Covering Problems for Partial Words and for Indeterminate Strings

Maxime Crochemore[1,2] Costas S. Iliopoulos[1,3]
Tomasz Kociumaka[4] Jakub Radoszewski[4]
Wojciech Rytter[4] Tomasz Waleń[4]

[1]King's College London, UK
[2]Université Paris-Est, France
[3]University of Western Australia
[4]University of Warsaw, Poland

ISAAC 2014
Jeonju, South Korea; December 15, 2014

Partial words

$\texttt{b b} \diamondsuit \texttt{ a b b} \diamondsuit \texttt{ a b b} \diamondsuit \texttt{ b a b b b} \diamondsuit \diamondsuit$

- $\diamondsuit$ — *don't care* symbol (*hole*)
- $\diamondsuit \approx c$ for every $c \in \Sigma \cup \{\diamondsuit\}$

Partial words

$$
\begin{array}{cccccccccccccccccc}
\mathtt{b} & \mathtt{b} & \Diamond & \mathtt{a} & \Diamond \\
 & & \wr\wr \\
\mathtt{b} & \mathtt{b} & \Diamond & \mathtt{a} & \mathtt{b} & \mathtt{b} & \Diamond & \mathtt{a} & \mathtt{b} & \mathtt{b} & \Diamond & \mathtt{b} & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{b} & \Diamond & \Diamond
\end{array}
$$

- $\Diamond$ — *don't care* symbol (*hole*)
- $\Diamond \approx c$ for every $c \in \Sigma \cup \{\Diamond\}$

Partial words

$$
\begin{array}{cccccccccccccccccc}
 & b & b & \lozenge & a & \lozenge & & & & & & & & & & & & \\
 & & & \not\approx & & & & & & & & & & & & & & \\
b & b & \lozenge & a & b & b & \lozenge & a & b & b & \lozenge & b & a & b & b & b & \lozenge & \lozenge
\end{array}
$$

- $\lozenge$ — *don't care* symbol (*hole*)
- $\lozenge \approx c$ for every $c \in \Sigma \cup \{\lozenge\}$

Indeterminate strings

a	b	c	d	a	c	a	b	c	a	b	a	c	a	b	a	b	c
	c			b		d			c		c	d		c			d
									d								

- Symbols — non-empty subsets of Σ

Indeterminate strings

a	b	c	d	a	c	a	b	c	a	b	a	c	a	b	a	b	c
	c			b		d			c		c	d		c			d
									d								

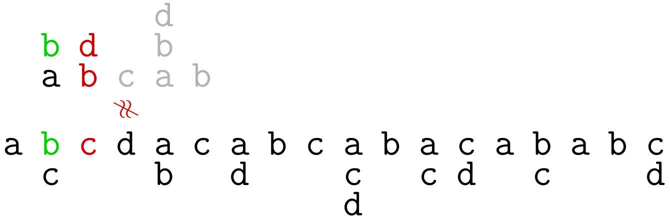
- Symbols — non-empty subsets of Σ
- Non-solid symbols — subsets of size at least 2

Indeterminate strings

			d														
b	d		b														
a	b	c	a	b													
		$\approx$															
a	b	c	d	a	c	a	b	c	a	b	a	c	a	b	a	b	c
	c			b		d			c		c	d		c			d
									d								

- Symbols — non-empty subsets of Σ
- Non-solid symbols — subsets of size at least 2
- $A \approx B$ if $A \cap B \neq \emptyset$

Indeterminate strings



- Symbols — non-empty subsets of Σ
- Non-solid symbols — subsets of size at least 2
- $A \approx B$ if $A \cap B \neq \emptyset$

Covering a solid string

Definition (Apostolico, Farach, Iliopoulos; 1991)

A string S is a *cover* of a string T if each position in T lies within an occurrence of S.

```
                                a a b a a
                           a a b a a
                   a a b a a
          a a b a a
a a b a a

a a b a a a b a a b a a a b a a a a b a a
```

Covering a solid string

Definition (Apostolico, Farach, Iliopoulos; 1991)

A string S is a *cover* of a string T if each position in T lies within an occurrence of S.

```
                          a a b a a
                     a a b a a
                 a a b a a
              a a b a a
          a a b a a
          a a b a a a b a a b a a a b a a a a b a a
```

- $S =$ aabaa is a cover.

