



SCORING AND COMPLETE SEARCH SYSTEM

THIS LECTURE

- Speeding up vector space ranking
- Putting together a complete search system
 - Will require learning about a number of miscellaneous topics and heuristics

COMPUTING COSINE SCORES

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[ $d$ ] + =  $w_{t,d} \times w_{t,q}$ 
7  Read the array Length
8  for each  $d$ 
9  do Scores[ $d$ ] = Scores[ $d$ ] / Length[ $d$ ]
10 return Top  $K$  components of Scores[]
```

EFFICIENT COSINE RANKING

- Find the K docs in the collection “nearest” to the query $\Rightarrow K$ largest query-doc cosines.
- Efficient ranking:
 - Computing a single cosine efficiently.
 - Choosing the K largest cosine values efficiently.
 - Can we do this without computing all N cosines?

EFFICIENT COSINE RANKING

- What we're doing in effect: solving the K -nearest neighbor problem for a query vector
- In general, we do not know how to do this efficiently for high-dimensional spaces
- But it is solvable for short queries, and standard indexes support this well

SPECIAL CASE – UNWEIGHTED QUERIES

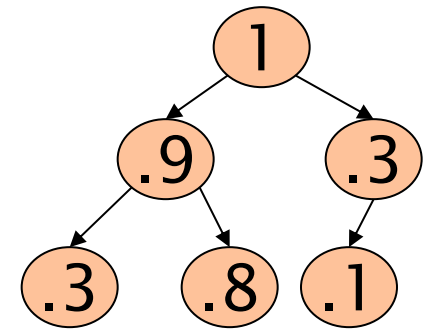
- No weighting on query terms
 - Assume each query term occurs only once
- Then for ranking, don't need to normalize query vector
 - Slight simplification of algorithm from previous lecture

COMPUTING THE K LARGEST COSINES: SELECTION VS. SORTING

- Typically we want to retrieve the top K docs (in the cosine ranking for the query)
 - not to totally order all docs in the collection
- Can we pick off docs with K highest cosines?
- Let J = number of docs with nonzero cosines
 - We seek the K best of these J

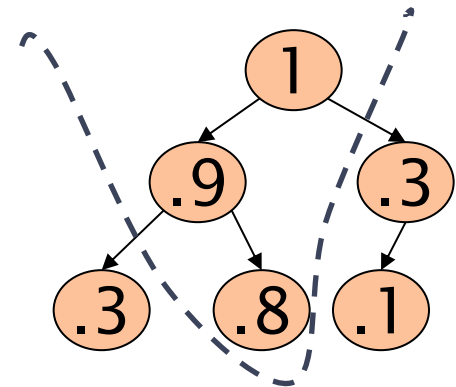
USE BINARY HEAP FOR SELECTING TOP K

- Shape property: a binary heap is a *complete binary tree*; that is, all levels of the tree, except possibly the last one (deepest) are fully filled, and, if the last level of the tree is not complete, the nodes of that level are filled from left to right.
- Heap property: the key stored in each node is either greater than or equal to (\geq) or less than or equal to (\leq) the keys in the node's children, according to some total order. Root stores the max or min value



USE BINARY HEAP FOR SELECTING TOP K

- Takes (worst-case) $2J$ operations to construct, then each of K “winners” read off in $2\log J$ steps. (Floyd’s method)
- For $J=1\text{M}$, $K=100$, this is about 10% of the cost of sorting.



QUIZ: HEAP

- Construct a binary max heap from the following array of keys:
[43, 21, 9, 23, 5, 28, 6, 12]
- Draw the binary heap as your answer.
- How many steps does it take to build this binary heap?
- How many steps does it take to get 5 “winners”?

BOTTLENECKS

- Previous approach was “exact”
- Primary computational bottleneck in scoring: cosine computation
- Can we avoid all this computation?
- Yes, but may sometimes get it wrong
 - a doc *not* in the top K may creep into the list of K output docs
 - Is this such a bad thing?

COSINE SIMILARITY IS ONLY A PROXY

- User has a task and a query formulation
- Cosine matches docs to query
- Thus cosine is anyway a proxy for user happiness
- If we get a list of K docs “close” to the top K by cosine measure, should be ok
 - Approximate solution!

GENERIC APPROACH

- Find a set A of *contenders*, with $K < |A| \ll N$
 - A does not necessarily contain the top K , but has many docs from among the top K
 - Return the top K docs in A
- Think of A as pruning non-contenders
- The same approach is also used for other (non-cosine) scoring functions
- Will look at several schemes following this approach

INDEX ELIMINATION

- Basic algorithm:
 - cosine computation algorithm only considers docs containing at least one query term
- Take this further:
 - Only consider high-idf query terms
 - Only consider docs containing many query terms

HIGH-IDF QUERY TERMS ONLY

- For a query such as *catcher in the rye*
- Only accumulate scores from *catcher* and *rye*
- Intuition: *in* and *the* contribute little to the scores and so don't alter rank-ordering much
- Benefit:
 - Postings of low-idf terms have many docs → these (many) docs get eliminated from set A of contenders
 - Save a lot of time!

DOCS CONTAINING MANY QUERY TERMS

- Any doc with at least one query term is a candidate for the top K output list
- For multi-term queries, only compute scores for docs containing several of the query terms
 - Say, at least 3 out of 4
 - Imposes a “soft conjunction” on queries seen on web search engines (early Google)
- Easy to implement in postings traversal

3 OF 4 QUERY TERMS

Antony	⇒	3	4	8	16	32	64	128	
Brutus	⇒	2	4	8	16	32	64	128	
Caesar	⇒	1	2	3	5	8	13	21	34
Calpurnia	⇒	13	16	32					

Concurrent Traversal

Scores only computed for docs 8, 16 and 32.

CHAMPION LISTS

- Precompute for each dictionary term t , the r docs of highest weight in t 's postings
 - Call this the champion list for t
 - (aka fancy list or top docs for t)
- Note that r has to be chosen at index build time
 - Thus, it's possible that $r < K$
- At query time, only compute scores for docs in the champion list of some query term
 - Pick the K top-scoring docs from amongst these

STATIC QUALITY SCORES

- We want top-ranking documents to be both *relevant* and *authoritative*
- *Relevance* is being modeled by cosine scores
- *Authority* is typically a query-independent property of a document
- Examples of authority signals
 - Wikipedia among average websites
 - Articles in certain newspapers
 - A paper with many citations
 - Many bitly's, diggs or del.icio.us marks
 - Pagerank

Quantitative





COME AT ME, BRO

Why Ukraine Has Been So Successful In Defending Itself Against Russia

Video

Ukraine has been able to mount a formidable defense against one of the world's strongest armies. Here's how they've pulled it off so far.

