The Communication Complexity of Pattern Matching with Edits Revisited

Tomasz Kociumaka[1] **Jakob Nogler**[2] Philip Wellnitz[3]

[1]Max Planck Institute for Informatics, SIC

[2]MIT

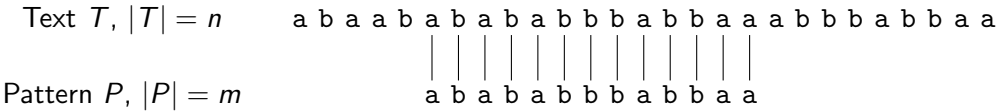
[3]National Institute of Informatics, SOKENDAI

Pattern Matching (PM)

Text T, $|T| = n$ a b a a b a b a b a b b b a b b a a a b b b a b b a a

Pattern P, $|P| = m$ a b a b a b b b a b b a a

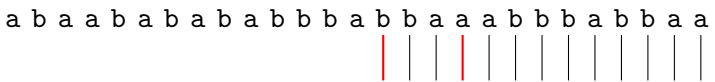
Pattern Matching (PM)



- **Exact PM:** Compute $\mathrm{Occ}(P, T) := \{x \mid T[x..x+m) = P\}$.

Pattern Matching (PM)

Text T, $|T| = n$

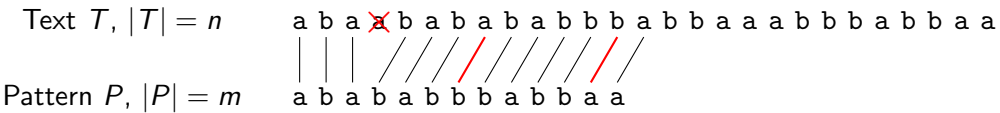


Pattern P, $|P| = m$

$k = 2$

- **Exact PM:** Compute $\mathrm{Occ}(P, T) := \{x \mid T[x \,..\, x + m) = P\}$.
- **PM with mismatches:** Compute $\mathrm{Occ}_k^H(P, T) := \{x \mid \mathrm{HD}(T[x \,..\, x + m), P) \leq k\}$.

Pattern Matching (PM)

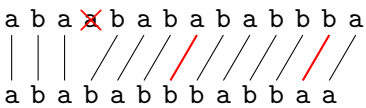


- **Exact PM:** Compute $\mathrm{Occ}(P, T) := \{x \mid T[x\,..\,x+m) = P\}$.
- **PM with mismatches:** Compute $\mathrm{Occ}_k^H(P, T) := \{x \mid \mathrm{HD}(T[x\,..\,x+m), P) \leq k\}$.
- **PM with edits:** Compute $\mathrm{Occ}_k^E(P, T) := \{x \mid \exists y\ \mathrm{ED}(T[x\,..\,y), P) \leq k\}$.

Pattern Matching (PM)

Text T, $|T| = n$

Pattern P, $|P| = m$



$k = 3$

- **Exact PM:** Compute $\mathrm{Occ}(P, T) := \{x \mid T[x \,..\, x + m) = P\}$.
- **PM with mismatches:** Compute $\mathrm{Occ}_k^H(P, T) := \{x \mid \mathrm{HD}(T[x \,..\, x + m), P) \leq k\}$.
- **PM with edits:** Compute $\mathrm{Occ}_k^E(P, T) := \{x \mid \exists y \ \mathrm{ED}(T[x \,..\, y), P) \leq k\}$.

Communication Complexity

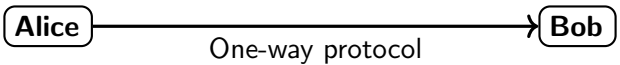
What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?

Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

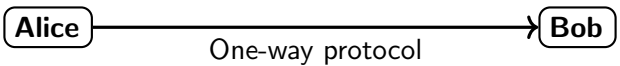
Can be modeled through a protocol:



Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Can be modeled through a protocol:

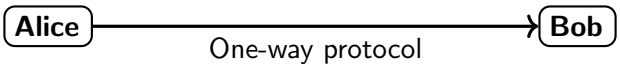


1. Alice receives a PM instance.
   *T*ext T, Pattern P, Threshold k

Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Can be modeled through a protocol:



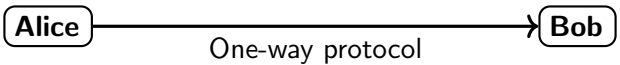
(1) Alice receives a PM instance.
*T*ext T, Pattern P, Threshold k

(2) Alice compresses the input.

Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Can be modeled through a protocol:



(1) Alice receives a PM instance.
*T*ext T, Pattern P, Threshold k

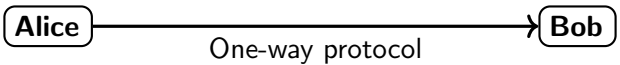
(2) Alice compresses the input.

(3) Alice sends compressed data to Bob.

Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Can be modeled through a protocol:



(1) Alice receives a PM instance.
Text T, Pattern P, Threshold k

(2) Alice compresses the input.

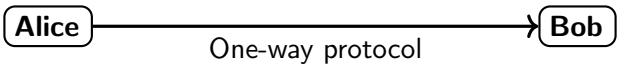
(3) Alice sends compressed data to Bob.

(4) Bob needs to reconstructs the output of the instance.
Set $Occ_k^E(P, T)$

Communication Complexity

What is the minimum amount of space we can compress the solution of an (approximate) pattern matching instance?
Solution: $\mathrm{Occ}(P, T)$, $\mathrm{Occ}_k^H(P, T)$ or $\mathrm{Occ}_k^E(P, T)$

Can be modeled through a protocol:



(1) Alice receives a PM instance.
Text T, Pattern P, Threshold k

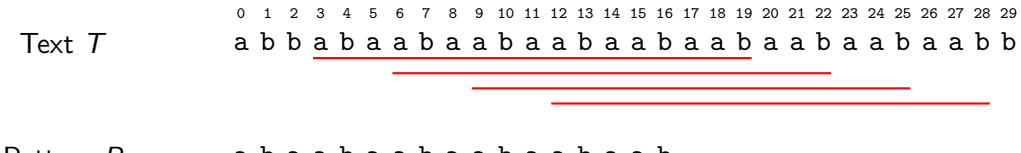
(2) Alice compresses the input.

(3) Alice sends compressed data to Bob.

(4) Bob needs to reconstructs the output of the instance.
Set $Occ_k^E(P, T)$

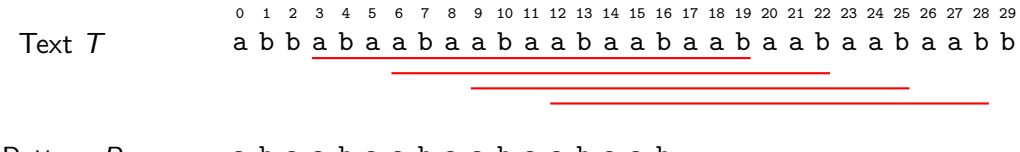
Communication Complexity = "minimum bits to send to Bob"

Example for Exact Pattern Matching



Alice needs to send to Bob the set $\mathrm{Occ}(P, T) = \{3, 6, 9, 12\}$

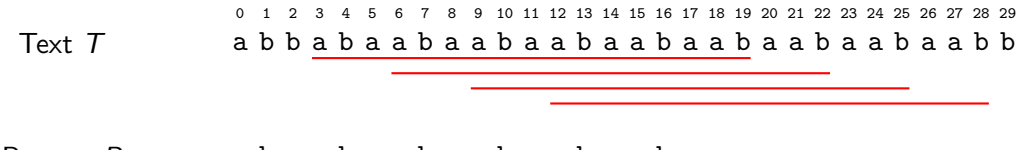
Example for Exact Pattern Matching



Alice needs to send to Bob the set $\mathrm{Occ}(P, T) = \{3, 6, 9, 12\}$

She has more than one way how to do it:

Example for Exact Pattern Matching

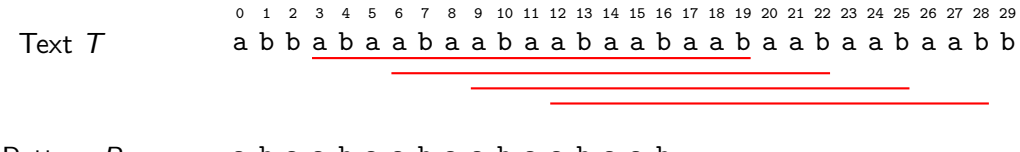


Alice needs to send to Bob the set $\mathrm{Occ}(P, T) = \{3, 6, 9, 12\}$

She has more than one way how to do it:

1. She can send $\mathrm{Occ}(P, T)$ explicitly.

Example for Exact Pattern Matching

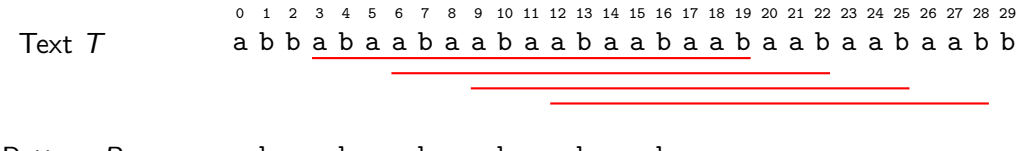


Alice needs to send to Bob the set $\mathrm{Occ}(P, T) = \{3, 6, 9, 12\}$

She has more than one way how to do it:

1. She can send $\mathrm{Occ}(P, T)$ explicitly.
2. She can send $\mathrm{Occ}(P, T)$ in a compressed form (e.g., arithmetic progression).

Example for Exact Pattern Matching



Alice needs to send to Bob the set $\mathrm{Occ}(P, T) = \{3, 6, 9, 12\}$

She has more than one way how to do it:

1. She can send $\mathrm{Occ}(P, T)$ explicitly.
2. She can send $\mathrm{Occ}(P, T)$ in a compressed form (e.g., arithmetic progression).
3. She can send P and T directly.

General Assumptions

1. $n \leq 3/2 \cdot m$

General Assumptions

1. $\boldsymbol{n \leq 3/2 \cdot m}$

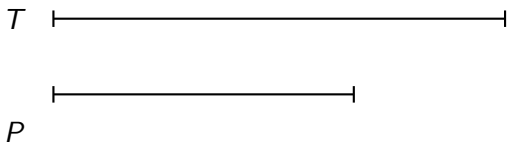
- Divide T into $\Theta(n/m)$ blocks of length $n \leq 3/2 \cdot m$, and apply protocol on each block.

General Assumptions

1. $\boldsymbol{n \leq 3/2 \cdot m}$

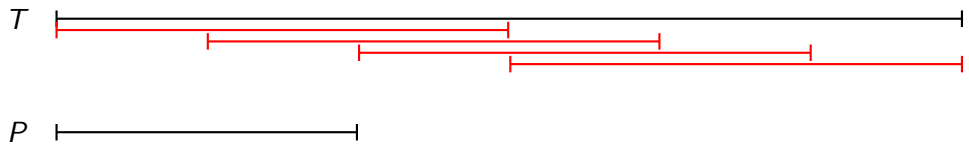
- Divide T into $\Theta(n/m)$ blocks of length $n \leq 3/2 \cdot m$, and apply protocol on each block.

2. **An exact/k-mismatch/k-edit occurrence of P aligns with prefix and suffix of T**



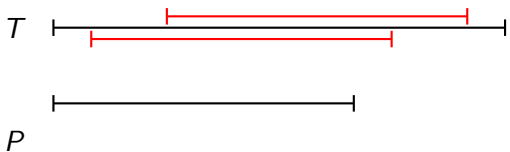
General Assumptions

1. $\boldsymbol{n \leq 3/2 \cdot m}$



- Divide T into $\Theta(n/m)$ blocks of length $n \leq 3/2 \cdot m$, and apply protocol on each block.

2. **An exact/k-mismatch/k-edit occurrence of P aligns with prefix and suffix of T**

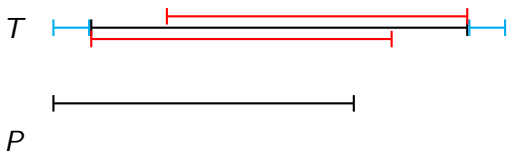


General Assumptions

1. $\boldsymbol{n \leq 3/2 \cdot m}$

- Divide T into $\Theta(n/m)$ blocks of length $n \leq 3/2 \cdot m$, and apply protocol on each block.

2. **An exact/k-mismatch/k-edit occurrence of P aligns with prefix and suffix of T**



Previous Results

	Upper Bound $\mathcal{O}(\cdot)$	**Lower Bound $\Omega(\cdot)$**	**Reference**
Exact PM	$\log m$	$\log m$	[FW65]

Previous Results

	Upper Bound $\mathcal{O}(\cdot)$	**Lower Bound $\Omega(\cdot)$**	**Reference**
Exact PM	$\log m$	$\log m$	[FW65]
PM with mismatches	$k \cdot \log(m\|\Sigma\|/k)$	$k \log(m\|\Sigma\|/k)$	[CKP19]

Blue: We report also the positions and characters of mismatches/edits.

