Design of Worst-Case-Optimal Spaced Seeds

Jens Zentgraf, Sven Rahmann

WABI 2025

Motivation

k-mers

A k-mer is a substring of length k.

Motivation

k-mers

A *k*-mer is a substring of length *k*.

Alignment-free applications

- Taxonomic classification
- Contamination removal
- Gene expression
- Phylogenetic tree reconstruction
- Pangenomics
- ...

Motivation

k-mers

A k-mer is a substring of length k.

Alignment-free applications

- Taxonomic classification
- Contamination removal
- Gene expression
- Phylogenetic tree reconstruction
- Pangenomics
- ...

Advantages

- Fast information retrieval (hashing, sorting, ...)
- Subsampling
  - Minimizers
  - Open/Closed Syncmers
  - ...

Motivation

k-mers

A k-mer is a substring of length k.

Alignment-free applications

- Taxonomic classification
- Contamination removal
- Gene expression
- Phylogenetic tree reconstruction
- Pangenomics
- ...

Advantages

- Fast information retrieval (hashing, sorting, ...)
- Subsampling
  - Minimizers
  - Open/Closed Syncmers
  - ...

Disadvantages

- Intolerant against errors
- Reference bias

Disadvantages of k-mers

An error changes k **consecutive** k-mers in a read.

Disadvantages of k-mers

An error changes k **consecutive** k-mers in a read.

```
TACGAGCTAGCATCAGC
  CGAGCTA
   GAGCTAG
    AGCTAGC
     GCTAGCA
      CTAGCAT
       TAGCATC
        AGCATCA
```

Disadvantages of k-mers

An error changes k **consecutive** k-mers in a read.

Large k

- **High** specificity
- **Lower** number of k-mers
- **Few (No)** unchanged k-mers if an error occurs

```
TACGAGCTAGCATCAGC
TACGAGCTAGCATCA
 ACGAGCTAGCATCAG
  CGAGCTAGCATCAGC
```

Disadvantages of k-mers

An error changes k **consecutive** k-mers in a read.

```
TACGAGCTAGCATCAGC
TAC AGC AGC TCA
 ACG GCT GCA CAG
  CGA CTA CAT AGC
   GAG TAG ATC
```

Large k

- **High** specificity
- **Lower** number of k-mers
- **Few (No)** unchanged k-mers if an error occurs

Small k

- **Low** specificity
- **High** number of k-mers
- **Many** unchanged k-mers if an error occurs

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)

$k = 5, \ w = 7$

```
TACGAGCTAGCATCAGC
###__##
```

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)

$k = 5,\ w = 7$

TACGAGCTAGCATCAGC

###__##

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)
- Constraints for masks:
  - Symmetric
  - Start and end with #

$k = 5,\ w = 7$

```
TACGAGCTAGCATCAGC
###__##
```

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)
- Constraints for masks:
  - Symmetric
  - Start and end with #

$k = 5,\ w = 7$

```
TACGAGCTAGCATCAGC
#_###_#
```

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)
- Constraints for masks:
  - Symmetric
  - Start and end with #

$k = 5,\ w = 7$

```
TACGAGCTAGCATCAGC
T CGA C
 A GAG T
  C AGC A
   G GCT G
    A CTA C
     G TAG A
      C AGC T
       T GCA C
        A CAT A
         G ATC G
          CATCAGC
```

Spaced seeds (gapped k-mers)

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)
- Constraints for masks:
  - Symmetric
  - Start and end with #

$k = 5,\ w = 7$

```
TACGAGCTAGCATCAGC
T CGA C
 A GAG T
  C AGC A
   G GCT G
    A CTA C
     G TAG A
      C AGC T
       T GCA C
        A CAT A
         G ATC G
          CATCAGC
```

Spaced seeds (gapped k-mers)

$$k = 5, \; w = 7$$

An error changes k distributed k-mers.

Spaced seeds

- k significant positions (#)
- Window of size w
- $w - k$ ignored positions, gaps, ...(_)
- Mask, or tuple of offsets of #
  - Mask: #_#_#
  - Tuple: (0,2,4)
- Constraints for masks:
  - Symmetric
  - Start and end with #

Designing spaced seeds

Lossless setting ([NR08, Bri14, FCLST07])

- Given k, w, n (length of the sequence), c number of changes.
- Find a seed that guarantees at least one hit independently of the error positions.
- Decision problem already NP-complete.

