信息检索与数据挖掘

第5章 向量模型及检索系统

——第二讲 检索系统

课程内容

- 第1章　绪论
- 第2章 布尔检索及倒排索引
- 第3章 词典查找及扩展的倒排索引
- 第4章 索引构建和索引压缩
- 第5章 向量模型及检索系统
  - **向量模型**
  - **检索系统的评分计算**
- **第6章 检索的评价**
- 第7章 相关反馈和查询扩展
- 第8章 概率模型
- 第9章 基于语言建模的检索模型
- 第10章 文本分类
- 第11章 文本聚类
- 第12章 Web搜索
- 第13章 多媒体信息检索
- 第14章 其他应用简介

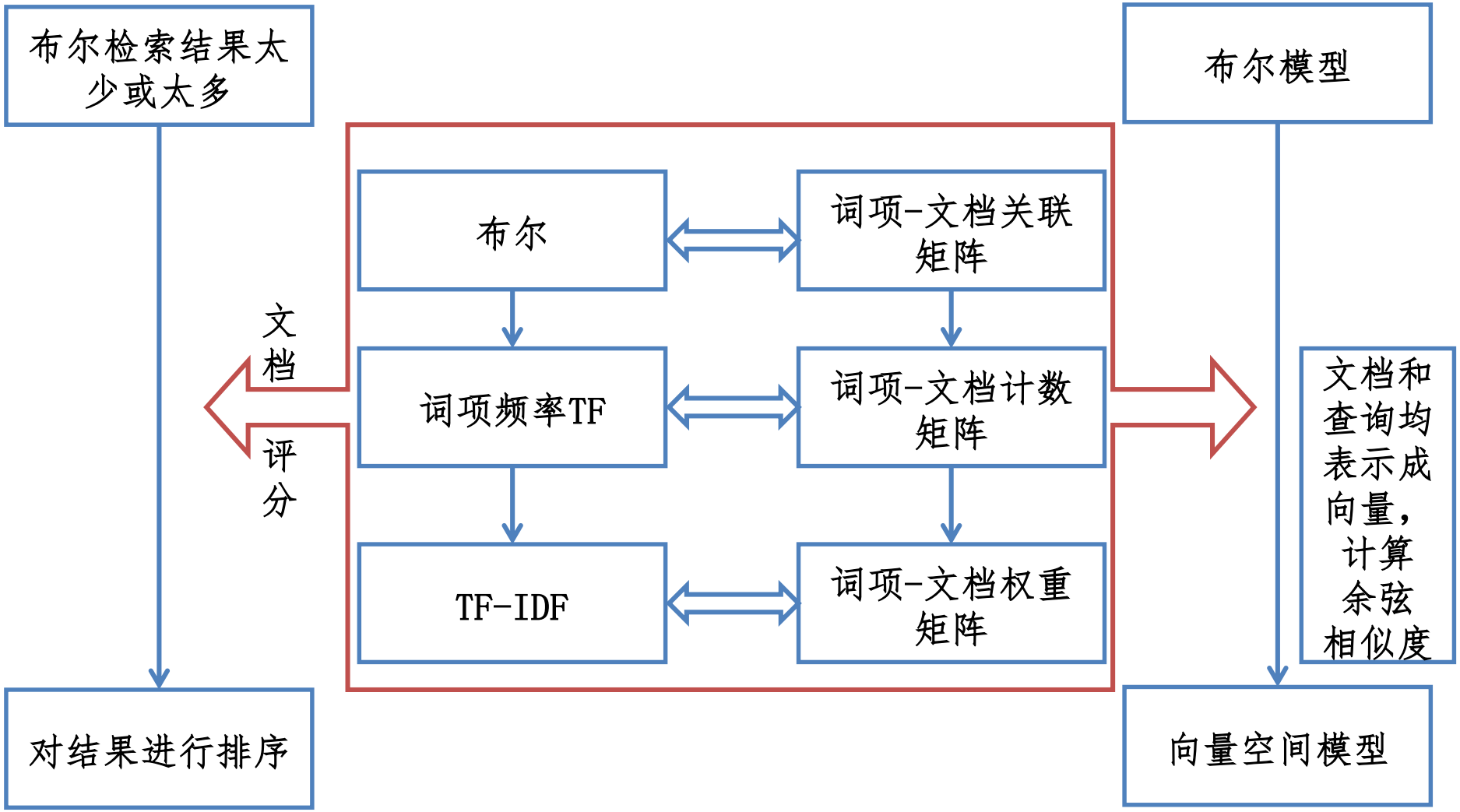
本讲提纲

1. 上一讲回顾
2. 结果排序的重要性
3. 结果排序的实现
4. 完整的搜索系统

提纲

1. 上一讲回顾
2. 结果排序的重要性
3. 结果排序的实现
4. 完整的搜索系统

回顾：从布尔模型到向量空间模型



回顾：词项频率tf

- t 在 d 中的对数词频权重定义如下：

$$w_{t,d} = \begin{cases} 1 + log_{10} tf_{t,d} & if \quad tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$$

- 文档–词项的匹配得分

$$\sum_{t \in q \cap d} (1 + log_{10} tf_{t,d})$$

回顾：idf权重

- df_t 是出现词项 t的文档数目
- df_t 是和词项 t的信息量成反比的一个值
- 于是可以定义词项 t的idf权重：

$$idf_t = log_{10}(\frac{N}{df_t})$$

（其中N 是文档集中文档的数目）

- idf_t 是反映词项 t的信息量的一个指标

逆文档频率：多个文档中都会出现的常见词、高频词idf较低；反之罕见词的idf高

回顾：tf-idf权重

- tf-idf权重

$$w_{t,d} = (1 + log_{10} tf_{t,d}) \times log_{10}(\frac{N}{df_t})$$

- tf-idf 是信息检索中最著名的权重计算方法
- tf-idf值随着词项在单个文档中出现次数增加而增大
- tf-idf值随着词项在文档集中数目减少而增加

某一特定文件内的高词语频率，以及该词语在整个文件集合中的低文件频率，可以产生出高权重的TF-IDF。因此，TF-IDF倾向于过滤掉常见的词语，保留重要的词语。

词袋模型（*Bag of Words*）

- 不考虑词在文档中出现的顺序
- “John is quicker than Mary” 和“Mary is quicker than John”的表示结果一样
- 这就是**词袋模型**

- TF、DF、IDF、TF-IDF都只考虑：**词袋模型**

图像的特征

Bag-of-words representation for an image

Object

Bag-of-words

二值关联矩阵

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . . .
ANTHONY	1	1	0	0	0	1
BRUTUS	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
. . .						

每篇文档表示成一个二值向量$\in \{0, 1\}^{|V|}$

二值矩阵→词频矩阵

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . . .
ANTHONY	157	73	0	0	0	1
BRUTUS	4	157	0	2	0	0
CAESAR	232	227	0	2	1	0
CALPURNIA	0	10	0	0	0	0
CLEOPATRA	57	0	0	0	0	0
MERCY	2	0	3	8	5	8
WORSER	2	0	1	1	1	5
. . .						

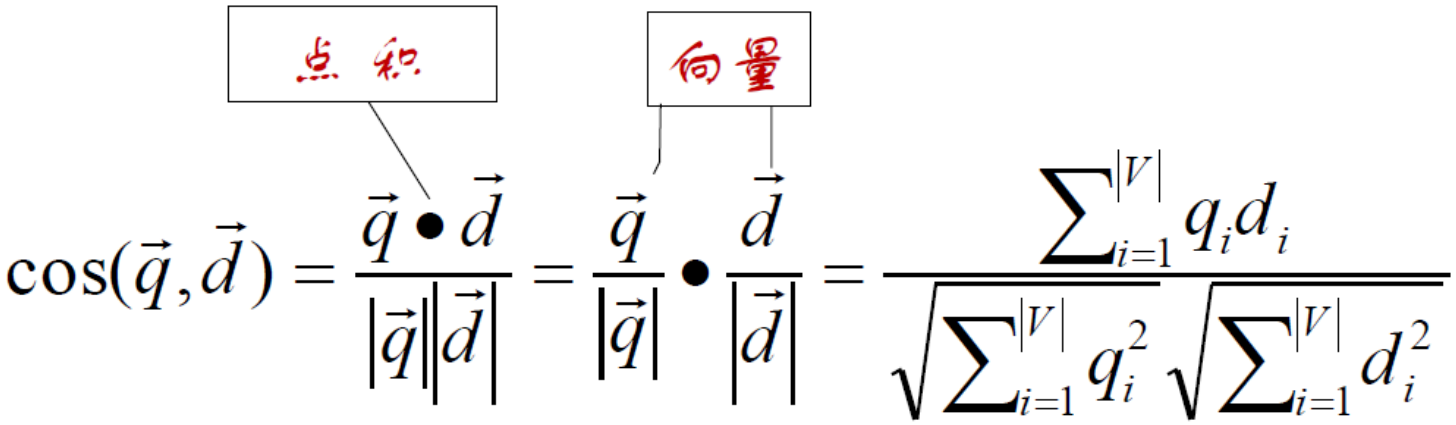
每篇文档表示成一个词频向量$\in \mathbb{N}^{|V|}$

二值 → 词频 → 权重矩阵

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . . .
ANTHONY	5.25	3.18	0.0	0.0	0.0	0.35
BRUTUS	1.21	6.10	0.0	1.0	0.0	0.0
CAESAR	8.59	2.54	0.0	1.51	0.25	0.0
CALPURNIA	0.0	1.54	0.0	0.0	0.0	0.0
CLEOPATRA	2.85	0.0	0.0	0.0	0.0	0.0
MERCY	1.51	0.0	1.90	0.12	5.25	0.88
WORSER	1.37	0.0	0.11	4.15	0.25	1.95
. . .						

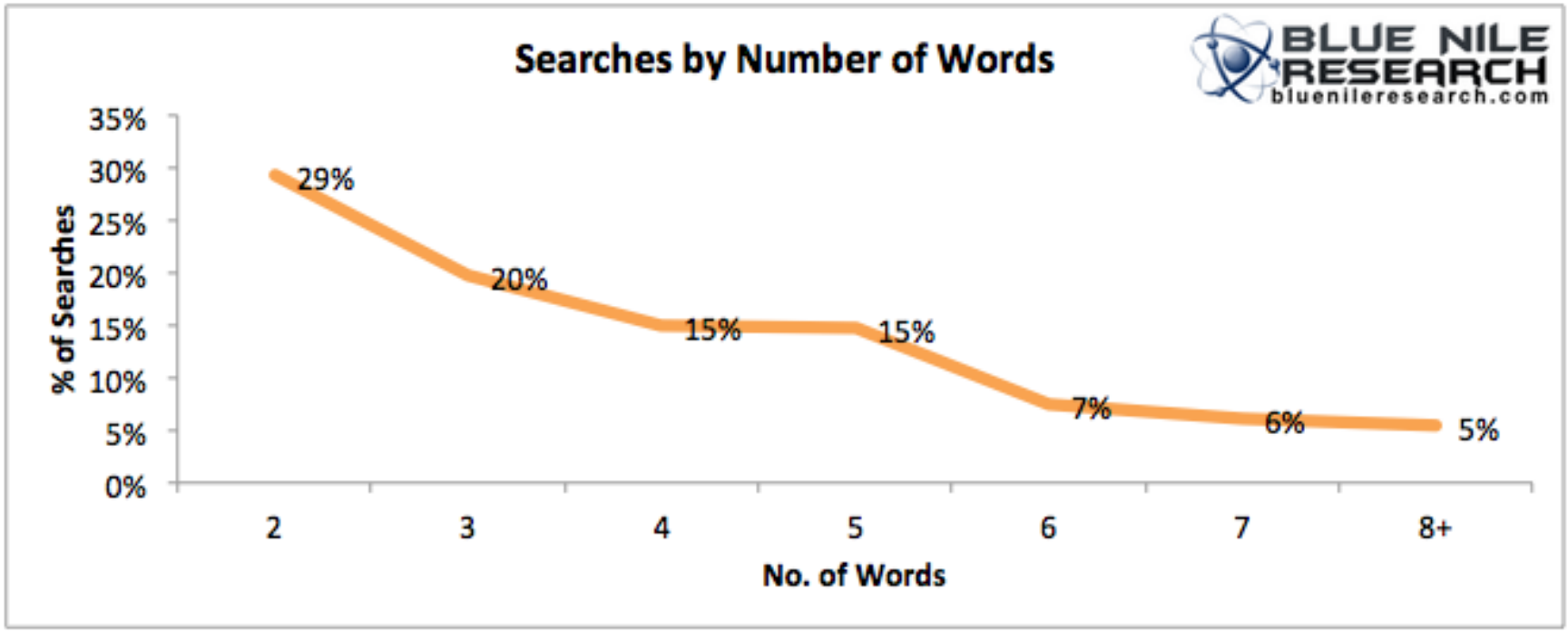
每篇文档表示成一个基于tf-idf权重的实值向量 $\in R^{|V|}$

回顾：查询和文档之间的余弦相似度计算

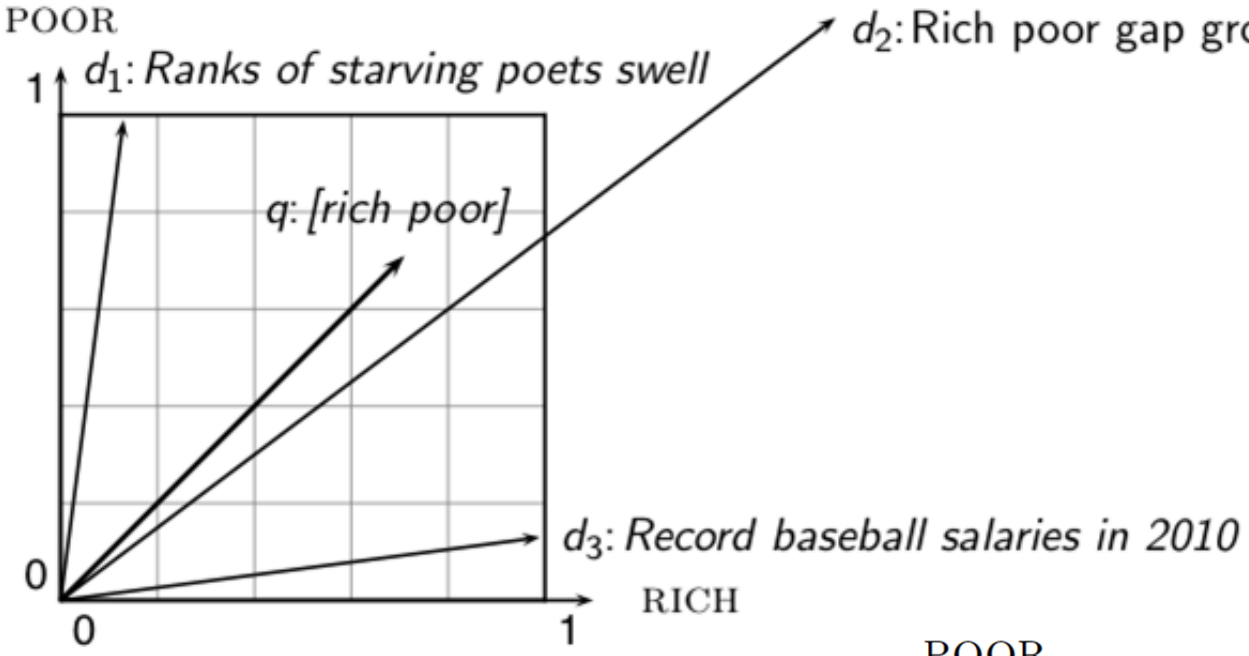


- q_i 是词项i在query中的tf-idf 权值
- d_i 是词项i在文档中的tf-idf 权值
- $\cos(\vec{q},\vec{d})$ $\vec{q}$与$\vec{d}$的余弦相关性
- 等价于向量$\vec{q}$与$\vec{d}$夹角的余弦值

