Discrete Mathematics and Theoretical Computer Science DMTCS vol. **12:4**, 2010, 109–126

Grammatical compression: compressed equivalence and other problems

Alberto Bertoni[1†] and Roberto Radicioni[2‡]

[1]*Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy*
[2]*Dipartimento di Informatica e Comunicazione, Università degli Studi dell'Insubria, Italy*

received 30th October 2009, *revised 1st June 2010*, *accepted 11th October 2010*.

In this work, we focus our attention to algorithmic solutions for problems where the instances are presented as straight-line programs on a given algebra. In our exposition, we try to survey general results by presenting some meaningful examples; moreover, where possible, we outline the proofs in order to give an insight of the methods and the techniques.
We recall some recent results for the problem PosSLP, consisting of deciding if the integer defined by a straight-line program on the ring Z is greater than zero; we discuss some implications in the areas of numerical analysis and strategic games. Furthermore, we propose some methods for reducing Compressed Word Problem from an algebra to another; reductions from trace monoids to the semiring of nonnegative integers are exhibited and polynomial time algorithms for compressed equivalence in monoids related to Dyck reductions are shown. Finally, we consider inclusion problems for context-free languages, proving how in some cases efficient algorithms for these problems benefit from the ability to work with compressed data.

Keywords: grammatical compression, computational complexity, algebra representations

1 Introduction

Consider the problem SQRT-Sum of deciding whether $\sum_{i=1}^{n} \sqrt{a_i} > t$, for integers $a_1, \dots, a_n, t$, and the problem Incl(D_A) of deciding whether $L \subseteq D_A$ for context-free languages L, where D_A is the Dyck language on the alphabet $A \cup \overline{A}$. Have these problems something in common? In this paper, we point out that efficient algorithms for these problems take advantage from the ability to work with compressed data.

Compressing means to give a succinct description of data that can be reconstructed by means of a decompression algorithm. Compression is ubiquitous. Signal coding benefits, among the others, from Huffman or Lempel-Ziv compression algorithms; for translating analog into digital data, the signal is

†Email: `bertoni@dsi.unimi.it`
‡Partially supported by Project M.I.U.R. PRIN 2007–2009: Mathematical aspects and forthcoming applications of automata and formal languages. Email: `roberto.radicioni@uninsubria.it`

compressed by sampling and decompressed by the reconstruction filter; basic unsupervised learning algorithms such Principal Component Analysis (PCA) can be interpreted in terms of optimal linear compression.

Probably, the most famous notion of compression is related to Kolmogorov complexity, a powerful measure of nonrandomness of data (Kolmogorov (1968)). Roughly speaking, Kolmogorov complexity $K(D)$ is the length of the smallest program printing D. Unfortunately, $K(D)$ is not computable, therefore, in a practical context, the notion of "program printing D" should be limited in some sense. For example, we can require that the program contains only assignment instructions. This restriction corresponds to consider *straight-line programs* (SLP): in this case, compressing D means to give a short SLP computing D. Such a representation is very flexible; for instance, in the case of strings, SLPs can be efficiently converted in other compressed representations, such as Lempel-Ziv factorization, and vice versa (Rytter (2003)).

In this work, we focus our attention to algorithmic solutions for problems where the instances are presented as SLPs on a given algebra. In our exposition, we try to survey general results by presenting some meaningful examples; moreover, where possible, we outline the proofs in order to give an insight of the methods and the techniques.

In Section 2, we introduce SLPs as a sort of "shorten representations" of terms of an algebra: the term can be viewed as a string and the SLP as a context-free grammar generating a language containing that unique string.

Given an algebra A and a relation $R \subseteq A^s$, a natural problem requires to decide, given s terms $t_1, \ldots, t_s$ respectively holding the elements $a_1, \ldots, a_s \in A$, whether $R(a_1, \ldots, a_s)$ is true or not; we call *Relation*(R,A) such a problem, and *Compressed Relation*(R,A) the analogous problem having as instances SLPs $\Phi_1, \ldots, \Phi_s$ that are compressed versions of terms $t_1, \ldots, t_s$. If R is the identity relation $=_A$, we call *Equivalence*(A) and *Compressed Equivalence*(A) the problems Relation($=_A$,A) and Compressed Relation($=_A$,A), respectively; in literature, such problems are also called *Word Problem* and *Compressed Word Problem*. Other interesting instances of Compressed Relation are Compressed Substring Problem (or Compressed Pattern Matching, Lifshits (2007)) and Compressed First-Order Unification (Gascón et al. (2009)).

In Section 3, we observe that the gap between Compressed Equivalence(A) and Equivalence(A) can be at most exponential depending on the considered algebra. We present some examples; for instance, in the free monoid the gap is polynomial, while in the semiring of finite languages with union and product the gap becomes provably exponential.

Compressed Relation(R,A) has been studied mostly for algebraic structures such as monoids or semirings. In Section 4, we recall methods and results for the problem PosSLP, i.e. *Compressed relation (R,A)* where A is the ring of integer numbers $\mathbf{Z}$ with the operations $+, -, \times$ and R is the relation "greater than 0". This problem is proved to be in the counting hierarchy CH (Allender et al. (2008/09)); we discuss some implications of this result in the area of numerical analysis and in that of strategic games. PosSLP is a touchstone for many problems such as SQRT-Sum (Garey et al. (1976); Allender et al. (2008/09)) and (Strong) ϵ-approximation of Nash equilibria (Nash (1951); Etessami and Yannakakis (2007)).

In Section 5, we discuss some methods useful to efficiently reduce Compressed Equivalence(A) to Compressed Equivalence(B), for some given algebras A and B.

First, we look for a notion of representation of an algebra into another preseving the complexity of Compressed Equivalence. As an example, we show a representation of free partially commutative monoids (Mazurkiewicz (1977); Cartier and Foata (1969)) in the semiring of integer numbers. A direct consequence is that, given a trace monoid M, Compressed Equivalence(M) can be solved in randomized polynomial time.

Secondly, we recall the notion of composition system (Gasieniec et al. (1996)), a flexible concept applied to efficiently solve Compressed Equivalence(M) for certain monoids M. As an application, we present a polynomial time algorithm for Compressed Equivalence(D_m), where D_m is the monoid with $2m$ generators obtained by Dyck reductions over m types of parentheses, a class of monoids studied in the context of XML-grammars (Berstel and Boasson (2002)).

Finally, in Section 6, we observe that, in some cases, apparently uncorrelated problems can benefit from the solution of Compressed Equivalence. For example, we consider the problem Incl(L_0) of deciding whether a context-free language is contained in a fixed language L_0, and we prove that, in some cases, it can be reduced to Compressed Equivalence(A) for some monoid A (Bertoni et al. (2009)). As a consequence, we point out that Incl(D_A) can be solved in polynomial time for every Dyck language D_A.

2 Straight-line Programs and Grammatical Compression

Given a finite set Σ of operation symbols σ with fixed arity $m(\sigma)$, a Σ-algebra A is a pair $\langle A, \Sigma_A\rangle$, where A (with an abuse of notation) is the underlying set and $\sigma_A : A^{m(\sigma)} \longrightarrow A$ are the operations. If $m(\sigma) = 0$, then σ_A is called a constant. In the following, for the sake of simplicity, we will omit the constants whenever the context is clear.

