ERGODIC PROPERTIES OF COMPUTER ORBITS FOR SIMPLE PIECEWISE MONOTONIC TRANSFORMATIONS

N. Friedman
School of Computer Science, McGill University, Montreal, Canada

A. Boyarsky and M. Scarowsky
Department of Mathematics, Concordia University, Montreal, Canada

(Received 11 December 1987; in revised form 19 January 1988)

Communicated by E. Y. Rodin

Abstract—Using a general model for normalized floating point arithmetic, we study the computer generated orbit of simple piecewise monotonic transformations on \([0, 1]\) of the form \(\tau(x) = kx \mod 1\). Conditions under which the computer orbit is eventually exact are established. Moreover it is shown that for sufficiently high precision of computation the computer orbits display the same ergodic properties as the exact orbits of the transformations.

1. INTRODUCTION

Let \(\tau: [0, 1] \rightarrow [0, 1]\) be a transformation and let \(C = \{\tau^n(x)\}_{n=0}^\infty\) denote an orbit that begins at \(x_0\) and ends at \(\tau^n(x_0)\). Let \(\tau\) admit an absolutely continuous ergodic measure \(\mu\), then the Birkhoff ergodic theorem states that for \(x \in S \subseteq [0, 1]\), \(\mu(S) = 1\),

\[
\lim_{m \to \infty} \frac{1}{m} \sum_{k=0}^{m-1} f(\tau^k x) = \int_0^1 f \, d\mu,
\]

where \(f \in L_1\), i.e. for every \(x \in S\), the orbit \(\{\tau^n(x)\}_{n=0}^\infty\) exhibits the ergodic measure \(\mu\). If \(\mu\) is a continuous measure (and this is the only interesting situation), every point in \([0, 1]\) has \(\mu\)-measure 0 and therefore no matter what starting point is used we cannot be certain that it will exhibit \(\mu\) in the sense of equation (1).

In Ref. [1] it is shown that for certain simple piecewise monotonic transformations \(\tau\), there exists a sequence of periodic orbits \(\{P_n\}\) such that the measures supported on these periodic orbits approach \(\mu\) in the topology of weak convergence on measures. Therefore, given a starting point \(x_n \in P_n\), \(n\) large, equation (1) will be approximately satisfied, i.e.

\[
\frac{1}{\text{per}(x_n)} \sum_{\text{per}(x_n)} f(\tau^k(x_n)) \approx \int_0^1 f \, d\mu,
\]

where \(\text{per}(x_n)\) denotes the number of points in the periodic orbit containing \(x\) and the summation is over all points in \(\text{per}(x_n)\).

Let \(T\) be a computer algorithm with the property that, given the computer representation of some number in \([0, 1]\), it produces a value in \([0, 1]\) in computer representation form. Then the computer orbit \(C_T = \{x_i\}_{i=0}^\infty\) is the sequence of (computer representable) points such that \(x_{i+1}\) is obtained by applying algorithm \(T\) to \(x_i\). In Section 2 we give precise definitions of computer representations and computer orbits. Now, if \(T\) is an algorithm for computing \(\tau\), what has been observed in practice is that \(C_T\) exhibits the same statistical behavior as the exact orbit \(C\) in the sense of expression (2). This is remarkable since truncation and round-off errors quickly produce an orbit which has no resemblance to the exact orbit. In this paper we shall study this computer phenomenon and prove that for a certain class of piecewise linear maps, the computer generated orbits exhibit the appropriate measure.

In Ref. [2] the interplay between computer and theoretical orbits is studied for a Chebysev map. Whereas the approach of Ref. [2] is somewhat heuristic, in this paper a theoretical model for
computer arithmetic is used to demonstrate a precise relationship between computer and theoretical orbits for certain maps. For an interesting discussion on computer orbits see Chap. 1 of Ref. [3].

2. COMPUTER ORBITS

Consider the class of maps \( \tau : [0, 1] \to [0, 1] \) defined by

\[
\tau(x) = kx \pmod{1},
\]

where \( k \) is an integer \( > 0 \). In this section we shall examine properties of the computer generated orbits of this map, that is, the behavior of the sequence of points \( x_0, x_1, \ldots, x_i, \ldots \) where \( x_0 \) is the computer representation of some point in \((0, 1)\) and \( x_{i+1} \) is obtained by applying a computer algorithm for evaluating \( kx_i \pmod{1} \).

