一种基于 MapReduce 的不确定图上的相似性连接方法

缪丰羽[1] 王宏志[2] 阮群生[3]
(宁德师范学院信息与机电工程学院 福建 宁德 352100)[1]
(哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)[2] (厦门大学软件学院 福建 厦门 361000)[3]

摘 要 相比于确定图上的相似性连接,不确定图上的相似性连接通常具有更大的实际应用价值以及计算复杂性。文中研究了基于 MapReduce 分布式编程框架的不确定图上的相似性连接问题,提出了基于概率和的 Map 方剪枝和 Reduce 方剪枝的两种剪枝策略。Map 方剪枝策略在映射过程中过滤掉了不可能具有相似图的不确定图。Reduce 方剪枝策略用于减少约减过程中的候选图对。基于这两种剪枝策略,文中提出了一种基于 MapReduce 框架的不确定图上的相似性连接算法 MUGSJoin。实验结果证明,该算法与同类算法相比具有更好的性能和可扩展性。

关键词 MapReduce,不确定图,相似性连接
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.12.048

Method of Similarity Join on Uncertain Graphs Using MapReduce

MIAO Feng-yu[1] WANG Hong-zhi[2] RUAN Qun-sheng[3]
(College of Information & Mechanical and Electrical Engineering, Ningde Normal University, Ningde, Fujian 352100, China)[1]
(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)[2]
(College of Software, Xiamen University, Xiamen, Fujian 361000, China)[3]

Abstract Compared to the similarity join on the deterministic graph, the similarity join on the uncertain graph usually has greater practical application value and higher computational complexity. This paper studied the similarity join on uncertain graph databases based on MapReduce distributed programming framework, and proposed two pruning strategies using the existence probability, namely Map Side Pruning and Reduce Side Pruning. The Map Side Pruning can filter out the uncertain graphs at the Map side which have no chance to be similar with any other uncertain graph at the Map side. The Reduce Side Pruning can reduce the candidate pairs at Reduce side in the process of reduction. Based on the above pruning approaches, this paper proposed a similarity join algorithm MUGSJoin on uncertain graph databases based on MapReduce. The experiment results show that the proposed approach has much better performance and scalability than the similar algorithms.

Keywords MapReduce, Uncertain graph, Similarity join

1 引言

图的相似性连接作为图数据挖掘以及图信息集成中的一种重要的基本操作已被广泛应用,尤其在图数据的预处理阶段,其典型应用包括结构数据清理和近复本结构检测等[1]。例如,在图信息集成中,多个自主的图数据库对于相同的真实世界实体可能具有不同的结构描述,图的相似性连接可以找出这些图数据库中差别小于给定阈值的相似图对(事实上该图对可能代表着同一实体),以进行后期的数据合并或删除。对于图的相似性连接目前已经有了许多的研究成果[1-4]。

图的相似性连接可以分为两类:一类是给定两个图数据库 R 和 S,以及一个相似性阈值 λ,返回 R 和 S 中相似性大于或等于 λ 的所有图对 (r_i, s_j),其中 $r_i \in R, s_j \in S$;另一类是给定一个图数据库 R 以及一个相似性阈值 λ,返回 R 中相似性大于或等于 λ 的所有图对 (r_i, r_j),其中 $r_i, r_j \in R$。两类问题的解法相似,本文主要讨论第二类相似性连接。

目前为止,图的相似性连接问题大部分是基于确定图进行的。然而由于数据源的多样性和不一致性,以及数据在提取过程中的技术精度等,在实际应用中,图数据经常会引入不确定信息,因此不确定图上的相似性连接成为了图数据管理中亟待解决的问题[5]。例如,在社会网络中,由于更新不及时,可能出现相同的社区具有若干不同的关系或标签,若采用

到稿日期:2017-12-28 返修日期:2018-03-16 本文受国家自然科学基金重点项目(U1509216),国家重点研发计划项目(2016YFB1000703),国家自然科学基金面上项目(61472099,61602129),福建省自然科学基金面上项目(2018J01555),福建省教育厅中青年项目(JAT170653)资助。
缪丰羽(1983-),女,硕士,讲师,主要研究方向为图数据管理、XML 数据库查询优化,E-mail:feathen1983@163.com(通信作者);王宏志(1978-),男,博士,副教授,博士生导师,主要研究方向为大数据、数据质量、图数据管理;阮群生(1979-),男,博士生,副教授,主要研究方向为大数据、人工智能。

不确定图进行建模，则相似性连接结果可能会包含该社区对。这相比于确定图中的相似性连接更具有容错性，也更接近于人的分析判断方式[6]。

因此，研究不确定图上的相似性连接具有更加普遍的应用价值。相比于确定图上的相似性连接，不确定图上的相似性连接通常具有更大的计算量和更高的算法复杂度，因此该领域的研究具有更大的挑战性。

由于图的相似性连接问题的计算量较大，大部分该领域的研究都选择采用近似算法来减少计算过程的中间数据，且测试数据的规模较小[1-4]。随着大数据时代的到来，现实生活中产生的数据的规模已达到了 ZB 级[5]。单机版的各种图数据算法已不能满足目前海量图数据的处理需要，人们开始越来越多地借助于并行计算平台并开发各种相应的图数据并行处理算法[7-10]。

MapReduce[11]是一种能够有效处理大规模数据集的分布式编程模型。随着各种信息处理领域数据量的急速增加，这种海量数据的并行处理编程框架开始被广泛使用。MapReduce 框架使用计算机集群来对海量数据进行并行处理，把数据存储在集群上的分布式文件系统中。基于 MapReduce 框架的大规模数据相似性连接问题，主要包括对集合相似性连接、字符串相似性连接和向量相似性连接的研究[5]。近年来也出现了一些基于 MapReduce 框架的图相似性连接的研究[11-13]，但由于不确定图的相似性连接问题的复杂性，目前还没有关于不确定图上相似性连接问题的并行算法。

本文的主要贡献有：1)设计了基于 MapReduce 框架的不确定图相似性连接的基本算法，并分析了该算法的网络传输代价和时间复杂度；2)提出了基于概率和的 Map 方剪枝和 Reduce 方剪枝两种剪枝策略；3)设计了基于以上剪枝策略的不确定图的相似性连接算法 MUGSJoin，并分析了该算法的网络传输代价和时间复杂度。

本文第 2 节讨论了相关工作；第 3 节给出了不确定图的相似性连接的形式化定义；第 4 节提出了基于 MapReduce 框架的不确定图的相似性连接的基本算法；第 5 节介绍了基于概率和的过滤策略，以及基于这些过滤策略的算法实现，包括 Map 方剪枝的 Hadoop 连接、Reduce 方剪枝的 Hadoop 连接和同时使用 Map 方剪枝和 Reduce 方剪枝的混合连接；第 6 节给出了使用剪枝策略后的相似性连接算法 MUGSJoin；第 7 节给出了实验分析；最后总结全文并展望未来。

