Doctopic: Algorithms, automata, complexity and games ARTICLE IN PRESS TCS:11378

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

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Linear-space recognition for grammars with contexts

Mikhail Barash [a,b,∗], Alexander Okhotin [c]

[a] Turku Centre for Computer Science, Turku FI-20500, Finland
[b] Åbo Akademi University, Turku FI-20500, Finland
[c] St. Petersburg State University, 7/9 Universitetskaya nab., Saint Petersburg 199034, Russia

ARTICLE INFO

Article history:
Received 16 September 2015
Received in revised form 8 November 2017
Accepted 13 November 2017
Available online xxxx
Communicated by D. Perrin

Keywords:
Context-free grammars
Conjunctive grammars
Context-sensitive grammars
Space complexity
Parsing

ABSTRACT

Grammars with contexts are an extension of context-free grammars equipped with operators for referring to the left and the right contexts of a substring being defined. These grammars are notable for still having a cubic-time parsing algorithm, as well as for being able to describe some useful syntactic constructs, such as declaration before use. It is proved in this paper that every language described by a grammar with contexts can be recognized in deterministic linear space.

1. Introduction

Formal grammars model the syntax of languages, both natural and artificial. The basic and the most important model is known in the literature as a *context-free grammar*. These grammars define the syntax of longer strings by concatenating shorter strings whose properties have already been defined: for example, a rule $S \to \mathrm{NP}\ \mathrm{VP}$ states that a concatenation of any noun phrase with any verb phrase is a sentence. The problem of parsing a string according to a grammar—that is, testing whether a string is syntactically correct and producing its parse tree—can be solved by various efficient algorithms. The basic algorithm, known as the Cocke–Kasami–Younger algorithm, works in time $\mathcal{O}(n^3)$, where n is the length of the input. Another, more sophisticated algorithm by Valiant [32], works in time $\mathcal{O}(n^\omega)$, where ω, with $\omega < 3$, is the exponent in the time complexity of matrix multiplication. For certain subclasses of context-free grammars, parsing can be performed in time less than $\mathcal{O}(n^\omega)$; the most well-known of these subclasses are the LL(k) grammars and the LR(k) grammars, which can be parsed in linear time.

Context-free grammars can be further generalized by adding explicit operations of intersection and complementation of languages, leading to *conjunctive grammars* [16] and *Boolean grammars* [17], respectively. These grammars are capable of expressing many interesting languages that are beyond the capacity of ordinary grammars [16,17]. At the same time, grammars with contexts maintain many practical properties of ordinary grammars [16–18], such as a version of tabular parsing algorithm working in cubic time and the generalization of Valiant's parsing algorithm, working in the same time $\mathcal{O}(n^\omega)$ [21], as in the case of context-free grammars. Also there are subclasses with square-time and linear-time parsing [1,18].

∗ Corresponding author.
E-mail addresses: mikhail.barash@abo.fi (M. Barash), alexander.okhotin@spbu.ru (A. Okhotin).

https://doi.org/10.1016/j.tcs.2017.11.006
0304-3975/© 2017 Published by Elsevier B.V.

In both conjunctive and Boolean grammars, the applicability of a rule to a certain substring does not depend on the *context*, in which this substring occurs, and these grammars thus remain "context-free" in the general sense of the word. A further extension of conjunctive grammars with operators referring to the left and right contexts of the current substring was recently proposed by the authors under the name of *grammars with contexts* [3,4]. Such grammars can have rules of the form $A \to BC \,\&\, \lhd D \,\&\, \rhd H$, meaning that a string is defined by the nonterminal A, if this string can be represented as a concatenation of substrings defined by B and C, and it is preceded by a string of the form D and followed by a string of the form H. These grammars can conveniently describe such constructs as *declaration before use* [3], and a full-sized example of a grammar defining a typed programming language was recently given by the first author [2]; on the other hand, the only known conjunctive grammar for *declaration before use* [20, Example 3] is much less intuitive, and consequently less suitable for practical language specification.

