Detecting the Origin of Text Segments Efficiently

by Abdel-Hamid O., Behzadi B, Christoph S., Henzinger M.

Presentation on Hot Topic in Information Retrieval

by, Besnik Fetahu.

Introduction

- Problem of replicated content (i.e.,stealing of intellectual property).

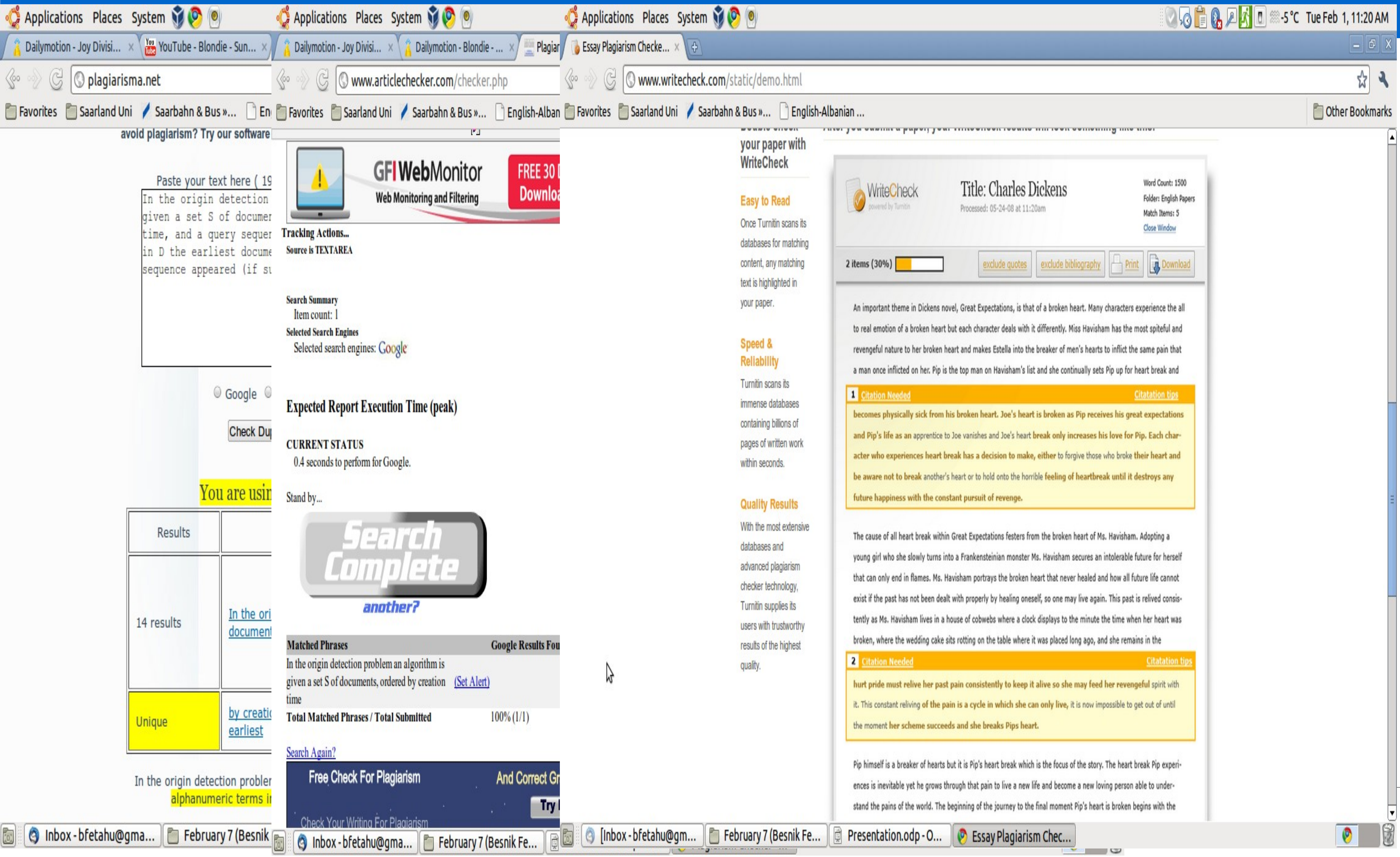
- Solution to this problem, two main philosophies:
  - Prevention
  - Detection.

Introduction – Prevention & Detection

- Prevention mechanisms:
  - Subscription
  - Distribute by CD-ROMs, example IEEE
  - Watermark
  - Active Documents, etc.

