



第九届中国数据库技术大会
DATABASE TECHNOLOGY CONFERENCE CHINA 2018

基于机器学习和图数据库 实现即时推荐引擎

俞方桦 博士

Neo4j Inc. APAC



DTCC
2018

2018.05.10 – 12 北京国际会议中心



IT168.com

ChinaUnix

ITPUB

俞方桦 博士 | Joshua Yu

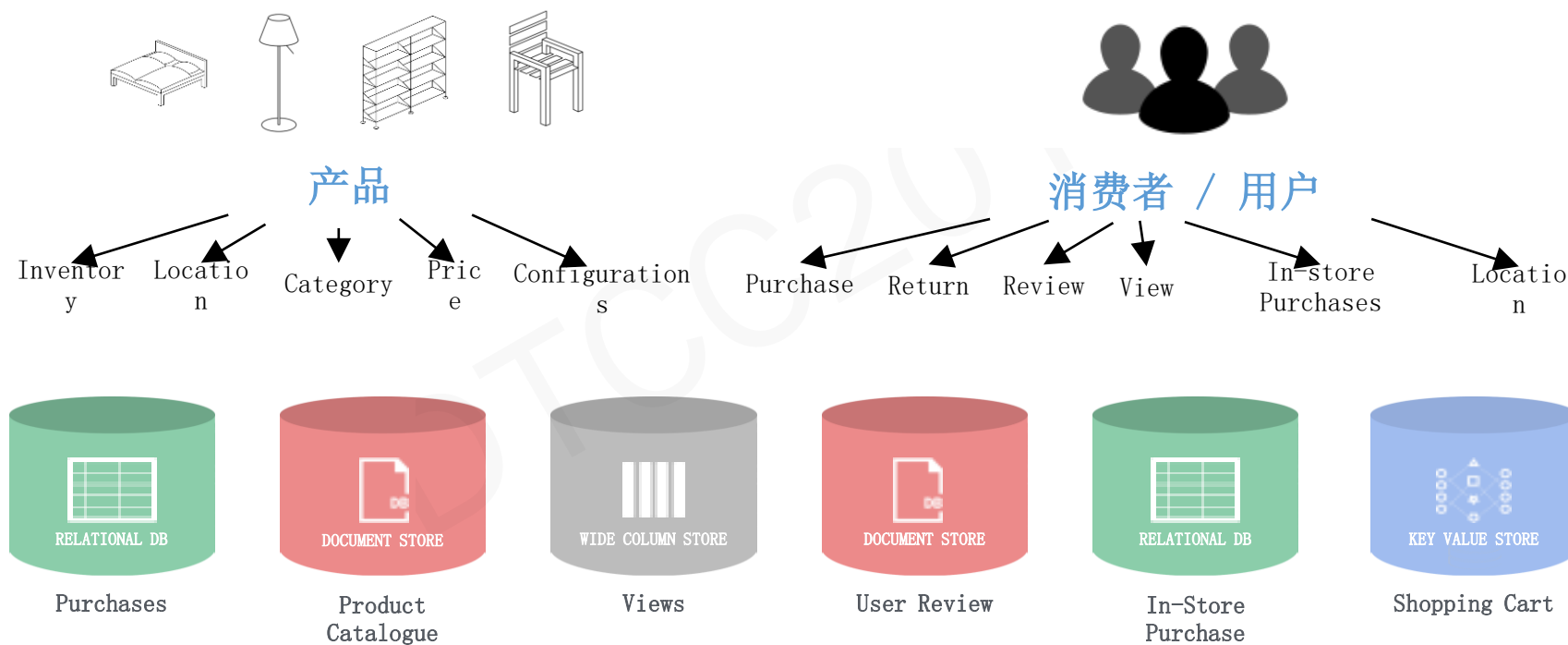
Field Engineering, Neo4j Inc. APAC

Joshua.yu@neo4j.com



推荐引擎的**即时性**、**精准性**、**相关性**是大数据时代网络营销的成功秘诀。

来自架构的挑战



使用图数据库建立关于产品和客户的“知识图谱”

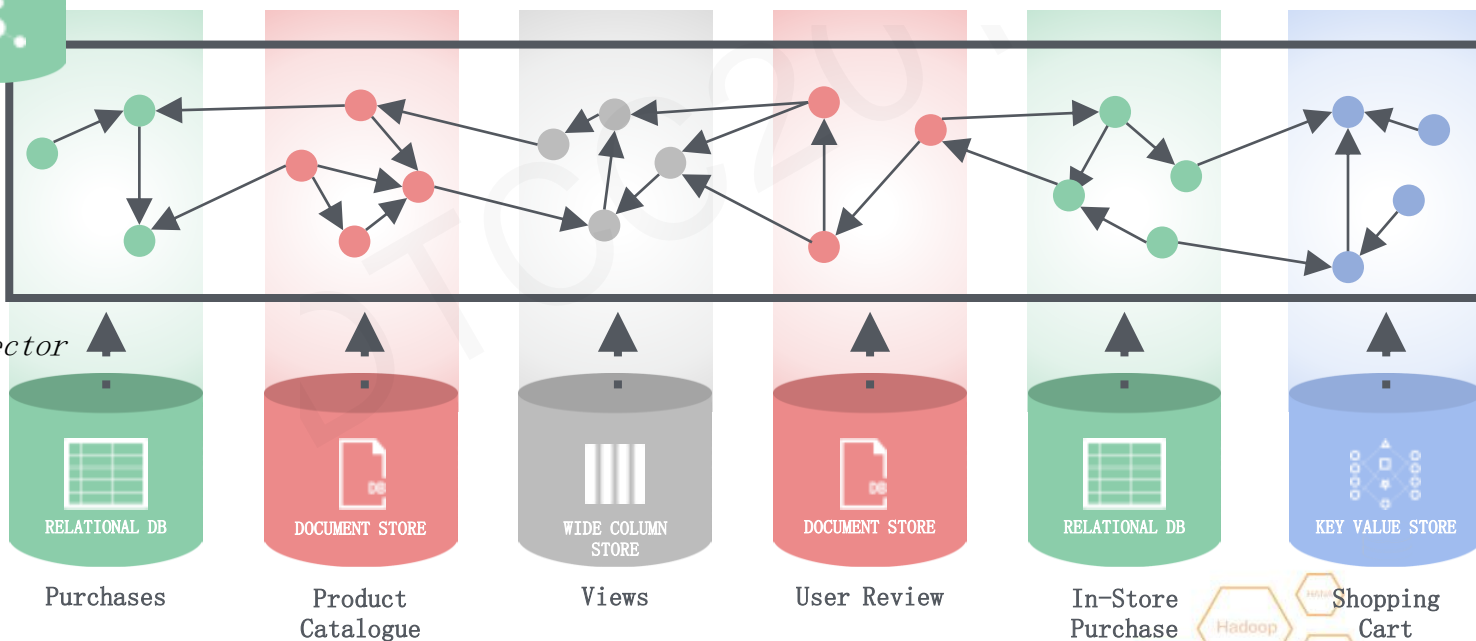


Apps and Systems

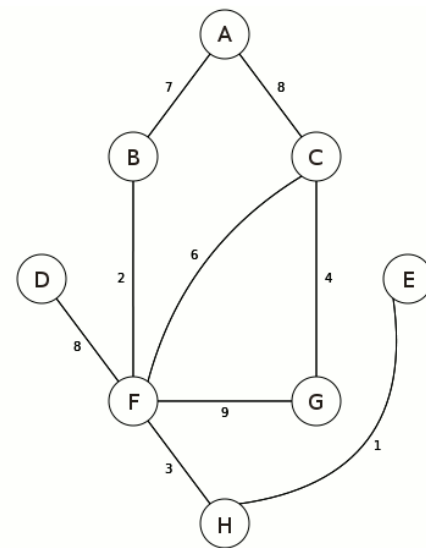
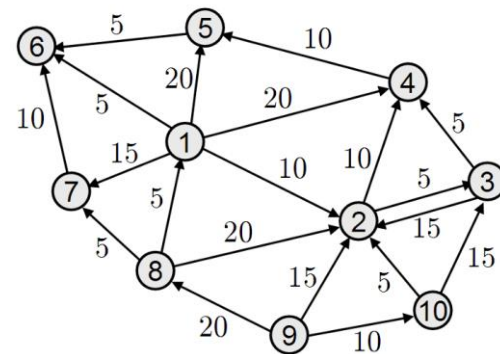
Drivers: Java | JavaScript | Python | .Net | PHP | Go
| Ruby

