现代信息检索
Modern Information Retrieval

第17讲 信息采集
Crawling

*改编自"An introduction to Information retrieval"网上公开的课件，地址 http://nlp.stanford.edu/IR-book/

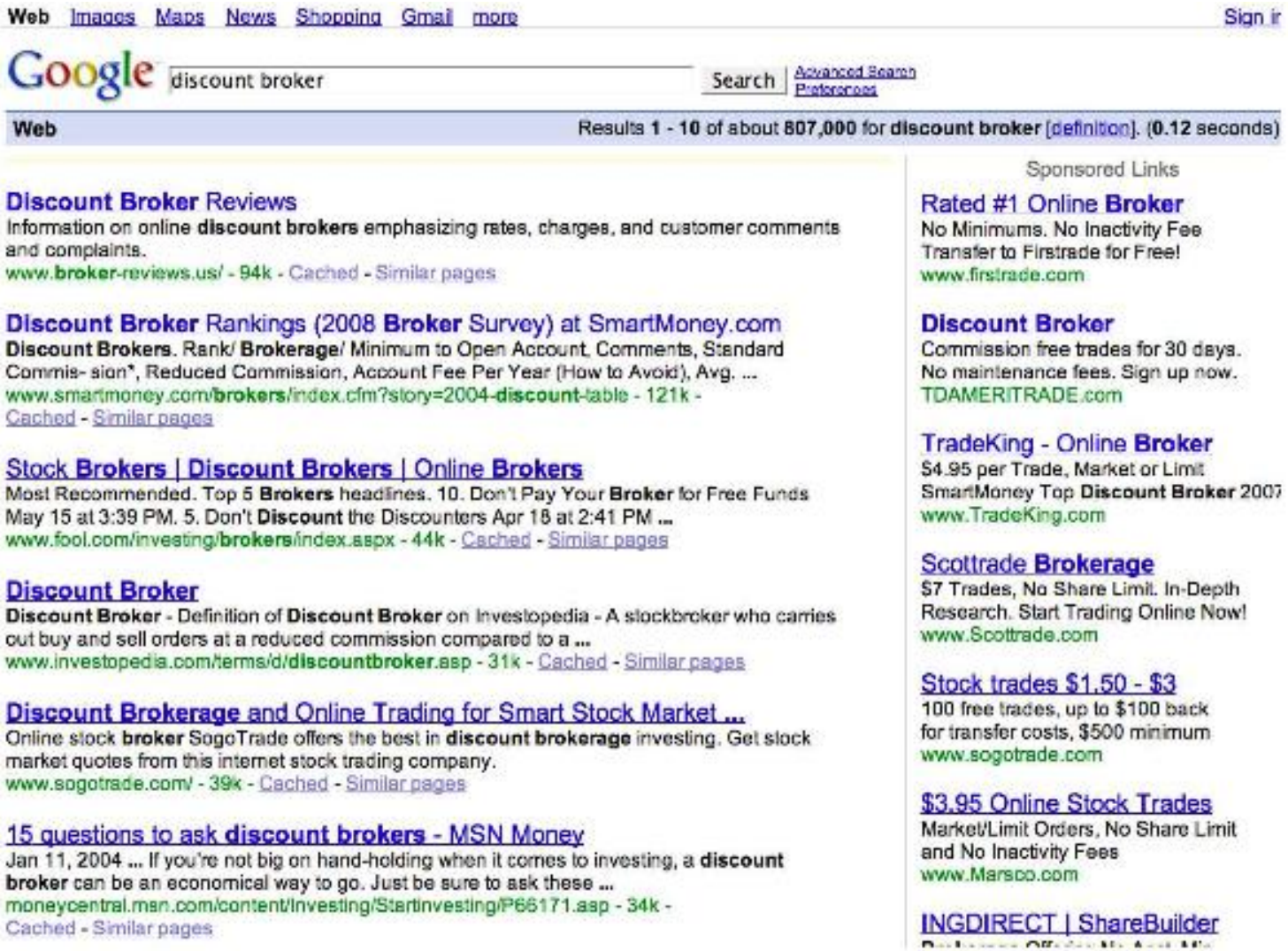
提纲

1. 上一讲回顾
2. 一个简单的采集器
3. 一个真实的采集器

提纲

1. 上一讲回顾
2. 一个简单的采集器
3. 一个真实的采集器

搜索广告


Web Images Maps News Shopping Gmail more
Sign in
Google
discount broker
Search
Advanced Search
Preferences
Web
Results 1 - 10 of about 807,000 for discount broker [definition]. (0.12 seconds)
Discount Broker Reviews
Information on online discount brokers emphasizing rates, charges, and customer comments and complaints.
www.broker-reviews.us/ - 94k - Cached - Similar pages
Discount Broker Rankings (2008 Broker Survey) at SmartMoney.com
Discount Brokers. Rank/ Brokerage/ Minimum to Open Account, Comments, Standard Commis- sion*, Reduced Commission, Account Fee Per Year (How to Avoid), Avg. ...
www.smartmoney.com/brokers/index.cfm?story=2004-discount-table - 121k - Cached - Similar pages
Stock Brokers | Discount Brokers | Online Brokers
Most Recommended. Top 5 Brokers headlines. 10. Don't Pay Your Broker for Free Funds May 15 at 3:39 PM. 5. Don't Discount the Discounters Apr 18 at 2:41 PM ...
www.fool.com/investing/brokers/index.aspx - 44k - Cached - Similar pages
Discount Broker
Discount Broker - Definition of Discount Broker on Investopedia - A stockbroker who carries out buy and sell orders at a reduced commission compared to a ...
www.investopedia.com/terms/d/discountbroker.asp - 31k - Cached - Similar pages
Discount Brokerage and Online Trading for Smart Stock Market ...
Online stock broker SogoTrade offers the best in discount brokerage investing. Get stock market quotes from this internet stock trading company.
www.sogotrade.com/ - 39k - Cached - Similar pages
15 questions to ask discount brokers - MSN Money
Jan 11, 2004 ... If you're not big on hand-holding when it comes to investing, a discount broker can be an economical way to go. Just be sure to ask these ...
moneycentral.msn.com/content/Investing/Startinvesting/P66171.asp - 34k - Cached - Similar pages
Sponsored Links
Rated #1 Online Broker
No Minimums. No Inactivity Fee
Transfer to Firstrade for Free!
www.firstrade.com
Discount Broker
Commission free trades for 30 days.
No maintenance fees. Sign up now.
TDAMERITRADE.com
TradeKing - Online Broker
$4.95 per Trade, Market or Limit
SmartMoney Top Discount Broker 2007
www.TradeKing.com
Scottrade Brokerage
$7 Trades, No Share Limit. In-Depth Research. Start Trading Online Now!
www.Scottrade.com
Stock trades $1.50 - $3
100 free trades, up to $100 back for transfer costs, $500 minimum
www.sogotrade.com
$3.95 Online Stock Trades
Market/Limit Orders, No Share Limit and No Inactivity Fees
www.Marsco.com
INGDIRECT | ShareBuilder

Google次高竞标价格拍卖机制

advertiser	bid	CTR	ad rank	rank	paid
A	$4.00	0.01	0.04	4	(minimum)
B	$3.00	0.03	0.09	2	$2.68
C	$2.00	0.06	0.12	1	$1.51
D	$1.00	0.08	0.08	3	$0.51

- bid: 每个广告商为每次点击给出的最大投标价格
- CTR: 点击率，即一旦被显示后被点击的比率。CTR是一种相关性度量指标。
- ad rank: bid × CTR: 这种做法可以在 (i) 广告商愿意支付的价钱 (ii) 广告的相关度高低 之间进行平衡。
- rank: 拍卖中的排名
- paid: 广告商的次高竞标价格

Google次高竞标价格拍卖机制

advertiser	bid	CTR	ad rank	rank	paid
A	$4.00	0.01	0.04	4	(minimum)
B	$3.00	0.03	0.09	2	$2.68
C	$2.00	0.06	0.12	1	$1.51
D	$1.00	0.08	0.08	3	$0.51

次高竞标价格拍卖： 广告商支付其维持在拍卖中排名所必须的价钱(加上一分钱) (用它的下一名计算其支付价格)

