搜索引擎及信息检索技术

Search Engine & Information Retrieval

杨 逍

2010年12月4日

The First

- 20世纪70年代出现了最早的商业搜索引擎；

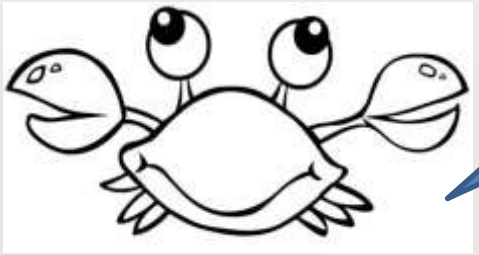
  检索方式：布尔检索

  盈利模式：按照检索次数收费

  提供内容：法律、生物等专业化知识库

  用户：律师以及科研工作人员

  负载：最多每天处理70万次查询请求



如何吃第一个螃蟹？

刀耕火种：最原始的实现方式

- 检索模型：布尔检索

接受布尔表达式进行查询，即通过AND、OR、NOT等逻辑操作符将词项连接起来的查询。比如“我爸”AND“是”AND“李刚”NOT“凤姐”

- 为什么选择布尔方式？

(1)大规模文档集条件下的快速查找，需要在几十亿到上万亿单词的数据规模下进行查找，传统匹配算法突破不了速度瓶颈；

(2)用户需要更为灵活的查询方式，比如需要查询包含在同一个句子中在5个词以内出现“爸”和“李刚”，传统的扫描匹配很难做到这种灵活的方式；

(3)需要对结果进行排序，很多情况下，用户希望在多个满足自己需求的文档中得到最佳答案；

(4)传统的串行匹配技术往往都是精确查找，我们需要模糊和包含语义信息的查询技术（用户追求的是灵活快速查询，而对模式匹配的精确位置信息并不敏感）；

刀耕火种：如何实现布尔搜索

➢数学家的做法；

	Doc1	Doc2	Doc3	Doc4	Doc5	Doc6
Antony	1	1	0	0	0	1
Brusts	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

表1 词项-文档关联矩阵

查询：Brusts AND Caesar NOT Calpurnia

分别取出Brusts、 Caesar 、 Calpurnia对应的行向量，并对Calpurnia对应的行向量求反，然后进行基于位的与操作，得到：

110100AND110111AND101111=100100

结果向量中的第1和第4个元素为1，这表明该查询对应的文档是Doc1和Doc4。

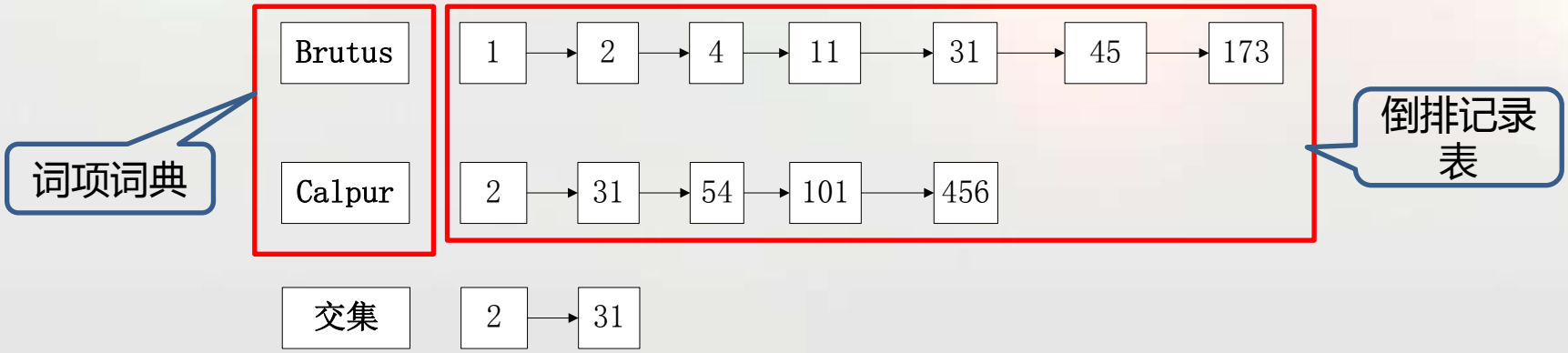
刀耕火种：如何实现布尔搜索

➢数学家做法太理想化：

例子：如果有100万个文档，共有50万个词项，那么对应的词项-文档矩阵大概有5000亿(50万X100万)个取布尔值的元素，远远大于内存容量，带来程序处理时的多维灾难，且这个庞大的矩阵具有高度的稀疏性。

➢程序员的做法：

使用倒排索引(inverted index)：



查询 Brutus AND Calpur ：

词项的倒排记录表采用单链表或者变长数组进行存储，这样就转化为求链表的交集问题；

一种线性的链表求交集算法O(N)：

```
INTERSECT(p1,p2)
1    anser= <>
2   while p1!=null and p2!=null
3   do if docID(p1)=docID(p2)
3     then ADD(anser,docID(p1))
5        p1=next(p1)
6        p2=nex(p2)
7    else if docID(p1)<docID(p2)
8          then p1=next(p1)
9          else p2=nex(p2)
10  return anser
```

两个倒排表合并

```
INTERSECT(<t1,.....tn>)
1   terms=SortByIncreasingFREQUENCY (<t1,.....tn>)
2   result=postings(first(terms))
3   terms=rest(terms)
4   while terms!+null and result!=null
5   do result = INTERSECT(result,postings(first(terms)))
6      terms=rest(terms)
7    return result
```

多个词项倒排表合并

改进倒排表的存储方式，可以通过常数时间而非线性时间实现查找：如使用B+树，Trie树，Hash B树等结构。

这就是最早的布尔搜索引擎系统的基本实现原理；

真正的搜索引擎

1.综合性搜索引擎

Baidu 百度　Google 谷歌　必应 bing　有道 youdao　SOSO 搜搜

2.垂直搜索引擎

Gougou 狗狗　SHOOTER　去哪儿? Qunar.Com 聪明你的旅行　Microsoft Travel Guide Beta 旅游指南　搜房 SouFun www.soufun.com

