《大数据管理》第5章作业 2023.9.24

大数据2101班 李嘉鹏 U202115652

1.（1）RDF表示法的陈述采用什么语法？简述RDF一句陈述包含的各项内容。

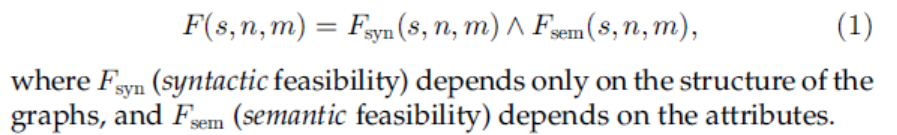
采用XML语法。RDF陈述的形式是一个<资源，属性，属性值>三元组，类似于（主语，谓语，宾语）的表达。

（2）RDFS的目的是要解决RDF的什么问题？OWL又在RDFS之后解决了什么问题？OWL语言的基础及其对知识的处理能力有哪些？

RDFS可以解决RDF无法描述新概念的问题。它在RDF基础上增加了一些词汇来拓展RDF的能力，可以自定义词汇用来描述新概念。

OWL解决了无法理解词汇之间联系的问题。它具有更强的机器解释能力，可以理解词汇之间的联系（例如反义词、同义词）。OWL的语言基础是计算逻辑，其对知识的处理能力包括推理、验证知识的一致性、使隐形知识显性化。

2.（1）简述VF2子图查询算法的7条启发式规则的含义，其中第2条规则可分为哪3部分？



（1）规则1也就是Fsem(s,n,m)，需要保证新加入节点对后，对应节点和边的label要全部相同。

（2）规则2也就是Fsyn(s,n,m)，可以分为三个部分：

第一，将新节点对加入状态s后，检查新的状态s′还是一致状态，对应规则3、4​；

第二，考虑1-lookahead，判断未来一步是否可能构成一致状态s′′，对应规则5、6；

第三，考虑2-lookahead，判断未来两步是否可能构成一致状态s′′′，对应规则7。



（3）前驱节点规则：也就是Rpred。对于查询节点的每一个前驱节点，目标节点一定存在与之对应的前驱节点；

（4）后继节点规则：也就是Rsucc。对于查询节点的每一个后继节点，目标节点一定存在与之对应的后继节点；

（5）1-lookahead的in节点（前驱）规则：也就是Rin。在查询图中，以已匹配节点为前驱节点（也就是指向已匹配节点）的小图节点数量一定不大于大图节点数量；

（6）1-lookahead的out节点（后继）规则：也就是Rout。在查询图中，以已匹配节点为后继节点（也就是从已匹配节点出发）的小图节点数量一定不大于大图节点数量；

（7）2-lookahead的in和out节点规则：也就是Rnew。在查询图中，与“未匹配节点”邻接的小图节点数一定不大于大图相对应的节点数量。

（2）大数据环境下若要通过查询图分解完成子图查询，请简述计算过程。

①每个节点上的子查询都完成一个顶点的边匹配；

②节点k执行针对子查询qi的结果汇总，包括来自其它节点的结果；

③节点k按照stwig的排列顺序执行结果之间的join操作，得到本地完整子图集合；

④最终结构由所有节点汇总而成。

3.现有如下所示属性图，完成下列3个小题：

1）写出该属性图的（V,L,E,A）四元组形式的内容描述；

V={N1, N2, N3, N4}

L={人，电影，出演，导演}

E={E1, E2, E3, E4}

A={姓名，性别，名称，角色，来源}

N1.L={人}，N2.L={电影}，N3.L={人}，N4.L={人}

N1.A={姓名，性别}，N2.A={名称}，N3.A={姓名，性别}，N4.A={姓名，性别}，N1[姓名]=张三，N1[性别]=男，N2[名称]=饼侠，N3[姓名]=李四，N3[性别]=男，N4[姓名]=赵某，N4[性别]=女

E1.L={出演}，E2.L={导演}，E3.L={出演}，E4.L={出演}

E1.A={角色，来源}，E3.A={角色，来源}，E4.A={角色，来源}，E1[角色]=男一号，E1[来源]=百度百科，E3[角色]=男二号，E3[来源]=百度百科，E4[角色]=女一号，E4[来源]=搜狐

2）要找出编号为N1和N3的两个人之间的一条长度不超过10的最短路径，写出相应的Cypher查询语句；

MATCH(s:人{姓名:‘张三’})

MATCH(e:人{姓名:‘李四’})

MATCH p = shortestPath( (s)-[\*..10]->(e))

RETURN p

或：

MATCH (s) WHERE id(s) = ‘N1’

MATCH (e) WHERE id(e) = ‘N3’

MATCH p = shortestPath( (s)-[\*..10]->(e))

RETURN p

3）要找出参演过“张三”自导自演电影的演员姓名及电影名称，写出相应Cypher查询语句。

MATCH (n1:人{姓名:’张三’})-[:导演]->(n2:电影)

MATCH (n1:人{姓名:’张三’})-[:出演]->(n2:电影)

MATCH (n3)-[:出演]->(n2:电影)

RETURN distinct n3.姓名,n2.名称

4.（1）写出图计算BSP模型的代码模板；

void Compute(MessageIterator\* msgs) {

//遍历由顶点入边传入的消息列表

for(; !msgs->Done(); msgs->Next())

doSomething()

//生成新的顶点值

\*MutableVertexValue() = …

//生成沿顶点出边发送的消息

SendMessageToAllNeighbors(…);

}

（2）若采用GAS图计算模型，请按照其代码模板写出“查询所有结点二跳邻居数”的伪代码。（注：可在函数参数中包含图计算的迭代轮次数）

# 共两轮迭代  
for iterations in range(2):  
    # Gather阶段：对收到的消息进行汇总  
    for each vertex u in graph:  
        for each incoming edge VU to u:  
            received\_msg = receive\_message(VU)  
             msg\_sum = sum\_messages(received\_msg)  
  
        # Apply阶段：更新节点状态  
        apply(u, msg\_sum)  
  
    # Scatter阶段，向邻居发送消息  
    if iterations == 0: # 第一次迭代所有点向邻居点传播一个带自身ID，生命值为2的消息  
        for each vertex u in graph:  
           for each outgoing edge UV from u to V:  
              send\_message(UV, create\_message(u.id, 2))  
    else: # 第二次迭代，所有点将收到的消息向邻居点再转发一次，生命值为1  
        for each vertex u in graph:  
            for msg in u.msg\_sum:  
                for each outgoing edge UV from u to V:  
                     send\_message(UV, create\_message(msg.id, 1))

for each vertex u in graph:  
    for each incoming edge VU to u:  
        received\_msg = receive\_message(VU)  
     if msg.HP == 1

TwoStepNeighborList(u) += msg.id #将这个ID计入顶点u的二跳邻居列表