2 相关工作

本节主要介绍本文的相关工作，包括 MapReduce 框架、基于 MapReduce 框架的图的相似性连接和不确定图上的相似性连接。

2.1 MapReduce

MapReduce 是由 Google 于 2004 年提出的一种编程模型，通常用于大规模数据集(大于 1TB)的并行计算。Map(映射)和 Reduce(归约)的概念，以及它们的主要思想和特性，都是从函数式编程语言和矢量编程语言里借用来的。它为编程人员在不会分布式并行编程的情况下，将自己的程序在分布式系统上运行提供了极大的方便。MapReduce 编程模型分为两个步骤：Map 和 Reduce。其中 Map 过程负责处理每个输入的键值对(k_1,v_1)，将其映射为一组中间的键值对 $list(k_2,v_2)$。Reduce 过程负责将新的键值对$(k_2,list(v_2))$作为输入，产生另一个键值对列表 $list(k_3,v_3)$。这些输入和输出数据被保存在分布式文件系统中，如 Google File System[14]。

目前有许多研究工作都利用 MapReduce 框架对原有的算法进行扩展。文献[15-17]扩展了 MapReduce 模式来处理传统的 DBMS 中的连接操作。其中，文献[15]提出了一些在 MapReduce 环境下优化多重连接的方法；文献[16]主要利用 MapReduce 框架处理 θ 连接；文献[17]在原始的 MapReduce 框架上做了一些修改使得其上的连接操作可以更加有效。还有一些工作研究了在 MapReduce 环境下处理复杂的连接，如模糊连接[18]、海量高维数据集上的相似性连接[19]和 top-k 相似性连接[20]。

2.2 基于 MapReduce 框架的图的相似性连接

在单机上执行的图的相似性连接已有很多成果[1-4]。随着大数据时代的到来，这些相似性连接方法已经不能满足海量数据的处理要求，因此，各种借助 MapReduce 编程模式的并行算法应运而生[11-13]。文献[13]提出了一种 MapReduce 框架下的图的相似性连接算法。该算法包含了一种适用于 q-gram 字母表的可扩展前缀过滤策略；一种有效减少候选图对的策略，可以大大降低算法的通信代价；一种二轮数据访问方案，用于减少数据访问开销。文献[11]提出了一种名为 MGSJoin 的可扩展算法，该算法基于过滤-验证框架来进行有效的图的相似性连接。MGSJoin 算法通过计算图特征码的重叠部分来过滤出不可能匹配的图对；此外，该算法还集成了多种连接策略来增强验证效果。文献[11]的作者还提出了一种基于 MapReduce 的方法来进行 GED 的计算。鉴于文献[11]提出的算法在过滤阶段键值对的数量庞大，文献[12]引入了光谱布隆过滤技术来对算法进行改进，并设计了 BMGSJoin 算法来减少过滤阶段的键值对。

2.3 不确定图上的相似性连接

上述文献的研究对象都是确定图。目前为止，对于不确定图上的相似性连接的研究相对较少。文献[2]研究了大规模不确定图的相似性查询，其中包含子图相似性查询和超图相似性查询。但这种包含查询与我们所研究的相似性连接并不相同。

3 不确定图上的相似性连接

本节首先介绍确定图的定义，并基于该定义给出了本文所讨论的不确定图的形式化定义和可能世界图的定义；最后给出不确定图的相似性连接的形式化定义。

定义 1(确定图)[2] 一个确定图 $g^c=(V,E,\Sigma,L)$。其中，V 是顶点集；E 是边集；Σ 是标签集；$L:V\cup E->\Sigma$ 是一个函数，该函数将标签映射到顶点和边上。如果 g^c 中的某些边射入同一个顶点或者组成一个三角形，则称它们为邻边，记为 ne。

定义 2(不确定图)[2] 一个不确定图 $g=(g^c,X_E)$，其中 g^c 为一个确定图，X_E 是一个由 E 索引的二进制随机变量集。X_E 中的元素 x_e 取值为 0 或者 1，用来表示边 e 存在的概率。

图中的每个邻边集都有一张对应的联合概率分布表(JPT),表中给出了该邻边集中邻边的所有随机变量的分布情况以及相应的概率值 Pr。

例如,图 1 所示的不确定图有 3 个邻边集,分别是(e_1,e_2,e_3),(e_3,e_4,e_5)和(e_4,e_5,e_6)。对应的联合概率分布表为图 2 中的 JPT1,JPT2 和 JPT3。JPT1 中的第 2 行数据 $Pr(e_1=1,e_2=1,e_3=0)=0.2$ 表示边 e_1 和 e_2 存在且边 e_3 不存在的概率为 0.2。

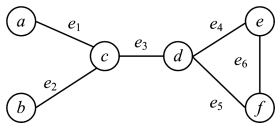

图 1　不确定图

Fig. 1　Uncertain graph

e_1	e_2	e_3	Pr
1	1	1	0.2
1	1	0	0.2
1	0	1	0.1
1	0	0	0.1
0	1	1	0.1
0	1	0	0.1
0	0	1	0.1
0	0	0	0.1

JPT1

e_3	e_4	e_5	Pr
1	1	1	0.30
1	1	0	0.40
1	0	1	0.05
1	0	0	0.15
0	1	1	0.03
0	1	0	0.02
0	0	1	0.04
0	0	0	0.01

JPT2

e_4	e_5	e_6	Pr
1	1	1	0.25
1	1	0	0.15
1	0	1	0.05
1	0	0	0.15
0	1	1	0.08
0	1	0	0.20
0	0	1	0.04
0	0	0	0.08

JPT3

图 2　JPTs

Fig. 2　JPTs

定义 3(可能世界图)[2]　一个可能世界图 $g'=(V',E',\Sigma',L')$是一个确定图,它是一个不确定图 $g=((V,E,\Sigma,L),X_E)$的实例,其中 $V'=V,E'\subseteq E,\Sigma'\subseteq\Sigma$。

例如,将 JPT1 中的元组($e_1=1,e_2=1,e_3=1$)同 JPT2 中的元组($e_3=1,e_4=1,e_5=1$)以及 JPT3 中的元组($e_4=1,e_5=1,e_6=1$)进行联接,得到图 1 的一个可能世界图,它表示图 1 中所有的顶点和边都存在。该可能世界图的概率值为 $0.2*0.3*0.25=0.015$。图 1 的可能世界图的个数为 3 个联合分布表进行联接得到的表的元组数。

定义 4(不确定图的相似性连接,UGSJ)　给定一个不确定图数据集 $G=\{g_1,g_2,\cdots,g_N\}$,一个相似性阈值 $\gamma\in(0,1]$。一个概率阈值 $\alpha\in(0,1]$,不确定图的相似性连接指的是从 G 中找出所有 Pr 值大于或等于 α 的图对(g_i,g_j)。

$$UGSJ=\{(g_i,g_j)\mid g_i\in G \text{ and } g_j\in G \text{ and } Pr\{Mc(g_i,g_j)\geqslant\gamma\}\geqslant\alpha\} \tag{1}$$

其中:

$$Pr\{Mc(g_i,g_j)\geqslant\gamma\}=\sum_{\forall g'\in g_i}\sum_{\forall g''\in g_j}p(g')*p(g'')*\chi(Mc(g',g'')\geqslant\gamma) \tag{2}$$