♡ 1,103 📌 f 🐦



YOU GOTTA WATCH THIS!

50 Of The Best Cult Classic Movies Released This Century

lifehacker.com

If you've ever shouted "Oh hi, Mark!" for no good reason, you know what we're talking about.

♡ 476 📌 f 🐦



FOLLOW FOR UPDATES

Read The Latest News From The Russian Invasion Of Ukraine Here

Russia began invading Ukraine on Thursday, February 24. Since then thousands of people have been killed and over a million refugees have been displaced. Multiple talks between the two countries to negotiate peace have failed.

♡ 14,148 📌 f 🐦



PUT THE SCISSORS DOWN

Apparently, We've Been Opening Spaghetti Packets The Wrong Way

Video

Our life has changed thanks to Nona Elda Cooks.

♡ 1,763 📌 f 🐦



NO CAKE WALK

As Russian Troop Deaths Climb, Morale Becomes An Issue, Officials Say

nytimes.com

More than 7,000 Russian troops have been killed in less than three weeks of fighting, according to conservative U.S. estimates.

♡ 640 📌 f 🐦



THE SIX NOT-SO-SECRET INGREDIENTS

What Is SPAM Made Of?

mentalfloss.com

The recipe for SPAM isn't as complicated as its reputation would have you think.

♡ 679 📌 f 🐦

SHORTEN. SHARE. MEASURE.

Join Bitly, the world's leading link management platform.

Paste a link to shorten it

SHORTEN

SIGN UP FOR FREE

or Learn more →

UNLEASH THE POWER OF THE LINK

The link connects your marketing efforts to how and where your customers experience your brand. Are you getting the most out of it?



SHORTEN



SHARE



MEASURE

MODELING AUTHORITY

- Assign to each document a *query-independent quality score* in $[0,1]$ to each document d
 - Denote this by $g(d)$
- Thus, a quantity like the number of citations is scaled into $[0,1]$
- Quiz: suggest a formula for this quality score.

NET SCORE

- Consider a simple total score combining cosine relevance and authority
- $\text{net-score}(q,d) = g(d) + \text{cosine}(q,d)$
 - Can use some other linear combination
 - Indeed, any function of the two “signals” of user happiness – more later
- Now we seek the top K docs by net score

TOP K BY NET SCORE – FAST METHODS

- First idea: Order all postings by $g(d)$
- Key: there is a common ordering for all postings
- Thus, can concurrently traverse query terms' postings for
 - Postings intersection
 - Cosine score computation

WHY ORDER POSTINGS BY $G(D)$?

- Under $g(d)$ -ordering, top-scoring docs likely to appear early in postings traversal
- In time-bound applications (say, we have to return whatever search results we can in 50 ms), this allows us to stop postings traversal early
 - Short of computing scores for all docs in postings

QUIZ: G(D)-ORDERING

- How shall we order the postings by $g(d)$ score, by increasing order or decreasing order? Why?

CHAMPION LISTS IN $G(D)$ -ORDERING

- Can combine champion lists with $g(d)$ -ordering
- Maintain for each term a champion list of the r docs with highest $g(d) + \text{tf-idf}_{td}$
- Seek top- K results from only the docs in these champion lists

HIGH AND LOW LISTS

- For each term, we maintain two postings lists called *high* and *low*
 - Think of *high* as the champion list
