式集合中选择的解差别微乎其微。

4.5.2.2　Cluster-TopK 算法的关键技术

如果先得到完整的频繁图模式集合 F，再使用传统的聚类算法 (例如 k-means 算法) 对 F 进行聚类，时间复杂性将是 $O(|F|^2)$，这显然是不能接受的。为了使 Cluster-TopK 算法高效可扩展，Cluster-TopK 算法要实现如下两个目标:(1) 只扫描算法输出的图模式一遍而能得到一个代表模式 (簇中心点) 集合；(2) 裁剪图模式空间中不产生 (或极少产生) 代表模式的那些分枝。下面两个小节分别介绍实现这两个目标的关键技术。

4.5.2.2.1　产生代表模式

这里介绍产生代表模式的方法。该方法利用了第 2 章 RP-GD 算法的一些技术。图 4 显示了图模式空间中的一个分枝，每个结点代表一个频繁子图。大部分频繁子图挖掘算法 (gSpan [145]，FFSM [66] 等) 都采用深度优先遍历方式访问该空间中的每个结点。采用深度优先方式将会访问每个结点两次: 第一次是从父亲结点到当前结点的访问，例如在图 4 中从结点 A 到结点 C 的访问。第二次是在完成了所有后裔的访问后再次回到当前结点，例如在图 4 中访问完结点 E，F 和 G 之后第二次访问结点 C。目前的频繁子图挖掘算法是在第一次访问某个结点时就输出它。因此，对图 4 中的结点，输出顺序是 $\ldots, A, B, C, E, F, G, D, \ldots$。

为方便描述，我们将 δ-覆盖或 Δ-覆盖都简称为覆盖。我们的目标是产生一个代表模式集合能覆盖所有的频繁图模式。对一个给定的图模式，例如图 4 中的结点 A，根据定义，能覆盖结点 A 的图模式必然是结点 A 的超图。因此，图 4 中的结点 P，Q 和 A 的后裔都有可能覆盖结点 A。

然而，我们在第二章中证明了，如果采用 gSpan [145] 的枚举框架，在一个结点第二次访问之后，所有能覆盖该结点的图模式都已经被输出。例如: 采用 gSpan 的枚举

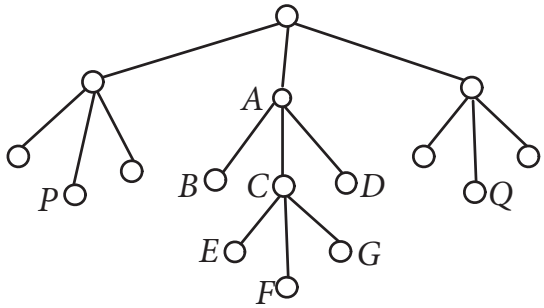


图 4　图模式空间中的一个分枝

框架，结点 Q 就不能覆盖结点 A。

因此，我们调整图模式空间中结点的输出顺序: 只有当一个结点第二次被访问时，我们才输出它。这样一来就可以保证当一个图模式 P 输出时，所有能覆盖 P 的图模式都已经被输出，便于我们为 P 选择对应的代表模式。对图 4 中的结点，调整之后的输出顺序是 $\ldots, B, E, F, G, C, D, A, \ldots$。

采用上面调整之后的输出顺序，我们依此处理每个输出的图模式。假定 RS 表示已经产生的部分代表模式集合。如果当前输出的图模式 P 能被 RS 中的某个代表模式覆盖，我们继续处理下一个输出的图模式。如果 P 不能被 RS 中的任何代表模式覆盖，我们创建一个新的能覆盖 P 的代表模式。

为了使最终产生的代表模式数量尽可能少，在创建新的能覆盖 P 的代表模式时，我们采用了贪心策略，选择 P 的后裔中能覆盖 P 的最大图模式。因为这样选择的图模式有更大的可能性来覆盖以后输出的图模式。假设需要为图 4 中的结点 A 创新一个新的代表模式并且已知结点 B，C，F，D 都能覆盖 A，根据贪心策略，我们应创建一个新的代表模式 F，因为结点 F 大于结点 B，C，D。

4.5.2.2.2　裁剪不产生代表模式的分枝

如果图模式搜索空间中的某个分枝不产生新的代表模式 (或者极少产生新的代表模式)，完全遍历该分枝会使算法的性能急剧下降。这里我们研究裁剪这些分枝的方法，该方法利用了第 2 章 RP-Leap 算法的一些技术。

图 5 显示了整个图模式空间中以图模式 p 为根的一棵子树。通过向模式 p 中增加一条新边，p 可以被扩展成一系列新的图模式 $p \diamond e_1, p \diamond e_2, \ldots, p \diamond e_n$。$p \diamond e_1$ 为根的分枝含有 $p \diamond e_1$ 的超图。$p \diamond e_2$ 为根的分枝含有 $p \diamond e_2$ 的超图，但不含有 $p \diamond e_1$ 的超图。$p \diamond e_i$ 为根的分枝含有 $p \diamond e_i$ 的超图，但不含有任何 $p \diamond e_j (j < i)$ 的超图。

对 $p \diamond e_2$ 为根的分枝中的任何图模式 $p \diamond e_2 \diamond e'$，很可能存在 $p \diamond e_2 \diamond e'$ 的超图 $p \diamond e_1 \diamond e_2 \diamond e'$，并且 $p \diamond e_1 \diamond e_2 \diamond e'$ 出现在 $p \diamond e_1$ 为根的分枝中。如果在图数据库中，p 和 $p \diamond e_1$ 经常一起出现的话，那么 $p \diamond e_2 \diamond e'$ 和 $p \diamond e_1 \diamond e_2 \diamond e'$ 有很大的可能性也经常

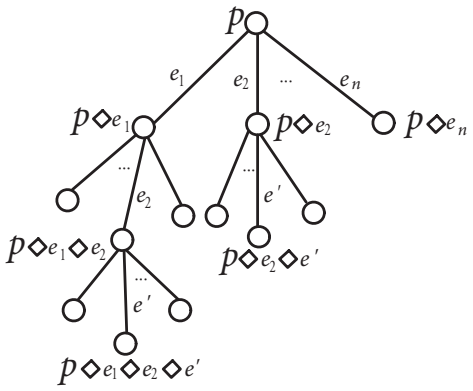


图 5　图模式空间中的以 p 为根的一棵子树

一起出现。这也就意味着，$p \diamond e_2 \diamond e'$ 和 $p \diamond e_1 \diamond e_2 \diamond e'$ 在支持度上非常接近。

如果 $p\diamond e_1$ 为根的分枝已经被遍历，那么对该分枝中的任意图模式 $p\diamond e_1\diamond e_2\diamond e'$，我们在遍历 $p\diamond e_2$ 为根的分枝之前，已经产生了一个代表模式 R 能覆盖 $p\diamond e_1\diamond e_2\diamond e'$。因为 R 能覆盖 $p\diamond e_1\diamond e_2\diamond e'$，根据覆盖的定义，$R$ 是 $p\diamond e_1\diamond e_2\diamond e'$ 的超图。如果 p 和 $p\diamond e_1$ 经常一起出现，则很可能 $\text{support}(p\diamond e_1) \approx \text{support}(p)$，并且 $\text{support}(p\diamond e_1\diamond e_2\diamond e') \approx \text{support}(p\diamond e_2\diamond e')$，那么，$\text{support}(p \diamond e_2 \diamond e') - \text{support}(R) \approx \text{support}(p \diamond e_1 \diamond e_2 \diamond e') - \text{support}(R)$。根据覆盖的定义 ($\Delta$-覆盖或者 δ-覆盖)，R 具有很大的可能性覆盖 $p \diamond e_2 \diamond e'$。因此，如果 p 和 $p \diamond e_1$ 经常一起出现，以 $p \diamond e_2$ 为根的分枝不产生 (或者很少产生) 新的代表模式，我们可以跳过该分枝的遍历，以提高算法的效率。类似地，如果存在一条边 e_i，使得 p 和 $p \diamond e_i$ 经常一起出现，我们就可以跳过那些以 $p \diamond e_j (j < i)$ 为根的分枝，因为这些分枝几乎不产生新的代表模式。

我们根据 p 和 $p \diamond e$ 的支持度定义了它们之间的一个距离函数 $\text{Dis}(p, p \diamond e) = 1 - \frac{\text{support}(p\diamond e)}{\text{support}(p)}$。用户说明一个小的距离阀值参数 $\epsilon (0 < \epsilon < 1)$，如果 $\text{Dis}(p, p \diamond e) < \epsilon$，就可以认为 p 和 $p \diamond e$ 经常一起出现。

请注意，尽管使用距离阀值参数 ϵ 可以跳过很多搜索分枝，但是我们也不可避免地会丢失一些代表模式。ϵ 值越大，丢失的数量越多。在实验中，我们发现使用一个很小的 ϵ 值 (例如 0.01)，就可以得到绝大部分的代表模式，同时使算法的性能提高 1-2 个数量级。

下面分析丢失少量代表模式对挖掘结果质量的影响非常小。假设一个代表模式 R 丢失了，R 覆盖的某个频繁图模式 p 能有另一个代表模式 R_1 所覆盖，R_1 没有丢失。因为 R 能覆盖 p，说明 R 和 p 在结构和支持度上很接近。同样，因为 R_1 能覆盖 p，说明 R_1 和 p 在结构和支持度上也很接近。因此，R 和 R_1 在结构和支持度上也会很接近。R 对意义度量的贡献能由 R_1 近似代替，因为我们的目标是选择 K 个图模式联合起来使某一意义度量最大化，丢失的少量代表模式对意义度量的贡献能由其他的代

表模式近似地代替，因此，挖掘结果质量不受什么影响。实验中，我们发现不同的 ϵ 值对挖掘结果质量影响非常小，而 ϵ 值对改进算法效率却起着巨大的作用。

4.5.2.3 Cluster-TopK 算法描述

本小节将 4.5.2.2 节中的关键技术集成到 gSpan [145] 的 DFS 编码搜索框架中，给出挖掘 Top-K 图模式的完整算法 Cluster-TopK，如算法 9 所示。在 gSpan 中，每个频繁子图都对应一个最小 DFS 编码 (边的序列)。DFS 编码搜索框架采用深度优先搜索方法挖掘频繁子图，只在最小 DFS 编码上进行最右扩展。因为 gSpan 是经典的频繁子图挖掘算法，我们省略了它的细节描述。DFS 编码、最右扩展等具体概念请见参考文献 [145]。

Algorithm 9: Cluster-TopK 算法

Input: 图数据库 D，最小支持度 $\min\_\sup$，联合意义度量 M，输出图模式个数 K，聚类质量参数 Δ(或 δ)，距离阀值 ϵ
Output: 使 M 最大化的 K 个图模式

```
1  扫描 D，得到所有频繁边；
2  删除 D 中不频繁的结点和边；
3  S^1={所有频繁边的最小 DFS 编码}；/* 根据 DFS 字典顺序，排序 S^1 中的所有 DFS 编码 */
4  GS = ∅；/* GS 是一个全局堆栈 */
5  RS = ∅；/* RS 是一个存放代表模式的全局数据结构 */
6  for S^1 里的每个 DFS 编码 s do
7      s.min_distance = 1；/* 初始化最大值 */
8      调用 MiningReprePatterns(s, NULL, D, min_sup, Δ, ϵ, RS)；
9  从 RS 中选择一个图模式 p* 使得 M(p*) 最大；
10 T = {p*}；/* T 存放挖掘结果 */
11 while (|T| < K) do
12     从 RS − T 中选择一个图模式 p 使得 b(p) 最大；/*b(·) 是一个收益函数，见等式4.2中的贪心规则 */
13     T = T ∪ {p}；
14 输出 T；
```

算法 Cluster-TopK 如下工作: 初试化时，先扫描数据库，得到频繁边集合，然后删除数据库中不频繁的边和结点。在这之后，对每个 1-边频繁子图，调用子过程 MiningReprePatterns(如算法 10 所示) 进行深度优先搜索，发现所有代表模式。在得到所有代表模式集合 RS 之后，类似于算法 Greedy-TopK，从 RS 中贪心增量地选择 K 个图模式使意义度量最大化。

下面描述子过程 MiningReprePattern 的执行流程。第 1-3 行，算法测试以当前模式 s 为根的分枝是否能被裁剪。我们使用 $p.\min\_\text{distance}$ 表示 p 和它的已经被遍历的孩子之间的最小距离 ($Dis(p,.)$)。如果 $p.\min\_\text{distance} < \epsilon$，说明存在 p 的一个已经被遍历的孩子 c，c 和 p 经常一起出现。根据 4.5.2.2.2 节描述的思想，因为当前代表模式集合 RS 能覆盖以 c 为根分枝中的所有频繁模式，RS 也就具有很大的可能性

Algorithm 10: 子过程 MiningReprePatterns

Input: DFS 编码 s，s 的父亲编码 p，图数据库 D，最小支持度 $\min\_\sup$，聚类质量参数 Δ(或 δ)，距离阀值 ϵ，代表模式集合 RS

Output: 代表模式集合 RS

```
if p! = NULL 或或 p.min_distance < ε then
    返回;
if s ≠ min(s) then
    返回;
p.min_distance = min(p.min_distance, Dis(p, s));
把 s 的最后一条边放入全局堆栈 GS;
for GS 里的每个入口 Q(图模式 Q) do
    if s 能覆盖 Q，并且 s 大于 Q 的候选代表模式 Q.R then
        用 s 代替 Q.R;
扫描 D 一次，得到 s 的所有频繁最右扩展孩子;
for s 的每个频繁最右扩展孩子 s ◇r e do
    (s ◇r e).min_distance = 1; /* 初始化最大值 */
    调用 MiningReprePatterns(s ◇r e, s, D, min_sup, Δ, ε, RS);
if s.covered = False then
    在 RS 中寻找一个能覆盖 s 的代表模式 R;
    if 不存在这样的代表模式 R then
        用 s 的候选代表模式，创建一个新的代表模式 R_new，把 R_new 放入 RS;
        for GS 里的每个入口 Q(图模式 Q) do
            if R_new 能覆盖 Q then
                Q.covered = True;
    else
        for GS 里的每个入口 Q(图模式 Q) do
            if R 能覆盖 Q then
                Q.covered = True;
弹出 GS 的栈顶;
```