Designing spaced seeds

Lossless setting ([NR08, Bri14, FCLST07])

- Given k, w, n (length of the sequence), c number of changes.
- Find a seed that guarantees at least one hit independently of the error positions.
- Decision problem already NP-complete.

Probabilistic setting ([BM08, CP10, No 17])

- No fixed number of changes.
- Compare two sequences that differ at each position with a probability.
- Compute probability distribution of the random number of hits (at least one hit).
- Computing sensitivity of a mask is NP-hard.

Goal

Goal

- Compute the following guarantees for a mask:

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

`CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG`

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

$k = 27$ $w = 27$ ###########################

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

$k = 27$ $w = 27$ ###########################

CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

$k = 27$ $w = 29$ `#########_#########_#########`

`CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG`

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.

$k = 27$ $w = 29$ #########_#########_#########

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
       #########_#########_#########         #########_#########_#########
                 #########_#########_#########                           #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.

$k = 27$ $w = 29$ #########_#########_#########

CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.

$k = 27$ $w = 29$ #########_#########_#########

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
#########_#########_#########     #########_#########_#########        #########_#########_#########
     #########_#########_#########          #########_#########_#########
               #########_#########_#########         #########_#########_#########
                        #########_#########_#########          #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ #########_#########_#########

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
#########_#########_#########     #########_#########_#########        #########_#########_#########
     #########_#########_#########          #########_#########_#########
               #########_#########_#########         #########_#########_#########
                        #########_#########_#########          #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ `#########_#########_#########`

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
                                  #########_#########_#########
     #########_#########_#########
                                                     #########_#########_#########
                                                               #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ #########_#########_#########

```
     ######### #########↓##################↓############################↓######### #########    (82)
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
                                  #########_#########_#########
     #########_#########_#########
                                                     #########_#########_#########
                                                               #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ `#########_#########_#########`

`CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG`

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ `#########_#########_#########`

```
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
                                      #########_#########_#########
                                       #########_#########_#########
                                        #########_#########_#########
                                         #########_#########_#########
                                          #########_#########_#########
                                                    #########_#########_#########
                                                              #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

$k = 27$ $w = 29$ `#########_#########_#########`

```
                            ↓        ↓#################################↓######### #########     (51)
CTATTTGTGTAATAGCCCATCGAGACGGCCCCTCATGGCGCTCAAGTGAACCTCTATTCGGGAGTCCGTCATATGCATGTCAGCGGTGACGGTATAGATG
                                      #########_#########_#########
                                       #########_#########_#########
                                        #########_#########_#########
                                         #########_#########_#########
                                          #########_#########_#########
                                                    #########_#########_#########
                                                              #########_#########_#########
```

Goal

Goal

- Compute the following guarantees for a mask:
  - How many errors c are tolerated, until we cannot find at least one k-mer.
  - For a given c, what is the minimal number of k-mers we can find.
  - For a given c, what is the minimal number of bases covered by k-mers.

Solution

We provide 3 ILPs to compute these values for a given mask.

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

$$k=9$$

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

`(0,1,2,4,5,6,8,9,10)`

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\overset{k=9}{\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}}$$

$$\texttt{(0,1,2,4,5,6,8,9,10)}$$

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}^{k=9}$$

(0,1,2,4,5,6,8,9,10)

n=28 CTATTTGTGTAATAGCCCATCGAGACGG

x 0000000000000000000000000000

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

k=9

$\underbrace{\texttt{###\_###\_###}}_{w=11}$

(0,1,2,4,5,6,8,9,10)

n=28 CTATTTGTGTAATAGCCCATCGAGACGG

x 0000000000001000000000000000

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

n=28 CTATTTGTGTAATAGCCCATCGAGACGG

x 1111111111111111111111111111

$$\overset{k=9}{\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}}$$

(0,1,2,4,5,6,8,9,10)

Number of Tolerated Errors

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\overbrace{\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}}^{k=9}$$