查询表示为向量
只会有少数分量非零

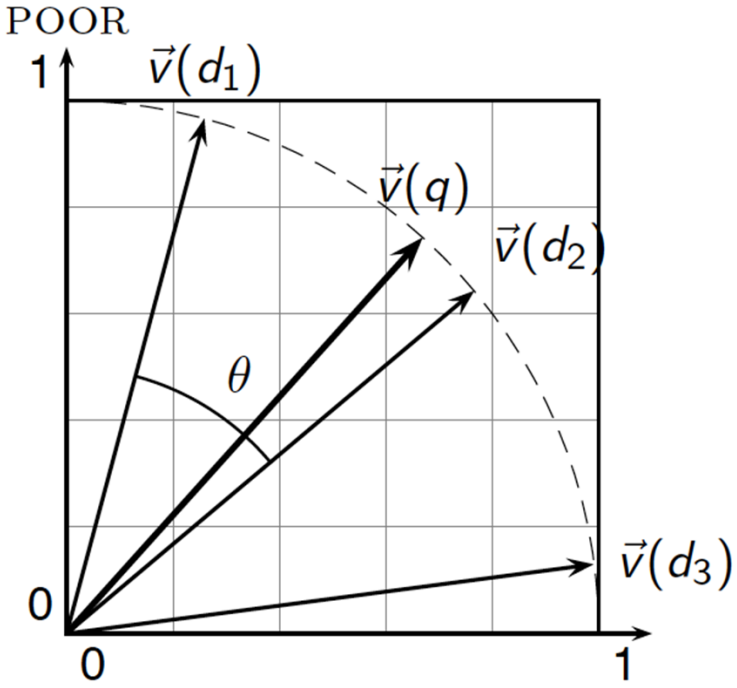


Research Reveals The Distinct Ways Users Search
http://searchengineland.com/research-reveals-distinct-ways-users-search-220977



长度归一化后，余弦的计算可直接通过点积的方式得到→

$$\cos(\vec{q},\vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$



文档长度的回转归一化

- 基于欧氏长度将每个文档向量归一化成单位向量，这样做会丢失原始的文档长度信息，也可能会隐藏长文档的一些细微性质：第一，由于长文档包含更多的词项数目，因此长文档中词项的频率tf 可能更高；第二，长文档可能包含更多的不同词项，即词汇量可能更大。这些因素会提高长文档的评分结果，这至少对某些信息需求来说是很不正常的。

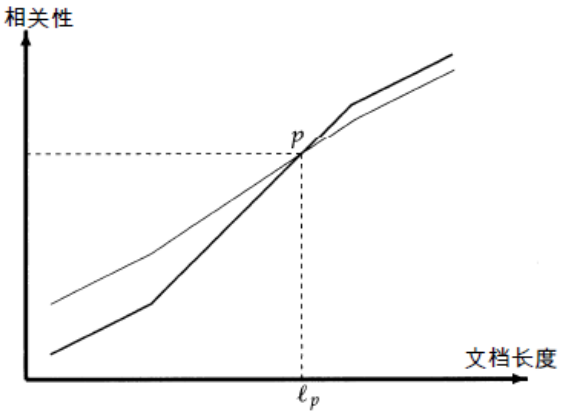


图 6-16 回转文档长度归一化

可以将相关概率（probability of relevance）看成文档长度的函数。

余弦归一化得到的计算结果的相关度和真实的相关度之间存在着差异。回转文档长度归一化的思路是，将余弦归一化结果曲线以p 点为轴逆时针旋转，使之能够和真实的基于文档长度的相关度曲线高度吻合。

文档权重和query权重（p89）

- 不同检索系统中的权重机制并不相同

- SMART标记：即对于每种不同的权重计算方法采用不同的标记。文档向量和query向量权重计算方法的组合字母表示为*ddd. qqq*

  - 例如： lnc. ltn
    - 文档：对数tf，无idf因子，余弦长度归一化
    - 查询：对数tf，idf，无归一化

回顾：tf-idf 权重机制变形

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log\frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0,\log\frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

例如： lnc. ltn

文档：对数tf，无idf因子，余弦长度归一化

查询：对数tf，idf，无归一化

Inc.ltc举例（p86）

- 文档： *car insurance auto insurance*
- Query： *best car insurance*

词项	查询						文档				内积
	tf-raw	tf-wt	df	idf	wt	n'lize	tf-raw	tf-wt	wt	n'lize	
auto	0	0	5000	2.3	0	0	1	1	1	0.52	0
best	1	1	50000	1.3	1.3	0.34	0	0	0	0	0
car	1	1	10000	2.0	2.0	0.52	1	1	1	0.52	0.27
insurance	1	1	1000	3.0	3.0	0.78	2	1.3	1.3	0.68	0.53

- tf-raw:未计算权重的词项频率
- tf-wt :采用了对数计算方法的词项频率
- n'lize：归一化后的权重

文档长度= $\sqrt{1^2+0^2+1^2+1.3^2} \approx 1.92$

Score = 0+0+0.27+0.53 = 0.8

提纲

1. 上一讲回顾
2. **结果排序的重要性**
3. 结果排序的实现
4. 完整的搜索系统

排序的重要性


alphago_百度搜索
alphago - 搜狗搜索
alphago - 必应
https://www.baidu.com/s?wd=alphago&rsv_spt=1&rsv_iqid=0xc48983720004ce8d&issp=1&f=8&rsv_bp=1&rsv_idx=2&ie=utf-8&tn=baiduh
Bai du 百度
alphago
百度一下
网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»
百度为您找到相关结果约2,220,000个
搜索工具
您可以仅查看：英文结果
AlphaGo出现漏洞 李世石扳回一局
热点
AlphaGo之父：李世石78步点中了电脑"bug"
人机大战连胜三场，AlphaGo技惊天下，令不少职业棋手大呼意外，而现在，它再次有了让人意外的表...详情>>
来源：新浪科技
图文回顾：谷歌PK李世石第四场
赛后复盘：AlphaGo疑似死机
柯洁约战AlphaGo：让风暴更猛
AlphaGo_百度百科
AlphaGo
阿尔法围棋（AlphaGo）是一款围棋人工智能程序，由位于英国伦敦的谷歌（Google）旗下DeepMind公司的戴维·西尔弗、艾佳·黄和戴密斯·哈萨比斯与他们的团队开发，这个程序利用...
程序原理 主要成绩
baike.baidu.com/
alphago的最新微博结果
网易新大话西游2 V：谷歌的阿尔法狗(AlphaGo)对战人类最强大脑,已经连下三城,如果它来大话参加比武大会,谁敢与之一战[偷乐] 查看全文>>
20分钟前 - 新浪微博 转发(0) | 评论(0)
香港经济日报 V：南韩围棋王李世乭对AlphaGo的人机战中,李在连输3场后,终
知名围棋高手
展开
李世石 获韩国围棋最优秀棋手
聂卫平 中国棋坛十大杰出人物
常昊 中国著名围棋九段棋手
曹薰铉 韩国第一个九段棋手
黑嘉嘉 围棋六段的混血美女
陈小匀 围棋5段的美女主播
唐莉 公认的围棋第一美女
棋坛小龙女 棋艺高超的清纯女孩
那些对抗人类智慧的机器人
深蓝 曾击败世界第一棋手
浪潮天梭 以一敌五的铁将
沃森 察言观色的全才学霸

排序的重要性


alphago_百度搜索
alphago - 搜狗搜索
alphago - 必应
https://www.sogou.com/web?query=alphago&ie=utf8&_ast=1457879168&_asf=null&w=01029901&p=40040100&dp=1&cid=&sut=1551&s
搜狗搜索
新闻 网页 微信 知乎 问问 图片 视频 地图 购物 更多
alphago
搜狗搜索
全部时间
李世石逆转胜AlphaGo 比分扳为3:1_凤凰网
阿尔法狗出现失误
也许这就是狗狗的弱点，看情况不好的时候，就会乱下。 详情>>
阿尔法三胜棋王 搜狗全员放假一天
AlphaGo表现出了哪些计算和智能
AlphaGo开发者：不存在秘密协议
AlphaGo之父：天才少年 象棋大师
李世石二战AlphaGo现场图
凤凰网 - www.ifeng.com - 2016-3-13
方寸之间，世纪之战！
谷歌人机大战
谷歌人机大战
直播时间 2016-03-13 12:00:00
直播内容 谷歌人机大战
直播热点 李世石人机大战
立即观看
阿尔法围棋 - 搜狗百科
阿尔法围棋（AlphaGo）是一款围棋人工智能程序，由位于英国伦敦的谷歌（Google）旗下DeepMind公司的戴维·西尔弗、艾佳·黄和杰米斯·哈萨比斯与他们的团队...
程序原理 - 主要成绩 - 挑战对象 - 比赛介绍
搜狗百科 - baike.sogou.com/v... - 2016-3-13 - 快照
谷歌人工智能破解围棋比赛_科技时代_新浪网
今日热点
1 新iPhone或发布 NEW
2 狗胜节放假
3 男女老少共洗泥浴 NEW
4 少女每天五孔流血 NEW
5 暴怒车主疯撞4S店 NEW
6 尸检男子突坐起 NEW
7 核电站因漏水关闭 NEW
8 宋仲基妹妹曝光
9 游客披棉被赏雪
10 王思聪雪梨吃火锅
围棋棋手
展开
李昌镐
古力
时越
吴清源
朴廷桓
罗洗河
常昊
陈耀烨

排序的重要性


alphago_百度搜索
alphago - 搜狗搜索
alphago - 必应
https://cn.bing.com/search?q=alphago&go=提交&qs=n&form=QBRE&pq=alphago&sc=8-5&sp=-1&sk=&
必应
alphago
Switch to Bing
网页 图片 视频 学术 词典 网典 地图 更多
提示:当前显示为 全部结果 | En 英文搜索 | 仅中文
AlphaGo - 话题精华 - 知乎
李巨擘能战胜 AlphaGo 吗？6276个最新问答，点击查看更多>> ... -"珍珠港遭到空袭！这不是演习！这不是演习！这不是演习！"——2016年1月28日凌晨-从昨晚开始，一条声称 ...
https://www.zhihu.com/topic/20038840 · 2 天前
Alphago的最新相关信息
哈撒比斯：李世石将AlphaGo推向极致 期待下一场 中国新闻网 · 3 小时前
3月13日，李世石与谷歌围棋AlphaGo人机五局大战第四局，继续在韩国首尔钟路区四季酒店进行。人类代表李世石在前三局比赛中0-3落后，实际已经提前宣告失利。按照双方赛前的约定，随后两局对局照常举行。今天的第四局比赛，李世石执白终于战胜AlphaGo...
神之一手！李世石首胜谷歌AlphaGo 腾讯科技 · 12 分钟前
AlphaGo创始人点评大战第四局：79手出错 87手才"意识"到 DoNews · 2 小时前
alphago_互动百科
谷歌旗下DeepMind公司的戴维·西尔弗、艾佳·黄和杰米斯·哈萨比斯与他们的团队，开发了一个叫"AlphaGo"的程序，利用"价值网络"去计算局面，用"策略网络"去选择下子。训 ...
www.baike.com/wiki/alphago 2016-1-28
AlphaGo - Wikipedia, the free encyclopedia 翻译此页
AlphaGo is a computer program developed by Google DeepMind to play the board game Go. In October 2015, it became the first computer Go program to beat a ...
https://en.wikipedia.org/wiki/AlphaGo 2016-3-10
AlphaGo开发者：人工智能将如何塑造未来|AlphaGo|人工智能 ...
导语：3月9日，谷歌人工智能AlphaGo战胜韩国棋手李世石引起巨
当前热搜
1 改变世界中国六大科技
2 阿尔法换张棋盘就不行了
3 微信确认将推企业版微信
4 中国电影票房超美被疑造假
5 个人二手房交易将缴增值税
6 赵薇醉酒主动献吻男子
7 全球最帅囚犯出狱了
8 全球游客最爱的二十大城市
相关搜索
alphago 棋谱
google alphago
alphago下载
alphago download
alphago 算法
alphago 李世石
alphago wikipedia
alphago挑战

排序的重要性


alphago 李世石_百度搜索
alphago 李世石 - 搜狗搜索
alphago 李世石 - 必应
https://www.baidu.com/s?wd=alphago%20李世石&rsv_spt=1&rsv_iqid=0xc48983720004ce8d&issp=1&f=8&rsv_bp=1&rsv_idx=2&ie=utf-8&
Bai du 百度
alphago 李世石
百度一下
网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»
百度为您找到相关结果约1,050,000个
搜索工具
AlphaGo出现漏洞 李世石扳回一局
热点
AlphaGo之父：李世石78步点中了电脑"bug"
人机大战连胜三场，AlphaGo技惊天下，令不少职业棋手大呼意外，而现在，它再次有了让人意外的表...详情>>
来源：新浪科技
图文回顾：谷歌PK李世石第四场
赛后复盘：AlphaGo疑似死机
柯洁约战AlphaGo：让风暴更猛
李世石:AlphaGo执黑有缺陷 下场我要执黑 新浪体育 3小时前
李世石出现"神之一手" 第四局战胜AlphaGo 环球网 2小时前
柯洁:李世石发挥完美 自己对AlphaGo胜率六到七成 网易体育 3小时前
AlphaGo之父:李世石将程序推向极致 难解释崩盘 搜狐体育 3小时前
柯洁:李世石赢得解气!今天的AlphaGo挑我不够格 凤凰科技 4小时前
alphago 李世石的相关视频 在线观看_百度视频
爱奇艺 03:15
酷6 09:09
爱奇艺 11:52
PPTV 00:31
人机大战：李世石变身黑客Alp...
机器赢了！AlphaGo首局战胜李...
谷歌AlphaGo终败北 李世石背...
围棋人机战第二场：李世石再...
查看全部338个相关视频>>
v.baidu.com/
alphago 李世石的最新微博结果
张向东 V：不管是AlphaGo赢,还是李世石赢,最后赢的,都是人性中美好智慧的
知名围棋高手
展开
李世石
获韩国围棋最优秀棋手
聂卫平
中国棋坛十大杰出人物
常昊
中国著名围棋九段棋手
曹薰铉
韩国第一个九段棋手
黑嘉嘉
围棋六段的混血美女
陈小匀
围棋5段的美女主播
唐莉
公认的围棋第一美女
棋坛小龙女
棋艺高超的清纯女孩
那些对抗人类智慧的机器人
深蓝
曾击败世界第一棋手
浪潮天梭
以一敌五的铁将
沃森
察言观色的全才学霸
阿尔法围棋
首度不被让子战胜人类
搜索热点
排名
搜索指数
1 张惠妹获男友告白 新 178575

