Annales Univ. Sci. Budapest., Sect. Comp. 28 (2007) 141-156

DECISION AND CLASSIFICATION ALGORITHMS FOR GENERALIZED NUMBER SYSTEMS

P. Burcsi, A. Kovács and Zs. Papp-Varga

(Budapest, Hungary)

Dedicated to Professor Imre Kátai on the occasion of his 70th birthday

Abstract. We present algorithms for the decision and classification of generalized number systems. In the first part of the article, an algorithm using an enclosing parallelepiped for the set of fractions is considered. We mainly focus on minimizing the number of lattice points in the parallelepiped by choosing the basis optimally. In the second part we generalize Brunotte's canonical number system decision algorithm for generalized systems and we extend the results for the classification problem. Finally, we compare the algorithms by their performances in practice.

1. Introduction

Let $M \in GL(n, \mathbb{Z})$, called the base, and $0 \in D \subseteq \mathbb{Z}^n$ a finite set, called the digit set. The pair (M, D) is called a *generalized number system* (GNS) if every $x \in \mathbb{Z}^n$ has a unique finite representation in the form $x = \sum_{i=0}^{k} M^i d_{j_i}$ with $d_{j_i} \in D$. The following necessary conditions must hold in order to form a GNS ([14]).

- M must be expansive, i.e. $|\lambda| > 1$ for every eigenvalue λ.
- D is a complete residue system mod M, i.e. $|D| = |\det M|$ and, for distinct $d, d' \in D$, we have $d - d' \notin M\mathbb{Z}^n$,

- $\det(M - I) \neq \pm 1$, where I denotes the identity matrix.

All of these properties can easily be checked for a given M and D. For details and other known results cf. the surveys [1] or [8]. Throughout the paper we will assume that the pair (M, D) meets these conditions.

A reformulation of the unique representation property is possible using the function $\phi : \mathbb{Z}^n \to \mathbb{Z}^n$, $x \mapsto M^{-1}(x - d)$ for the unique $d \in D$ satisfying $x \equiv d \pmod{M}$. Since M^{-1} is contractive and D is finite, there exists a norm on $\mathbb{Z}^n$ and a constant C such that the orbit of every $x \in \mathbb{Z}^n$ eventually enters the finite set $S = \{x \in \mathbb{Z}^k \mid \|x\| < C\}$ for the repeated application of ϕ. This means that the sequence $x, \phi(x), \phi^2(x), \ldots$ is eventually periodic for all $x \in \mathbb{Z}^n$.

Clearly, (M, D) is a GNS iff for every $x \in \mathbb{Z}^n$ the orbit of x eventually reaches 0. A point x is called periodic if $\phi^k(x) = x$ for some $k > 0$. The orbit of a periodic point is called a *cycle*. Hence (M, D) is a GNS iff the only periodic point is 0, or equivalently, the only cycle is $0 \to 0$. In this paper we investigate the following two problems. The decision problem for (M, D) asks if they form a GNS or not. The more general classification problem means finding all cycles.

No general fast algorithm is known for these problems. Special cases have been treated with success. A digit set is called *canonical* if

$$D = \{(i, 0, 0, \ldots, 0)^T \mid 0 \leq i < |\det M|\}.$$

A GNS is a canonical number system (CNS), if M is the companion matrix of a monic, integer polynomial, and the digit set is canonical. Important results are known for the canonical case. Quadratic CNS-polynomials were classified by I. Kátai and B. Kovács [9, 10] and independently by W.J. Gilbert [7]. Cubic and quartic CNS-polynomials were investigated by S. Akiyama, H. Brunotte, A. Pethő [2], H. Brunotte [5], and K. Scheicher, J.M. Thuswaldner [17].

The following was discovered by B. Kovács for irreducible polynomials [12], and slightly generalized by A. Pethő [16]. Let $c(x) = c_0 + c_1 x + \ldots + c_{n-1}x^{n-1} + x^n \in \mathbb{Z}^n$. If $c_0 \geq 2$, $c_{n-1} \leq \cdots \leq c_1 \leq c_0$ and $c(x)$ is not divisible by a cyclotomic polynomial, then $c(x)$ is a CNS-polynomial. S. Akiyama, A. Pethő [3], S. Akiyama, H. Rao [4] and K. Scheicher, J.M. Thuswaldner [17] showed characterization results under the "dominant" condition $c_0 > |c_1| + \cdots + |c_{n-1}|$.