3.个性化搜索引擎




SE first：

➢我们先抛开“信息检索” --IR，从系统分析的角度来看搜索引擎的原理和实现；

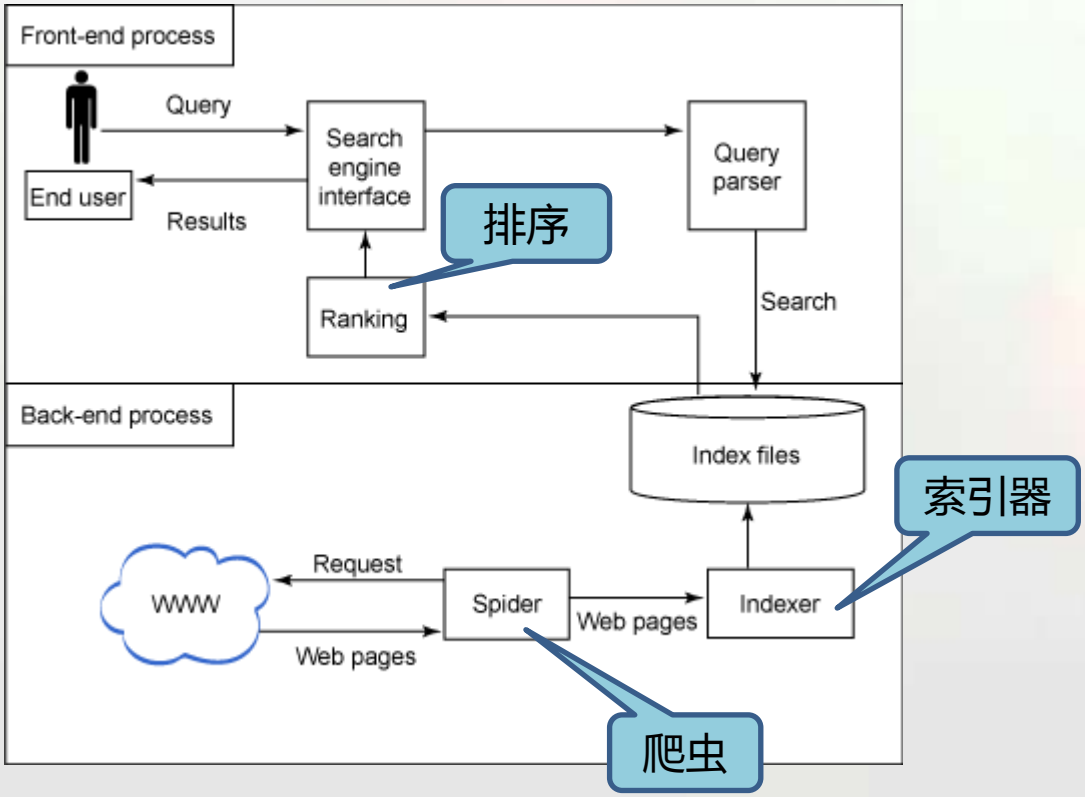


图1 经典的架构

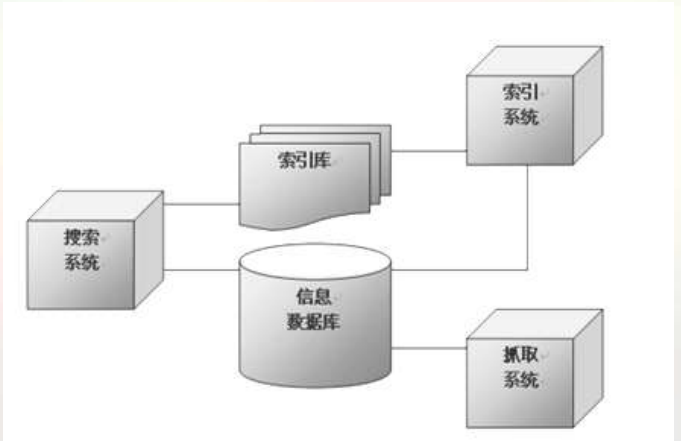


图2 模块划分

搜索引擎：

- 再复杂一点；

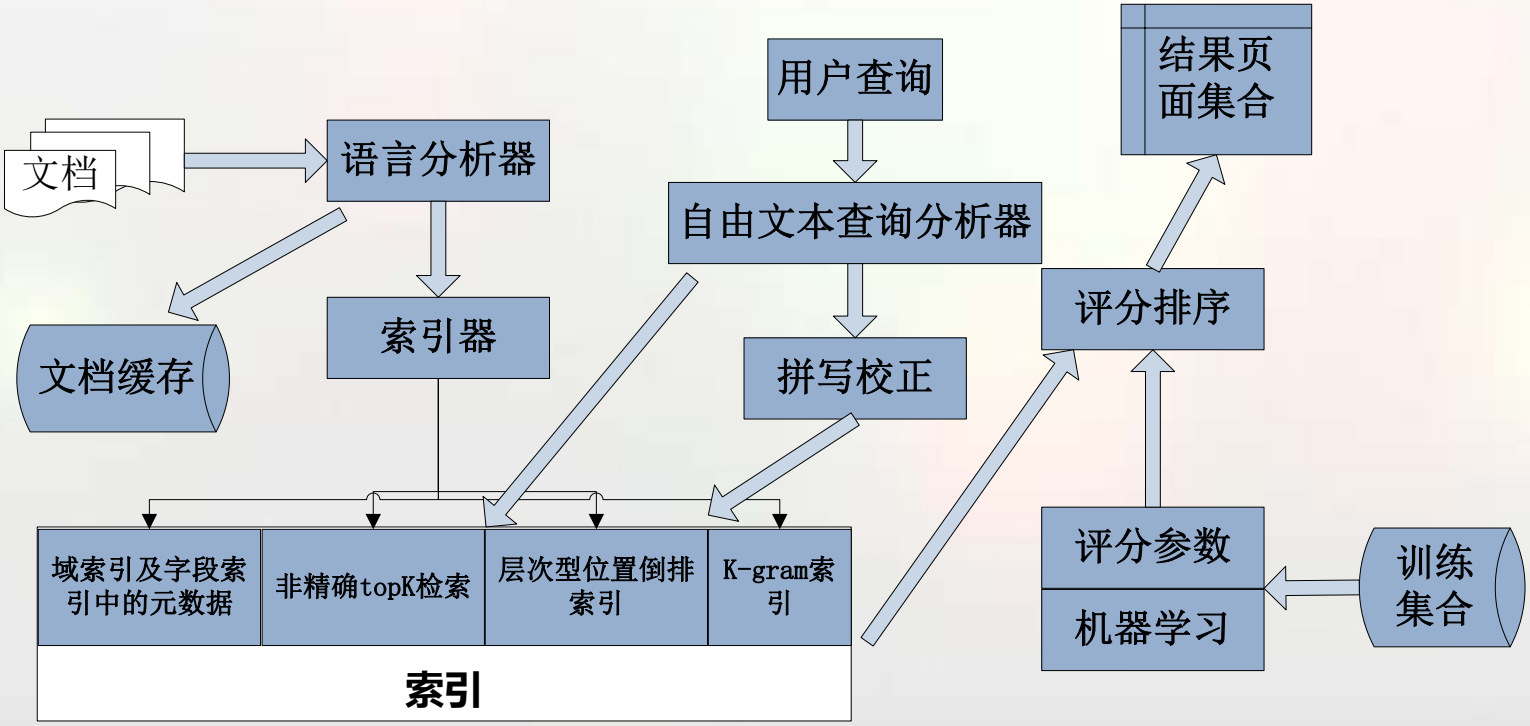


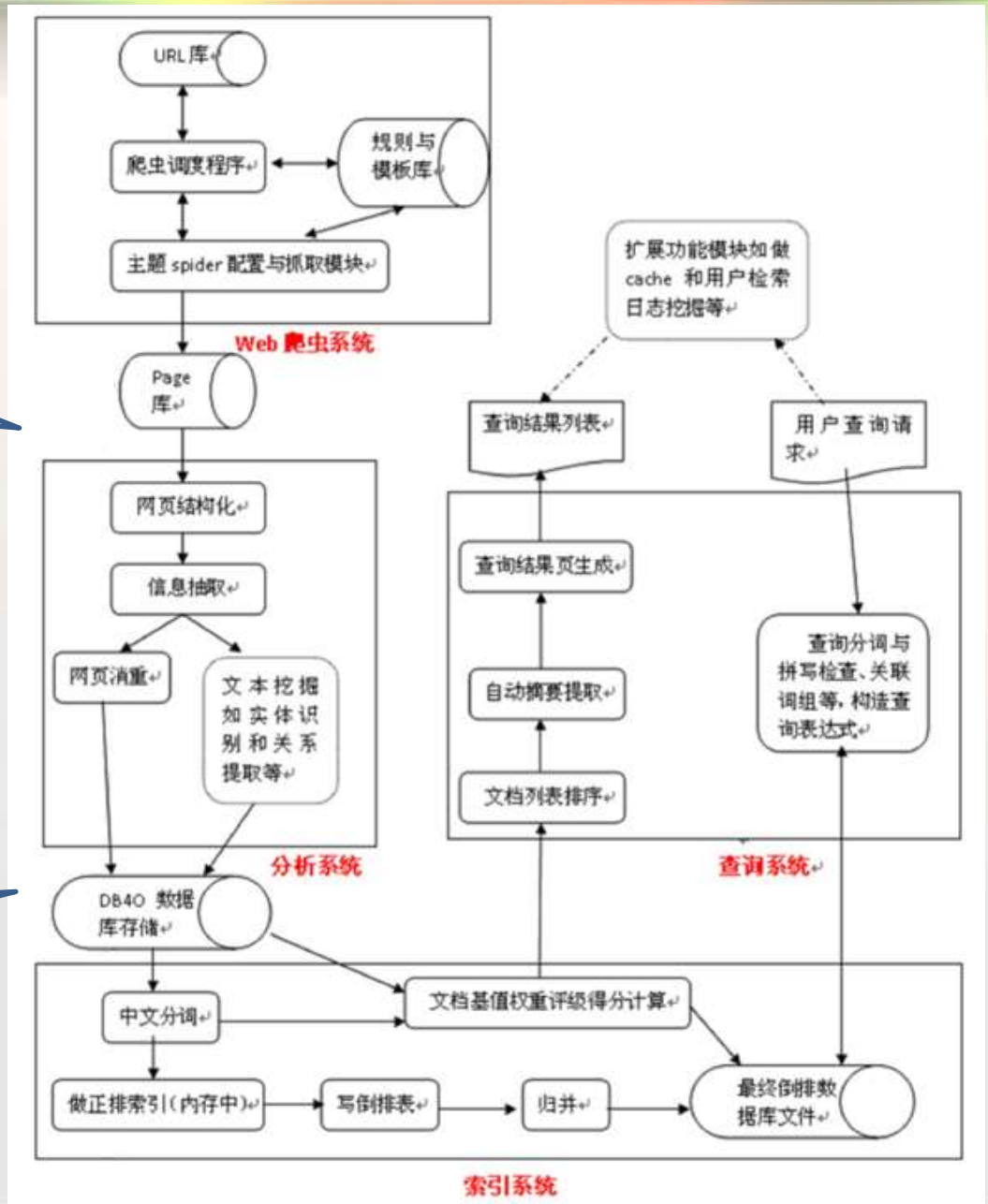
图3 搜索引擎架构(偏学术)

来自:《Search Engines: Information Retrieval in Practice》

我的工作：

V-SearchEngine 1.0
版本系统整体架构图

开发人员眼里的
搜索引擎实现



自己动手写SE：

- 关于 **V-SearchEngine：**

开发时间：2009年10月到12月；

开发目的：做出一个可配置化的垂直搜索引擎框架系统；

开发平台：windows、C++\C#、前台查询页面aspx；

系统功能：是一个完整的垂直搜索引擎框架系统。通过对主题爬虫的配置，可以实现针对各个领域如新闻、房产、购物、求职、博客、论坛等方向的垂直搜索；

性能：在1107138网页数据集上平均查询时间在十毫秒级；

开源力量：HtmlAgilityPack.1.4(一个开源html分析器) 、DB4O(一个开源纯面向对象数据库)、Log4net(一个开源日志记录工具);

最新版本：1.0；(已停止开发)

后续工作：整体移植到linux系统下，加入动态平衡树的在线索引功能、变长索引压缩，优化查询效率，即插即用方式添加多元化检索模型；(目前大致完成了50%)

最后：以子系统切分的形式开源；

V-SearchEngine：

主题爬虫

搜索主界面
(以菜谱数据
源为例)

V-SearchEngine：

通过对原始网页的去重和信息结构化提取，去除网页中广告、外链等无用信息，提供给用户清洗后的价值信息，进行了浅层的文本挖掘；


食谱

京酱肉丝

搜 索

京酱肉丝卷

http://www.douguo.com/cookbook/3985

京酱肉丝

http://www.meishichina.com/Eat/RMenu/200909/68016.html

京酱肉丝

http://www.meishichina.com/Eat/RMenu/200911/72030.html

京酱肉丝

http://www.douguo.com/cookbook/4360

京酱肉丝

http://www.douguo.com/cookbook/5425

对于关键字"京酱肉丝"进行搜索时结果的第一页下半部分：

京酱肉丝

http://www.douguo.com/cookbook/3716

京酱肉丝

http://www.douguo.com/cookbook/255

京酱肉丝

http://www.douguo.com/cookbook/5490

记录数:6087 总页数:609 1 [2] [3] [4] [5] 下一页

对于关键字"土豆"进行搜索时结果的第一页上半部分：

V-SearchEngine：

校园网内开放一天测试，一天中共有312次访问256个不同IP地址链接，其中共有182个用户成功实现了搜索访问链接建立和相应的用户检索日志记录。仅仅是demo测试，然后就废掉了；

