

# 《知识图谱：概念与技术》

## 知识图谱管理

---

# 本章大纲

---

- 知识图谱基本操作
- 知识图谱管理方法

# 本章大纲

---

- 知识图谱基本操作
- 知识图谱管理方法

# 基本操作

---

- 类似于关系数据，知识图谱上的基本操作也有以下几个
  - 选择
  - 连接

# 选择

- **选择**在知识图谱中选择满足给定条件的知识图谱片段

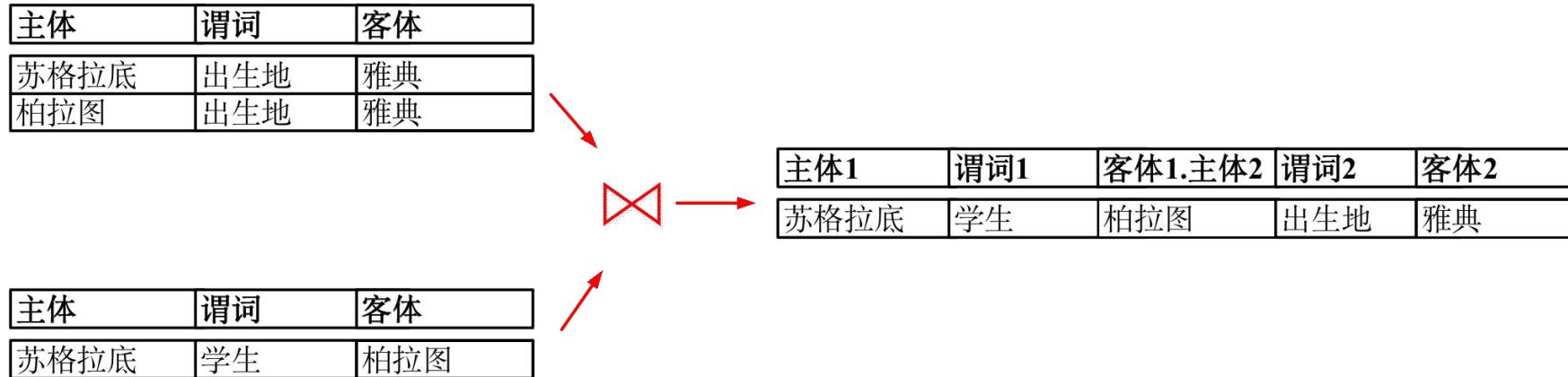
主体	谓词	客体
苏格拉底	学生	柏拉图
苏格拉底	出生时间	公元前469年
苏格拉底	英文译名	Aristotle
苏格拉底	出生地	雅典
苏格拉底	isA	哲学家
柏拉图	代表作品	《理想国》
柏拉图	出生时间	公元前427年
柏拉图	英文译名	Plato
柏拉图	出生地	雅典
柏拉图	isA	唯心主义哲学家
《理想国》	地位	最具影响力的20本学术书
《理想国》	isA	书籍
雅典	外文名称	Athens
雅典	isA	城市
唯心主义哲学家	subclassOf	哲学家

Diagram illustrating the selection process from a large knowledge graph to a smaller subset:

- The original knowledge graph (left) contains 18 triples.
- Selection conditions are applied to filter triples where the subject is either "苏格拉底" (Socrates) or "柏拉图" (Plato), and the predicate is "出生地" (place of birth).
- These selected triples are grouped by subject (Socrates and Plato) and then aggregated into a single row per subject.
- The resulting subset (right) contains 4 triples, showing the place of birth for both Socrates and Plato.

# 连接

- **连接**是按照一定条件从两个知识图谱的笛卡儿积中选取知识图谱片段



# 知识图谱应用特点

---

- 常见的应用中知识图谱数据操作的特点如下：
  1. 选择：选择度高
  2. 连接：连接数量多

# 本章大纲

---

- 知识图谱基本操作
- 知识图谱存储

# 知识图谱的物理存储

---

- 知识图谱数据的基本操作
- 知识图谱的存储形式

# 基本操作

---

- 类似于关系数据，知识图谱上的基本操作也有以下几个
  - 选择
  - 连接

# 选择

- **选择**在知识图谱中选择满足给定条件的知识图谱片段

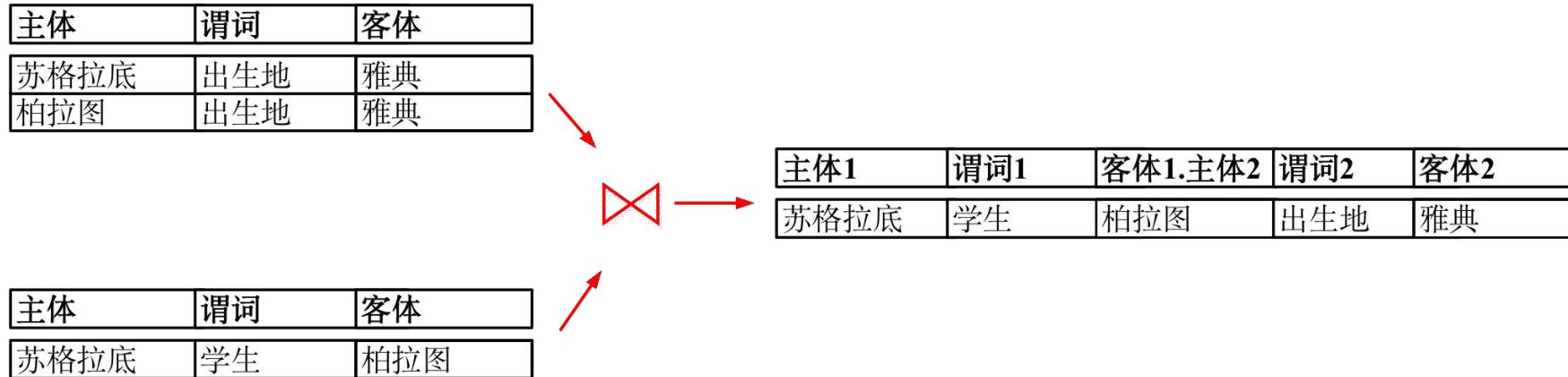
主体	谓词	客体
苏格拉底	学生	柏拉图
苏格拉底	出生时间	公元前469年
苏格拉底	英文译名	Aristotle
苏格拉底	出生地	雅典
苏格拉底	isA	哲学家
柏拉图	代表作品	《理想国》
柏拉图	出生时间	公元前427年
柏拉图	英文译名	Plato
柏拉图	出生地	雅典
柏拉图	isA	唯心主义哲学家
《理想国》	地位	最具影响力的20本学术书
《理想国》	isA	书籍
雅典	外文名称	Athens
雅典	isA	城市
唯心主义哲学家	subclassOf	哲学家

Diagram illustrating the selection process from a large knowledge graph to a smaller subset. Red boxes highlight specific triples: (柏拉图, 英文译名, Aristotle), (柏拉图, 出生地, 雅典), and (柏拉图, 出生地, 雅典). Red arrows point from these selected triples to a smaller table on the right.

主体	谓词	客体
苏格拉底	出生地	雅典
柏拉图	出生地	雅典

# 连接

- **连接**是按照一定条件从两个知识图谱的笛卡儿积中选取知识图谱片段



# 知识图谱应用特点

---

- 常见的应用中知识图谱数据操作的特点如下：
  1. 选择：选择度高
  2. 连接：连接数量多

# 知识图谱的物理存储

---

- 知识图谱数据的基本操作
- 知识图谱管理方法

# 本章大綱

---

- 知识图谱数据的基本操作
- 知识图谱管理方法
  - 单机知识图谱管理方法
  - 分布式知识图谱管理方法

# 知识图谱的关系表存储

- 现有方法：利用关系数据库技术定义一张三列的表

主体	属性	客体
亚里士多德	导师	柏拉图
亚里士多德	出生时间	公元前384年
亚里士多德	英文译名	Aristotle
亚里士多德	出生地	雅典
亚里士多德	isA	哲学家
柏拉图	代表作品	《理想国》
柏拉图	出生时间	公元前427年
柏拉图	英文译名	Plato
柏拉图	出生地	斯塔基拉
柏拉图	isA	唯心主义哲学家
《理想国》	地位	最具影响力的20本学术书
《理想国》	isA	书籍
雅典	外文名称	Athens
雅典	isA	斯塔基拉
唯心主义哲学家	subclassOf	哲学家

# Virtuoso

---

- Virtuoso是OpenLink公司开发的知识图谱管理系统，有免费的社区版
- 一个数据集只有一张大表，加上若干索引
- 下载地址：<https://virtuoso.openlinksw.com/>

# Virtuoso表结构

主体	属性	客体
亚里士多德	导师	柏拉图
亚里士多德	出生时间	公元前384年
亚里士多德	英文译名	Aristotle
亚里士多德	出生地	雅典
亚里士多德	isA	哲学家
柏拉图	代表作品	《理想国》
柏拉图	出生时间	公元前427年
柏拉图	英文译名	Plato
柏拉图	出生地	斯塔基拉
柏拉图	isA	唯心主义哲学家
《理想国》	地位	最具影响力的20本学术书
《理想国》	isA	书籍
雅典	外文名称	Athens
雅典	isA	斯塔基拉
唯心主义哲学家	subclassOf	哲学家

主键  
12字符以内的直接存值；  
12字符以上的对应到一个整数ID

另外，分别建立以主语和宾语为引导列的多列索引

# 三列表的优缺点

---

- 优点：基于现有的关系数据库，方法成熟
- 缺点：选择需要扫描整个数据集，效率低下；自连接多，连接操作效率低

# 三种典型基于关系数据库的优化策略

---

- 属性表方法 Jena
- 垂直划分方法 SW-Store
- 全索引方法 Hexastore、RDF-3x

# 属性表方法

- 在单张大三元组表之外还支持利用属性表进行RDF数据管理

主体	属性	客体
ID1	type	BookType
ID1	title	"XYZ"
ID1	author	"Fox, Joe"
ID1	copyright	"2001"
ID2	type	CDType
ID2	title	"ABC"
ID2	artist	"Orr, Tim"
ID2	copyright	"1985"
ID2	language	"French"
ID3	type	BookType
ID3	title	"MNO"
ID3	language	"English"
ID4	type	DVDType
ID4	title	"DEF"
ID5	type	CDType
ID5	title	"GHI"
ID5	copyright	"1995"
ID6	type	BookType
ID6	copyright	"2004"

