

Web Dynamics

Part 3 – Searching the Dynamic Web

3.1 Crawling and recrawling policies

3.2 Accessing the Hidden Web

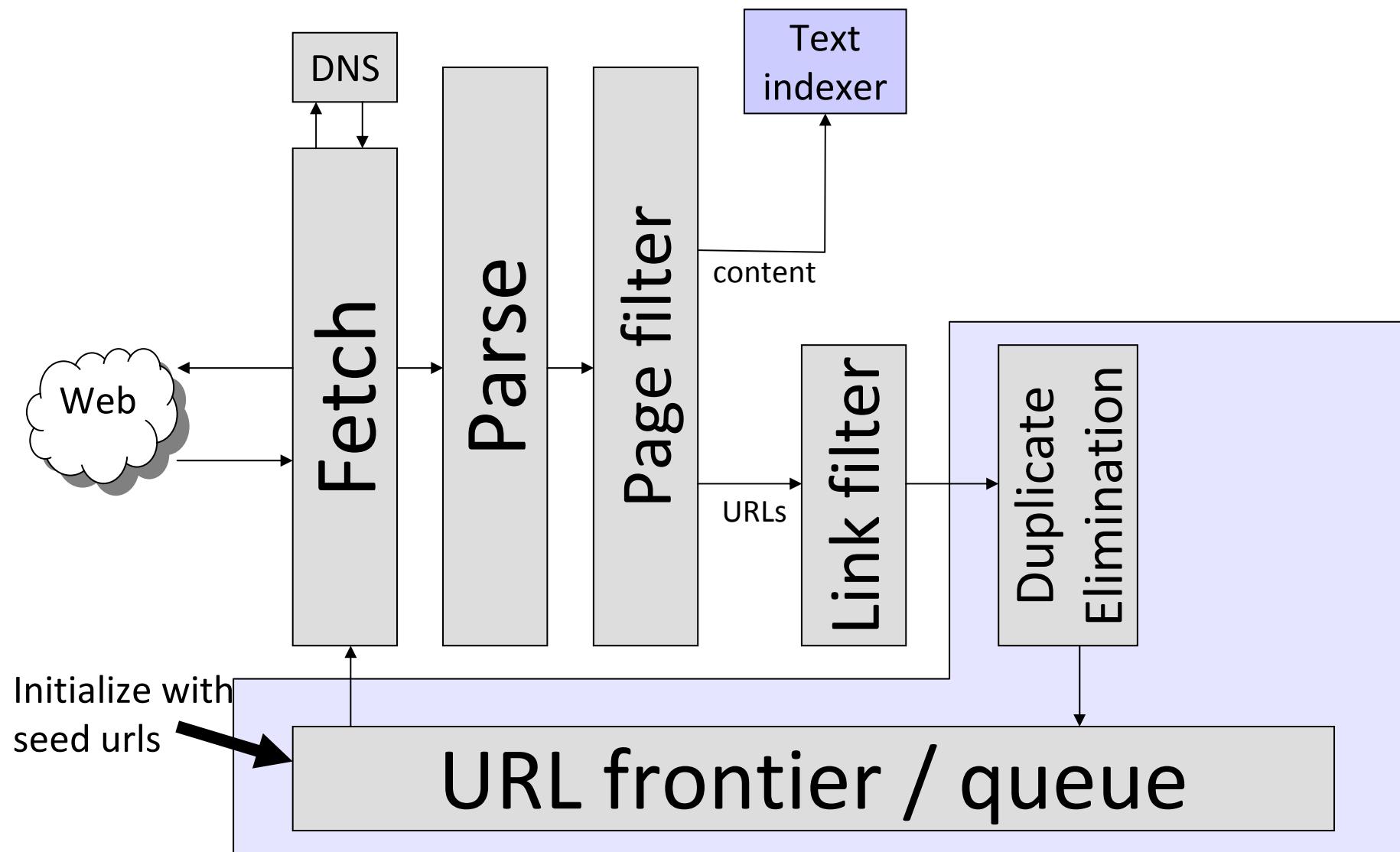
Why crawling is difficult

- Huge size of the Web (billions of pages)
- High dynamics of the Web (page creations, updates, deletions)
- High diversity in the Web (page importance, quality, formats, conformance to standards)
- Huge amount of noise, malicious content (spam), duplicate content (Wikipedia copies)

Requirements for a Crawler

- **Robustness:** resilience to (malicious or unintended) crawler traps
- **Politeness:** respect servers' policies for accessing pages (which & how frequent)
- **Quality:** focus on downloading “important” pages
- **Freshness:** make sure that crawled snapshots correspond to current version of pages
- **Scalability:** cope with growing load by adding machines & bandwidth
- **Efficiency:** make efficient use of system resources
- **Extensibility:** possible to add new features (data formats, protocols)

Basic Crawler Architecture



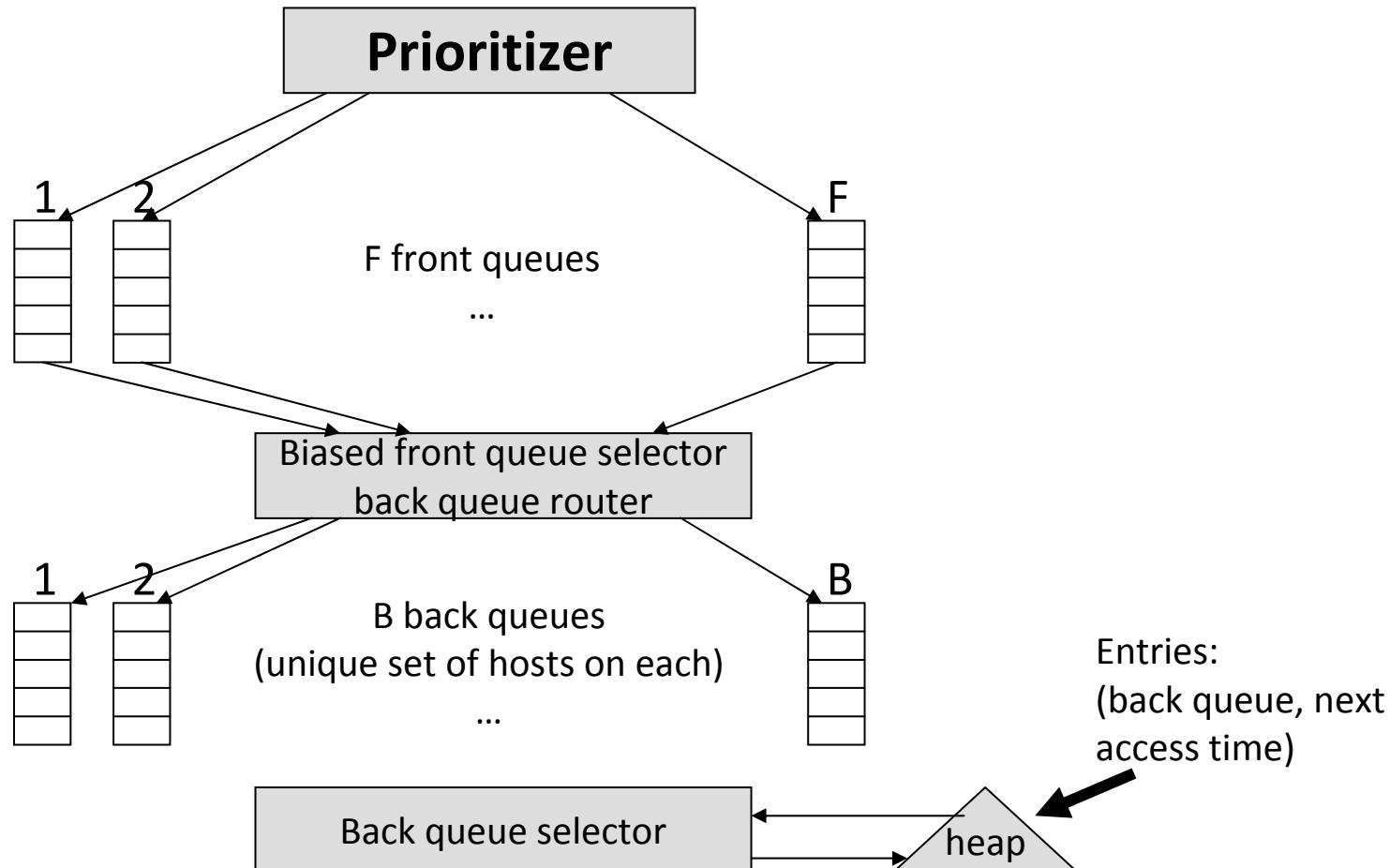
Crawler Types

- ***Snapshot crawler***: get at most one snapshot of each page (important for archiving)
- ***Batch-mode crawler***: revisit known pages periodically (collection is fixed)
- ***Steady crawler***: continuously revisit known pages (collection is fixed)
- ***Incremental crawler***: continuously revisit known pages and increase crawl quality by finding new good pages

Queue design for snapshot crawlers

Goals:

- Allow for different crawl priorities, but provide fairness
- Keep crawler busy while being polite



Modeling page changes over time

Observation:

Page changes can be modeled by Poisson process with change rate λ :

$$f_T(t) = \begin{cases} \lambda \exp(-\lambda t) & \text{for } t > 0 \\ 0 & \text{for } t \leq 0. \end{cases}$$

