Theoretical Computer Science 640 (2016) 41–51

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Computing abelian complexity of binary uniform morphic words

F. Blanchet-Sadri [a,*], Daniel Seita [b], David Wise [c]

[a] Department of Computer Science, University of North Carolina, P.O. Box 26170, Greensboro, NC 27402-6170, USA
[b] Computer Science Division, University of California, Berkeley, 387 Soda Hall, Berkeley, CA 94720-1776, USA
[c] Department of Mathematical Sciences, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15289, USA

A R T I C L E   I N F O

Article history:
Received 4 December 2015
Received in revised form 24 March 2016
Accepted 31 May 2016
Available online 6 June 2016
Communicated by D. Perrin

Keywords:
Formal languages
Algorithms
Analysis of algorithms
Abelian complexity
Morphisms
Morphic words
Automatic sequences

A B S T R A C T

Although a lot of research has been done on the factor complexity (also called subword complexity) of morphic words obtained as fixed points of iterated morphisms, there has been little development in exploring algorithms that can efficiently compute their abelian complexity. The factor complexity counts the number of factors of a given length n, while the abelian complexity counts that number up to letter permutation. We propose and analyze a simple $\mathcal{O}(n)$ algorithm for quickly computing the exact abelian complexities for *all* indices from 1 up to n, when considering binary uniform morphisms. Using our algorithm we also analyze the structure in the abelian complexity for that class of morphisms. Our main result implies, in particular, that the infinite word over the alphabet $\{-1, 0, 1\}$ constructed from the consecutive forward differences of the abelian complexity of a fixed point of a binary uniform morphism is in fact an automatic sequence with the same morphic length. Since the proof produces morphisms that typically contain many redundant letters, we present an efficient algorithm to eliminate them in order to simplify the morphisms and to see the patterns produced more clearly.

1. Introduction

The concept of *abelian complexity* is relatively new as compared to its classical counterpart of *factor complexity*. On the one hand, the factor complexity of a given infinite word w is a function $\rho_w : \mathbb{N} \to \mathbb{N}$ that maps an integer n to the number of distinct factors of w of length n. On the other hand, the abelian complexity of w is a function $\rho_w^{ab} : \mathbb{N} \to \mathbb{N}$ that maps an integer n to the number of distinct *Parikh vectors* of factors of w of length n. Parikh vectors, which appear in the literature under various names such as compomers [5,6], jumbled patterns [8,7], permutation patterns [9,18,24], commutative closures [19], content vectors [19], to name a few, are vectors that record the frequency of each letter in the factors. So the abelian complexity counts the number of distinct factors of a given length up to letter permutation. For a survey on abelian concepts such as abelian complexity as well as applications, we refer the reader to [11].

We focus on the abelian complexity of those words obtained as fixed points of iterated morphisms starting at some letter. For example, the *Thue–Morse word* is the fixed point starting at 0 of the uniform morphism $0 \mapsto 01, 1 \mapsto 10$ over the

* Corresponding author.
E-mail addresses: blanchet@uncg.edu (F. Blanchet-Sadri), seita@berkeley.edu (D. Seita), dwise@andrew.cmu.edu (D. Wise).

http://dx.doi.org/10.1016/j.tcs.2016.05.046
0304-3975/

binary alphabet $\{0, 1\}$, i.e.,

$$0110100110010110\cdots.$$

A lot of research has been done on the factor complexity of fixed points of morphisms, e.g., Frid [20] obtained an explicit formula for the factor complexity of the words obtained from a binary uniform morphism.

As to research done on the abelian complexity of fixed points of morphisms, there are two main approaches that have been taken so far. The first approach consists in deriving a formula. This is in general very difficult to do, so it has only been done for a few special cases over a binary or a ternary alphabet: Sturmian words [12], the Thue–Morse word [27], the Tribonacci word, i.e., the fixed point starting at 0 of the morphism $0 \mapsto 01, 1 \mapsto 02, 2 \mapsto 0$ [26], the paper-folding word [22], and quadratic Parry words, i.e., a morphism of the type $0 \mapsto 0^p1, 1 \mapsto 0^q$ with $p \geq q \geq 1$ or of the type $0 \mapsto 0^p1, 1 \mapsto 0^q1$ with $p > q \geq 1$, [2]. Constant and ultimately constant abelian complexity of infinite words has also been studied [13,28]. The second approach consists in sliding a window of size n on a sufficiently long prefix of a fixed point w of a morphism to count the number of distinct Parikh vectors. This is also in general very difficult to do as the required prefix's length grows to infinity as n goes to infinity, exhausting computer memory. Reference [30] combines these two approaches: it considers an infinite word w belonging to a subclass of Parry words, and instead of sliding a window of size n on a sufficiently long prefix of w, it finds a walk on a transition diagram of a discrete finite-state automaton constructed from w, a finite graph that is independent of n. Equivalently, it is shown how to find a transition function and an output function that allow the evaluation of the value $\rho^{ab}_w(n)$ in $\mathcal{O}(\log n)$ steps.

