Computations and Applications of Minimal Absent Words

Carl Barton[1], Alice Heliou[2,3],
Laurent Mouchard[4] and Solon P. Pissis[5]

[1] The Blizard Institute, Barts and The London School of Medicine and Dentistry, Queen Mary University of London, UK
[2] Inria Saclay-Île de France, AMIB, Bâtiment Alan Turing, France
[3] Laboratoire d'Informatique de l'École Polytechnique (LIX), CNRS UMR 7161, France
[4] University of Rouen, LITIS EA 4108, TIBS, Rouen, France
[5] Department of Informatics, King's College London, London, UK

Outline

'Negative' information

Principle

Given a sequence of letters, we focus on words that don't occur.
Their absence may have a signification.

'Negative' information

Principle

Given a sequence of letters, we focus on words that don't occur. Their absence may have a signification.

Example

In a random sequence S, we expect that every word of size less than $\log_\sigma(|S|)$ occurs in S.

'Negative' information

Principle

Given a sequence of letters, we focus on words that don't occur.
Their absence may have a signification.

Example

In a random sequence S, we expect that every word of size less than $\log_\sigma(|S|)$ occurs in S.
The human genome contains around 3G nucleotides (A, C, G, T).
Yet some words of size 11, are absent $(11 < \log_4(3 * 10^9) = 15,7)$

'Negative' information

Application

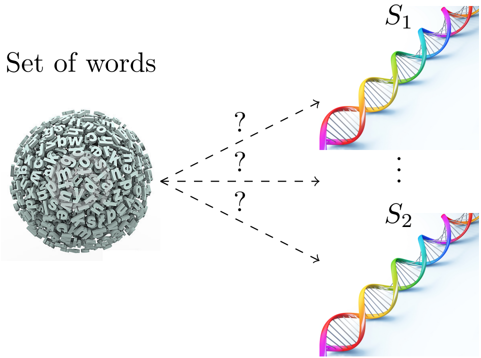
Three minimal sequences found in Ebola virus genomes and absent from human DNA, [Silva et al.], 2015

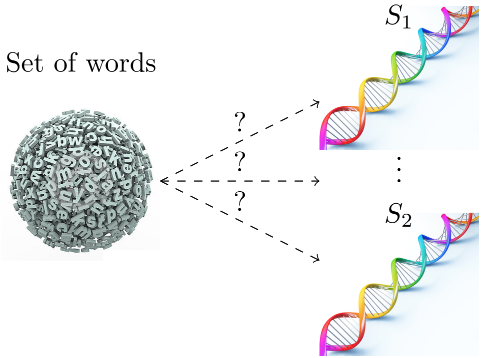
3 small sequences (TTTCGCCCGACT, TACGCCCTATCG, CCTACGCGCAAA) that appear in the Ebola genome as coding for proteins, are absent from the Human genome.

This was obtained by analyzing 99 virus and the Human genome reference GRC-38.

Unicity
$\longrightarrow$

Set of Absent Words


Set of words
?
?
?
S_1
S_2



Property

For each set of words $\mathcal{M}$ if there exists a sequence $\mathcal{S}$ such that $\mathcal{M}$ is its set of absent words, then $\mathcal{S}$ is unique.

Outline

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{\mathrm{A}}\,\overset{1}{\mathrm{A}}\,\overset{2}{\mathrm{C}}\,\overset{3}{\mathrm{A}}\,\overset{4}{\mathrm{C}}\,\overset{5}{\mathrm{A}}\,\overset{6}{\mathrm{C}}\,\overset{7}{\mathrm{C}}$$

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{\mathrm{A}}\,\overset{1}{\mathrm{A}}\,\overset{2}{\mathrm{C}}\,\overset{3}{\mathrm{A}}\,\overset{4}{\mathrm{C}}\,\overset{5}{\mathrm{A}}\,\overset{6}{\mathrm{C}}\,\overset{7}{\mathrm{C}}$$

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}$$

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}$$

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}$$

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}$$

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Definition : Minimal Absent Word

A minimal absent word of a sequence is an absent word whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

An upper bound on the number of minimal absent words is $\mathcal{O}(\sigma n)$.

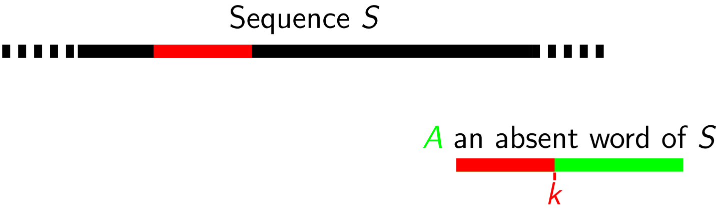
Crochemore et al. 1998, Mignosi et al. 2002

