Subsequences With Gap Constraints: Complexity Bounds for Matching and Analysis Problems

Joel D. Day [1] Maria Kosche [2] Florin Manea [2] Markus Schmid [3]

[1]Loughborough University, UK

[2]Göttingen University, Germany

[3]Humboldt University of Berlin, Germany

16th International Conference on Reachability Problems
17 – 19 October 2022
Kaiserslautern

Classical Subsequences

Subsequences

$$w \quad \boxed{b \quad a \quad c \quad b \quad a \quad a \quad b \quad a \quad d \quad a}$$

Subsequences

ababa

d

bbb

ba

bc

w $\boxed{b \quad a \quad c \quad b \quad a \quad a \quad b \quad a \quad d \quad a}$

baab

a

bb

c

aaaaa

b

Subsequences

ababa

d

bbb

abc

ba

bc

w $\boxed{b \quad a \quad c \quad b \quad a \quad a \quad b \quad a \quad d \quad a}$

baab

a

bb

c

aaaaa

b

ababa

d

bbb

ba

~~abc~~

bc

w $\boxed{b \quad a \quad c \quad b \quad a \quad a \quad b \quad a \quad d \quad a}$

baab

a

bb

c

aaaaa

b

Subsequences

Σ is a finite alphabet, e. g., $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$
\begin{array}{ccccccccc}
 & u[1] & u[2] & u[3] & u[4] & u[5] & u[6] & u[7] & u[8] \\
u = & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{a} & \mathtt{c} & \mathtt{b} & \mathtt{a} & \mathtt{b}
\end{array}
$$

$$
\begin{array}{cccc}
v = & \mathtt{b} & \mathtt{b} & \mathtt{a} \\
 & v[1] & v[2] & v[3]
\end{array}
$$

Subsequences

Σ is a finite alphabet, e. g., $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$\begin{array}{ccccccccc} & u[1] & u[2] & u[3] & u[4] & u[5] & u[6] & u[7] & u[8] \\ u = & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{a} & \mathtt{c} & \mathtt{b} & \mathtt{a} & \mathtt{b} \end{array}$$

$$\begin{array}{cccc} v = & \mathtt{b} & \mathtt{b} & \mathtt{a} \\ & v[1] & v[2] & v[3] \end{array}$$

Subsequences

Σ is a finite alphabet, e. g., $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$\begin{array}{ccccccccc}
 & u[1] & u[2] & u[3] & u[4] & u[5] & u[6] & u[7] & u[8] \\
u = & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{a} & \mathtt{c} & \mathtt{b} & \mathtt{a} & \mathtt{b}
\end{array}$$

$$\begin{array}{cccc}
v = & \mathtt{b} & \mathtt{b} & \mathtt{a} \\
 & v[1] & v[2] & v[3]
\end{array}$$

Notation

Embedding: $e : \{1, \ldots, |v|\} \to \{1, \ldots, |u|\}$ with $e(1) < \ldots < e(|v|)$.

Subsequences

Σ is a finite alphabet, e. g., $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$\begin{array}{ccccccccc} & u[1] & u[2] & u[3] & u[4] & u[5] & u[6] & u[7] & u[8] \\ u = & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{a} & \mathtt{c} & \mathtt{b} & \mathtt{a} & \mathtt{b} \end{array}$$

$$\begin{array}{cccc} v = & \mathtt{b} & \mathtt{b} & \mathtt{a} \\ & v[1] & v[2] & v[3] \end{array}$$

Notation

Embedding: $e : \{1, \ldots, |v|\} \to \{1, \ldots, |u|\}$ with $e(1) < \ldots < e(|v|)$.
$v \preceq_e u$: e is an embedding with $v[i] = u[e(i)]\ \forall i \in \{1, 2, \ldots, |v|\}$.