Covering a solid string

Definition (Apostolico, Farach, Iliopoulos; 1991)

A string S is a *cover* of a string T if each position in T lies within an occurrence of S.

```
                                a a b a
                      a a b a
              a a b a
        a a b a
a a b a

a a b a a a b a a b a a a b a a a a b a a
```

Covering a solid string

Definition (Apostolico, Farach, Iliopoulos; 1991)

A string S is a *cover* of a string T if each position in T lies within an occurrence of S.

```
                                a a b a
                        a a b a
                a a b a
          a a b a
    a a b a
a a b a a a b a a b a a a b a a a a b a a
```

- $S =$ aaba is not a cover.

Results for solid strings

Linear-time algorithms for natural problems:

1991; Apostolico et al. shortest cover

1992; Breslauer shortest cover (online)

1994; Moore & Smyth all covers

2002; Li & Smyth all covers (online)

Results for solid strings

Linear-time algorithms for natural problems:

1991; Apostolico et al. shortest cover

1992; Breslauer shortest cover (online)

1994; Moore & Smyth all covers

2002; Li & Smyth all covers (online)

Numerous generalizations:

1992; Iliopoulos & Smyth k-covers

1993; Iliopoulos et al. seeds

2002; Sim et al. approximate covers

2012; Flouri et al. enhanced covers

2013; K. et al. partial covers

Covering partial words and indeterminate strings

Definition

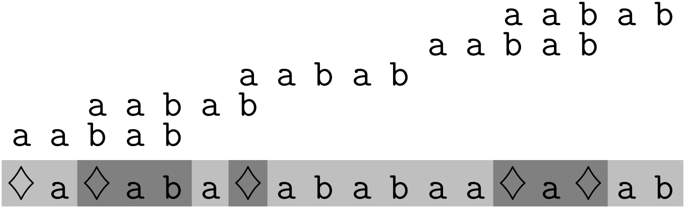
A *solid* string S is a cover of an *indeterminate* string T if each position in T lies within an occurrence of S.

```
                          a a b a b
                      a a b a b
            a a b a b
    a a b a b
a a b a b
◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b
```

Covering partial words and indeterminate strings

Definition

A *solid* string S is a cover of an *indeterminate* string T if each position in T lies within an occurrence of S.



- $S =$ aabab is a cover of T.

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

Our results:

- NP-hardness, already for binary partial words,

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

Our results:

- NP-hardness, already for binary partial words,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

Our results:

- NP-hardness, already for binary partial words,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \le n$)

Our results:

- NP-hardness, already for binary partial words,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

In this talk:

- NP-hardness, already for binary partial words,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Results for partial words and indeterminate strings

Problem

Find a shortest cover of a given indeterminate string T.

- n length of T
- k number of non-solid symbols in T
- σ alphabet size ($\sigma \leq n$)

In this talk:

- NP-hardness, already for binary partial words,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(\cancel{nk^4} n^{\mathcal{O}(1)} + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b b a b b ◇ a b b b b a b b b ◇ ◇
```

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b b a b b ◇ a b b b b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
            b b b a b             b b b a b
b b b a b             b b b a b
```

b b b a b b $\diamondsuit$ a b b b b a b b b $\diamondsuit$ $\diamondsuit$

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b           b b b a b
b b b a b           b b b a b
b b b a b b ◇ a b b b b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

b b b a b   b b b a b

b b b a b   b b b a b

b b $\diamondsuit$ a b b $\diamondsuit$ a b b b b a b b b $\diamondsuit$ $\diamondsuit$