- When traversing postings on a query, only traverse *high* lists first
 - If we get more than K docs, select the top K and stop
 - Else proceed to get docs from the *low* lists
- Can be used even for simple cosine scores, without global quality $g(d)$
- A means for segmenting index into two tiers

IMPACT-ORDERED POSTINGS

- So far: *global ordering* and *concurrent traversal*
- We only want to compute scores for docs for which $tf_{t,d}$ is high enough
- We sort each postings list by $tf_{t,d}$
- Now: not all postings in a common order!
 - high $tf_{t,d}$ doesn't mean high authority
- How do we compute scores in order to pick off top K ?
 - Two ideas follow

1. EARLY TERMINATION

- When traversing t 's postings, stop early after either
 - a fixed number of r docs
 - $tf_{t,d}$ drops below some threshold
- Take the **union** of the resulting sets of docs
 - One from the postings of each query term
- Compute only the scores for docs in this union

2. IDF-ORDERED TERMS

- When considering the postings of query terms
- Look at them in order of **decreasing idf**
 - High idf terms likely to contribute most to score
- As we update score contribution from each query term
 - Stop if doc scores relatively unchanged
- Can apply to cosine or some other net scores

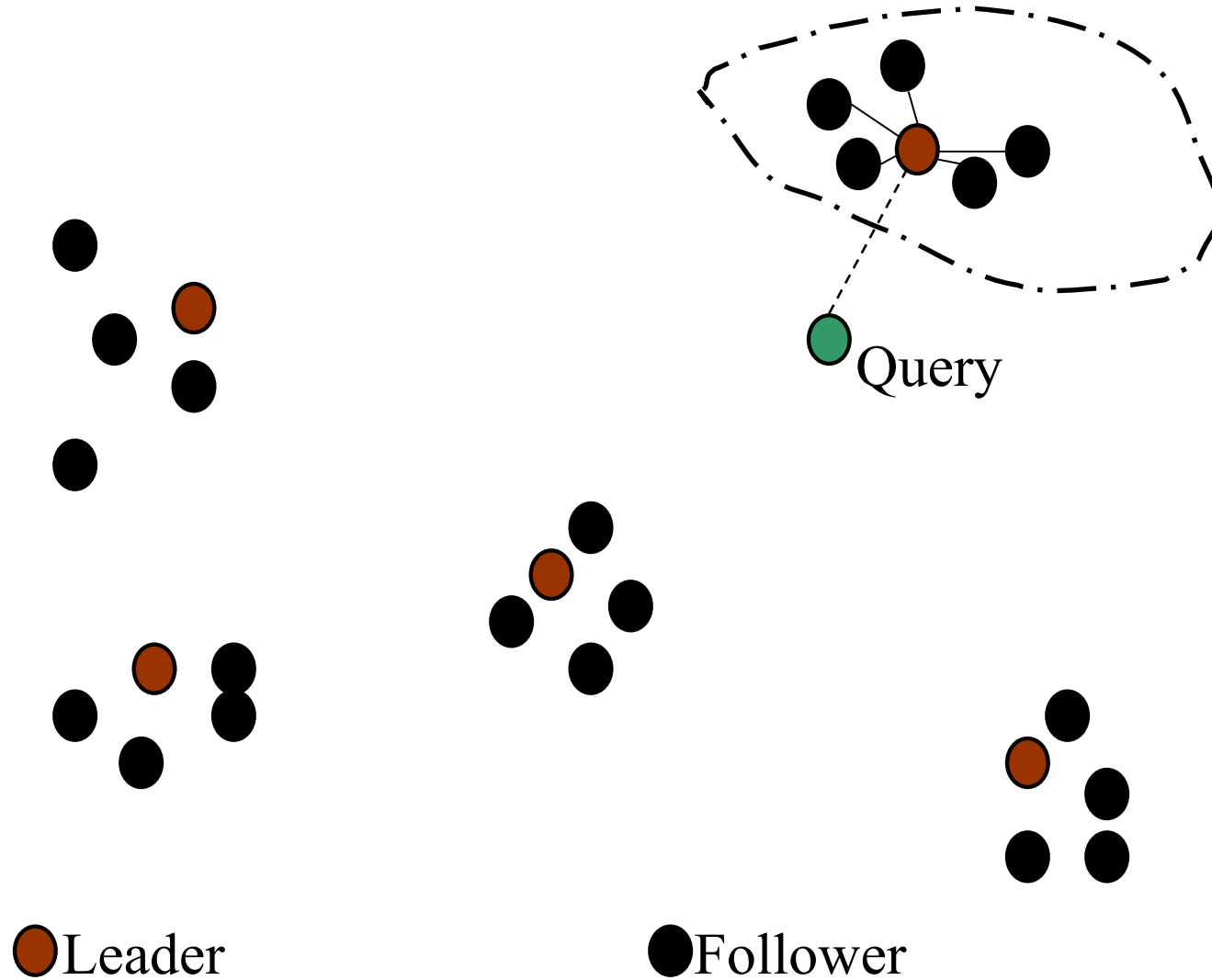
CLUSTER PRUNING: PREPROCESSING

- Pick \sqrt{N} *docs* at random: call these *leaders*
- For every other doc, pre-compute nearest leader
 - Docs attached to a leader: its *followers*;
 - Likely: each leader has $\sim \sqrt{N}$ followers.

CLUSTER PRUNING: QUERY PROCESSING

- Process a query as follows:
 - Given query Q , find its nearest *leader* L .
 - Seek K nearest docs from among L 's followers.

VISUALIZATION



WHY USE RANDOM SAMPLING

- Fast
- Leaders reflect data distribution

GENERAL VARIANTS

- Have each follower attached to $b_1=3$ (say) nearest leaders.
- From query, find $b_2=4$ (say) nearest leaders and their followers.
- Can recurse on leader/follower construction.

QUIZ: NEAREST LEADER

- Given a query, and N docs, to find the nearest leader, how many cosine computations do we do?
 1. N
 2. $N \log N$
 3. N^2
 4. \sqrt{N}

PARAMETRIC AND ZONE INDEXES

- Thus far, a doc has been a sequence of terms
- In fact documents have multiple parts, some with **special semantics**:
 - Author
 - Title
 - Date of publication
 - Language
 - Format
 - etc.
- These constitute the metadata about a document

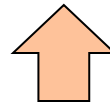
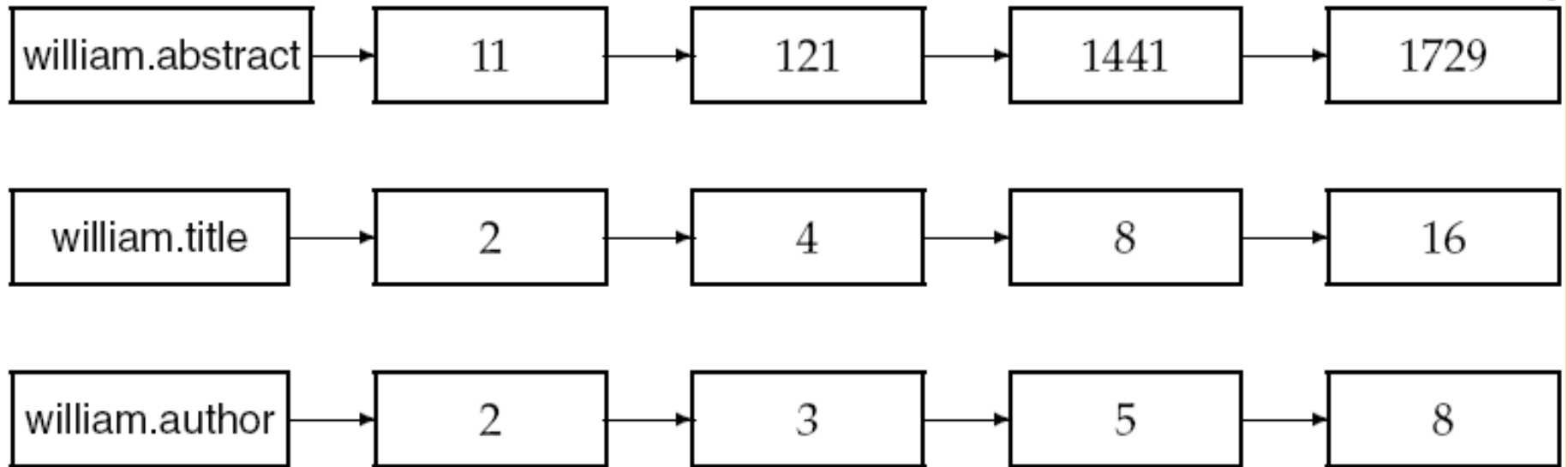
FIELDS

- We sometimes wish to search by these metadata
 - E.g., find docs authored by William Shakespeare in the year 1601, containing *alas poor Yorick*
- Year = 1601 is an example of a field
- Also, author last name = shakespeare, etc.
- Field or parametric index: postings for each field value
 - Sometimes build range trees (e.g., for dates)
- Field query typically treated as conjunction
 - (doc *must* be authored by shakespeare)

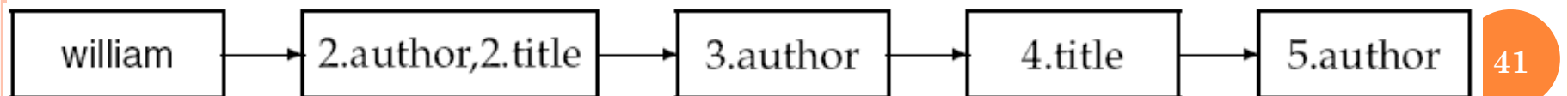
ZONE

- A zone is a region of the doc that can contain an arbitrary amount of text, e.g.,
 - Title
 - Abstract
 - References ...