In this paper we give two algorithms for the decision/classification problem in the general case. The original version of the first method, using a covering of the set of fractions, was first applied by the second author [13]. It gives lower and upper bounds on the coordinates of periodic points. The method is combined here with a basis transformation using a randomized algorithm in order to improve the bounds.

The second method is a generalization of Brunotte's CNS decision algorithm [5]. We show how one can modify it so that it handles arbitrary digit sets and

the classification problem. We also give detailed pseudocode implementation of the algorithm. In order to distinguish between the two methods, we refer to the first one as algorithm α and the second one as algorithm β.

Finally, we compare the methods by their practical performance.

2. Covering the set of fractions

The method is based on a covering construction that appeared in [13]. We will refer to it as algorithm α. The set of fractions is defined as

$$H = \left\{ \sum_{i=1}^{\infty} M^{-i} d_{j_i} \mid d_{j_i} \in D \right\} .$$

Let x be a periodic point, with period length k. Then $\phi^k(x) = \boldsymbol{x}$, thus, from the definition of ϕ, $M^{-k}x = x + \sum_{i=1}^{k} M^{-i} d_{j_i}$ for some j_i. If we denote the $n \times n$ identity matrix by I, we have

$$\begin{aligned} -x &= \left(I - M^{-k}\right)^{-1} \left(\sum_{i=1}^{k} M^{-i} d_{j_i} \right) = \\ &= \left(I + M^{-k} + M^{-2k} + \cdots\right) \left(\sum_{i=1}^{k} M^{-i} d_{j_i} \right) = \\ &= \sum_{l=0}^{\infty} \sum_{i=1}^{k} M^{-lk-i} d_{j_i} \in H , \end{aligned}$$

where the absolute convergence and the validity of the calculation follows from the expansivity of M.

If we find lower and upper bounds l_m, u_m $(m = 1, \ldots, n)$, such that $-H \subseteq$ $\subseteq T = \{(x_1, x_2, \ldots, x_n) \mid l_i \leq x_i \leq u_i\}$, then all periodic points lie in the brick T. We briefly describe a method for finding such bounds.

Let $\|\cdot\|$ denote the maximum norm in $\mathbb{R}^n$. Choose a positive real number $c < 1$ and $k \in \mathbb{N}$ with $\left\|M^{-k}\right\| = \delta \leq c$.

Let $p_m : \mathbb{R}^n \to \mathbb{R}$ denote the projection in the mth coordinate for $m = 1, \ldots, n$. Let

$$\begin{aligned} a_{m,j} &= \min_{d \in D} p_m(M^{-j} d) \quad \text{and} \\ b_{m,j} &= \max_{d \in D} p_m(M^{-j} d) \end{aligned}$$

for $j = 1, \ldots, k$. Let

$$W = \left\{ (x_1, \ldots, x_n)^T \;\middle|\; -\sum_{j=1}^{k} b_{m,j} \leq x_m \leq -\sum_{j=1}^{k} a_{m,j} \right\}.$$

Clearly, $-\sum_{i=1}^{k} M^{-i} d_i \in W$ for an arbitrary sequence $d_i \in D$. So $-H \subseteq W + M^{-k}W + M^{-2k}W + \cdots$. Now let A be the largest absolute value among the $\sum a_{m,j}$ and $\sum b_{m,j}$. We have $\|x\| \leq A$ for every $x \in W$, and $\left\|M^{-ik}\right\| \leq \delta^i$ for $i \geq 1$. Let $\gamma = A(\delta + \delta^2 + \cdots) = A\frac{\delta}{1-\delta}$. Then

