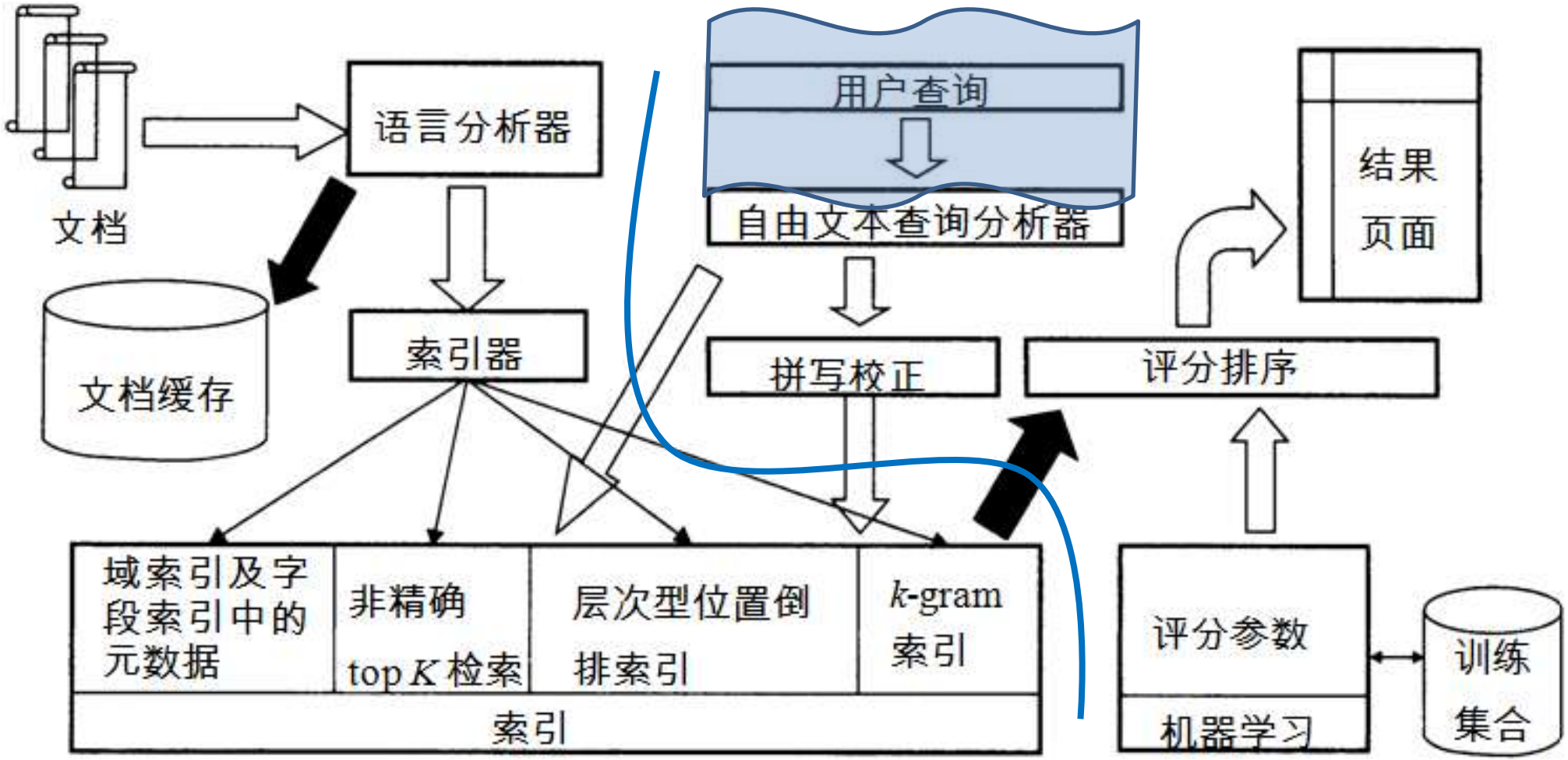
布尔查询
-基于倒排索引

苏州大学计算机学院
贡正仙

假定索引已经构建好

- 如何利用该索引来处理查询?