主体	Type	Title	copyright
ID1	BookType	"XYZ"	"2001"
ID2	CDType	"ABC"	"1985"
ID3	BookType	"MNO"	NULL
ID4	DVDType	"DEF"	NULL
ID5	CDType	"GHI"	"1995"
ID6	BookType	NULL	"2004"

主体	Title	Author	copyright
ID1	"XYZ"	"Fox, Joe"	"2001"
ID3	"MNO"	NULL	NULL
ID6	NULL	NULL	"2004"

主体	Title	Author	copyright
ID1	"ABC"	"Orr, Tim"	"1985"
ID2	"GHI"	NULL	"1995"

主体	属性	客体
ID1	author	"Fox, Joe"
ID2	artist	"Orr, Tim"
ID2	language	"French"
ID3	language	"English"

聚类属性表

主体	属性	客体
ID2	language	"French"
ID3	language	"English"
ID4	type	DVDType
ID4	title	"DEF"

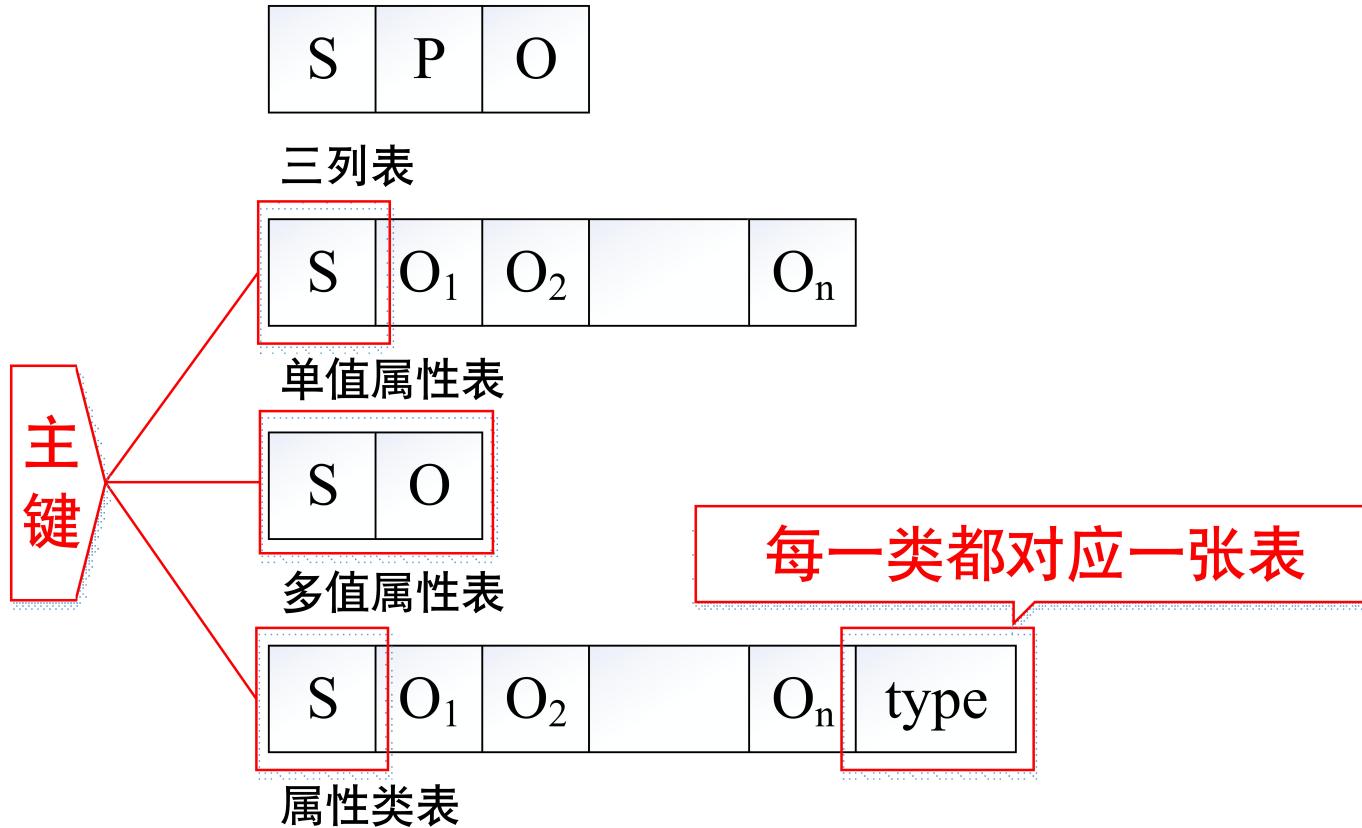
分类属性表

# Jena

---

- Jena是惠普实验室开发的知识图谱管理系统，现已由Apache管理
- Jena在三元组表之外，维护了三种属性表：单值属性表，多值属性表和属性类表
- 下载地址：<http://jena.apache.org/index.html>

# Jena属性表



# 属性表的优缺点

---

- 优点：选择操作不用扫描整个数据集，效率相对较高；连接操作相对于自连接，效率有所提高
- 缺点：表中会存在一定数量空值

# 垂直划分方法

- 对RDF知识图谱数据按照属性分割成若干表的方法

主体	属性	客体
亚里士多德	导师	柏拉图
亚里士多德	出生时间	公元前384年
亚里士多德	英文译名	Aristotle
亚里士多德	出生地	雅典
亚里士多德	isA	哲学家
柏拉图	代表作品	《理想国》
柏拉图	出生时间	公元前427年
柏拉图	英文译名	Plato
柏拉图	出生地	斯塔基拉
柏拉图	isA	唯心主义哲学家
《理想国》	地位	最具影响力的20本学术书
《理想国》	isA	书籍
雅典	外文名称	Athens
雅典	isA	斯塔基拉
唯心主义哲学家	subclassOf	哲学家



出生时间

主体	客体
苏格拉底	公元前469年
柏拉图	公元前427年

出生地

主体	客体
苏格拉底	雅典
柏拉图	雅典

英文译名

主体	客体
苏格拉底	Socrates
柏拉图	Plato

i SA

主体	客体
苏格拉底	哲学家
柏拉图	唯心主义哲学家
《理想国》	书籍
雅典	城市

学生

主体	客体
苏格拉底	柏拉图

代表作品

主体	客体
柏拉图	《理想国》

地位

主体	客体
《理想国》	最具影响力的20本学术书

外文名称

主体	客体
雅典	Athens

subcl assOf

主体	客体
唯心主义哲学家	哲学家

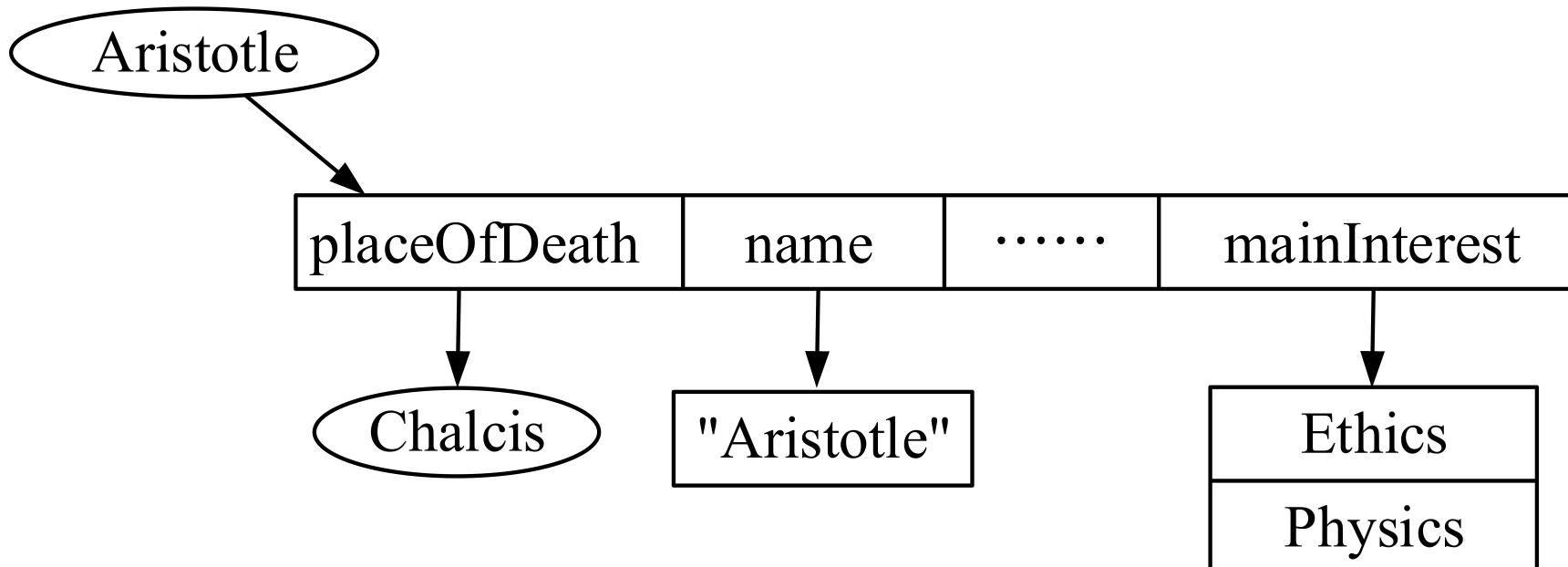
# 垂直划分的优缺点

---

- 优点：避免了大量自连接；存储时无空值
- 缺点：属性为变量的时候访问效率低下

# 全索引方法

- 将三元组在主体、属性、客体之间各种排列下能形成各种形态构建都枚举出来，然后为它们构建索引



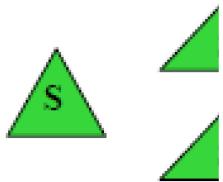
# RDF-3X

---

- RDF-3X是德国马克斯普朗克实验室开发的知识图谱管理系统，主要特点就是建立全索引
- 下载地址：<https://code.google.com/p/rdf3x/>

# RDF—3X索引

- 在全索引之外，RDF—3X还有建立了针对任意两列组合出的值对的聚集索引来记录统计信息



count<sub>SP</sub>

count<sub>SO</sub>

count<sub>PS</sub>

count<sub>PO</sub>

count<sub>OS</sub>

count<sub>OP</sub>

# 全索引的优缺点

---

- 优点：选择操作效率极高
- 缺点：连接操作依然低效；索引更新代价高

# 基于图模型的存储方式

- 通过将RDF 三元组看作带标签的边，RDF知识图谱数据自然的符合图模型结构，下图展示了一个知识图谱与其对应的邻接表

点	邻接表
柏拉图	(isA, 唯心主义哲学家), (出生时间, 公元前 427年), (出生地, 雅典), (英文译名, Plato), (代表作品, 《理想国》 )
苏格拉底	(isA, 哲学家), (出生时间, 公元前469年), (出生地, 雅典), (英文译名, Socrates), (学生, 柏拉图)
.....	.....