排序的重要性


alphago 李世石_百度搜索
alphago 李世石 - 搜狗搜索
alphago 李世石 - 必应
https://www.sogou.com/web?query=alphago+李世石&ie=utf8&_ast=1457879296&_asf=null&w=01029901&p=40040100&dp=1&cid=&cid=
搜狗搜索
新闻 网页 微信 知乎 问问 图片 视频 地图 购物 更多
alphago 李世石
搜狗搜索
全部时间
李世石逆转胜AlphaGo 比分扳为3:1 凤凰网
阿尔法狗出现失误
也许这就是狗狗的弱点，看情况不好的时候，就会乱下。 详情>>
阿尔法三胜棋王 搜狗全员放假一天
AlphaGo表现出了哪些计算和智能
AlphaGo开发者：不存在秘密协议
AlphaGo之父：天才少年 象棋大师
李世石二战AlphaGo现场图
凤凰网 - www.ifeng.com - 2016-3-13
方寸之间，世纪之战！
谷歌人机大战
谷歌人机大战
直播时间 2016-03-13 12:00:00
直播内容 谷歌人机大战
直播热点 李世石人机大战
立即观看
今日热点
1 新iPhone或发布 NEW
2 狗胜节放假
3 男女老少共洗泥浴 NEW
4 少女每天五孔流血 NEW
5 暴怒车主疯撞4S店 NEW
6 尸检男子突坐起 NEW
7 核电站因霜水关闭 NEW
8 宋仲基妹妹曝光
9 游客披棉被赏雪
10 王思聪雪梨吃火锅
alphago 李世石的最新相关信息
三大原因促李世石首胜 ALphaGo算法决定其弱点-搜...
搜狐体育 16分钟前
三大原因促李世石首胜 ALphaGo算法决定其弱点 谷歌的"深思"团队领头人哈萨比斯第四局赛后在脸书上发文，称到79手的时候阿尔法还认为自己的胜率有70%，到了89手突...
AlphaGo 对李世石"放水"？棋手说不可能！ 网易数码 2小时前
神之一手！李世石首胜谷歌AlphaGo 站长之家 2小时前
李世石：AlphaGo执黑有缺陷 下场我要执黑_棋牌 新浪体育 3小时前
孔杰：AlphaGo或有致命缺陷 可惜李世石发现太晚！... 凤凰科技 5小时前
alphago 李世石在线观看-搜狗影视
高清 搜狐视频 1:56 世纪围棋人机大战 al
高清 搜狐视频 1:13 "围棋人机大战"大局
高清 搜狐视频 00:44 人机对弈 alphago三
高清 搜狐视频 3:05 全球聚焦:李世石大战
著名棋手 展开
李世石
李昌镐
古力
元晟溱
围棋比赛 展开
十番棋
三星杯世界围棋公开赛
Chunlan 春兰
春兰杯世界职业围棋锦标赛
古李十番棋
其他人还搜 展开

排序的重要性


alphago 李世石_百度搜索
alphago 李世石 - 搜狗搜索
alphago 李世石 - 必应
https://cn.bing.com/search?q=alphago+李世石&go=提交&qs=n&form=QBRE&pq=alphago+李世石&sc=0-8&sp=-1&sk=&cvid=60E
必应
alphago 李世石
Switch to Bing in English
登录
网页 图片 视频 学术 词典 网典 地图 更多
Alphago 李世石的最新相关信息
哈撒比斯：李世石将AlphaGo推向极致 期待下一场
中国新闻网 · 4 小时前
3月13日，李世石与谷歌围棋AlphaGo人机五局大战第四局，继续在韩国首尔钟路区四季酒店进行。人类代表李世石在前 ...
AlphaGo 对李世石"放水"？专业棋手说不可能！
雷锋网 · 3 小时前
就在舆论普遍认为李世石将会继续败北的时候，人机大战的第四场，李世石出人意料地挽回了胜局，将比分改写为"3:1"。 ...
李世石：AlphaGo执黑有缺陷 下场我要执黑
海外网 · 2 小时前
新浪科技讯 3月13日下午消息，人机围棋对决今日进行第四场比赛，李世石九段凭借第78"神之一手"让AlphaGo陷入混乱，最终将AlphaGo首次拖 ...
人机大战现场-AlphaGo连扳 李世石退让诱敌深入_棋牌_新浪 ...
新浪体育讯 北京时间3月13日，李世石与谷歌AlphaGo五番棋人机对决第4局即将在韩国四季酒店打响。新浪体育首尔报道团为您带来现场第 ...
sports.sina.com.cn/go/2016-03-13/doc-ifxqhm... 9 小时前
3：0！谷歌AlphaGo再赢李世石 人类终败北_科技_腾讯网
3：0！谷歌AlphaGo再赢李世石 人类终败北 世界围棋冠军李世石遭遇三连败，五番棋比赛中谷歌AlphaGo已赢得胜利。
tech.qq.com/a/20160312/034432.htm 1 天前
人工智能攻克围棋！AlphaGo三比零胜李世石|人工智能 ...
图文直播全程回顾新浪科技讯北京时间3月12日下午消息，谷歌人工智能AlphaGo与韩国棋手李世石今日进行了第三场较量，最终AlphaGo战胜 ...
tech.sina.com.cn/it/2016-03-12/doc-ifxqhmvc2363285.shtml 1 天前
围棋棋手
李昌镐 聂卫平 小林光一 范廷钰 江铸久
当前热搜
1 改变世界中国六大科技
2 阿尔法换张棋盘就不行了
3 微信确认将推企业版微信
4 中国电影票房超美被疑造假
5 个人二手房交易将缴增值税
6 赵薇醉酒主动献吻男子
7 全球最帅囚犯出狱了
8 全球游客最爱的二十大城市
相关搜索
千世石缘
中国世石
李世石女儿李慧琳
三生三世石
魅世红颜文 三世石
魅世红颜 三世石
Nature AlphaGo
Google Go AI

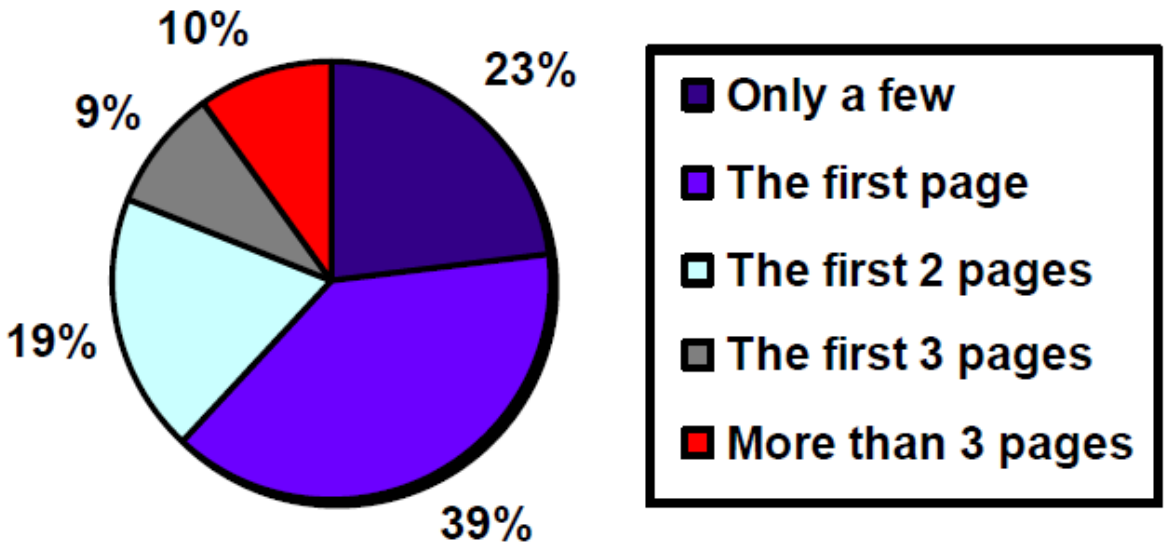
排序的重要性

- 上一讲：不排序的问题严重性
  - 用户只希望看到一些而不是成千上万的结果
  - 很难构造只产生一些结果的查询
  - 即使是专家也很难
  - → 排序能够将成千上万条结果缩减至几条结果，因此非常重要
- 实际上，大部分用户只看1到3条结果

用户浏览的页数

iProspect Search Engine User Behavior (2006)

"When you perform a search on a search engine and are looking over the results, approximately how many entries do you typically review before clicking one? (Select One)"



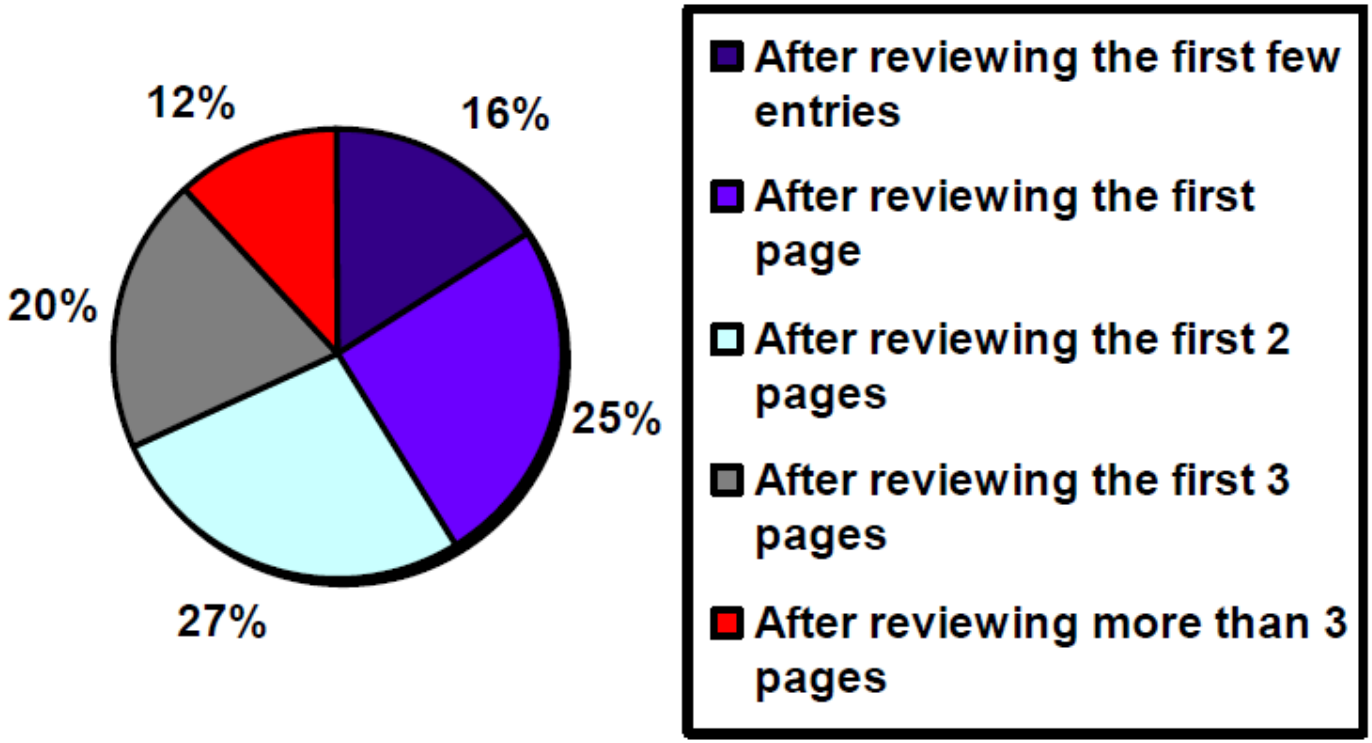
90%的用户不会浏览搜索结果页三页之后的页面，而62%的用户只会关注搜索结果页第一页的内容。

http://district4.extension.ifas.ufl.edu/Tech/TechPubs/WhitePaper_2006_SearchEngineUserBehavior.pdf

用户的耐心

iProspect Search Engine User Behavior (2006)

