ArchSummit

全球架构师峰会（深圳）2014

移动环境下的内容推荐初探

姚从磊, 豌豆荚

Agenda

- 什么是推荐系统
- 移动内容推荐的特点
- 豌豆荚移动内容推荐关键技术
- 总结

什么是推荐系统

Recommender systems or recommendation systems (sometimes replacing "system" with a synonym such as platform or engine) are a subclass of information filtering system that seek to predict the 'rating' or 'preference' that user would give to an item.

- from wikipedia

Recommender systems/engines are a subclass of information retrieval systems/engines in passive mode.

Search systems/engines are a subclass of information retrieval systems/engines in active mode.

常用的推荐功能

- 有哪些值得看的电影？
- 去哪儿下馆子腐败？
- 淘点啥宝贝？
- 装什么有意思的应用？
- 大家都在讨论什么话题？
- Netflix Prize $1 Million Challenge?

豆瓣电影 movie.douban.com

大众点评 dianping.com

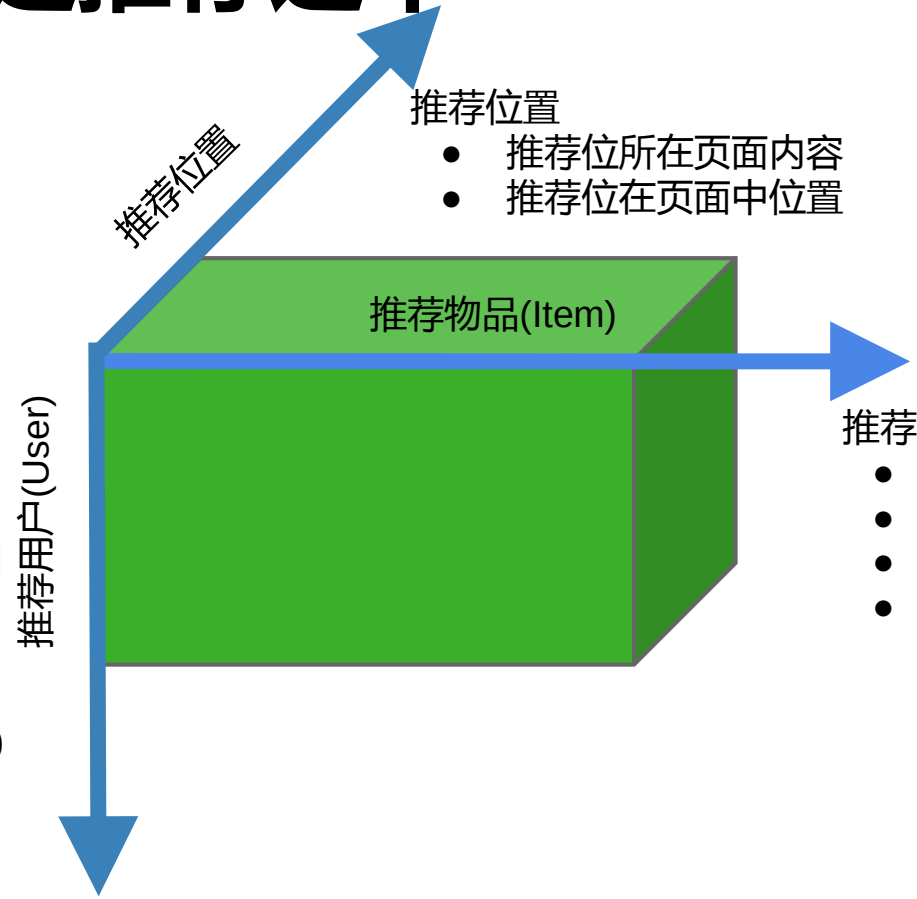
淘宝网 Taobao.com

新浪微博

推荐技术应用场景

应用场景	排行榜推荐	个性化推荐	使用海量数据
电影推荐	多	多	是
餐馆推荐	多	少	是
商品推荐	多	多	是
应用推荐	多	多	是
话题推荐	多	少	是
广告推荐	少	多	是