可以覆盖以 s 为根分枝中的所有频繁模式。因此，我们可以跳过以 s 为根的分枝 (裁剪该子树)。在第 4 行，$s \neq \min(s)$ 判断 s 是否是当前模式的最小 DFS 编码。如果不是，可以跳过以 s 为根的分枝 [145]。在第 7 行，算法根据 p 和 s 的之间的距离 $\mathrm{Dis}(p, s)$ 更新 $p.\min\_\mathrm{distance}$。在第 8 行，算法将当前模式 s 的 DFS 编码中的最后一条边放入全局栈 GS。GS 跟踪图模式空间中从根结点到当前结点 (当前模式 s) 的所有图模式。GS 中的每个入口对应一个图模式 Q，含有如下信息:(1)Q 的最小 DFS 编码中的最后一条边；(2)Q 的支持度；(3)Q 的覆盖标记 $Q.covered$；(4)Q 的候选代表模式 $Q.R$。第 9-13 行，扫描 GS 中的每个图模式 Q，测试当前模式 s 是否能覆盖 Q。如果 s 能覆盖 Q 并且 s 大于 Q 的候选代表模式 $Q.R$，根据 4.5.2.2.1 中的贪心策略，用 s 替换 $Q.R$。第 14 行，扫描数据库发现当前模式 s 的所有频繁最右扩展孩子。第 15-18 行，对 s 的每个频繁最右扩展孩子 $s \diamond_r e$，$(s \diamond_r e).\mathrm{min\_distance}$ 被初试化为最大值 1，递归调用子过程 MiningReprePatterns 继续深度优先搜索。随着对 $s \diamond_r e$ 的孩子的遍历，$(s \diamond_r e).\mathrm{min\_distance}$ 的值被逐渐减小。当 $(s \diamond_r e).\mathrm{min\_distance}$ 小于距离阀值 ϵ 时，根据 4.5.2.2.2 节中描述的思想，$s \diamond_r e$ 的剩余的没被遍历的孩子分枝就可以被裁剪掉。在遍历当前模式 s 的所有后裔之后，算法在第 19 行测试 s 是否已经被覆盖。如果 s 没被覆盖，在第 20 行扫描当前的代表模式集合 RS，试图发现一个代表模式 R 能覆盖 s。如果这样的代表 R 不存在，算法在第 22 行用 s 的候选代表模式创建一个新的代表模式 R_{new}，放入 RS，并且算法扫描 GS 中的每个图模式 Q(第 23 行)，判断 R_{new} 能否覆盖 Q(第 24 行)。如果 R_{new} 能覆盖 Q，则标记 Q 已经被覆盖 $Q.\mathrm{covered} = \mathrm{True}$(第 25 行)。如果在 RS 中发现了某个代表模式 R 能覆盖 s，也扫描 GS 中的每个图模式 Q(第 29 行)，判断 R 是否能覆盖 Q(第 30 行)。如果 R 能覆盖 Q，则标记 Q 已经被覆盖 (第 31 行)。

4.6 实验结果及分析

我们进行了大量的实验来考查算法的挖掘结果质量、执行效率、可扩展性，以及

不同参数对算法性能和结果质量的影响。

4.6.1　实验设置

我们从一个真实的化合物数据中导出了若干个图集合。该化合物数据用来测试化合物对艾滋病病毒的抑制作用，含有大约 44 000 个化合物。根据对艾滋病病毒的抑制程度，每个化合物都被分为下面三类中的一类: CA(confirmed active)，CM(confirmed moderately) 和 CI(confirmed inactive)。其中，CA 含有 422 个化合物，CM 含有 1 081 个化合物，CI 含有剩余的化合物。我们根据类别，分别导出了三个图集合 CA，CM 和 CI。其中，CI 图集合是从所有 CI 化合物中随机选择 5 000 个化合物。

本章算法使用 C++ 语言实现，用带有-O3 优化选项的 g++ 编译。用于实验的计算机具有 PIV 3.0GHz CPU 和 1GB 内存，运行 RedHat Linux 8.0 操作系统。

Cluster-TopK 算法需要两个额外的参数: 聚类质量参数 δ(或 Δ)；跳跃距离阀值 ϵ。在下面的实验中，若无特别说明，δ 取 0.1，ϵ 取 0.01。参数 δ 和 ϵ 对算法的影响和在实际应用中的选择将在 4.6.4 节描述。

4.6.2　比较挖掘结果的质量

本小节比较算法 Greedy-TopK 和 Cluster-TopK 之间的挖掘结果质量，下一小节比较算法 Greedy-TopK 和 Cluster-TopK 之间的性能。表 2 显示了在不同的数据集合上，固定最小支持度 min_sup 等于 10%，变化不同的 K 值，算法 Greedy-TopK 和 Cluster-TopK 选择的 K 个图模式而得到的联合熵的大小 (MES 问题)。表 3 显示了在相同条件下，算法选择的 K 个图模式而得到的信息增益的大小 (MIGS 问题)。

表 1　根据求得的联合熵, 比较 Greedy-TopK 和 Cluster-TopK (min_sup=10%)

K	CA		CM		CI	
	Greedy-TopK	Cluster-TopK	Greedy-TopK	Cluster-TopK	Greedy-TopK	Cluster-TopK
5	4.533 27	4.563 31	4.632 26	4.606 91	4.635 27	4.738 49
10	6.562 51	6.592 19	7.710 09	7.693 38	7.935 17	7.903 42
15	7.312 81	7.242 08	8.842 97	8.804 95	9.735 96	9.678 08
20	7.675 65	7.603 08	9.237 50	9.214 60	10.585 2	10.496 3

可以看出，算法 Greedy-TopK 和 Cluster-TopK 挖掘出的 Top-K 模式在质量上非常接近。尽管 Cluster-TopK 算法没有理论上的近似比保证，但它输出的意义度量值 (联合熵和信息增益) 与 Greedy-TopK 算法相比最多差 1%。有时，Cluster-TopK 算法得到的结果还稍微优于 Greedy-TopK 算法。我们也将最小支持度的取值在 5% 到

30% 之间变化，用以比较结果差异。结果显示，Greedy-TopK 和 Cluster-TopK 给出的 Top-K 模式在质量上仍然非常接近。

表 2　根据求得的信息增益，比较 Greedy-TopK 和 Cluster-TopK(min_sup=10%)

K	CA(res. CM)		CM(res. CA)		CI(res. CA)	
	Greedy-TopK	Cluster-TopK	Greedy-TopK	Cluster-TopK	Greedy-TopK	Cluster-TopK
5	0.193 026	0.187 017	0.167 959	0.165 682	0.128 673	0.128 654
10	0.316 993	0.304 763	0.300 546	0.315 074	0.231 147	0.230 494
15	0.471 530	0.453 959	0.462 120	0.476 579	0.303 742	0.296 234
20	0.584 803	0.557 849	0.565 636	0.573 212	0.341 27	0.336 603

为了进一步验证不同算法的结果质量，我们使用算法 Greedy-TopK 和 Cluster-TopK 最大化信息增益产生的 Top-K 模式进行分类实验，比较分类性能。我们构造两个分类任务:(1) 对 CA 类和 CM 类中的化合物进行分类; (2) 对 CA 类和 CI 类中的化合物进行分类。带有缺省参数的 LIBSVM [26] 被用作分类模型。分类准确率使用 5 次交叉验证进行评价。ROC 曲线下的面积 (AUC) 被用来度量分类性能，AUC 越大，表示分类性能越好。

此外，我们也抽取了传统的 Top-K 模式 (根据信息增益对所有模式进行排序，从中选择前 K 个信息增益最大的模式) 进行比较。Trad-TopK 表示抽取传统 Top-K 模式的算法。

我们使用算法 Trad-TopK, Greedy-TopK 和 Cluster-TopK 分别从正类和反类中抽取 K 个图模式, 建立分类模型。表 3 显示了 K 变化时, 不同的 Top-K 模式构造的分类器的分类性能 (AUC)。可以看出, Greedy-TopK 和 Cluster-TopK 产生的 Top-K 模式在分类性能方面远远优于传统的 Top-K 模式。因为传统的 Top-K 模式只考虑优化单一模式的意义度量，导致结果集中存在很多结构类似的图模式，降低了整体的分类性能。Greedy-TopK 和 Cluster-TopK 优化模式集合中所有模式的联合意义, 隐含排斥结构类似的图模式，从而保证了整体的分类性能。此外，Greedy-TopK 和 Cluster-TopK 产生的 Top-K 模式在分类性能方面也非常接近，再次说明 Greedy-TopK 和 Cluster-TopK 的挖掘结果质量差异很小。

表 3　不同 Top-K 模式构造的分类器的分类性能 (AUC)(min_sup=10%)

K	CA vs CM			CA vs CI		
	Trad-TopK	Greedy-TopK	Cluster-TopK	Trad-TopK	Greedy-TopK	Cluster-TopK
5	0.603 2	0.753 0	0.738 8	0.722 2	0.828 4	0.826 4
10	0.660 8	0.796 6	0.801 3	0.569 9	0.907 4	0.902 0
15	0.684 7	0.803 0	0.799 7	0.616 1	0.921 6	0.925 5
20	0.717 9	0.802 3	0.795 8	0.763 5	0.924 4	0.923 2

最后，我们从结构上比价一下不同算法产生的 Top-K 模式。图 ??、图 6 和图 7 分别显示了算法 Trad-TopK，Greedy-TopK 和 Cluster-TopK 产生的 Top-5 模式。可以看出，Trad-TopK 产生的 Top-5 图模式在结构上有很大重叠，因为它只考虑单一模式的意义，而不考虑模式的联合意义。Greedy-TopK 和 Cluster-TopK 考虑了模式的联合意义，产生的 Top-5 图模式在结构上彼此之间有很大的差异。在实际应用中，这恰恰是用户想要挖掘的模式类型。

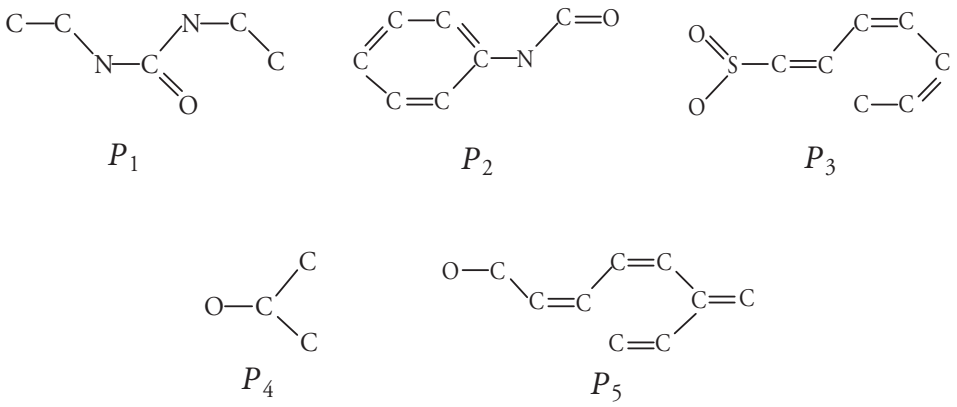


图 6　Greedy-TopK 算法产生的 Top-5 模式 (MIGS，min_sup=10%)

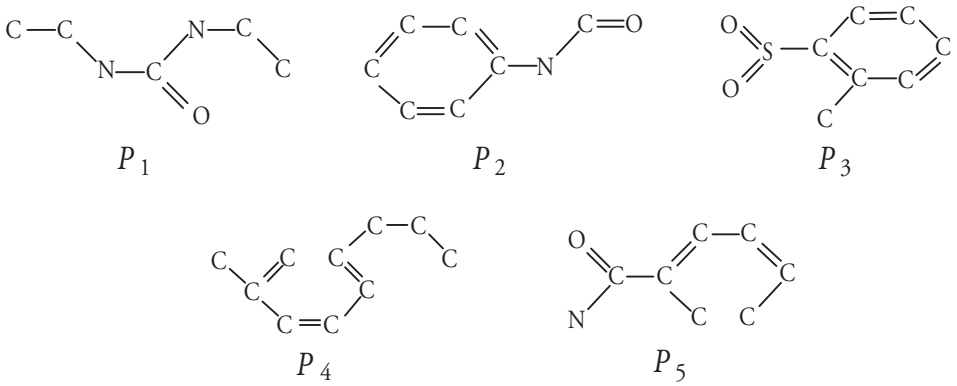


图 7　Cluster-TopK 算法产生的 Top-5 模式 (MIGS，min_sup=10%)

4.6.3　比较算法的效率

本小节比较算法 Greedy-TopK 和 Cluster-TopK 的执行效率。在 Greedy-TopK 的第一步，既可以使用 gSpan [145] 挖掘频繁图模式，又可以使用 CloseGrpah [146] 挖掘频繁闭图模式。因此，我们比较了 Greedy-TopK 的两个版本，gSpan+Greedy-TopK 和

CloseGrpah+Greedy-TopK。注意到，在与 Cluster-TopK 比较时，Greedy-TopK 利用了 4.5.1.1 节和 4.5.1.2 节中的裁剪技术以获得最高的执行效率。

图 8 显示了当最小支持度变化时，不同算法求解 MES 问题所用的时间。图 9 显示了当最小支持度变化时，不同算法求解 MIGS 问题所用的时间。如果一个算法不能在一个小时内完成，我们就终止算法。因此，图中 gSpan+Greedy-TopK 的结果是不完整的。

