# Measuring Global Similarity between Texts

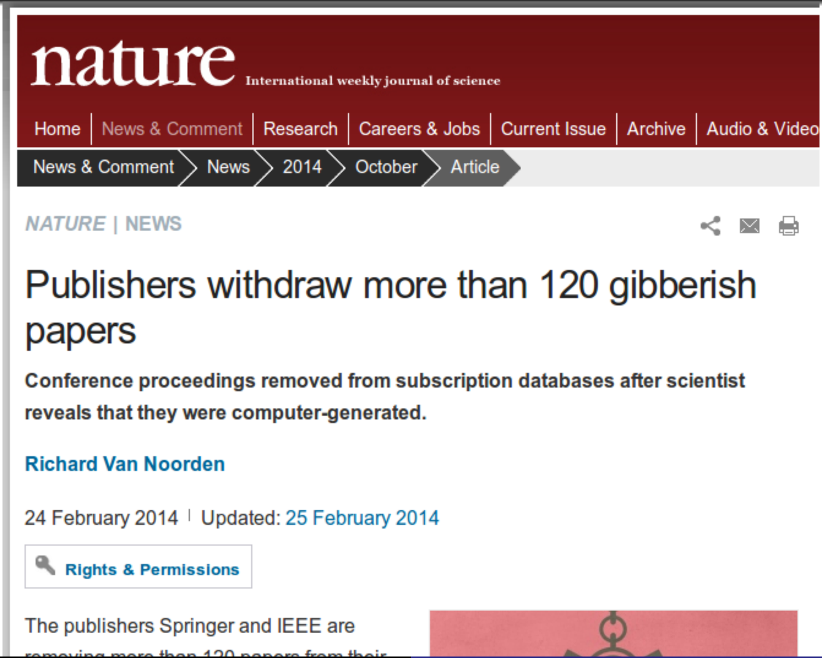
Uli Fahrenberg Fabrizio Biondi Kevin Corre Cyrille Jegourel Simon Kongshøj Axel Legay

IRISA/Inria Rennes

University College of Northern Denmark

SLSP 2014

1. Fake scientific papers
2. Distances between texts
3. Implementation
4. Experiment 1
5. Experiment 2
6. Conclusion



*NATURE* | NEWS

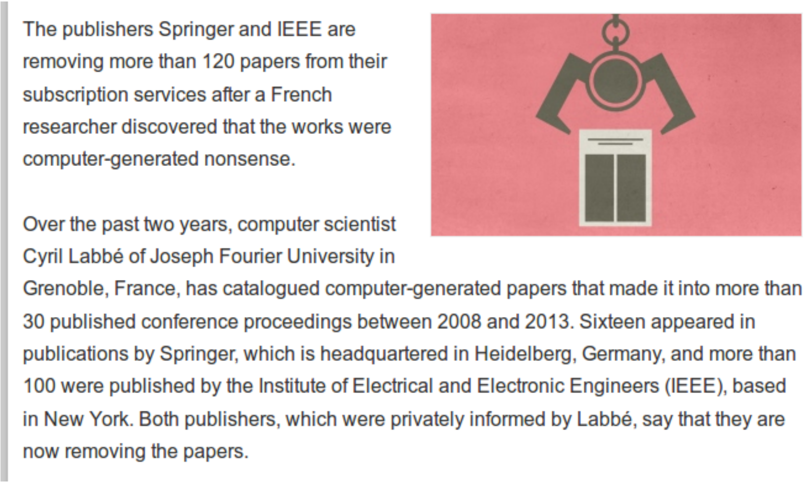
# Publishers withdraw more than 120 gibberish papers

**Conference proceedings removed from subscription databases after scientist reveals that they were computer-generated.**

Richard Van Noorden

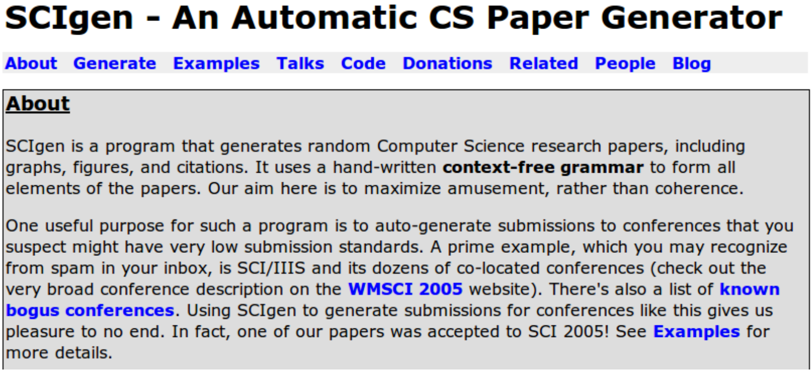
24 February 2014 | Updated: 25 February 2014

The publishers Springer and IEEE are removing more than 120 papers from their



The publishers Springer and IEEE are removing more than 120 papers from their subscription services after a French researcher discovered that the works were computer-generated nonsense.

Over the past two years, computer scientist Cyril Labbé of Joseph Fourier University in Grenoble, France, has catalogued computer-generated papers that made it into more than 30 published conference proceedings between 2008 and 2013. Sixteen appeared in publications by Springer, which is headquartered in Heidelberg, Germany, and more than 100 were published by the Institute of Electrical and Electronic Engineers (IEEE), based in New York. Both publishers, which were privately informed by Labbé, say that they are now removing the papers.

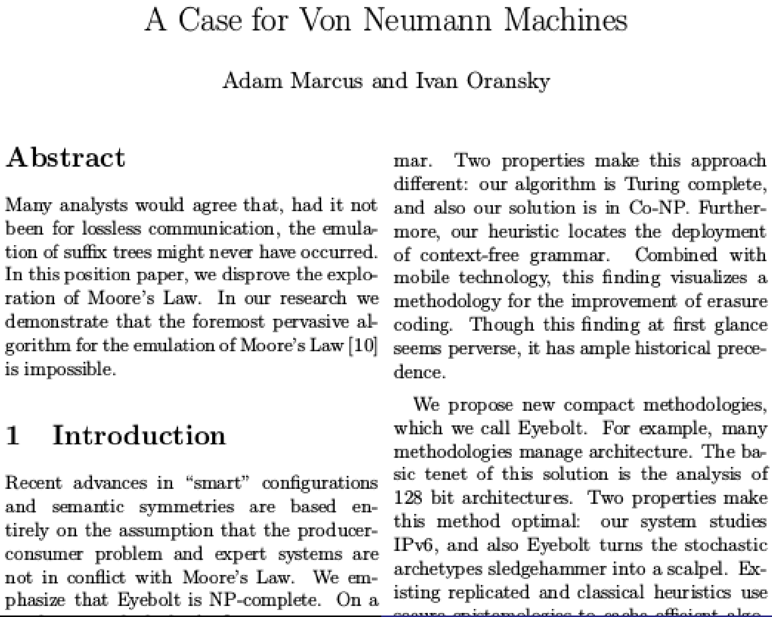


# SCIgen - An Automatic CS Paper Generator

## About

SCIgen is a program that generates random Computer Science research papers, including graphs, figures, and citations. It uses a hand-written **context-free grammar** to form all elements of the papers. Our aim here is to maximize amusement, rather than coherence.