海量数据是推荐之本

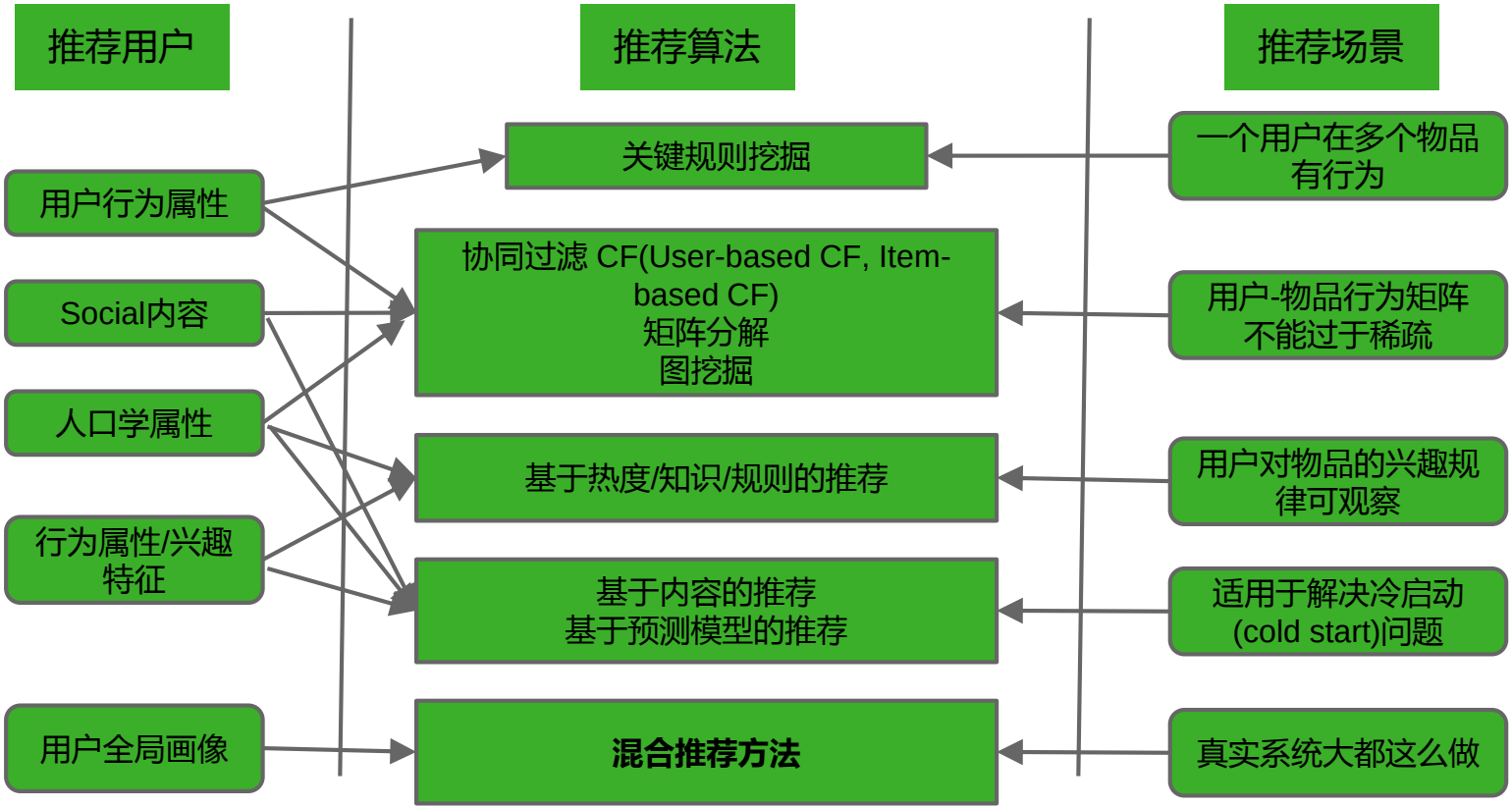


推荐位置
- 推荐位所在页面内容
- 推荐位在页面中位置

推荐用户
- 人口学属性(性别、年龄等)
- 行为属性/兴趣特征(浏览、搜索、缓存)
- Social内容(社交关系、小组发帖)
- 上下文(时间、地点等)
- ……