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}$$

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

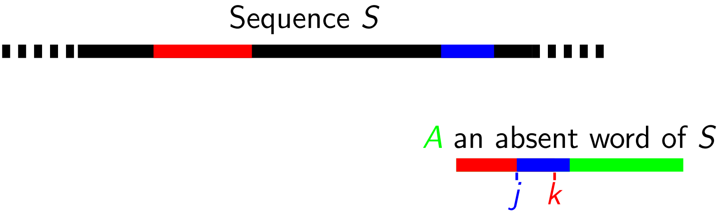
An absent word has a minimal absent word as factor

An absent word has a minimal absent word as factor



k, such that $A[0..k]$ occurs in S but not $A[0..k+1]$

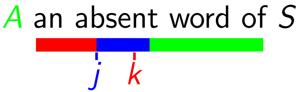
An absent word has a minimal absent word as factor



k, such that $A[0..k]$ occurs in S but not $A[0..k+1]$

j, such that $A[j..k+1]$ occurs S but not A[j-1..k+1]

An absent word has a minimal absent word as factor



k, such that $A[0..k]$ occurs in S but not $A[0..k+1]$

j, such that $A[j..k+1]$ occurs S but not A[j-1..k+1]

A[j-1..k+1] is a minimal absent word of S

because $A[j..k+1]$ and $A[j-1..k]$ occur in S.

Outline

Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Lemma

If *awb* is a minimal absent word of S, then there exist positions i and j such that (i, j, w) is a maximal repeated pair of S.

Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Lemma

If awb is a minimal absent word of S, then there exist positions i and j such that (i, j, w) is a maximal repeated pair of S.

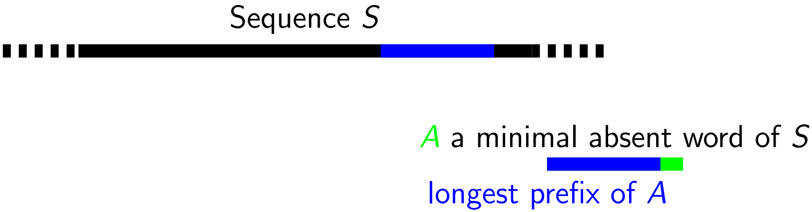
Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Lemma

If awb is a minimal absent word of S, then there exist positions i and j such that (i, j, w) is a maximal repeated pair of S.



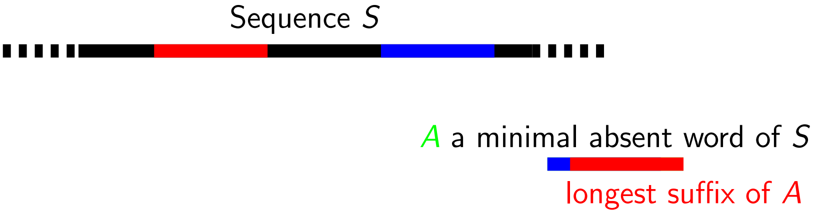
Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Lemma

If awb is a minimal absent word of S, then there exist positions i and j such that (i, j, w) is a maximal repeated pair of S.



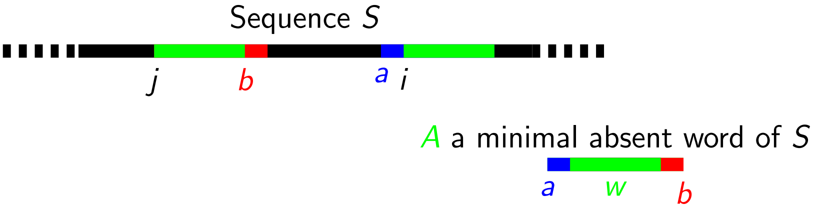
Definition: Maximal repeated pair

A maximal repeated pair in a S is a triple (i, j, w) such that:

- w occurs in S at positions i and j
- $S[i-1] \neq S[j-1]$
- $S[i+|w|] \neq S[j+|w|]$

Lemma

If awb is a minimal absent word of S, then there exist positions i and j such that (i, j, w) is a maximal repeated pair of S.



Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

$\Rightarrow$

pos	
8	#

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	**A**	**A**	**C**	**A**	**C**	**A**	**C**	**C**	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

⇒

pos									
8	#								
0	A	A	C	A	C	A	C	C	#

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	**A**	**C**	**A**	**C**	**A**	**C**	**C**	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

$\Rightarrow$

pos									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	**A**	**C**	**A**	**C**	**C**	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of *y*

⇒

pos									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
3	A	C	A	C	C	#			

Ordered suffixes of *S*

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

$\Rightarrow$

pos									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
3	A	C	A	C	C	#			
5	A	C	C	#					
7	C	#							
2	C	A	C	A	C	C	#		
4	C	A	C	C	#				
6	C	C	#						

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of right maximal repeated pairs.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

$\Rightarrow$

SA									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
3	A	C	A	C	C	#			
5	A	C	C	#					
7	C	#							
2	C	A	C	A	C	C	#		
4	C	A	C	C	#				
6	C	C	#						

Ordered suffixes of S

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of patterns.