Real-Time
Queries

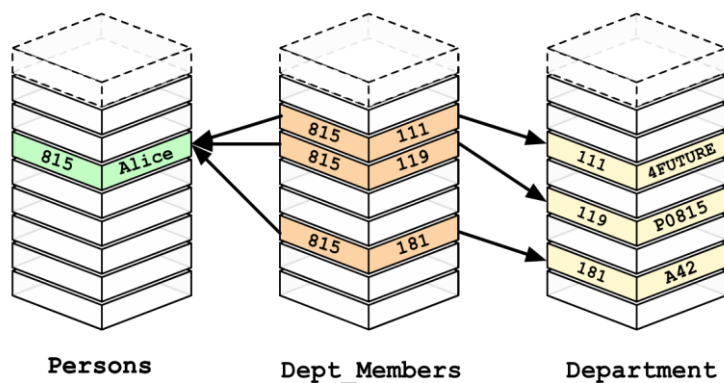
Connector



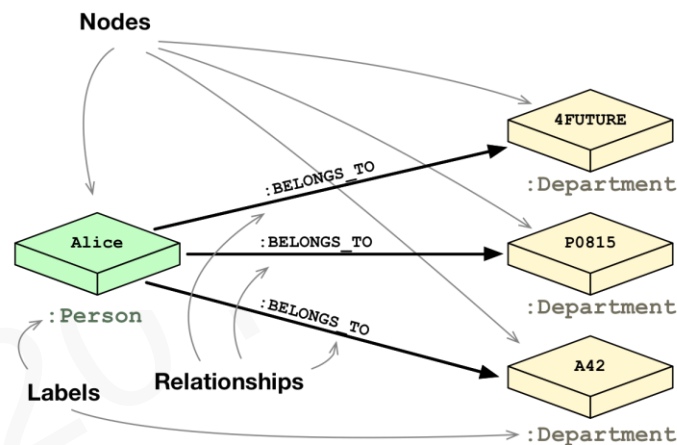
什么是“图”数据库？



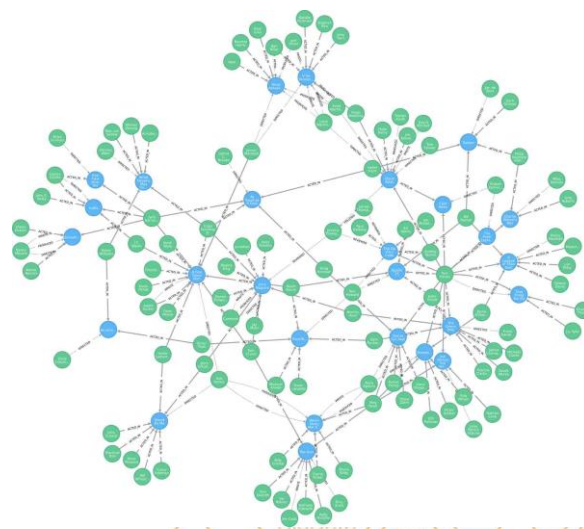
对比：关系型和图数据库



存储



模式



比较图和关系数据库：复杂查询的性能



Table 2-1. Finding extended friends in a relational database versus efficient finding in Neo4j

Depth	RDBMS execution time(s)	Neo4j execution time(s)	Records returned
2	0.016	0.01	~2500
3	30.267	0.168	~110,000
4	1543.505	1.359	~600,000
5	Unfinished	2.132	~800,000

测试说明：

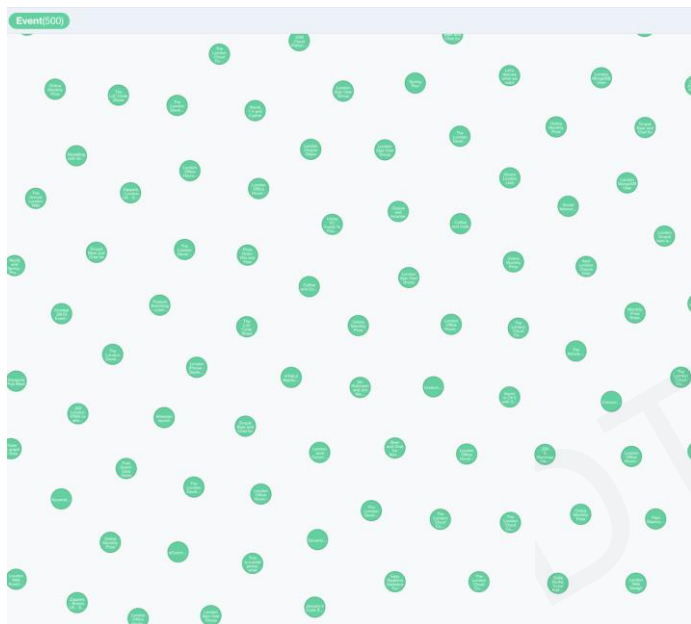
- 1百万成员的社交网络
- 平均每个人有50个朋友

参见：Graph Database 2nd Edition by O'Reilly 2017

推荐引擎的抽象表示 – 问题定义



无序的数据项



相关知识

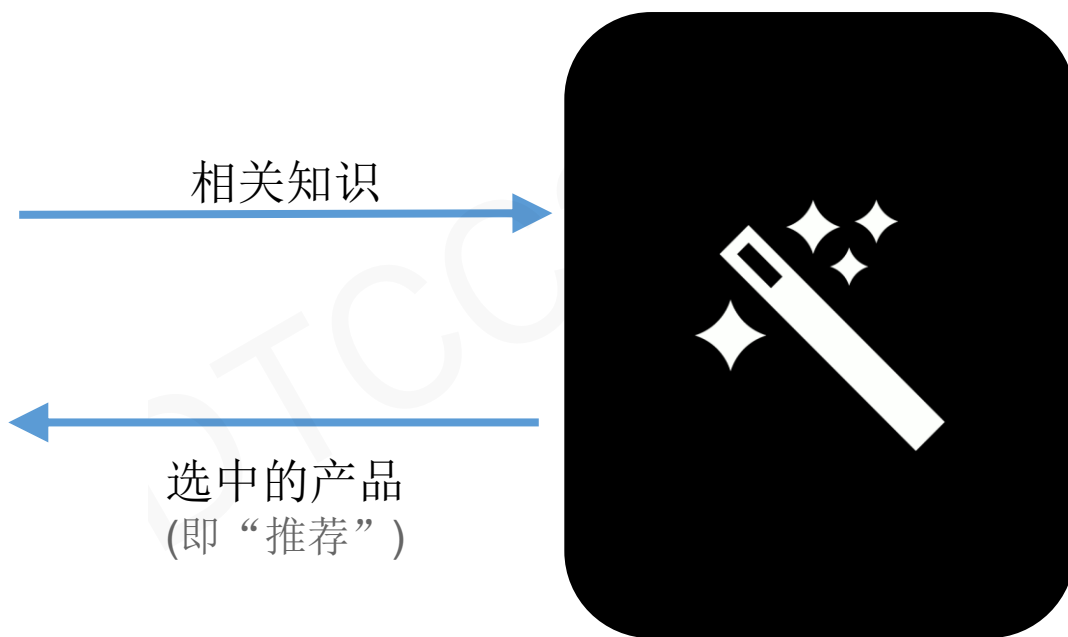
高度相关的查询结果

Recommendation
Intro to Graphs with Neo4j
Social Network Analysis and Visualization with Neo4j
Intro to Graphs with Neo4j
Social Network Analysis and Visualization with Neo4j
Neo4j's New Clustering Architecture: Raft consensus and causal consistency
Intro to Graphs with Neo4j
Neo4j and the Panama Papers
RDF to Neo4j. A worked example
Intro to Graphs with Neo4j
Fraud Detection using Neo4j