Previous Results

	Upper Bound $\mathcal{O}(\cdot)$	**Lower Bound $\Omega(\cdot)$**	**Reference**
Exact PM	$\log m$	$\log m$	[FW65]
PM with mismatches	$k \cdot \log(m\|\Sigma\|/k)$	$k \log(m\|\Sigma\|/k)$	[CKP19]
PM with edits	$k^3 \log(m\|\Sigma\|)$		[CKW20]

Blue: We report also the positions and characters of mismatches/edits.

Previous Results

	Upper Bound $\mathcal{O}(\cdot)$	**Lower Bound $\Omega(\cdot)$**	**Reference**
Exact PM	$\log m$	$\log m$	[FW65]
PM with mismatches	$k \cdot \log(m\lvert\Sigma\rvert/k)$	$k \log(m\lvert\Sigma\rvert/k)$	[CKP19]
PM with edits	$k^3 \log(m\lvert\Sigma\rvert)$		[CKW20]
PM with edits	$k \log m \log(m\lvert\Sigma\rvert)$	$k \log(m/k)$	[K**N**W24]

Blue: We report also the positions and characters of mismatches/edits.

Previous Results

	Upper Bound $\mathcal{O}(\cdot)$	**Lower Bound $\Omega(\cdot)$**	**Reference**
Exact PM	$\log m$	$\log m$	[FW65]
PM with mismatches	$k \cdot \log(m\lvert\Sigma\rvert/k)$	$k \log(m\lvert\Sigma\rvert/k)$	[CKP19]
PM with edits	$k^3 \log(m\lvert\Sigma\rvert)$		[CKW20]
PM with edits	$k \log m \log(m\lvert\Sigma\rvert)$	$k \log(m/k)$	[K**N**W24]
PM with edits	$\boldsymbol{k \log(m\lvert\Sigma\rvert/k)}$	$\boldsymbol{k \log(m\lvert\Sigma\rvert/k)}$	**This work**

Blue: We report also the positions and characters of mismatches/edits.

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].
- Best classical algorithm for pattern matching with edits runs in time $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3.5})$ [CKW22];

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].
- Best classical algorithm for pattern matching with edits runs in time $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3.5})$ [CKW22]; but lower bound is only $\mathcal{O}(n + n/m \cdot k^{2-\Omega(1)})$ [BI18];

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].

- Best classical algorithm for pattern matching with edits runs in time $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3.5})$ [CKW22];
  but lower bound is only $\mathcal{O}(n + n/m \cdot k^{2-\Omega(1)})$ [BI18];
  but there is a barrier: [CKW22] relies on output representation of $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3})$ bits.

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].

- Best classical algorithm for pattern matching with edits runs in time $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3.5})$ [CKW22];
  but lower bound is only $\mathcal{O}(n + n/m \cdot k^{2-\Omega(1)})$ [BI18];
  but there is a barrier: [CKW22] relies on output representation of $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3})$ bits.

- We can prove that there are P', T' such that:
  1. P', T' are a compression of P, T of size $\widetilde{\mathcal{O}}(k)$,
  2. $\mathrm{Occ}_E^k(P, T) = \mathrm{Occ}_E^k(P', T')$. ($P', T'$ preserve approximate occurrences)

Applications

- For mismatches, these results are at the basis for the $\widetilde{\mathcal{O}}(k)$-space streaming algorithm [CKP19].

- Best classical algorithm for pattern matching with edits runs in time $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3.5})$ [CKW22];
  but lower bound is only $\mathcal{O}(n + n/m \cdot k^{2-\Omega(1)})$ [BI18];
  but there is a barrier: [CKW22] relies on output representation of $\widetilde{\mathcal{O}}(n + n/m \cdot k^{3})$ bits.

- We can prove that there are P', T' such that:
  1. P', T' are a compression of P, T of size $\widetilde{\mathcal{O}}(k)$,
  2. $\mathrm{Occ}_E^k(P, T) = \mathrm{Occ}_E^k(P', T')$. ($P', T'$ preserve approximate occurrences)

- The last results are at the basis of our quantum algorithms [K**N**W24], [K**N**W25].