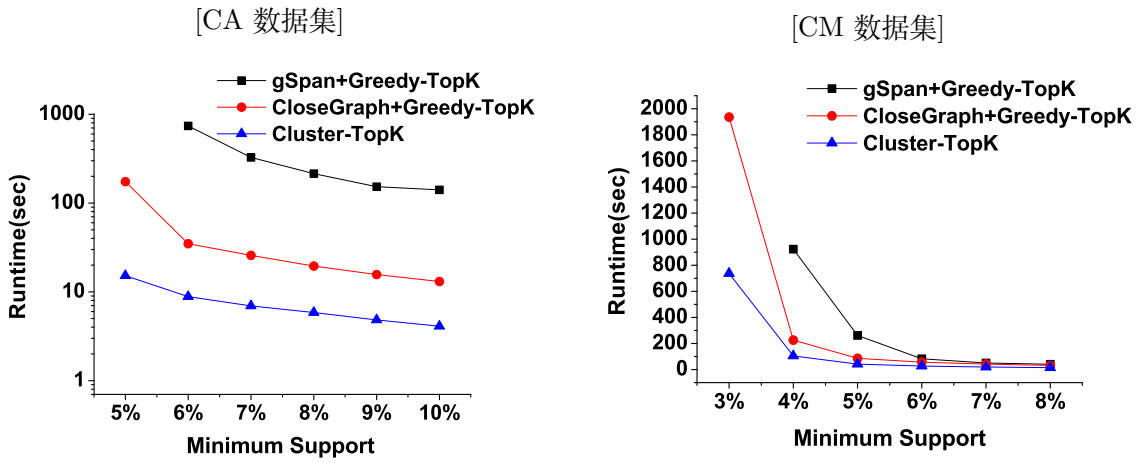


图 8　支持度的变化对算法执行时间的影响 (MES，K=10)

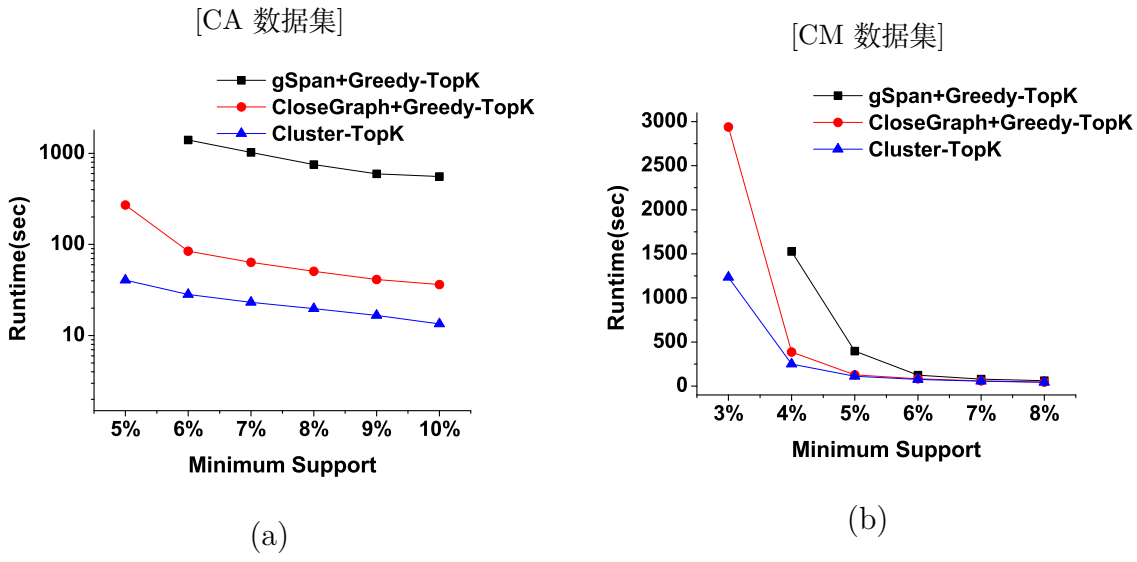


图 9　支持度的变化对算法执行时间的影响 (MIGS，K=10)

从图 8 和图 9 可以看出，在执行效率方面，算法 Cluster-TopK 极大地优于 Greedy-TopK。例如: 在 CA 数据集合上，Cluster-TopK 比 gSpan+Greedy-TopK 快 2 个数量级，比 CloseGrpah+Greedy-TopK 快 1 个数量级。而且，支持度越低，Cluster-TopK 和 Greedy-TopK 的执行效率差异越大。Cluster-TopK 的高效率来源于两个方面:(1) 在挖掘代表模式时，因为 Cluster-TopK 裁剪掉了很多不产生 (或少产生) 代表模式的分枝，与 CloseGraph 挖掘闭图模式相比，Cluster-TopK 遍历了模式空间中更少的结点，获得了更高的执行效率。(2) 在贪心选择时，因为 Cluster-TopK 产生的代表图模式数量比 CloseGraph 产生的闭图模式数量少很多，因此 Cluster-TopK 贪心选择的执行效率也比 Greedy-TopK 贪心选择的执行效率快很多。

下面评价算法 Greedy-TopK 中使用的裁剪技术 (见 4.5.1.1 节和 4.5.1.2 节) 的有效性。为此，我们形成了 Greedy-TopK 算法的四个变体: Greedy-TopK-1 表示不使用裁剪技术的 gSpan+Greedy-TopK; Greedy-TopK-2 表示使用裁剪技术的 gSpan+Greedy-TopK; Greedy-TopK-3 表示不使用裁剪技术的 CloseGraph+Greedy-TopK; Greedy-TopK-4 表示使用裁剪技术的 CloseGraphn+Greedy-TopK。图 10 显示了当最小支持度变化时，算法 Greedy-TopK 的不同变体所需的执行时间。可以看出，4.5.1.1 节和 4.5.1.2 节中的裁剪技术能有效地改善算法 Greedy-TopK 的效率，支持度越低，改善越明显。

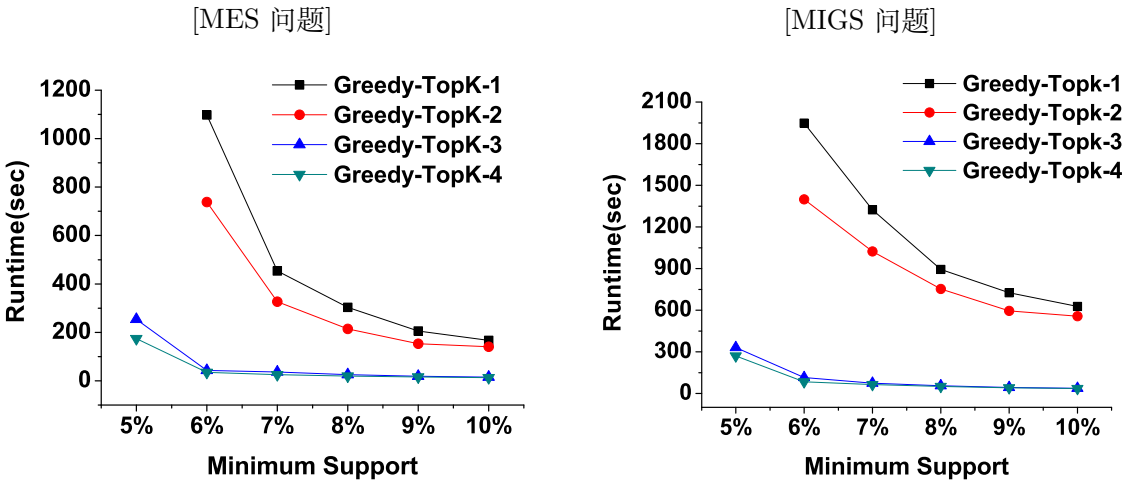


图 10　支持度变化时算法 Greedy-TopK 的不同变体所需的执行时间 (CA 数据集，K=10)

4.6.4　不同参数对 Cluster-TopK 算法的影响

本小节评价算法 Cluster-TopK 中所使用的两个参数: 聚类质量参数 δ 和跳跃距离阀值 ϵ，对算法的挖掘结果和运行时间的影响。由于相对跳跃距离阀值 δ(相对值) 和

Δ(绝对值) 具有完全相同的作用，我们只给出对 δ 的评价。

我们先评价聚类质量参数 δ 对算法的影响，固定跳跃距离阀值 $\epsilon = 0.01$，变化 δ 的值。Cluster-TopK 算法的运行时间由两部分构成，聚类时间和贪心选择时间。我们使用 T_{Cluster}，T_{Greedy} 和 T_{Total} 分别表示算法的聚类时间、贪心选择时间和总的时间(单位为秒)。*Rep* 表示 Cluster-TopK 算法聚类之后产生的代表模式数量，M 表示最后输出的度量值。表 4 显示了当 δ 值变化时，算法 Cluster-TopK 的结果质量和运行时间。可以看出，当 δ 增加时, 聚类时间 T_{Cluster} 不受影响。然而，随着 δ 值的增加，输出的代表模式数量逐渐减少，因此，贪心选择时间 T_{Greedy} 逐渐减小，总的时间也逐渐减小。一个有趣的现象是 δ 值对最后度量值 M 的影响微乎其微。在 4.5.2.1 节，我们分析了当 δ 值给定时，Cluster-TopK 的结果与 Greedy-TopK 的结果在理论上的最大可能差异。而实际中的差异比这个理论上的最大差异少得多，这再次说明了算法 Cluster-TopK 采用的聚类策略非常适合于本章所研究的问题。

表 4　聚类质量参数 δ 对算法 Cluster-TopK 的挖掘结果和运行时间的影响

δ	*Rep*	CA，min_sup=5%							
		Joint Entropy(MES)				Information gain(MIGS) (res. CM)			
		M	T_{Cluster}	T_{Greedy}	T_{Total}	M	T_{Cluster}	T_{Greedy}	T_{Total}
0.05	976	6.574 83	5.91	11.66	17.57	0.297 455	5.92	45.51	51.43
0.1	760	6.592 19	5.90	9.14	15.04	0.303 155	5.91	34.85	40.76
0.15	631	6.491 19	5.89	7.78	13.7	0.301 489	5.90	29.54	35.44
0.2	551	6.491 2	5.91	7.04	12.95	0.301 288	5.89	26.55	32.44
0.25	489	6.446 4	5.90	6.33	12.23	0.284 938	5.91	23.83	29.74
0.3	453	6.459 05	5.92	5.85	11.77	0.291 065	5.90	21.81	27.71

下面评价跳跃距离阀值 ϵ 对算法的影响，固定聚类质量参数 δ=0.1，变化 ϵ 的值。表 5 显示了当 ϵ 值变化时，算法 Cluster-TopK 的结果质量和运行时间。可以看出，当 ϵ 增加时，聚类时间 T_{Cluster} 明显减少。这是因为大的 ϵ 值使得图模式空间中更多的分枝被裁剪掉，从而聚类时间 T_{Cluster} 被大大缩减。当 ϵ 增加时，由于输出的代表模式数量也在逐渐减少，因此贪心选择时间 T_{Greedy} 和总的时间也逐渐减小。与聚类质量参数 δ 相比，跳跃距离阀值 ϵ 对结果质量的影响更小。例如: 当 ϵ 从 0.01 变化到 0.1 时，最后的度量值几乎不发生变化。在前面的实验中，我们只使用了小的 ϵ 值 0.01，就能使算法 Cluster-TopK 比 CloseGraph+Greedy-TopK 快 1 个数量级。如果使用更大的 ϵ 值 (例如 0.1) 会使算法 Cluster-TopK 更快，而且对结果质量几乎没有影响。这恰恰说明了算法 Cluster-TopK 所采用的裁剪策略能在保证结果质量的前提下，极大地改善算法的效率。

表 5　距离阀值 ϵ 对算法 Cluster-TopK 的挖掘结果和运行时间的影响

ϵ	*Rep*	CA，min_sup=5%							
		Joint Entropy(MES)				Information gain(MIGS) (res. CM)			
		M	T_{Cluster}	T_{Greedy}	T_{Total}	M	T_{Cluster}	T_{Greedy}	T_{Total}
0.01	760	6.592 19	5.9	9.31	15.21	0.303 155	5.9	34.71	40.61
0.02	731	6.592 19	5.36	9.01	14.37	0.303 155	5.36	33.78	39.14
0.03	697	6.592 19	4.49	8.14	12.63	0.303 155	4.49	31.2	35.69
0.04	675	6.592 19	3.93	7.98	11.91	0.303 155	3.93	30.13	34.06
0.05	644	6.592 19	3.13	7.08	10.21	0.303 155	3.12	27.05	30.17
0.06	633	6.592 19	2.93	6.95	9.88	0.303 155	2.93	26.71	29.64
0.07	626	6.592 19	2.9	6.93	9.83	0.303 155	2.9	26.63	29.53
0.08	616	6.592 19	2.71	6.78	9.49	0.303 155	2.71	26.15	28.86
0.09	612	6.592 19	2.59	6.72	9.31	0.300 521	2.59	25.89	28.48
0.1	574	6.592 19	2.44	6.45	8.89	0.300 521	2.44	24.53	26.97
0.15	465	6.598 96	1.83	5.14	6.97	0.292 042	1.83	19.63	21.46
0.2	355	6.522 53	1.31	4.23	5.54	0.284 786	1.31	16.37	17.69

4.6.5　算法可扩展性

本小节研究算法 Cluster-TopK 的可扩展性。我们从 1 倍到 10 倍复制 CA 数据集，选择 min_sup = 5% 和 6%，运行 Cluster-TopK，实验结果如图 11 所示。从图 11 可以看出，算法 Cluster-TopK 有很好的可扩展性。随着数据量的增加，Cluster-TopK 的运行时间也在线性地增加。

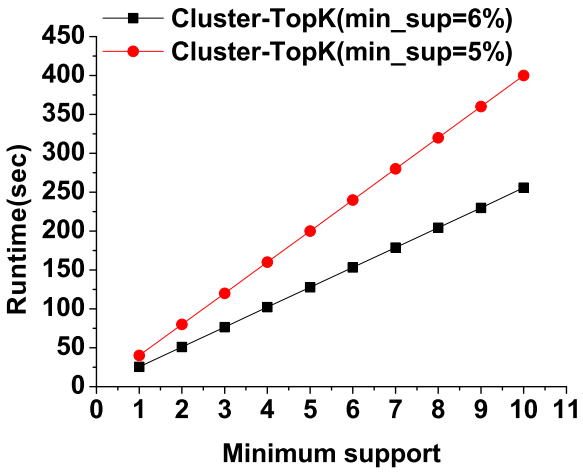


图 11　Cluster-Topk 的可扩展性 (MIGS，K=10)

4.7　本章小结

本章提出了一个新的研究问题，基于联合意义度量挖掘 Top-K 图模式，并给出两个高效算法 Greedy-TopK 和 Cluster-TopK。Greedy-TopK 有近似比保证，但是当

