知识图谱划分算法研究综述

王 鑫 陈蔚雪 杨雅君 张小旺 冯志勇

(天津大学智能与计算学部 天津 300354)

(天津市认知计算与应用重点实验室 天津 300354)

摘 要 知识图谱是人工智能的重要基石,因其包含丰富的图结构和属性信息而受到广泛关注.知识图谱可以精确语义描述现实世界中的各种实体及其联系,其中顶点表示实体,边表示实体间的联系.知识图谱划分是大规模知识图谱分布式处理的首要工作,对知识图谱分布式存储、查询、推理和挖掘起基础支撑作用.随着知识图谱数据规模及分布式处理需求的不断增长,如何对其进行划分已成为目前知识图谱研究的热点问题.从知识图谱和图划分的定义出发,系统性地介绍当前知识图谱数据划分的各类算法,包括基本、多级、流式、分布式和其他类型图划分算法.首先,介绍 4 种基本图划分算法:谱划分算法、几何划分算法、分支定界算法、KL 及其衍生算法,这类算法通常用于小规模图数据或作为其他划分算法的一部分;然后,介绍多级图划分算法,这类算法对图粗糙化后进行划分再投射回原始图,根据粗糙化过程分为基于匹配的算法和基于聚合的算法;其次,描述 3 种流式图划分算法,这类算法将顶点或边加载为序列后进行划分,包括 Hash 算法、贪心算法、Fennel 算法,以及这 3 种算法的衍生算法;再次,介绍以 KaPPa、JA-BE-JA 和轻量级重划分为代表的分布式图划分算法及它们的衍生算法;同时,在其他类型图划分算法中,介绍近年来新兴的 2 种图划分算法:标签传播算法和基于查询负载的算法.通过在合成与真实知识图谱数据集上的丰富实验,比较了 5 类知识图谱代表性划分算法在划分效果、查询处理与图数据挖掘方面的性能差异,分析实验结果并推广到推理层面,获得了基于实验的知识图谱划分算法性能评价结论.最后,在对已有方法分析和比较的基础上,总结目前知识图谱数据划分面临的主要挑战,提出相应的研究问题,并展望未来的研究方向.

关键词 知识图谱;图划分;多级划分;流划分;分布式

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2021.00235

Research on Knowledge Graph Partitioning Algorithms: A Survey

WANG Xin CHEN Wei-Xue YANG Ya-Jun ZHANG Xiao-Wang FENG Zhi-Yong

(*College of Intelligence and Computing, Tianjin University, Tianjin* 300354)

(*Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin* 300354)

Abstract Knowledge graphs are important cornerstones of artificial intelligence, which not only have the graph structure of the general graph model but also contain rich attribute information of vertices and edges. Therefore, knowledge graphs have received widespread attention in practical tasks. Knowledge graphs can use accurate semantics to describe the various entities and their connections in the real world, where the vertices represent entities, and the edges represent connections among these entities. Knowledge graph partitioning is a fundamental task for distributed processing of large-scale knowledge graphs, and it is the essential support for a series of graph processing operations including distributed storage, query processing, reasoning, and data mining. With the increasing scale of knowledge graphs and more requirements of distributed processing, it is difficult to partition large-scale knowledge graphs, which has become a hot issue in the current

收稿日期:2019-09-05;在线发布日期:2020-02-05.本课题得到国家重点研发计划项目(2019YFE0198600)、国家自然科学基金(61972275)资助.**王 鑫**,博士,教授,中国计算机学会(CCF)杰出会员,主要研究领域为知识图谱数据管理、图数据库、大规模知识处理. E-mail:wangx@tju.edu.cn. **陈蔚雪**,硕士研究生,主要研究方向为知识图谱数据管理、图数据库、大规模知识处理. **杨雅君**,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为图数据库、图数据挖掘. **张小旺**,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为数据库、人工智能、知识图谱. **冯志勇**,博士,教授,中国计算机学会(CCF)杰出会员,主要研究领域为知识工程、服务计算.

research on knowledge graphs. Starting from the definitions of knowledge graphs and graph partitioning, we systematically introduce various algorithms currently available for knowledge graph partitioning, including basic graph partitioning algorithms, multilevel graph partitioning algorithms, streaming graph partitioning algorithms, distributed graph partitioning algorithms, and other types of graph partitioning algorithms. First, four basic graph partitioning algorithms are reviewed, including spectral partitioning algorithms, geometric partitioning algorithms, branch and bound algorithms, KL and its derivative algorithms. This type of algorithms usually works on small-scale graphs or as subroutines for more sophisticated algorithms. Second, two multilevel graph partitioning algorithms are introduced, one of which uses the matching-based algorithm during the coarsening phase, and the other uses the aggregation-based algorithm. After coarsening, they partition the much smaller graph and then project the result back to the original graph. Third, streaming graph partitioning algorithms are described, which load a graph as a sequence of vertices or edges and assign each element to the corresponding partitions. The streaming graph partitioning algorithms can be further divided into three subcategories: the hash algorithm, the greedy algorithm, the Fennel algorithm, and their derivative algorithms. Fourth, we introduce distributed graph partitioning algorithms for partitioning large-scale graphs in distributed environments. We choose three representative algorithms, namely the KaPPa algorithm, the JA-BE-JA algorithm, the lightweight repartitioning algorithm, and then describe derivative algorithms of these distributed algorithms. Meanwhile, as the other types of graph partitioning algorithms, we present two recent graph partitioning algorithms, one is the label propagation algorithm, and the other is the query workload-based algorithm. By conducting extensive experiments on both synthetic and real knowledge graph datasets, we compare the performance of five representative knowledge graph partitioning algorithms in partitioning effects, query processing, and graph data mining. We analyze the experimental results in detail and draw general conclusions, and extend them to the reasoning tasks. The performance evaluation conclusions of knowledge graph partitioning algorithms are obtained based on these experiments and analyses. Finally, based on the analysis and comparison of existing methods, we summarize the main challenges in the current research on knowledge graph partitioning, propose the corresponding issues that can be investigated, and look forward to the future research directions on this topic.

Keywords knowledge graph; graph partitioning; multilevel partitioning; streaming partitioning; distribution

1 引 言

近年来,知识图谱得到了越来越多的关注和发展.知识图谱[1]最早由 Google 在 2012 年提出,现如今规模已经达到百万顶点(10^6)和上亿条边(10^8),常用的知识图谱有 DBpedia[2]、YAGO[3]、Wikidata[4]、Freebase[5]以及中文百科知识图谱 CN-DBpedia[6]、Zhishi.me[7]等.知识图谱数据模型基于图结构,用顶点表示实体,用边表示实体间的联系.由于同时包含了图的结构信息和属性信息,知识图谱能够更好地表示现实生活中的复杂关系.

知识图谱不仅需要根据实际情况随时更新数据,还要进行存储、查询、推理、挖掘等一系列操作.如今知识图谱的数据规模正在不断增大,链接开放数据(Linked Open Data)2019 年 3 月发布的 LOD 云图中有很多知识图谱数据集规模超过 10 亿条三元组[8].以维基百科知识图谱 DBpedia 为例,它所包含的三元组数据已经超过了 30 亿条.由于知识图谱数据的急剧增长,传统的集中式数据处理已经不能满足当前需求,必须通过分布式集群来存储和处理大规模知识图谱数据.知识图谱数据的分布式存储面临的第一个问题即是知识图谱划分.这就使得如何划分大规模知识图谱,同时满足跨分区边数目最小化且有效提高知识图谱查询处理性能成为迫切需要深入研究的课题.

知识图谱划分作为大规模知识图谱分布式处理的首要工作，随着知识图谱数据规模及分布式处理需求的增长面临着新的挑战.一方面，现有的知识图谱划分方法在大图上的划分效果并不理想，时间复杂度较高，特别是对于包含数据量大且关系复杂的知识图谱.产业界对于知识图谱划分通常使用分布式处理框架，需要在集群上找到最佳的数据分布，然而高质量的划分必须同时满足负载平衡（在机器之间均匀地分配数据）和最小化切割边或顶点的数量（其顶点或边属于不同的分区），这增加了机器之间的通信成本.另一方面，目前主要利用现有的图划分算法对知识图谱进行划分，然而这些算法忽略了知识图谱的属性信息，如何充分利用知识图谱数据进行划分、如何制定统一的划分标准以及如何将划分与后续的查询处理相结合是当前研究的难点问题.

知识图谱划分有着十分重要的现实意义，例如复杂网络[9]、图像分割[10]、超大规模集成电路设计[11]等.在复杂网络方面，最典型的就是社交网络分析[12]，每一个用户都是一个顶点，可以根据用户个人信息以及用户之间的联系对其进行分类，更好地对具有相同属性的用户进行分析.这同样适用于道路交通网络[13]和生物网络[14]，道路交通网络需要构建图来分析车辆轨迹和人的行动路线，生物网络则通过图来表示蛋白质间的相互作用以及基因序列.图像分割[15]是图像处理的重要内容，它需要将像素分配到相应对象中，在图划分中，每个顶点代表一个像素，图像分割即对图中顶点进行划分.超大规模集成电路（VLSI）设计是另一种常见的图划分应用技术，划分的目的是为了降低 VLSI 设计的复杂性.

图划分是一个经典的 NP 难问题，没有恒定因子的近似算法[16].目前已有许多关于图划分方法的研究，但是现有的图划分算法综述都只基于图论中的图 $G=(V,E)$，并不包括知识图谱的相关内容，知识图谱划分算法的综述尚未发现.鉴于此，本文主要对现有知识图谱划分算法研究进行较为全面地综述，整体框架如图 1 所示.

图 1　知识图谱划分算法框架

图 2 按照时间顺序列出了本文所综述的主要算法.从图中可以看出，KL 算法是最早出现的图划分

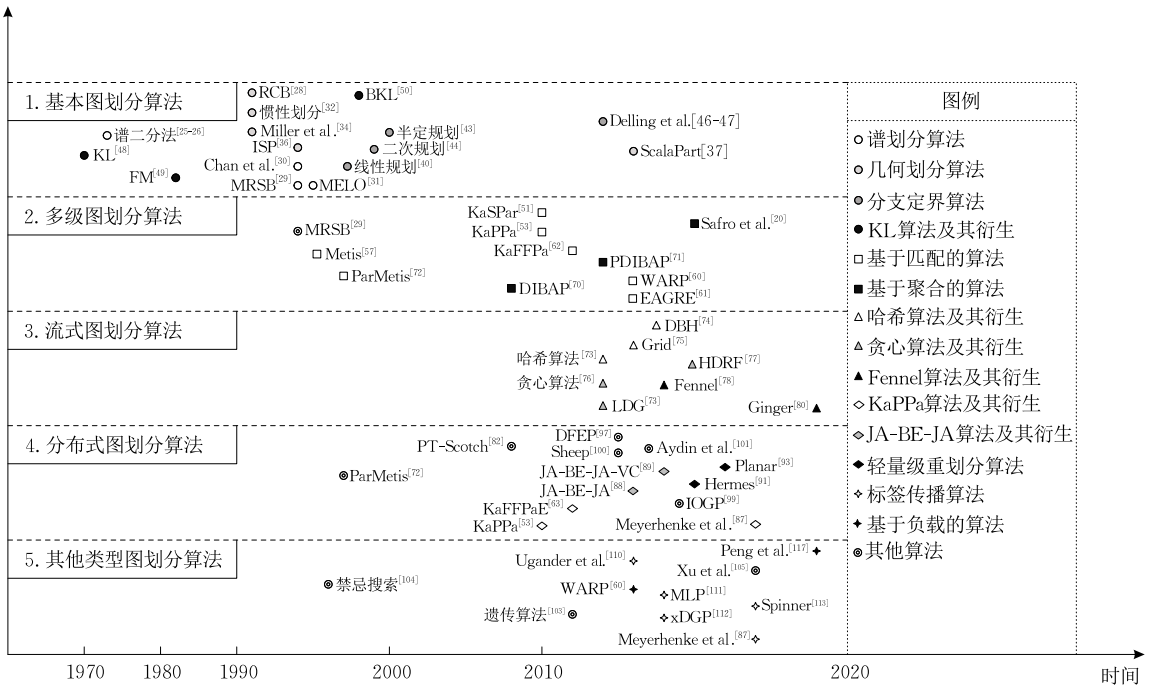


图 2　知识图谱划分算法时间轴

算法，随后出现的是谱划分算法，之后很长一段时间内都是这两类算法，直到 20 世纪 90 年代图划分算法的种类才逐渐增多，有了几何划分算法和分支定界算法. 早期算法主要在单机环境下运行，随着图数据规模的增大，多级图划分算法逐渐兴起，其将原始图粗糙化为较小图进行划分，再将结果投射回原始图. 之后随着大规模知识图谱的出现，多级算法也不能处理现有的大图，人们开始思考如何将现有算法置于分布式环境下运行，于是有了分布式图划分算法. 并且将动态图作为划分的输入图，以适应现实中不断更新的数据. 近几年提出的流式图划分算法将顶点或边转化成流输入，是另一种划分大规模图的思路. 当前知识图谱得到广泛关注，一些专门针对 RDF 图的划分算法开始出现，例如基于查询负载的算法.

相关工作. 文献[17]对近年来出现的图划分算法进行了综述，但是侧重于文献列举，算法介绍较为简单，本文不仅详细描述了各种算法，还比文献[17]多了流式图划分算法、标签传播算法和基于查询负载的算法等内容. 文献[18]综述了以顶点为中心的分布式图计算框架，关于图划分一节的介绍较为简单，包含的算法类型并不全面. 文献[19]着重分析了分布式框架下的图划分算法，更侧重于不同框架的性能比较. 文献[20]对已有多级图划分算法进行了综述. 文献[21-22]从顶点划分和边划分两个方面综述了流式图划分算法. 文献[23]综述了云计算环境下进行大规模图数据处理的关键问题，从单指标和多指标两个方面分析了部分图划分算法. 本文详细介绍 5 种类型的知识图谱划分算法，比上述文献更加综合全面.

2 预备知识

知识图谱目前并没有一个统一的严格定义. 知识图谱是图 $G=(V,E)$ 的某种扩展形式，其中 V 是顶点集合，表示实体，E 是边集合，表示实体间的联系. 实际上，知识图谱是图数据模型的继承和发展，其在一般图模型的顶点和边上附加更多的属性信息，用于描述现实世界中事物的广泛联系. 知识图谱主要有以下 4 种类型，分别是 RDF 图、属性图、异构信息网络和有向标签图. 有关知识图谱数据管理的详细内容可参见文献[24].

RDF 图定义为 RDF 三元组的有限集合. RDF 是万维网联盟制定的一种表示网络信息的标准，通过资源、属性和值来描述特定信息. RDF 采用三元组 (s,p,o) 的形式，其中 s,p,o 分别代表语义数据中的主语、谓语和宾语. RDF 图的顶点集合是三元组中主语和宾语的集合，边集合是谓语的集合. RDF 图的形式化定义如下.

定义 1. RDF 图. 设 U、B 和 L 为互不相交的无限集合，分别代表 URI、空顶点(blank node)和字面量(literal). 一个三元组 $(s,p,o)\in(U\cup B)\times U\times(U\cup B\cup L)$ 称为 RDF 三元组，其中 s 是主语，p 是谓语，o 是宾语. RDF 图 G 是三元组 (s,p,o) 的有限集合.

属性图是另一种常用的知识图谱数据模型. 与 RDF 图相比，属性图多了一个键值对来表示顶点或边的信息. 属性图的形式化定义如下.

定义 2. 属性图. 属性图 G 是四元组 (V,E,λ,δ). 其中：

(1) V 是顶点的集合；

(2) $E\subseteq V\times V$ 是有向边的集合，如 $e=(v_1,v_2)$ 表示 e 是从 v_1 到 v_2 的有向边；

(3) 设 Lab 是标签集合，函数 $\lambda:(V\cup E)\rightarrow Lab$ 为顶点或边赋予标签，如 $l\in Lab$，$\lambda(v)=l$ 表示顶点 v 的标签是 l；

(4) 设 $Prop$ 是属性集合，Val 是值集合，函数 $\delta:(V\cup E)\times Prop\rightarrow Val$ 为顶点或边关联属性，如 $p\in Prop$，$a\in Val$，$\delta(v,p)=a$ 表示顶点 v 的属性 p 的值是 a.