其中,g'为 g_i 的可能世界图,g''为 g_j 的可能世界图,$p(g')$为 g'的概率,$p(g'')$为 g''的概率。

$$\chi(Mc(g',g'')\geqslant\gamma)=\begin{cases}1, & \text{if } Mc(g',g'')\geqslant\gamma\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

其中,$Mc(g',g'')$表示可能世界图 g'和 g''的相似程度。本文所采用的确定图的相似度计算方法为文献[21]中提出的方法。

4　基本算法

本节介绍基于 MapReduce 的不确定图相似性连接的基本算法,并分析这种算法实现的网络传输代价和时间复杂度。

本算法的基本思想是将相似性连接过程拆分成几个不同的阶段,再将每个阶段分解成若干个子任务,并将其交由不同的计算结点并行完成,最后对并行计算结果进行合并。这种编程模式可以将计算任务分配到多台物理机上,从而提高系统处理大规模数据的可扩展性,使处理大规模的图结构数据成为可能。

4.1　基本算法

基于 MapReduce 的不确定图相似性连接的基本算法需要两个 MapReduce 作业。

作业 1 中,每个 Map 函数输入的是一个不确定图 g。输入的(key,$value$)对中的 key 是 g 的 id,$value$ 是 g 的邻接表(由顶点集 V、边集 E 和每条边上的存在概率 P 构成),输出的(key,$value$)对中的 key 是 g 的 id,$value$ 是 g 对应的一组可能世界图 g(PWG)。为了生成候选的不确定图对,可以在 Mapper 类中增加一个全局变量 $list(g)$来存储所有的不确定图的 id,并在 Close 函数中输出用不确定图的 id 表示的候选图对。该阶段的 Mapper 类为:

```
Class Mapper
  Method Initialize
    list(g)←∅
  Method Map (gid,g)
    list(JPT)=generateJPTs(V,E,P)
    g(PWG)=Join(list(JPT))
    list(g) ← list(g)∪gid
    输出(gid,g(PWG))
  Method Close
  输出候选图对(gᵢid,gⱼid)
```

其中,generateJPTs 函数为不确定图 g 生成一组 JPT;Join 函数将 g 的所有 JPT 进行联接,生成 g 的一组可能世界图 g(PWG),该函数的具体实现算法参考文献[6]。

在 Reduce 函数中生成包含图 id 和 PWG 列表的候选图对($g_j$id,g_i(PWG))。Reducer 类为:

```
Class Reducer
    Method Reduce(gᵢid,gᵢ(PWG)/list(gⱼid))
      for gⱼid ∈list(gⱼid) do
        输出(gⱼid,gᵢ(PWG))
```

作业 2 中,Map 函数的输入为包含图 id 和 PWG 列表的候选图对($g_j$id,g_i(PWG))或者用 id 和相应的 PWG 列表表示的不确定图($g_j$id,g_j(PWG)),并只需要原样输出。在 Reduce 函数中,对于每个不确定图对($g_i$id,$g_j$id),根据定义 4 求 Pr 值,并将 Pr 值不小于 α 的不确定图对作为相似性连接结果输出。作业 2 为:

```
Class Mapper
    Method Map ((gⱼid,gᵢ(PWG))/(gⱼid,gⱼ(PWG)))
      输出(gⱼid,gᵢ(PWG)/gⱼ(PWG))
Class Reducer
    Method Reduce(gⱼid,gⱼ(PWG)/list(gᵢ(PWG)))
```

```
for g_i(PWG)∈list(g_i(PWG)) do
  Pr←0
  对于 g_i 以及 g_j 中所有的可能世界图对(g′,g″),计算相应的
  Mc(g′,g″)
  if (Mc(g′,g″)>=γ) then
    将 g′和 g″的概率乘积累加进 Pr 中
  if (Pr>=α) then
    输出(g_iid,g_jid)
```

4.2 复杂度分析

在分析算法复杂度时,主要考虑 MapReduce 作业的 Map 阶段、Reduce 阶段以及 Shuffle 阶段的复杂性。其中 Map 阶段考虑从分布式文件系统 DFS 中读取数据的 IO 开销及时间复杂度;Reduce 阶段主要考虑将数据写入 DFS 的 IO 开销及时间复杂度;Shuffle 阶段考虑网络数据传输的复杂度。

在作业 1 中,Map 函数需要读入 G 中的所有不确定图。设 G 中不确定图的数目为 N,则 Map 函数的 IO 开销为 $O(N)$。对于每个 Map 函数,首先为不确定图 g 生成一组 JPT,并将其联接生成一组可能世界图 $g(PWG)$。设每个不确定图的顶点数为 n,边数为 m,则构造一组 JPT 的时间复杂度为 $O(n+2m)$。设 g 生成的 JPT 的数目为 t,每个 JPT 的行数为 r,则生成一组可能世界图的时间复杂度为 $O(2^{tr})$。设 g 的可能世界图数目为 p,则输出($g$id,$g(PWG)$)的时间复杂度为 $O(p)$。最后,Close 函数的时间复杂度为 $O(1/2*N^2)$。因此,作业 1 的 Map 阶段的时间复杂度总和为 $O(N*(n+2m+2^{tr}+p)+1/2*N^2)$。Shuffle 阶段的传输复杂度为 Map 阶段输出的所有键值对的数量,即 $O(p*N+1/2*N^2)$。每个 Reduce 函数输出包含某一不确定图 g_i 的所有候选图对。Reduce 阶段在最坏情况下的时间复杂度总和为 $O(p*N)$,IO 开销为 $O(1/2*p*N^2)$。

在作业 2 中,Map 函数读取作业 1 输出的所有键值对,因此 IO 开销为 $O(1/2*p*N^2)$。由于作业 2 仅是将输入作为结果输出,因此 Map 的时间复杂度和 Shuffle 的传输复杂度均为 $O(1/2*p*N^2)$。Reduce 函数根据定义 4 计算每个候选图对的 Pr 值,若 Pr 值不小于 α,则将该图对作为结果输出。计算 Pr 值的过程中,计算候选图对中每个可能世界图对的相似性所需的时间为 $O(n\log n+m)$,其计算方法及时间复杂度分析请参见文献[20]。设最后的结果图对数量为 C,则 Reduce 函数的总时间复杂度为 $O(1/2*p^2*N^2*(n\log n+m))$,IO 开销为 $O(C)$。

5 基于概率和的过滤策略

上述基本算法的主要思路是将不确定图数据集 G 中的不确定图映射到分布式计算结点上进行处理,先将每个不确定图转化成若干个可能世界图,再根据定义 4 找出 G 中所有满足条件的不确定图对。虽然该算法将数据和计算分布到了多个计算结点,较单机算法其可行性较高,但从复杂度分析来看,该算法的 IO 开销和时间复杂度也是相当大的。

在实际情况中,真正能够满足相似性阈值的图对很少。因此,我们设计了一组基于概率和的过滤策略,可以在算法运行的各个阶段进行有效过滤,以减少最后进行匹配率计算的可能世界图对,提高算法的运行效率。这一组过滤策略分为单图过滤策略和图对过滤策略,对应的函数实现分别为 Map 方剪枝的 Hadoop 连接和 Reduce 方剪枝的 Hadoop 连接。