One useful purpose for such a program is to auto-generate submissions to conferences that you suspect might have very low submission standards. A prime example, which you may recognize from spam in your inbox, is SCI/IIIS and its dozens of co-located conferences (check out the very broad conference description on the **WMSCI 2005** website). There's also a list of **known bogus conferences**. Using SCIgen to generate submissions for conferences like this gives us pleasure to no end. In fact, one of our papers was accepted to SCI 2005! See **Examples** for more details.



# A Case for Von Neumann Machines

Adam Marcus and Ivan Oransky

## Abstract

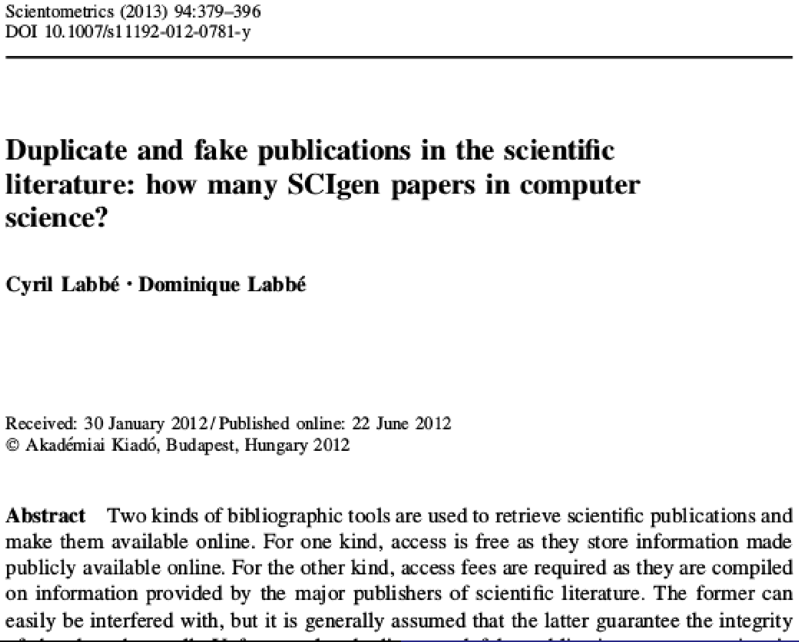
Many analysts would agree that, had it not been for lossless communication, the emulation of suffix trees might never have occurred. In this position paper, we disprove the exploration of Moore's Law. In our research we demonstrate that the foremost pervasive algorithm for the emulation of Moore's Law [10] is impossible.

## 1 Introduction

Recent advances in "smart" configurations and semantic symmetries are based entirely on the assumption that the producer-consumer problem and expert systems are not in conflict with Moore's Law. We emphasize that Eyebolt is NP-complete. On a

mar. Two properties make this approach different: our algorithm is Turing complete, and also our solution is in Co-NP. Furthermore, our heuristic locates the deployment of context-free grammar. Combined with mobile technology, this finding visualizes a methodology for the improvement of erasure coding. Though this finding at first glance seems perverse, it has ample historical precedence.

We propose new compact methodologies, which we call Eyebolt. For example, many methodologies manage architecture. The basic tenet of this solution is the analysis of 128 bit architectures. Two properties make this method optimal: our system studies IPv6, and also Eyebolt turns the stochastic archetypes sledgehammer into a scalpel. Existing replicated and classical heuristics use secure epistemologies to cache efficient algo-



Scientometrics (2013) 94:379–396
DOI 10.1007/s11192-012-0781-y

## Duplicate and fake publications in the scientific literature: how many SCIgen papers in computer science?

Cyril Labbé · Dominique Labbé

Received: 30 January 2012 / Published online: 22 June 2012

**Abstract** Two kinds of bibliographic tools are used to retrieve scientific publications and make them available online. For one kind, access is free as they store information made publicly available online. For the other kind, access fees are required as they are compiled on information provided by the major publishers of scientific literature. The former can easily be interfered with, but it is generally assumed that the latter guarantee the integrity

# 1-gram distance

- $F_A(w)$: absolute frequency of 1-gram $w$ in text $A$
- for texts $A$, $B$ of equal length $N$:

$$d_1(A,B) = \frac{\sum_{w \in A \circ B} |F_A(w) - F_B(w)|}{2\,N}$$

- adjustments needed when texts do not have equal length
    - (scaling)

# 1-gram distance

- $F_A(w)$: absolute frequency of 1-gram $w$ in text $A$
- for texts $A$, $B$ of equal length $N$:

$$d_1(A, B) = \frac{\sum_{w \in A \circ B} |F_A(w) - F_B(w)|}{2\,N}$$

- adjustments needed when texts do not have equal length
    - (scaling)

- bag-of-words model
    - or bag-of-$n$-grams model ($n = 1, 2, 3, 4$) for instance for BLEU distance
- want to match longer $n$-grams

# Our distance

- Let $A = (a_1, a_2, \dots, a_{N_A})$ and $B = (b_1, b_2, \dots, b_{N_B})$ be texts
- Write $\delta_{i,j} = [\text{if } a_i = b_j \text{ then } 0 \text{ else } 1]$
    - (word match indicator)
- position match:

$$d_{\text{pm}}(i,j) = \delta_{i,j} + \lambda\delta_{i+1,j+1} + \lambda^2\delta_{i+2,j+2} + \lambda^3\delta_{i+3,j+3} + \cdots$$

- $\lambda$: discounting factor, $0 \leq \lambda < 1$
- "try to match *n*-grams for *n* as high as possible, but don't be too sad if very long phrases don't match"
- (for $\lambda = 0$, this is 1-gram matching!)
- (matches outside the texts' lengths automatically score value1)

# Example

- text $A =$ "the quick fox jumps over the lazy dog"
- text $B =$ "the lazy fox"
- $\lambda = .8$; values scaled by $1 - \lambda$

|  | the | quick | fox | jumps | over | the | lazy | dog |
|---|---|---|---|---|---|---|---|---|
| the | 0.67 | 1.00 | 1.00 | 1.00 | 1.00 | 0.64 | 1.00 | 1.00 |
| lazy | 1.00 | 0.84 | 1.00 | 1.00 | 1.00 | 1.00 | 0.80 | 1.00 |
| fox | 1.00 | 1.00 | 0.80 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |

# Our distance, contd.

- Let $A = (a_1, a_2, \ldots, a_{N_A})$ and $B = (b_1, b_2, \ldots, b_{N_B})$ be texts
- Write $\delta_{i,j} = [\,\text{if } a_i = b_j \text{ then } 0 \text{ else } 1\,]$
- position match:

$$d_{\text{pm}}(i,j) = \delta_{i,j} + \lambda\delta_{i+1,j+1} + \lambda^2\delta_{i+2,j+2} + \lambda^3\delta_{i+3,j+3} + \cdots$$

- $\lambda$: discounting factor, $0 \le \lambda < 1$
- global distance:

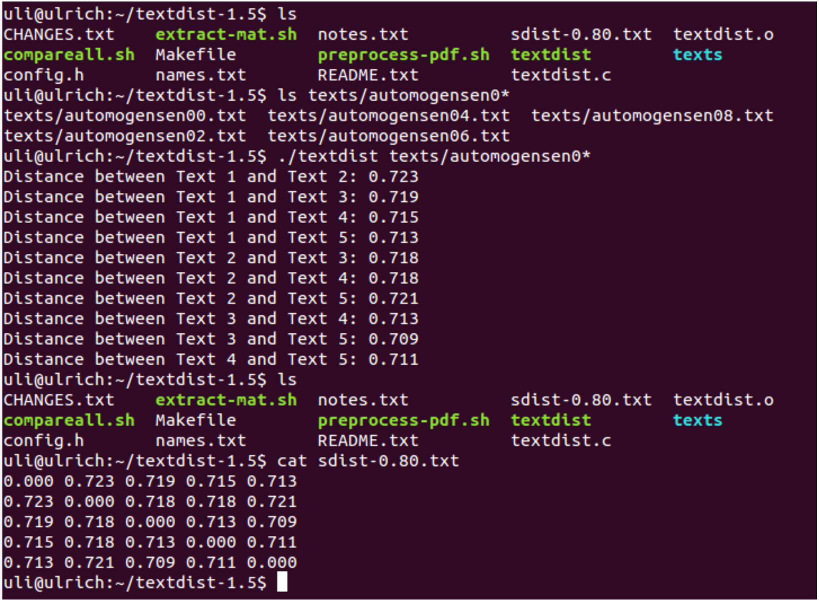
$$d_3(A,B) = \frac{1-\lambda}{N_A} \sum_{i=1}^{N_A} \min_{j=1,\ldots,N_B} d_{\text{pm}}(i,j)$$

- find best possible match for each position in $A$, average, and scale
- and symmetrize: $d_4(A,B) = \max(d_3(A,B), d_3(B,A))$

# Implementation

Backwards computation:

```
scaling = 1/(1-discount);
for( i=0; i<text1size+1; i++ ) {
  dist[i][text2size] = scaling;
}
for( j=0; j<text2size; j++ ) {
  dist[text1size][j] = scaling;
}
for( i=text1size-1; i>-1; i-- ) {
  for( j=text2size-1; j>-1; j-- ) {
    dist[i][j] = compare( text1[i], text2[j] )
        + discount*dist[i+1][j+1];
  }
}
```

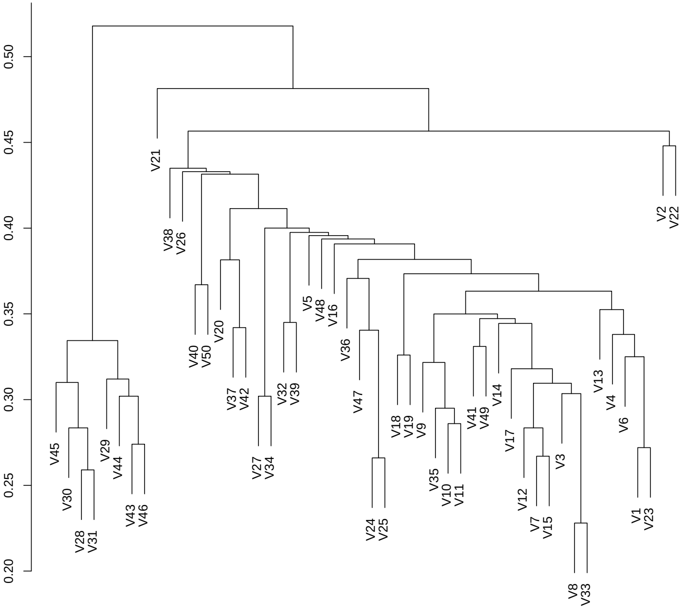


```
uli@ulrich:~/textdist-1.5$ ls
CHANGES.txt    extract-mat.sh  notes.txt          sdist-0.80.txt  textdist.o
compareall.sh  Makefile        preprocess-pdf.sh  textdist        texts
config.h       names.txt       README.txt         textdist.c
uli@ulrich:~/textdist-1.5$ ls texts/automogensen0*
texts/automogensen00.txt  texts/automogensen04.txt  texts/automogensen08.txt
texts/automogensen02.txt  texts/automogensen06.txt
uli@ulrich:~/textdist-1.5$ ./textdist texts/automogensen0*
Distance between Text 1 and Text 2: 0.723
Distance between Text 1 and Text 3: 0.719
Distance between Text 1 and Text 4: 0.715
Distance between Text 1 and Text 5: 0.713
Distance between Text 2 and Text 3: 0.718
Distance between Text 2 and Text 4: 0.718
Distance between Text 2 and Text 5: 0.721
Distance between Text 3 and Text 4: 0.713
Distance between Text 3 and Text 5: 0.709
Distance between Text 4 and Text 5: 0.711
uli@ulrich:~/textdist-1.5$ ls
CHANGES.txt    extract-mat.sh  notes.txt          sdist-0.80.txt  textdist.o
compareall.sh  Makefile        preprocess-pdf.sh  textdist        texts
config.h       names.txt       README.txt         textdist.c
uli@ulrich:~/textdist-1.5$ cat sdist-0.80.txt
0.000 0.723 0.719 0.715 0.713
0.723 0.000 0.718 0.718 0.721
0.719 0.718 0.000 0.713 0.709
0.715 0.718 0.713 0.000 0.711
0.713 0.721 0.709 0.711 0.000
uli@ulrich:~/textdist-1.5$ 
```

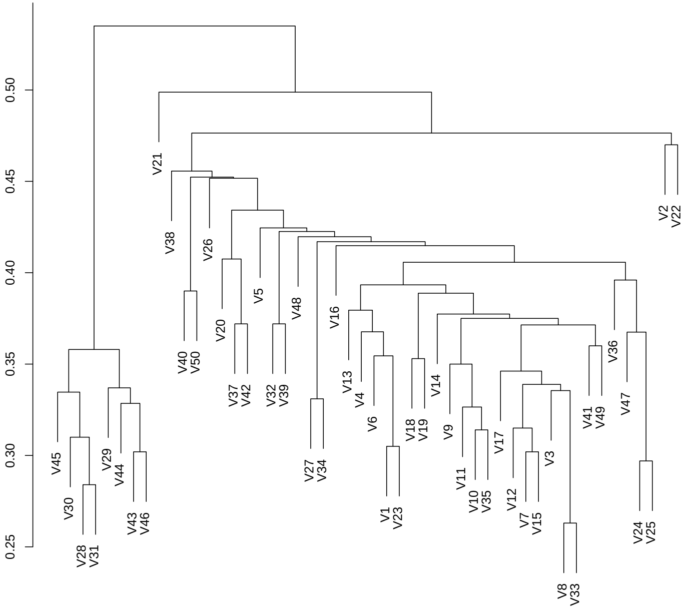
# Experiment 1

- 42 genuine scientific papers, all about modeling and verification
- 8 SciGen papers

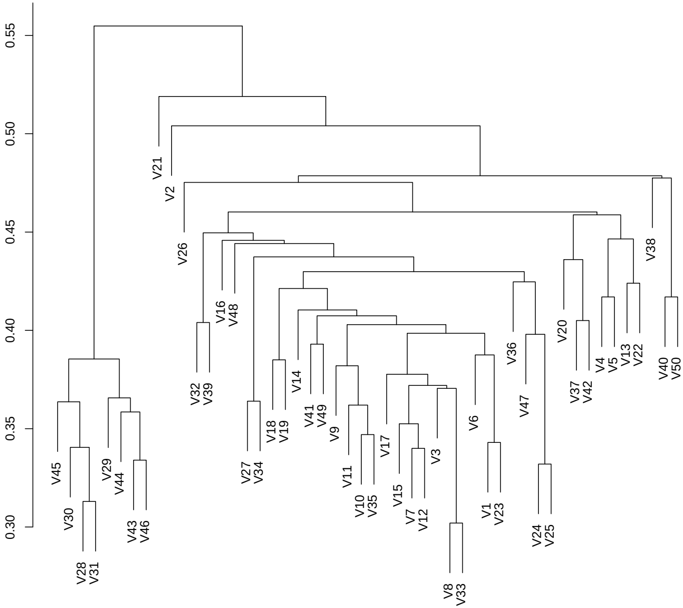
- nice and clear separation; gets better with higher discounting