异构信息网络源自信息网络. 信息网络是指带有对象或链接类型映射的有向图，如果信息网络中对象或链接的类型总数大于 1，则称为异构信息网络. 其形式化定义如下.

定义 3. 异构信息网络. 异构信息网络 G 由三元组 (V,E,λ) 构成. 其中

(1) V 是顶点的集合；

(2) $E\subseteq V\times V$ 是有向边的集合；

(3) 设 $Type$ 是类型集合，函数 $\lambda:(V\cup E)\rightarrow Type$ 为顶点或边赋予类型，如 $l\in Type$，$\lambda(v)=l$ 表示顶点 v 的所属类型是 l；

有向标签图是在有向图 $G=(V,E)$ 的基础上为每个顶点添加了标签. 其形式化定义如下.

定义 4. 有向标签图. 有向标签图 G 由三元组 (V,E,L) 构成. 其中

(1) V 是顶点的集合；

(2) $E\subseteq V\times V$ 是有向边的集合；

(3) L 是顶点和边上标签的集合.

知识图谱泛化、统一了各种图模型结构，知识图谱与各种图模型的关系如图 3 所示. 可以看出，知识图谱扩展了一般图模型，通常意义上的图只包含顶点和边，而知识图谱增加了顶点和边的属性信息，不但能够划分和存储，还能进行查询、推理和挖掘等操作. 有向标签图是最简单的知识图谱模型，其在一般图模型的基础上增加了标签集合，图中的任意顶点或边至少关联一个标签. RDF 图可以看作是特殊的有向标签图，其特殊之处在于一个三元组中的谓语也可作为另一个三元组的主语或宾语，反映在有向标签图中，即边亦可作为顶点，顶点与边交集非空. 属性图在一般图模型的基础上进行扩展，其增加了顶点属性和边属性的内置支持，目前被图数据库业界广泛采用. 异构信息网络和有向标签图相似，不同的是将标签换成了所属类型，且每个顶点或边可以有不止一个所属类型.

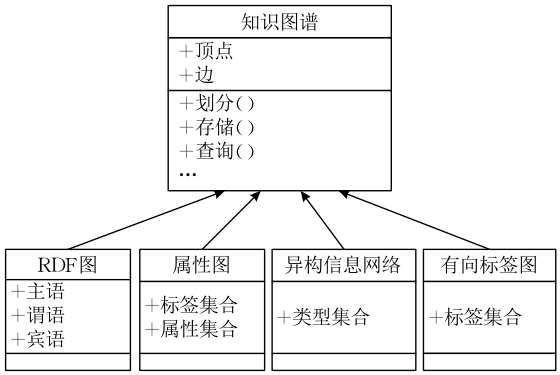


图 3　知识图谱的 4 种类型

知识图谱与一般图模型相比含有更加丰富的信息，通常在一般图模型上的各种图划分算法也适用于知识图谱. 为简便起见，如无特别说明，本文介绍的图划分算法均普遍适用于各类知识图谱的划分. 限于篇幅，这里以 RDF 图作为代表，给出一个知识图谱示例，如图 4 所示.

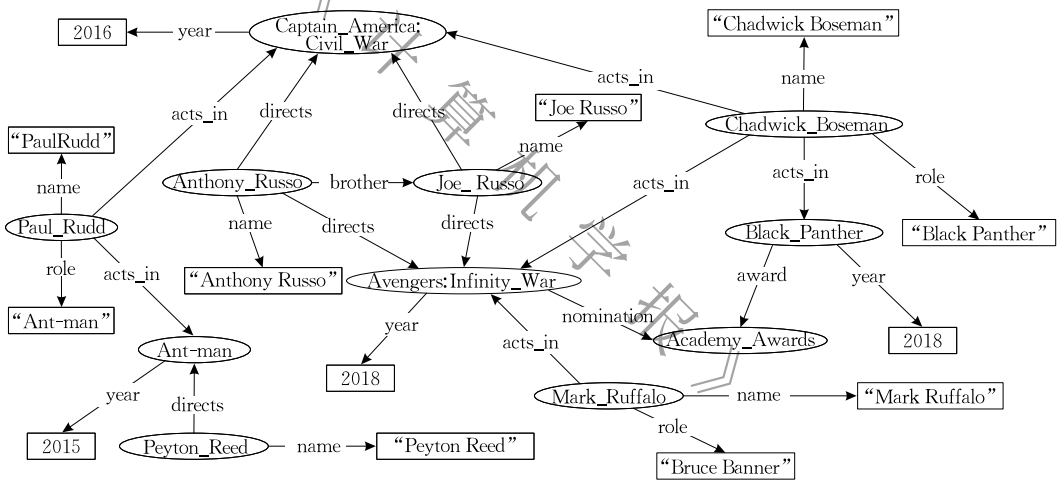


图 4　知识图谱示例：RDF 图

定义 5.　图划分. 给定图 $G=(V,E)$，图划分是将图 G 划分成 k 部分，每一部分用 P_i 表示，其中 $i\in[1,k]$. 对于任意 $i\neq j$，有 $P_i\cap P_j=\varnothing$ 且 $\bigcup_{i=1}^{k}P_i=G$. 图划分需要满足式(1)或式(2)以实现负载平衡，同时最小化跨越分区的顶点或边，这两项也是判断划分质量好坏的标准.

$$n(1-\theta)/k\leq|V_i|\leq n(1+\theta)/k \tag{1}$$

$$m(1-\theta)/k\leq|E_i|\leq m(1+\theta)/k \tag{2}$$

其中，$\theta\in[0,1]$是浮动因子，m 和 n 分别代表图的顶点数和边数，$|V_i|$和$|E_i|$分别是第 i 个分区的顶点数和边数. 图划分问题已经被证明是 NP 难问题[16].

当输入图随时间变化时(例如增加或删除一些顶点)，划分可能会失去平衡. 因此对于动态图，需要在划分前设定一个阈值，一旦失衡超过该阈值，则会重新进行划分.

在图划分算法中，有两种切分方法，一种是对顶点进行划分，另一种是对边进行划分. 图 5 给出了两种划分方法的示例.

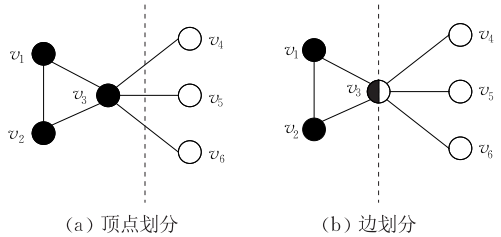


图 5　两种划分方法的示例((a)顶点划分将顶点划分至不同分区，造成边切割；(b)边划分将边划分至不同分区，造成顶点切割)

在顶点划分中，可能存在跨越分区的边（如图 5(a)中的边(v_3,v_4)的两个顶点分别属于不同分区），更多的跨越边意味着更大的通信开销，因此顶点划分的主要优化目标是减少边切割. 同样地，边划分的主要优化目标是减少顶点切割. 由于已有的图划分算法大部分是对图的顶点进行划分，在本文中如无特别说明图划分算法均采用顶点划分.

表 1 给出了图划分算法中的一些常用符号.

表 1 常用符号

符号	描述
G /G_i	输入图/G 的第 i 个子集
V /V_i	G 中顶点的集合/V 的第 i 个子集
E /E_i	G 中边的集合/E 的第 i 个子集
v	G 的一个顶点
(v_i,v_j)	v_i 和 v_j 之间的边
n	G 的顶点数量
m	G 的边数量
k	G 的分区数
P/P_i	G 的分区集合/G 的第 i 个分区
$P(v)$	v 所在的分区
$N(v)$	v 的相邻顶点集合
$deg(v)$	v 的度数
$c(v)$	v 的权重

3 基本图划分算法

本节主要包括早期出现的较为经典的图划分算法. 由于条件限制，这些算法通常作用于小规模图，或者作为子程序应用于更复杂的图划分方法. 这些算法大部分仅用于图的二分，但可以推广到 k 个分区（例如通过递归过程）.

3.1 谱划分算法

谱二分法是一种常用的图划分算法，最早由 Donath 等人[25]和 Fiedler[26-27]提出，它通过计算图对应的 Laplace 矩阵 $\boldsymbol{L}$ 的第二小特征值的特征向量 $\boldsymbol{y}$，即 Fiedler 向量来确定划分标准. 其中 $\boldsymbol{L}=\boldsymbol{D}-\boldsymbol{A}$，$\boldsymbol{D}$ 是图的度矩阵，$\boldsymbol{A}$ 是图的邻接矩阵. $\boldsymbol{L}=(l_{i,j})_{n\times n}$ 的定义如下：

$$l_{i,j}=\begin{cases}-1, & \text{如果 } e_{i,j}\in E\\ deg(v_i), & \text{如果 } i=j\\ 0, & \text{否则}\end{cases} \tag{3}$$

通过确定 $\boldsymbol{y}$ 的中位数 $\bar{m}$ 并且将小于或等于 $\bar{m}$ 的顶点分配给 P_1，其余顶点分配给 P_2 完成图划分. 当分区数大于 2 时，可递归地使用谱二分法（RSB）[28]，直至达到划分目标.

然而，RSB 的运行时间较长，造成较高的成本. 文献[29]使用多级方法来快速获得 Fiedler 向量，算法结构类似于第 4 节中介绍的多级图划分算法，它先构造一系列较小的图，在粗糙化图上计算 Fiedler 向量，再将向量逐步插值到更高级别的图中，其间使用 Rayleigh 商迭代进行局部改进. 文献[30-31]通过使用 Laplace 矩阵的多个特征向量将图直接划分成多个部分，进一步提高了效率.

3.2 几何划分算法

几何划分算法利用顶点的坐标信息对图进行划分. 坐标二分法[28]是其中最简单的算法，它基于以下假设：除了顶点集合 $V=(v_1,v_2,\cdots,v_n)$ 之外，还存在可用于顶点的二维或三维坐标. 对于任意 $v_i\in V$，存在二维坐标 $v_i=(x_i,y_i)$ 或三维坐标 $v_i=(x_i,y_i,z_i)$. 然后选择任意坐标方向（例如 y 方向）作为基准，根据顶点的 y 坐标对所有顶点进行排序. 将 y 坐标较小的一半顶点分配给 P_1，其他顶点分配给 P_2，再使用递归坐标二分法（RCB）即可对图进行多路划分.

坐标二分法的一个缺点是太依赖坐标，导致同一图在不同的坐标系中有着完全不同的划分结果. 惯性划分[32-33]的出现解决了这个问题，它可以看作是坐标二分法的改进，但它没有利用坐标轴，而是通过选择划分的基准线（或平面）L，使得所有顶点到 L 的平方和最小化. 如图 6 所示.

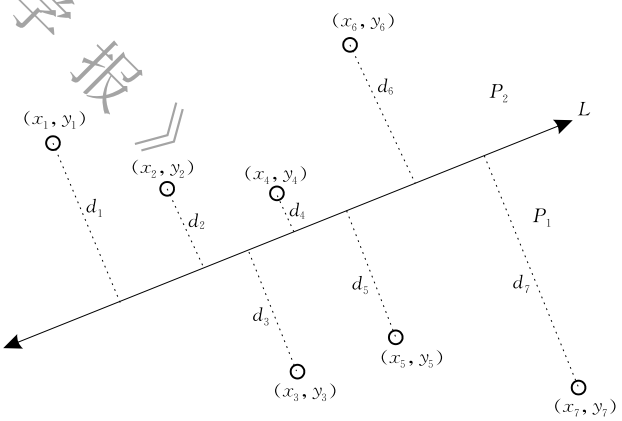


图 6 惯性二分法

文献[34-35]提出了 k-重叠图的概念，其特殊类型包括平面图、k-近邻图和有限元相关图，划分时需将 d 维顶点随机投影至 $d+1$ 维球体，由通过中心点的平面进行二等分.

此外，还有将 d 维顶点降低至一维空间进行划分的空间填充曲线法[36-37]，以及使用图嵌入将坐标赋予图进行划分的方法[38].

3.3 分支定界算法

NP 难问题通常使用分支定界[39]算法解决，这对图划分同样适用. 分支定界通过将原始问题分成

两个或多个子问题，递归地进行求解，从而找到最佳解决方案. 分支定界树的每个节点对应不同的子问题，算法在原始问题的解决方案基础上保持上界 U，U 可以在算法改进时更新. 要处理树中的节点，需要在相应子问题的解决方案上计算下界 L. 如果 $L\geq U$ 则剪掉节点，否则继续分支，创建两个或更多的子问题. 整个过程如图 7 所示.

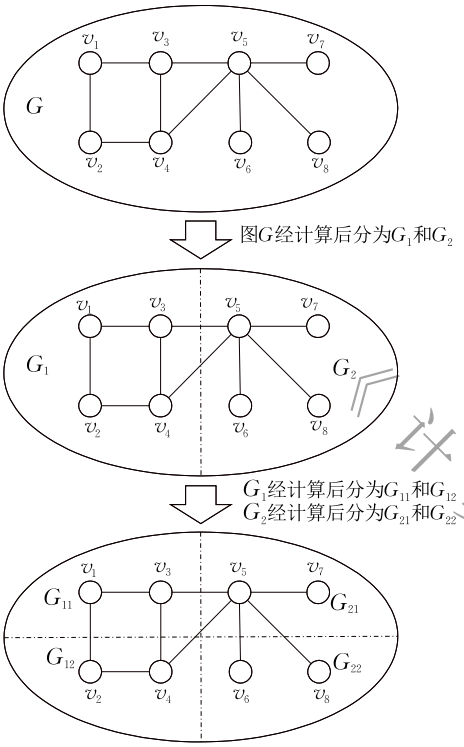


图 7　分支定界图划分示例

在图划分问题中，获得上界和下界的方式有很多，例如利用线性规划[40-42]、半定规划[41-43]、二次规划[44-45]等. 文献[46-47]采用组合的方法，可以划分更大规模的图，首先假设 A 和 B 是 V 的两个不相交子集，其他不在 A 或 B 中的顶点属于未分配状态，对于 A 可以扩展成的任意 A^+，计算 B 和 A^+ 之间的最小边切割，从而进行划分. 该方法可以划分具有上百万个顶点的图，但其在大规模图上的运行时间较长.

3.4　KL 算法及其衍生

上述算法都是直接作用于原始图进行划分，除此之外还有基于已划分图的改进算法，Kernighan 和 Lin 提出的 KL 算法[48]通常被认为是第一个此类图划分算法. KL 算法从任意两个分区开始，通过交换顶点或顶点集来创建新的划分方案，从而改进初始划分方案. 对新的划分方案重复该过程，直到不能改进为止.

假设 A 和 B 是 V 的两个初始分区，顶点 v 的增益 g_v 是将 v 从原有分区移至新分区后所减少的切割边数，这里的 g_v 可以是负数. 交换顶点 $a\in A$ 和 $b\in B$ 的增益 $g(a,b)$ 为 $g_a+g_b-2\delta(a,b)$，其中 $\delta(a,b)$ 用来判断顶点 a 与 b 之间是否存在边，定义如下：

$$\delta(a,b)=\begin{cases}1, & \text{如果 } (a,b)\in E\\ 0, & \text{否则}\end{cases} \tag{4}$$

选择使 g_v 最大化的顶点对 (a,b)，一旦选择 a 和 b，之后交换不再考虑 a 或 b. 依次选择顶点对 $(a_1,b_1),(a_2,b_2),\cdots,(a_{|v|/2-1},b_{|v|/2-1})$，找出能使 $\sum_{i=1}^{l}g(a_i,b_i)$ 最大化的 l，交换顶点集 $\{a_1,a_2,\cdots,a_l\}$ 和 $\{b_1,b_2,\cdots,b_l\}$. KL 算法重复以上过程，直到没有改进为止. 图 8 是顶点交换的一个例子.