推荐物品
- 种类、主题、关键词
- 生命周期、投放规则
- 大小、颜色、价格
- ……

推荐算法

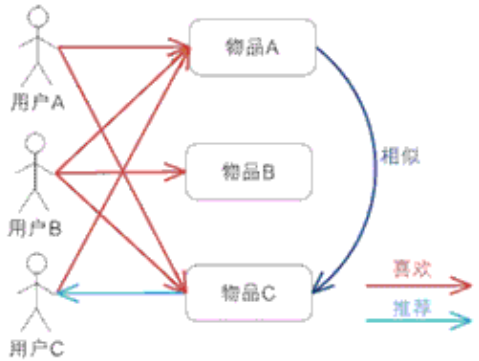

推荐用户
推荐算法
推荐场景
用户行为属性
Social内容
人口学属性
行为属性/兴趣特征
用户全局画像
关键规则挖掘
协同过滤 CF(User-based CF, Item-based CF)
矩阵分解
图挖掘
基于热度/知识/规则的推荐
基于内容的推荐
基于预测模型的推荐
混合推荐方法
一个用户在多个物品有行为
用户-物品行为矩阵不能过于稀疏
用户对物品的兴趣规律可观察
适用于解决冷启动(cold start)问题
真实系统大都这么做

常用推荐算法比较

推荐算法	优点	缺点	示例
基于知识的推荐算法(基于人工观察和经验，抽象出规则进行推荐)	1. 方法简单 2. 规则易解释、易运营 3. 所需数据少	1. 规则来源于人工知识，覆盖率差 2. 有效周期难以判断	好友推荐
基于内容的推荐算法	1. 可以解决冷启动问题 2. 方便整合各种用户/物品特征 3. 推荐结果易于理解	1. 受限于特征提取的准确性 2. 需要大量的支撑数据 3. 常局限于特征稀疏的情况	Google News Amazon 淘宝 豌豆荚
协同过滤的推荐算法	1. 原理简单有效，易于理解 2. 算法灵活，支持海量用户/物品 3. 无需考虑复杂特征，支持特征空间极其复杂的用户/物品的高效计算	1. 很明显的冷启动问题 2. 常受限于用户/物品行为矩阵稀疏的情况 3. 受限于用户/物品rating的正确性	Netfix 淘宝 Amazon 豆瓣 豌豆荚