Fixed a set X of variables, t is a Σ-term with variables X (or a Σ-expression) of A if either $t = x$, for some $x \in X$, or $t = \sigma t_1 \cdots t_{m(\sigma)}$, for some $\sigma \in \Sigma$ and some Σ-terms $t_1, \ldots, t_{m(\sigma)}$. Σ-expressions can be seen as the words of the fixed point of the following language equation with unique (context-free) solution

$$T_\Sigma(X) = X \cup \bigcup_{\sigma\in\Sigma} \sigma T_\Sigma(X)^{m(\sigma)}.$$

The free Σ-algebra T_Σ has as underlying set $T_\Sigma = T_\Sigma(\emptyset)$ and the operations are given by $\sigma_A(t_1, \ldots, t_{m(\sigma)}) = \sigma t_1 \cdots t_{m(\sigma)}$; we usually refer to elements of T_Σ as Σ-terms. It is well known that, for every Σ-algebra A, there is a unique morphism $\tau_A : T_\Sigma \longrightarrow A$. Moreover, we can consider a Σ-expression e with variables $\{x_1, \ldots, x_g\}$ as a function $e : T_\Sigma^g \longrightarrow T_\Sigma$, defined as follows. Let $\underline{t} = (t_1, \ldots, t_g) \in T_\Sigma^g$. Then,

$$e(\underline{t}) = \begin{cases} e & \text{if } e \in T_\Sigma; \\ t_i & \text{if } e = x_i; \\ \sigma e_1(\underline{t}) \cdots e_{m(\sigma)}(\underline{t}) & \text{if } e = \sigma e_1 \cdots e_{m(\sigma)} \end{cases}$$

Given a Σ-algebra A, a *straight-line program* (SLP) Φ is a sequence of n assignments of the form $X_i = \sigma_A(X_{j_1}, \ldots, X_{j_{m(\sigma)}})$, for some $\sigma \in \Sigma$, $1 \le i \le n$ and $j_1, \ldots, j_{m(\sigma)} < i$. We call n the size of Φ and denote it by $|\Phi|$. Every variable X_i is associated with $\text{eval}_A(X_i) \in A$, inductively defined by:

$$\text{eval}_A(X_i) = \sigma_A(\text{eval}_A(X_{j_1}), \ldots, \text{eval}_A(X_{j_{m(\sigma)}})).$$

Finally, we set $\text{eval}_A(\Phi) = \text{eval}_A(X_n)$. In the following, we call X_n the output variable.

A SLP Φ is basically a DAG, but can also be viewed as a context-free grammar G_Φ, where $X_1, \ldots, X_n$ are the metasymbols, $X_i \longrightarrow \sigma X_{j_1} \cdots X_{j_{m(\sigma)}}$ are the production rules and X_n is the axiom. G_Φ generates a unique string $t_\Phi \in T_\Sigma$ such that $\mathrm{eval}_A(\Phi) = \tau_A(t_\Phi)$. The name *grammatical compression* derives from the fact that Φ can be viewed as a context-free grammar generating a unique term representing t_Φ.

In the case A is a monoid, a slightly different approach should be used. If the monoid A is generated by the constants $\Sigma = \{c_1, \ldots, c_l\}$, then the free monoid Σ^* should be considered instead of the free algebra. In this case, a SLP Φ turns out to be a Chomsky Normal Form grammar generating a unique word $t_\Phi \in \Sigma^*$.

For both free algebras and free monoids a SLP Φ can be an exponentially succinct representation of t_Φ.

Example 2.1 *Consider the following SLP Φ_n on the semiring $\langle \mathbf{N}, +, \cdot, 0, 1\rangle$:*

$$X_0 = 1;\ \ X_1 = X_0 + X_0;\ \ X_2 = X_1 \cdot X_1;\ \ldots\ X_n = X_{n-1} \cdot X_{n-1}.$$

It computes the number $2^{2^{n-1}}$, representable in binary notation by $2^{n-1} + 1$ bits. The term t_{Φ_n} is the unique word generated by the context-free grammar associated with Φ_n and the length of t_{Φ_n} is $2^{n+1} - 1$.

Example 2.2 *Consider the SLP Φ_n on the free monoid $\{a, b\}^*$ given by the CNF grammar with the rules:*

$$X_0 \longrightarrow a,\ X_1 \longrightarrow b,\ X_i \longrightarrow X_{i-1}X_{i-2} \quad \textit{for} \quad 1 < i \leq n.$$

It generates the n-th Fibonacci's word; in this case, $|t_{\Phi_n}| \sim \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+1}$.

Finally, we recall a construction due to Lohrey (Lohrey (2006)) useful for studying the computational complexity of some compressed word problems.

Example 2.3 *A well known NP-complete problem is* SUBSETSUM *(Garey and Johnson (1979)): given a vector $\underline{w}$ of integers and a target integer t, decide if there is at least one selection of entries in $\underline{w}$ the sum of which is t, i.e., there exists $\underline{x} \in \{0, 1\}^n$ such that $\underline{x} \cdot \underline{w} = t$.*

Let $I = (\underline{w}, t)$ be an instance of SUBSETSUM, *with $\underline{w} \in \mathbf{N}^n$ and define $s = \underline{1} \cdot \underline{w}$, where $\underline{1}$ is the vector with all entries* 1. *The* Lohrey strings *of I are the two words $\xi_{ab}(I)$, $\xi'_{ab}(I)$ in $\{a, b\}^*$ given by*

$$\xi_{ab}(I) \;=\; (a^{s-t}ba^t)^{2^n} \qquad \xi'_{ab}(I) \;=\; \prod_{\substack{x\in\{0,1\}^n \\ b(x)=0..2^n-1}} (a^{\underline{x}\cdot\underline{w}}ba^{s-\underline{x}\cdot\underline{w}})$$

Informally, $\xi_{ab}(I)$ encodes t by 2^n blocks of length $s+1$ made of a in all places except in the $(t+1)$-th. On the other hand, $\xi'_{ab}(I)$ encodes the sums of all the possible subsets of $\underline{w}$ by setting b only in position $\underline{x} \cdot \underline{w}$ in the x-th block, for every $x \in \{0, 1\}^n$.

Lohrey strings admit an exponential compression rate (Lohrey (2006)):

Lemma 1 *Let $I = (\underline{w}, t)$ be an instance of* SUBSETSUM *with $\underline{w} \in \mathbf{N}^n$. Then, the length of Lohrey strings $\xi_{ab}(I)$, $\xi'_{ab}(I)$ is $(s + 1)2^n$, where $s = \underline{1} \cdot \underline{w}$ and they can be represented by SLPs on the free monoid $\{a, b\}^*$ of size $n^{O(1)}$.*

The previous examples exhibit SLPs with exponential compression rate with respect to their corresponding term. Now, we observe that the compression rate is *at most* exponential:

Fact 1 *Consider a Σ-algebra A and let $m = \max\{m(\sigma)|\sigma \in \Sigma\}$. If $m > 1$, then $|t_\Phi| = m^{O(|\Phi|)}$ for every SLP Φ.*

A natural problem is, given as input a Σ-term (a word) t in the Σ-algebra T_Σ (resp. in the free monoid Σ^*), to find a smallest SLP generating t. Even if the study of these problems is not a topic of this work, we notice that the situation is much different in the case of free algebras and that of free monoids. In fact, in the first case, the problem is solvable in polynomial time. In the second, the problem is NP-hard to approximate within a constant (Lehman and Shelat (2002)), while it is approximable up to a logarithmic factor (Rytter (2003)). In particular, this last result is obtained by constructively proving that, given a word w, the size of its Lempel-Ziv factorization and the smallest size of a SLP generating w are equal up to a logarithmic factor in the length of w.

3 Problems on Compressed Terms

The computation of a SLP Φ over an algebra A can be conceptually factorized in two steps: first, the term t_Φ in the free algebra is considered and then it is evaluated obtaining $\tau_A(t_\Phi)$. The first step leads us to consider Φ as a sort of compressed representation of t_Φ. In this context, some natural problems on algebras can be presented either in standard or compressed version: in the first case, the input of the problem is represented by terms, in the second by SLPs. Fixed a Σ-algebra A and a relation $R \subseteq A^s$, we consider the problem:

PROBLEM: *Compressed relation (R,A) (*CompRel(R,A)*)*
INSTANCE: s SLPs $\Phi_1, \dots, \Phi_s$ on A;
QUESTION:is $R(\text{eval}_A(\Phi_1), \dots, \text{eval}_A(\Phi_s))$ true?