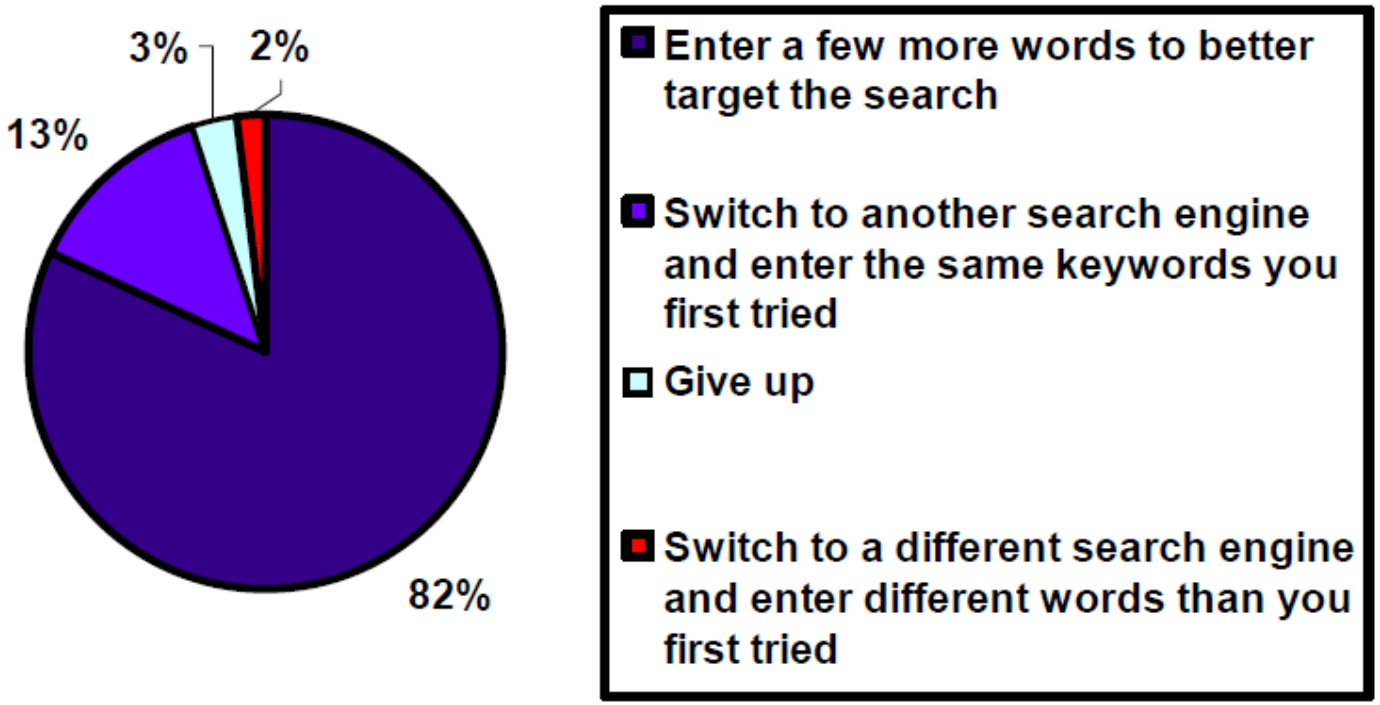
"When you perform a search on a search engine and don't find what you are looking for, at what point do you typically either revise your search, or move on to another search engine? (Select one)"



修改查询词、换搜索引擎、放弃

iProspect Search Engine User Behavior (2006)

"When you perform a search on a search engine and don't find what you are looking for, what are you typically more likely to do? (Select one)"



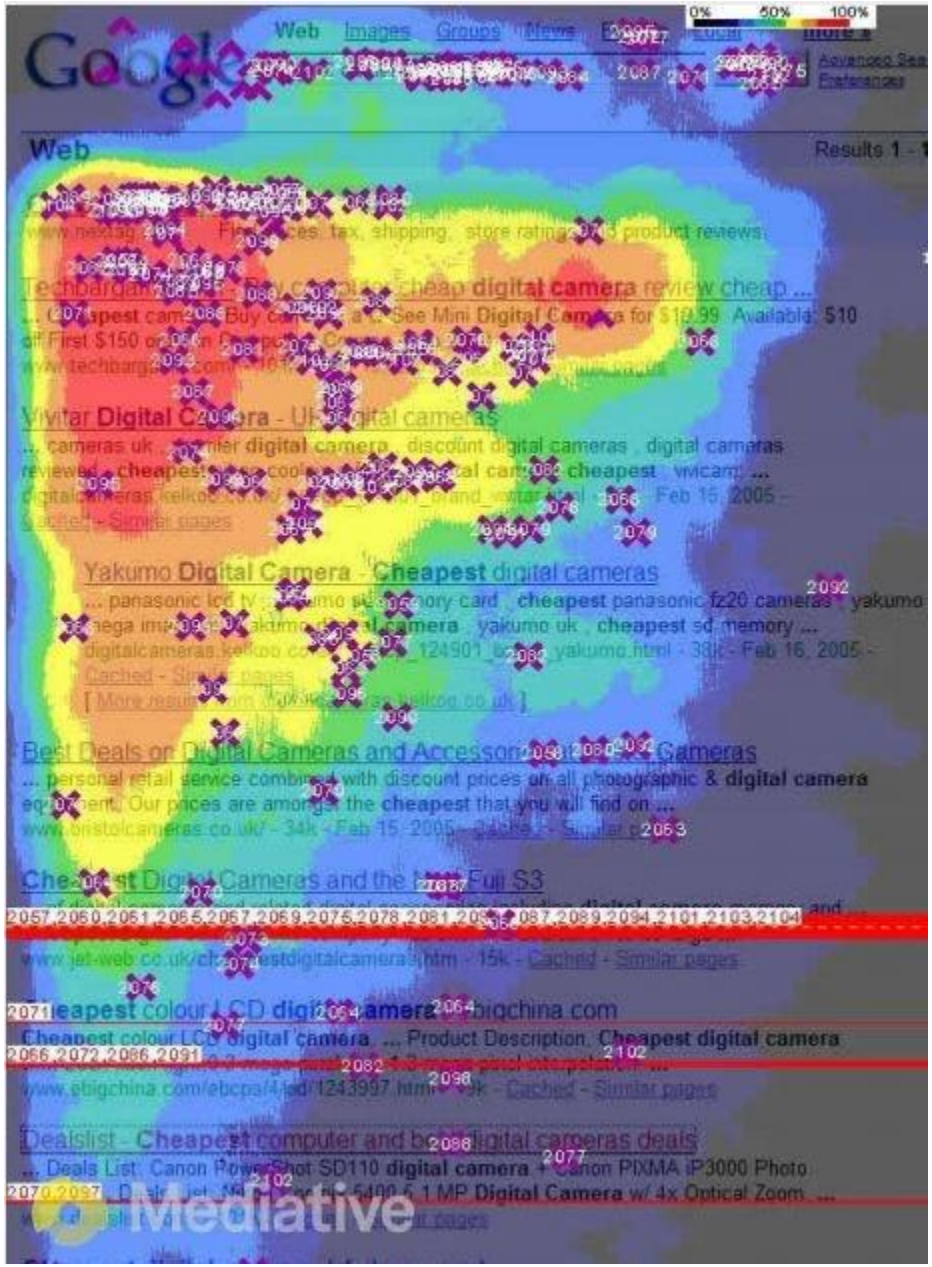
十年来用户点击习惯的变化

the number one listing will still get more clicks, but the percentage is getting more evenly distributed between spots two to four.

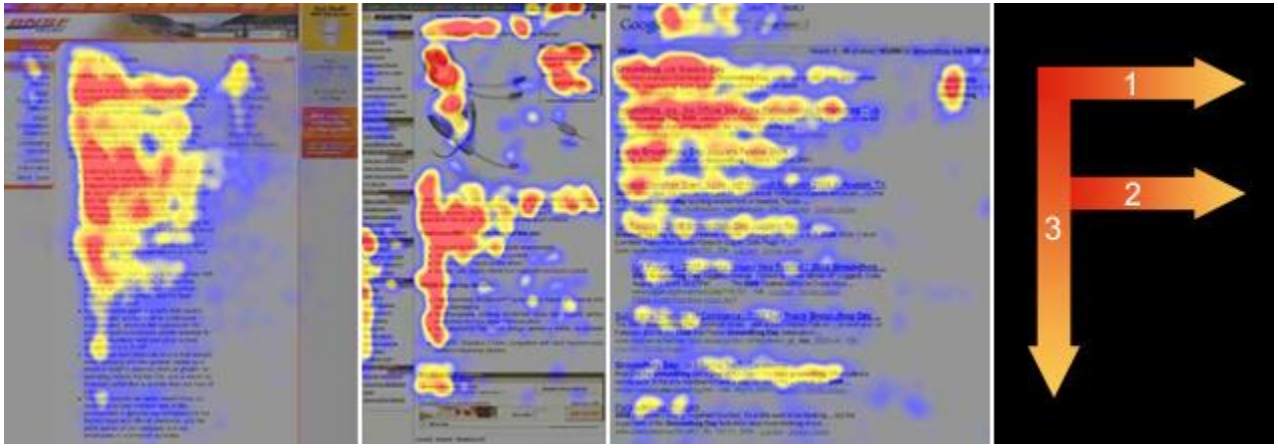
The Evolution of SERPs and User Behaviors
https://searchenginewatch.com/sew/how-to/2374414/the-evolution-of-serps-and-user-behaviors
The evolution of Google search results pages & their effect on user behaviour
http://120.52.72.39/storage.pardot.com/c3pr90ntcsf0/22222/39230/The_Evolution_of_Google_Search_Results_Pages_and_Their_Effect_on_User_Behaviour_New_.pdf

Golden Triangle

In 2005, if you weren't in the top three listings, you probably weren't getting clicks. Users didn't scroll. They trusted that whatever Google was showing them as first was the best possible option.



Jakbo Nielsen的F形浏览热区



Jakbo Nielsen曾对232位用户浏览几千个页面的过程中的眼动情况进行追踪，发现用户在不同站点上的浏览行为有明显的一致性，将浏览热点可视化后呈现出类似F形的图案。这种浏览行为有三个特征：

1. 用户首先会在内容区的上部进行横向浏览。
2. 用户视线下移一段距离后在小范围内再次横向浏览。
3. 最后用户会在内容区的左侧做快速的纵向浏览。

2013年中国搜索引擎用户搜索体验分析

iResearch 艾瑞咨询集团

中国搜索引擎页面主要广告位

搜索引擎页面主要有六个广告位，分别对应于右侧图片中的1号位置到6号位置，各广告在页面中的具体位置为：

1号位置：位于页面顶部的广告位；

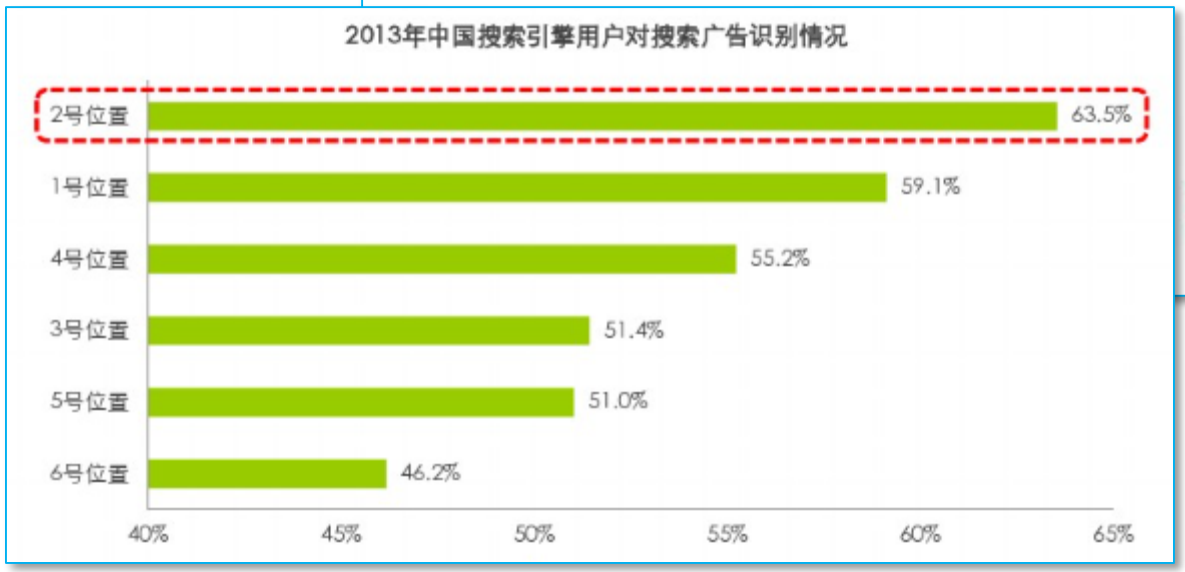
2号位置：位于页面右侧顶部的广告位；

3号位置：位于页面中间偏上位置的广告位；

4号位置：位于页面右侧中间位置的广告位；

5号位置：位于页面中部的广告位；

6号位置：位于页面底部的广告位；



排序的重要性: 小结

- 摘要阅读(Viewing abstracts): 用户更可能阅读前几页(1, 2, 3, 4)的结果的摘要
- 点击(Clicking): 点击的分布甚至更有偏向性
  - 一半情况下，用户点击排名最高的页面
  - 即使排名最高的页面不相关，仍然有30%的用户会点击它
- → 正确排序相当重要
- → 把最相关的页面放在首页非常重要

提纲

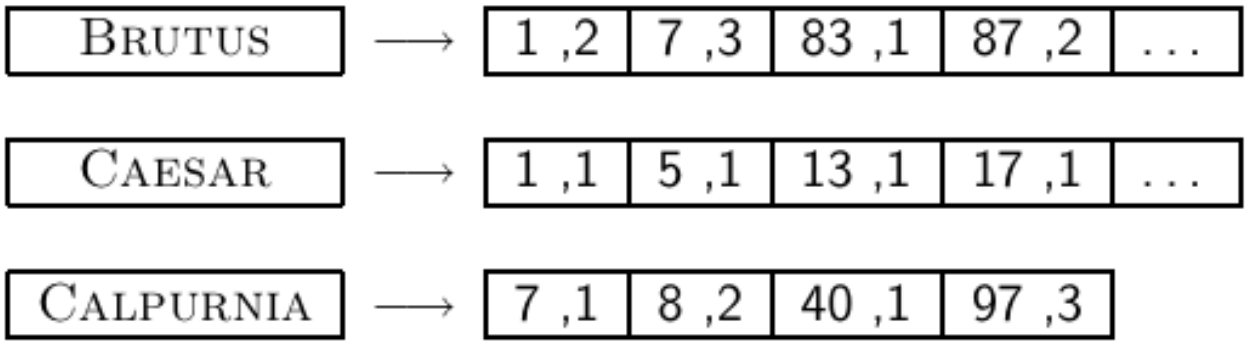
1. 上一讲回顾
2. 结果排序的重要性
3. 结果排序的实现
   - 精确top K 检索及其加速办法
   - 非精确top K检索
4. 完整的搜索系统

精确top K 检索及其加速办法

- 方法一：快速计算余弦
- 方法二：堆排序法N中选K
- 方法三：提前终止计算

词项频率tf和文档频率idf的存储

- 词典中保存每个词的idf值
- 词项频率tf存入倒排索引中



  - 当然也需要位置信息，上面没显示出来

倒排索引中的词项频率存储

- 每条倒排记录中，除了docID_d 还要存储$\mathrm{tf}_{t,d}$
- 通常存储是原始的整数词频，而不是对数词频对应的实数值
- 这是因为取对数后的实数值不易压缩
- 对tf采用一元码编码效率很高
- 总体而言，额外存储tf所需要的开销不是很大：采用位编码压缩方式，每条倒排记录增加不到一个字节的存储量
- 或者在可变字节码方式下每条倒排记录额外需要一个字节即可

余弦相似度计算算法

```
CosineScore(q)
 1  float Scores[N] = 0
 2  float Length[N]
 3  for each query term t
 4  do calculate w_{t,q} and fetch postings list for t
 5      for each pair(d, tf_{t,d}) in postings list
 6      do Scores[d] += w_{t,d} × w_{t,q}
 7  Read the array Length
 8  for each d
 9  do Scores[d] = Scores[d]/Length[d]
10  return Top K components of Scores[]
```

能不能加快？

精确top K 检索及其加速办法

- 目标：从文档集的所有文档中找出K 个离查询最近的文档
- (一般)步骤：对每个文档评分(余弦相似度)，按照评分高低排序，选出前K个结果
- 如何加速：
  - 思路一：加快每个余弦相似度的计算
  - 思路二：不对所有文档的评分结果排序而直接选出Top K篇文档
  - 思路三：能否不需要计算所有N篇文档的得分？

精确top K检索加速方法一：快速计算余弦

- 检索排序就是找查询的K近邻
- 一般而言，在高维空间下，计算余弦相似度没有很高效的方法
- 但是如果查询很短，是有一定办法加速计算的，而且普通的索引能够支持这种快速计算

特例– 不考虑查询词项的权重

- 查询词项无权重
  - 相当于假设每个查询词项都出现1次

- 于是，不需要对查询向量进行归一化
  - 可以对上一讲给出的余弦相似度计算算法进行轻微的简化

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\bullet\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

快速余弦相似度计算: 无权重查询

for each pair$(d, \text{tf}_{t,d})$ in postings list
do $Scores[d]$ += $\text{wf}_{t,d} \times \text{w}_{t,q}$

FastCosineScore(q)

```
1   float Scores[N] = 0
2   for each d
3   do Initialize Length[d] to the length of doc d
4   for each query term t
5   do calculate w_{t,q} and fetch postings list for t
6      for each pair(d, tf_{t,d}) in postings list
7      do add wf_{t,d} to Scores[d]
8   Read the array Length[d]
9   for each d
10  do Divide Scores[d] by Length[d]
11  return Top K components of Scores[]
```

Figure 7.1 A faster algorithm for vector space scores.