Both approaches being difficult, there has also been work on the *asymptotic* abelian complexity of some morphic words, e.g., the fixed point starting at 0 of the non-uniform morphism $0 \mapsto 012, 1 \mapsto 02, 2 \mapsto 1$ [3]. Such works are mainly concerned with the asymptotic behaviors of the abelian complexities rather than their specific values. The classification of the asymptotic growths of the abelian complexities of fixed points of binary morphisms has been undertaken and a complete classification in the case of binary uniform morphisms has been derived [4]. This classification had previously been done for the factor complexity of fixed points of morphisms [14–17,23].

There has been little development in exploring algorithms that can efficiently compute abelian complexity values. Our paper presents an efficient algorithm for binary uniform morphisms, and also exposes an interesting mathematical connection between a class of morphisms and the abelian complexities of their fixed points.

The contents of our paper are as follows: In Section 2, we discuss terminology related to the abelian complexity of fixed points of morphisms. In Section 3, we describe and analyze a simple $\mathcal{O}(n)$ algorithm for computing the abelian complexities $\rho^{ab}(n)$ for *all* indices from 1 up to n, when considering binary uniform morphisms. In Section 4, we discuss morphic words and prove some of their properties not shared by fixed points of morphisms. They are constructed from two morphisms σ and τ. In Section 5, using our algorithm, we analyze the structure in the abelian complexity of fixed points of binary uniform morphisms. We state and prove our main result which implies, in particular, that if φ is a binary uniform morphism with morphic length ℓ and the number of zeroes in $\varphi(0)$ minus the number of zeroes in $\varphi(1)$ is not equal to 1, then the infinite word over the alphabet $\{-1, 0, 1\}$ constructed from the forward differences of $\rho^{ab}_{\varphi^\omega(0)}$, where $\varphi^\omega(0)$ denotes the fixed point of φ starting at 0, is an ℓ-automatic sequence. In Section 6, we prove some results on self-similarity. In Section 7, we give an efficient algorithm to eliminate redundancies in automatic sequences. This algorithm is of interest because the morphisms σ and τ constructed in the proof of our main result typically contain many redundant letters, so it is nice to be able to simplify the morphisms to see the pattern produced more clearly. Finally in Section 8, we conclude with some remarks and open problems for future work.

2. Preliminaries

Let Σ denote an arbitrary alphabet. A *word* w over Σ is a (finite or infinite) sequence of characters from Σ. The character at position i of w is denoted by $w[i]$ (position labeling starts at 0) and the *factor* from position i to position j inclusive by $w[i..j]$. If j is non-inclusive, the factor is denoted by $w[i..j)$. Here $[i..j]$ (resp., $[i..j)$) denotes the set $\{i, i+1, \dots, j\}$ (resp., $\{i, i+1, \dots, j-1\}$). When w is finite, the number of characters in w is denoted by $|w|$ and is called the *length* of w. The *empty word* ε is the word of length 0 and Σ^* denotes the set of all finite words over Σ, including ε. Equipped with the concatenation of words, Σ^* forms a monoid with ε acting as the identity.

Define the *factor complexity* ρ_w of a word w over Σ to be the function that, for each positive integer n, returns the number of distinct factors of length n of w. Likewise, define the *abelian complexity* ρ^{ab}_w of a word w over Σ to be the function that, for each positive integer n, returns the number of abelian equivalence classes of factors of length n of w. Here two factors are *abelian equivalent* if one can be obtained from the other by a permutation of characters. If w is understood, we often drop it and simply write ρ, ρ^{ab} respectively. To compute $\rho^{ab}(n)$, we count the number of *Parikh vectors* for all length-n factors, where the Parikh vector of a factor v is the vector whose ith entry is the number of occurrences of the ith letter in v; e.g., the Parikh vector of *ababbca* is $(3, 3, 1)$.

A *morphism* φ over $\Sigma = \{0, 1, \dots, k-1\}$ is a function $\Sigma^* \to \Sigma^*$ such that if $u, v \in \Sigma^*$, we have $\varphi(uv) = \varphi(u)\varphi(v)$. We denote it as $\varphi = (x_0, x_1, \dots, x_{k-1})$, where $\varphi(i) = x_i$. The *fixed point* $\varphi^\omega(a)$ of morphism φ over Σ starting at $a \in \Sigma$ is the limit as $n \to \infty$ of $\varphi^n(a)$, if it exists. A morphism is *binary* if it is over a 2-letter alphabet. A morphism φ over Σ is *uniform* if $|\varphi(a)| = |\varphi(b)|$ for all $a, b \in \Sigma$ (it is *ℓ-uniform* if $|\varphi(a)| = \ell$ for all $a \in \Sigma$).