Next Generation of Knowledge Graph

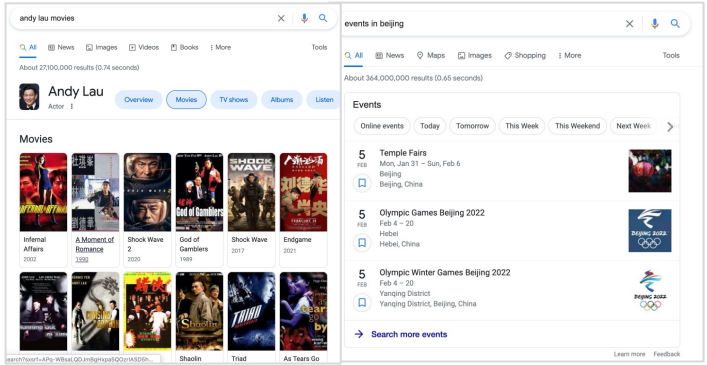
-- Semantic-enhanced Programmable Graph

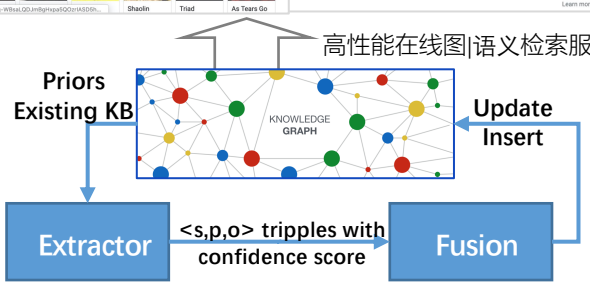
蚂蚁集团 | 梁磊

2023.06.19

目录

从通用知识图谱到领域知识图谱

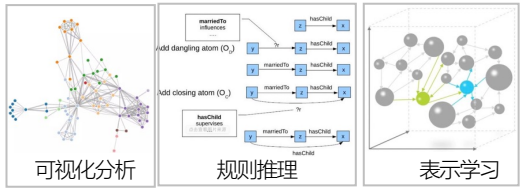




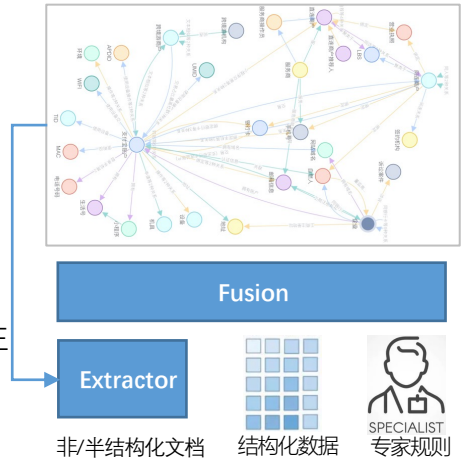
图谱构建以
文档抽取式为主

- 从开放数据中抽取<s,p,o>三元组而构建的知识库，搜/推/QA等应用
- Google KV/百度/MS Satori(企)、Wikipedia/Freebase/YAGO (开)为代表

通用知识图谱(2012-)



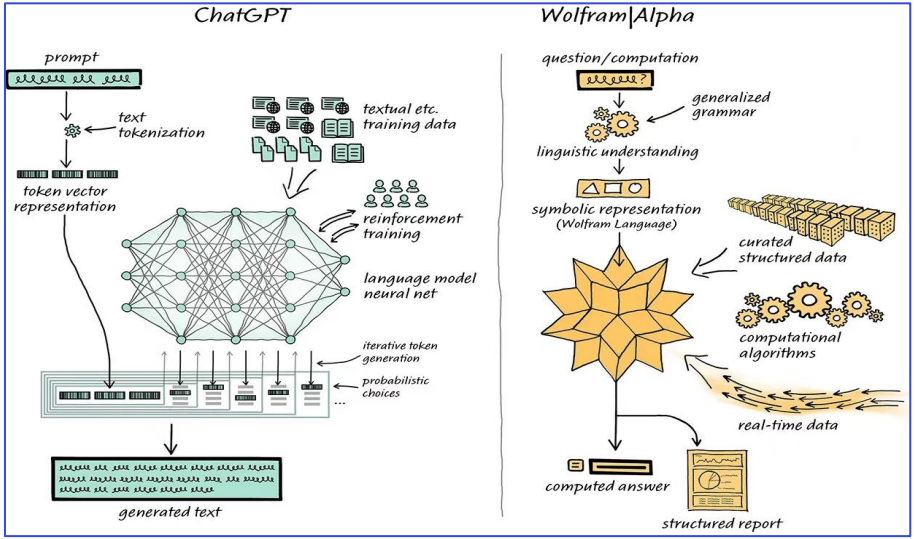
图谱构建以
结构化数据为主

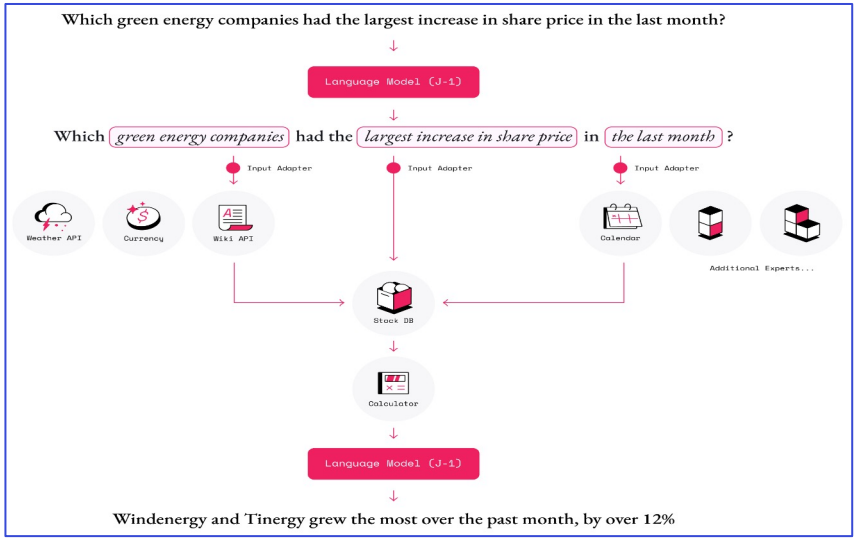


- 知识获取从开放域到以封闭领域知识为主，知识与经验积累并重
- 以发现领域稀薄知识为主要推理任务，如风控、信贷、保险等
- Data fabric: 连接企业内数据孤岛，建立知识驱动的数据底盘
- Neo4j: Graph + Semantics = Knowledge Graph

领域知识图谱(2018-)