$$l_m = -\sum_{j=1}^{k} b_{m,j} - \gamma \text{ , and}$$

$$u_m = \sum_{j=1}^{k} a_{m,j} + \gamma$$

are appropriate lower and upper bounds. The number of periodic lattice points is bounded by $\mathrm{Vol}(M, D) = \prod_{m=1}^{n} \lfloor u_m - l_m + 1 \rfloor$. Classification can be done by examining the points in the brick determined by these bounds. The running time of the classification is roughly proportional to $\mathrm{Vol}(M, D)$. Choosing δ sufficiently small (0.1 seemed reasonable in our experiments) guarantees that the bounds obtained from the covering brick are almost always sharp. We cannot directly reduce the value of $\mathrm{Vol}(M, D)$. Clearly, a basis transformation (applied to both M and D) does not affect the periodicity of points. Hence we can perform a basis transformation *before* calculating $\mathrm{Vol}(M, D)$. Below, we give an algorithm for choosing a basis that yields significant decrease in the value of $\mathrm{Vol}(M, D)$. If T is a transformation matrix, we will simply denote $\mathrm{Vol}(TMT^{-1}, TD)$ by $\mathrm{Vol}(T)$ (M and D are fixed). For a matrix U, we denote by $incr(U, i, j)$ and $decr(U, i, j)$ the matrix that differs from U only at position (i, j) by $+1$ and -1, respectively. The main algorithm uses the following routine:

Function CHANGE-AT-POSITION(M, D, i, j, T)

```
1  U ← T ;
2  OldVol ← Vol(T) ;
3  while Vol(incr(U, i, j)) < OldVol do
4      U ← incr(U, i, j) ;
5      OldVol ← Vol(U) ;
6  end
7  V ← T ;
8  OldVol ← Vol(T) ;
9  while Vol(decr(V, i, j)) < OldVol do
10     V ← decr(V, i, j) ;
11     OldVol ← Vol(V) ;
12 end
13 return [U, V]
```

This function increases (decreases) $T_{i,j}$ as long as smaller volumes are received. The best matrix is put into U (resp. V).

Algorithm FIND-BASIS-TRANSFORMATION(M, D)

```
   Input: M, D, CandNum, ProbeNum, NoImprLim, TargetVol
   Output: Upper triangular basis transformation matrix T
1  Candidates ← CandNum sized list of identity matrices ;
2  NoImpr ← 0 ;
3  while NoImpr < NoImprLim and Vol(Candidates[1]) > TargetVol do
4      NewCands ← [] ;
5      forall T ∈ Candidates do
6          for k ← 1 to ProbeNum do
7              Choose random position (i, j) above the diagonal ;
8              [U, V] ← CHANGE-AT-POSITION(M, D, i, j, T) ;
9              NewCands ← NewCands + [U, V] ;
10         end
11     end
12     Sort NewCands increasingly by values of Vol(T)
13     if Vol(NewCands[1]) < Vol(Candidates[1]) then
14         NoImpr ← 0 ;
15     else
16         NoImpr ← NoImpr+1 ;
17     Copy first CandNum elements of NewCands into Candidates ;
18 end
19 return Candidates[1]
```

The computation goes along several branches. It keeps track of the best

CandNum transformation matrix candidates and, in each iteration, it tries to improve them *probeNum* times. The computation ends when no improvement occurs for *NoImprLim* consecutive iterations, or when a sufficiently small volume, lower than *TargetVol* is reached.

Below we give a detailed example of the algorithm.

We use the shorthand $((i_1, j_1, x_1), ..., (i_k, j_k, x_k))$ to denote a matrix that differs from the identity matrix at positions (i_h, j_h), holding the values x_h, $(h = 1 \dots k)$.

We input

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & -7 \\ 1 & 0 & 0 & 0 & 6 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

D is canonical, $CandNum = 2$, $ProbeNum = 2$, $NoImprLim = 3$, $TargetVol = 5000$. The initial volume is 17500.

After the first iteration of the while loop (lines 3–18) we have $Candidates = [((3, 4, -1)), ((4, 5, -1))]$, $\mathrm{Vol}(Candidates) = [10500, 10500]$.

This means that the random method found two distinct positions where a change in the matrix results in a smaller volume.