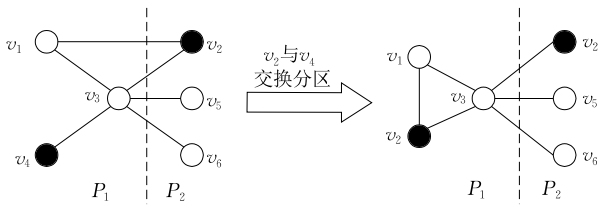


图 8　顶点交换示例(顶点 v_2 和顶点 v_4 交换后，跨分区边数由 4 减少为 3)

KL 算法的一个重大改进是 Fiduccia-Mattheyses (FM)算法[49]. KL 和 FM 之间的主要区别在于顶点移动的单位. KL 一次交换一对顶点，而 FM 一次只移动一个顶点. 这使得 FM 比 KL 效率更高且没有划分质量的损失.

KL 算法通常用于其他图划分算法的细化，特别是在多级图划分算法的反粗糙化阶段. 文献[50]在 KL 算法的基础上提出 KL(1)算法和边界 KL 算法(BKL)，KL(1)算法只运行一次 KL 算法，不再进行迭代过程；BKL 对跨越分区的边所连接的顶点集使用 KL 算法，在保证质量的同时减少了划分时间. KaSPar[51]考虑了高度局部化的 FM 算法，仅从处于边界的未进行粗糙化的顶点开始搜索，进一步提高了划分效率.

表 2 是对基本图划分算法的一个小结. 谱划分算法和分支定界算法虽然时间复杂度较高，但是有着良好的划分质量，多适用于在单机上划分规模较小的图. 相反，几何划分算法耗时较少，划分质量要差一些，但可以扩展到分布式环境下，从而划分大规模图数据. KL 算法及其改进算法通常与其他算法尤其是第 4 节将介绍的多级图划分算法相结合使用，作为反粗糙化阶段的细化算法.

表 2 基本图划分算法小结

类型	优点	缺点	算法	描述
谱划分算法	引入 Laplace 矩阵的特征向量,在小规模图上划分质量较高	时间复杂度较高,不适用于大规模图的划分	谱二分法[25-27]	利用 Laplace 矩阵的第二小特征值的特征向量(即 Fiedler 向量)将图划分为两部分
			RSB[28]	递归使用谱二分法,直至达成目标
			MRSB[29]	在粗糙化图上计算 Fiedler 向量,再将其逐步插值回原始图
			MELO[31]	使用 Laplace 矩阵的多个特征向量直接将图划分成多个部分
几何划分算法	能够充分利用顶点的坐标信息,计算简单,可用于大规模图	只考虑坐标而不是顶点间的联系,导致跨越分区的边数较多	RCB[28]	递归地根据顶点某一方向的坐标将顶点平均分成两部分
			惯性划分[32-33]	使所有顶点到基准线(或平面)的平方和最小化
			Miller 等人[34-35]	将 d 维顶点投影至 $d+1$ 维球体,由通过中心点的平面进行划分
			ISP[36]	将 d 维顶点降低至一维空间,利用 Hilbert 曲线进行划分
			ScalaPart[37]	使用图嵌入将坐标赋予图,再进行划分
分支定界算法	利用已有算法计算划分的上界和下界,在小规模图上取得较好的划分质量	计算较为复杂,不适用于大规模图的划分	线性规划[40-42]	利用线性规划算法确定划分边界
			半定规划[41-43]	利用半定规划算法确定划分边界
			二次规划[44-45]	利用二次规划算法确定划分边界
			Delling 等人[46-47]	利用组合算法确定划分边界
KL 算法及其衍生算法	计算较为简单,可以只对图的局部顶点做交换而不用搜索整个图	划分时间随迭代次数成正比	KL[48]	在已有划分方案基础上交换不同分区的顶点或顶点集
			FM[49]	在已有划分方案基础上移动顶点到其他分区
			BKL[50]	只对划分边界的顶点进行交换

对于知识图谱而言,使用基本图划分算法时,需要忽略顶点和边的属性信息,将知识图谱当作无向图进行划分.由于基本图划分算法是在单机环境下进行的,通常只能划分具有上万个顶点和边的知识图谱,不符合现实需要,一般情况下不单独使用.但是这些算法的基本思想可以被利用并加以改进,例如在多级图划分算法中将图粗糙化后的划分阶段,此时图的规模已足够小,可直接使用基本图划分算法;另一种情况是将这些算法扩展到分布式环境下从而进行大规模知识图谱的划分.

4 多级图划分算法

随着图数据规模的增长,基本图划分算法已经不能满足实际需要,于是产生了多级图划分算法[52].这类算法首先将图 G 粗糙化为只包含几百个顶点的摘要图,在这个小得多的图上进行划分,然后将划分结果投影回原始图.整个过程分为三个阶段:粗糙化、初始划分和反粗糙化,如图 9 所示.

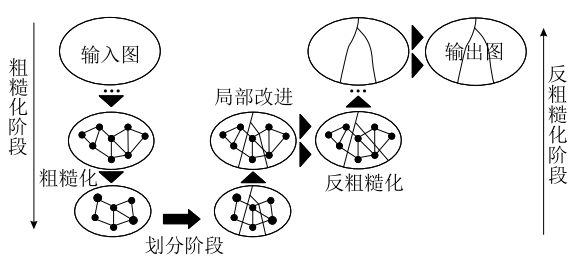


图 9 多级图划分算法示意

粗糙化的主要目的是生成具有较少自由度的输入图,这是通过构造一系列具有更小尺寸的粗糙化图来实现的.这一阶段通常需要收缩顶点集,这意味着用粗糙化图中的单个顶点 u 代替原始图中的部分顶点,假设 L 是 V 中部分顶点的集合,那么 $c(u)=\sum_{v\in L}c(v)$.收缩时可能会产生平行边,类似地,用单个边代替这些边,其权重等于平行边的权重之和.这样使得粗糙化图中的划分结果能够反映原始图中的划分结果.

当图已经粗糙化至足够小时(即可以使用基本图划分算法进行处理时),停止粗糙化.本文中提到的任何基本图划分算法,只要可以处理顶点和边的权重,皆可用于初始划分.一些代价较高的划分方法在粗糙化图上拥有较好的划分效果,但是不意味着在原始图上拥有同样的效果.有些多级划分算法采用多种简单方法的组合进行划分,也可以获得很好的划分效果[17].

反粗糙化即将粗糙化图的划分逐步映射至原始图的划分.由于原始图比粗糙化图具有更多的自由度,粗糙化图的最佳划分不一定是原始图的最佳划分.因此,在反粗糙化阶段需要利用一些算法进行细化(如 KL 算法),反粗糙化和细化同时进行,直至划分结果没有太大变化.

多级图划分算法的难点在于构造粗糙化图,划分和细化算法可参见本文其他小节,本节重点介绍粗糙化的两种类型以及对应的划分算法,分别是基

于匹配的算法和基于聚合的算法.

4.1　基于匹配的算法

最常见的粗糙化方法是通过最大匹配得到的.给定图 G,匹配 $M\subseteq E$ 是一组边集,集合中任意两条边不存在公共顶点,最大匹配是指包含最多数量的边的匹配.在粗糙化阶段,匹配中每条边连接的顶点都用单个顶点代替.图 10 是匹配的一个例子,其中 $M=\{(v_1,v_2),(v_3,v_4)\}$.

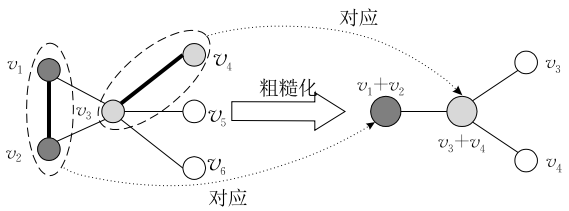


图 10　粗糙化阶段的匹配示例

一个好的匹配应该包含尽可能多的权重高的边,但同时也希望顶点权重保持平衡[53].针对这个问题,文献[53]使用评级函数,对边的权重和均匀性之间的进行权衡.假设 $w(e)$ 代表边 e 的权重,$c(v)$ 代表顶点 v 的权重,文献[53]经实验验证,最优评级函数为

$$f(u,v)=\frac{w(u,v)}{c(u)c(v)} \tag{5}$$

使用深度优先算法或随机算法可以快速获得最大匹配,文献[50]进一步提出了重边匹配、轻边匹配和重团匹配等启发式算法.其中重边匹配在文献[50]中被证明划分效果最好,它以随机顺序访问顶点,优先选择连接顶点的权重最高的未匹配边添加到匹配.

有一些接近线性时间的算法需要较大开销,但是可以获得更好的质量保证,例如贪心算法[54]、路径增长算法(PGA)[55]以及结合贪心算法和 PGA 的全局路径算法(GPA)[56].贪心算法的思想非常简单,即重复地将当前图 G 中未匹配的权重最高的边添加到匹配中,直到没有可选择的边.PGA 需要构建一组顶点不相交路径.每条路径从任意顶点开始,将与当前顶点 v 相邻的未粗糙化顶点 u 的权重最大的边 $e=(v,u)$ 添加到路径中,并删除与 v 相邻的所有边.如果当前路径无法扩展,PGA 将开始构建新的路径,直至没有剩余边.在增长路径的同时,选择成为路径的边将交替地添加到匹配 M_1 和 M_2 中,最终返回二者中的较大值.GPA 在上述两种算法的基础上,按照权重降低的顺序扫描边,构建路径的集合 E'.然后使用动态规划算法为这些路径计算最大权重匹配.在第一轮 GPA 完成后,可在剩余边上运行第二轮 GPA,多数情况下在第二轮过后已经没有剩余边[56].

METIS[57-58]是最常见的多级图划分算法,它在粗糙化阶段选择重边匹配,在划分和细化时使用 KL 算法.文献[59]首次将 METIS 作用于 RDF 图.WARP[60]进一步扩展了这种方法,它将图划分与跨分区的三元组查询负载感知复制相结合,并通过匹配查询模式扩展片段,实现高效的查询处理.EAGRE[61]通过在 RDF 数据中使用实体概念来粗糙化 RDF 图,通过对同一类的实体进行分组,可以减少 RDF 的查询时间.

KaPPa[53]是一种基于 GPA 匹配的分布式多级图划分算法,其分区数量需要等于所使用的处理器的数量.KaPPa 以及其改进算法 KaFFPa[62]和 KaFFPaE[63]将在 6.1 节详细介绍.此外,KaSPar[51]是一个基于极限思想的图划分算法,它仅收缩两个级别之间的单条边,使得不同级别的图只有少量变化,从而保证划分质量.

4.2　基于聚合的算法

除了基于匹配的算法,另一种常见的粗糙化方法是基于权重聚合的方法[64-66].这种方法源自代数多重网格法(AMG)[67].在权重聚合中,每个顶点可以分属不同的聚合.粗糙化阶段包含三个步骤:首先选择图的部分顶点作为聚合的种子;然后确定计算规则,可得到属于每个聚合的非种子顶点的权重;最后计算粗糙化顶点之间的连接强度.

权重聚合首先选择一组顶点 $C\subset V$,使得 $F=V/C$ 中的所有其他顶点强耦合到 C.从 $C=\varnothing$ 和 $F=V$ 开始,将顶点从 F 转移到 C 直至剩余 $i\in F$ 满足

$$\sum_{j\in C}w_{ij}\Big/\sum_{j\in V}w_{ij}\leq\Theta \tag{6}$$

Θ 是聚合参数(通常 $\Theta\approx 0.5$).

对于 $i\in F$,定义 C 中 i 的粗糙化邻域 N_i.设 $I(j)$ 是图中的种子 j 的相邻顶点在粗糙化图中的聚合序号.经典的 AMG 矩阵 $\boldsymbol{P}$(大小为 $|V|\times|C|$)的定义如下:

$$P_{iI(j)}=\begin{cases}w_{ij}\Big/\sum\limits_{k\in N_i}w_{ik}, & \text{如果 } i\in F,j\in N_i\\ 1, & \text{如果 } i\in C,j=i\\ 0, & \text{否则}\end{cases} \tag{7}$$

$P_{iI(j)}$ 代表 i 属于第 $I(j)$ 聚合的可能性,第 p 个聚合的大小为 $\sum\limits_j v_jP_{jp}$,连接两个聚合 p 和 q 的边的权重为 $w_{pq}=\sum\limits_{k\neq l}P_{kp}w_{kl}P_{lq}$.

加权聚合可能导致更密集的粗糙化图,所以需要谨慎选择 AMG 方法,同时保证负载均衡.文献[20]使用代数距离[68]作为顶点之间连通性的度量,以

获得高质量且负载均衡的划分结果. 文献[69]提出了一个粗糙化框架,其目标是保留图的谱特征,使用随机游走推导粗糙化的特征向量,但是代价非常高. DIBAP[70]和 PDIBAP[71]在细化阶段使用基于扩散的算法,能够提升划分质量并且更好地并行化.

表 3 给出了多级图划分算法的小结. 基于匹配的算法因其时间复杂度较小更为常见,其中 METIS 以提出时间早、速度快、质量高等原因通常作为其他算法的比较标准. 然而,METIS 无法直接应用到大规模图数据上,后续算法主要围绕如何实现更大规模图数据的划分以及如何在分布式条件下进行划分这两个方向发展. 目前产业界常用的知识图谱(如 DBpedia)在划分时都采用 METIS 算法的主要思想,常见做法是将 METIS 算法集成至分布式系统中以便在不同机器上存储大规模数据.

表 3 多级图划分算法小结

类型	优点	缺点	算法	描述
基于匹配的算法	每一级别的顶点只能粗糙化到下一级别的一个顶点上,计算较为简单	时间复杂度与粗糙化的级别有关和图的规模有关,单机情况下只能划分小规模图	METIS[57-58]	粗糙化阶段使用匹配算法收缩边,细化阶段使用 KL 改进算法
			WARP[60]	将 METIS 算法应用于 RDF 图,并通过匹配查询模式进行扩展
			KaPPa[53]	粗糙化阶段使用 GPA 算法计算匹配,并且使用 FM 算法进行细化
			KaFFPa[62]	在 KaPPa 的基础上,在粗糙化和反粗糙化阶段进行迭代
			KaSPar[51]	粗糙化阶段每级仅收缩单条边,细化阶段使用 FM 算法
基于聚合的算法	引入 AMG 算法计算顶点分数,划分质量较高	计算较为复杂,单个顶点可粗糙化至下一级的多个顶点,易导致更密集的粗糙化图	Safro 等人[20]	粗糙化阶段使用 AMG 算法,并以代数距离作为顶点连通性的度量
			DIBAP[70]	粗糙化阶段使用 AMG 算法,细化阶段使用基于扩散的算法进行改进

同基本图划分算法相似,多级图划分算法不考虑知识图谱的属性信息,将知识图谱当作无向图粗糙化后进行划分. 不同之处在于多级图划分算法可划分包含数十万个顶点和上百万条边的知识图谱,适用范围要比基本图划分算法更广. 但它仍属于单机环境下的划分算法,需要与第 6 节的分布式图划分算法结合才能划分大规模知识图谱,其中的典型算法是 METIS 的并行版本 ParMETIS[72].

5 流式图划分算法

流式图划分算法[73]是近年来提出的一种图划分方法,流式图划分的目的是为了对大规模流式图数据进行高效率划分. 流式图划分算法将图加载为包含顶点或边的序列,再将每个元素分配到对应的分区,整个过程如图 11 所示.

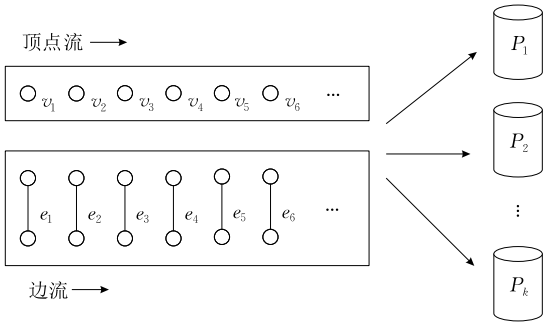


图 11 流式图划分算法流程示意图

5.1 Hash 算法及其衍生

在流式图划分算法中,最简单的是 Hash 算法,它通过构建基于散列的映射函数 $h:\mathbb{N}\to\mathbb{N}$ 进行划分,函数 h 的输入可以是顶点或边的编号. 以边为例,边 $e=(v_i,v_j)$ 的分区为 $P(e)=h(e)\bmod(k)$. 这种启发式算法通常会造成大量的顶点切割,虽然时间开销较少但往往不能获得较高的图划分质量. DBH[74]是对 Hash 算法的一种改进,它是一种边划分算法,主要思想是优先切割更高度数的顶点. 在 DBH 算法中,$h(e)$的定义如下:

$$h(e)=\begin{cases}h(v_i), & \text{如果 } deg(v_i)<deg(v_j)\\ h(v_j), & \text{否则}\end{cases} \tag{8}$$

Gird[75]同样是边划分算法,在 Hash 算法的基础上,增加了对分区的约束:

(1) $P(v_i)\bigcap P(v_j)\neq\varnothing$;

(2) $P(v_i)\not\subset P(v_j)$且 $P(v_j)\not\subset P(v_i)$;

(3) $|P(v_i)|=|P(v_j)|$.