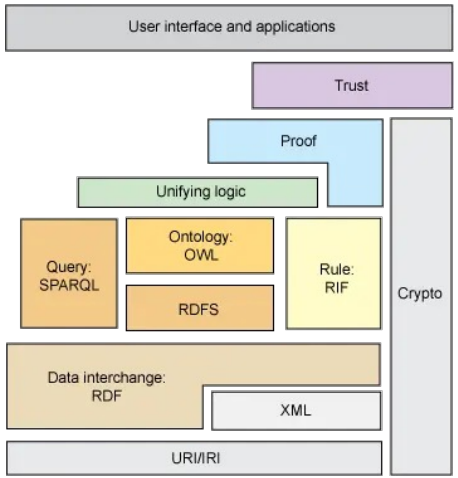
LLM下基于知识图谱的领域知识外挂



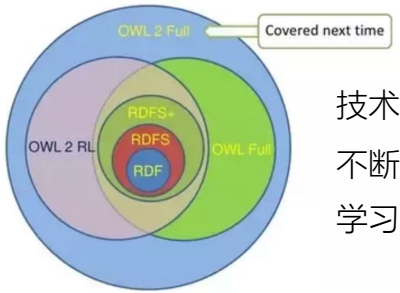


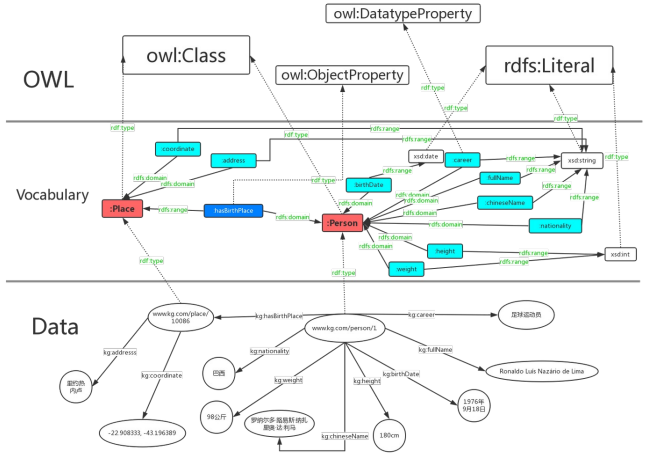
- **知识外挂：** LLM模型压缩，知识外化/卸载，超大LLM模型Transformer的稀疏化，能根据个性化需求提供信息
- **LLM +KG:** 神经+符号，与知识图谱的深度融合，结合符号系统提升领域知识、复杂推理能力
- **资源更新：** 集成多资源提供动态知识，应对实时更新的问题。Toolformer: Language Models Can Teach Themselves to Use Tools。

RDF/OWL - 知识管理的不足(举例)



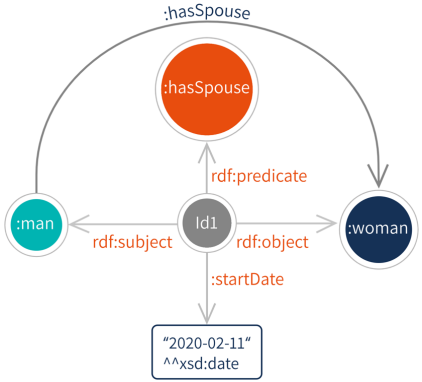
From RDF to OWL 2 Full



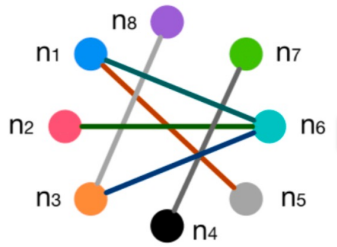


技术栈复杂
不断打补丁
学习成本高

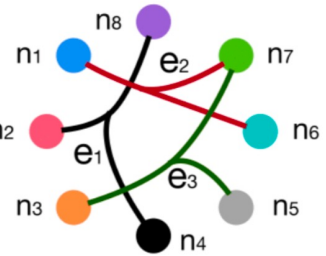
- 语义严谨，语法体系复杂
- 三元组结构展开，存储开销大
- 类型、实例定义耦合，图谱构建成本高



又扩展了RDF-Star



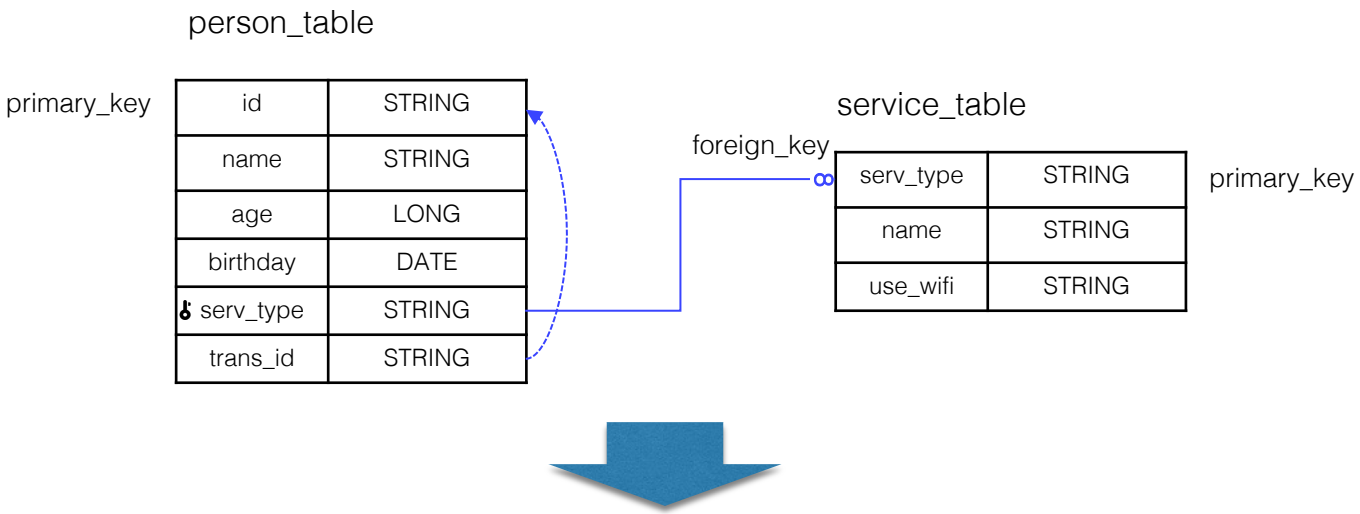
<s,p,o>主要表示二元结构



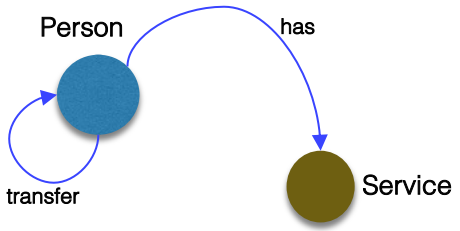
<s,p,o>实例结构难以表示工业应用下复杂多元结构的拓展(如事件)

最关键的，缺少工业化应用中的成功案例

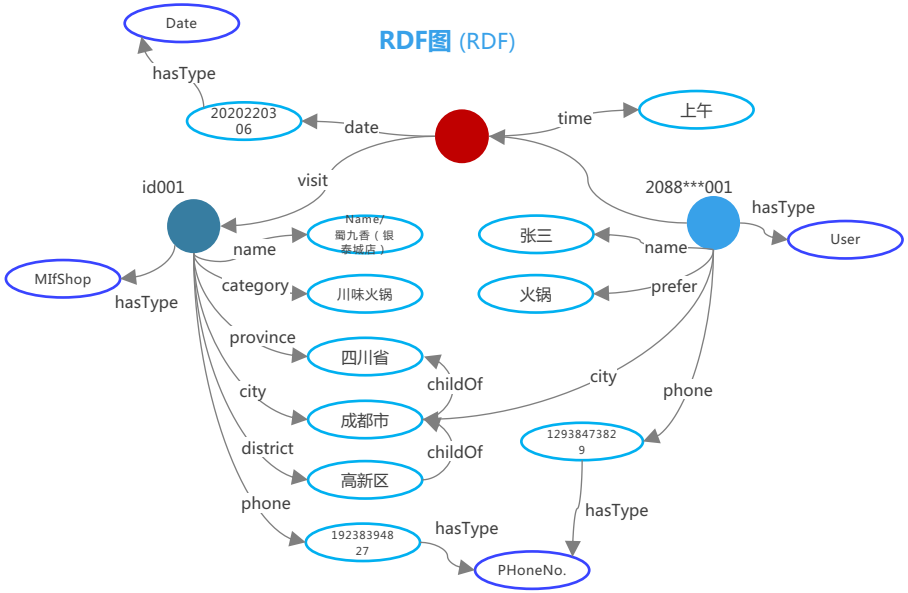
LPG - 外键关系化的数据结构

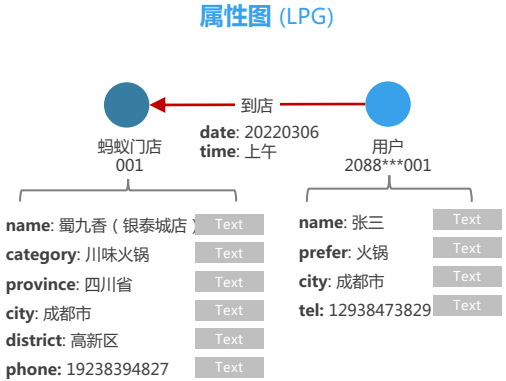



```
CREATE GRAPH TYPE fraudGraphType STRICT {
      ( personType : Person { name STRING , age LONG, birthday DATE})
      ( serviceType : Service { name STRING , use_wifi STRING})
      (: personType)-[ hastype : has]->(: serviceType )
      (: personType)-[ transferType : transfer]->(: personType)
}
```