完整的搜索系统示意图


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

布尔检索

针对布尔查询的检索，布尔查询是指利用 AND, OR 或者 NOT操作符将词项 连接起来的查询

– 信息 AND 检索

– 信息 OR 检索

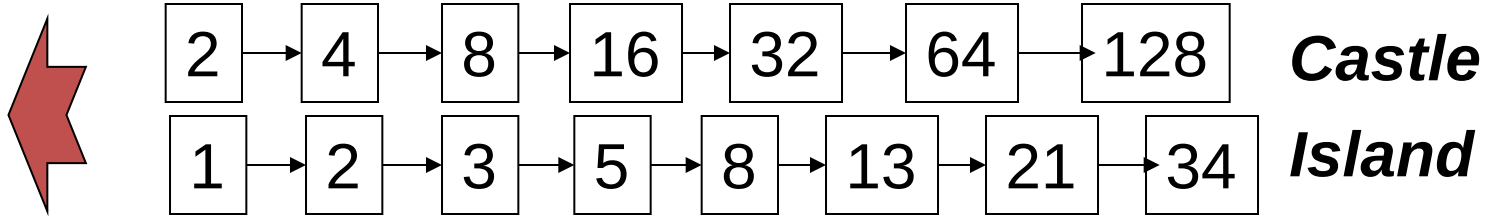
– 信息 AND 检索 AND NOT 教材

AND查询的处理

考虑如下查询（从简单的布尔表达式入手）：

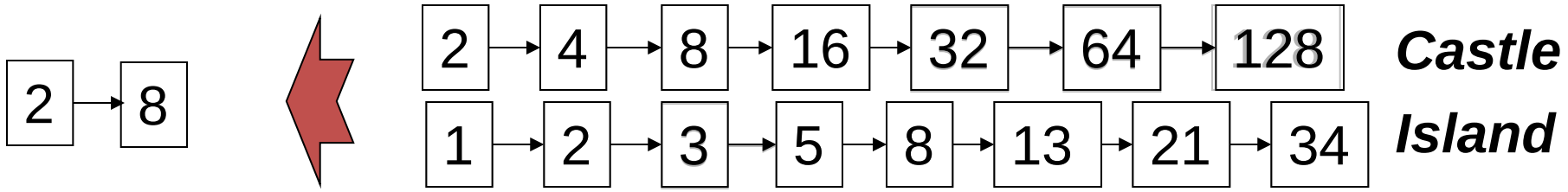
Castle AND Island

– 在词典中定位 Castle
  - 返回对应倒排记录表(对应的docID)
– 在词典中定位 Island
  - 再返回对应倒排记录表
– 合并(Merge)两个倒排记录表，即求交集



合并过程

每个倒排记录表都有一个定位指针，两个指针同时从前往后扫描，每次比较当前指针对应倒排记录，然后移动某个或两个指针。



合并时间为两个表长之和的线性时间

假定表长分别为x 和y, 那么上述合并算法的复杂度为 O(*x+y*)