- Build inverted indexes on zones as well to permit querying
- E.g., “find docs with *merchant* in the title zone and matching the query *gentle rain*”

EXAMPLE ZONE INDEXES



Encode zones in dictionary vs. postings.



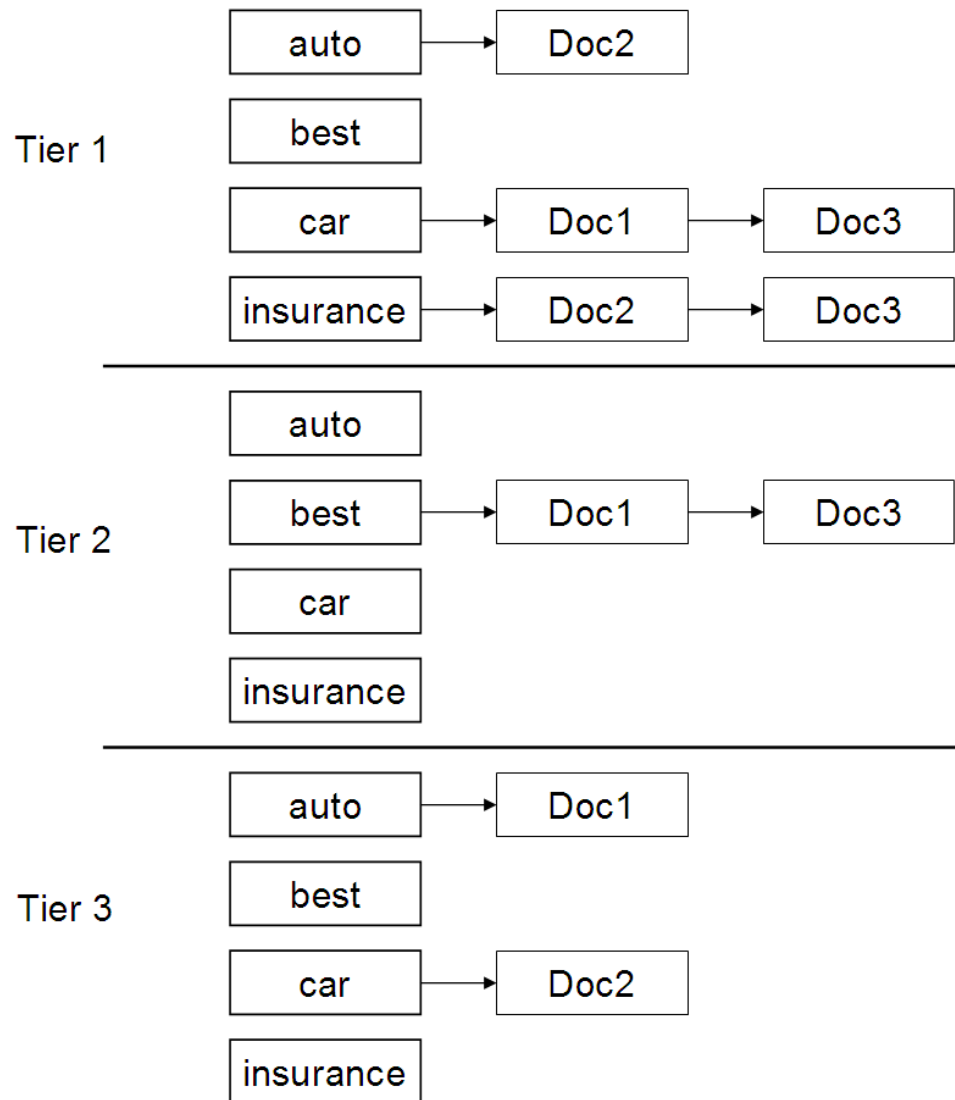
QUIZ: ZONE INDEX

- Which one of the previous approaches is better if I want to search for “William” in the abstract of a doc? Why?
 - a) encode zones in the dictionary
 - b) encode zones in the postings

TIERED INDEXES

- Break postings up into a hierarchy of lists
 - Most important
 - ...
 - Least important
- Can be done by $g(d)$ or another measure
- Inverted index thus broken up into tiers of decreasing importance
- At query time use top tier unless it fails to yield K docs
 - If so drop to lower tiers

EXAMPLE TIERED INDEX



QUERY TERM PROXIMITY

- Free text queries: just a set of terms typed into the query box – common on the web
- Users prefer docs in which query terms occur within close proximity of each other
- Let w be the smallest window in a doc containing all query terms, e.g.,
 - For the query *strained mercy* the smallest window in the doc *The quality of **mercy is not strained*** is 4 (words)
- Would like scoring function to take this into account – how? (we don't know yet.)