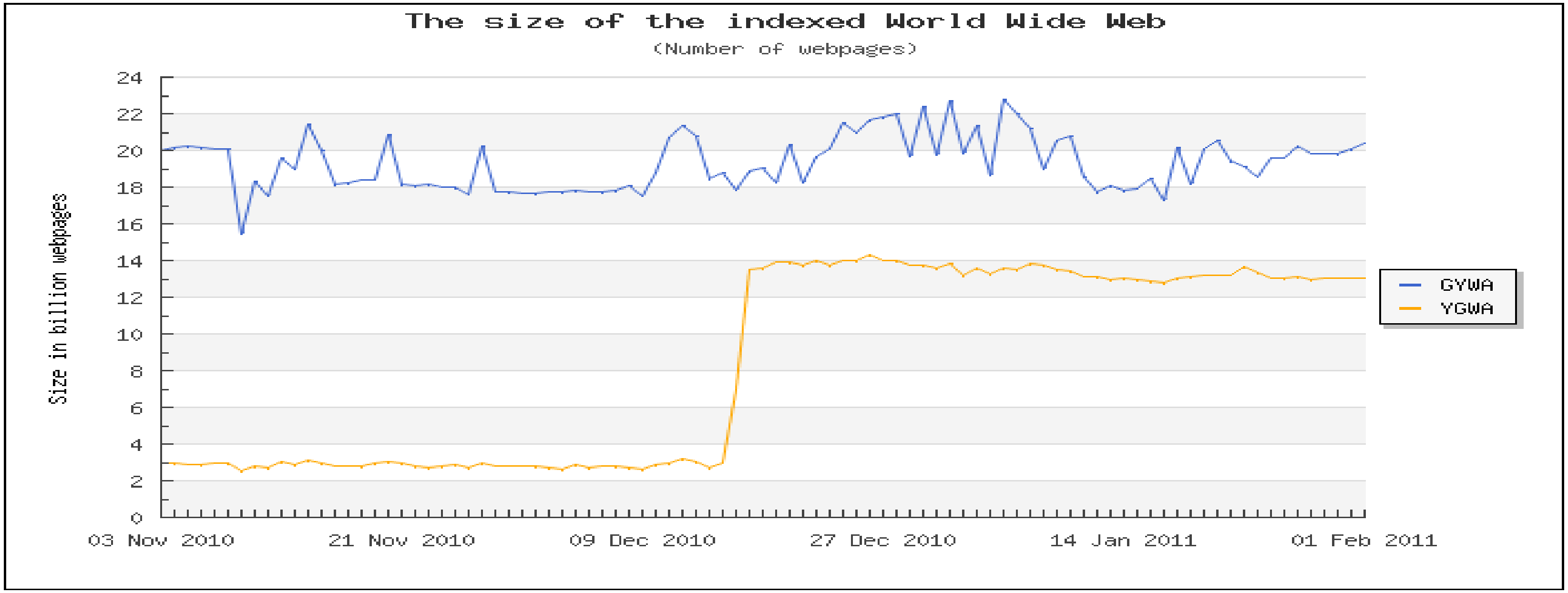
- Detection mechanism:
  - Match new content to previously published ones.
  - Operate on semantic level.

Introduction - Examples



Introduction - Problems

- Problem: huge amount of information available.



GYBA = Sorted on Google, Yahoo!, Bing and Ask
YGBA = Sorted on Yahoo!, Google, Bing and Ask

Source: http://www.worldwidewebsize.com/

Types of Replication

- Billions of pages on the Internet, categorized as follows:
  - *20 – 40 % identical copies*
  - *Near duplicate pages*
  - *Partial replication*
  - *Semantic duplication.*

- Research is focused into these categories:
  - *Paragraph replication*
  - *Problem with spammers*
  - *Return search snippets from search engines*
  - *Mark novel information on web browsers.*

Previous work

- Main papers that are referenced in this work are:
  - *Fingerprinting by Random Polynomials - Rabin O. M.*
  - Syntactic Clustering of the Web - Broder A. et.al.,
  - Copy Detection Mechanism for Digital Documents - Brin S., et. al.

Previous work - Fingerprints

- Given an *n-bit message* $\boldsymbol{m_0,\ldots,m_{n-1}}$, we view it as a polynomial of degree n-1 over the finite field.

$$f(x) = m_0 + m_1 x + \ldots + m_{n-1} x^{n-1}$$

- Pick a random irreducible polynomial **p(x)** of degree k over **GF(2)**.

- Define a fingerprint of m to be the remainder **r(x)** of **f(x) / p(x)** over **GF(2)** which can be viewed as a polynomial of degree k-1 or as a k-bit number.

Previous work – Syntactic Clustering

- Makes use of shingles.
- Clusters documents on semantic level
- Interesting approach on measuring document similarity.
- Reduced time complexity, without limitations.

Previous work – Copy Detection Mechanism for D.D.

- A framework called ***COPS***, which performs operations (*Subset,* Overlap, and *Plagiarism*)
- Chunking methods (used in other papers).

Strategy	Summary	Example on ABCDEF (k=3)	Space	#units	SEC <=
A	1 unit	A, B, C, D, E, F	|r|	1	|r|
B	K units, 0 over	ABC, DEF	|r| / k	K	1
C	K units, k – 1 over	ABC, BCD, CDE, DEF	|r|	K	|r| / k
D	Hashed breakpoints	AB, CDEF	|r| / k	K	|r|

Outline of the algorithm

- An important aspect for which the authors had to take into consideration were:
  - *Space efficient*
  - *Real-time*.

- A rough outline of the algorithm looks like this:
  - Fingerprint each document,
  - ***Selection algorithms***: shingles to save in the hash table.
  - ***Estimation algorithms***: determine the origin of a shingle.
  - ***Eviction algorithms***: determine, which shingle to keep.

Outline of the algorithm – Cont.

- Input to the algorithm:
  - A set S of sequence of tokens.
  - A “query” which consist of an additional sequence D.
  - A parameter k - number of consecutive tokens in a shingle.

- Phases of the algorithm:
  - Selection Phase
  - Hashing Phase, and
  - Estimation Phase.