Grid 通过构造矩阵 $X\times Y$ 使得 $XY=k$,其中 k 是分区数,每一个元素代表一个分区,再将顶点 v 通过函数 h 映射至矩阵中对应的分区编号,计算与 v 所属分区同在一行或一列的分区编号集合,这种方式保证了每对顶点计算后的交集都有两个分区,之后再选择负载较小的分区作为该对顶点对应边的最终分区. Gird 需要随时维护分区分配以计算负载最小的分区.

5.2　贪心算法及其衍生

贪心算法[76]是一种基于规则的流式图划分算法，属于边划分算法，对于给定边 $e\in E$ 的顶点 $v_i,v_j\in V$，它遵循的规则如下：

(1) 如果 $P(v_i)\bigcap P(v_j)\neq\varnothing$，则将 e 划分至 $P(v_i)\bigcap P(v_j)$ 的最小负载分区中；

(2) 如果 $P(v_i)\bigcap P(v_j)=\varnothing$ 且 $P(v_i)\bigcup P(v_j)\neq\varnothing$，则将 e 划分至 $P(v_i)\bigcup P(v_j)$ 的最小负载分区中；

(3) 如果 $P(v_i)\neq\varnothing$ 且 $P(v_j)=\varnothing$，则将 e 划分至 $P(v_i)$ 的最小负载分区中；

(4) 如果 $P(v_i)=\varnothing$ 且 $P(v_j)=\varnothing$，则将 e 划分至现有的最小负载分区中.

LDG[73]在贪心算法的基础上，增加了分区的负载限制. HDRF[77]是对贪心算法的一种改进，类似于 DBH，将度数纳入考虑范围，优先对度数较高的顶点进行划分.

对于给定边 $e=(v_i,v_j)$，HDRF 定义分区 P_l 为 $C^{\mathrm{HDRF}}(v_i,v_j,l)$ 的最大值：

$$\theta(v_i)=\frac{deg(v_i)}{deg(v_i)+deg(v_j)}=1-\theta(v_j) \tag{9}$$

$$g(v,l)=\begin{cases}1+(1-\theta(v)), & \text{如果}l\in P(v)\\ 0, & \text{否则}\end{cases} \tag{10}$$

$$C^{\mathrm{HDRF}}(v_i,v_j,l)=\lambda\times\frac{maxsize-|l|}{\varepsilon+maxsize-minsize}+g(v_i,l)+g(v_j,l) \tag{11}$$

其中，$maxsize$ 和 $minsize$ 分别是具有最大负载和最小负载的分区的大小. 当 $\lambda\leq1$ 时，该方法与贪心算法相似，可能会造成分区不平衡. 为了解决这个问题，通常设置 $\lambda>1$. 如果 $\lambda\to\infty$，该方法将趋近于随机划分.

HDRF 可以在处理输入流的同时维护度信息并对其进行更新，而不是在分区之前预先计算度，适用于动态图的划分.

5.3　Fennel 算法及其衍生

Fennel[78]是另一种常见的流式图划分算法，它的核心思想是最大化相邻顶点和最小化不相邻顶点. Fennel 与 LDG 相似，都需要计算相邻顶点数，不同的是 Fennel 需要根据每个分区的负载限制设置最大分配顶点数的阈值. 对于负载低于阈值并且输入顶点被分配给具有最大分数的分区，计算分数 $\delta g(v,P_i)$.

$$\delta g(v,P_i)=|P_i\bigcap N(v)|-\alpha\gamma|P_i|^{\gamma-1} \tag{12}$$

其中，参数 α 和 γ 是可调的，顶点所在的分区为 $\arg\max_i\{\delta g(v,P_i)\}$. 此外，文献[79]提出了基于 Fennel 的再划分方法 re-Fennel，这是一种迭代算法，通过利用先前的划分结果来提高图的划分质量，实验表明其效果可与 METIS 算法相当.

受到 Fennel 算法的启发，文献[80]提出了一种名为 Ginger 的算法. 它对较低度数的顶点使用类 Fennel 算法对顶点及其入边进行划分，与 Fennel 不同的是，Ginger 的目标函数同时考虑了顶点和边. 而对度数较高的顶点，Ginger 使用 Hash 算法对顶点的入边进行划分.

然而，Fennel 算法在计算时要提前获知顶点数 n 和边数 m，这使得其在执行划分时需要保持输入图不变，不适用于动态图的划分.

表 4 给出了流式图划分算法的小结，其他相关细节以及实验结果比较可参见文献[21-22]. 其中文献[22]是流式图划分算法的最新实验性分析结果，通过大量比较实验得出以下结论：Hash 算法是简单但有效的一种方法，尤其是在降低延迟方面；Fennel 算法更适用于对度数较低的图进行划分，例如道路交通网络；Ginger 算法在像社交网络这样具有长尾分布的图上更加有效；对于幂率图而言，HDRF 则是最佳算法，因为其较低的顶点切割和均衡的负载可以使图划分获得最佳性能.

表 4　流式图划分算法小结

类型	优点	缺点	算法	描述
Hash 算法及其衍生	计算简单，时间复杂度低，可以在划分过程中随时对图进行更新	初始随机划分，导致跨越分区的顶点数较多，划分质量较差	Hash 算法[73]	对顶点或边随机选择分区
			DBH[74]	在 Hash 算法的基础上，优先对有更高度数的顶点进行切割
			Grid[75]	通过构造矩阵得到两个候选分区，再选择负载较小的分区
贪心算法及其衍生	适用于动态图的划分，对幂率图的划分效果明显	贪心算法划分质量较差；改进后的 HDRF 时间复杂度较高	贪心算法[76]	基于特定规则选择负载较小的分区
			HDRF[77]	在贪心算法的基础上，优先对有更高度数的顶点进行切割
Fennel 算法及其衍生	同时考虑相邻顶点数和非相邻顶点数，划分质量较高	计算较为复杂；需要提前获知顶点数和边数，不适用于动态图的划分	Fennel[78]	在最大化相邻顶点和最小化不相邻顶点之间进行插值
			Ginger[80]	低度数顶点使用类 Fennel 算法，高度数顶点使用 Hash 算法

与之前的图划分算法不同，流式图划分算法在划分时需要将知识图谱加载为包含顶点或边的序列，在整个过程中只需一次图遍历，而不是将图载入后进行多次迭代计算，所以能够作用于包含上亿个顶点和边的知识图谱，符合大规模知识图谱的划分要求.

6 分布式图划分算法

随着图数据规模的不断增大，如何在分布式环境下实现图划分算法已成为必然需求. 最常见的方法是开发已有图划分算法的并行版本，例如，METIS 算法的并行版本 ParMETIS[72]、Scotch 算法[81]的并行版本 PT-Scotch[82]. 这两种都是基于递归二分法的算法，它们比直接 k 分区算法更难于并行化，因为在最初的两个分区中，可用的并行性较少. 这两种算法属于早期分布式图划分算法，本节将主要介绍最近提出的几种典型分布式图划分算法.

6.1 KaPPa 算法及其衍生

Holtgrewe 等人[53]提出了一种分布式多级图划分算法 KaPPa. 该算法在粗糙化阶段使用文献[83]提供的分布式匹配算法，并且使用 FM 算法进行细化，使用邻接矩阵存储顶点和边的权重，使用 Hash 表存储迁移的顶点和对应的边. 在进行局部改进算法之前，需要在每个分区的边界执行广度优先搜索生成顶点集，细化操作将在这些顶点集之间进行.

KaFFPa[62]在 KaPPa 的基础上，不考虑并行化，而是专注于大规模图的划分，扩展了文献[84-85]引入的多级迭代算法，其主要思想是在粗糙化和反粗糙化阶段进行迭代，一旦图被划分，两个分区之间的边就不会收缩，这样可以确保分区质量. 传统的多级划分算法只执行一次粗糙化和反粗糙化，也被称作 V 循环[62]. KaFFPa 提出了两种新的全局搜索方法，分别是 W 循环和 F 循环. W 循环的工作原理如下：在每个级别上使用不同的随机种子进行两次独立的粗糙化和反粗糙化，从而得到更好的划分结果. F 循环类似于 W 循环，区别在于每一级上最多执行两次递归调用. 不同循环的例子见图 12.

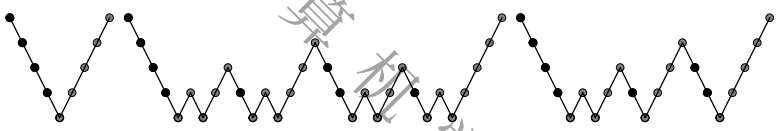


图 12 从左到右依次为多级划分算法的 V 循环、W 循环和 F 循环[62]

KaFFPaE[63]是 KaFFPa 的进化算法(EA)版本，从图的初始划分开始，以边切割数作为划分标准，经过多轮选择后，得出最优划分结果. 为了避免陷入局部最优，引入变异算子以减缓算法收敛. KaFFPaE 定义了两个变异算子 M_1 和 M_2，它们随机作用于不同顶点，均执行 F 循环.

KaFFPaE 的并行化过程如下：首先需要估计图的分区数 S，每个处理单元执行划分步骤并测量划分时间 t，再选择 S 使得创建 S 个分区的时间约为 t_{total}/f，其中 f 是调整参数，而 t_{total} 是图划分的总运行时间. 每个处理单元多次调用 KaFFPa 创建 S 个分区，并以一定概率插入变异算子. 只要有时间剩余，算法就会循环进行.

文献[86]改进了 KaFFPaE，能够实现完全平衡的划分. 文献[87]将 KaFFPaE 与第 7 节的标签传播算法结合，得到了更高质量的图划分效果.

6.2 JA-BE-JA 算法及其衍生

Rahimian 等人[88]提出了名为 JA-BE-JA 的分布式图划分算法，该算法使用局部搜索和模拟退火技术进行图划分. JA-BE-JA 算法首先随机均匀地为每个顶点分配一种颜色，π_v 指顶点 v 的颜色，具有相同颜色的顶点属于同一分区. 将 $N_v(c)$ 定义为具有颜色 c 的 v 的邻居集合，顶点 v 的邻居数用 d_v 表示，$d_v(c)=|N_v(c)|$ 是具有颜色 c 的 v 的邻居数. 图的能量定义为具有不同颜色的顶点之间的边数. 因此，顶点的能量是其具有不同颜色的邻居的数量，并且图的能量是顶点的能量的总和：

$$E(G)=\frac{1}{2}\sum_{v\in V}(d_v-d_v(\pi_v)) \tag{13}$$

然后将相邻顶点和随机顶点集作为候选顶点进行搜索，如果交换顶点颜色后图的能量降低，则进行交换，当搜索不到可交换颜色的顶点时，算法终止. 交换过程如图 13 所示.

图 13 p 和 q 交换前，与 p 颜色相同的邻居数为 1，与 q 颜色相同的邻居数为 0；p 和 q 交换后，与 p 颜色相同的邻居数为 3，与 q 颜色相同的邻居数为 1，3+1>1+0，交换后图的能量更高

$$d_p(\pi_q)^\alpha+d_q(\pi_p)^\alpha>d_p(\pi_p)^\alpha+d_q(\pi_q)^\alpha \quad (14)$$

其中，α 是能量函数的参数. 利用模拟退火技术防止陷入局部最优，引入温度 T 并让它随时间减少，类似于冷却过程，T 降为 1 时不再下降.

JA-BE-JA-VC[89]是 JA-BE-JA 的一个变体，其基于边划分而不是顶点划分. 该算法首先将一组边随机划分至不同的分区，同时允许顶点被复制到多个分区中，然后定义每个边和每个分区的能量函数，根据系统能量判断是否交换边. 该算法通过应用局部搜索算法迭代地改进初始划分.

6.3　轻量级重划分算法

为了适应动态图的计算，需要根据输入图的更新信息在已有划分基础上再次划分，这个过程称为重划分[90]. 如果重划分算法仅依赖于要重新划分的图中的少量信息，则称其为轻量级重划分算法. 当新顶点加入图或图的权重发生变化时，轻量级重划分将被触发，同时通过迭代过程减少边切割，适用于动态图的划分. 该算法利用聚合顶点权重信息作为其辅助数据，相比于图的结构信息，该数据要小得多. 假设 v 是分区 P_s 中的顶点，增益 $g(v)$ 是将 v 从 P_s 移至 P_t 所减少的边切割数，增益可以是负值. 在每次迭代中定义两个阶段，第一阶段仅允许从具有较低编号到较高编号的分区移动顶点，而第二阶段仅允许在相反方向上移动顶点. 如果顶点可以移动至多个分区，则选择增益最大的分区，当某分区权重超过负载时，不再向该分区迁移顶点. 当没有顶点需要移动时，算法结束. 整个过程如图 14 所示.

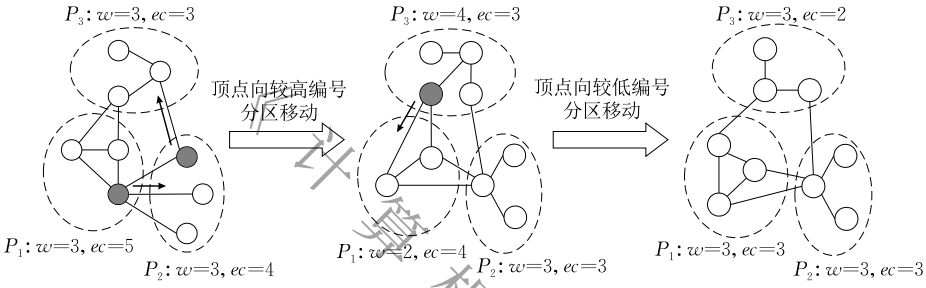


图 14　左图为初始划分图，P_1，P_2，P_3 代表分区，w 代表权重，ec 代表边切割数，灰色的顶点表示该点需要移动至编号更高的分区中(即从 P_1 到 P_2，P_2 到 P_3)；中间图的灰色顶点需要移动至编号更低的分区中(即从 P_3 到 P_1)；右图为算法结束时的划分图

文献[91]提出一种将轻量级重划分集成到 Hermes 系统中的方法，并将其嵌入至 Neo4j 图数据库[92]，以支持在分布式环境下进行图划分. 文献[93]将划分分为两个阶段，逻辑顶点迁移和物理顶点迁移，在降低通信成本的同时能够更好地应用于现实中的大规模图数据.