频繁子图数量较多时挖掘效率较低。Cluster-TopK 没有近似比保证，但是挖掘效率更高。实验结果显示本章提出的 Top-K 挖掘优于传统的 Top-K 挖掘。Cluster-TopK 比 Greedy-TopK 快至少一个数量级。而且，Cluster-TopK 的挖掘结果质量非常接近于 Greedy-TopK 的挖掘结果质量。本章提出的算法是一种通用算法，也适用于其他的联合意义度量和模式类型 (例如: 项集模式、序列模式、树模式等)。

第 5 章　基于显露模式的图分类方法

5.1　引言

近年来，由于图模型在生物信息学、化学、社会网络分析等领域的广泛应用，图挖掘和图数据管理逐渐成为热点研究内容。很多基于图的数据挖掘算法和查询处理技术已经被提出。例如: 文献 [20] [66] [83] [103] [145] 研究了图的频繁模式挖掘算法；文献 [34] 和 [147] 研究了图的索引方法和查询处理方法；文献 [47] 和 [123] 研究了社会网络中结点之间的关系。

图分类与预测是图挖掘的一个重要研究分支，在医学、生物学等领域有着广泛的应用背景。例如: 在药物设计方面，对化合物进行化学实验以检测它对病毒的抑制作用及其对人体的副作用将是一个耗时且昂贵的实验过程。为了设计一个新的抗病毒药物，对数据库中所有可能的化合物进行这种化学实验，时间和费用都是无法接受的。如果使用图分类技术，根据少量化合物的实验结果建立分类模型，扫描化合物数据库，识别出那些最有可能抗病毒的化合物，再在识别出的化合物上进行化学实验，将极大地降低药物设计的费用和周期。再例如: 在计算生物学领域，蛋白质的氨基酸序列通过折叠可以形成蛋白质二级结构和蛋白质三级结构。蛋白质二级结构可以用平面标号图来表示，而蛋白质三级结构可以用带有几何信息的空间几何图来表示。通过在蛋白质二级结构和三级结构上建立分类预测模型，可以帮助生物学家分析和识别蛋白质的基本功能。

传统的分类方法 [54](如决策树归纳、贝叶斯方法等) 并不能直接用于图数据，因为这些方法都假定数据以关系或向量的形式存在。为此，研究人员提出了一些针对图数据的分类方法。这些分类方法大致可分成两类:(1) 基于特征的图分类方法；(2) 基于核函数的图分类方法。在基于特征的图分类方法中，最有代表性的方法 [42,43] 是使用频繁子图作为分类特征进行分类，包含三个主要步骤:(1) 挖掘频繁子图；(2) 选择分类特征；(3) 构造分类模型。该方法存在的主要问题是挖掘频繁子图时如何设置最小支持度。设置过高，只能产生少量极其频繁的子图，影响分类器的性能；设置过低，频繁子图挖掘算法又不能在合理时间内完成挖掘。使用基于核函数的图分类方法构造的核函数来度量图之间的相似性。代表性的方法包括: 基于环模式的图核方法 [63]，基于随机游走的图核方法 [77,99]，基于最短路径的图核方法 [22]，基于最优局部分配的图核方法 [48] 等。基于核函数的图分类方法存在的主要问题是: 很多图核函数的计算都是 NP

完全问题。因此，很多基于核函数的图分类方法只适用于小规模的图。

如果图模式 P 的支持度从一个类别到另一个类别急剧变化 (例如:P 在正类的支持度是 35%，而 P 在反类的支持度是 0.02%)，直觉上，P 具有很强的分类能力，非常适合作为分类特征。这样的模式在文献 [44] 中被称为显露模式，并被用于关系数据的分类，取得了很好的分类效果。本章重点研究如何利用显露模式对图数据进行分类。

本章提出了一种基于频繁闭显露模式的图分类方法 CEP。CEP 包括三个主要步骤:(1) 挖掘频繁闭图模式；(2) 过滤非显露模式；(3) 构造分类规则。第一步，CEP 使用闭图挖掘算法 CloseGraph [146] 挖掘所有频繁闭图模式作为候选分类特征。与频繁子图挖掘算法相比，闭图挖掘算法可以使用更小的支持度来挖掘更大的图集合，产生更多对分类有重要作用的候选分类特征。而且，由闭图模式构成的候选分类特征既不会丢失对分类有用的信息，在数量上又远远小于由频繁子图构成的候选分类特征。第二步，CEP 保留频繁闭图模式中的显露模式。第三步，CEP 根据剩余的显露模式构造分类规则。在构造分类规则时，我们提出了一个新的特征选择方法，该方法既考虑显露模式如何覆盖图数据，又考虑显露模式在图数据中如何分布。这是因为，相同特征在图中的不同位置很可能导致对图的作用不相同。第三步产生的分类规则也容易被特定领域的专业人员理解和利用。在化合物上的分类实验结果显示:CEP 的分类性能优于文献 [43] 中的图分类方法，考虑特征分布的特征选择方法能有效地提高分类性能。

综上，本章的主要贡献如下: 第一，提出了一个新的图分类方法 CEP。第二，给出了一个新的特征选择方法，该方法既考虑特征如何覆盖图数据，又考虑特征在图数据中如何分布。第三，通过理论分析，给出图模式的支持度与其分类预测能力之间的关系。第四，实验结果验证了 CEP 的分类性能和新的特征选择方法的有效性。

本章的内容安排如下:5.2 节介绍相关工作；5.3 节介绍预备知识并给出问题定义；5.4 节详细介绍 CEP 方法；5.5 节通过实验验证 CEP 方法并与文献 [43] 中的分类方法进行比较；5.6 节对本章进行小结。

5.2 相关工作

分类是机器学习和数据挖掘领域中的一个基本研究问题。研究人员已经提出了大量的分类方法，主要包括判定树归纳分类方法、贝叶斯分类方法、神经网络分类方法和支持向量机分类方法等 [54]。这些传统分类方法通常假定数据是以关系或者向量的形式存在。分类问题的一个最大特点是: 不存在任何分类方法在所有环境下都是最优的。必须针对特定的领域和特定的数据类型，设计合适的分类方法。随着数据类型的变化，研究人员又提出了针对特定数据类型的分类方法。例如: 文献 [156] 研究了对文本数据的分类；文献 [90] 研究了对序列数据的分类。

近年来，由于图数据的大量出现，对图数据的分类也自然引起了研究人员的极大兴趣。图分类方法大致可分成两类: 基于特征的分类方法；基于核函数的分类方法。

按照特征来源的不同，基于特征的分类方法又可分为: (1) 基于物理化学特性的分类方法；(2) 基于拓扑特征或几何特征的分类方法。基于物理化学特性的分类方法需要领域专家根据应用背景和领域知识事先指定合适的分类特征。其优点是方法简单，避免出现" 过拟合" 现象。缺点是表达能力不足，会丢失图数据的重要信息，导致分类性能差。基于拓扑特征或几何特征的分类方法主要是使用频繁拓扑子图或频繁几何子图作为分类特征进行分类 [42,43]。该类方法的优点是可以获得任意拓扑 (或几何) 结构的特征，具有良好的分类性能和扩展性。缺点是挖掘频繁子图时，最小支持度参数很难确定。如果最小支持度设置过高，只能产生少量极其频繁的图模式，极大地降低了分类器的准确率；如果最小支持度设置过低，产生的图模式数量又太多，分类模型很难构造，甚至频繁子图挖掘算法都不能在合理的时间内完成挖掘。

基于核函数的分类方法在很多应用中已被证明优于其他的方法 [63]。为此，研究人员针对图分类问题，提出了一些图核函数 [22,48,63,77,99]。本质上，图核函数可以看作两个图之间的相似程度的一种度量。例如: 将图分解成基本成分集合，两个图的核函数可以定义为对应基本成分集合交集的大小。文献 [63] 给出了一种基于环模式的图核方法。该方法根据图中环模式集合和树模式集合定义图核函数，然后利用支持向量机进行分类。然而，该核函数的计算只适用于环个数被某一常数所限制的图。而且这种分类方法只在图中具有天然环结构 (例如化合物) 的情况下才具有较高的分类性能。如果图中没有环结构或者只有少量环结构，该方法并不可行。文献 [77] 和 [99] 提出了一种基于随机游走的图核方法。文献 [22] 提出了一种基于最短路径的图核方法。文献 [48] 提出了一种最优局部分配的图核方法。通常，基于核函数的分类方法具有良好的分类性能。然而，很多图核函数的计算都是 NP 完全问题。因此，很多图核方法可扩展性差，只适用于小规模的图数据。

随着频繁模式挖掘技术的进步，基于频繁模式的分类方法也受到了普遍关注。文献 [31] 分析了基于频繁模式的分类方法在数据分类时具有的优点，并给出了一个基于频繁模式的分类方法。该方法首先挖掘所有频繁模式，然后使用特征选择方法挑选强分类特征，再将数据映射到特征空间中，用支持向量机建立分类模型。该方法在对关系数据分类时，取得了很好的分类效果。然而，该方法的缺点是需要挖掘所有频繁模式，当支持度设置过低，该方法的可扩展性差。为此，文献 [32] 利用分枝限界技术从数据库中直接挖掘强分类特征，用来改善基于频繁模式的分类方法的效率和可扩展性。基于频繁模式的分类方法如果使用所有频繁模式将会导致过适应问题，因而需要一个特征选择过程从所有可用的频繁模式中选择对分类起主要作用的特征。目前的特征选择方法包括 CBA [95] 和 CMAR [94]，这些特征选择方法只考虑特征如何覆盖数据，不

考虑特征在数据中如何分布。文献 [43] 指出，在图数据中，相同特征在图中的不同位置很可能导致对图的作用也是不相同的。因此，对图数据分类时，特征选择方法需要考虑特征在图中的分布情况。显露模式的概念最早在文献 [44] 中提出，并用于关系数据的分类。

本章提出了一种新的图分类方法 CEP。CEP 使用频繁闭显露模式来构造分类规则。CEP 与现有分类方法的区别在于:(1) 提出了一个新的特征选择方法，该方法既考虑特征对数据库图的覆盖，又考虑特征在数据库图中如何分布；(2) 在选择分类特征之后，直接根据特征构造分类规则，而不是将数据映射到特征空间；(3) 分类特征从频繁闭显露模式中选择，而不是从所有频繁模式中选择。实验结果显示，CEP 分类方法在分类性能方面优于文献 [43] 中的图分类方法。

5.3 问题定义

详细介绍 CEP 方法之前，首先给出有关的基本概念和问题定义。

现用 $\text{support}_{\text{rel}}(p; D)$ 表示图模式 p 在数据库 D 中的相对支持度；用 $\text{support}_{\text{abs}}(p; D)$ 表示图模式 p 在数据库 D 中的绝对支持度。

定义 5.3.1(显露比) 给定两个图数据库 D_1 和 D_2，一个图模式 p 从 D_1 到 D_2 的显露比定义为 $\text{EMR}(p; D_1, D_2) = \text{support}_{\text{rel}}(p; D_1)/\text{support}_{\text{rel}}(p; D_2)$。

直觉上，如果图模式 p 从一个类别图数据库到另一个类别图数据库的显露比非常大的话 (例如:p 在正类图数据库的相对支持度是 35%，p 在反类图数据库的相对支持度是 0.02%)，则图模式 p 应该被选作分类特征。

定义 5.3.2(显露模式) 给定两个图数据库 D_1 和 D_2 和一个显露比阀值 min_EMR，如果图模式 p 从 D_1 到 D_2 的显露比 $\text{EMR}(p; D_1, D_2) \geqslant \text{min\_EMR}$，则称图模式 p 为从 D_1 到 D_2 的显露模式。

本章研究的图分类问题定义如下:

输入: 图数据库 $D = \{(G_i, C_i)|i = 1, 2, \ldots, n\}$，其中，$G_i$ 是一个图结构，C_i 表示 G_i 所属的类别。

输出: 一个分类模型，用来判断新图所属类别。

为了便于描述，本章只讨论二元分类问题。对 CEP 方法稍加改动，也可以用于多元分类问题。

5.4 CEP 分类方法

CEP 分类方法的框架如图 1 所示。CEP 包括三个主要步骤:(1) 挖掘频繁闭图模

式; (2) 过滤非显露模式; (3) 构造分类规则。第一步, 将图数据库根据类别分为正例图数据库 PD 和反例图数据库 ND。然后使用闭图模式挖掘算法 CloseGraph [146], 分别对正例图数据库 PD 和反例图数据库 ND 进行挖掘, 得到正例中的频繁闭图模式集合 PFC 和反例中的频繁闭图模式集合 NFC。第二步, 对 PFC 和 NFC 分别进行非显露模式过滤, 得到正例中的闭显露模式集合 PEP 和反例中的闭显露模式集合 NEP。第三步, 从 PEP 和 NEP 分别构造正例分类规则集合 PCR 和反例分类规则集合 NCR。下面将详细描述以上各个步骤。

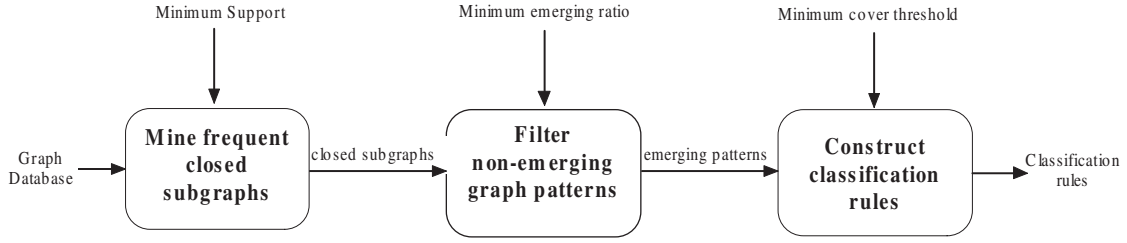


图 1　CEP 分类方法框架

5.4.1　挖掘频繁闭图模式

在基于频繁模式的图分类方法中, 一个重要的问题就是最小支持度的设置。如果最小支持度设置得过高, 只能产生少量高度频繁的模式, 将严重影响分类器的性能。如果最小支持度设置得过低, 模式挖掘算法需要太长的运行时间或者由于内存耗尽而不能完成挖掘任务。即使完成了挖掘任务, 由于产生的图模式太多, 也会严重影响分类模型的构造以及分类模型的实际应用。

