Periodic sets of integers

Armando B. Matos
Centro de Informática de Universidade do Porto, Rua do Campo Alegre 823, 4100 Porto, Portugal

Communicated by M. Nivat
Received June 1992
Revised December 1992

Abstract

Matos, A.B., Periodic sets of integers, Theoretical Computer Science 127 (1994) 287–312.

Consider the following kinds of sets:
- the set of all possible distances between two vertices of a directed graph;
- any set of integers that is either finite or periodic for all n greater or equal to some n_0 (such a set is called *ultimately periodic*);
- a context-free language over an alphabet with one letter (such a language is also regular);
- the set of all possible lengths of words of a context-free language.

All these sets are isomorphic relatively to the operations of union (or sum), concatenation and Kleene (or transitive) closure. Furthermore, they all share a particularly important property which is not valid in some similar algebraic structure – the concatenation is commutative.

The purpose of this paper is to investigate the representation and properties of these sets and also the algorithms to compute the operations mentioned above. The concepts of linear number and Δ-sum are developed in order to provide convenient methods of representation and manipulation.

It should be noted that although Δ-sums and regular expressions (or finite automata) over a one-letter alphabet denote essentially the same sets, the corresponding algebras are quite different. For example, it is always possible to eliminate the closure and concatenation operations from a Δ-sum by expanding it as a sum of linear numbers. No such elimination is possible for regular expressions (although special forms of regular expressions or finite automata are sufficient to denote regular sets *over one-letter alphabets*). The algorithms using Δ-sums are often faster and simpler than those based on finite automata or regular expressions over a one-letter alphabet. We think that this improvement comes from the fact that a set of words over one letter is represented by the set of their lengths and manipulated by arithmetic operations.

We apply these methods to the first kind of sets listed above and present new algorithms dealing with a variety of problems related to distances in directed graphs.

1. Introduction

Ultimately periodic sets of *integers* correspond to *lengths* of the words of regular languages and are isomorphic to regular sets over a one-letter alphabet. They can, of

Correspondence to: A.B. Matos, Centro de Informática de Universidade de Porto, Rua do Campo Alegre 823, 4100 Porto, Portugal. Email: acm@ ncc.up.pt.

course, be characterized in standard ways (finite automata, regular expressions or linear grammars). However, in this paper we argue that, from the algorithmic point of view, a more direct and "arithmetic" characterization may be preferable since more compact representations and more efficient algorithms seem to be possible.

As an example of an ultimately periodic set, consider

$$\{0,3\}\cup\{2+4i:\ i\geqslant 0\}\cup\{7+3i:\ i\geqslant 0\}=\{0,2,3,6,7,10,13,14,16,\ldots\}.$$

This set is periodic for all integers greater than or equal to 5 and the (shortest) period is 12. In this paper those sets will be denoted by *Δ-sums* which are essentially sums like

$$0\oplus 3\oplus(2+4k)\oplus(7+3k),$$

where each term is called a *linear number*. This specific Δ-sum denotes the set above. Any ultimately periodic set of natural numbers can be denoted by a Δ-sum.

We formulate in more general terms the correspondence between a language and the lengths of its words. Given a language L over an alphabet Σ we define N_L as the set of all lengths of words in L

$$N_L=\{|w|:w\in L\}.$$

Clearly N_L is a subset of $\mathbf{N}$, the set of nonnegative integers. Suppose that L belongs to a certain family of languages $\mathscr{L}$. An interesting question is: how is the family of sets N_L related to $\mathscr{L}$? In particular, we consider the families of the Chomsky hierarchy (see for instance [9]).

It is not difficult to see that, for every recursively enumerable set of integers S, there is a type-0 language L such that $N_L=S$ and reciprocally that N_L is recursively enumerable for every type-0 language L. In other words, lengths of words of type-0 languages correspond exactly to recursively enumerable sets of integers.

To our knowledge, no simple characterization of the lengths of words for languages of type 1 is known.

Consider now a context free language L (type 2) characterized by a grammar with a set of productions P. We define a new language L' over a single-letter alphabet $\{a\}$ by replacing in every production in P every terminal symbol by a. It is easy to see that N_L and $N_{L'}$ are equal, i.e. the lengths of the words in L and L' are the same. The language L' is also regular and the family of sets N_L is the family of ultimately periodic sets of integers. We can summarize the correspondence between the languages and the length of their words as in Table 1.

Table 1

Family of languages	Length of words
Type 0	Recursively enumerable sets of integers
Type 1	?
Type 2	Ultimately periodic sets of integers
Type 3	Ultimately periodic sets of integers

We see that Δ-sums are isomorphic to regular (and to context free) languages over an alphabet with one letter. Their closure properties are a consequence of this isomorphism but we prove them directly using the Δ-sum formalism because the insight so gained will be helpful for the design of the correspondent algorithms.

Concepts from standard language theory – such as finite automata or regular expressions – can be used to represent and manipulate sets denoted by Δ-sums. However, the direct use of Δ-sums, besides being *simpler*, gives *shorter descriptions and faster algorithms.* As a simple example, consider the Δ-sum $17 \oplus (2+30k)$; it is easy to find an equivalent (nondeterministic) automaton which has 49 states and the equivalent regular expression

$$\overbrace{aa\cdots a}^{17} + aa(\overbrace{aa\cdots a}^{30})^*$$

has 53 symbols! Algorithms with automata or regular expressions this large are extremely inefficient. With Δ-sums, sets of words over one letter are represented as sets of their lengths and manipulated by arithmetic operations; this is the reason for the improvement in compactness and efficiency. For instance, the concatenation of a^n with a^n is executed as a sum $n+n$, which takes time $O(\log n)$ instead of $O(n)$.

Although Δ-sums and regular expressions over one letter denote isomorphic sets, *they are not identical from the algebraic point of view.* For instance, the operators of concatenation and closure $(*)$ contained in a Δ-sum can always be eliminated by expanding it as a sum of linear numbers (see Section 3.3). Such elimination is not possible in general for regular expressions.

Clearly, the properties of Δ-sums correspond to the specific properties of the regular expressions (or finite automata) *with a one-letter alphabet.* For instance, the fact that Δ-sums denote ultimately periodic sets corresponds to the possibility of writing every regular expression over a one-letter alphabet in the form

$$a^{n_1}+\cdots+a^{n_k}+(a^{m_1}+\cdots+a^{m_l})(a^p)^* \quad (k\geqslant 0,\ l\geqslant 0).$$

The properties of regular sets over two- or more-letter alphabet are much more complex. For instance, Conway [3] proved that any complete system of identities *on a two-letter alphabet* must contain an infinity of identities in two or more variables. As another example, the star height (see for instance [8]) of a rational expression *over a one-letter alphabet* is 1 (see the expression above). *In this paper we explore the relative simplicity of regular sets over a one-letter alphabet* arguing that they are conveniently denoted by Δ-sums.