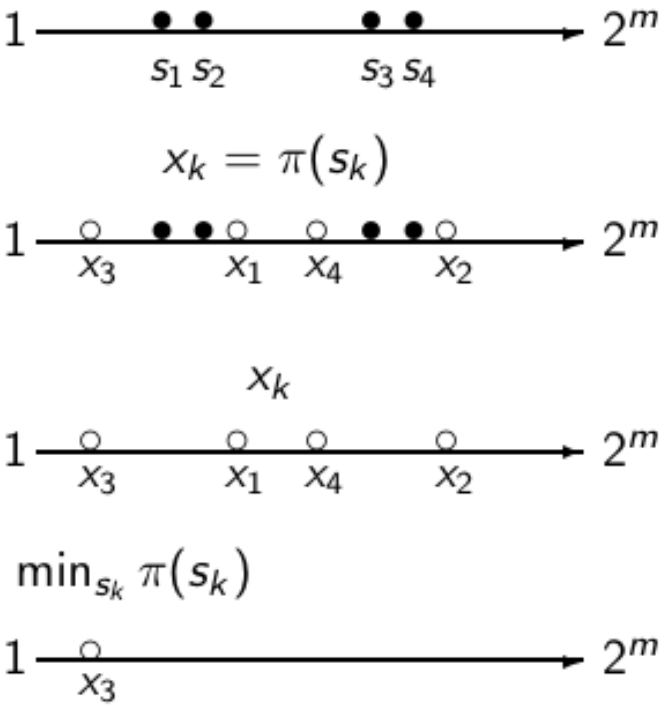
$\text{price}_1 \times \text{CTR}_1 = \text{bid}_2 \times \text{CTR}_2$ (使得排名rank_1=rank_2)

$\text{price}_1 = \text{bid}_2 \times \text{CTR}_2 / \text{CTR}_1$

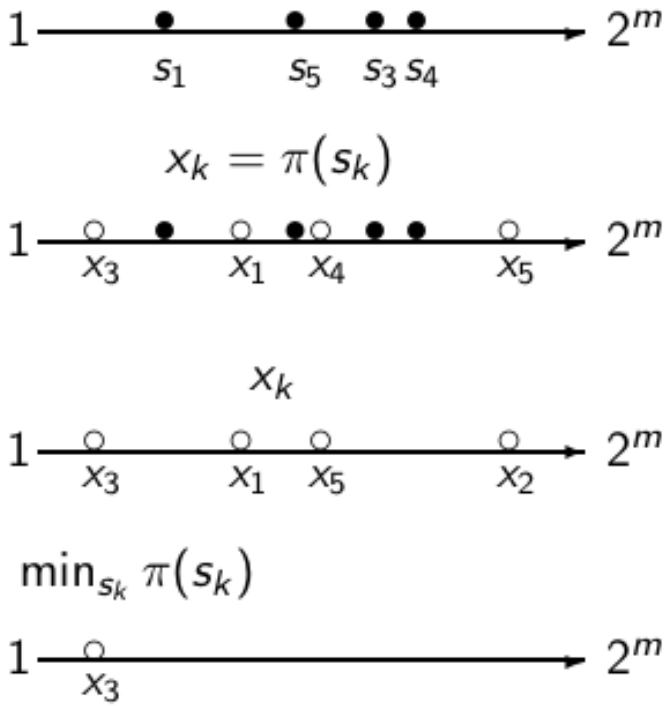
$p_1 = \text{bid}_2 \times \text{CTR}_2/\text{CTR}_1 = 3.00 \times 0.03/0.06 = 1.50$
$p_2 = \text{bid}_3 \times \text{CTR}_3/\text{CTR}_2 = 1.00 \times 0.08/0.03 = 2.67$
$p_3 = \text{bid}_4 \times \text{CTR}_4/\text{CTR}_3 = 4.00 \times 0.01/0.08 = 0.50$

置换和最小值：例子

文档 1: $\{s_k\}$



文档2: $\{s_k\}$



使用$\min_{s \in d1} \pi(s) = \min s \in d_2\ \pi(s)$ 作为文档 d_1 和 d_2是否近似重复的测试条件？ 该例子中置换π表明: $d_1 \approx d_2$

例子

	d_1	d_2
s_1	1	0
s_2	0	1
s_3	1	1
s_4	1	0
s_5	0	1

$h(x) = x \bmod 5$

$g(x) = (2x + 1) \bmod 5$

$\min(h(d_1)) = 1 \neq 0 = \min(h(d_2))$ $\min(g(d_1)) =$

$2 \neq 0 = \min(g(d_2))$

$\hat{J}(d_1, d_2) = \frac{0+0}{2} = 0$

	d_1 slot		d_2 slot	
		∞		∞
		∞		∞
$h(1) = 1$	1	1	–	∞
$g(1) = 3$	3	3	–	∞
$h(2) = 2$	–	1	2	2
$g(2) = 0$	–	3	0	0
$h(3) = 3$	3	1	3	2
$g(3) = 2$	2	2	2	0
$h(4) = 4$	4	1	–	2
$g(4) = 4$	4	2	–	0
$h(5) = 0$	–	1	0	0
$g(5) = 1$	–	2	1	0

最终的梗概

本讲内容

- 网页采集的概念
- 一个简单的采集器
- 一个真实的采集器

提纲

1. 上一讲回顾
2. 一个简单的采集器
3. 一个真实的采集器

采集会有多难?

- Web搜索引擎必须要要采集网页文档
- 其他有些IR系统获得文档内容相对容易一些
  - 比如，对硬盘上所有文档建立索引只需要基于文件系统进行迭代式扫描即可
- 但是对于Web IR系统来说，获得文档内容需要更长的时间 . . .
- . . . 这是因为存在延迟
- 但是这真的是系统设计中的一个难点吗？

基本的采集过程

- 初始化采集URL种子队列；
- 重复如下过程：
  - 从队列中取出URL
  - 下载并分析网页
  - 从网页中抽取更多的URL
  - 将这些URL放到队列中
- 这里有个“Web的连通性很好”的基本假设

课堂思考题: 下列爬虫有什么问题?

```
urlqueue := (some carefully selected set of seed urls)
while urlqueue is not empty:
myurl := urlqueue.getlastanddelete()
mypage := myurl.fetch()
fetchedurls.add(myurl)
newurls := mypage.extracturls()
for myurl in newurls:
if myurl not in fetchedurls and not in urlqueue:
urlqueue.add(myurl)
addtoinvertedindex(mypage)
```

上述简单采集器的问题

- 规模问题: 必须要分布式处理
- 我们不可能索引所有网页，必须要从中选择部分网页，如何选择？
- 重复网页：必须要集成重复检测功能
- 作弊网页和采集器陷阱：必须要集成作弊网页检测功能
- 礼貌性问题： 对同一网站的访问按遵照协议规定，并且访问的间隔必须要足够
- 新鲜度(freshness)问题：必须要定期更新或者重采
  - 由于Web的规模巨大，我们只能对一个小的网页子集频繁重采
  - 同样，这也存在一个选择或者优先级问题

采集规模的数量级

- 如果要在一个月内采集20,000,000,000个页面. . .
- . . . 那么必须要在一秒内大概采集 8000个网页!
- 由于我们采集的网页可能重复、不可下载或者是作弊网页，实际上可能需要更快的采集速度才能达到上述指标

采集器必须做到

礼貌性

- 不要高频率采集某个网站
- 仅仅采集robots.txt所规定的可以采集的网页

鲁棒性

- 能够处理采集器陷阱、重复页面、超大页面、超大网站、动态页面等问题

Robots.txt文件

- 1994年起使用的采集器协议(即规定了采集器对网站的访问限制)
- 例子：
  - User-agent: *

    Disallow: /yoursite/temp/
  - User-agent: searchengine

    Disallow: /
- 采集时，要将每个站点的 robots.txt放到高速缓存中，这一点相当重要

Example of a robots.txt (nih.gov)

```
User-agent: PicoSearch/1.0
Disallow: /news/information/knight/
Disallow: /nidcd/
...
Disallow: /news/research_matters/secure/
Disallow: /od/ocpl/wag/
User-agent: *
Disallow: /news/information/knight/
Disallow: /nidcd/
...
Disallow: /news/research_matters/secure/
Disallow: /od/ocpl/wag/
Disallow: /ddir/
Disallow: /sdminutes/
```

https://www.jd.com/robots.txt

```
User-agent: *
Disallow: /?*
Disallow: /pop/*.html
Disallow: /pinpai/*.html?*
User-agent: EtaoSpider
Disallow: /
User-agent: HuihuiSpider
Disallow: /
User-agent: GwdangSpider
Disallow: /
User-agent: WochachaSpider
Disallow: /
```