This problem can be interpreted as the compressed version of the problem to decide, given s terms $t_1, \dots, t_s \in T_\Sigma$, whether $R(\tau_A(t_1), \dots, \tau_A(t_s))$ is true; we call Rel(R,A) such a problem.

A natural relation defined for every algebra A is the identity relation $=_A$; we call *Equivalence (A)* (Eq(A)) and *Compressed Equivalence (A)* (CompEq(A)) the problems Rel($=_A$,A) and CompRel($=_A$,A), respectively; such problems are also called *Word Problem* and *Compressed Word Problem* in literature. By Fact 1, the gap between the complexities of Eq(A) and CompEq(A) can be at most exponential, and it depends on the considered algebra. In the following we present four examples. The first two are borderline examples; in the first, the complexity gap is polynomial while in the second it becomes provably exponential. The last two examples represent intermediate situations.

Example 3.1 *Consider the monoid $\langle \Sigma^*, \cdot \rangle$ over the alphabet Σ. Then, both* Eq(Σ^*) *and* CompEq(Σ^*) *are in P.*

Eq(Σ^*) is trivially in P, while CompEq(Σ^*) has been proved to be in P by Plandowski (1994). In section 5, we show a randomized polynomial time algorithm for CompEq(Σ^*).

Example 3.2 *Consider the algebra of finite languages FL_Σ over the alphabet $\Sigma = \{0, 1\}$ equipped with the operations union $\cup$ and language product $\cdot$. Then,* Eq(FL_Σ) *is coNP-complete, while* CompEq(FL_Σ) *is coNEXPTIME-complete.*

This fact, quoted in Hunt III et al. (2000), is an easy consequence of a result of Meyer and Stockmeyer (1972). First of all, we observe that the problem of deciding if $w \in \text{eval}_{FL_\Sigma}(\Phi)$, where $w \in \Sigma^*$ and Φ is a SLP over FL_Σ, can be solved in polynomial time. Indeed, consider the new context-free grammar

$G(\Phi) = \langle \Sigma, Q, P, q_n \rangle$, where $Q = \{q_1, \ldots, q_n\}$ contains the instruction variables of Φ and the production rules in P are obtained from the instructions of Φ as follows: each instruction $q_i = 0/1$ generates the rule $q_i \longrightarrow 0/1$, each instruction $q_i = q_j \cup q_s$ generates the rules $q_i \longrightarrow q_j/q_s$ and each instruction $q_i = q_j \cdot q_s$ generates the rule $q_i \longrightarrow q_j q_s$. Clearly, $w \in \text{eval}_{FL_\Sigma}(\Phi) \Leftrightarrow w \in L_{G(\Phi)}$, hence the membership problem can be checked in polynomial time.

Now, we recall that $\text{eval}_{FL_\Sigma}(\Phi_1) \neq \text{eval}_{FL_\Sigma}(\Phi_2)$ if and only if there exists a word w such that $w \in \text{eval}_{FL_\Sigma}(\Phi_1) \Leftrightarrow w \notin \text{eval}_{FL_\Sigma}(\Phi_2)$. Since $|w| = 2^{O(\max(|\Phi_1|,|\Phi_2|))}$, we conclude that CompEq(FL_Σ) $\in$ *co*NEXPTIME. With a similar argument, Eq(FL_Σ) $\in$ *co*NP.

The completeness of Eq(FL_Σ) is obtained by reduction from 3-TAUT, i.e., the problem of deciding if a disjunction F of 3-monomials is a tautology. Each 3-monomial $M = x_i^{q_i} \wedge x_j^{q_j} \wedge x_s^{q_s}$, where $s > j > i$ and $q_i, q_j, q_s \in \{0, 1\}$, can be associated with the language $L_M = \Sigma^{i-1} q_i \Sigma^{j-i-1} q_j \Sigma^{s-j-i-1} q_s \Sigma^{n-s-j-i-1}$ and the disjunction F of 3-monomials is associated with the language $L_F = \bigcup_{M \in F} L_M$, so that $L_F = \Sigma^*$ if and only if F is a tautology.

At the end, in order to prove the completeness of CompEq(FL_Σ), consider the algebra

$$\mathcal{F}_\Sigma^2 = \langle FL_\Sigma, \{\cup, \cdot, \emptyset, 0, 1, ()^2\} \rangle,$$

where $()^2$ is the squaring operation $(L)^2 = L \cdot L$. The problem Eq($\mathcal{F}_\Sigma^2$) has been proved to be *co*NEXPTIME-complete in Meyer and Stockmeyer (1972). On the other hand, every term e of size n in $\mathcal{F}_\Sigma^2$ can be easily translated in a SLP Φ_e over FL_Σ such that $\text{eval}_{FL_\Sigma}(\Phi_e) = \tau_{\mathcal{F}_\Sigma^2}(e)$ and $|\Phi_e| = O(|e|)$.

Example 3.3 *Consider the ring* $\langle \mathbf{Z}, +, \cdot, -, 0, 1 \rangle$. *Then,* Eq($\mathbf{Z}$) $\in$ *P, while* CompEq($\mathbf{Z}$) $\in$ *coRP.*

This result is presented in Schönhage (1979). The main idea follows a classical fingerprint technique, depicted by the following randomized algorithm that verifies if $\text{eval}_{\mathbf{Z}}(\Phi) = 0$:

```
Input: a SLP $\Phi$ on $\langle \mathbf{Z}, +, \cdot, -, 0, 1 \rangle$ such that $|\Phi| = n$.
  $h \leftarrow \text{rand}(2, n^2 2^n)$;  // select $h$ uniformly at random in $\{2, \ldots, n^2 2^n\}$.
  $x \leftarrow \text{eval}_{\mathbf{Z}_h}(\Phi)$;      // evaluate $\Phi$ using arithmetic modulo $h$.
  if $x \neq 0$ then $s \leftarrow$ "not 0";
            else $s \leftarrow$ "probably 0";
Output: $s$.
```

Clearly, this algorithm works in randomized polynomial time and fails in the case $\text{eval}_{\mathbf{Z}}(\Phi) \neq 0$ but $\text{eval}_{\mathbf{Z}}(\Phi) \mod h = 0$. The probability of error can be obtained in the following way. Call a prime p "good" if p does not divide $\text{eval}_{\mathbf{Z}}(\Phi)$: in this case, $\text{eval}_{\mathbf{Z}}(\Phi) \mod p \neq 0$. Let $\{b_1, \ldots, b_k\}$ be the set of "not good" primes. Since $2^{2^n} \geq |\text{eval}_{\mathbf{Z}}(\Phi)| \geq \prod_{j=1}^{k} b_j \geq 2^k$, there are at most 2^n not good primes, while in the range $\{2, \ldots, n^2 \cdot 2^n\}$ there are $\Theta(n \cdot 2^n)$ primes. Hence, the probability of error is at most $(n^2 2^n - \Theta(n 2^n) + 2^n)/(n^2 2^n) = 1 - \Theta(1/n)$.

