Fast Equality Test for Straight-Line Compressed Strings

Manfred Schmidt-Schauß∗, Georg Schnitger†

Institut für Informatik
Goethe-Universität
Postfach 11 19 32
D-60054 Frankfurt, Germany

Abstract

The paper describes a simple and fast randomized test for equality of grammar-compressed strings. The thorough running time analysis is done by applying a logarithmic cost measure.

Key words: randomized algorithms, straight line programs, grammar-based compression

1. Introduction

Compression of data like strings and trees improves space usage, a well-known method is Lempel-Ziv encoding [7]. The standard way of applying algorithms to the data is a decompression prior to the application perhaps followed by a compression of the generated or modified data. Algorithms that can be translated such that they work efficiently on the compressed data are of interest, and complement the space efficiency by also improving running times.

To avoid the peculiarities of a specialized compression mechanism and to keep the generality of analyses, grammar based compression was proposed.

∗Corresponding author
Email addresses: schauss@cs.uni-frankfurt.de (Manfred Schmidt-Schauß),
georg@thi.informatik.uni-frankfurt.de (Georg Schnitger)
†Partially supported by DFG SCHN503/4-1
It is used for string compression and algorithms on strings \cite{10,12,8} as well as for compressions of trees and algorithms on them \cite{3,4,5}.

A central algorithmic problem used as a subalgorithm in several other algorithms on compressed data is the following: given two compressed representations $r_1$ and $r_2$, say of strings $s_1$ and $s_2$, respectively, decide whether $s_1 = s_2$. The first efficient algorithm that works without prior decompression is Plandowski’s algorithm \cite{10,11}. It uses grammars as compression device and shows that the equality test can be done in time polynomially in the size of the grammars. An improvement of this equality test is in \cite{9} where an algorithm is described that works in time $O(n^3)$, where $n$ is the size of the grammar.

Randomized algorithms for variants of this test are described in \cite{6} using 2 matrices and in \cite{2} for the generalisation to two-dimensional strings, where a polynomial interpretation is used.

In this paper we describe a randomized algorithm for Plandowski’s equality problem that runs in quadratic time even using a logarithmic cost measure for arithmetic operations. It is correct for identical strings and does not detect inequality with a small probability $\delta$, and with $\delta^n$ after $n$ repetitions of the test. The algorithm requires prime numbers that are exponential in the size of $G$, which leads to the conjecture that also prime numbers of polynomial size are sufficient.

The randomized equality test is faster than the Lifshits-test \cite{9}, which has a cubic running time, but presumably an $O(n^4)$ running time using the logarithmic cost measure. The equality test is also applicable to grammar-compressed ranked trees by applying it to the SLCF grammar representing the preorder traversals, which can be generated in linear time (see \cite{3,4}).
2. Grammars and Equality

Definition 2.1. (a) A straight-line context-free grammar (SLCFG) $G$ is a quadruple $(\Sigma, \mathcal{N}, S, \mathcal{R})$ where

1. $\Sigma$ is a finite alphabet, (we assume $|\Sigma| = O(1)$)
2. $\mathcal{N} = \{B_1, \ldots, B_N\}$ is a set of nonterminals,
3. $S = B_N$ is the start symbol and
4. $\mathcal{R}$ is a finite set of productions. A production has either the form $B_i \to B_j B_k$ for $i > j, k$ or $B_i \to a$ for $a \in \Sigma$. Moreover for each nonterminal $B_i$ there is exactly one production in $\mathcal{R}$.

(b) Every nonterminal $A \in \mathcal{N}$ generates exactly one string $\text{val}(A)$. The string generated by the start symbol $B_N$ is denoted by $\text{val}(G)$.

(c) The size $|G|$ of $G$ is the number of productions of $\mathcal{R}$.

The length of $\text{val}(G)$ may be as large as $2^{|G|}$. As an example, for every integer $n > 1$ there is an SLCFG $G_n$ of size $\lceil (\log_2(n)) \rceil$ such that $\text{val}(G_n)$ is a string of 0’s of length $n$.