对于关键字"土豆"进行搜索时结果的第一页上半部分：

图 3-18 对于关键字“土豆”的搜索结果的下半部分

V-SearchEngine：

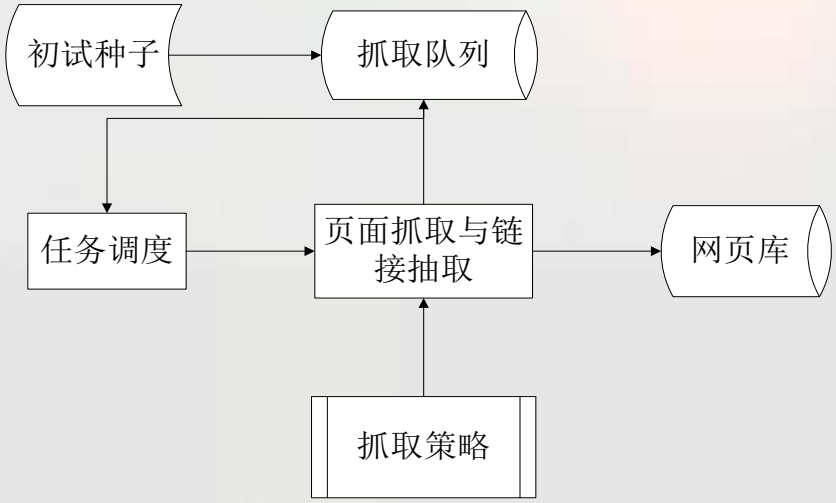
通过基于模板和规则的命名实体识别和实体关系提取，挖掘出了两种关系提供给用户查询(结果显示借鉴Google罗盘效果)，如上图所示，不过准确率不是很高。

系统介绍完毕， Simple？ Interesting ？ Useful？

解剖一：Spider or Crawler

- 网络爬虫是一个自动提取网页的程序，它为搜索引擎从万维网上下载网页，是搜索引擎的重要组成。传统爬虫从一个或若干初始网页的URL开始，获得初始网页上的URL，在抓取网页的过程中，不断从当前页面上抽取新的URL放入队列，直到满足系统的一定停止条件。

- 从理想的角度看，原理十分简单；



Spider 的问题：

- 网络不是和谐的~~
- 爬虫需要注意的问题：

(1)增量式抓取：保持抓取网页的新鲜度，涉及到更新策略和网页以及URL去重，对网页进行抓取更新时的等级划分；

(2)分布式抓取：时间和地理上的分布式，例如腾讯搜搜的爬虫服务器覆盖全国二十多个城市；

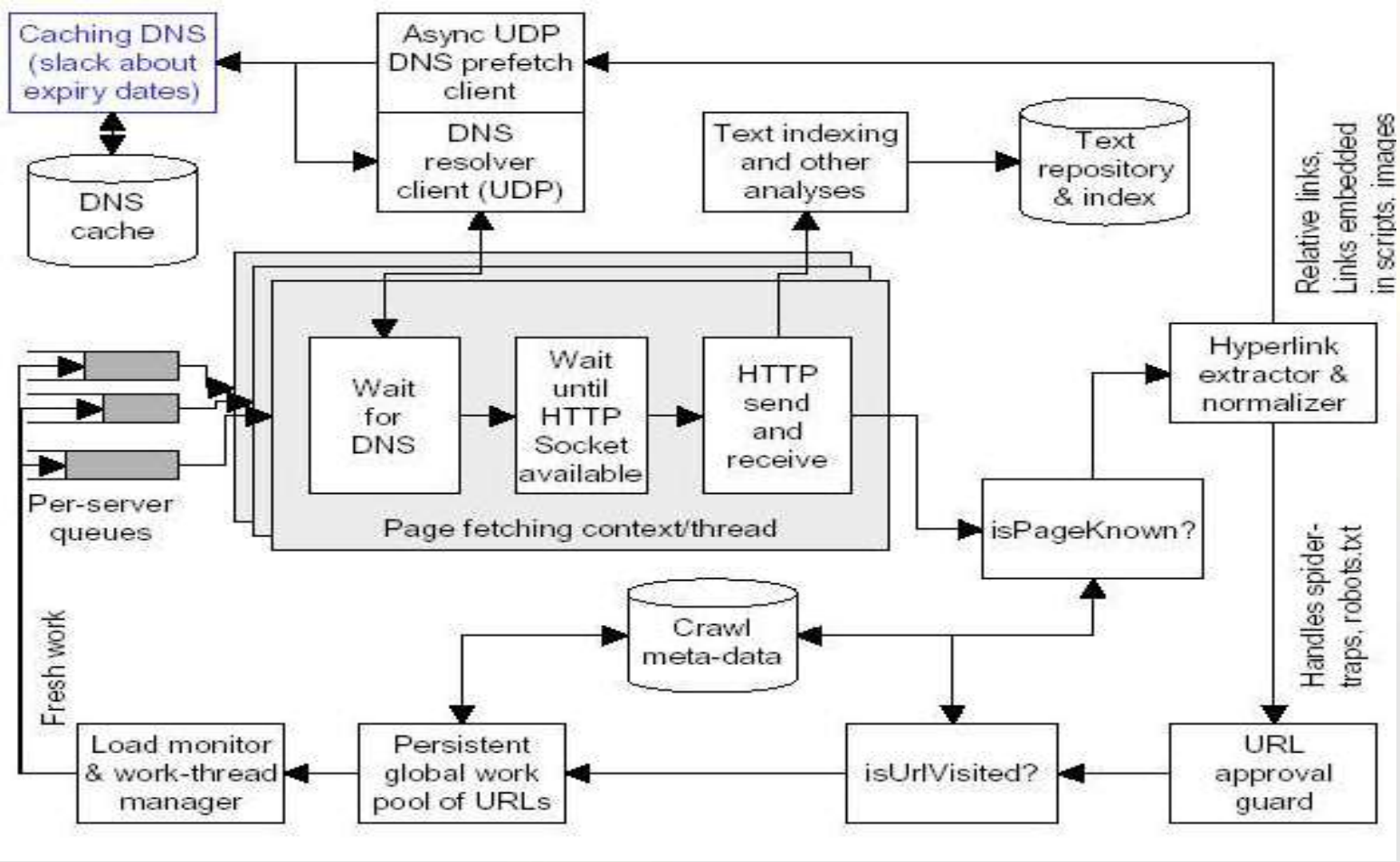
(3)绅士抓取：平衡礼貌策略，一般要遵守Robots exclusion protocol，Google的爬虫要比百度的绅士的多；

(4)Deep Web的抓取：如何抓取40%的“暗网”，搜索引擎的一个挑战，寻找“入口”，软接口、Deep web数据集成技术等；

(5)欺骗与黑洞：SEO针对爬虫的欺骗，以及网页中的链接黑洞；

一种Crawler的体系结构：

- 大规模爬取器的一种结构图：



解剖二：自然语言处理(NLP)

- 搜索引擎中对NLP的需求：

(1)分词； (已比较成熟)

(2)词性标注与词义消歧；(已比较成熟)

(3)句法分析、语法分析、语义分析；(尚有很大的调高空间)

(4)命名实体挖掘； (研究热点)

(5)文本分类； (已比较成熟)

(6)文本聚类； (已比较成熟)

(7)自动文摘； (已比较成熟，但在搜索引擎中一直很难找到自己的位置，Snippet)

(8)复述(paraphrasing)； (研究热点)

(9)情感计算；(即情感信息的抽取、情感信息的分类以及情感信息的检索与归纳)

(10)机器翻译；(用于CLIR)

中文分词：

张华平/n 1995年/t 离开/v 江西/n 鄱阳/n 老家/n 就读/v 于/p 北方/s 工业/n 大学/n ，/w 如今/t 已经/d 是/v 中科院/n 计算/v 所/q 的/u 副/b 研究员/n ，/w 他/r 说/v ICTCLAS/x 就/d 像/v 是/v 他/r 的/u 孩子/n 一样/u 珍爱/v 。/w

➢主流中文分词方法与技术：

一.基于层叠隐马尔可夫模型技术
代表产品：中科院ICTCLAS 3.0

二.基于语言模型与统计和规则相结合的技术
代表产品：海量中文智能分词组件（准确率较高）

三.基于CRF语言模型技术
代表产品：哈工大IR实验室LTP分词模块

四.其他研究型的分词模型

中文分词：

- 现阶段人们提出了许多中文分词的算法，主要可以分成以下三类：

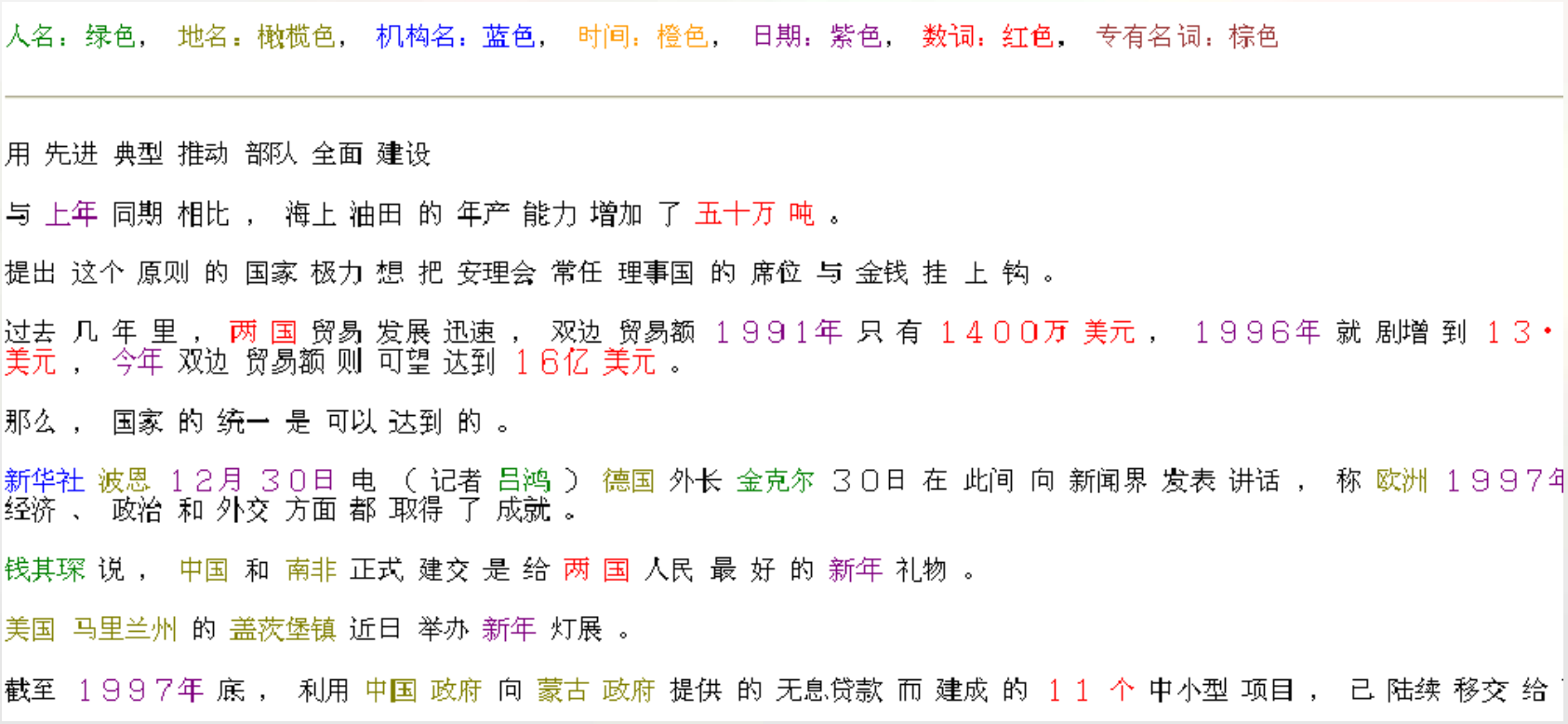
a．基于字符串匹配的分词方法：待分析的中文字符串与一定规模的词典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功(识别出一个词)。按照扫描方向的不同，串匹配分词方法又可分为正向和逆向匹配；按照优先匹配的原则，可分为最大和最小匹配。