The properties of languages with one-letter alphabets – or equivalently, the properties of the lengths of the words of a language – have been studied before. For instance, it is proved in [6] that every context-free language over an alphabet with one letter is regular, it is mentioned in [9] (Exercise 6.8 in pp. 142 and 143) that the set of the lengths of the words of a regular languages over an alphabet with one letter is ultimately periodic and in [4, Ch. V] the relationship between regular sets with a single-letter alphabet and sets of integers is thoroughly discussed; in particular, the "finite union of arithmetic progressions" in [4, Ch. V, Proposition 1.1] corresponds to our Δ-sums. The main contribution of this paper is algorithmic: a new representation of regular languages over an alphabet with one letter is developed and applied to language and graph algorithms.

A brief summary of Sections 2–5 is as follows. In Section 2, we present basic definitions about linear numbers and Δ-sums, and state some known properties (related to the Frobenius problem) about the possibility of obtaining a number as a linear combination of two fixed integers using nonnegative integer coefficients. These results are useful to characterize the concatenation and closure of Δ-sums.

Closure and algebraic properties of Δ-sums are studied in Section 3. The set of Δ-sums forms a closed semiring [11] where the "multiplication" (which corresponds to concatenation) is commutative.

In Section 4, algorithms to compute the results of operations between Δ-sums are presented and their complexity is analyzed. We implemented these algorithms in Prolog and checked the examples of this work.

The study of all possible lengths of paths between two nodes of a directed graph is an important application of Δ-sums. It is well known that the set of all paths between two nodes can be represented by a regular expression. If, instead of using alphabet symbols, we use edge lengths, we get a similar algebraic structure with one difference: concatenation (which corresponds to the sum of path lengths) is commutative. This difference makes it possible to have simpler algorithms and shorter representations. A Δ-sum is a representation of all possible lengths of paths between two nodes; when expanded as a sum of linear numbers, it allows straightforward answers to a number of questions related to those lengths. In Section 5, we discuss the use of Δ-sums to solve several problems about distances and describe an efficient method for the computation of a Δ-sum expression denoting the set of all distances between two vertices. The reader may find interesting to start by looking at the examples in that section which illustrate an important application of the concept of Δ-sum.

2. Linear numbers and Δ-sums

A *linear number* represents a set of integers; for example, $(2,7)$ or, equivalently, $2+7k$ denotes the (linear) set

$$\{2+7i: i\geqslant 0\}=\{2,9,16,\ldots\}.$$

This nomenclature is taken from [9, p. 143] where a language $\{0^{p+iq}: i\geqslant 0\}$ is called linear.

Definition 2.1. *A linear number* is a pair (a,b) of nonnegative integers. It denotes the set

$$\mathscr{S}((a,b))=\{a+bi: i\geqslant 0\}.$$

The *period* of the linear number is b. We use the notation $a+bk$ for the linear number (a, b); it denotes the set of numbers having the form $a+bk$ where k is a nonnegative integer. The set denoted by a linear number is called a *linear set*. Note that such a set either contains exactly one element (if $b=0$) or is infinite (if $b\geqslant 1$).

The notation $a+bk$ for linear numbers is similar to the notation of complex numbers by $a+bi$. In both cases we have a symbolic representation for a pair of numbers which does not involve any algebraic operation.

The linear number $a+0k$ denotes the singleton set $\{a\}$ and may be written simply a. Similarly, bk and $a+k$ stand for $0+bk$ and $a+1k$, respectively. A linear number $a+bk$ where $b\neq 0$ is called *noninteger*.

Definition 2.2. A set S of nonnegative integers is *ultimately periodic* if there exist integers n_0, p (period) m, $i_1, \dots, i_m$ with

$$0\leqslant i_1<i_2<\cdots<i_m<p \quad (m\geqslant 0),$$

and a finite set F of integers less n_0 such that

$$S=F\cup\{n_0+i_1+pl\colon l\geqslant 0\}\cup\cdots\cup\{n_0+i_m+pl\colon l\geqslant 0\}.$$

Any positive multiple of p is also a period.

Note that *every finite set is ultimately periodic* (take $m=0$ in the definition).

Definition 2.3. A *Δ-sum* is a finite "sum"

$$x_1 \oplus x_2 \oplus \cdots \oplus x_m,$$

where each x_i is a linear number and $m\geqslant 0$. It denotes the set

$$\mathcal{S}(x_1 \oplus \cdots \oplus x_m)=\mathcal{S}(x_1)\cup\cdots\cup\mathcal{S}(x_m).$$

If $m=0$ the Δ-sum is written $\emptyset$ end denotes the set $\emptyset$. Every noninteger linear number is a Δ-sum (take $m=1$ in the definition). By definition two Δ-sums are *equal* if they denote the same set

$$x_1=x_2 \Leftrightarrow \mathcal{S}(x_1)=\mathcal{S}(x_2).$$

The $+$ in $a+bk$ binds more tightly than the $\oplus$ operator so that

$$a+bk\oplus c+dk$$

stands for

$$(a+bk)\oplus(c+dk).$$

The sum of Δ-sums is defined in the obvious way. Their concatenation is defined as follows.

Definition 2.4. If x_1 and x_2 are Δ-sums, their concatenation x_1x_2 denotes the set

$$\mathscr{S}(x_1x_2)=\{a+b\colon a\in\mathscr{S}(x_1),\ b\in\mathscr{S}(x_2)\}.$$

It will be proved later (Theorem 3.3) that this set can always be denoted by a Δ-sum. The concatenation of m Δ-sums all equal to x is represented by x^m. For all x we have $\mathscr{S}(x^0)=\{0\}$.

The name *concatenation* is borrowed from language theory; the following concatenation of two languages over the alphabet $\{1\}$ illustrates this origin

$$\{1^{a+bi}\colon i\geqslant 0\}\ \{1^{a'+b'i}\colon i\geqslant 0\}=\{1^n\colon n\in\mathscr{S}((a+bk)(a+b'k))\}.$$

The left-hand side is a concatenation of two sets; on the right-hand side, the expression $(a+bk)(a+b'k)$ is a concatenation of two linear numbers.

As an example, consider the set denoted by the concatenation $(2+3k)(1+4k)$,

$$\begin{aligned}\mathscr{S}((2+3k)(1+4k))&=\mathscr{S}((3)(3k)(4k))\\&=\{3\}(\{0,3,6,9,\dots\}\cup\{4,7,10,13,\dots\}\\&\quad\cup\{8,11,14,17,\dots\}\cup\cdots)\\&=\{3\}\{0,3,4,6,7,8,9,10,\dots\}\\&=\{3,6,7,9,10,11,12,\dots\}\\&=\mathscr{S}(3\oplus 6\oplus 7\oplus 9+k).\end{aligned}$$

As the two Δ-sums denote the same set, they are equal:

$$(2+3k)(1+4k)=3\oplus 6\oplus 7\oplus 9+k.$$

Definition 2.5. Let x be a Δ-sum. The *Kleene* closure (or just closure) of x is represented by x^* and denotes the set

$$\mathscr{S}(x^*)=\bigcup_{m\geqslant 0}\mathscr{S}(x^m).$$

We see that x^* denotes the set containing the integers that can be obtained by summing any (nonnegative) number of integers of $\mathscr{S}(x)$. In *particular*, 0 belongs to $\mathscr{S}(x^*)$ because $\mathscr{S}(x^0)$ is $\{0\}$. It will be proved later (Theorem 3.6) that the Kleene closure of x can always be denoted by a Δ-sum.