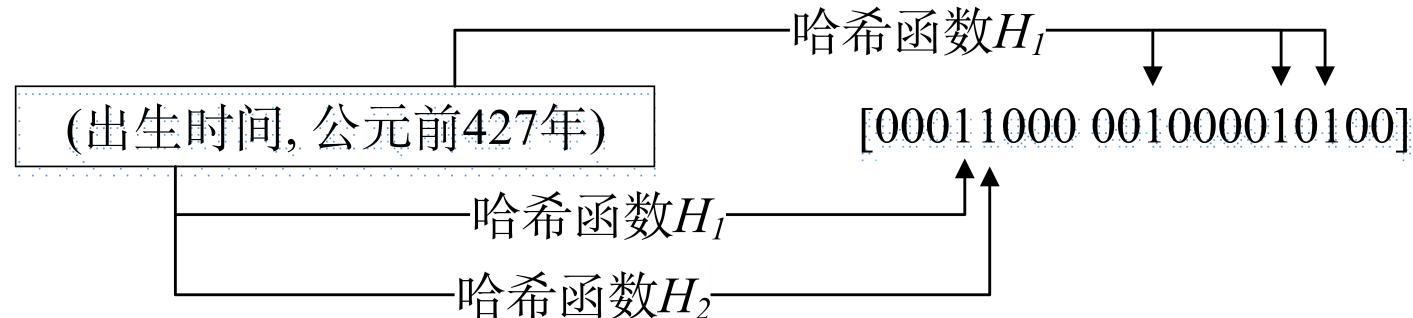
# gStore

---

- gStore是北京大学计算机科学技术研究生开发的知识图谱管理系统，主要特点是建立针对知识图谱图结构构建基于签章树的索引
- 下载地址：<http://www.gstore-pku.com/>

