

《知识图谱: 概念与技术》

第 9 章

知识图谱 表示与存储

彭鹏

湖南大学

hnu16pp@hnu.edu.cn

本章大纲

- 图论基础
- 知识图谱表示
- 知识图谱存储

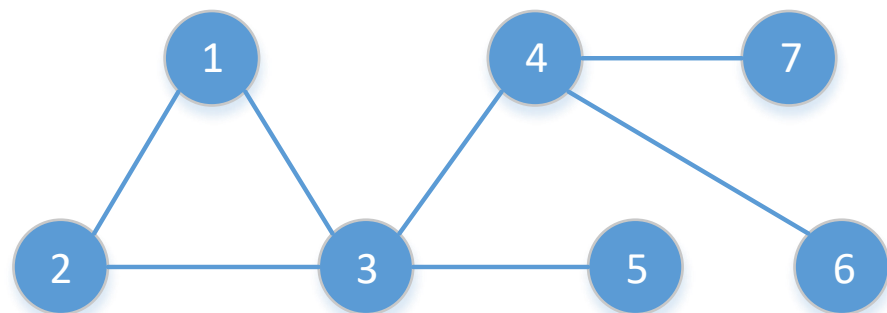
图论基础

图的定义

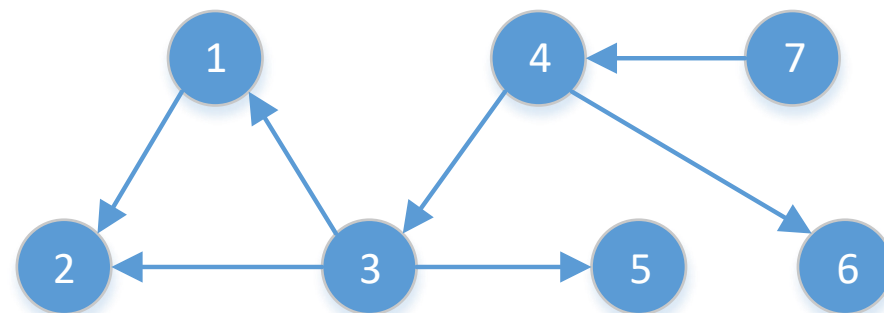
- 图可以表示成一个二元组 $G = \langle V, E \rangle$ ，其中 V 是点集合， $E \subseteq V \times V$ 表示边集合，其中每条边由 V 中两个点相连接所构成
- 此外， G 中的边 e 可以被赋上一个权重，记为 $w(e)$

无向图和有向图

➤ 如果图中的边都没有方向，那么 G 被称为无向图



➤ 如果图中的边有方向，那么 G 被称为有向图

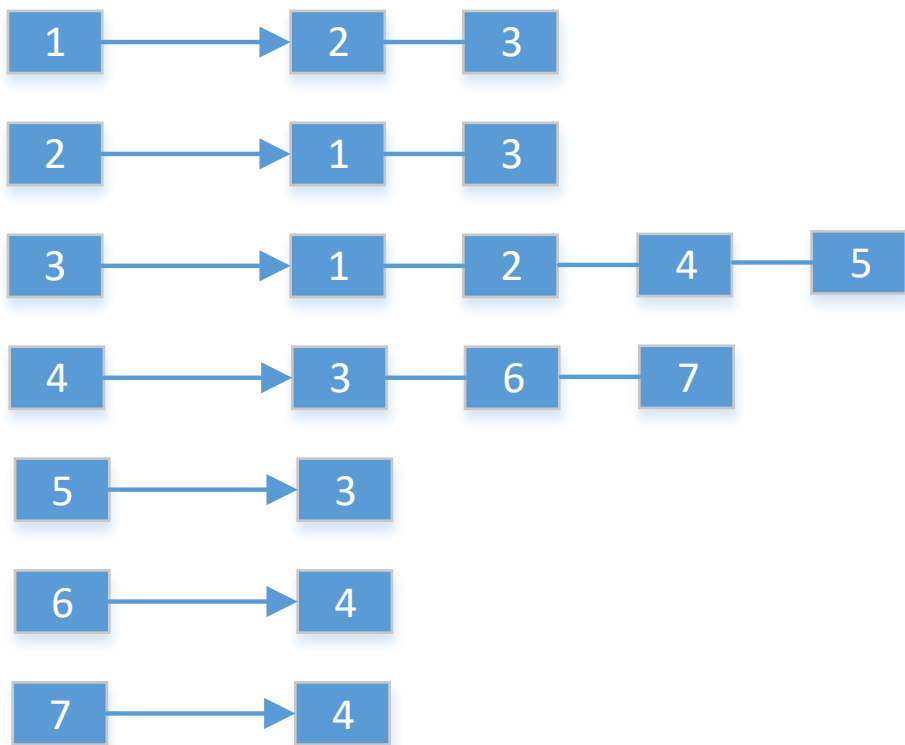


点的度数

- 给定一个无向图 $G=(V, E)$, 对于任意的 $v \in V$, 称 v 作为 G 中边的端点的次数之和为 v 的**度数**, 简称**度**, 记作 $\deg_G(v)$, 在不发生混淆的情况下, 也可以简写为 $\deg(v)$
- 给定一个有向图 $G=(V, E)$, 对于任意的 $v \in V$, 称 v 作为 G 中边的起点的次数之和为 v 的**出度**, 记作 $\deg_G^+(v)$, 简记作 $\deg^+(v)$; 称 v 作为 G 中边的终点的次数之和为 v 的**入度**, 记作 $\deg_G^-(v)$, 简记作 $\deg^-(v)$ 。
 $\deg^+(v) + \deg^-(v)$ 被称为 v 的**度数**, 记作 $\deg(v)$

图的表示

邻接表



邻接矩阵

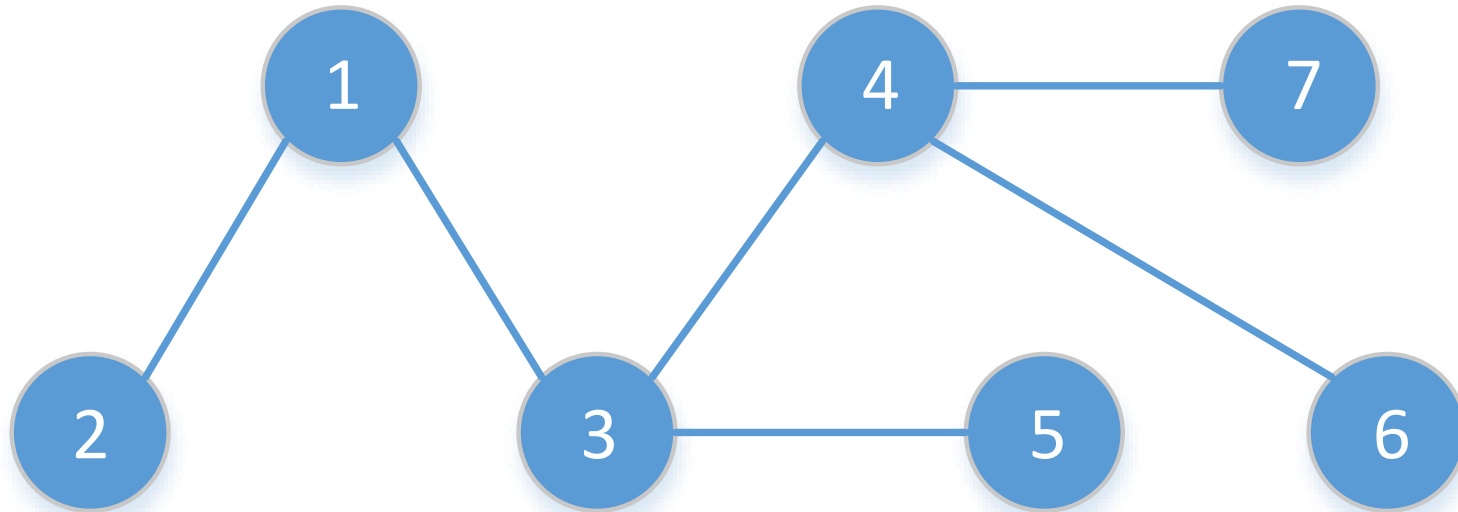
0	1	1	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	1	0	0
0	0	1	0	0	1	1
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	1	0	0	0

图上的路径

- 给定一个无向图 $G = (V, E)$, G 中顶点与边的交替序列 $\pi = v_0 e_1 v_1 e_2 \dots e_l v_l$ 称为点 v_0 到点 v_l 的**通路**, 其中 v_{r-1} 和 v_r 是 e_r 的端点 ($1 \leq r \leq l$) ; 若 v_0 和 v_l 是同一个点, 则 π 被称为一条**回路**。
- 有向图中**通路**的定义与无向图中定义类似, 只是要注意在有向图中通路和回路中边的方向的一致性, 即 v_{r-1} 必须是 e_r 的起点且 v_r 是 e_r 的终点 ($1 \leq r \leq l$)

树

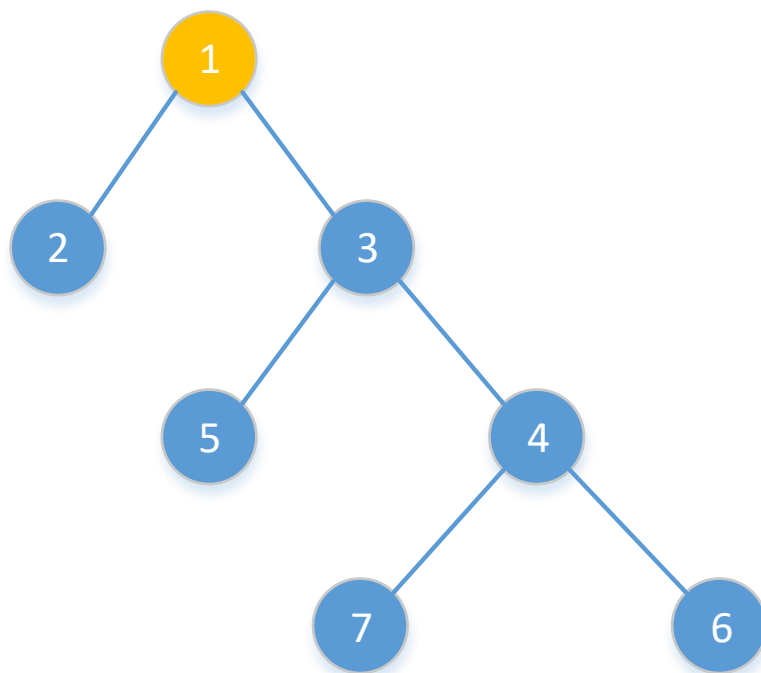
- 任意两个顶点间有且只有一条路径的图称为**树**。或者说，只要没有回路的连通图就是**树**
- **森林**是指互不相交并树的集合



有根树

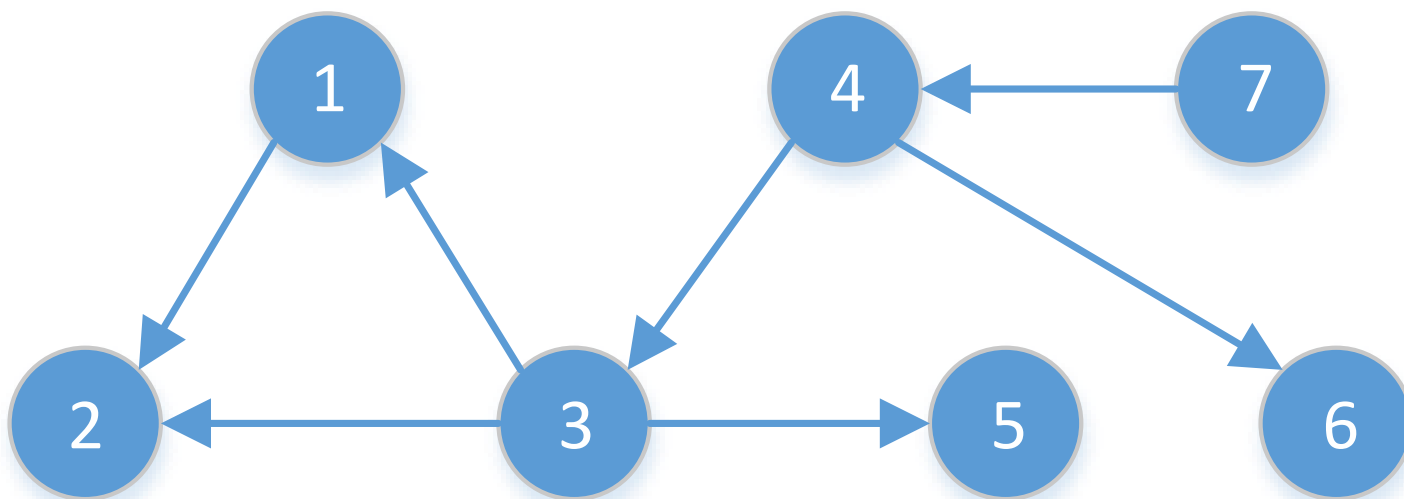
- 一棵树中可以指定一个特殊的点：**根**。一个有根的树叫**有根树**
- 有根树中的点可以根据到根的距离分**层**。一颗有根树的层数叫做这棵树的**高度**。点最多的那一层的点数叫做这棵树的**宽度**。
- 一条边的两个端点中，靠近根的那个点叫做另一个点的**父亲点**，相反的，距离根比较远的那个点叫做另一个点的**孩子点**
- 父亲方向的所有点都叫做这个点的**祖先**，儿子方向的所有点都叫做这个点的**子孙**。没有子点的子点叫做**叶子点**