排序 /
相关度

限制 / 最多条目

推荐引擎的抽象表示 – 基本功能



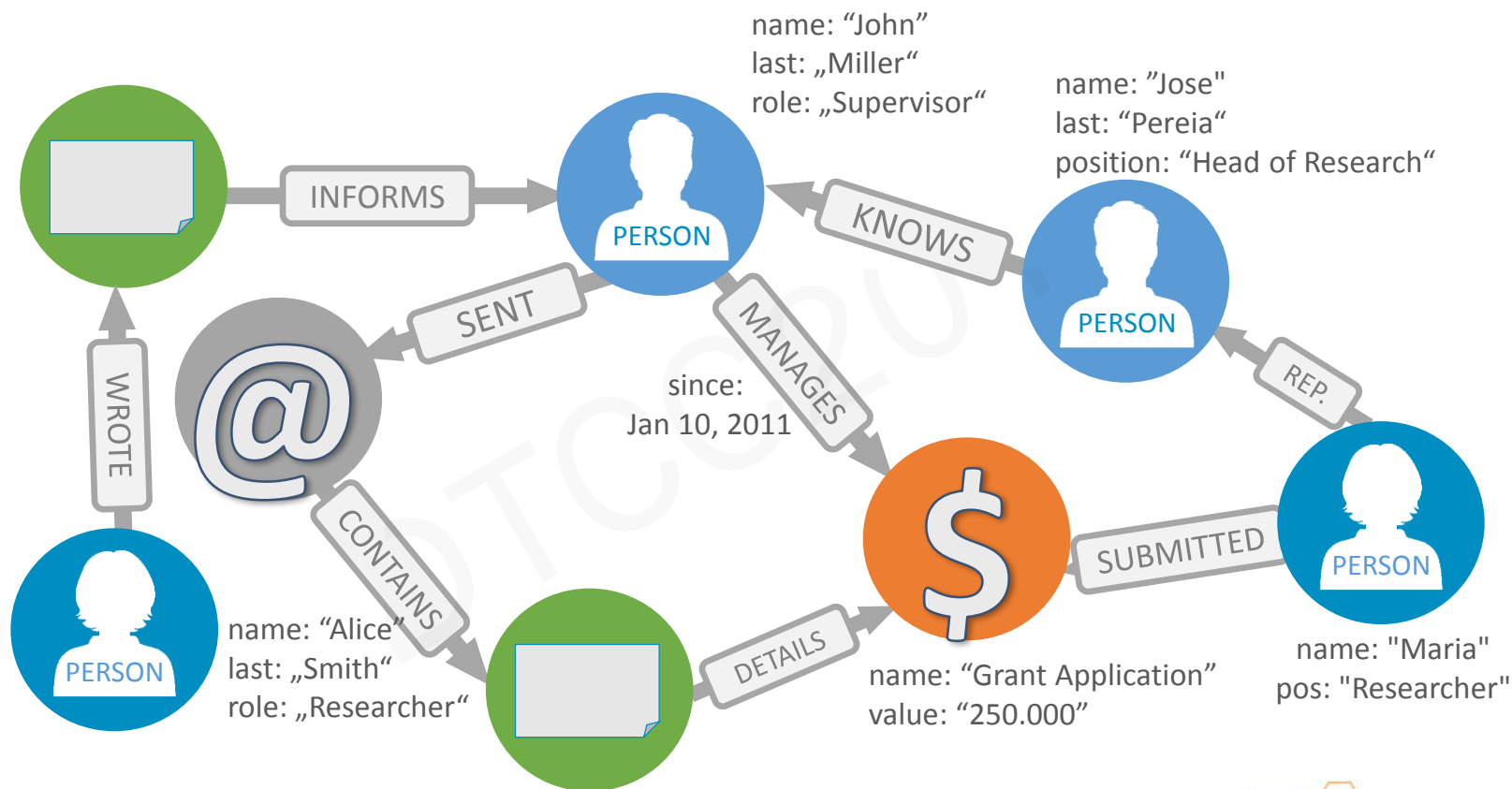
提高精准度需要借助相关知识(Context)

- 用户
- 用户过去的交互
- 已经输入的搜索内容
- 一天中的时间段/一周中的某天/季节
- 天气状况
- 评价
- ...

推荐引擎的抽象表示 – 完整的功能



图提供丰富的相关知识



应用实例 – Meetup活动推荐



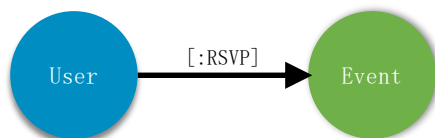
Meetup

如何向用户推荐最符合他/她需求和兴趣的聚会活动，
以提高活动的报名人数和出席率？

Meetup活动推荐 – 推理和发现



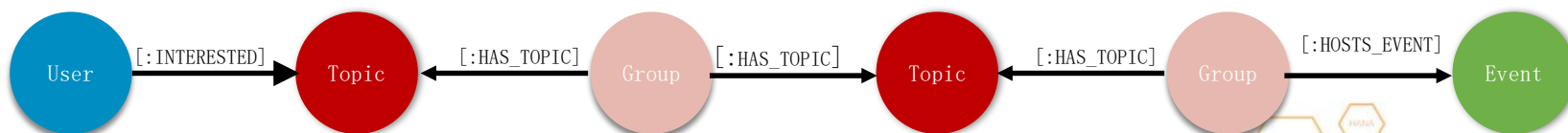
- 无效的推荐(已经参加的活动)



- 有效的推荐：匹配感兴趣的主题



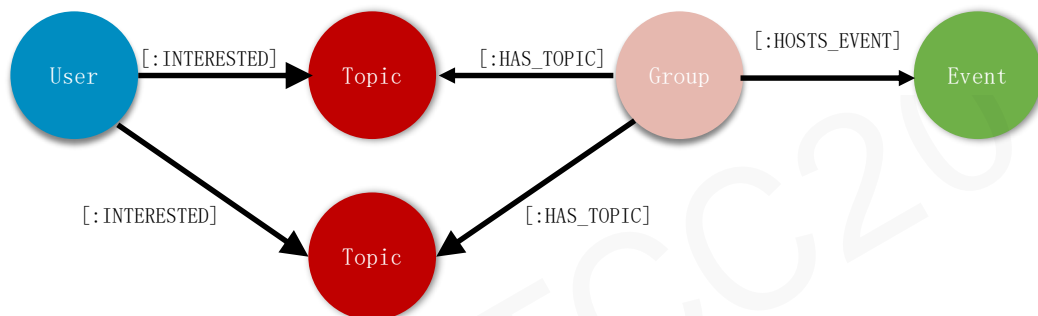
- 接近度：越远的关系相关度越小



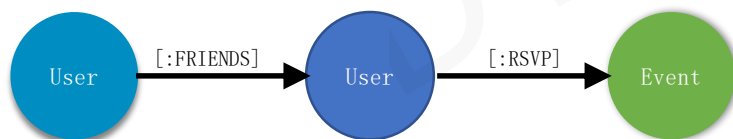
Meetup活动推荐 – 推理和发现(续)



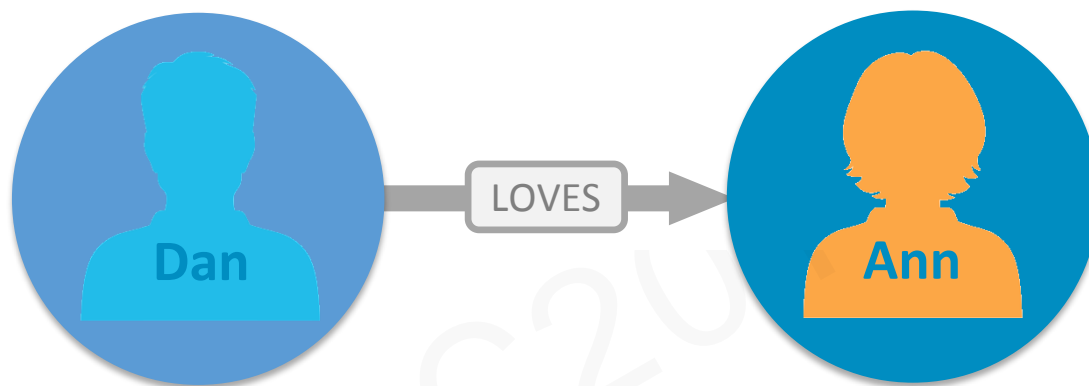
- 密度: 越多路径达到的节点越相关



- 替代的路径



Cypher : 图数据库查询语言



NODE

Relationship

NODE

`(:Person { name:"Dan" }) -[:LOVES]-> (:Person { name:"Ann" })`

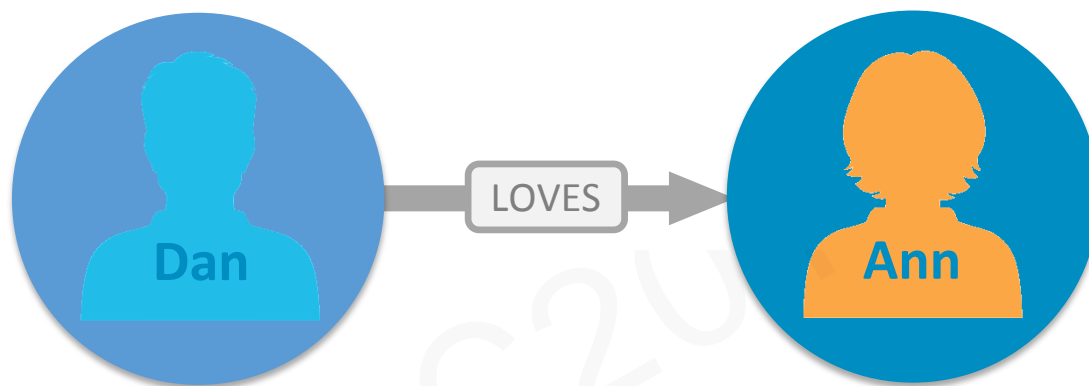
LABEL

PROPERTY

LABEL

PROPERTY

Cypher : 创建节点和关系



NODE

Relationship

NODE

```
CREATE (:Person { name:"Dan" }) -[:LOVES]-> (:Person { name:"Ann" })
```