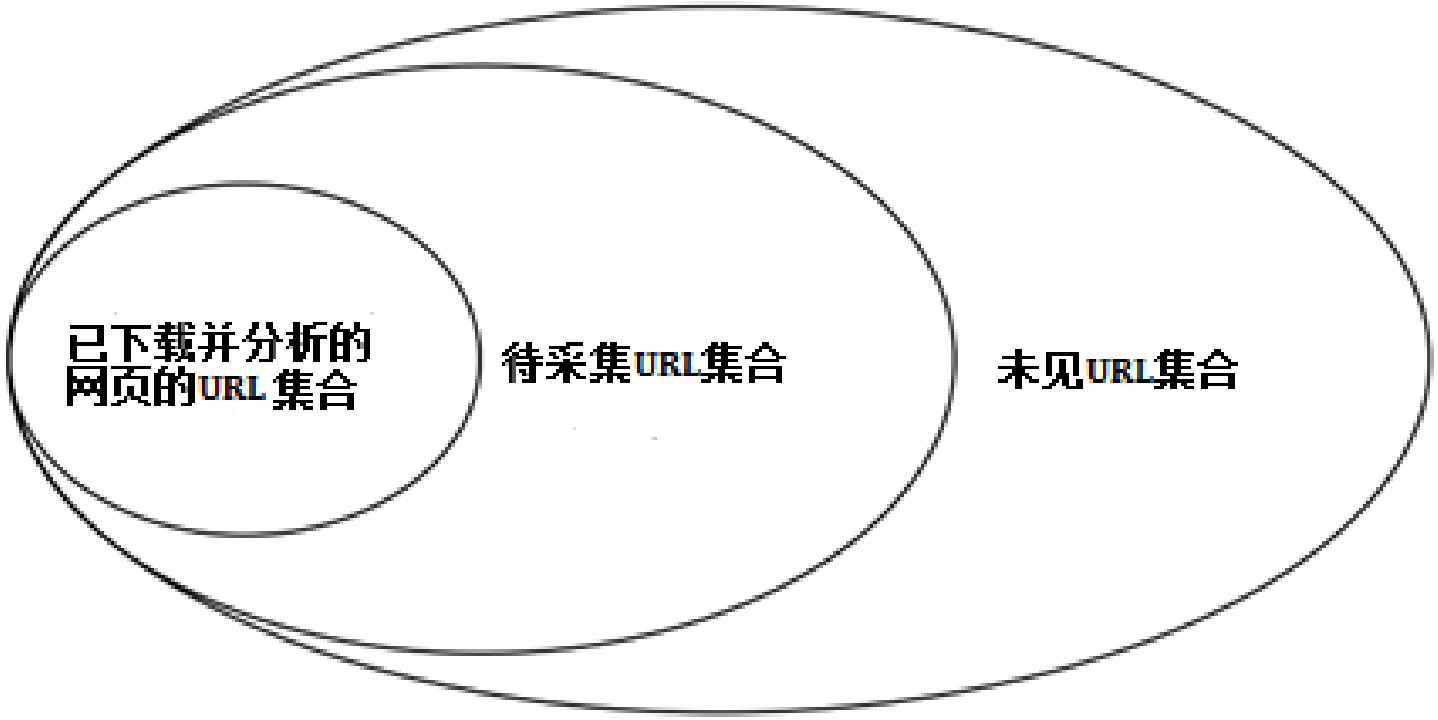
任意一个采集器应该做到

- 能够进行分布式处理
- 支持规模的扩展：能够通过增加机器支持更高的采集速度
- 优先采集高质量网页
- 能够持续运行：对已采集网页进行更新

提纲

1. 上一讲回顾
2. 一个简单的采集器
3. 一个真实的采集器

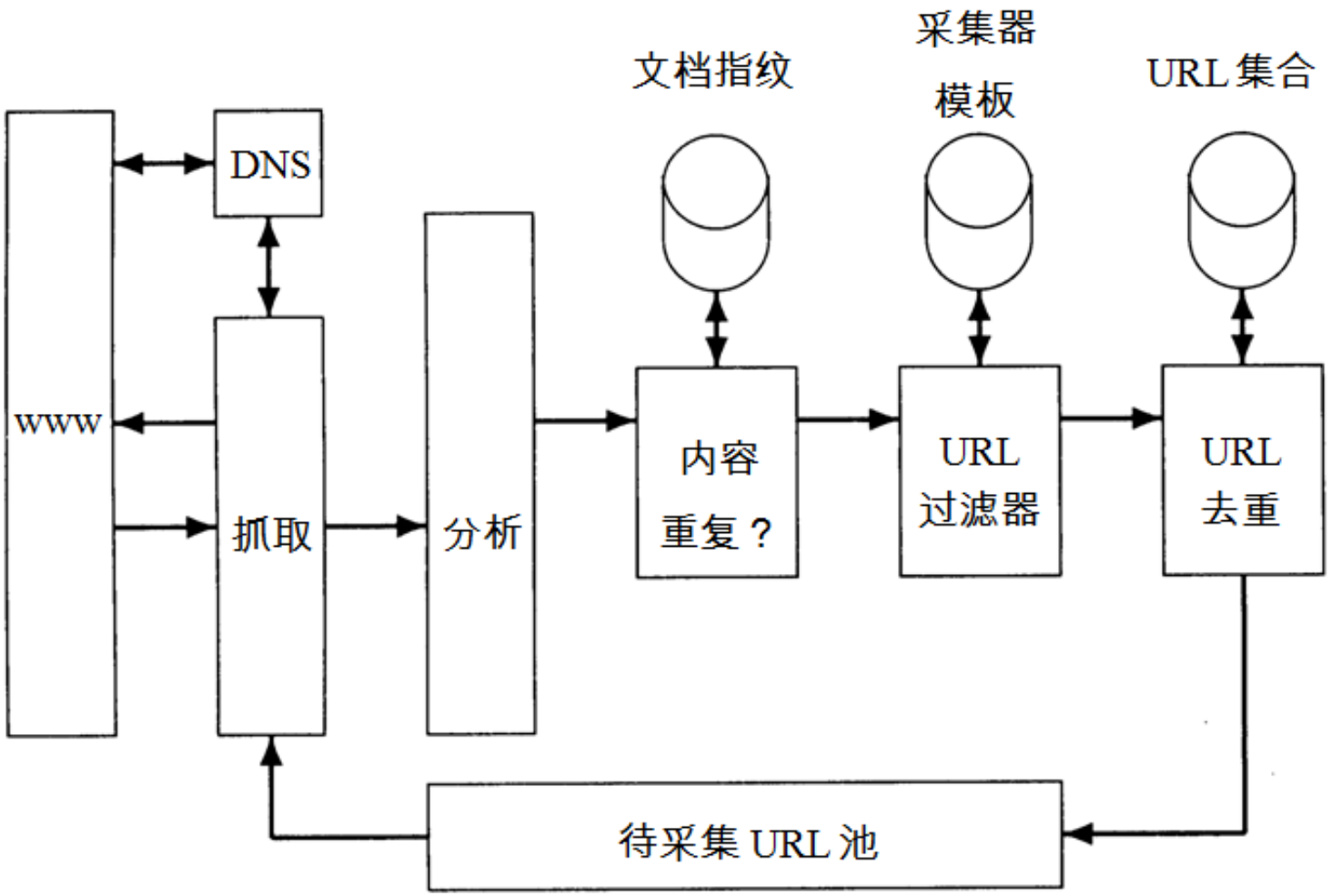
待采集URL池(URL frontier)



待采集URL池

- 待采集URL池是一个数据结构，它存放并管理那些已经看到但是还没有采集的URL集合
- 可能包含来自同一主机的不同页面
- 必要要避免在同一时间采集这些来自同一主机的页面
- 必须要保证采集线程任务饱和

基本的采集架构



URL 规范化(normalization)

- 从网页中抽取的URL有些是相对地址
- 比如，在http://mit.edu网站下，我们会采集页面aboutsite.html
  - 该页面的绝对地址为： http://mit.edu/aboutsite.html
- 在网页分析过程中，必须要将相对URL地址规范化

内容重复判别(Content seen)