有根树示例



有向无环图

- 如果一个有向图从任意顶点出发无法经过若干条边回到该点（即不存在回路），则这个图是一个**有向无环图**



概率图^[28]

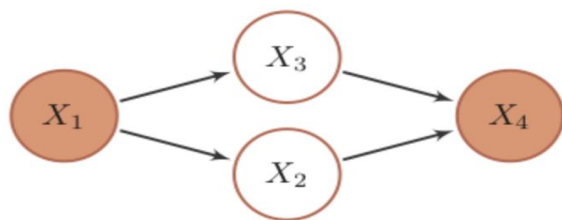
- 概率图模型 (Probabilistic Graphical Model, PGM)，是指一种用图结构来描述多元随机变量之间条件独立关系的概率模型。在概率图中，每个节点表示一个/组随机变量，边表示随机变量之间的概率依赖关系
- 核心思想：
 - 通过独立性假设有效减少参数量；
 - 使用图结构简单、直观地描述随机变量之间的条件独立性，并将复杂的联合概率模型分解成一些简单条件概率模型的组合。

概率图实例

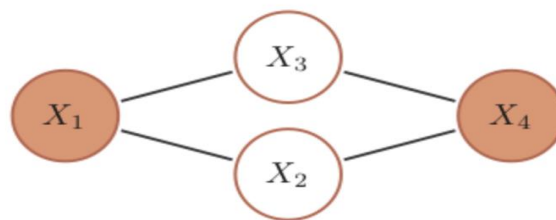
- 假设四个二值变量 X_1, X_2, \dots, X_k ，若不知道变量依赖关系的情况下，可以用一个联合概率表来记录每一种取值的概率 $P(X_{1:4})$ ，共 2^4-1 共15个参数。
- 若假设给定 X_1 时， X_2 与 X_3 独立；给定 X_2 和 X_3 时， X_4 和 X_1 独立，则：
 - $p(x_2|x_1, x_3) = p(x_2|x_1)$
 - $p(x_3|x_1, x_2) = p(x_3|x_1)$
 - $p(x_4|x_1, x_2, x_3) = p(x_4|x_2, x_3)$
- 则联合概率 $p(x)$ 可分解为：
 - $$\begin{aligned} p(x) &= p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)p(x_4|x_1, x_2, x_3) \\ &= p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2, x_3) \end{aligned}$$
- 只需 $1+2+2+4=9$ 个独立参数

概率图实例

- 常见的概率图模型可分为有向图模型与无向图模型：
 - 有向图模型图结构为有向无环图，每条边表示了两个变量的因果关系；
 - 无向图模型中每条边代表两个变量之间有概率依赖关系。
- 下图分别给出了两种图模型下 $X_{1:4}$ 条件独立性的可视化描述：



(a) 有向图：贝叶斯网络



(b) 无向图：马尔可夫随机场

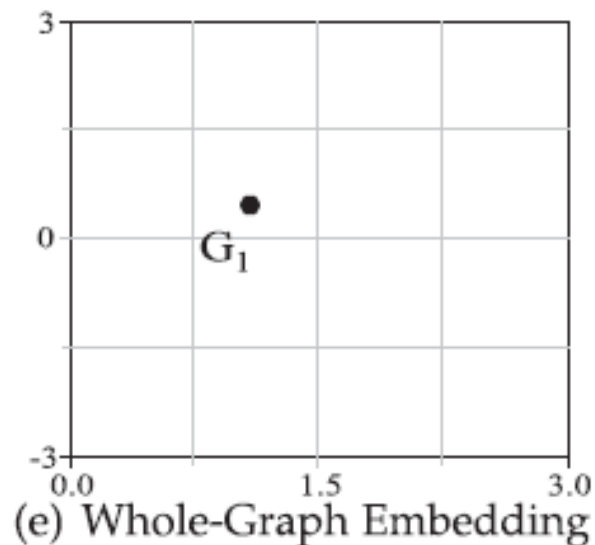
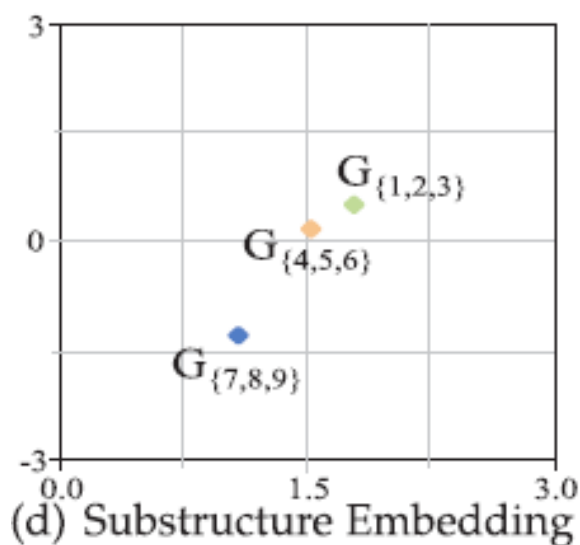
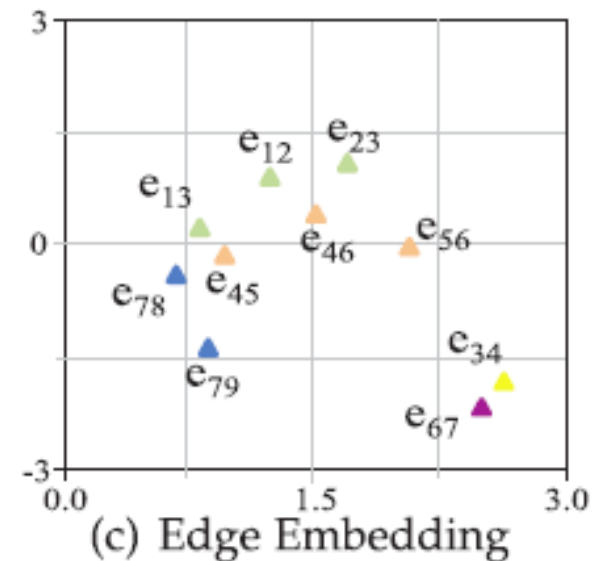
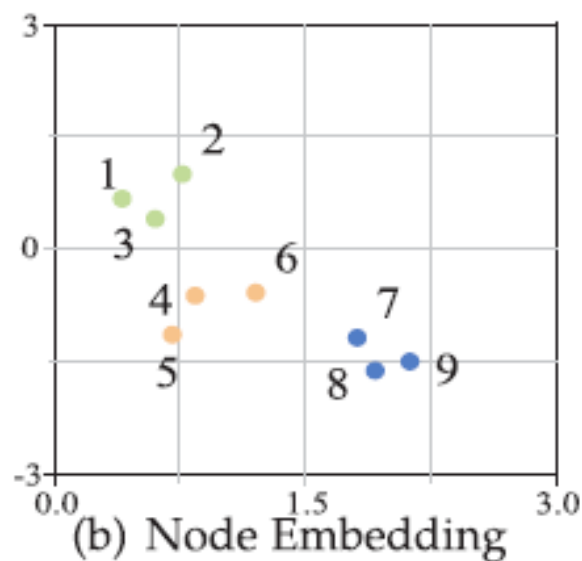
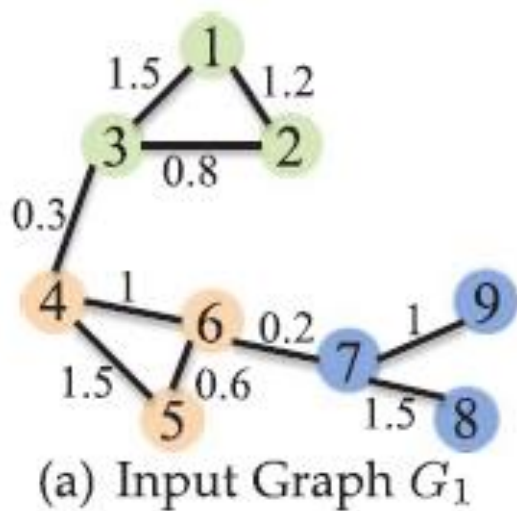
有向图和无向图示例。带阴影的节点表示可观测到的变量，不带阴影的节点表示隐变量，连边表示两变量间的条件依赖关系

三个基本问题

- 表示问题：如何通过图结构来描述一个概率模型中变量之间的依赖关系；
- 学习问题：包括图结构与参数的学习，其中，给定图结构下计算联合分布 $p(x_1, x_2, \dots, x_k)$ ，其中 x_1, \dots, x_k 为模型中的参数
- 推断问题：已知部分变量时，计算其他变量的后验概率分布

图嵌入 (Graph Embedding)

- 在**维持图的性质**的情况下将图转化成**低维向量**
- 常见四种嵌入方式：**点嵌入** (Node Embedding) 、 **边嵌入** (Edge Embedding) 、 **子图嵌入** (Substructure Embedding) 、 **整图嵌入** (Whole-Graph Embedding)



H. Cai, V. W. Zheng, and K. C. Chang, A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications, IEEE Trans. Knowl. Data Eng., 30 (9) (2018) 1616-1637

常见图嵌入方法

- 基于矩阵分解的方法
- 基于随机游走的方法
- 基于深度学习的方法

知识图谱逻辑表示

知识

- 知识是结构化的经验、价值、相关信息和专家洞察力的融合，提供了评价和产生新的经验和信息的框架
- 所谓知识表示，构建一个符号系统来将知识符号化、形式化或模型化

Knowledge Representation Hypothesis implies that we will want to construct systems for which the intentional stance is grounded by design in symbolic representations.

——《*Knowledge Representation and Reasoning*》，
Ronald J. Brachman, Hector J. Levesque

产生式规则表示法

- 简介

- 是**专家系统**中应用最广泛的一种知识表示方法
- 最适合表示各种启发式的**经验性规则**，领域专家可以无须知识工程工具就能把自己的知识转换成规则的形式

- 表示形式

IF a THEN b

- a称为**前件**，表示**前提条件**，各个条件由**逻辑连接词**（合取、析取等）组成各种不同的组合
- b称为**后件**，表示当前提条件**为真**时，应采取的行动或所得的**结论**

举例： 动物识别系统IDENTIFIER

- R1: IF 有毛发 THEN 是哺乳动物
- R2: IF 能产乳 THEN 是哺乳动物
- R3: IF 有羽毛 THEN 是哺乳动物
- R4: IF 能飞行 AND 能生蛋 THEN 是鸟类动物
-

一阶逻辑

- 一阶逻辑也叫一阶谓词演算，允许量化陈述的公式，是一种形式系统
- 存在两种合法的表示式：**项**和**公式**
- 比如：人都会死，表示成 $\forall x (M(x) \rightarrow H(x))$
苏格拉底是人，表示成 $M(s)$
苏格拉底会死的，表示成 $M(s) \rightarrow H(s)$

马尔科夫逻辑网^[29]

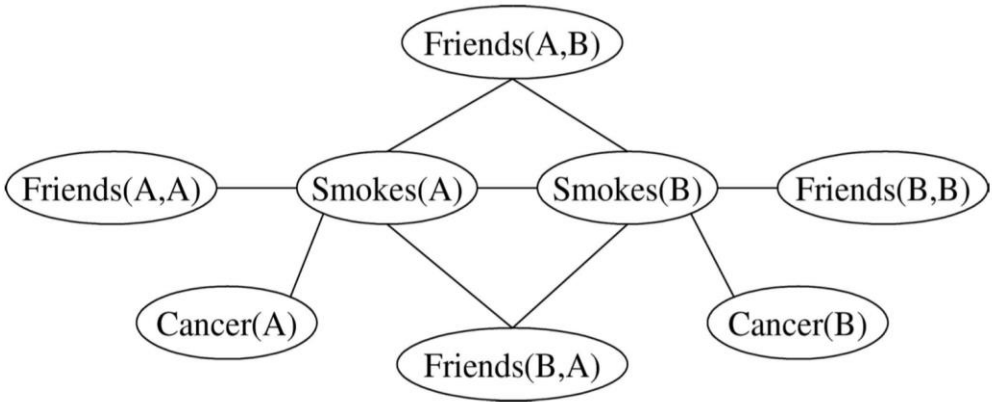
- 马尔科夫逻辑网(Markov Logic Networks, MLNs) 是一种把概率图模型与一阶谓词逻辑结合起来的统计关系学习模型
- 核心思想：
 - 一阶谓词逻辑知识库即为在一个可能世界的集合上建立了硬性规则，如果一个世界违反了其中的某一条规则，那么这个世界的存在概率即为零。；
 - 通过为规则加上一个特定权重的方法将一阶谓词逻辑规则中的硬性约束 (hard constraints) 进行软化，权重越大，约束能力越强，权重无限大即为硬性规则；
 - MLNs实质上是一个规则-权重对的集合。
- 应用：
 - 对知识图谱建模后，进行事实、规则及其权重的学习与推断。

马尔科夫逻辑网实例

- 马尔科夫逻辑网的定义可以得到一个图结构，原子事实即图中的节点，节点之间的边则表示相邻两个事实之间存在于同一规则中

Table 1 Example of a first-order knowledge base and MLN. Fr() is short for Friends(), Sm() for Smokes(), and Ca() for Cancer()