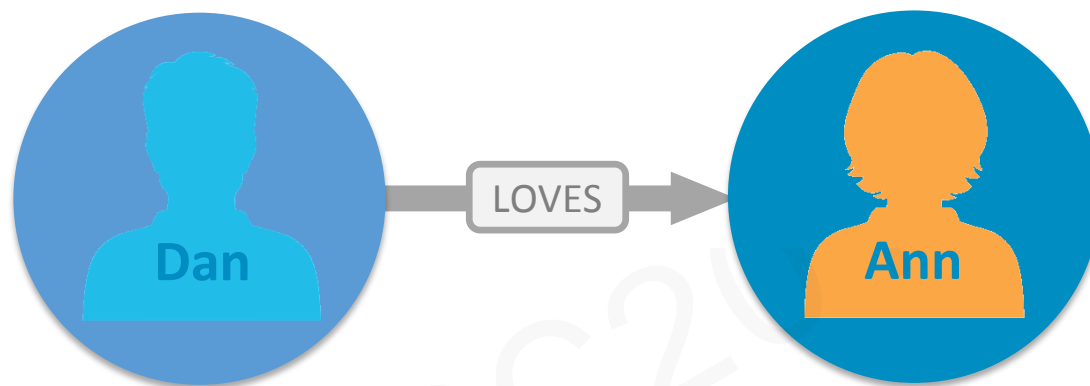
LABEL

PROPERTY

LABEL

PROPERTY

Cypher : 匹配图模式



NODE

Relationship

NODE

```
MATCH (:Person { name:"Dan" } ) -[:LOVES]-> ( whom ) RETURN whom
```

LABEL

PROPERTY

VARIABLE

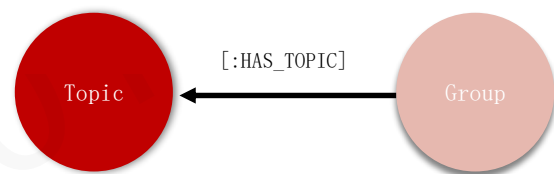
Meetup活动推荐 – 推理和发现(续)



注：以下的查询均用Cypher语言实现。

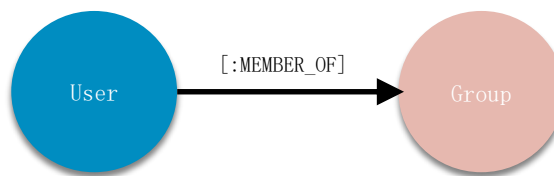
- 发现最热门的主题

```
1 MATCH (t:Topic)<-[:HAS_TOPIC]-()  
2 RETURN t.name, COUNT(*) AS count  
3 ORDER BY count DESC
```



- 发现已经参加的聚会

```
1 MATCH (m:Member)-[:MEMBER_OF]->(group)  
2 WHERE m.name = {name}  
3 RETURN group
```

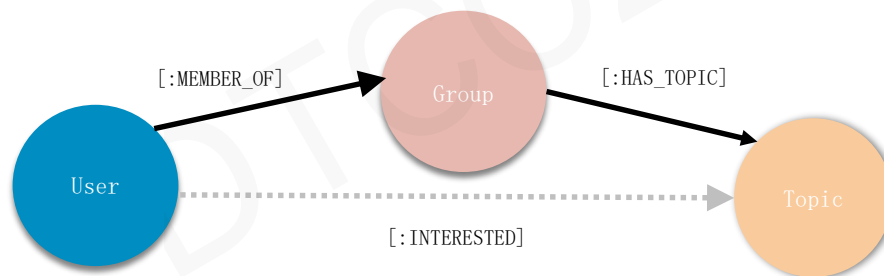


Meetup活动推荐 – 推理和发现(续)



- 推导最可能感兴趣的主题

```
1 MATCH (member:Member {name: {name}})-[:MEMBER_OF]->()-[:HAS_TOPIC]->(topic)
2 WHERE NOT ((member)-[:INTERESTED_IN]->(topic))
3 RETURN topic.name, COUNT(*) AS times
4 ORDER BY times DESC
```



- 三元闭包(Triadic Closure)的应用

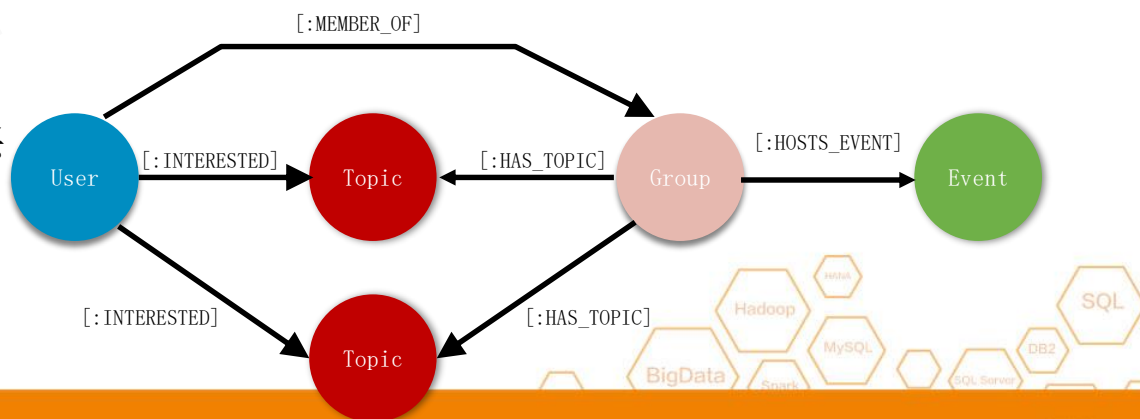
Meetup活动推荐 – 推理和发现(续)



- 推荐最可能感兴趣的活动

```
1 MATCH (member:Member) WHERE member.name CONTAINS {name}
2 MATCH (futureEvent:Event) WHERE futureEvent.time > timestamp()
3
4 WITH member, futureEvent, EXISTS((member)-[:MEMBER_OF]->()-[:HOSTED_EVENT]->(futureEvent)) AS myGroup
5 OPTIONAL MATCH (member)-[:INTERESTED_IN]->()-[:HAS_TOPIC]->()-[:HOSTED_EVENT]->(futureEvent)
6
7 WITH member, futureEvent, myGroup, COUNT(*) AS commonTopics
8 MATCH (futureEvent)<-[:HOSTED_EVENT]-(group)
9
10 RETURN futureEvent.name, futureEvent.time, group.name, commonTopics, myGroup
11 ORDER BY futureEvent.time
```

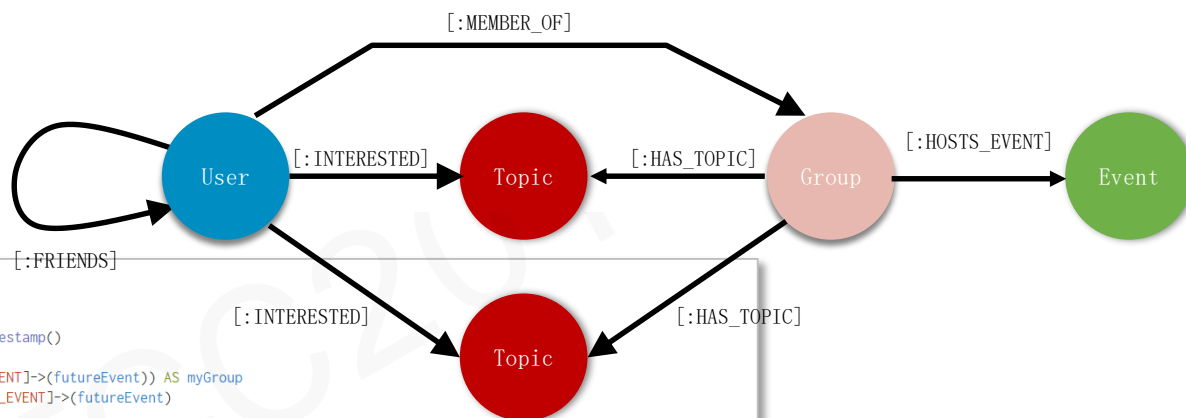
1. 找到所有未来的活动;
2. 看看这些活动是不是属于已经加入的兴趣组;
3. 统计共同的兴趣主题;
4. 找到那些组织这些活动的兴趣组;
5. 按照时间对结果排序。



Meetup活动推荐 – 推理和发现(续)