Subsequences

Σ is a finite alphabet, e. g., $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$\begin{array}{cccccccccc} & u[1] & u[2] & u[3] & u[4] & u[5] & u[6] & u[7] & u[8] \\ u = & \mathtt{a} & \mathtt{b} & \mathtt{b} & \mathtt{a} & \mathtt{c} & \mathtt{b} & \mathtt{a} & \mathtt{b} \end{array}$$

$$\begin{array}{cccc} v = & \mathtt{b} & \mathtt{b} & \mathtt{a} \\ & v[1] & v[2] & v[3] \end{array}$$

Notation

Embedding: $e : \{1, \ldots, |v|\} \to \{1, \ldots, |u|\}$ with $e(1) < \ldots < e(|v|)$.
$v \preceq_e u$: e is an embedding with $v[i] = u[e(i)]\ \forall i \in \{1, 2, \ldots, |v|\}$.

v is subsequence of u ($v \preceq u$) if there is some embedding e with $v \preceq_e u$.

Subsequences

Set of Subsequences of length k

For given integer k and word w, let $Subseq(k, w)$ denote the set of subsequences of length k of w.

$$Subseq(k, w) = \{v \in \Sigma^k \mid v \preceq w\}.$$

Example:

$$\begin{aligned} Subseq(3, abbacbab) = \{&aaa, aab, aac, aba, abb, abc, aca, acb,\\ &baa, bab, bac, bba, bbb, bbc, bca, bcb,\\ &cab, cba, cbb\} \end{aligned}$$

Subsequences in TCS

Subsequences are a central concept in many different areas of TCS:

- Formal languages and logics (piecewise testable languages, subword order and downward closures).
- Combinatorics on words.
- Modelling concurrency.
- Database theory (event stream processing).
- Algorithms (longest common subsequence, shortest common supersequence).

Computational Problems for Subsequences

Matching

Input:	$u, v \in \Sigma^*$
Question:	$v \preceq u$?

Computational Problems for Subsequences

Matching

Input: $u, v \in \Sigma^*$

Question: $v \preceq u$?

Analysis Problems

Input: $u, v \in \Sigma^*,\ k \in \mathbb{N}$

Questions:

$Subseq(k, u) = Subseq(k, v)$? (Equivalence)

$Subseq(k, u) \subseteq Subseq(k, v)$? (Containment)

$Subseq(k, u) = \Sigma^k$? (Universality)

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

a b c b c a b c a b a c

c a b a

Subsequence Problems are Simple

`a b c b c a b c a b a c`

`c a b a`

Subsequence Problems are Simple

Observation

Matching is trivially solvable in linear time.

Subsequence Problems are Simple

Observation

Matching is trivially solvable in linear time.

Observation

We can construct a DFA for $\mathit{Subseq}(k, u)$ of size $\mathrm{O}(k|u|)$.
$\Rightarrow$ the analysis problems can be solved in polynomial time.

Subsequence Problems are Simple

Observation

Matching is trivially solvable in linear time.

Observation

We can construct a DFA for $\mathit{Subseq}(k, u)$ of size $\mathrm{O}(k|u|)$.
$\Rightarrow$ the analysis problems can be solved in polynomial time.

Theorem (Gawrychowski et al., STACS 2021)

Equivalence can be decided in linear time.

Subsequences with Gap Constraints

Shift from Classical Scenario to Gap Constraints

Classical subsequences . . .

. . . are usually considered with arbitrary embeddings.

For practical scenarios, it is reasonable to introduce *gap constraints*.

We restrict the length of gaps.

- Alignments of bio-sequences.
- Modelling single processor scheduling with fairness properties.

And we restrict allowed symbols that can occur in gaps.

- Complex event processing $\rightarrow$ forbidding events in specific positions of a subsequence.

$u =$ `a b c b c a b c a b a c`

$v =$ `c a b a`

$u =$ a b c b c a b c a b a c

$v =$ c a b a

$$\mathrm{gap}_e(u, i) = u[e(i) + 1..e(i+1) - 1] \qquad i \in \{1, \dots, |v| - 1\}$$

$u =$ `a b c b c a b c a b a c`

$v =$ `c a b a`

$$\mathsf{gap}_e(u, 1) = \texttt{bc}$$
$$\mathsf{gap}_e(u, 2) = \varepsilon$$
$$\mathsf{gap}_e(u, 3) = \texttt{c}$$