不考虑查询中查询词的权重

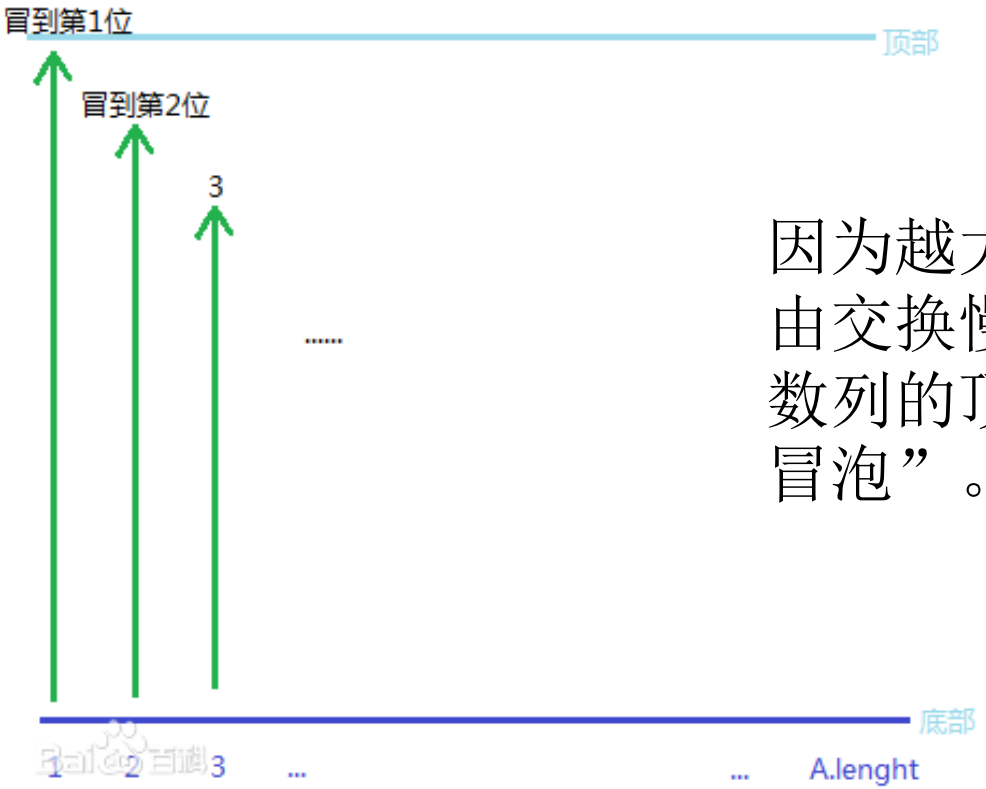
精确top k检索加速方法二：堆排序法N中选K

- 检索时，通常只需要返回前K条结果
  - 可以对所有的文档评分后排序，选出前K个结果，但是这个排序过程可以选用快速算法

- 令 J = 具有非零余弦相似度值的文档数目
  - 从J中选K个最大的

冒泡排序（Bubble Sort）

重复地走访过要排序的数列，一次比较两个元素，如果他们的顺序错误就把他们交换过来。走访数列的工作是重复地进行直到没有再需要交换，也就是说该数列已经排序完成。



因为越大的元素会经由交换慢慢“浮”到数列的顶端，故名“冒泡”。

冒泡排序的计算量

若文件的初始状态是正序的，一趟扫描即可完成排序。所需的关键字比较次数C和记录移动次数M均达到最小值：$C_{min}=n-2$，$M_{min}=0$。所以，冒泡排序最好的时间复杂度为$O(n)$。

若初始文件是反序的，需要进行$n-1$趟排序。每趟排序要进行$n-i$次关键字的比较($1\le i\le n-1$)，且每次比较都必须移动记录三次来达到交换记录位置。在这种情况下，比较和移动次数均达到最大值：

$$C_{\max}=\frac{n(n-1)}{2}=O(n^2)$$

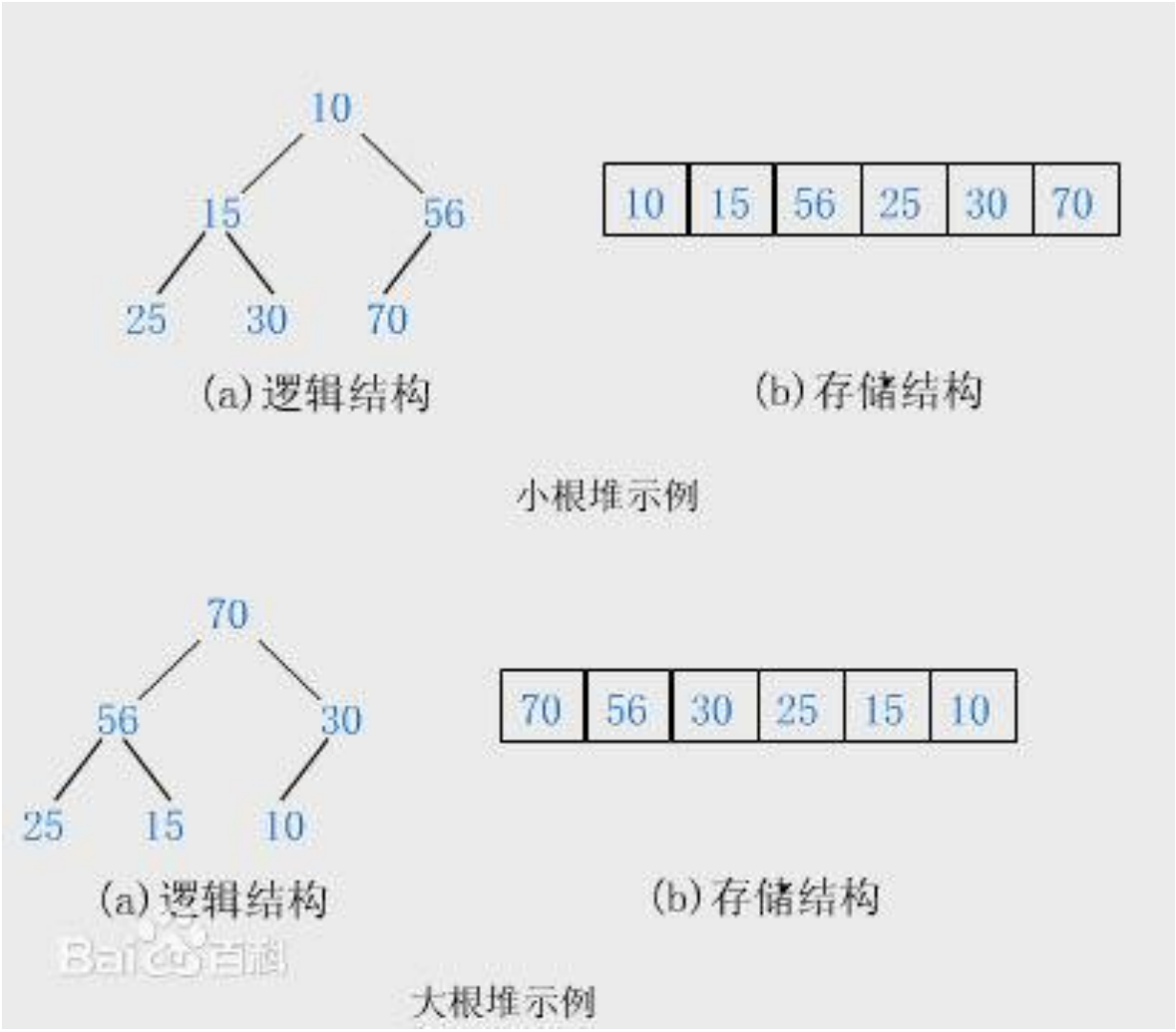
$$M_{\max}=\frac{3n(n-1)}{2}=O(n^2)$$

堆排序法

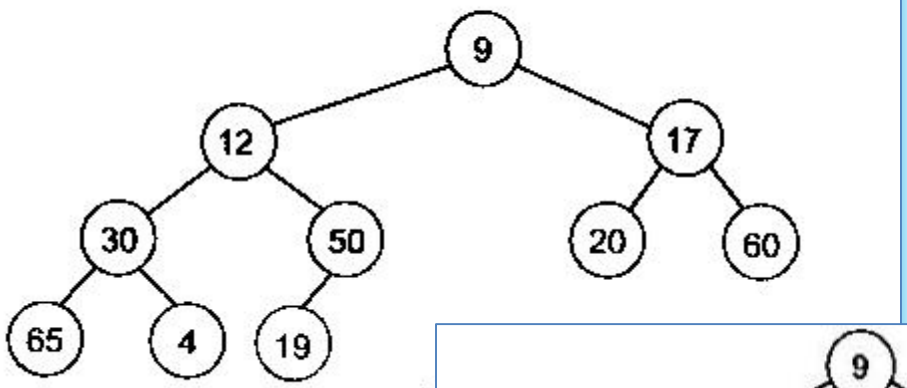
- **堆排序法**
  - 1991年的计算机先驱奖获得者、斯坦福大学计算机科学系教授罗伯特·弗洛伊德(Robert W．Floyd）和威廉姆斯(J．Williams）1964年共同发明了堆排序算法（ Heap Sort)
- **小根堆，大根堆**
  - n个关键字序列Kl，K2，…，Kn称为（Heap），当且仅当该序列满足如下性质（简称为堆性质）：ki<=k(2i）且ki<=k(2i+1)(1≤i≤ n/2），这是小根堆，大根堆则换成>=号。
  - 大根堆和小根堆：根结点（亦称为堆顶）的关键字是堆里所有结点关键字中最小者的堆称为小根堆，又称最小堆。根结点（亦称为堆顶）的关键字是堆里所有结点关键字中最大者，称为大根堆，又称最大堆。

堆排序法:小根堆，大根堆

堆可以被看成是一棵树，结点在堆中的**高度**可以被定义为从本结点到叶子结点的最长简单下降路径上边的数目；定义堆的高度为树根的高度。

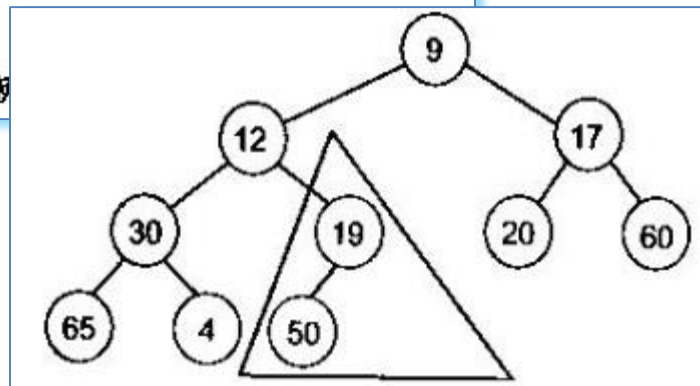


int A[0] = {9,12,17,30,50,20,60,65,4,49};

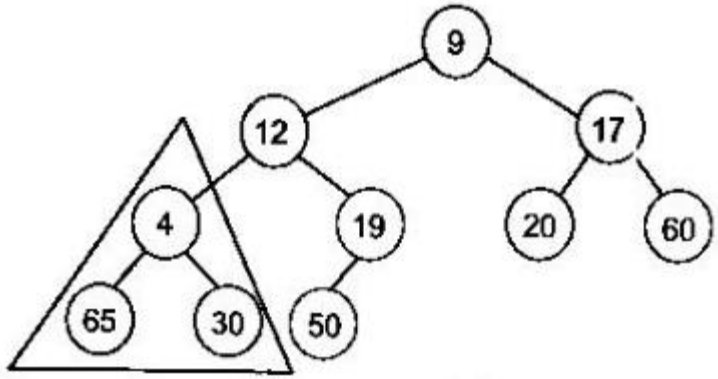


对叶子结点来说，可认为它已经是一个合法的堆了即20，60， 65， 4， 49都分别是一个合法的堆。故从A[4]=50开始向下调整；再取A[3]=30，A[2] = 17，A[1] = 12，A[0] = 9分别作一次向下调整操作就可以了。

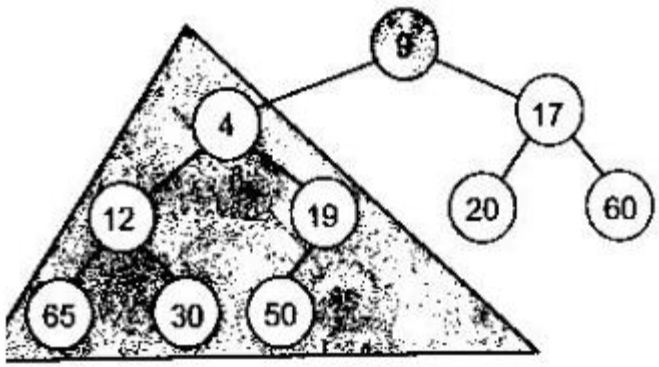
堆排序的时间复杂度为O(N*logN)