Example 3.4 *Consider the monoid M presented by the rewrite system $\langle \{a, b, c, d\}, R \rangle$ where a, b, c, d are the generators and $ac \longrightarrow \epsilon$, $bc \longrightarrow \epsilon$ and $ad \longrightarrow \epsilon$ are the relators. Then,* Eq(M) $\in$ *P, while* CompEq(M) *is coNP-hard.*

This result is proved in (Lohrey, 2006, Th. 5.2) as a consequence of the fact that the problem of deciding if $\text{eval}_M(\Phi) = \epsilon$ is *co*NP-complete. We sketch the proof, which takes advantage of Lohrey strings (see

Example 2.3). With every instance I of SUBSETSUM of size n, we associate the term $T(I)$ on $\{a,b,c,d\}^*$ given by

$$T(I) \;=\; \xi_{ab}(I)\xi'_{cd}(I) \;=\; (a^{s-t}ba^t)^{2^n} \prod_{\substack{x\in\{0,1\}^n \\ b(x)=0..2^n-1}} (c^{\underline{x}\cdot\underline{w}}dc^{s-\underline{x}\cdot\underline{w}})$$

By Lemma 1, $T(I)$ can be represented by a SLP of polynomial size. Now, observe that:

$$\tau_M(a^{s-t}ba^tc^{t'}dc^{s-t'}) = \begin{cases} \epsilon & \text{if} \quad t \neq t', \\ bd & \text{if} \quad t = t'. \end{cases}$$

As a consequence:

$$\tau_M(T(I)) = \epsilon \quad \Longleftrightarrow \quad \underline{x}\cdot\underline{w} \neq t \;\; \forall\, x \in \{0,1\}^n \quad \Longleftrightarrow \quad I \notin \text{SUBSETSUM}.$$

4 CompRel(R,A): an example in the area of arithmetical SLPs

Often, a deep insight on a problem about SLPs on a fixed algebraic structure can give unexpected benefits in other areas. In this section we recall methods and results concerning the problem PosSLP, together with its relation with problems such as Square Root Sum (Garey et al. (1976)) and (Strong) ϵ-Approximation of Nash equilibrium (Nash (1951); Etessami and Yannakakis (2007)). PosSLP is defined on the arithmetical algebra $\langle \mathbf{Z}, +, \cdot, - \rangle$:

PROBLEM: *Positivity of SLP (*PosSLP*)*
INSTANCE: a SLP Φ on $\langle \mathbf{Z}, +, \cdot, - \rangle$;
QUESTION:is $\text{eval}_{\mathbf{Z}}(\Phi) > 0$?

This apparently somewhat artificial problem is motivated in Allender et al. (2008/09) for better understanding the Blum-Shub-Smale model of computation (Blum et al. (1997)); moreover, it captures important stability aspects of floating point numerical computation.

In the matter of its complexity, observe that for a SLP Φ of size n it holds $-2^{2^n} < \text{eval}_{\mathbf{R}}(\Phi) < 2^{2^n}$, therefore:

$$(2^{2^n} - 1 + \text{eval}_{\mathbf{R}}(\Phi)) : 2^{2^n} \;=\; \begin{cases} 1 & \text{if } \text{eval}_{\mathbf{R}}(\Phi) > 0 \\ 0 & \text{otherwise} \end{cases}$$

where : is the integer division. Since Eq($\langle \mathbf{Z}, +, \cdot, -, : \rangle$) is PSPACE-complete (Bertoni et al. (1981)), we conclude that PosSLP is in PSPACE. Is it possible to do something better? A nice result in Allender et al. (2008/09) proves that PosSLP is in the counting hierarchy. The (polynomial) counting hierarchy CH was introduced by Wagner (Wagner (1986)) with the goal of classifying the complexity of certain combinatorial problems where counting is involved. Let PP be the class of languages L for which there exists a polynomial time bounded probabilistic Turing machine M such that $x \in L$ if and only if x is accepted with probability larger that $1/2$. Then, for every integer k, the k-th level C_k of the counting hierarchy is defined by:

$$C_{k+1} = \text{PP}^{C_k} \qquad \text{and} \qquad C_0 = \text{P}$$

CH is the union of all classes C_k. CH is contained in PSPACE, moreover CH collapses to P if P=PP.

Fact 2 PosSLP *is in* CH.

Fact 2 is a consequence of methods and results developed on circuit theory in the 90s, particularly in the area of neural networks with a fixed number of layers. In circuit theory, one is interested to state whether there are (or not) efficient circuits for fixed functions. For instance, consider the Boolean function majority:

$$\text{majority}(x_1, \ldots, x_n) = 1 \quad \Longleftrightarrow \quad \sum_{k=1}^{n} x_k > \frac{n}{2}$$

It has been proved in Razborov (1987) that majority is not in AC_0, the class of Boolean functions computable by polynomial size unbounded fan-in circuits with constant depth. It follows as a natural question to study the power of circuits using majority as gates, by considering the class TC_0 of Boolean functions computable by polynomial size constant depth circuits containing constants, negation and majority.

In this area there has been a sequence of surprising results. For instance, in the case of iterated multiplication and division of integers, in Reif and Tate (1992) threshold circuits of polynomial size and depth 4 have been found. A further solution with strong uniformity requirements has been given in Hesse et al. (2002), where DLOGTIME-uniform circuits performing these tasks are exhibited. The main tool is the construction of a family D_n of DLOGTIME-uniform threshold circuits of polynomial size and constant depth that, having as input the Chinese Remainder Representation of an integer z of n bits, compute the binary representation of z. The Chinese Remainder Representation of an integer z of n bits is a sequence $z(p, k)$ of binary values indexed by (p, k), where $z(p, k)$ is the k-th bit of the integer $z \mod p$, for each prime $p < n^2$.

For proving that PosSLP is in CH, given a SLP Φ of size n, it needs to construct a SLP Ψ such that $\text{eval}_{\mathbf{Z}}(\Psi) = 2^{2^n} + \text{eval}_{\mathbf{Z}}(\Phi)$. Observe that, if z is a string of n bits representing in binary notation the integer $\text{eval}_{\mathbf{Z}}(\Psi)$, the first bit of z is 1 if and only if $\text{eval}_{\mathbf{Z}}(\Phi) > 0$. A circuit E_n computing z using Ψ is constructed as follows: gates of level 1 are labelled by (p, k) (p and k integers represented with $2n$ bits, p prime) and (p, k) outputs the k-th bit of $\text{eval}_{\mathbf{Z}}(\Psi) \mod p$; the other gates are those of D_n, i.e. majority gates. Given a gate F of E_n, let $u(\Phi, F)$ be the output of F if E_n uses Ψ. Consider now:

$$L_d \ = \ \{(\Phi, F, b) \mid F \text{ is the label of a gate of level } d, \ b = u(\Phi, F)\}$$

Clearly L_1 is in C_1. Let $d > 1$. For a gate F at level d, since F is a majority gate, it holds that $u(\Phi, F) = 1$ if and only if

$$\text{Prob}(u(\Phi, F') = 1 \mid F' \text{ connected with } F, \text{ hence } F' \text{ is at level } d - 1) > 1/2$$

Since the family D_n is DLOGTIME-uniform, given gates F and F' it can be verified in polynomial time if F' is connected to F. Therefore, if L_{d-1} is in C_{d-1} then L_d is in C_d. Since E_n has a constant depth, PosSLP is in CH.

This result has consequences for several problems in areas such as computational geometry, numerical analysis and game theory. Here we recall the relations connecting PosSLP with Square Root Sum Problem (SQRT-Sum) (Allender et al. (2008/09)) and (Strong) ϵ-Approximation of Nash Equilibrium (Etessami and Yannakakis (2007)).

PosSLP vs. SQRT-Sum

SQRT-Sum consists of deciding, given as input $n+1$ positive integers $a_1, \dots, a_n, r$, if $\sum_{k=1}^{n} \sqrt{a_k} > r$. It appeared in Garey et al. (1976) in relation with Traveling Salesman Problem in the euclidean plane, which was only shown to be NP-hard, because of the difficulty of checking if a given path has length less than r. So, a natural still open question posed in Garey et al. (1976) is whether SQRT-Sum is in NP. At present, the best classification of SQRT-Sum is the following, obtained in Allender et al. (2008/09) combining arguments in Tiwari (1992):

Fact 3 SQRT-Sum *is in* CH.