- 推荐朋友已经报名的活动

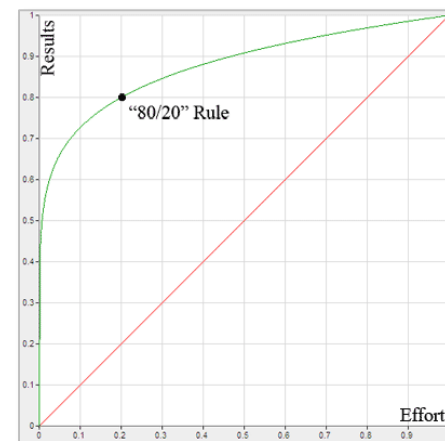


```
1 MATCH (member:Member) WHERE member.name CONTAINS {name}
2
3 MATCH (futureEvent:Event)
4 WHERE timestamp() + (7 * 24 * 60 * 60 * 1000) > futureEvent.time > timestamp()
5
6 WITH member, futureEvent, EXISTS((member)-[:MEMBER_OF]->()-[:HOSTED_EVENT]->(futureEvent)) AS myGroup
7 OPTIONAL MATCH (member)-[:INTERESTED_IN]->()-[:HAS_TOPIC]->()-[:HOSTED_EVENT]->(futureEvent)
8
9 WITH member, futureEvent, myGroup, COUNT(*) AS commonTopics
10 WHERE commonTopics >= 3
11 OPTIONAL MATCH (member)-[rsvp:RSVPD]->(previousEvent)-[:HOSTED_EVENT]->()-[:HOSTED_EVENT]->(futureEvent)
12 WHERE previousEvent.time < timestamp()
13
14 WITH member, futureEvent, commonTopics, myGroup, COUNT(rsvp) AS previousEvents
15
16 OPTIONAL MATCH (member)-[:FRIENDS]->(:Member)-[rsvpFriend:RSVPD]->(futureEvent)
17 WITH member, futureEvent, commonTopics, myGroup, previousEvents, COUNT(rsvpFriend) AS friendsGoing
18
19 MATCH (venue)-[:VENUE]->(futureEvent)-[:HOSTED_EVENT]->(group)
20
21 WITH member, futureEvent, group, venue, commonTopics, myGroup, previousEvents, friendsGoing, distance(point(venue), point({training-location})) AS distance
22 OPTIONAL MATCH (member)-[rsvp:RSVPD]->(previousEvent)-[:VENUE]->(aVenue)
23 WHERE previousEvent.time < timestamp() AND abs(distance(point(venue), point(aVenue))) < 500
24
25 WITH futureEvent, group, venue, commonTopics, myGroup, previousEvents, friendsGoing, distance, COUNT(previousEvent) AS eventsAtVenue
26 WITH futureEvent, group, venue, commonTopics, myGroup, previousEvents, friendsGoing, distance, eventsAtVenue, CASE WHEN myGroup THEN 5 ELSE 0 END AS myGroupScore
27 WITH futureEvent, group, venue, commonTopics, myGroup, previousEvents, friendsGoing, distance, eventsAtVenue, myGroupScore, round((futureEvent.time - timestamp()) / (24.0*60*60*1000)) AS days
28
29 RETURN futureEvent.name, futureEvent.time, group.name, venue.name, commonTopics, myGroup, previousEvents, friendsGoing, days, distance, eventsAtVenue, myGroupScore + commonTopics + eventsAtVenue + (friendsGoing / 2) - days AS score
30 ORDER BY score DESC
```

Meetup活动推荐 – 推理和发现(续)



- 对结果进行打分
 - 运用Pareto规则对推荐进行打分
 - 配置权重
 - 调整不同推荐结果的重要性
 - 计算的总分
 - 按照总分进行排序



```
1 // ... ..
2
3 CALL apoc.scoring.existence(5, myGroup) YIELD value AS myGroupScore
4 CALL apoc.scoring.pareto(1, 3, 10, days) YIELD value AS daysScore
5 CALL apoc.scoring.pareto(1, 5, 10, commonTopics) YIELD value AS topicsScore
6 CALL apoc.scoring.pareto(1, 7, 20, eventsAtVenue) YIELD value AS eventsAtVenueScore
7 CALL apoc.scoring.pareto(1, 5, 20, friendsGoing) YIELD value AS friendsGoingScore
8
9 RETURN futureEvent.name, futureEvent.time, group.name, venue.name, commonTopics, myGroup, previousEvents, friendsGoing, friends[..5],
10 days, distance, eventsAtVenue, myGroupScore + topicsScore + eventsAtVenueScore + friendsGoingScore - daysScore AS score
11 ORDER BY score DESC
```

Meetup活动推荐 – 进一步的改进



- 基于协同的规则
 - 共同参加的活动，余弦相似度，Jaccard相似度, Dice相似度, ...
- 基于内容的规则
 - 属于同一领域的其他主题和活动，例如“关系数据库”兴趣组，可以推荐“建模”、“SQL”、“JDBC”
- 特定规则
 - 最多人参加的活动、新活动、特邀嘉宾出席的活动等
- 综合各种规则的系统



Meetup活动推荐 – 进一步的改进

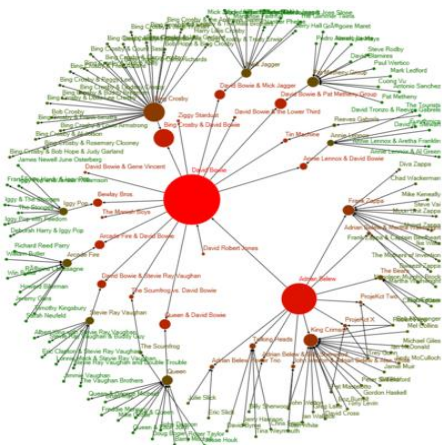
图论算法



中心性

决定节点在一个网络中的重要性。

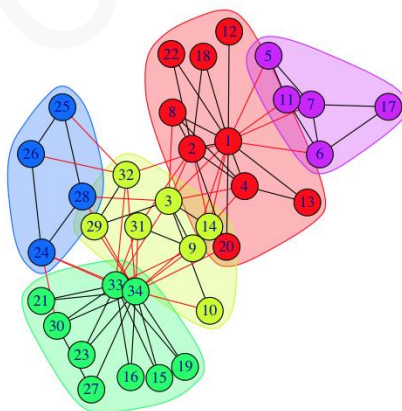
- 页面排行
- 间接中心性
- 紧密中心性
- 调和中心性



社区检测

对一个网络中节点自动进行分类、分区，决定网络的集群。

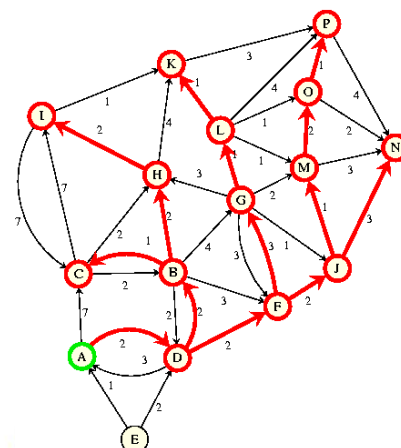
- Louvain方法
- 标签传播



路径寻找

寻找最短路径、生成子图。

- 最小权重生成子树
- 最短路径

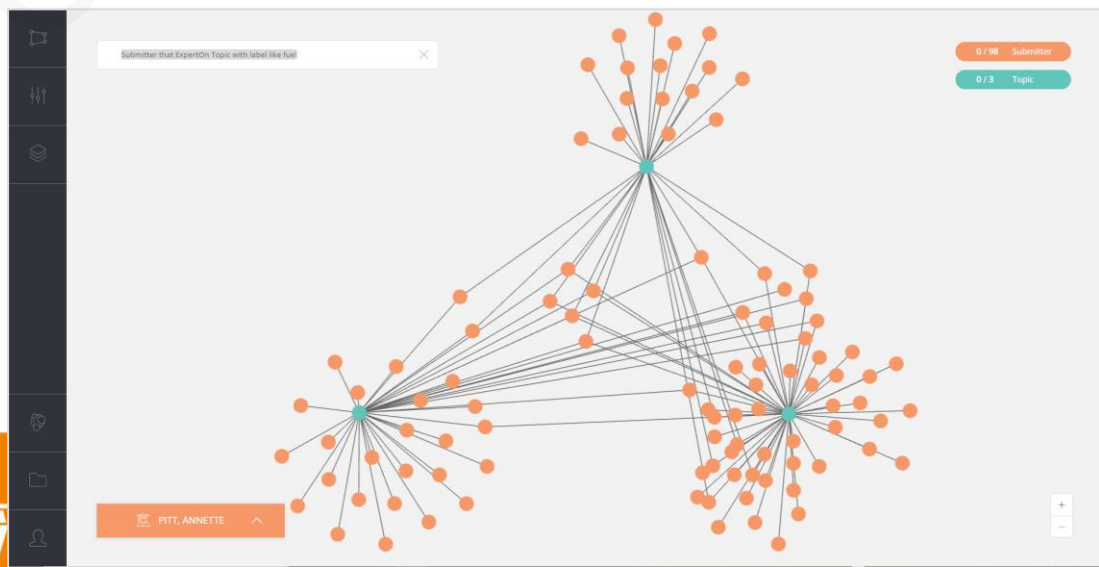


Meetup活动推荐 – 进一步的改进(续)



- 寻找兴趣相同的用户
 - 社区检测算法
 - 计算兴趣组的相似度
 - 除了“朋友”关系，还可以推荐“同类人”参加的活动

```
1 CALL apoc.algo.community
2   (25,null,'partition','X','OUTGOING','weight',10000)
```