Probability for at least one change until t :

$$\Pr\{T \leq t\} = \int_0^t f_T(t) dt = 1 - \exp(-\lambda t)$$

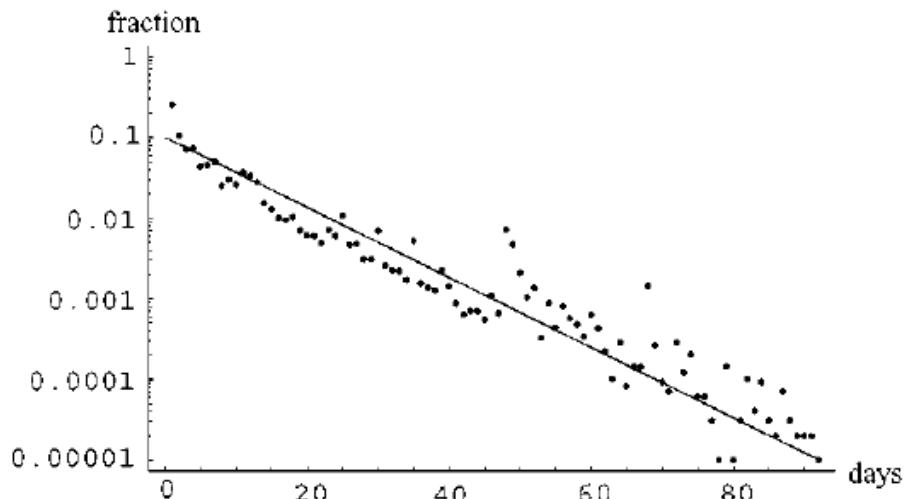
Expectation: $E[t] = 1/\lambda$, variance: $\text{var}[t] = 1/\lambda^2$

Note: change rates differ per page (and maybe over time)

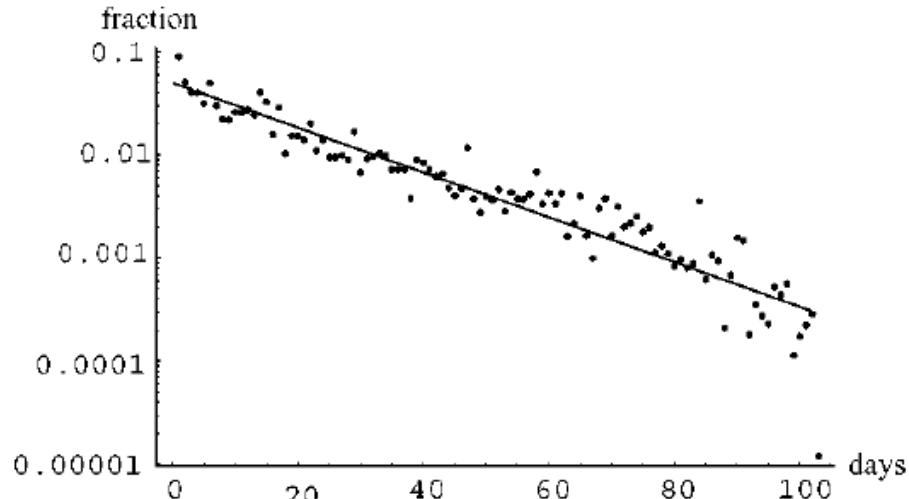
Poisson processes on real data

Cho & Garcia-Molina, TODS 2003:

- Daily crawl of 720,000 pages from 270 sites over approx 4.5 months
- Seeds: popular pages from large Web crawl



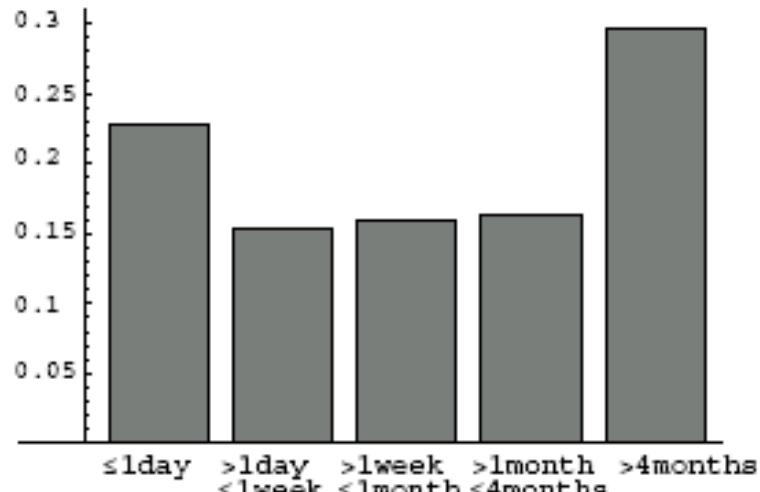
(a) For the pages that change every 10 days on average



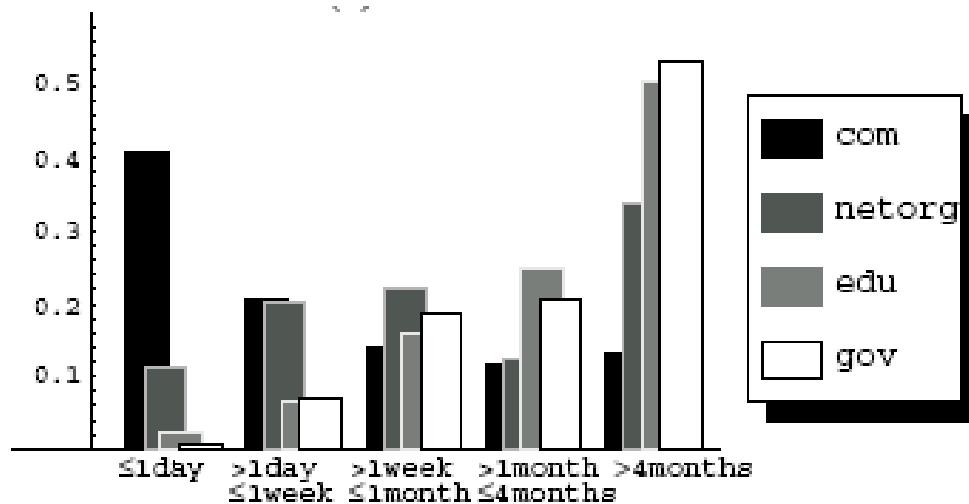
(b) For the pages that change every 20 days on average

Fig. 14. Change intervals of pages.

Change rate distributions on the Web



(a) Over all domains



(b) For each domain

Sampling change rates

Goal: determine λ_i for fixed page i

Simple estimator:

For X_i monitored updates in time T_i , estimate

$$\bar{\lambda}_i := \frac{X_i}{T_i}$$

Question: is this a good estimator?

- Is it unbiased? **No.**
- Is it consistent? **No.**

Better estimator: n_i accesses with frequency f_i , page was
not changed Y_i times

$$\bar{\lambda}_i := -f \cdot \log \frac{Y_i + 0.5}{n_i + 0.5}$$

Crawling the dynamic Web

Challenges:

- How do we model the „up-to-dateness“ of our index
- How frequently do we recrawl?
 - On average, update each of N pages once *within I time units* (average update frequency $f=1/I$)
- How frequently do we schedule per-page revisits?
 - uniformly vs. depending on the change rates
- In which order do we revisit pages
 - fixed order vs. recrawl (random) vs. purely random

Measures for recency of the index (1)

Definition:

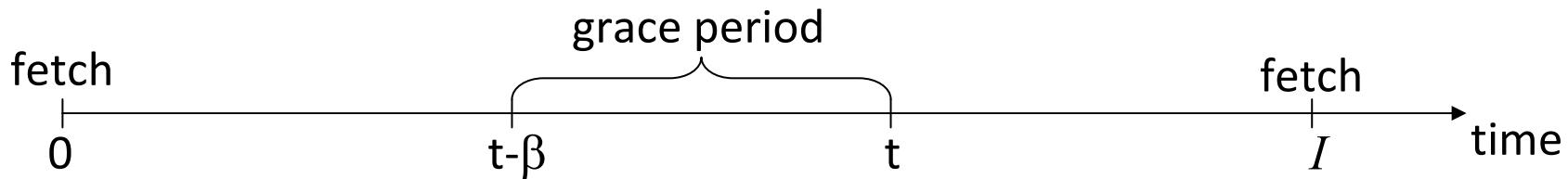
Index is **(α, β)-current at time t** when the probability that random page has been up-to-date β time units ago is at least α .