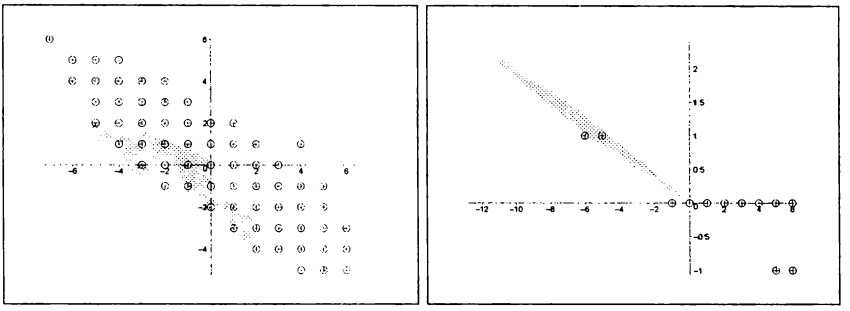
In the next iteration it tries to improve the first element of *Candidates*. No improvement is found for this matrix. For the second one, however, positions $(3, 4)$ and $(1, 4)$ were chosen, both positions are incremented and decremented. Incrementing at position $(3, 4)$ does not improve, but decrementing it to -1 yields volume 6525, incrementing it further gives 10500. Decrementing at position $(1, 4)$ does not improve (12375 for -1), but incrementing it yields the volumes 9000, 7875, 6750, 6000 and 4500, after the next iteration the volume becomes 6750. After sorting, we have $Candidates = [((4, 5, -1), (1, 4, 5)), ((4, 5, -1), (3, 4, -1))]$, $\mathrm{Vol}(Candidates) = [4500, 6525]$.

Since 4500 is smaller than *TargetVol*, we stop. We succeeded in reducing the size of the covering set to 4500 from the original volume 17500. The transformed matrix is

$$\begin{pmatrix} 0 & 0 & 5 & 0 & -7 \\ 1 & 0 & 0 & -5 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix},$$

and the digit set remains canonical. We note that letting the algorithm run a bit longer (through playing with the parameters) gives an even smaller volume, below 1000.

We give two more examples in Figure 1.



(a) (b)

Figure 1: Decreasing the volume of the covering set

Example (a) shows $M = \left(\begin{smallmatrix} 1 & -2 \\ 1 & 3 \end{smallmatrix}\right)$ with $D = \{(0,0), (1,0), (0,1), (4,1), (-7,6)\}$. Changing the basis to $\{(1,0),(1,1)\}$ decreases the volume from 42 to 24. Example (b) shows $M = \left(\begin{smallmatrix} 0 & -7 \\ 1 & 6 \end{smallmatrix}\right)$ with canonical digit set. Replacing basis vector $(0,1)$ with $(-5,1)$ gives volume 4 instead of 64. In both pictures, grey areas represent $-H$. The circles represent the set E in Brunotte's algorithm (cf. function CONSTRUCT-SET-E).

In order to test the algorithm on a large data set, we used algorithms in [6] for the generation of random expansive polynomials. Figure 2 shows the average improvement in orders of magnitude, as the constant term takes values $6, \ldots, 100$, and dimension is changed from 3 to 8.

A possible way to improve the algorithm is to perform the algorithm for the transformed system with a lower triangular transformation matrix as well, and combine the two transformations. We performed it in four different ways: only upper, only lower, upper followed by lower and lower followed by upper triangular transformation matrices. The average reduction in orders of magnitude were $3.218, 0.603, 3.417$ and 3.168, respectively. The smaller figure at lower triangular matrices may be due to the special form of test cases: for their importance only companion matrices were considered. We will continue our experiments with arbitrary matrices to see if this caused the difference. Generating random expansive matrices seems difficult. One can apply an integer basis transformation to the companion matrix of a polynomial, but we know from [15] and [18] that this method generates all expansive matrices only if the class number of the order corresponding to the polynomial is 1.

3. Generalization of Brunotte's algorithm

3.1. The decision algorithm

Brunotte's canonical number system decision algorithm first appeared in [5]. An alternative treatment of the algorithm can be found in [4]. We reformulate and prove Brunotte's results on decision in the case of arbitrary generalized number systems. We refer to this family of algorithms as method β. The statement of the theorem and the proof idea are slightly modified compared to the original version. The generalized algorithm starts with the construction of a set E. We use the function ϕ defined in the introduction.

Function CONSTRUCT-SET-E(M, D)

```
1 E ← D , E' ← ∅ ;
2 while E ≠ E' do
3     E' ← E ;
4     forall e ∈ E and d ∈ D do
5         put φ(e + d) into E ;
6     end
7 end
8 return E
```

Proposition 3.1. *The above algorithm terminates.*

Proof. Let $x \in \mathbb{Z}^n$, and let $d \in D$ be the digit with $\phi(x) = M^{-1}(x - -d)$. Since M^{-1} is contractive, there exists a norm in $\mathbb{R}^n$ with induced norm $\|M^{-1}\| = r < 1$. Let $m = \max_{d,d' \in D} \|d - d'\|$. If $\|x\| \leq mr/(1-r)$, then $\|\phi(x + d')\| = \|M^{-1}(x + d' - d)\| \leq r\|x\| + rm \leq mr^2/(1-r) + mr = mr/(1 - -r)$. Thus, no vector with $\|x\| > mr/(1-r)$ will ever get into E.