5.1 单图过滤策略

对于给定的不确定图数据集 G,如果在算法运行初期,先筛选掉一部分不可能找到相似图的不确定图,就可以大幅减少候选图对,尽量避免后期的无效计算。

通过对第 3 节相关公式进行分析和推导,可以得出不确定图数据集 G 中任意不确定图上的一个定理。

定理 1 设 g_i 为不确定图数据集 G 中的任意不确定图,g'为 g_i 的可能世界图,$p(g')$为 g'的概率。若 g_i 满足条件:$\sum_{\forall g'\in g_i} p(g')<\alpha$,则可以判断 g_i 在 G 中没有满足条件的相似图,并将其剪枝。

证明:根据式(2)可得:

$$\begin{aligned}Pr\{Sim(g_i,g_j)\geqslant\gamma\}&=\sum_{\forall g'\in g_i}\sum_{\forall g''\in g_j}p(g')*p(g'')*\\&\quad\chi(Sim(g',g'')\geqslant\gamma)\\&\leqslant\sum_{\forall g'\in g_i}\sum_{\forall g''\in g_j}p(g')*p(g'')\\&=\sum_{\forall g'\in g_i}p(g')*\sum_{\forall g''\in g_j}p(g'')\end{aligned}$$

由可能世界图的性质可知,$\sum_{\forall g''\in g_j}p(g'')\leqslant 1$,因此:

$$\begin{aligned}Pr\{Sim(g_i,g_j)\geqslant\gamma\}&=\sum_{\forall g'\in g_i}p(g')*\sum_{\forall g''\in g_j}p(g'')\\&\leqslant\sum_{\forall g'\in g_i}p(g')\end{aligned}$$

定理得证。

根据上述定义得到单图上界过滤策略,若不确定图 g_i 的所有可能世界图的概率之和小于 α,则将其进行过滤。

设不确定图数据集 G 中的不确定图有 m 个,如果过滤掉 1 个不确定图,则可以减少 $m-1$ 个候选图对,如果过滤掉 2 个不确定图,则可以减少$(m-1)+(m-2)$个候选图对,如果过滤掉 k 个不确定图,则可以减少$\sum_{i=1}^{k}m-i$ 个候选图对。

5.2 Map 方剪枝的 Hadoop 连接

根据单图过滤策略,可以对 4.1 节的作业 1 进行修改,得到如下的 Map 方剪枝的 Hadoop 连接。

```
Class Mapper
    Method Initialize
      list(g)←Ø
    Method Map (gid,g)
      list(JPT)=generateJPTs(V,E,P)
      g(PWG)=Join(list(JPTs))
      sum ← 0
      将 g 中每个可能世界图的概率值累加进 sum 中
      If (sum>= α) then
        将 gid 添加进 list(g)
        输出(gid,g(PWG))
    Method Close
      输出候选图对(g_iid,g_jid)
Class Reducer
    Method Reduce(g_iid,g_i(PWG)/list(g_jid))
      for g_jid∈list(g_jid) do
        输出(g_jid,g_i(PWG))
```

该作业在 Mapper 类的 Map 方法里增加了一个 for 循环,计算不确定图 g 对应的所有 PWG 的概率值之和,然后将

不满足过滤条件的不确定图 g 加入候选集 $list(g)$ 中,最后输出不满足过滤条件的不确定图的 id 及其对应的 $g(PWG)$。

5.3 图对过滤策略

根据单图上界过滤策略过滤掉部分不确定图之后,剩余的不确定图将组成候选图对。如果从这些候选图对中进一步过滤掉一些不可能相似的图对,则会大幅减少后期的计算量。

在分析第 3 节相关公式的过程中,进一步得出了有关候选图对的一个定理。

定理 2　设 (g_i,g_j) 为不确定图数据集 G 中的任意一个图对,g' 为 g_i 的可能世界图,g'' 为 g_j 的可能世界图,$p(g')$ 为 g' 的概率,$p(g'')$ 为 g'' 的概率。若 (g_i,g_j) 满足条件:$\sum_{\forall g'\in g_i} p(g') * \sum_{\forall g''\in g_j} p(g'')<\alpha$,则可以判断图对 (g_i,g_j) 不满足相似性条件,可以被剪枝。

证明:从定理 1 的证明过程可知:

$$Pr\{Sim(gi,gj)\geqslant\gamma\}=\sum_{\forall g'\in g_i} p(g') * \sum_{\forall g''\in g_j} p(g'')$$

结合式(2),定理 2 得证。

根据上述定理,得到图对上界过滤策略,若图对 (g_i,g_j) 的所有可能世界图对的概率乘积之和小于 α,则将其过滤。

图对上界过滤策略通过计算所有可能世界图对的概率乘积之和来估计一个图对的相似性上界,避免了对所有可能世界图对的相似性的盲目计算。过滤图对的数量取决于具体的不确定图数据集。实验证明,在一般情况下,该步骤的过滤策略可以减少大量的相似性计算。

5.4 Reduce 方剪枝的 Hadoop 连接

根据图对过滤策略,可以设计得到如下的 Reduce 方剪枝的 Hadoop 连接:

```
Class Mapper
    Method Map ((gᵢid,gᵢ(PWG))/(gⱼid,gⱼ(PWG)))
      输出(gⱼid,gᵢ(PWG)/gⱼ(PWG))
Class Reducer
    Method Reduce(gⱼid,gⱼ(PWG)/list(gᵢ(PWG)))
      for gᵢ(PWG)∈list(gᵢ(PWG)) do
        sum ← 0
        对于 gᵢ 以及 gⱼ 中所有的可能世界图对(g',g''),将其概率乘
        积 p(g') * p(g'')累加到 sum 中
        If (sum>=α) then
          输出(gᵢid,gⱼ(PWG))
```

该作业在 Reducer 类的 Reduce 方法中增加了一个 for 循环,用来检查所有包含 g_j 的候选图对,将所有可能世界图对的概率乘积之和小于 α 的图对过滤掉。

6 基于 MapReduce 的不确定图相似性连接算法 MUGSJoin

根据基本算法以及第 5 节提出的过滤策略,本节提出同时使用 Map 方剪枝和 Reduce 方剪枝的混合连接,即完整的不确定图的相似性连接算法 MUGSJoin。该算法采用过滤-验证的框架进行,整个算法分为 3 个作业,其中过滤阶段采用两个 MapReduce 作业完成。

6.1 过滤阶段

作业 1:Map 函数以数据库中的不确定图 g 及其 id 作为输入的键值对,在 Map 过程中为每个不确定图 g 生成一组 JPT,并将其联接生成一组可能世界图 $g(PWG)$。随后 Map 函数使用单图过滤策略对不确定图 g 进行检查,并将其所有 PWG 的概率值相加,若概率之和不小于 α,则将不确定图 g 的 id 及其对应的 $g(PWG)$ 作为键值对输出。在作业 1 的 Mapper 类中用到了一个公共变量 $list(g)$,该变量用来记录不满足单图过滤条件的所有候选图的相应 id,并在 Mapper 类的 Close 函数中通过两个 for 循环输出第一阶段的候选图对。Reduce 函数获得包含某一不确定图 g_i 的所有候选图对以及 g_i 和它的 PWG 列表,输出用 id 和 PWG 列表表示的候选图对。