Consider the expression $(2+3k)^*$; we get

$$\begin{aligned}\mathscr{S}((2+3k)^*)&=\{0\}\cup\{2,5,8,11,\dots\}\cup\{4,7,10,13,\dots\}\cup\{6,9,12,15,\dots\}\cup\cdots\\&=\{0,2,4,5,6,7,8,\dots\}\\&=\mathscr{S}(0\oplus 2\oplus 4+k).\end{aligned}$$

Theorems 3.3 and 3.6 provide a justification for the following recursive definition of Δ-sum expressions.

- A linear number is a Δ-sum expression.
- $\emptyset$ is a Δ-sum expression.
- If x_1 and x_2 are Δ-sum expressions then so are $x_1 \oplus x_2$, $x_1 x_2$ and x_1^*.

When writing Δ-sum expressions, we use the following priority order (increasing order)

$$\oplus, +, \cdot, *,$$

where, of course, $+$ is not an operator over Δ-sums.

For reasons that will get clear later (see Section 5) Δ-sum expressions will also be called *path length* or *distance* expressions.

The concatenation of $c+ak$ and $d+bk$ denotes the set of all integers $c+d+ai+bj$ where i and j are nonnegative integers. Theorem 2.6 characterizes this set for the case $c=d=0$.

Theorem 2.6. *Let a and b be positive integers, d their greatest common divisor, n the integer $(a-d)(b-d)/d$ and S the set*

$$\{ai+bj:\ i,j\geqslant 0\}.$$

Then

(1) *all members of S are multiples of d;*
(2) *for all $i\geqslant 0$, the integer $n+di$ belongs to S;*
(3) *the integer $n-d$ does not belong to S;*
(4) *the number of elements of S less than n is $n/(2d)$.*

Proof. Follows easily from known results about the Frobenius' problem (see for instance [10, 12, 13]). □

As a simple example, consider $a=6$ and $b=15$. The corresponding set is

$$\mathscr{S}=\{6i+15j:\ i,j\geqslant 0\}=\{6,12,15,18,\dots\}.$$

We have $d=3$ and $n=(3\times 12)/3=12$. Let G_{12} be the set of integers greater than or equal to 12; from parts (1) and (2) we have

$$S\cap G_{12}=\{12+3i:\ i\geqslant 0\}.$$

Part (3) says that 9 is not a member of S. We get

$$S=\{0,6\}\cup\{12+3i:\ i\geqslant 0\}.$$

The first set has two elements in accordance with part (4) $(2=12/(2\times 3))$.

3. Properties of Δ-sums

In this section we study the closure and algebraic properties of Δ-sums. We show that the family of ultimately periodic sets is exactly the family of sets characterized by Δ-sums.

3.1. Δ-Sums and ultimately periodic sets

We now prove that every ultimately periodic set is denoted by some Δ-sum and reciprocally that sets denoted by Δ-sums are ultimately periodic.

Theorem 3.1. *Every ultimately periodic set is denoted by a Δ-sum.*

Proof. Any ultimately periodic set S can be written as a disjoint union (see Definition 2.2 for the meaning of the symbols used below)

$$S = F \cup \{n_0 + i_1 + pl:\ l \geqslant 0\} \cup \cdots \cup \{n_0 + i_m + pl:\ l \geqslant 0\},$$

where F is the finite set containing the elements of S which are less than n_0

$$F = \{a_1, a_2, \ldots, a_k\}.$$

Then S is denoted by the Δ-sum

$$a_1 \oplus \cdots \oplus a_k \oplus ((n_0 + i_1) + pk) \oplus \cdots \oplus ((n_0 + i_m) + pk).$$

Note that in this expression the $+$ symbol is used both to denote the addition of integers (in $n_0 + i_1, \ldots$) and as a notation for linear numbers. $\square$

Theorem 3.2. *Every Δ-sum denotes an ultimately periodic set.*

Proof. It is enough to consider the sum of two linear numbers

$$a + bk \oplus c + dk.$$

Let S be the set denoted by this Δ-sum and let m be the least common multiple of b and d. We show that for all $i \geqslant 0$ the set $S \cap S_i$ where

$$S_i = [n_0 + im, n_0 + (i+1)m - 1]$$

(where by $[a, b]$ we represent the set $\{a, a+1, \ldots, b\}$) is always the same providing that n_0 is large enough. This proves that S is ultimately periodic.

Consider an element of S. If it is of the form

$$x = a + bi,$$

for some $i \geqslant 0$, then the integer

$$a + bi + m = a + bi'$$

also belongs to S. A similar conclusion holds for elements $c+di$ showing that

$$S \cap S_i \subseteq S \cap S_{i+1}.$$

Reciprocally, if $a+bi \in S$ and $bi \geqslant m$, we have (recall that m is multiple of b and d)

$$a+bi-m=a+bi' \in S$$

(where $i' \geqslant 0$) and, if $c+di \in S$ with $di \geqslant m$,

$$c+di-m=c+di' \in S$$

(where again $i' \geqslant 0$). This shows that for n_0 large enough

$$S \cap S_i \subseteq S \cap S_{i-1},$$

which concludes the proof that S is ultimately periodic. □

As an example, consider the Δ-sum presented at the beginning of Section 1. It can also be written as

$$0 \oplus 2 \oplus 3 \oplus 6+12k \oplus 7+12k \oplus 10+12k \oplus 13+12k \oplus 14+12k \oplus 16+12k,$$

showing that it is ultimately periodic with period 12.

3.2. Closure properties

The closure properties of Δ-sums are a consequence of the fact that they are isomorphic to regular languages over a one-letter alphabet. Yet, it is instructive to prove them directly using the Δ-sum formalism because the insight so gained will be helpful for the design of the correspondent algorithms.

By definition it is clear that the sum of two Δ-sums is a Δ-sum. We now consider the concatenation of two Δ-sums and prove that it is also a Δ-sum.

Theorem 3.3. *Δ-sums are closed under addition* ($\oplus$) *and concatenation.*

Proof. Obvious for the addition. Consider the concatenation of two linear numbers

$$(a+bk)(a'+b'k).$$

From Theorem 2.6 it is clear that this concatenation is a sum

$$a_1 \oplus \cdots \oplus a_m \oplus n+dk,$$

showing that the concatenation of two linear numbers is a Δ-sum. The general result follows easily. □

As an example, consider the concatenation

$$(1+12k)(2+42k)(3+66k).$$

It can be shown that it denotes the set

$$\{6, 18, 30\} \cup \{42 + 6i\colon i \geqslant 0\},$$

which is also denoted by the Δ-sum

$$6 \oplus 18 \oplus 30 \oplus 42 + 6k.$$

We now study the Kleene closure of a Δ-sum, considering first the closure of a single linear number.

Lemma 3.4. *Let $a + bk$ be a linear number different from 0. Then*

$$(a + bk)^* = F \oplus c + dk,$$

where d is the greatest common divisor of a and b, c is the least common multiple of a and b and F is a finite sum of integers all of them multiples of d.