关键原因: 倒排记录表按照docID排序

其它布尔查询的处理

- OR表达式：Castle or Island
- 两个倒排记录表的并集

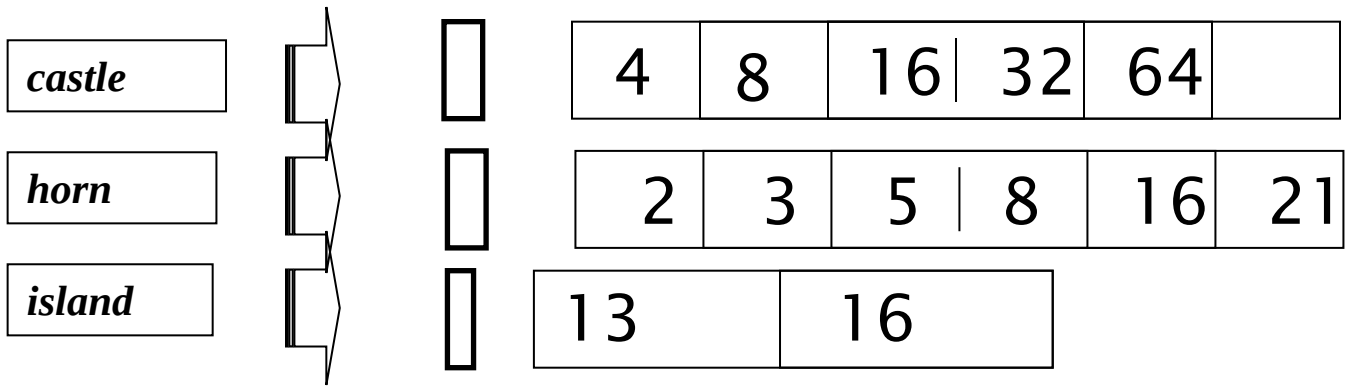
- NOT表达式： Castle AND NOT Island
- 两个倒排记录表的减

还可以有更复杂的
(Castle OR Island) AND NOT
(Horn OR River)

复杂的布尔查询需要考虑效率问题！

布尔查询优化

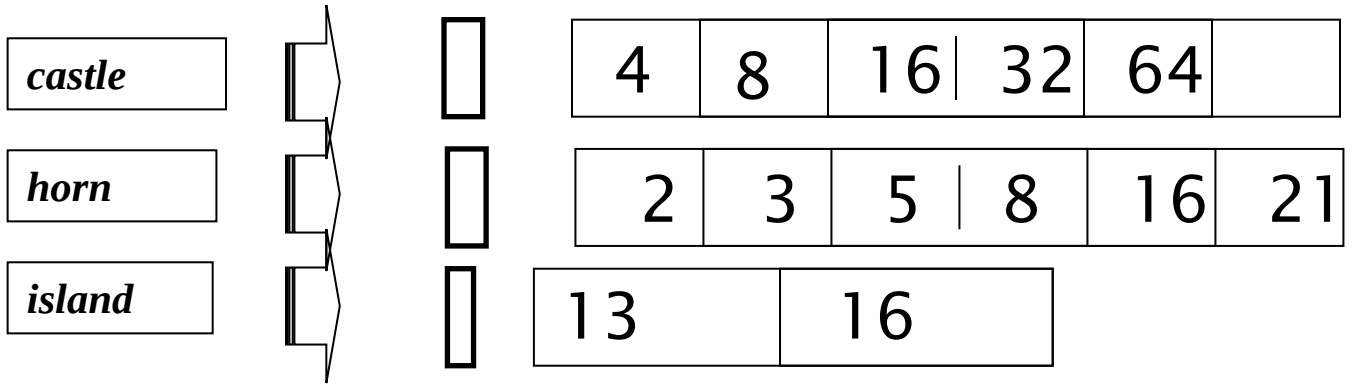
- 查询处理中是否存在处理的顺序问题？
- 考虑n 个词项的 AND
- 对每个词项，取出其倒排记录表，然后两两合并



查询: ***castle*** *AND* ***horn*** *AND* island

查询优化

- 按照表从小到大(即df从小到大)的顺序进行处理:
  - 每次从最小的开始合并



查询: ***castle*** *AND* ***horn*** *AND* island

相当于处理查询 (***island*** *AND* ***castle)*** *AND* ***horn***.

布尔检索的优点

- 构建简单，或许是构建IR系统的一种最简单方式
  - 在30多年中是最主要的检索工具
  - 当前许多搜索系统仍然使用布尔检索模型:
    - 电子邮件、文献编目、Mac OS X Spotlight工具

布尔检索的缺点

- 布尔查询构建复杂，不适合普通用户。构建不当，检索结果过多或者过少
- 没有充分利用词项的频率信息
  - 1 vs. 0 次出现
  - 2 vs. 1次出现
  - 3 vs. 2次出现, …
  - 通常出现的越多越好，需要利用词项在文档中的词项频率(term frequency, tf)信息
- 不能对检索结果进行排序