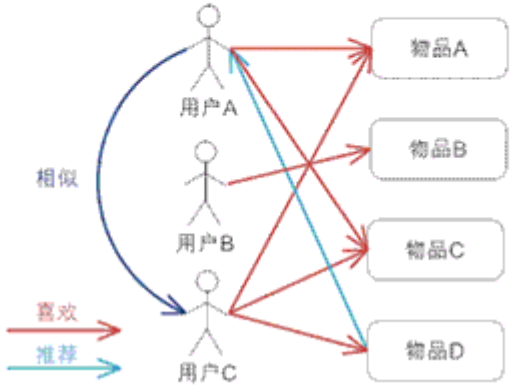
协同过滤-User-based CF

用户/物品	物品A	物品B	物品C
用户A	√		√
用户B	√	√	√
用户C	√		推荐



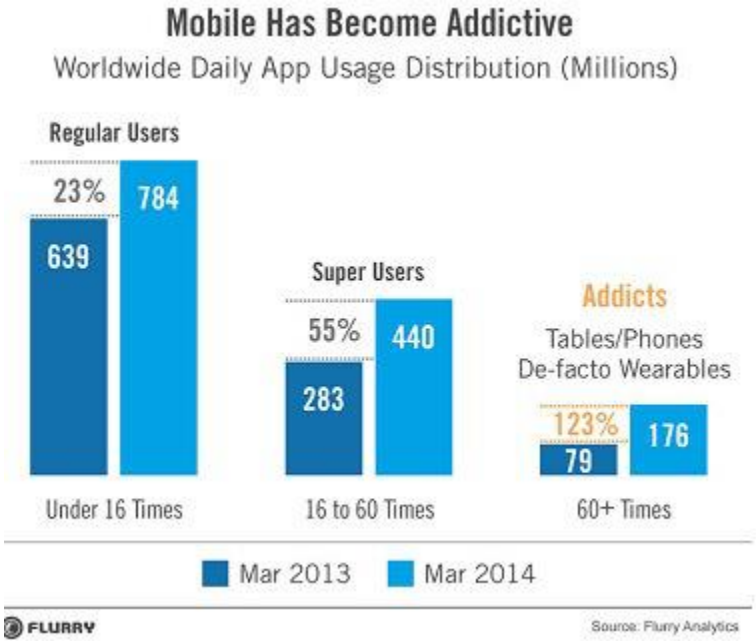
协同过滤-Item-based CF

用户/物品	物品A	物品B	物品C	物品D
用户A	√		√	推荐
用户B		√		
用户C	√		√	√

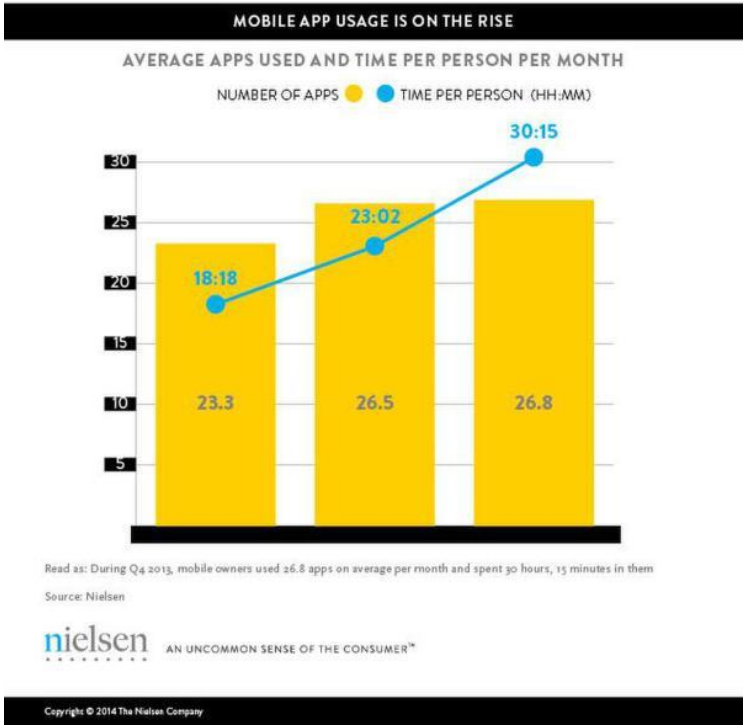


移动推荐的特点

使用次数多


Mobile Has Become Addictive
Worldwide Daily App Usage Distribution (Millions)
Regular Users
23%
784
639
Under 16 Times
Super Users
55%
440
283
16 to 60 Times
Addicts
Tables/Phones
De-facto Wearables
123%
176
79
60+ Times
Mar 2013
Mar 2014
FLURRY
Source: Flurry Analytics

时间碎片化


MOBILE APP USAGE IS ON THE RISE
AVERAGE APPS USED AND TIME PER PERSON PER MONTH
NUMBER OF APPS
TIME PER PERSON (HH:MM)
30
25
20
15
10
5
18:18
23:02
30:15
23.3
26.5
26.8
Read as: During Q4 2013, mobile owners used 26.8 apps on average per month and spent 30 hours, 15 minutes in them
Source: Nielsen
nielsen
AN UNCOMMON SENSE OF THE CONSUMER™
Copyright © 2014 The Nielsen Company

内容消费占大头

用户易定位(用户<->设备)

移动推荐的新特点

特点	原因
实时推荐非常重要	移动设备/应用使用次数多
推荐的准确率要求更高	移动设备/应用使用时间碎片化
内容推荐是关键	移动设备/应用消费中内容是大头
可利用用户信息更丰富	移动设备可以用来准确定位用户