Proof. An integer belongs to $\mathscr{S}(a + bk)^*$ if and only if, for some nonnegative integers $j, i_1, \ldots, i_j$ with $j \geqslant 0$, it can be written as

$$a + bi_1 + a + bi_2 + \cdots + a + bi_j,$$

i.e. if it is of one of the following two kinds:

$$\begin{cases} 0, & \\ aj + bi & \text{for } j \geqslant 1,\ i \geqslant 0. \end{cases}$$

These integers are denoted by the following Δ-sum, whre $n = b/d$:

$$x = 0 \oplus a + bk \oplus 2a + bk \oplus \cdots \oplus na + bk.$$

The n integers ia with $1 \leqslant i \leqslant n$ are all multiples of d and, when taken modulo b, they are all distinct, which implies that $F \oplus c + dk$ is a Δ-sum expression for the closure of $a + bk$; here $c = na = ab/d$ is the least common multiple of a and b and F contains the integers of x less than c. The proof remains valid if $a = 0$, $b \neq 0$ ($d = b$, c defined as 0) or if $a \neq 0$, $b = 0$ ($d = a$, c defined as 0). However, when $a = b = 0$, we have simply

$$(a + bk)^* = 0. \qquad \square$$

As an example, we have

$$(4 + 3k)^* = 0 \oplus 4 \oplus 7 \oplus 8 \oplus 10 \oplus 11 \oplus 12 + k.$$

This expression can be simplified as

$$0 \oplus 4 \oplus 7 \oplus 8 \oplus 10 + k.$$

We now generalize this result considering the closure of an arbitrary sum of linear numbers. The fact that it contains (at most) only one noninteger linear number is somewhat surprising.

Lemma 3.5. *Consider a Δ-sum x expressed as a finite sum of linear numbers,*

$$x = a_1 + b_1 k \oplus \cdots \oplus a_m + b_m k.$$

Then x^* can also be denoted by a finite sum of linear numbers,

$$x^* = F \oplus c + dk,$$

where

$$d = \gcd(a_1, \ldots, a_m, b_1, \ldots, b_m), \tag{1}$$

$$c = \operatorname{lcm}(a_1, \ldots, a_m, b_1, \ldots, b_m), \tag{2}$$

and F is a finite sum of integers.

Proof. Similar to Lemma 3.4. □

Theorem 3.6. *Δ-sums are closed under the Kleene closure. The closure of any Δ-sum contains at most one noninteger linear number.*

Proof. Use the previous lemma and induction on the structural complexity of the Δ-sum. □

3.3. The commutative semiring of Δ-sums

The following properties of Δ-sums are easy to check. If x, y and z are Δ-sums we have

$(x \oplus y) \oplus z = x \oplus (y \oplus z)$	(associativity of $\oplus$),
$x \oplus y = y \oplus x$	(commutativity of $\oplus$),
$x \oplus \emptyset = \emptyset \oplus x = x$	(neutral element of $\oplus$),
$(xy)z = x(yz)$	(associativity of concatenation),
$xy = yx$	(commutativity of concatenation),
$x0 = 0x = x$	(neutral element of concatenation),
$x\emptyset = \emptyset x = \emptyset$	(product by 0),
$x(y \oplus z) = xy \oplus xz$	(distributivity),
$(x \oplus y)z = xz \oplus yz$	(distributivity).

These properties show that

$$(\Delta, \oplus, \cdot, \emptyset, 0),$$

where $\cdot$ stands for concatenation, is a *commutative semiring* [11], i.e. $(\Delta, \oplus, \emptyset)$ and $(\Delta, \cdot, 0)$ are commutative monoids where for all x, y and z we have $x\emptyset = \emptyset x = \emptyset$, $x(y \oplus z) = xy \oplus xz$ and $(x \oplus y)z = xz \oplus yz$.

For space reasons, other properties of Δ-sums are not discussed here but we would like to emphasize three important (although related) differences between the algebras of regular expressions (see for instance [3]) and of Δ-sums (these properties were discussed in Section 3).

(1) Concatenation is commutative.

(2) The closure and concatenation operators can be eliminated from every Δ-sum.

(3) Arithmetic can be used for the computation of concatenation and closure of Δ-sums.

Properties (2) and (3) are not valid for regular expressions even when the alphabet is restricted to one letter.

4. Algorithms for the manipulation of Δ-sums

In this section we consider the computer manipulation of Δ-sums. We begin with some simple results that show that the theory of Δ-sums is essentially decidable. Then algorithms for the sum, concatenation, closure and simplification of Δ-sums will be described and their complexity will be analyzed briefly.

We begin by discussing the *representation* of linear numbers and Δ-sums.

(1) *For linear numbers* we use the notation $a + bk$, or just a if $b = 0$.

(2) A *Δ-sum* can always be represented by a sum of linear numbers if we use the distributive law and make the corresponding "concatenations" using Algorithm 1. The sum can always be expressed as

$$F \oplus P,$$

where F is a finite sum of integers (corresponding to linear numbers $a + 0k$) and P is finite sum of numbers $a + bk$ with $b \geqslant 1$. The pair (F, P) is called an *expanded representation* of a Δ-sum. In the sequel both F and P will be treated as *sets* so that instead of saying, for instance, "a belongs to $\mathscr{S}(F)$" we just say "a belongs to F".

(3) An expanded representation (F, P) is called *canonical* if the following properties hold:

(a) All linear numbers in P have the same period ($2 \oplus 5 + 2k \oplus 9 + 4k$ is not canonical).

(b) The sets denoted by F and P are disjoint ($2 \oplus 9 \oplus 5 + 2k$ is not canonical).

(c) The size of F is as small as possible ($2 \oplus 5 \oplus 7 + 2k$ is not canonical).

Clearly, a canonical representation is *always possible and unique.* The three Δ-sums above denote the same set; the equivalent canonical expression is $2 \oplus 5 + 2k$.

(4) Nonexpanded representations will also be used. In fact, any syntactically correct expression involving sums, concatenations and closures may denote a Δ-sum;

this may be much more compact than an expanded representation (see for instance the example in Section 4.4.1).

In this paper we do not explain how these representations can efficiently be implemented in a programming language. In fact, our algorithms will only be described in an abstract way (in a language similar to C or Pascal), using high-level constructs like “sets of integers”.

4.1. Some decision problems

In this subsection we show that it is decidable whether two Δ-sums are equal, i.e. whether they denote the same set.

Theorem 4.1. *It is decidable whether*

(1) *a given integer belongs to a Δ-sum,*
(2) *a finite set of integers belongs to a Δ-sum,*
(3) *a linear number belongs to a Δ-sum,*
(4) *a Δ-sum is included in another Δ-sum,*
(5) *two Δ-sums are equal.*

Proof. Parts (1) and (2) are straightforward if we represent the Δ-sum as a sum of linear numbers. We consider only questions (3) and (4).

To prove (3), denote the linear number and the Δ-sum by $a+bk$ and (F,P), respectively, and suppose that $b>0$. Note that, as the Δ-sum represents an ultimately periodic set with values of n_0 and p (period) that can be computed, we have only to check that
- every integer $a+bi$ less than n_0 belongs to F,
- every integer $a+bi$ satisfying

$$n_0 \leqslant a+bi < n_0+p$$

belongs to a set denoted by some linear number in P.