如果仅从分类性能角度考虑, 原则上应该使用尽可能低的支持度阀值, 这样将会得到更多对分类起关键作用的图模式。下面从信息论的角度分析图模式的支持度与其分类能力之间的关系。

为了便于分析, 假设只有两个类别 $C = \{0, 1\}$。v_p 是表示某个图模式 p 的随机变量。在一个图实例中观察到图模式 p 时, 获得的信息增益是 $IG(C|v_p) = H(C) - H(C|v_p)$, 其中 $H(C)$ 是无条件熵, $H(C|v_p)$ 是在图模式 p 出现的条件下的条件熵。在类分布固定的情况下, 无条件熵 $H(C)$ 是一个常数。设 $H_{\min}(C|v_p)$ 是 $H(C|v_p)$ 的最小值, $IG_{\max}(C|v_p)$ 是 $IG(C|v_p)$ 的最大值, 我们有 $IG_{\max}(C|v_p) = H(C) - H_{\min}(C|v_p)$。

假设图模式 p 的相对支持度为 s, 那么 $P(v_p = 1) = s$。再假设 $P(C = 1) = m$, $P(C = 1|v_p = 1) = n$, 我们有 $H(C|v_p) = -\sum\limits_{v_p \in \{0,1\}} P(v_p) \sum\limits_{C \in \{0,1\}} P(C|v_p) \log P(C|v_p) = -s(n \log n + (1-n)\log(1-n)) - (1-s)(\frac{m-sn}{1-s} \log \frac{m-sn}{1-s} + \frac{(1-s)-(m-sn)}{1-s} \log \frac{(1-s)-(m-sn)}{1-s})$。

显然，$H(C|v_p)$ 是 s，m 和 n 的函数。容易证明，当 s 和 m 为常量时，$H(C|v_p)$ 是 n 的凹函数。如果 s 和 m 为满足不等式 $s \leqslant m \leqslant 1/2$ 的常量，则 $H(C|v_p)$ 在 $n=0$ 或者 $n=1$ 时取得最小值。假设 $H(C|v_p)$ 在 $n=0$ 时取得最小值，那么 $n=1$ 的情况可进行类似分析。$H_{\min}(C|v_p) = (s-1)(\frac{m}{1-s}\log\frac{m}{1-s} + \frac{1-s-m}{1-s}\log\frac{1-s-m}{1-s})$。在类分布固定的情况下，$m$ 是一个常数。$H_{\min}(C|v_p)$ 对 s 求偏导数，有 $\frac{\partial H_{\min}(C|R)|_{n=0}}{\partial s} = \log\frac{1-s-m}{1-s} \leqslant 0$。

因此，当 $s \leqslant m \leqslant 1/2$ 时，$H_{\min}(C|R)_{n=0}$ 随着 s 而单调递减，$IG_{\max}(C|R)$ 随着 s 而单调递增，一个图模式的支持度与它可能具有的最大分类预测能力成正比。

通过以上分析，可以得出下面的结论: 在图分类方法中，应该使用尽可能小的支持度阀值来挖掘图模式，这样会减少对分类有用的图模式的丢失。

然而，太低的支持度阀值又导致图模式挖掘算法无法完成。为解决支持度阀值选择问题，图分类方法 [42,43] 在挖掘图模式时，尽可能地降低支持度阀值到某一极限，如果再降低，产生的图模式太多而不能被有效地处理。例如: 图分类方法 [43] 使用频繁子图挖掘算法 FSG [83] 挖掘 NCI/CA 时，支持度阀值最小只能取 10%，将产生 2 万多个频繁图模式。

为了获得更高的分类性能，本章的 CEP 分类方法只挖掘频繁闭图模式。与挖掘频繁图模式相比，挖掘频繁闭图模式有下面几个优点:(1) 在绝大多数情况下，频繁图模式的数量远远大于频繁闭图模式的数量。例如: 在图模式挖掘和图分类经常使用的实验数据 NCI/CA [146] 中，当最小支持度为 5% 时，将产生大约 100 万个频繁图模式。而在这些频繁图模式中，只有不到 2 000 个闭图模式。因此，挖掘闭图模式既可以加速挖掘过程，又加快了 CEP 框架中显露模式生成、分类规则构造等后续步骤，从而缩短了分类模型的构造时间。同时，也有利于分类模型在实际中的应用。(2) 前面已经证明图模式的分类能力与其支持度密切相关，小的支持度阀值会减少对分类有用的图模式信息的丢失。如果使用闭图模式挖掘算法，就可以使用更小的支持度阀值，产生更多对分类至关重要的图模式信息。(3) 更为重要的是，挖掘闭图模式不会丢失任何对分类有用的信息，因为从闭图模式集合，能恢复所有的频繁图模式及其支持度。而且，在实际应用中，领域专家只对闭图模式感兴趣。

在二元分类问题中，正例的数量通常远远小于反例的数量。例如: 在 CA 和 CI 的分类问题 [43,63] 中，正例的数量只是反例数量的 1%。在这种情况下，如果在整个图集合中挖掘闭图模式，正例中很多对分类至关重要的图模式将丢失。因此，在挖掘之前，我们先将整个图数据库根据类别分成正例图数据库 PD 和反例图数据库 ND。然后使用闭图模式挖掘算法 CloseGraph [146] 分别挖掘正例图数据库 PD 和反例图数据库 ND，得到正例中的频繁闭图模式集合 PFC 和反例中的频繁闭图模式集合 NFC。

5.4.2　过滤非显露图模式

是否所有的频繁闭图模式对分类都非常重要呢? 当然不是。如果一个图模式在正例和反例中的支持度相差不大，那么该图模式对分类的预测作用也非常有限。只有那些其支持度从一个类别到另一个类别变化巨大的图模式才能对分类起到重要的预测作用。例如: 假设一个图模式 p 在正例中的支持度为 35%，在反例中的支持度为 0.2%。如果一个未知类别的新图 G 含有图模式 p，那么我们可以说图 G 在正例中的几率为 99.4%。这样的图模式经常被称为显露模式 [44]。

因此，在 CEP 方法的第二步，我们分别从正例的频繁闭图模式集合 PFC 和反例的频繁闭图模式集合 NFC 中过滤掉非显露模式。算法 11 给出了如何过滤 PFC 中非显露模式，过滤 NFC 中非显露模式可类似的处理。

Algorithm 11: 过滤 PFC 中非显露模式的算法

Input: 正例中的频繁闭图模式集合 PFC，反例图数据库 ND，显露比阀值 λ
Output: 正例中的显露模式集合 PEP

```
for PFC 里的每个图模式 p do
    计算 p 在 ND 中的相对支持度 support_rel(p, ND);
    if support_rel(p, PD)/support_rel(p, ND) ⩾ λ then
        把 p 放入 PEP;
输出 PEP;
```

在算法 11 中，需要 $|PFC| \times |ND|$ 个子图同构测试。子图同构测试是 NP 完全问题，因此当闭图模式数量很大时算法 11 的效率很低。而且，如果 ND 太大不能完全装入内存，算法 11 需要多遍磁盘扫描。为提高效率，应该尽可能减少子图同构测试次数和磁盘扫描次数。

下面我们给出一个更高效的算法来过滤 PFC 中的非显露模式。该算法首先将 PFC 中的所有频繁闭图模式组织成一个树型结构，用 T 表示，T 的根结点为空图。对 PFC 中的任意两个图模式 p_1 和 p_2，如果满足下面三个条件:(1)$p_1 \subset p_2$，(2) 不存在 $p \in PFC$ 使得 $p_1 \subset p \subset p_2$，(3)$p_1$ 的 DFS 编码小于 p_2 的 DFS 编码，则将 p_2 作为 p_1 的孩子结点。如果在得到所有频繁闭图模式之后，再构造这样的树 T，不可避免地会执行大量的子图同构测试。实际上，在闭图模式挖掘算法 CloseGraph [146] 深度优先遍历图模式空间的过程中，我们很容易得到闭图模式之间的超图–子图关系，并不需要额外的子图同构测试。因此，我们可以在 CloseGraph [146] 执行过程中同时构造树 T。CloseGraph 运行完成，T 也被构造完成。

在得到 T 之后，对 ND 中的每个图 G，采用深度优先方式遍历树 T。如果 G 包含当前结点 $p(p \subseteq G)$，p 的计数加 1，继续遍历 p 的孩子结点。如果 G 不包含当前结点 $p(p \not\subseteq G)$，根据 Aprior(反单调) 性质，G 也不可能包含 p 的孩子结点，我们可以

略过对 p 为根的子树的遍历。这样做，我们只需扫描 NS 一次，就可以得到 PFC 中所有图模式在 NS 中的 (相对) 支持度，极大地减少了子图同构测试次数。改进的算法如算法 12 所示。

Algorithm 12: 过滤 PFC 中非显露模式的改进算法

Input: 正例中的频繁闭图模式集合 PFC，反例图数据库 ND，PFC 中闭图模式形成的树型结构 T，显露比阀值 λ
Output: 正例中的显露模式集合 PEP

```
for ND 里的每个图 G do
    node = T.root;
    调用 Traverse_Tree(G, node);
for PFC 里的每个图模式 p do
    support_rel(p, ND) = p.count/|ND|;
    if support_rel(p, PD)/support_rel(p, ND) ⩾ λ then
        把 p 放入 PEP;
输出 PEP;
```

Algorithm 13: 遍历树 T 的子过程

Input: 反例图数据库 ND 中的一个图 G，树 T 的一个结点 node
Output: 如果 G 包含 node 为根子树中的某个结点 p，则 p 的计数加 1

```
for node 的每个孩子 c do
    if c ⊆ G then
        c.count++;
        调用 Traverse_Tree(G, c);
```

5.4.3 构造分类规则

显然，PEP 中的每个显露模式 p 都对应一个分类规则:$p \rightarrow$ 正例 (p 在正例中的支持度，p 的显露比)。NEP 中的每个显露模式 q 也对应一个分类规则:$q \rightarrow$ 反例 (q 在反例中的支持度，q 的显露比)。尽管可以直接使用这些分类规则对一个新实例进行分类，然而这样做却有下面的问题。

为了不丢失对分类有重要预测作用的显露模式，显露比阀值 λ 通常取很小的数值。这样，显露模式对应的分类规则的数量仍然非常大。在对一个新实例分类之前，首先需要检查它与哪些分类规则相匹配 (包含哪些显露模式)，这涉及大量的子图同构测试。子图同构测试是 NP 完全问题，时间复杂性很高。在实际应用中 (例如: 在新药研制方面)，分类模型通常是在小的化合物训练集上构造而后用于含有几百万个化合物的数据库。因此，如果直接使用所有的分类规则将严重限制分类方法的实际应用。

因此，有必要从所有的分类规则中选择少量对分类起决定作用的分类规则。分类规则选择相当于从显露模式中进行特征选择，因为每个显露模式都对应一个分类规则。在介绍 CEP 的特征选择方法之前，先定义一个分类规则等级的概念。

分类规则等级: 任意两个分类规则 R_1 和 R_2，如果下面的条件之一成立，我们就说 R_1 的等级高于 R_2，用 $R_1 \succ R_2$ 表示。(1)R_1 的显露比高于 R_2。(2) R_1 和 R_2 的显露比相等，R_1 的支持度高于 R_2。(3)R_1 和 R_2 的显露比和支持度都相等，R_1 的规模小于 R_2。

根据分类规则等级，正例 (或反例) 的分类规则可以形成线性顺序。这时，我们可以采用类似 CBA [95] 或者 CMAR [94] 的顺序覆盖方法来选择少量对分类起决定作用的分类规则。但是，CBA 和 CMAR 只考虑数据包含哪些模式，并没有考虑这些模式在数据中如何分布。

在图的环境中，除了要考虑每个图包含哪些模式，还应该考虑这些模式在图中如何分布。例如: 设 p_1 和 p_2 是两个显露模式且 $p_1 \subset p_2$，如果 p_1 在一个图 G 中的每一次出现都伴随着 p_2 的出现，那么 p_2 对图 G 的分类将起主要的预测作用。在已经选择 p_2 的前提下，再选择 p_1 就没有什么意义。

本章的特征 (或分类规则) 选择算法，充分考虑了模式在图中的分布情况，进一步提高了 CEP 的分类性能，这一点已经在实验中得到验证。

在正例 (或反例) 的分类规则排成线性顺序之后，我们给每个规则分配一种颜色。例如: 用颜色 1 代表等级最高的分类规则，用颜色 2 代表等级第二高的分类规则，依此类推。用 color(R) 表示分类规则 R 对应的颜色。因为分类规则和显露模式一一对应，显露模式 p 的颜色也用 color(p) 表示。对 PEP 中的每个显露模式 p，定义 cover_color$(p, PEP) = \{\text{color}(p_i) | p_i \subset p, p_i \in PEP\}$。cover_color$(p, PEP)$ 是 PEP 中所有能被 p 包含的显露模式的颜色集合。利用过滤非显露模式时构造的树结构 T(见 5.4.2 节的描述)，很容易计算 cover_color(p, PEP)。算法 14 给出了如何从 PEP 中产生对正例分类起决定作用的分类规则。类似地，CEP 方法也从 NEP 中产生对反例分类起决定作用的分类规则。

我们简单解释一下算法 14。输入参数 θ 是覆盖阀值，它控制一个图被几个显露模式覆盖时才可以被删除。图的一个顶点可能被多个分类规则的显露模式所覆盖，为了表示这种情况，我们允许每个顶点可以有多种颜色，每种颜色对应一个分类规则。我们用 color(v) 表示顶点 v 所包含的颜色集合。在第 3-7 行将所有顶点的颜色集合初试化为空。第 12 行，每当一个显露模式 p 与图 G 中的一个子图 emb 相匹配 (子图同构) 时，我们先检查第 13 行中的条件是否成立。如果条件成立，则说明子图 emb 目前还没有被包含 p 的显露模式所覆盖，显露模式 p 对图 G 应该起预测作用。在第 14-16 行，我们先删除被 p 包含的显露模式在子图 emb 中的覆盖信息 (被 p 包含的显露模式的预测作用被 p 代替)，然后记录显露模式 p 覆盖 emb(记录 p 的预测作用)。第 20-22 行，若图 G 被 θ 个显露模式所覆盖，则从 PD 中删除 G。第 23-25 行，若整个图 G