$u =$ `a b c b c a b c a b a c`

$v =$ `c a b a`

$$\mathsf{gap}_e(u, 1) = \texttt{bcabc}$$

$$\mathsf{gap}_e(u, 2) = \varepsilon$$

$$\mathsf{gap}_e(u, 3) = \varepsilon$$

$u =$ a b c b c a b c a b a c

$v =$ c a b a

$$\mathrm{gap}_e(u,1) = \texttt{bcabc}$$
$$\mathrm{gap}_e(u,2) = \varepsilon$$
$$\mathrm{gap}_e(u,3) = \varepsilon$$

Gap constraints

$gc = (C_1, \ldots, C_{|v|-1})$, where $C_i \subseteq \Sigma^*$ for every $i \in \{1, \ldots, |v|-1\}$.
(tuple of gap constraints)
The embedding e *satisfies* gc *w.r.t.* v, if, for every $i \in [|v|-1]$, $\mathrm{gap}_e(u,i) \in C_i$.

$u = \mathtt{a\ b\ c\ b\ c\ a\ b\ c\ a\ b\ a\ c}$

$v = \mathtt{c\ a\ b\ a}$

$$\mathrm{gap}_e(u,1) = \mathtt{bc} \qquad C_1 = \{\mathtt{ab}, \mathtt{ac}, \mathtt{bc}, \mathtt{bcabc}\}$$

$$\mathrm{gap}_e(u,2) = \varepsilon \qquad C_2 = \{\varepsilon, \mathtt{a}, \mathtt{b}\}$$

$$\mathrm{gap}_e(u,3) = \mathtt{c} \qquad C_3 = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$$

Gap constraints

$gc = (C_1, \dots, C_{|v|-1})$, where $C_i \subseteq \Sigma^*$ for every $i \in \{1, \dots, |v|-1\}$.
(tuple of gap constraints)
The embedding e *satisfies* gc *w.r.t.* v, if, for every $i \in [|v|-1]$, $\mathrm{gap}_e(u,i) \in C_i$.

$u =$ a b c b c a b c a b a c

$v =$ c a b a

$\mathrm{gap}_e(u, 1) = \texttt{bcabc}$

$\mathrm{gap}_e(u, 2) = \varepsilon$

$\mathrm{gap}_e(u, 3) = \varepsilon$

$C_1 = \{\texttt{ab}, \texttt{ac}, \texttt{bc}, \texttt{bcabc}\}$

$C_2 = \{\varepsilon, \texttt{a}, \texttt{b}\}$

$C_3 = \{\texttt{a}, \texttt{b}, \texttt{c}\}$

Gap constraints

$gc = (C_1, \ldots, C_{|v|-1})$, where $C_i \subseteq \Sigma^*$ for every $i \in \{1, \ldots, |v| - 1\}$.
(tuple of gap constraints)
The embedding e *satisfies* gc *w.r.t.* v, if, for every $i \in [|v| - 1]$, $\mathrm{gap}_e(u, i) \in C_i$.

Gap Constraints

Notations

$v \preceq_{gc} u$: $v \preceq_e u$ for some embedding e satisfying gc.
(v is a gc-subsequence of u)

Gap Constraints

Notations

$v \preceq_{gc} u$: $v \preceq_e u$ for some embedding e satisfying gc.
(v is a gc-subsequence of u)

$$Subseq(gc, u) = \left\{ v \in \Sigma^{|gc|+1} \mid v \preceq_{gc} u \right\}.$$

Gap Constraints

Notations

$v \preceq_{gc} u$: $v \preceq_e u$ for some embedding e satisfying gc.
(v is a gc-subsequence of u)

$$Subseq(gc, u) = \{v \in \Sigma^{|gc|+1} \mid v \preceq_{gc} u\}.$$

Example

$gc = (C_1, C_2)$ with $C_1 = C_2 = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

$$Subseq(gc, \mathtt{abbacbab}) = \{\mathtt{abc}, \mathtt{bab}, \mathtt{bca}, \mathtt{abb}\}.$$

Problems for Gap Constrained Subsequences

Matching

Input:	$u, v \in \Sigma^*$, $k := \lvert v \rvert$, $(k-1)$-tuple gc of gap constraints.
Question:	$v \preceq_{gc} u$?