(0,1,2,4,5,6,8,9,10)

n=28 CTATTTGTGTAATAGCCCATCGAGACGG

x 0000000000001000000000000000

Number of Tolerated Errors

Minimize $\sum_{i \in [n]} x_i$,

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

k=9

$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$

(0,1,2,4,5,6,8,9,10)

n=28 `CTATTTGTGTAATAGCCCATCGAGACGG`

x `0000000000001000000000000000`

Number of Tolerated Errors

Minimize $\sum_{i \in [n]} x_i$,

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\underbrace{\texttt{###\_###\_###}}_{w=11}^{k=9}$$

(0,1,2,4,5,6,8,9,10)

```
        ###_###_###
n=28 CTATTTGTGTAATTAGCCCATCGAGACGG
   x 0000000000001000000000000000
```

Number of Tolerated Errors

$$\text{Minimize} \quad \sum_{i \in [n]} x_i\,,$$

$$\text{such that} \quad \sum_{j \in \kappa} x_{p+j} \geq 1 \quad p \in [n-w+1]\,.$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\underbrace{\overset{k=9}{\texttt{###\_###\_###}}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

```
       ###_###_###
n=28 CTATTTGTGTAATAGCCCATCGAGACGG
   x 0000000000001000000000000000
```

Number of Tolerated Errors

$$\text{Minimize} \quad \sum_{i \in [n]} x_i \,,$$

$$\text{such that} \quad \sum_{j \in \kappa} x_{p+j} \geq 1 \quad p \in [n-w+1] \,.$$

Tolerated Errors

$$c = \sum_{i \in [n]} x_i - 1$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\underbrace{\overset{k=9}{\texttt{###\_###\_###}}}_{w=11}$$

`(0,1,2,4,5,6,8,9,10)`

Minimal Number of Hits

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions

$$\overset{k=9}{\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}}$$

$\texttt{(0,1,2,4,5,6,8,9,10)}$

Minimal Number of Hits

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}^{k=9}$$

(0,1,2,4,5,6,8,9,10)

Minimal Number of Hits

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

n=28 `CTATTTGTGTAATAGCCCATCGAGACGG`

x 0000000000001000000000000000

$$k{=}9$$

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

$$(0,1,2,4,5,6,8,9,10)$$

Minimal Number of Hits

Variables

- k: weight of the mask
- w: window length
- κ: k-tuple of offsets
- n: sequence length
- c: **allowed changes**

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

```
      ###_###_###↓
n=28  CTATTTGTGTAATTAGCCCATCGAGACGG
x     0000000000001000000000000000
y     110001000100011111
```

$$k=9$$

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$

$$(0,1,2,4,5,6,8,9,10)$$

Minimal Number of Hits

Minimize $\sum_{p \in [n-w+1]} y_p$,

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

```
          ###_###_###↓
n=28 CTATTTGTGTAATAGCCCATCGAGACGG
   x 0000000000001000000000000000
   y 110001000100011111
```

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}^{k=9}$$

(0,1,2,4,5,6,8,9,10)

Minimal Number of Hits

$$\text{Minimize} \quad \sum_{p\in[n-w+1]} y_p \,,$$

$$\text{such that} \quad \sum_{i\in[n]} x_i = c \,,$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- **c: allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

```
          ###_###_###
n=28 CTATTTGTGTAATAGCCCATCGAGACGG
   x 0000000000001000000000000000
   y 110001000100011111
```

k=9

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal Number of Hits

Minimize $$\sum_{p\in[n-w+1]} y_p \,,$$

such that $$\sum_{i\in[n]} x_i = c\,,$$

$$y_p \geq 1 - \sum_{j\in\kappa} x_{p+j} \quad p \in [n-w+1]\,,$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

```
      ###_###_###↓
n=28 CTATTTGTGTAATAGCCCATCGAGACGG
   x 0000000000001000000000000000
   y 110001000100011111
```

$$\overset{k=9}{\underbrace{\texttt{###\_###\_###}}_{w=11}}$$

(0,1,2,4,5,6,8,9,10)