Question to answer:

How frequently do we need to recrawl to guarantee to be **(95%, 1 week)-current?**

Answer: every 18 days for 800 million page sample
[Brewington and Cybenko 2000]

Brewington&Cybenko Model



Probability that a specific document is β -current in interval $[0;I]$:

$$\underbrace{\int_0^{\beta} \frac{1}{I} dt}_{\text{if } t < \beta, \text{ prob. is 1}} + \underbrace{\int_{\beta}^I \frac{1}{I} e^{-\lambda(t-\beta)} dt}_{\text{if } t > \beta, \text{ prob. decays exponentially with delay}} = \frac{\beta}{I} + \frac{1 - e^{-\lambda(I-\beta)}}{\lambda I}$$

Now average over all documents (assuming distribution $w(\lambda)$ for change rates):

$$\alpha = \int_0^{\infty} w(\lambda) \left[\frac{\beta}{I} + \frac{1 - e^{-\lambda(I-\beta)}}{\lambda I} \right] d\lambda$$

(see paper: $1/\lambda$ is Weibull-distributed)

Measures for recency of the index (2)

- **Freshness $F(p;t)$ of a page p at time t :**
1 if p is up-to-date at time t , 0 otherwise
- **Age $A(p;t)$ of a page p at time t :**
time since the last update of p that is not reflected in the index

- **Freshness $F(t)$ of the index at time t :**

$$F(t) = \frac{1}{N} \sum_{i=1}^N F(p_i, t)$$

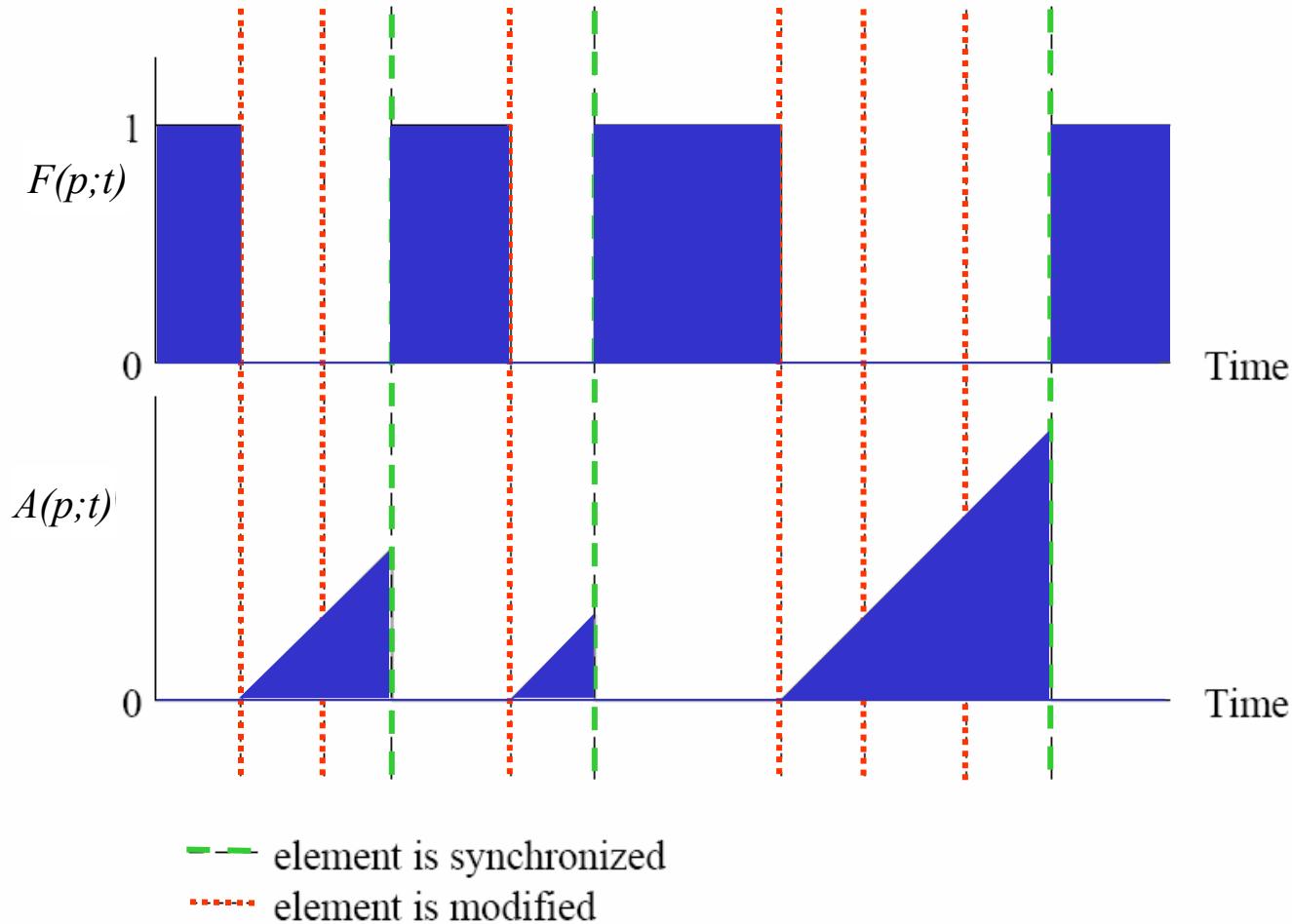
- **Average Freshness $F(p)$ of a page p :**

$$F(p) = \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t F(p; t) dt$$

- **Average Freshness of the index:**

$$F = \frac{1}{N} \sum_{i=1}^N F(p_i)$$

Example: freshness and age for page p



Freshness and age for different crawlers

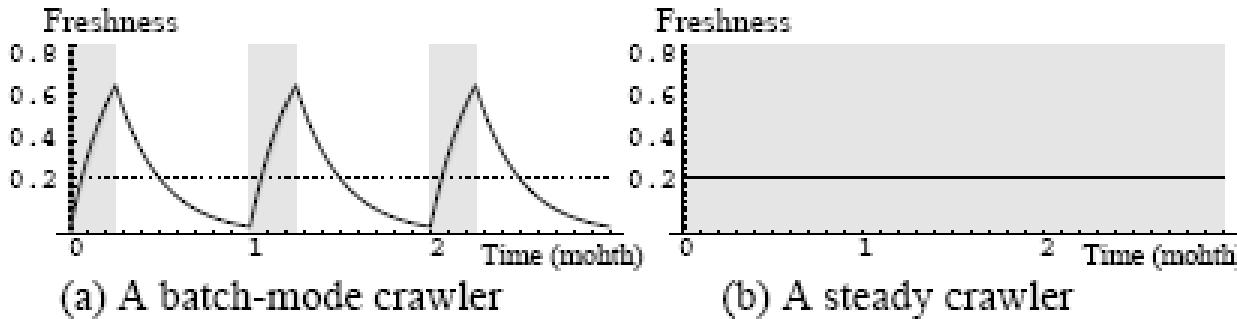


Figure 7: Freshness evolution of a batch-mode/steady crawler

grey area: time when crawler is active

solid line: $F(t)$

dotted line: average of $F(t)$

Theorem:

Average freshness is the same for both crawlers if load is the same

Expected freshness and age of a page

Assume for page p:

- p changes with rate λ
- p is synch'ed at time 0

Then

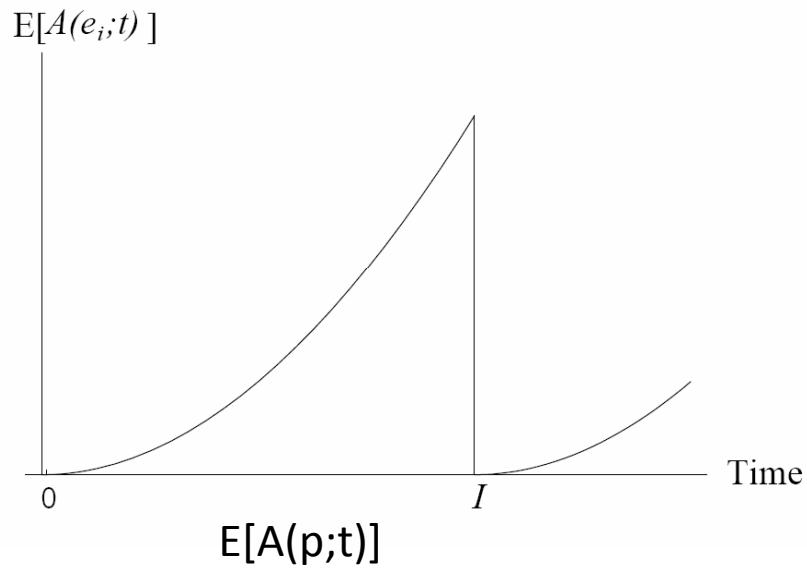
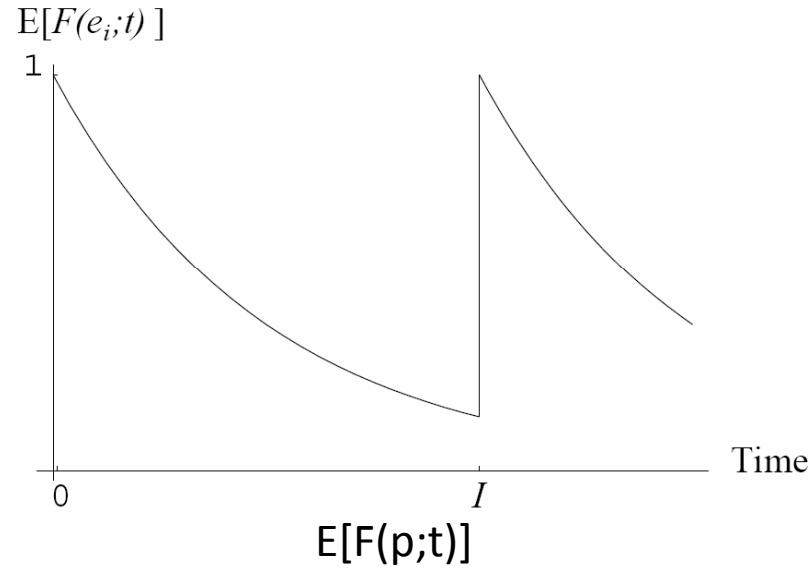
- Expected freshness of p at time $t \geq 0$:

$$E[F(p; t)] = 0 \cdot \underbrace{\left(1 - e^{-\lambda t}\right)}_{P[p \text{ changed at time } t]} + 1 \cdot e^{-\lambda t} = e^{-\lambda t}$$