Protocol Sketch

- I will sketch main ideas between (old) construction
- Disclaimer: A lot of technical details are omitted

Protocol Sketch

- I will sketch main ideas between (old) construction
- Disclaimer: A lot of technical details are omitted

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.

2. Some additional characters in P and T.

Protocol Sketch

- I will sketch main ideas between (old) construction
- Disclaimer: A lot of technical details are omitted

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
   - Will include one as a prefix and one as a suffix.
   - Alignment information contains: positions and characters of edits.
   - Each alignment can be sent using $\mathcal{O}(k \log(m|\Sigma|))$ bits.
2. Some additional characters in P and T.

Protocol Sketch

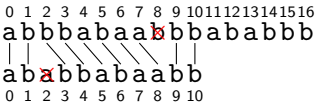
- I will sketch main ideas between (old) construction
- Disclaimer: A lot of technical details are omitted

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma| m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
   - Will include one as a prefix and one as a suffix.
   - Alignment information contains: positions and characters of edits.
   - Each alignment can be sent using $\mathcal{O}(k \log(m|\Sigma|))$ bits.
2. Some additional characters in P and T.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



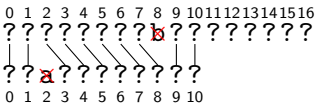
$\mathcal{A}_1$

$\mathcal{A}_2$

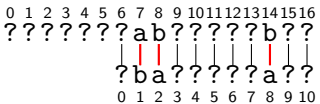
1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



$\mathcal{A}_1$

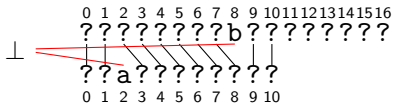


$\mathcal{A}_2$

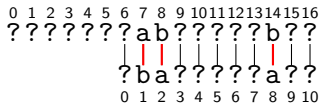
1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



$\mathcal{A}_1$

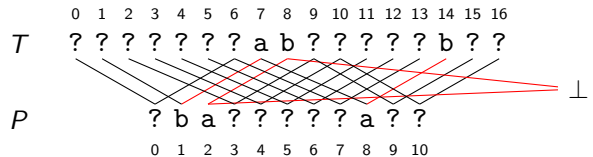


$\mathcal{A}_2$

1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

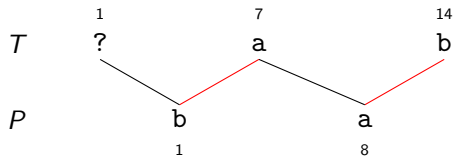
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

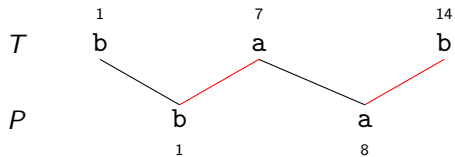
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit):

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

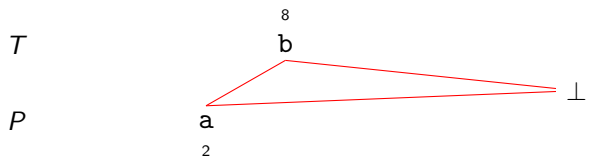
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

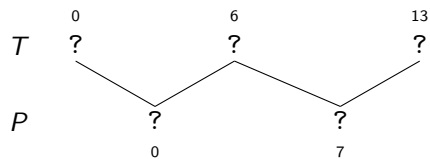
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

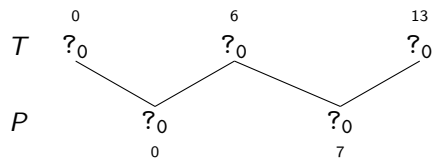
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit):

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

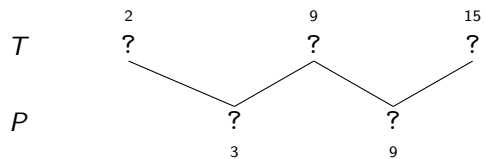
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

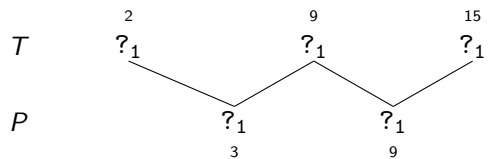
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

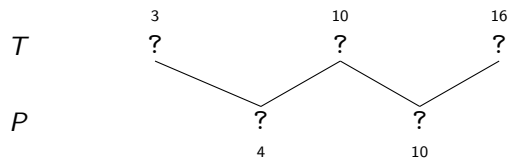
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