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b ◇ a b b ◇ a b b b b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S, T)$. Succeed if $T[i..i + \ell - 1]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b ◇ a b b ◇ a b b b b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

b b b a b   b b b a b

b b b a b   b b b a b

b b $\diamondsuit$ a b b $\diamondsuit$ a b b $\diamondsuit$ b a b b b $\diamondsuit$ $\diamondsuit$

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S, T)$. Succeed if $T[i..i + \ell - 1]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
            b b b a b                 b b b a b
    b b b a b                 b b b a b
    b b ◇ a b b ◇ a b b ◇ b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1]$ is solid.
3. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1] \cap T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

b b b a b   b b b a b

b $\Diamond$ b a b   b b $\Diamond$ a b

b b $\Diamond$ a b b $\Diamond$ a b b $\Diamond$ b a b b b $\Diamond$ $\Diamond$

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1]$ is solid.
3. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1] \cap T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b ◇ a b b ◇ a b b ◇ b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S, T)$. Succeed if $T[i..i+\ell-1]$ is solid.
3. Guess $\ell = |S|$ and $i \in Occ(S, T)$. Succeed if $T[i..i+\ell-1] \cap T[1..\ell]$ is solid.

How to guess a cover? (in typical cases)

- Checking whether a solid string S is a cover is easy.
- We can afford verifying polynomially many candidates.

```
        b b b a b         b b b a b
b b b a b         b b b a b
b b ◇ a b b ◇ a b b ◇ b a b b b ◇ ◇
```

Attempts:

1. Guess $\ell = |S|$. Succeed if $T[1..\ell]$ is solid.
2. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1]$ is solid.
3. Guess $\ell = |S|$ and $i \in Occ(S,T)$. Succeed if $T[i..i+\ell-1] \cap T[1..\ell]$ is solid.

Which covers could we miss?

```
    a a b a b         a a b a b
a a b a b   a a b a b     a a b a b
◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b
```

Which covers could we miss?

```
    ◇ a ◇ a b         a a b a b
◇ a b a ◇   a a b a b     a a b a b
◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b
```

Which covers could we miss?

```
        a a b a b               a a b a b
◇ a b a b     ◇ a ◇ a b               a a b a b
◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b
```

Which covers could we miss?

a a b a b ◇ a ◇ a b

a a ◇ a ◇ a a b a b a a b a b

◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b

Which covers could we miss?

```
    a a b a b         a a b a b
◇ a ◇ a b   a a b a b   ◇ a ◇ a b
◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b
```

Which covers could we miss?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

- If S was not found, then for each $i \in Occ(S,T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$.

Which covers could we miss?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

- If S was not found, then for each $i \in Occ(S,T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1

Which covers could we miss?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3

Which covers could we miss?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1 + j] = T[i + j] = \diamondsuit$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5

Which covers could we miss?

$$\begin{array}{cccccccccccccccccc} & & & & & & \diamondsuit & a & \diamondsuit & a & b & a & \diamondsuit & a & b & a & b & a \\ \diamondsuit & a & \diamondsuit & a & b & a & \diamondsuit & a & b & a & b & a & a & \diamondsuit & a & \diamondsuit & a & b \end{array}$$

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1 + j] = T[i + j] = \diamondsuit$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7

Which covers could we miss?

$$\diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \mathtt{b}$$

$$\diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}$$

- If S was not found, then for each $i \in Occ(S,T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8

Which covers could we miss?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$. Moreover, $T[1..j] \approx T[i..i+j-1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8

Which covers could we miss?

$$\begin{array}{cccccccccccccccccc} & & & & & & & & & \diamondsuit & \mathtt{a} & \diamondsuit & \mathtt{a} & \mathtt{b} & \mathtt{a} & \diamondsuit & \mathtt{a} & \mathtt{b} \\ \diamondsuit & \mathtt{a} & \diamondsuit & \mathtt{a} & \mathtt{b} & \mathtt{a} & \diamondsuit & \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{a} & \diamondsuit & \mathtt{a} & \diamondsuit & \mathtt{a} & \mathtt{b} \end{array}$$