两种知识建模表示方式: RDF vs LPG



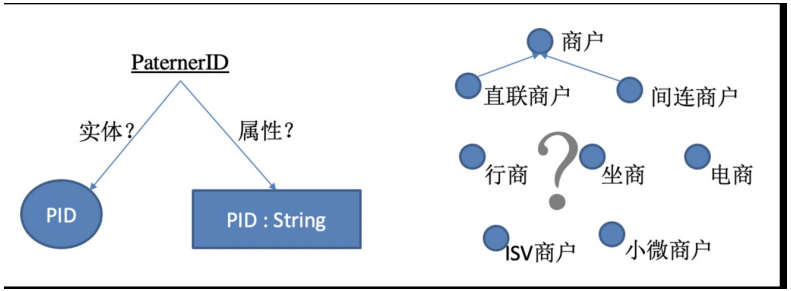


知识表示(工业落地效率与知识标准化之间的矛盾)

- RDF（Resource Description Framework）：用于知识交换，强语义，高门槛
- LPG（Labeled Property Graph）：用于图存储与查询，弱语义，低门槛

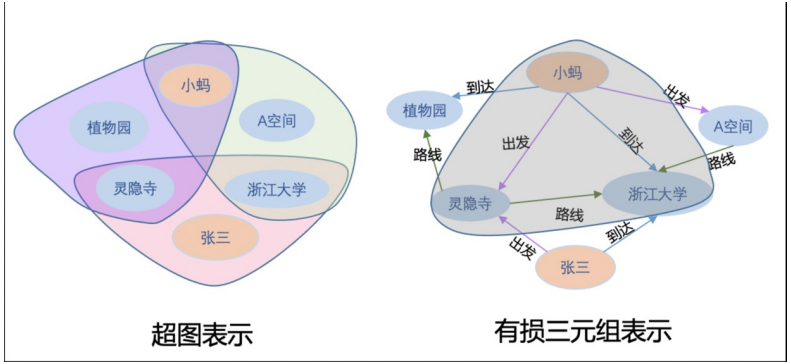
目录

图谱应用业务痛点(结构定义)



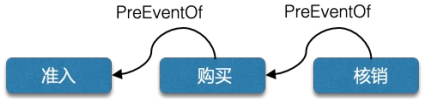
实体类型定义颗粒度不同造成重复构建和潜在不一致

用户	出发地点	出发时间	抵达地点	到达时间
小蚂	灵隐寺	8:00	植物园	10:00
张三	灵隐寺	9:00	浙江大学	10:00
小蚂	A空间	17:00	浙江大学	18:00

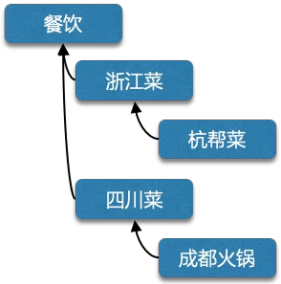


时空多元关联的建模和构建问题

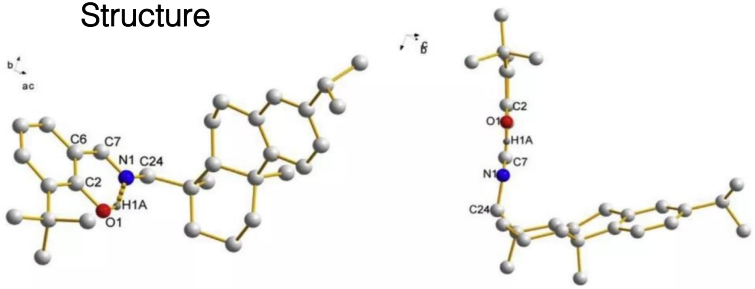
Pipeline



Tree



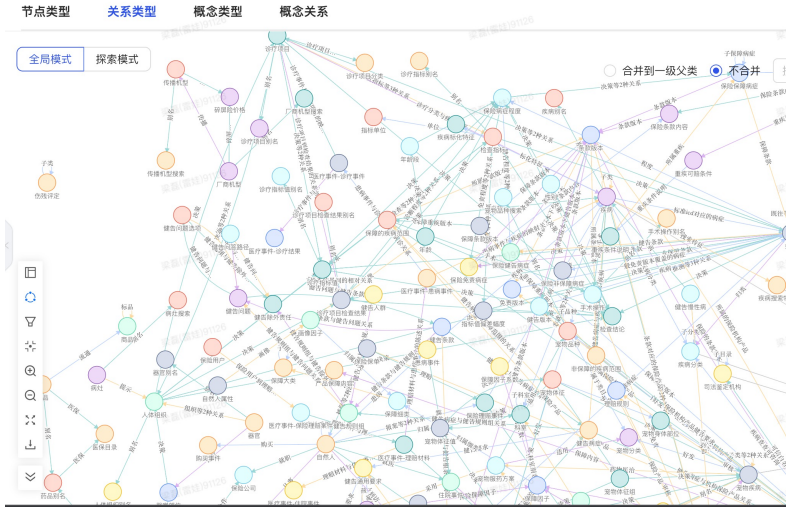
Structure



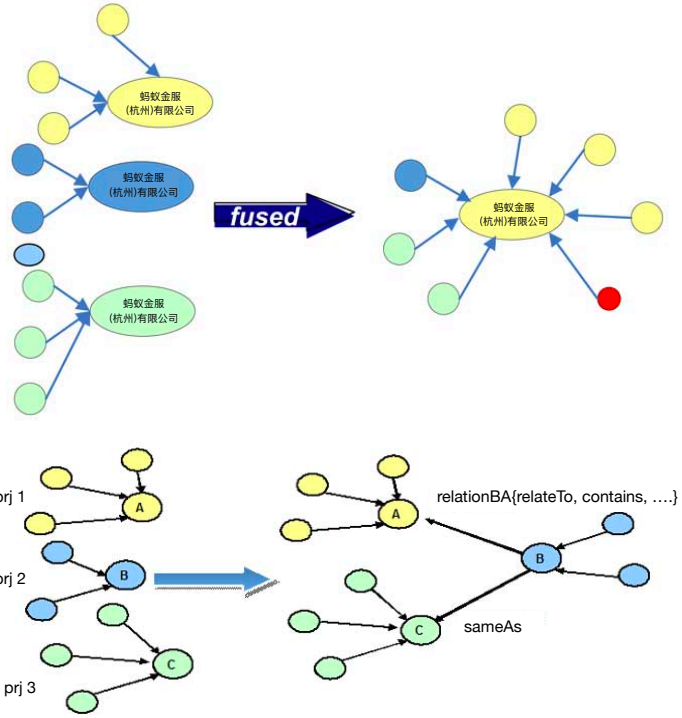
复杂拓扑结构的表示，如何表示知识的领域内聚性

图谱应用业务痛点(迭代演化)

- 每类EntityType，<EntityType1, relation1, EntityType2>独立构建
- 理论上每个属性都可以构建为实体，纠结于成本和效率
- 可迭代性差，Schema膨胀到一定程度，维度爆炸，只能重构

