THEORIE-TAG 2023
A. Lin, G. Zetzsche (Eds.): Theorietag 2023, Kaiserslautern, 4. – 6.10.2023
RPTU & MPI-SWS Kaiserslautern, Technical Report, pp. 67–70.

Matching Patterns with Variables Under Simon's Congruence

Pamela Fleischmann[(A)] Sungmin Kim[(C)] Tore Koß[(B)]
Florin Manea[(B)] Dirk Nowotka[(A)] Stefan Siemer[(B)]
Max Wiedenhöft[(A)]

[(A)]Department of Computer Science, Kiel University, Germany
{fpa,dn,maw}@informatik.uni-kiel.de

[(B)]Department of Computer Science, University of Göttingen, Germany
{tore.koss,florin.manea,stefan.siemer}@cs.uni-goettingen.de

[(C)]Department of Computer Science, Yonsei University, Republic of Korea
rena_rio@yonsei.ac.kr

Abstract

We introduce and investigate a series of matching problems for patterns with variables under Simon's congruence and give a thorough picture of their computational complexity.

1. Introduction

A *pattern with variables* is a string $\alpha \in (\Sigma \cup \mathcal{X})^*$ consisting of *constant letters* (or *terminals*) from a finite alphabet $\Sigma = \{1, ..., \sigma\}$ of size $\sigma \geq 2$ and a potentially infinite set of *variables* $\mathcal{X}$ such that $\Sigma \cap \mathcal{X} = \emptyset$. Here, we assume σ to be bounded by a constant. A pattern is mapped by a *substitution* $h : (\Sigma \cup \mathcal{X})^* \to \Sigma^*$ which is a morphism that acts as the identity on Σ and maps each variable of $\mathcal{X}$ to a (potentially empty) string over Σ. For example, we can map the pattern $\alpha = xx\mathtt{abab}yy$ to the string of constants $\mathtt{aaaaababbb}$ by the substitution h with $h(x) = \mathtt{aa}$ and $h(y) = \mathtt{b}$ and by that $h(\alpha) = \mathtt{aaaaababbb}$. If a pattern α can be mapped to a string of constants w, we say that α *matches* w. The problem of deciding whether there exists a substitution h for a pattern α such that $h(\alpha) = w$ for a given word w is called the *(exact) matching problem*, `Match`. This heavily studied problem is `NP`-Complete in general [1], but a series of classes of patterns, defined by structural restrictions, for which `Match` is in `P` were identified [4]. Moreover, for most of the parameterised classes, `Match` is $W[1]$-hard [3] w.r.t. the structural parameters used to define the respective classes. Recently, Gawrychowski et. al. [7, 8] studied `Match` in an approximate setting. In general: given a pattern α and a word w, decide whether there exists a substitution h such that $h(\alpha)$ is similar to w w.r.t. some similarity measure. Thus, it seems natural to consider other string-equivalence relations as similarity measures. Here, we consider an approximate variant of `Match` using Simon's congruence $\sim_k$ [13].

Matching under Simon's Congruence: $\texttt{MatchSimon}(\alpha, w, k)$
Input: Pattern α, $|\alpha| = m$, word w, $|w| = n$, and number $k \in [n]$.
Question: Is there a substitution h with $h(\alpha) \sim_k w$?

A string u is a *subsequence* of a string w if u results from w by deleting some letters of w. Let $\mathbb{S}_k(w)$ be the set of all subsequences of a given string w up to length $k \in \mathbb{N}_0$. Two strings v and v' are k-Simon congruent iff $\mathbb{S}_k(v) = \mathbb{S}_k(v')$ [13]. Then, we write $v \sim_k v'$. As a similarity measure for strings, $\sim_k$ was optimally solved in [2, 6]. Thus, it seems natural to consider, in a general setting, the problem of checking whether one can map a given pattern α to a string which is similar to w w.r.t. $\sim_k$. One of the congruence-classes of Σ^* w.r.t. $\sim_k$ received much attention: the class of k-subsequene universal words [11, 2] which are those words which contain all k-length words as subsequences. Here, we consider the following problem, where $\iota(w)$ (universality index of w) is the largest integer ℓ for which w is ℓ-subsequence universal.

Matching a Target Universality: $\texttt{MatchUniv}(\alpha, k)$
Input: Pattern α, $|\alpha| = m$, and $k \in \mathbb{N}_0$.
Question: Is there a substitution h with $\iota(h(\alpha)) = k$?