b．基于理解的分词方法：这种分词方法是通过让计算机模拟人对句子的理解，达到识别词的效果，也称人工智能法。

c．基于统计的分词方法；这种方法只需对语料中的字组频率进行统计，而不需要切分词典，因而又叫作无词典分词法或统计取词方法，其主要包括基于引马尔可夫模型、基于最大熵模型、基于条件随机场模型的方法。

命名实体识别：

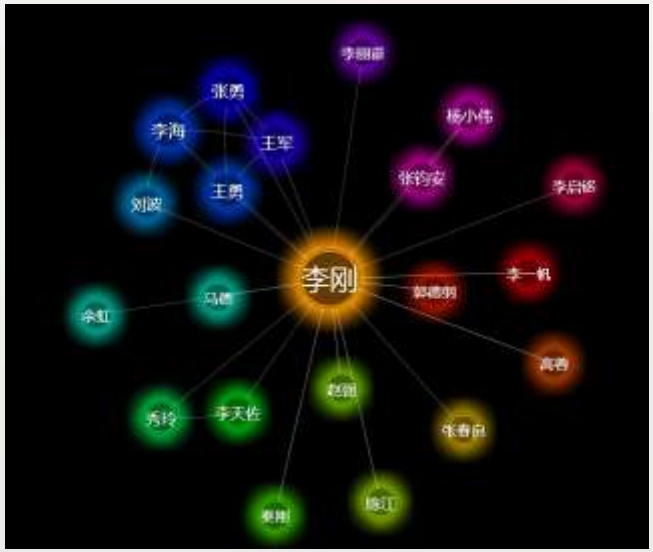
➢现有的任务主要分为七大类，同时也可面向特定领域进行识别：

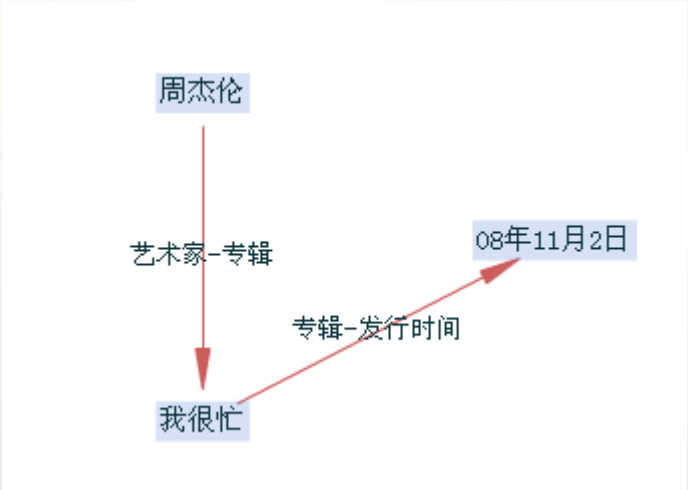


➢现有的主要实现方法：规则与HMM结合、规则与CRF结合等等。总体思想：基于规则与统计进行；

小扩展：实体关系提取(RE)