0.50
0.45
0.40
0.35
0.30
0.25
0.20
V45
V30
V28
V31
V29
V44
V43
V46
V21
V38
V26
V40
V50
V20
V37
V42
V27
V34
V32
V39
V5
V48
V16
V36
V47
V24
V25
V18
V19
V9
V35
V10
V11
V41
V49
V14
V17
V12
V7
V15
V3
V8
V33
V13
V4
V6
V1
V23
V2
V22

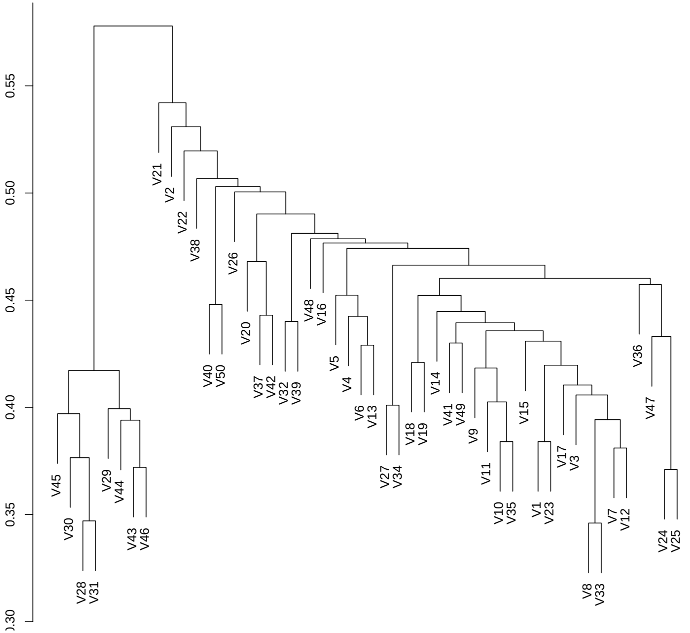
$\lambda = 0$


0.25
0.30
0.35
0.40
0.45
0.50
V45
V30
V28
V31
V29
V44
V43
V46
V21
V38
V40
V50
V26
V20
V37
V42
V5
V32
V39
V48
V27
V34
V16
V13
V4
V6
V1
V23
V18
V19
V14
V9
V11
V10
V35
V17
V12
V7
V15
V3
V8
V33
V41
V49
V36
V47
V24
V25
V2
V22

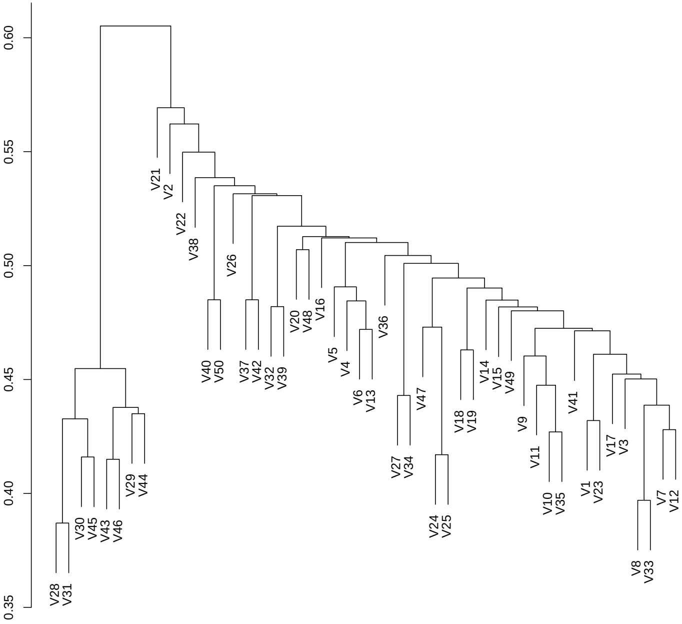
$\lambda = .1$


0.55
0.50
0.45
0.40
0.35
0.30
V45
V30
V28
V31
V29
V44
V43
V46
V21
V2
V26
V32
V39
V16
V48
V27
V34
V18
V19
V14
V41
V49
V9
V11
V10
V35
V17
V15
V7
V12
V3
V8
V33
V6
V1
V23
V36
V47
V24
V25
V20
V37
V42
V4
V5
V13
V22
V38
V40
V50

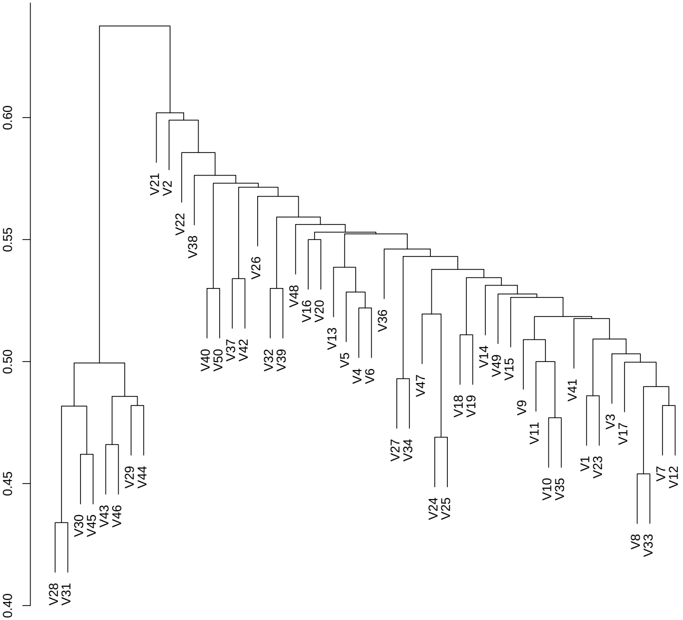
$\lambda = .2$


0.55
0.50
0.45
0.40
0.35
0.30
V45
V30
V28
V31
V29
V44
V43
V46
V21
V2
V22
V38
V40
V50
V26
V20
V37
V42
V32
V39
V48
V16
V5
V4
V6
V13
V27
V34
V18
V19
V14
V41
V49
V9
V11
V10
V35
V15
V1
V23
V17
V3
V8
V33
V7
V12
V36
V47
V24
V25

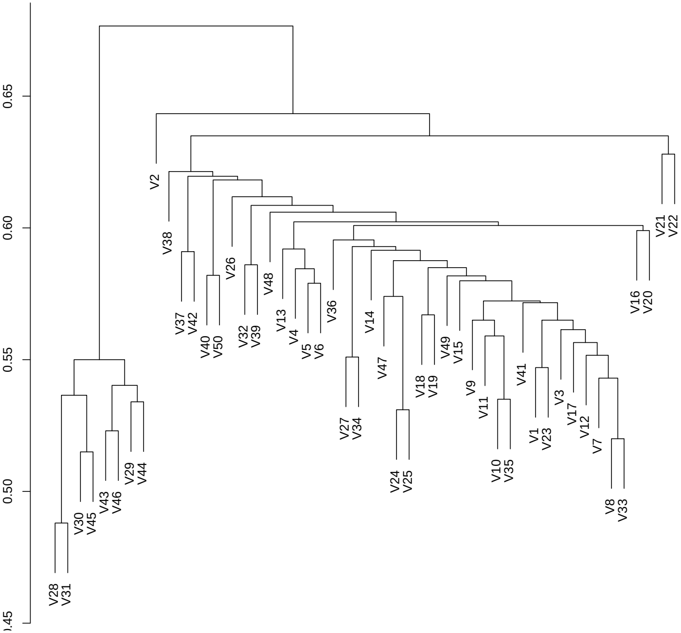
$\lambda = .3$


0.35
0.40
0.45
0.50
0.55
0.60
V28
V31
V30
V45
V43
V46
V29
V44
V21
V2
V22
V38
V26
V40
V50
V37
V42
V32
V39
V20
V48
V16
V5
V4
V6
V13
V36
V27
V34
V47
V24
V25
V18
V19
V14
V15
V49
V9
V11
V10
V35
V41
V1
V23
V17
V3
V8
V33
V7
V12