- Expected age of p at time $t \geq 0$:

$$E[A(p; t)] = \int_0^t (t-s) \underbrace{\lambda e^{-\lambda s}}_{P[\text{first change of } p \text{ after time } s]} ds = t \left(1 - \frac{1 - e^{-\lambda t}}{\lambda t}\right)$$

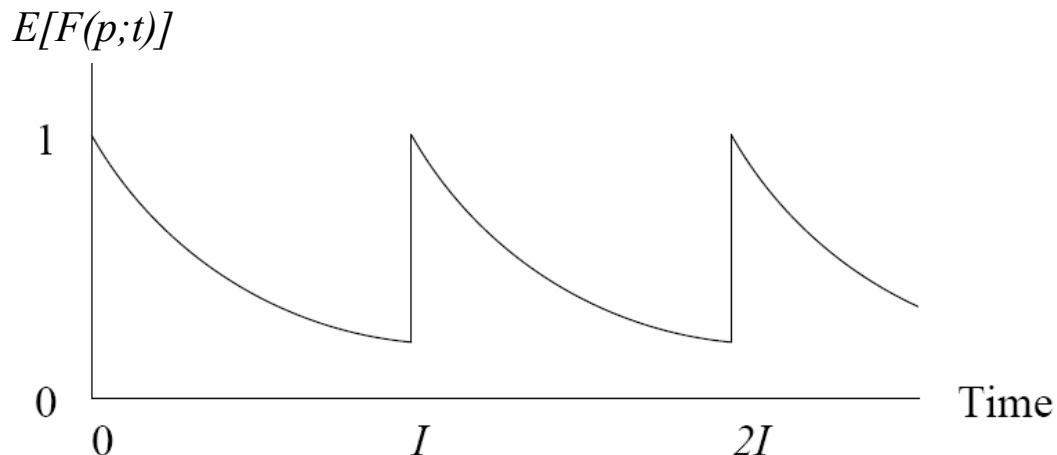
Expected freshness and age over time



Which avg. freshness can we achieve?

Assume that

- All pages change at *the same rate* λ
- All pages are sync'ed *every* I time units (at rate $f=1/I$)
- Pages are always sync'ed *in a fixed order*

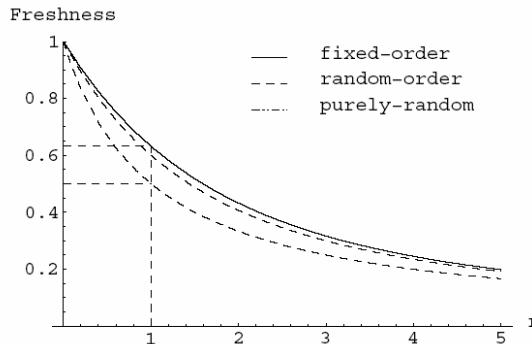


Theorem:

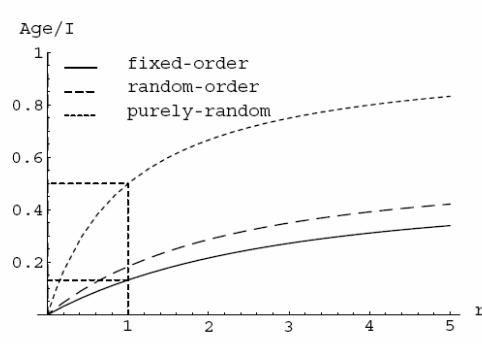
$$F(p) = \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t E[F(p; t)] dt = \frac{1}{I} \int_0^I E[F(p; t)] dt = \frac{1 - e^{-\lambda I}}{\lambda I} = \frac{1 - e^{-\lambda/f}}{\lambda / f} = F$$

Are other orders better?

- **Random order:**
update all pages once, but in random order (e.g., by recrawling)
- **Purely random order:**
pick page to update at random



(a) Freshness graph over $r = \lambda/f$



(b) Age graph over $r = \lambda/f$

<i>policy</i>	<i>Freshness</i>	<i>Age</i>
Fixed-order	$\frac{1-e^{-r}}{r}$	$\frac{1}{f}(\frac{1}{2} - \frac{1}{r} + \frac{1-e^{-r}}{r^2})$
Random-order	$\frac{1}{r}(1 - (\frac{1-e^{-r}}{r})^2)$	$\frac{1}{f}(\frac{1}{3} + (\frac{1}{2} - \frac{1}{r})^2 - (\frac{1-e^{-r}}{r^2})^2)$
Purely-random	$\frac{1}{1+r}$	$\frac{1}{f}(\frac{r}{1+r})$

Non-uniform update frequencies

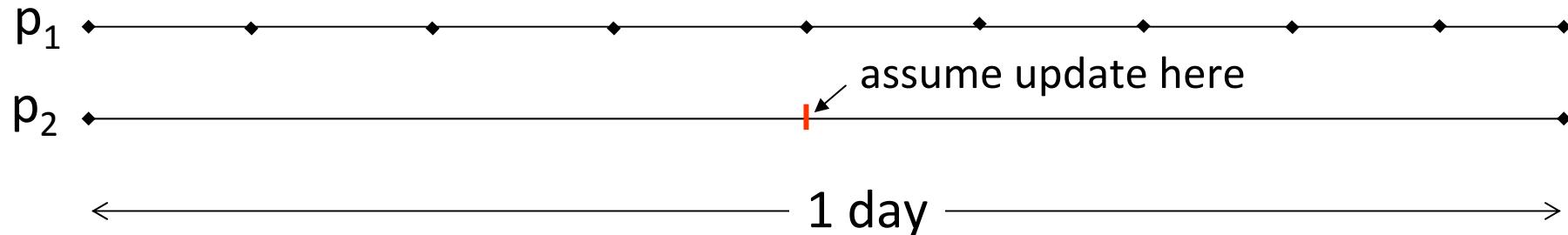
Now

- page p_i changes with rate λ_i
- page p_i is updated at fixed interval $I_i (=1/f_i)$

Question: How are f_i and λ_i related?

Simple answer $f_i \propto \lambda_i$ is wrong!

Simple example: two pages, one update



Assume

- p_1 changes once per interval (=9 times/day)
- p_2 changes once per day
- probability for change uniform in each interval

Now estimate *expected benefit of updating p_2* in the middle of the day

- with prob. $\frac{1}{2}$ change occurs later \Rightarrow benefit 0
- with prob. $\frac{1}{2}$ change occurs before \Rightarrow benefit $\frac{1}{2}$
- Expected benefit: $\frac{1}{2} * \frac{1}{2} = \frac{1}{4}$

Similar computation for p_1 (update in the middle of any interval):

- Expected benefit: $\frac{1}{2} * \frac{1}{18} = \frac{1}{36}$

Two pages, more updates

row	$f_1 + f_2$	f_1	f_2	benefit	best
(a)	1	1	0	$\frac{1}{2} \times \frac{1}{18} = \frac{1}{36}$	0 1
(b)		0	1	$\frac{1}{2} \times \frac{1}{2} = \frac{9}{36}$	
(c)	2	2	0	$\frac{1}{2} \times \frac{1}{18} + \frac{1}{2} \times \frac{1}{18} = \frac{2}{36}$	0 2
(d)		1	1	$\frac{1}{2} \times \frac{1}{18} + \frac{1}{2} \times \frac{1}{2} = \frac{10}{36}$	
(e)		0	2	$\frac{1}{3} \times \frac{2}{3} + \frac{1}{3} \times \frac{1}{3} = \frac{12}{36}$	
(f)	5	3	2	$\frac{3}{36} + \frac{12}{36} = \frac{30}{72}$	2 3
(g)		2	3	$\frac{2}{36} + \frac{6}{16} = \frac{31}{72}$	
(h)	10	9	1	$\frac{9}{36} + \frac{1}{4} = \frac{36}{72}$	7 3
(i)		7	3	$\frac{7}{36} + \frac{6}{16} = \frac{41}{72}$	
(j)		5	5	$\frac{5}{36} + \frac{15}{36} = \frac{40}{72}$	

Rules of thumb:

- When sync frequency (f_1+f_2) much smaller than change frequency ($\lambda_1+\lambda_2$), don't sync quickly changing pages
- Even for $f_1+f_2 \approx \lambda_1+\lambda_2$, uniform (5:5) better than proportional (9:1)

Can we prove this?