➢对相同实体以及不同实体间的关系进行机器学习和挖掘：



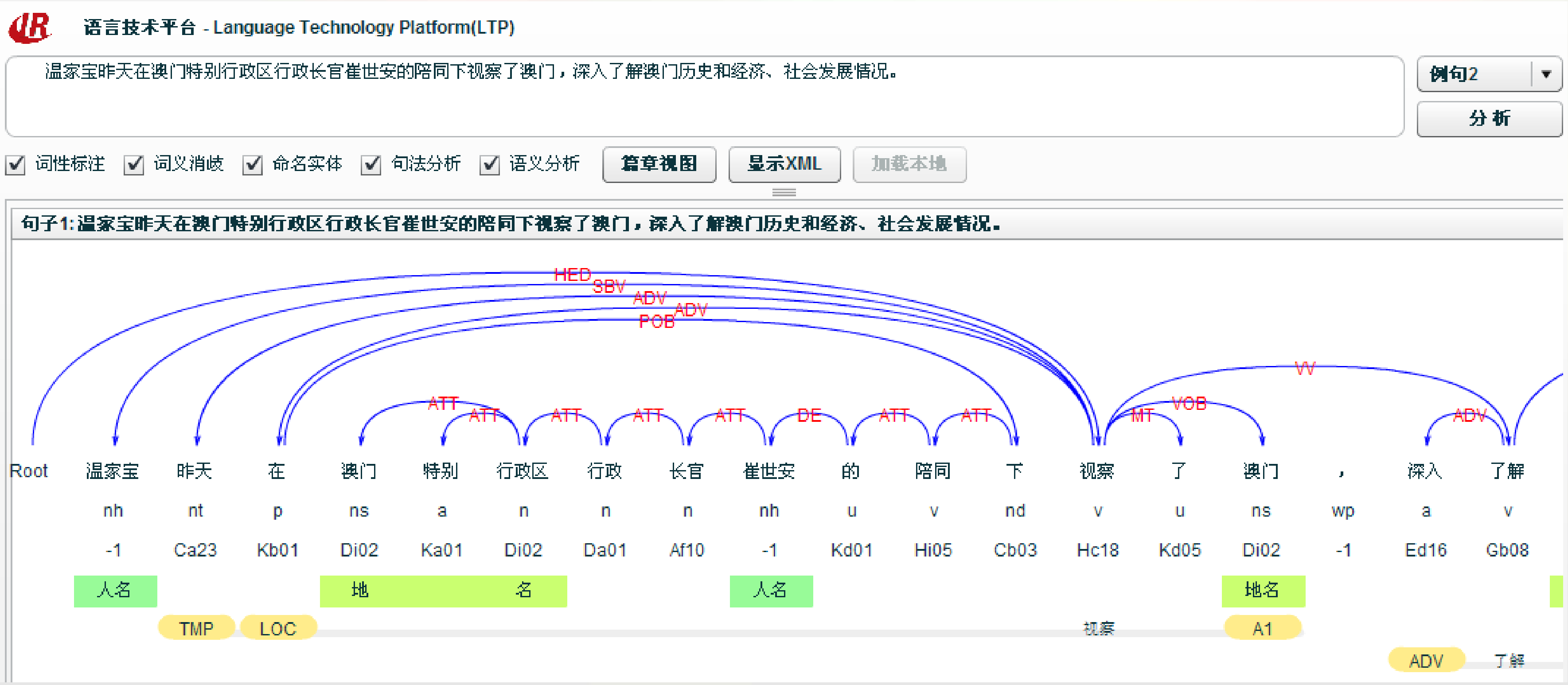


➢目前来说整体准确率都不高，封闭条件下准确率不超过90%，开放环境下准确率很低；

➢方法：分三种：有指导的学习、半指导、无指导。基于特征向量、基于序列模式挖掘、基于树核函数等；

难点与挑战：句法分析与语义分析

➢NLP亟需解决的问题：



➢如何提高准确率：？？？？？？？？？？？？？？？？？？？？？？？？？？

文本分类与文本聚类：

➢文本分类：对文本集(或其他实体或物件)按照一定的分类体系或标准进行自动分类标记；

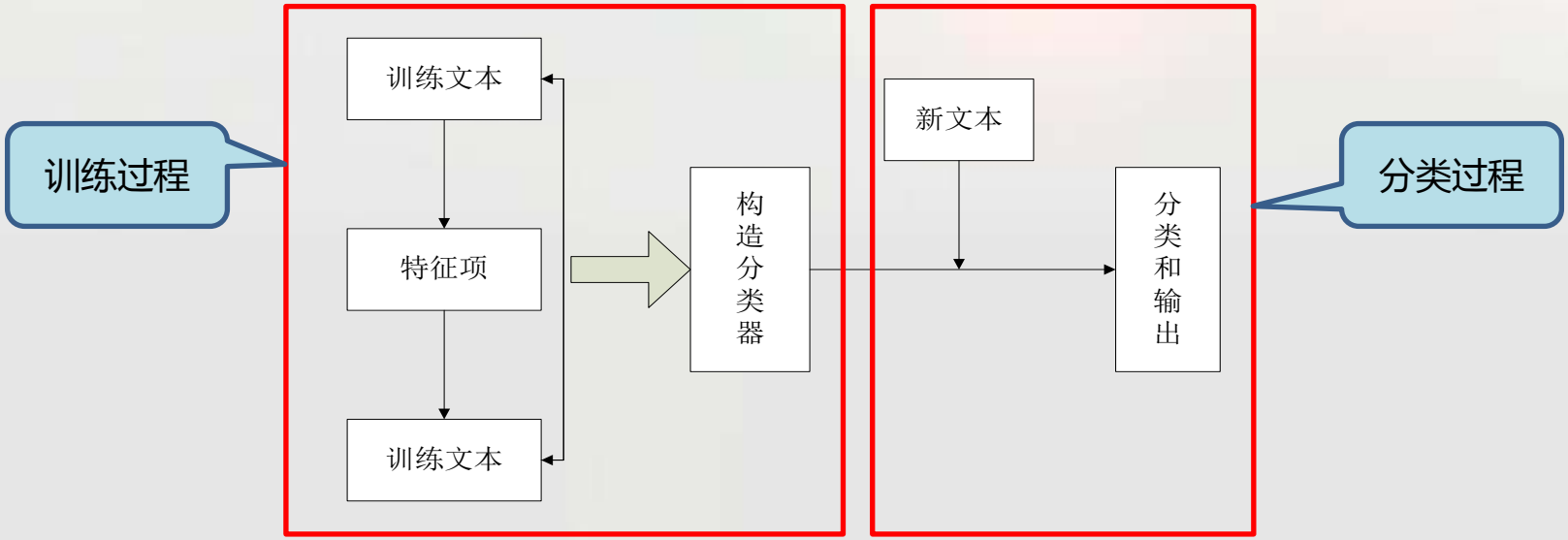
(1)有指导或者半指导学习，需要样本标注和训练过程；

(2)参考算法：决策树，Rocchio，朴素贝叶斯，神经网络，支持向量机，线性最小平方拟合，kNN，遗传算法，最大熵，Generalized Instance Set等；

➢文本聚类

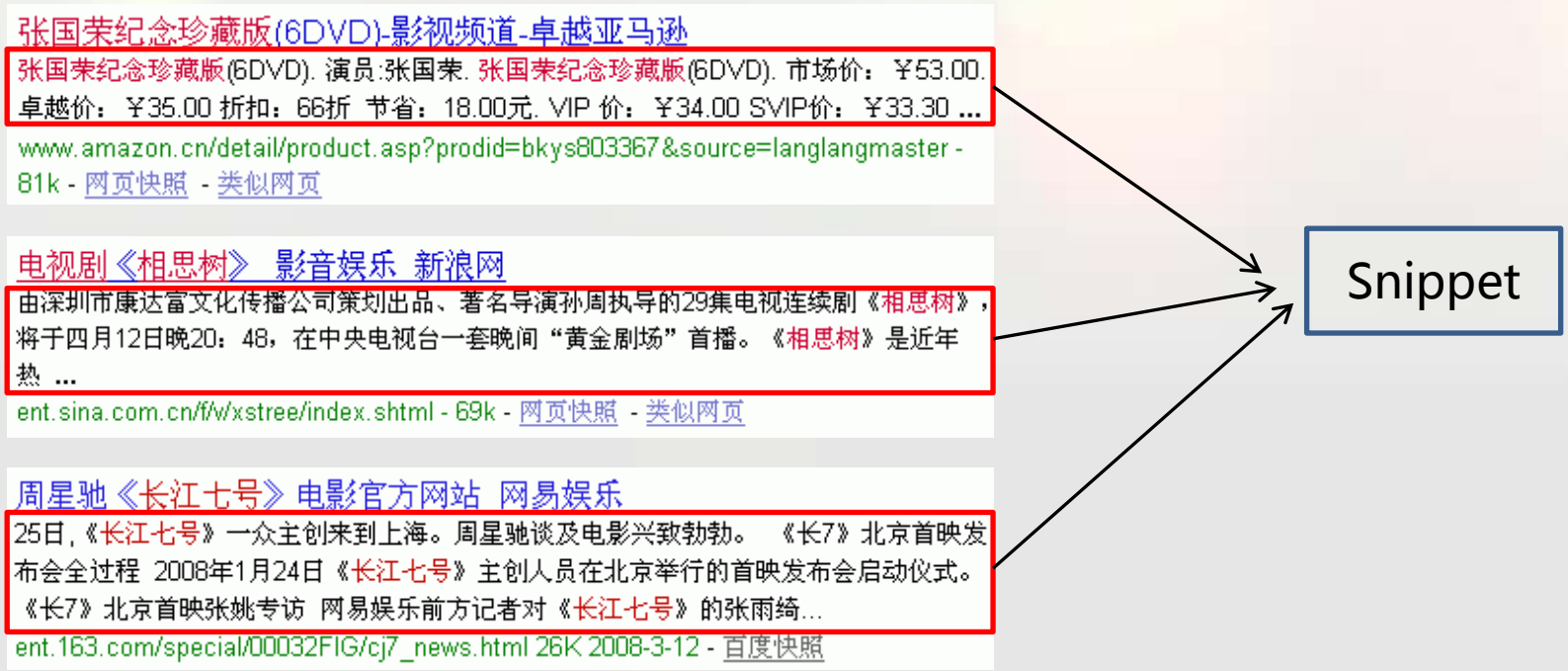
(1)无指导学习；

(2)参考算法： K-MEANS算法、K-MEDOIDS算法、CLARANS算法、BIRCH算法、CURE算法、CHAMELEON算法、DBSCAN算法、OPTICS算法、DENCLUE算法等；



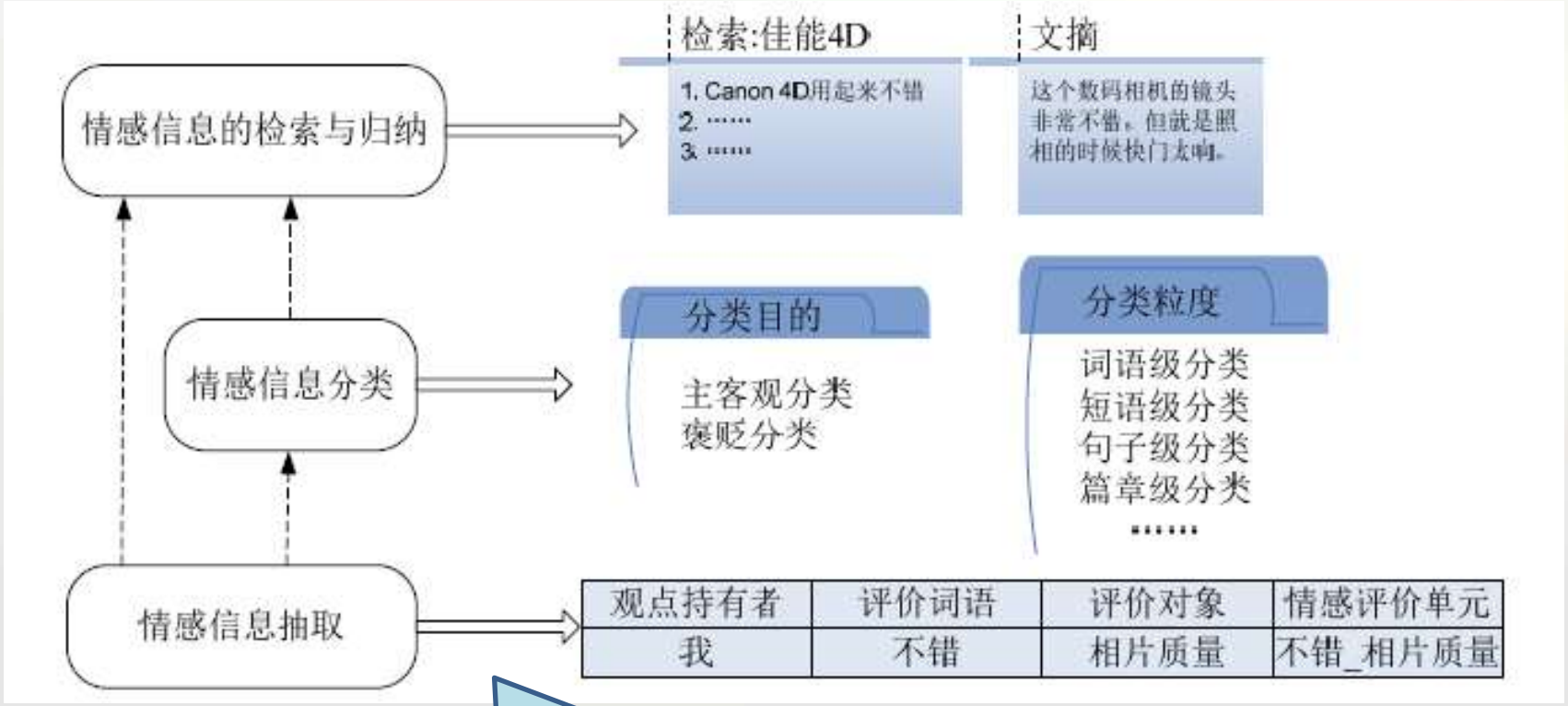
自动文摘与Snippet：

- 文本的自动文摘研究历史已经非常古老了，但是在应用转化上一直处于一个比较尴尬的境地，经常出现开着装甲车偷白菜的情景；
- 在搜索引擎中快速而方便的Snippet已经基本足够，不需要非常复杂的自动文摘技术；