Map 函数和 Reduce 函数如下:

Map:$(gid,g)\rightarrow(gid,g(PWG)/g_jid)$

Reduce:$(g_iid,g_i(PWG)/list(g_jid))\rightarrow(g_jid,g_i(PWG))$

作业 2:Map 函数的输入为 $(g_jid,g_i(PWG))$ 或 $(g_jid,g_j(PWG))$,然后使用 g_jid 作为键,$g_i(PWG)$ 或者 $g_j(PWG)$ 作为值,构成输出的键值对。在 Reduce 函数中,检查所有包含 g_j 的候选图对,使用图对过滤策略将图对中两个不确定图的所有 PWG 组合的概率乘积相加,若和值不小于 α,则将该图对作为第二阶段的候选结果输出。

Map 函数和 Reduce 函数如下:

Map:$((g_jid,g_i(PWG))/(g_jid,g_j(PWG)))\rightarrow(g_jid,g_i(PWG)/g_j(PWG))$

Reduce:$(g_jid,g_j(PWG)/list(g_i(PWG)))\rightarrow(g_iid,g_j(PWG))$

6.2 验证阶段

验证阶段对第二阶段的候选图对 $(g_iid,g_j(PWG))$ 进行验证。其中 Map 函数使用类似于作业 2 中 Map 函数的方法,将 $(g_iid,g_i(PWG))$ 或 $(g_iid,g_j(PWG))$ 作为输入,键值对 $(g_iid,g_i(PWG)/g_j(PWG))$ 作为输出。在 Reduce 函数中,检查所有包含 g_i 的候选图对,根据定义 4 计算候选图对的 Pr 值,若候选图对的所有满足相似性阈值的 PWG 组合的概率乘积之和不小于 α,则将该图对作为最终结果之一输出。

Map 函数和 Reduce 函数如下:

Map:$((g_iid,g_i(PWG))/(g_jid,g_j(PWG)))\rightarrow(g_iid,g_i(PWG)/g_j(PWG))$

Reduce:$(g_iid,g_i(PWG)/list(g_j(PWG)))\rightarrow(g_iid,g_jid)$

6.3 MUGSJoin 算法

MUGSJoin 算法结合了 Map 方剪枝和 Reduce 方剪枝两种过滤策略,并对剪枝后剩余的候选图对进行验证,最终输出相似性连接结果。其中过滤阶段的作业 1 对应的是 5.2 节的 Map 方剪枝策略,作业 2 对应的是 5.3 节的 Reduce 方剪枝策略,验证阶段的作业 3 对应的是 4.1 节基本算法中的作业 2。根据定义 4 计算候选图对的 Pr 值,并输出将满足条件的结果。

算法 1　MUGSJoin 算法

```
输入:不确定图集合 G,概率阈值 α,相似性阈值 γ
输出:所有相似图对(gᵢid,gⱼid)
/* 过滤:作业 1 */
Class Mapper
  Method Initialize
```

```
    list(g)←∅
  Method Map (gid,g)
    list(JPT)=generateJPTs(V,E,P)
    g(PWG)=Join(list(JPTs))
    sum ← 0
    将 g 中每个可能世界图的概率值累加进 sum 中
    If (sum>=α) then
      将 gid 添加进 list(g)
      输出(gid,g(PWG))
  Method Close
    输出候选图对(g_i id,g_j id)
Class Reducer
  Method Reduce(g_i id,g_i(PWG)/list(g_j id))
    for g_j id∈list(g_j id) do
      输出(g_j id,g_i(PWG))
/* 过滤:作业 2 */
Class Mapper
  Method Map((g_i id,g_i(PWG))/(g_j id,g_j(PWG)))
    输出(g_j id,g_i(PWG)/g_j(PWG))
Class Reducer
  Method Reduce(g_j id,g_j(PWG)/list(g_i(PWG)))
    for g_i(PWG)∈list(g_i(PWG)) do
    sum←0
    对于 g_i 以及 g_j 中所有的可能世界图对(g′,g″),将其概率乘积
    p(g′)*p(g″)累加到 sum 中
    If (sum>=α) then
      输出(g_i id,g_j(PWG))
/* 验证:作业 3 */
Class Mapper
  Method Map ((g_i id,g_i(PWG))/(g_i id,g_j(PWG)))
    输出(g_i id,g_i(PWG)/g_j(PWG))
Class Reducer
  Method Reduce(g_i id,g_i(PWG)/list(g_j(PWG)))
    for g_i(PWG)∈list(g_i(PWG)) do
      Pr←0
      对于 g_i 以及 g_j 中所有的可能世界图对(g′,g″),计算相应的
      Mc(g′,g″)
      if (Mc(g′,g″)>=γ) then
        将 g′和 g″的概率乘积累加进 Pr 中
      if (Pr >=α) then
        输出(g_i id,g_j id)
```

6.4 正确性和复杂度分析

算法 1 能够正确返回所有满足定义 4 的不确定图对,其正确性通过单图过滤和图对过滤的正确性来保证。

在过滤阶段的作业 1 中,Map 函数的 IO 开销为 $O(N)$。对于每个 Map 函数,构造一组 JPT 的时间复杂度为 $O(n+2m)$,生成一组可能世界图的时间复杂度为 $O(2^{tr})$。随后 Map 函数使用单图过滤策略对不确定图 g 进行检查,将其所有 PWG 的概率值相加,若概率之和不小于 α,则将不确定图 g 的 id 及其对应的 g(PWG)作为键值对输出。设 g 的可能世界图数目为 p,则该操作的时间复杂度为 $O(p)$。设 Map 阶段过滤后得到的候选图数目为 C_1,则 Map 函数的输出为 $O(p*C_1)$,Close 函数的时间复杂度为 $O(1/2*C_1^2)$。因此作业 1 的 Map 阶段的时间复杂度总和为 $O(N*(n+2m+2^{tr}+p)+p*C_1+1/2*C_1^2)$。Shuffle 阶段的传输复杂度为 Map 阶段输出的所有键值对的数量,即 $O(p*C_1+1/2*C_1^2)$。每个 Reduce 函数输出包含某一不确定图 g_i 的所有候选图对。Reduce 阶段在最坏情况下的时间复杂度总和为 $O(p*C_1)$,IO 开销为 $O(1/2*p*C_1^2)$。相比于基本算法中的作业 1,Map 阶段的时间复杂度中增加了 $O(p*C_1)$,这主要是因为加入了单图过滤操作;若 $C_1 \ll N$,则其反而会降低 Map 的时间复杂度,并且使 Shuffle 阶段以及 Reduce 阶段的复杂度都降低。实验证明,在大部分情况下,该过滤策略效果显著。

相比于基本算法,MUGSJoin 算法增加了一个 MapReduce 作业进行图对过滤。该作业的 Map 函数读取作业 1 输出的所有键值对以及作业 1 过滤后得到的所有不确定图,因此 IO 开销为 $O(1/2*p*C_1^2+p*C_1)$。Map 函数仅仅是将输入作为结果输出,因此其时间复杂度和 Shuffle 的传输复杂度都为 $O(1/2*p*C_1^2+p*C_1)$。Reduce 函数检查所有包含某一不确定图 g_j 的候选图对,若不满足图对过滤策略则将其输出。设作业 2 过滤后得到的候选图对数目为 C_2($C_2<1/2*C_1^2$),则 Reduce 函数的时间复杂度总和为 $O(1/2*p^2*C_1^2)$,IO 开销为 $O(p*C_2)$。