表 5 给出了上述几种分布式图划分算法的小结. 划分时通常会使用分布式图处理框架 MapReduce[94]，通过引入编程模型来促进高效分布式算法执行，同时抽象出底层细节. 除此之外，常用的分布式图处理框架还有 Pregel[95]、GraphX[96]、PowerLyra[80] 等，其划分质量与所使用的划分算法有关，具体细节及相关比较可参见文献[19]. 近年来出现的质量较高的分布式图划分算法还有在 Hadoop 和 Spark 等分布式框架上运行的 DFEP[97] 和 DynamicDFEP[98]、增量在线图划分算法 IOGP[99]、将图转化成树的

表 5　分布式图划分算法小结

类型	优点	缺点	算法	描述
KaPPa 算法及其衍生	引入分布式匹配算法，使得多级算法能够划分大规模图	需要提前存储顶点和边，不适用于动态图的划分	KaPPa[53]	粗糙化阶段使用 GPA 算法计算匹配，并且使用 FM 算法进行细化
			KaFFPaE[63]	在 KaPPa 的基础上，在粗糙化和反粗糙化阶段进行 F 循环
			Meyerhenke 等人[87]	将标签传播算法与 KaFFPaE 相结合
JA-BE-JA 算法及其衍生	计算简单，局部性强，且能避免陷入局部最优	运行时间与迭代次数呈正相关，不适用于动态图的划分	JA-BE-JA[88]	为每个顶点随机分配颜色，定义顶点能量，通过交换顶点减少图的总能量
			JA-BE-JA-VC[89]	为每个顶点随机分配颜色，定义边的能量，通过交换边减少图的总能量
轻量级重划分算法	内存较小，可随时更新顶点	运行时间与迭代次数呈正相关	Hermes[91]	按照指定顺序迁移顶点，并将其集成到 Hermes 系统中
			Planar[93]	将轻量级重划分分为逻辑顶点迁移和物理顶点迁移两个阶段

Sheep 算法[100]、基于线形嵌入的划分算法[101]以及基于定向边交换的分布式在线大图划分算法[102]. 可以看出,分布式图划分算法从原来的研发已有算法的并行版本逐渐演变为直接在分布式环境下提出新算法. 输入图也不局限于静态图,而增加了结合实际需要随时可更新数据的动态图.

分布式图划分算法可用于具有上亿个顶点和边的大规模图数据的划分,是目前进行大规模真实知识图谱划分的主流算法. 但现有大部分分布式图划分算法不能充分利用知识图谱的丰富属性信息.

7 其他类型图划分算法

除了上述 4 种类型的图划分算法外,还有一些其他类型的算法有着较好的图划分效果,例如,基于遗传/进化算法进行图划分[103]、基于禁忌搜索的顶点交换方法[104]、基于语义的 RDF 图划分算法[105]、基于 BSP 模型的大图划分算法[106-107]以及动态平衡图划分算法[108]. 限于篇幅,本节对其中两种常用的算法类型进行介绍.

7.1 标签传播算法

标签传播算法(LPA)[109]首先为每个顶点随机均匀分配一个标签,然后迭代地更新顶点标签. 在每次迭代中,顶点将其相邻顶点中数量最多的标签作为自己的标签. 标签不再更改时,该过程终止. 具有相同标签的顶点属于同一分区. 标签传播算法中顶点更新的过程如图 15 所示.

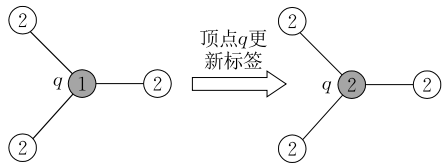


图 15 标签传播算法中顶点 q 更新自身标签

但 LPA 无法保证分区大小平衡,文献[110]通过提供精确约束,即给定分区顶点数量的上限和下限以解决该问题. 文献[111]提出与 METIS 结合的算法 MLP,在粗糙化过程中使用 LPA,可划分具有 10 亿以上顶点的图.

以上两种方法只适用于静态图,xDGP[112]是一个基于 LPA 和顶点迁移的自适应分布式图划分系统. Spinner[113]是基于 LPA 以顶点为中心在 Pregel 下实现的图划分算法,其根据权重和分区大小为每个顶点赋予对应的分数,并以此判定顶点是否迁移至其他分区. 文献[114]将 KaFFPa 与 LPA 结合,并在文献[87]中实现了其分布式算法,可划分上亿顶点的图,且质量要高于 ParMETIS 和 PT-Scotch,但是该方法在划分知识图谱时需要将其转化为无向图,未利用知识图谱的属性信息.

7.2 基于查询负载的算法

近年来由于知识图谱的广泛应用,出现了一些提升 RDF 知识图谱查询性能的图划分算法,文献[115]中使用对 RDF 谓语进行随机划分的算法 PB,可在一定程度上提高查询效率.

还有一些方法则是直接基于查询负载对 RDF 图进行划分[60,116-117],其中文献[117]增加了查询负载 $\mathbb{Q}=\{Q_1,Q_2,\cdots,Q_r\}$. 该算法的目的是根据边属性出现的频繁程度对边进行划分,从而提高查询性能. 为了寻找具有高访问频率的模式(FAP),需要定义 FAP 使用值 $use(Q,p)$ 以记录 FAP 的访问频率,其中 Q 是一个 SPARQL 查询,p 是一种频繁访问模式.

$$use(Q,p)=\begin{cases}1, & \text{如果 } p \text{ 是 } Q \text{ 的子图}\\ 0, & \text{否则}\end{cases} \tag{15}$$

给定模式 p,p 命中查询 Q 的增益为 $Benefit(p,Q)=|E(p)|\times use(Q,p)$,其中 $E(p)$ 是 p 中的一组边. 对于一组频繁模式 $P=\{p_1,p_2,\cdots,p_x\}$,它在 $\mathbb{Q}$ 上的增益是其所有 FAP 的增益和.

$$Benefit(P,\mathbb{Q})=\sum_{Q_i\in\mathbb{Q},i=1,\cdots,r}\max_{p\in P}\{Benefit(p,Q_i)\} \tag{16}$$

算法的优化目标是最大化受存储约束限制的增益. 由于这个问题是 NP 难的,文献[117]采用贪心算法选择频繁访问模式. 图 16 是频繁访问模式的一个示例.

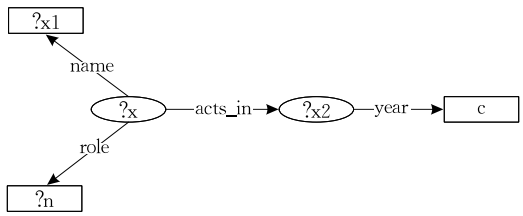


图 16 频繁访问模式

文献[117]提出了两种分片策略:垂直分片和水平分片.

垂直分片需要将匹配到相同频繁访问模式的图放入同一个分片. 因为单个 RDF 图包含的频繁访问模式有限,过滤掉不相关的元素可以提高查询性能. 对于图 4 的 RDF 图,采用图 16 的频繁访问模式,垂直分片如图 17 所示.

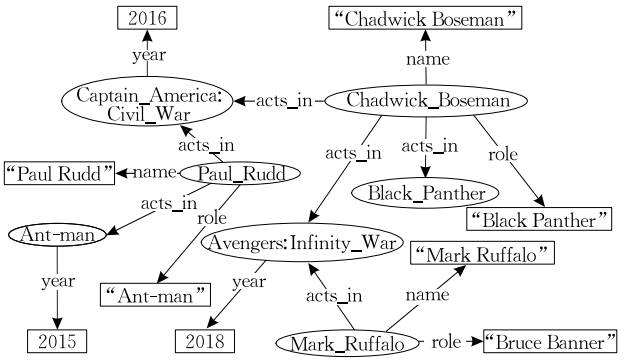


图 17　基于查询负载的垂直分片

对于水平分片，需要将相同频繁访问模式的匹配放入不同的分片中. 首先定义结构简单谓语 sp 为单个频繁访问模式对应的等式或不等式. 假设频繁访问模式 p 含有变量集合 $Var=\{var_1,var_2,\cdots,var_n\}$，$sp$ 的定义如下：

$$sp:p(var_i)\theta Value \tag{17}$$

其中 $\theta\in\{=,\neq\}$，$Value$ 是常量约束.

以图 16 的频繁访问模式为例，可以生成结构简单谓语如下：

(1) $sp_1:p(?x1)=$"Chadwick Boseman"；

(2) $sp_2:p(?x1)\neq$"Chadwick Boseman"；

(3) $sp_3:p(?x2)=$Avengers：Infinity_War；

(4) $sp_4:p(?x2)\neq$Avengers：Infinity_War.

再将结构最小谓语定义为相同频繁访问模式的 sp 的合取. 给定一组 $SP=\{sp_1,sp_2,\cdots,sp_y\}$，结构最小谓语 $M=\{mp_1,mp_2,\cdots,mp_z\}$ 的定义如下：

$$M=\{mp_i\mid \bigwedge_{sp_k\in SP} sp_k^*,1\leq k\leq y\} \tag{18}$$

其中，$sp_k^*=sp_k$ 或 $sp_k^*=\neg sp_k$. 然后根据结构最小谓语对图进行划分.

依然以图 16 的频繁访问模式为例，对应的结构最小谓语如下：

(1) mp_1：$p(?x1)=$"Chadwick Boseman" $\wedge$ $p(?x2)=$Avengers：Infinity_War；

(2) mp_2：$p(?x1)=$"Chadwick Boseman" $\wedge$ $p(?x2)\neq$Avengers：Infinity_War；

(3) mp_3：$p(?x1)\neq$"Chadwick Boseman" $\wedge$ $p(?x2)=$Avengers：Infinity_War；

(4) mp_4：$p(?x1)\neq$"Chadwick Boseman" $\wedge$ $p(?x2)\neq$Avengers：Infinity_War.

图 18 显示了从上述结构最小谓语生成的所有水平分片.

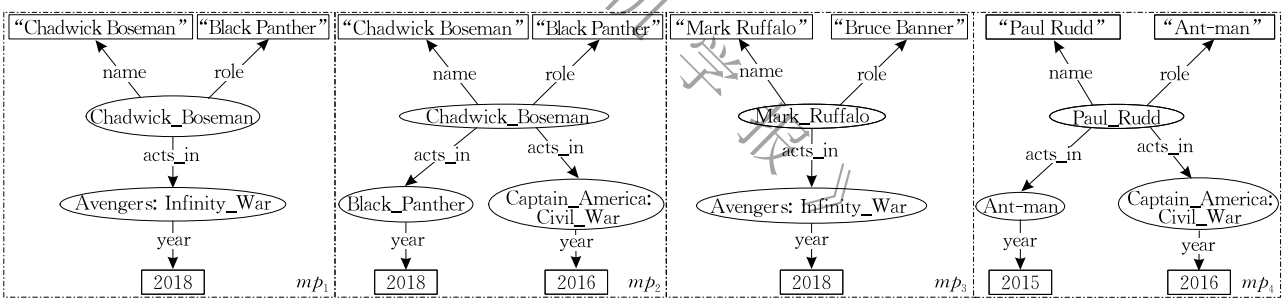


图 18　基于查询负载的 RDF 图划分中结构最小谓语 mp_1、mp_2、mp_3 和 mp_4 对应的水平分片

垂直分片和水平分片的侧重点不同，前者利用查询的局部性来提高吞吐量，后者旨在通过并行化子查询来缩短查询响应时间. 文献[117]最后在权衡这两种分片策略优势的基础上，提出混合分片策略.

表 6 给出了以上两种划分算法的一个小结.

表 7 汇总了本文介绍的所有类型的知识图谱划分算法，包括每种算法的文献和主要特点描述. 从表中可以看出，流式图划分算法、分布式图划分算法以及基于查询负载的算法有很大的发展空间，应该成为未来知识图谱划分的重点研究方向.

表 6　其他类型图划分算法小结

类型	优点	缺点	算法	描述
标签传播算法	计算简单、直接，划分后的边切割数较少	本身不保证负载平衡，需要加限制条件或与其他算法结合	Ugander 等人[110]	顶点根据约束条件更新自身标签
			MLP[111]	在粗糙化阶段使用标签传播算法
			xDGP[112]	在用标签传播算法进行划分后进行顶点迁移
			Spinner[113]	根据权重和分区大小为每个顶点赋予分数，以此判定顶点是否迁移
基于查询负载的算法	充分考虑了图的结构信息和属性信息	只针对 RDF 图，计算较为复杂	WARP[60]	将 METIS 算法应用于 RDF 图，并通过匹配查询模式进行扩展
			Peng 等人[117]	根据频繁访问模式对 RDF 图进行划分

表 7 知识图谱划分算法总结

类型	名称	文献	描述
基本图划分算法	谱划分算法	[25]~[31]	利用 Laplace 矩阵的特征向量对图进行划分
	几何划分算法	[28][32]~[37]	根据顶点坐标进行图划分
	分支定界算法	[40]~[47]	将图划分分成多个子问题递归求解,从而找到最优解
	KL 算法及其衍生	[48][49][50][51]	在已有图划分基础上交换顶点
多极图划分算法	基于匹配的算法	[50]~[63]	将图粗糙化为小图进行划分,再将划分结果投影回原始图
	基于聚合的算法	[64]~[71]	基于 GPA 匹配的多级划分
流式图划分算法	Hash 算法及其衍生	[73][74][75]	构建散列函数,实现随机划分
	贪心算法及其衍生	[76][77]	基于特定规则选择负载较小的分区
	Fennel 算法及其衍生	[78][79][80]	在最大化相邻顶点和最小化不相邻顶点之间进行插值
分布式图划分算法	KaFFPaE 算法及其衍生	[53][63][87]	在每个级别上执行多次粗糙化/反粗糙化过程
	JA-BE-JA 算法及其衍生	[88][89]	使用顶点交换和模拟退火技术进行图划分
	轻量级重划分算法	[90][91][93]	根据图的更新信息进行划分,顶点按照指定顺序迁移
其他类型图划分算法	标签传播算法	[87][109]~[114]	更新顶点标签为最多邻居顶点标签,多次迭代后得到结果
	基于查询负载的算法	[60][116][117]	从 RDF 图中选出常用的查询负载模式,并以此作为划分依据

8 实验分析

前文中提到,知识图谱划分是为知识图谱的分布式存储、查询、推理和挖掘提供了基础支撑.本节选取查询和挖掘,将划分后的数据集上传至集群进行分布式存储,通过实验比较不同划分算法对查询和挖掘的响应时间,分析划分效果与二者的内在联系,并延伸至推理层面.

8.1 实验环境

实验分析前简要介绍本次实验使用的硬件和软件配置.硬件配置:实验集群由 5 台腾讯云标准型 S5 服务器构成,每台机器都有 4 个 CPU 和 16 GB 内存,系统盘大小 50 GB,网络连接为以太网.

软件配置:操作系统为 CentOS 7.6,每台机器安装分布式 RDF 知识图谱数据库管理系统 gStoreD[①] 来进行查询,并使用 MPICH-3.3.2 来进行集群中多个机器节点间的通信.实验在合成数据集 LUBM 和真实数据集 DBpedia 2019-08-30[②] 的实体间链接图文件 mappingbased_objects_en 上完成,对数据集中的 RDF 图进行划分,数据集的具体内容如表 8 所示,其中平均度的定义如下:

$$\text{平均度}=\frac{\text{顶点数}}{\text{边数}} \tag{19}$$

表 8 最大负载率

算法	LUBM100	DBpedia
Hash	1.00	1.00
METIS	1.03	1.03
HDRF	1.00	1.00
JA-BE-JA	1.00	1.00
PB	1.61	1.24

实验从前文提到的图划分算法中选取 5 种典型算法,对其图划分效果进行实验验证.算法包括:基于随机划分思想的 Hash 算法、多级图划分算法 METIS(见 4.1 小节)、流式图划分算法 HDRF[③](见 5.2 小节)、分布式图划分算法 JA-BE-JA[④](见 6.2 小节)以及基于 RDF 谓语的划分算法 PB[⑤](见 7.2 小节).