Due to the round-off and truncation errors in conventional computation, computer arithmetic is inherently inexact. In order to focus on the effects of such errors on the computer generated orbit we shall assume that \( k \) is an integer greater than 1 which has an exact floating point representation. Without such a restriction the algorithm representing the map \( \tau \) would in fact be attempting to compute the orbit or an approximation to the map, thereby complicating the task of relating the computer orbit to exact orbits of \( \tau \). The restriction that \( k \) be greater than 1 guarantees interesting ergodic behavior since \( \tau \) will admit an absolutely continuous invariant measure [4] on \([0, 1]\). In fact the measure is a Lebesgue measure.

To study the computer generated orbit, we use a general model for normalized floating point arithmetic as described in Chap. 4 of Ref. [5]. Given a normalized floating point number \( x \), our algorithm computes \( kx \) using the multiplication algorithm described in Ref. [5], and if the result is greater than 1, performs a left shift followed by a normalization operation to compute the mod 1 operation. For the sake of completeness we describe Knuth's model and multiplication algorithm. Since our algorithm involves positive numbers and no addition is necessary, the algorithm can be presented in a simplified form.

The following definition describes the computer representation of real numbers as a base \( b \) floating point values. The restrictions on the exponent, \( e \), and fraction, \( f \), are motivated by the standard practice of allocating a fixed, finite number of digits for the representation of each of these values. The use of an unsigned representation for the exponent requires us to specify a value \( q \) in order to represent values \( b^i \) for \( i \leq 0 \). Other representations of exponents can be used without affecting the results of this paper.

**Definition 1**

A base \( b \), excess \( q, p \) digit floating point representation of a real number is a pair \((e, f)\) representing the number \( fb^{e - q} \). The value, \( e \), is a non-negative integer of bounded size, \( f \) is a signed fraction such that \( b^{-p} \leq f < b^{-p} \), i.e. \( f \) has at most \( p \) digits and \( |f| < 1 \). We assume \( q > p \) to allow the representation of all such fractions in \((0, 1)\).

**Definition 2**

A floating point representation \((e, f)\) is said to be normalized if \( b^{-1} \leq |f| < 1 \) or if \( f = 0 \) and \( e = 0 \).

For example, if \( b = 2, q = 0, p = 5 \), then \((11, 0.10111) \) [or \((3, \frac{11}{2}) \) in base 10] is the 5 digit normalized floating point representation of the real number 5.75 (in base 10).

Let \( k \), the slope of the map \( \tau(x) = kx \pmod{1} \), have normalized floating point (nfp) representation \( (r + q, f) \) where \( k = b^r f \), is an integer. Since \( k > 1 \), \( r = \lfloor \log_b k \rfloor + 1 \), where \( \lfloor A \rfloor \) is the largest integer \( \leq A \). Let \( x \in (0, 1) \) have nfp representation \((e_x, f_x)\). Given such an \( x \), define \( m(x) \) to be the smallest integer, \( l \), such that \( b^{l + \varepsilon} f_x < b^{-1} \) is an integer. Note that since \( x < 1, e_x \leq q \) and hence \( m(x) \geq 0 \). Define

\[
\delta_x = \begin{cases} 
1, & \text{if } f_x < b^{-1} \\
0, & \text{otherwise}
\end{cases}
\]

The following algorithm is equivalent to that in Ref. [5] for multiplying and then normalizing two
Ergodic properties of computer orbits

nfp numbers when restricted to numbers of the form \( k \) and \( x \) given above. The result is an nfp number. The presentation below is a form which simplifies the ensuing discussion, but is not necessarily as efficient as an actual implementation might be.

**Algorithm 1**

*Input:* \( k = (r + q, f_x) \), \( x = (e_x, f_x) \) (as described above).

*Output:* \((e, f)\) representing \( k \otimes x \) (where \( \otimes \) is the floating point multiplication operation).