Let (F_1,P_1) and (F_2,P_2) be the two *canonical* representations of the Δ-sums mentioned in (4). They denote the same set if and only if $F_1=F_2$ and $P_1=P_2$ (recall that F_1, F_2, P_1 and P_2 are sets). □

4.2. Algorithms for sum, concatenation and closure

The sum of two Δ-sums (F_1,P_1) and (F_2,P_2) is just the pair

$$(F_1 \cup F_2, P_1 \cup P_2).$$

To concatenate EF, first expand E and F in a sum of linear numbers, then use the distributivity laws to get a sum of concatenations of two linear numbers. Finally, use the algorithm described in Fig. 1 to compute those concatenations. The computation of F can easily be made finite and is not described here.

```
$\Delta$-sum $n\_concat((a+bk),(c+dk))$
% Concatenates 2 linear numbers, returns $\Delta$-sum $(F,P)$.
% Based on Theorem 1
integer $g$, $n$;
finite integer set $F$;
    if $b=d=0$ then return $(\{b+d\},\{\})$;
    $g=\gcd(b,d)$;
    $n=a+c+((b-g)(d-g))/g$;
    $F=\{x : x=a+c+bi+dj, i\geq 0, j\geq 0, x\leq n-2g\}$;
    return $(F,\{(n+gk)\})$
end
```

Fig. 1. Concatenation of two linear numbers.

```
$\Delta$-sum $closure1(a+bk)$
% Returns the closure of $a+bk$.
integer $d$;
    if $a=b=0$ then return $(\{0\},\{\})$;
    if $b=0$ then return $(\{\},\{0+ak\})$;
    $n=b/\gcd(a,b)$;
    return $(\{0\},\{a+bk, 2a+bk,\cdots,na+bk\})$;
end
```

Fig. 2. Closure of a linear number.

```
$\Delta$-sum $closure(x_1\oplus\cdots\oplus x_n)$
% Returns the closure of the $\Delta$-sum.
finite integer set $L$;
linear number set $R$;
    $L=\emptyset$;
    $R=\emptyset$;
    compute $x_1^*$, ..., $x_n^*$ (using $closure1$);
    for each subset $\{y_1,\cdots,y_m\}$ of $\{x_1,\cdots,x_n\}$ do
        $(L_s,R_s)= y_1^* y_2^*\cdots y_m^*$
        $L=L\cup L_s$;
        $R=R\cup R_s$;
    return$(L,R)$;
end
```

Fig. 3. Closure of a Δ-sum.

Function *closure*1 in Fig. 2 describes the computation of the Kleene closure of a linear number (see Theorem 3.6) while function *closure* (Fig. 3) returns the closure of an arbitrary Δ-sum expressed as a sum of linear numbers. An algorithm for the closure of a Δ-sum can also be based directly on Lemmas 3.4 and 3.5.

4.3. *Simplifying Δ-sums*

When dealing with practical algorithms, it is important that Δ-sum expressions are simplified frequently; otherwise their size may become very large.

Consider an expanded representation (F, P) of a Δ-sum. We describe some simple simplifications which might be applicable to (F, P).

(1) Remove a from F if $a \in P$ or remove repeated occurrences of a from F, e.g.

$$2 \oplus 5 \oplus 6 \oplus 5 + 2k = 2 \oplus 6 \oplus 5 + 2k.$$

(2) Remove $a + bk$ from P if all but a finite number of the integers $a + bi$ is in P; the size of F may increase, e.g.

$$2 \oplus 7 + 2k \oplus 3 + 4k = 2 \oplus 3 \oplus 7 + 2k.$$

(3) "Compact" P by replacing two or more linear numbers by a new one. New integers may have to be inserted in F, e.g.

$$5 + 3k \oplus 6 + 3k \oplus 7 + 3k = 5 + k.$$

In the following example, the three kinds of simplification are used. Consider the Δ-sum

$$S = (2 \oplus 3 + 8k \oplus 2 + 9k)(1 + 6k) \oplus 19 + 2k.$$

We use the distributive law and compute the concatenations (using the algorithm in Fig. 1)

$$(3 + 8k)(1 + 6k) = 4 \oplus 10 \oplus 12 \oplus 16 + 2k,$$

and

$$(2 + 9k)(1 + 6k) = 3 \oplus 9 + 3k.$$

So we have

$$\begin{aligned} S &= 3 + 6k \oplus 4 \oplus 10 \oplus 12 \oplus 16 + 2k \oplus 3 \oplus 9 + 3k \oplus 19 + 2k \\ &= 4 \oplus 10 \oplus 3 + 6k \oplus 9 + 3k \oplus 16 + 2k \oplus 19 + 2k \\ &= 3 \oplus 4 \oplus 10 \oplus 9 + 3k \oplus 16 + 2k \oplus 19 + 2k \\ &= 3 \oplus 4 \oplus 10 \oplus 16 \oplus 18 \oplus 9 + 3k \oplus 19 + k \\ &= 3 \oplus 4 \oplus 10 \oplus 16 \oplus 9 + 3k \oplus 18 + k \\ &= 3 \oplus 4 \oplus 9 \oplus 10 \oplus 12 \oplus 15 \oplus 16 \oplus 18 + k. \end{aligned}$$

The final expression has only one linear number with nonnull k coefficient. The following simplification steps were used:

- remove 12, which is included in $9+3k$, and 3, which is included in $3+6k$,
- remove $3+6k$, which, except for 3, is included in $9+3k$,
- compact $16+2k$ and $19+2k$; we get $19+k$ and the integers 16 and 18,
- $18 \oplus 19+k=18+k$,
- except for $\{9,12,15\}$, $9+3k$ is a subset of $18+k$.

4.4. Complexity

When dealing with the complexity of Δ-sum algorithms, different kinds of questions should be considered.

(1) How to represent Δ-sums during the computations? Following are the two possibilities.

(a) as a pair (F,P) where F is a finite set of integers and P a finite set of linear numbers (expanded representation), Hash tables are appropriate for the implementation of F and P;

(b) as a path (possibly using an implicit representation [2, pp. 418–426]) length expression.

(2) What is the complexity of basic operations – sum, concatenation and the Kleene closure? The answer to this question depends on the representation method.

(3) How does the complexity of Δ-sum algorithms to solve problems about distances compare with similar algorithms using regular expression?

(4) How to analyze the complexity of matrix algorithms [11, 14, 15] involving Δ-sums?

Some of these problems are difficult. In particular, question (1) must be answered (i.e. a concrete representation method must be selected) before the complexity of the algorithms can be analyzed. In this paper we concentrate on high-level algorithmic ideas without going down to the program level, so that the actual representation of Δ-sums is not chosen and detailed complexity analysis is not done.

We only make some remarks about the size of a Δ-sum when expanded as a sum of linear numbers and present a simple classification of their complexity.

4.4.1. On the size of Δ-sums

The size of an expanded and simplified Δ-sum is usually much smaller (and never greater) than the size of a corresponding regular expression. For example, to the regular expression over $\{a,b\}$

$$(a+(abab)^*)^*$$

corresponds the simpler Δ-sum

$$(1 \oplus (4k)^*)^*=(1 \oplus 4k)^*=k,$$

which expresses that all nonnegative lengths of paths are possible.