Lemma 3.1. *Let us assume that every vector in E has finite expansion. Let $e \in E$ and $x = \sum_{i=0}^{k} M^i d_{j_i}$ an arbitrary vector with finite expansion. Then $e + x$ has finite expansion as well.*

Proof. The proof goes by induction on k. If $k = 0$, then $x \in D$. Thus, $\phi(e + x) \in E$ by the construction of E, and it has finite expansion by the assumption. Then $x + e$ has finite expansion as well.

If $k > 0$, then put $y = \phi(x)$, so that $x = My + d_{j_0}$. By the construction of E, $e + d_{j_0} = Me' + d'$ for some $e' \in E$ and $d' \in D$, giving

$$\phi(e + x) = \phi(e + My + d_{j_0}) = \phi(My + Me' + d') = y + e' .$$

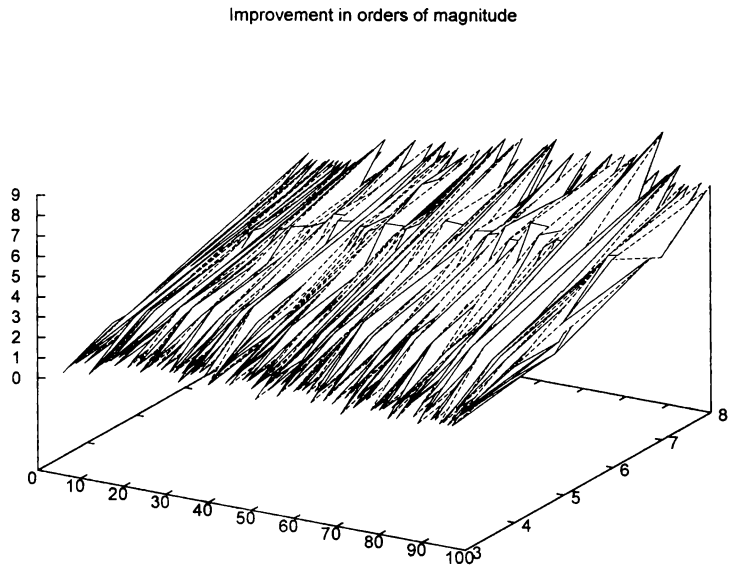


Figure 2: The average improvement in the volume of the covering set after algorithm FIND-BASIS-TRANSFORMATION, expressed in orders of magnitude (base 10).

It suffices to show that $y + e'$ has finite expansion, but that follows from the induction hypothesis.

The above lemma and proof are best understood if we look at E as the set of possible carries that occur when two points with finite expansion are added. We can now give the generalized algorithm for number system decision. Let the set $B = \{(0, \ldots, 0, \pm 1, 0, \ldots, 0)\}$ (i.e. the n basis vectors and their opposites).

Function SIMPLE-DECIDE(M, D)

```
1 E ← CONSTRUCT-SET-E(M, D) ;
2 forall p ∈ B ∪ E do
3     if p has no finite expansion then
4         return false
5
6 end
7 return true
```

Theorem 3.1. *Algorithm* SIMPLE-DECIDE *returns true if and only if* (M, D) *is a number system.*

Proof. It is clear that a vector with no finite expansion yields rejection. On the other hand, if every vector in E has finite expansion, then, by Lemma 3.1, vectors with finite expansion form an additive sub-semigroup $S \leq \mathbb{Z}^n$. Take a set that generates the whole lattice as a semigroup, like B. If this set has finite expansion, then S contains it, giving $S = \mathbb{Z}^n$.

3.2. The classification algorithm

For classification, one needs to find *all* periodic points. Of course, if the decision algorithm accepts (M, D), we know that the origin is the only periodic point. If it rejects, we have a point without finite expansion, which is either a point of the semigroup-basis, or a point in set E. Thus, following the orbit of this point, we also find a non-zero periodic point, a witness.