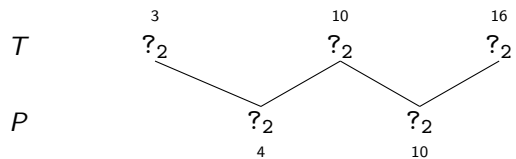
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

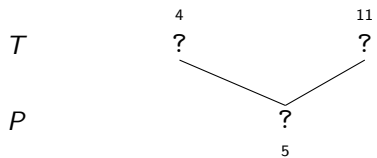
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

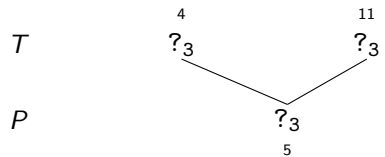
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

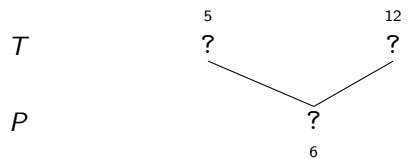
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

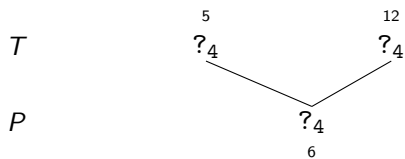
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

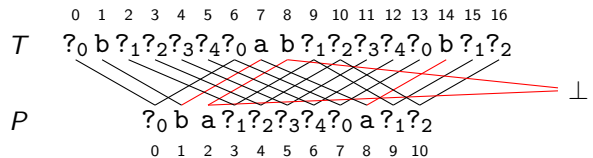
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
   Red Components (at least one red edge=edit): Bob can learn every character in them.
   Black Components (no red edge=edit): Bob can not learn any character, but all characters are the same.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.



1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.

T_S $?_0$ $?_1?_2?_3?_4?_0$ $?_1?_2?_3?_4?_0$ $?_1?_2$

P_S $?_0$ $?_1?_2?_3?_4?_0$ $?_1?_2$

1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
3. Constructs P_S and T_S from P and T by only retaining characters in black components.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.

$$T_S = \quad ?_0?_1?_2?_3?_4?_0?_1?_2?_3?_4?_0?_1?_2$$

$$P_S = \quad ?_0?_1?_2?_3?_4?_0?_1?_2$$

1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
3. Constructs P_S and T_S from P and T by only retaining characters in black components.

   P_S and T_S are periodic. Period = # of black bomponents.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T. ⟵ What can Bob infer from S?
2. Some additional characters in P and T.

$$T_S = \quad \mathbf{0\,1\,2\,4\,0\,1\,2\,3\,4\,0\,1\,2}$$

$$P_S = \quad \mathbf{0\,1\,2\,3\,4\,0\,1\,2}$$

1. Bob receives $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
2. Bob constructs an inference graph $\mathbf{G}_S$ and looks at the connected components.
3. Constructs P_S and T_S from P and T by only retaining characters in black components.

   P_S and T_S are periodic. Period = # of black bomponents.

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

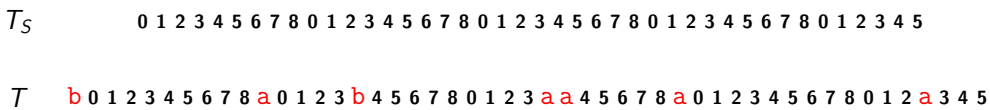
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.

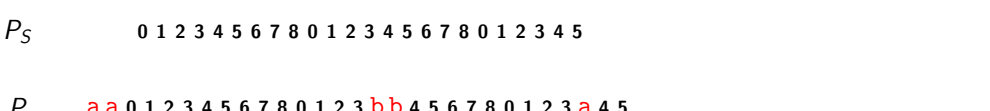
T_S **0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5**

P_S **0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5**

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

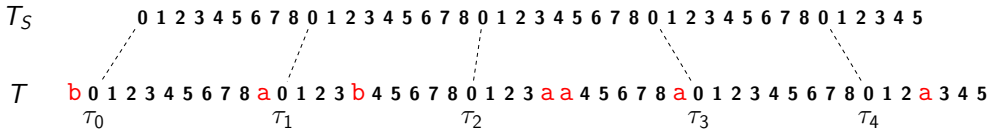
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.