$$S = \overset{0}{A}\overset{1}{A}\overset{2}{C}\overset{3}{A}\overset{4}{C}\overset{5}{A}\overset{6}{C}\overset{7}{C}\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of y

$\Rightarrow$

SA									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
3	A	C	A	C	C	#			
5	A	C	C	#					
7	C	#							
2	C	A	C	A	C	C	#		
4	C	A	C	C	#				
6	C	C	#						

Ordered suffixes of S

Linear-time computation of minimal absent words using suffix array. BMC Bioinformatics, 2014

Pre-computation

Construction :

- Suffix Array, linear time and space since 2003
- Longest Common Prefix table, linear time and space with the SA and the sequence as input

Computation

- Travel twice through those tables, in order to construct the set of letters that occurs just before each right-maximal repetition.
- Deduce the set of minimal absent words.

Linear-time computation of minimal absent words using suffix array. BMC Bioinformatics, 2014

Pre-computation

Construction :

- Suffix Array, linear time and space since 2003
- Longest Common Prefix table, linear time and space with the SA and the sequence as input

Computation

- Travel twice through those tables, in order to construct the set of letters that occurs just before each right-maximal repetition.
- Deduce the set of minimal absent words.

Perfomances

Computation for the whole human genome :
$\simeq$ 9000s with 130GB of RAM

Suffix Array by Manber& Myers in 1990

Table containing the starting position of the suffixes when they are in alphabetical order. It allows fast localisation of patterns.

$$S = \overset{0}{A}\,\overset{1}{A}\,\overset{2}{C}\,\overset{3}{A}\,\overset{4}{C}\,\overset{5}{A}\,\overset{6}{C}\,\overset{7}{C}\,\overset{8}{\#}$$

0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
2	C	A	C	A	C	C	#		
3	A	C	A	C	C	#			
4	C	A	C	C	#				
5	A	C	C	#					
6	C	C	#						
7	C	#							
8	#								

Suffixes of *y*

⇒

SA									
8	#								
0	A	A	C	A	C	A	C	C	#
1	A	C	A	C	A	C	C	#	
3	A	C	A	C	C	#			
5	A	C	C	#					
7	C	#							
2	C	A	C	A	C	C	#		
4	C	A	C	C	#				
6	C	C	#						

Ordered suffixes of *S*

Improvements for the computation of minimal absent words

Parallelising the Computation of Minimal Absent Words, PPAM 2105

- High scalabilty of the computation
- Computation for the whole human genome: $\simeq$ 5000s with 4 cores and 130GB of RAM

Improvements for the computation of minimal absent words

Parallelising the Computation of Minimal Absent Words, PPAM 2105

- High scalabilty of the computation
- Computation for the whole human genome: $\simeq$ 5000s with 4 cores and 130GB of RAM

Using external memory

The precomputation step, is also done in external memory, using recent implementations by Karkkainen et al.
Computation for the whole human genome:

- $\simeq$ 8000s with 8GB of RAM
- $\simeq$ 12000s with 1GB of RAM

Improvements for the computation of minimal absent words

Parallelising the Computation of Minimal Absent Words, PPAM 2105

- High scalabilty of the computation
- Computation for the whole human genome: $\simeq$ 5000s with 4 cores and 130GB of RAM

Using external memory

The precomputation step, is also done in external memory, using recent implementations by Karkkainen et al.
Computation for the whole human genome:

- $\simeq$ 8000s with 8GB of RAM
- $\simeq$ 12000s with 1GB of RAM

Implementations are available:
`https://github.com/solonas13/maw`

Computations of minimal absent words using different structures

References	Time for fixed size alphabet	Space	Structure
Crochemore et al., Information Processing Letters 1998 Automata and forbidden words	$\mathcal{O}(n)$	$\mathcal{O}(n)$	suffix automata
Belazzougui et al. Algorithms - ESA 2013 Versatile Succinct Representations of the Bidirectional Burrows-Wheeler Transform	$\mathcal{O}(n)$	$\mathcal{O}(n)$	compact bidirectional BWT
Ota et al. TCS 2014 Dynamic construction of an antidictionary with linear complexity	$\mathcal{O}(n)$	$\mathcal{O}(n)$	suffix tree, dynamic approach
Belazzougui et al. CPM 2015 Space-efficient detection of unusual words	randomized $\mathcal{O}(n)$	$\mathcal{O}(n)$	BWT & few additional structures

Outline

Applications

Computational Biology

- Define a distance based on the of MAWs $\rightarrow$ mesure the difference between 2 sequences (Chairungsee and Crochemore, 2012)
- Alignment free sequence comparison, linear time and space (Crochemore et al, 2016)

Computer Science

- Data Compression Using Antidictionaries (Crochemore et al., 2000, Fiala and Holub, 2008)

Perspective

- Compute the minimal absent words inside a sliding window.
  $\rightarrow$ Find the best match using a minimal absent words measure

Thank you

Thank you

Questions ?