English	First-order logic	Clausal form	Weight
Friends of friends are friends	$\forall x \forall y \forall z \text{ Fr}(x, y) \wedge \text{Fr}(y, z) \Rightarrow \text{Fr}(x, z)$	$\neg \text{Fr}(x, y) \vee \neg \text{Fr}(y, z) \vee \text{Fr}(x, z)$	0.7
Friendless people smoke	$\forall x (\neg(\exists y \text{ Fr}(x, y)) \Rightarrow \text{Sm}(x))$	$\text{Fr}(x, g(x)) \vee \text{Sm}(x)$	2.3
Smoking causes cancer	$\forall x \text{ Sm}(x) \Rightarrow \text{Ca}(x)$	$\neg \text{Sm}(x) \vee \text{Ca}(x)$	1.5
If two people are friends, either both smoke or neither does	$\forall x \forall y \text{ Fr}(x, y) \Rightarrow (\text{Sm}(x) \Leftrightarrow \text{Sm}(y))$	$\neg \text{Fr}(x, y) \vee \text{Sm}(x) \vee \neg \text{Sm}(y),$	1.1
		$\neg \text{Fr}(x, y) \vee \neg \text{Sm}(x) \vee \text{Sm}(y)$	1.1



概率软逻辑

- 概率软逻辑是马尔科夫逻辑网的延伸，使用“软”逻辑建模不确定的规则与事实，使原子事实的真值在 $[0,1]$ 区间内取连续值。
- PSL使用软逻辑作为逻辑组成部分，并使用马尔科夫网作为其统计模型。
- 优势：
 - 使用简单逻辑语法定义模型，通过快速凸优化进行运算；
 - 增强了MLNs的不确定性处理能力，能在其基础上进一步建模不确定性的规则与事实；
 - 使用连续的软真值将MLNs的离散优化问题简化为连续优化问题。
- 应用：
 - 对知识图谱建模后，进行事实、规则及其权重的学习与推断。

概率软逻辑实例^[30, 31]

- 不确定规则


- 0.3: $\text{friend}(B, A) \wedge \text{votesFor}(A, P) \rightarrow \text{votesFor}(B, P)$

- 0.8: $\text{spouse}(B, A) \wedge \text{votesFor}(A, P) \rightarrow \text{votesFor}(B, P)$

- 不确定事实

- $I(\text{spouse}(B, A)) = 1; I(\text{votesFor}(A, P)) = 0.9; I(\text{votesFor}(B, P)) = 0.3$

规则建模


$$\begin{aligned} I(r_{body}) &= I(spouse(B, A) \wedge votesFor(A, P)) \\ \bullet \quad &= \max\{0, I(spouse(B, A)) + I(votesFor(A, P)) - 1\} \\ &= \max\{0, 1 + 0.9 - 1\} = 0.9 \\ \bullet \quad I(r_{head}) &= I(votesFor(B, P)) = 0.3 \\ \bullet \quad d(r) &= \max\{0, I(r_{body}) - I(r_{head})\} = 0.6 \end{aligned}$$

框架表示法

- 简介

- 是以**框架理论**为基础发展起来的一种结构化的知识表示

- 框架理论的基本观点

- 人类对**现实世界**中各种事物的**认识**都是以**框架**的结构存储在**记忆**中
 - 当人面临**新的情景**时，就从记忆中找出一个合适的框架，这个框架是以前记忆的一个知识空框，具体内容依新的情景而**改变**，对这空框的细节**加工、修改和补充**，形成对**新情景的认识**又记忆于人脑中

框架表示形式

- 基本组成

- 框架通常由描述事物的各个方面的“槽” (slot) 组成
- 槽
 - 用于描述所论对象某一方面的属性
 - 每个槽可以拥有多个侧面
- 侧面
 - 用于描述相应属性的一个方面
 - 每个侧面可以拥有多个值
- 槽和侧面所具有的属性值分别称为槽值和侧面值

- 表示形式

- <框架名>
- <槽名1>: <槽值1>
- <槽名2>: <侧面21>: <值211>

框架名: <教师>

类属: <知识分子>

工作: 范围: (教学, 科研)

缺省: 教学

性别: (男, 女)

学历: (学士, 硕士, 博士)

类型: (<小学教师>, <中学教师>, <大学教师>)

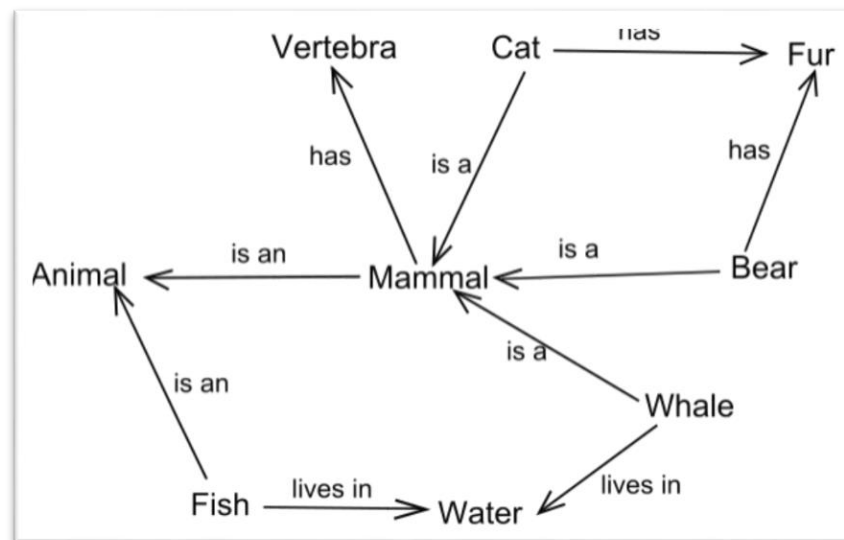
语义网络

- 简介

- 是知识表示中**最重要**的方法之一

- 组成

- 是由**节点**和连接节点的**边**所组成的有向图
 - 节点表示**概念**
 - 边表示**关系**



https://en.wikipedia.org/wiki/Semantic_network

本体

- 本体是关于共享概念的一致约定。共享概念包括用来对领域知识进行建模的概念框架、需要互操作的主体之间用于交互的与内容相关的协议，以及用于表示特定领域的理论的共同约定。在知识共享的情况下，本体的形式特化为具有代表性的词汇的定义
- 一种最简单的形式是一种层次结构，用来详细描述类和它们之间的包含关系^[1, 2]

WordNet

- WordNet是一个由普林斯顿大学建立和维护的英语字典，可以视为有向无环图

```
dog, domestic dog, Canis familiaris
├─ canine, canid
│   └─ carnivore
│       └─ placental, placental mammal, eutherian, eutherian mammal
│           └─ mammal
│               └─ vertebrate, craniate
│                   └─ chordate
│                       └─ animal, animate being, beast, brute, creature, fauna
└─ ...
```

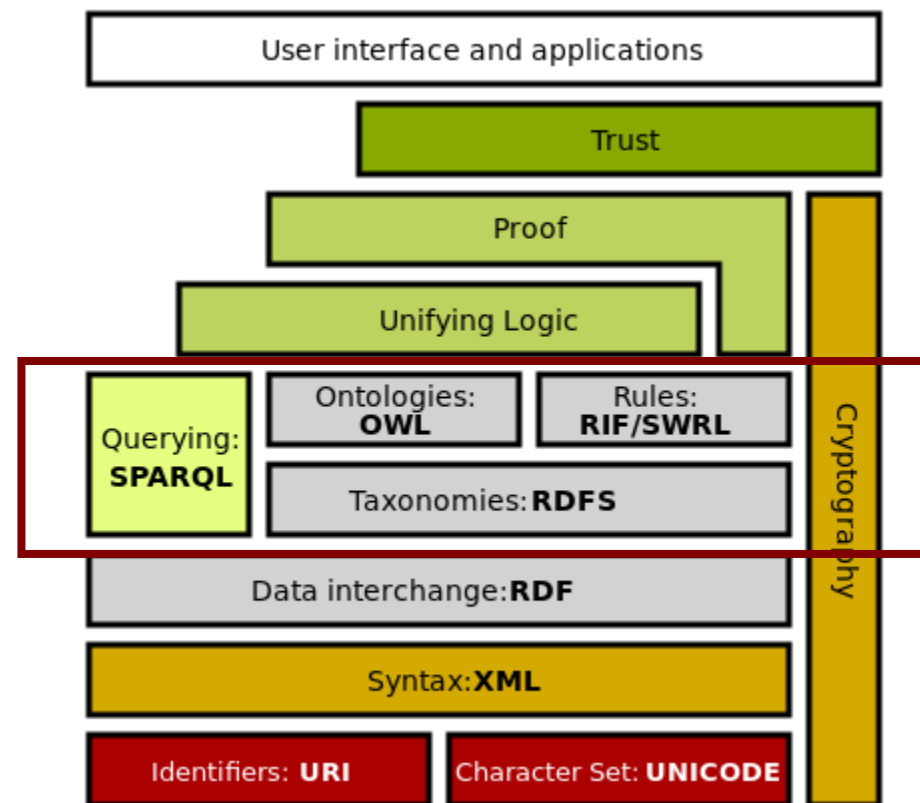
<https://en.wikipedia.org/wiki/WordNet>

本体描述语言

- Ontolingua
- Cycl
- KIF
- RDF(S)
- DAML
- OWL
-

资源描述框架(RDF)数据

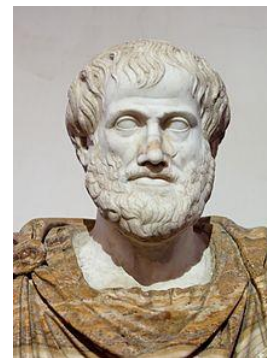
- RDF是知识图谱数据的事实标准
- RDF是由W3C组织提出的一种描述资源概念模型的语言
- RDF是语义网的一个基石 (Building Block)
- 语义网的目标是网络上的资源是“机器可理解”(Machine understandable)



RDF 数据模型

- 任何事物都被表示成资源，通过URI来标识
- 每个资源都有属性及相应属性值
- 每个资源还能和其它资源有关系
- 多个资源相互连接就形成了一个语义网络

dp:Aristotle



dp:name "Aristotle"@en
dp:dateOfBirth "384 B.C."

dp:placeOfDeath



dp:Chalcis

RDF三元组模型

- 每个资源的一个属性及属性值，或者它与其他资源的一条关系，都被称为一条知识。这些属性以及关系就能表示成三元组
- RDF 数据中的三元组又称**声明**，通常被表示为<主体, 属性, 客体>。三元组也可表示为<主体, 属性, 属性值>，这种方式主要用来描述实体的相关属性
- 一个RDF 数据集D是一系列三元组的集合

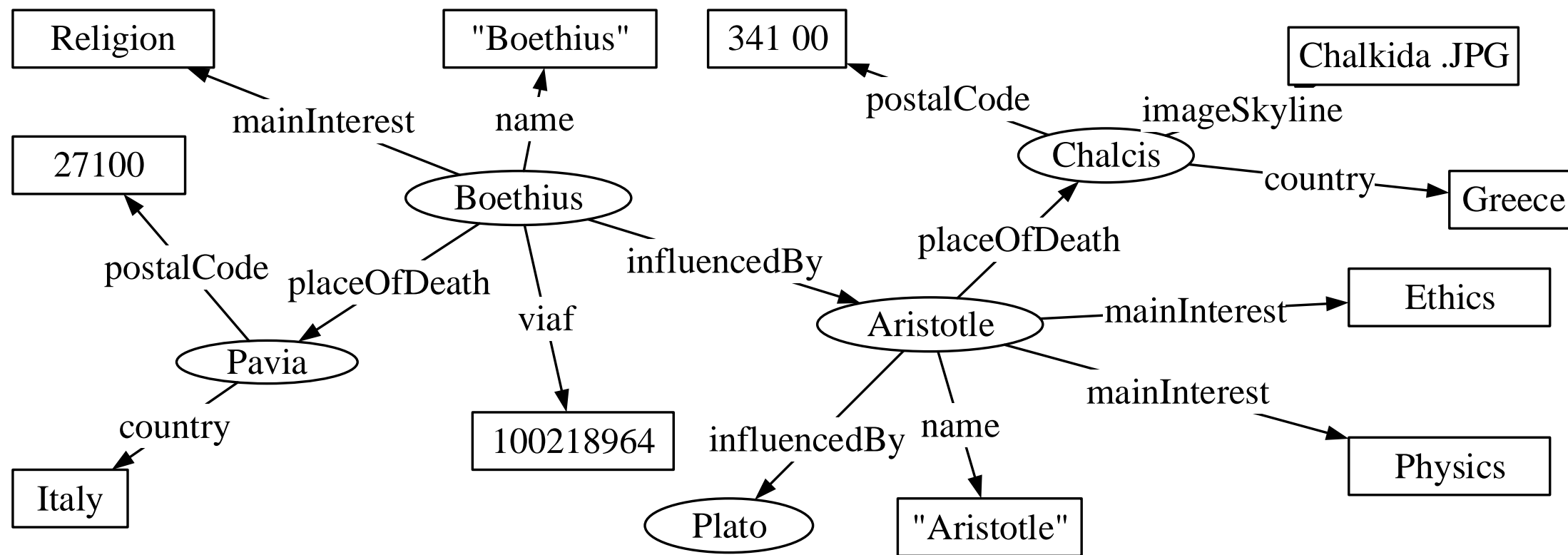
RDF三元组示例

主语	谓词	宾语
Aristotle	influencedBy	Plato
Aristotle	mainInterest	Ethics
Aristotle	name	"Aristotle"
Aristotle	placeOfDeath	Chalcis
Aristotle	mainInterest	Physics
Boethius	influencedBy	Aristotle
Boethius	mainInterest	Religion
Boethius	name	"Boethius"
Boethius	placeOfDeath	Pavia
Boethius	viaf	100218964
Chalcis	imageSkyline	Chalkida .JPG
Chalcis	country	Greece
Chalcis	postalCode	341 00
Pavia	country	Italy
Pavia	postalCode	27100