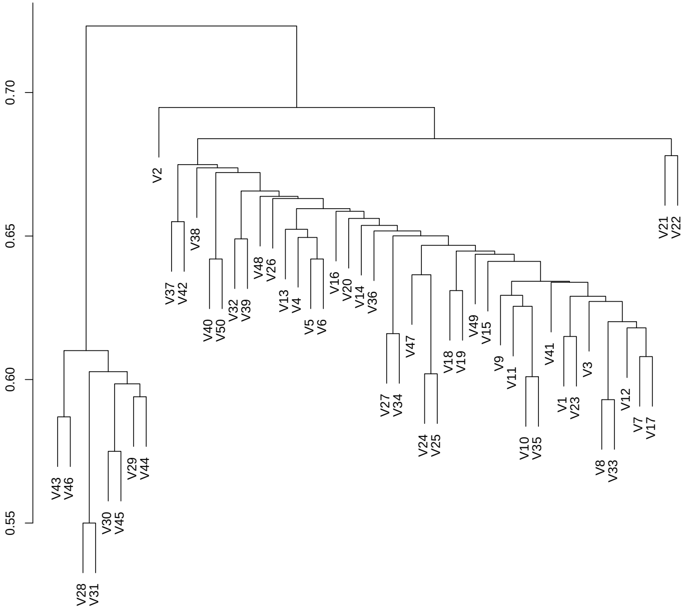
$\lambda = .4$


0.40
0.45
0.50
0.55
0.60
V28
V31
V30
V45
V43
V46
V29
V44
V21
V2
V22
V38
V40
V50
V37
V42
V26
V32
V39
V48
V16
V20
V13
V5
V4
V6
V36
V27
V34
V47
V24
V25
V18
V19
V14
V49
V15
V9
V11
V10
V35
V41
V1
V23
V3
V17
V8
V33
V7
V12

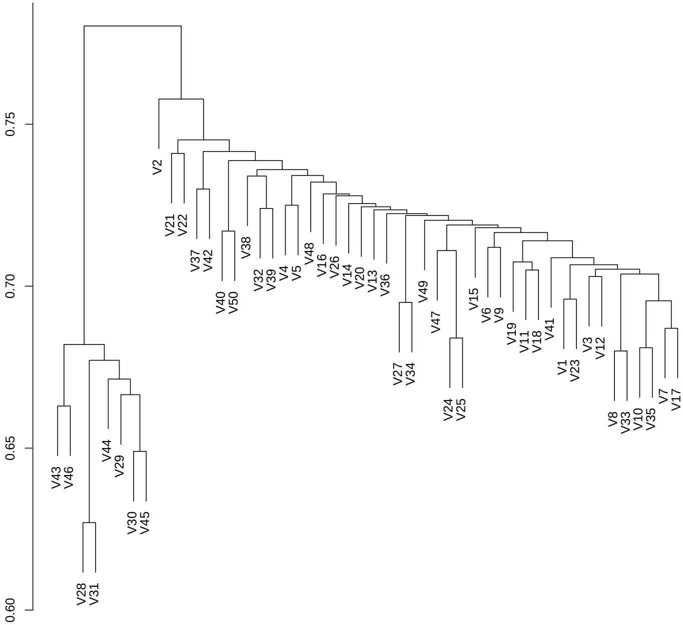
$\lambda = .5$


0.65
0.60
0.55
0.50
.45
V28
V31
V30
V45
V43
V46
V29
V44
V2
V38
V37
V42
V40
V50
V26
V32
V39
V48
V13
V4
V5
V6
V36
V27
V34
V14
V47
V24
V25
V18
V19
V49
V15
V9
V11
V10
V35
V41
V1
V23
V3
V17
V12
V7
V8
V33
V16
V20
V21
V22

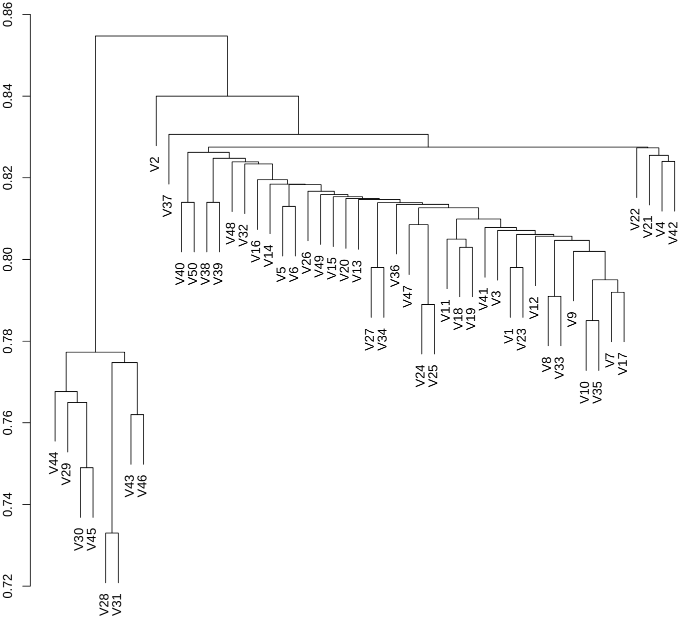
$\lambda = .6$


0.70
0.65
0.60
0.55
V43
V46
V28
V31
V30
V45
V29
V44
V2
V37
V42
V38
V40
V50
V32
V39
V48
V26
V13
V4
V5
V6
V16
V20
V14
V36
V27
V34
V47
V24
V25
V18
V19
V49
V15
V9
V11
V10
V35
V41
V1
V23
V3
V8
V33
V12
V7
V17
V21
V22

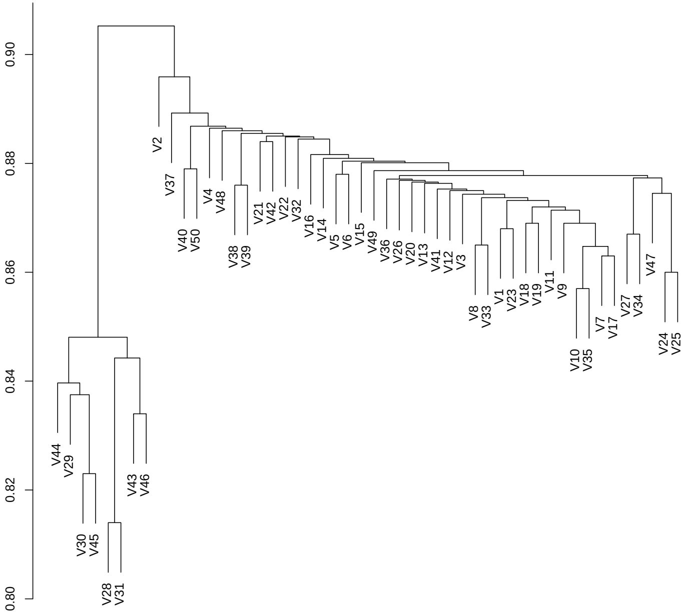
$\lambda = .7$


0.60
0.65
0.70
0.75
V43
V46
V28
V31
V44
V29
V30
V45
V2
V21
V22
V37
V42
V40
V50
V38
V32
V39
V4
V5
V48
V16
V26
V14
V20
V13
V36
V27
V34
V49
V47
V24
V25
V15
V6
V9
V19
V11
V18
V41
V1
V23
V3
V12
V8
V33
V10
V35
V7
V17