Selection Phase

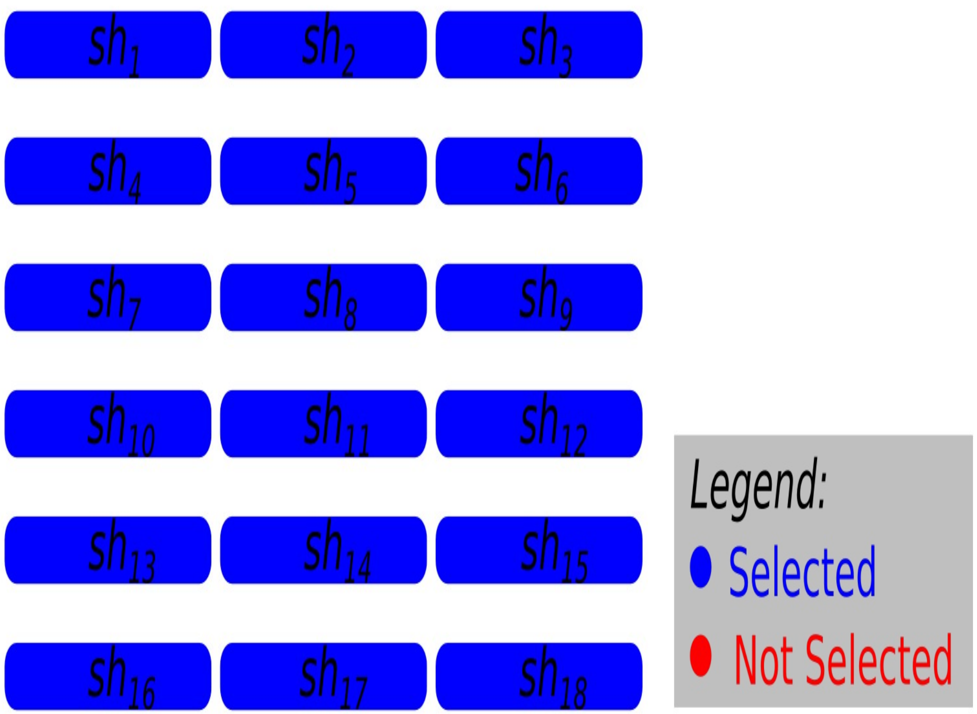
- Each document is converted into a set of ***fingerprints***:
  - All shingles of D are generated and converted into a 62 bit fingerprint.
  - A subset of shingles is selected based on their fingerprints.

Selection Phase - Methods

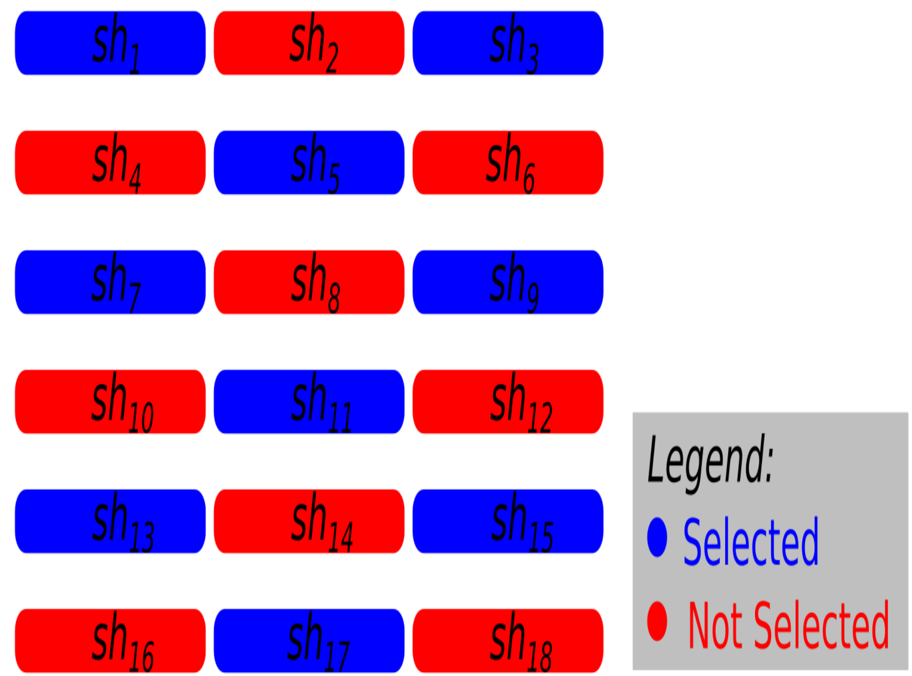
- Experimented with a numerous selection methods:
  - ***All*** – a baseline algorithm that selects all shingles.
  - ***Every l-th*** (lth)
  - ***Modulo l*** (M-l)
  - ***Winnowing w*** (W-w)
  - ***Revised Hash-breaking*** (Hb-p)
  - ***DCT*** (DCT-p), and
  - ***Hailstorm***.