The compactness of the Δ-sum representation is mainly due to the commutativity of concatenation which has a number of simplifying effects. For instance, any Δ-sum E can be written in such a way that *all noninteger linear numbers have the same period.* This is a direct consequence of the fact that every Δ-sum denotes an ultimately periodic set (see the proof of Theorem 3.1). The common period is the least common multiple of all the periods in an expansion of E. The number of noninteger linear numbers, i.e. the size of P does not exceed this common period.

Normally, the size of F is also small. Consider for instance the concatenation

$$(a'+ak)(b'+bk).$$

Using Theorem 2.6 (see the definitions of n and d) this may be written as (F, P), where

$$P=\{(a'+b'+n)+dk\},$$

$$|F|=\frac{n}{2d}.$$

The set F is often sparse, in the sense that the quotient

$$\frac{|F|}{a'+b'+n}=\frac{n}{2d(a'+b'+n)}$$

is much less than 1 if either d is much greater than 1 or $a'+b'$ is much greater than n. Similar remarks apply to the closure of a Δ-sum.

However, in some cases the size of the expanded Δ-sum may be exponential on the size of the initial expression as the following example, involving only integer linear numbers, shows (it corresponds to the lengths of paths from a to b in Fig. 8)

$$(2^n \oplus 0)(2^{n-1} \oplus 0)\cdots(2^0 \oplus 0).$$

The length of this expression is polynomial in n; more specifically, if we recall that 2^n is a shorthand for $\overbrace{22\cdots 2}^{n}$, we see that it is $O(n^2)$. However, both the corresponding set

$$\{0,1,2,3,\ldots,2^{n+1}-1\}$$

and the expanded Δ-sum

$$0\oplus 1\oplus 2\oplus 3\oplus\cdots\oplus 2^{n+1}-1$$

have exponential size.

4.4.2. *Complexity of Δ-sums: a simple classification*

Consider a general path length expression E. As a step for classifying the complexity of Δ-sums, we define a function *cpx* in Fig. 4 such that $cpx(E)$ is

- -1 if $\mathscr{S}(E)=\emptyset$,
- 0 if $\mathscr{S}(E)=\{0\}$,
- 1 if $\mathscr{S}(E)$ is finite and contains some nonzero integer,

```
integer $cpx(E)$
% Returns an integer denoting the complexity of
% $\mathcal{S}(E)$ as explained in the text.
    % Simple cases:
    if $E = \emptyset$ then return $-1$;
    if $E = 0$ then return $0$;
    if $E = a$ and $a > 0$ then return $1$;
    if $E = a + bk$ and $b > 0$ then return $2$;
    % Sum:
    if $E = F \oplus G$ then return $max(cpx(F), cpx(G))$;
    % Concatenation:
    if $E = FG$ then
        if $cpx(F) = -1$ or $cpx(G) = -1$ then return $-1$;
        if $cpx(F) = 0$ then return $cpx(G)$;
        if $cpx(G) = 0$ then return $cpx(F)$;
        if $cpx(F) = 1$ and $cpx(G) = 1$ then return $1$;
        return $2$;
    % Closure:
    if $E = F^*$
        if $cpx(F) \leq 0$ then return $0$;
        return $2$;
end
```

Fig. 4. The function $cpx(E)$.

• 2 if $\mathcal{S}(E)$ is infinite, i.e. E contains some noninteger linear number.

As the expression to be classified is never "expanded", the algorithm is efficient, being in fact $O(|E|)$. The reader may find it instructive to compute $cpx(E)$ where

$$E = (2 \oplus 4 + 6k)(\emptyset)((4 + 6k \oplus 8 + 4k)(4 + 6k \oplus 8 + 4k))^* \oplus (\emptyset)^*.$$

5. An application: all lengths of paths

As an application of the Δ-sum concept, we consider several problems related to the length of paths in directed graphs. Let $G = (V, E, f)$ be a directed graph where $f: E \to \Sigma$ is a function labeling the edges of the graph with letters of the alphabet Σ. It is well known [15] that the set of all paths between two vertices can be represented by a regular expression. We now prove that the set of all *lengths of paths* (which can be infinite if the graph has cycles) may always be denoted by a Δ-sum which, when expressed in a convenient way (for instance, as a sum of linear numbers), may provide straightforward answers to a number of questions related to the length of the paths such as "what is the shortest length?", "is a given length possible?" or "are there arbitrarily large lengths?".

Theorem 5.1. *Let $G=(V, E)$ be a directed graph and let a and b be two vertices of V. There is a Δ-sum x denoting all distances from a to b, i.e. such that an integer n belongs to $\mathcal{S}(x)$ if and only if there is a path from a to b having length n.*

Proof. A method to obtain a length path expression $r(x)$ for x is as follows:

(1) label each edge of G with the same symbol a;
(2) get the regular expression x that represents all paths from a to b;
(3) make the following replacements in x, resulting in the expression $r(x)$:
  (a) a^i by i; this includes the replacement of λ (the empty word) by 0,
  (b) $y+z$ by $r(y) \oplus r(z)$,
  (c) yz by $r(y)r(z)$,
  (d) $(x)^*$ by $(r(x))^*$.

It is not difficult to see that $r(x)$ denotes the set of all lengths of paths from vertex a to b. □

The method used in this proof shows that the set of *lengths of paths* between two nodes is isomorphic to the set of *paths* between those nodes *if all edges are labeled with the same symbol*. This restriction is algebraically important: it makes concatenation commutative. A similar situation occurs if we restrict the terminal alphabet of a context-free grammar to one letter; the resulting grammar is regular [6]. The following sets are isomorphic:

- the set of Δ-sums,
- the set of context free languages over an alphabet with one letter,
- the set of regular languages over an alphabet with one letter.

Two observations should be made at this point. The first one concerns the method described in the previous proof. It is not necessary to find the regular expression denoting the paths from A to B and convert it to a path expression; the computation with the algebra of Δ-sums is substantially more efficient. To the length of each edge we may either assign 1 or the nonnegative integer specified by some function

$$f: E \to \mathbf{Z}_0^+.$$

The second observation is that we do not need to consider only the paths between two fixed vertices. Well-known matricial methods [7, 14] which are used in a number of different situations – such as the Floyd shortest path algorithm [5], Warshall's transitive closure computation [16] and Kleene's computation of a regular expression equivalent to an automaton – can of course be applied to distance problems.

5.1. Examples

We now describe a simple example that illustrates how the solution of several problems can be based on the computation of a Δ-sum describing all distances between two vertices of a graph.

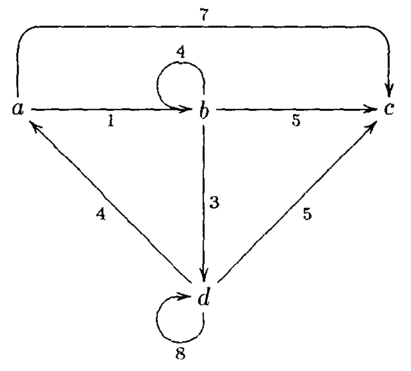

Fig. 5. A graph.