We outline a proof based on a reduction to PosSLP via Canny's gap theorem (Canny (1988a)):

Theorem 1 (Canny's gap) *Let $x_1, x_2, \dots, x_n$ be an arbitrary solution of an algebraic system of n equations and n unknowns, having a finite number of solutions, with maximal total degree d, with relative integer coefficients smaller or equal to M in absolute value. Then, for all $i = 1, \dots n$, either $x_i = 0$ or $|x_i| > \epsilon$ where $\epsilon = (3Md)^{-nd^n}$.*

Theorem 1 gives a way to numerically prove whether an algebraic number is zero, greater than zero or less than zero: compute a (guaranteed) interval containing it, of width smaller than ϵ. As soon as the interval does not contain zero, the number is clearly not zero and its sign is known. Otherwise, if the interval contains zero and has width less than ϵ, then the number can only be zero.

By Theorem 1, we know that

$$\sum_{k=1}^{n} \sqrt{a_k} - r \;=\; 0 \quad \text{(exclusive) or} \quad \left|\sum_{k=1}^{n} \sqrt{a_k} - r\right| \;>\; \max_{1 \le k \le n} (a_k)^{-2^{\Theta(n)}}.$$

Hence, it suffices to approximate $\sqrt{a_i}$ up to $\delta = \frac{1}{3n} \max(a_k)^{-2^{\Theta(n)}}$, for $1 \le i \le n$. To this end, consider the following (concise representation of a) SLP on the rational field $\langle \mathbf{Q}, +, \cdot, -, / \rangle$ approximating $\sqrt{a_i}$ by the Newton Method:

$$X_0 = \lceil \sqrt{a_i} \rceil; \quad X_1 = X_0/2 + a_i/(2X_0); \quad \dots; \quad X_k = X_{k-1}/2 + a_i/(2X_{k-1}).$$

Since Newton Method guarantees a quadratic convergence, it holds

$$|X_k - \sqrt{a_i}| \le 2^{-2^{O(k)}}.$$

By choosing k so that $2^{-2^{O(k)}} = \delta$, we approximate $\sqrt{a_i}$ with sufficient precision by a SLP of size $O(n + \log \max(a_i))$. Therefore, we can decide $\sum_i \sqrt{a_i} - r > 0$ by checking if a polynomial size SLP Φ on $\langle \mathbf{Q}, +, \cdot, -, / \rangle$ represents a number greater than $r \in \mathbf{Q}$.

This last problem can be reduced to PosSLP. Indeed, recalling that every rational x is $x = a/b$ for two integers a and b, with every SLP Φ on $\langle \mathbf{Q}, +, \cdot, -, / \rangle$ we can associate in polynomial time two SLPs Π and Δ on $\langle \mathbf{Z}, +, \cdot, - \rangle$ such that $\text{eval}_{\mathbf{Q}}(\Phi) = \text{eval}_{\mathbf{Z}}(\Pi)/\text{eval}_{\mathbf{Z}}(\Delta)$.

Hence, the problem of checking $\text{eval}_{\mathbf{Q}}(\Phi) > r$ can be reduced to solve PosSLP for a SLP of size $O(|\Phi|)$ representing $(\text{eval}_{\mathbf{Z}}(\Pi) - r \cdot \text{eval}_{\mathbf{Z}}(\Delta)) \cdot \text{eval}_{\mathbf{Z}}(\Delta)$. By Fact 2, SQRT-Sum is in CH.

PosSLP vs. Nash Equilibrium

Nash equilibrium is a central concept in the area of strategic games, which model situations where several actors are making decisions at the same time, knowing that the gain of each one depends on the decisions of the others. In a n-player game, each player $k \in \{1, \dots, n\}$ disposes of a finite set S_k of strategies together with a payoff function $u_k : S \longrightarrow Q$, where $S = \prod_k S_k$; the rational number $u_k(s_1, \dots, s_n)$ represents the "gain" of player k if player j adopts the strategy s_j, for all j. The players can adopt probabilistic strategies, called mixed strategies: a mixed strategy of player k is a probability distribution p_k on S_k; the situation where each player k chooses mixed strategy p_k is described by the strategy profile $p = (p_1, \dots, p_n)$. A profile p induces the probability distribution $\hat{p}$ on S given by $\hat{p}(s_1, \dots, s_n) = p_1(s_1) \cdot \dots \cdot p_n(s_n)$, and the expected payoff of player k on the profile p is:

$$U_k(p) = E_{\hat{p}}[u_k].$$

The target of each player is to maximize his expected payoff, but that depends on the choices of the others: a Nash equilibrium is a profile p^* where no player can increase its own payoff by unilaterally switching its strategy. More formally, with (p, k, g_k) we denote the profile obtained from p by substituting the strategy p_k of player k with the new strategy g_k; then a Nash equilibrium is a profile p^* such that, for every player k and a mixed strategy g_k, $U_k(p^*) \geq U_k(p^*, k, g_k)$.

A central result presented in Nash (1951), previously proved by von Neumann and Morgenstern (1947) in the case of sum-zero games, is:

Fact 4 *Every finite game has a Nash Equilibrium.*

This result is proved in a non constructive way, via Brower's Theorem. Then, a non trivial task is to search for a Nash equilibrium. Moreover, while for games with two players the Nash equilibria are vectors with rational entries, as pointed out in Nash (1951) there are 3-player games with only irrational equilibria, that we can only approximate. As a consequence, a very natural problem is:

PROBLEM: *(Strong) ϵ-Approximation of a Nash Equilibrium (*Nash ϵ-App*)*
INSTANCE: a finite game G with 3 players;
OUTPUT: a profile z s. t. $\exists$ Nash Equilibrium p^* with $||z - p^*||_\infty < \epsilon$,

where, for a vector $x \in \mathbf{R}^d$, $||x||_\infty = \max |x_i|$.

Given a game G, a Nash Equilibrium is a fixed point $z = F_G(z)$ for a suitable function F_G, obtained as a composition of operations $+$, $\cdot$, $/$ and $\max$ on the set of real numbers $\mathbf{R}$. Fixed two vectors a and b, the expression

$$\exists z \ \ (z = F_G(z) \ \wedge \ a \leq z \leq b)$$

can be easily transformed in a formula on the Existential Theory of Reals. Since this theory can be decided in PSPACE (Canny (1988b)), Nash ϵ-App can be solved in PSPACE. Further improvements to this result are limited from the fact that Nash ϵ-App is more difficult than PosSLP (Etessami and Yannakakis (2007)).

In particular, Etessami and Yannakakis give a construction that associates with every SLP Φ on $\langle \mathbf{Z}, +, \cdot, - \rangle$ a 3-player game G that admits a unique Nash equilibrium p^* such that the first strategy s_1 of Player 3 verifies:

$$\begin{aligned} \text{eval}_{\mathbf{Z}}(\Phi) > 0 &\implies p_3^*(s_1) > 1 - \epsilon, \\ \text{eval}_{\mathbf{Z}}(\Phi) \leq 0 &\implies p_3^*(s_1) = 0. \end{aligned}$$

That means that any nontrivial approximation of a Nash equilibrium is computationally difficult at least as PosSLP:

Fact 5 PosSLP *is polinomially reducible to* Nash ϵ-App.

5 Algebra representations

In this section, we discuss some methods useful to reduce CompEq(A) to CompEq(B) efficiently, for some given algebras A and B.

First, we look for a notion of representation of an algebra into another preseving the complexity of CompEq. As an example, we show a representation of free partially commutative monoids (Mazurkiewicz (1977); Cartier and Foata (1969)) in the semiring of integer numbers. A direct consequence is an efficient randomized polynomial time algorithm to solve CompEq(M) for trace monoids M.

