The Membership Problem for Pattern Languages in a Nutshell

We want to check whether or not given words follow a given pattern. Consider the example below:

Words:

(1) ABCBACCCACBBACCACBCBACCAB
(2) ABABABABBAABAB
(3) BCBABACBAABBAABBCBABAB
(4) CBAACBCBAABBAACBCBAA
(5) ABABABABABABABABAB
(6) ABABCBAAABABBA
(7) ABCCBABBABC
(8) ABABABABABABABABABAB
(9) CBBAABBAAAAAAAABBAAC
(10) ABCABCABCABCABCCABCABABCABC

Pattern:

🟥🟦🟨🟦🟩🟨🟩🟦🟨🟥

Positive:

(1)	AB	C	BACC	C	ACB	BACC	ACB	C	BACC	AB
(2)	AB	A	B	A	BA	B	BA	A	B	AB
(4)	CBAA	C	B	C	BAA	B	BAA	C	B	CBAA
(8)	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB
(9)	C	BB	AA	BB	AAA	AA	AAA	BB	AA	C
(10)	ABC	AB	C	AB	CABCAB	C	CABCAB	AB	C	ABC

Negative:

(3) BCBABACBAABBAABBCBABAB
(5) ABABABABABABABABAB
(6) ABABCBAAABABBA
(7) ABCCBABBABC

More formally, this is the problem to decide on the existence of a morphism between two words. For further literature on the complexity of the membership problem of pattern languages, on their expressive power, decidability questions and (algorithmic) learnability see, e. g., [1, 2, 4, 3, 5, 6].

References

[1] D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.

[2] A. Ehrenfeucht and G. Rozenberg. Finding a homomorphism between two words is NP-complete. *Information Processing Letters*, 9:86–88, 1979.

[3] D.D. Freydenberger and D. Reidenbach. Bad news on decision problems for patterns. *Information and Computation*, 208:83–96, 2010.

[4] A. Mateescu and A. Salomaa. Patterns. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, pages 230–242. Springer, 1997.

[5] D. Reidenbach. Discontinuities in pattern inference. *Theoretical Computer Science*, 397:166–193, 2008.

[6] D. Reidenbach and M. L. Schmid. A polynomial time match test for large classes of extended regular expressions. In *Proc. 15th International Conference on Implementation and Application of Automata, CIAA 2010*, volume 6482 of *Lecture Notes in Computer Science*, pages 241–250, 2011.