*Algorithm:*

1. \( e' = e_x + r \)
   \( f' = f_x f_x \).
2. Compute \( \delta_x \) and set
   \( e'' = e' - \delta_x \)
   \( f'' = b^{e''} f' \).
3. (Round \( f'' \))
   \[ f''' = b^{-r} \left\lfloor b f'' + \frac{1}{2} \right\rfloor. \]
   (Note—this works because \( f''' > 0 \). If \( f''' \) were less than 0, we would round it up.)
   If \( f''' = 1 \) then set \( f = 0.1, \ e = e'' + 1 \).
   Otherwise set \( f = f''', e = e'' \).

For a given computer representation, the range of possible values for \( e \) is fixed. Multiplication of two numbers may produce a value \( e < 0 \) or a value of \( e \) too large to be represented. In the first case, exponent underflow is said to occur and in the second case exponent overflow occurs. Now in general \( e_x + r - 1 \leq e \leq e_x + r + 1 \). In our case, since \( r \geq 1 \) and \( e_x \leq q, e_x \leq e \leq r + q + 1 \). Thus exponent underflow cannot occur. We assume that \( r + q \) is strictly less than the maximum possible exponent, in order to guarantee that overflow will also not occur. (Note that if \( k \) is to be represented at all, \( r + q \) must be less than or equal to this maximum value and the restriction just precludes equality.)

In examining the algorithm given above for computing \( k \otimes x \), we note that the only source of computational error is the rounding step (step 3).

The following algorithm can be used to compute \( x(\text{mod} \, 1) \) where \( x \) has nfp representation \((e_x, f_x)\). All the steps can be implemented as simple shifts and introduce no computational errors.

**Algorithm 2**

*Input:* \( x = (e_x, f_x) \)

*Output:* \((e, f)\) representing \( x(\text{mod} \, 1) \) in normalized form.

*Algorithm:*

1. If \( e_x \leq q, e = e_x, f = f_x \), otherwise set
   \( e' = q \)
   \( f' = b^{e_x - q} f_x - \lfloor b^{e_x - q} f_x \rfloor. \)
   (Note that this is accomplished by shifting left \( e_x - q \) digits.)
   If \( f' = 0 \), then set \( f = e = 0 \) and exit.
2. (Normalization)
   Let \( n \) be the largest non-negative integer \( l \) such that \( f' < b^{-1} \). Since \( 0 < f' < 1 \), \( n \) is well defined. Then \( e = q - n, f = b^n f' \). Since \( n \leq p \leq q, e \geq 0 \) and underflow does not occur.

Remark. If \( b f_x \) is an integer, then \( b^{e_x - q} f' \) and hence \( b^{e_x - q} f \) are integers.

Let \( \tau(x) = k x(\text{mod} \, 1) \) be a given function with \( k \) satisfying the restrictions mentioned above. Suppose \( x \) has a base \( b \), excess, \( q, p \), digit representation. Then \( T(x) \) is the result of applying Algorithms 1 and 2 to the normalized representation of \( x \). The computer orbit of such a point...
x = x₀ is the sequence of points with nfp representations \( \{x_i\}_{i=0}^{\infty} \), where \( x_{i+1} = T(x_i) \). Given a point \( x \) with an nfp representation, we say the computation is exact if \( T(x) = \tau(x) \).

In the remainder of this section we investigate the relationships between this computer orbit and the exact orbit \( \{\tau(x_i)\}_{i=0}^{\infty} \).

For values of \( x \) in which the term \( f'' \) computed in algorithm 1 has \( p \) or fewer digits, no error in either Algorithm 1 or 2 occurs and the computation is exact. This leads to the following result:

**Lemma 1**

Let \( k \) and \( x \) have nfp representation \((r + q, f)\) and \((e_x, f_x)\), respectively where \( b^qf \) is an integer. Let \( k \otimes x \) be the result of applying algorithm 1 to \( k \) and \( x \). If \( m(x) + r \ll p \), the computation is exact, i.e. \( T(x) = \tau(x) \). Moreover, if \( y = T(x) \), \( m(y) \ll m(x) \).