文本情感分析：

➢文本情感分析,又称意见挖掘,简单而言,是对带有情感色彩的主观性文本进行分析、处理、归纳和推理的过程。目前在搜索引擎的应用主要还是在用户查询分析上。



赵妍妍学姐总结的情感分析层次框架

文本情感分析：

- 一个文本倾向性分析示例，基于情感词抽取和知网(HowNet)语义距离计算实现：

- 关于NLP就暂时介绍到这里，其他的我暂时没有涉及过，没有实践没有发言权；

解剖三：网页信息提取

- 可将网页信息提取任务分成以下三类：

  (1)网页主题信息提取，如新闻、博客网页的正文；

  (2)多记录信息提取，如BBS帖子、校内及微博状态；

  (3)定制性价值信息提取，如淘宝页面交易记录、用户评论、社会化标签；

- 为什么要提取：

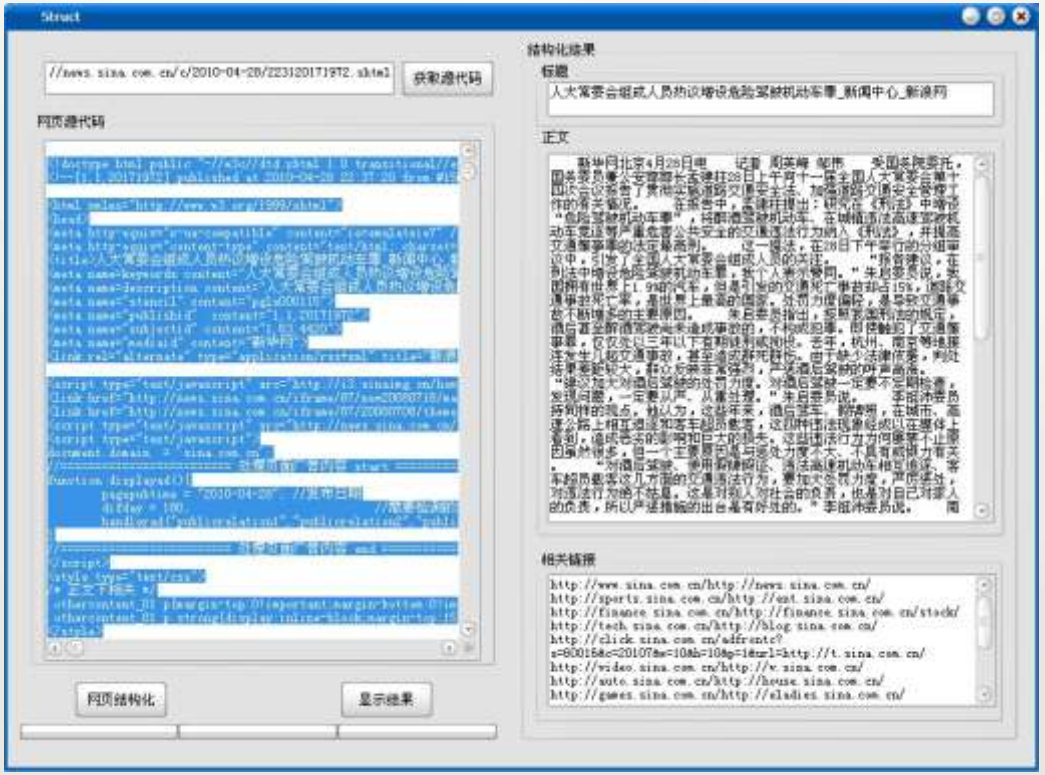
  (1)网页净化，去除广告等噪声；

  (2)提取需要被索引的信息，去除SEO的欺骗内容；

  (3)为用户展示更加结构化的数据；

网页正文提取：

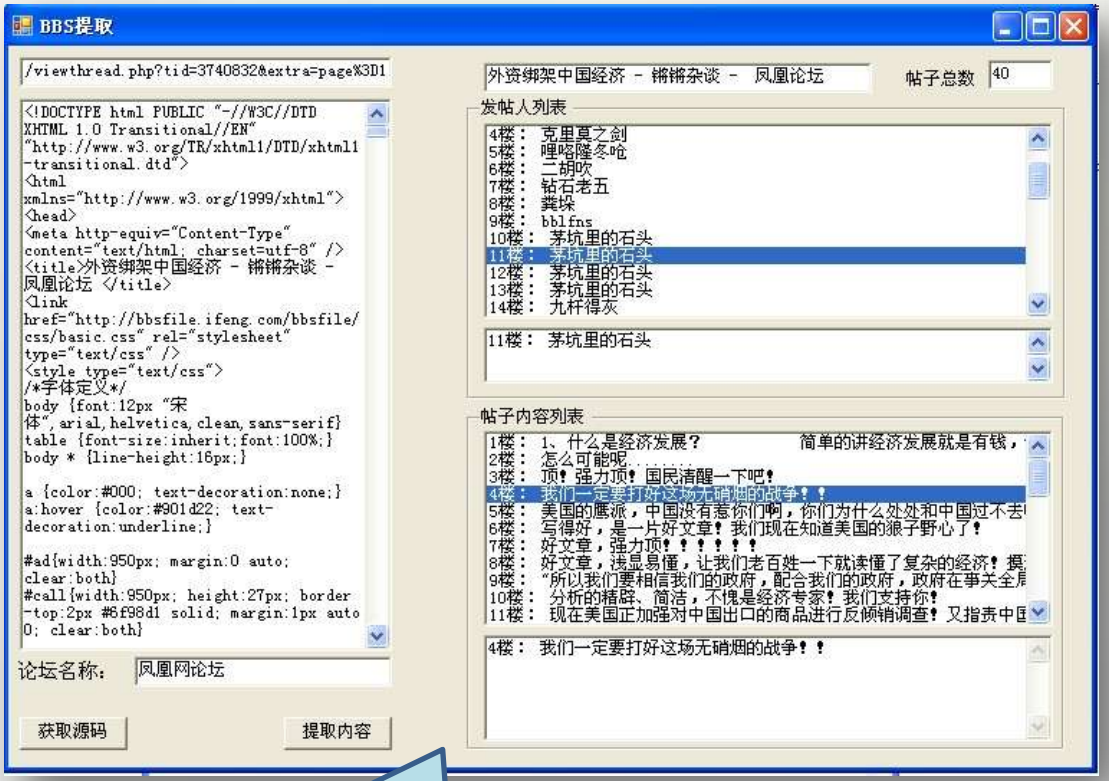
- 识别与定位：如何哪一部分是正文？
- 方法非常多：基于统计的，基于规则的，基于NLP的等等，准确率都已经非常高了。



- 基于网页DOM树和统计规则的主题正文信息提取

多记录型网页提取：

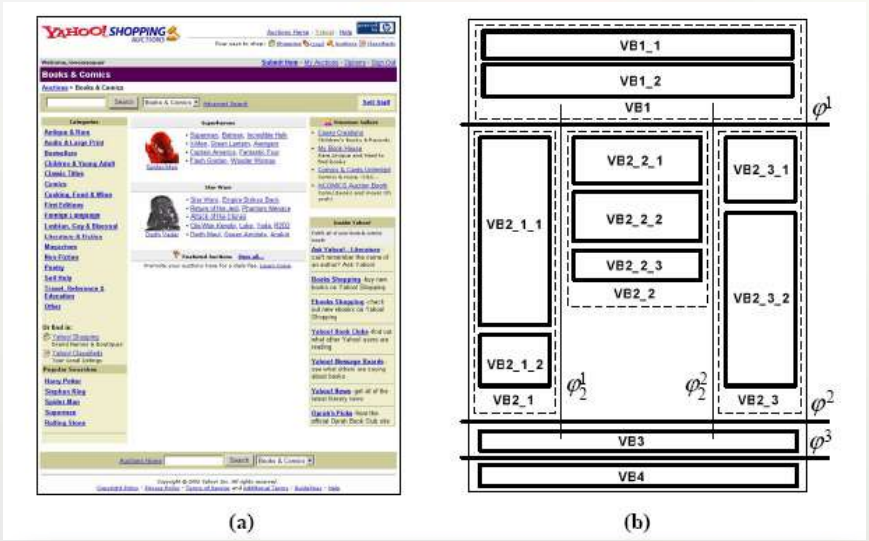
- 针对BBS的帖子、Twitter、微博等：
- 基于信息量、规则、视觉等方式进行提取；

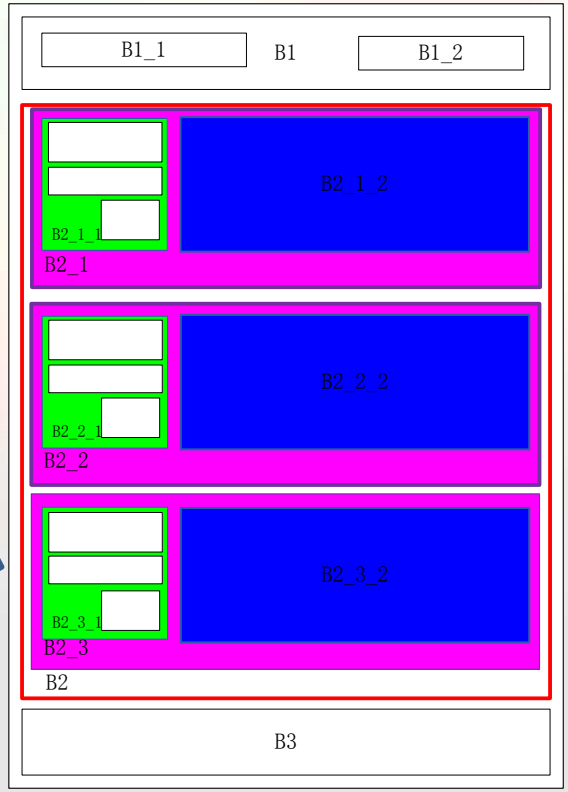


基于HtmlParser的BBS发言文本内容提取

基于视觉特征的网页信息提取：

➢思想：对网页进行切割，然后对视觉块进行定位提取；





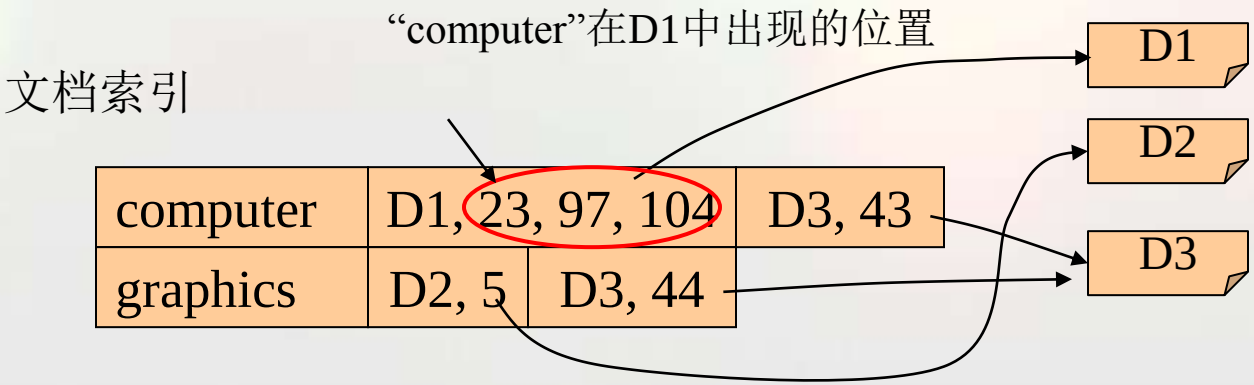
解剖四：倒排索引

➢建立索引的目的

(1)对文档或文档集合建立索引，以加快检索速度；

(2)倒排文档（或倒排索引）是一种最常用的索引机制；

(3)倒排文档的索引对象是文档或文档集合中的单词等。例如，有些书往往在最后提供的索引（单词—页码列表对），就可以看成是一种倒排索引；



索引结构: ***hashing, B+-trees, tries***.