Note that `MatchUniv` can be formulated in terms of `MatchSimon`. One very important difference, though, is that we are not explicitly given a target word w but instead, we are given the number k which represents the target more compactly (using only $\log k$ bits).

A well-studied extension of `Match` is the satifiability problem for word equations (e.g. see [10]). Here, we extend `MatchSimon` to the problem of solving word equations under $\sim_k$:

Word Equations under Simon's Congruence: $\texttt{WESimon}(\alpha, \beta, k)$
Input: Patterns α, β, $|\alpha| = m$, $|\beta| = n$, and $k \in [m+n]$.
Question: Is there a substitution h with $h(\alpha) \sim_k h(\beta)$?

We present a rather comprehensive picture of the problems' computational complexity, starting with `MatchUniv` and showing that it is `NP`-complete. Also, we present a series of structurally restricted classes of patterns for which it can be solved in polynomial time. Then, we discuss `MatchSimon` and show its `NP`-completeness. Finally, we discuss `WESimon` and its variants, characterise their computational complexity, and point to a series of future research directions.

2. The NP-Completeness of `MatchUniv` and `MatchSimon`

To show that `MatchUniv` is `NP`-hard, we reduce the `NP`-complete problem `3CNFSAT` (see [9, 5]) to `MatchUniv`. The idea is to construct several gadgets which allow us to encode a `3CNFSAT`-instance φ as a `MatchUniv` instance (α, k). Thus, we can find a substitution h for the instance (α, k) such that $\iota(h(\alpha)) = k$ iff φ is satisfiable. We recall `3CNFSAT`.

3-Satisfiability for formulas in conjunctive normal form, `3CNFSAT`.
Input: Clauses $\varphi := \{c_1, c_2, \ldots, c_m\}$, where $c_j = (y_j^1 \vee y_j^2 \vee y_j^3)$ for $1 \leq j \leq m$, and y_j^1, y_j^2, y_j^3 from a finite set of boolean variables $X := \{x_1, x_2, \ldots, x_n\}$ and their negations $\bar{X} := \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$.
Question: Is there an assignment for X, which satisfies all clauses of φ?

Further, we get `NP`-containment by using a slight variation of *subsequence universality signatures* [12] such that the maximal length of certificates is polynomial in the input.

Theorem 2.1 `MatchUniv` *is* `NP`*-complete.*

By restricting the input patterns, we get two classes of patterns such that MatchUniv can be solved in polynomial time.

Proposition 2.2 MatchUniv$(\alpha, k) \in$ P *if there exists a variable that occurs only once in* α. *So,* MatchUniv$(\alpha, k) \in$ P *for regular patterns (see e.g. [4])* α. *Also,* MatchUniv$(\alpha.k) \in$ P *if* $|$var$(\alpha)|$ *is constant.*

Further, we discuss the MatchSimon problem. In case of MatchSimon we are given a pattern α, a word w, and a natural number $k \leq |w|$ and we want to check the existence of a substitution h such that $h(\alpha) \sim_k w$. We immediately get that MatchSimon is NP-hard, because MatchSimon$(\alpha, w, |w|)$ is equivalent to Match(α, w) and Match is NP-complete. Notice that this result followed much easier than the corresponding lower bound for MatchUniv because in MatchSimon we only ask for $h(\alpha) \sim_k w$ and allow $h(\alpha) \sim_{k+1} w$, while in MatchUniv $h(\alpha)$ has to be strict k-universal but not $(k+1)$-universal. Thus, we consider the following problem.

Matching under Strict Simon's Congruence: MatchStrictSimon(α, w, k)
Input: Pattern α, $|\alpha| = m$, word w, $|w| = n$, and $k \in [n]$.
Question: Is there a substitution h with $h(\alpha) \sim_k w$ and $h(\alpha) \not\sim_{k+1} w$?

Adapting the reduction used for Theorem 2.1, we can show that MatchStrictSimon is NP-hard. For the NP-containment, we know that it is enough to only consider strings of length up to $O((k+1)^\sigma)$ as potential substitutions of the variables in a substitution h for a pattern α. Longer strings can be replaced with shorter ones which are $\sim_k$-congruent with the same impact on the sets $\mathbb{S}_k(h(\alpha))$.