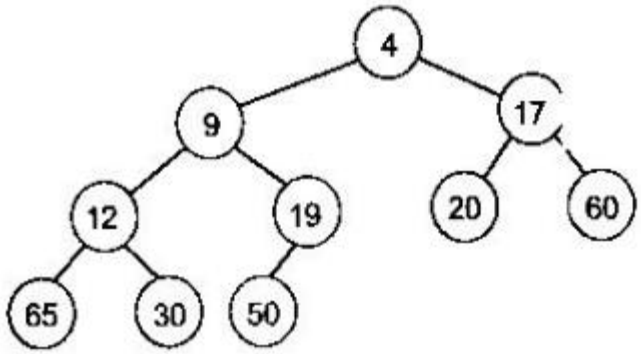
(A) FilterDown(4)放置50



(B) FilterDown(3)放置30



(C)



(D)

小结：堆排序

- 特点
  - 堆排序其实也是一种选择排序，是一种树形选择排序。只不过直接选择排序中，为了从R[1...n]中选择最大记录，需比较n-1次，然后从R[1...n-2]中选择最大记录需比较n-2次。事实上这n-2次比较中有很多已经在前面的n-1次比较中已经做过，而树形选择排序恰好利用树形的特点保存了部分前面的比较结果，因此可以减少比较次数。
- 运算量
  - 对于n个关键字序列，最坏情况下每个节点需比较log2(n)次，因此其最坏情况下时间复杂度为nlogn。

冒泡排序（Bubble Sort），时间复杂度为O(n^2)，空间复杂度为O(1)，即就地排序

精确top K 检索加速方法三：提前终止计算

- 到目前为止的倒排记录表都按照docID排序
- 接下来将采用与查询无关的另外一种反映结果好坏程度的指标(静态质量)
  - 例如：页面d的PageRank g(d)，就是度量有多少好页面指向d的一种指标（参考第 21章）
  - 于是可以将文档按照PageRank排序 g(d1) > g(d2) > g(d3) > . . .
  - 将PageRank和余弦相似度线性组合得到文档的最后得分

    net-score(q, d) = g(d) + cos(q, d)

提前终止计算

- 假设:
  - (i) g → [0, 1];
  - (ii) 检索算法按照d1,d2,…, 依次计算(为文档为单位的计算, document-at-a-time), 当前处理的文档的g(d) < 0.1;
  - (iii) 而目前找到的top K 的得分中最小的都 > 1.2
- 由于后续文档的得分不可能超过1.1 (cos(q,d) <1)
- 所以, 我们已经得到了top K结果, 不需要再进行后续计算

小结：精确top K 检索及其加速

- 方法一：快速计算余弦
  - 无权重
- 方法二：堆排序法N中选K
  - 堆构建：需要 2J 次操作(J = 具有非零余弦相似度值的文档数目)
  - 选出前K个结果：每个结果需要2log J 步
- 方法三：提前终止计算
  - net-score(q, d) = g(d) + cos(q, d)

提纲

1. 上一讲回顾
2. 结果排序的重要性
3. 结果排序的实现
   - 精确top K 检索及其加速办法
   - 非精确top K检索
4. 完整的搜索系统

非精确top K检索

- 策略一：索引去除(Index elimination)
- 策略二：胜者表
- 策略三：静态得分
- 策略四：影响度排序
- 策略五：簇剪枝方法——预处理
- 策略六：参数化索引以及域索引
- 策略七：层次索引

精确top K检索的问题

- 仍然无法避免大量文档参与计算

- 一个自然而言的问题就是能否尽量减少参与计算文档数目，即使不能完全保证正确性也在所不惜。
  - 即采用这种方法得到的top K虽然接近但是并非真正的top K----非精确top K检索

非精确top K检索的可行性

- 检索是为了得到与查询匹配的结果，该结果要让用户满意
- 余弦相似度是刻画用户满意度的一种方法
- 非精确top K的结果如果和精确top K的结果相似度相差不大，应该也能让用户满意

一般思路

- **找一个文档集合A，K<|A|<<N，利用A中的top K结果代替整个文档集的top K结果**
  - 即给定查询后，A是整个文档集上**近似剪枝**得到的结果
  - 上述思路不仅适用于**余弦相似度**得分，也适用于其他相似度计算方法

策略一：索引去除(Index elimination)

- 对于一个包含多个词项的查询来说，很显然我们可以仅仅考虑那些至少包含一个查询词项的文档

- 可以进一步拓展这种思路
  - 只考虑那些词项的idf值超过一定阈值的文档
  - 只考虑包含多个查询词项（一个特例是包含全部查询词项）的文档

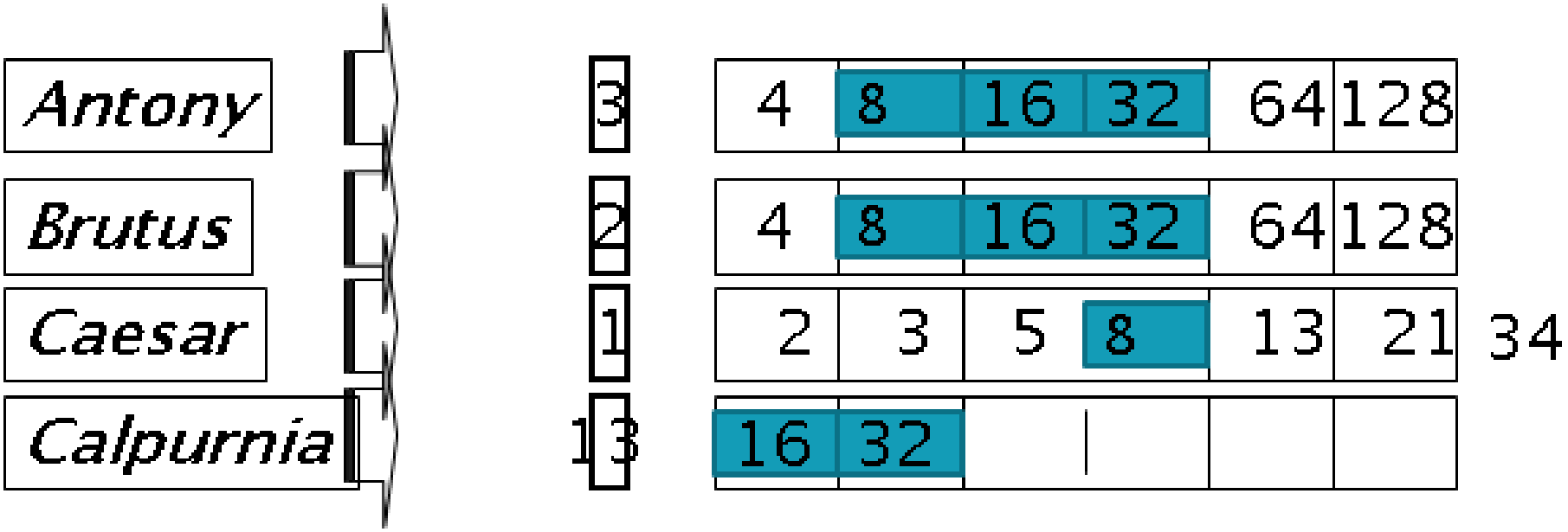
词项的idf 值超过一定阈值的文档

- 在查询*catcher in the rye* 时
- 只有*catcher* 和*rye* 的倒排记录表才会被遍历
- 显而易见：*in* 和*the* 的作用很小
- 优点：
  - 含有低idf值的词项的文档非常多，采用这种方法可以将大量无关的文档从候选集合*A*中去除

包含多个查询词项的文档

- 那些至少包含一个查询词项的文档才有可能成为候选文档
- 对于多词项查询，只考虑那些包含较多查询词项的文档
  - 比如，至少含有超过3/4的查询词项
- 可以在倒排记录表遍历过程中实现

4中含3



仅对文档8、16和 32进行计算

策略二：胜者表

- 对于词典中的每个词项t，预先计算出**r 个最高权重的文档**
  - 词项t 所对应的tf 值最高的**r 篇文档构成t 的胜者表**
  - 也称为**优胜表（fancy list）或高分文档（top doc）**
- 其中r 的值需要在索引建立之时给定
  - 因此，有可能出现$r < K$ 的情况
- 给定查询 q，对查询q 中所有词项的胜者表求并集，并可以生成集合A
  - 根据余弦相似度大小从A中选取前top K 个文档

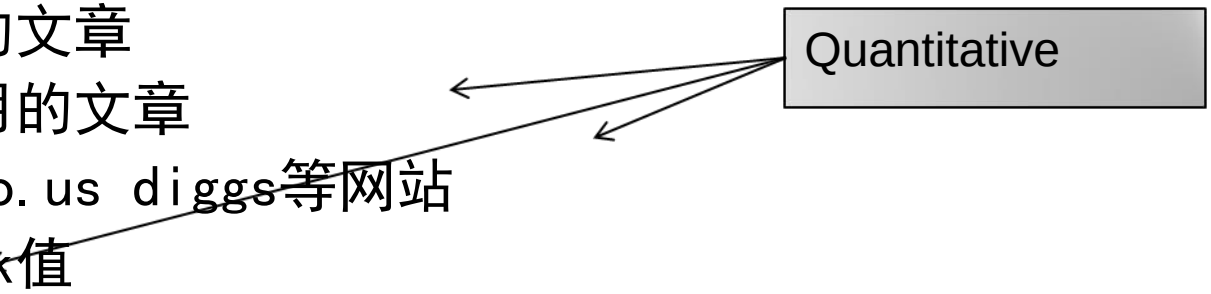
课堂思考

- 胜者表方式和前面的索引去除方式有什么关联？如何融合它们？

- 如何在一个倒排索引当中实现胜者表？
  - 提醒：胜者表与docID大小无关

策略三：静态得分

- 我们希望排序靠前的文档既是相关的又是权威的
- 相关性通过余弦相似度得分来判断
- 权威性是与query无关的文档本身的属性决定的
- 权威性标志举例
  - 维基百科
  - 报纸上的文章
  - 很多引用的文章
  - del.icio.us diggs等网站
  - Pagerank值



权威度计算

- 为每篇文档赋予一个与查询无关的(query-independent) [0,1]之间的值，记为g(d)

- 同前面一样，最终文档排名基于g(d)和相关度的线性组合。
  - **net-score(q,d) = g(d) + cosine(q,d)**
  - 可以采用等权重，也可以采用不同权重
  - 可以采用任何形式的函数，而不只是线性函数

- 接下来我们的目标是找net-score最高的top K文档（非精确检索）

基于net-score的Top K文档检索

- 首先按照g(d)从高到低将倒排记录表进行排序
- 该排序对所有倒排记录表都是一致的(只与文档本身有关)
- 因此，可以并行遍历不同查询词项的倒排记录表来
  - 进行倒排记录表的合并
  - 余弦相似度的计算

利用g(d)排序的优点

- 这种排序下，高分文档更可能在倒排记录表遍历的前期出现

- 在时间受限的应用当中（比如，任意搜索需要在50ms内返回结果），上述方式可以提前结束倒排记录表的遍历

全局胜者表

- 对于选择好的r 值，对每个词项t 构建一个全局胜者表
- 其中包含了$g(d)$ + tf-idf_{td}得分最高的r 篇文档
- 当查询提交以后，对所有全局胜者表的并集中的文档计算其最后得分
- 根据最终得分选择top K

高端表(High list)和低端表(Low list)

- 对每个词项，维护两个倒排记录表 ，分别成为高端表和低端表
  - 比如可以将高端表看成胜者表
- 遍历倒排记录表时，仅仅先遍历高端表
  - 如果返回结果数目超过K，那么直接选择前K篇文档返回
  - 否则，继续遍历低端表，从中补足剩下的文档数目
- 上述思路可以直接基于词项权重，不需要全局量g(d)
- 实际上，相当于将整个索引分层

策略四：影响度排序

- 将词项t 对应的所有文档d 按照tf_{td} 值降序排列
- 不同词项倒排记录表中文档所采用的排序方式就不是统一的
- 不能通过并发扫描多个倒排记录表的方式来计算文档的得分
- 有两种思路可以显著降低用于累加得分的文档数目

思路1：提前结束

- 对某个查询词项t 对应的倒排记录表进行从前往后扫描时，可以在某个阶段停止
  - 停止条件一：扫描了r 篇固定数目的文档

或者采用

  - 停止条件二：是当前记录的tf_{td}已经低于某个阈值

思路2：词项按照idf 降序排列

- 词项按照idf 降序排列（query里面的词项）
  - 对最终得分贡献最大的查询词项首先被考虑
- 在查询处理过程中进行自适应处理
  - 当遇到具有较低idf 值的查询词项时，可以根据和前一个查询词项的文档得分的改变值来决定是否需要处理，改变值达到最小限度的时候终止。
- 例如：查询*catcher in the rye* 时，按idf排序应该是catcher，rye，in，the，当我们依次处理，处理到in时发现改变值很小，对排序影响很小，所以终止。
- 上述思路给出所有方法所共有的一个一般形式。我们也可以实现另外的静态得分情况下的倒排索引