在验证阶段的作业 3 中,Map 函数读取作业 2 输出的所有键值对以及作业 2 过滤后得到的所有不确定图,因此 IO 开销为 $O(p*C_1+p*C_2)$。类似于作业 2,验证阶段的 Map 函数仅仅是将输入作为结果输出,因此其时间复杂度和 Shuffle 的传输复杂度也都为 $O(p*C_1+p*C_2)$。Reduce 函数检查所有包含某一不确定图 g_i 的候选图对,根据定义 4 计算其 Pr 值,若 Pr 值不小于 α,则将该图对作为结果输出。在计算 Pr 值的过程中,计算候选图对中每个可能世界图对的相似性所需的时间为 $O(n\log n+m)$,其计算方法及时间复杂度分析请参见文献[21]。设最后的结果图对数量为 C_3,则 Reduce 函数的总时间复杂度为 $O(1/2*p^2*C_1^2*(n\log n+m))$,IO 开销为 $O(C_3)$。

相比于基本算法,MUGSJoin 算法主要增加了两个过滤操作,其中单图过滤在基本算法的作业 1 中添加,因此没有使复杂度大幅变化,而图对过滤算法则另外设置了一个 MapReduce 作业来完成,从理论上说,增加了一定的时间复杂度和传输复杂度。影响两个算法效率的主要因素取决于过滤的质量,若 $C_1 \ll N$,$C_2 \ll 1/2*C_1^2$,$C_3 \ll C_2$,则不仅可以抵消过滤步骤产生的代价,实际上还可以达到大幅提高算法效率的目的。

7 实验分析

7.1 实验平台和实验数据

本实验的实验平台采用亚马逊云服务平台(Amazon Web Services,AWS),使用 AWS 的弹性计算云(Elastic Compute Cloud,EC2)。在 EC2 中每个计算节点称为一个实例。本实验一共使用 10 个实例,每个实例都是 m3.xlarge,其配置为 64 位 CPU 架构,主频 2.5GHZ,4 个 vCPU,13 个 ECU,15 GB 内存,2×40 G SSD,网络传输性能高,其中 vCPU 代表 EC2 虚拟内核,ECU 代表 EC2 计算单元,一个 ECU 大概相当于 1GHz Xeon 处理器的性能。

实验中使用了一个真实的不确定图数据集。该数据集取

自 STRING 数据库[22],包含 BioGRID 数据库[23]中的有机物 PPI 网络。PPI 网络中的顶点代表蛋白质,边代表蛋白质间的相互作用,顶点标签为 STRING 数据库提供的蛋白质的 COG 功能注释[24],边的存在概率 P 也是由 STRING 数据库提供。

在 STRING 数据库中随机生成了 5 个不确定图数据集 G_1,G_2,G_3,G_4 和 G_5,各数据集相应的不确定图数分别为 100 M,200 M,300 M,400 M 和 500M。每个不确定图平均包含 365 个结点和 647 条边。每条边的平均存在概率为 0.413。

实验的其他参数设置如下:概率阈值 α 为 0.3～0.5,默认值为 0.4;相似性阈值 γ 为 0.5～0.7,默认值为 0.6。

7.2　算法性能测试及验证

由于目前在基于 MapReduce 的不确定图数据集的相似性连接问题上并没有可直接进行对比的算法,因此先对本文所提出的方法及过滤策略进行单独的测试和验证。

首先测试本文提出的相似性连接算法的有效性(见图 3),并分析两个参数 α 和 γ 对算法运行时间的影响,如图 4、图 5 所示。

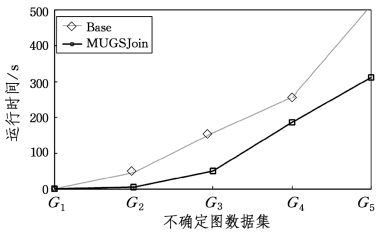

图 3　算法的运行时间对比

Fig. 3　Comparison of running time of algorithms

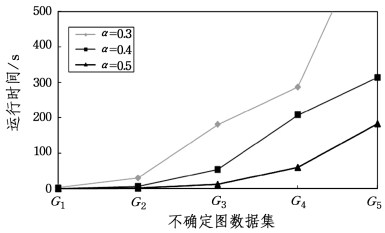

图 4　α 值对算法运行时间的影响

Fig. 4　Influence of α on running time

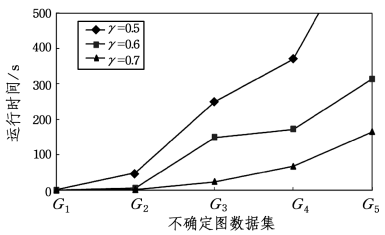

图 5　γ 值对算法运行时间的影响

Fig. 5　Influence of γ on running time

图 3 显示了本文提出的基本算法(用 Base 来表示)和同时使用 Map 方剪枝和 Reduce 方剪枝的混合连接算法 MUGSJoin 在不同大小的不确定图数据集中的运行时间对比。算法运行过程中概率阈值 α 和相似性阈值 γ 取默认值。

从图 3 可以看出,随着不确定图数据集的增大,基本算法和 MUGSJoin 算法的运行时间都明显延长。相比于基本算法,MUGSJoin 可以实现更快的相似性连接速度,因为 Map 方剪枝以及 Reduce 方剪枝大大减少了后期计算相似性度的候选图对。但由于数据集是随机产生的,每个实验数据集中符合过滤条件的候选图对的数目并没有明显的规律,因此不同的数据集的过滤能力有所不同。

图 4 给出了在不同的概率阈值下,混合连接算法 MUGSJoin 在不确定图数据集中的运行时间。算法运行过程中相似性阈值 γ 取默认值。

从实验数据可以看出,随着数据集中不确定图数量的增多,运行时间逐渐延长。但各数据集运行算法的时间都会随着概率阈值 α 的增大而缩短。这是因为 α 值越大,在 Map 阶段和 Reduce 阶段移除的不确定图或者不确定图对越多,下一阶段的计算量就越少,从而使得算法的整体运行时间减小。

图 5 给出了在不同的相似性阈值下,混合连接算法 MUGSJoin 在不确定图数据集中的运行时间。算法运行过程中概率阈值 α 取默认值。

从图 5 可以看出,在同一个不确定图数据集中,算法的运行时间也随着相似性阈值 γ 的增大而缩短。这是因为 γ 值越大,在 Reduce 阶段过滤出的可能世界图对越多,从而使得计算相似度算法的运行时间减少,进而缩短算法的整体运行时间。