Selection Phase – All & I-th

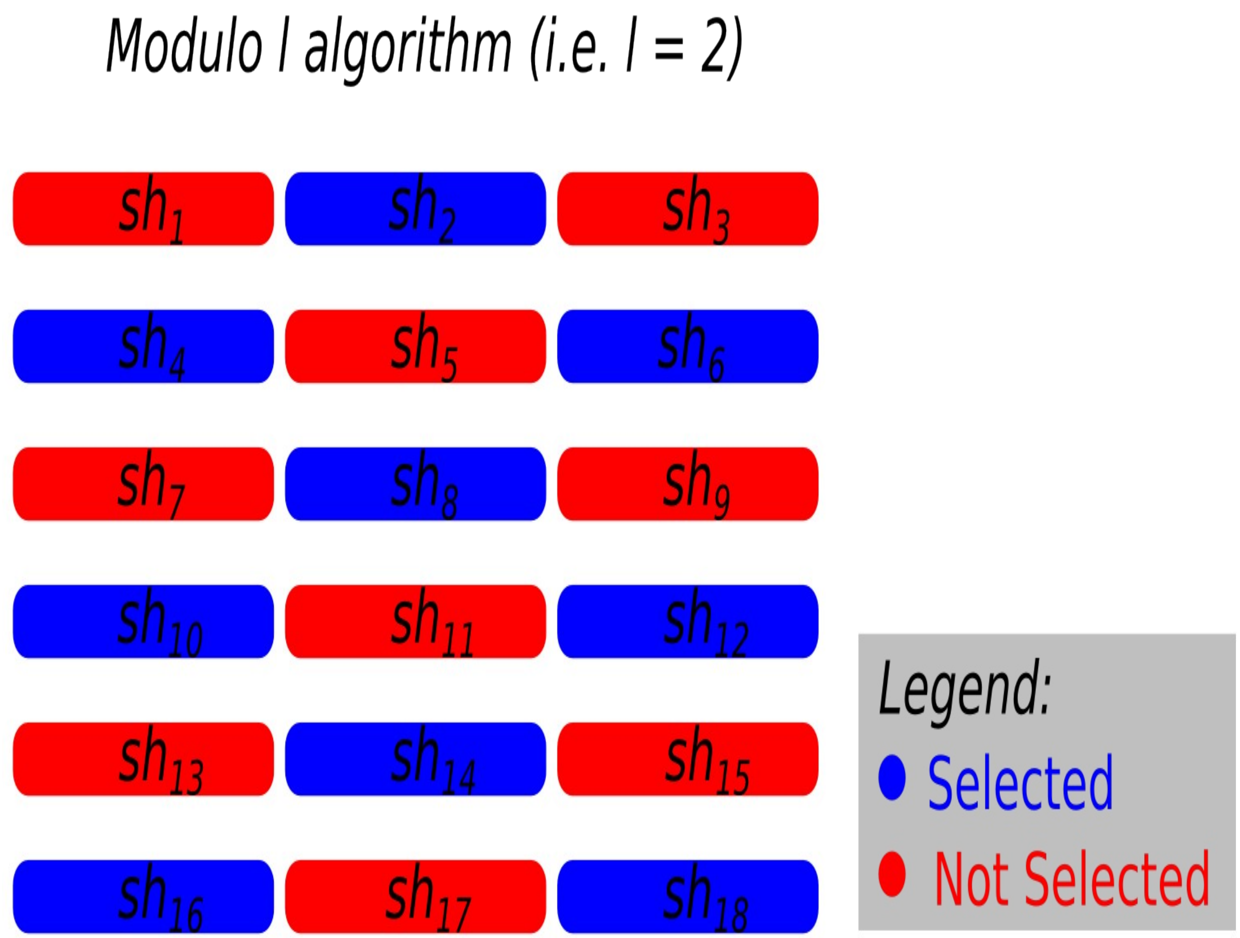
All - baseline algorithm



Every I-th shingle algorithm (i.e., 2nd)



Selection Phase – M-l



Selection Phase - W-w

```
A do run run run, a do run run
```
(a) Some text.

```
adorunrunrunadorunrun
```
(b) The text with irrelevant features removed.

```
adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun
```
(c) The sequence of 5-grams derived from the text.

```
77 74 42 17 98 50 17 98 8 88 67 39 77 74 42
17 98
```
(d) A hypothetical sequence of hashes of the 5-grams.

(77, 74, 42, **17**)	(74, 42, 17, 98)
(42, 17, 98, 50)	(17, 98, 50, **17**)
(98, 50, 17, 98)	(50, 17, 98, **8**)
(17, 98, 8, 88)	(98, 8, 88, 67)
(8, 88, 67, 39)	(88, 67, **39**, 77)
(67, 39, 77, 74)	(39, 77, 74, 42)
(77, 74, 42, **17**)	(74, 42, 17, 98)

(e) Windows of hashes of length 4.

```
17 17 8 39 17
```
(f) Fingerprints selected by winnowing.

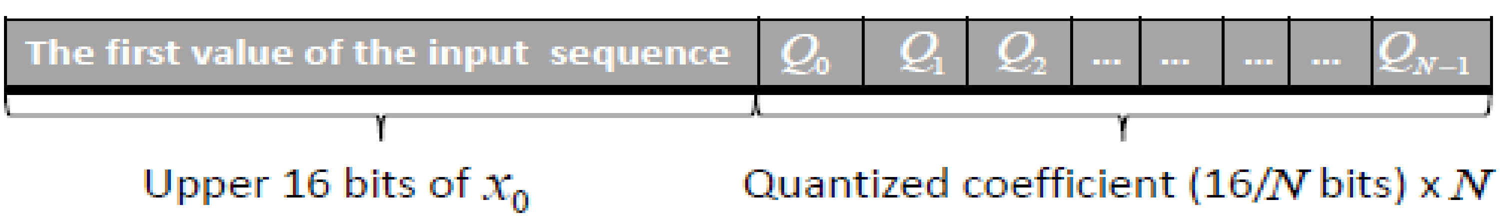
```
[17,3] [17,6] [8,8] [39,11] [17,15]
```
(g) Fingerprints paired with 0-base positional information.

Selection Phase – Hb-p

- ***Revised Hash-breaking***
  - Apply a hash function ***h*** to each token.
  - Break the document into non-overlapping segments.
  - Fingerprint all the tokens contained in a segment.
  - Number of segments is **1/p**.