$\lambda = .8$

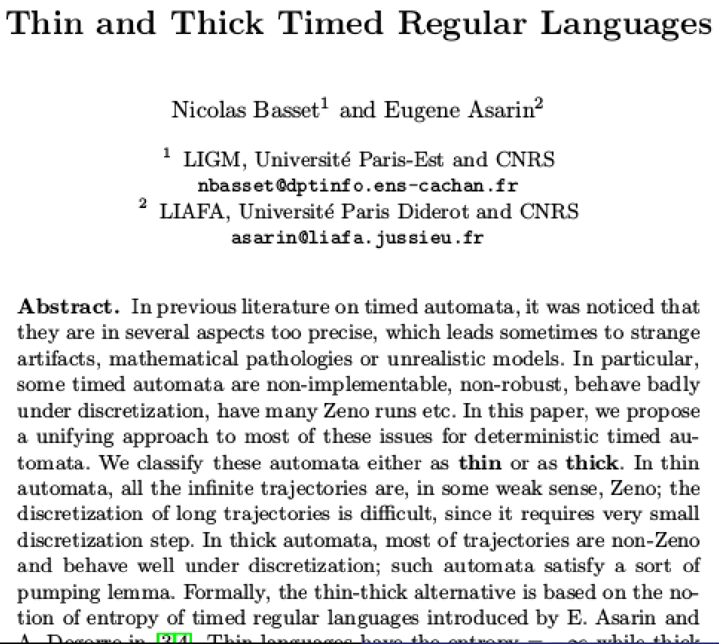

0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
V44
V29
V30
V45
V28
V31
V43
V46
V2
V37
V40
V50
V38
V39
V48
V32
V16
V14
V5
V6
V26
V49
V15
V20
V13
V27
V34
V36
V47
V24
V25
V11
V18
V19
V41
V3
V1
V23
V12
V8
V33
V9
V10
V35
V7
V17
V22
V21
V4
V42

$\lambda = .9$


0.90
0.88
0.86
0.84
0.82
0.80
V44
V29
V30
V45
V28
V31
V43
V46
V2
V37
V40
V50
V4
V48
V38
V39
V21
V42
V22
V32
V16
V14
V5
V6
V15
V49
V36
V26
V20
V13
V41
V12
V3
V8
V33
V1
V23
V18
V19
V11
V9
V10
V35
V7
V17
V27
V34
V47
V24
V25

$\lambda = .95$

- paper 8: Nicolas Basset and Eugene Asarin. Thin and thick timed regular languages. FORMATS, 2011
- paper 33: Eugene Asarin and Aldric Degorre. Volume and entropy of regular timed languages. HAL, 2009

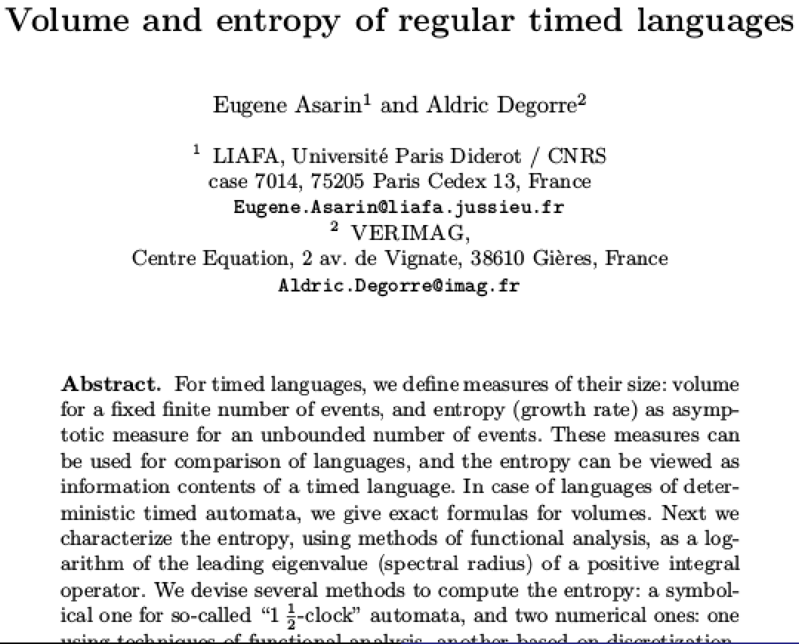


# Thin and Thick Timed Regular Languages

Nicolas Basset[1] and Eugene Asarin[2]

[1] LIGM, Université Paris-Est and CNRS
nbasset@dptinfo.ens-cachan.fr
[2] LIAFA, Université Paris Diderot and CNRS
asarin@liafa.jussieu.fr

**Abstract.** In previous literature on timed automata, it was noticed that they are in several aspects too precise, which leads sometimes to strange artifacts, mathematical pathologies or unrealistic models. In particular, some timed automata are non-implementable, non-robust, behave badly under discretization, have many Zeno runs etc. In this paper, we propose a unifying approach to most of these issues for deterministic timed automata. We classify these automata either as **thin** or as **thick**. In thin automata, all the infinite trajectories are, in some weak sense, Zeno; the discretization of long trajectories is difficult, since it requires very small discretization step. In thick automata, most of trajectories are non-Zeno and behave well under discretization; such automata satisfy a sort of pumping lemma. Formally, the thin-thick alternative is based on the notion of entropy of timed regular languages introduced by E. Asarin and A. Degorre in [3,4]. Thin languages have the entropy $= -\infty$ while thick



# Volume and entropy of regular timed languages

Eugene Asarin[1] and Aldric Degorre[2]

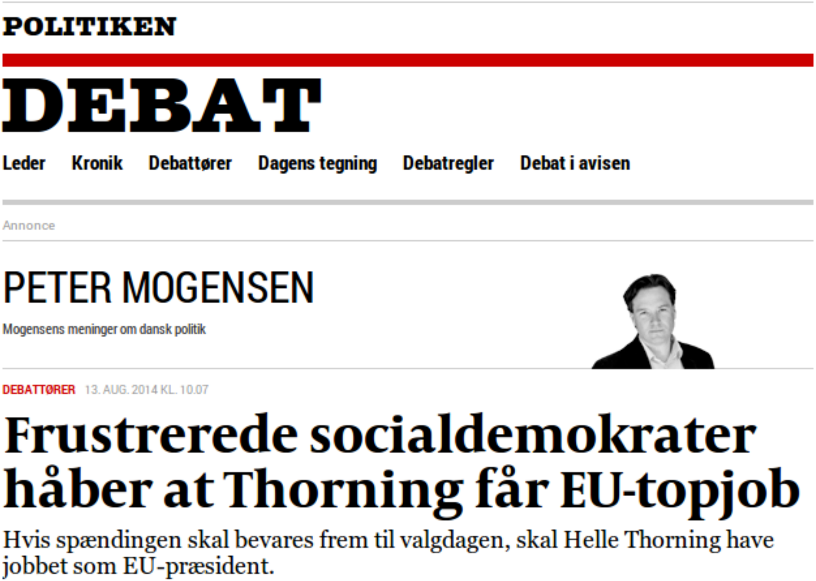
[1] LIAFA, Université Paris Diderot / CNRS
case 7014, 75205 Paris Cedex 13, France
Eugene.Asarin@liafa.jussieu.fr
[2] VERIMAG,
Centre Equation, 2 av. de Vignate, 38610 Gières, France
Aldric.Degorre@imag.fr