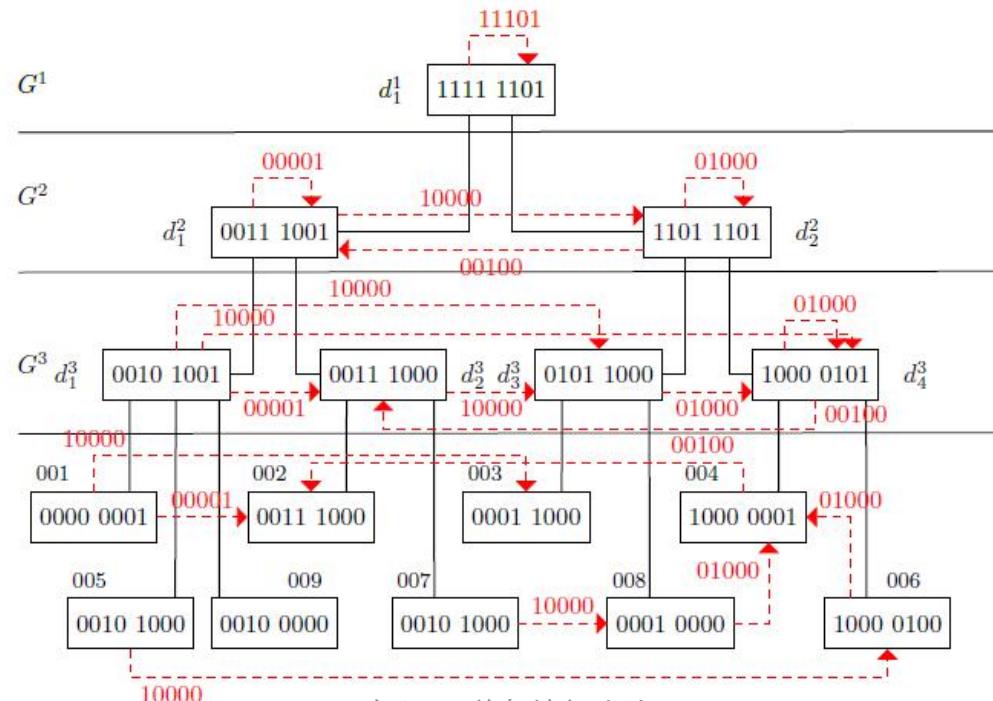
# gStore

- 利用哈希函数将知识图谱中每个实体邻接表转化成一个位串，同时关系也转化成位串



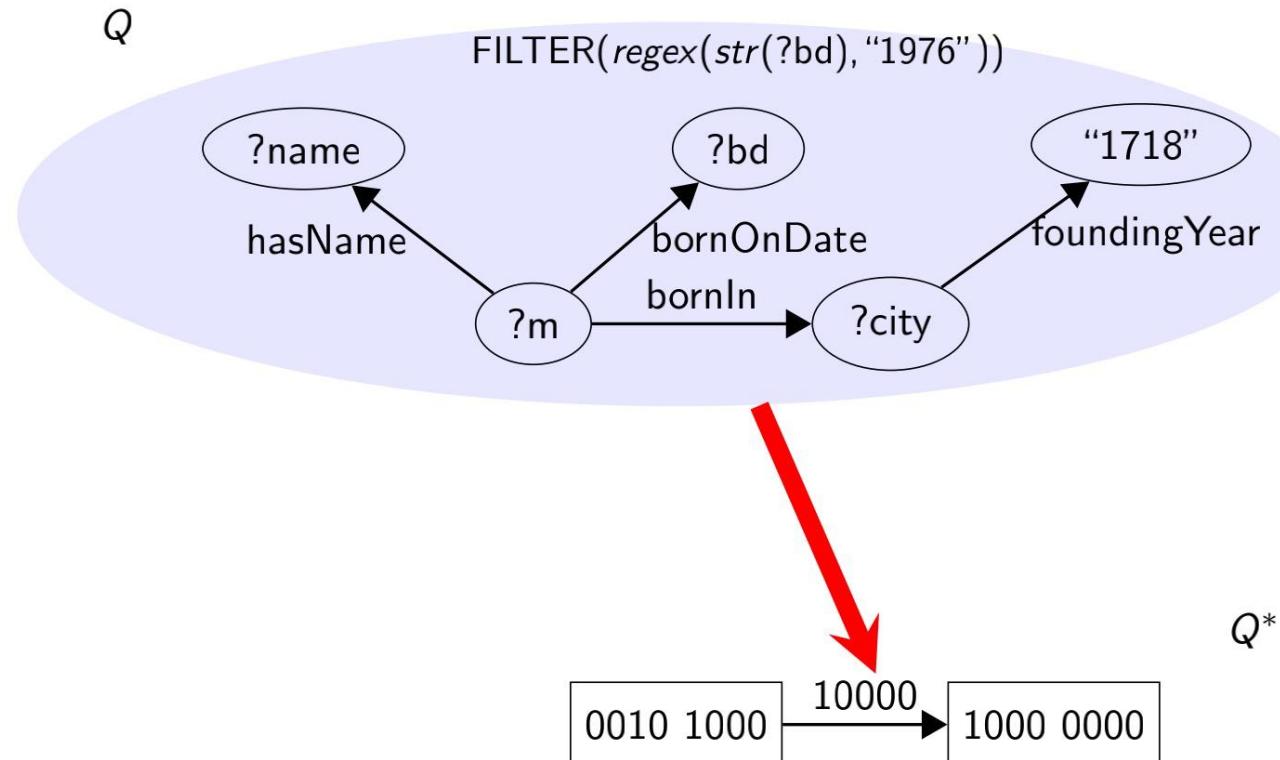
# gStore

- 然后将上述实体的位串组织成一颗签章树，同时签章树中不同实体按照图谱中三元组连接上带关系位串的边



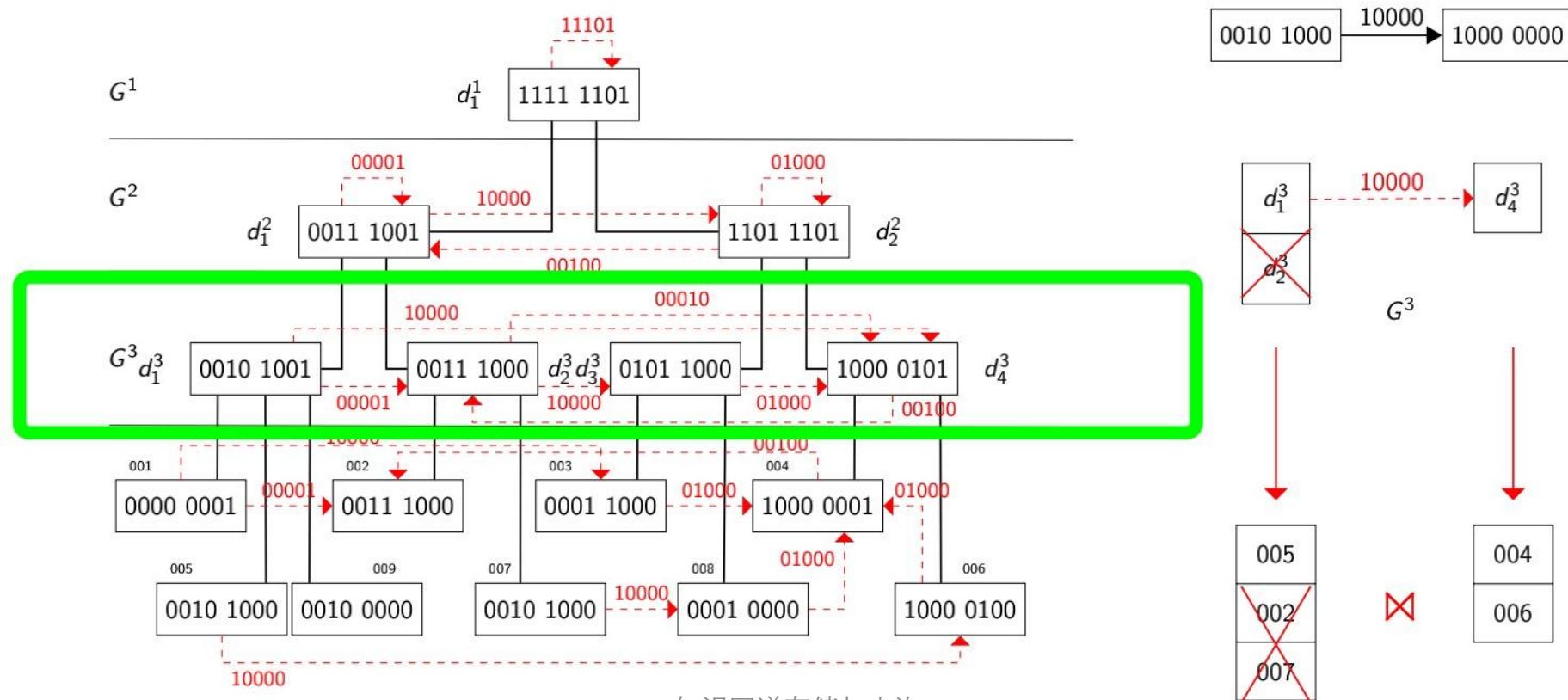
# gStore

- 查询也转化成二进制位串



# gStore

- 先得到候选，再连接



# 基于邻接矩阵的存储方式

---

- 这些系统首先给知识图谱中的主体、谓词和客体（或属性和属性值）进行编号，然后构建一个 $|V_s| \times |V_p| \times |V_o|$ 的三维矩阵M，其中 $|V_s|$ 、 $|V_p|$ 和 $|V_o|$ 分别表示主体、谓词及客体的数量

# BitMat

---

- BitMat就是一个典型的基于邻接矩阵存储知识图谱的系统。因为知识图谱中的常见查询是以谓词为常量的查询，BitMat 将上述 $|V_s| \times |V_p| \times |V_o|$ 的三维矩阵切分成 $|V_p|$ 个 $|V_s| \times |V_o|$ 和 $|V_o| \times |V_s|$ 的二维矩阵
- 同时，为了提高基于主体和客体查询的性能，BitMat 还切分出 $|V_s|$ 个 $|V_p| \times |V_o|$ 的二维矩阵和 $|V_o|$ 个 $|V_p| \times |V_s|$ 的二维矩阵
- 由于实际的知识图谱数据往往比较稀疏，BitMat 对每个矩阵中的每一行交替存储0 和1 的数量以实现压缩

# 图存储的优缺点

---

- 优点：基于整个图结构构建索引，对于复杂访问很高效
- 缺点：简单的访问不一定比基于关系数据库的方法快

# 本章大纲

---

- 知识图谱数据的基本操作
- 知识图谱管理方法
  - 单机知识图谱管理方法
  - 分布式知识图谱管理方法

# 大数据时代的分布式知识图谱管理方法

---

- 基于Hadoop的分布式知识图谱管理方法
- 基于其他云平台的分布式知识图谱管理方法

# 大数据时代的分布式知识图谱管理方法

---

- 这类方法都是利用已有大数据处理平台进行知识图谱数据的存储，并利用已有云平台上成熟的任务处理模式处理知识图谱数据任务
- 被最多地利用的大数据处理平台就是Hadoop；其他的大数据处理平台如Trinity、Spark等也有相关应用

# 大数据发展流程

---

- Hadoop
- Spark/Trinity
- Pregel

# Hadoop概述

---

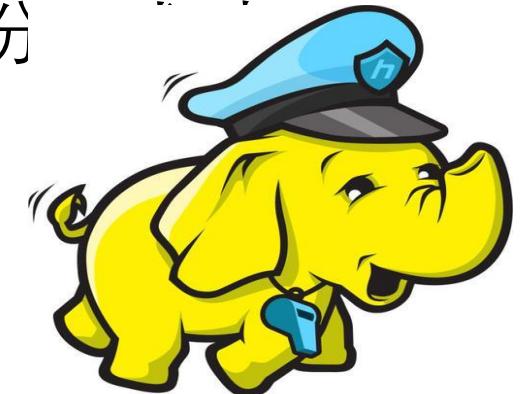
- Hadoop是一个开源的、可靠的、可扩展的分布式并行计算框架
- 主要组成：分布式文件系统HDFS和MapReduce算法执行
- 作者： Doug Cutting
- 语言： Java， 支持多种编程语言， 如：Python、C++
- 系统平台： Linux



# Hadoop起源

---

- Hadoop是Google的集群系统的开源实现
  - Google集群系统：GFS(Google File System)、MapReduce、BigTable
  - Hadoop主要由HDFS(Hadoop Distributed File System Hadoop分布式文件系统)、MapReduce和HBase组成
- Hadoop的初衷是为解决 Nutch 的海量数据爬取和存储的需要
- Hadoop于2005年秋天作为 Lucene的子项目Nutch的一部分 Apache基金会。
- 名称起源：Doug Cutting儿子的黄色大象玩具的名字



# Hadoop优势

---

- **可扩展**: 不论是存储的可扩展还是计算的可扩展都是Hadoop的设计根本;
- **经济**: 它在通常可用的计算机集簇间分配数据和处理，这些集簇可以被计入选以千计的节点当中
- **高效**: 通过分配数据，Hadoop能够在存放数据的节点之间平行的处理它们，因此其处理速度非常快。
- **可信**: Hadoop能够自动保存数据的多份副本，并且能够自动地将失败的任务重新分配

# Hadoop概述

---

- Hadoop是一个分布式系统基础架构，是一个能够对大量数据进行分布式处理的软件框架，由Apache基金会开发。用户可以在不了解分布式底层细节的情况下，开发分布式程序，充分利用集群的威力高速运算和存储。

Hadoop框架中最核心的设计就是：**MapReduce**和**HDFS**。

# Spark

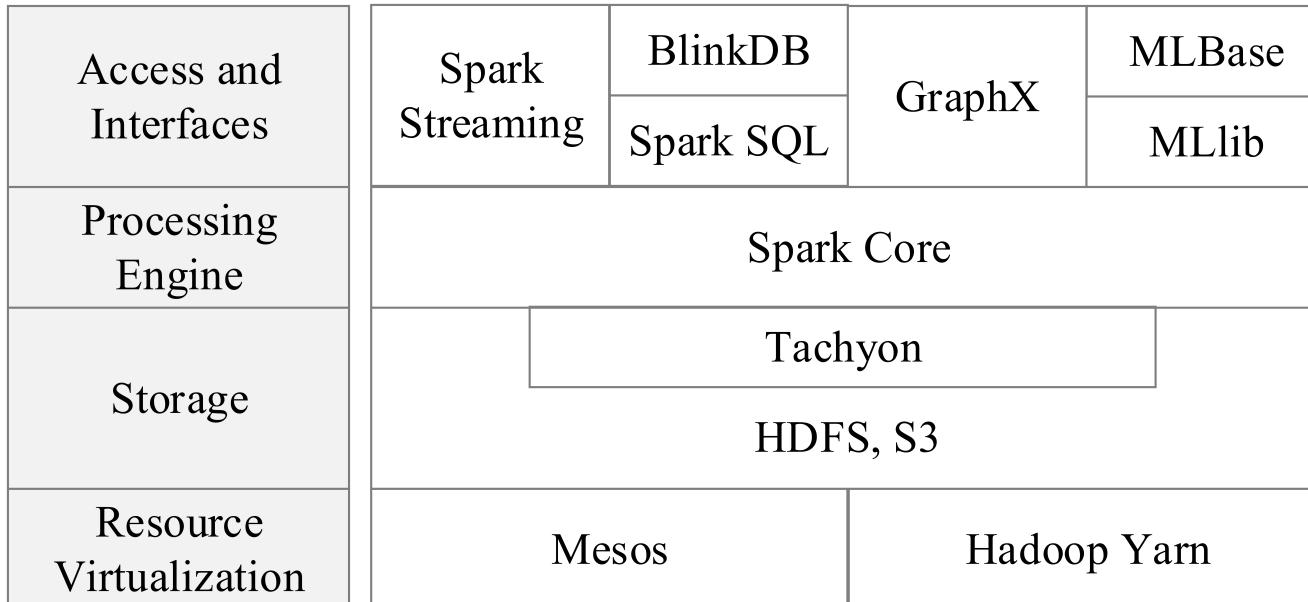
---

- Spark是UC Berkeley AMP Lab所开源的类Hadoop MapReduce的通用并行框架，Spark
- Spark中Job中间输出结果可以保存在内存中，从而不再需要读写HDFS，因此Spark能更好地适用于数据挖掘与机器学习等需要迭代的MapReduce的算法
- 可以理解为运行在HDFS上的Scala虚拟机