Selection Phase - *Dct-p*

- *DCT fingerprinting:*
  - Text segments, using ***Hb-p***,
  - Hash values for words in the segments,
  - Vertical translation of hash values, ***median*** located at 0,
  - ***Normalize*** the values by the max hash value,
  - Perform ***DCT*** with the normalized values,
  - ***Quantize*** each coefficient to be fitted into a small number of bits 2, 3, or 4,
  - Form a fingerprint with the quantized coefficients Q_k's.



Selection Phase - *Hs*

- *Hailstorm (Hs)*
  - Fingerprint every token
  - Select a shingle s iff the minimum fingerprint value of all tokens occurs at the first or last position of s.
  - Probability that a shingle is chosen is **2/k** if all tokens are different.

Selection Phase - Properties

- Contribution on this phase ***Hailstorm Alg***.

- Properties of the algorithms:
  - ***Winnowing*** – fulfills locality property.
  - ***Modulo l*** and ***Hailstorm*** – fulfill context freeness (better).

•**Lemma**: In every document D, any token is covered by at least one k-shingle selected by algorithm Hailstorm.

Hashing Phase

- Fixed-size hash table, split into ***buckets*** each containing up to ***64*** shingles.

- Space needed to store a shingle with its accompanying information varies between algorithms, is between ***14-18 bytes.***

Hashing Phase

bucket[1,1]
bucket[1,2]
bucket[1,3]
bucket[1,4]
•
•
•
bucket[1,61]
bucket[1,62]
bucket[1,63]
bucket[1,64]
bucket[2,1]
bucket[2,2]
bucket[2,3]
bucket[2,4]
bucket[2,5]
bucket[2,6]
•
•
•
bucket[k,59]
bucket[k,60]
bucket[k,61]
bucket[k,62]
bucket[k,63]
bucket[k,64]

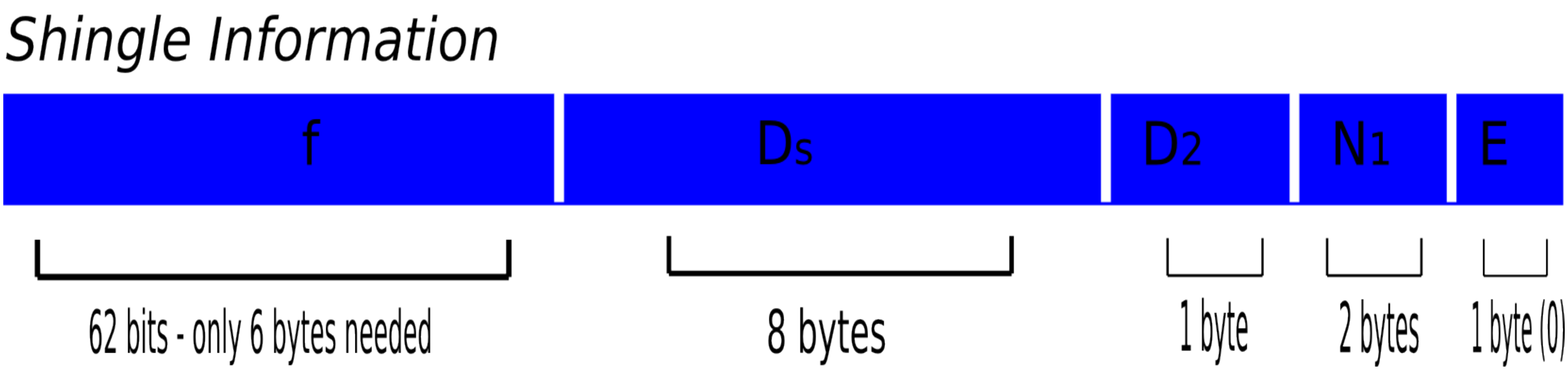
Shingle

t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8

In each cell of a bucket is saved a shingle, of size k where in our example it is k = 8.

Hashing Phase – Shingle Information

- Parts contained in a shingle:
  - The ***fingerprint*** of s itself,
  - Its ***origin D_s***,
  - Its ***offset D_2***,
  - Information about ***neighboring shingles in D_s***,
  - Information for the ***eviction algorithm***.



Hashing Phase - Methods

- The work focuses on mainly three algorithms:
  - ***Random*** – evict a random shingle.
  - ***Copy – Count (CC)*** – copy count for each shingle.
  - ***Lucky Shingle (LS)*** – 1-byte score, gives a weighted variant of copy count.

Hashing Phase – Copy Count

Copy Count - Scores

$shingle_1$	1
$shingle_2$	1
$shingle_3$	1
$shingle_4$	1
$shingle_5$	1
$shingle_6$	1
$shingle_7$	1
$shingle_8$	1
$shingle_9$	2
$shingle_{10}$	1
$shingle_{11}$	1
$shingle_{12}$	1
....	1
$shingle_{63}$	1
$shingle_{64}$	1

for shingle' find a shingle from the hash table that matches.

if $shingle_9$ == shingle'
score = score + 1

Hashing Phase – Lucky Score

- ***Lucky Score (LS)*** – incremental steps:
  - Set lucky score to 1 for each shingle,
  - Increment the score of the first and last shingle of a copied block by $floor(\sqrt{b-2})$
  - If a shingle is the first or the last of its document, increment score additionally by 3
  - For every y-th selected shingle in D, increment the score by 1 (y=7).
  - If average score of all shingles reaches some limit, divide by 2.