图 6 通过对比基本算法和混合连接算法 MUGSJoin 相应阶段的候选图对数量来分析两个过滤策略的剪枝能力。

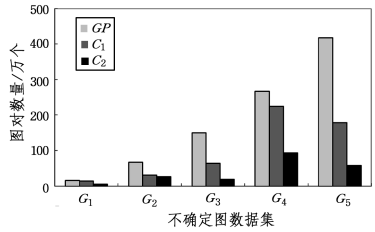

图 6　剪枝能力分析

Fig. 6　Analysis of pruning ability

图 6 中,GP 表示基本算法的作业 1 中 Map 阶段输出的候选图对数量,C_1 表示 MUGSJoin 算法的作业 1 中 Map 阶段输出的候选图对数量(即进行单图过滤后剩余的候选图对数量),C_2 表示 MUGSJoin 算法的作业 1 中 Reduce 阶段输出的候选图对数量(即进行单图过滤以及图对过滤后剩余的候选图对数量)。算法运行过程中概率阈值和相似性阈值取默认值。可以看出,在随机生成的 5 个数据库中,单图过滤策略以及图对过滤策略都可以进行有效剪枝,但具体的剪枝数量与数据库中不确定图的概率值有关。

最后,对算法的查准率和查全率进行分析,结果如图 7 和图 8 所示。

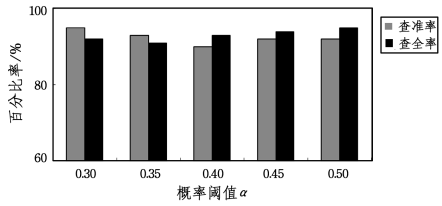

图 7　不同概率阈值下的查准率和查全率

Fig. 7　Precision and recall under different probability thresholds

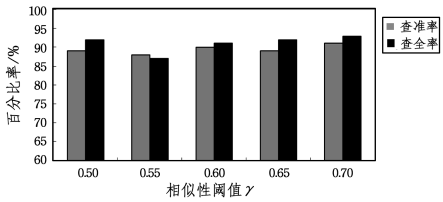

图 8 不同相似性阈值下的查准率和查全率

Fig. 8 Precision and recall under different similarity thresholds

查准率是指在返回的相似图对中正确结果的百分比，查全率是指返回的相似图对在所有正确结果中的百分比。为了获得不确定图数据集 G_1-G_5 中正确的相似图对的数目，将数据库中的不确定图通过软件进行转换，使其可以在可视化状态下进行相似性连接的判断。然后将这些正确的相似图对和实验结果进行对比，从而计算出本文算法的查准率和查全率。由图 7 和图 8 可知，本文所提方法在查准率和查全率上都具有很高的近似质量，且不会随着概率阈值和相似性阈值的变化而产生较大的波动，其百分比都大于 85%。

图 9 对比了基本算法以及混合连接算法 MUGSJoin 的扩展性。取数据集 G_3 进行测试，概率阈值 α 和相似性阈值 γ 取默认值，并将平台的实例数目设置为在 5～10 之间变化。从图 9 可以看出，随着实例个数的增加，两种算法的运行时间都会不断缩短，虽然当实例数大于 7 时，运行时间缩短的趋势有所减缓，但从总体上仍然可以看出算法具有良好的可扩展性。

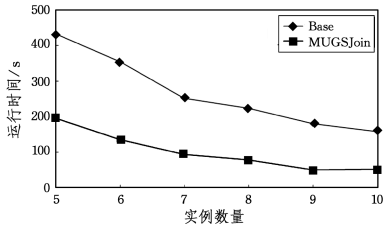

图 9 扩展性分析

Fig. 9 Scalability analysis

从以上测试数据及实验分析可以得出，本文所提出的基于 MapReduce 的不确定图数据集上的相似性连接算法 MUGSJoin 具有较好的可行性和有效性，算法中所采用的单图过滤及图对过滤策略可以过滤掉大部分的不相似图对，具有较好的剪枝能力和剪枝效率，且算法可以达到较高的查询质量。从实验数据还可以看出，随着不确定图数量的增多或者不确定图中顶点数和边数的增加，算法的运行时间大幅延长，虽然可以采用增加实例数量或者提高实例配置等方式缩短运行时间，但不能从本质上降低算法的时间复杂度，后续将进一步设计更加高效的并行算法。

7.3 相近算法对比分析

文献[12]提出了一种基于 MapReduce 的图相似度连接算法 BMGSJoin。该算法选用编辑距离度量图之间的相似度，使用基于路径的 q-gram 方法进行计数过滤，并采用了布隆过滤技术减少候选图对的数目，属于比较典型的基于MapReduce 的图相似度连接算法。下面，将本文提出的图相似性连接方法与该算法进行比较和分析。

由于 BMGSJoin 算法处理的是确定图，在实验之前，需要对 7.1 节给出的测试数据进行以下处理：1）选取不确定图数据集 G_3，去掉所有图上边的概率值，得到对应的确定图数据集 DG_3，并以此作为对比实验的输入数据；2）将本文提出的 MUGSJoin 算法进行相应的修改，保留其进行确定图的相似性连接的处理步骤，并将修改后得到的新算法记为 DMSJoin。

图 10 给出了 BMGSJoin 算法与 DMSJoin 算法在相似性阈值 γ 变化的情况下对于确定图数据集 DG_3 的运行时间。由于采用的分布式平台实例性能的提高，BMGSJoin 的运行时间也大幅度缩短，但相比之下，DMSJoin 算法仍然表现出了较好的运行效率。

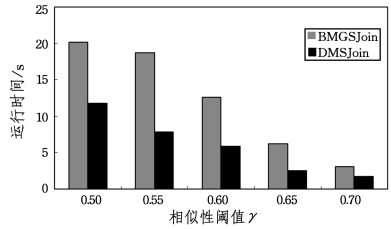

图 10 运行时间对比

Fig. 10 Comparison of running time

结束语 本文研究了基于 MapReduce 编程模型的不确定图的相似性连接操作，提出了使用 MapReduce 的不确定图的相似性连接基本算法。由于基本算法会产生大量的中间键值对，导致网络传输复杂度增加，本文提出了一组基于概率和的过滤策略，并在此基础上设计了 Map 方剪枝的 Hadoop 连接和 Reduce 方剪枝的 Hadoop 连接两种方法。Map 方剪枝的 Hadoop 连接主要用于过滤不确定图数据集中在给定阈值下不可能具有相似图的不确定图，Reduce 方剪枝的 Hadoop 连接主要用于进一步过滤不满足阈值条件的候选图对。最后提出了同时采用 Map 方剪枝和 Reduce 方剪枝的混合连接算法 MUGSJoin。实验证明，MUGSJoin 算法相比于基本算法具有更好的性能和可扩展性。

在后续的工作中，我们将继续对不确定图数据集上的相似性连接问题做深入的分析，进一步研究分布式编程的优化方式，并设计更高效的过滤策略来提高不确定图的相似性连接算法的效率。

参 考 文 献

[1] ZHAO X, XIAO C, LIN X, et al. Efficient Graph Similarity Joins with Edit Distance Constraints[C]// Proceedings of IEEE Conference on Data Engineering. IEEE Press, 2012: 834-845.

[2] YUAN Y, WANG G, CHEN L, et al. Graph similarity search on large uncertain graph databases[J]. Vldb Journal, 2015, 24(2): 271-296.