8.2 划分效果分析

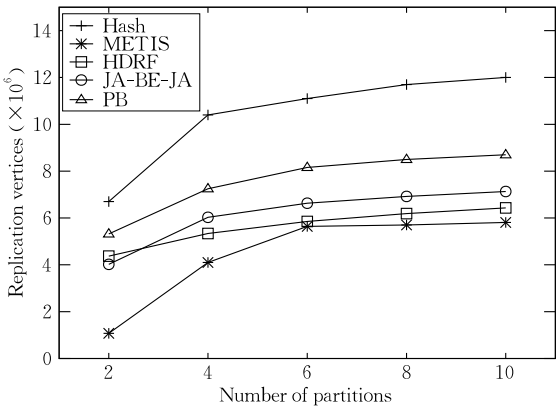
由于 RDF 图是三元组的集合,故对于顶点划分算法,需要将宾语复制到主语所在分区,以保证三元组的完整性.第 2 节给出了图划分的两个目标,分别是最小化跨越分区的顶点数或边数以及负载平衡.本小节针对这两个指标展开实验,选择 LUBM100 作为 LUBM 数据集的代表,在其和 DBpedia 数据集上使用 5 种具有代表性的图划分算法,分别在 2 到 10 个分区进行实验.为了统一划分标准,通过统计复制顶点数来比较划分效果,复制顶点数越少说明该算法的划分效果越好,其结果如图 19 所示.并统计了最大分区顶点数与理想状态下各分区顶点数的比值作为最大负载率,在表 9 给出,其定义如下:

$$\text{最大负载率}=\frac{\text{最大分区顶点数}}{\text{负载平衡时各分区的顶点数}} \tag{20}$$

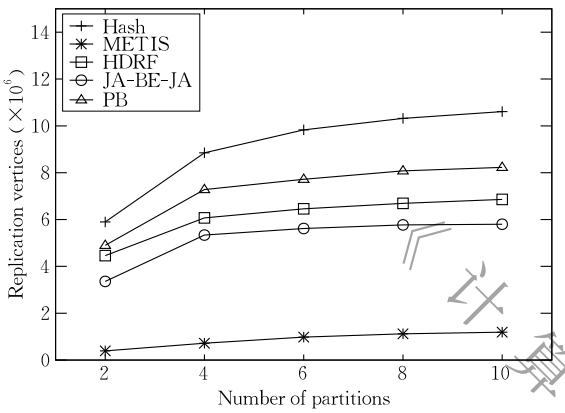
从图表中可以得出以下结论:

(1) Hash 算法的复制顶点数最多,这是由于 Hash 算法是一种随机划分的算法,不能保证划分质量,同时,Hash 算法严格按照平衡分区进行计算.由前文分析可知,各类图划分算法的目的均在于减少

① Using gStoreD link. https://github.com/bnu05pp/gStoreD.html 2019,2,16
② DBpedia datasets. https://wiki.dbpedia.org/develop/datasets.html 2019,8,30
③ Using HDRF link. https://github.com/fabiopetroni/VGP.html 2015,7,6
④ Using JA-BE-JA link. https://github.com/fatemehr/jabeja.html 2016,6,7
⑤ Using PB link. https://github.com/dice-group/rdf-partitioning.html 2018,7,12



(a) LUBM100



(b) DBpedia

图 19　不同划分算法的复制顶点数

不同机器间的通信开销，而 Hash 算法的目的只是为了快速划分，并不考虑划分质量，故其他 4 种算法的复制顶点数均小于 Hash 算法.

(2) METIS 算法在这两个数据集上的综合划分效果最好，特别是在 DBpedia 数据集下，其平均复制顶点数仅为 Hash 算法的 9.2%，并且明显少于另外 3 种算法，这也是目前在较小规模图上最常用 METIS 算法的原因. 此外，METIS 算法在粗糙化图上进行平衡划分，但是在投射回原始图时会造成一定的偏差，导致最终负载稍有不平衡.

(3) HDRF 算法在 LUBM 数据集下的划分效果更好，主要原因在于 HDRF 算法适用于幂率图，而 LUBM 数据集的平均度更高，说明其包含的高次顶点数要多于 DBpedia 数据集，更有利于 HDRF 算法的运行.

(4) JA-BE-JA 算法的划分效果在这 5 种算法中处于中等水平，因其适用范围较广，更适合在大规模数据集上进行划分. 由于该算法是在已经划分好的平衡分区上进行顶点交换，所以可以保持负载完全平衡.

(5) PB 算法的复制顶点数较多，仅次于 Hash 算法，这是因为它在划分时主要基于的是一种随机方法，同时，该算法没有考虑负载平衡，在一定程度上降低了复制顶点数.

表 9　实验所用数据集

	数据集	大小	顶点数	边数	平均度=边数/顶点数
LUBM	LUBM20	341 M	663647	2687596	4.05
	LUBM40	676 M	1308274	5495742	4.20
	LUBM60	0.99 G	1971871	8002472	4.06
	LUBM80	1.33 G	2642810	10726415	4.06
	LUBM100	1.66 G	3301718	13403134	4.06
DBpedia	mappingbased_objects_en	1.52 G	4256892	11018249	2.59

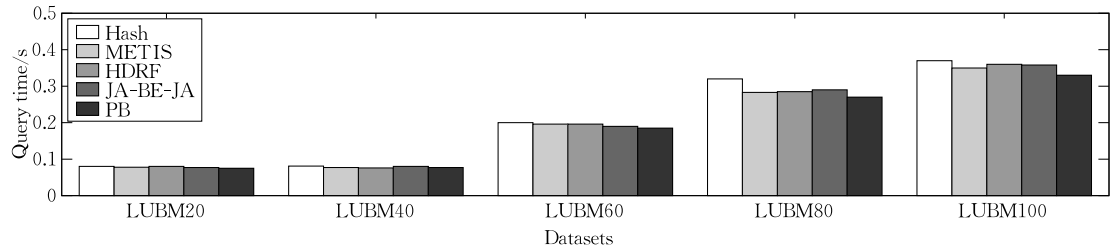
8.3　查询时间分析

本组实验对比了不同划分算法在分区数为 4 时的查询时间，具体实现方法是在 LUBM 数据集和 DBpedia 数据集上进行划分后，对 LUBM 数据集使用查询 Q1、Q2 和 Q3，对 DBpedia 数据集使用查询 DQ1、DQ2 和 DQ3，同时记录查询的响应时间，其中 Q1 和 DQ1 属于星型查询，Q2、Q3、DQ2 和 DQ3 属于复杂查询，Q3 和 DQ3 匹配的查询数要远超过 Q2 和 DQ2，查询难度依次增加，查询语句可在附录中查看，图 20 和图 21 分别展示了这两组数据集上的实验结果.

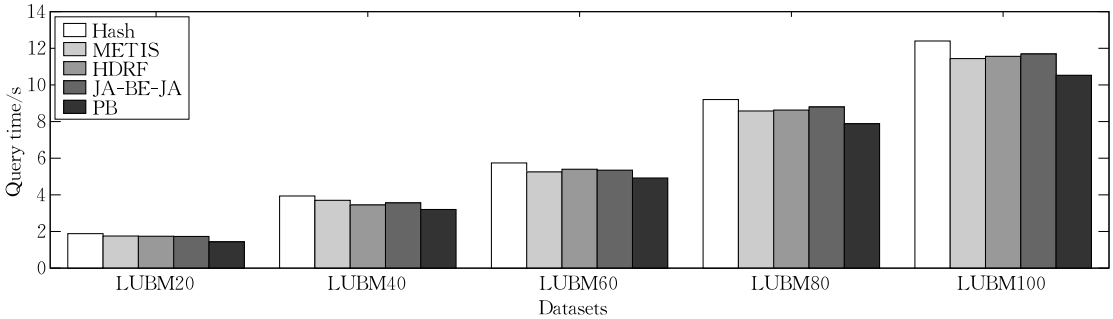
在 LUBM 数据集上，显然 Hash 算法的查询时间在 5 种算法中是最长的，PB 算法的查询时间最短，而其他 3 种划分算法并无显著差别. 从图 20(a) 可以看出，各类算法在简单查询上的查询时间差别不大，尤其是在较小数据集 LUBM20 和 LUBM40 上，二者的查询时间几乎相同. 随着查询难度的增加，图 20(b)开始有较明显的差距，直到在图 20(c)中，Hash 算法在 LUBM100 数据集上的查询时间与 PB 算法相差 12.9 s，达到最大.

在 DBpedia 数据集上的结果与 LUBM 数据集相似，不同的是差距要小一些，在图 21(c)上方能看到较为清楚的结果，这与两个数据集的平均度有关系，DBpedia 数据集的平均度小，导致可匹配的查询数少，对应的查询时间降低. 相比较而言，在 DBpedia 数据集上 METIS 算法的查询效果要比 LUBM 数据集上更好一些.

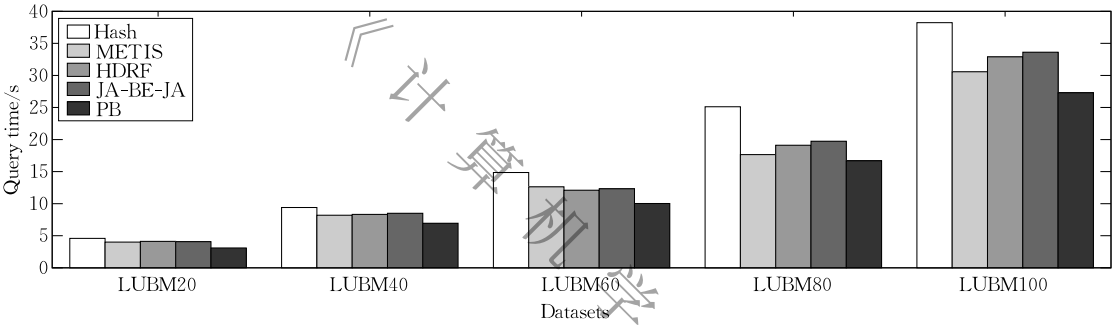
通过对比划分效果图可知，查询时间与复制顶



(a) 查询Q1的运行时间

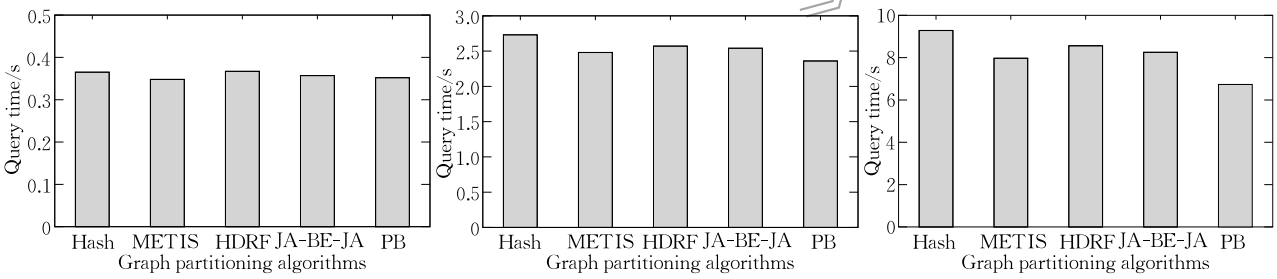


(b) 查询Q2的运行时间



(c) 查询Q3的运行时间

图 20 不同划分算法在 LUBM 数据集上的查询时间



(a) DQ1 (b) DQ2 (c) DQ3

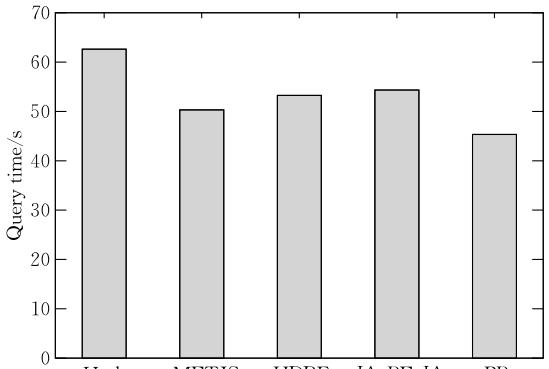
图 21 不同划分算法在 DBpedia 数据集上的查询时间

点数总体上呈正相关. Hash 算法的划分效果最差，所以查询耗时最长，METIS 算法在 DBpedia 数据集上的划分效果较好，对应的查询用时较少. 这是由于随机分配导致图处理时跨机器的计算较多，通信时间大大增加，查询时间随之增长. PB 算法的情况特殊，因其增加了根据谓语进行划分这一环节，提高了查询效率，所以它的查询时间要少于其他划分算法. 由此可得出结论，对于大部分知识图谱划分算法，跨越分区的顶点数或边数越多，基于此种知识图谱划分算法的查询处理时间就越长.

8.4 挖掘时间分析

由于数据挖掘是从大量数据中搜索隐藏信息，这里采用复杂查询方式达到这一目的，在 LUBM100 数据集和 DBpedia 数据集上分别使用挖掘语句 DM1 和 DM2，挖掘语句可参见附录，图 22 展示了不同划分算法在两个数据集上分区数为 4 时的挖掘时间.

与查询结果相似，Hash 算法的挖掘时间明显超过其他 4 种算法，PB 算法的时间最短，METIS 算



(a) LUBM100

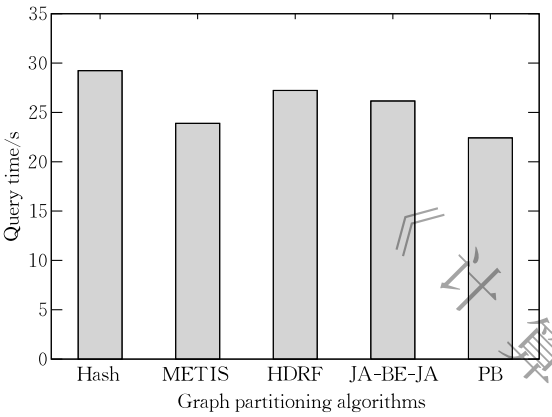


图 22　不同划分算法的挖掘时间

法次之. 不同的是，因为挖掘耗时较长，各个算法间的差距比查询时更大. 所以对于挖掘来说，其运行时间主要与算法的划分效果有关，跨越分区的顶点数或边数越多，执行数据挖掘时的通信开销就越大，相应的运行时间也越长.

我们相信，该结论同样适用于知识图谱的推理，因为任何在知识图谱上进行的分布式计算都需要考虑通信开销的问题，而图划分的目标是减少跨越分区的顶点或边，从而减少通信时间. 根据上述实验分析可知，知识图谱划分效果越好，划分后进行查询、推理和挖掘等操作的时间就越少.

8.5　实验总结

表 10 给出了本节实验效果的一个总结. 显然，Hash 算法的划分效果、查询效果和挖掘效果较差，但是它的时间复杂度低，容易实现，适合在大规模图上需要快速划分的情况；METIS 算法的整体效果较好，在 DBpedia 数据集上表现最为明显，但也有其局限性，只适用于本地环境下规模较小的数据集；HDRF 算法和 JA-BE-JA 算法处于中间水平，不同的是 HDRF 算法更适用于幂率图，所以在 LUBM 数据集上的划分、查询、挖掘效果要比 JA-BE-JA 算法更好；PB 算法的划分效果较差，但是查询和挖掘效果最好，运行时间少，适合需要快速进行 RDF 图计算的情况. 综上所述，没有哪一种算法在所有情况下都是最好的，需要根据实际情况选择合适的算法，从而高效地实现知识图谱分布式存储、查询、推理、挖掘等一系列操作.

表 10　实验效果总结

算法	划分效果	查询效果	挖掘效果
Hash	负载平衡，复制顶点数最多	查询时间最长	挖掘时间最长
METIS	负载稍不平衡，但复制顶点数最少，特别是在 DBpedia 数据集上	查询时间较短	挖掘时间较短
HDRF	负载平衡，复制顶点数较少，在 LUBM 数据集上效果更好	查询时间较长	挖掘时间较长
JA-BE-JA	负载平衡，复制顶点数较少	查询时间较长	挖掘时间较长
PB	负载不平衡，复制顶点数较多	查询时间最短	挖掘时间最短

9　未来研究方向

通过上述分析与比较，可以看出知识图谱划分已经取得了一定的成果，提出了不少成熟的理论和算法. 但是，现存的知识图谱划分算法的理论、方法与技术尚有许多局限性，仍面临着许多问题和挑战，进一步的研究重点包括：

(1) 知识图谱划分标准与算法效率如何权衡

本文介绍的知识图谱的划分算法大部分是在一般图划分算法的基础上进行的，由于知识图谱包含信息要比一般的图模型丰富和复杂，划分通常和后期的存储查询相结合，增加了划分难度. 传统的图划分方法以降低顶点或边的切割数为划分目标，以负载是否平衡、是否具有较小的时间复杂度等作为衡量划分好坏的标准. 但是对于知识图谱而言，除了这些以外，还应以知识图谱的图结构和属性信息作为划分标准，划分是为了更好地执行后续的存储、查询、推理和挖掘操作. 此外，知识图谱数据模型尚未有统一定义，也没有确定的标准. 所以需要深入研究针对知识图谱的划分算法，做到既要有利于支持知识图谱查询的快速执行，又要避免划分算法复杂度过高.