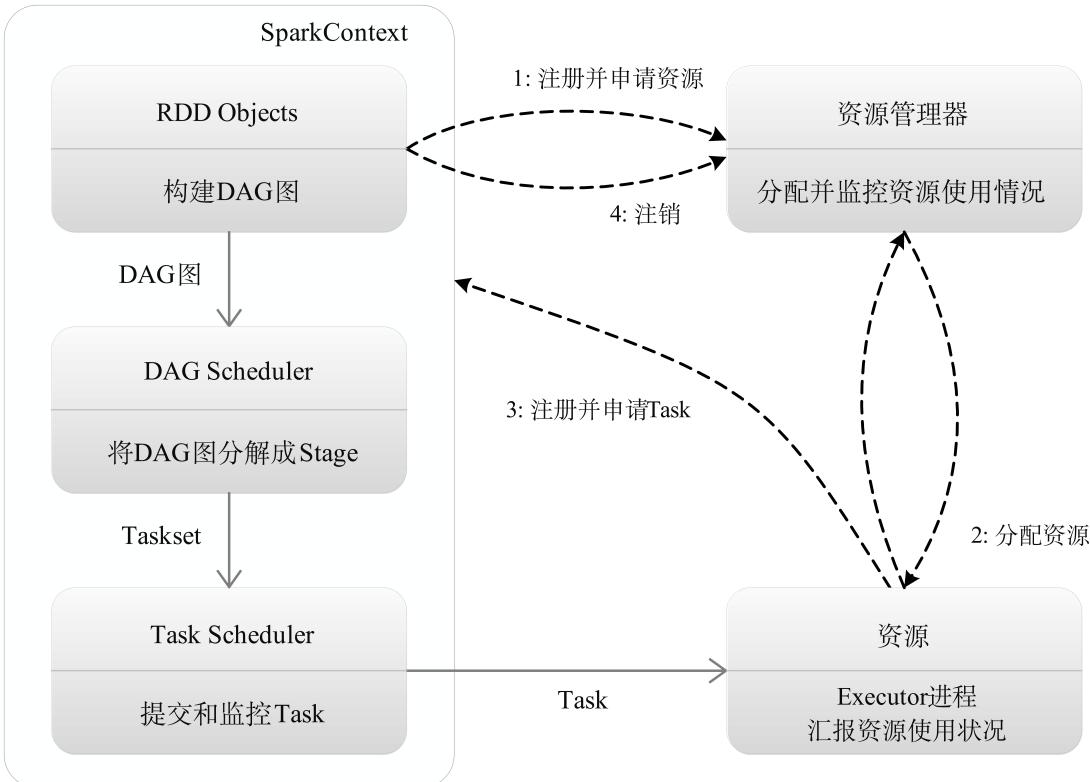
# Spark

---

- Spark的生态系统主要包含了Spark Core、Spark SQL、Spark Streaming、MLlib和GraphX 等组件



# Spark运行基本流程



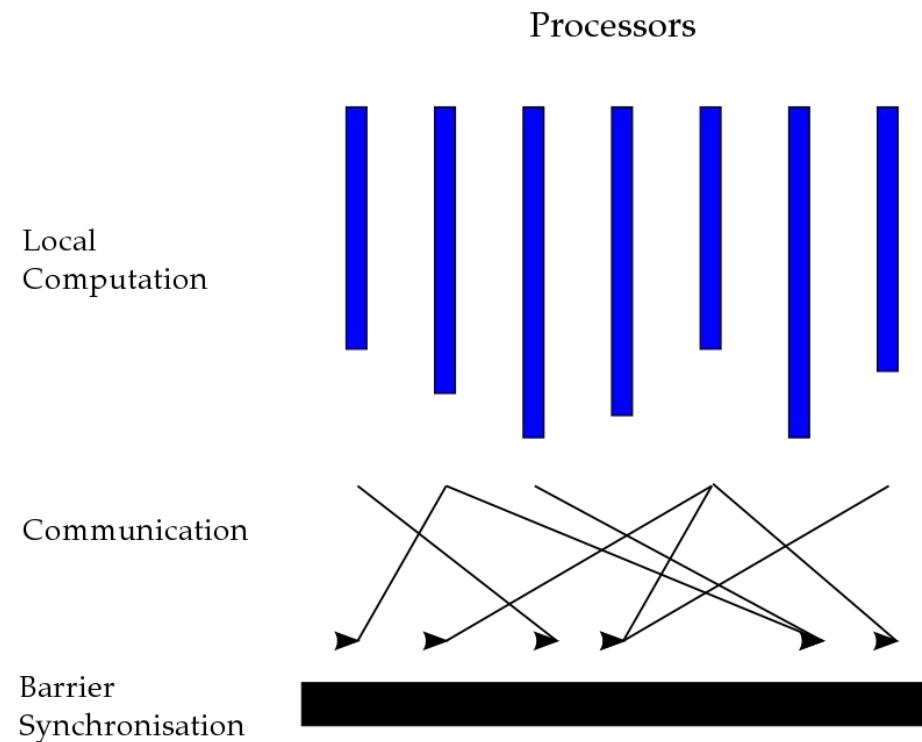
- (1) 首先为应用构建起基本的运行环境，即由Driver创建一个SparkContext，进行资源的申请、任务的分配和监控
- (2) 资源管理器为Executor分配资源，并启动Executor进程
- (3) SparkContext根据RDD的依赖关系构建DAG图，DAG图提交给DAGScheduler解析成Stage，然后把一个个TaskSet提交给底层调度器TaskScheduler处理；Executor向SparkContext申请Task，Task Scheduler将Task发放给Executor运行，并提供应用程序代码
- (4) Task在Executor上运行，把执行结果反馈给TaskScheduler，然后反馈给DAGScheduler，运行完毕后写入数据并释放所有资源

# Pregel

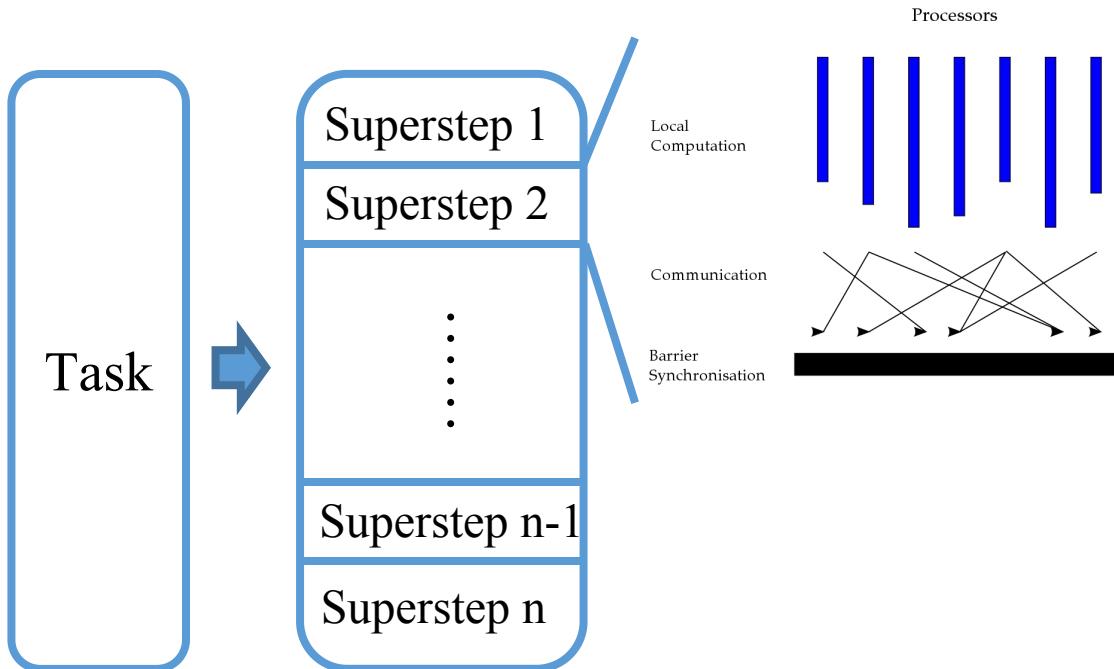
---

- 针对大规模网络上的网页链接分析等实际问题，Pregel是谷歌提出的大规模分布式图计算平台
- Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, Grzegorz Czajkowski: Pregel: a system for large-scale graph processing. SIGMOD Conference 2010: 135–146

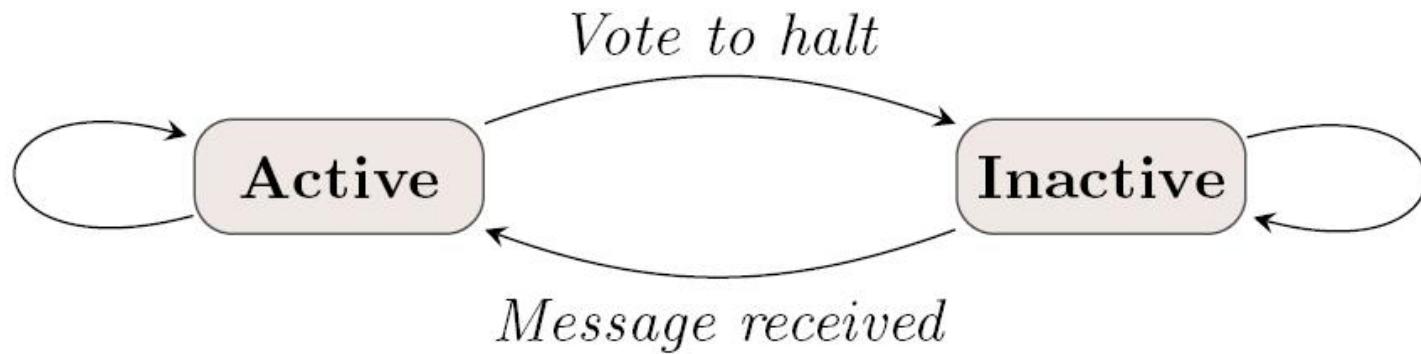
# 计算模型 (BSP)



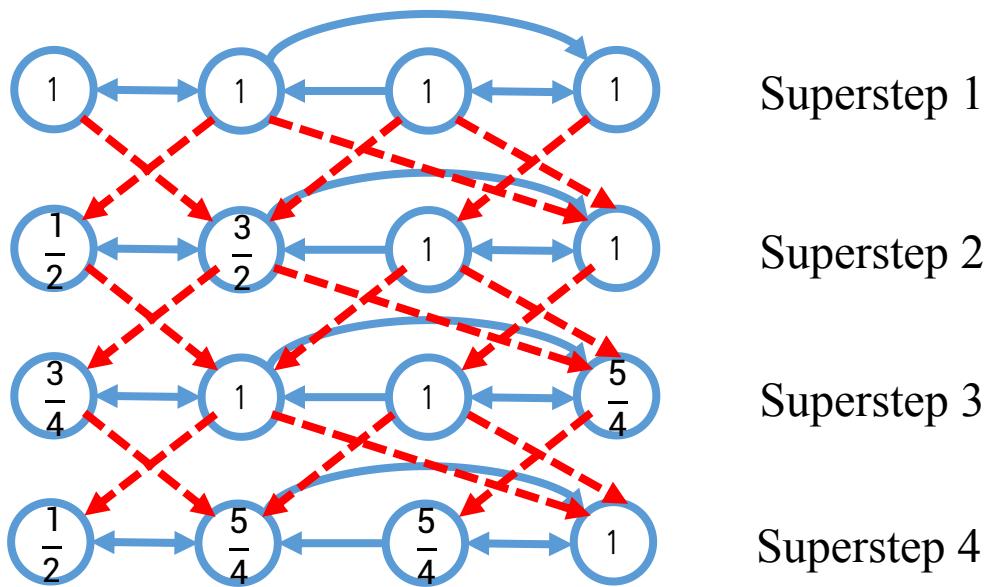
# Pregel



# 以点为核心的计算模型



# 示例



Superstep 1

Superstep 2

Superstep 3

Superstep 4

# 基于Hadoop的分布式知识图谱管理方法

---

- 这一类方法又可细分为两类
  - 基于三元组的存储，如SHARD
  - 基于图的存储，如EAGRE

# SHARD

- SHARD以知识图谱数据中的主体为核心进行数据划分，与一个主体相关的所有三元组被聚集起来并被存储为HDFS文件中的一行

Aristotle	influencedBy	Plato
Aristotle	mainInterest	Ethics
Aristotle	name	"Aristotle"
Aristotle	placeOfDeath	Chalcis