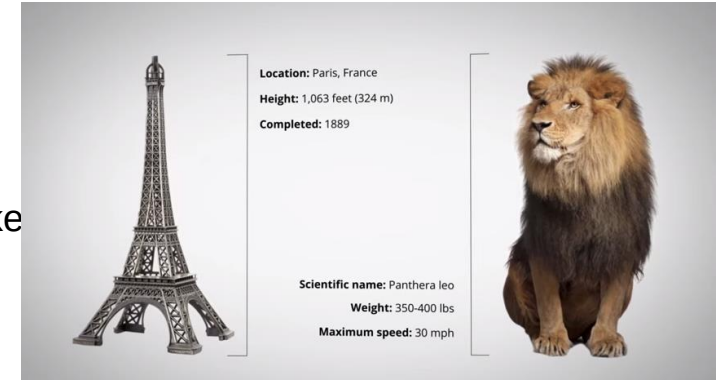
豌豆荚移动内容推荐关键技术

豌豆荚移动内容推荐框架

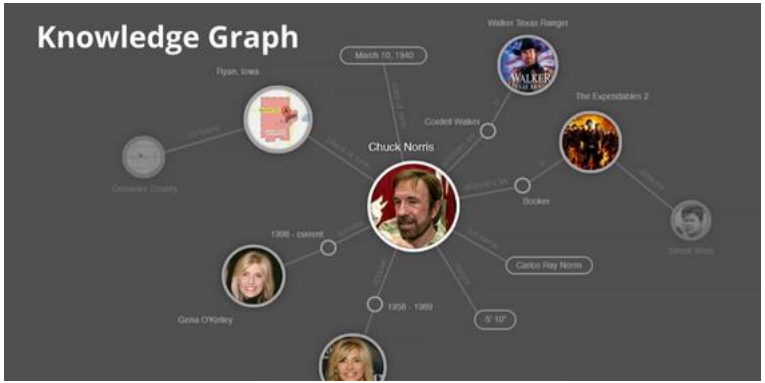
关键技术之
知识图谱

知识图谱 - Knowledge Graph

Knowledge Graph understands real-world entities and their relationships to one another: things, not strings, helps the machine to understand the world a bit more like people do.



Knowledge Graph consists of nodes with edges between them. Each node corresponds to an entity in real work, each edge describe the relationship between the begin and end nodes.



Google Knowledge Graph

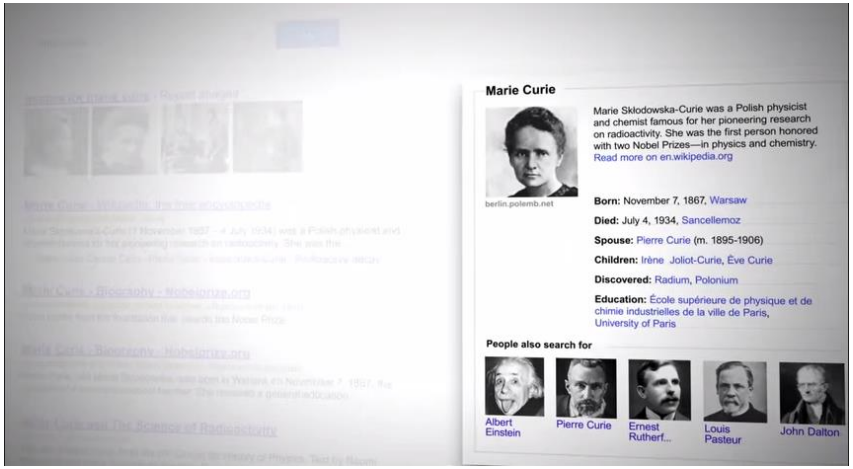
Data Sources

- Public Sources(Wikipedia, Freebase, CIA World Factbook)
- Knowledge Mining from Web pages(Lists, Tables, etc.)
- Knowledge Mining from user queries(xx attributes of yy entity)

Google Knowledge Graph

Usages

- Find the right thing
  - Recognize the entities in queries and associates them with the relevant information, and disambiguate entities using context information.
- Get the best summary
  - Use the users search behavior to determine the most needed aspects for each entity.
- Go deeper and broader
  - Help users to get some unexpected discoveries.