Estimation Phase

- Input information: retrieved shingles, guess the origin of each shingle.

- Main work is focused on these methods:
  - ***No Bridging (NB)***
  - ***Expansion (E)***
  - ***Bridging Algorithm (B)***
  - ***Bridging with Expansion (BE).***

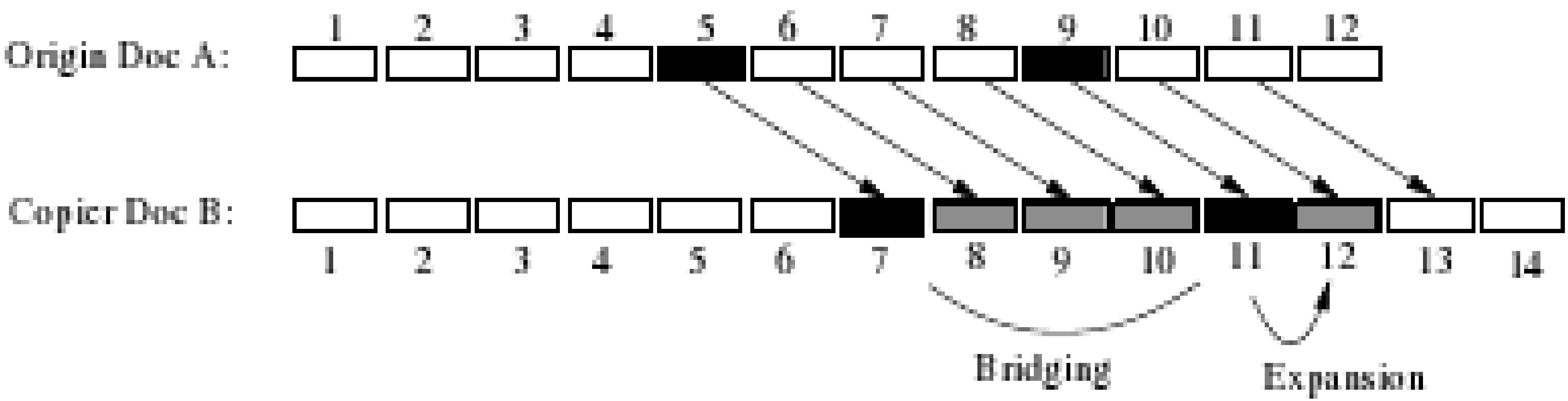
Estimation Phase - Bridging

- ***Bridging Alg.*** (***their work***), for each two selected shingles ***s*** and ***s'***:

  - The *offset* of ***s*** in **D** is less than the *offset* of ***s'***
  - For ***s*** and ***s'***, same *origin* is stored in the hash table.
  - Difference of their *offset* in **D** equals to the *offset* stored in the hash table.
  - None of the shingles that occur after ***s*** and ***s'*** in **D** fulfill the previous properties.

Estimation Phase – Bridging with Expansion

- ***Bridging with Expansion*** – previous properties hold, and two additional ones:



Evaluation

- Evaluation was done on two separate datasets:
  - ***Blog data set*** – 8.6 million pages
  - ***German pages*** – 1.3 million pages

- Various sizes of shingles were used, **k = 8** achieved highest results.

- Metrics to measure the performance of the framework:
  - ***Dominant Origin (DO)***
  - ***Selected Shingle Ratio (SSR)***
  - ***Token Freshness (TF).***

Evaluation - Statistics

- Statistics on the two separate dataset:

data set	# of documents	# of shingles	avg # of shingles per doc	with dom. origin	avg size of cop. blocks	shingle copy ratio
blogs	8,666,731	1.71 bil.	197	94%	17	0.36
Swiss	1,360,393	0.78 bil.	570	92%	13	0.16

Table 1: Various statistics of our data sets.

m	5000	2000	1000	500	200	100	50	20
Blogs	34.2	13.7	6.8	3.3	1.4	0.7	0.3	0.1
Swiss	57.5	23.0	11.5	5.8	2.3	1.2	0.6	0.2

Table 2: Different hash table sizes m in MB and the percentage of all shingles that fit into the hash table at one time (in percent) for both datasets.

Evaluation – Experimental Setups

- Selection Phase
  - ***Version A: random eviction & no estimation***
  - ***Version B: lucky eviction & BE estimation***
- Eviction Phase
  - ***Version A: All the baseline algorithm***
  - ***Version B: NHs***
- Estimation Phase
  - ***Version A: Copy Count***
  - ***Version B: Lucky Shingle***

Evaluation - Selection Alg.

Every l-th			Modulo l			
4th	6th	8th	NM2	NM3	M4	NM4
25%	17%	13%	15%	15%	25%	14%
Modulo l						
M5	NM5	M6	NM6	M7	NM7	M8
20%	14%	16%	12.5%	14%	11%	12%
Winnowing w						
NW5	NW6	NW7	W8	NW8	W9	NW9
16%	17%	16.5%	23.5%	17%	21%	16%
Dct p (same ssr as HB-p)					Hailstorm	M-l
Dct3	Dct4	Dct5	Dct6	Dct8	NHs	NM8
20%	15%	12%	10%	8%	17%	10%

Table 3: Percentage of shingles sent to the hash table by the selection algorithms in the blogs data set.