**Abstract.** For timed languages, we define measures of their size: volume for a fixed finite number of events, and entropy (growth rate) as asymptotic measure for an unbounded number of events. These measures can be used for comparison of languages, and the entropy can be viewed as information contents of a timed language. In case of languages of deterministic timed automata, we give exact formulas for volumes. Next we characterize the entropy, using methods of functional analysis, as a logarithm of the leading eigenvalue (spectral radius) of a positive integral operator. We devise several methods to compute the entropy: a symbolical one for so-called "1 $\frac{1}{2}$-clock" automata, and two numerical ones: one using techniques of functional analysis, another based on discretization

# Experiment 2

- 97 `arxiv` papers (search with author = “Smith”)
- 50 SciGen papers (modified SciGen to generate longer papers)
- 50 `automogensen` papers
  - simple 3-place Markov chain text generator
  - fed with the genuine corpus from Experiment 1 (length 247,000; 14,000 unique words)

- not so clear separation with average clustering
- `arxiv` papers have wildly varying distances; disturb
- clear separation with Ward clustering



POLITIKEN

DEBAT

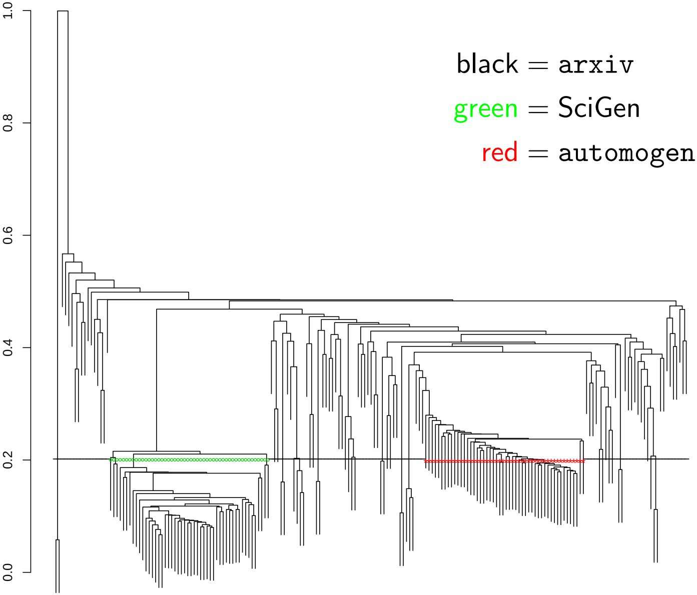
PETER MOGENSEN

Mogensens meninger om dansk politik

DEBATTØRER 13. AUG. 2014 KL. 10.07

# Frustrerede socialdemokrater håber at Thorning får EU-topjob

Hvis spændingen skal bevares frem til valgdagen, skal Helle Thorning have jobbet som EU-præsident.