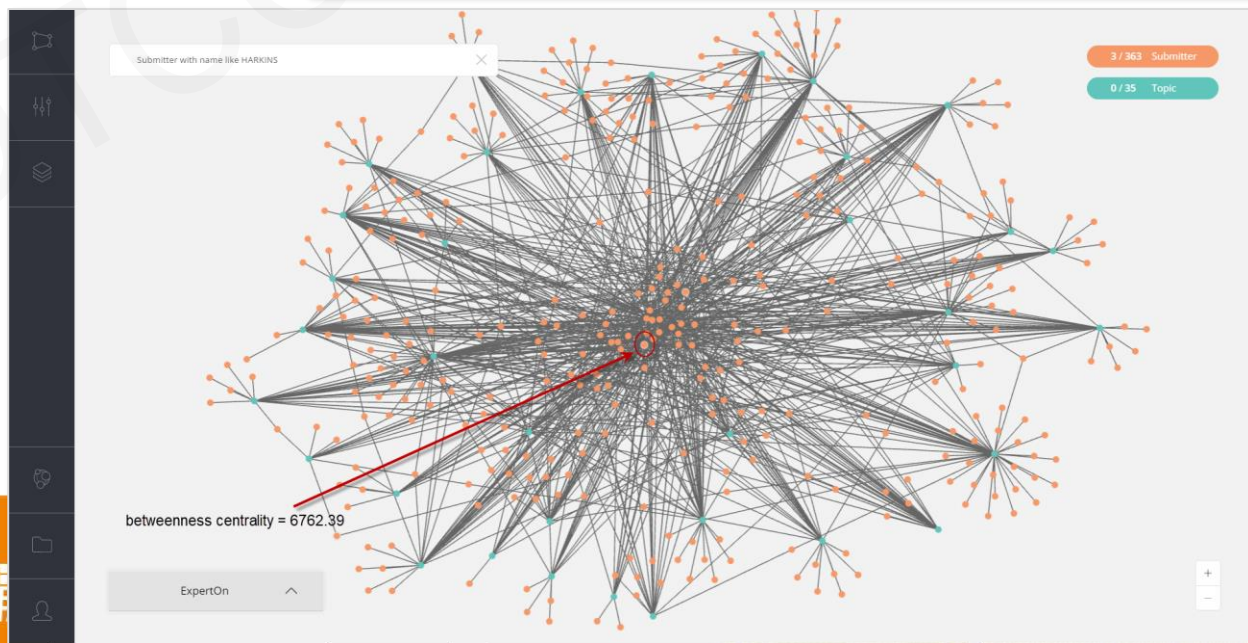


Meetup活动推荐 – 进一步的改进(续)



- 发现“公众人物” / “领域专家”
 - 运用中心性算法寻找“公众人物”
 - 推荐公众人物的选择
 - 专家对活动的评价

```
1 MATCH (node:User)
2 WITH collect(node) AS nodes
3 CALL apoc.algo.betweenness(['RSVP'], nodes, 'BOTH')
4   YIELD node, score
5 RETURN id(node), node.name, score
6 ORDER BY score DESC
```

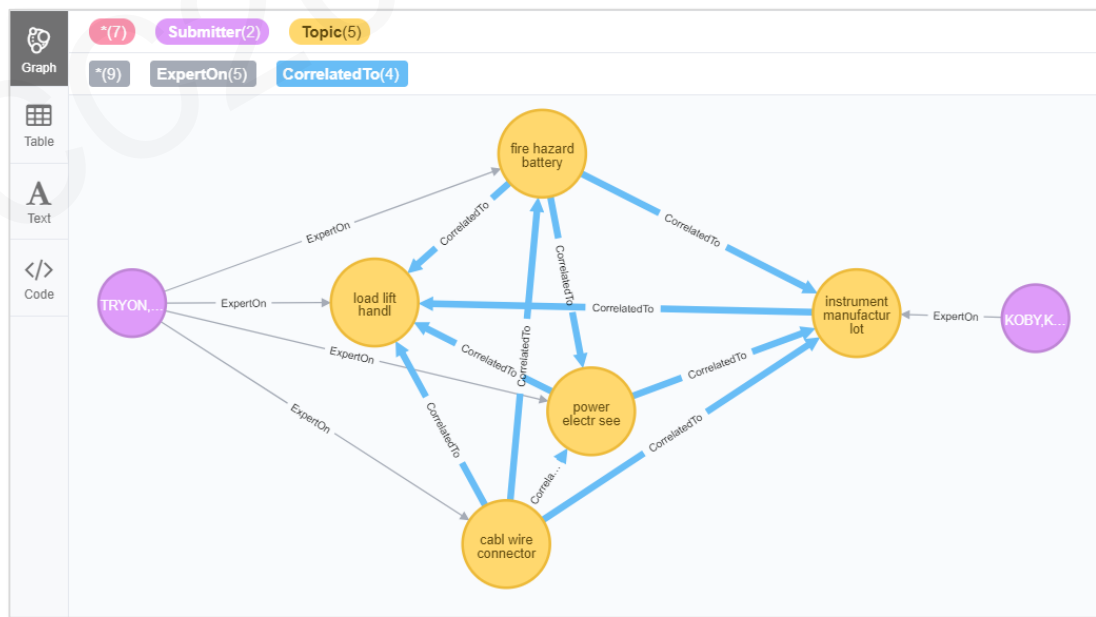


Meetup活动推荐 – 进一步的改进(续)



- 最短路径算法
 - 推荐新的朋友
 - 发现共同的兴趣
 - 优化活动安排的时间和地点

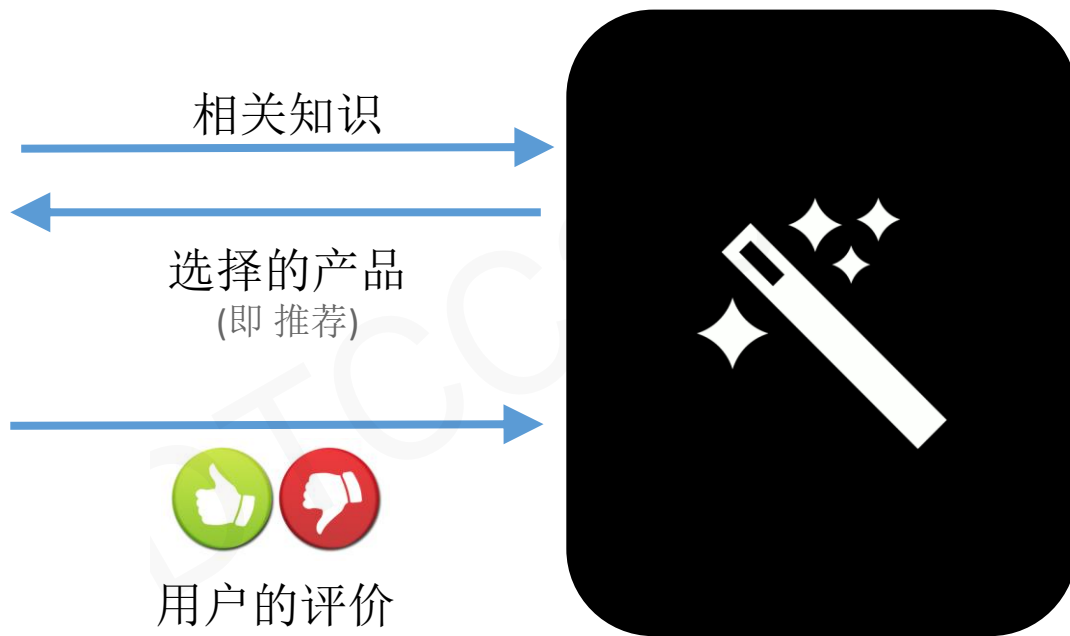
```
1 MATCH (s1:User{name:'KOBY,KEN'}),  
2     (s2:User{name:'TRYON,ROY'}),  
3 p = allShortestPaths((s1) -[*]- (s2))  
4 RETURN p;
```



Meetup活动推荐 – 进一步的改进(续)





- 将用户的反馈也考虑进来



Meetup活动推荐 – 进一步的改进(续)



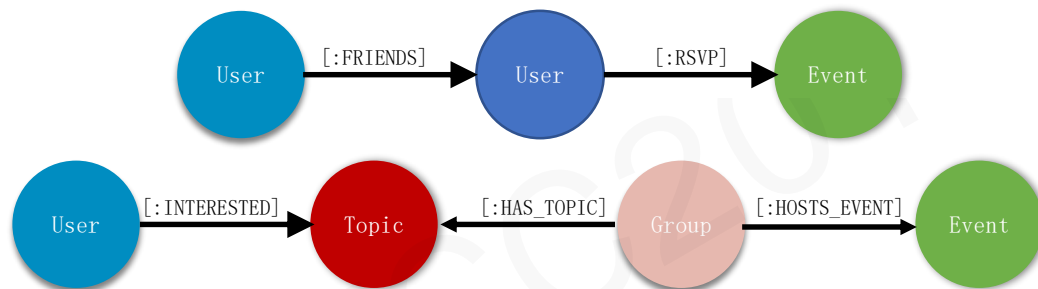
- 分析用户反馈
 - 明确的  
 - 隐含的
 - 浏览过的活动
 - 浏览活动的详细内容
 - 添加活动到“收藏”、“关注”
 - 参加过的活动
 - 从头到尾阅读相关文章
- 从初始的推荐列表中过滤掉那些产生过负面反馈的项目