Minimal Number of Hits

$$\text{Minimize} \quad \sum_{p \in [n-w+1]} y_p \,,$$

$$\text{such that} \quad \sum_{i \in [n]} x_i = c \,,$$

$$y_p \geq 1 - \sum_{j \in \kappa} x_{p+j} \qquad p \in [n-w+1] \,,$$

$$y_p \leq 1 - x_{p+j} \qquad p \in [n-w+1], \; j \in \kappa \,.$$

```
     ###_###_###↓
n=28 CTATTTGTGTAATAGCCCATCGAGACGG
   x 0000000000001000000000000000
   y 1100010001000011111
```

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

$$k{=}9$$

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers

$$\underbrace{\overset{k=9}{\texttt{###\_###\_###}}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i \in [n]}$, indicators of covered positions

$$k{=}9$$
$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$
(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

n=28 CTATTTGTGTAATAGCCCATCGAGACGG

x 0000000000001000000000000000

y 110001000100011111

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i\in[n]}$, indicators of covered positions

$$\overset{k=9}{\underbrace{\texttt{###\_###\_###}}_{w=11}}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

```
      ###_###_###↓
n=28 CTATTTGTGTAATTAGCCCATCGAGACGG
   x 0000000000001000000000000000
   y 11000100010001111
   z 1111111111110111111111111111
```

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i \in [n]}$, indicators of covered positions

k=9

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

Minimize $\sum_{i \in [n]} z_i$,

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i \in [n]}$, indicators of covered positions

$$k{=}9$$
$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$
$$\texttt{(0,1,2,4,5,6,8,9,10)}$$

Minimal number of Covered Bases

$$\text{Minimize} \quad \sum_{i \in [n]} z_i \,,$$

$$\text{such that} \quad \sum_{i \in [n]} x_i = c \,,$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i \in [n]}$, indicators of covered positions

$$\overset{k=9}{\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

$$\text{Minimize} \quad \sum_{i\in[n]} z_i\,,$$

$$\text{such that} \quad \sum_{i\in[n]} x_i = c\,,$$

$$y_p \geq 1 - \sum_{j\in\kappa} x_{p+j} \quad p \in [n-w+1]\,,$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i\in[n]}$, indicators of covered positions

k=9

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

$$\text{Minimize} \quad \sum_{i\in[n]} z_i\,,$$

$$\text{such that} \quad \sum_{i\in[n]} x_i = c\,,$$

$$y_p \geq 1 - \sum_{j\in\kappa} x_{p+j} \qquad p \in [n-w+1]\,,$$

$$y_p \leq 1 - x_{p+j} \qquad p \in [n-w+1]\,, \; j \in \kappa\,,$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i\in[n]}$, indicators of covered positions

k=9

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Minimal number of Covered Bases

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{i \in [n]} z_i \,, \\
\text{such that} \quad & \sum_{i \in [n]} x_i = c \,, \\
& y_p \geq 1 - \sum_{j \in \kappa} x_{p+j} && p \in [n-w+1] \,, \\
& y_p \leq 1 - x_{p+j} && p \in [n-w+1] \,, \\
& && j \in \kappa \,, \\
& z_{p+j} \geq y_p && p \in [n-w+1] \,, \\
& && j \in \kappa \,,
\end{aligned}
$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i \in [n]}$, indicators of changed positions
- $y = (y_p)_{p \in [n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i \in [n]}$, indicators of covered positions

$$k{=}9$$

$$\underbrace{\texttt{###\_###\_###}}_{w=11}$$

$$\texttt{(0,1,2,4,5,6,8,9,10)}$$

Minimal number of Covered Bases

$$\begin{aligned}
\text{Minimize} \quad & \sum_{i\in[n]} z_i\,, \\
\text{such that} \quad & \sum_{i\in[n]} x_i = c\,, \\
& y_p \geq 1 - \sum_{j\in\kappa} x_{p+j} && p \in [n-w+1]\,, \\
& y_p \leq 1 - x_{p+j} && p \in [n-w+1]\,, \\
& && j \in \kappa\,, \\
& z_{p+j} \geq y_p && p \in [n-w+1]\,, \\
& && j \in \kappa\,, \\
& z_i \leq \sum_{j\in\kappa,\ i-j\geq 0} y_{i-j} && i \in [n]\,.
\end{aligned}$$

Variables

- k: weight of the mask
- n: sequence length
- w: window length
- c: **allowed changes**
- κ: k-tuple of offsets

We use the following **binary** variables:

- $x = (x_i)_{i\in[n]}$, indicators of changed positions
- $y = (y_p)_{p\in[n-w+1]}$, indicators of (starting positions of) unchanged κ-mers
- $z = (z_i)_{i\in[n]}$, indicators of covered positions

k=9

$$\underbrace{\texttt{\#\#\#\_\#\#\#\_\#\#\#}}_{w=11}$$

(0,1,2,4,5,6,8,9,10)

Evaluation

Read mapper using spaced seeds

- Index which contains the position of each k-mer in the reference.
- Query all k-mers in a read.
- Subtract relative position in the read to get a start position.
- Combine near positions to one.

Evaluation

Read mapper using spaced seeds

- Index which contains the position of each k-mer in the reference.
- Query all k-mers in a read.
- Subtract relative position in the read to get a start position.
- Combine near positions to one.

Challenge data set

- Restrict to area without repeats in the genome.
- All k-mers in these areas occur only once.
- Also all k-mers do not have a Hamming-distance one neighbor anywhere in the genome.

Evaluation

Read mapper using spaced seeds

- Index which contains the position of each k-mer in the reference.
- Query all k-mers in a read.
- Subtract relative position in the read to get a start position.
- Combine near positions to one.

Challenge data set

- Restrict to area without repeats in the genome.
- All k-mers in these areas occur only once.
- Also all k-mers do not have a Hamming-distance one neighbor anywhere in the genome.
- Create reads of length 100.
- Insert errors in equidistant positions.
- Add a random offset uniformly chosen from $\{-3, \ldots, 0, \ldots, 3\}$

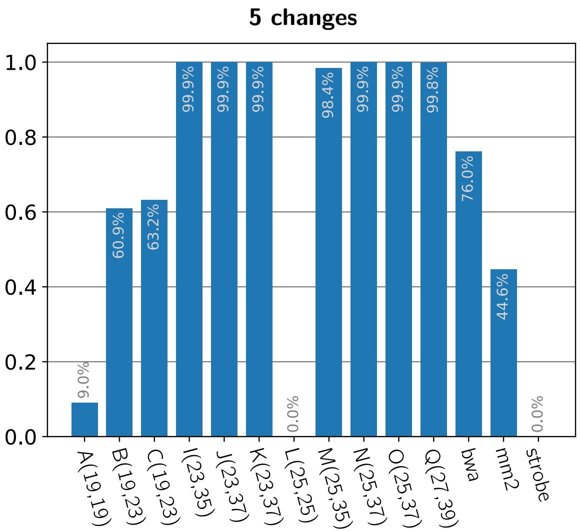
Evaluation

	(k, w) shape	mask	$c = 3$ MH	$c = 3$ MC	$c = 4$ MH	$c = 4$ MC	$c = 5$ MH	$c = 5$ MC
A	$(19, 19)$	###################	25	43	6	24	**0**	**0**
B	$(19, 23)$	####_####_###_####_####	21	68	11	48	**6**	42
C	$(19, 23)$	####_###_#####_###_####	21	68	11	48	**6**	42
I	$(23, 35)$	###_####__#_###__#__###_#__####_###	12	59	5	48	**2**	**34**
J	$(23, 37)$	#####__#_#_#_#__#####__#_#_#_#__#####	11	55	4	45	**2**	**34**
K	$(23, 37)$	###_###__#_###_#__#__#_###_#__###_###	12	58	**6**	47	**2**	**34**
L	$(25, 25)$	#########################	1	25	**0**	**0**	**0**	**0**
M	$(25, 35)$	#######__###__#_###_#__###__#######	9	55	**4**	42	**0**	**0**
N	$(25, 37)$	#####_#_#_##_##__###__##_##_#_#_#####	9	53	**4**	45	**0**	**0**
O	$(25, 37)$	#####_#_#__####__###__####__#_#_#####	8	54	2	**47**	**0**	**0**
Q	$(27, 39)$	##_####_###__#_###_#_###_#__###_####_##	7	52	**2**	**42**	**0**	**0**