策略五：簇剪枝方法——预处理

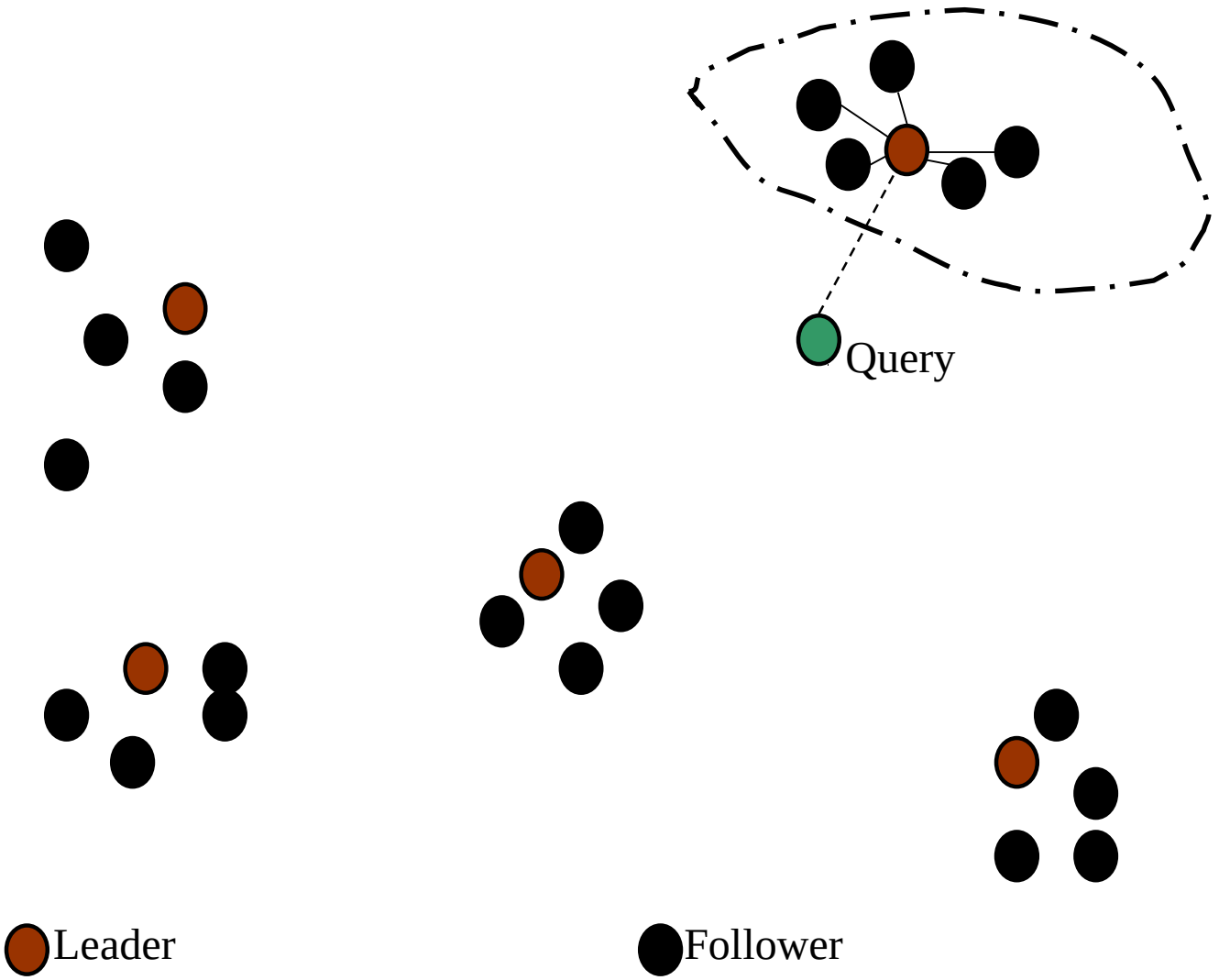
- 从N 篇文档组成的文档集中随机选出$\sqrt{N}$ 篇文档，它们称为先导者（leader）集合
- 对于每篇不属于先导者集合的文档，计算离之最近的先导者
  - 不属于先导者集合的文档称为追随者
  - 对于N 篇随机选出的先导者文档，其期望分配到的追随者个数大约为$\sqrt{N}$

先导者：通过文档聚类得到

簇剪枝方法——查询处理

- 查询处理：
  - 给定查询q，通过与先导者计算余弦相似度，找出和它最近的先导者L
  - 候选集合A 包括L 及其追随者，然后对A 中的所有的文档计算余弦相似度

可视化示意图



为什么采用随机抽样？

- 速度快
- 先导者能够反映数据的分布情况

常见变形

- 预处理时，我们将每个追随者分配给离它最近的$b1$个先导者
- 查询处理时，我们将考虑和查询q 最近的$b2$ 个先导者
- 很显然，前面讲到的方法只是该方法在$b1 = b2 = 1$ 情况下的一个特例

思考一下

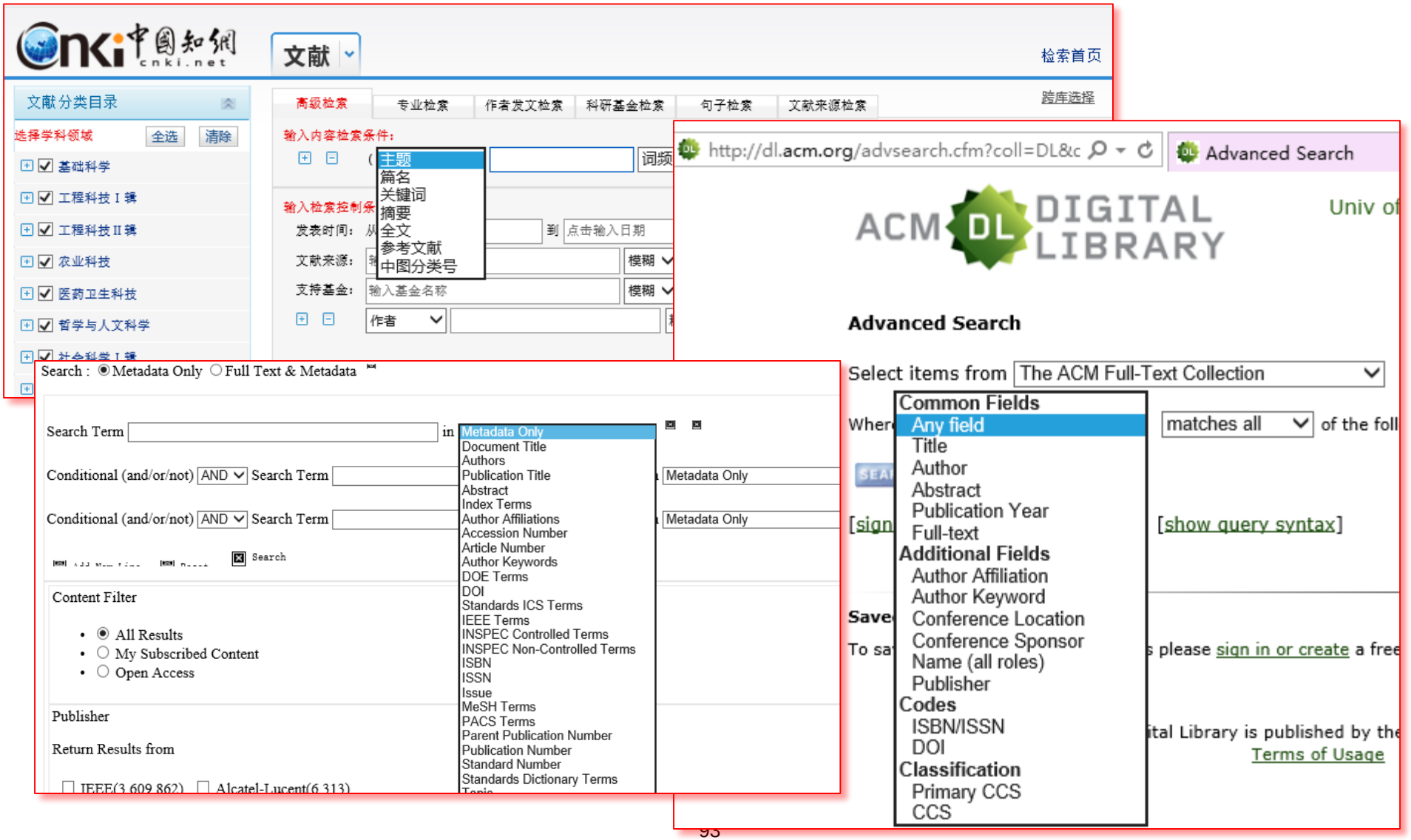
- 在簇剪枝方法中，第一步发现最近先导者过程中需要多少次余弦计算？
- 前一页中变量 *b1*，*b2* 的作用？

策略六：参数化索引以及域索引

参数化索引，p76

- 迄今，我们一直将文档看成由词项组成的有序排列
- 事实上，一个文档一般是由几个部分组成，这些部分有不同的意义，如：
  - 作者
  - 题目
  - 正文
  - 语言
  - 格式
  - ……
- 这些构成了一个文档的元数据

很多学术资源支持参数化索引



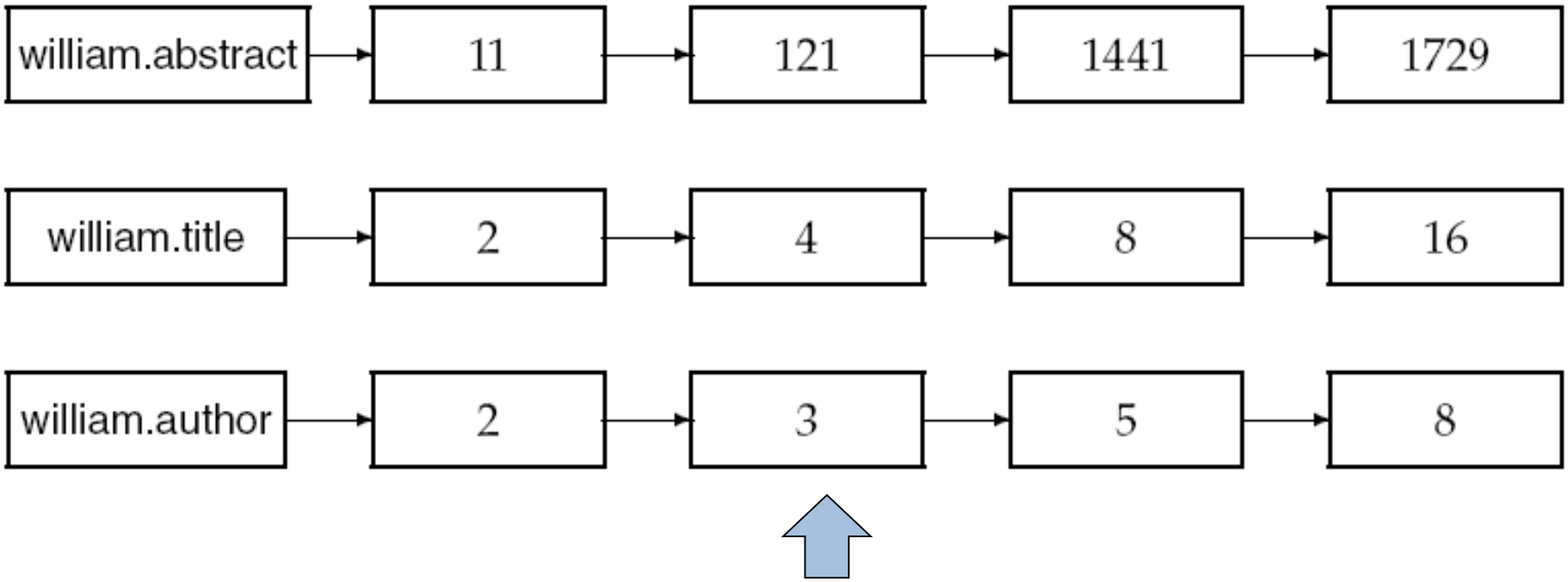
字段（Field）

- 我们经常希望检索这些元数据
  - 如：寻找莎士比亚在1601年写的小说，文中包含*alas poor Yorick*这几个词
- （Year=1601）就是一个字段
- 同样，（作者=莎士比亚）也是一个字段
- 按字段查询一般都属于联合查询
  - 即每个域条件都满足

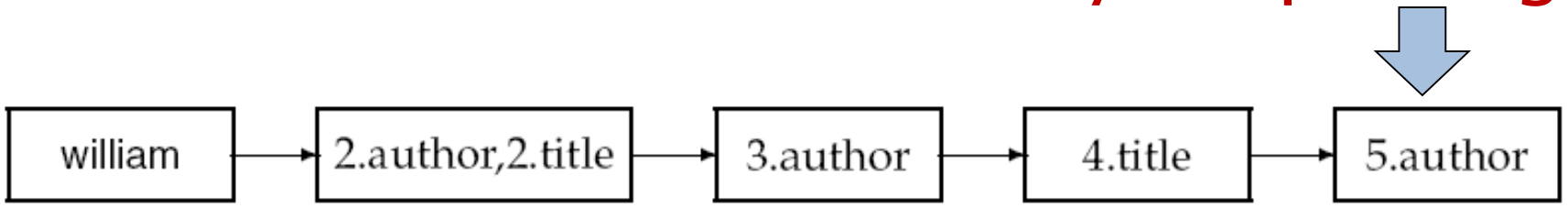
域（zone）

- 域是一个可以包含任意内容的区域，如：
  - 题目
  - 摘要
  - 引用…
- 在域上建立倒排索引同样可以进行查询
- 例如：“寻找标题中出现merchant、作者中出现william且正文中出现gentle rain的文档”

域索引



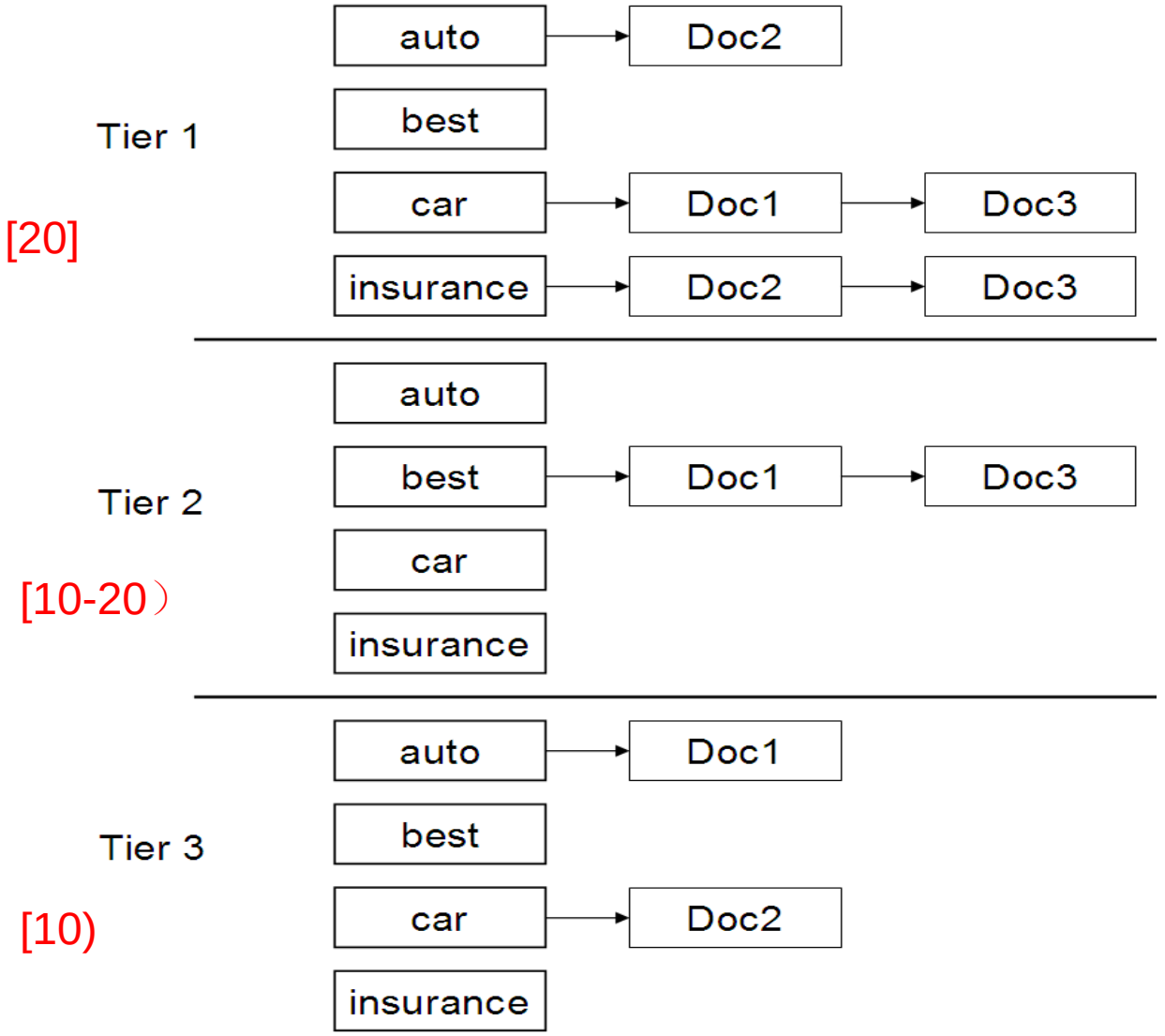
Encode zones in dictionary vs. postings.