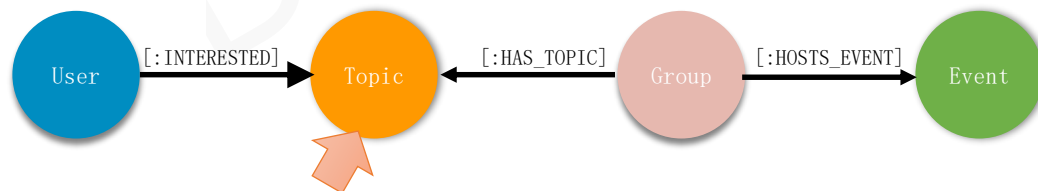
Meetup活动推荐 – 进一步的改进(续)



- 基于实际数据进行训练:
 - 用户接受度：因果关系



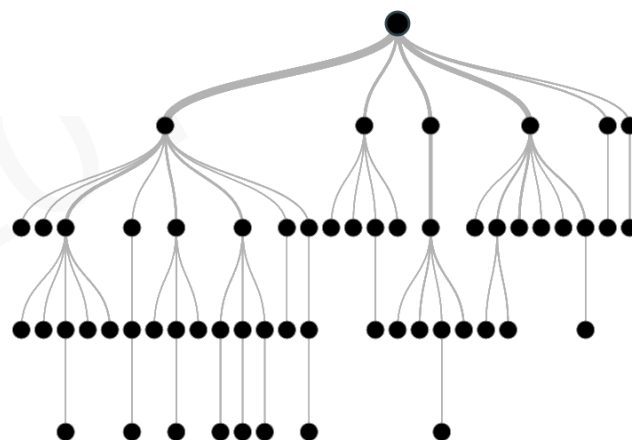
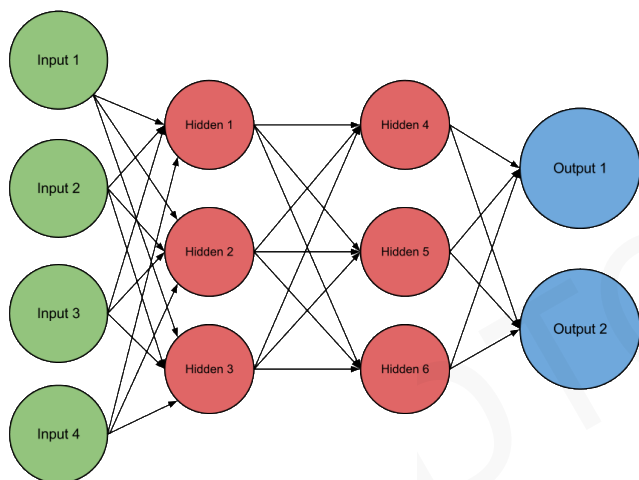
- 用户接受度：在关键路径上出现的节点



机器学习和图数据库



几乎所有的人工智能算法都是“图算法”。

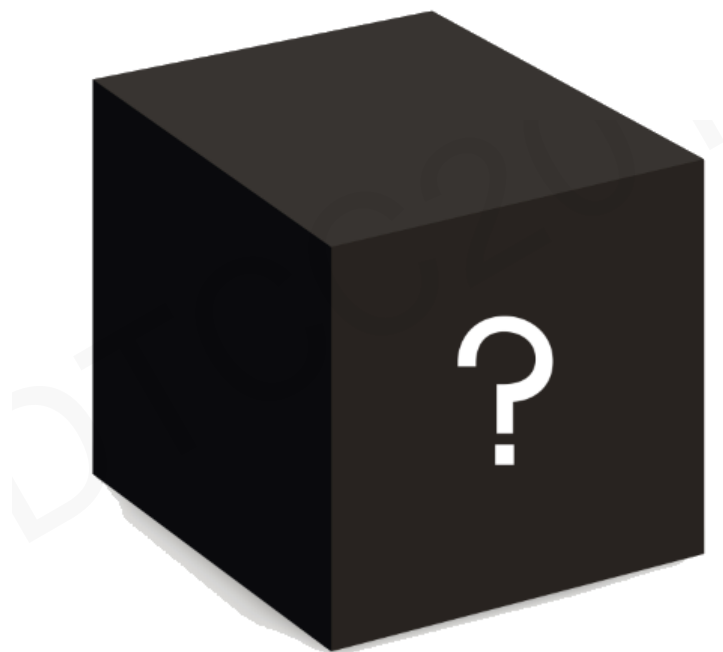


那为什么以前在机器学习中没有见到过图数据库？

机器学习和图数据库(续)