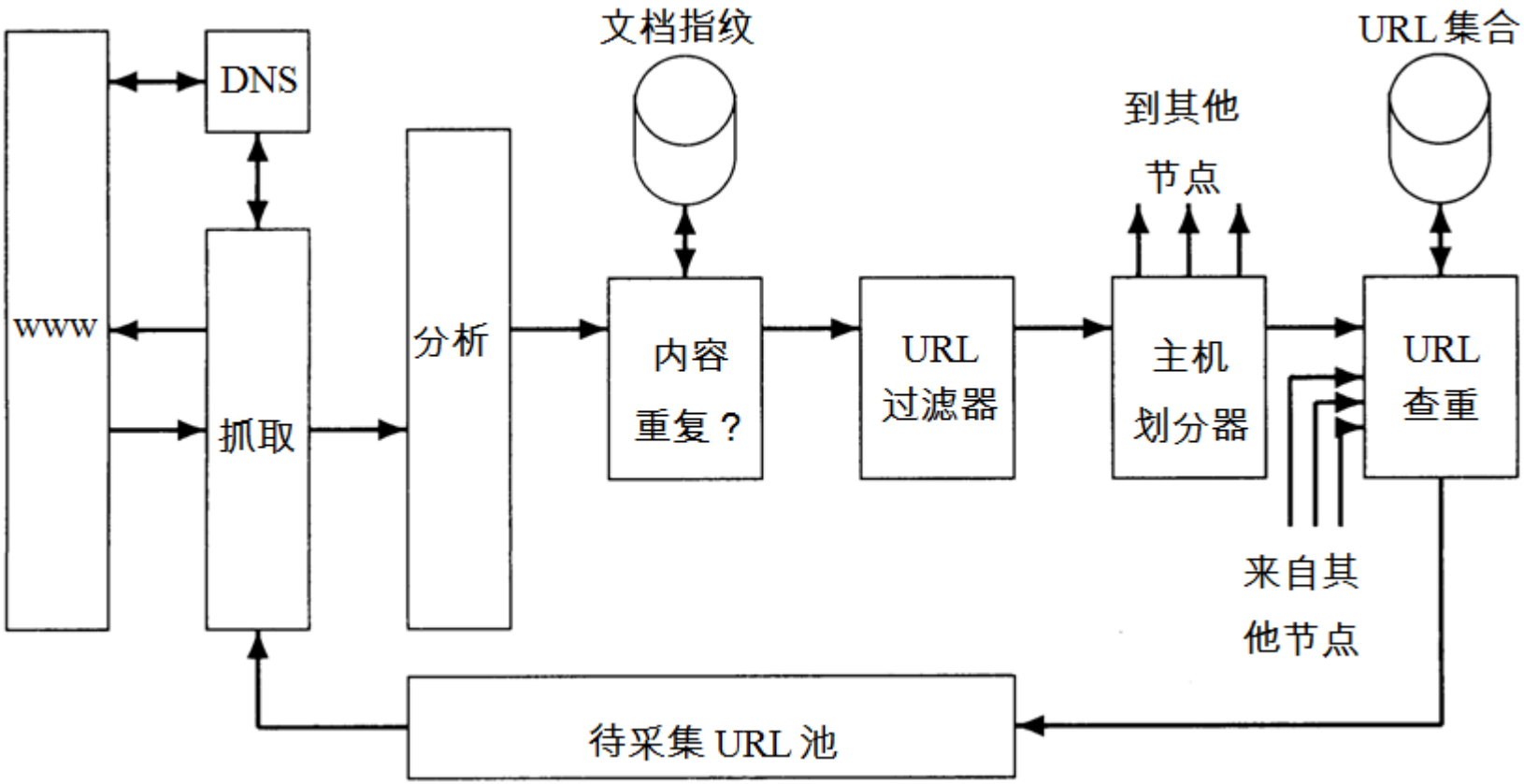
- 对每个抓取的页面，判断它是否已在索引当中
- 可以采用文档指纹或者shingle的方法判别
- 忽略那些已经在索引中的重复页面

分布式采集

- 运行多个采集线程，这些线程可以分布在不同节点上
  - 这些节点往往在地理上分散在不同位置
- 将采集的主机分配到不同节点上

Google 数据中心(wazfaring. com)

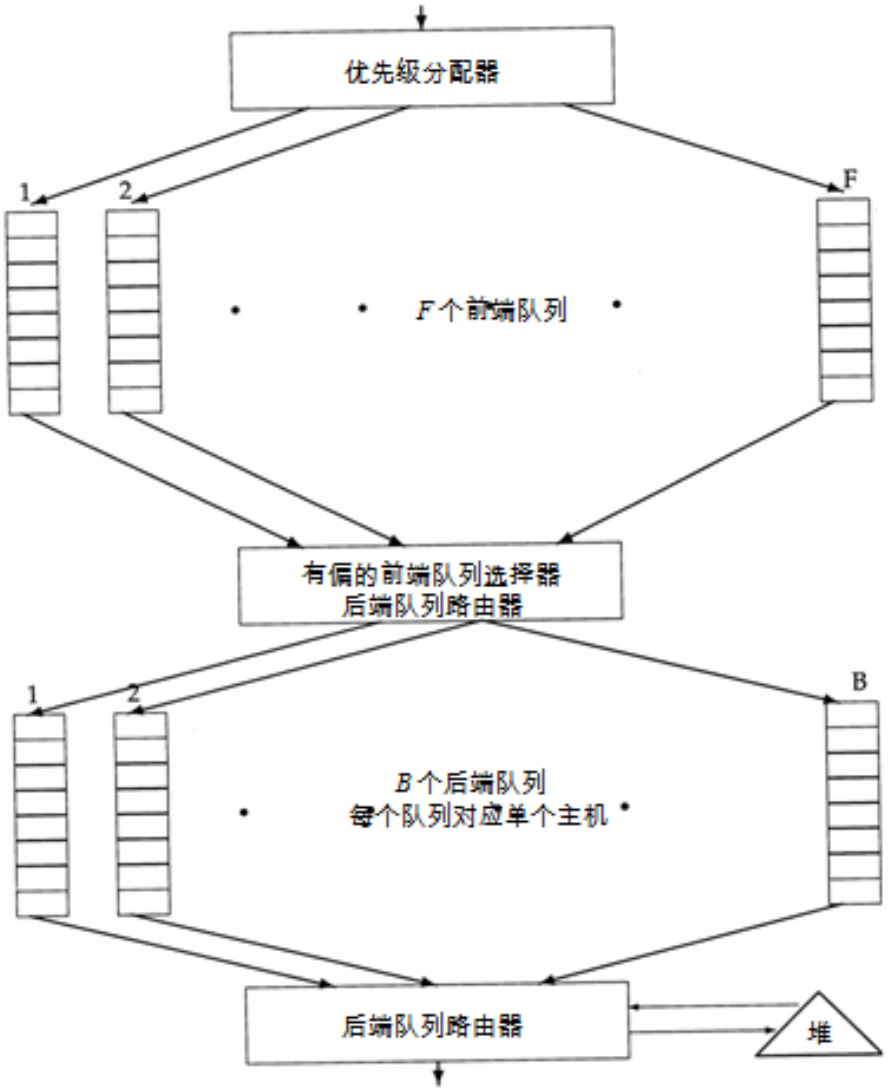
分布式采集器


文档指纹
URL 集合
DNS
到其他
节点
www
抓取
分析
内容
重复？
URL
过滤器
主机
划分器
URL
查重
来自其
他节点
待采集 URL 池

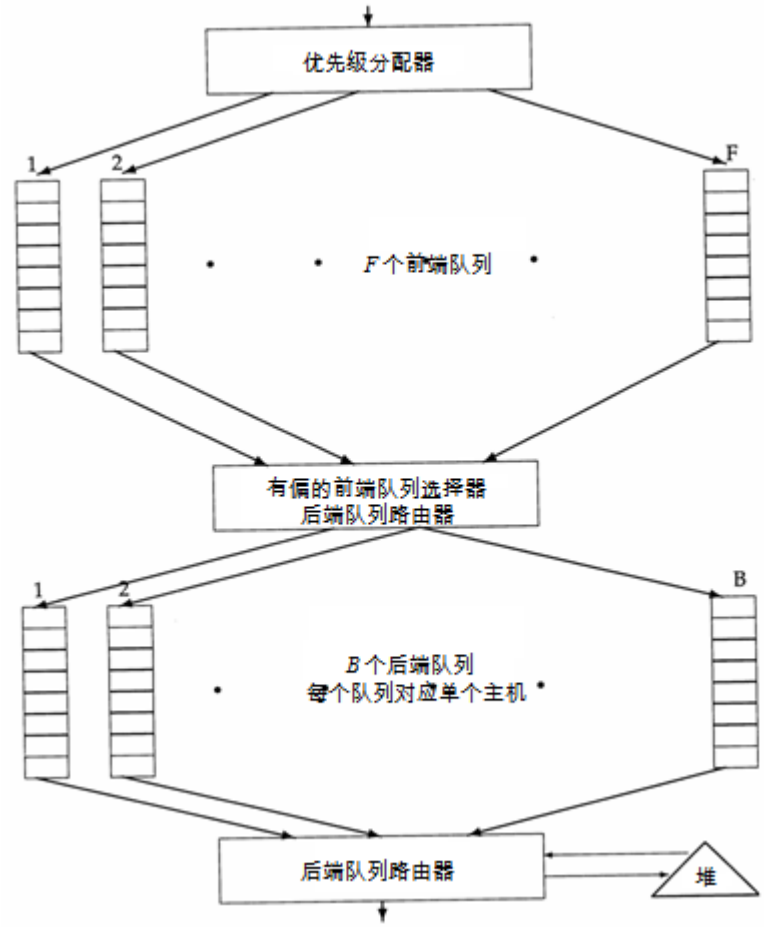
待采集URL池: 主要考虑两点

- 礼貌性: 不要非常频繁第访问某个Web服务器
  - 比如，可以在两次服务器访问之间设置一个时间间隔
- 新鲜度: 对某些网站的采集频率(如新闻网站)要高于其他网站
- 要实现上述功能并不容易，一个简单的优先级队列难以成功