**Proof** The number \( b^qf \) is an integer and by definition \( b^{e_x + qf_x} \) is an integer. Therefore,

\[
b^{e_x + qf_x} \ll b^{e_x + p - r - qf_x} \quad \text{[since \( m(x) \ll p - r \)]}
\]

\[
b^{e_x - q} \ll 0 \quad \text{[since \( e_x - q \ll 0 \)]}
\]

which is therefore integral. Now

\[
b^{e_x + qf_x} = b^{e_x + b^qf_x f_{e_x}} = (b^{e_x + qf_x})(b^qf_x)
\]

and is therefore an integer. It follows that

\[
\lfloor b^{e_x + qf_x} + \frac{1}{2} \rfloor = b^{e_x + qf_x}
\]

i.e. no round-off error occurs and hence \( k \otimes x = kx \). Since Algorithm 2 introduces no new computational errors, we have \( T(x) = \tau(x) \).

Now let \( y = T(x) \) have nfp representation \((e_y, f_y)\). To show that \( m(y) \ll m(x) \), it suffices to show that \( b^{e_x + e_y - qf_y} \) is an integer. Consider

\[
b^{m(x) + e_y - qf_y} = b^{m(x) + e_x - r - qf_x} (b^qf_x f_{e_x})
\]

\[
= (b^{m(x) + e_x - qf_x})(b^qf_x)
\]

which is an integer. Now if \( e'' \ll q, e_y = e'' \) and \( f_y = f'' \) we are done. If \( e'' \ll q, e_y \ll q \) and \( f_y = b^{e'' - q} (b^{e'' - q} - \lfloor b^{e'' - q} \rfloor) \), the remark following algorithm 2 yields:

\[
b^{m(x) + e_y - q - (e'' - q)} + e_y - qf_y = b^{m(x) + e_y - qf_y},
\]

is an integer.

Q.E.D.

An immediate consequence of this result is found in the following Theorem.

**Theorem 1**

If \( k \) and \( x_0 \) satisfy the hypothesis of Lemma 1, then the computer orbit \( \{x_i\}_{i=0}^{\infty} \) is exact, where \( x_{i+1} = T(x_i) \).

This theorem can be viewed as follows: given a slope \( k \) for the map \( \tau(x) = kx \mod 1 \), where \( k \) is an integer with \( |k| < b^r \), and a starting point \( x \) of the form \( 0.a_1a_2a_3 \ldots a_m \) (using a base \( b \) representation), the computer is guaranteed to generate an exact orbit [i.e. \( T^n(x) = \tau^n(x) \mod 1 \)] if the arithmetic can be done to a precision of at least \( r + m \). In some recent computer languages (such as Ada) it is possible to specify the desired precision of computation and thus guarantee, a priori, the computation of an exact orbit for starting points of a given precision.

**Corollary 1**

Let \( n \ll p - r \) be a positive integer and let \( a \ll b^r \) be a positive integer, where \( p \) denotes the number of digits in the floating point representation and \( b \) is the base. Then if \( x_0 \) is the nfp representation of \( a/b^r \), the computer orbit described in Theorem 1 is exact.
Proof. Let $x_0$ have nfp representation $(e, f)$, where
\[ f = \frac{a}{b} \]
Then $e = i + q - n$ and since $b'f$ is an integer, $m(x) \leq i - e + q = n$. Thus the hypothesis of Lemma 1 is satisfied.

In the remainder of this section we study computer orbits which may not be exact and attempt to demonstrate conditions under which they are eventually exact. The following two lemmas show that under certain conditions potential round-off errors do not actually occur.

**Lemma 2**

Let $k$, $x$ and $y$ be as in Lemma 1.

(a) If $m(x) + r = p + 1$ and $\delta_x = 1$, then the computation is exact and $m(y) \leq m(x)$.

(b) If $m(x) + r = p + 1$, $\delta_x = 0$ and $e_x < q$, then the computation is exact and $m(y) \leq m(x)$.