Secondly, we recall the notion of composition system (Gasieniec et al. (1996)), a flexible concept applied to efficiently solve CompEq(M) for certain monoids M. As an application, we present a polynomial time algorithm for CompEq(D_m), where D_m is the monoid with $2m$ generators obtained by Dyck reductions over m types of parentheses, a class of monoids studied in the context of XML-grammars (Berstel and Boasson (2002)); a more general result can be found in Lohrey (2006).

5.1 Representations

Consider two algebras $\langle A, \Sigma_A\rangle$ and $\langle B, \Gamma_B\rangle$ and, for the sake of simplicity, let $\max_{\sigma\in\Sigma_A}\{m(\sigma)\} = 2$.

Definition 1 *A* representation *of A in B is an injective function $f = (f_1, \dots, f_s) : T_\Sigma(X) \longrightarrow T_\Gamma(X)^s$ for some $s \in \mathbf{N}$, such that, for every operation $\sigma \in \Sigma$, there exist s Γ-expressions $t_{\sigma,1}, \dots, t_{\sigma,s} \in T_\Gamma(X)$ satisfying*

$$\tau_B(f_i(\sigma xy)) = \tau_B(t_{\sigma,i}(f_1(x), \dots, f_s(x), f_1(y), \dots, f_s(y))), \quad (1)$$

for every i, $1 \le i \le m$.

We say that A is *representable* in B if there exists a representation of A in B.

Theorem 2 *Let A and B be two algebras. If A is* representable *in B, then* CompEq(A) *is polynomial time reducible to* CompEq(B).

Proof: Consider a SLP Φ in A with variables $X_1, \dots X_n$. Recalling Eq. (1), we associate with Φ a SLP Ψ in B with variables $Y_{j,i}$, where $1 \le j \le n$ and $1 \le i \le s$. More precisely, each instruction $X_k = \sigma X_l X_m$ in Φ corresponds to s macro-instructions in Ψ of the form

$$\begin{aligned} Y_{k,1} &= t_{\sigma,1}(Y_{l,1}, \dots, Y_{l,s}, Y_{m,1}, \dots, Y_{m,s}), \\ &\vdots \\ Y_{k,s} &= t_{\sigma,s}(Y_{l,1}, \dots, Y_{l,s}, Y_{m,1}, \dots, Y_{m,s}). \end{aligned}$$

With macro-instruction $Y_{k,i} = t_{\sigma,i}(Y_{l,1}, \dots, Y_{l,s}, Y_{m,1}, \dots, Y_{m,s})$ we intend the SLP straightforwardly obtainable from $t_{\sigma,i}$, interpreted as a derivation tree of the grammar $T_\Gamma(X)$. For example, from $t(x, y, z) = \gamma_1(x, \gamma_2(y, z))$ we obtain the SLP whose instructions are $X_1 = \gamma_2(y, z)$, $X_2 = \gamma_1(x, X_1)$.

The structure of Eq. (1) implies that, for all $1 \leq i \leq s$:

$$\tau_B(f_i(t_\Phi)) = \mathrm{eval}_B(Y_{n,i}).$$

Denote by Ψ_i the SLP obtained from Ψ considering $Y_{n,i}$ as output variable, so that $\mathrm{eval}_B(\Psi_i) = \mathrm{eval}_B(Y_{n,i})$. Given an instance (Φ, Φ') of CompEq(A), we construct $\Psi_i, \ldots, \Psi_s$ and $\Psi'_1, \ldots, \Psi'_s$. Injectivity of f implies that $\mathrm{eval}_A(\Phi) = \mathrm{eval}_A(\Phi')$ if and only if

$$\bigwedge_{1 \leq i \leq s} \mathrm{eval}_A(\Psi_i) = \mathrm{eval}_A(\Psi'_i).$$

Since the sizes of the translated SLPs are polynomially bounded, the theorem is proved. □

In the following, we show some examples of representations reducing, directly or undirectly, the problem CompEq on different monoids to the same problem on the semiring of natural numbers.

Free Monoid vs. Integer Numbers

Let $\Sigma = \{0, 1\}$ and consider the free monoid Σ^*. Then, Σ^* is representable in the semiring $\langle \mathbf{N}, +, \cdot \rangle$ as follows.

Consider a function $b : \Sigma^* \longrightarrow \mathbf{N}$ where $b(\epsilon) = 0$ and $b(w) = \sum_{i=0}^{n-1} x_i \cdot 2^i$ for $w = x_0 x_1 \cdots x_{n-1}$. The representation of Σ^* in $\mathbf{N}$ can be realized by the injective function $f : \Sigma^* \longrightarrow \mathbf{N}^2$ defined as $f(w) = (f_1(w), f_2(w))$, where $f_1(w) = b(w)$ and $f_2(w) = 2^{|w|}$. Indeed, $f(\epsilon) = (0, 1)$, $f(0) = (0, 2)$ and $f(1) = (1, 2)$. Moreover,

$$f_1(w \cdot v) = f_1(w) \cdot f_2(v) + f_1(v) \quad \text{and} \quad f_2(w \cdot v) = f_2(w) \cdot f_2(v).$$

As a consequence, recalling Example 3.3, CompEq(Σ^*) can be solved in polynomial time by a randomized algorithm. It should be remarked that, thanks to Plandowski (1994), the same problem has been solved in polynomial time deterministically (Example 3.1).

Trace Monoids vs. Free Monoids

Trace monoids were introduced in Mazurkiewicz (1977) for representing concurrent processes. They are defined by undirected graphs $\langle \Sigma, E \rangle$ where Σ is the set of generators and E, called independence relation, summarizes commutations $ab = ba$ for $(a, b) \in E$. The trace monoid $T(\Sigma, E)$ is the free partially commutative monoid generated by $\langle \Sigma, E \rangle$ (Cartier and Foata (1969)) and its elements are called traces.

Every trace monoid $T(\Sigma, E)$ is representable in the free monoid $\{0, 1\}^*$. Indeed, let $<$ be a total order on Σ and consider, for $\sigma \in \Sigma$, the morphism $f_\sigma(\sigma') = 0$ if $\sigma = \sigma'$ and $f_\sigma(\sigma') = \epsilon$ otherwise, and, for $(a, b) \notin E$ and $a < b$, $f_{ab}(a) = 0$, $f_{ab}(b) = 1$, $f_{ab}(\sigma) = \epsilon$ if $\sigma \neq a$ or $\sigma \neq b$. In Duboc (1986) it is proved that, for all traces $t \neq t'$, either there is $\sigma \in \Sigma$ for which $f_\sigma(t) \neq f_\sigma(t')$ or there is $(a, b) \notin E$ such that $f_{ab}(t) \neq f_{ab}(t')$. Hence, there is an injective morphism $f : T(\Sigma, E) \longrightarrow (\{0, 1\}^*)^s$, where $s = |\Sigma| \left(|\Sigma| + 1\right)/2 - |E|$, proving the reprentability of $T(\Sigma, E)$ in $\{0, 1\}^*$.

The injectivity of f is proved by the fact that the f_σ's guarantee that traces with different Parikh vectors have different images. Moreover, a swap of two adjacent noncommuting symbols in a trace t necessarily causes a swap in at least one of the components f_{ab} (Duboc (1986)).

Therefore, recalling Example 3.1, also CompEq($T(\Sigma, E)$) can be solved in polynomial time for every trace monoid $T(\Sigma, E)$.

Finally, we observe that there are useful polynomial time reductions that are not representations. For instance, consider the semiring $\langle \mathbf{N}[x], +, \cdot \rangle$ of polynomials with nonnegative coefficients and the semiring $\langle \mathbf{N}, +, \cdot \rangle$ of nonnegative integers. The function

$$p(x) \longmapsto p(2^{p(1)+1})$$

is not a representation in the sense of Definition 1, but is useful for reducing CompEq($\mathbf{N}[x]$) to CompEq($\mathbf{N}$), since, for polynomials with the same degree, $p \neq q$ implies $p(2^{p(1)+1}) \neq q(2^{q(1)+1})$.