Consider the directed graph represented in Fig. 5. The following expression denotes the set of all distances from vertex a to c. It can be obtained by inspection of the graph

$$E(a,c)=(1\cdot 4^*\cdot 3\cdot 8^*\cdot 4)^*\cdot(7\oplus 1\cdot 4^*\cdot 5\oplus 1\cdot 4^*\cdot 3\cdot 8^*\cdot 5).$$

For more complex graphs an algorithm similar to Kleene's conversion of an automaton to a regular expression may be used. The Δ-sum so obtained is then converted to a sum of linear numbers and simplified.

Using the computation rules described previously, it is possible to express this Δ-sum as a sum of linear numbers (in fact these expressions were obtained by a computer program)

$$E(a,c)=7\oplus 6+4k\oplus 9+4k\oplus 15+4k.$$

This sum provides almost immediate answers to several questions related to the lengths of paths from a to c. This is illustrated in Table 2.

Table 2

Question	Method	Answer
Shortest path	Smallest in $\{7\}\cup\{6,9,15\}$	6
Longest path	Infinite if $P\neq\emptyset$	∞
Is there a length 71?	Search in F and P	Yes, $15+4\times 14$
Are there arbitrarily large prime lengths?	True if $\mathcal{S}(P)$ contains almost all odd integers	Yes
What multiples of 4 are path lengths?	Inspect F and P	None

Note that the method used to answer the fourth question corresponds to a condition which is only sufficient. We conjecture that a condition which is also necessary is: P contains some $a+bk$ with $b\geqslant 1$ and $\gcd(a,b)=1$.

As another example, consider the graph in Fig. 7. The expressions denoting the lengths of paths from a to b and to c are

$$P(a,b)=1 \oplus 3+3k \oplus 3+5k,$$

$$P(a,c)=(1 \oplus 3+3k \oplus 5+5k)(1+5k).$$

The last expression can be expanded in a sum which has only one noninteger linear number,

$$P(a,c)=2 \oplus 4 \oplus 7 \oplus 9 \oplus 10 \oplus 12+k.$$

The integers less than 30 belonging to $\mathscr{S}(P(a,b))$ are represented in the diagram below

0					5					10					15					20					25				
·	×	·	×	·	·	×	·	×	×	·	·	×	×	·	×	·	·	×	·	·	×	·	×	×	·	·	×	×	·

Fig. 6a.

This set is ultimately periodic with period 15. The diagram for $P(a,c)$ is

0					5					10					15					20					25				
·	·	×	·	×	·	·	×	·	×	×	·	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×

Fig. 6b.

5.2. *Computing all distances: an efficient method*

Consider a directed graph $G=(V,E,f)$, where f is a function which assigns to each edge a nonnegative number. Although a Δ-sum expression denoting all distances between two particular vertices is often relatively short (and typically much shorter than the corresponding regular expression), it may also be very large if there are many paths between those two vertices *and* if f takes very large values. As an example, consider the graph in Fig. 8; there are 2^{n+1} paths between a and b and the corresponding Δ-sum has 2^{n+1} terms:

$$0 \oplus 1 \oplus 2 \oplus 3 \oplus \cdots \oplus 2^{n+1}-1.$$

This example shows that the size of the Δ-sum can be exponential in the number of nodes (this graph has $2n+2$ nodes).

It is not difficult to design an algorithm *working in polynomial time* that, from a given Δ-sum E, outputs a graph G with two selected vertices a and b such that the set of distances between those two vertices is denoted by E; of course the size of G is also

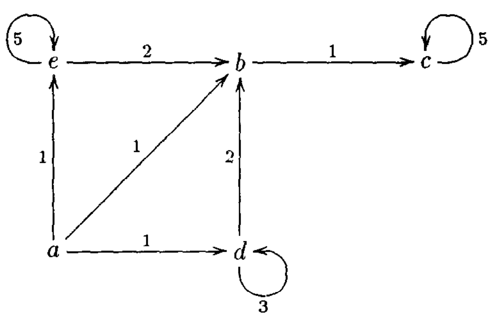

Fig. 7. Another graph.

polynomial in $|E|$. Then the observations made in Section 4.4 about the exponential or polynomial size of Δ-sums can also be made about graphs.

In the following, we use the decomposition of the graph into strongly connected components in order to design an algorithm for the computation of a Δ-sum denoting the set of all distances between two vertices. In many practical cases this results in a relatively short expression; in particular this will be the case *if* either the function f does not take large values *or if* the number of strongly connected components is small.

The set of vertices V of a directed graph $G=(V,E)$ can be partitioned (see for example [1]) into equivalence classes $V_1, V_2, \ldots, V_r$ such that two vertices a and b are equivalent if and only if there is a path from a to b and a path from b to a. Each $G_i=(V_i, E_i)$ where E_i is the set of edges connecting two vertices in V_i is called a *strongly connected component* (SCC). The edges of E that do not belong to any E_i connect two distinct strongly connected component. Any cycle of G is contained in some strongly connected component.

In [1] an algorithm to find the partition of a graph into strongly connected components (having execution time $O(\max(n,e))$ where n and e are, the number of vertices and the number of edges of the graph, respectively) is described.

In order to describe our algorithm more clearly, we define two graphs associated to every directed graph G. The SCC-*graph* G_s is the graph whose vertices are the strongly connected components of G and which has an edge between components A and B if and only if for one or more edges (a,b) of E we have $a\in A$ and $b\in B$. Clearly G_s is acyclic. The SCC-*multigraph* is the multigraph whose vertices are the strongly connected components of G and whose edges between components A and B are the edges (a,b) of E such that $a\in A$ and $b\in B$.

5.2.1. *Inside a strongly connected component*

We now prove that the Δ-sum expression denoting the set of distances between two vertices in the same SCC needs only to have one noninteger linear number.

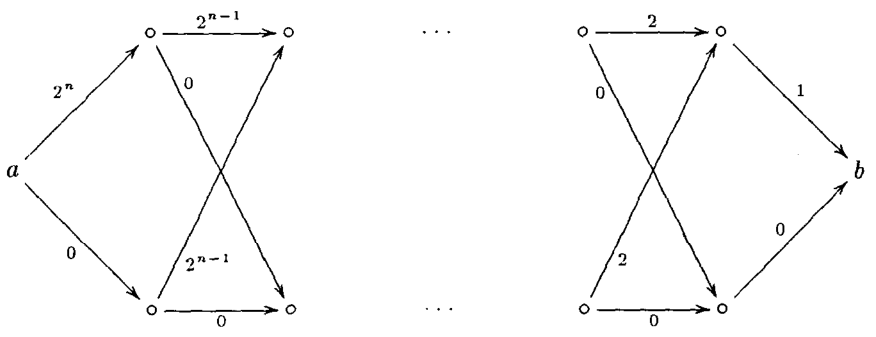

Fig. 8. A graph with 2^{n+1} paths between a and b.