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1 + j] = T[i + j] = \diamondsuit$. Moreover, $T[1..j] \approx T[i..i + j - 1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8, 10

Which covers could we miss?

$$\diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \diamondsuit$$

$$\diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \mathtt{b}\ \mathtt{a}\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \diamondsuit\ \mathtt{a}\ \mathtt{b}$$

- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$. Moreover, $T[1..j] \approx T[i..i+j-1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8, 10, 12

Which covers could we miss?

$$\begin{array}{cccccccccccccccccc} & & & & & & & & & & & & \diamondsuit & \texttt{a} & \diamondsuit & \texttt{a} & \texttt{b} \\ \diamondsuit & \texttt{a} & \diamondsuit & \texttt{a} & \texttt{b} & \texttt{a} & \diamondsuit & \texttt{a} & \texttt{b} & \texttt{a} & \texttt{b} & \texttt{a} & \texttt{a} & \diamondsuit & \texttt{a} & \diamondsuit & \texttt{a} & \texttt{b} \end{array}$$

- If S was not found, then for each $i \in Occ(S,T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$. Moreover, $T[1..j] \approx T[i..i+j-1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8, 10, 12, 14

Which covers could we miss?

$$\diamondsuit\ \texttt{a}\ \texttt{a}\ \diamondsuit$$

$$\diamondsuit\ \texttt{a}\ \diamondsuit\ \texttt{a}\ \texttt{b}\ \texttt{a}\ \diamondsuit\ \texttt{a}\ \texttt{b}\ \texttt{a}\ \texttt{b}\ \texttt{a}\ \texttt{a}\ \diamondsuit\ \texttt{a}\ \diamondsuit\ \texttt{a}\ \texttt{b}$$

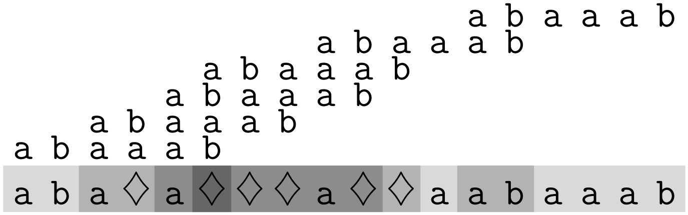
- If S was not found, then for each $i \in Occ(S, T)$ there exists j such that $T[1+j] = T[i+j] = \diamondsuit$. Moreover, $T[1..j] \approx T[i..i+j-1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8, 10, 12, 14, and 16.

Which covers could we miss?

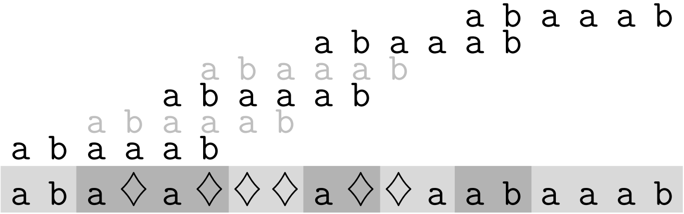
$\Diamond$ a $\Diamond$ a b a $\Diamond$ a b a b a a $\Diamond$ a $\Diamond$ a b

- If S was not found, then for each $i \in Occ(S,T)$ there exists j such that $T[1+j] = T[i+j] = \Diamond$.
  Moreover, $T[1..j] \approx T[i..i+j-1]$.
- Here *ambiguous* positions ($\mathcal{A}$) are: 1, 3, 5, 7, 8, 10, 12, 14, and 16.
  In general, there are at most $1 + \frac{k(k-1)}{2} = \mathcal{O}(k^2)$ ambiguous positions.