(2) 面向知识图谱高层语义如何划分

知识图谱的一个重要特点是对本体的表示. 以

RDF 图为例,其源于语义 Web 的发展,其在标准制定之初即面向 Web 上全局资源的表示、发布和集成,在 RDF 图之上定义了 RDF 模式语言(RDF Schema)和 Web 本体语言(OWL),可用于表示丰富的高级语义知识.虽然目前已经有了针对 RDF 图语义划分的算法,但是还没有考虑利用本体语言作为启发信息进行划分.如何利用知识图谱高层语义指导划分是非常重要的研究方向.

(3)面向知识图谱应用需求如何划分

知识图谱的应用领域非常广泛,在社交网络、道路交通、电子商务等方面都有着重要作用.以电子商务为例,企业可以构建用户的知识图谱,对用户的商品查询情况、购买记录进行分析,将用户划分至对应的消费人群.知识图谱的领域应用研究正处于起步阶段,目前各个领域特定的知识图谱划分结果还没有相应的质量评价指标,如何针对应用需求进行知识图谱划分还需要进行深入研究.

(4)超大规模知识图谱如何划分

目前,大规模知识图谱划分普遍使用分布式算法,例如,基于 MapReduce、Pregel、MPI 等已有分布式处理框架进行划分.但是随着具有数十亿三元组规模的知识图谱不断涌现,现有分布式方法已不足以应对超大规模知识图谱的划分,通常会导致算法运行时间增长,不同机器间跨越边数或顶点数增多,通信成本增加等.这就需要进一步研究超大规模知识图谱的高效划分方法,使得知识图谱在负载尽量均衡的情况下减少不同机器间的通信开销,在满足划分目标的同时降低划分算法的复杂度.

(5)基于知识图谱表示学习如何划分

近年来,知识图谱表示学习开始兴起,其主要思想是将知识图谱的实体和关系嵌入到低维向量空间中,在保留结构信息的同时有效支持下游任务.基于知识图谱表示学习的划分方法是一个值得研究的新方向,目前已有的方法是在图嵌入后利用坐标信息,使用几何划分方法进行划分.但这些算法只限于单机环境,还没有扩展到分布式框架上.如何更好地将表示学习与其他常用的划分方法特别是分布式图划分算法结合,从而提高划分的效率和质量是非常有意义的研究课题.

(6)知识图谱划分如何支持存储查询

知识图谱划分是分布式存储的必要手段,划分的目的是为了更好地执行后续的存储查询.所以划分阶段需要考虑存储查询,同样地,执行存储查询时也需要考虑划分.当前已经有基于查询负载的知识图谱划分算法,但都是基于静态图的划分.在现实应用中,知识图谱数据会随时间而动态变化,需要及时更新划分的输入图.如何使其与动态图划分算法更好地结合,以及如何对划分算法与存储查询方案融合进行优化设计仍需进一步研究.

10 总 结

知识图谱以精确的语义描述了现实世界中的实体以及实体间的关系,有着广泛的应用.知识图谱划分作为大规模知识图谱分布式处理的首要工作,具有十分重要的意义.本文对现有知识图谱划分算法进行了研究综述,介绍了 4 种基本图划分算法,比较了多级图划分算法的 2 种类型,给出了流式图划分算法的主要思想,描述了 3 种分布式图划分算法,介绍了最近提出的 2 种新型图划分算法,在合成与真实知识图谱数据集上探究了 5 种知识图谱代表性划分算法在划分效果、查询处理与图数据挖掘方面的性能差异,最后展望了知识图谱划分算法的未来研究方向.

致 谢 感谢湖南大学信息科学与工程学院彭鹏博士在实验环境配置中提供的帮助,同时感谢编辑部和审稿人给予本文的宝贵意见!

参 考 文 献

[1] Pujara J, Miao H, Getoor L, et al. Knowledge graph identification//Proceedings of the International Semantic Web Conference. Sydney, Australia, 2013: 542-557

[2] Lehmann J, Isele R, Jakob M, et al. DBpedia — A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web, 2015, 6(2): 167-195

[3] Suchanek F M, Kasneci G, Weikum G. YAGO: A core of semantic knowledge//Proceedings of the 16th International Conference on World Wide Web. Banff, Canada, 2007: 697-706

[4] Vrandečić D, Krötzsch M. Wikidata: A free collaborative knowledgebase. Communications of the ACM, 2014, 57(10): 78-85

[5] Bollacker K, Evans C, Paritosh P, et al. Freebase: A collaboratively created graph database for structuring human knowledge//Proceedings of the International Conference on Management of Data. Vancouver, Canada, 2008: 1247-1250

[6] Xu B, Xu Y, Liang J, et al. CN-DBpedia: A never-ending Chinese knowledge extraction system//Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems. Arras, France, 2017: 428-438

[7] Niu X, Sun X, Wang H, et al. Zhishi.me-weaving Chinese linking open data//Proceedings of the International Semantic Web Conference. Bonn, Germany, 2011: 205-220

[8] Heath T, Bizer C. Linked data: Evolving the Web into a global data space. Synthesis Lectures on the Semantic Web: Theory and Technology, 2011, 1(1): 1-136

[9] Newman M E J. The structure and function of complex networks. SIAM Review, 2003, 45(2): 167-256

[10] Peng B, Zhang L, Zhang D. A survey of graph theoretical approaches to image segmentation. Pattern Recognition, 2013, 46(3): 1020-1038

[11] Kahng A B, Lienig J, Markov I L, et al. VLSI Physical Design: From Graph Partitioning to Timing Closure. Heidelberg, Germany: Springer Science & Business Media, 2011

[12] Boyd D M, Ellison N B. Social network sites: Definition, history, and scholarship. Journal of Computer-Mediated Communication, 2007, 13(1): 210-230

[13] Liben-Nowell D, Novak J, Kumar R, et al. Geographic routing in social networks. Proceedings of the National Academy of Sciences, 2005, 102(33): 11623-11628

[14] Alon U. Biological networks: The tinkerer as an engineer. Science, 2003, 301(5641): 1866-1867

[15] Camilus K S, Govindan V K. A review on graph based segmentation. International Journal of Image, Graphics & Signal Processing, 2012, 4(5)

[16] Bui T N, Jones C. Finding good approximate vertex and edge partitions is NP-hard. Information Processing Letters, 1992, 42(3): 153-159

[17] Buluç A, Meyerhenke H, Safro I, et al. Recent advances in graph partitioning//Kliemann L, Sanders P, eds. Algorithm Engineering. Cham: Springer, 2016: 117-158

[18] McCune R R, Weninger T, Madey G. Thinking like a vertex: A survey of vertex-centric frameworks for large-scale distributed graph processing. ACM Computing Surveys, 2015, 48(2): 25

[19] Verma S, Leslie L M, Shin Y, et al. An experimental comparison of partitioning strategies in distributed graph processing. Proceedings of the VLDB Endowment, 2017, 10(5): 493-504

[20] Safro I, Sanders P, Schulz C. Advanced coarsening schemes for graph partitioning. Journal of Experimental Algorithmics, 2015, 19(1): 2-16

[21] Abbas Z, Kalavri V, Carbone P, et al. Streaming graph partitioning: An experimental study. Proceedings of the VLDB Endowment, 2018, 11(11): 1590-1603

[22] Pacaci A, Özsu M T. Experimental analysis of streaming algorithms for graph partitioning//Proceedings of the International Conference on Management of Data. Amsterdam, The Netherlands, 2019: 1375-1392

[23] Yu Ge, Gu Yu, Bao Yu-Bin, et al. Large scale graph data processing on cloud computing environments. Chinese Journal of Computers, 2011, 34(10): 1753-1767(in Chinese)
(于戈，谷峪，鲍玉斌等. 云计算环境下的大规模图数据处理技术. 计算机学报，2011，34(10)：1753-1767)

[24] Wang Xin, Zou Lei, Wang Chao-Kun, et al. Research on knowledge graph data management: A survey. Journal of Software, 2019, 30(7): 2139-2174(in Chinese)
(王鑫，邹磊，王朝坤等. 知识图谱数据管理研究综述. 软件学报，2019，30(7)：2139-2174)

[25] Donath W E, Hoffman A J. Lower bounds for the partitioning of graphs. IBM Journal of Research and Development, 1973, 17(5): 420-425

[26] Fiedler M. Algebraic connectivity of graphs. Czechoslovak Mathematical Journal, 1973, 23(2): 298-305

[27] Fiedler M. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. Czechoslovak Mathematical Journal, 1975, 25(4): 619-633

[28] Simon H D. Partitioning of unstructured problems for parallel processing. Computing Systems in Engineering, 1991, 2(2): 135-148

[29] Barnard S T, Simon H D. Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. Concurrency: Practice and experience, 1994, 6(2): 101-117

[30] Chan P K, Schlag M D F, Zien J Y. Spectral k-way ratio-cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1994, 13(9): 1088-1096

[31] Hendrickson B, Leland R. An improved spectral graph partitioning algorithm for mapping parallel computations. SIAM Journal on Scientific Computing, 1995, 16(2): 452-469

[32] Williams R D. Performance of dynamic load balancing algorithms for unstructured mesh calculations. Concurrency: Practice and Experience, 1991, 3(5): 457-481

[33] Farhat C, Lesoinne M. Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics. International Journal for Numerical Methods in Engineering, 1993, 36(5): 745-764

[34] Miller G L, Teng S H, Vavasis S A. A unified geometric approach to graph separators//Proceedings of the 32nd Annual Symposium on Foundations of Computer Science. San Juan, Puerto Rico, 1991: 538-547

[35] Gilbert J R, Miller G L, Teng S H. Geometric mesh partitioning: Implementation and experiments. SIAM Journal on Scientific Computing, 1998, 19(6): 2091-2110

[36] Pilkington J R, Baden S B. Partitioning with spacefilling curves. Department of Computer Science and Engineering, UC San Diego: Technical Report: CS94-349, 1994

[37] Hungershöfer J, Wierum J M. On the quality of partitions based on space-filling curves//Proceedings of the International Conference on Computational Science. Amsterdam, The Netherlands, 2002: 36-45

[38] Kirmani S, Raghavan P. Scalable parallel graph partitioning//Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. Denver, USA, 2013: 51-61

[39] Land A H, Doig A G. An automatic method of solving discrete programming problems. Econometrica, 1960, 28(3): 497-520

[40] Brunetta L, Conforti M, Rinaldi G. A branch-and-cut algorithm for the equicut problem. Mathematical Programming, 1997, 78(2): 243-263

[41] Lisser A, Rendl F. Graph partitioning using linear and semidefinite programming. Mathematical Programming, 2003, 95(1): 91-101

[42] Armbruster M, Fügenschuh M, Helmberg C, et al. A comparative study of linear and semidefinite branch-and-cut methods for solving the minimum graph bisection problem//Proceedings of the International Conference on Integer Programming and Combinatorial Optimization. Bertinoro, Italy, 2008: 112-124

[43] Karisch S E, Rendl F, Clausen J. Solving graph bisection problems with semidefinite programming. INFORMS Journal on Computing, 2000, 12(3): 177-191

[44] Hager W W, Krylyuk Y. Graph partitioning and continuous quadratic programming. SIAM Journal on Discrete Mathematics, 1999, 12(4): 500-523

[45] Hager W W, Phan D T, Zhang H. An exact algorithm for graph partitioning. Mathematical Programming, 2013, 137(1-2): 531-556

[46] Delling D, Goldberg A V, Razenshteyn I, et al. Exact combinatorial branch-and-bound for graph bisection//Proceedings of the Fourteenth Workshop on Algorithm Engineering and Experiments. Westin Miyako, Japan, 2012: 30-44

[47] Delling D, Werneck R F. Better bounds for graph bisection//Proceedings of the European Symposium on Algorithms. Ljubljana, Slovenia, 2012: 407-418

[48] Kernighan B W, Lin S. An efficient heuristic procedure for partitioning graphs. The Bell system technical journal, 1970, 49(2): 291-307

[49] Fiduccia C M, Mattheyses R M. A linear-time heuristic for improving network partitions//Proceedings of the 19th Design Automation Conference. Las Vegas, USA, 1982: 175-181

[50] Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on scientific Computing, 1998, 20(1): 359-392

[51] Osipov V, Sanders P. n-level graph partitioning//Proceedings of the European Symposium on Algorithms. Liverpool, UK, 2010: 278-289

[52] Hendrickson B, Leland R. A multi-level algorithm for partitioning graphs. Proceedings Supercomputing, 1995, 95(28): 1-14

[53] Holtgrewe M, Sanders P, Schulz C. Engineering a scalable high quality graph partitioner//Proceedings of the IEEE International Symposium on Parallel & Distributed Processing. Atlanta, USA, 2010: 1-12

[54] Avis D. A survey of heuristics for the weighted matching problem. Networks, 1983, 13(4): 475-493

[55] Drake D E, Hougardy S. A simple approximation algorithm for the weighted matching problem. Information Processing Letters, 2003, 85(4): 211-213

[56] Maue J, Sanders P. Engineering algorithms for approximate weighted matching//Proceedings of the International Workshop on Experimental and Efficient Algorithms. Rome, Italy, 2007: 242-255

[57] Karypis G, Kumar V. Multilevel k-way partitioning scheme for irregular graphs. Journal of Parallel and Distributed Computing, 1998, 48(1): 96-129

[58] Karypis G, Kumar V. Multilevel k-way hypergraph partitioning. VLSI Design, 2000, 11(3): 285-300

[59] Huang J, Abadi D J, Ren K. Scalable SPARQL querying of large RDF graphs. Proceedings of the VLDB Endowment, 2011, 4(11): 1123-1134

[60] Hose K, Schenkel R. WARP: Workload-aware replication and partitioning for RDF//Proceedings of the International Conference on Data Engineering Workshops. Brisbane, Australia, 2013: 1-6

[61] Zhang X, Chen L, Tong Y, et al. EAGRE: Towards scalable I/O efficient SPARQL query evaluation on the cloud//Proceedings of the International Conference on Data Engineering. Brisbane, Australia, 2013: 565-576

[62] Sanders P, Schulz C. Engineering multilevel graph partitioning algorithms//Proceedings of the European Symposium on Algorithms. Saarbruecken, Germany, 2011: 469-480

[63] Sanders P, Schulz C. Distributed evolutionary graph partitioning//Proceedings of the Fourteenth Workshop on Algorithm Engineering and Experiments. Westin Miyako, Japan, 2012: 16-29

[64] Safro I, Ron D, Brandt A. Multilevel algorithms for linear ordering problems. Journal of Experimental Algorithmics, 2009, 13(1): 4-24

[65] Meyerhenke H, Monien B, Schamberger S. Accelerating shape optimizing load balancing for parallel fem simulations by algebraic multigrid//Proceedings of the International Parallel & Distributed Processing Symposium. Rhodes Island, Greece, 2006: 10-20

[66] Chevalier C, Safro I. Comparison of coarsening schemes for multilevel graph partitioning//Proceedings of the International Conference on Learning and Intelligent Optimization. Trento, Italy, 2009: 191-205

[67] Ron D, Wishko-Stern S, Brandt A. An algebraic multigrid based algorithm for bisectioning general graphs. Department of Computer Science and Applied Mathematics, the Weizmann Institute of Science, Rehovot, Israel: Technical Report: MCS05-01, 2005

[68] Chen J, Safro I. Algebraic distance on graphs. SIAM Journal on Scientific Computing, 2011, 33(6): 3468-3490

[69] Lafon S, Lee A B. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28(9): 1393-1403

[70] Meyerhenke H, Monien B, Sauerwald T. A new diffusion-based multilevel algorithm for computing graph partitions of very high quality//Proceedings of the International Symposium on Parallel and Distributed Processing. Miami, USA, 2008: 1-13

[71] Meyerhenke H. Shape optimizing load balancing for MPI-parallel adaptive numerical simulations. Graph Partitioning and Graph Clustering, 2012, 588(1): 67-81

[72] Karypis G, Kumar V. Parallel multilevel series k-way partitioning scheme for irregular graphs. SIAM Review, 1999, 41(2): 278-300

[73] Stanton I, Kliot G. Streaming graph partitioning for large distributed graphs//Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Beijing, China, 2012: 1222-1230