The $EQ$ problem for SLCFGs is: given an SLCFG $G$ and two nonterminals $A_1, A_2$ determine whether $\text{val}(A_1) = \text{val}(A_2)$.

The $EQPREF$ problem for SLCFGs is: given an SLCFG $G$ and two nonterminals $A_1, A_2$ determine whether $\text{val}(A_1) = \text{val}(A_2)$ and if $\text{val}(A_1) \neq \text{val}(A_2)$ then determine the length of the longest common prefix of $\text{val}(A_1)$ and $\text{val}(A_2)$.

Let $b \geq |\Sigma| + 1$ be a number (the base) and let $\text{num}$ be an injective function $\text{num} : \Sigma \to \{1, \ldots, b - 1\}$. Observe that a string $d = d_1 \cdots d_m$ over $\Sigma$ can be interpreted as the $b$-ary representation of the natural number $\text{num}_b(d) = \sum_{i=0}^{m-1} \text{num}(d_{m-i}) \cdot b^i$. Hence $d = d'$ iff $\text{num}_b(d) = \text{num}_b(d')$. 

3
The EQ problem is non-trivial, since the strings \( \text{val}(A) \) may have length exponential in \( |G| \), i.e. as large as \( 2^{|G|} \). Thus the associated natural numbers \( \text{num}_b(A) := \text{num}_b(\text{val}(A)) \) may have size exponential in \( |G| \). Therefore we determine \( \text{num}_b(A) \) modulo a randomly selected integer \( m \) smaller than some bound and check whether \( \text{num}_b(A_1) \equiv \text{num}_b(A_2) \mod m \) holds.

For computing running times we apply a slightly adapted logarithmic cost measure, i.e., \( n_1 \circ n_2 \) requires running time \( O(\log n) \) for addition-like arithmetic operations where \( n = \max(n_1, n_2) \) and \( O(\log n \cdot M(n)) \) for multiplication and division where \( M(n) \) is significantly smaller than \( \log n \). We also use \( M(n^k) \leq kM(n) \).

3. A Randomized Equality Test

Let \( e = 2.718\ldots \) be the Euler-number. The randomized equality test for natural numbers has the following properties.

**Fact 3.1.** Let \( c \geq e \) be arbitrary and let \( a \) be a positive integer. For any two natural numbers \( x, y < a \), if \( x \neq y \) then

\[
x \equiv y \mod m
\]

holds with probability at most 0.5, provided a number \( m \leq (2 \ln a)^2 \) is selected uniformly at random.

**Proof.** First we show that asymptotically the number of prime divisors of an integer \( a \) is less than \( \pi(\ln a) \) where \( \pi(z) \) is the number of primes less than \( z \). First observe that \( \sum_{p \leq N} \ln p \sim N \), where we sum over all primes at most \( N \) (see [1]). As a consequence \( \ln(\prod_{p \leq N} p) \geq N/(1 + \delta) \) for \( \delta > 0 \) and hence

\[
\prod_{p \leq N} p \geq e^{N/(1+\delta)}.
\]

Let \( 0 < x < a \) and \( P \) be the set of all primes \( p \) with \( x \equiv 0 \mod p \). Then \( \prod_{p \in P} p \mid x \) and in particular \( \prod_{p \in P} p < a \).
If $|P| \geq \pi((1 + \delta) \cdot \ln a)$, then by our previous argument

$$a \leq \prod_{p, p \leq (1 + \delta) \cdot \ln a} p \leq \prod_{p \in P} p < a$$

and hence $|P| \leq \pi(\ln a)$ follows.