Algorithm 14: 产生分类规则的算法 (特征选择算法)

Input: 正例中的显露模式集合 PEP，正例图数据库 PD，覆盖阀值 θ
Output: 正例分类规则集合 PCR

```
1  把对应 PEP 中每个显露模式的分类规则放入 RS;
2  根据分类规则的等级排序 RS;
3  for PD 里的每个图 G do
4      for G 里的每个结点 v do
5          color(v) = ∅;
6  for RS 里的每个分类规则 R do
7      R.matching = false;
8      p = R 的显露模式;
9      for PD 里的每个图 G do
10         for p 在 G 中的每次出现 emb do
11             if ⋂_{v∈emb} color(v) = ∅ then
12                 for emb 里的每个结点 v do
13                     color(v) = (color(v) − cover_color(p, PEP)) ∪ color(r);
14                 R.matching = true;
15         if |⋃_{v∈G} color(v)| = θ then
16             从 PD 中删除 G;
17         if ∀v ∈ G, color(v) ≠ ∅ then
18             从 PD 中删除 G;
19     if R.matching = true then
20         把 R 放入 PCR;
```

的所有顶点已经被显露模式所覆盖，则从 PD 中删除 G。最后，若分类规则 R 至少对一个图 G 起预测作用，则把 R 作为分类规则输出。

5.4.4 使用分类规则进行分类

得到正例分类规则集合 PCR 和反例分类规则集合 NCR 之后，就可以对一个未知类别的图 G 进行分类。为了尽可能精确地分类，CEP 分类方法从 PCR 和 NCR 分别寻找若干个匹配图 G 的分类规则。然后，根据它们的支持度和显露比预测图 G 的类别。算法 15 给出了如何对一个未知类别的图进行分类。

算法 15 实现时，要注意两点: (1) 在第 1-2 步寻找匹配规则时，要像 5.4.3 节构造分类规则那样，考虑显露模式彼此包含情况，使得某个分类规则的预测作用不会完全被另一个分类规则所代替。(2) 经常会存在这种情况，真正匹配的规则数少于给定的数量，这时就使用实际匹配的规则。

Algorithm 15: ClassifyGraph 算法

Input: 要分类的图 G，正例分类规则集合 PCR，反例分类规则集合 NCR，正例分类规则的最小匹配数 M_1，反例分类规则的最小匹配数 M_2

Output: G 的类别 (正例或者反例)

按照等级从高到低的顺序，从 PCR 中寻找 M_1 个能匹配 G 的分类规则 $\{R_1^+, R_2^+, \ldots, R_{M_1}^+\}$;

按照等级从高到低的顺序，从 NCR 中寻找 M_2 个能匹配 G 的分类规则 $\{R_1^-, R_2^-, \ldots, R_{M_2}^-\}$;

$\text{Score} = \sum_{i=1}^{M_1} R_i^+.s \times R_i^+.r - \sum_{i=1}^{M_2} R_i^-.s \times R_i^-.r$; /* $R_i^+.s$ 是 R_i 的显露模式的支持度，$R_i^+.r$ 是 R_i 的显露模式的显露比 */

if Score > 0 **then**

    预测 G 属于正类;

else

    预测 G 属于负类;

5.5　实验结果及分析

所有实验都在内存 512M, CPU 为 Pentium-4 3.2G, 操作系统为 Redhat 7.0 的 PC 机上进行。所有的算法都在 STL 库支持下用 C++ 实现，用带有 O3 选项的 g++ 编译。在 CEP 分类方法中, 需要挖掘频繁闭图模式, 因此我们也实现了 CloseGraph [146]。

为了与文献 [43] 和 [63] 中的分类方法比较，本章使用了与文献 [43] 和 [63] 相同的实验数据: NCI-HIV 化合物数据。所有 NCI-HIV 化合物根据它们对人体 CEM 细胞的保护程度被分成三个类别: CA(强烈保护)，CM(一般保护) 和 CI(没有保护)。在 44 000 多个 NCI-HIV 化合物中，422 个属于 CA，1 081 个属于 CM，其余属于 CI。与文献 [43] 和 [63] 中的分类问题一样，我们也考虑下面的三个二元分类问题:(1) CA vs CM；(2) CA+CM vs CI；(3) CA vs CI。

因为文献 [43] 和 [63] 用 ROC 曲线 [109] 来度量分类器的性能，为了能与文献 [43] 和 [63] 中分类方法进行比较，我们也用 ROC 曲线来度量 CEP 分类器的性能。在二维 ROC 空间中，分类器对应的 ROC 曲线在 x 轴上显示被错误分类的反例的比率，在 y 轴上显示被正确分类的正例的比率。一个分类器对应的 ROC 曲线面积越大，说明该分类器的分类性能越好。

对每个分类问题, 实验都进行 5 次交叉验证, 求 ROC 曲线面积的均值和方差。表 1 给出了 CEP 分类器构造时使用的各种参数以及输出的平均分类规则数和平均构造时间。

表 1　CEP 分类器构造时使用的参数

Parameters	CA vs CM	CA+CM vs CI	CA vs CI
minimum support	(0.05,0.03)	(0.03,0.02)	(0.05,0.02)
minimum emerging ratio	(2,2)	(2,2)	(2,2)
minimum cover threshold	(5,5)	(5,5)	(5,5)
number of output rules	(47,62)	(156,27)	(112,35)
construction time(s)	413	1586	1131

表 1 中括号里的数值分别是正例和反例的参数设置。在每次交叉验证中，分类器构造完成之后，设置算法 5.4.4 中正例分类规则和反例分类规则的最小匹配数为 (5,5)，用算法 15 进行分类。表 2 给出了 CEP 分类方法对应的 ROC 曲线面积。为了与文献 [43] 和 [63] 中的分类方法比较，表 2 也显示了在文献 [43] 和 [63] 中给出的 ROC 曲线面积。

表 2　各种分类方法对应的 ROC 面积

Classification Approaches	CA vs CM	CA+CM vs CI	CA vs CI
FSG	0.810	0.794	0.908
FSG+3D	0.821	0.819	0.940
CPK	0.8273(0.013)	0.8011(0.017)	0.9285(0.010)
γCPK	0.8398(0.010)	0.8332(0.013)	0.9426(0.007)
CEP	0.8439(0.011)	0.8457(0.006)	0.9611(0.009)

在表 2 中，FSG 和 FSG+3D 是文献 [43] 中的分类方法。FSG 根据频繁拓扑子结构来建立分类模型，FSG+3D 根据频繁几何子结构来建立分类模型。CPK 和 γCPK 是文献 [63] 中的分类方法。CPK 是通过普通环模式核来建立分类模型，γCPK 是通过高斯环模式核来建立分类模型。表 2 中的数值是每种分类方法对应的 ROC 曲线面积的均值和方差。在表 2 中，我们给出了文献 [43] 和 [63] 中的分类方法在各种参数下最好的结果。通过比较 ROC 曲线的面积，可以看出 CEP 在分类性能上要明显优于文献 [43] 和 [63] 中的分类方法。尽管 CEP 没有使用几何图模式，但它的分类性能仍然要优于 FSG+3D。将来我们要考虑如何在 CEP 中利用几何模式进一步提高 CEP 的分类性能。

下面，我们通过实验验证 CEP 中所使用的特征选择方法的有效性。现用 CEP+CBA 表示在 CEP 框架中使用传统的特征选择方法 CBA。表 3 给出了 CEP 使用不同特征选择方法时的分类性能。从表 3 可以看出，本章给出的特征选择方法优于传统的特征选择方法 CBA。

表 3　测试 CEP 中特征选择方法的有效性

Classification Approaches	CA vs CM	CA+CM vs CI	CA vs CI
CEP+CBA	0.840 7(0.008)	0.841 9(0.012)	0.956 8(0.012)
CEP	0.843 9(0.011)	0.845 7(0.006)	0.961 1(0.009)

CEP 好的分类性能主要得益于下面三个方面:(1) 使用更低的支持度来挖掘频繁闭图模式。(2) 过滤频繁闭图模式中的非显露模式。(3) 在分类规则选择时，既考虑显露模式如何覆盖图数据，又考虑显露模式在图中如何分布。

此外，CEP 的分类模型也容易被理解和使用。CEP 的分类模型由一系列分类规则构成。领域专家很容易理解和利用这种形式的分类规则。例如: 化学家可以分析这

些分类规则，设计更好的化合物。

5.6　本章小结

本章给出了一种基于频繁闭显露模式的图分类方法 CEP。CEP 首先挖掘频繁闭图模式，然后从闭图模式中得到显露模式，最后根据显露模式构造一系列分类规则。实验结果显示，在对化合物数据分类时，CEP 在分类性能上优于文献 [43] 的图分类方法。除了图数据，CEP 分类框架也可用于其他数据类型的分类，例如: 关系数据分类、文本分类等。文献 [43] 中已经显示，在对化合物分类时，基于几何模式的图分类方法要优于基于拓扑模式的图分类方法。在我们的 CEP 分类框架中，利用几何模式有可能会进一步提高 CEP 的分类性能。我们以后的研究工作将在这方面开展。

第 6 章　结　　论

图挖掘在化学、计算生物学、Web 网络分析、社会网络分析等领域都有广泛的应用。本书重点研究了图挖掘中的模式挖掘技术，提出了一些新的图模式挖掘问题和有效的解决方法。本书主要创新点如下:

提出从图数据库中挖掘代表模式问题及其有效解决方法。给出了该问题的形式化定义，并证明了该问题是 NP-hard 问题。提出了一系列新的概念:δ-覆盖图、跳跃值、δ-跳跃模式等。发现了 δ-跳跃模式的一个重要性质:δ-跳跃模式一定是代表模式。提出了挖掘代表模式的三个算法: RP-FP, RP-GD, RP-Leap。RP-FP 和 RP-GD 挖掘完整的代表模式集合，RP-Leap 挖掘近似的代表模式集合。RP-FP 从频繁闭图模式中计算代表模式，具有紧的近似比保证，但效率和可扩展性差。RP-GD 直接从图数据库中挖掘代表模式，没有近似比保证，但效率和可扩展更高。RP-Leap 以丢失少量代表模式的代价，取得了比 RP-GD 快至少一个数量级的性能改善。实验结果表明: 挖掘代表模式可以极大地减少图模式的输出数量，基于代表模式构造的分类器在分类性能方面并不低于基于闭图模式构造的分类器，从而证明了代表模式中仍然保留了重要而有意义的图模式。由于图代表了最通用的模式类型，本书给出的三个代表模式挖掘算法也能用来挖掘其他类型的代表模式 (例如: 项集代表模式、序列代表模式、树代表模式等)。

提出从图数据库中挖掘核心子结构问题及其有效的解决方法。根据实际应用中核心子结构的特征，给出了它的形式化定义，称为 Δ-跳跃模式。发现了 Δ-跳跃模式的很多重要性质，例如: Δ-跳跃模式是稳定的，它们对噪声和数据的变化不敏感，Δ 值越大，它们的抗干扰能力越强。为高效挖掘 Δ-跳跃模式，提出了两种新的裁剪技术，基于内扩展的裁剪和基于外扩展的裁剪。利用这两种裁剪技术，设计了一个高效的挖掘算法 GraphJP。理论上证明了这两种裁剪技术的正确性及算法 GraphJP 的正确性。实验结果表明: 这两种新的裁剪技术能有效地裁剪图模式搜索空间，算法 GraphJP 高效可扩展，发现的核心子结构具有重要的应用价值。由于图代表了最通用的模式类型，本书给出的 Δ-跳跃模式定义和挖掘算法 GraphJP 也容易被扩展来挖掘其他类型的跳跃模式 (例如: 项集跳跃模式、序列跳跃模式、树跳跃模式等)。

提出基于联合意义度量的 Top-K 图模式挖掘问题及其有效解决方法。讨论了适用于图模式集合的联合意义度量，并利用信息论中的概念 (联合熵和信息增益) 给出了两个具体的问题定义 MES 和 MIGS，证明了它们是 NP-hard 问题。提出了两个高效的 Top-K 挖掘算法 Greedy-TopK 和 Cluster-TopK。针对 MES 和 MIGS 这两个具体问题

的意义度量，设计了一系列新的裁剪技术，进一步提高了算法的效率。当意义度量满足 submodular 性质时，Greedy-TopK 能提供近似比保证，但当频繁图模式数量较多时，Greedy-TopK 效率和扩展性差，Cluster-TopK 没有近似比保证，但 Cluster-TopK 的挖掘效率远远高于 Greedy-TopK 的挖掘效率。实验结果表明: 本书提出的 Top-K 挖掘在结果质量和可用性方面要远远优于传统的 Top-K 挖掘，Cluster-TopK 比 Greedy-TopK 快一到两个数量级，而且 Cluster-TopK 的挖掘结果质量非常接近于 Greedy-TopK 的挖掘结果质量。Greedy-TopK 和 Cluster-TopK 是通用的 Top-K 挖掘算法，也适用于其他的联合意义度量和模式类型 (例如: 项集模式、序列模式、树模式等)。

提出了一种基于频繁闭显露模式的图分类框架 CEP。CEP 包括三个主要步骤:(1) 挖掘频繁闭图模式；(2) 过滤非显露模式；(3) 构造分类规则。在过滤非显露模式时，需要计算图模式在不同类别数据库中的支持度，提出了一种有效的算法，可以极大地减少此过程所需的子图同构测试次数。在构造分类规则时，提出了一个新的特征选择方法，该方法既考虑显露模式如何覆盖图数据，又考虑显露模式在图数据中如何分布。实验结果表明:CEP 具有良好的分类性能，而且，领域专家容易理解和利用 CEP 产生的分类规则。

理论分析和实验结果表明，本书提出的图模式挖掘技术的研究具有一定的创新性，具体工作已被国内外同行多次引用 [27,164,165,169]，具有重要的学术和应用价值。由于现实世界中的图数据还可能具有动态性、不确定性，以及数据流等多种形式，因此本书的后续研究将在如下几个方面开展:

(1) 从动态图中挖掘图模式；

(2) 从不确定图中挖掘图模式；

(3) 从图流中挖掘图模式。

参考文献

[1] The international network for social network analysis. http://www.insna.org/.