Marie Curie
Marie Skłodowska-Curie was a Polish physicist and chemist famous for her pioneering research on radioactivity. She was the first person honored with two Nobel Prizes—in physics and chemistry.
Read more on en.wikipedia.org
berlin.polemb.net
Born: November 7, 1867, Warsaw
Died: July 4, 1934, Sancellemoz
Spouse: Pierre Curie (m. 1895-1906)
Children: Irène Joliot-Curie, Ève Curie
Discovered: Radium, Polonium
Education: École supérieure de physique et de chimie industrielles de la ville de Paris, University of Paris
People also search for
Albert Einstein
Pierre Curie
Ernest Rutherf...
Louis Pasteur
John Dalton


NOBEL PRIZE


THE RENAISSANCE


Most popular Dubai destinations
Cities in UAE
Skyscrapers in Dubai

豌豆荚知识图谱

数据来源

- 公开来源(Wikipedia, Freebase, IMDB、豆瓣电影)
- 应用内内容/网页信息中解析出来的实体和实体关系信息(应用，游戏，书，视频，音乐等)
- 豌豆荚用户贡献的User profile信息和UGC内容
- 用户-实体的关系(搜索、浏览、消费、评价等)

豌豆荚知识图谱

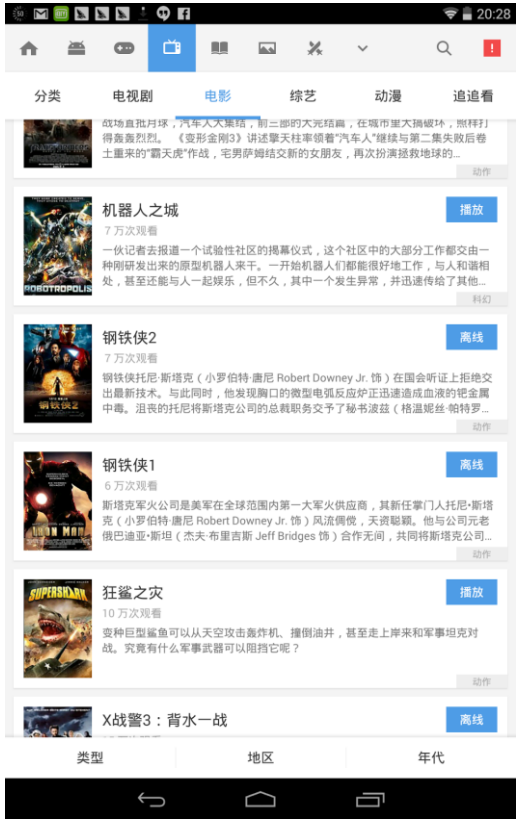
数据来源

- 公开来源(Wikipedia, Freebase, IMDB、豆瓣电影)
- **应用内内容**/网页信息中解析出来的实体和实体关系信息(应用，游戏，书，视频，音乐等)
- **豌豆荚用户贡献的User profile信息和UGC内容**
- **用户-实体的关系(搜索、浏览、消费、评价等)**

豌豆荚知识图谱

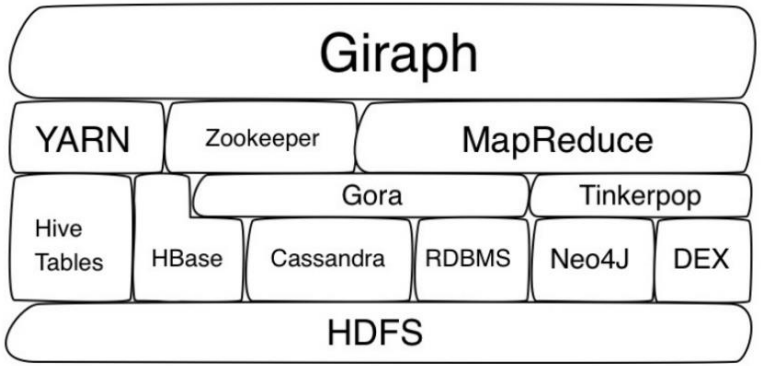
用途

- 信息展示
  - 全面展示内容相关的实体及相互关系
- 用户意图理解
  - 重名区分
  - 查询意图识别
- **内容推荐**
  - 豌豆荚首页内容推荐 - 追新有趣
  - 豌豆荚详情页内容推荐 - 相关而非相似
  - 豌豆荚各频道列表页内容推荐 - 追热有趣
  - ……