In an interval $[k, k^2]$, the number of multiples $k'p \in [k, k^2]$ of primes $p \in [k, k^2]$ can be estimated as

$$\int_{k}^{k^2} \frac{k^2}{(x \ln(x))} dx = k^2 (\ln \ln(k^2) - \ln \ln k) = k^2 \ln 2.$$

Note that every multiple occurs only once in the interval. Since some primes from $P$ may be in the interval, a lower bound for the number of integers in $[k, k^2]$ with a prime divisor not in $P$ is $k^2 \ln 2 - |P|k$. For $k = (2 \ln a)$, we obtain a ratio 

$$(k^2 \ln 2 - |P|k)/k^2 = \ln 2 - 1/(2 \ln \ln a) > 0.5.$$ 

If we select $m \leq (2 \ln a)^2$ then $x - y \equiv 0 \mod m$ holds with probability at most 0.5. \hfill \square

4. Equality-Test Algorithms

By utilizing a table with $A \rightarrow |\text{val}(A)|$ computed as $|\text{val}(A)| := |\text{val}(B)| + |\text{val}(C)|$ if $A \rightarrow BC$ is the production for $A$, we obtain:

**Observation 4.1.** For an SL{CFG} $G$ the length of $\text{val}(A)$ can be determined simultaneously for all nonterminals $A$ in time $O(|G| \cdot \log |\text{val}(G)|)$.

Given a positive integer $m$, we store the values $b^{\text{val}(A)} \mod m$ in a table $\tau$ computed as follows: $\tau(A) = b \mod m$ for $A \rightarrow a$, and $\tau(A) = \tau(B) \cdot \tau(C) \mod m$ for $A \rightarrow BC$. We also determine $\text{num}_a(A) \mod m$ for all nonterminals $A$ using another table $\sigma$ computed as follows: $\sigma(A) = \text{num}(a) \mod m$ if $A \rightarrow a$ and $\sigma(A) = \sigma(B) \cdot \tau(C) + \sigma(C) \mod m$ if $A \rightarrow BC$. Since addition and multiplication can be done modulo $m$, the entries in $\sigma, \tau$ are smaller than $m$. 

5
Observation 4.2. Assume that an SLCFG $G = (\Sigma, N, S, R)$ and a positive integer $m$ is given. Then the numbers $b^{\text{val}(A)} \mod m$ and $\text{num}_b(A) \mod m$ can be determined simultaneously for all $A \in N$ in time $O(|G| \cdot \log m \cdot M(m))$.

Checking whether $\text{val}(A_1) = \text{val}(A_2)$ holds, requires first to determine the lengths $|\text{val}(A)|$ for all nonterminals $A$ with Observation 4.1. We then randomly select a number $m \leq (2 \cdot \ln |\text{val}(G)|)^2 = O(|\text{val}(G)|^2)$ and determine $\text{num}_b(G) \mod m$ with Observation 4.2. Finally, with Fact 3.1, we do not detect inequality with probability at most 0.5.

Theorem 4.3 (Modulo-test for small primes). Assume that an SLCFG $G$ and two nonterminals $A_1, A_2$ of $G$ are given.

If $\text{val}(A_1) \neq \text{val}(A_2)$, then we do not detect inequality with probability at most 0.5. If $\text{val}(A_1) = \text{val}(A_2)$ our algorithm is always correct. The running time is bounded by $O(|G| \cdot \log |\text{val}(G)| \cdot M(|\text{val}(G)|))$.

Since $|\text{val}(G)| \leq 2^{|G|}$, the running time is at most quadratic if we disregard the $M(\cdot)$-contribution.

As a special case, we look at a one-letter alphabet. If we have to check whether $\text{val}(A_1) = \text{val}(A_2)$ holds it is sufficient to check whether $|\text{val}(A_1)| = |\text{val}(A_2)|$. Therefore we use Fact 3.1 with $a = |\text{val}(G)|$, and can exploit $\ln a \leq |G|$. We randomly select a number $m \leq (2 \cdot |G|)^2$ and determine $\text{num}_b(A)$ with Observation 4.2. Finally, with Fact 3.1, we do not detect inequality with probability at most 0.5.

Theorem 4.4. For a one-letter alphabet $\Sigma$ Theorem 4.3 holds with a running time $O((|G| \cdot \log |G| \cdot M(|G|))$.