Problems for Gap Constrained Subsequences

Matching

Input: $u, v \in \Sigma^*$, $k := |v|$, $(k-1)$-tuple gc of gap constraints.
Question: $v \preceq_{gc} u$?

Analysis Problems

Input: $u, v \in \Sigma^*$, $(k-1)$-tuple gc of gap constraints.
Questions: $Subseq(gc, u) = Subseq(gc, v)$? (Equivalence)
$Subseq(gc, u) \subseteq Subseq(gc, v)$? (Containment)
$Subseq(gc, u) = \Sigma^k$? (Universality)

Problems for Gap Constrained Subsequences

Matching

Input: $u, v \in \Sigma^*$, $k := |v|$, $(k-1)$-tuple gc of gap constraints.
Question: $v \preceq_{gc} u$?

Analysis Problems

Input: $u, v \in \Sigma^*$, $(k-1)$-tuple gc of gap constraints.
Questions: $Subseq(gc, u) = Subseq(gc, v)$? (Equivalence)
$Subseq(gc, u) \subseteq Subseq(gc, v)$? (Containment)
$Subseq(gc, u) = \Sigma^k$? (Universality)

Remark

k always means the length of subsequences!
(I. e., $gc = (C_1, \ldots, C_{k-1})$.)

Types of Gap-Constraints

Gap constraints $gc = (C_1, \ldots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.

Types of Gap-Constraints

Gap constraints $gc = (C_1, \ldots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.

... *length constraints* if every $C_i = \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$ for some $L^-(i), L^+(i) \in \mathbb{N} \cup \{0, +\infty\}$.

Types of Gap-Constraints

Gap constraints $gc = (C_1, \ldots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.

... *length constraints* if every $C_i = \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$ for some $L^-(i), L^+(i) \in \mathbb{N} \cup \{0, +\infty\}$.

... *reg-len constraints* if every $C_i = C_i' \cap \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$.

Types of Gap-Constraints

Gap constraints $gc = (C_1, \dots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.
(Represented as: DFAs.)

... *length constraints* if every $C_i = \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$ for some $L^-(i), L^+(i) \in \mathbb{N} \cup \{0, +\infty\}$.

... *reg-len constraints* if every $C_i = C'_i \cap \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$.

Types of Gap-Constraints

Gap constraints $gc = (C_1, \ldots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.
(Represented as: DFAs.)

... *length constraints* if every $C_i = \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$ for some $L^-(i), L^+(i) \in \mathbb{N} \cup \{0, +\infty\}$.
(Represented as: $(L^-(i), L^+(i)) \in \mathbb{N}^2$ in binary encoding.)

... *reg-len constraints* if every $C_i = C_i' \cap \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$.

Types of Gap-Constraints

Gap constraints $gc = (C_1, \ldots, C_{k-1})$ are ...

... *regular constraints* if every C_i is a regular language.
(Represented as: DFAs.)

... *length constraints* if every $C_i = \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$ for some $L^-(i), L^+(i) \in \mathbb{N} \cup \{0, +\infty\}$.
(Represented as: $(L^-(i), L^+(i)) \in \mathbb{N}^2$ in binary encoding.)

... *reg-len constraints* if every $C_i = C_i' \cap \{v \in \Sigma^* \mid L^-(i) \leq |v| \leq L^+(i)\}$.
(Represented as: $((L^-(i), L^+(i)), A_i')$, where A_i' is DFA for C_i'.)

Research Questions

Main Research Question

Investigate the complexity of matching and analysis problems in the presence of gap constraints.

Research Questions

Main Research Question

Investigate the complexity of matching and analysis problems in the presence of gap constraints.

Conditional Lower Bound Hypotheses

- *Exponential Time Hypothesis* (ETH):
  No $2^{o(n)}\,\mathrm{poly}(n+m)$ algo. for 3-Satisfiability.
- *Strong Exponential Time Hypothesis* (SETH):
  $\forall \epsilon > 0 \exists k$: No $O(2^{n(1-\epsilon)}\,\mathrm{poly}(n))$ algo. for k-Satisfiability.
- *Orthogonal Vectors Hypothesis* (OVH):
  $\forall \epsilon > 0$: No $\mathrm{O}(n^{2-\epsilon}\,\mathrm{poly}(d))$ algo. for OV.