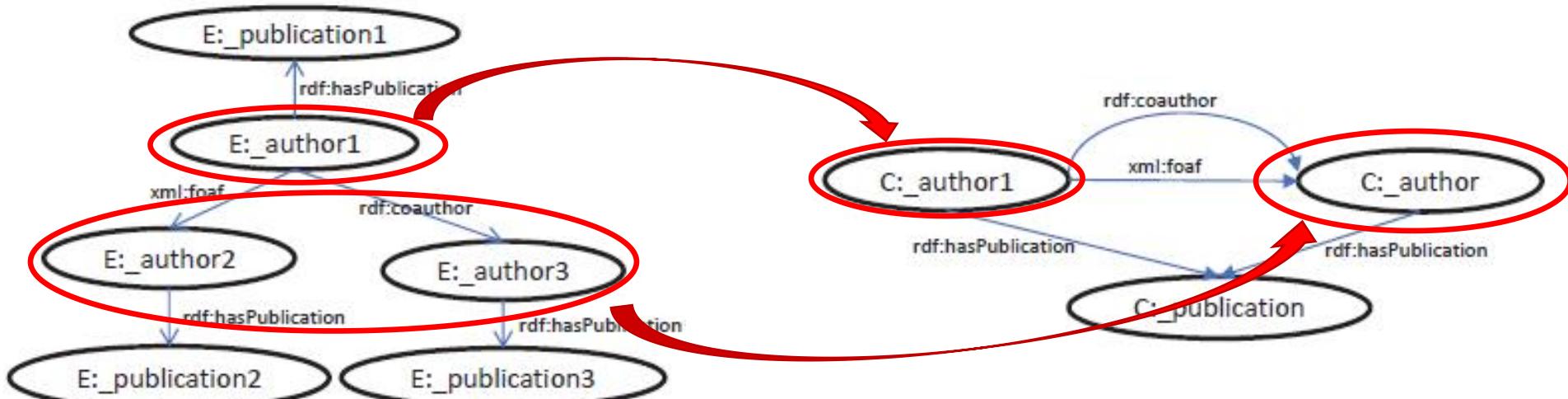
Aristotle influencedBy Plato mainInterest Ethics name "Aristotle" placeOfDeath Chalcis