策略七：层次索引

- 可以看成是优胜表的一般化形式
  - 最重要
  - …
  - 最不重要
- 可以用静态得分或者其它得分衡量
- 倒排记录表按照重要性降序转化成层次索引
- 查询是只用上层索引，除非上层索引返回结果小于K
  - 上层返回结果小于K则再从下一层中检索

层次索引


Tier 1
[20]
auto
Doc2
best
car
Doc1
Doc3
insurance
Doc2
Doc3
Tier 2
[10-20）
auto
best
Doc1
Doc3
car
insurance
Tier 3
[10)
auto
Doc1
best
car
Doc2
insurance

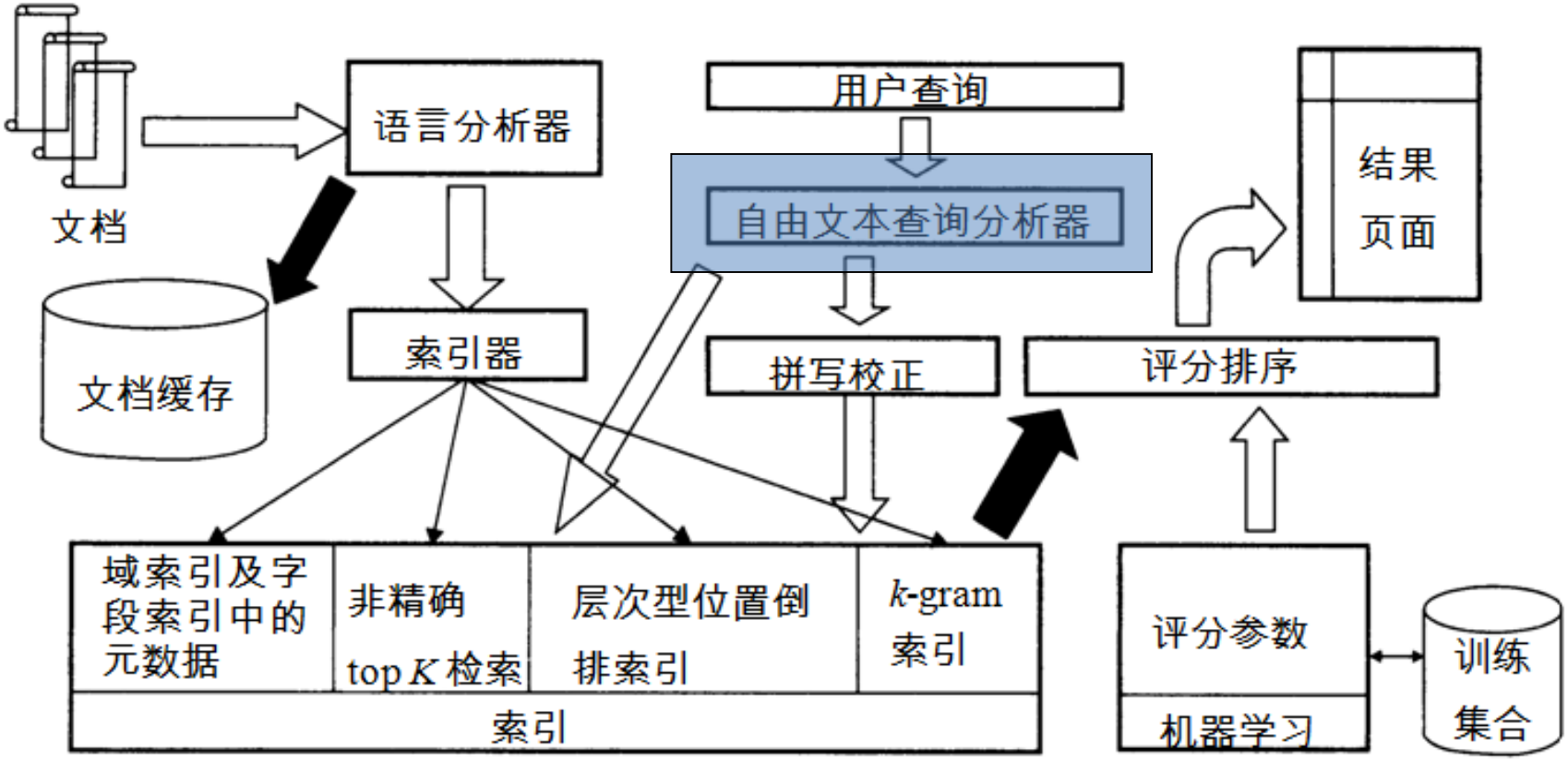
小结：非精确top K检索

- 策略一：索引去除(Index elimination)
- 策略二：胜者表
- 策略三：静态得分
- 策略四：影响度排序
- 策略五：簇剪枝方法——预处理
- 策略六：参数化索引以及域索引
- 策略七：层次索引

提纲

1. 上一讲回顾
2. 结果排序的重要性
3. 结果排序的实现
4. 完整的搜索系统

搜索系统组成


文档
语言分析器
用户查询
自由文本查询分析器
结果
页面
文档缓存
索引器
拼写校正
评分排序
域索引及字
段索引中的
元数据
非精确
top K 检索
层次型位置倒
排索引
k-gram
索引
索引
评分参数
机器学习
训练
集合

查询词项的邻近性

- 自由文本查询：用户输入几个词项到搜索框——一般的互联网检索
- 用户往往希望返回的文档中大部分或者全部查询词项之间的距离比较近
- 令文档d中包含所有查询词项的最小窗口大小为ω，其取值为窗口内词的个数
- 假设某篇文档仅仅包含一个句子*The quality of mercy is not strained*，那么查询*strained mercy*在此文档中的最小窗口大小是4
- 用窗口大小来度量位置关系

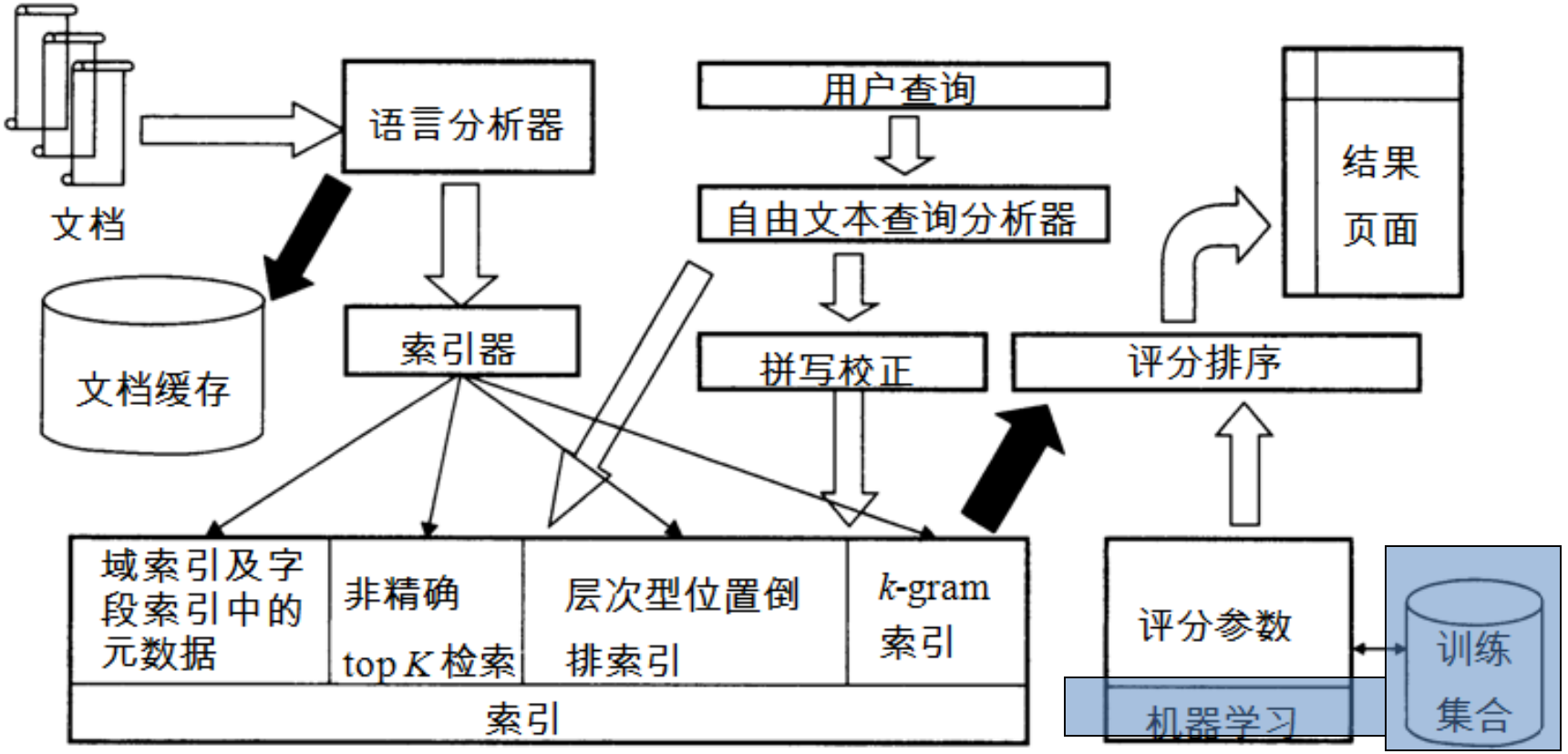
词项的邻近性示例


https://www.baidu.com/s?wd=%E4%B8%87%E7%
万立骏 化学_百度搜索
侯建国副部长做主题报告
Baidu百度
万立骏 化学
百度一下
添加百度到桌面，搜索更便捷!
百度首页 消息 设置 fa...7@tom.com
网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»
百度为您找到相关结果约65,800个
万立骏_百度百科
姓名：万立骏
生日：1957年7月 职业：物理化学家
简介：万立骏，博士，物理化学家，中国科学院化学研究所研究员。1987年获大连理工大学硕士学位，1996年获日本东北大学博士学位。主要从事电化学、SPM技术、纳米化学和纳米材料的研究...
个人简介 履历 学术成就 所获奖项 代表性论著
baike.baidu.com/ 2014-04-17
万立骏 化学的最新相关信息
万立骏任中科大校长 曾有二十余项发明专利(图)
至此,中国五所著名大学的校长全部完成换届,巧合的是,万立骏与此前上任的北京大学新校长林建华、清华大学新校长邱勇同为化学家出身,中国三所顶尖高校...
新浪新闻 1天前
万立骏:"当好大家的服务员" 中国教育新闻网 1天前
万立骏任职中科大校长 环球网 1天前
中科大新任校长万立骏,系海归院士的杰出... 网易新闻 2天前
万立骏 - 万立骏 简历 资料 新闻-中国党... 中国共产党新闻网 3天前
中国3所顶尖高校校长组成"化学三掌门"怎么解释 万立骏院士资料
2天前 - 中国3所顶尖高校校长组成"化学三掌门"怎么解释 万立骏院士资料。值得一提的是,万立骏与此前上任的北京大学[微博]新校长林建华、清华大学[微博]新...
edu.qlwb.com.cn/0327/3... - 百度快照 - 83%好评
相关人物 展开
曹淑敏 多少人羡慕不已的骄子
袁帅 清华大学教师
刘焕香 现任兰州大学教授
翁冰莹 厦门大学人文学院博士
肖益鸿 福大化学化工学院就职
江雷 中国著名纳米材料专家
白春礼 化学和纳米科技
钱逸泰 中国科学院院士
其他人还搜 展开
方璐
侯建国 中国科学技术大学校长
徐良杰 博士生导师
于吉红 科学家

查询分析器

- 自由文本查询对用户输入的关键词可能需要基于底层索引结果对多个查询进行处理，如查询*rising interest rates*之类*query*时，查询分析器可能做如下操作：
  - 1. 将用户输入的查询字符串看成一个短语查询
  - 2. 如果包含短语rising interest rates 的文档数目少于10 篇，那么会将原始查询看成rising interest 和interest rates 两个查询短语，同样通过向量空间方法来计算。
  - 3. 如果结果仍然少于10 个，重新利用向量空间模型求解，认为3 个查询词项之间是互相独立的

搜索系统组成


文档
语言分析器
用户查询
自由文本查询分析器
结果
页面
文档缓存
索引器
拼写校正
评分排序
域索引及字
段索引中的
元数据
非精确
top K 检索
层次型位置倒
排索引
k-gram
索引
索引
评分参数
机器学习
训练
集合

综合评分

- 已经介绍的评分函数有余弦相似度、静态得分、近邻性等。

- 如何将这些评分组合才是最优的？
- 通用方法——机器学习

机器学习有下面几种定义：“机器学习是一门人工智能的科学，该领域的主要研究对象是人工智能，特别是如何在经验学习中改善具体算法的性能”。“机器学习是对能通过经验自动改进的计算机算法的研究”。“机器学习是用数据或以往的经验，以此优化计算机程序的性能标准。”一种经常引用的英文定义是：A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

课后练习

- 习题7-3
- 习题7-5
- 习题7-7

思考：

各种查询操作在向量空间模型中的实现


谢谢大家!