可以进行部分匹配: ’ %comput% ’

可以进行短语搜索:查找包含“computer graphics”的文档

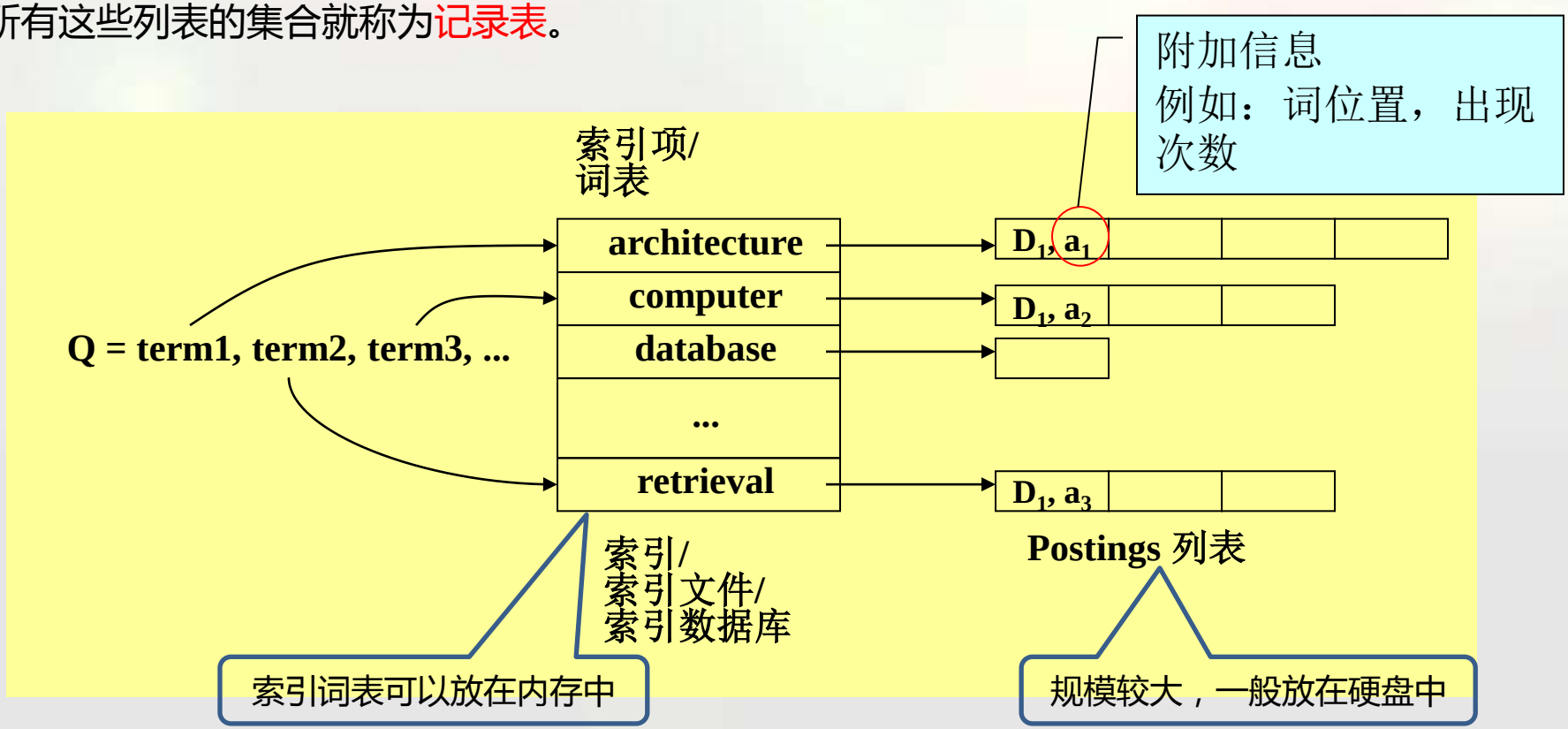
倒排索引：

➢倒排文档组成：

(1)倒排文档一般由两部分组成：词汇表（vocabulary）和记录表（posting list）。

(2)词汇表是文本或文本集合中所包含的所有不同单词的集合。

(3)对于词汇表中的每一个单词，其在文本中出现的位置或者其出现的文本编号构成一个列表，所有这些列表的集合就称为记录表。



倒排索引：

➢建立索引的过程：

- 识别文档中的词（分词）
- 删除停用词(stop words)（如“的、是”等）
- 提取词干(stemming)（将派生相关词合并在一起）
- 用索引项的标号代替词干(stems)
- 统计词干的数量(*tf*)
- (可选) 对低频词项使用同义词词典(thesaurus)（在大型搜索引擎中一般不这么做，会大大增加索引文件的大小）
- (可选) 对高频词项构成短语
- 计算所有单个词项、短语和语义类的权重等附加信息
- 建立倒排表(在内存中进行分块倒排，达到一定大小后写入到磁盘中)

倒排索引：

➢倒排索引的问题：

●倒排索引的压缩：目前使用变长索引压缩技术可以将索引大小压缩到原始文档大小的10%以下。良好的索引压缩可以导致在内存中存放更大的索引缓存，提高查询效率。

●倒排索引的更新：爬虫抓取网页的不断更新要求需要支持实时在线的索引技术，可以使用基于动态平衡树来实现；

●分布式索引建立：对于大型搜索引擎，采用基于文档分割的分布式索引技术。基于Mapreduce，Map阶段和Reduce阶段将计算任务划分成子任务块：map阶段将输入的数据片映射成键-值对。Reduce阶段将同一键(词项ID)的所有值(文档ID)集中存储，以便快速读取和处理。可以基于Hadoop进行扩展。

解剖五：搜索结果页生成

➢最重要也是SE最核心的技术：链接分析与排序；(用户一般只看前面的结果)

网页之间的超链接：

●链接反映的是网页之间形成的“参考”、“引用”和“推荐”关系；

●可以合理的假设，若一篇网页被较多的其他网页链接，则它相对较被人关注，其内容应该是较重要、或者较有用；

●因此，可以认为一个网页的“入度”（指向它的网页的个数）是衡量它重要程度的一种有意义的指标。这和科技论文的情况类似，被引用较多的就是较好的文章；

●同时，人们注意到，网页的“出度”（从它连出的超链个数）对分析网上信息的状况也很有意义的，因此可以考虑同时用两个指标来衡量网页 ；

链接分析技术：

- 链接分析和商业排序算法是SE公司的存在灵魂；
- PageRank：基于「**从许多优质的网页链接过来的网页，必定还是优质网页**」的回归关系

Sergey Brin 和 Lawrence Page于1998年提出PageRank算法，2001获得美国专利，布林和佩齐都是Google的创始人；

$$PR(A) = (1-d) + d\ (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))$$

公式解释：其中PR(A)表示的是从一个外部链接站点t1上，依据Pagerank系统给你的网站所增加的PR分值；PR(t1)表示该外部链接网站本身的PR分值；C(t1)则表示该外部链接站点所拥有的外部链接数量；d是阻尼系数(在0和1之间)。通过全局计算的不断收敛最后确定页面的PR值；

- 参考资料：

关于pagerank最经典的两篇文章：

Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, 'The PageRank Citation Ranking: Bringing Order to the Web', 1998.

Taher H. Haveliwala, 'Efficient Computation of PageRank', Stanford Technical Report, 1999.

Google 的秘密- PageRank 彻底解说 中文版：http://www.kreny.com/pagerank_cn.htm

链接分析技术：

- HITS：由乔恩·克莱因伯格(Jon Kleinberg)于1998年设计提出

基本思想：

1个网页有两个属性： Hub/Authority ：

Hub表明了贡献度(hub网页指向权威网页)；

Authority表明权威程度，按照此排序；

好的Hub页指向了好的Authority 页；

好的Authority 页被好的Hub页指；

HITS（Hyperlink - Induced Topic Search）算法是利用Hub/Authority方法的搜索方法；

算法如下：

将查询q提交给传统的基于关键字匹配的搜索引擎;

搜索引擎返回很多网页，从中取前n个网页作为根集(root set)，用R表示，R满足如下3个条件:

R中网页数量相对较小

R中网页大多数是与查询q相关的网页

R中网页包含较多的权威网页

通过向R中加入被R引用的网页和引用R的网页将R扩展成一个更大的集合S;

采用迭代的方法计算A/H值，最终按照A进行排序;

Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment . Jour nal of the ACM ,1999 ;46 (5)

链接分析技术：

- PageRank算法中对于向外链接的权值贡献是平均的，Hits算法考虑了不同链接的重要性
- PageRank与Hits算法：

它们都利用了网页和超链组成的有向图，根据相互链接的关系进行递归的运算。

但是，两者又有很大的区别，主要在于运算的时机：

Google是在网页搜集告一段落时，离线的使用一定的算法计算每个网页的权值，在检索时只需要从数据库中取出这些数据即可，而不用做额外的运算，这样做的好处是检索的速度快，但丧失了检索时的灵活型。

HITS使用即时分析运算策略，每得到一个检索，它都要从数据库中找到相应的网页，同时提取出这些网页和链接构成的有向子图，再运算获得各个网页的相应链接权值。这种方法虽然灵活性强，并且更加精确，但在用户检索时进行如此大量的运算，检索效率显然不高。