Mercator 中的待采集URL缓冲池

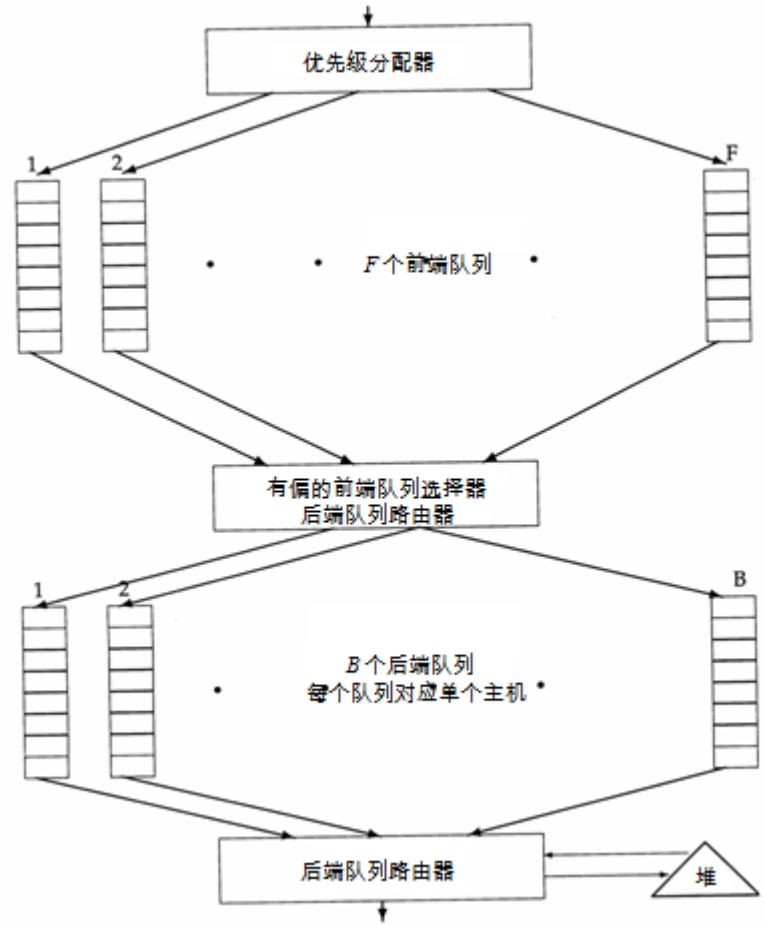


Mercator 中的待采集URL缓冲池



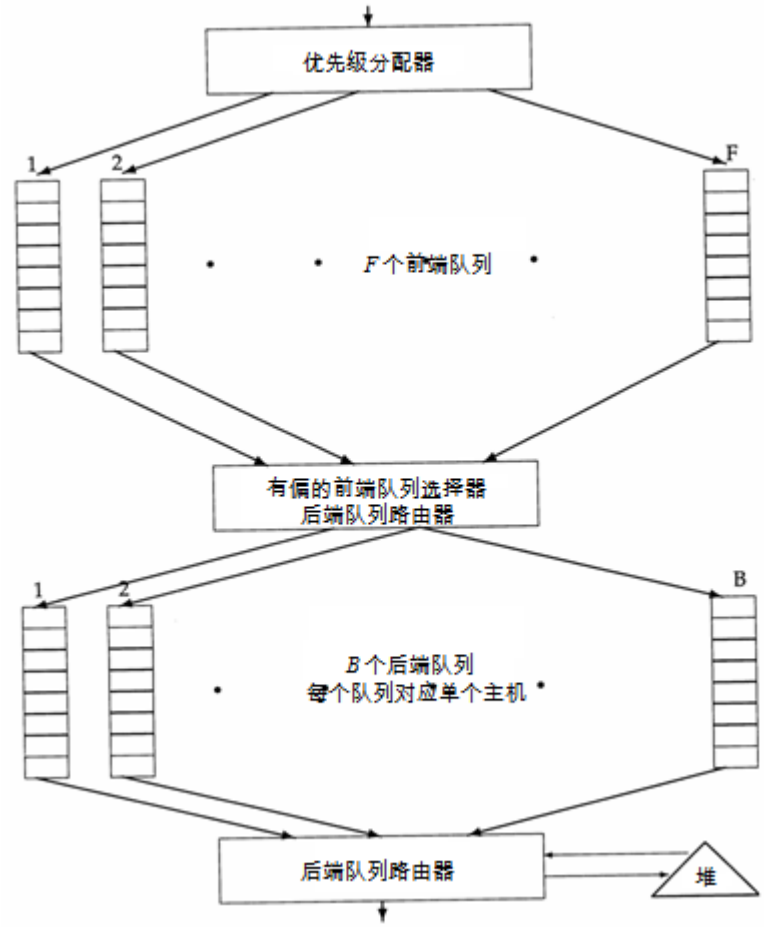
- URL从上部流入缓冲池

Mercator 中的待采集URL缓冲池



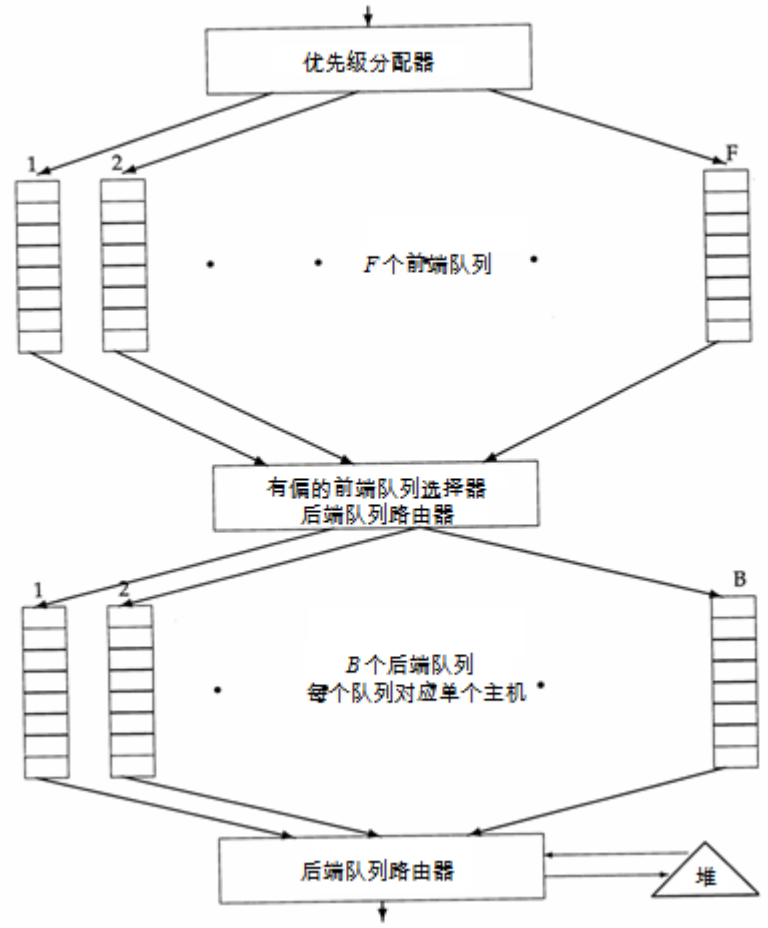
- URL从上部流入缓冲池
- 前端队列(Front queue)管理优先级

Mercator 中的待采集URL缓冲池



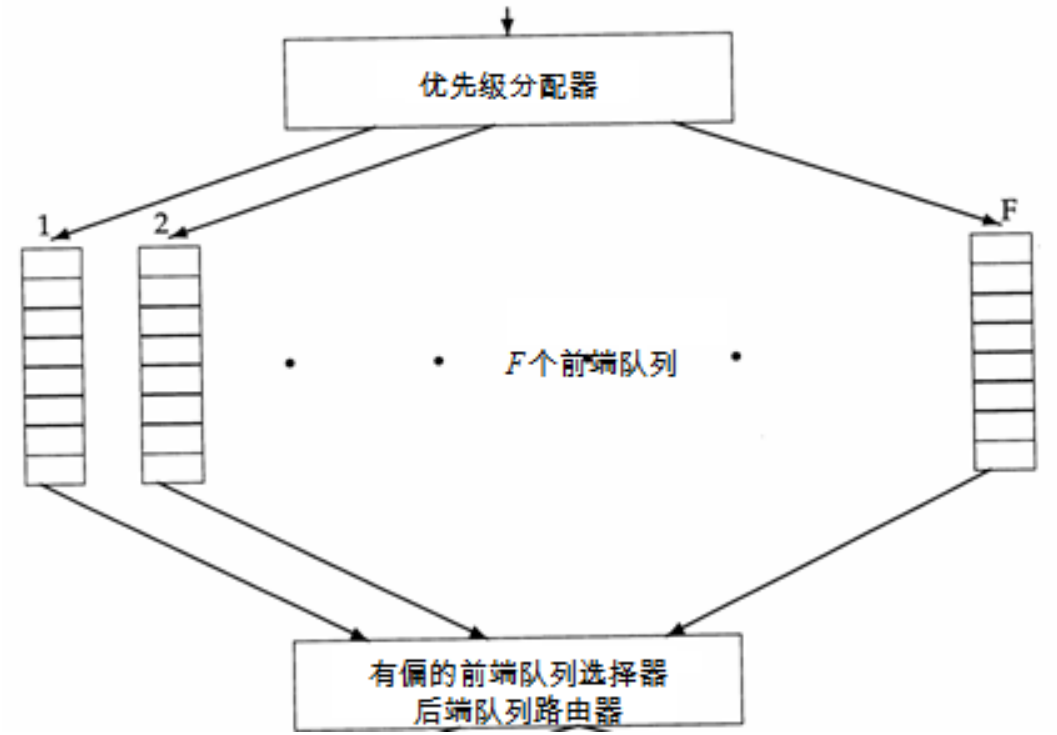
- URL从上部流入缓冲池
- 前端队列(Front queue)管理优先级
- 后端队列(Back queue) 实现礼貌性