In a similar way, by considering the semiring $\langle \mathbf{N}[x, y], +, \cdot \rangle$ of bivariate polynomials, CompEq($\mathbf{N}[x, y]$) can be reduced to CompEq($\mathbf{N}$). This fact gives the intuition that CompEq($\langle \Sigma^{**}, \ominus, \oslash \rangle$) can be solved in randomized polynomial time (Berman et al. (1997)), where $\langle \Sigma^{**}, \ominus, \oslash \rangle$ is the algebra of bidimensional words with horizontal and vertical concatenations. In fact, in Berman et al. (1997) is observed that $\langle \Sigma^{**}, \ominus, \oslash \rangle$ admits a natural representation in $\langle \mathbf{N}[x, y], +, \cdot \rangle$.

It is a still open problem whether CompEq($\langle \Sigma^{**}, \ominus, \oslash \rangle$) is in P.

5.2 Composition systems

Let M be a monoid presented by a confluent and terminating semi-Thue system (see Book and Otto (1993)) with generators in Σ. In this case, there is a function red : $\Sigma^* \longrightarrow \Sigma^*$ that associates with every $w \in \Sigma^*$ its *normal form* red(w) so that the monoid M is isomorphic to $\langle \text{red}(\Sigma^*), \odot, \epsilon \rangle$, where $x \odot y = \text{red}(xy)$; a SLP Φ on M represents the word eval$_M(\Phi) \in \Sigma^*$.

Suppose now to enrich $\langle \Sigma^*, \cdot, \epsilon \rangle$ with new operations $\mathcal{O}$ on Σ^* so that CompEq($\langle \Sigma^*, \cdot, \mathcal{O}, \epsilon \rangle$) remains in P. Then, a method for solving CompEq(M) is to consider the following two steps:

1. Find a function f that associates every SLP Φ on M with a SLP $f(\Phi)$ on an (even infinite) algebra $\langle \Sigma^*, \cdot, \mathcal{O}, \epsilon \rangle$ such that eval$_M(\Phi) = \text{eval}_{\Sigma^*}(f(\Phi))$;

2. Given two SLPs Φ, Ψ on M, check eval$_{\Sigma^*}(f(\Phi)) = \text{eval}_{\Sigma^*}(f(\Psi))$.

In fact, if the function f is polynomial time computable, then CompEq(M) is in P.

A good choice for $\mathcal{O}$ turned out to be the operations of prefix and suffix selection; that leads to the concept of composition system (Gasieniec et al. (1996)).

Given the monoid $\langle \Sigma^*, \cdot \rangle$ and a word $w \in \Sigma^*$, let $w[i]$ the symbol of w in position i, with $1 \leq i \leq |w|$ and let $w[i, j]$ be the factor $w[i]w[i+1] \cdots w[j]$, where $i \leq j$. A *composition system* is a SLP generating words in Σ^* where the instructions are the concatenation, the prefix and the suffix selection, denoted respectively by

$$A = BC \qquad A = {}^{[i]}B \qquad \text{and} \qquad A = B^{[i]}.$$

The semantics of the concatenation is obvious, while for the others it holds eval$_{\Sigma^*}({}^{[i]}B) = w[1, i]$ and eval$_{\Sigma^*}(B^{[i]}) = w[|w| - i, |w|]$, with $w = \text{eval}_{\Sigma^*}(B)$.

In Gasieniec et al. (1996) it has been proved the following:

Lemma 2 *The compressed equivalence problem for composition systems is solvable in polynomial time.*

Such a result is used in Lohrey (2006) for proving that CompEq(M) is solvable in polynomial time if M is presented by suitable semi-Thue systems (2-homogeneous confluent N-free rewrite systems). Here, we present the particular (but important) case of Dyck reductions.

Dyck reductions

Given a finite alphabet Σ, let $\overline{\Sigma} = \{\overline{\sigma} \mid \sigma \in \Sigma\}$ a disjoint copy of σ. Let $\langle \Sigma \cup \overline{\Sigma}, R\rangle$ be the rewrite system where $R = \{\sigma\overline{\sigma} \longrightarrow \epsilon \mid \sigma \in \Sigma\}$, so that $w_1 \longrightarrow_R w_2$ if and only if $w_1 = v\sigma\overline{\sigma}u$ and $w_2 = vu$, for some $\sigma \in \Sigma$.

Now, let $\sim_R$ be the smallest congruence relation containing $\longrightarrow_R$ and consider the quotient monoid $M = (\Sigma \cup \overline{\Sigma})^* / \sim_R$; notice that $\{w \mid w \sim_R \epsilon\}$ is the Dyck language on Σ. Every element in $(\Sigma \cup \overline{\Sigma})^*$ can be mapped into a reduced word red(w) having no factors of the form $\sigma\overline{\sigma}$, for all $\sigma \in \Sigma$, by using rewrite rules regardless of the order of their application. As a consequence, M is isomorphic to $\langle \text{red}(\Sigma^*), \odot, \epsilon\rangle$, where red$(\Sigma^*)$ is the set of reduced words and

$$w \odot v \;=\; \text{red}(w \cdot v) \;=\; w[1, |w| - i]v[i + 1, |v|], \tag{2}$$

being i the maximum integer for which $\text{red}(w[|w| - i + 1, |w|]v[1, i]) = \epsilon$. Let w' be the string in Σ^* such that $\overline{w}'[j] = w[|w| - j]$, for $j = 1, \ldots, |w|$. Then, i is the length of the maximum common prefix of w' and v and can be computed in polynomial time by a binary search, recalling $\log |v|$ times the algorithm proving Lemma 2 (see Gasieniec et al. (1996) for further details).

Every SLP on $\langle \text{red}(M), \odot\rangle$ can be inductively translated in a composition system by replacing each instruction $A = B \odot C$ with the sequence of instructions

$B' = {}^{[|\text{eval}_M(B)|-i]}B$
$C' = C^{[|\text{eval}_M(C)|-i+1]}$
$A \;= B'C'$

where i, defined as in Eq. (2), is obtained by doing the previously described binary search. Hence, Step 1 of the method can be computed in polynomial time.

Such a coding from SLPs on M to compositions systems, together with Lemma 2, also allows solving Step 2 in polynomial time, proving:

Fact 6 CompEq(M) *is in* P.

6 CompEq and Inclusion Problem for CF languages

In this section, we point out the benefit of studying Compressed Equivalence Problem for efficiently solving, in some cases, the inclusion problem of context-free languages. Given a context-free language L_0, the inclusion problem for L_0 can be formulated as follows:

PROBLEM: *Inclusion for* L_0 *(*Incl(L_0)*)*
INSTANCE: a CF grammar G;
QUESTION:does $L_G \subseteq L_0$?

This problem is known to be indecidable even for some deterministic languages L_0 (see, for example, Valiant (1973)) and the set of CF grammars generating languages L_0 for which the problem is decidable is not recursive (Hopcroft (1969)). However, if the language L_0 is superdeterministic, then the problem turns out to be decidable, even though the best algorithm proposed is doubly exponential (Greibach and Friedman (1980)).

A natural problem is that of finding a class of languages L_0 for which the problem is solvable in polynomial time. To this matter, let M be a monoid presented by a terminating and confluent semi-Thue system; for every string w, let red(w) be its normal form (see Subsection 5.2) and let $L_M = \text{red}^{-1}(\epsilon)$. If M is cancellative (i.e. $zxw = zyw$ implies $x = y$, for every $x, y, z, w \in M$), then Incl(L_M) and CompEq(M) are related by the following (Bertoni et al. (2009)).