[3] ZHENG W, ZOU L, LIAN X, et al. Graph similarity search with edit distance constraint in large graph databases[C]// ACM International Conference on Conference on Information & Knowledge Management. ACM, 2013: 1595-1600.

[4] WANG Y, WANG H, LI J, et al. Efficient graph similarity join for information integration on graphs[J]. Frontiers of Computer Science, 2016, 10(2): 317-329.

[5] PANG J, YU G, XU J, et al. Similarity Joins on Massive Data Based on MapReduce Framework[J]. Computer Science, 2015, 42(1): 1-5. (in Chinese)
庞俊，于戈，许嘉，等. 基于 MapReduce 框架的海量数据相似性连接研究进展[J]. 计算机科学，2015，42(1)：1-5.

[6] MIAO F Y, WANG H Z. A Method of Similarity Join on Uncer-

tain Graph Database[J]. Journal of Software, 2018, 29(10): 3150-3163. (in Chinese)
缪丰羽,王宏志. 一种不确定图数据库上的相似性连接方法[J]. 软件学报,2018,29(10):3150-3163.

[7] LIN J, SCHATZ M. Design patterns for efficient graph algorithms in MapReduce[C]// Eighth Workshop on Mining & Learning with Graphs(Mlg 10). 2010:78-85.

[8] PLIMPTON S J, DEVINE K D. MapReduce in MPI for Large-scale graph algorithms[J]. Parallel Computing, 2011, 37(9): 610-632.

[9] QIN L, YU J X, CHANG L, et al. Scalable big graph processing in MapReduce[C]// Proceedings of the ACM SIGMOD International Conference on Mangement of Data. 2014:22-27.

[10] KOLDA T G, PINAR A, PLANTENGA T, et al. Counting Triangles in Massive Graphs with MapReduce [J]. Siam Journal on Scientific Computing, 2013, 36(5): 48-77.

[11] CHEN Y, ZHAO X, XIAO C, et al. Efficient and Scalable Graph Similarity Joins in MapReduce[J]. The Scientific World Journal, 2014, 2014(3): 749028.

[12] CHEN Y F, ZHAO X, HE P J, et al. BMGSJoin: A MapReduce Based Graph Similarity Join Algorithm[J]. Pattem Recognition and Artificial Intelligence, 2015, 28(5): 472-480. (in Chinese)
陈一帆,赵翔,何培俊,等. BMGSJoin:一种基于 MapReduce 的图相似度连接算法[J]. 模式识别与人工智能,2015,28(5):472-480.

[13] PANG J, GU Y, XU J, et al. Efficient Graph Similarity Join with Scalable Prefix-Filtering Using MapReduce[J]. Lecture Notes in Computer Science, 2014, 8485: 415-418.

[14] SANJAY G, HOWARD G, SHAN-TAK L. The Google file system[J]. ACM SIGOPS Operationg Systems Review, 2003, 37(5): 29-43.

[15] AFRATI F N, ULLMAN J D. Optimizing joins in a map-reduce environment[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(9): 1282-1298.

[16] OKCAN A, RIEDEWALD M. Processing theta-joins using MapReduce[C]// ACM SIGMOD International Conference on Management of Data(SIGMOD 2011). Athens, Greece, DBLP, 2011: 949-960.

[17] YANG H C, DASDAN A, HSIAO R L, et al. Map-reduce-merge: simplified relational data processing on large clusters[C]// ACM SIGMOD International Conference on Management of Data. ACM, 2007: 1029-1040.

[18] AFRATI F N, SARMA A D, MENESTRINA D, et al. Fuzzy Joins Using MapReduce[C]// IEEE International Conference on Data Engineering. IEEE, 2012: 498-509.

[19] LUO W, TAN H, MAO H, et al. Efficient Similarity Joins on Massive High-Dimensional Datasets Using MapReduce[C]// IEEE International Conference on Mobile Data Management. IEEE Computer Society, 2012: 1-10.

[20] WANG Y, WANG H, LI J, et al. Efficient subgraph join based on connectivity similarity[J]. World Wide Web-internet & Web Information Systems, 2015, 18(4): 871-887.

[21] KIM Y, SHIM K. Parallel Top-K Similarity Join Algorithms Using MapReduce[C]// IEEE International Conference on Data Engineering. IEEE Computer Society, 2012: 510-521.

[22] MORRIS R, MORRIS R. ExOR: opportunistic multi-hop routing for wireless networks[C]// Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. ACM, 2005: 133-144.

[23] CEOL A, CHATR ARYAMONTRI A, LICATA L, et al. MINT, the molecular interaction database: 2009 update. [J]. Nucleic Acids Research, 2010, 38(suppl1): D532-D539.

[24] CHUA H N, SUNG W K, WONG L. Exploiting Indirect Neighbours and Topological Weight to Predict Protein Function from Protein-Protein Interactions[M]// Data Mining for Biomedical Applications. Springer Berlin Heidelberg, 2006: 1623-1630.

(上接第 292 页)

[8] MORZY M. Prediction of moving object location based on frequent trajectories[C]// The 21st International Symposium on Computer and Information Sciences(ISCIS'2006). 2006: 583-592.

[9] AGRAWAL R, SRIKANT R. Fast algorithms for mining association rules[C]// Proceedings of the 20th International Conference on Very Large Data Bases. VLDB, 1994: 487-499.

[10] GIDÓFALVI G, DONG F. When and where next: individual mobility prediction [C] // Proceedings of the First ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems. ACM, 2012: 57-64.

[11] LV M Q, CHEN L, CHEN G C. Position Prediction Based on Adaptive Multi-Order Markov Model[J]. Journal of Computer Research and Development, 2010, 47(10): 1764-1770. (in Chinese)
吕明琪,陈岭,陈根才. 基于自适应多阶 Markov 模型的位置预测[J]. 计算机研究与发展,2010,47(10):1764-1770.

[12] YU X G, LIU Y H, WEI D, et al. Hybrid Markov mode for mobile path prediction [J]. Journal on Communications, 2006, 27(12): 61-69. (in Chinese)
余雪岗,刘衍珩,魏达,等. 用于移动路径预测的混合 Markov 模型[J]. 通信学报,2006,27(12):61-69.

[13] CHEN Z B, SHEN H T, ZHOU X F. Discovering popular routes from trajectories[C]// Proceedings of the 27th ICDE International Conference on Data Engineering. Germany, IEEE, 2011: 900-911.

[14] DEMIRYUREK U, SHAHABI C. Indexing network voronoi diagrams[C]// Database Systems for Advanced Applications. Springer Berlin Heidelberg, 2012: 526-543.

[15] CHEN C. The establishment and application of voronoi diagram in computer mapping[J]. Acta Geodaetica et Cartographica Sinica, 1987, 16(3): 223-231. (in Chinese)
陈春. 泰森多边形的建立及其在计算机制图中的应用[J]. 测绘学报,1987,16(3):223-231.

[16] PAVAN M, PELILLO M. Dominant sets and pairwise clustering [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(1): 167-172.

[17] YUAN J, ZHENG Y, XIE X, et al. Driving with knowledge from the physical world[C]// Proceedings of International Conference on the 17th ACM SIGKDD Knowledge Discovery and Data Mining. USA, ACM, 2011: 316-324.