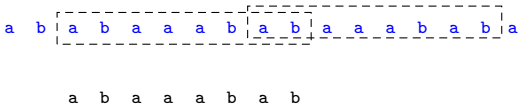
Jakub Radoszewski, jrad@mimuw.edu.pl
Project title: "Non-Standard Stringology: Algorithms and Combinatorics"

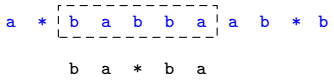
Different Models of Equivalence

The key problem of standard stringology is pattern matching. It is stated as follows: given two strings: a text t of length n and a pattern p of length m, find all occurrences of the pattern in the text (as a substring).



In this project we investigate non-standard models in stringology that stem from various applications. It is most convenient to use the simple pattern matching problem in their formulations (even though this problem has already been solved in all models).

Strings with Wildcards The pattern and the text can contain wildcard symbols (don't care symbols) that match any other symbol of the alphabet.



Weighted Sequences A weighted sequence (position weight matrix, PWM) specifies the probability of occurrence $\pi_i(a)$ for each position i and letter $a \in \Sigma$. The sum of probabilities $\pi_i(a)$ over all $a \in \Sigma$ for a given position i is 1.

a $\frac{1}{4}$	a 1	a $\frac{3}{4}$	a $\frac{1}{2}$	a 1
b $\frac{3}{4}$	b 0	b $\frac{1}{4}$	b $\frac{1}{2}$	b 0

Usually a threshold $\frac{1}{z}$ is specified. A string is said to match the weighted sequence if the probability of its occurrence is at least $\frac{1}{z}$.

The motivation comes from Computational biology. Primary uses:

- Motifs is biological sequences
- Phylogenetic trees
- Profiles
- ECG annotations

In the Weighted Pattern Matching problem, the input are: t – a weighted sequence of length n, p – a string pattern of length m, $\frac{1}{z}$ – a cut-off probability, Σ – the alphabet. The output are all positions i in t where p matches $t[i, i+m-1]$ with probability at least $\frac{1}{z}$.

$\frac{1}{z} = \frac{1}{8}$

t	a $\frac{1}{4}$	a 1	a $\frac{3}{4}$	a $\frac{1}{2}$	a 1
	b $\frac{3}{4}$	b 0	b $\frac{1}{4}$	b $\frac{1}{2}$	b 0
p		a	a	b	$\frac{3}{8} \geq \frac{1}{z}$ **YES**

Parameterized Model We say that two equal-length strings u and v over an alphabet Σ parameterized match (p-match) if there is a bijection $f : \Sigma \mapsto \Sigma$ such that $f(u) = v$.

For example, for $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$, the following two strings match:

$$\mathtt{aacabcdaaca} \approx \mathtt{ccdcbdaccdc}$$

as we have the bijection: $f(\mathtt{a}) = \mathtt{c}$, $f(\mathtt{b}) = \mathtt{b}$, $f(\mathtt{c}) = \mathtt{d}$, $f(\mathtt{d}) = \mathtt{a}$.

This problem is also generalized to the case when $\Sigma = \Sigma_S \cup \Sigma_P$, Σ_S are static symbols, and Σ_P are parameterized symbols. Then we say that two equal-length strings u and v over an alphabet $\Sigma = \Sigma_S \cup \Sigma_p$ parameterized match if there is a function $f : \Sigma \mapsto \Sigma$ that is constant on Σ_S and a bijection on Σ_P such that $f(u) = v$.

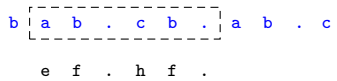
For example, for $\Sigma_S = \{\mathtt{.}, \mathtt{:}, \mathtt{,}\}$, $\Sigma_P = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$, the following two strings match:

$$\mathtt{ab.aabc:a,b} \approx \mathtt{bd.bbdc:b,d}$$

due to the bijection: $f(\mathtt{.}) = \mathtt{.}$, $f(\mathtt{:}) = \mathtt{:}$, $f(\mathtt{,}) = \mathtt{,}$, $f(\mathtt{a}) = \mathtt{b}$, $f(\mathtt{b}) = \mathtt{d}$, $f(\mathtt{c}) = \mathtt{c}$, $f(\mathtt{d}) = \mathtt{a}$.

The motivation for parameterized matching comes from Plagiarism detection and Refactoring of code.

In the Parameterized Pattern Matching problem, given two strings: a text t of length n and a pattern p of length m, we are to find all substrings of the text that p-match the pattern.