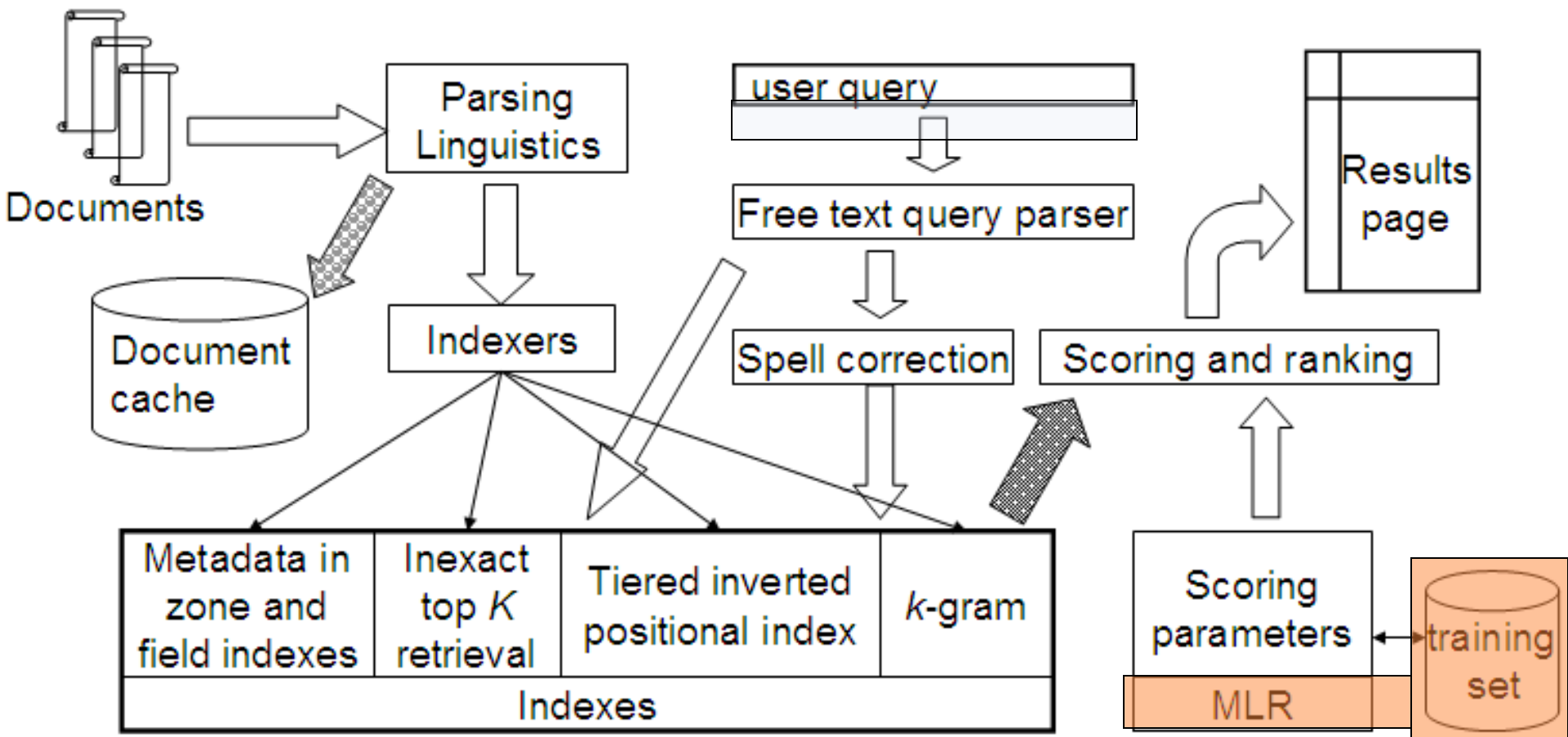
QUERY PARSERS

- Free text query from user may in fact spawn one or more queries to the indexes, e.g., query *rising interest rates*
 - Run the query as a phrase query
 - If $<K$ docs contain the phrase *rising interest rates*, run the two phrase queries *rising interest* and *interest rates*
 - If we still have $<K$ docs, run the vector space query *rising interest rates*
 - Rank matching docs by vector space scoring
- This sequence is issued by a query parser

AGGREGATE SCORES

- We've seen that score functions can combine cosine, static quality, proximity, etc.
- How do we know the best combination?
- Some applications – *expert-tuned*
- Increasingly common: *machine-learned*
 - In later lecture

PUTTING IT ALL TOGETHER



Components we have introduced thus far

- Document preprocessing (linguistic and otherwise)
- Positional indexes
- Tiered indexes
- Spelling correction
- k -gram indexes for wildcard queries and spelling correction
- Query processing
- Document scoring
- Term-at-a-time processing

Components we haven't covered yet

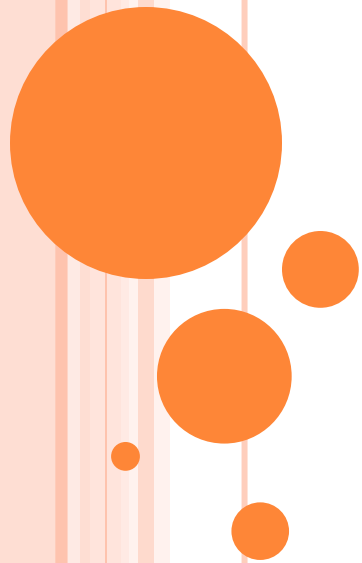
- Document cache: we need this for generating snippets (=dynamic summaries)
- Machine-learned ranking functions
- Proximity ranking (e.g., rank documents in which the query terms occur in the same local window higher than documents in which the query terms occur far from each other)
- Query parser

Vector space retrieval: Interactions

- How do we combine phrase retrieval with vector space retrieval?
- We do not want to compute document frequency / idf for every possible phrase. Why?
- How do we combine Boolean retrieval with vector space retrieval?
 - For example: “+”-constraints and “-”-constraints
- Post-filtering is simple, but can be very inefficient – no easy answer.
- How do we combine wild cards with vector space retrieval?
 - Again, no easy answer

RESOURCES

- IIR 7, 6.1



WEB SEARCH BASICS

OVERVIEW

- Big picture
- Ads – they pay for the web
- Duplicate detection – addresses one aspect of chaotic content creation
- Spam detection – addresses one aspect of lack of central access control
- Size of the web

The left side of the slide features a series of vertical stripes in shades of brown, tan, and grey. Overlaid on these stripes are several orange circles of varying sizes. One large circle is positioned near the top left, and several smaller circles are scattered below it, some overlapping the stripes.

Big Picture

55

BRIEF (NON-TECHNICAL) HISTORY

- Early keyword-based engines ca. 1995-1997
 - Altavista, Excite, Infoseek, Inktomi, Lycos
- Paid search ranking: Goto (morphed into Overture.com → Yahoo!)
 - Your search ranking depended on how much you paid
 - Auction for keywords: **casino** was expensive!

BRIEF (NON-TECHNICAL) HISTORY

- 1998+: Link-based ranking pioneered by Google
 - Blew away all early engines save Inktomi
 - Great user experience in search of a business model
 - Meanwhile Goto/Overture's annual revenues were nearing \$1 billion
- Result: Google added paid search “ads” to the side, independent of search results
 - Yahoo followed suit, acquiring Overture (for paid placement) and Inktomi (for search)
- 2005+: Google gains search share, dominating in Europe and very strong in North America
 - 2009: Yahoo! and Microsoft propose combined paid search offering

nigritude ultramarine - Google Search - Mozilla Firefox

File Edit View Go Bookmarks Yahoo! Tools Help

http://www.google.com/search?hl=en&q=nigritude+ultramarine&btnG=Google+Search

Getting Started Latest Headlines

pragh60@gmail.com | My Account | Sign out

Web Images Groups News Froogle Local more »

Google nigritude ultramarine Search Advanced Search Preferences

Web Results 1 - 10 of about 185,000 for **nigritude ultramarine**. (0.35 seconds)

Anil Dash: Nigritude Ultramarine
Do me a favor: Link to this post with the phrase **Nigritude Ultramarine**. ... Just placed a link to your **Nigritude Ultramarine** article on my weblog. Cheers! ...
www.dashes.com/anil/2004/06/04/nigritude_ultra - 101k - Mar 1, 2006 -
[Cached](#) - [Similar pages](#)

Nigritude Ultramarine FAQ
Nigritude Ultramarine FAQ - frequently asked questions about **nigritude ultramarine** and the realted SEO contest.
www.nigritudeultramarines.com/ - 59k - [Cached](#) - [Similar pages](#)

SEO contest - Wikipedia, the free encyclopedia
The **nigritude ultramarine** competition by SearchGuild is widely acclaimed as ...