There are all reasons to expect grammars with contexts to have greater expressive power than conjunctive grammars. For instance, the language of *declaration before or after use* can be described by a grammar with contexts [3,4], whereas no way of constructing a conjunctive grammar for this language is known. Similarly, there is a grammar with contexts for the language $\{\, a^n b^{kn} \mid k, n \geqslant 1 \,\}$ [5], which has also defied all attempts at description by a conjunctive grammar; it has only been proved that this language is not *linear conjunctive* [33]. Unfortunately, since no methods for proving non-representability of languages by conjunctive grammars have been discovered yet [20, Problem 1], at the moment, grammars in contexts cannot be proved to be strictly more powerful than conjunctive grammars.

In spite of allowing a richer set of operators in the rules, grammars with contexts still have a cubic-time parsing algorithm, presented by Rabkin [25]. If only one-sided contexts are allowed, then the parsing time can be reduced to $\mathcal{O}(\frac{n^3}{\log n})$ [3,22]. Square-time parsing is guaranteed for a subclass known as *linear grammars with one-sided contexts*, in which, furthermore, the use of concatenation is restricted to concatenating terminal symbols from the left and from the right [5], as in the classical *linear context-free grammars*.

Besides the time complexity of parsing, which is some low-degree polynomial for every sensible grammar model, there are other important computational complexity aspects, which reveal further differences between various kinds of grammars. Consider that for the classical linear grammars, parsing is an NL-complete problem [31]. For the full class of ordinary (context-free) grammars, an algorithm using space $\mathcal{O}((\log n)^2)$, at the expense of exponential time, was discovered by Lewis et al. [15]. Later, based on the same underlying idea, a parsing method using circuits of depth $\mathcal{O}((\log n)^2)$ was given by Brent and Goldschlager [8] and by Rytter [27], thus showing that all these languages belong to the complexity class NC. For LR grammars, Cook [10] gave a different $\mathcal{O}((\log n)^2)$-space algorithm that runs in polynomial time—this is the SC complexity class. Tradeoffs between time and space for LR grammars were further studied by von Braunmühl et al. [7].

In contrast, linear conjunctive grammars can describe some P-complete sets [11,19], which puts them one level higher in the hierarchy of complexity classes. For this reason, under the usual assumptions of the complexity theory, there a parsing algorithm for linear conjunctive grammars that would use space $(\log n)^{\mathcal{O}(1)}$ should not exist. This expected lower bound also applies to the full class of conjunctive grammars, and further to grammars with contexts.

The only known upper bound on the space complexity of conjunctive and Boolean grammars is that all languages they define lie in the deterministic linear space, DSPACE(n); no better upper bound is known even for linear conjunctive grammars [30]. The purpose of this paper is to establish the same upper bound on the space complexity of the languages defined by grammars with two-sided contexts.

2. Grammars with contexts

The class of formal grammars considered in this paper are *grammars with two-sided contexts*, which allow unrestricted use of conjunction, disjunction, concatenation and four types of context operators.

Definition 1 *(Barash and Okhotin [3,4]).* A grammar with two-sided contexts is a quadruple $G = (\Sigma, N, R, S)$, where

- Σ is the alphabet of the language being defined;
- N is a finite set of nonterminal symbols, which denote the properties of strings defined in the grammar;
- R is a finite set of grammar rules, each of the form

$$\begin{aligned} A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd \beta_1 \,\&\, \ldots \,\&\, \lhd \beta_m \,\&\, \trianglelefteq \gamma_1 \,\&\, \ldots \,\&\, \trianglelefteq \gamma_n \,\& \\ \&\, \trianglerighteq \kappa_1 \,\&\, \ldots \,\&\, \trianglerighteq \kappa_{m'} \,\&\, \rhd \delta_1 \,\&\, \ldots \,\&\, \rhd \delta_{n'}, \end{aligned} \tag{1}$$

  with $A \in N$, $k \geqslant 1$, $m, n, m', n' \geqslant 0$ and $\alpha_i, \beta_i, \gamma_i, \kappa_i, \delta_i \in (\Sigma \cup N)^*$;
- $S \in N$ is the *initial symbol*, which represents the syntactically well-formed sentences in the language.

Intuitively, according to such a grammar, *a substring w of a string uwv is described by A*, if there exists a rule (1), for which all of the following conditions hold.

- For each *base conjunct* $\alpha_i = X_1 \ldots X_\ell$, with $X_1, \ldots, X_\ell \in \Sigma \cup N$, the string w can be represented as a concatenation $w = w_1 \ldots w_\ell$ of any substrings described by $X_1, \ldots, X_\ell$, respectively.