关键技术之
Giraph-图挖掘平台

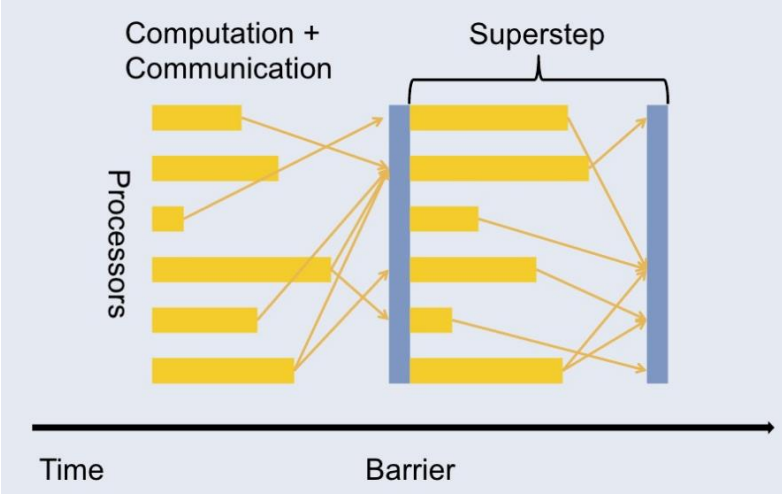
Giraph - 图挖掘平台

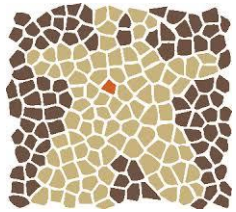


Giraph 是Google Pregel的开源实现，它基于Hadoop，通过在Hadoop上运行map-only的job来实现常见的图挖掘算法。

Giraph 提供了一个简单的“面向节点”(Think like a vertex)的编程模型。所有的图挖掘算法都可简化为在每一个Vertex上的两类操作：1) 在vertext上进行数值计算(computation)；2) 将数值计算的结果通过边传递给相邻的全部/部分节点(communication)。

上述编程模型极大简化了图挖掘算法的实现，使得每个算法都可以迭代地实现，这一切是建立在Giraph内嵌的bulk synchronous parallel编程模型基础上。





Giraph Timeline

- 受Google Pregel启发开发 (2010)
- 被Yahoo!捐献给ASF (2011)
- 成为Top-level Apache Project (2012)
- 发布1.0版本 (2013)
- 发布1.1版本 (2014)

Giraph典型应用

- Ranking
  - Popularity, Importance, etc.
- Label Propagation
  - Location, School, Interest, gender, etc.
- Community
  - Groups, Clusters

Map-only Map-Reduce

- Think like a key-value pair

```
public class Mapper<
      KEYIN
      VALUEIN
      KEYOUT
      VALUEOUT>{
   void map(KEYIN key,
      VALUEIN value,
      Context context)
      throws IOException,
      InterruptedException;
}
```

Giraph

- Think like a vertex

```
public class Vertext<
      I extends
         WritableComparable,
      V extends Writable,
      E extends Writable,
      M extends Writable>{
   void compute(
      Iterator<M> msgIterator);
}
```

Giraph API(当前Vertex上可用)