Proof (1)

Notation/Definition:

- $F(\lambda_i, f_i)$: average freshness of p_i when p_i changes with rate λ_i and is updated with rate f_i
 - $\lambda = \frac{1}{N} \sum \lambda_i$ average change rate
 - function $f(x)$ is convex if
$$\frac{1}{n} \sum_{i=1}^n f(x_i) \geq f\left(\frac{1}{n} \sum_{i=1}^n x_i\right)$$
- $F(\lambda_i, f_i)$ is convex in λ_i independent of the sync strategy

Proof(2)

With *uniform* update frequency ($f_i=f$):

$$F_u = \frac{1}{N} \sum F(\lambda_i, f_i) = \frac{1}{N} \sum F(\lambda_i, f)$$

With *proportional* update frequency:

$$F_p = \frac{1}{N} \sum F(p_i) = \frac{1}{N} \sum F(\lambda, f) = F(\lambda, f)$$

here, $F(p_i)=F(\lambda_i, f_i)=F(\lambda, f)$ because $F(p_i)$ depends only on $r=\lambda/f$

Then:

$$F_u = \frac{1}{N} \sum F(\lambda_i, f) \geq F\left(\frac{1}{N} \sum \lambda_i, f\right) = F(\lambda, f) = F_p$$

Optimization Problem

Given λ_i ($i=1..N$), find f_i ($i=1..N$) that maximize

$$F = \frac{1}{N} \sum_{i=1}^N F(\lambda_i, f_i)$$

under the constraint that

$$\sum_{i=1}^N f_i = f \text{ and } f_i \geq 0 \text{ } (i = 1..N)$$

Using Lagrange multipliers, this transforms to

$$\frac{\partial F(\lambda_i, f_i)}{\partial f_i} F = \mu \quad \sum_{i=1}^N f_i = f$$

which can be solved numerically (for fixed order)

Solving for the two-page example

($\lambda_1=9$, $\lambda_2=1$, $f=2$)

```
fsolve({diff((1-exp(-9/f1))/(9/f1),f1)=y, diff((1-exp(-1/f2))/(1/f2),f2)=y,\n f1+f2=2},{f1,f2,y},{f1=0...2,f2=0...2,y=0...100});
```

{ $f_1 = 0.2358671910$, $f_2 = 1.764132809$, $y = 0.1111111111$ }

($\lambda_1=9$, $\lambda_2=1$, $f=10$)

```
fsolve({diff((1-exp(-9/f1))/(9/f1),f1)=y, diff((1-exp(-1/f2))/(1/f2),f2)=y,\n f1+f2=10},{f1,f2,y},{f1=0...10,f2=0...10,y=0...100});
```

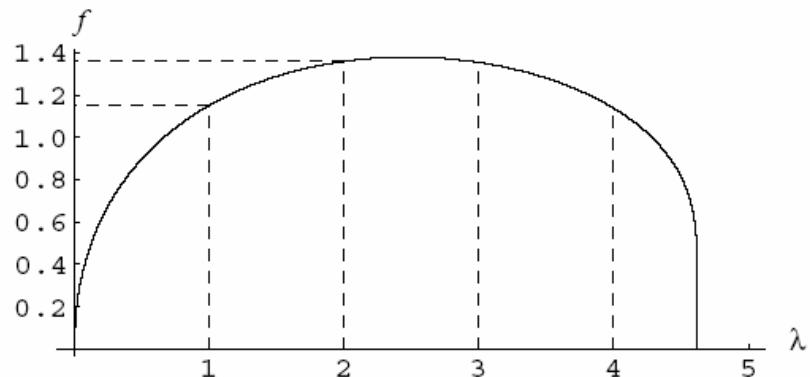
{ $f_1 = 6.885783095$, $f_2 = 3.114216905$, $y = 0.04174104014$ }

May require grouping of pages with similar frequency to be scalable

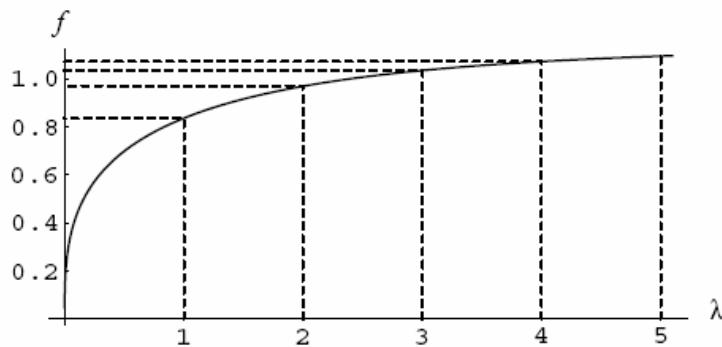
Sync freq. as function of change freq.

	e_1	e_2	e_3	e_4	e_5
(a) change frequency	1	2	3	4	5
(b) synchronization frequency (freshness)	1.15	1.36	1.35	1.14	0.00
(c) synchronization frequency (age)	0.84	0.97	1.03	1.07	1.09

Table IV. The optimal synchronization frequencies of Example 5.4



(a) synchronization frequency as a function of change frequency for freshness optimization



(b) synchronization frequency as a function of change frequency for age optimization

Fig. 11. Solution of the freshness and age optimization problem of Example 5.4

Predicted Freshness/Age

	overall		com		gov	
	Freshness	Age	Freshness	Age	Freshness	Age
Proportional	0.12	400 days	0.07	386 days	0.69	19.3 days
Uniform	0.57	5.6 days	0.38	8.5 days	0.82	2.0 days
Optimal	0.62	4.3 days	0.44	7.4 days	0.85	1.3 days

Table VII. Freshness and age prediction based on the real Web data

(assuming 1 billion pages, sync interval=1 month,
reasonable distribution of change rates)

Extension: Page Weights

Goal:

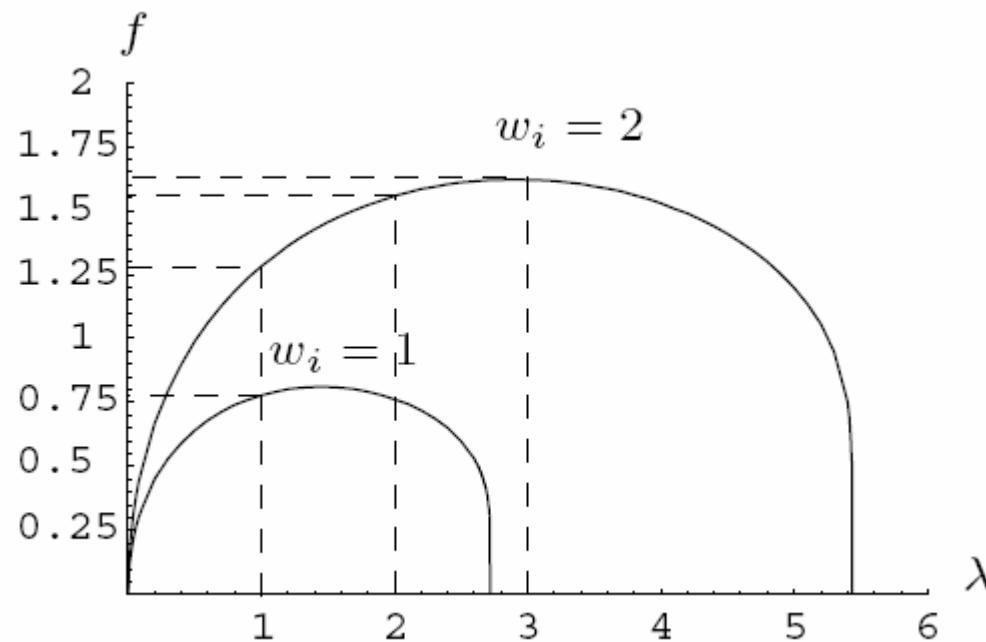
Provide high freshness/low age for „important“ pages
(e.g., measured by click rates, pagerank, ...)

Solution:

Consider weighted freshness/age: $F = \frac{\sum_{i=1}^N w_i F(p_i)}{\sum_{i=1}^N w_i}$
(leads to similar opt. problem)

Example with Page Weights

	e_{11}	e_{12}	e_{13}	e_{21}	e_{22}	e_{23}
(a) weight (w_i)		1			2	
(b) change frequency (times/day)	1	2	3	1	2	3
(c) synchronization frequency (freshness)	0.78	0.76	0.00	1.28	1.56	1.62
(d) synchronization frequency (age)	0.76	0.88	0.94	0.99	1.17	1.26



Crawling to Improve Result Quality

So far: every page change considered equal
⇒ often too conservative (advertisements,
date/time, dynamic links, ...)