**Proof.** To show the computation is exact, as in Lemma 1, it suffices to show that $b\phi f''$ is an integer. Consider
\[ b\phi f'' = b\phi f + \delta_x f = b\phi f_x f, \]
In part (a), $\delta_x = 1$. Since $e_x - q \leq 0$,
\[ b\phi f'' = b m(x) + f_x f \geq (b m(x) + e_x - q f)(b'f), \]
which is an integer by the definition of $m(x)$ and $r$. In part (b), since $e_x < q$ and both are integers, $e_x - q \leq -1$ and hence
\[ b m(x) + e - q f'' = b m(x) + e + r - \delta_x - q + b_x f_x f, \]
which is an integer. Completing the argument as in Lemma 1, the result follows.

Q.E.D.

When $m(x) + r = p + 1$, there is a potential error as a result of the multiplication. Lemma 2 presents conditions under which this round-off does not occur. The next result considers the only remaining case and shows that if any error occurs, the result is such that no further error will occur in computing the following points of the orbit.

**Lemma 3**

Let $k$, $x$ and $y$ be as in Lemma 1. If $m(x) + r = p + 1$, $\delta_x = 0$ and $e_x = q$, then $m(y) < m(x)$.

**Proof.** In this case,
\[ b\phi f'' = b m(x) + r - 1 f_x f \geq b^{-1}(b m(x) + e_x - q f)(b'f), \]
Since $m(x)$ is minimal this is not necessarily an integer and round-off and/or truncation error may occur in step 3 of Algorithm 1. Then
\[ f'' = b^{-1}(b\phi f'' + \frac{1}{2}). \]

**Case (1).** If $f'' = 1$, then $e = e_x + r + 1$ and $f = b^{-1}$. It follows that
\[ b m(x) + e - q f = b m(x) + e + r - q - 1 = b m(x) + r - 1, \]
which is an integer since $r \geq 1$ and $m(x) \geq 0$.

**Case (2).** If $f = f''$, then
\[ b m(x) - 1 + e - q f = b m(x) - 1 + e + r - q f \]
\[ = b f = \lfloor b f'' + \frac{1}{2} \rfloor, \]
which is an integer. Therefore applying the argument at the end of Lemma 1, $m(y) < m(x) - 1 < m(x)$.

Q.E.D.
It is now possible to show that under certain conditions a computer orbit is eventually an exact orbit.

**Theorem 2**

If \( \{x_j\}_{j=0}^\infty \) is a computer generated orbit of the transformation \( kx \mod l \) and \( x_i > b^{-1} \) for some \( i \), then there exists a \( t \) such that \( T^t(x_i) = \tau(x_i) \forall j \), i.e. the orbit is exact from \( x_i \) on.

**Proof.** Let \( m_j = m(x_j), e_j = e_{x_j}, f_j = f_{x_j}, \delta_j = \delta_{x_j} \forall j \). Now \( m_i \leq p \) since \( e_i = q \) and \( b^r f_j \) is an integer. If \( r = 1, m_i + r \leq p + 1 \). In computing \( k \otimes x_i \), assuming \( f'' \neq 1 \) (if \( f'' = 1, \) we are done), we have

\[
\begin{align*}
e &= q + r - \delta_i \\
f &= b^{-p}[b^r f + \frac{1}{2}].
\end{align*}
\]

If \( r > 1 \), we apply Algorithm 2 and obtain

\[
e_{i+1} = q - n,
\]

and a corresponding \( f_{i+1} \) with the property that \( b^{e-r+2} f_{i+1} \) is an integer. Therefore,

\[
m_{i+1} \leq p - r + \delta_i.
\]

Thus, \( m_{i+1} + r \leq p + 1 \) in either case.

By Lemmas 1 and 2, the sequence \( \{x_j\}_{j=i+1}^\infty \) is either exact (in which case \( t = i + 1 \)) or

\[
m_{i+1} + r = p + 1, \quad \delta_{i+1} = 0 \quad \text{and} \quad e_{i+1} = q.
\]

In this case, by Lemma 3, \( m_{i+2} + r \leq p \) and applying Lemma 1 again we can take \( t = i + 2 \).

Q.E.D.

### 3. THE LENGTH OF PERIODIC ORBITS

In Section 2 it was shown that, for transformations \( T(x) = kx \mod 1 \), the computer orbit for certain rational starting points is exact. More specifically, using \( p \) precision base \( b \) floating point arithmetic, any starting point which can be represented using fewer than \( p - \left\lfloor \log_b k \right\rfloor \) digits yields an exact orbit. Since all the values computed are rational, the orbit is eventually periodic.