Covering set of a cover



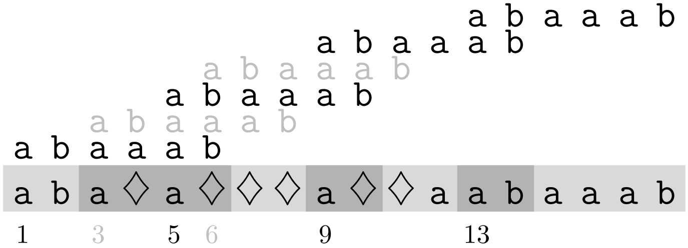
Covering set of a cover



Definition

A set $\mathcal{P} \subseteq Occ(S, T)$ is a *covering set* for S if every position in T is covered by at least one occurrence of S in $\mathcal{P}$.

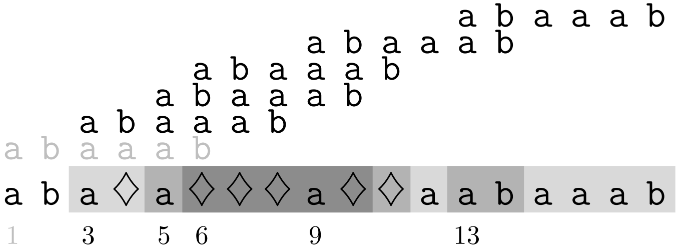
Covering set of a cover



Definition

A set $\mathcal{P} \subseteq Occ(S, T)$ is a *covering set* for S if every position in T is covered by at least one occurrence of S in $\mathcal{P}$.

When $\mathcal{P}$ is a covering set?



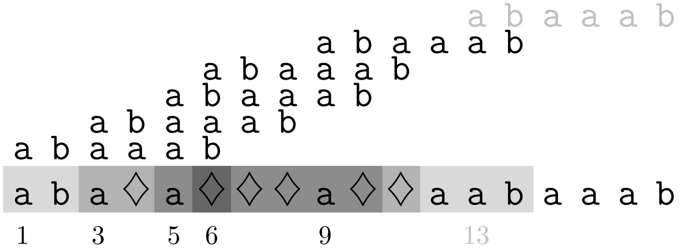
Definition

A set $\mathcal{P} \subseteq Occ(S, T)$ is a *covering set* for S if every position in T is covered by at least one occurrence of S in $\mathcal{P}$.

When $\mathcal{P}$ is a covering set?

- $\min \mathcal{P} = 1$,

Covering set of a cover



Definition

A set $\mathcal{P} \subseteq Occ(S, T)$ is a *covering set* for S if every position in T is covered by at least one occurrence of S in $\mathcal{P}$.

When $\mathcal{P}$ is a covering set?

- $\min \mathcal{P} = 1$,
- $\max \mathcal{P} = |T| - |S| + 1$,

```
                              a b a a a b
                        a b a a a b
                  a b a a a b
                a b a a a b
            a b a a a b
        a b a a a b
        a b a ◇ a ◇ ◇ ◇ a ◇ ◇ a a b a a a b
        1   3   5 6     9       13
```

Definition

A set $\mathcal{P} \subseteq Occ(S,T)$ is a *covering set* for S if every position in T is covered by at least one occurrence of S in $\mathcal{P}$.

When $\mathcal{P}$ is a covering set?

- $\min \mathcal{P} = 1$,
- $\max \mathcal{P} = |T| - |S| + 1$,
- $|q - p| \leq |S|$ for any two consecutive $p, q \in \mathcal{P}$.