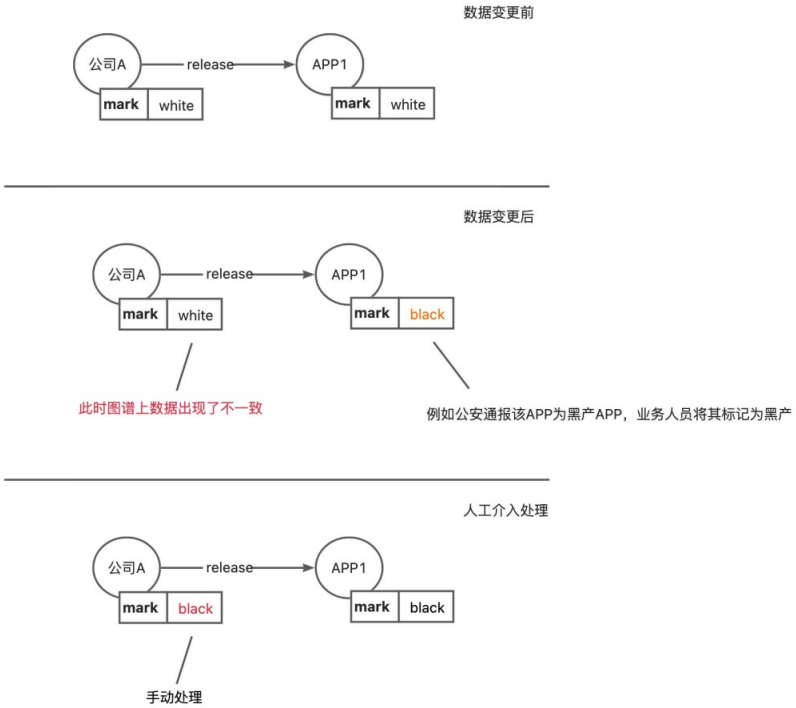


因实体类型颗粒度缺少管控，长期迭代带来Schema爆炸

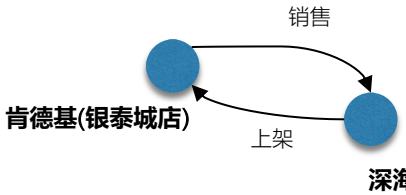


不同领域图谱的类型/实例对齐及互联复用

图谱应用业务痛点(逻辑依赖)



逻辑依赖缺失带来不一致问题

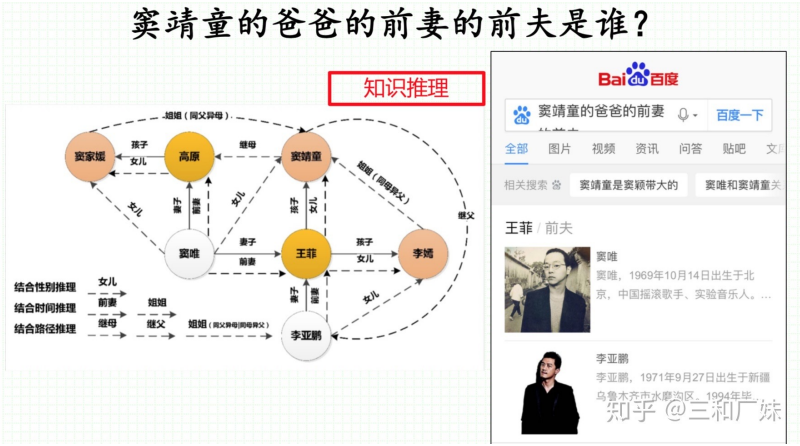


<肯德基(银泰城店), 销售, 深海鳕鱼堡>

->

<深海鳕鱼堡, 上架, 肯德基(银泰城店)>

定义逻辑依赖，可减少重复构建



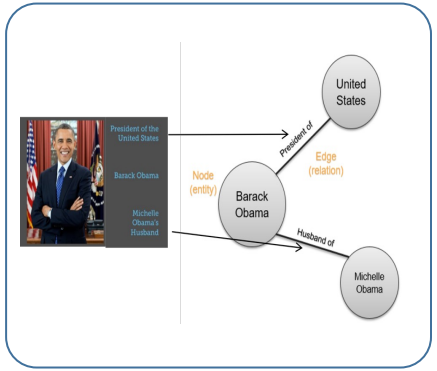
inverseOf: Person(x1) ^ Person(x2) ^ 前妻(x1, x2) -> 前夫(x2, x1)

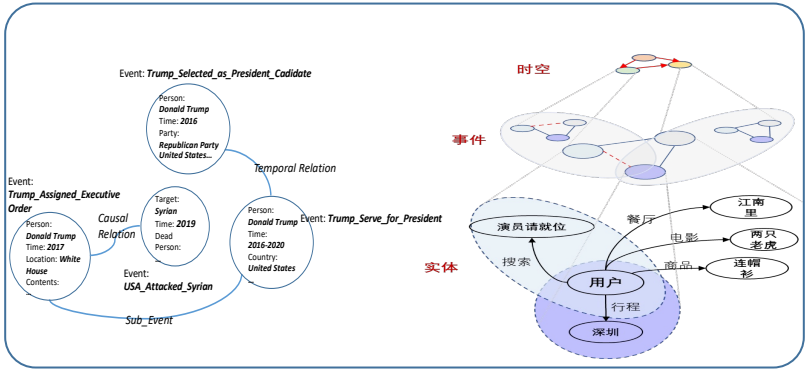
inverseOf: Person(x1) ^ Person(x2) ^ Male(x2) ^ 女儿(x1, x2) -> 爸爸(x2, x1)

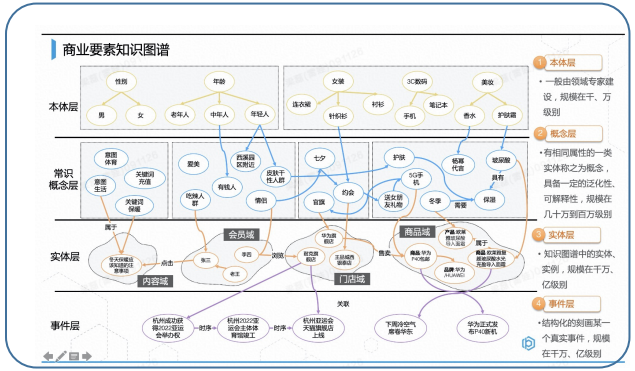
定义逻辑依赖，可实现逻辑推理

新知识管理范式的期待

知识类型从简单到复杂，从静态到动态，从广域到垂域，从平面到时空







(Barack Obama, Spouse, Michelle)