Comparison of search results for **nigritude ultramarine** during and after the ...
en.wikipedia.org/wiki/Nigritude_ultramarine - 37k - [Cached](#) - [Similar pages](#)

Slashdot | How To Get Googled, By Hook Or By Crook
The current 3rd result showcases the "**Nigritude Ultramarine** Fighting Force" who ... When discussing **nigritude ultramarine** [slashdot.org] it is important to ...
slashdot.org/article.pl?sid=04/05/09/1840217 - 110k - [Cached](#) - [Similar pages](#)

The Nigritude Ultramarine Search Engine Optimization Contest
It's sweeping the web -- or at least search engine optimizers -- a new contest to rank tops for the term **nigritude ultramarine** on Google.
searchenginewatch.com/sereport/article.php/3360231 - 57k - [Cached](#) - [Similar pages](#)

Sponsored Links

Business Blogging Seminar
ing to L.A. March 16
Top bloggers reveal key techniques
www.blogbusinesssummit.com
Los Angeles, CA

Full-Time SEO & SEM Jobs
Find companies big & small hiring full-time SEO & SEM pros right now
CareerBuilder.com

SEO Contests
Information on SEO Contests like the **Nigritude Ultramarine** contest.
www.seo-contests.com/

The SEO Book
Nigritude Ultramarine & SEO secrets
Fun, free, raw, & different.
www.seobook.com

Algorithmic results.

Done

All

News

Images

Videos

Maps

More

Settings

Tools

About 254,000,000 results (0.72 seconds)

Top things to do in Cape Town







Table Mountain
Mountain for hiking, climbing & biking




Cape of Good Hope
Scenic spot & tip of the Cape Peninsula



Kirstenbosch National Botanica...
Nature reserve with mountain views



Boulders Beach
Sandy cove with resident penguin

 Cape Town travel guide

Cape Town Tourism: The Official Guide To Cape Town



www.capetown.travel/ ▾

The official guide on things to do, places to see, the best restaurants to eat at, and everthing you need to know about staying in **Cape Town**.

