Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

The chop of languages ☆

Markus Holzer, Sebastian Jakobi, Martin Kutrib *

Institut für Informatik, Universität Giessen, Arndtstr. 2, 35392 Giessen, Germany

ARTICLE INFO

Article history:
Received 30 July 2016
Received in revised form 26 January 2017
Accepted 1 February 2017
Available online xxxx

Keywords:
Language operation
Closure properties
Descriptional complexity
Decidability
Chomsky hierarchy

ABSTRACT

We investigate chop operations, which can be seen as generalized concatenation. For several language families of the Chomsky hierarchy we prove (non)closure properties under chop operations and incomparability to the family of languages that are the chop of two regular languages. We also prove non-closure of that language family under Boolean operations and closure under reversal. Further, the representation of a regular language as the chop of two regular expressions can be exponentially more succinct than its regular expression. By considering the chop of two linear context-free languages we already obtain language families that have non-semi-decidable problems such as emptiness or finiteness.

1. Introduction

One of the most important algebraic structures in formal language theory is the free semigroup (or monoid) that naturally is defined *via* the operation of concatenation. The investigation of closure properties of families of formal languages under certain language operations such as, for example, homomorphism, inverse homomorphism, intersection with regular sets, and union led to the rich theory of abstract families of languages, which is the equivalent of abstract families of automata theory. For the general treatment of machines and languages we refer to Ginsburg [15].

The importance of language operations for the representation of language families is emphasized by the concept of regular expressions and variants. Regular expressions with the operations of union ($\cup$), concatenation ($\cdot$), and Kleene star (*), allow an appealing set-theoretic characterization of languages accepted by finite automata. Since these languages are closed under several more operations, the approach to add operations like intersection ($\cap$), complementation ($\sim$), or squaring (^{2}) does not increase the expressive power of regular expressions. However, the descriptional power, that is, the succinctness of such expressions can be increased. On the other hand, growing succinctness of the expressions, can increase the computational complexity of decision problems. This further motivates the investigation of operations on formal language families. Besides the aforementioned classical operations, also more involved ones were investigated in the literature, inspired by, for example, biology, logic, etc. For instance, the study of biological processes on DNA molecules turned out to be a host of language operations. The simplest are hairpin loop with inverted pointers (*hi*), loop with direct pointers (*ld*), and double loop with alternating direct pointers (*dlad*), which were studied in [8–10] in detail.

☆ This paper is a revised and expanded version of a paper presented at the *13th International Conference Automata and Formal Languages, Debrecen, Hungary, August 17–22, 2011.*

* Corresponding author.
E-mail addresses: holzer@informatik.uni-giessen.de (M. Holzer), sebastian.jakobi@informatik.uni-giessen.de (S. Jakobi), kutrib@informatik.uni-giessen.de (M. Kutrib).

http://dx.doi.org/10.1016/j.tcs.2017.02.002
0304-3975/

In this paper we consider an operation that was inspired by logic, to be more precise, by the duration calculus, which is an interval logic for real time systems. Recently it was shown in [1] that the model checking problem for the discrete duration calculus can be reduced to so-called (extended) *chop* expressions (with tests). Simply speaking a chop expression is nothing else than a regular expression, where the operations of concatenation and Kleene star are replaced by the chop operation and its iterated version. The *chop* or *fusion* of two words is built by overlapping some non-empty suffix and prefix of the first and second word, respectively, if these subwords coincide; otherwise the operation is not defined. Thus, the chop operation can be seen as a generalization of concatenation that allows one to verify whether the end of the first word overlaps with the beginning of the second word. The easiest form of fusing two words is by single letters. This operation and its descriptional complexity is studied in [21], where exponentially more succinct representations compared to ordinary regular expressions are obtained, while still having similar descriptional complexity bounds as concatenation and Kleene star (for finite automata). Here we investigate the more general variant of the chop operation, allowing the fusion of (non-empty) words in general. Some aspects of this operation were also studied in [7], where the operation is called self-assembly, and in [14], under the name overlap assembly. Since this operation has some "built-in nondeterminism," we also consider two "deterministic" chop operation variants, namely the max- and min-chop operation. Various other operations that are more or less closely related to the herein studied chop operations can be found in [5,11,12,25].