Evaluation – Selection Alg.

m	All	NHs	4th	NW8	NM3	Hb3	Dct8
5000	83.2	**98.8**	89.5	98.2	96.7	96.9	90.0
2000	79.8	**97.3**	83.2	96.2	95.7	95.6	90.0
1000	77.2	84.6	78.5	84.3	85.1	84.6	**89.5**
500	75.7	79.8	76.5	79.5	79.4	79.3	**82.8**
200	74.3	77.5	75.1	77.3	77.3	77.1	**77.6**
100	73.6	75.8	74.0	75.7	75.7	75.6	**76.3**
50	73.2	74.7	73.4	74.6	74.6	74.6	**75.3**
20	72.9	73.7	72.8	73.6	73.7	73.6	**74.0**
AvgDO	76.2	**82.8**	77.9	82.4	82.3	82.2	81.9

Table 4: Blogs data set: the DO score (in %) of the best performing selection algorithms for different hash table sizes (in MB) when combined with random eviction and no estimation. The maximum in each row is highlighted.

Evaluation - Selection Alg.

m	All	NHs	4th	NW8	NM3	Hb3	Dct8
5000	**99.2**	98.5	88.9	98.0	96.3	96.8	85.1
500	90.6	**93.7**	86.0	92.9	92.0	91.8	85.1
50	79.7	**84.3**	79.6	83.5	83.1	82.5	82.8
AvgDO	88.0	**91.0**	84.0	90.2	89.3	89.2	84.1

Table 5: Blogs data set: the DO score (in %) of the best performing selection algorithms for different hash table sizes (in MB) when combined with lucky eviction and BE estimation.

Data& Version	All	NHs	4th	NW8	NM3	Hb3	Dct8
Blogs A	81.0	**83.6**	79.5	83.4	82.7	82.6	82.2
Blogs B	87.9	**89.1**	83.6	88.7	87.6	87.6	84.3
Swiss A	80.9	81.6	76.0	81.4	**81.8**	81.1	79.3
Swiss B	90.6	**88.0**	79.2	87.6	87.7	86.7	59.7

Table 7: The overall score, i.e., the mean of the average DO and the average TF metric (in percent) on both data sets and both versions for the best performing selection algorithms. The maximum in each row (ignoring All) is highlighted.

Evaluation – Selection Alg.

m	All	NHs	4th	NW8	NM3	Hb3	Dct8
Version A: Random Eviction + No Estimation							
5000	**97.7**	93.7	89.7	93.4	91.6	90.7	87.5
500	**86.9**	86.2	82.7	86.2	84.9	84.8	84.4
50	**78.2**	77.2	75.0	77.3	76.5	77.1	77.3
AvgTF	**85.7**	84.5	81.1	84.3	83.2	83.1	82.5
Version B: Lucky Eviction + BE Bridging							
5000	**98.4**	93.6	89.7	93.5	91.7	90.8	86.2
500	**90.5**	89.6	85.1	89.4	88.1	88.1	85.9
50	81.4	81.6	77.9	81.4	80.5	81.6	**82.7**
AvgTF	**87.9**	87.2	83.2	87.0	85.8	86.0	84.6

Table 6: Blogs data set: the TF score (in %) of the best performing selection algorithms for different hash table sizes (in MB).

Evaluation - Eviction Alg.

m	All				NHs			
	R	LRU	CC	LS	R	LRU	CC	LS
Dominant Origin Metric								
5000	83.2	93.0	**96.1**	95.0	98.8	98.8	98.8	98.8
500	75.7	77.6	**80.9**	80.7	79.8	88.2	**93.1**	87.5
50	73.2	73.4	72.6	**73.5**	74.7	75.9	**78.5**	77.8
Avg	76.2	78.9	**81.1**	80.4	82.8	86.0	**88.3**	86.7
Token Freshness Metric								
5000	97.6	96.1	96.8	**98.6**	93.5	93.5	93.5	93.5
500	86.9	83.8	84.1	**89.6**	86.2	88.1	**88.8**	86.0
50	78.2	76.7	71.8	**79.8**	77.2	**77.9**	75.1	77.1
Avg	85.7	83.7	81.8	**87.7**	84.3	**85.3**	84.2	84.4
Overall Score								
	80.7	81.3	81.4	**84.1**	83.5	85.6	86.2	**85.5**

Table 8: Blogs data set: the DO and TF score (in %) of all eviction algorithms for the selection algorithms All and NHs with no estimation algorithm for different hash table sizes (in MB).

Evaluation - Estimation Alg.

m	NHs + CC				NHs + LS			
	NB	E	B	BE	NB	E	B	BE
Dominant Origin Metric								
5000	**98.8**	**98.8**	97.4	98.5	**98.8**	**98.8**	97.4	98.5
500	93.1	93.1	92.1	92.7	87.5	92.3	92.6	**93.7**
50	78.5	78.8	77.9	79.2	77.8	82.4	83.0	**84.3**
Avg	88.3	88.7	87.6	88.4	86.7	89.7	89.7	**91.0**
Token Freshness Metric								
5000	93.5	93.5	**93.6**	**93.6**	93.5	93.5	**93.6**	**93.6**
500	88.8	88.6	89.3	88.6	86.0	88.4	89.5	**89.6**
50	75.1	75.5	76.4	75.8	77.1	80.0	81.5	**81.6**
Avg	84.2	84.2	85.0	84.5	84.4	86.2	87.1	**87.2**
Overall Score								
	86.2	86.4	86.3	86.5	85.5	87.9	88.4	**89.1**

Table 9: Blogs data set: the DO and TF score (in %) of all bridging algorithms with the selection algorithm NHs and the eviction algorithms CC and LS, for different hash table sizes m (in MB).