Causal relation, Temporal relation, Co-reference relation, Sub-class relation…

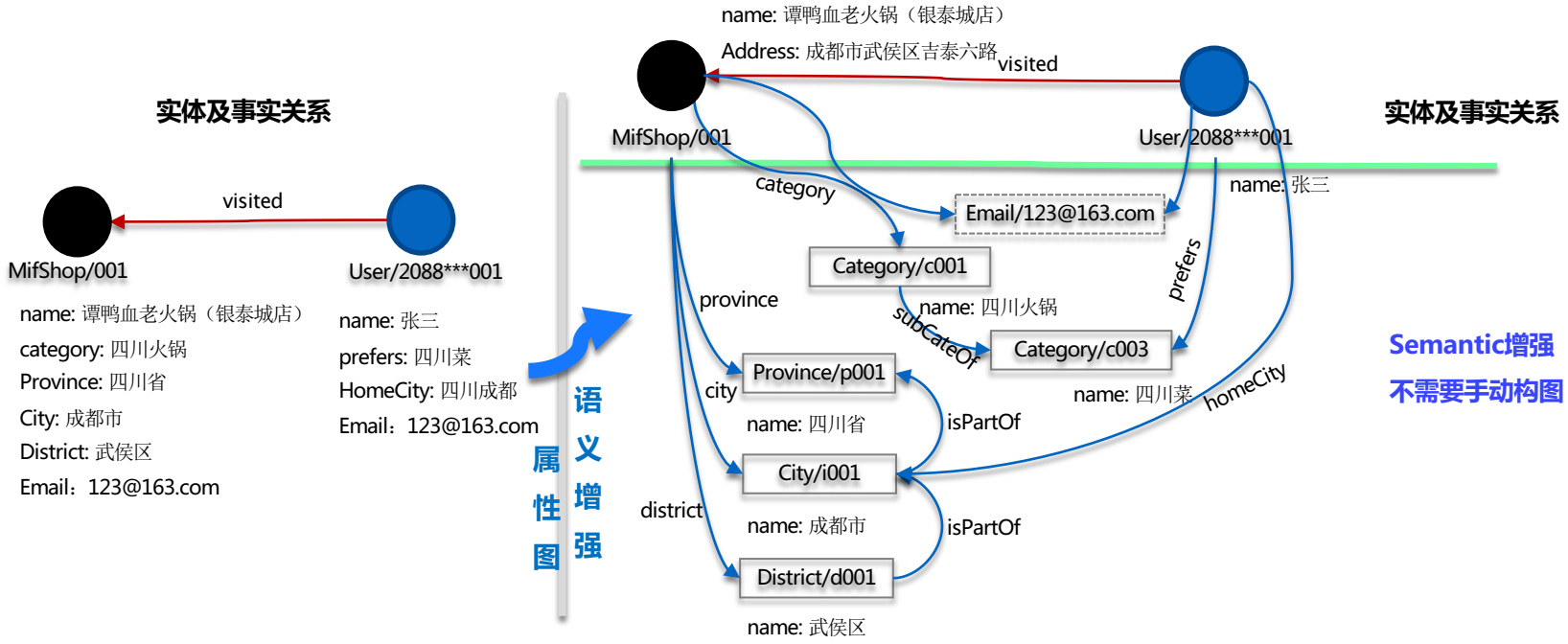
SPO三元组

时空多元关联

构建知识分层，实现动、静分离

目录

SPG: Semantic-enhanced Programmable Graph(语义增强示意)



Things, not Strings

知识的三个显著特点:

1、必须有明确的领域类型(every Thing has a Class)

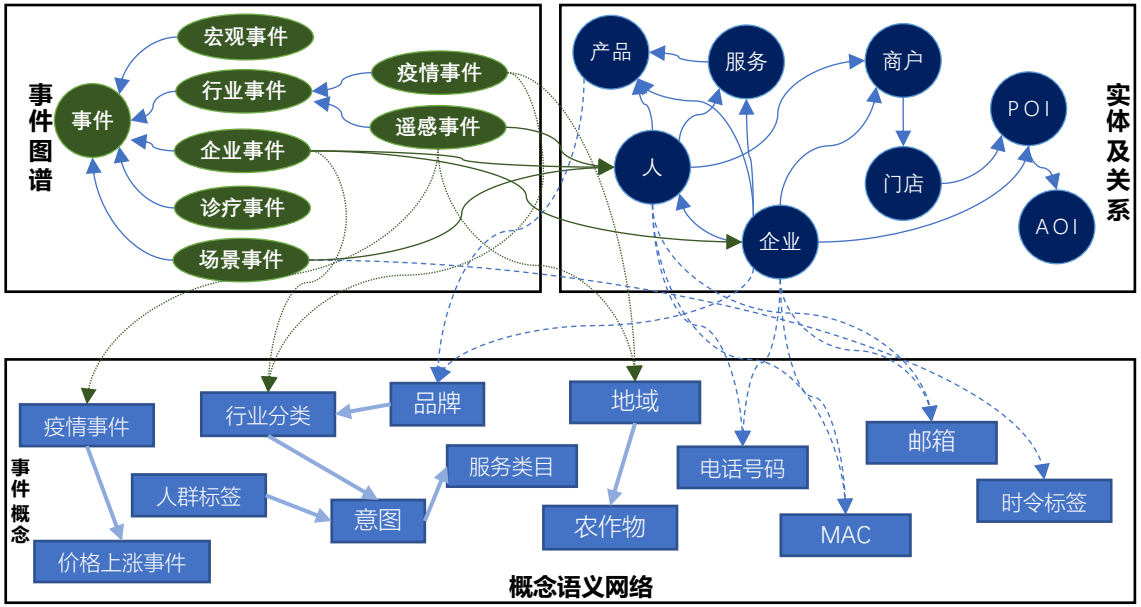
2、每个实例类型内必唯一(each instance is unique within an Entity Class)

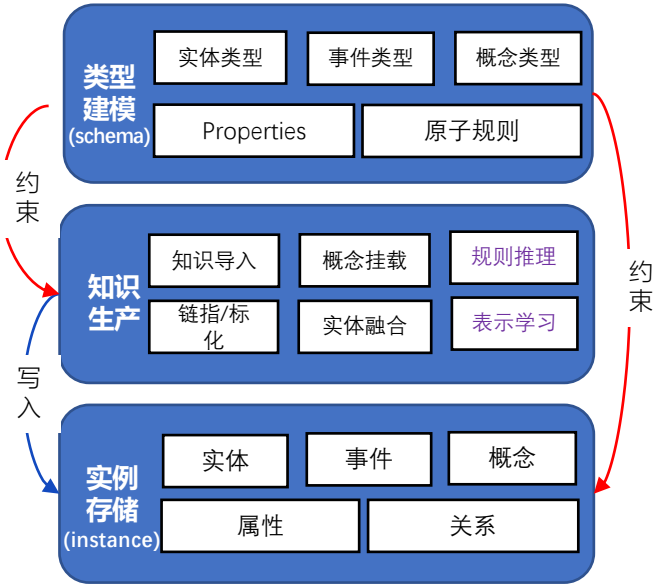
3、语义明确的谓词修饰(nothing exists in isolation)

SPG: 主体知识分类模型(Class–Instance Paradigm)

业界主流的划分为实体、概念，但使用阶段并无清晰界限，我们对知识类型的定义：

- **实体：**业务相关性比较强的客观实例，通过实体Properties(属性、关系)刻画个体画像，如用户、企业、商户等
- **概念：**实体从具体到一般的抽象，表述的是一组实体集合。相对静态、具有较强复用性，如人群标签、领域标准类型、语义词汇(如HowNet)等
- **事件：**加入时间、空间、标的等约束的实体类型，如通过NLP、CV等抽取出来的行业事件、企业事件、诊疗事件等

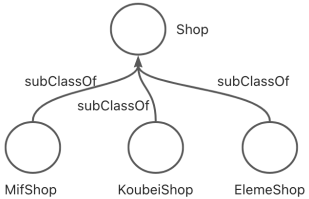




SPG: 主体知识分类模型

实体

支撑业务决策的复合类型，具有唯一实例及ID



伪码表示

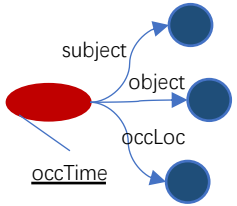
```
EntityClass Shop {
    id          String
    name        String
    shopCate    Category(类目)
    locateAt    AdminArea(行政区划)/省/市/区
    relatePOI   POI(高德POI)
    locCoord    LBSPoint(经纬度)
}
EntityClass MifShop subClassOf Shop{
    mccCate     MccCategory(MCC类目)
    override relatePOI    AntPOI(蚂蚁POI)
}
```

built-in Class predicates:

- subClassOf: 表示schema的复用，父/子类可独立实例化
- deepSubClasssOf: 深度继承
- equivalentClass： 等价类

事件

时空约束的多元要素组合



伪码表示

```
EventClass CompanyEvent {
    subject     Company(企业), Person(自然人)
    object      Company(企业), Person(自然人)
    occTime     Timestamp(时间)
    occLoc      POI(POI)
    ... ...
}
```

Built-in properties:

- subject (multi-class supported)
- object (multi-class supported)