RDF图模型

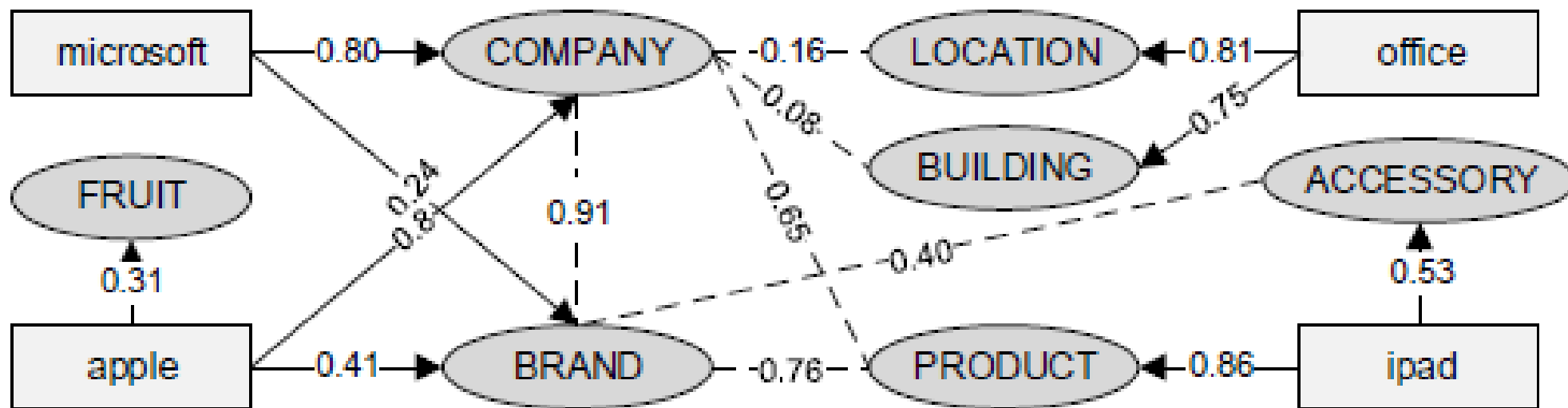
- RDF知识图谱数据可以天然的被视为一个图
- 在这个图中，每个资源或者数据集中出现过的字符串可以被视为图上的点，每个三元组可以视为连接主体及客体的有向边，而三元组中的谓词就可以视为有向边上的标签

RDF数据图示例



Probase^[3]

- Probase是微软构建的带概率的概念知识图谱，只有isA关系，所以是带概率的有向无环图



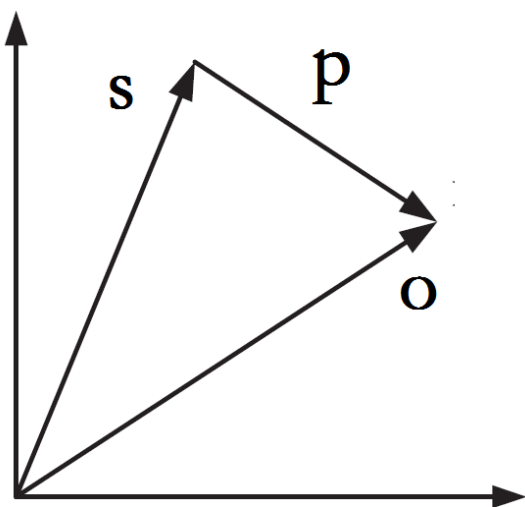
Yashen Wang, Heyan Huang, Chong Feng: Query Expansion Based on a Feedback Concept Model for Microblog Retrieval. WWW 2017: 559-568

知识嵌入 (Knowledge Embedding)

- 给定一个知识图谱，在**保留语义**的情况下将其转化为低维向量
- 基本步骤：
 - 1、将知识图谱中的实体转化为向量，关系转化成向量空间中操作；
 - 2、定义一个估值函数 f ，来度量每个三元组的可能性；
 - 3、找出最优的实体向量来最优化这个代价函数 f

TransE模型^[4]

- 位移假设 (translation assumption): $\mathbf{s} + \mathbf{p} \approx \mathbf{o}$
 - China – Beijing = France – Paris = <capital-of>
 - Beijing + <capital-of> = China
 - Paris + <capital-of> = France



TransE

实体表示: 向量 \mathbf{s} 、 \mathbf{o}

关系表示: 向量 \mathbf{p}

位移操作: $\mathbf{s} + \mathbf{p} \approx \mathbf{o}$

三元组打分: $f_p(s, o) = -\|\mathbf{s} + \mathbf{p} - \mathbf{o}\|_1$

TransE模型

- 首先，将实体和关系都转化成向量
- 然后，给定三元组 $\langle s, p, o \rangle$ ，定义估值函数

$$f_p(s, o) = -\|\mathbf{s} + \mathbf{p} - \mathbf{o}\|_{1/2}$$

- 最后，基于知识图谱数据中已有三元组，最优化上述这个函数

TransE模型

- 优化问题构造

$$\min_{\{\mathbf{e}_i\}, \{\mathbf{r}_k\}} \sum_{t^+ \in \mathcal{O}} \sum_{t^- \in \mathcal{N}_{t^+}} [\delta + f(e_i, r_k, e_j) - f(e'_i, r_k, e'_j)]_+$$

- 观测三元组（正例）得分 $f(e_i, r_k, e_j)$
- 相应未观测三元组（负例）得分 $f(e'_i, r_k, e'_j)$
- 排序损失：若正负例得分差距大于给定阈值 δ ，损失为零；否则损失大于零
- 排序损失最小化：正负例得分差距尽可能大

TransE模型扩展

- 动机：弥补TransE在自反/多对一/一对多型关系上的不足

- 自反型关系： $(e_i, r_k, e_j) \in \mathcal{O}, (e_j, r_k, e_i) \in \mathcal{O}$

$$\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j = \mathbf{0}, \mathbf{e}_j + \mathbf{r}_k - \mathbf{e}_i = \mathbf{0} \Rightarrow \mathbf{r}_k = \mathbf{0}, \mathbf{e}_i = \mathbf{e}_j$$

- 多对一型关系： $\forall i \in \{1, \dots, n\}, (e_i, r_k, e_j) \in \mathcal{O}$

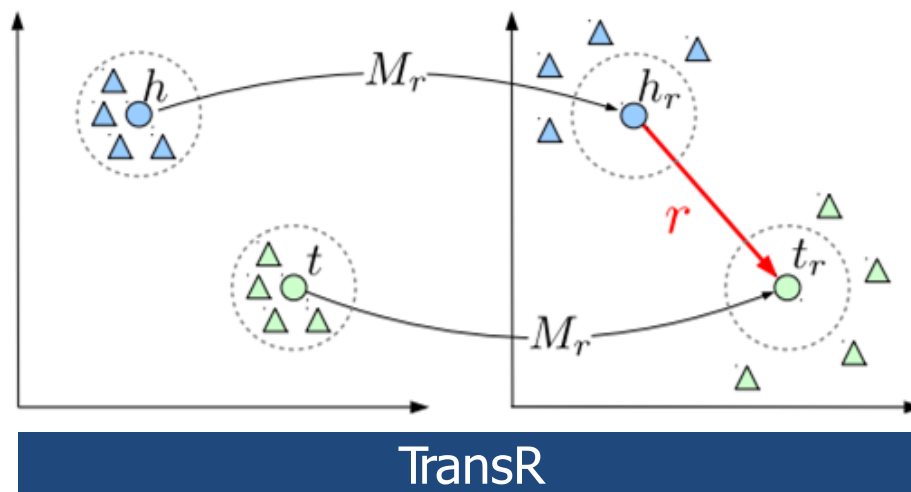
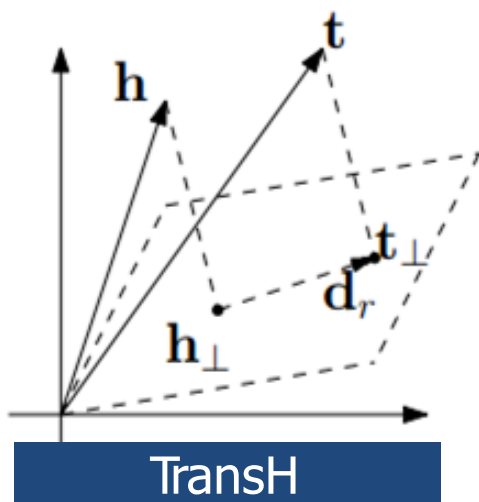
$$\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j = \mathbf{0}, \forall i \in \{1, \dots, n\} \Rightarrow \mathbf{e}_1 = \mathbf{e}_2 = \dots = \mathbf{e}_n$$

- 一对多型关系： $\forall j \in \{1, \dots, m\}, (e_i, r_k, e_j) \in \mathcal{O}$

$$\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j = \mathbf{0}, \forall j \in \{1, \dots, m\} \Rightarrow \mathbf{e}_1 = \mathbf{e}_2 = \dots = \mathbf{e}_m$$

TansH和TransR

- 解决方案：同一实体在不同关系下有不同的表示
 - TransH: 关系专属超平面 (relation-specific hyperplanes)
 - TransR: 关系专属投影矩阵 (relation-specific projection matrices)



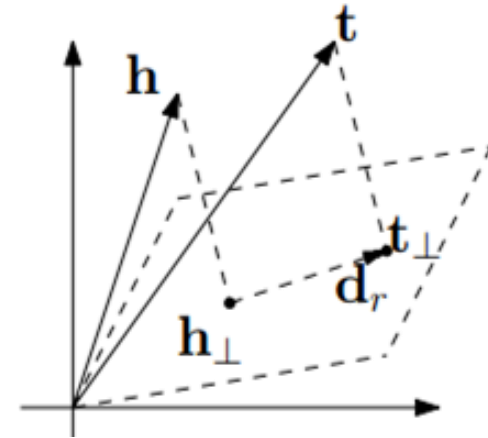
TransH^[18]

- 实体和关系的向量空间表示

- 实体：向量 $\mathbf{e} \in \mathbb{R}^d$
- 关系：位移向量 $\mathbf{r} \in \mathbb{R}^d$ ，超平面法向量 $\mathbf{w} \in \mathbb{R}^d$

- 打分函数定义

- 头实体投影： $\mathbf{e}_{\perp}^i = \mathbf{e}_i - \mathbf{w}_k^T \mathbf{e}_i \mathbf{w}_k$
- 尾实体投影： $\mathbf{e}_{\perp} = \mathbf{e}_j - \mathbf{w}_k^T \mathbf{e}_j \mathbf{w}_k$
- 位移操作： $\mathbf{e}_{\perp} + \mathbf{r}_k \approx \mathbf{e}_{\perp}$
- 距离模型： $f(e_i, r_k, e_j) = \|(\mathbf{e}_i - \mathbf{w}_k^T \mathbf{e}_i \mathbf{w}_k) + \mathbf{r}_k - (\mathbf{e}_j - \mathbf{w}_k^T \mathbf{e}_j \mathbf{w}_k)\|_1$



TransH

- 优化问题构造

$$\min_{\{\mathbf{e}_i\}, \{\mathbf{r}_k\}} \sum_{t^+ \in \mathcal{O}} \sum_{t^- \in \mathcal{N}_{t^+}} [\delta + f(e_i, r_k, e_j) - f(e'_i, r_k, e'_j)]_+$$

- 观测三元组（正例）得分 $f(e_i, r_k, e_j)$
- 相应未观测三元组（负例）得分 $f(e'_i, r_k, e'_j)$
- 排序损失：若正负例得分差距大于给定阈值 δ ，损失为零；否则损失大于零
- 排序损失最小化：正负例得分差距尽可能大

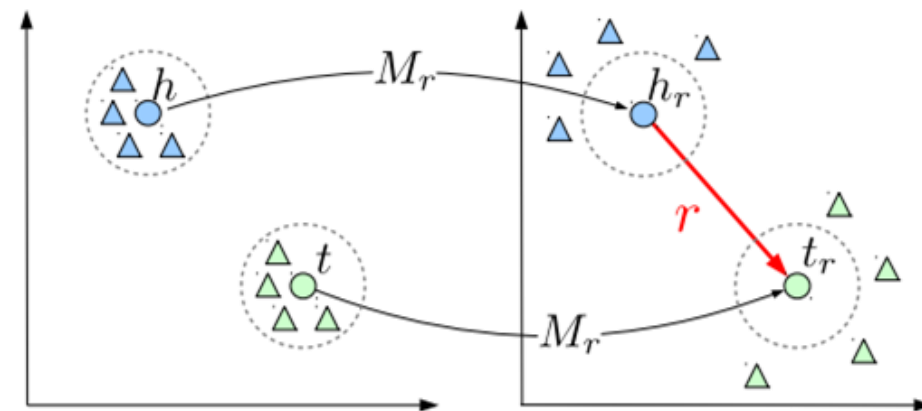
TransR^[19]

- 实体和关系的向量空间表示

- 实体：向量 $\mathbf{e} \in \mathbb{R}^d$
- 关系：位移向量 $\mathbf{r} \in \mathbb{R}^d$ ，投影矩阵 $\mathbf{M} \in \mathbb{R}^{d \times d}$