Goal:

Update page only for *important* changes

Metric 1: Result Quality

Query results **on the index** should be identical to
Query results **on the live Web**
(see Pandey/Olston, WWW05, for details)

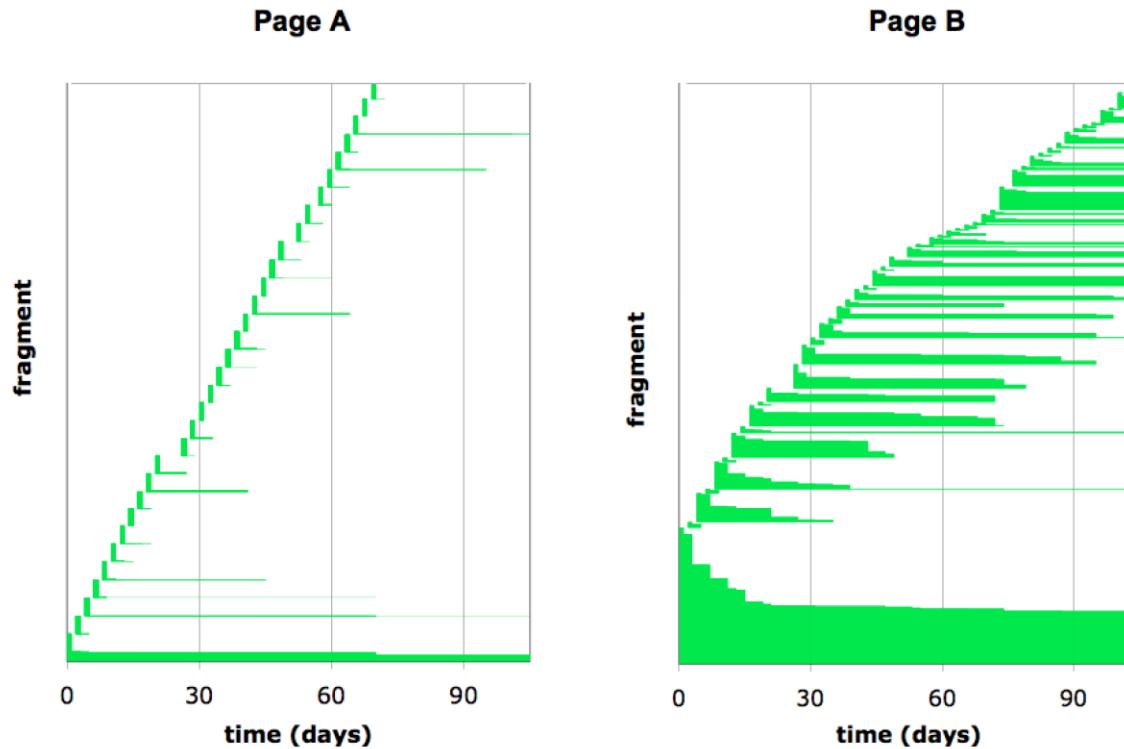
Metric 2: Information Longevity

Assumption:

Content that „lasts for a while“ more important than

Content that is „transient“ (such as ads)

Example: lifespan of (word-level) shingles on two pages



Quantifying Freshness / Staleness

Notation:

- $S(p)$ = set of shingles of page p
- p_t version at time t

Freshness of p at time t (relative to index time t_p):

$$F(p; t_p; t) = \frac{|S(p_{t_p}) \cap S(p_t)|}{|S(p_{t_p}) \cup S(p_t)|} \quad (\text{Jaccard coefficient})$$

Staleness of p at time t (relative to index time t_p):

$$D(p; t_p; t) = 1 - \frac{|S(p_{t_p}) \cap S(p_t)|}{|S(p_{t_p}) \cup S(p_t)|}$$

Optimizing Staleness

Assume staleness never decreases over time

⇒ estimate $D(p; t_p; t)$ by monotone function $D_p^*(t - t_p)$

Theorem:

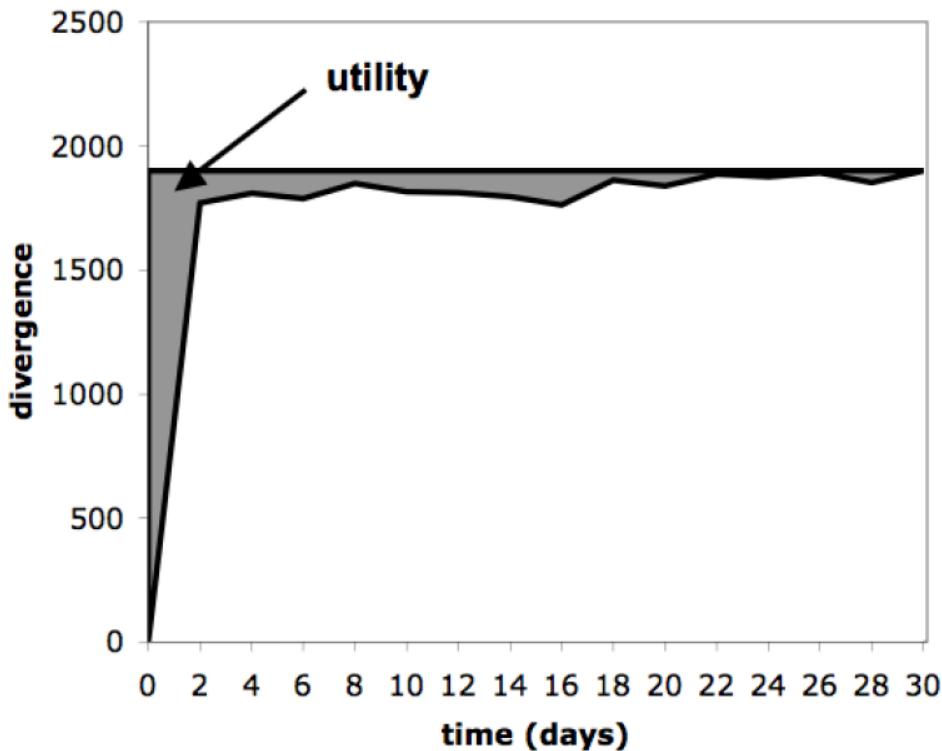
At each point in time t , refresh exactly those pages
that have $U_p(t - t_p) \geq T$, where

$$U_p(t) = t \cdot D_p^*(t) - \int_0^t D_p^*(x) dx$$

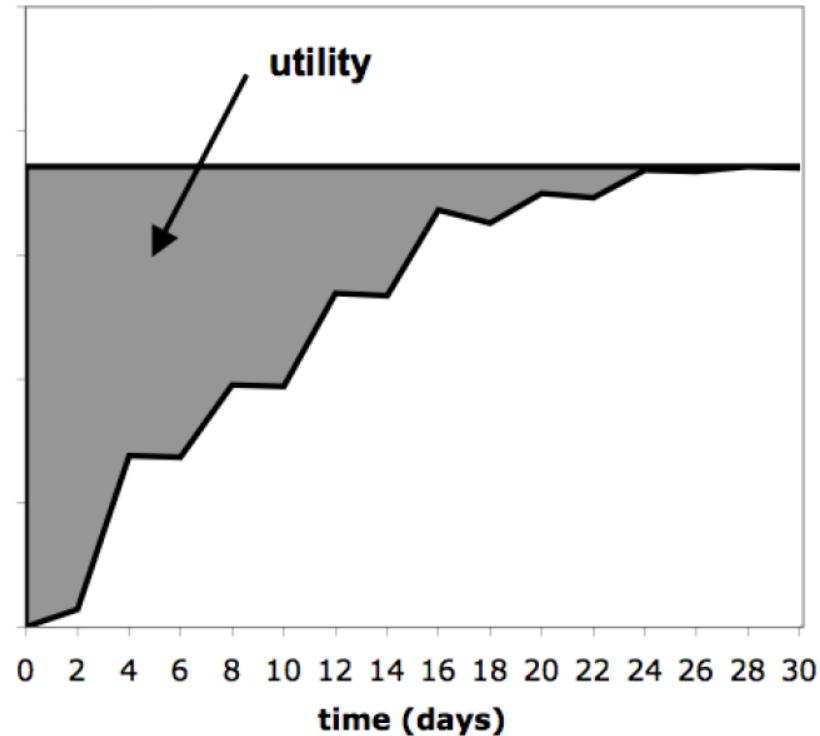
This yields optimal staleness among all schedules
that do the same number of refreshes.