A first idea for classification is to search the set $E \cup B$ for points with no finite expansion. Unfortunately, there can be periodic points that evade this search. We show this by the following simple example. Take the expansive polynomial $3 - 3x + x^2$ with canonical digit set. Then there are two non-zero periodic points, both of period 1: $(-4, 2)$ and $(-2, 1)$. Only the latter one is in the set $E \cup B$. $5 - 4x + x^2$ is another example.

The idea of the classification algorithm is to increase the set E so that it contains all periodic points. This is done by iterating the function CONSTRUCT-SET-E several times, replacing set D by a larger set including already known periodic points.

We now give the algorithm for classification.

Function SIMPLE-CLASSIFY(M, D)

```
1  D ← D ;
2  finished ← false ;
3  while not finished do
4      E ← CONSTRUCT-SET-E(M, D) ;
5      finished ← true ;
6      forall p ∈ E ∪ B do
7          if p does not run eventually into D then
8              put newly found periodic points into D ;
9              finished ← false ;
10
11     end
12 end
13 return D \ D (the set of non-zero periodic points)
```

Lemma 3.2. *The classification algorithm terminates.*

Proof. Note that $\mathcal{D}$ may only contain periodic points and digits, so CONSTRUCT-SET-E is only called a finite number of times. It remains to show that the execution of this call terminates for any $\mathcal{D}$. This follows from the finiteness of $\mathcal{D}$ in the same way as in proposition 3.1.

Lemma 3.3. *Assume that every vector in $\mathcal{E}$ eventually runs into $\mathcal{D}$. Let $e \in \mathcal{E}$, and $x = \sum_{i=0}^{k-1} M^i d_{j_i} + M^k d$ an arbitrary vector that runs into $\mathcal{D}$ (i.e. $d \in \mathcal{D}$). Then $e + x$ also runs into $\mathcal{D}$ eventually.*

Proof. The proof goes by induction on k. The case $k = 0$ follows from the construction of $\mathcal{E}$. If $k > 0$, then let $y = \phi(x)$. We know that $e + d_{j_0} = Me' + d'$ for some $e' \in \mathcal{E}$ and $d' \in D$, giving

$$\phi(e + x) = \phi(e + My + d_{j_0}) = \phi(My + Me' + d') = y + e' \quad ,$$

so the lemma holds by the induction hypothesis for e' and y.

Theorem 3.2. *Algorithm* SIMPLE-CLASSIFY *is correct.*

Proof. The only thing to show is that on termination $\mathcal{D}$ contains every periodic point. We show that every vector in $\mathbb{Z}^n$ eventually runs into $\mathcal{D}$. This is true for $\mathcal{E}$ by the previous lemma, so the points eventually running into $\mathcal{D}$ form an additive semigroup. Since this semigroup contains B, the proof is complete.

3.3. Implementation

We give directly programmable pseudocode below that optimizes the decision and classification algorithms.

We will use the notation $\phi(X + Y) = \{\phi(x + y) \mid x \in X, y \in Y\}$. Suppose that the loop in lines 4–6 of function CONSTRUCT-SET-E is executed exactly k times. Let us denote the set E before the ith execution of the loop by E_i, let $E_0 = \varnothing$ and $\Delta E_i = E_i \setminus E_{i-1}$. Then we have

$$D = E_1 \subsetneq E_1 \subsetneq E_2 \ldots \subsetneq E_k \,,$$

and

$$\begin{aligned} E_{i+1} &= \phi(E_i + D) = \phi((E_{i-1} \cup \Delta E_i) + D) = \\ &= \phi(E_{i-1} + D) \cup \phi(\Delta E_i + D) = E_i \cup \phi(\Delta E_i + D) \end{aligned}$$

for $i = 0, 1, \ldots, k-1$. This shows that in order to obtain E_{i+1}, the calculation of $\phi(e + D)$ for $e \in \Delta E_i$ is sufficient. This observation suggests the following algorithm.