Theorem 3 *Let M be a monoid presented by a terminating and confluent semi-Thue system. If M is cancellative, then* Incl(L_M) *is polynomially reducible to* CompEq(M).

This result is based on the following property of cancellative monoids: let G be a CF grammar with nonterminal symbols $X_0, \ldots, X_n$, where X_0 is the axiom. If M is cancellative, then

$$L_G \subseteq L_M \quad \Longleftrightarrow \quad |\text{red}(L_{X_i})| = 1,\ i = 0, \ldots, n \ \wedge\ \text{red}(L_{X_0}) = \{\epsilon\}. \tag{3}$$

In light of this, Incl(L_M) can be decided in two steps.

First, associate with every X_i in G a SLP Φ_i, generating a word in L_{X_i} with derivation tree of minimal length.

Second, for all the productions $X_i \to X_j X_k$ (resp. $X_i \to \sigma$) in G, check if red$(w_i) =$ red$(w_j w_k)$ (resp. red$(w_i) = \sigma$). These conditions are not verified on the words w_1, but through the SLPs Φ_i representing the words. If at least one of them is not true or red$(w_0) \neq \epsilon$, then the left side of the condition of (3) is false.

In the matter of the computational complexity, the first step can be executed in polynomial time, while the second requires to call a polynomial number of times an algorithm for CompEq(M).

Example 6.1 *Consider the problem Incl(D_A), where D_A is the Dyck language with parenthesis $A \cup \overline{A}$. Incl(D_A) is solvable in polynomial time.*

Such a problem is proved to be decidable by Knuth (1967) in the case $|A| = 1$; a doubly exponential algorithm has been exhibited by Berstel and Boasson (2002) in the general case. Now, every Dyck language is the preimage $L_M = \text{red}^{-1}(\epsilon)$ in a monoid M presented by Dyck reductions (see Subsection 5.2). Since, by Fact 6, CompEq(M) is in P and it is easy to verify that M is cancellative, then Incl(D_A) is solvable in polynomial time.

References

E. Allender, W. Hesse, and D. A. Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. System Sci.*, 65(4):695–716, 2002. Special issue on complexity, 2001 (Chicago, IL).

E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. B. Miltersen. On the complexity of numerical analysis. *SIAM J. Comput.*, 38(5):1987–2006, 2008/09.

P. W. Beame, S. A. Cook, and H. J. Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

P. Berman, M. Karpinski, L. L. Larmore, W. Plandowski, and W. Rytter. On the complexity of pattern matching for highly compressed two-dimensional texts. In *Combinatorial pattern matching (Aarhus, 1997)*, volume 1264 of *Lecture Notes in Comput. Sci.*, pages 40–51. Springer, Berlin, 1997.

J. Berstel and L. Boasson. Formal properties of XML grammars and languages. *Acta Inform.*, 38(9): 649–671, 2002.

A. Bertoni, G. Mauri, and N. Sabadini. A characterization of the class of functions computable in polynomial time on random access machines. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 168–176. ACM New York, NY, USA, 1981.

A. Bertoni, C. Choffrut, and R. Radicioni. The inclusion problem of context-free languages: Some tractable cases. In V. Diekert and D. Nowotka, editors, *Developments in language theory*, volume 5583 of *Lecture Notes in Comput. Sci.*, pages 103–112. Springer-Verlag, Berlin Heidelberg, 2009.

L. Blum, M. Shub, F. Cucker, and S. Smale. *Complexity and real computation*. Springer Verlag, 1997.

R. V. Book and F. Otto. *String-rewriting systems*. Springer-Verlag, London, UK, 1993.

J. Canny. *The complexity of robot motion planning*, volume 1987 of *ACM Doctoral Dissertation Awards*. MIT Press, Cambridge, MA, 1988a.

J. Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 460–469. ACM New York, NY, USA, 1988b.

P. Cartier and D. Foata. *Problèmes combinatoires de commutation et réarrangements*. Lecture Notes in Mathematics, No. 85. Springer-Verlag, Berlin, 1969.

C. Duboc. *Commutations dans les monoïdes libres: un cadre théorique pour l'étude du parallélisme*. PhD thesis, Université de Rouen, 1986.

K. Etessami and M. Yannakakis. On the Complexity of Nash Equilibria and Other Fixed Points (Extended Abstract). In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 113–123. IEEE Computer Society, 2007.

M. Garey, R. Graham, and D. Johnson. Some NP-complete geometric problems. In *Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 10–22. ACM New York, NY, USA, 1976.

M. R. Garey and D. S. Johnson. *Computers and intractability*. W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences.

A. Gascón, G. Godoy, and M. Schmidt-Schauß. Unification with singleton tree grammars. In *Rewriting Techniques and Applications*, pages 365–379. Springer, 2009.

L. Gasieniec, M. Karpinski, W. Plandowski, and W. Rytter. Efficient algorithms for lempel-ziv encoding. *Lecture Notes in Computer Science*, 1097:392–403, 1996.

S. A. Greibach and E. P. Friedman. Superdeterministic PDAs: a subcase with a decidable inclusion problem. *J. Assoc. Comput. Mach.*, 27(4):675–700, 1980.

W. Hesse, E. Allender, and D. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65(4):695–716, 2002.

J. E. Hopcroft. On the equivalence and containment problems for context-free languages. *Math. Systems Theory*, 3:119–124, 1969.

H. Hunt III, R. Stearns, and M. Marathe. Relational representability, local reductions, and the complexity of generalized satisfiability problems. *Manuscript*, 2000.

D. E. Knuth. A characterization of parenthesis languages. *Inform. Control*, 11(3):269–289, 1967.

A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Internat. J. Comput. Math.*, 2:157–168, 1968.

E. Lehman and A. Shelat. Approximation algorithms for grammar-based compression. In *SODA '02: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 205–212, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics.

Y. Lifshits. Processing compressed texts: A tractability border. In *Combinatorial Pattern Matching*, volume 4580 of *Lecture Notes in Comput. Sci.*, pages 228–240. Springer, Berlin, 2007.

M. Lohrey. Word problems and membership problems on compressed words. *SIAM J. Comput.*, 35(5): 1210–1240, 2006.

A. Mazurkiewicz. Concurrent program schemes and their interpretation. Technical Report DAIMI PB-78, Aarhus University, Comp. Science Depart., July 1977.

A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. pages 125–129, Washington, DC, USA, 1972. IEEE Computer Society.

J. Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):286–295, 1951.

W. Plandowski. Testing equivalence of morphisms on context-free languages. In *Algorithms—ESA '94 (Utrecht)*, volume 855 of *Lecture Notes in Comput. Sci.*, pages 460–470. Springer, Berlin, 1994.

A. Razborov. Lower bounds on the size of bounded depth networks over a complete basis with logical addition. *Mathematicheskie Zametki*, 41(4):598–607, 1987.

J. H. Reif and S. R. Tate. On threshold circuits and polynomial computation. *SIAM J. Comput.*, 21(5): 896–908, 1992.

W. Rytter. Application of Lempel-Ziv factorization to the approximation of grammar-based compression. *Theoret. Comput. Sci.*, 302(1-3):211–222, 2003.

A. Schönhage. On the power of random access machines. In *Proceedings of the 6th Colloquium, on Automata, Languages and Programming*, pages 520–529. Springer-Verlag London, UK, 1979.

P. Tiwari. A problem that is easier to solve on the unit-cost algebraic RAM. *J. Complexity*, 8(4):393–397, 1992.

J. Torán. Complexity classes defined by counting quantifiers. *J. Assoc. Comput. Mach.*, 38(3):753–774, 1991.

L. G. Valiant. *Decision procedures for families of deterministic pushdown automata*. PhD thesis, University of Warwick Coventry, UK, 1973.

J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, N. J., 1947. 2d ed.

K. W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Inform.*, 23(3):325–356, 1986.