Mercator 中的待采集URL缓冲池

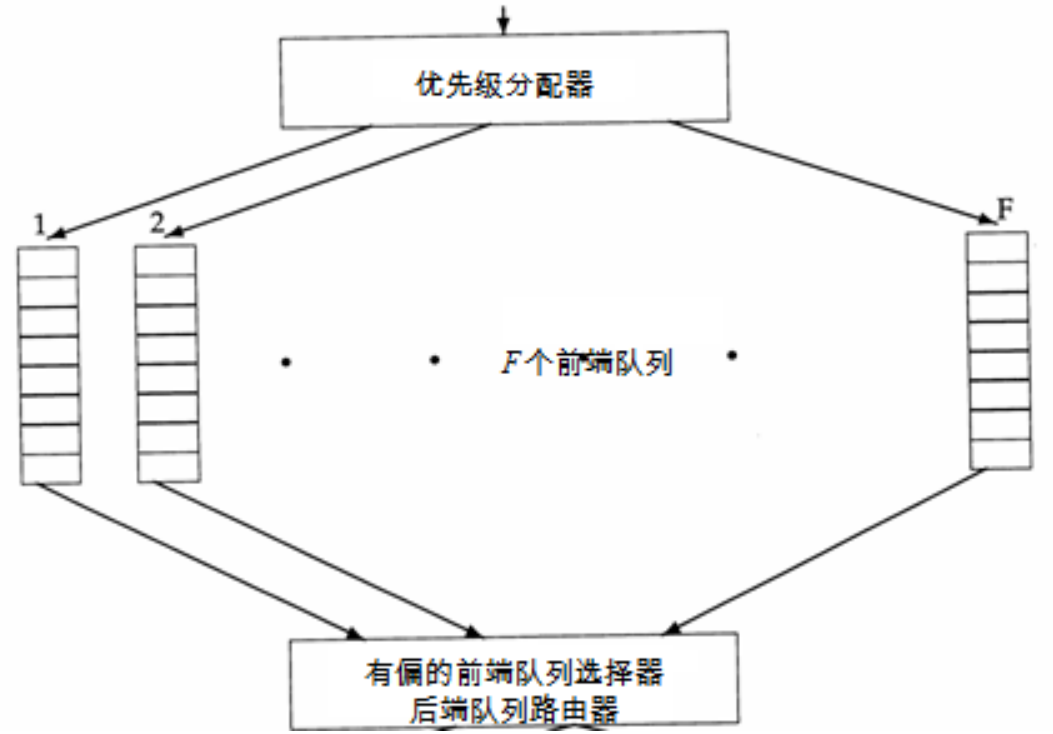


- URL从上部流入缓冲池
- 前端队列(Front queue)管理优先级
- 后端队列(Back queue) 实现礼貌性
- 每个队列都是FIFO (First in First out，先进先出)

Mercator 中的待采集URL缓冲池：前端队列(Front queue)

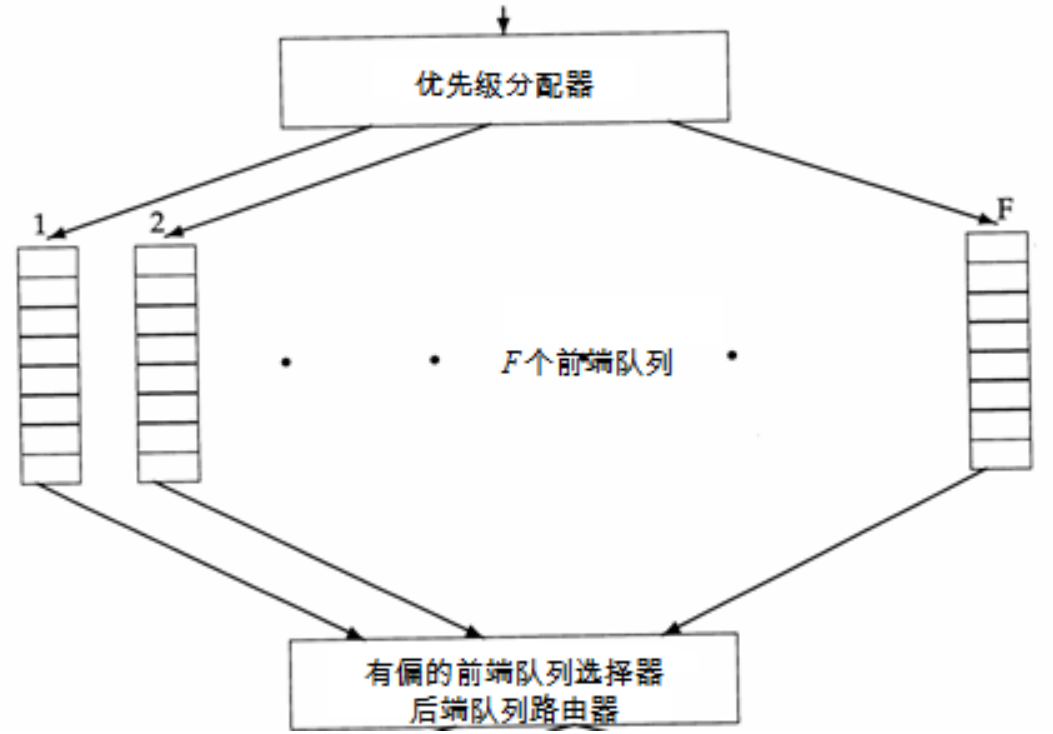


Mercator 中的待采集URL缓冲池：前端队列(Front queue)



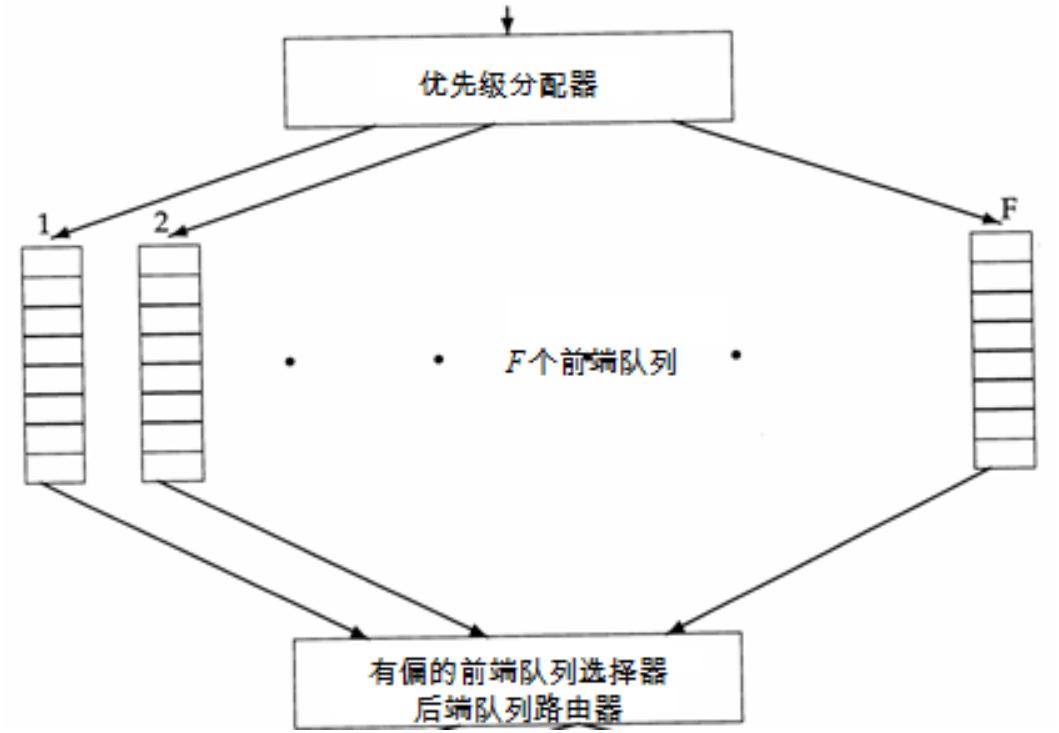
- 优先级分配器给每个URL分配一个0到F之间的优先级整数

Mercator 中的待采集URL缓冲池：前端队列(Front queue)