- 研究人员和工程师还提出了很多其他的优秀的网页权重分析方法；

解剖六：用户查询及优化

- 主要是基于NLP技术；
- 一个典型的例子：百度的框计算——搜索框查询内容的分发；
- 根据用户输入的词语或者句子构造查询表达式、纠错与自动补全；
- 支持高级检索：支持词条、布尔、范围、前缀、短语、多短语、模糊、通配符、跨度检索；
- 根据用户查询日志挖掘和基于NLP的查询自动扩展；
- 相关反馈与伪相关反馈；

SE End：

解剖完毕

信息检索： Information Retrieval

- 信息检索衍生于图书和情报管理学科；
- 主要讨论的是Ad Hoc信息检索；
- 信息检索评测的两个重要指标：

(1)准确率

$$\text{准确率}=\frac{\text{检索返回的相关文档集数目}}{\text{检索返回的文档数目}}$$

(2)召回率

$$\text{召回率}=\frac{\text{检索返回的相关文档集数目}}{\text{实际相关的文档数目}}$$

- Web搜索与广义信息检索的区别：

(1)在web检索中，召回率的估计非常困难；

(2)用户不关心准确率，只关心前几页是不是有确切的相关信息；

(3)在网络环境下，对响应时间的要求非常苛刻；

信息检索： Information Retrieval

➢我们前面尚未涉及到的有关信息检索的其他内容：

- 信息检索模型；
- 跨语言检索；
- 自动问答系统：如百度知道，天涯问答等；
- 全文检索系统、站内检索系统；
- 多媒体检索；
- P2P检索；
- 信息检索的评测；
- 用户行为分析；
- 结构化数据检索与XML检索；
- 数据集成与融合；

信息检索模型：

- 检索模型提供了一种度量查询和文档之间相似度的方法，一般基于一种共同的理念：文档和查询共有的term越多，则认为文档和该查询越相关；

  处理对象是查询Q和文档集合{D1,D2,.....Dn}，处理过程就是计算每篇文档Di和这个查询的相似度SC(Q，Di).

- 常用的信息检索模型有如下几种：

  (1)向量空间模型；将查询和文档都表示为词项空间的向量。

  (2)概率模型；基于文档集词项出现的可能性，对文档和查询匹配词项计算联合概率。

  (3)语言模型；建立一个语言模型，并计算出各文档“生成”查询的概率。

  (4)推理网络；采用贝叶斯网络来推断文档和查询间的相关性。

  (5)布尔检索；对查询词项赋予布尔权重，根据权重计算查询和文档的相似度。

  (6)LSI(隐性语义检索)；对词项-文档矩阵进行奇异值分解降维，计算文档在多维空间的语义距离。

  (7)神经网络方法；（不推荐）

  (8)遗传算法；通过进化的方法来得到查找相关文档的最优查询，通过适应度来收敛。

  (9)模糊集检索；文档映射为模糊集，根据模糊集的交并补等操作计算文档和查询间的隶属度。

举个例子：向量空间模型(VSM)

- 原理：查询向量<q0,q1…qn>,文档向量<d0,d1…dn>，转换为计算向量之间的相近性，如可以计算两个向量之间的夹角，内积等等；
- 基础：使用IDF(逆文档概率)来计算，为每篇文档建立一个对应的向量。

首先定义：

t—文档集中不同的词项的个数；

tfij —词项tj在文档Di中出现的次数，也就是词频；

dfj—包含词项tj的文档的篇数；

idfj—lg(d/dfj),其中d表示所有文档的篇数，这就是逆文档频率；

例子：

Q:gold silver truck

D1:shipment of gold damaged in a fire

D2: Delivery of silver arrived in a silver truck

D3: Shipment of gold arrived in a truck

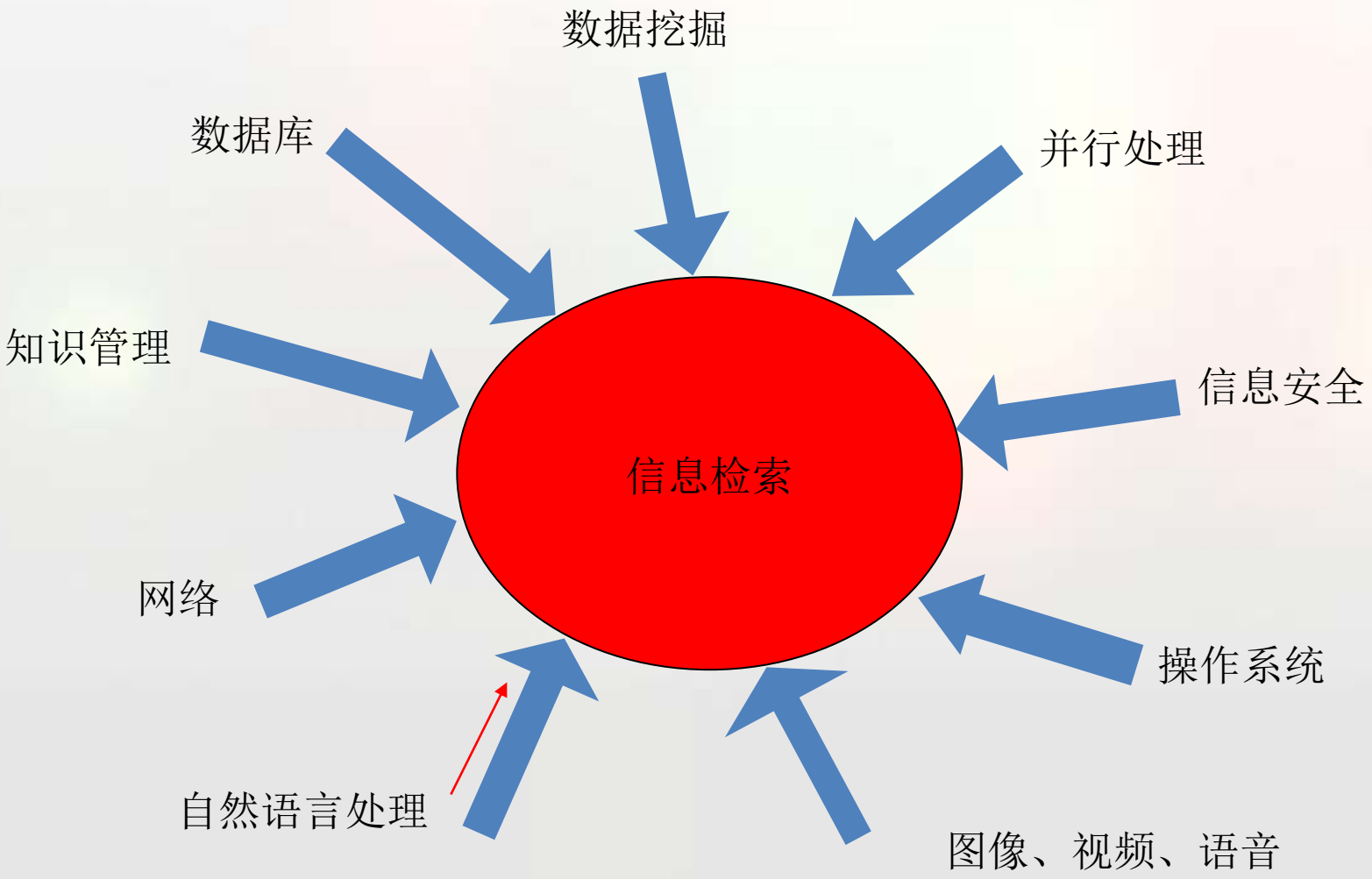
举个例子：向量空间模型(VSM)

*idf*a=0	*idf*arrived=0.176	*idf*damaged=0.477	*idf*delivery=0.477	*idf*fire=0.477	*idf*gold=0.176
*idf*in=0	*idf*of=0	*idf*silver=0.477	*idf*shipment=0.176	*idf*truck=0.176	

docid	a	arrived	damaged	delivery	fire	gold	in	of	shipment	silver	truck
D1	0	0	0.477	0	0.477	0.176	0	0	0.176	0	0
D2	0	0.176	0	0.477	0	0	0	0	0	0.954	0.176
D3	0	0.176	0	0	0	0.176	0	0	0.176	0	0.176
Q	0	0	0	0	0	0.176	0	0	0	0.477	0.176

- SC(Q，D1)=0X0+0X0+0X0.477+0X0+0X0.477+0.176X0.176+0X0+0X0+0X0.176+0.477X0+0X0.176 =0.176X0.176=0.031
- SC(Q，D2)=0.486
- SC(Q，D3)=0.062
- 故最终检索结果顺序为D2，D3，D1;

信息检索： Information Retrieval



开源力量：

➢开源搜索引擎：

(1)Nutch: http://nutch.apache.org/

目前最新版本： **24 September 2010 - Apache Nutch 1.2 Released**

利用Nutch，十五分钟即可配置搭建一个搜索引擎；

开源力量：

- 开源全文检索系统：

(1)Lucene: http://lucene.apache.org/

最新版本： 3 December 2010 - Lucene Java 3.0.3 and 2.9.4 available

(2)firtex: http://www.firtex.org/

最新版本： 1.2.0 RC 20090705

2007年3月，FirteX获2006’第二届中国开源软件竞赛学生组银奖（金奖空缺）

开源力量：

- 开源爬虫：

(1) Larbin：http://larbin.sourceforge.net/

(2) Heritrix：http://crawler.archive.org/

(3) WebSPHINX：http://www.cs.cmu.edu/~rcm/websphinx/

(4) Jspider ：http://j-spider.sourceforge.net/

开源力量：

- 开源中文分词：

(1) ICTCLAS 3.0 ：http://ictclas.org/index.html

(2) IKAnalyzer3.0： http://code.google.com/p/ik-analyzer/

(3) Paoding ‘s Knives： http://code.google.com/p/paoding/

(4) imdict-chinese-analyzer: http://code.google.com/p/imdict-chinese-analyzer/

……

- 还有很多其他优秀的开源作品供我们参考，开源的力量是无穷的~~

End：

Gmail : yxyx3258@gmail.com

豆瓣： http://www.douban.com/people/13869034/

微博： http://t.sina.com.cn/yangxiaovenus