[The best time of year to visit ...](#) · [Contact Us](#) · [50 things to do in Cape Town](#) · [Stay](#)

People also ask

- Is Cape Town safe for tourists? ▾
- What is the best time to go to Cape Town? ▾
- Is it safe to vacation in South Africa? ▾
- Is South Africa safe in 2018? ▾



data ©2019 AfriGIS (Pty) Ltd, Google

Cape Town


Travel


Cape Town is a port city on South Africa's southwest coast, on a peninsula beneath the imposing Table Mountain. Slowly rotating cable cars climb to the mountain's flat top, from which there are sweeping views of the city, the busy harbor and boats heading for Robben Island, the notorious prison that once held Nelson Mandela, which is now a living museum.

Local time: Friday 6:00 AM

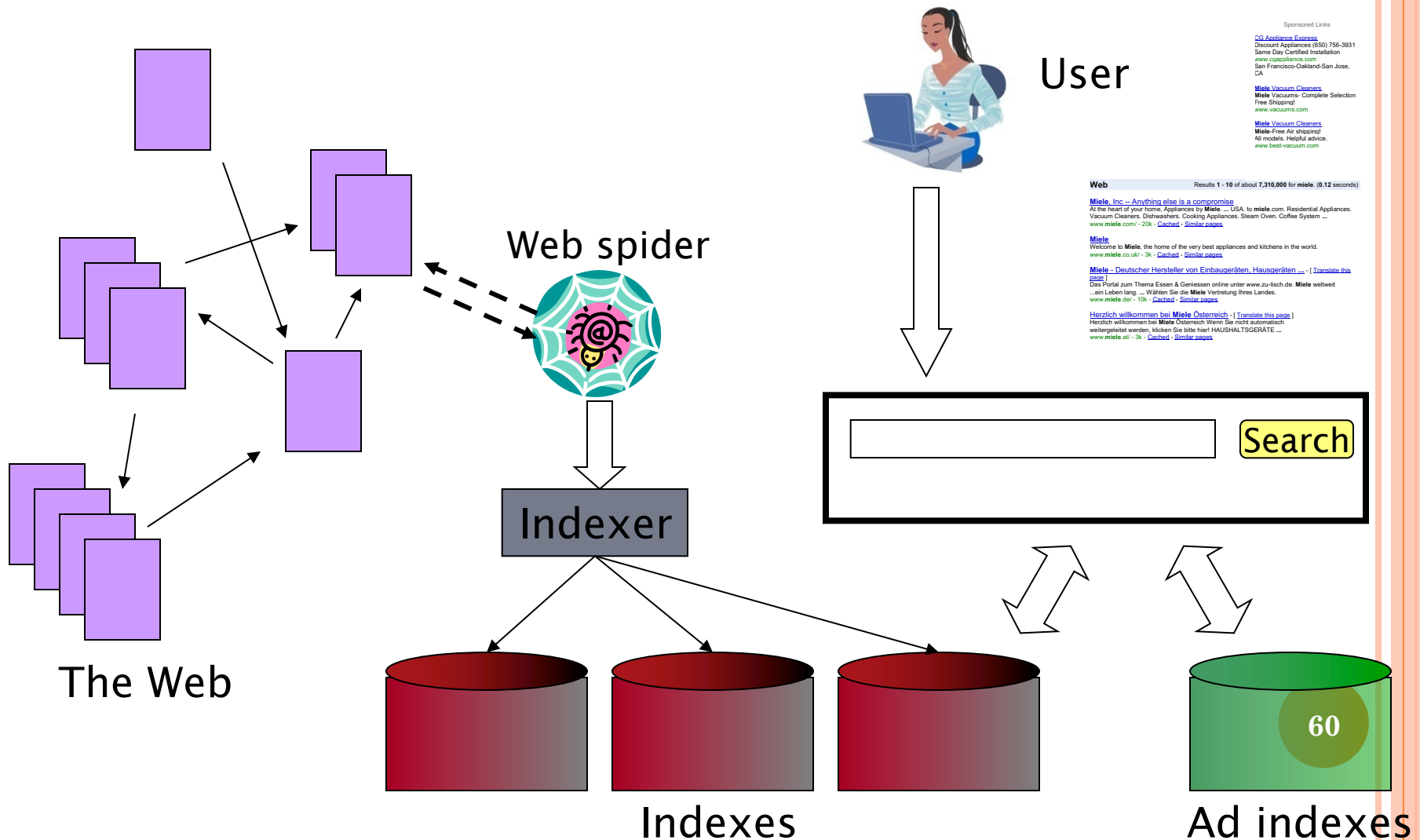
Weather: 64°F (18°C), Wind S at 3 mph (5 km/h), 92% Humidity

Plan a trip

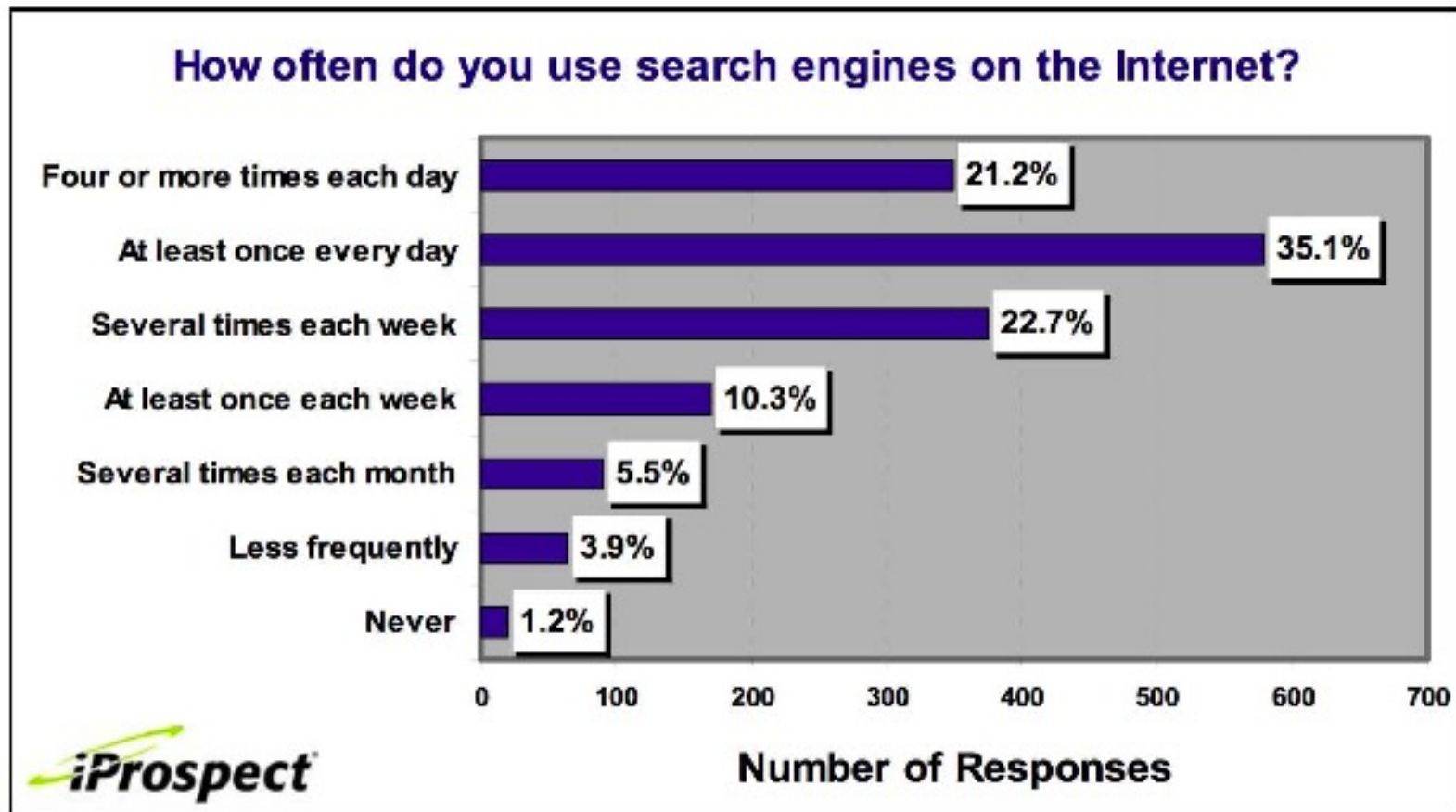
 Cape Town travel guide

 3-star hotel averaging \$67, 5-star averaging \$175

WEB SEARCH BASICS



Search is the top activity on the web



Without search engines, the web wouldn't work

- Without search, content is hard to find.
- → Without search, there is no incentive to create content.
 - Why publish something if nobody will read it?
 - Why publish something if I don't get ad revenue from it?
- Somebody needs to pay for the web.
 - Servers, web infrastructure, content creation
 - A large part today is paid by search ads.
 - Search pays for the web.

Interest aggregation

- Unique feature of the web: A small number of geographically dispersed people with similar interests can find each other.
 - Elementary school kids with hemophilia
 - People interested in translating R5R5 Scheme into relatively portable C (open source project)
 - Search engines are a key enabler for interest aggregation.

IR on the web vs. IR in general

- On the web, search is not just a nice feature.
 - Search is a key enabler of the web: . . .
 - . . . financing, content creation, interest aggregation etc.