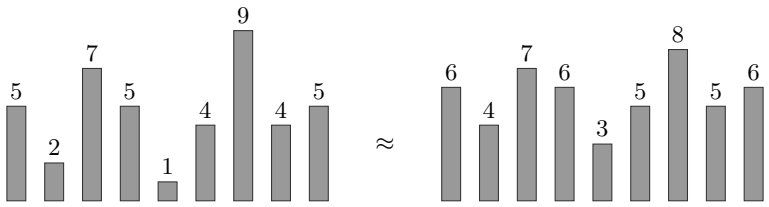


Order-Preserving Model We say that two equal-length strings u and v over an integer alphabet Σ order-preserving match (op-match) if there is a 1-1 increasing function $f : \Sigma \mapsto \Sigma$ such that $f(u) = v$.

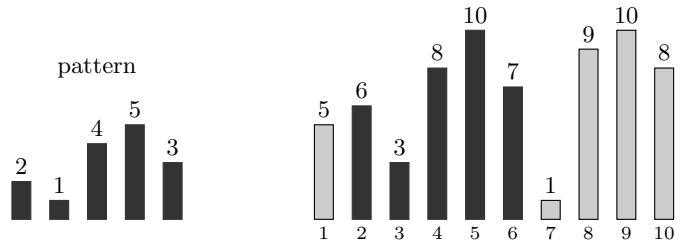
For example:

$$5\,2\,7\,5\,1\,4\,9\,4\,5 \approx 6\,4\,7\,6\,3\,5\,8\,5\,6$$

with the bijection: $f(1) = 3$, $f(2) = 4$, $f(4) = 5$, $f(5) = 6$, $f(7) = 7$, $f(9) = 8$.



In the Order-Preserving Pattern Matching, we want to find all such matches. E.g.:



The motivation comes from:

- Trend discovery in stock market
- Melody matching based on relative heights of pitches

Jumbled Model Two strings are said to jumbled match (histogram match; Abelian match) if the multisets of their letters are the same. For example:

$$\texttt{aabcbaab} \approx \texttt{ababacab}$$

(Here: $\{\texttt{a}, \texttt{a}, \texttt{a}, \texttt{a}, \texttt{b}, \texttt{b}, \texttt{b}, \texttt{c}\}$).

The motivation for this model is mainly combinatorial (first posed by P. Erdos).

Problem 0. Pattern Matching

As it has already been mentioned, the pattern matching problem is already well studied in all the considered models. Assume: n – text length, m – pattern length.

model	time complexity
standard	$O(n+m)$
wildcards	$O(n \log m)$
weighted strings	$O(n \log z)$
parameterized	$O(n+m)$
order-preserving	$O(n+m)$
jumbled	$O(n+m)$

It is worthwile to acquaint oneself with these results.

Problem 1. Indexing

Given a text of length n, preprocess it to answer multiple pattern-matching queries (m – query length). Index parameters: space, preprocessing time, query time.

The following table presents the known complexities of indices. For the jumbled index, we assume the binary alphabet.

model	space	preprocessing	query
standard	$O(n)$	$O(n)$	$O(m)$
wildcards	$O(n \log^k n)$	$O(n \log^k n)$	$O(m + 2^k \log \log n)$
weighted strings	$O(nz)$	$O(nz)$	$O(m)$
parameterized	$O(n)$	$O(n)$	$O(m)$
order-preserving	$O(n)$	$O(n \log \log n)$	$O(m)$
jumbled	$O(n)$	$O(n^{1.859})$	$O(m)$

Problem: Faster index construction Can we construct the index in the: weighted, order-preserving, or jumbled model faster? Also, investigate different variants of the index for wildcards.

Problem 2. Repetitions

A string v is called a square if $v = uu$ for some string u. The following table lists examples of squares in different equivalence models.

model	a square	comment
standard	121314 121314	just a square
parameterized	121314 424143	the bijection: $1 \to 4$, $2 \to 2$, $3 \to 1$, $4 \to 3$
order-preserving	121314 131516	the increasing bijection: $1 \to 1$, $2 \to 3$, $3 \to 5$, $4 \to 6$
Abelian	121314 123114	both words are permutations of the multiset $\{1, 1, 1, 2, 3, 4\}$

Problem: Combinatorial bounds We ask the question: "What is the maximum number of squares of respective types in a string of length n?" The following table presents the known results. Can we improve the bounds for non-standard models?

model	**bounds**
standard	$\Theta(n)$
parameterized	$\Theta(n)$ for a constant alphabet
order-preserving	$\Theta(n)$ for a constant alphabet
Abelian	$\Omega(n^{1.5})$ and $O(n^{11/6})$ non-equivalent squares