- 打分函数定义

- 头实体投影： $\mathbf{e}_\perp = \mathbf{M}_k \mathbf{e}_i$
- 尾实体投影： $\mathbf{e}_\perp = \mathbf{M}_k \mathbf{e}_j$
- 位移操作： $\mathbf{e}_\perp + \mathbf{r}_k \approx \mathbf{e}_\perp$
- 距离模型： $f(e_i, r_k, e_j) = \|\mathbf{M}_k \mathbf{e}_i + \mathbf{r}_k - \mathbf{M}_k \mathbf{e}_j\|_1$



TransR

- 优化问题构造

$$\min_{\{\mathbf{e}_i\}, \{\mathbf{r}_k\}} \sum_{t^+ \in \mathcal{O}} \sum_{t^- \in \mathcal{N}_{t^+}} [\delta + f(e_i, r_k, e_j) - f(e'_i, r_k, e'_j)]_+$$

- 观测三元组（正例）得分 $f(e_i, r_k, e_j)$
- 相应未观测三元组（负例）得分 $f(e'_i, r_k, e'_j)$
- 排序损失：若正负例得分差距大于给定阈值 δ ，损失为零；否则损失大于零
- 排序损失最小化：正负例得分差距尽可能大

统一框架

- 相同的优化方式

$$\min_{\{\mathbf{e}_i\}, \{\mathbf{r}_k\}} \sum_{t^+ \in \mathcal{O}} \sum_{t^- \in \mathcal{N}_{t^+}} [\delta + f(e_i, r_k, e_j) - f(e'_i, r_k, e'_j)]_+$$

- 不同的实体/关系表示方式和打分函数

Method	Entity/Relation embeddings	Scoring function
TransE (Bordes et al., 2013)	$\mathbf{e}, \mathbf{r} \in \mathbb{R}^d$	$f_{ij}^{(k)} = \ \mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j\ _{\ell_1/\ell_2}$
SME (lin) (Bordes et al., 2014)	$\mathbf{e}, \mathbf{r} \in \mathbb{R}^d$	$f_{ij}^{(k)} = (\mathbf{W}_{u1}\mathbf{r}_k + \mathbf{W}_{u2}\mathbf{e}_i + \mathbf{b}_u)^T (\mathbf{W}_{v1}\mathbf{r}_k + \mathbf{W}_{v2}\mathbf{e}_j + \mathbf{b}_v)$
SME (bilin) (Bordes et al., 2014)	$\mathbf{e}, \mathbf{r} \in \mathbb{R}^d$	$f_{ij}^{(k)} = ((\mathbf{W}_u \bar{\times}_3 \mathbf{r}_k) \mathbf{e}_i + \mathbf{b}_u)^T ((\mathbf{W}_v \bar{\times}_3 \mathbf{r}_k) \mathbf{e}_j + \mathbf{b}_v)$
SE (Bordes et al., 2011)	$\mathbf{e} \in \mathbb{R}^d, \mathbf{R}^u, \mathbf{R}^v \in \mathbb{R}^{d \times d}$	$f_{ij}^{(k)} = \ \mathbf{R}_k^u \mathbf{e}_i - \mathbf{R}_k^v \mathbf{e}_j\ _{\ell_1}$

知识图谱物理存储

基于关系模型的存储方式

- 现有方法：利用关系数据库技术^[5-10]定义一张三列的表

主语	谓词	宾语
Aristotle	influencedBy	Plato
Aristotle	mainInterest	Ethics
Aristotle	name	"Aristotle"
Aristotle	placeOfDeath	Chalcis
Aristotle	wikiPageUsesTemplate	Template:Planetmath
Boethius	influencedBy	Aristotle
Boethius	mainInterest	Religion
Boethius	name	"Boethius"
Boethius	placeOfDeath	Pavia
Boethius	viaf	100218964
Chalcis	imageSkyline	Chalkida .JPG
Chalcis	country	Greece
Chalcis	postalCode	341 00
Pavia	country	Italy
Pavia	postalCode	27100

Virtuoso^[33]

- 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

主语	谓词	宾语
Aristotle	influencedBy	Plato
Aristotle	mainInterest	Ethics
Aristotle	name	'Aristotle'
Aristotle	placeOfDeath	Chalcis
Aristotle	wikiPageUsesTemplate	Template:Planetmath
Boethius	influencedBy	Aristotle
Boethius	mainInterest	Religion
Boethius	name	"Boethius"
Boethius	placeOfDeath	Pavia
Boethius	viaf	100218964
Chalcis	imageSkyline	Chalkida.JPG
Chalcis	country	Greece
Chalcis	postalCode	341 00
Pavia	country	Italy
Pavia	postalCode	27100

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

三种典型基于关系数据库的优化策略

- 属性表方法 Jena^[5]、Sesame^[32]、Oracle^[6]
- 垂直划分方法 SW-Store^[11]
- 全索引方法 Hexastore^[12]、RDF-3x^[13, 14, 15]

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”

RDF三元组示例

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	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	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”

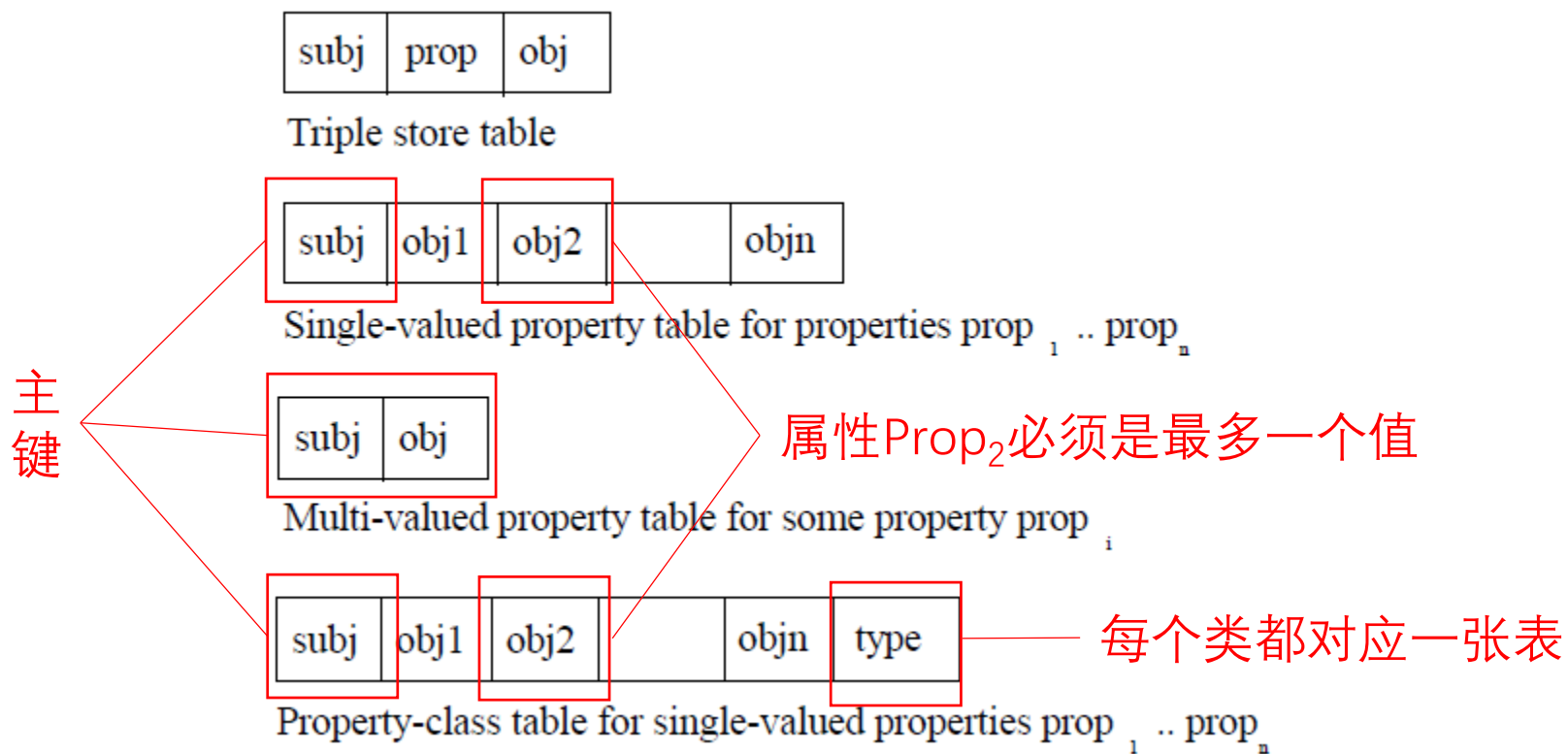
主体	属性	客体
ID2	language	“French”
ID3	language	“English”
ID4	type	DVDType
ID4	title	“DEF”

分类属性表

Jena^[5]

- 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



垂直划分方法

- 对RDF知识图谱数据按照谓词（或属性）分割成若干表的方法

Subj.	Prop.	Obj.
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"



Type	
ID1	BookType
ID2	CDType
ID3	BookType
ID4	DVDType
ID5	CDType
ID6	BookType

Author	
ID1	"Fox, Joe"

Title	
ID1	"XYZ"
ID2	"ABC"
ID3	"MNO"
ID4	"DEF"
ID5	"GHI"

Artist	
ID2	"Orr, Tim"

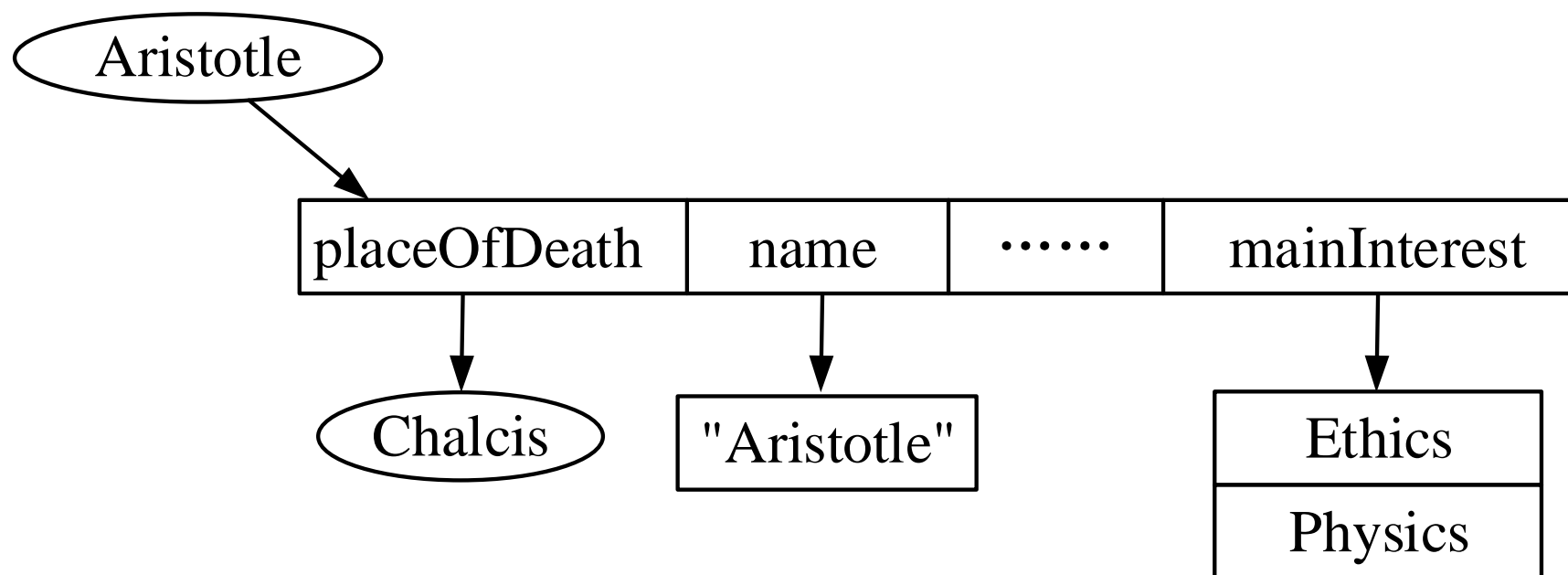
Copyright	
ID1	"2001"
ID2	"1985"
ID5	"1995"
ID6	"2004"

Language	
ID2	"French"
ID3	"English"

Daniel J. Abadi, Adam Marcus, Samuel Madden, Kate Hollenbach:
SW-Store: a vertically partitioned DBMS for Semantic Web data
management. VLDB J. 18(2): 385-406 (2009)

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

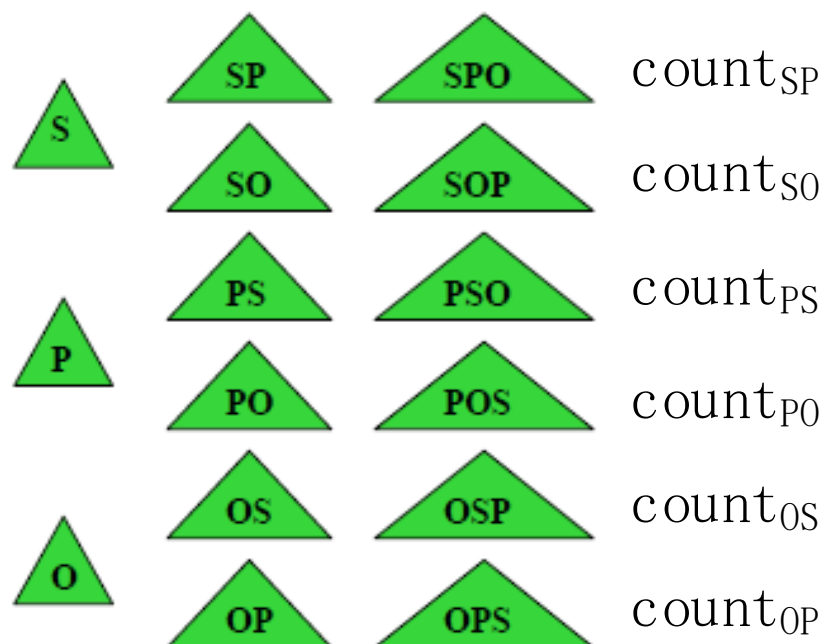


RDF-3X^[14]