Note that this is faster than the naive comparison $|\text{val}(A_1)| = |\text{val}(A_2)|$, which runs in time $O(|G| \cdot \log |\text{val}(G)| \cdot M(|G|))$, resp. quadratic in the worst-case.
Remark 4.5 (Lower Bound for Primes). We show that prime factors greater than \((\ln 2) \cdot |G|\) are required for the equality test. More rigorously:

Let \(\Sigma = \{0, \ldots, b - 1\}\), \(n\) be an integer and let \(G\) be an SLCFG such that \(\text{val}(A_0)\) is a string of only 0’s of length \(n\), and \(\text{val}(A_1)\) is a string of only 1’s of length \(n\). \(G\) can be chosen such that \(|G| \leq 2^{\lceil \log_2(n) \rceil}\). Let \(b < p_1 < p_2 < \ldots p_k \leq N\) be the sequence of all primes greater than \(b\) and bounded by \(N\), and let \(n = (p_1 - 1) \cdot \ldots \cdot (p_k - 1)\). For all \(p \in \{p_1, \ldots, p_k\}\):

\[
\sum_{i=0}^{p-2} b^i \cdot (b - 1) = b^{p-1} - 1 \equiv 0 \mod p, \text{ since } p > b.
\]

Hence \((\sum_{i=0}^{p-2} b^i) \equiv 0 \mod p\), and \(\text{val}(A_0)\) and \(\text{val}(A_1)\) are indistinguishable by the modulo algorithm for all primes \(p \in \{p_1, \ldots, p_k\}\). Using the equations in the proof of Fact 3.1, we obtain \(\ln n = \ln \prod_{p,b<p\leq N}(p - 1) \leq \ln \prod_{p,p\leq N} p \sim N\). Thus \(\ln \prod_{p,p\leq N} p < 1.1N\), for \(N\) not too small, and also \(\ln n = (\ln 2) \cdot \log_2(n) \geq (\ln 2) \cdot (|G| - 1)\). Thus \(N > c \cdot |G|\) with \(c \approx \ln 2/1.1 \approx 0.6\).

Hence, the upper bound on primes must be larger than \((\ln 2) \cdot |G|\).

Assuming ideal properties, the following computation is possible and gives a rough estimate for the necessary range of tiny primes (mathematically unsafe, but useful as a practical hint). Given \(|G|\) and assuming \(|G| \geq b\), there are at most \(|G|^{|G|^2}\) different grammars of size \(|G|\). Assuming that the generated numbers are all different and are exactly \(1, \ldots, |G|^{|G|^2}\), then using Fact 3.1 we obtain that primes in the range up to \(\log(|G|) \cdot |G|^2\) have to be chosen for a useful modulo test with tiny primes.

Remark 4.6. Our algorithm can also be applied to the EQPREF-problem. This problem was also considered in [6]. We perform an interval bisection method. Given an SLCFG \(G\), a single bisection step requires to compute the length, and two grammars \(G_1, G_2\) according to the bisection, where \(G_1, G_2\) are smaller than \(G\). The number of bisection steps is \(O(\log |\text{val}(G)|)\).
The construction can be done in running time $O(|G| \cdot \log |\text{val}(G)|)$ and the test in time $O(|G| \cdot \log |\text{val}(G)| \cdot M(|\text{val}(G)|))$. The test must be repeated $O(\log \log |\text{val}(G)|)$ times in every step. This sums up to $O(|G| \cdot \log^2 |\text{val}(G)| \cdot M(|\text{val}(G)|) \cdot \log \log |\text{val}(G)|)$. Under a uniform cost measure we obtain $O(|G| \cdot \log |\text{val}(G)| \cdot \log \log |\text{val}(G)|)$.

5. Summary

The following table summarizes the complexities of the different randomized or sample equality tests, where we use the logarithmic cost measure, and omit the $M(\cdot)$-factor.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>worst case</th>
<th>general case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulo-test with small primes</td>
<td>$O(</td>
<td>G</td>
</tr>
<tr>
<td>Modulo-test with tiny primes</td>
<td>$O(</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>(of order $\log</td>
<td>G</td>
</tr>
<tr>
<td>Modulo-test for $</td>
<td>\Sigma</td>
<td>= 1$</td>
</tr>
</tbody>
</table>

Remark 5.1. The complexities for any of the above case are $O(|G|)$ using a uniform cost measure for arithmetic.

References