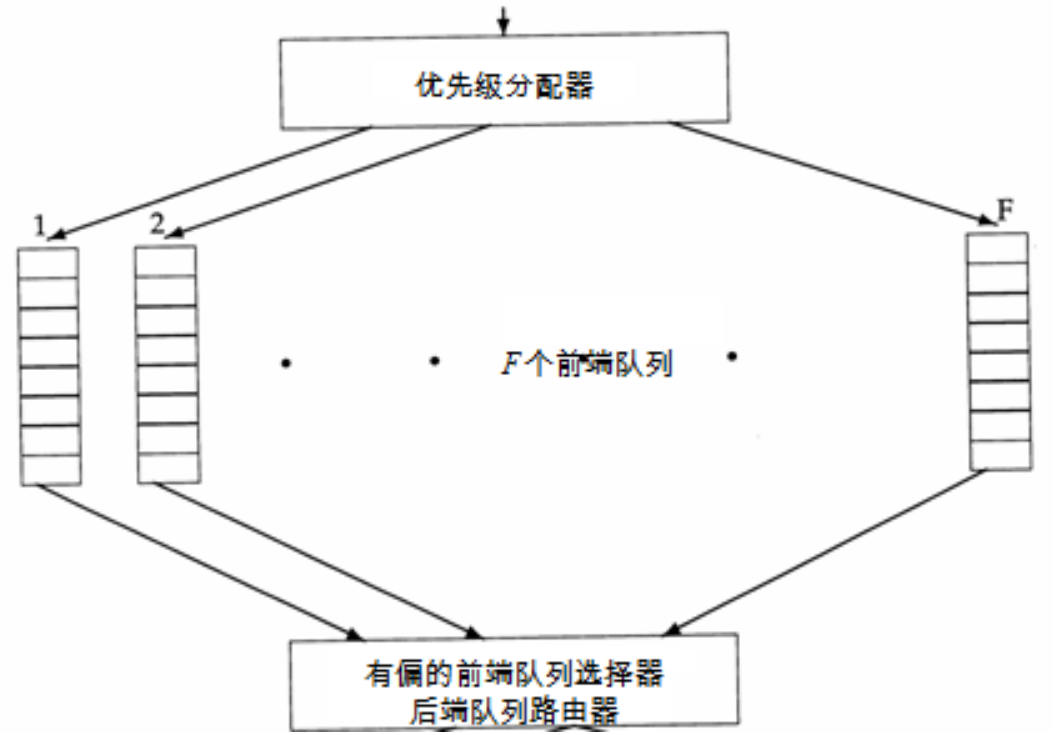
- 优先级分配器给每个URL分配一个0到F之间的优先级整数
- 然后将URL添加到相应的队列中

Mercator 中的待采集URL缓冲池：前端队列(Front queue)



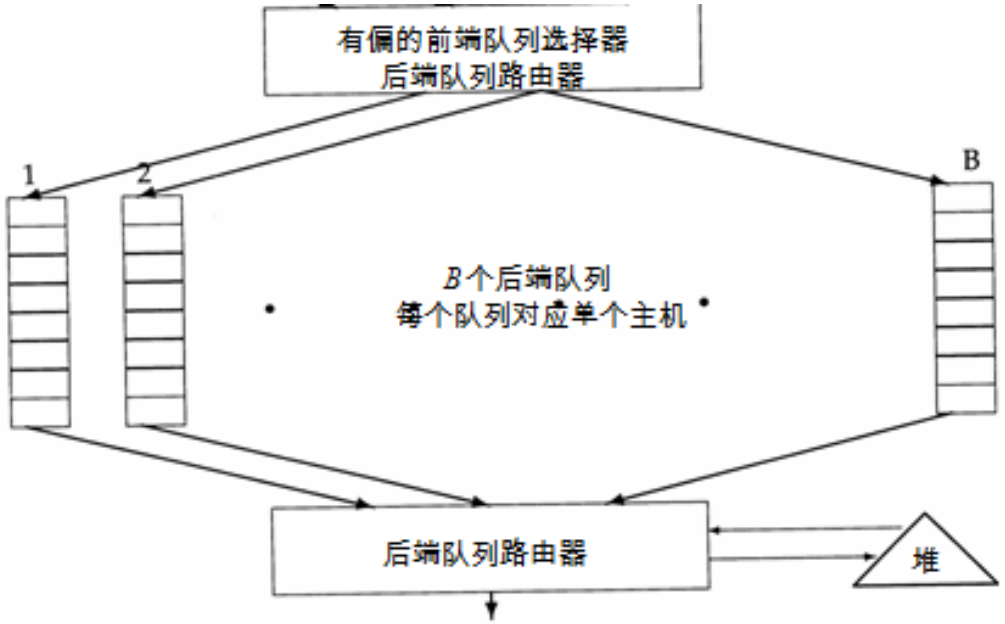
- 优先级分配器给每个URL分配一个0到F之间的优先级整数
- 然后将URL添加到相应的队列中
- 分配优先级可以基于启发式信息：更新率、PageRank等等

Mercator 中的待采集URL缓冲池：前端队列(Front queue)

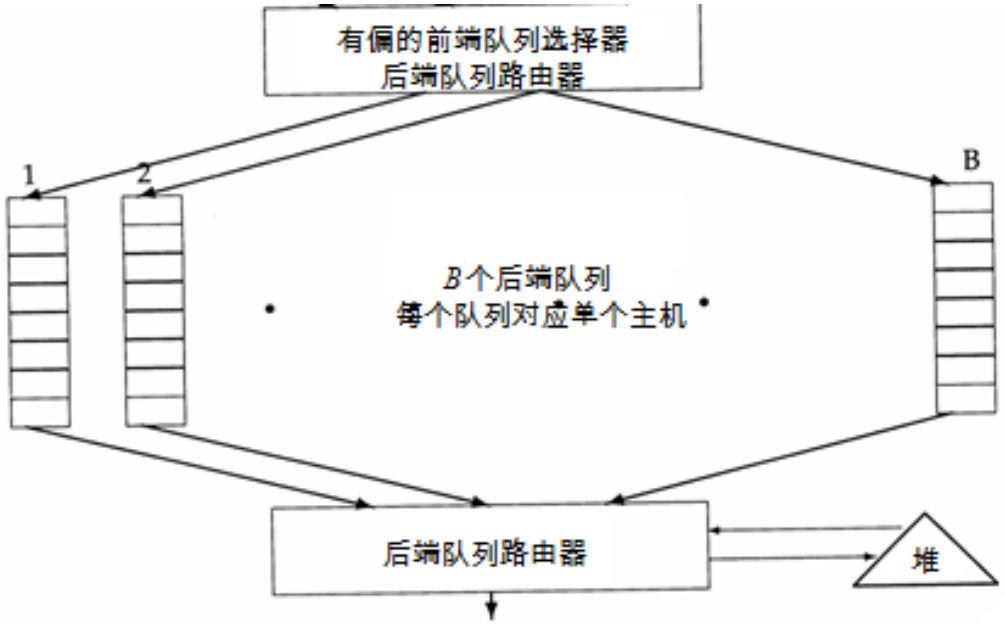


- 从前端队列中进行选择由后端队列发起
- 选择一个前端队列来选择下一个URL：轮询法(Round robin)、随机法或者更复杂的方法
- 但是上述选择过程倾向于高优先级的前端队列

Mercator 中的待采集URL缓冲池：后端队列(Back queue)

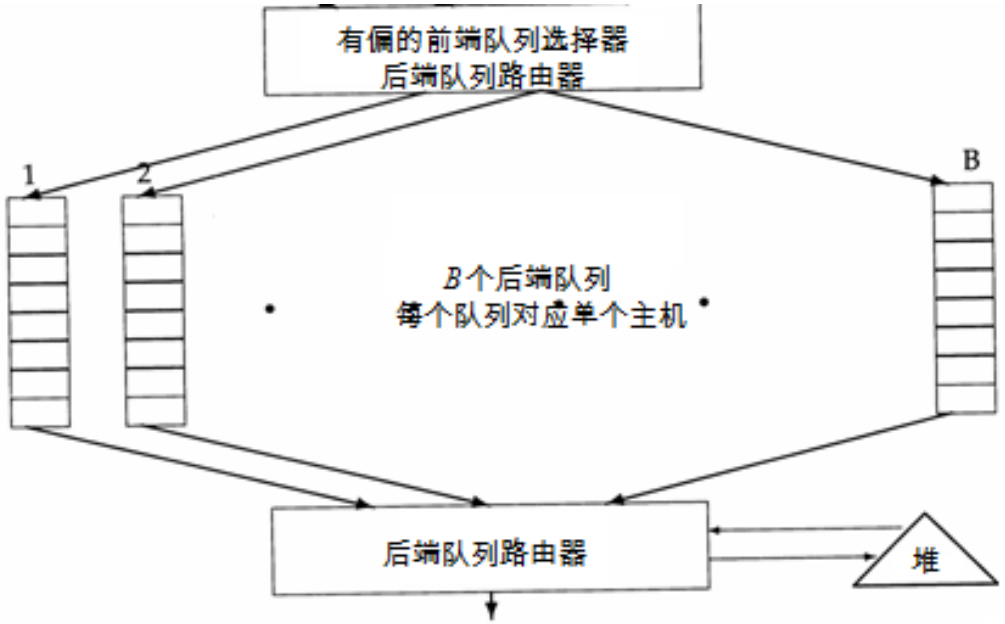


Mercator 中的待采集URL缓冲池：后端队列(Back queue)