Intuition for Utility Function

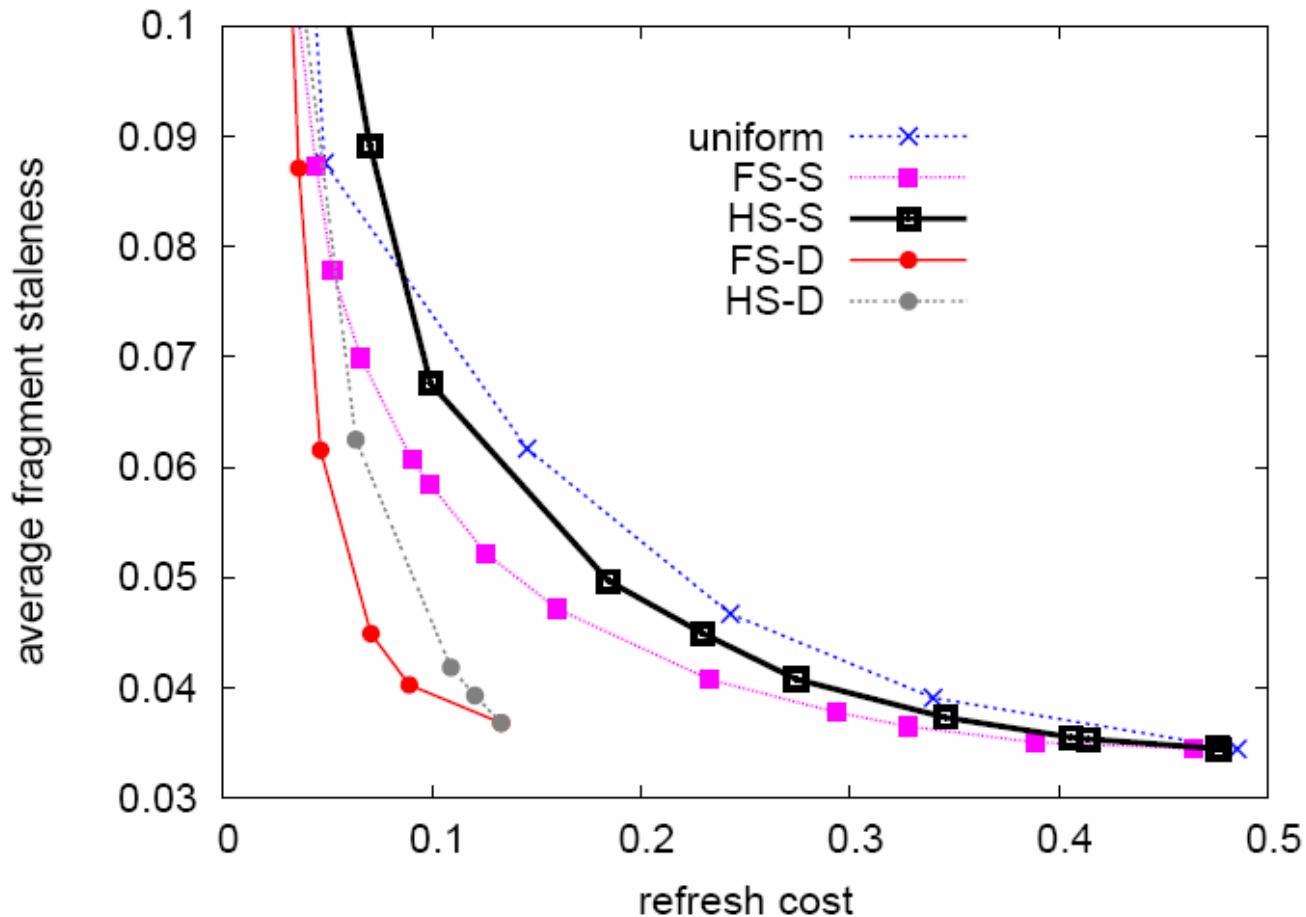
Page A



Page B



Experiment: Staleness



HS: freshness-based („holistic“)

FS: staleness-based („fragment“)

-S: estimation of D^* over all snapshots

-D: estimation of D^* for each snapshot

Web Dynamics

Part 3 – Searching the Dynamic Web

3.1 Crawling and recrawling policies

3.2 Accessing the Hidden Web

Introducing the Hidden/Deeb/Invisible Web



The screenshot shows the cars.com website's search interface. At the top is the cars.com logo. Below it, a large search bar contains the placeholder text "Search For a Car". Underneath the search bar, there's a section titled "New Cars" with the sub-instruction "Build or find your car". This section includes dropdown menus for "Make" (set to "Acura") and "Model" (set to "All"), and a "Maximum Price" dropdown set to "No Max". Below these are fields for "Search Within" (set to "30 miles") and "Your ZIP" (with a blank input field). A red "Search New" button is at the bottom of this sidebar.

Search
Home > Advanced Search

Books Used & Out of Print

Advanced Book Search

Fill in one or more of the fields below to search. You can narrow your search by selecting from the dropdown menus.

Title

Author

Keywords

Publisher

Price All Prices

Format All Formats

Age All Ages

Subject All Subjects

Search for exact phrase

[Advanced Search](#) | [1982 and Older](#)

BioMyn

Integrated knowledge for systems biology

Beta version. Only human data available.

[BioMyn Toys](#)

examples: ['cancer apoptosis'](#), ['methylation'](#), ['brca2'](#),
['ENSP00000369497'](#), ['kegg apoptosis'](#), ['diabetes'](#), ['kinase'](#)

[MPII Home](#) - [About BioMyn](#) - [Download](#) - [Contact](#) - [Blog](#)

Data only accessible through Web forms
estimate: ~100 petabytes from approx. 100 million sources,
compared to ~200 terabytes for the „surface Web“)

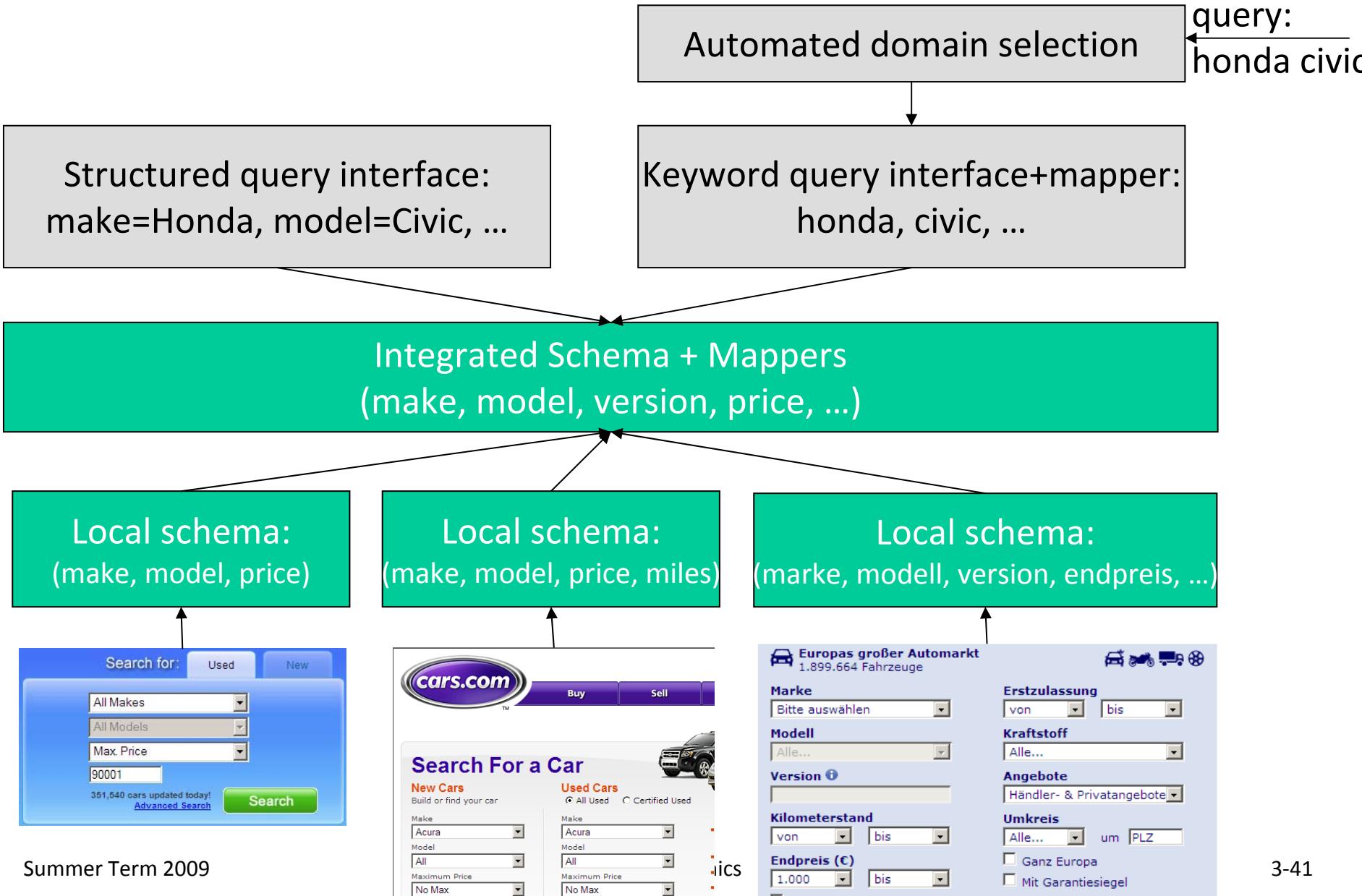
Diversity of „Hidden Web“-Sources

„Hidden Web“-sources differ in

- amount of provided data
- quality/authority of provided data
- freshness of provided data
- covered application domain(s)
- richness of interface (text box, selection,...)

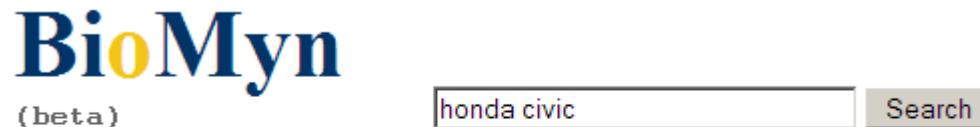
How can such dynamic sources be included in a search engine's index?