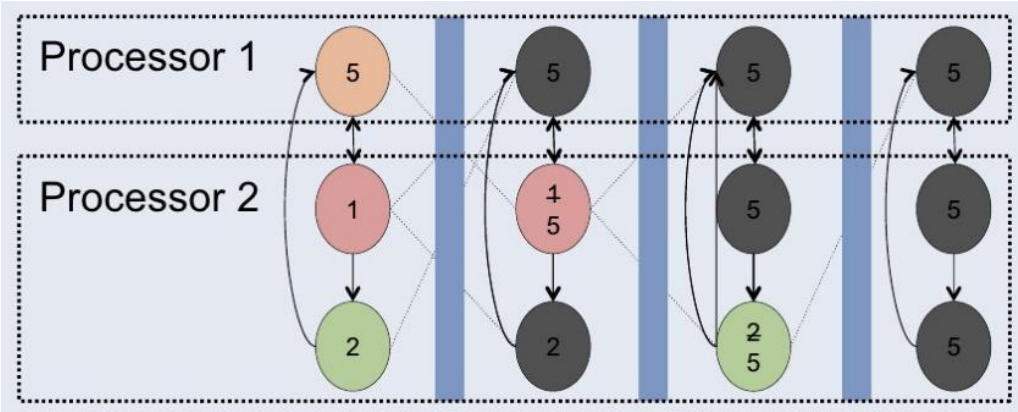
I getVertextId()
V getVertextValue()
void setVertextValue(V vertextValue)
Iterator<I> iterator()
E getEdgeValue(I targetVertextId)
boolean hasEdge(I targetVertextId)
boolean addEdge(I targetVertextId, E edgeValue)
E removeEdge(I targetVertextId)
void voteToHalt()
boolean isHalted()

Giraph API(Vertex间传递消息可用)

void sendMsg(I targetVertexid, M msg)
void sendMsgToAllEdges(M msg)
void addVertextRequest(BasicVertex(I, V, E, M> vertex)
void removeVertextRequest(I vertexId)
void addEdgeRequest(I sourceVertexId, Edge<I, E> edge)
void removeEdgeRequest(I sourceVertextId, I destVertextId)

一个简单的例子 - 计算最大节点值

```
public class MaxValueVertex extends EdgeListVertex<
    IntWritable, IntWritable, IntWritable, IntWritable>{
    @override
     public void compute(Iterator<IntWritable> msgIterator){
       boolean changed = false;
       while(msgIterator.hasNext()){
          IntWritable msgValue = msgIterator.next();
          if(msgValue.get() > getVertexValue().get())}
             setVertexValue(msgValue);
             changed = true;
           }
        }
        if(getSuperStep() == 0 || changed) {
          sendMsgToAllEdges(getVertexValue());
        }  else {
           voteToHalt();
       }
   }
  }
```



复杂点的例子 - PageRank

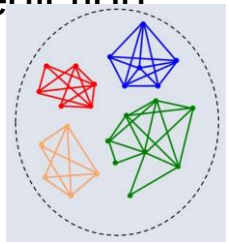
```
public class SimplePageRankVertex extends EdgeListVertex<
    LongWritable, DoubleWritable, FloatWritable, DoubleWritable>{
    @override
     public void compute(Iterator<DoubleWritable> msgIterator){
         if(getSuperStep() >= 1) {
             double sum = 0;
             while(msgIterator.hasNext()) {
                  sum += msgIterator.next().get();
             }
             setVertexValue(new DoubleWritable((0.15f /
getNumVertices()) + 0.85f * sum);
         }
         if(getSuperStep() < 300) {
            long edges = getNumOutEdges();
            sendMsgToAllEdges(new
DoubleWritable(getVertexValue.get() / edges));
         } else {
            voteToHalt();
         }
     }
}
```

Mahout (Hadoop) **854行**

豌豆荚知识图谱 + Giraph

用途

- 用户性别、地域、教育水平的prediction



- Item/User-based CF，支撑内容推荐

- 作弊用户识别, opinion leader识别

总结

豌豆荚移动内容推荐框架

豌豆荚知识图谱独有内容

- **应用内内容**中解析出来的实体和实体关系信息(应用，游戏，书，视频，音乐等)
- **豌豆荚用户贡献的User profile信息和UGC内容**
- **用户-实体的关系(搜索、浏览、消费、评价等)**

豌豆荚知识图谱 + Giraph

- **用户属性prediction**
- **社区发现**
- **内容推荐**
- **spammer/opinion leader发现**

谢谢！