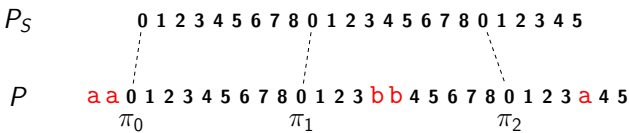




Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

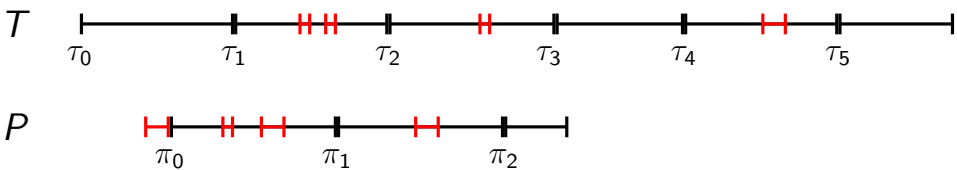
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.





Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

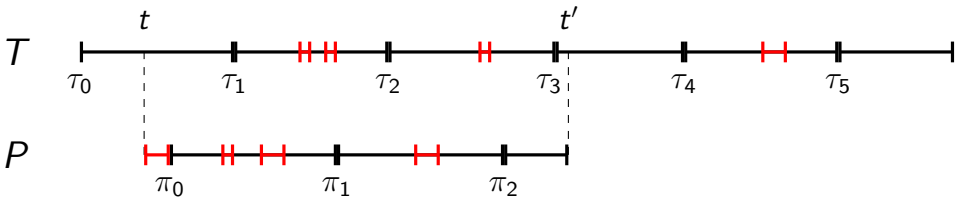
1. A set S of alignments of cost $\leq k$ of P onto fragments of T. $\longleftarrow$ What can Bob infer from S?
2. Some additional characters in P and T.



Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

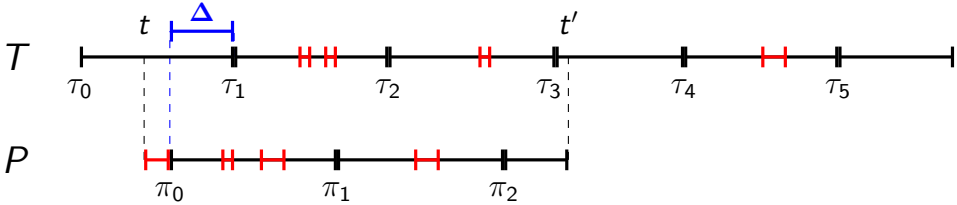
- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.



Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

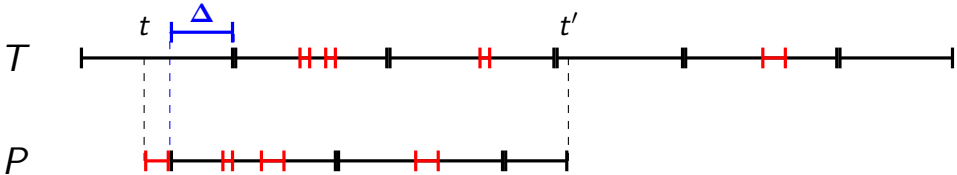
- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.



Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.

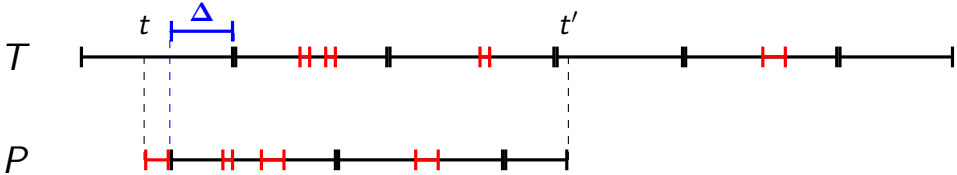


Case $\boldsymbol{\Delta} \gg k$ (large)

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.



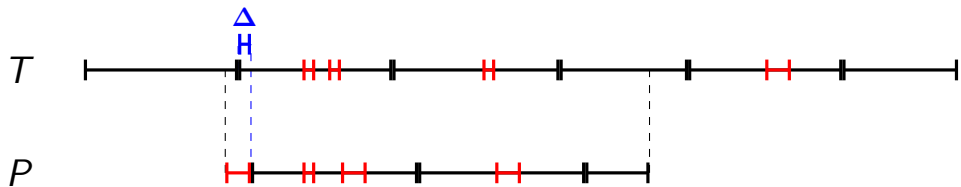
Case $\boldsymbol{\Delta} \gg k$ (large) $\Longrightarrow$ if $\mathcal{X}$ is added to S, the # of black components at least halves.