Lemma 5.2. *Let a, b, c and d be vertices belonging to the same SCC of a directed graph and let the distances between a and b and between c and d be denoted by Δ-sums S_{ab} and S_{cd}, respectively. Then if $e+fk \in S_{ab}$ there is an integer m such that $(e+m)+fk \in S_{cd}$.*

Proof. We can go from c to d following a path $c \to a \to b \to d$. Let m be the sum of the lengths in $c \to a$ and in $b \to d$. Then as $e+fk$ denotes (some of the) distances from a to b, $(e+m)+fk$ denotes (some of the) distances from c to d. $\square$

Essentially, this lemma says that the period of a noninteger linear number of the Δ-sum that denotes the distances between two vertices belonging to the same SCC does not depend on those vertices. We now see that, inside the same SCC, at most one noninteger linear number is needed.

Theorem 5.3. *Let a and b be the vertices belonging to the same SCC of a directed graph. There is a Δ-sum denoting the set of distances between a and b having the form*

$$e_1 \oplus \cdots \oplus e_m \oplus f + pk,$$

where the linear number $f+pk$ does not depend on a or b. If $a=b$ then $f=0$.

Proof. Consider first the case $a=b$. The set of all distances between a and a is denoted by some Δ-sum x. Then x^* also denotes that set and the statement follows from Theorem 3.6. Note that x does not depend on a.

If a and b are distinct vertices, every path from a to b can be described by a path from a to a followed by a simple path from a to b followed by a path from b to b. Then, its total length is described by $F \oplus x^*$ where x^* is the expression mentioned above and F corresponds to the finite number of simple paths. $\square$

5.2.2. The general case

We now describe a faster method to compute a Δ-sum representing the set of distances between two vertices in a graph and analyze its complexity. In particular we study the number of periods (the k-coefficients of noninteger linear numbers) needed in the Δ-sum.

Consider a path in the SCC-graph between the strongly connected components containing the two vertices a and b

$$C_1, C_2, \ldots, C_m,$$

where $a \in C_1$ and $b \in C_m$. In general this path corresponds to several paths in the original graph because there may be several edges connecting the same pair of strongly connected components and because there are (in general) infinite paths inside the same SCC. A Δ-sum denoting the set of distances for a particular choice of the edges connecting distinct strongly connected components has the form (we are using Theorem 5.3):

$$(F^1_{ae_1} \oplus f^1_{ae_1} + p_1 k) d_{12} (F^2_{i_2 e_2} \oplus f^2_{i_2 e_2} + p_2 k) d_{23} \cdots$$
$$d_{(m-1)m} (F^m_{i_m b} \oplus f^m_{i_m b} + p_m k). \tag{3}$$

The factors in this concatenation denote, respectively, the distances between a and the exit vertex e_1 of the first component, the length of the edge that connects the first to the second component, ..., and the distances between the entrance vertex i_m of the last component and b. Superscripts are used to identify the strongly connected components. Using the proof of Theorem 3.3, we see that every expression for the same path in the SCC-graph can be written using the period

$$p = \gcd(p_1, p_2, \ldots, p_m).$$

It is now clear how we can compute the Δ-sum denoting the distances between a and b. The result is a sum of terms like (3) above each corresponding to a particular selection of edges connecting components for each path between a and b in the SCC-graph. Although the Δ-sum so obtained may be big, practice shows that in many cases it is much shorter (and the corresponding computation is much faster) than the one obtained by more direct algorithms not based on graph decomposition.

As an example, consider again the distances between a and c in the graph of Fig. 5. The components are $\{a, b, d\}$ and $\{c\}$. The corresponding expression is (before any simplification)

$$E(a,c) = (1 \cdot 4^* \cdot 3 \cdot 8^* \cdot 4)^* \cdot 7 \oplus (1 \cdot 4^*) \cdot (4^* \cdot 3 \cdot 8^* \cdot 4 \cdot 1)^* \cdot 5 \oplus$$
$$(1 \cdot 4^* \cdot 3 \cdot 8^*) \cdot (8^* \cdot 4 \cdot 1 \cdot 4^* \cdot 3)^* \cdot 5.$$

We can be more precise about the length of the Δ-sum denoting the distances between a and b. The proof of the following theorem is omitted.

Theorem 5.4. *The length of the Δ-sum denoting the distances between two vertices of a graph can be exponential in the size of the graph (the maximum of the number of vertices and edges) only if one of the following occurs.*

(1) *The number of paths in the SCC-graph between the components of a and b is exponential.*

(2) *The number of paths in the SCC-multigraph that correspond to some path in the SCC-graph between the components of a and b is exponential.*

(3) *The length of at least one edge in some SCC belonging to some path of the SCC-graph between the components of a and b is exponential.*

6. Conclusions

For problems related to the length of the words of context-free or regular languages (or equivalently, to the words of regular languages when the alphabet is restricted to one letter), Δ-sums seem to be more compact and efficient than standard representations such as finite automata or regular expressions. The reasons for this improvement may be related to the algebraic properties of Δ-sums (where for instance, closure and concatenation operators can always be completely eliminated) and to the arithmetic character of their manipulation.

In this paper, we have analyzed the properties of Δ-sums and presented algorithms for the basic operations. Further work is needed in order to obtain more accurate complexity results. In particular, the relationship between known algorithms for the Frobenius problem and algorithm for the concatenation of Δ-sums should be studied.

The application to distance problems in directed graphs (see Section 5) is another promising area for further research. Two specific problems are: (i) the design of efficient algorithms for the computation of an expression denoting the distances between two vertices of the same strongly connected component and (ii) a study of the application of Δ-sums for *dynamic graphs* (graphs whose vertices and edges can be added and removed).

References

[1] A. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Reading, MA, 1974).
[2] A. Aho and J.D. Ullman, *Principles of Compiler Design* (Addison-Wesley, Reading, MA, 1974).
[3] J. Conway, *Regular Algebra and Finite Machines* (Chapman and Hall, London, 1971).
[4] S. Eilenberg, *Automata, Languages and Machines, Vol. A* (Academic Press, New York, 1974).
[5] R. Floyd, Algorithm 97: shortest path, *Comm. ACM* **5**(6) (1962) 345.
[6] S. Ginsburg and H.G. Rice, Two families of languages related to Algol, *J. ACM* **9**(3) (1962) 350–371.
[7] M. Gondran, Algèbre linéaire et cheminement dans un graphe, *Rev. Française d'Automatique, Informatique et Recherche Opérationnelle* **9**(1) (1975) 77–99.
[8] K. Hashigushi, Algorithms for determining relative star height, *Inform. and Control* **78** (1987) 124–169.
[9] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, Reading, MA, 1979).
[10] R. Kannan, Solution of the Frobenius problem, Tech. Report CMU-CS-89-204, Carnegie-Mellon University, Pittsburgh, PA, 1989.
[11] D.J. Lehmann, Algebraic structures for transitive closure, *Theoret. Comput. Sci.* **4** (1977) 59–76.

[12] A. Schrijver, *Theory of Linear and Integer Programming* (Wiley, New York, 1989).
[13] J.J. Sylvester, Problem 7382, *Math. from the Ed. Times with Solutions* **41** (1884) 21.
[14] R.E. Tarjan, Fast algorithms for solving path problems, *J. ACM* **28**(3) (1981) 594–614.
[15] R.E. Tarjan, A unified approach to path problems, *J. ACM* **28**(3) (1981) 577–593.
[16] S. Warshall, A theorem on boolean matrices, *J. ACM* **9**(1) (1962) 11–12.