We study the closure properties of the language families of the Chomsky hierarchy, and of the deterministic context-free and linear context-free languages under the introduced chop operations. For regular languages, which we will focus on, the situation is a little bit involved. On the one hand, regular languages are closed under the nondeterministic variant of the chop operation, while both deterministic variants lead to even non-context-free languages, when applied to two appropriate regular languages. The induced language families are shown to be incomparable to linear context-free, deterministic context-free, context-free languages, and even Church–Rosser languages. Moreover, these incomparability results go hand in hand with the non-closure of these language families under the Boolean operations. We also prove the existence of an infinite hierarchy of language families built by successive chops of regular languages. Finally, we prove that the chop operations give rise to highly succinct representation of languages, even to non-recursive trade-offs, between different language families obtained by the application of a single chop operation on two linear context-free languages.

2. Definitions

We assume the reader is familiar with some basic notions of formal language theory as presented in [24]. Concerning our notations, we write Σ^* for the set of all words over the finite alphabet Σ. The empty word is denoted by λ, and $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$. For the length of a word w we write $|w|$. The reversal of a word w is denoted by w^R and the reversal of a language L is defined to be $L^R = \{ w^R \mid w \in L \}$. We use $\subseteq$ for inclusions and $\subset$ for strict inclusions. We call two languages L_1 and L_2 *equivalent*, $L_1 \equiv L_2$, if and only if they differ at most by the empty word, that is, if $L_1 \setminus \{\lambda\} = L_2 \setminus \{\lambda\}$. We write REG, LIN, CFL, CSL, RE, and CRL for the families of regular, linear context-free, context-free, context-sensitive, recursively enumerable, and Church–Rosser languages, respectively. For more on the family of Church–Rosser languages we refer to [30].

Next we turn to *chop operations*. The chop of two words $u, w \in \Sigma^*$ is defined as

$$u \odot w = \{ u'vw' \mid u = u'v,\ w = vw',\ \text{and } v \in \Sigma^+ \}.$$

Observe, that $u \odot w$ can be the empty set, since the suffix of u, which is also the prefix of w, has to be non-empty. In [7], this operation is called self-assembly, and is denoted by $\mathscr{S}(u, w)$. This chop operation is also a special case of the $\odot_N$ operation from [12], which requires the length of the overlap of the words to be at least N. Another related operation from [25] is $\otimes$, which requires the non-overlapping parts to be non-empty.

In addition, we define two deterministic variants of the chop operation, the max-chop and the min-chop operation, by choosing the non-empty suffix of the first word (the non-empty prefix of the second word) as long as possible or as short as possible, respectively. More precisely, we define $u \odot_{\max} w = u'vw'$, with $u = u'v$, $w = vw'$, and $v \in \Sigma^+$, where v is the longest suffix of u that is also a prefix of w. If there is no such suffix $u \odot_{\max} w$ is undefined. Similarly we define $u \odot_{\min} w = u'vw'$, with $u = u'v$, $w = vw'$, and $v \in \Sigma^+$, where v is the shortest suffix of u that is also a prefix of w. Again, if there is no such suffix $u \odot_{\min} w$ is undefined. An operation similar to $\odot_{\max}$, namely the short concatenation $\circ$, is investigated in [5], where the overlap may also be empty, which in this case results in the concatenation of the two words. For two languages L_1 and L_2, define

$$L_1 \odot L_2 = \bigcup_{u \in L_1, w \in L_2} u \odot w,$$

and for the generalization to the chop of two language families $\mathscr{L}_1$ and $\mathscr{L}_2$ we set $\mathscr{L}_1 \odot \mathscr{L}_2 = \{ L_1 \odot L_2 \mid L_1 \in \mathscr{L}_1 \text{ and } L_2 \in \mathscr{L}_2 \}$. Similarly one defines the max- and min-chop of two languages and of two language families. As in the case of words, the chop (max-chop, min-chop, respectively) of two languages can be the empty set.

In order to clarify our notation we give an example (see [5]).