Complexity Bounds for Matching

Complexity Bounds for Matching

Upper Bound

The matching problem can be solved in time $O(|u||gc|)$.

Complexity Bounds for Matching

Upper Bound

The matching problem can be solved in time $O(|u||gc|)$.

Remark

For gap-constraints gc, let $|gc|$ be

- the total number of the DFAs states, if gc are regular constraints or reg-len constraints.
- the number of constraints with $L^+(i) - L^-(i) > 0$, if gc are length constraints.

Complexity Bounds for Matching

Conditional Lower Bound

The matching problem cannot be solved in time $O(|u|^h|gc|^g)$ with $g + h = 2 - \epsilon$ for some $\epsilon > 0$ unless OVH fails.

Complexity Bounds for Matching

Conditional Lower Bound

The matching problem cannot be solved in time $O(|u|^h |gc|^g)$ with $g + h = 2 - \epsilon$ for some $\epsilon > 0$ unless OVH fails.

For length constraints: this even holds for $|\Sigma| = 4$ and length constraints $(0, \ell)$ with $\ell \leq 6$.

For regular constraints: this even holds for $|\Sigma| = 4$ and all regular constraints are expressed by constant size DFAs.

Complexity Bounds for Matching

Conditional Lower Bound

The matching problem cannot be solved in time $O(|u|^h |gc|^g)$ with $g + h = 2 - \epsilon$ for some $\epsilon > 0$ unless OVH fails.

For length constraints: this even holds for $|\Sigma| = 4$ and length constraints $(0, \ell)$ with $\ell \leq 6$.

For regular constraints: this even holds for $|\Sigma| = 4$ and all regular constraints are expressed by constant size DFAs.

Proof Sketch

Fine-grained reduction from the *orthogonal vectors* problem (OV).

Complexity Bounds for Analysis Problems

Remark

We only discuss the **non-universality** problem with length constraints.
$\rightarrow$ We are checking $Subseq(gc, u) \neq \Sigma^k$.

All results hold analogously for non-equivalence and non-containment.

Complexity Bounds for Analysis Problems

Remark

We only discuss the **non-universality** problem with length constraints.
$\rightarrow$ We are checking $Subseq(gc, u) \neq \Sigma^k$.

All results hold analogously for non-equivalence and non-containment.

Upper Bound

The non-universality problem can be solved in time $O(|\Sigma|^k |gc||u|)$.
(Achieved by a brute force algorithm.)

Complexity Bounds for Analysis Problems

Conditional Lower Bounds

For every fixed alphabet Σ with $|\Sigma| \geq 3$, the non-universality problem with length constraints can be solved in time $2^{\mathrm{O}(k)}|gc||u|$. Moreover, it cannot be solved

- in subexponential time $2^{\mathrm{o}(k)}\,\mathrm{poly}(|u|, k)$ (unless ETH fails),
- in time $\mathrm{O}(2^{k(1-\epsilon)}\,\mathrm{poly}(|u|, k))$ (unless SETH fails).

These lower bounds hold even if all length constraints are $(1, 5)$.

Complexity Bounds for Analysis Problems

Conditional Lower Bounds

For every fixed alphabet Σ with $|\Sigma| \geq 3$, the non-universality problem with length constraints can be solved in time $2^{\mathrm{O}(k)}|gc||u|$. Moreover, it cannot be solved

- in subexponential time $2^{\mathrm{o}(k)}\,\mathrm{poly}(|u|, k)$ (unless ETH fails),
- in time $\mathrm{O}(2^{k(1-\epsilon)}\,\mathrm{poly}(|u|, k))$ (unless SETH fails).

These lower bounds hold even if all length constraints are $(1, 5)$.

Theorem

For every fixed alphabet Σ with $|\Sigma| = 2$, non-universality with length constraints is NP-complete even if each length constraint is $(0, 0)$ or $(3, 9)$.

Proof

Reduction from CNF-Sat.