HDFS



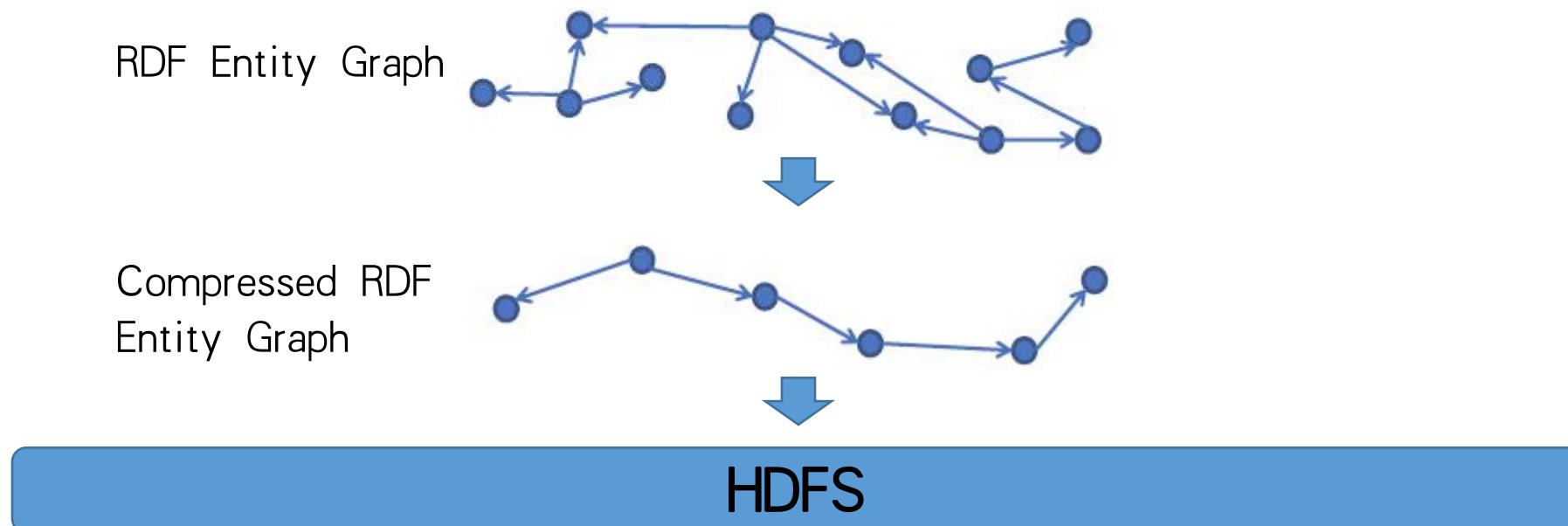
# EAGRE

- EAGRE 将知识图谱中相似的实体聚类成一个实体类，进而形成一个压缩实体图



# EAGRE

- 上述压缩实体图存储在内存中，并利用METIS 进行图划分，然后根据划分结果将RDF 实体以及相应三元组放到不同机器上



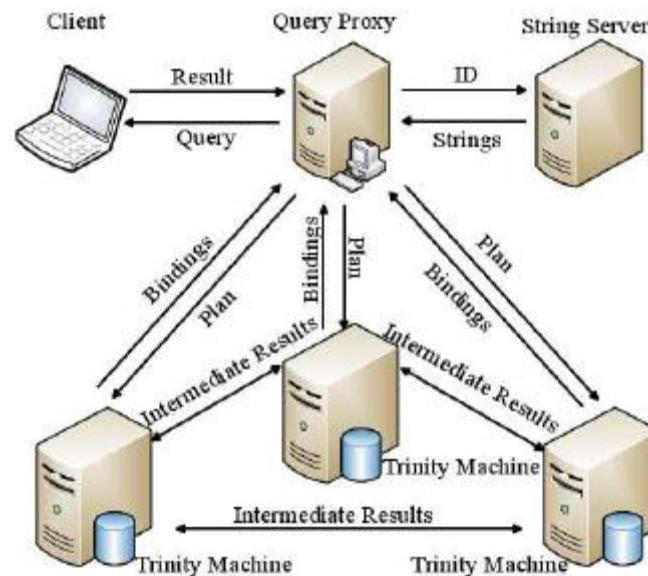
# 基于其他云平台的分布式知识图谱存储方法

---

- 基于HBase 的H2RDF
- 基于Trinity 的Trinity.RDF、Stylus
- 基于Spark 的S2RDF

# Trinity .RDF

- Trinity是微软开发的一个分布式内存的图数据管理系统
- 知识图谱可以直接以图的方式存在Trinity上

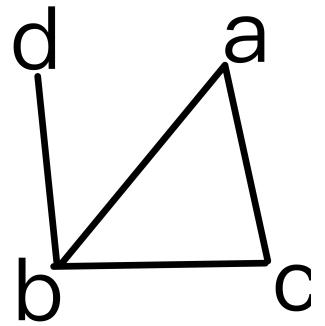


# 查询处理流程

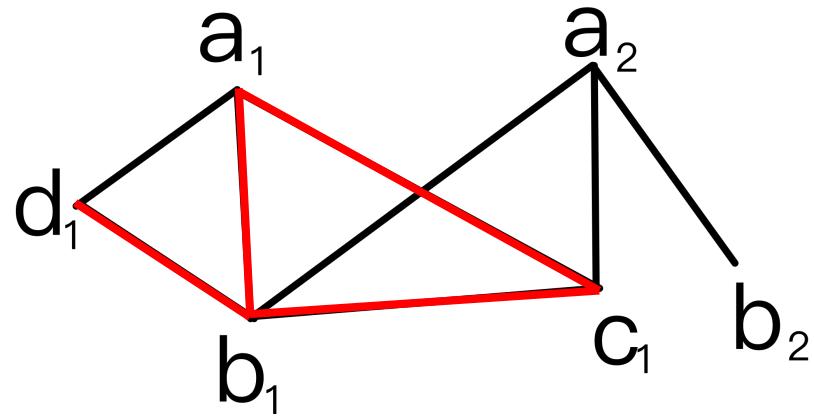
---

1. 将图查询分解为小的基本单元
2. 并行地在数据图中匹配这些基本单元
3. 联结所得结果

# 子图匹配

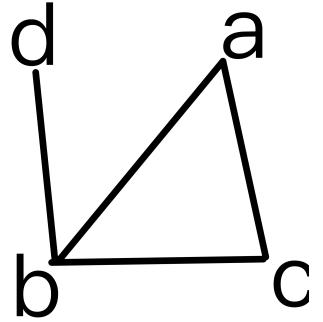


$G_1$

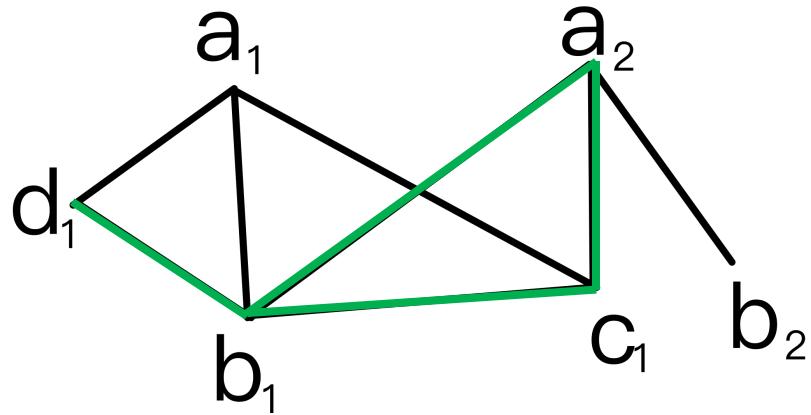


$G_2$

# 子图匹配

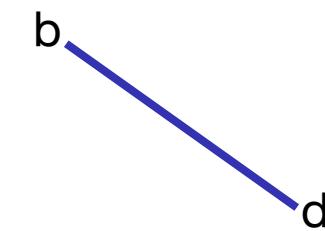
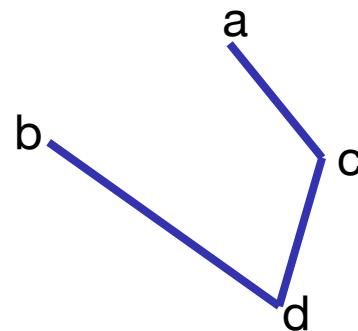
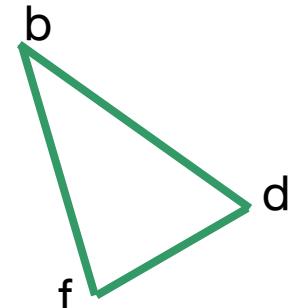
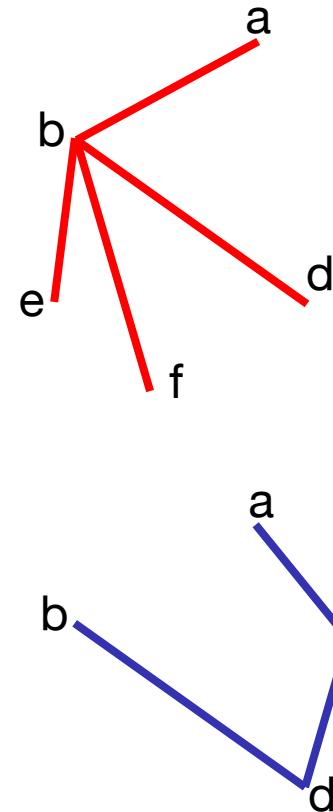
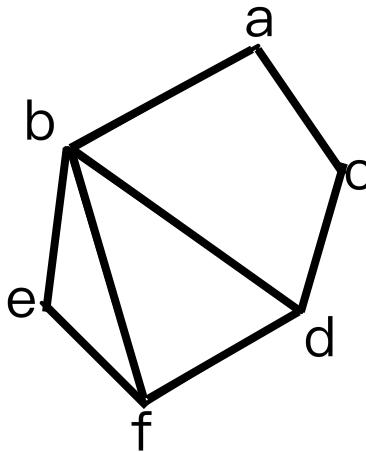


$G_1$



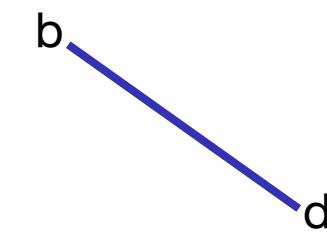
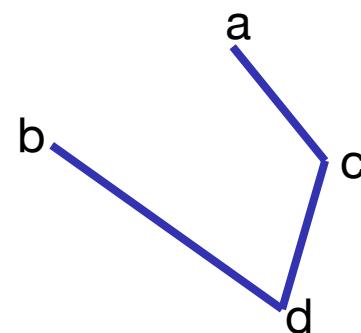
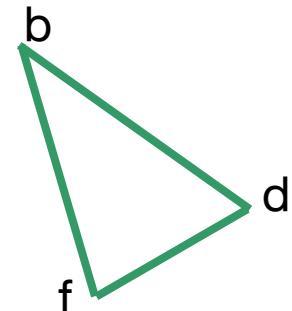
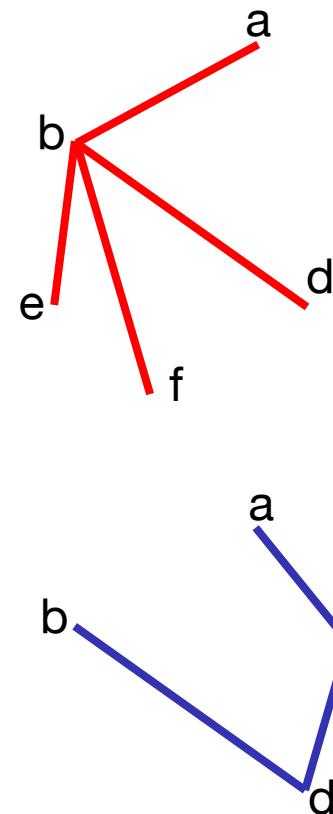
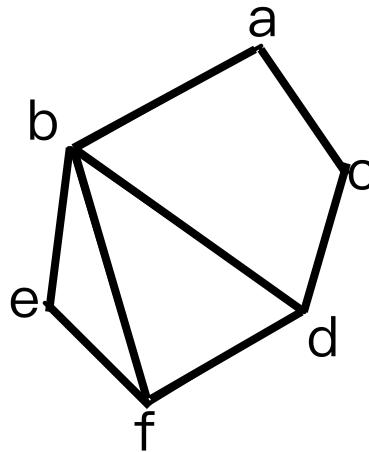
$G_2$

# 分布式子图匹配的基本单元



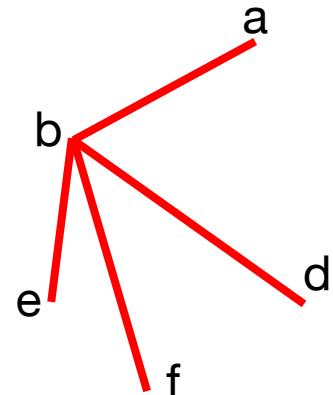
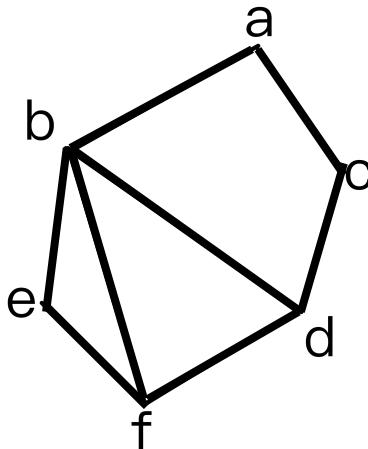
作为基本单元，哪一个才是最优的？

