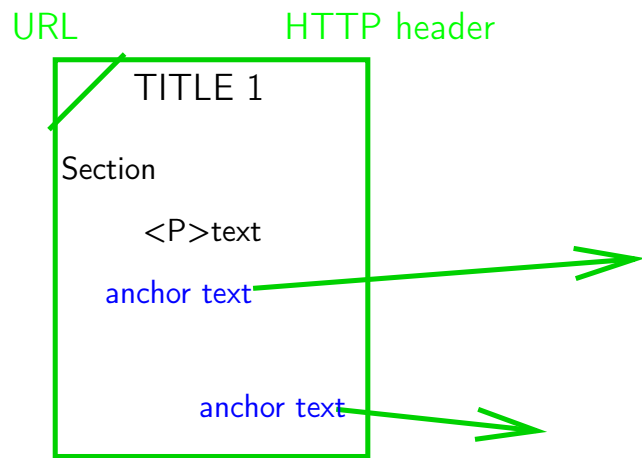


Web Document Models for Web Information Retrieval

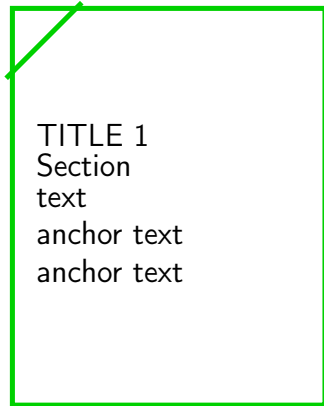
Michel Beigbeder

Web page model



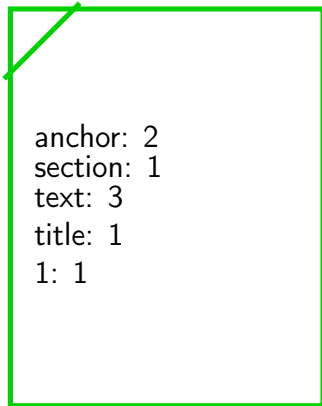
HTML:
(Internal) Structure
Links (External structure)

Flat Web page model



No structure
Links for gathering only

Vector Web page model (1)

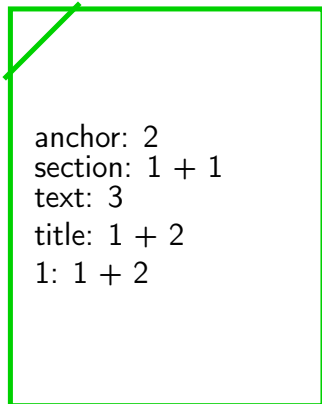


No structure

No links

No text: bag of words

Vector Web page model (2)



anchor: 2
section: 1 + 1
text: 3
title: 1 + 2
1: 1 + 2

No structure

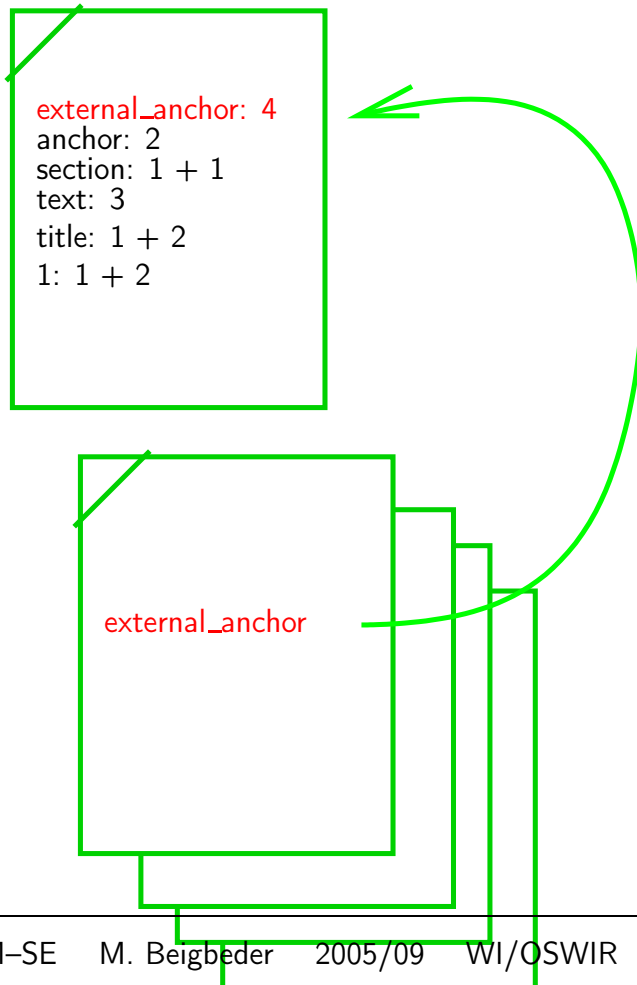
No links

No text: bag of words

Improvements:

Tag dependant weight (vector) or
occurrence (boolean)

Vector Web page model (3)



No structure

No links

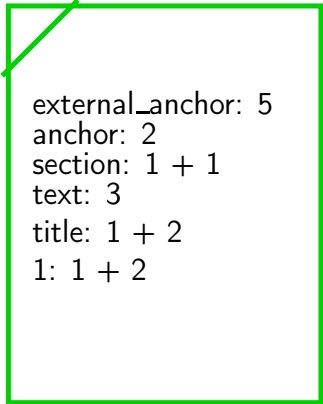
No text: bag of words

Improvements:

Tag dependant weight (vector) or
occurrence (boolean)

Indexation with anchor text of inlinks

Vector Web page model (4)



```
external_anchor: 5
anchor: 2
section: 1 + 1
text: 3
title: 1 + 2
1: 1 + 2
```

PageRank: 0.8

Hubbiness: 0.2

Authority: 0.7

No structure

No links

No text: bag of words

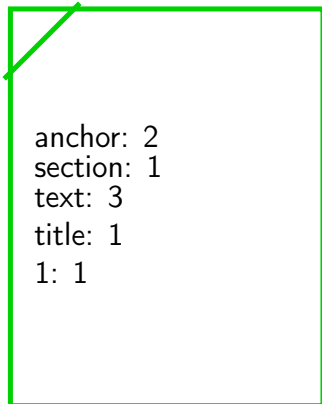
Improvements:

Tag dependant weight (vector) or
occurrence (boolean)

Indexation with anchor text of inlinks

PageRank (Google) & HITS

Vector Web page model (5)



anchor: 2
section: 1
text: 3
title: 1
1: 1

No structure

No links

No text: bag of words

Improvements:

Tag dependant weight (vector) or
occurrence (boolean)

Indexation with anchor text of inlinks

PageRank (Google) & HITS

Occurrence proximity (Google)

What features to use?

URL
HTTP header
n-grams
stems
words
text
internal structure
external inlinks anchor text

Web graph usage

Index enhancement and relevance propagation

Page ranking

Topic page gathering

Page categorization

Page classification

Similar page discovery

Replica discovery

Logical Units Discovery

Communities discovery

Web graph usage

Index enhancement and relevance propagation
Page ranking

Topic page gathering
Page categorization
Page classification
Similar page discovery
Replica discovery
Logical Units Discovery
Communities discovery

→ clustering

Integrate clusterization in search engines

Relevance, popularity, etc. produce ranked lists thus the user as an $O(n)$ access

⇒

Organize results in trees (clusters) to have an $O(\log n)$ access.

How:

- clustering
 - *before* or *after* the query
 - in a site, in the Web
 - based on different similarities according to the user needs
- categorizing
 - Using (open) categorizations (www.dmoz.org)