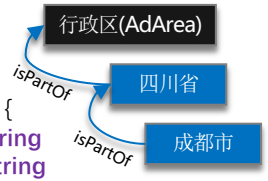
概念

领域知识标准化建模

1、结合领域模型定义

2、领域模型共性抽象

3、较强跨业务迁移性




```
ConceptClass AdminArea {
    hyperConcept        String
    hyperPredicateName  String
    hasAlias List<String>
    synonyms List<String>
}
```

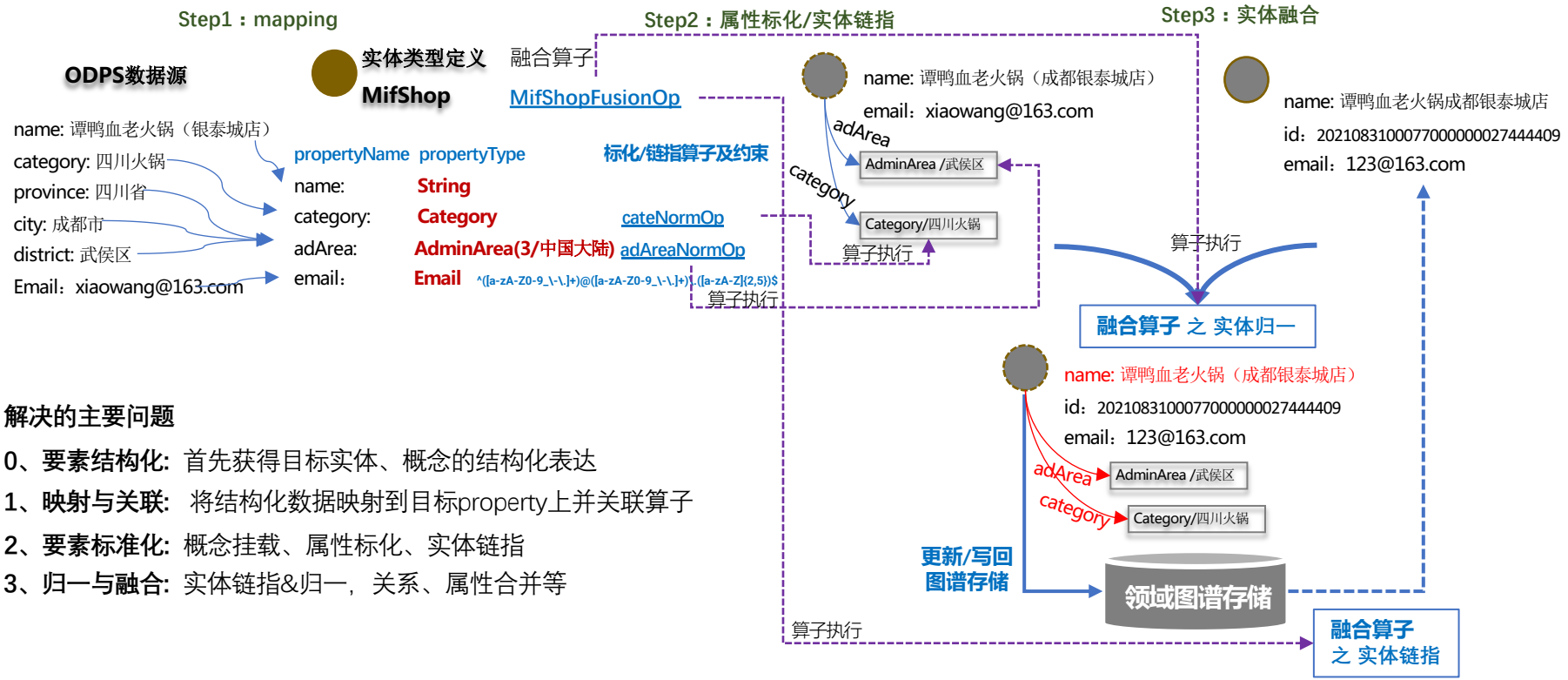
Built-in Concept predicates:

- isPartOf
- locateAt
- isA
- subCategoryOf
- childOf

Built-in Concept properties:

- hypernym
- hasAlias

SPG: 非完备数据集下图谱的构建(Programmable)



解决的主要问题

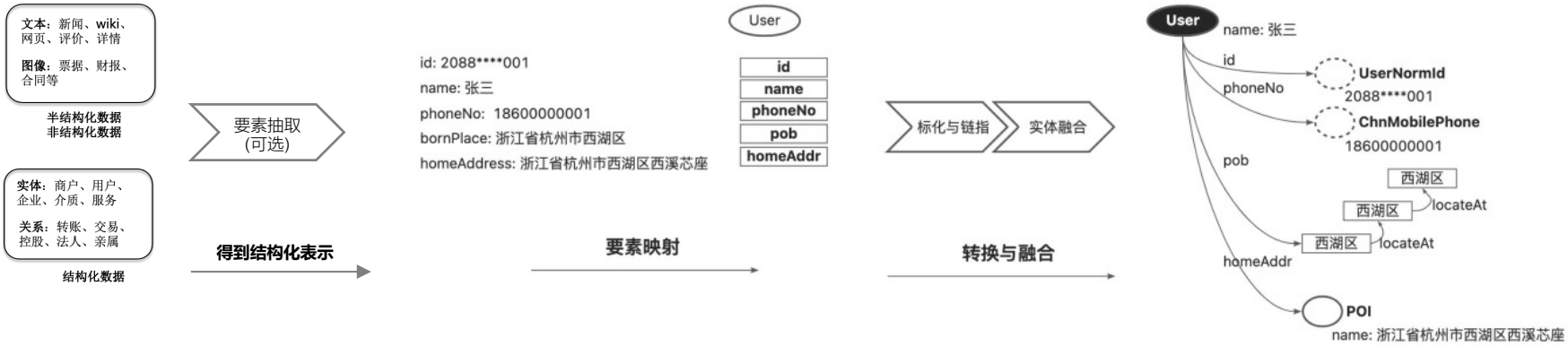
0、要素结构化: 首先获得目标实体、概念的结构化表达

1、映射与关联: 将结构化数据映射到目标property上并关联算子

2、要素标准化: 概念挂载、属性标化、实体链指

3、归一与融合: 实体链指&归一，关系、属性合并等

SPG: 数据到知识转化的编程范式

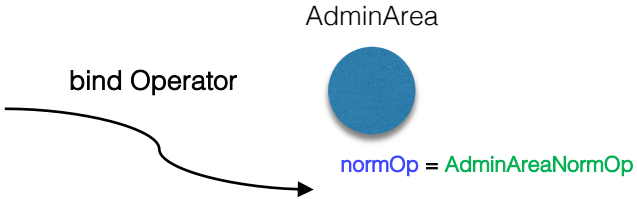


实体类型定义

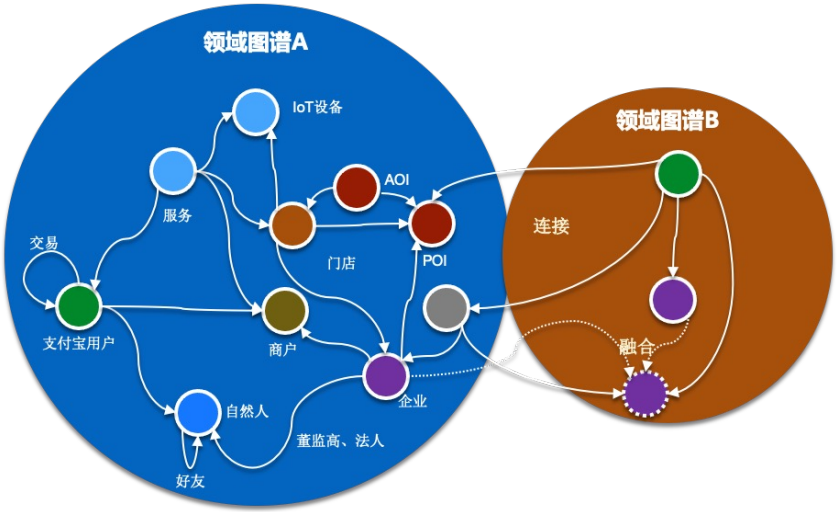
```
EntityClass User {
    id              UserNormId
    name            String
    phoneNo         ChnMobilePhone
    bornPlace       AdminArea
    homeAddress     POI(高德POI)
}
```