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

◊ a ◊ a b a ◊ a b a b a a ◊ a ◊ a b

i	ℓ	1	3	6	13	15
1						
3						
5						
7						
12						
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3						
5						
7						
12						
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

♢ a ♢ a b a ♢ a b a b a a ♢ a ♢

♢ a ♢ a b a ♢ a b a b a a ♢ a ♢ a b

i	ℓ	1	3	6	13	15
1	**18**	♢	♢	♢	♢	♢
3	10	♢	b	b	-	-
5						
7						
12						
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

◊ a ◊ a b a ◊ a b a b a a ◊

◊ a ◊ a b a ◊ a b a b a a ◊ a ◊ a b

i	ℓ	1	3	6	13	15
1	**18**	◊	◊	◊	◊	◊
3	10	◊	b	b	-	-
5	8	b	◊	b	-	-
7						
12						
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a

$\diamondsuit$ a $\diamondsuit$ a b a $\diamondsuit$ a b a b a a $\diamondsuit$ a $\diamondsuit$ a b

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12						
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

♢ a ♢ a b a ♢

♢ a ♢ a b a ♢ a b a b a a ♢ a ♢ a b

i	ℓ	1	3	6	13	15
1	**18**	♢	♢	♢	♢	♢
3	10	♢	b	b	-	-
5	8	b	♢	b	-	-
7	8	♢	b	a	-	-
12	**7**	a	♢	b	-	-
14						
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

◇ a ◇ a b

◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b

i	ℓ	1	3	6	13	15
1	**18**	◇	◇	◇	◇	◇
3	10	◇	b	b	-	-
5	8	b	◇	b	-	-
7	8	◇	b	a	-	-
12	**7**	a	◇	b	-	-
14	**5**	◇	◇	-	-	-
16						

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

◇ a ◇

◇ a ◇ a b a ◇ a b a b a a ◇ a ◇ a b

i	ℓ	1	3	6	13	15
1	**18**	◇	◇	◇	◇	◇
3	10	◇	b	b	-	-
5	8	b	◇	b	-	-
7	8	◇	b	a	-	-
12	**7**	a	◇	b	-	-
14	**5**	◇	◇	-	-	-
16	**3**	◇	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1, 3, 7, 12, 14, 16\}$$

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1,3,7,12,14,16\}$$

$|S| = 3 \quad 12 - 7 > 3$ NO

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1, 7, 12\}$$

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S, T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1, 7, 12\}$$

$|S| = 7$ mismatch at position 6 NO

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1, 3, 7, 12, 14\}$$

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to check covering sets?

- We could miss covers S with $Occ(S,T) \subseteq \mathcal{A}$.
- We will find covers which admit a covering set $\mathcal{P} \subseteq \mathcal{A}$.
- There are $2^{\mathcal{O}(k^2)}$ possibilities for $\mathcal{P}$. How to check one?

$$\mathcal{P} = \{1, 3, 7, 12, 14\}$$

$|S| = 5$ $S[1] = \mathtt{a}$ $S[3] = \mathtt{b}$ YES

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.

Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \Diamond$ for some $j < |S|$.

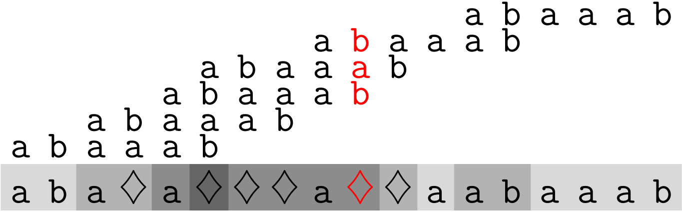
- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.

```
                        a b a a a b
                a b a a a b
          a b a a a b
        a b a a a b
    a b a a a b
a b a a a b
a b a ◊ a ◊ ◊ ◊ a ◊ ◊ a a b a a a b
```

Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

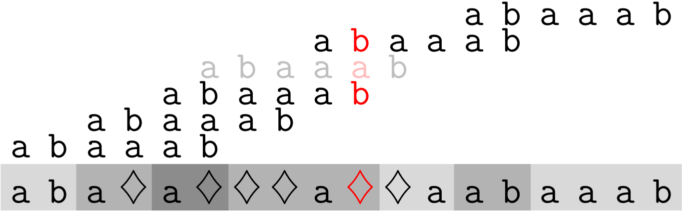
- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.



Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

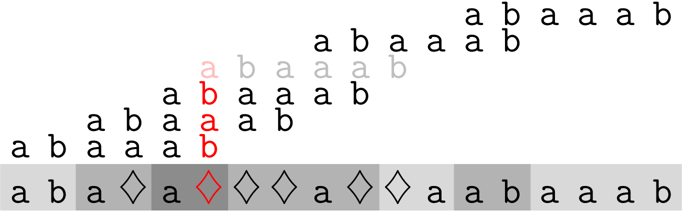
- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.



Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.



Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

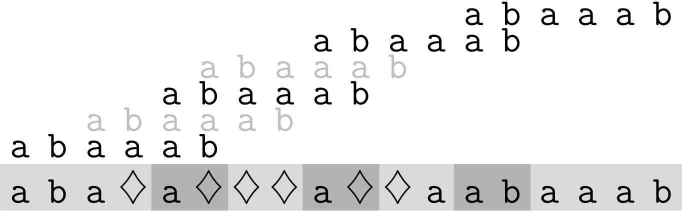
- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.

```
                            a b a a a b
                        a b a a a b
                    a b a a a b
                a b a a a b
            a b a a a b
        a b a a a b
        a b a ◇ a ◇ ◇ ◇ a ◇ ◇ a a b a a a b
```

Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.



- If $\mathcal{P}$ is minimal, every position is covered by 1 or 2 occurrences $i \in \mathcal{P}$.

Do we need to verify all subsets?

We could only miss covers S such that for each $i \in Occ(S,T)$ we have $T[1+j] = T[i+j] = \diamondsuit$ for some $j < |S|$.

- Every occurrence of $i \in \mathcal{P}$ covers a non-solid position.

```
                        a b a a a b
                a b a a a b
          a b a a a b
        a b a a a b
    a b a a a b
a b a a a b
a b a ◇ a ◇ ◇ ◇ a ◇ ◇ a a b a a a b
```

- If $\mathcal{P}$ is minimal, every position is covered by 1 or 2 occurrences $i \in \mathcal{P}$.
- It suffices to consider $\mathcal{P}$ with $|\mathcal{P}| \le 2k$,

$$\binom{|\mathcal{A}|}{\le 2k} = \mathcal{O}(2^{2k \log |A|}) = 2^{\mathcal{O}(k \log k)}.$$

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Can guessing still help?

1. $\max \mathcal{P} = |T| - |S| + 1$ ($\mathcal{O}(|\mathcal{A}|)$ choices)

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Can guessing still help?

1. $\max \mathcal{P} = |T| - |S| + 1$ ($\mathcal{O}(|\mathcal{A}|)$ choices)
2. Some $i \in \mathcal{P}$ ($\mathcal{O}(|\mathcal{A}|)$ choices).

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	$\diamondsuit$	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Can guessing still help?

1. $\max \mathcal{P} = |T| - |S| + 1$ $\quad (\mathcal{O}(|\mathcal{A}|)$ choices)
2. Some $i \in \mathcal{P}$ $\quad (\mathcal{O}(|\mathcal{A}|)$ choices).
3. How to fill $\diamondsuit$'s in the i-th row $\quad (\mathcal{O}(|\mathcal{A}|^h)$ choices).
   Fail if more than h $\diamondsuit$'s, check candidates otherwise.

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	b	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Can guessing still help?

1. $\max \mathcal{P} = |T| - |S| + 1$ $\quad (\mathcal{O}(|\mathcal{A}|)$ choices)
2. Some $i \in \mathcal{P}$ $\quad (\mathcal{O}(|\mathcal{A}|)$ choices).
3. How to fill $\diamondsuit$'s in the i-th row $\quad (\mathcal{O}(|\mathcal{A}|^h)$ choices).
   Fail if more than h $\diamondsuit$'s, check candidates otherwise.