[2] National cancer institute. http://dtp.nci.nih.gov/docs/aids/searches/list.html/.

[3] National library of medicine. http://chem.sis.nlm.nih.gov/chemidplus/.

[4] S. salamone. data storage trends require new management solutions. http://m.zdnet.com.au/120265411.htm.

[5] E. Abdelhamid, M. Canim, M. Sadoghi, B. Bhattacharjee, Y.-C. Chang, and P. Kalnis. Incremental frequent subgraph mining on large evolving graphs. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2710–2723, 2017.

[6] N. Acosta-Mendoza, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, A. Gago-Alonso, and J. E. Medina-Pagola. Mining clique frequent approximate subgraphs from multi-graph collections. *Applied Intelligence*, 50(3):878–892, 2020.

[7] F. Afrati, A. Gionis, and H. Mannila. Approximating a collection of frequent sets. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 12–19, Seattle, WA, USA, 2004. ACM Press.

[8] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 207–216, Washington, D.C., USA, 1993. ACM Press.

[9] R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *20th International Conference on Very Large Data Bases (VLDB)*, pages 487–499, Trondheim, Norway, 1994. VLDB Endowment.

[10] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pages 37–48, 2013.

[11] A. G. Alonso, J. E. M. Pagola, and J. A. Carrasco-Ochoa. Mining frequent connected subgraphs reducing the number of candidates. In *Proceeding of the 12th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 365–376, Antwerp, Belgium, 2008. Springer-Verlag.

[12] A. Angel, N. Koudas, N. Sarkas, and D. Srivastava. Dense subgraph maintenance

under streaming edge weight updates for real-time story identification. *arXiv preprint arXiv:1203.0060*, 2012.

[13] S. Aridhi and E. M. Nguifo. Big graph mining: Frameworks and techniques. *Big Data Research*, 6:1–10, 2016.

[14] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *12nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 44–54, Philadelphia, PA, USA, 2006. ACM Press.

[15] D. Bandyopadhyay, J. Huan, J. Liu, J. Prins, J. Snoeyink, W. Wang, and A. Tropsha. Structure-based function inference using protein family-specific fingerprints. *Protein Science*, 15(6):1537–1543, 2006.

[16] H. M. Berman, J. Westbrook, Z. Feng, and G. G. et al. The protein data bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[17] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà. Mining frequent closed graphs on evolving data streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 591–599, 2011.

[18] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann. Netgan: Generating graphs via random walks. In *International conference on machine learning*, pages 610–619. PMLR, 2018.

[19] M. Boley, T. Horvath, and S. Wrobel. Efficient discovery of interesting patterns based on strong closedness. In *The 9th SIAM International Conference on Data Mining*, pages 1002–1013, Sparks, NV, USA, 2009. Society for Industrial and Applied Mathematics.

[20] C. Borgelt and M. R. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *IEEE International Conference on Data Mining (ICDM)*, pages 51–58, Maebashi, City, Japan, 2002. IEEE Computer Society Press.

[21] C. Borgelt, T. Meinl, and M. R. Berthold. Moss: A program for molecular substructure mining. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, pages 6–15, New York, NY, USA, 2005. ACM Press.

[22] K. M. Borgwardt and H.-P. Kriegel. Shortest-path kernels on graphs. In *IEEE International Conference on Data Mining (ICDM)*, pages 74–81, Houston, Texas, USA, 2005. IEEE Computer Society Press.

[23] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[24] G. Buehrer, S. Parthasarathy, and Y.-K. Chen. Adaptive parallel graph mining for cmp architectures. In *IEEE International Conference on Data Mining (ICDM)*, pages 97–106, Hong Kong, China, 2006. IEEE Computer Society Press.

[25] S. Cao, W. Lu, and Q. Xu. Deep neural networks for learning graph representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[26] C. Chang and C. Lin. Libsvm: a library for support vector machines. software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 2001.

[27] C. Chen, C. X. Lin, X. Yan, and J. Han. On effective presentation of graph patterns: a structural representative approach. In *Proceeding of the 17th ACM conference on Information and knowledge management (CIKM)*, pages 299–308, Napa Valley, California, USA, 2008. ACM Press.

[28] C. Chen, X. Yan, P. S. Yu, J. Han, D.-Q. Zhang, and X. Gu. Towards graph containment search and indexing. In *33th International Conference on Very Large Data Bases (VLDB)*, pages 926–937, University of Vienna, Austria, 2007. VLDB Endowment.

[29] M. Chen, I. W. Tsang, M. Tan, and T. J. Cham. A unified feature selection framework for graph embedding on high dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 27(6):1465–1477, 2014.

[30] H. Cheng, X. Yan, and J. Han. Mining graph patterns. In *Frequent pattern mining*, pages 307–338. Springer, 2014.

[31] H. Cheng, X. Yan, J. Han, and C.-W. Hsu. Discriminative frequent pattern analysis for effective classification. In *Proceedings of the 23th International Conference on Data Engineering (ICDE)*, pages 716–725, Istanbul, Turkey, 2007. IEEE Computer Society Press.

[32] H. Cheng, X. Yan, J. Han, and P. S. Yu. Direct discriminative pattern mining for effective classification. In *Proceedings of the 24th International Conference on Data Engineering (ICDE)*, pages 169–178, Cancun, Mexico, 2008. IEEE Computer Society Press.

[33] J. Cheng, Y. Ke, and W. Ng. δ-tolerance closed frequent itemsets. In *IEEE International Conference on Data Mining (ICDM)*, pages 139–148, Hong Kong, China, 2006. IEEE Computer Society Press.

[34] J. Cheng, Y. Ke, W. Ng, and A. Lu. Fg-index: Towards verification free query processing on graph databases. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 857–872, Beijing, China, 2007. ACM Press.

[35] Y. Chi, Y. Yang, and R. R. Muntz. Indexing and mining free trees. In *IEEE International Conference on Data Mining (ICDM)*, pages 509–512, Melbourne, Florida, USA, 2003. IEEE Computer Society Press.

[36] Y. Chi, S. Zhu, X. Song, J. Tatemura, and B. L. Tseng. Structural and temporal analysis of the blogosphere through community factorization. In *13rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 163–172, San Jose, California, USA, 2007. ACM Press.

[37] J. Cohen. Graph twiddling in a mapreduce world. *Computing in Science & Engineering*, 11(4):29–41, 2009.

[38] D. J. Cook. *MINING GRAPH DATA*. John Wiley & Sons, Incorporated., Publication, March 2007.

[39] S. A. Cook. The complexity of theorem-proving procedures. In *Proceeding of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 151–158, Shaker Heights, Ohio, USA, 1971. ACM Press.

[40] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, September 2001.

[41] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1st edition, 8 1991.

[42] M. Deshpande, M. Kuramochi, and G. Karypis. Frequent sub-structure based approaches for classifying chemical compounds. In *IEEE International Conference on Data Mining (ICDM)*, pages 35–42, Melbourne, Florida, USA, 2003. IEEE Computer Society Press.

[43] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. Frequent sub-structure based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1036–1050, 2005.

[44] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 43–52, San Diego, CA, USA, 1999. ACM Press.

[45] G. Ehud, S. E. Shimony, and V. Natalia. Discovering frequent graph patterns using disjoint paths. *IEEE Transactions on Knowledge and Data Engineering*,

18(11):1441–1456, 2006.

[46] M. Elseidy, E. Abdelhamid, S. Skiadopoulos, and P. Kalnis. Grami: Frequent subgraph and pattern mining in a single large graph. *Proceedings of the VLDB Endowment*, 7(7):517–528, 2014.

[47] C. Faloutsos, K. S. McCurley, and A. Tomkins. Fast discovery of connection subgraphs. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 118–127, Seattle, WA, USA, 2004. ACM Press.

[48] H. Frohlich, J. K. Wegner, F. Sieker, and A. Zell. Optimal assignment kernels for attributed molecular graphs. In *Processings of the 22th International Conference on Machine Learning (ICML)*, pages 225–232, New York, NY, USA, 2005. ACM Press.

[49] J. Gao, C. Zhou, J. Zhou, and J. X. Yu. Continuous pattern detection over billion-edge graph using distributed framework. In *2014 IEEE 30th International Conference on Data Engineering*, pages 556–567. IEEE, 2014.

[50] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *31th International Conference on Very Large Data Bases (VLDB)*, pages 721–732, Trondheim, Norway, 2005. VLDB Endowment.

[51] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer, 1971.

[52] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[53] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, 2002.

[54] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2nd edition, March 2006.

[55] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 1–12, Dallas, Texas, USA, 2000. ACM Press.

[56] M. A. Hasan, V. Chaoji, S. Salem, J. Besson, and M. J. Zaki. Origami: Mining representative orthogonal graph patterns. In *IEEE International Conference on Data Mining (ICDM)*, pages 153–162, Omaha, NE, USA, 2007. IEEE Computer Society Press.

[57] H. He and A. K. Singh. Closure-tree: An index structure for graph queries. In

Proceedings of the 22nd International Conference on Data Engineering (ICDE), pages 38–47, Atlanta, Georgia, USA, 2006. IEEE Computer Society Press.

[58] H. He and A. K. Singh. Graphrank: Statistical modeling and mining of significant subgraphs in the feature space. In *IEEE International Conference on Data Mining (ICDM)*, pages 885–890, Hong Kong, China, 2006. IEEE Computer Society Press.

[59] P. Hintsanen and H. Toivonen. Finding reliable subgraphs from large probabilistic graphs. *Data mining and knowledge discovery*, 17(1), 2008.

[60] L. B. Holder and D. J. Cook. Graph-based relational learning: Current and future directions. *ACM SIGKDD Explorations Newsletter*, 5(1):90–93, 2003.

[61] L. B. Holder, D. J. Cook, and S. Djoko. Substructures discovery in the subdue system. In *1st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 169–180, Seattle, Washington, USA, 1994. IEEE Computer Society Press.

[62] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer, 3rd edition, 1996.

[63] T. Horvath, T. Gartner, and S. Wrobel. Cyclic pattern kernels for predictive graph mining. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 158–167, Seattle, WA, USA, 2004. ACM Press.

[64] T. Horvath, P. Ramon, and S. Wrobel. Frequent subgraph mining in outerplanar graphs. In *12nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 797–802, Philadelphia, PA, USA, 2006. ACM Press.

[65] H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21(1):213–221, Jan 2005.

[66] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *IEEE International Conference on Data Mining (ICDM)*, pages 549–552, Melbourne, Florida, USA, 2003. IEEE Computer Society Press.

[67] J. Huan, W. Wang, J. Prins, and J. Yang. Spin: Mining maximal frequent subgraphs from graph databases. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 581–586, Seattle, WA, USA, 2004. ACM Press.

[68] A. Inokuchi. Mining generalized substructures from a set of labeled graphs. In *IEEE International Conference on Data Mining (ICDM)*, pages 415–418,

Brighton, UK, 2004. IEEE Computer Society Press.

[69] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceeding of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 13–23, London, UK, 2000. Springer-Verlag.

[70] A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50(3):321–354, 2003.

[71] R. J. and B. Jr. Efficiently mining long patterns from databases. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 85–93, Seattle, Washington , USA, 1993. ACM Press.

[72] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1st edition, June 1998.

[73] G. Jeh and J. Widom. Mining the space of graph properties. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 187–196, Seattle, WA, USA, 2004. ACM Press.

[74] H. Jiang, H. Wang, P. S. Yu, and S. Zhou. Gstring: A novel approach for efficient search in graph databases. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE)*, pages 566–575, Istanbul, Turkey, 2007. IEEE Computer Society Press.

[75] R. Jin, M. Abu-Ata, Y. Xiang, and N. Ruan. Effective and efficient itemset pattern summarization: regression-based approaches. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 399–407, Las Vegas, Nevada, USA, 2008. ACM Press.

[76] R. Jin, C. J. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *11st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 606–611, Chicago, Illinois, USA, 2005. ACM Press.

[77] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Processings of the 20th International Conference on Machine Learning (ICML)*, pages 321–328, Washington, DC, USA, 2003. ACM Press.

[78] Y. Ke, J. Cheng, and W. Ng. Correlation search in graph databases. In *13rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 390–399, San Jose, California, USA, 2007. ACM Press.

[79] Y. Ke, J. Cheng, and W. Ng. Efficient correlation search from graph databases.

IEEE Transactions on Knowledge and Data Engineering, 20(12):1601–1615, 2008.

[80] N. S. Ketkar, L. B. Holder, and D. J. Cook. Comparison of graph-based and logic-based multi-relational data mining. *ACM SIGKDD Explorations Newsletter*, 7(2):64–71, 2005.

[81] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[82] Y. Koren, S. C. North, and C. Volinsky. Measuring and extracting proximity in networks. In *12nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 245–255, Philadelphia, PA, USA, 2006. ACM Press.

[83] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *IEEE International Conference on Data Mining (ICDM)*, pages 313–326, San Jose, California, USA, 2001. IEEE Computer Society Press.

[84] M. Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. In *IEEE International Conference on Data Mining(ICDM)*, pages 258–265, Maebashi, City, Japan, 2002. IEEE Computer Society Press.

[85] M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1038–1051, 2004.

[86] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. In *Proceedings of the 2004 SIAM Data Mining Conference*, pages 243–271, Lake Buena Vista, Florida, USA, 2004. Springer Netherlands.

[87] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. *Data mining and knowledge discovery*, 11(3):243–271, 2005.

[88] N. G. L., W. L. A., and F. M. L. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

[89] M. Lahiri and T. Y. Berger-Wolf. Structure prediction in temporal networks using frequent subgraphs. In *IEEE Symposium on Computational Intelligence and Data Mining*, pages 35–42, Honolulu, Hawaii, USA, 2007. IEEE Computer Society Press.