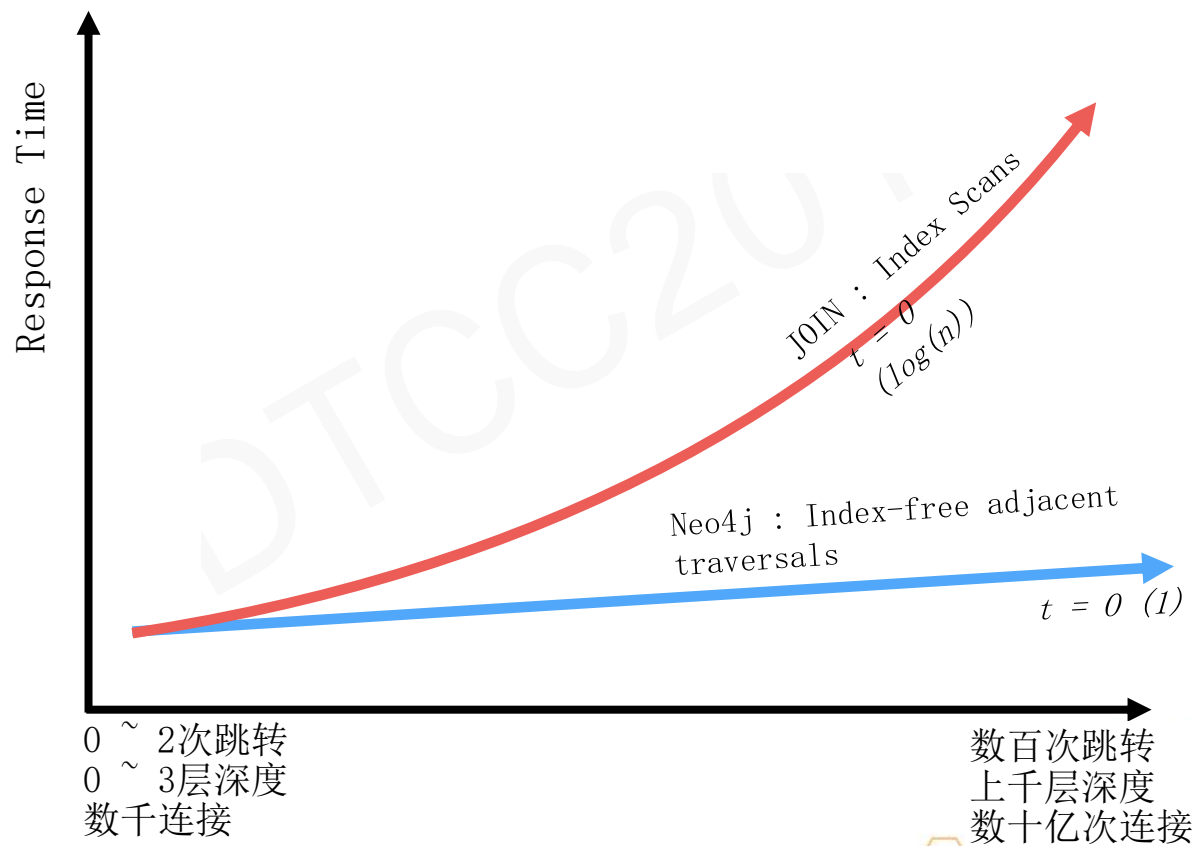


以前的机器学习应用中，数据模型是个黑盒子。

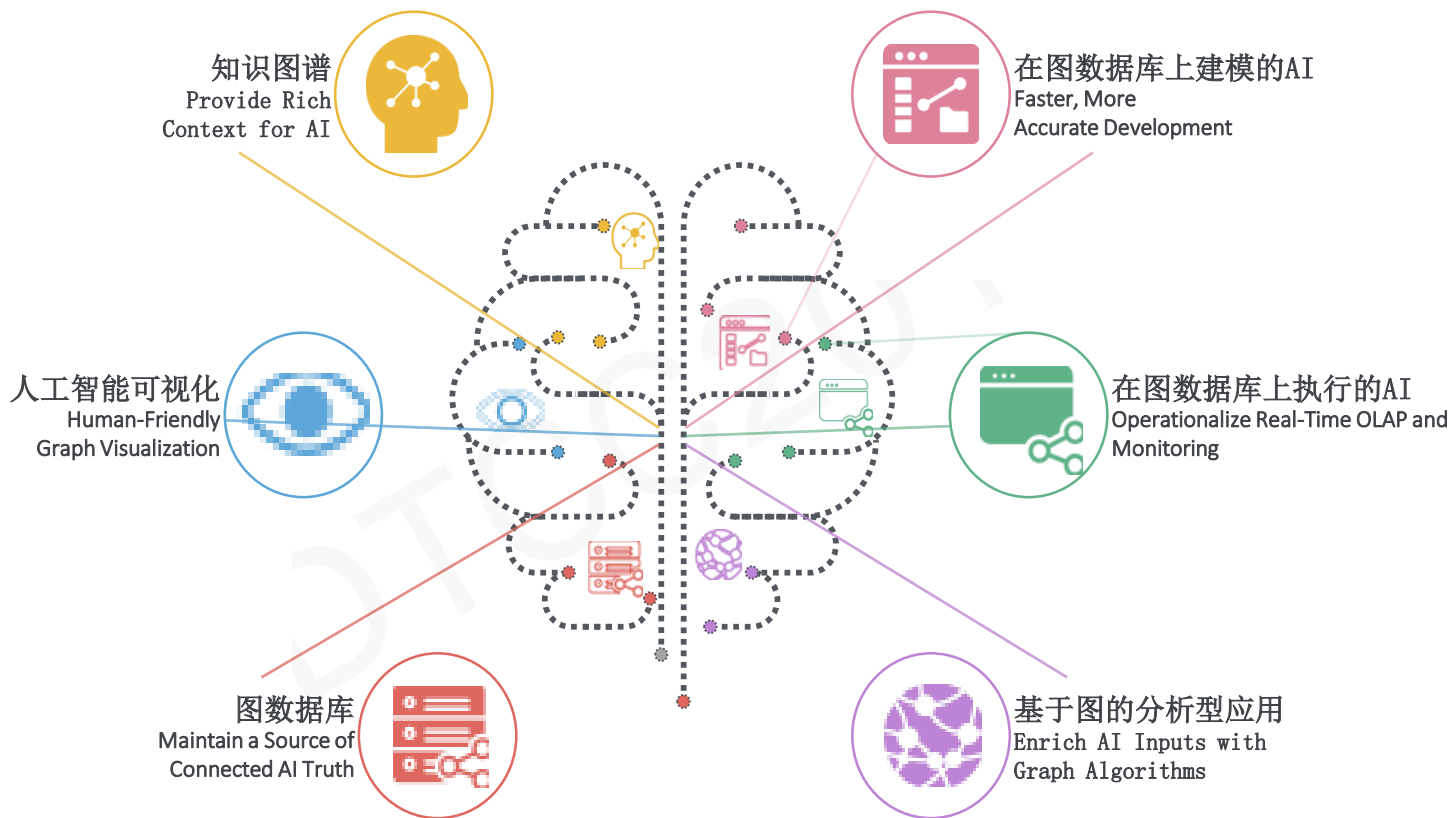


机器学习和图数据库(续)

在大数据时代，数据库的类型将决定性能，而性能对实时的推荐引擎至关重要。



图数据库技术将极大增强AI类应用



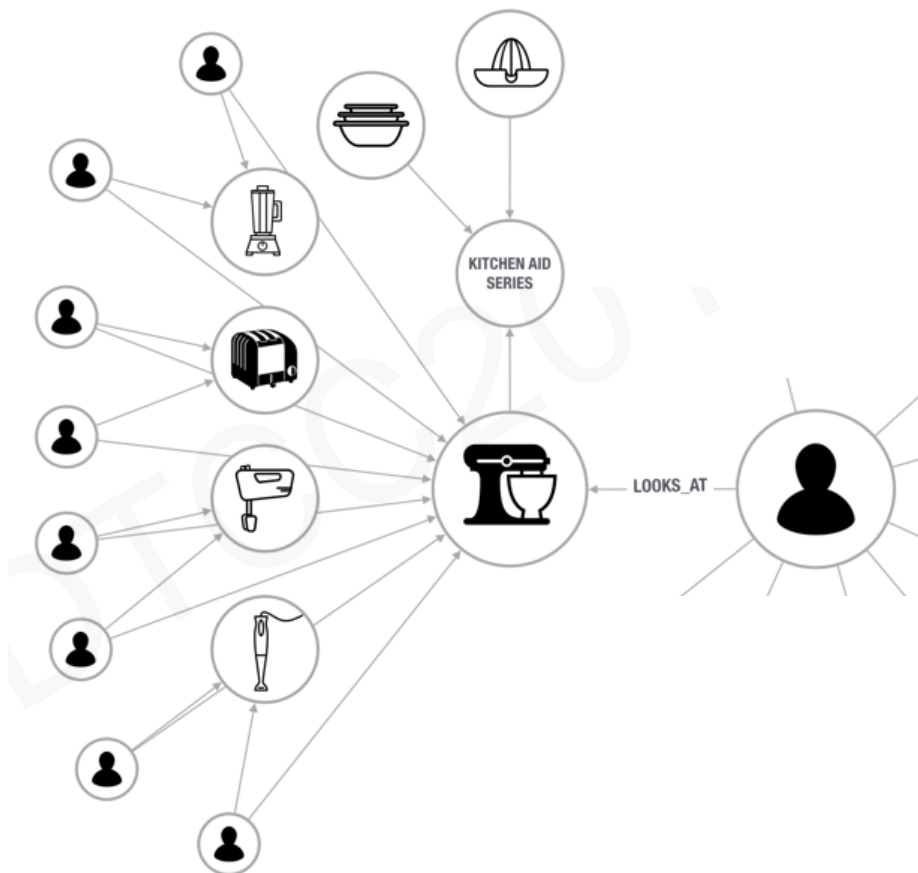
总结



- 基于图数据库实现机器学习
 - 数据模型的一致性
 - 性能优势，因为无须做连接(JOIN)
 - 丰富的语义和关系
 - 图论算法
 - 模型的灵活性

结合图数据库和机器学习，使得实现**即时、精准、相关性高**的推荐引擎成为可能。

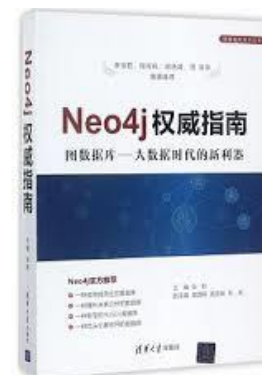
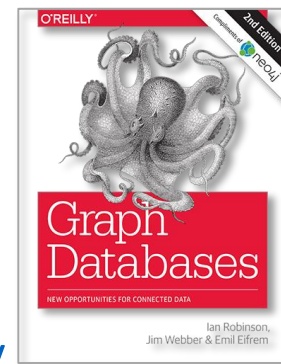
展望：建立在对产品和客户的全面视图之上的即时推荐引擎



如需了解更多，请访问以下免费资源



- Website: <http://neo4j.com>
- 下载桌面版: <https://neo4j.com/download/>
- 在线数据库沙箱: <https://neo4j.com/sandbox-v2/>
- 技术问题(英文): <http://stackoverflow.com>
- Github开源项目和代码库: <https://github.com/neo4j-contrib/>
- 中文社区: <http://neo4j.com.cn>
- QQ 群: Neo4j 中文社区 / 547190638
- 图数据库(电子书): <https://neo4j.com/graph-databases-bo>



电子邮件: apac@neo4j.com (亚太地区)



THANKS





讲师申请

联系电话（微信号）：18612470168

关注“ITPUB”更多
技术干货等你来拿~

与百度外卖、京东、魅族等先后合作系列分享活动



让学习更简单

微学堂是以ChinaUnix、ITPUB所组建的微信群为载体，定期邀请嘉宾对热点话题、技术难题、新产品发布等进行移动端的在线直播活动。

截至目前，累计举办活动期数60+，参与人次40000+。

ITPUB学院

ITPUB学院是盛拓传媒IT168企业事业部（ITPUB）旗下
企业级在线学习咨询平台
历经18年技术社区平台发展
汇聚5000万技术用户
紧随企业一线IT技术需求
打造全方式技术培训与技术咨询服务
提供包括企业应用方案培训咨询（包括企业内训）
个人实战技能培训（包括认证培训）
在内的全方位IT技术培训咨询服务

ITPUB学院讲师均来自于企业
一些工程师、架构师、技术经理和CTO
大会演讲专家1800+
社区版主和博客专家500+

培训特色

无限次免费播放
随时随地在线观看
碎片化时间集中学习
聚焦知识点详细解读
讲师在线答疑
强大的技术人脉圈

八大课程体系

基础架构设计与建设
大数据平台
应用架构设计与开发
系统运维与数据库
传统企业数字化转型
人工智能
区块链
移动开发与SEO



联系我们

联系人：黄老师
电话：010-59127187
邮箱：edu@itpub.net
网址：edu.itpub.net
培训微信号：18500940168