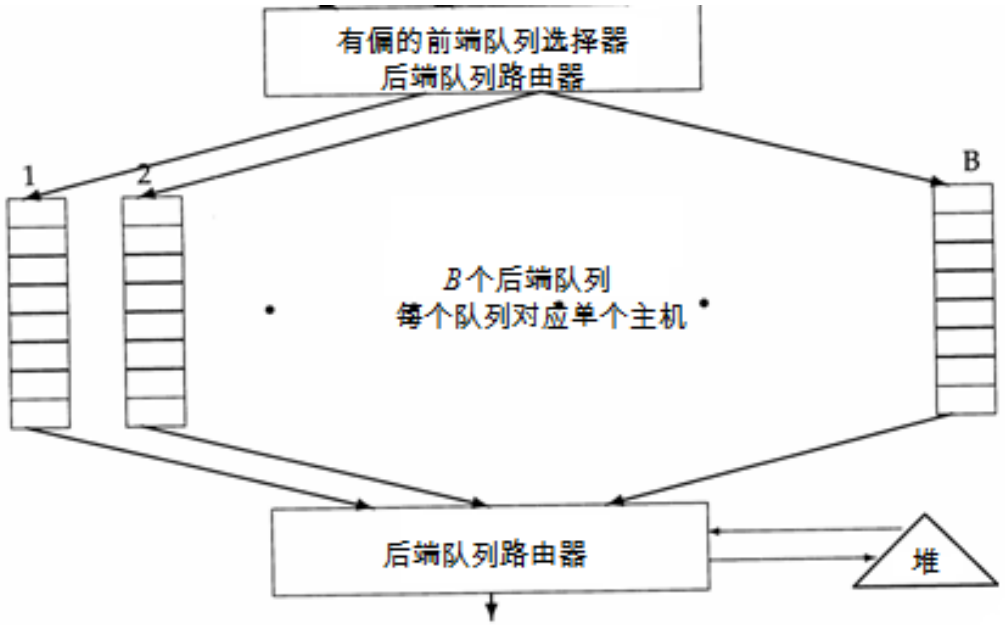
- 恒定情形1： 当采集器在运行时，每个后端队列不为空
- 恒定情形2： 每个后端队列中仅存放来自同一主机的URL
- 维护一张主机到后端队列的表

Mercator 中的待采集URL缓冲池：后端队列(Back queue)



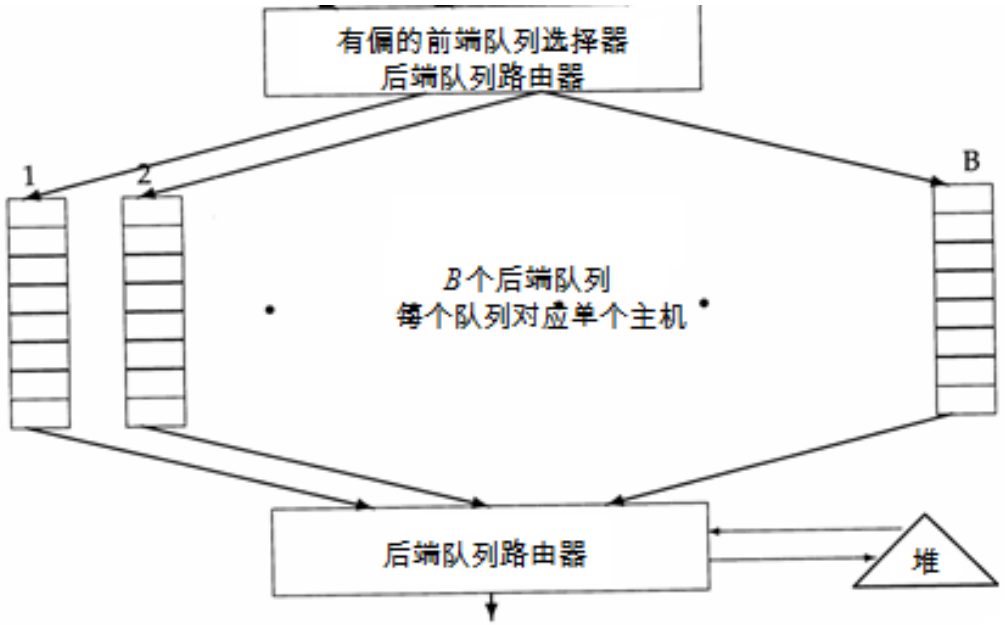
- 在堆中：
- 每个后端队列对应一个元素
- 元素值为该队列对应的主机重新访问的最早时间 t_e
- 该时间t_e 由下列因素确定 (i) 上次访问该主机的时间 (ii) 时间间隔的启发式方法

Mercator 中的待采集URL缓冲池 : 后端队列(Back queue)



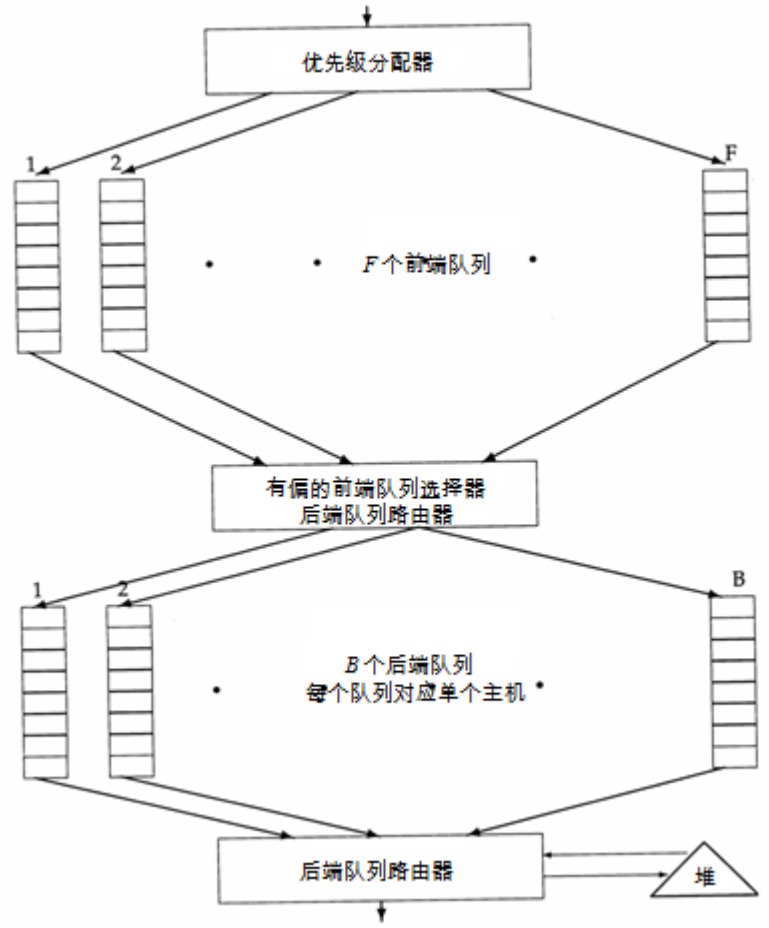
- 抓取器与后端队列交互方法：
- 重复下列操作： (i) 抽取堆中的当前根节点 q (q 是一个后端队列)
- 并且 (ii) 抓取q中的头部URL u . . .
- . . .直至 q 为空...
- (也就是说一直抓到 u为q中最后一个URL为止)

Mercator 中的待采集URL缓冲池 : 后端队列(Back queue)



- 一旦后端队列 q为空:
- 重复下列操作 (i) 从前端队列中将一系列URL u推入并且 (ii) 将 u 加到相应的后端队列中. . .
- . . . 直到得到一个u，u 的主机没有对应的后端队列为止
- 然后将 u 放入 q 并为它建立一个堆

Mercator 中的待采集URL缓冲池



- URLs从上流入到缓冲池
- 前端队列管理优先级
- 后端队列保证礼貌性

采集器陷阱(Spider trap)

- 一些恶意的服务器可以产生无穷的链接网页序列
- 一些复杂的采集器陷阱产生的页面不能简单地判断为动态页面

本讲小结

- 网页采集的概念
- 一个简单的采集器
- 一个真实的采集器Mercator

参考资料

- 《信息检索导论》第20章
- http://ifnlp.org/ir
  - Heydon等人撰写的有关Mercator的论文
  - 采集协议标准