- Web need financing → look at search ads
- The web is a chaotic und uncoordinated collection. → lots of duplicates – need to detect duplicates
- No control / restrictions on who can author content → lots of spam – need to detect spam
- The web is very large. → need to know how big it is

USER NEEDS

◦ Need [Brod02, RL04]

- **Informational** – want to learn about something (~40% / 65%)

Low hemoglobin

- **Navigational** – want to go to that page (~25% / 15%)

United Airlines

- **Transactional** – want to do something (web-mediated) (~35% / 20%)

Seattle weather

- Access a service

Mars surface images

- Downloads

Canon S410

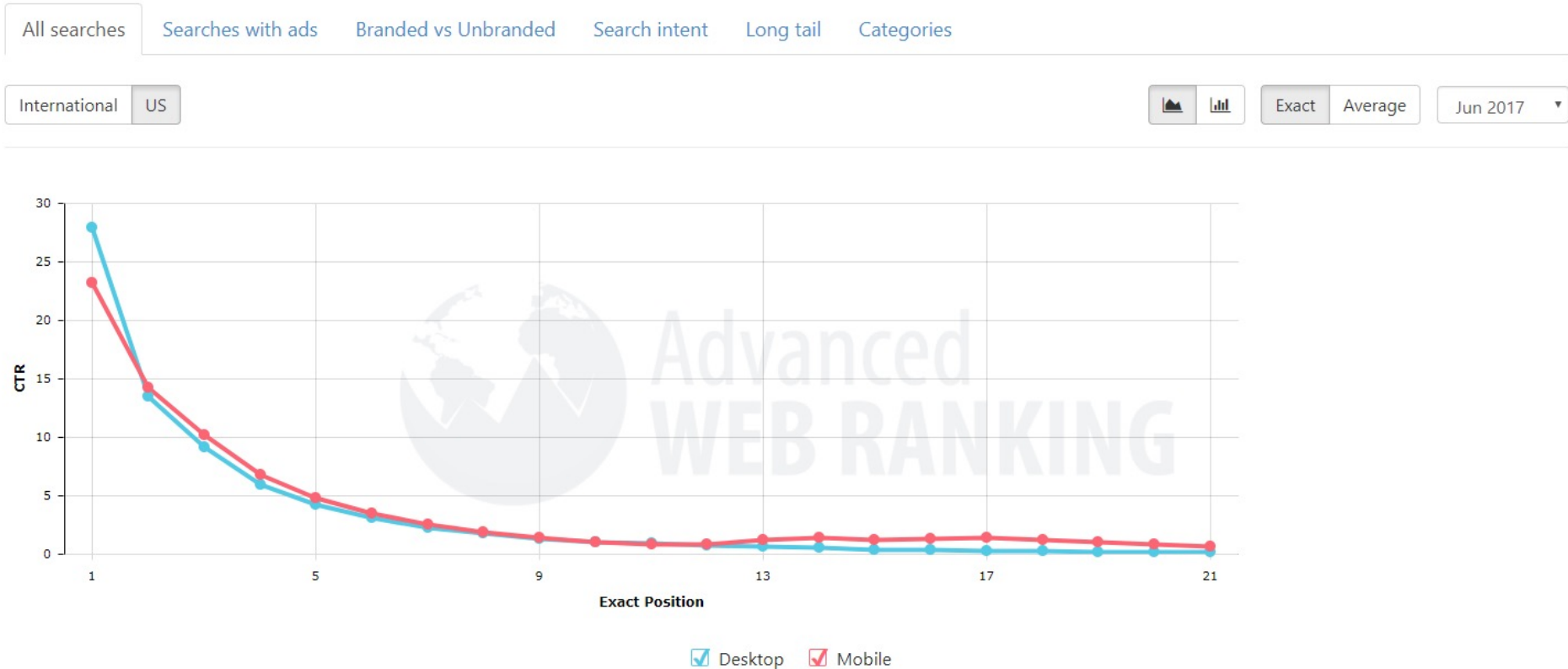
- Shop

- **Gray areas**

Car rental Brasil

- Find a good hub
- Exploratory search “see what’s there”

HOW FAR DO PEOPLE LOOK FOR RESULTS?



71% CTR for page 1, 6% for page 2 + 3 combined

Page 1 receives 95% traffic, remaining pages get 5%

$CTR = \text{clicks} / \text{impressions}$

USERS' EMPIRICAL EVALUATION OF RESULTS

- Quality of pages varies widely
 - Relevance is not enough
 - Other desirable qualities (non IR!!)
 - Content: Trustworthy, diverse, non-duplicated, well maintained
 - Web readability: display correctly & fast
 - No annoyances: pop-ups, etc.
- Precision vs. recall
 - On the web, recall seldom matters
- What matters
 - Precision at 1? Precision above the fold?
 - Comprehensiveness – must be able to deal with obscure queries
 - Recall matters when the number of matches is very small
- User perceptions may be unscientific, but are significant over a large aggregate

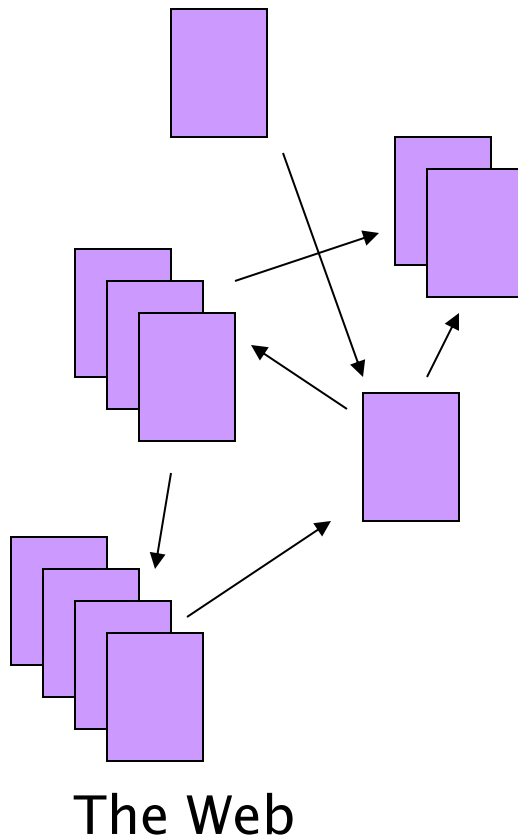
QUIZ: WINNING FACTOR

- What are the important factors in the quality of web search? (multiple correct answers)
 - a) Readability
 - b) Precision
 - c) Recall
 - d) Relevance

USERS' EMPIRICAL EVALUATION OF ENGINES

- Relevance and validity of results
- UI – Simple, no clutter, error tolerant
- Trust – Results are objective
- Coverage of topics for polysemic queries
- Pre/Post process tools provided
 - Mitigate user errors (auto spell check, search assist,...)
 - Explicit: Search within results, more like this, refine ...
 - Anticipative: related searches
- Deal with idiosyncrasies
 - Web specific vocabulary
 - Impact on stemming, spell-check, etc.
 - Web addresses typed in the search box
- “The first, the last, the best and the worst ...”

THE WEB DOCUMENT COLLECTION



- No design/co-ordination
- Distributed content creation, linking, democratization of publishing
- Content includes truth, lies, obsolete information, contradictions ...
- Unstructured (text, html, ...), semi-structured (XML, annotated photos), structured (Databases)...
- Scale much larger than previous text collections ... but corporate records are catching up
- Growth – slowed down from initial “volume doubling every few months” but still expanding
- Content can be *dynamically generated*