- 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/>

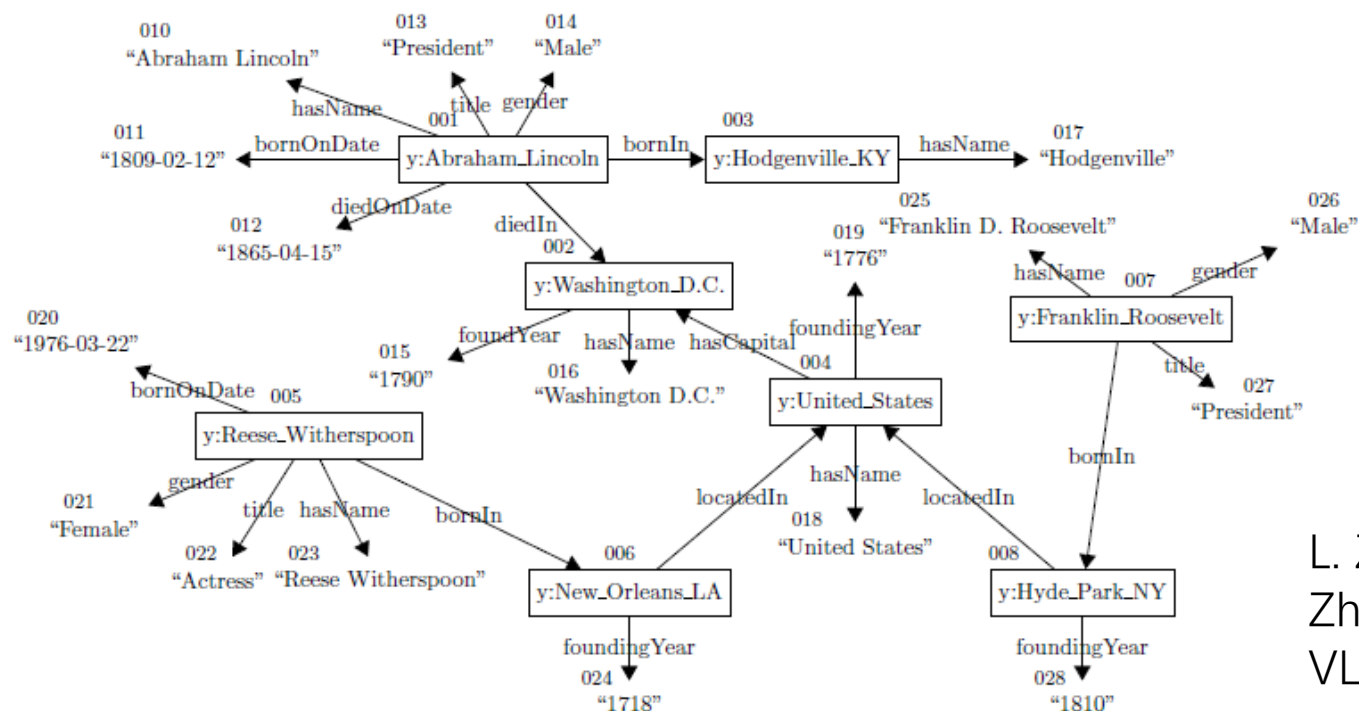
RDF-3X索引

- 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



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



<i>uLabel</i>	<i>adjList</i>
y:Abraham_Lincoln	(hasName, "Abraham Lincoln"), (bornOnDate, "1809-02-12"), (diedOnDate, "1865-04-15"), (diedIn, y:Washington_DC), (gender, "Male"), (title, "President"), (bornIn, y:Hodgenville_KY)
y:Washington_DC	(hasName, "Washington D.C."), (foundingYear, "1790")
...	...

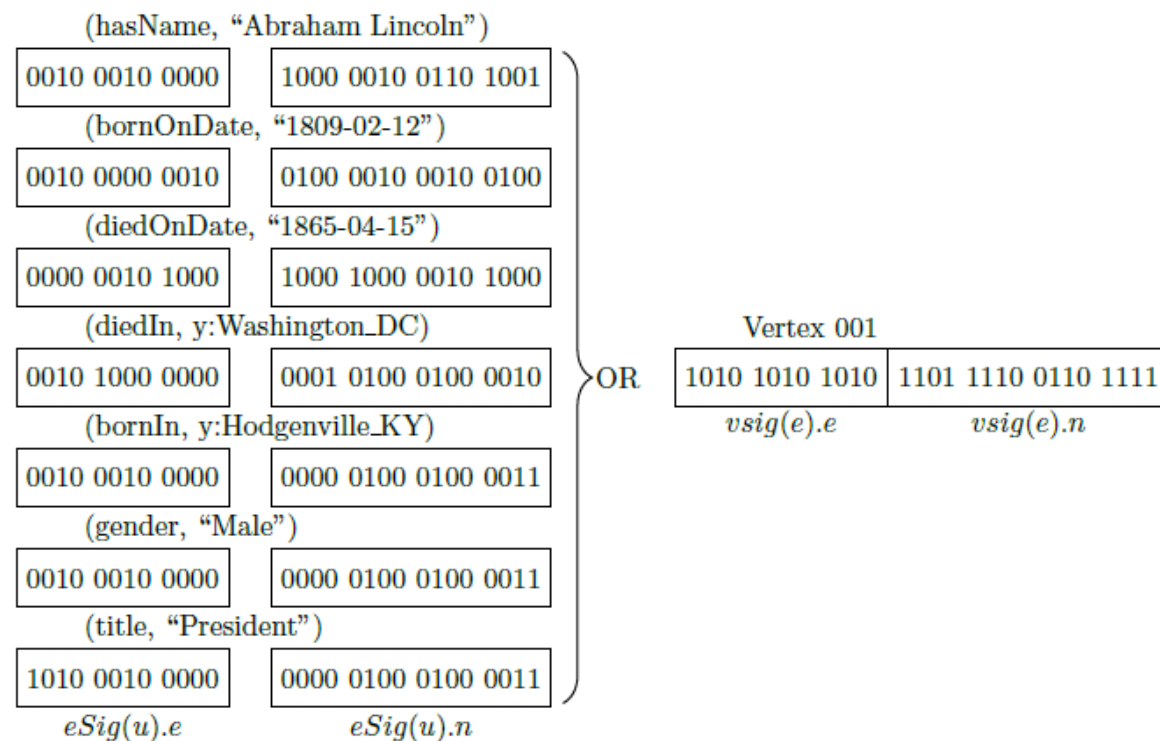
L. Zou, M. T. Özsu, L. Chen, X. Shen, R. Huang and D. Zhao, gStore: A Graph-based SPARQL Query Engine, VLDB J. 23 (4), 565–590 (2014).

gStore^[16,17]

- RDF-3X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/>

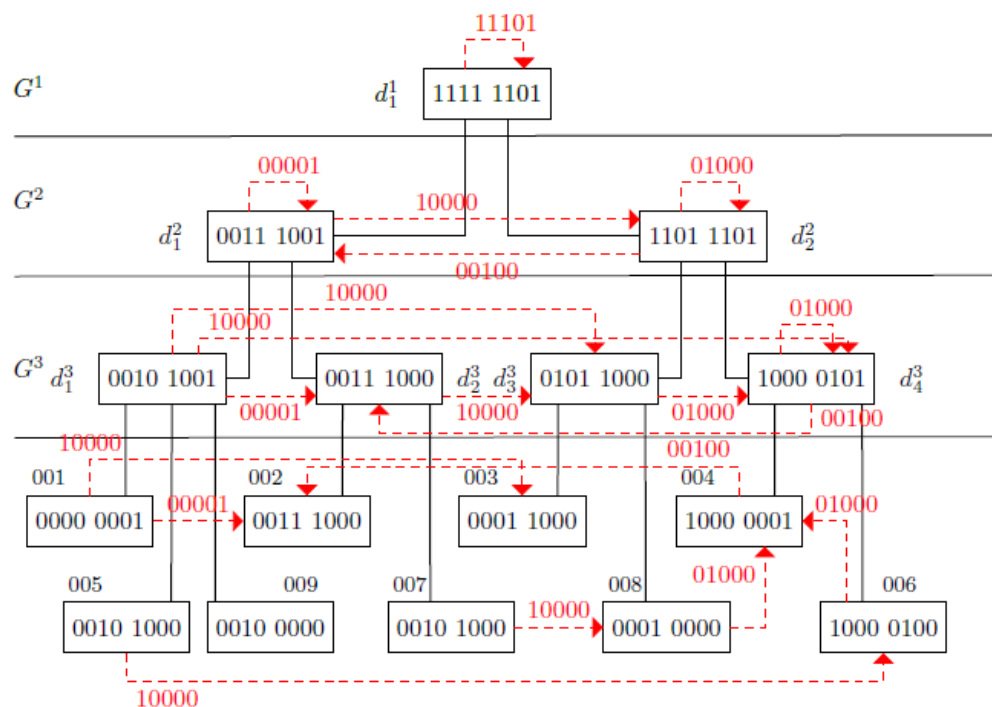
gStore^[16,17]

- 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^[16,17]

- 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



分布式知识图谱存储*

- 基于云平台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
- 基于数据划分的分布式知识图谱管理方法
 - 基于粗化（coarsening）的知识图谱划分方法
 - 基于局部模式的知识图谱划分方法

基于云平台的分布式知识图谱管理方法

- 这类方法都是利用已有云平台存储系统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
- 被最多地利用的云平台系统就是Hadoop; 其他的云平台系统如Trinity、Spark等也有相关应用

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^[20]
 - 基于图的存储，如EAGRE^[21]

SHARD^[20]

- 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

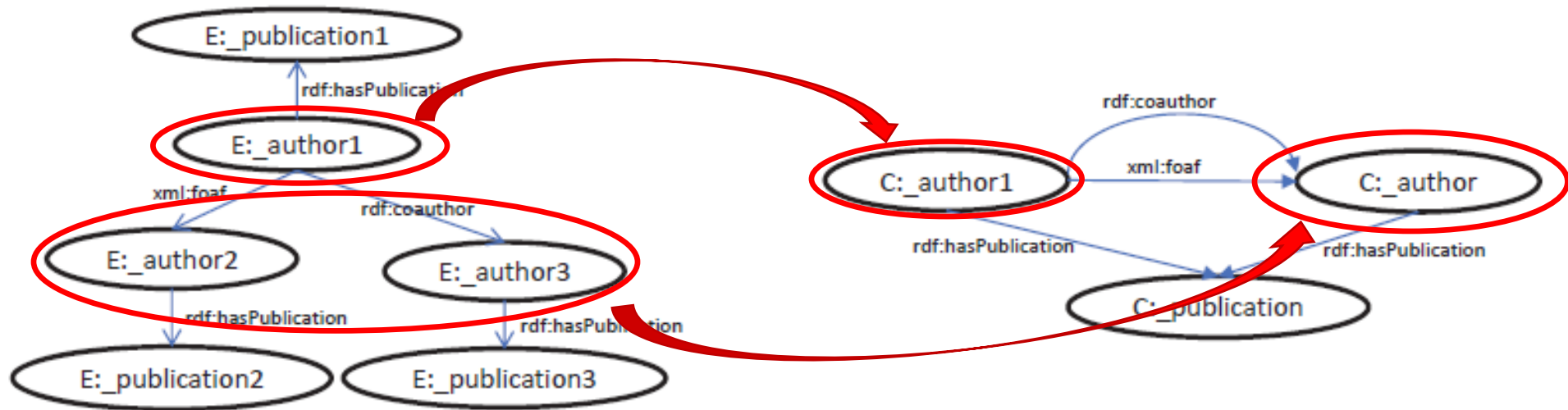
Aristotle influencedBy Plato mainInterest Ethics name "Aristotle" placeOfDeath Chalcis



HDFS

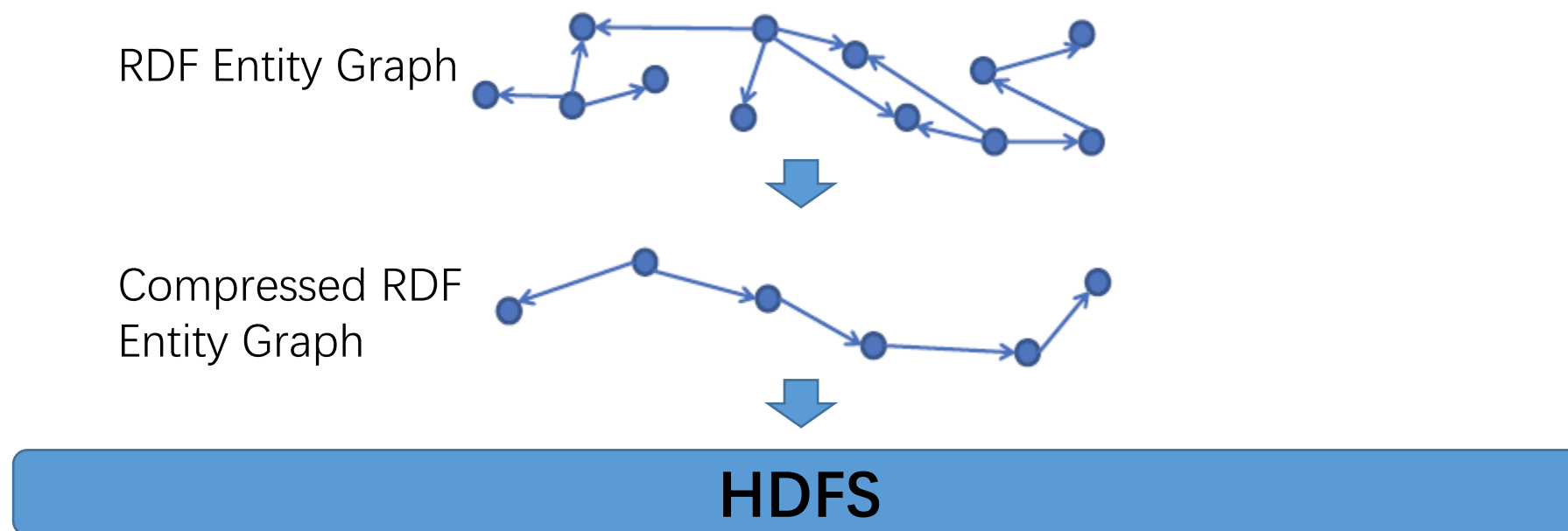
EAGRE^[21]