Algorithm DECIDE(M, D)

```
1  forall b ∈ B do
2      if b has no finite expansion then
3          return false
4
5  end
6  E ← D, oldΔE ← D, ΔE ← ∅ ;
7  while oldΔE ≠ ∅ do
8      forall e ∈ oldΔE, d ∈ D do
9          Orbit ← ∅, p ← φ(e + d) ;
10         while p ∉ Orbit and p ∉ E do
11             Orbit ← Orbit ∪ {p} ;
12             ΔE ← ΔE ∪ {p} ;
13             p ← φ(p) ;
14         end
15         if p ∈ Orbit then
16             return false
17         else
18             E ← E ∪ Orbit ;
19
20     end
21     oldΔE ← ΔE ;
22     ΔE ← ∅ ;
23 end
24 return true
```

When the while loop in lines 7–23 is entered for the ith time, the variables take the following values: $E = E_i$ and $old\Delta E = E_i \setminus E_{i-1}$. This is easily seen by induction, showing that our algorithm is correct.

Consider the classification algorithm. We reformulate the algorithm in a recursive manner as follows. Let $\mathcal{E}_1 = \mathcal{D}_1 = D$, and

$$\begin{aligned} \mathcal{E}_{i+1} &= \phi(\mathcal{E}_i + \mathcal{D}_i)\,, \\ \mathcal{D}_{i+1} &= \{\text{ periodic points in } \mathcal{E}_i \}\,. \end{aligned}$$

Clearly, these sequences are eventually stable.

Let $\Delta\mathcal{E}_i = \mathcal{E}_i \setminus \mathcal{E}_{i-1}$ and $\Delta\mathcal{D}_i = \mathcal{D}_i \setminus \mathcal{D}_{i-1}$. Then

$$\begin{aligned} \phi(\mathcal{E}_i + \mathcal{D}_i) &= \phi(\mathcal{E}_{i-1} + \mathcal{D}_{i-1}) + \phi(\Delta\mathcal{E}_i + \mathcal{D}_{i-1}) + \phi(\mathcal{E}_i + \Delta\mathcal{D}_i) = \\ &= \mathcal{E}_i + \phi(\Delta\mathcal{E}_i + \mathcal{D}_{i-1}) + \phi(\mathcal{E}_i + \Delta\mathcal{D}_i)\,, \end{aligned}$$

and $\mathcal{E}_i$ is already known.

We note that putting one periodic point per cycle into the set $\mathcal{D}$ is sufficient. This is because running into that point or running into the cycle is equivalent.

Algorithm CLASSIFY(M, D)

```
1  D ← D ∪ { periodic points on the orbits of B} ;
2  E ← D ;
3  oldΔE ← E, oldΔD ← D ;
4  PrevD ← ∅ ;
5  while oldΔE ≠ ∅ do
6      ΔE ← ∅, ΔD ← ∅ ;
7      CurrE ← E ;
8      forall (e, d) ∈ CurrE × oldΔD ∪ oldΔE × PrevD do
9          Orbit ← ∅, p ← φ(e + d) ;
10         while p ∉ Orbit and p ∉ E do
11             Orbit ← Orbit ∪ {p} ;
12             ΔE ← ΔE ∪ {p} ;
13             p ← φ(p) ;
14         end
15         if p ∈ Orbit then
16             ΔD ← ΔD ∪ {p} ;
17         E ← E ∪ Orbit ;
18     end
19     oldΔE ← ΔE ;
20     oldΔD ← ΔD ;
21     PrevD ← D ;
22     D ← D ∪ ΔD ;
23 end
24 return D \ D, which contains one point from every non-zero cycle
```

When we enter the main loop for the ith time, $\mathcal{E} = \mathcal{E}_i$, $old\Delta\mathcal{E} = \Delta\mathcal{E}_i$, $\mathcal{D} = \mathcal{D}_i$, $old\Delta\mathcal{D} = \Delta\mathcal{D}_i$ and $Prev\mathcal{D} = \mathcal{D}_{i-1}$. This can be proved by induction, establishing the correctness of the algorithm.

4. Comparison of the algorithms

In the article [6] the authors determine all binary canonical number system polynomials in higher degrees. The decision algorithms described in the present article were used to decide the number system property. We summarize our experiences below.

- Both algorithms α and β find some of the non-zero periodic points quickly, if such points exist. That is, if (M, D) is not a number system, the decision algorithms are fast.