[74] Xie C, Yan L, Li W J, et al. Distributed power-law graph computing: Theoretical and empirical analysis//Proceedings of the Advances in Neural Information Processing Systems. Montreal, Canada, 2014: 1673-1681

[75] Jain N, Liao G, Willke T L. GraphBuilder: Scalable graph ETL framework//Proceedings of the 1st International Workshop on Graph Data Management Experiences and Systems. New York, USA, 2013: 4-10

[76] Gonzalez J E, Low Y, Gu H, et al. PowerGraph: Distributed graph-parallel computation on natural graphs. OSDI, 2012, 12(1): 2

[77] Petroni F, Querzoni L, Daudjee K, et al. HDRF: Stream-based partitioning for power-law graphs//Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. Melbourne, Australia, 2015: 243-252

[78] Tsourakakis C, Gkantsidis C, Radunovic B, et al. FENNEL: Streaming graph partitioning for massive scale graphs//Proceedings of the 7th ACM International Conference on Web Search and Data Mining. New York, USA, 2014: 333-342

[79] Nishimura J, Ugander J. Restreaming graph partitioning: Simple versatile algorithms for advanced balancing//Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Chicago, USA, 2013: 1106-1114

[80] Chen R, Shi J, Chen Y, et al. PowerLyra: Differentiated graph computation and partitioning on skewed graphs. ACM Transactions on Parallel Computing, 2019, 5(3): 13

[81] Pellegrini F, Roman J. Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs//Proceedings of the International Conference on High-Performance Computing and Networking. Brussels, Belgium, 1996: 493-498

[82] Chevalier C, Pellegrini F. PT-Scotch: A tool for efficient parallel graph ordering. Parallel Computing, 2008, 34(6-8): 318-331

[83] Manne F, Bisseling R H. A parallel approximation algorithm for the weighted maximum matching problem//Proceedings of the International Conference on Parallel Processing and Applied Mathematics. Gdansk, Poland, 2007: 708-717

[84] Toulouse M, Thulasiraman K, Glover F. Multi-level cooperative search: A new paradigm for combinatorial optimization and an application to graph partitioning//Proceedings of the European Conference on Parallel Processing. Toulouse, France, 1999: 533-542

[85] Walshaw C. Multilevel refinement for combinatorial optimisation problems. Annals of Operations Research, 2004, 131(1-4): 325-372

[86] Sanders P, Schulz C. Think locally, act globally: Highly balanced graph partitioning//Proceedings of the International Symposium on Experimental Algorithms. Rome, Italy, 2013: 164-175

[87] Meyerhenke H, Sanders P, Schulz C. Parallel graph partitioning for complex networks. IEEE Transactions on Parallel and Distributed Systems, 2017, 28(9): 2625-2638

[88] Rahimian F, Payberah A H, Girdzijauskas S, et al. JA-BE-JA: A distributed algorithm for balanced graph partitioning//Proceedings of the 7th International Conference on Self-Adaptive and Self-Organizing Systems. Trento, Italy, 2013: 51-60

[89] Rahimian F, Payberah A H, Girdzijauskas S, et al. Distributed vertex-cut partitioning//Proceedings of the International Conference on Distributed Applications and Interoperable Systems. Berlin, Germany, 2014: 186-200

[90] Xu N, Chen L, Cui B. LogGP: A log-based dynamic graph partitioning method. Proceedings of the VLDB Endowment, 2014, 7(14): 1917-1928

[91] Nicoara D, Kamali S, Daudjee K, et al. Hermes: Dynamic partitioning for distributed social network graph databases//Proceedings of the Extending Database Technology. Brussels, Belgium, 2015: 25-36

[92] Vukotic A, Watt N, Abedrabbo T, et al. Neo4j in Action. New York, USA: Manning Publications Co., 2014

[93] Zheng A, Labrinidis A, Chrysanthis P K. Planar: Parallel lightweight architecture-aware adaptive graph repartitioning //Proceedings of the 32nd International Conference on Data Engineering. Helsinki, Finland, 2016: 121-132

[94] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. Communications of the ACM, 2008, 51(1): 107-113

[95] Malewicz G, Austern M H, Bik A J C, et al. Pregel: A system for large-scale graph processing//Proceedings of the International Conference on Management of Data. Indianapolis, USA, 2010: 135-146

[96] Gonzalez J E, Xin R S, Dave A, et al. GraphX: Graph processing in a distributed dataflow framework//Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation. Broomfield, USA, 2014: 599-613

[97] Guerrieri A, Montresor A. DFEP: Distributed funding-based edge partitioning//Proceedings of the European Conference on Parallel Processing. Vienna, Austria, 2015: 346-358

[98] Sakouhi C, Aridhi S, Guerrieri A, et al. DynamicDFEP: A distributed edge partitioning approach for large dynamic graphs//Proceedings of the 20th International Database Engineering & Applications Symposium. Montreal, Canada, 2016: 142-147

[99] Dai D, Zhang W, Chen Y. IOGP: An incremental online graph partitioning algorithm for distributed graph databases //Proceedings of the 26th International Symposium on High-Performance Parallel and Distributed Computing. Washington, USA, 2017: 219-230

[100] Margo D, Seltzer M. A scalable distributed graph partitioner. Proceedings of the VLDB Endowment, 2015, 8(12): 1478-1489

[101] Aydin K, Bateni M H, Mirrokni V. Distributed balanced partitioning via linear embedding//Proceedings of the 9th ACM International Conference on Web Search and Data Mining. San Francisco, USA, 2016: 387-396

[102] Wang Zhi-Gang, Gu Yu, Bao Yu-Bin, et al. OnFlyP: An online distributed partition algorithm for large scale graphs based on edge-exchange model. Chinese Journal of Computers, 2015, 38(9): 1838-1851(in Chinese)
(王志刚，谷峪，鲍玉斌等. OnFlyP：基于定向边交换的分布式在线大图划分算法. 计算机学报，2015，38(9)：1838-1851)

[103] Kim J, Hwang I, Kim Y H, et al. Genetic approaches for graph partitioning: A survey//Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation. Dublin, Ireland, 2011: 473-480

[104] Rolland E, Pirkul H, Glover F. Tabu search for graph partitioning. Annals of Operations Research, 1996, 63(2): 209-232

[105] Xu Q, Wang X, Wang J, et al. Semantic-aware partitioning on RDF graphs//Proceedings of the Asia-Pacific Web and Web-Age Information Management Joint Conference on Web and Big Data. Beijing, China, 2017: 149-157

[106] Zhao Xiang, Li Bo, Shang Hai-Chuan, et al. A revised BSP-based massive graph computation model. Chinese Journal of Computers, 2017, 40(1): 223-235(in Chinese)
(赵翔，李博，商海川等. 一种改进的基于 BSP 的大图计算模型. 计算机学报，2017，40(1)：223-235)

[107] Leng Fang-Ling, Liu Jin-Peng, Wang Zhi-Gang, et al. Edge Cluster Based Large Graph Partitioning and Iterative Processing in BSP. Journal of Computer Research and Development, 2015, 52(4): 960-971(in Chinese)
(冷芳玲，刘金鹏，王志刚等. BSP 模型下基于边聚簇的大图划分与迭代处理. 计算机研究与发展，2015，52(4)：960-971)

[108] Li Qi, Zhong Jiang, Li Xue. DyBGP: A dynamic-balanced algorithm for graph partitioning based on heuristic strategies. Journal of Computer Research and Development, 2017, 54(12): 2834-2840(in Chinese)
(李琪，钟将，李雪. 基于启发策略的动态平衡图划分算法. 计算机研究与发展，2017，54(12)：2834-2840)

[109] Zhu X, Ghahramani Z. Learning from labeled and unlabeled data with label propagation. Carnegie Mellon University, Pennsylvania: Technical Report: CMU-CALD-02-107, 2002

[110] Ugander J, Backstrom L. Balanced label propagation for partitioning massive graphs//Proceedings of the 6th ACM International Conference on Web Search and Data Mining. Rome, Italy, 2013: 507-516

[111] Wang L, Xiao Y, Shao B, et al. How to partition a billion-node graph//Proceedings of the 30th International Conference on Data Engineering. Chicago, USA, 2014: 568-579

[112] Vaquero L M, Cuadrado F, Logothetis D, et al. Adaptive partitioning for large-scale dynamic graphs//Proceedings of the 34th International Conference on Distributed Computing Systems. Calabria, Italy, 2014: 144-153

[113] Martella C, Logothetis D, Loukas A, et al. Spinner: Scalable graph partitioning in the cloud//Proceedings of the 33rd International Conference on Data Engineering. San Diego, USA, 2017: 1083-1094

[114] Meyerhenke H, Sanders P, Schulz C. Partitioning complex networks via size-constrained clustering//Proceedings of the International Symposium on Experimental Algorithms. Copenhagen, Denmark, 2014: 351-363

[115] Akhter A, Ngomo A C N, Saleem M. An empirical evaluation of RDF graph partitioning techniques//Proceedings of the Knowledge Engineering and Knowledge Management. Nancy, France, 2018: 3-18

[116] Harbi R, Abdelaziz I, Kalnis P, et al. Accelerating SPARQL queries by exploiting hash-based locality and adaptive partitioning. The International Journal on Very Large Data Bases, 2016, 25(3): 355-380

[117] Peng P, Zou L, Chen L, et al. Adaptive distributed RDF graph fragmentation and allocation based on query workload. IEEE Transactions on Knowledge and Data Engineering, 2019, 31(4): 670-685

附　录.

查询语句：

LUBM：

Q1：SELECT ?X WHERE {
?X 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#Publication〉.
?X 〈tju:#publicationAuthor〉
〈http://www.Department0.University0.edu/AssistantProfessor0〉.}

Q2：SELECT ?X ?Y ?Z WHERE {
?X 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#GraduateStudent〉.
?Y 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#University〉.
?Z 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#Department〉.
?X 〈tju:#memberOf〉 ?Z.
?Z 〈tju:#subOrganizationOf〉 ?Y.
?X 〈tju:#undergraduateDegreeFrom〉 ?Y.}

Q3：SELECT ?X ?Y WHERE {
?X 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#UndergraduateStudent〉.
?X 〈tju:#takesCourse〉 ?Y.
〈http://www.Department0.University0.edu/AssociateProfessor0〉
〈tju:#teacherOf〉 ?Y.}

DBpedia：

DQ1：SELECT ?X ?Y ?Z WHERE {
?X 〈http://www.w3.org/2000/01/rdf-schema#seeAlso〉 ?Y.
?X 〈http://dbpedia.org/ontology/deathPlace〉 ?Z.}

DQ2：SELECT ?X2 ?X1 ?Y1 ?Y2 ?Z WHERE {
?X1 〈http://dbpedia.org/ontology/philosophicalSchool〉 ?Y1.
?Y1 〈http://www.w3.org/2000/01/rdf-schema#seeAlso〉 ?Z.
?X1 〈http://dbpedia.org/ontology/era〉 ?Y2.
?X2 〈http://dbpedia.org/ontology/mainInterest〉 ?Z.}

DQ3：SELECT ?X1 ?X2 ?Y1 ?Y2 ?Z WHERE {
?X1 〈http://dbpedia.org/ontology/philosophicalSchool〉 ?Y1.
?Y1 〈http://www.w3.org/2000/01/rdf-schema#seeAlso〉?Z.
?X2 〈http://dbpedia.org/ontology/mainInterest〉 ?Z.
?X1 〈http://dbpedia.org/ontology/mainInterest〉 ?Y2.
?Y2 〈http://dbpedia.org/ontology/field〉 ?Z.}

挖掘语句：

LUBM：

DM1：SELECT ?X1 ?X2 ?X3 ?Y1 ?Y2 ?Y3 ?Z1 ?Z2 ?Z3
WHERE {
?X1 〈tju:#name〉 ?Y1.
?X1 〈tju:#teacherOf〉 ?Z1.
?X1 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#FullProfessor〉.
?Z2 〈tju:#publicationAuthor〉 ?X1.
?X2 〈tju:#name〉 ?Y2.
?X2 〈tju:#takesCourse〉 ?Z1.
?X2 〈tju:#memberOf〉 ?Z3.
?X3 〈tju:#name〉 ?Y3.
?X3 〈tju:#teachingAssistantOf〉 ?Z.
?Z1 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#type〉
〈tju:#Course〉.}

DBpedia：

DM2：SELECT ?X1 ?X2 ?X3 ?Y1 ?Y2 ?Y3 ?Y4 ?Z1 ?Z2
WHERE {
?X1 〈http://dbpedia.org/ontology/philosophicalSchool〉 ?Y1.
?Y1 〈http://www.w3.org/2000/01/rdf-schema#seeAlso〉?Z1.
?X2 〈http://dbpedia.org/ontology/mainInterest〉 ?Z1.
?X3 〈http://dbpedia.org/ontology/field〉 ?Z1.
?X1 〈http://dbpedia.org/ontology/mainInterest〉 ?Y2.
?Y2 〈http://www.w3.org/2000/01/rdf-schema#seeAlso〉 ?Z2.
?X3 〈http://dbpedia.org/ontology/birthPlace〉 ?Y3.
?X3 〈http://dbpedia.org/ontology/deathPlace〉 ?Y4.}

WANG Xin, Ph. D., professor. His main research interests include knowledge graph data management, graph database and large-scale knowledge processing.

CHEN Wei-Xue, M. S. candidate. Her main research interests include knowledge graph data management, graph database and large-scale knowledge processing.

YANG Ya-Jun, Ph. D., associate professor. His main research interests include graph database and graph data mining.

ZHANG Xiao-Wang, Ph. D., associate professor. His main research interests include database, artificial intelligence and knowledge graph.

FENG Zhi-Yong, Ph. D., professor. His main research interests include knowledge engineering and service computing.

Background

Knowledge graphs are important cornerstones of artificial intelligence, which contain rich graph structures and attribute information. Knowledge graphs can accurately describe various entities and connections in the real world. As the primary task of distributed processing of large-scale knowledge graphs, knowledge graph partitioning provides basic support for distributed storage, query processing, reasoning, and data mining. With the increasing scale of knowledge graphs and more requirements of distributed processing, knowledge graph partitioning faces new challenges. In recent years, distributed graph partitioning algorithms are commonly used when partitioning large-scale knowledge graphs. However, these algorithms ignore various attribute information of knowledge graphs. It is a challenging issue to make full use of rich attribute information of knowledge graphs in their partitioning tasks, which has become a hot topic in the current research.

There have been a large number of research works on graph partitioning, but most of which are based on the basic (un)directed graphs, and do not consider knowledge graphs. There does not yet exist a comprehensive survey of knowledge graph partitioning algorithms at present. In order to provide a systematic and comprehensive review on the research of knowledge graph partitioning, we introduce various algorithms currently available for knowledge graph partitioning, including basic graph partitioning algorithms, multilevel graph partitioning algorithms, streaming graph partitioning algorithms, distributed graph partitioning algorithms, and other types of graph partitioning algorithms. We first propose the definition of knowledge graphs and graph partitioning. Since we consider knowledge graphs as a general model for all four graph types, most basic graph partitioning algorithms can also be used to partition knowledge graphs. The basic graph partitioning algorithms include spectral partitioning algorithms, geometric partitioning algorithms, branch and bound algorithms, and KL and its derivative algorithms. Algorithms based on matching and aggregation are introduced in the category of multilevel graph partitioning algorithms. Three streaming graph partitioning algorithms, including hash, greedy and Fennel, are discussed. Representative distributed graph partitioning algorithms, i.e., KaPPa, JA-BE-JA, and lightweight repartitioning, are introduced. Graph partitioning algorithms using label propagation and query workload are also presented. Meanwhile, we compare the performance of five representative knowledge graph partitioning algorithms in partitioning effects, query processing, and graph data mining. Finally, based on the analysis and comparison of existing methods, we summarize the main challenges of knowledge graph partitioning, and look forward to the future research direction.

This work is supported by the National Key Research and Development Program of China "Key Technologies and Systems of Distributed Knowledge Graph Data Management (2019YFE0198600)" and the National Natural Science Foundation of China "Research on Key Technology of Distributed Storage and Query on Large-Scale Knowledge Graphs (61972275)".