算子开发与绑定

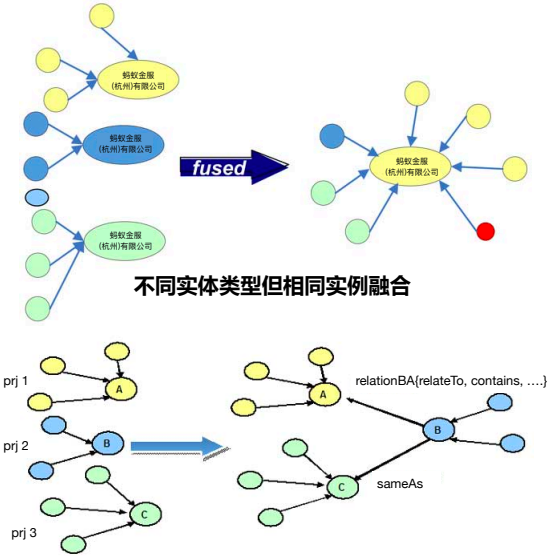
```python
@BaseOp.register("AdminAreaNormOp", bind_to="AdminArea", is_api_iface=True)
class AdminAreaNormOp(PropertyNormalizeOp):
    def eval(self, property: str, record: Vertex = None) -> Union[str, Trace]:
        # property = "中国成都市" ,需要标化到成都
        # 简单模式
        if "成都" in property:
            return "成都"
        # 外部调用,例如调用大模型或者其他NLP模型
        return LLMAdminAreaNorm(property)
```



SPG: 实现连接即可用的跨域知识复用



示意：领域图谱B 融合/复用 领域图谱A

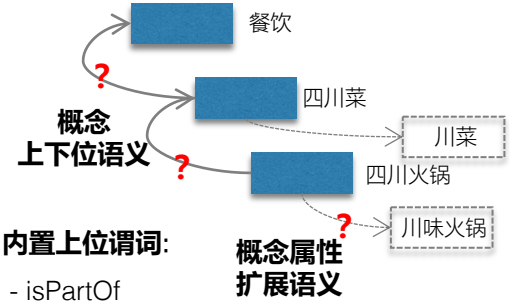


- **跨业务的知识复用：**基于图谱本体模型（面向对象），实现跨业务的知识连接、复用
- **减少无效数据拷贝：**减少无效的数据拷贝，连接即可应用，标准化知识服务链路
- **业务快速价值落地：**减少业务找数据的成本，通过知识复用带来更大业务价值，降本提效

SPG: 谓词语义与逻辑符号(Logical Symbols)

内置谓词:

- equivalentClass
- belongTo
- sameAs

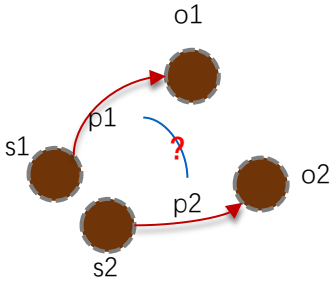


内置上位谓词:

- isPartOf
- subCategoryOf
- isA
- …

内置属性谓词:

- hasAlias
- synonym
- …



内置谓词:

- inverseOf
- mutexOf
- transitive
- equivalentProperty
- subPropertyOf
- symmetricProperty
- normalizedProperty

… …

伪码示例:

```
Create EntityType FusedPOI equivalentClass (
    fuse(AmapPOI, AlipayPOI)
    .withLinkFunction(samePoiSimilarityFunc)
    .withFuseStrategy(){
        FusedPOI.attr1 = isNotBlank(AmapPOI.attr1) ?
AmapPOI.attr1 : AlipayPOI:attr1
        FusedPOI.attrx = （ AmapPOI.attrx1 >
AlipayPOI.attrx2 ）
            … …
    }
)
```

提示词联想示例:

getPrompts(四川火锅)

上位词: 四川菜, 川菜

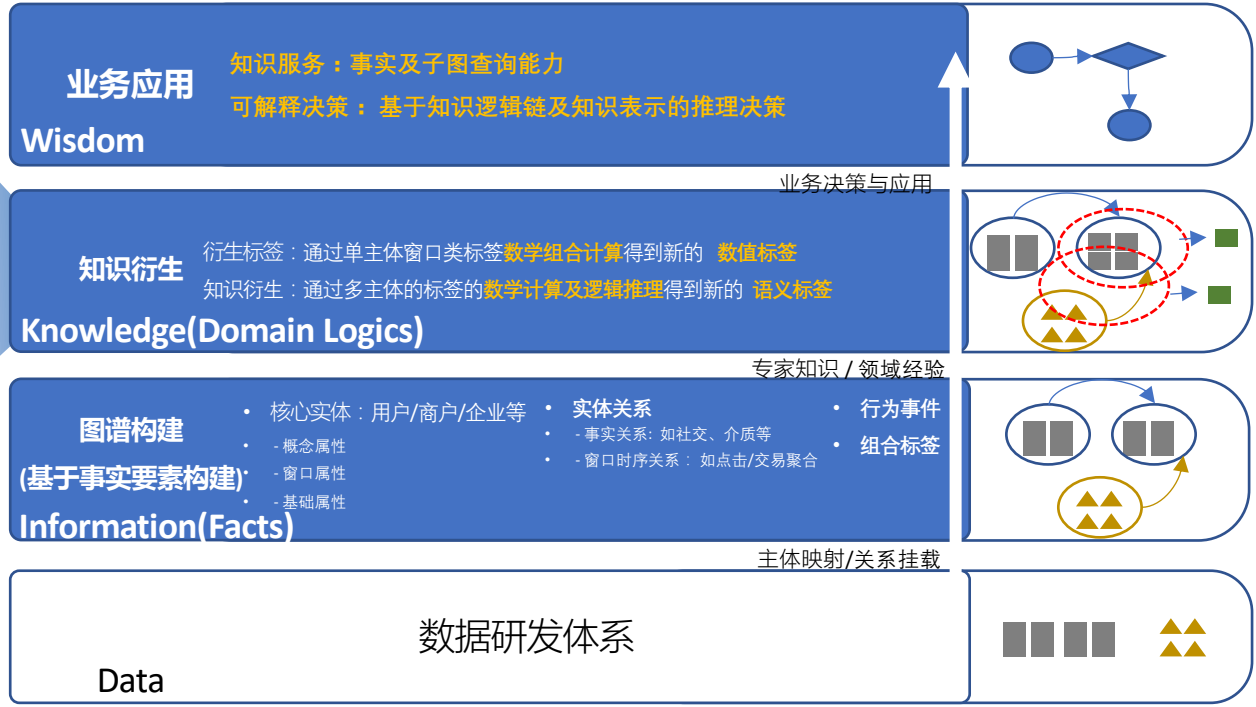
同义词: 川味火锅，川式火锅， 重庆火锅，巴蜀火锅

```
Define (s:MifShop) –[p:hasProduct]->(o:Product)
inverseOf
(s:Product) –[p:availableOn]->(o:MifShop) {
    Rule { }
}

Define (s:User)–[p:belongTo]->(o:Crowd/爱成都火锅人群)
{
    GraphStructure{}
    Rule {
        s.prefers contains(川式火锅)
    }
}

GraphStructure {
    (s1:Crowd/爱成都火锅人群) –[p:visited]->(o:MifShop)
    (s2:Product) –[p:availableOn]->(o:MifShop)
}
Rule {
    s2.category contains(四川菜)
}
Action {
    get(s2.name)
}
```