N.B. We can assume no such alignments exist. Communication cost $= \mathcal{O}(\underbrace{|S|}_{\leq \log m} \cdot k \cdot \log(m|\Sigma|))$

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.

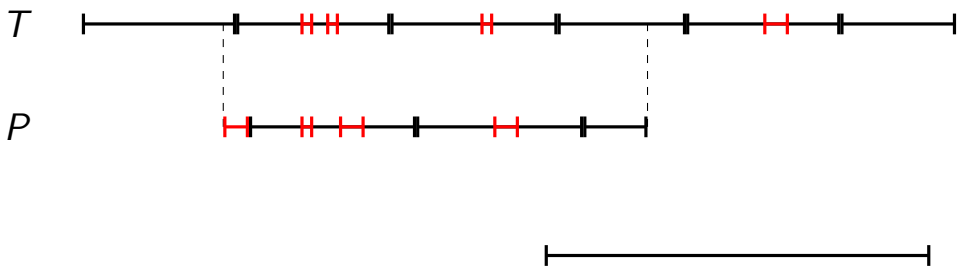


Case $\boldsymbol{\Delta} \lesssim k$ (small)

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[\,t\,..\,t'\,)$ such that $\mathrm{ED}(P, T[\,t\,..\,t'\,)) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.

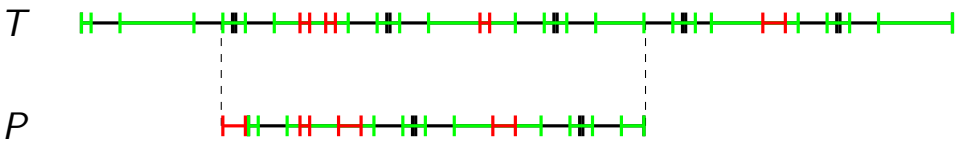


Case $\boldsymbol{\Delta} \lesssim k$ (small)

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[t..t']$ such that $\mathrm{ED}(P, T[t..t']) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.



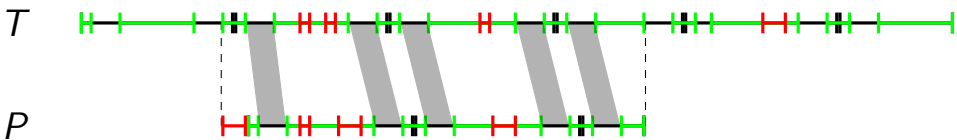
encoding cost $\approx$ same cost of what we encoded in 1.

Case $\boldsymbol{\Delta} \lesssim k$ (small)

Alice sends to Bob (at most $\approx k \cdot \log m \cdot \log(|\Sigma|m)$ bits):

1. A set S of alignments of cost $\leq k$ of P onto fragments of T.
2. Some additional characters in P and T.

- Consider an arbitrary fragment $T[t..t']$ such that $\mathrm{ED}(P, T[t..t']) \leq k$.
- Compute $\boldsymbol{\Delta}$ = period offset.



encoding cost $\approx$ same cost of what we encoded in 1.

Case $\boldsymbol{\Delta} \lesssim k$ (small) $\Longrightarrow$ an optimal alignment $P \rightsquigarrow T[t..t']$ matches characters in the same uncovered black components.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Challenge 1: Now S contains alignments of fragments of P onto fragments on T.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Challenge 1: Now S contains alignments of fragments of P onto fragments on T.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Challenge 1: Now S contains alignments of fragments of P onto fragments on T.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Challenge 1: Now S contains alignments of fragments of P onto fragments on T.

Challenge 2: Arguments become very tricky because some cyclic dependencies.

Improved Construction (Intuition)

Key intuition:

While adding "uncaptured" alignments, we can do better than adding the whole alignments:

When P_S has 2^i period occurrences we can add a subset of the alignment of cost $\approx k/2^i$.

Challenge 1: Now S contains alignments of fragments of P onto fragments on T.

Challenge 2: Arguments become very tricky because some cyclic dependencies.

Challenge 3: For each alignment we send $\Omega(\log m)$ bits, this is a problem when S grows to size $\omega(k)$. When this happens we: P and T are periodic in a stronger sense (in the sense of [CKW20]), and that in such case we can construct S of size 3.

Thanks!