Evaluation - Estimation Alg.

m	ALL + CC				ALL + LS			
	NB	E	B	BE	NB	E	B	BE
Dominant Origin Metric								
5000	96.8	96.6	96.6	96.4	**98.6**	98.5	98.4	98.4
500	84.1	84.0	84.4	83.8	89.6	**90.5**	90.3	**90.5**
50	71.8	71.9	72.0	71.9	79.8	81.3	81.2	**81.4**
Avg	81.8	81.8	81.9	81.7	87.7	88.5	88.4	**88.6**
Token Freshness Metric								
5000	96.1	96.6	95.2	96.1	95.0	**99.5**	97.9	99.2
500	80.9	82.9	82.1	82.3	80.7	90.2	89.2	**90.6**
50	72.6	75.5	74.4	75.3	73.4	78.9	78.2	**79.7**
Avg	81.0	82.7	81.5	82.3	80.3	87.7	86.6	**87.9**
Overall Score								
	81.4	82.2	81.7	82.0	84.0	88.1	87.5	**88.3**

Table 10: Blogs data set: the DO and TF score (in %) of all bridging algorithms with the selection algorithm All and the eviction algorithms CC and LS, for different hash table sizes m (in MB).

Evaluation – Estimation Alg.

m	number of extra-labeled shingles	Accuracy
5000	6694	10.5
2000	8139	30.8
1000	33003	84.8
500	122614	96.6
200	163484	98.3
100	155107	98.6
50	154093	98.8
20	133737	98.6

Table 11: Blogs data set and selection algorithm NHs: The number of extra-labeled shingles and the accuracy (in percent) of the origin of extra-labeled shingles for different hash table sizes (in MB).

Overall Evaluation

• The performance of the system is the best when using ***NHs*** with ***lucky eviction*** and ***BE estimation***, where with the decrease of **m** the performance decreases, while the results for both metrics (***DO*** *and* ***TF***) are quite good.

•Whereas if used algorithm ***All*** (when combined with ***lucky eviction*** and ***expansion***) results are lower than those produced by ***NHs*** algorithm.

Overall Evaluation

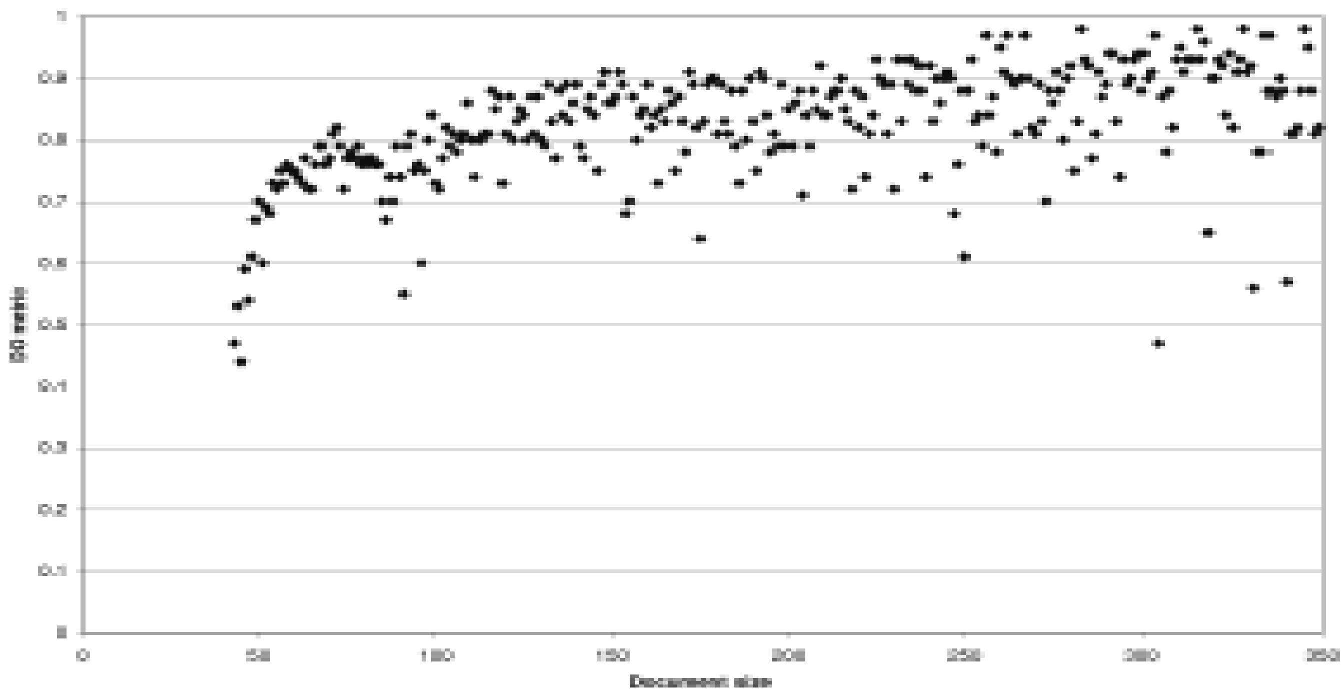


Figure 2: For m =20MB the DO metric averaged for all documents of a fixed size for the selection algorithm NHs with lucky eviction and BE estimation.

Conclusions

- Pro's
  - Fixed hash table size
  - Good results
  - ***Hailstorm***
- Con's
  - No reasoning on score changes in ***LS***
  - *Security*