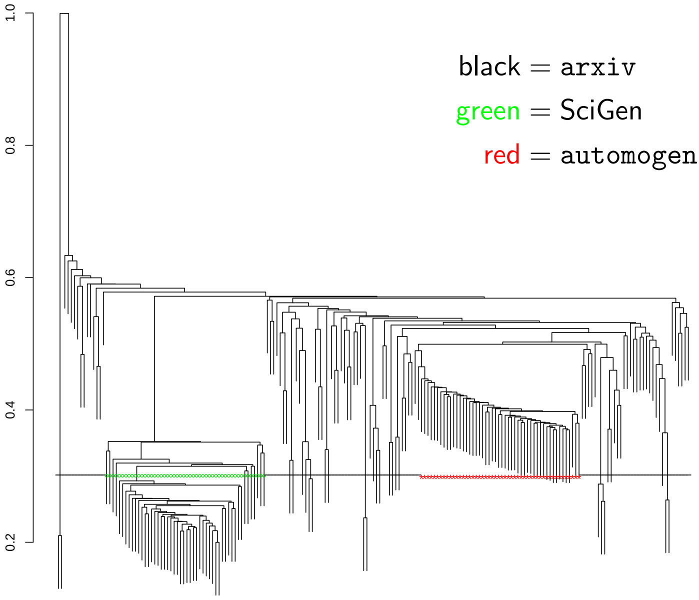


black = `arxiv`

green = SciGen

red = `automogensen`

$\lambda = 0$

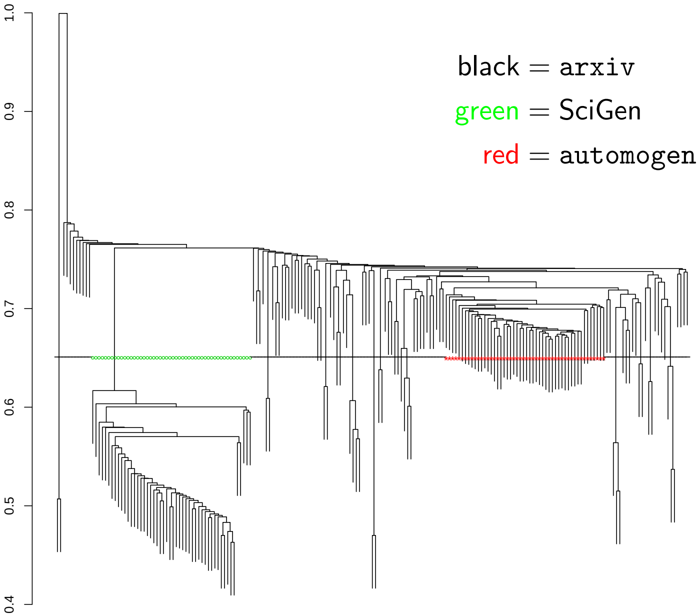


black = $\texttt{arxiv}$

green = SciGen

red = $\texttt{automogensen}$

$\lambda = .4$

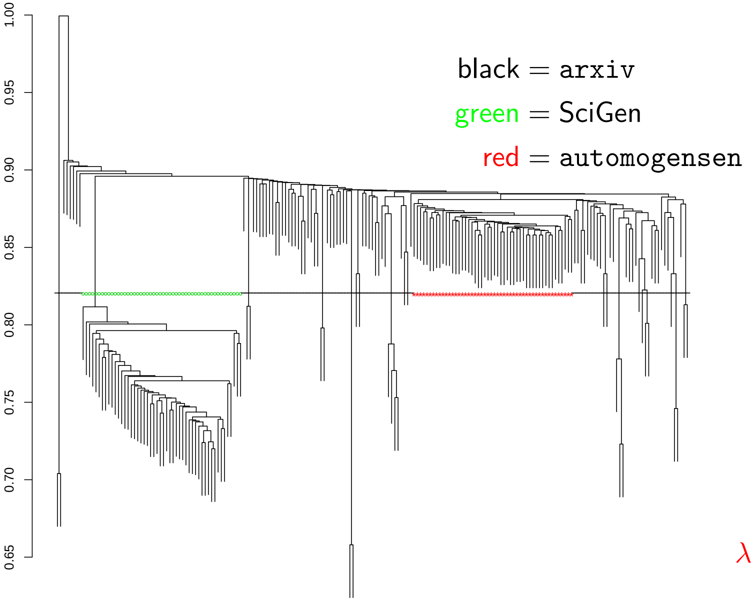


black = $\texttt{arxiv}$

green = SciGen

red = $\texttt{automogensen}$

$\lambda = .8$

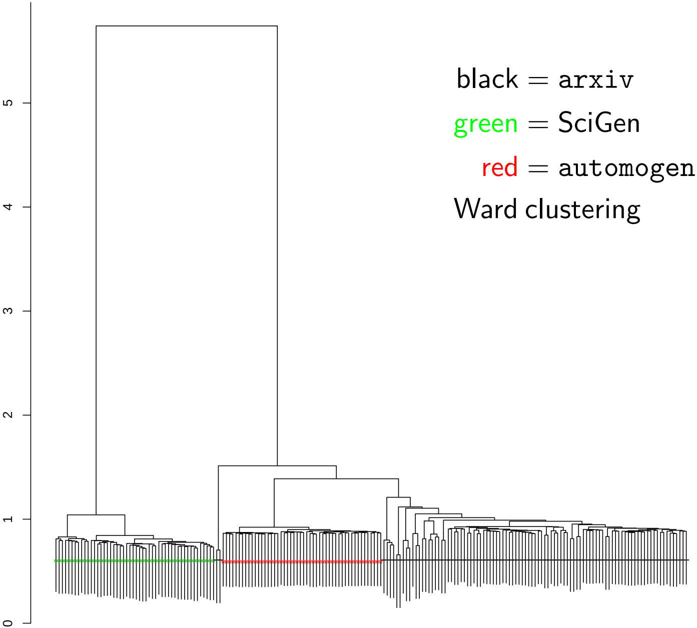


black = arxiv

green = SciGen

red = automogensen

$\lambda = .95$



black = `arxiv`

green = SciGen

red = `automogensen`

Ward clustering

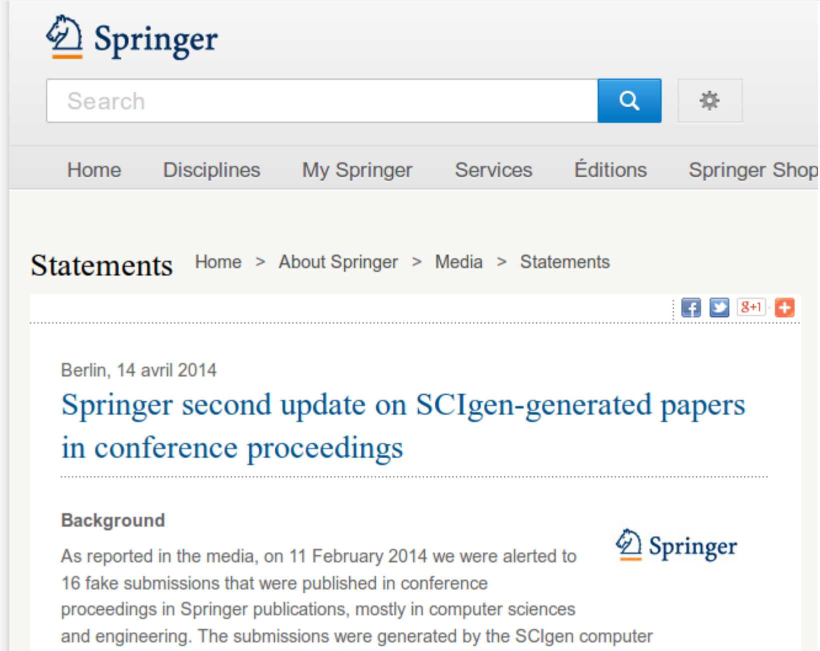
$\lambda = .95$

# Conclusion

- new intertextual distance which can measure differences in *n*-grams for all *n* simultaneously
- experiments show good separation of fake and genuine scientific papers
- more experiments needed, also on other types of papers
- need thorough comparison with state of the art

Future work:

- WordNet integration
- discounted edit distance (swaps and deletions)



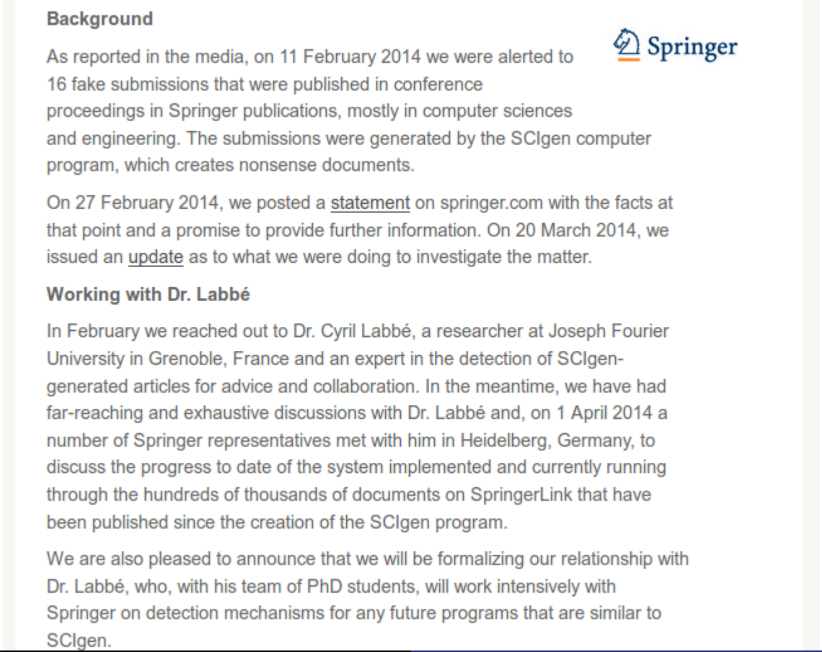
## Statements

Berlin, 14 avril 2014

## Springer second update on SCIgen-generated papers in conference proceedings

### Background

As reported in the media, on 11 February 2014 we were alerted to 16 fake submissions that were published in conference proceedings in Springer publications, mostly in computer sciences and engineering. The submissions were generated by the SCIgen computer



### Background

As reported in the media, on 11 February 2014 we were alerted to 16 fake submissions that were published in conference proceedings in Springer publications, mostly in computer sciences and engineering. The submissions were generated by the SCIgen computer program, which creates nonsense documents.

On 27 February 2014, we posted a statement on springer.com with the facts at that point and a promise to provide further information. On 20 March 2014, we issued an update as to what we were doing to investigate the matter.

### Working with Dr. Labbé

In February we reached out to Dr. Cyril Labbé, a researcher at Joseph Fourier University in Grenoble, France and an expert in the detection of SCIgen-generated articles for advice and collaboration. In the meantime, we have had far-reaching and exhaustive discussions with Dr. Labbé and, on 1 April 2014 a number of Springer representatives met with him in Heidelberg, Germany, to discuss the progress to date of the system implemented and currently running through the hundreds of thousands of documents on SpringerLink that have been published since the creation of the SCIgen program.

We are also pleased to announce that we will be formalizing our relationship with Dr. Labbé, who, with his team of PhD students, will work intensively with Springer on detection mechanisms for any future programs that are similar to SCIgen.