Approach 1: Meta Search Engines



Pros and Cons

Good: Information always up-to-date



But:

- Includes major manual work (mapper definition), does not scale well
- Requires complex domain detection („java“, „jaguar“)
- Cannot easily cope with interface changes
- Automated processes are error-prone

Approach 2: Surfacing

- Generate „reasonable“ inputs for each form
- Index result pages (identified by their url)
like static Web pages
- Follow outgoing links

<http://www.autoscout24.de>List.aspx?vis=1&make=9&model=18683&...>

1.330 Fahrzeuge für Sie gefunden

Ausgeföhrte Suche: Audi Q7, Preis ab € 1.000,-, Deutschland

Fahrzeuge 1 - 20

Länderspezifisch und europaweit suchen
Bitte auswählen

Aktuelle Topangebote zu Ihrer Suche:

Audi Q7	Audi Q7	Audi Q7
€ 37.500,- MwSt. ausweisbar 102.000 km EZ 05/2007	€ 36.700,- MwSt. ausweisbar 59.659 km EZ 05/2006	€ 42.800,- MwSt. ausweisbar 26.591 km EZ 08/2007

Listenansicht Bildergalerie Tabelle

Sortierung aufsteigend nach: Preis Ergebnisse: 20 pro Seite

3 vergleichen

Audi Q7 3.0 TDI DPF quattro tiptronic 1.Hand

D-71065 Sindelfingen - silber metallic - Automatik, Diesel, ABS, Airbag, Alarmanlage, Allrad, Alufelgen, Anhängerkupplung, Beifahrer Airbag, Bordcomputer, Dachträger, Elektr. Einparkhilfe... > mehr

7 Bilder Fahrzeug merken

Selecting a few good inputs is important

The screenshot shows a web form for searching new cars. It includes fields for 'Make' (Audi), 'Model' (Q7), 'Maximum Price' (\$10,000), and a search radius of '30 miles' from '66123'. There are also 'Search Within' and 'Your ZIP' fields. A large orange 'Search New' button is at the bottom.

11 values

New Cars
Build or find your car

Make: Audi

Model: Q7

Maximum Price: \$10,000

Search Within: 30 miles of: 66123 Your ZIP

Search New

~ 2000 combinations

100 values

~ 100,000 values

⇒ 220 billion combinations possible!
(but only 650,000 cars in the database)

(remember from part 1: the Web is infinite!)

Finding „good“ inputs: templates

Simplifying model:

selects, text fields

Web form with inputs $X_1 \dots X_n$ provides access to database D

Problem:

Find *query templates* $T \subseteq \{X_1 \dots X_n\}$ and input value sets V_i for each input $X_i \in T$ such that instantiating T by submitting input value combinations to the form (leaving $X_k \notin T$ blank)

1. yields good coverage of D  difficult to measure
2. is efficient, i.e., does not return too many duplicates



How can we measure this?

Template Informativeness

Fix ***signature function S(p)*** for Web pages
(e.g., word-level shingles)

Approach:

Measure ***informativeness*** of results for template T with input set $G(T)$ (all possible inputs) as

$$I(T) = \frac{|\{S(p) \mid p \text{ generated by } T \text{ with input } g \in G(T)\}|}{|G(T)|}$$

Definition:

Template T ***informative*** iff $I(T) \geq \tau$

In practice: Ignore T where $|G(T)| > 10,000$; consider only a sample of $G(T)$ (up to 200 inputs per template)

Finding Informative Templates: ISIT

```
informative:=∅; // inform. templates
candidates:={ {xi} };
while (candidates!=∅)
    for each x∈candidates:
        if (x not informative) remove x;
    informative=union(candidates);
    c:=∅; // set of cands for next step
    for each x∈candidates, I input:
        c=union(x,I);
    candidates=c;
```

Informative Templates: Experiments

ISIT efficiency: 500,000 (randomly chosen) HTML forms

Table 1: Distribution of HTML Forms by the number of inputs.

Num. Inputs	Num. Forms	Num. Inputs	Num. Forms
1	429271	6	1684
2	43776	7	1071
3	12987	8	418
4	7378	9	185
5	3101	10	129

Num. Inputs	Max Templates [dimension ≤ 3]	Average Templates Tested	Average Templates Informative	Average URLs Analyzed
1	1 [1]	1	1	23
2	3 [3]	2.82	1.76	136
3	7 [7]	5.38	3.04	446
4	15 [14]	8.80	3.75	891
5	31 [25]	12.66	5.93	1323
6	63 [41]	17.43	7.17	1925
7	127 [63]	24.81	10.62	3256
8	255 [92]	29.46	13.45	3919
9	511 [129]	35.98	15.80	4239
10	1023 [175]	41.46	17.71	5083

ISIT effectiveness: 12 selected forms

No.	URL with HTML form	Cartesian product	URLs Generated	Estimated Database	Records Retrieved
1	http://www2.imm.dtu.dk/pubdb/public/index_public.php	53550	1349	4518	4518
2	http://dermatology.jwatch.org/cgi/archive	32400	179	3122	1740
3	http://www.kles.com.au/listings.php	129654	81	62	61
4	http://aarpmagazine.org/food/recipeguide	150822	365	314	273
5	http://www.shadetrees.org/search.php	313170	181	1699	1756
6	http://www.dasregistry.org/listServices.jsp	2 million	627	295	267
7	http://www.nch.org.uk/ourservices/index.php?i=402	2.5 million	34	27	27
8	http://www.lodgealestate.co.nz/?nav=rentsearch	4.7 million	26	87	74
9	http://www.pipa.gov.ps/search_db.asp	68 million	1274	409	409
10	http://www.biztransact.com/	257 million	13,209	100,000+	10,937
11	http://projects.bebif.be/enbi/albertinerift/common/search	948 billion	2406	2287	1257
12	http://oraweb.aucc.ca/showcupid.html	3.2 trillion	34	27	27

Handling Text Inputs

Problem: No (small) set of input values available

Solution: Iterative sampling

1. Pick ***most important terms*** from the page with the text input (using tf*idf), check informativeness
2. To create next set of input values, pick most important terms ***from all pages generated*** so far (excluding those appearing on all pages)
3. Stop when set is stable

Experimental Results: Text Inputs

Table 5: Examples of HTML forms with text boxes comparing the the actual size of the database (number of records) against the number of URLs generated and the number of records retrieved: (first) on the first results page when using only the text box, (select) on the first page using only select menus, and (first++) on the first page and the pages that have links from it when using only the text box,

No.	Form URL	Estimated Database	Keywords Selected	Results Per Page	Records Retrieved		
					first	select	first++
1	http://kbase.gofrugaltech.com/	382	169	20	348	n.a.	381
2	http://nsidc.org/data/index.html	506	355	20	475	102	478
3	http://www.hhmi.org/news/search.html	1700	500	25	1485	320	1585
4	http://ww.azom.com/	4700	500	10	2136	n.a.	3124
5	https://secure.financeweek.co.uk/cgi-bin/iadmin.cgi?page=68	5000	500	50	4118	1316	4118
6	http://www.angis.org.au/Databases/BIRX/	18200	500	100	5315	n.a.	n.a.
7	http://english.ibd.com.cn/databank.asp	21200	500	10	2661	306	4697
8	http://federalgovernmentjobs.us/	60300	500	10	3096	n.a.	12127
9	http://praca.gratka.pl/	15800	500	20	4012	290	11043
10	http://www.rhdsc.gc.ca/fr/ministeriel/recherche/index.shtml	21800	500	200	13324	724	19332

Current challenges for Crawling/Indexing

- ***Non-HTML content*** (Flash, SilverLight)
⇒ single big „object“, encapsulates content and outgoing links
- ***Dynamic pages*** where program in the browser loads content directly from the server (AJAX)
⇒ only 1 URL for application, inaccessible for standard crawlers (no Javascript!)
- ***Highly dynamic data*** (social networks, Twitter)
- ***Non-textual data*** (images, videos, ...)

References

Main references:

- P. Baldi, P. Frasconi, P. Smyth: **Modeling the Internet and the Web**, chapter 6
- G. Pant et al.: **Crawling the Web**, Web Dynamics book, 153—178, 2004
- J. Bar-Ilan: **Search engine ability to cope with the changing Web**, Web Dynamics book, 195—218, 2004
- J. Cho, H. Garcia-Molina: **Effective page refresh policies for Web crawlers**, ACM Trans. Database Syst. 28(4), 390-426, 2003

Additional references:

- C.D. Manning et al.: **Introduction to Information Retrieval**, Chapter 20, Cambridge University Press, 2008
- J. Cho, H. Garcia-Molina: **Estimating frequency of change**, ACM Transactions on Internet Technology 3(3), 256—290, 2003
- I. Hellsten et al.: **Multiple presents: how search engines rewrite the past**, New Media & Society 8(6), 901—924, 2006
- D. Lewandowski: **A three-year study on the freshness of web search engine databases**, Information Science 34(6), 917—831, 2008
- R. Baeza-Yates et al.: **Crawling a country: better strategies than breadth-first for Web page ordering**, WWW Conference, 2005
- R. Baeza-Yates et al.: **Web structure, dynamics and page quality**, SPIRE conference, 2002
- J. Cho, H. Garcia-Molina: **Parallel crawlers**, WWW Conference, 2002
- B.E. Brewington, G.Cybenko: **How dynamic is the Web?** Computer Networks 33(1-6), 257-276, 2000
- S. Pandey, C. Olston: **User-Centric Web Crawling**, WWW Conference, 2005.
- C. Olston, S. Pandey: **Recrawl Scheduling Based on Information Longevity**, WWW Conference, 2008
- J. Madhavan et al.: **Google's Deep-Web Crawl**, VLDB Conference, 2008
- P.G. Ipeirotis et al.: **To Search or To Crawl? Towards a Query Optimizer for Text-Centric Crawls**, SIGMOD Conference, 2006.