- 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

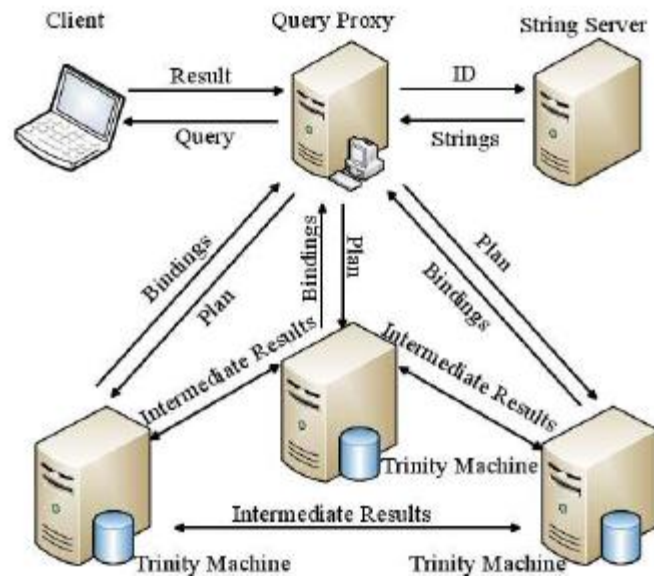


基于其他云平台的分布式知识图谱管理方法

- 基于HBase 的H2RDF^[22, 23]
- 基于Trinity 的Trinity.RDF^[24]
- 基于Spark 的S2RDF^[25]

Trinity.RDF^[24]

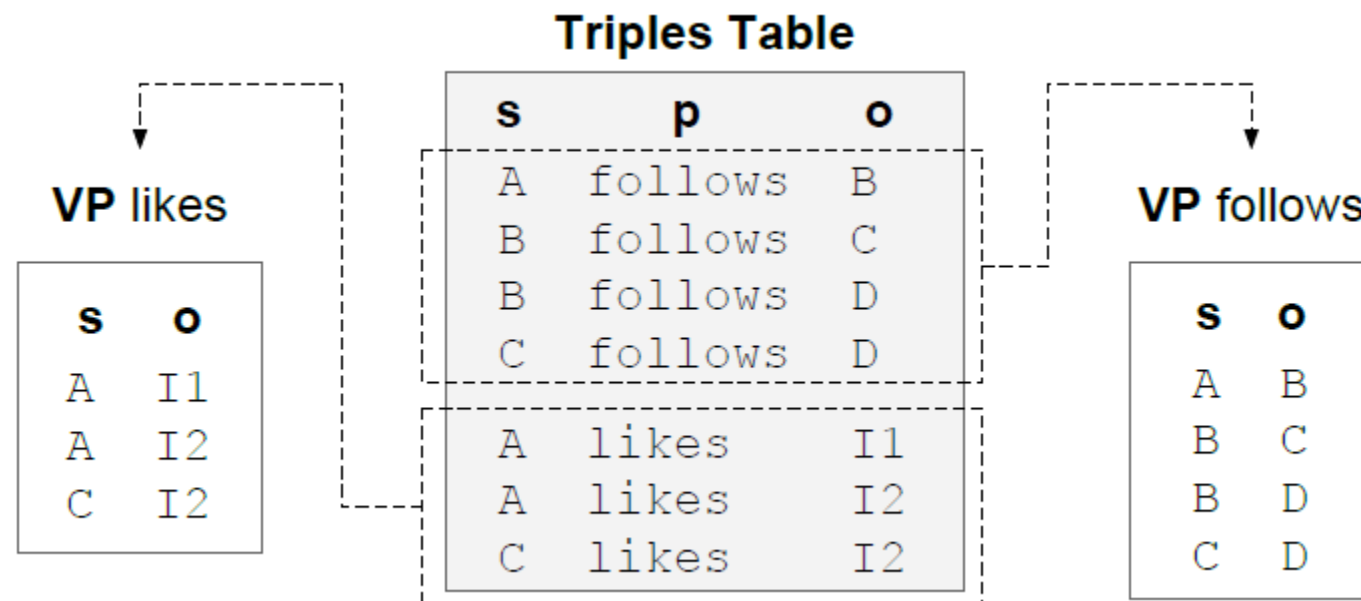
- Trinity是微软开发的一个分布式内存的图数据管理系统
- 知识图谱可以直接以图的方式存在Trinity上



K. Zeng, J. Yang, H. Wang, B. Shao and Z. Wang, A Distributed Graph Engine for Web Scale RDF Data, PVLDB 6 (4), 265–276 (2013).

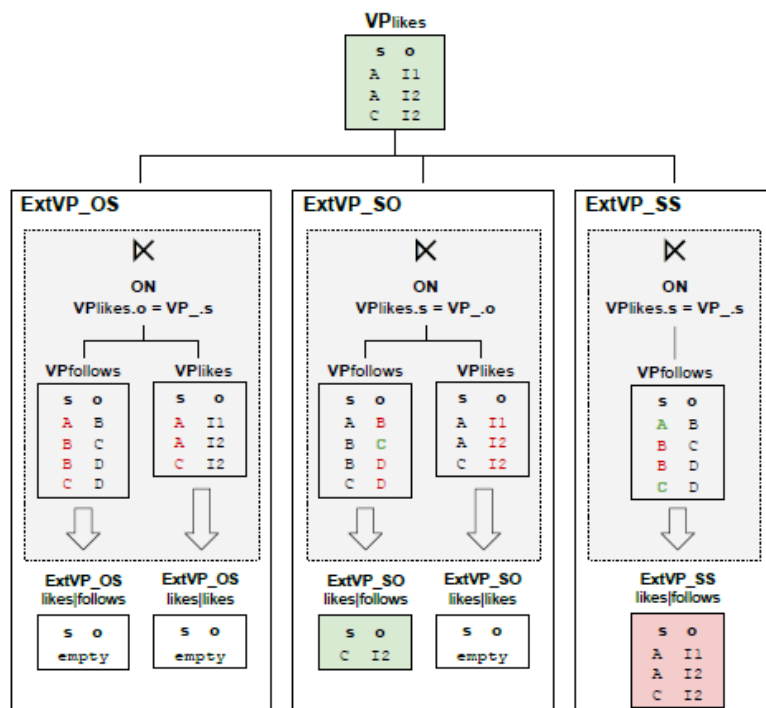
S2RDF^[25]

- 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



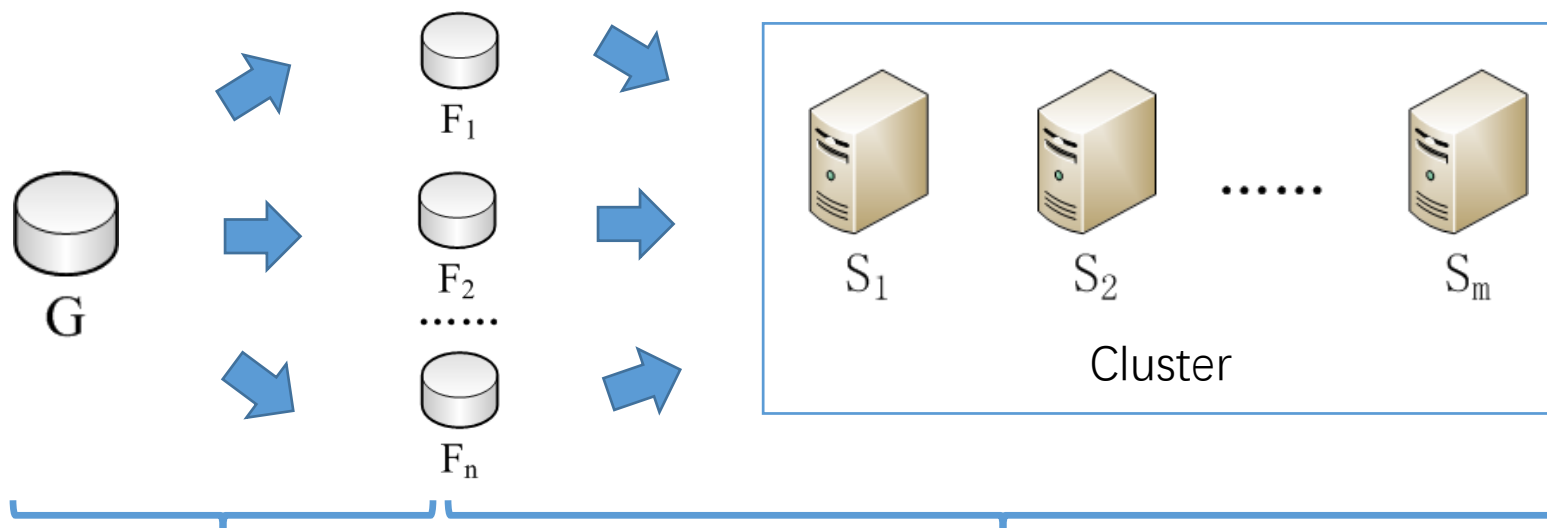
S2RDF

- 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



基于数据划分的分布式知识图谱管理方法

- 第一步，系统要将数据按照一定的算法划分成若干分片；第二步，系统将划分出来的分片分配到不同机器上去
- 这些方法相互之间主要的不同在于数据划分方式不同



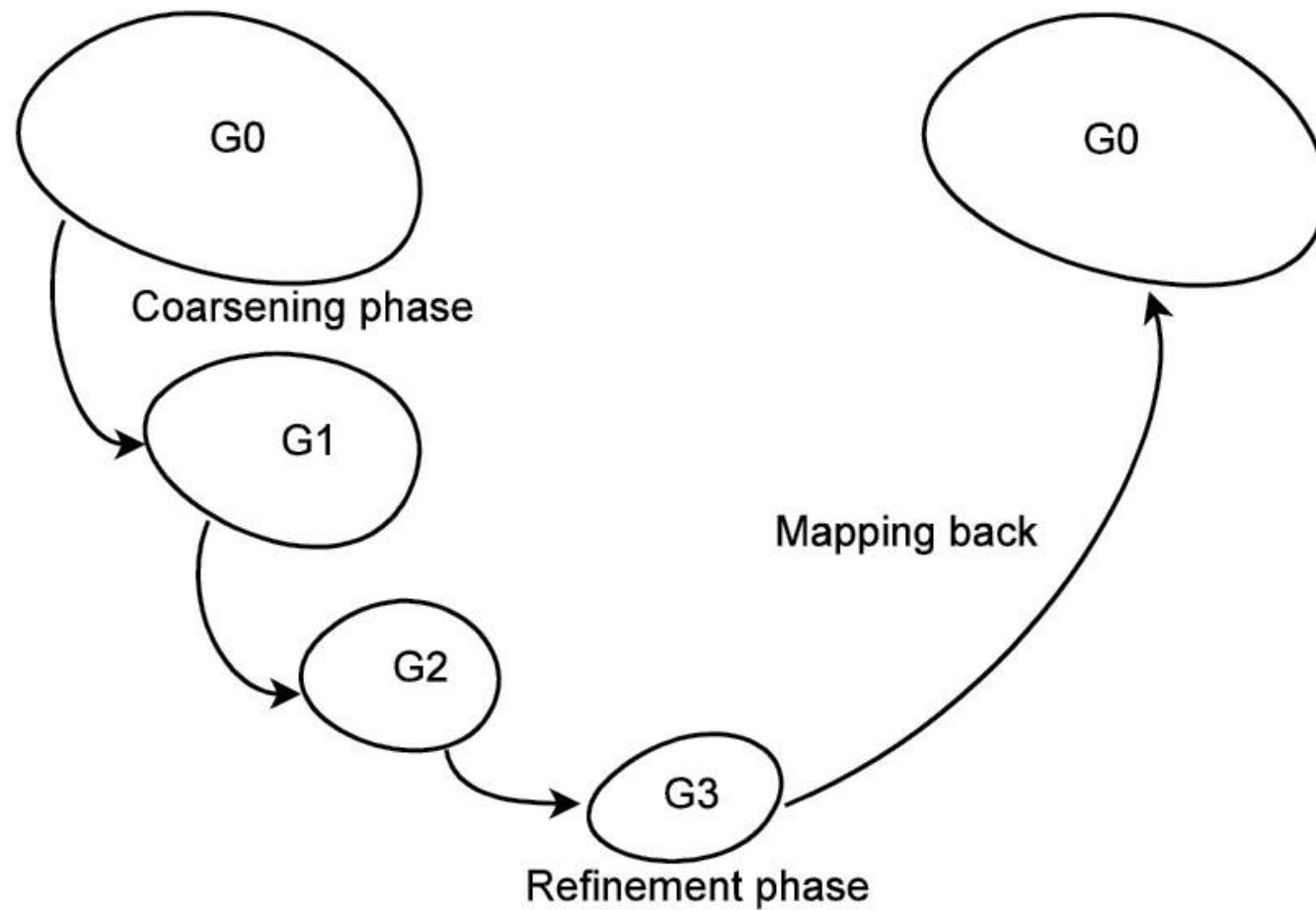
方法分类

- 基于粗化（coarsening）的划分方法，如Huang et al.^[26]
- 基于局部模式的划分方法，如SHAPE^[27]