- Both algorithms become exponentially memory and CPU-expensive in the worst cases as the degree of the polynomials grows. The largest set E we encountered was of size 21 223 091, for the polynomial $2+3x+3x^2+$ $+3x^3+3x^4+3x^5+3x^6+3x^7+3x^8+2x^9+x^{10}$. Storing such a large set of 10 dimensional vectors required smart programming tricks. The largest brick size we computed was even larger.

- We cannot say that either algorithm outperforms the other. In many cases, e.g. for polynomials fulfilling the monotonity condition of [12], Brunotte's algorithm is faster. In many cases, e.g. for the polynomial $2+x^3+x^7+x^{10}$, the algorithm using bricks is faster. Two small examples showing the set of fractions together with the set E are seen in Figure 1.

- The decision using covering bricks is easily parallelized, with no communication between processing units. This seems the only way to attack degree 12 for binary CNS-polynomials.

5. Further work

We are currently working on both algorithms. We try to give a deterministic version of FIND-BASIS-TRANSFORMATION. In the generalized version of Brunotte's algorithm, we try to address the memory problems we encountered in practice. Making the algorithm more memory-efficient is possible through space-time tradeoff.

References

[1] **Akiyama S., Borbély T., Brunotte H., Pethő A. and Thuswaldner A.,** On a generalization of the radix representation - a survey, *High primes and misdemeanours: Lectures in honour of the 60th birthday of Hugh Cowie Williams, Fields Inst. Commun.*, **41** (2004), 19-27.

[2] **Akiyama S., Brunotte H. and Pethő A.,** Cubic CNS-polynomials, notes on a conjecture of W.J. Gilbert, *J. Math. Anal. Appl.,* **281** (2003), 402-415.

[3] **Akiyama S. and Pethő A.,** On canonical number systems, *Theoret. Comput. Sci.,* **270** (2002), 921-933.

[4] **Akiyama S. and Rao H.,** New criteria for canonical number systems, *Acta Arith.,* **111** (1) (2004), 5-25.

[5] **Brunotte H.,** On trinomial bases of radix representations of algebraic integers, *Acta Sci. Math. Szeged,* **67** (2001), 407-413.

[6] **Burcsi P. and Kovács A.,** Algorithms for finding generalized binary number systems (submitted)

[7] **Gilbert W.J.,** Radix representation of quadratic fields, *J. Math. Anal. Appl.,* **83** (1981), 264-274.

[8] **Kátai I.,** Generalized number systems in Euclidean spaces, *Math. Comput. Model.,* **38** (2003), 883-892.

[9] **Kátai I. and Kovács B.,** Canonical number systems in imaginary quadratic fields, *Acta Math. Hungar.,* **37** (1981), 159-164.

[10] **Kátai I. and Kovács B.,** Kanonische Zahlensysteme bei reelen quadratischen Zahlen, *Acta Sci. Math. Szeged,* **42** (1980), 99-107.

[11] **Kátai and Szabó J.,** Canonical number systems for complex integers, *Acta Sci. Math. Szeged,* **37** (1975), 255-260.

[12] **Kovács B.,** Canonical number systems in algebraic number fields, *Acta Math. Hungar.,* **37** (1981), 405-407.

[13] **Kovács A.,** On computation of attractors for invertible expanding linear operators in $\mathbb{Z}^k$, *Publ. Math. Debrecen,* **56** (1-2) (2000), 97-120.

[14] **Kovács A.,** Number expansion in lattices, *Math. Comput. Modelling,* **38** (2003), 909-915.

[15] **Latimer C.G. and MacDuffee C.C.,** A correspondence between classes of ideals and classes of matrices, *Ann. Mathematics,* **34** (1933), 313-316.

[16] **Pethő A.,** On a polynomial transformation and its application to the construction of a public key cryptosystem, *Comput. Number Theory Proc.,* eds. A. Pethő, M. Pohst, H.G. Zimmer and H.C. Williams, Walter de Gruyter Publ. Comp., 1991, 31-43.

[17] **Scheicher K. and Thuswaldner J.M.,** On the characterization of canonical number systems, *Osaka J. Math.,* **41** (2) (2004), 327-351.

[18] **Taussky O.,** On a theorem of Latimer and MacDuffee, *Canad. J. Math.,* **1** (1949), 300-302.

P. Burcsi, A. Kovács and Zs. Papp-Varga
Department of Computer Algebra
Eötvös Loránd University
Pázmány Péter sét. 1/C
H-1117 Budapest, Hungary