Quick Overview

	Classical	Gap Constraints
Matching	$O(n)$	$O(\lvert u\rvert\lvert gc\rvert)$
Equivalence	$O(n)$	
Containment	$O(nm\lvert\Sigma\rvert)$	$\mathrm{O}(\lvert\Sigma\rvert^k\lvert gc\rvert\lvert u\rvert)$
Universality	$O(n)$	

Additional Results

Gap Length Equalities

In addition to gap constraints, we consider gap length equalities of the form "$|\mathrm{gap}_i| = |\mathrm{gap}_j|$" meaning $|\mathrm{gap}_e(w, i)| = |\mathrm{gap}_e(w, j)|$.

Additional Results

Gap Length Equalities

In addition to gap constraints, we consider gap length equalities of the form "$|\mathrm{gap}_i| = |\mathrm{gap}_j|$" meaning $|\mathrm{gap}_e(w, i)| = |\mathrm{gap}_e(w, j)|$.

(Or even more complex variant: $2|gap_7| + |gap_3| \leq |gap_2|$.)

Additional Results

Gap Length Equalities

In addition to gap constraints, we consider gap length equalities of the form "$|\mathrm{gap}_i| = |\mathrm{gap}_j|$" meaning $|\mathrm{gap}_e(w, i)| = |\mathrm{gap}_e(w, j)|$.

(Or even more complex variant: $2|gap_7| + |gap_3| \leq |gap_2|$.)

Theorem

Matching with gap constraints and gap length equalities is NP-complete even for $|\Sigma| = 2$ and gap constraints $C_i = \Sigma^*$.

Proof

Reduction from 3-CNF-SAT.

Multiplicities

Sets of Gap Constrained Subsequences with Multiplicities:

$Subseq(2, \texttt{abba}) = \{\texttt{aa}, \texttt{ab}, \texttt{ba}, \texttt{bb}\}$
$Subseq(2, \texttt{abab}) = \{\texttt{aa}, \texttt{ab}, \texttt{ba}, \texttt{bb}\}$

Multiplicities

Sets of Gap Constrained Subsequences with Multiplicities:

$Subseq(2, \texttt{abba}) = \{\texttt{aa}, \texttt{ab}, \texttt{ba}, \texttt{bb}\}$
$Subseq(2, \texttt{abab}) = \{\texttt{aa}, \texttt{ab}, \texttt{ba}, \texttt{bb}\}$

$Subseq^m(2, \texttt{abba}) = \{(\texttt{aa}, 1), (\texttt{ab}, 2), (\texttt{ba}, 2), (\texttt{bb}, 1)\}$
$Subseq^m(2, \texttt{abab}) = \{(\texttt{aa}, 1), (\texttt{ab}, 3), (\texttt{ba}, 1), (\texttt{bb}, 1)\}$

Multiplicities

Sets of Gap Constrained Subsequences with Multiplicities:

$Subseq(2, \mathtt{abba}) = \{\mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}\}$
$Subseq(2, \mathtt{abab}) = \{\mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}\}$

$Subseq^m(2, \mathtt{abba}) = \{(\mathtt{aa}, 1), (\mathtt{ab}, 2), (\mathtt{ba}, 2), (\mathtt{bb}, 1)\}$
$Subseq^m(2, \mathtt{abab}) = \{(\mathtt{aa}, 1), (\mathtt{ab}, 3), (\mathtt{ba}, 1), (\mathtt{bb}, 1)\}$

Theorem

The equivalence problem for gap-constrained subsequences with multiplicities can be decided in polynomial time.

Overview

The presented results and topics are also summarized in the survey paper presented at NCMA, 2022:

Combinatorial Algorithms for Subsequence Matching: A Survey
(MK, Tore Koß, Florin Manea, Stefan Siemer)

The presented results and topics are also summarized in the survey paper presented at NCMA, 2022:

Combinatorial Algorithms for Subsequence Matching: A Survey
(MK, Tore Koß, Florin Manea, Stefan Siemer)

Thank you!

Bonus

Theorem

The non-universality problem with length constraints cannot be solved in running time $O(f(k)\,\mathrm{poly}(|w|, k))$ for any computable function f (unless $\mathsf{FPT} = \mathsf{W}[1]$).