How to fill few holes?

i	ℓ	1	3	6	13	15
1	**18**	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$	$\diamondsuit$
3	10	$\diamondsuit$	b	b	-	-
5	8	b	$\diamondsuit$	b	-	-
7	8	a	b	a	-	-
12	**7**	a	$\diamondsuit$	b	-	-
14	**5**	$\diamondsuit$	$\diamondsuit$	-	-	-
16	**3**	$\diamondsuit$	b	-	-	-

Can guessing still help?

1. $\max \mathcal{P} = |T| - |S| + 1$ ($\mathcal{O}(|\mathcal{A}|)$ choices)
2. Some $i \in \mathcal{P}$ ($\mathcal{O}(|\mathcal{A}|)$ choices).
3. How to fill $\diamondsuit$'s in the i-th row ($\mathcal{O}(|\mathcal{A}|^h)$ choices).
   Fail if more than h $\diamondsuit$'s, check candidates otherwise.

Which subsets do we still need to check?

Which covers could we still miss?

Which subsets do we still need to check?

Which covers could we still miss?

- $T[1..|S|] \cap T[i..i+|S|-1]$ has at least $h+1$ holes for each $i \in Occ(S,T)$.

Which subsets do we still need to check?

Which covers could we still miss?

- $T[1..|S|] \cap T[i..i+|S|-1]$ has at least $h+1$ holes for each $i \in Occ(S,T)$.
- Every occurrence $i \in \mathcal{P}$ covers at least $h+1$ non-solid symbols.
- $|P| \le 2\frac{k}{h+1}$ if $\mathcal{P}$ is minimal.

$$\binom{|\mathcal{A}|}{\le 2\frac{k}{h+1}} = 2^{\mathcal{O}\left(\frac{k \log k}{h}\right)}.$$

Which subsets do we still need to check?

Which covers could we still miss?

- $T[1..|S|] \cap T[i..i+|S|-1]$ has at least $h+1$ holes for each $i \in Occ(S,T)$.
- Every occurrence $i \in \mathcal{P}$ covers at least $h+1$ non-solid symbols.
- $|P| \leq 2\frac{k}{h+1}$ if $\mathcal{P}$ is minimal.

$$\binom{|\mathcal{A}|}{\leq 2\frac{k}{h+1}} = 2^{\mathcal{O}\left(\frac{k \log k}{h}\right)}.$$

In total:

- $\mathcal{O}(|\mathcal{A}|^{h+2}) = 2^{\mathcal{O}(h \log k)}$ for guessing,
- $\mathcal{O}(|\mathcal{A}|^{h+2}) = 2^{\mathcal{O}(\frac{k}{h} \log k)}$ for finding missed covers,
- $2^{\mathcal{O}(\sqrt{k} \log k)}$ for $h = \sqrt{k}$.

Conclusions and open problems

Our results:

- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Conclusions and open problems

Our results:

- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Open problems:

- Are indeterminate strings inherently harder?
  - For constant alphabets, no!

Conclusions and open problems

Our results:

- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Open problems:

- Are indeterminate strings inherently harder?
  - For constant alphabets, no!
- What is the right exponent for partial words?
  - For constant alphabets, at most $\mathcal{O}(\sqrt{k \log k})$!

Conclusions and open problems

Our results:

- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(k \log k)})$-time algorithm,
- $\mathcal{O}(nk^4 + 2^{\mathcal{O}(\sqrt{k} \log k)})$-time version for partial words,
- $2^{o(\sqrt{k})} n^{\mathcal{O}(1)}$ lower bound already for binary partial words (under ETH).

Open problems:

- Are indeterminate strings inherently harder?
  - For constant alphabets, no!
- What is the right exponent for partial words?
  - For constant alphabets, at most $\mathcal{O}(\sqrt{k \log k})$!
- Polynomial kernel?
  - For partial words, yes! (obtained through NP-hardness)

Thank you

Thank you for your attention!

Full version available at `http://arxiv.org/abs/1412.3696`.