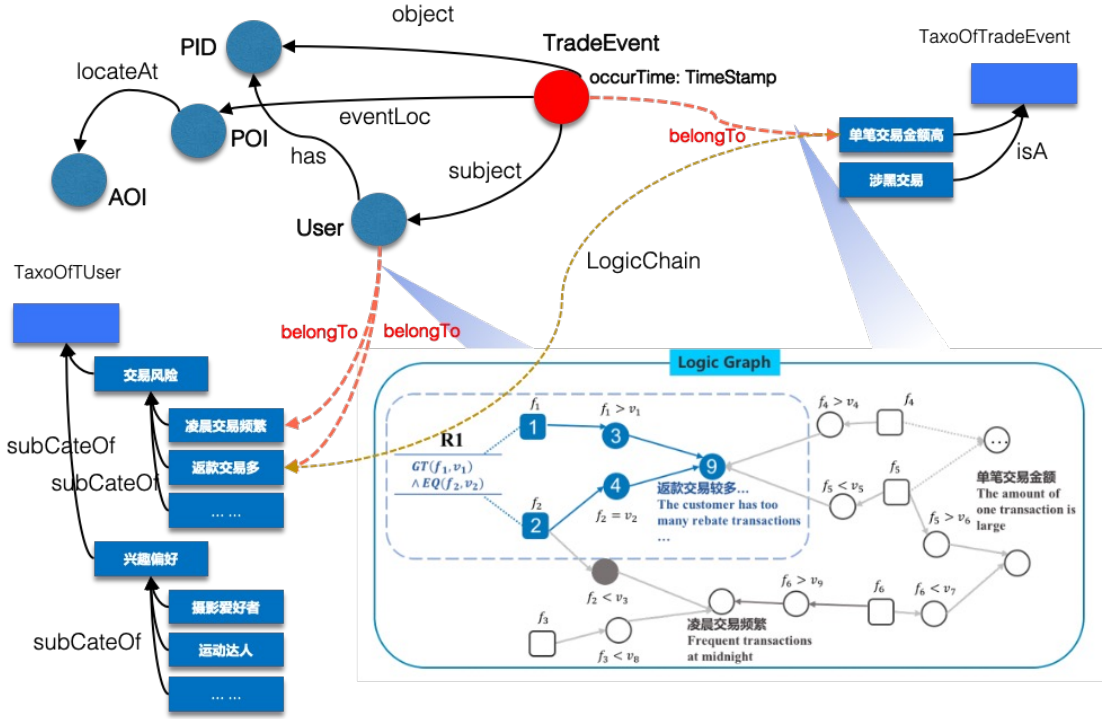
SPG: 基于语义逻辑实现知识分层

```
Define (s:User)-[p:belongTo]->(o:TuringCrowd/大龄未婚青年) {
  GraphStructure {
  }
  Rule {
    R1: s.age > 25 && s.age < 50
    R2: s.marriageStatus = '未婚'
    p: R1 && R2
  }
}
```

```
Define (s:User)-[p:belongTo]->(o:TuringCrowd/高收入大龄未婚青年) {
  GraphStructure {
    (s)-[p1:belongTo]->(o1:TuringCrowd/大龄未婚青年)
  }
  Rule {
    R1: s.incomeLevel = '大于等于4万'
    p : R1 and p1
  }
}
```



SPG: 基于语义逻辑实现LogicChain



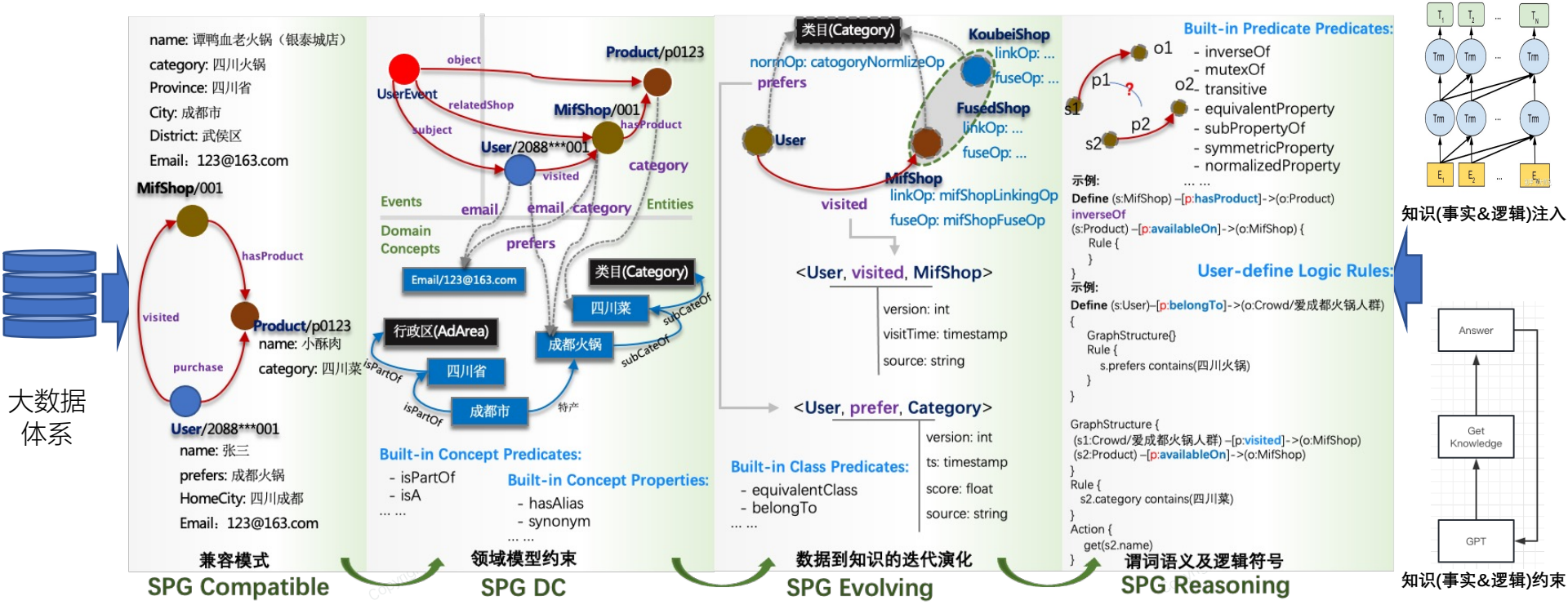
涉黄用户: 凌晨交易频繁 & 单笔交易金额高

大额套现用户：返款交易多

```
Define (e:TradeEvent)–[p:belongTo]->(o:TaxoOfTradeEvent/单笔交易金额高)
{
    GraphStructure{}
    Rule {
        e.amount > 500
    }
}
Define (s:User)–[p:belongTo]->(o:TaxoOfUser/交易风险/返款交易多)
{
    GraphStructure{
      (e1:TradeEvent)-[ps1:subject]->(su1:User)
      (e1:TradeEvent)-[pp1:object]->(sp1:PID)
      (e2:TradeEvent)-[ps2:subject]->(su2:User)
      (e2:TradeEvent)-[pp2:object]->(sp2:PID)
      (su1)-[has]->(sp2)
      (su2)-[has]->(sp1)
      (e2)-[pb:belongTo]->(o:/TaxoOfTradeEvent/单笔交易金额高)
    }
    Rule {
        s.id == su1.id
        e1.ts < e2.ts and hour(current_time()) - hour(e1.ts) < 24
        group(s).count() > 10
    }

}
Define (s:User)–[p:belongTo]->(o:TaxoOfUser/交易风险/凌晨交易频繁)
{
    GraphStructure{
      (e1:TradeEvent)-[ps1:subject]->(su1:User)
    }
    Rule {
        s.id == su1.id
        hour(e1.occurTime) between(0, 4)
    }
}
```

SPG: Semantic-enhanced Programmable Graph(L1 – L3)

From Big Data to LLMs, help machines understand the world



目录

谢谢！

SPG框架 + OpenKG 共建

2023/08月后逐步推出… ...