# 分布式子图匹配的基本单元



作为基本单元，哪一个才是最优的？

# 分布式子图匹配的基本单元

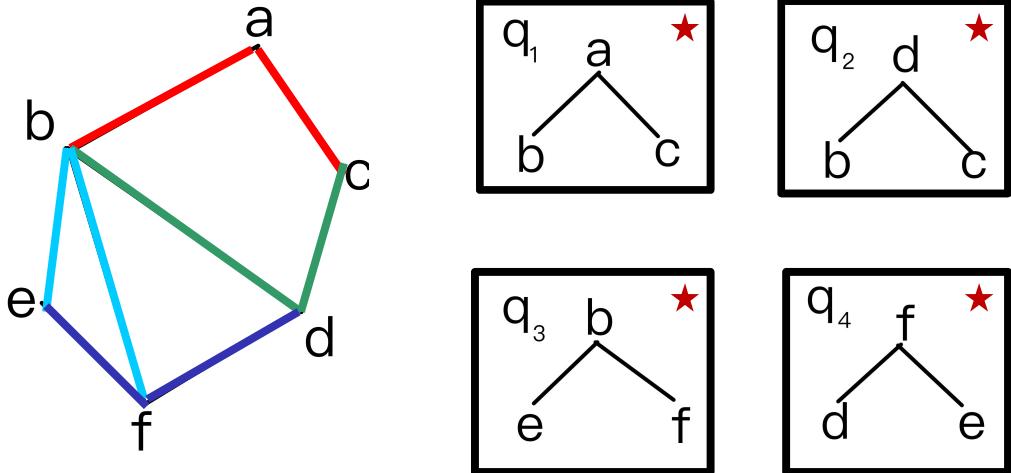


Twig

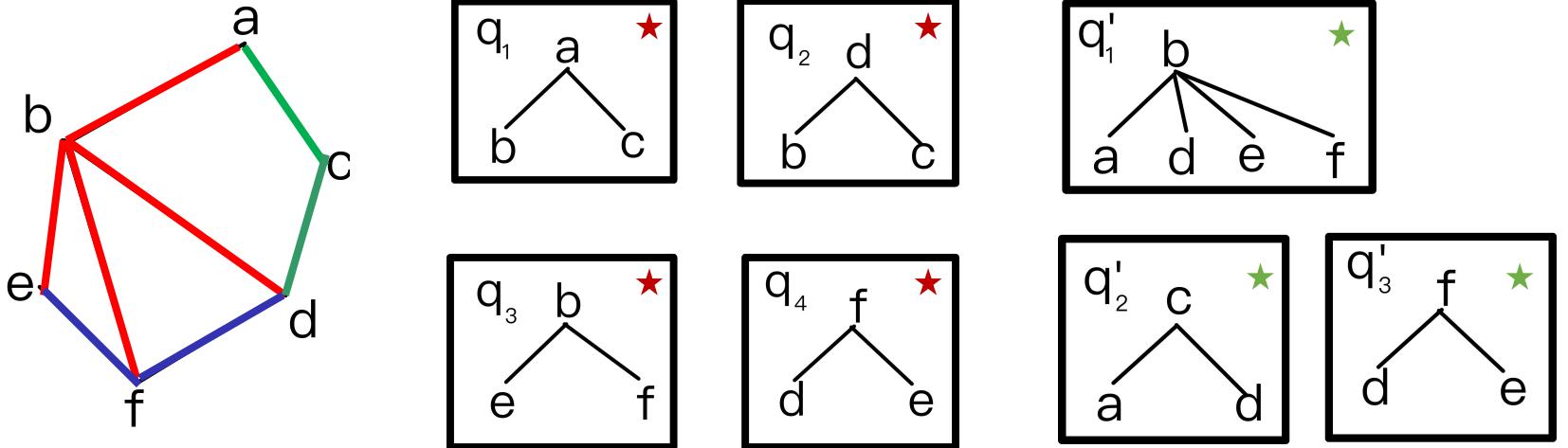
- 容易拆解
- 高度为一
- 最多一次跨网络访问

作为基本单元，哪一个才是最优的？

# 查询分解

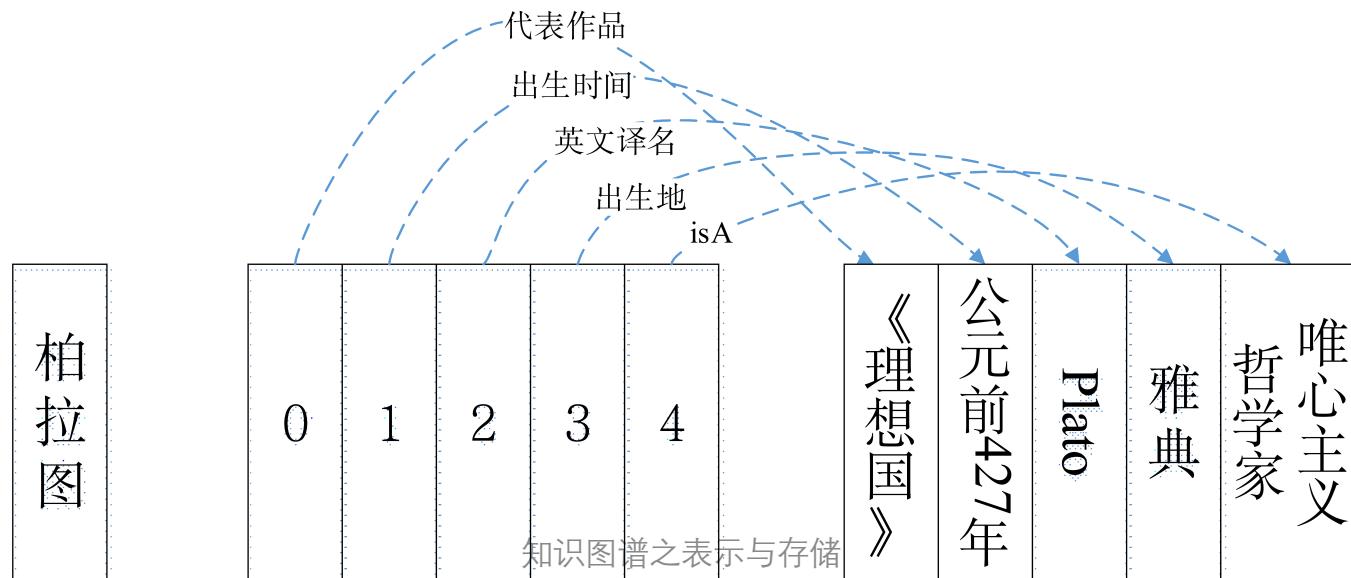


# 查询分解



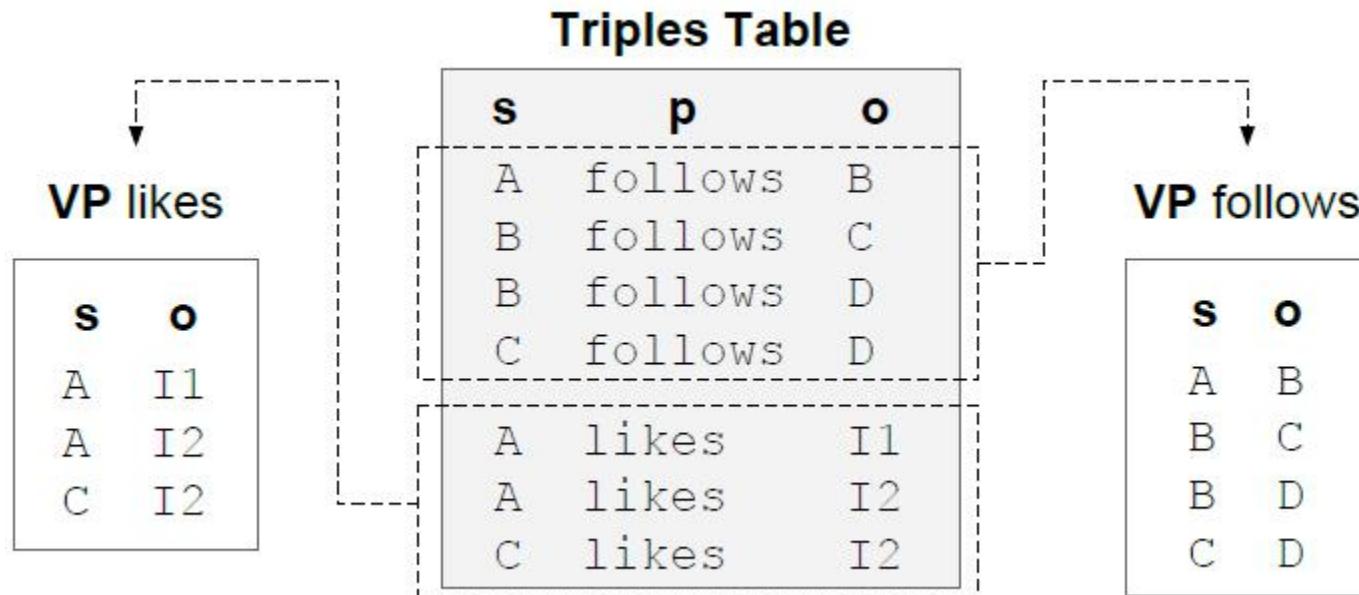
# Stylus

- 在邻接表的基础上，Stylus进一步优化图的邻接表的组织形式
- 本文定义一个叫xUDT的数据结构，用来将不同点的邻居信息按照其所相邻属性组织成不同形式



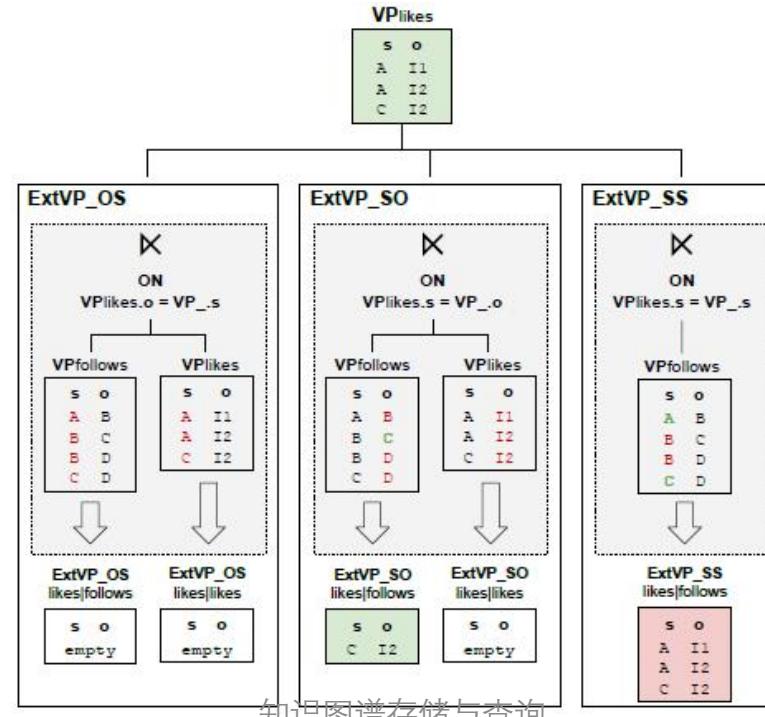
# S2RDF

- S2RDF利用Spark的关系数据库接口进行知识图谱数据管理
- S2RDF利用垂直划分对知识图谱数据进行划分



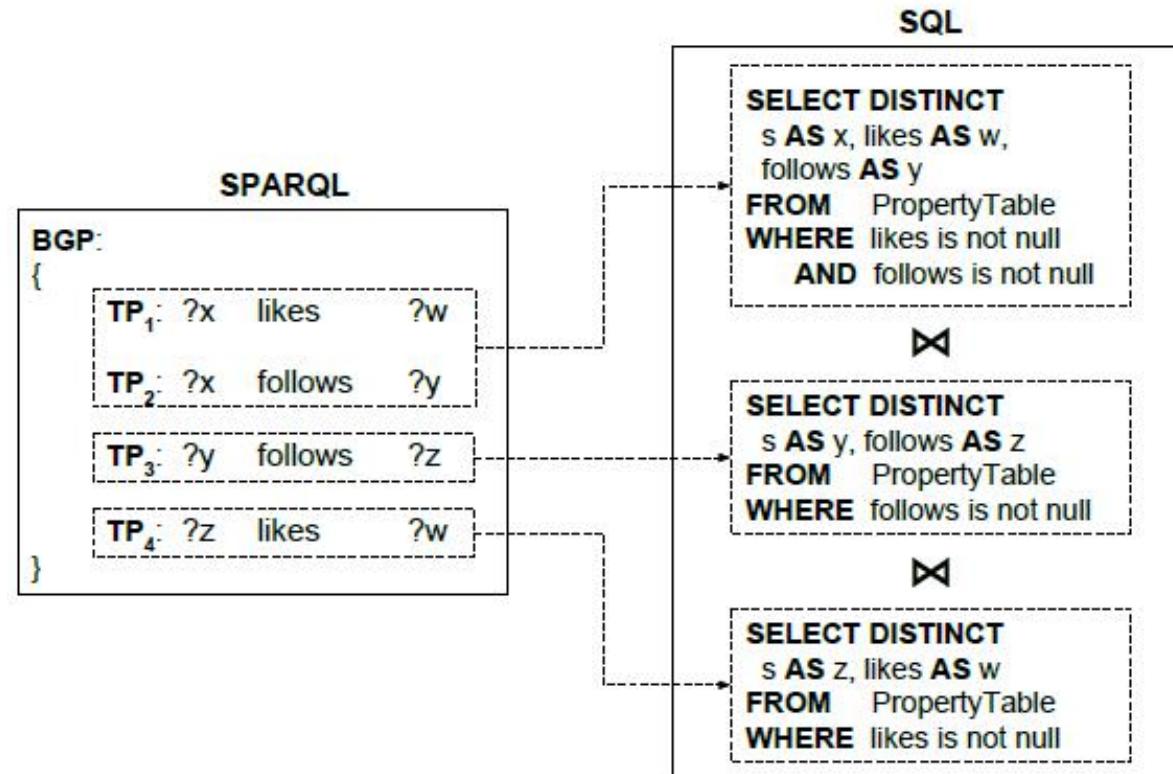
# S2RDF

- 在基本垂直划分基础上，S2RDF物化了部分垂直划分数据表之间的连接结果并也存储在关系数据表



# S2RDF

- 查询分解成基于垂直划分的子查询，并转化成SQL



# 《知识图谱：概念与技术》

谢谢！