基于粗化的划分方法

- 这类方法主要思想在于大的知识图谱中一些点迭代地合并起来形成能被处理的小图。然后对这个小图调用现有方法来进行划分，最后将对小图的划分结果回溯成大图的分
- 这个挑选点进行合并的过程称之为**粗化 (coarsening)**
- 现有方法主要区别在于粗化方式不一样

基于粗化的划分



Huang et al.^[26]

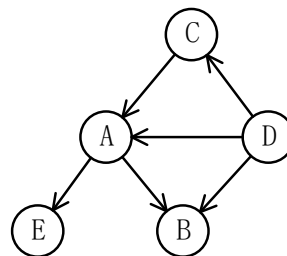
- 这个系统主要思想是将现有知识图谱数据图根据其结构信息进行水平划分
- 它对知识图谱的划分使用现有成熟工具METIS来实现。划分出来每个RDF 数据块对应一个数据分片
- 在此过程中，为了提高数据的局部性，每个块除了保存自身所有的点之外，还保存了它们的 n 步邻居以内所有点和边的副本

基于局部模式的划分方法

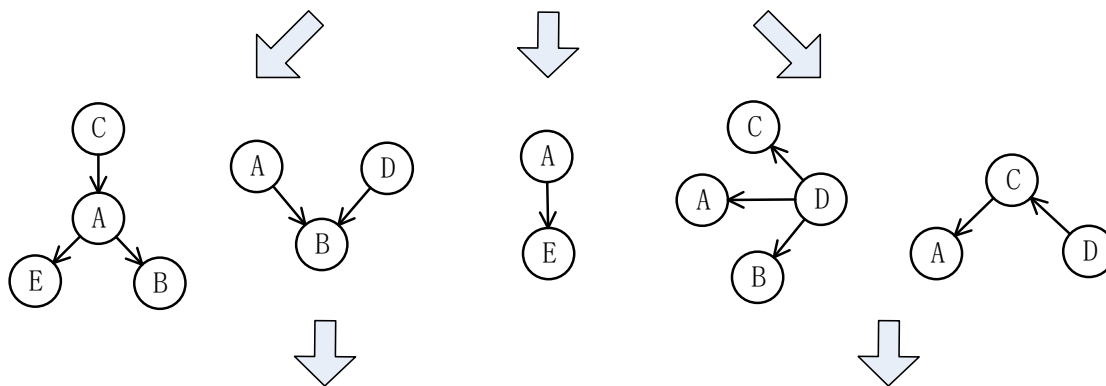
- 这类方法首先定义出若干模式，然后找出图上满足这些模式的子图，最后将这些子图按照模式划分成不同的分片
- 这类方法的关键在于如何定义模式

基于局部模式的划分方法

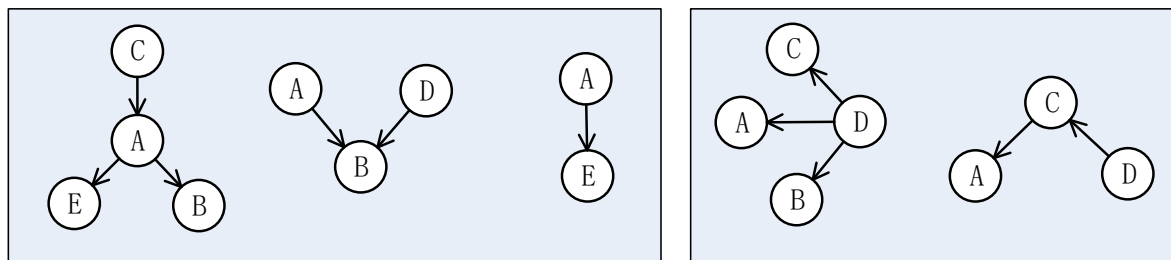
Original Graph



Patterns



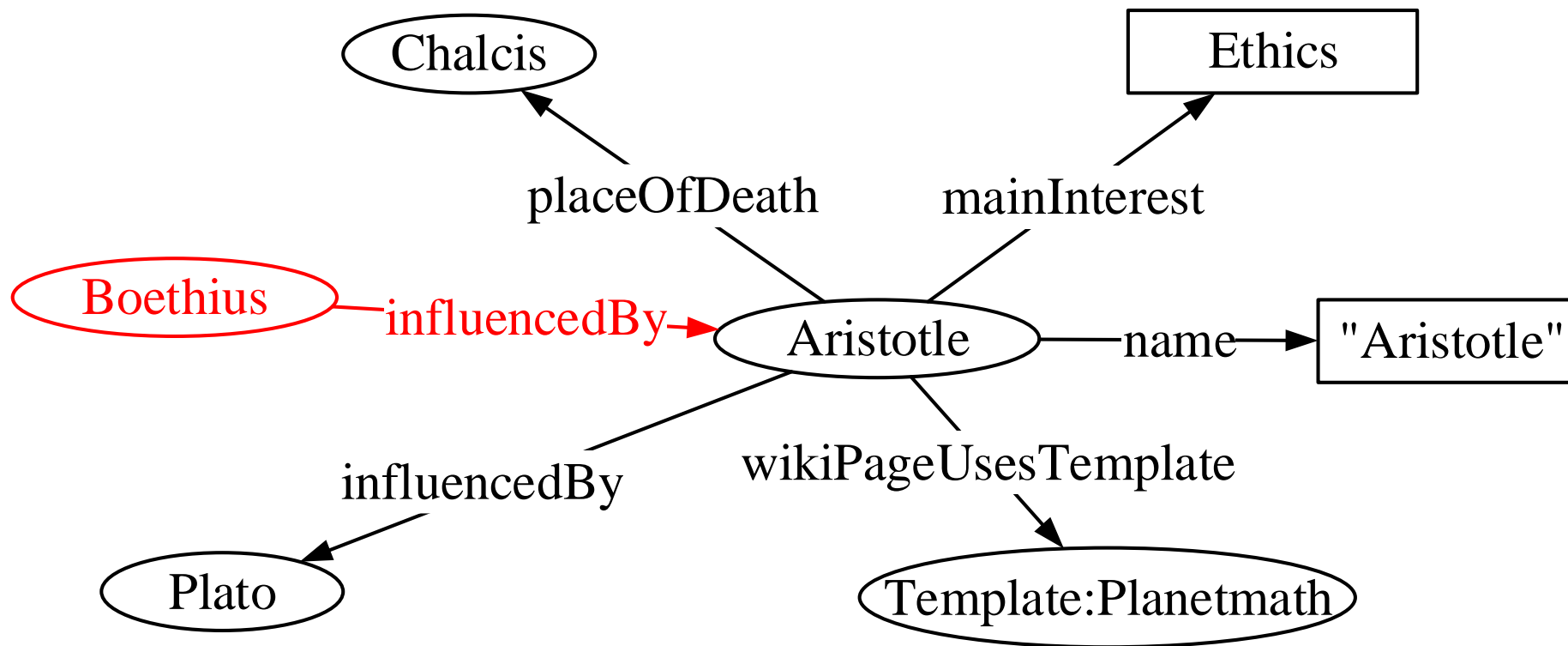
Partitions



SHAPE^[27]

- SHAPE 中，在进行知识图谱数据划分的时候，SHAPE 划分的基本单元为一个三元组群 (Triple Group)
- 所谓三元组群，其实就是图上的星状结构
- SHAPE 提出了三种三元组群：只含中心点出边的三元组群、只含中心点入边的三元组群、含中心点出边与入边的三元组群

SHAPE



- 黑色部分是只含中心点出边的三元组群；红色部分是只含中心点入边的三元组群；整个就是含中心点出边与入边的三元组群

《知识图谱: 概念与技术》

谢谢!



参考文献

1. Cf. T. R. Gruber. A translation approach to portable ontologies. Knowledge Acquisition, 5(2):199-220, 1993.
2. Thomas R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing, Revision: August 23, 1993.
3. Wentao Wu, Hongsong Li, Haixun Wang, Kenny Qili Zhu. Probase: a probabilistic taxonomy for text understanding. SIGMOD Conference 2012: 481-492
4. Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. NIPS 2013: 2787-2795
5. K. Wilkinson, Jena Property Table Implementation, in SSWS, Athens, Georgia, USA (2006), pp. 35-46.
6. Chong, E.I., Das, S., Eadon, G., Srinivasan, J.: An efficient SQLbased RDF querying scheme. In: VLDB, pp. 1216-1227 (2005)
7. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: a generic architecture for storing and querying RDF and RDF schema. In: ISWC, pp. 54-68 (2002)
8. Harris, S., Gibbins, N.: 3store: efficient bulk RDF storage. In: Proceedings of PSSS'03, pp. 1-15 (2003)
9. Lu, J., Ma, L., Zhang, L., Brunner, J.-S., Wang, C., Pan, Y., Yu, Y.: SOR: A practical system for ontology storage, reasoning and search. In: Proceedings of VLDB, pp. 1402-1405 (2007)
10. Wilkinson, K.: Jena property table implementation. In: SSWS (2006)


参考文献

11. Daniel J. Abadi, Adam Marcus, Samuel Madden, Kate Hollenbach: SW-Store: a vertically partitioned DBMS for Semantic Web data management. VLDB J. 18(2): 385–406 (2009)
12. C. Weiss, P. Karras and A. Bernstein, Hexastore. Sextuple Indexing for Semantic Web Data Management, PVLDB 1 (1), 1008–1019 (2008).
13. T. Neumann and G. Weikum, RDF-3X: A RISC-style Engine for RDF, PVLDB 1 (1), 647–659 (2008).
14. T. Neumann and G. Weikum, The RDF-3X Engine for Scalable Management of RDF Data, VLDB J. 19 (1), 91–113 (2010).
15. T. Neumann and G. Weikum, x-RDF-3X: Fast Querying, High Update Rates, and Consistency for RDF Databases, PVLDB 3 (1), 256–263 (2010).
16. L. Zou, J. Mo, L. Chen, M. T. Özsu and D. Zhao, gStore: Answering SPARQL Queries via Subgraph Matching, PVLDB 4 (8), 482–493 (2011).
17. L. Zou, M. T. Özsu, L. Chen, X. Shen, R. Huang and D. Zhao, gStore: A Graph-based SPARQL Query Engine, VLDB J. 23 (4), 565–590 (2014).
18. Z. Wang, J. Zhang, J. Feng, and Z. Chen, Knowledge graph embedding by translating on hyperplanes, in Proc. 28th AAAI Conf. Artif. Intell., 2014, pp. 1112–1119.
19. Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in Proc. 29th AAAI Conf. Artif. Intell., 2015, pp. 2181–2187.

参考文献

20. K. Rohloff and R. E. Schantz, High-performance, Massively Scalable Distributed Systems using the MapReduce Software Framework: the SHARD Triple-store, in PSI EtA (2010), p. 4.
21. X. Zhang, L. Chen, Y. Tong and M.Wang, EAGRE: Towards Scalable I/O Efficient SPARQL Query Evaluation on the Cloud, in ICDE (2013), pp. 565–576.
22. N. Papailiou, I. Konstantinou, D. Tsoumakos and N. Koziris, H2RDF: Adaptive Query Processing on RDF Data in the Cloud, in WWW (Companion Volume) (2012), pp. 397–400.
23. N. Papailiou, D. Tsoumakos, I. Konstantinou, P. Karras and N. Koziris, H2RDF+: An Efficient Data Management System for Big RDF Graphs, in SIGMOD (2014), pp. 909–912.
24. K. Zeng, J. Yang, H. Wang, B. Shao and Z. Wang, A Distributed Graph Engine for Web Scale RDF Data, PVLDB 6 (4), 265–276 (2013).
25. A. Schätzle, M. Przyjaciół-Zablocki, S. Skilevic, and G. Lausen. S2RDF: RDF Querying with SPARQL on Spark. PVLDB, 9(10):804–815, 2016
26. J. Huang, D. J. Abadi and K. Ren, Scalable SPARQL Querying of Large RDF Graphs, PVLDB 4 (11), 1123–1134 (2011).
27. K. Lee and L. Liu, Scaling Queries over Big RDF Graphs with Semantic Hash Partitioning, PVLDB 6 (14), 1894–1905 (2013).
28. 《神经网络与深度学习》，邱锡鹏
29. Markov logic networks , Richardson and Domingos, 2006

参考文献



- 30. A Short Introduction to Probabilistic Soft Logic, Kimming et al., 2012
- 31. Probabilistic Soft Logic for Semantic Textual Similarity, Beltağy et al. 2013
- 32. J. Broekstra, A. Kampman and F. van Harmelen, Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema, in ISWC (2002), pp. 54–68.
- 33. Orri Erling, Ivan Mikhailov: Virtuoso: RDF Support in a Native RDBMS. Semantic Web Information Management 2009: 501–519