Evaluation

Benchmark setup

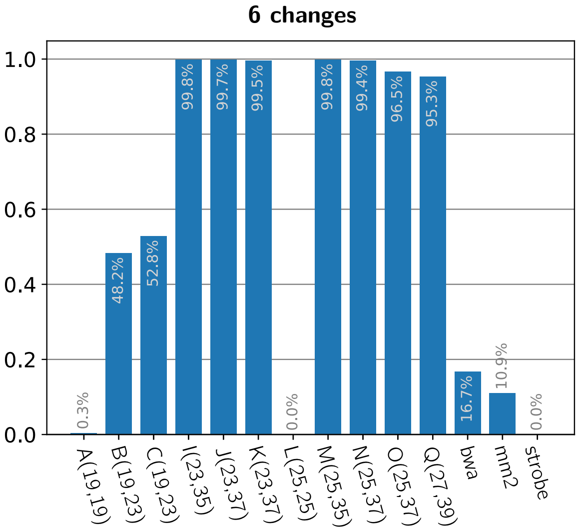
- AMD Ryzen 9 5950X 16-core CPU
- 128 GB RAM
- Gurobi (10.0.3)
- bwa-mem2 (v2.2.1)
- minimap2 (v2.26)
- strobealign (v0.11.0)



Evaluation

Benchmark setup

- AMD Ryzen 9 5950X 16-core CPU
- 128 GB RAM
- Gurobi (10.0.3)
- bwa-mem2 (v2.2.1)
- minimap2 (v2.26)
- strobealign (v0.11.0)



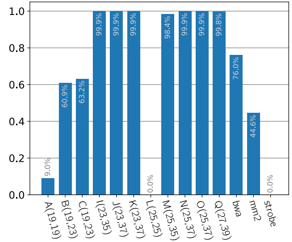
Summary

- Solved ILPs for
  - $n = 100$
  - $c \in \{3, 4, 5, 6, 7\}$
  - $k \in \{15, \ldots, 31\}$
  - $w \in \{15, \ldots, 47\}$
- Running time for one mask is small
- **Good** masks are rare

zentgraf@cs.uni-saarland.de
sven.rahmann@uni-saarland.de

https://doi.org/10.5281/zenodo.15690315

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{i \in [n]} z_i \,, \\
\text{such that} \quad & \sum_{i \in [n]} x_i = c \,, \\
& y_p \geq 1 - \sum_{j \in \kappa} x_{p+j} && p \in [n - w + 1] \,, \\
& y_p \leq 1 - x_{p+j} && p \in [n - w + 1] \,, \\
& && j \in \kappa \,, \\
& z_{p+j} \geq y_p && p \in [n - w + 1] \,, \\
& && j \in \kappa \,, \\
& z_i \leq \sum_{j \in \kappa,\ i-j \geq 0} y_{i-j} && i \in [n] \,.
\end{aligned}
$$



References I

Gary Benson and Denise Y. F. Mak.
Exact distribution of a spaced seed statistic for DNA homology detection.
In Amihood Amir, Andrew Turpin, and Alistair Moffat, editors, *String Processing and Information Retrieval, 15th International Symposium, SPIRE 2008, Melbourne, Australia, November 10-12, 2008. Proceedings*, volume 5280 of *Lecture Notes in Computer Science*, pages 282–293. Springer, 2008.

Karel Brinda.
Languages of lossless seeds.
In Zoltán Ésik and Zoltán Fülöp, editors, *Proceedings 14th International Conference on Automata and Formal Languages, AFL 2014, Szeged, Hungary, May 27-29, 2014*, volume 151 of *EPTCS*, pages 139–150, 2014.

W. H. Chung and S. B. Park.
Hit integration for identifying optimal spaced seeds.
BMC Bioinformatics, 11 Suppl 1(Suppl 1):S37, Jan 2010.

Martin Farach-Colton, Gad M. Landau, Süleyman Cenk Sahinalp, and Dekel Tsur.
Optimal spaced seeds for faster approximate string matching.
J. Comput. Syst. Sci., 73(7):1035–1044, 2007.

References II

L. Noé.
Best hits of 11110110111: model-free selection and parameter-free sensitivity calculation of spaced seeds.
Algorithms Mol Biol, 12:1, 2017.

François Nicolas and Eric Rivals.
Hardness of optimal spaced seed design.
J. Comput. Syst. Sci., 74(5):831–849, 2008.