[90] N. Lesh, M. J. Zaki, and M. Ogihara. Mining features for sequence classification. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 342–346, San Diego, CA, USA, 1999. ACM Press.

[91] J. Leskovec and C. Faloutsos. Sampling from large graphs. In *12nd ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining, pages 631–636, Philadelphia, PA, USA, 2006. ACM Press.

[92] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *11st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 177–187, Chicago, Illinois, USA, 2005. ACM Press.

[93] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27, 2014.

[94] W. Li, J. Han, and J. Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In *IEEE International Conference on Data Mining (ICDM)*, pages 369–376, San Jose, California, USA, 2001. IEEE Computer Society Press.

[95] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 80–86, New York City, New York, USA, 1998. ACM Press.

[96] G. Liu and L. Wong. Effective pruning techniques for mining quasi-cliques. In *Proceeding of the 12th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 33–49, Antwerp, Belgium, 2008. Springer-Verlag.

[97] Y. Liu, X. Jiang, H. Chen, J. Ma, and X. Zhang. Mapreduce-based pattern finding algorithm applied in motif detection for prescription compatibility network. In *International Workshop on Advanced Parallel Processing Technologies*, pages 341–355. Springer, 2009.

[98] Y. Luo, J. Guan, and S. Zhou. Towards efficient subgraph search in cloud computing environments. In *International Conference on Database Systems for Advanced Applications*, pages 2–13. Springer, 2011.

[99] P. Mahe, N. Ueda, T. Akutsu, J.-L. Perret, and J.-P. Vert. Extensions of marginalized graph kernels. In *Processings of the 21th International Conference on Machine Learning (ICML)*, pages 70–78, Banff, Alberta, Canada, 2004. ACM Press.

[100] P. Meysman, Y. Saeys, E. Sabaghian, W. Bittremieux, Y. Van de Peer, B. Goethals, and K. Laukens. Discovery of significantly enriched subgraphs associated with selected vertices in a single graph. In *Proceedings of the 14th International Workshop on Data Mining in Bioinformatics. BIOKDD*, volume 15, pages 1–8, 2015.

[101] S. Morishita. Carcinogenesis predictions using ilp. In *Proceedings of the First International Conference on Discovery Science(DS)*.

[102] A. Mrzic, P. Meysman, W. Bittremieux, P. Moris, B. Cule, B. Goethals, and K. Laukens. Grasping frequent subgraph mining for bioinformatics applications. *BioData mining*, 11(1):1–24, 2018.

[103] S. Nijssen and J. N. Kok. A quickstart in frequent structure mining can make a difference. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 549–552, Seattle, WA, USA, 2004. ACM Press.

[104] T. Ozaki and T. Ohkawa. Mining correlated subgraphs in graph databases. In *The Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD*, pages 272–283, Osaka, Japan, 2008. Springer-Verlag.

[105] B. Ozdemir, W. Abd-Almageed, S. Roessler, and X. W. Wang. isubgraph: integrative genomics for subgroup discovery in hepatocellular carcinoma using graph mining and mixture models. *PloS one*, 8(11):e78624, 2013.

[106] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang. Adversarially regularized graph autoencoder for graph embedding. *arXiv preprint arXiv:1802.04407*, 2018.

[107] J. Pei, D. Jiang, and A. Zhang. On mining cross-graph quasi-cliques. In *11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 228–238, Chicago, Illinois, USA, 2005. ACM Press.

[108] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[109] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.

[110] S. Raghavan and H. Garcia-Molina. Representing web graphs. In *Proceedings of the 19th International Conference on Data Engineering (ICDE)*, pages 405–416, Bangalore, India, 2003. IEEE Computer Society Press.

[111] S. Ranu and A. K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *Proceedings of the 25th International Conference on Data Engineering (ICDE)*, pages 844–855, Shanghai, China, 2009. IEEE Computer Society Press.

[112] A. Ray, L. Holder, and S. Choudhury. Frequent subgraph discovery in large attributed streaming graphs. In *Proceedings of the 3rd international workshop on big data, streams and heterogeneous source mining: algorithms, systems, program-*

ming models and applications, pages 166–181. PMLR, 2014.

[113] D. Shasha, J. T. L. Wang, and R. Giugno. Algorithmics and applications of tree and graph searching. In *Principles of database systems (PODS)*, pages 39–52, Madison, Wisconsin, USA, 2002. ACM Press.

[114] D. Shasha, J. T. L. Wang, and S. Zhang. Unordered tree mining with applications to phylogeny. In *Proceedings of the 20th International Conference on Data Engineering (ICDE)*, pages 708–719, Boston, USA, 2004. IEEE Computer Society Press.

[115] N. Shrivastava, S. Navlakha, and R. Rastogi. Graph summarization with bounded error. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 419–432, Vancouver, Canada, 2008. ACM Press.

[116] A. Srinivasan, R. D. King, S. Muggleton, and M. J. E. Sternberg. On classification and regression. In *Proceedings of the 7th International Workshop on Inductive Logic Programming (ILP)*, pages 273–287, Prague, Czech Republic, 1997. Springer-Verlag.

[117] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *13rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 687–696, San Jose, California, USA, 2007. ACM Press.

[118] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *12nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 374–383, Philadelphia, PA, USA, 2006. ACM Press.

[119] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 32–41, Edmonton, Alberta, Canada, 2002. ACM Press.

[120] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *13rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 717–726, San Jose, California, USA, 2007. ACM Press.

[121] A. Termier, M.-C. Rousset, and M. Sebag. Dryade: a new approach for discovering closed frequent trees in heterogeneous tree databases. In *IEEE International Conference on Data Mining (ICDM)*, pages 543–546, Brighton, UK, 2004. IEEE

Computer Society Press.

[122] L. T. Thomas, S. R. Valluri, and K. Karlapalem. Margin: Maximal frequent subgraph mining. In *IEEE International Conference on Data Mining (ICDM)*, pages 1097–1101, Hong Kong, China, 2006. IEEE Computer Society Press.

[123] H. Tong and C. Faloutsos. Center-piece subgraphs: Problem definition and fast solutions. In *12nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 404–413, Philadelphia, PA, USA, 2006. ACM Press.

[124] H. Tong, C. Faloutsos, and Y. Koren. Fast direction-aware proximity for graph mining. In *13rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 747–756, San Jose, California, USA, 2007. ACM Press.

[125] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, Ramasamy Uthurusamy. *Advances in knowledge discovery and data mining*. The MIT Press. Menlo Park, CA, 1996.

[126] V. V. Vazirani. *Approxiamation Algorithms*. Springer, 1st edition, March 2004.

[127] C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 730–735, Philadelphia, PA, USA, 2006. ACM Press.

[128] C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi. Scalable mining of large disk-based graph databases. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 316–325, Seattle, WA, USA, 2004. ACM Press.

[129] J. Wang, W. Hsu, M. L. Lee, and C. Sheng. A partition-based approach to graph mining. In *Proceedings of the 22th International Conference on Data Engineering (ICDE)*, pages 74–83, Atlanta, Georgia, USA, 2006. IEEE Computer Society Press.

[130] J. Wang, Z. Zeng, and L. Zhou. Clan: An algorithm for mining closed cliques from large dense graph databases. In *Proceedings of the 22th International Conference on Data Engineering (ICDE)*, pages 73–82, Atlanta, Georgia, USA, 2006. IEEE Computer Society Press.

[131] K. Wang, X. Xie, H. Jin, P. Yuan, F. Lu, and X. Ke. Frequent subgraph mining in graph databases based on mapreduce. In *Asia-Pacific Services Computing Conference*, pages 464–476. Springer, 2016.

[132] T. Washio and H. Motoda. State of the art of graph-based data mining. *ACM SIGKDD Explorations Newsletter*, 5(1):59–68, 2003.

[133] D. B. West. *Introduction to Graph Theory*. China Machine Press, 2nd edition, October 2004.

[134] D. W. Williams, J. Huan, and W. Wang. Graph database indexing using structured graph decomposition. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE)*, pages 976–985, Istanbul, Turkey, 2007. IEEE Computer Society Press.

[135] M. Worlein. *Extension and parallelization of a graph mining algorithm*. PhD thesis, Friedrich-Alexander-Universitat, Germany, 2006.

[136] M. Worlein, T. Meinl, I. Fischer, and M. Philippsen. A quantitative comparison of the subgraph miners mofa, gspan, ffsm, and gaston. In *Proceeding of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 392–403, Porto, Portugal, 2005. Springer-Verlag.

[137] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[138] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong. Random walks: A review of algorithms and applications. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(2):95–107, 2019.

[139] F. Xia, K. Sun, S. Yu, A. Aziz, L. Wan, S. Pan, and H. Liu. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence*, 2(2):109–127, 2021.

[140] S. Xiang, F. Nie, C. Zhang, and C. Zhang. Nonlinear dimensionality reduction with local spline embedding. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1285–1298, 2008.

[141] D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent pattern sets. In *31th International Conference on Very Large Data Bases (VLDB)*, pages 709–720, Santiago de Chile, Chile, 2005. VLDB Endowment.

[142] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):40–51, 2006.

[143] X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: A profile-based approach. In *11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 314–323, Chicago, Illinois, USA, 2005. ACM Press.

[144] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining significant graph patterns by

leap search. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 433–444, Vancouver, Canada, 2008. ACM Press.

[145] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *IEEE International Conference on Data Mining (ICDM)*, pages 548–551, Maebashi, City, Japan, 2002. IEEE Computer Society Press.

[146] X. Yan and J. Han. Closegraph: Mining closed frequent graph patterns. In *9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 286–295, Washington, DC, USA, 2003. ACM Press.

[147] X. Yan, P. S. Yu, and J. Han. Graph indexing: A frequent structure based approach. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 335–346, Paris, France, 2004. ACM Press.

[148] X. Yan, P. S. Yu, and J. Han. Graph indexing based on discriminative frequent structure analysis. *ACM Transactions on Database Systems (TODS)*, 30(4):960–993, Dec 2005.

[149] X. Yan, P. S. Yu, and J. Han. Substructure similarity search in graph databases. In *ACM Special Interest Group on Management of Data (SIGMOD)*, pages 766–777, Baltimore, Maryland, USA, 2005. ACM Press.

[150] X. Yan, X. J. Zhou, and J. Han. Mining closed relational graphs with connectivity constraints. In *11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 324–333, Chicago, Illinois, USA, 2005. ACM Press.

[151] X. Yan, F. Zhu, J. Han, and P. S. Yu. Searching substructures with superimposed distance. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*, pages 88–97, Atlanta, Georgia, USA, 2006. IEEE Computer Society Press.

[152] X. Yan, F. Zhu, P. S. Yu, and J. Han. Feature-based similarity search in graph structures. *ACM Transactions on Database Systems (TODS)*, 31(4):1418–1453, Dec 2006.

[153] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang. Network representation learning with rich text information. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[154] G. Yang. The complexity of mining maximal frequent itemsets and maximal frequent patterns. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 344–353, Seattle, WA, USA, 2004. ACM Press.

[155] Y. Yang, F. Nie, S. Xiang, Y. Zhuang, and W. Wang. Local and global regressive mapping for manifold learning with out-of-sample extrapolation. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[156] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Processings of the 14th International Conference on Machine Learning (ICML)*, pages 412–420, Nashville, TN, USA, 1997. ACM Press.

[157] K. Yoshida, H. Motoda, , and N. Indurkhya. Graph-based induction as a unified learning framework. *Journal of Applied Intelligence*, 4(3):297–316, Jul 1994.

[158] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[159] M. J. Zaki. Efficiently mining frequent trees in a forest. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 71–80, Edmonton, Alberta, Canada, 2002. ACM Press.

[160] Z. Zeng, J. Wang, J. Zhang, and L. Zhou. Fogger: an algorithm for graph generator discovery. In *12nd International Conference on Extending Database Technology (EDBT)*, pages 517–528, Saint-Petersburg, Russia, 2009. Springer-Verlag.

[161] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Coherent closed quasi-clique discovery from large dense graph databases. In *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 797–802, Philadelphia, PA, USA, 2006. ACM Press.

[162] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Out-of-core coherent closed quasi-clique mining from large dense graph databases. *ACM Transactions on Database Systems(TODS)*, 32(2), 2007.

[163] S. Zhang, M. Hu, and J. Yang. Treepi: A novel graph indexing method. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE)*, pages 966–975, Istanbul, Turkey, 2007. IEEE Computer Society Press.

[164] S. Zhang, J. Li, H. Gao, and Z. Zou. A novel approach for efficient supergraph query processing on graph databases. In *12nd International Conference on Extending Database Technology (EDBT)*, pages 204–215, Saint-Petersburg, Russia, 2009. Springer-Verlag.

[165] S. Zhang, H. Toivonen, W. Wu, J. Li, and H. Gao. Efficient algorithms for supergraph query processing on graph databases. *Journal of Combinatorial Optimization*, online publish, 2009.

[166] X. Zhang, F. Pan, W. Wang, and A. Nobel. Mining non-redundant high order correlations in binary data. In *34th International Conference on Very Large Data Bases (VLDB)*, pages 1178–1188, Auckland, New Zealand, 2008. VLDB Endowment.

[167] P. Zhao, J. X. Yu, and P. S. Yu. Graph indexing: Tree + delta >= graph. In *33th International Conference on Very Large Data Bases (VLDB)*, pages 938–949, University of Vienna, Austria, 2007. VLDB Endowment.

[168] F. Zhu, X. Yan, and J. Han. gprune: A constraint pushing framework for graph pattern mining. In *The Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD*, pages 388–400, Nanjing, China, 2007. Springer-Verlag.

[169] Z. Zou, J. Li, H. Gao, and S. Zhang. Frequent subgraph pattern mining on uncertain graph data. In *Proceeding of the 18th ACM conference on Information and knowledge management (CIKM)*, pages 583–592, Hong Kong, China, 2009. ACM Press.

[170] 唐德权. 基于图模式的数据挖掘算法研究与应用. PhD thesis, 湖南师范大学, 2021.

[171] 邹兆年. , 李建中, 高宏. 从不确定图中挖掘频繁子图模式. 软件学报. 2009, 20(11):2965–2976.