Theorem 2.3 MatchSimon *and* MatchStrictSimon *are* NP-*complete.*

If the patterns are regular, note that MatchSimon and MatchStrictSimon are in P.

Proposition 2.4 MatchSimon(α, w, k),MatchStrictSimon$(\alpha, w, k) \in$ P *if* α *is regular.*

3. An Analysis of WESimon

Finally, we address the WESimon problem, where we are given two patterns α and β and a natural number k and we want to check the existence of a substitution h with $h(\alpha) \sim_k h(\beta)$.

Theorem 3.1 WESimon *is* NP-*complete.*

To avoid trivial cases arising for WESimon, we also consider a stricter variant of this problem which, in contrast to WESimon, is NP-hard in all cases.

Word Equations under Strict Simon's Congruence: WEStrictSimon(α, β, k)
Input: Patterns α, β, $|\alpha| = m$, $\beta = n$, and $k \in [m+n]$.
Question: Is there a substitution h with $h(\alpha) \sim_k h(\beta)$ and $h(\alpha) \not\sim_{k+1} h(\beta)$?

Lemma 3.2 WEStrictSimon *is* NP-*hard, even if both patterns contain variables.*

Regarding the NP-membership, if k is upper bounded by a polynomial function in $|\alpha| + |\beta|$, we get that WEStrictSimon $\in$ NP. Otherwise, the question of the NP-membership remains open.

Theorem 3.3 WEStrictSimon *is* NP-*complete for all* $k \leq |\alpha| + |\beta|$.

4. Conclusion

We considered the problem of matching patterns with variables under Simon's congruence. Specifically, we considered the three main problems MatchUniv, MatchSimon, WESimon, strict variations MatchStrictSimon and WEStrictSimon, and have given a comprehensive image of their computaitonal complexity. In general, these problems are NP-complete, but have interesting particular cases which are in P. Interestingly, our NP and P algorithms work in (nondeterministic) polynomial time only in the case of a constant input alphabet. A characterisation of the parameterised complexity of these problems w.r.t. the parameter σ might be interesting. Another paramter of interest could be the number of variables of the considered patterns. We conjecture that the problems are $W[1]$-hard with respect to both of these parameters.

References

[1] D. Angluin, Finding Patterns Common to a Set of Strings. *J. Comput. Syst. Sci.* **21** (1980) 1, 46–62.

[2] L. Barker, P. Fleischmann, K. Harwardt, F. Manea, D. Nowotka, Scattered Factor-Universality of Words. In: *DLT 2020, Proceedings*. LNCS 12086, Springer, 2020, 14–28.

[3] R. G. Downey, M. R. Fellows, *Parameterized Complexity*. Monographs in Computer Science, Springer, 1999.

[4] H. Fernau, F. Manea, R. Mercas, M. L. Schmid, Pattern Matching with Variables: Efficient Algorithms and Complexity Results. *ACM Trans. Comput. Theory* **12** (2020) 1, 6:1–6:37.

[5] M. R. Garey, D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[6] P. Gawrychowski, M. Kosche, T. Koss, F. Manea, S. Siemer, Efficiently Testing Simon's Congruence. In: *STACS 2021*. LIPIcs 187, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 34:1–34:18.

[7] P. Gawrychowski, F. Manea, S. Siemer, Matching Patterns with Variables Under Hamming Distance. In: *46th ISMFCS, MFCS 2021*. LIPIcs 202, 2021, 48:1–48:24.

[8] P. Gawrychowski, F. Manea, S. Siemer, Matching Patterns with Variables Under Edit Distance, Springer, 2022, 275–289.

[9] R. M. Karp, Reducibility Among Combinatorial Problems. The IBM Research Symposia Series, Plenum Press, New York, 1972, 85–103.

[10] M. Lothaire, *Combinatorics on Words*. Cambridge University Press, 1997.

[11] P. Schnoebelen, P. Karandikar, The height of piecewise-testable languages and the complexity of the logic of subwords. *Logical Methods in Computer Science* **15** (2019).

[12] P. Schnoebelen, J. Veron, On Arch Factorization and Subword Universality for Words and Compressed Words. In: *WORDS 2023, Proceedings*. Lecture Notes in Computer Science 13899, 2023, 274–287.

[13] I. Simon, Piecewise testable events, Springer, 1975, 214–222.