It is possible that the period is small and then little ergodic information about the orbits of the dynamical system can be obtained. For example, if \( T(x_i) = 1 \), the computer orbit becomes a trivial exact orbit with period 1. In this section we use number theoretic arguments to show the existence of long periodic orbits for certain rational points. Such points which satisfy the hypothesis of Theorem 1 lead to long exact computer orbits. We then study the distribution of points in such periodic orbits and demonstrate that for certain computer representable starting points, the computer generated orbit has a distribution which approximates Lebesgue measure.

Lemmas 4 and 5 of this section are technical lemmas which are number theoretic in nature. We leave their proofs for the Appendix.

Let \( x > 0 \) be a rational point in \([0, 1]\) with a periodic orbit. Let \( \alpha/\beta \) be the reduced representation of \( x \) (i.e. \( (\alpha, \beta) = 1 \), where \( (\alpha, \beta) \) is the gcd of \( \alpha \) and \( \beta \)). Since \( x \) is periodic, we must have \( (k, \beta) = 1 \). (Otherwise the orbit could never return to its starting point.) Suppose \( x \) has period \( \ell \), i.e. \( \tau^\ell(x) = x \).

Since \( x \) under the transformation can be expressed as \( \tau(x) = kx - \left\lfloor kx \right\rfloor \), we have \( k^\ell x - A = x \), for some integer \( A \). Writing \( x = \alpha/\beta \) gives \( (k^\ell - 1)\alpha/\beta = A \), that is \( \beta | k^\ell - 1 \) (where \( w | y \) means \( w \) divides \( y \)). Let \( \beta = p_1^{n_1} p_2^{n_2} \ldots p_r^{n_r} \) be the prime factorization of \( \beta \) where \( p_0 = 2 \) and \( n_i > 0 \) for \( 1 < i < t \).

We allow \( n_0 = 0 \) if \( \beta \) is odd. Then \( p_0^{n_0} | k^\ell - 1 \forall 0 \leq i \leq t \). Note that \( (\alpha, p_0) = 1 \) and \( (k, p_i) = 1 \forall 0 \leq i \leq t \) since \( \alpha/\beta \) is in reduced form and \( (k, \beta) = 1 \).

By Fermat's theorem, \( p | k^{p-1} - 1 \) for any prime \( p \) with \( (k, p) = 1 \). Let \( l \) be the smallest exponent such that \( p_i | k^{l-1} \) (note that \( l \leq p_i - 1 \)) and let \( m_i, i > 0 \), be such that \( p_i^{m_i} | k^l - 1 \) (where \( w' | y \) means \( w' \) divides \( y \) but \( w' + 1 ! y \)).

**Lemma 4**

For \( i > 0 \), if \( n_i \geq m_i \), \( l \) is a multiple of \( l, p_i^{m_i - m} \); if \( n_i < m_i \), then \( l \) is a multiple of \( l_i \). Thus, \( l, p_i^{m_i - m} \) is the smallest integer \( l \) such that \( p_i^{n_i} | k^l - 1 \) and for this \( l, p_i^{n_i} | k^l - 1 \).
In the following lemma we establish a similar result for \( p_0 = 2 \). In this case, for \((\beta, 2) = 1\) Fermat's theorem gives \( l_0 = 1 \). Let \( m_0 \) be defined as follows.

**Case (i).** If \( 2^2|k - 1 \) then \( m_0 \) is such that \( 2^{m_0} | k - 1 \).

**Case (ii).** If \( 2|k - 1 \) then \( m_0 \) is such that \( 4^{m_0} | k^2 - 1 \).

**Lemma 5**

If \( 2^\infty |k^l - 1, n_0 > 0 \), then \( l \) is a multiple of \( 2^\infty - m_0 \) and in case (i), \( 2^\infty | k^{2^{\infty} - m_0} - 1 \).

Using these two lemmas, it is possible to calculate the period of \( x = \alpha/p_0^n \ldots p_i^n = \alpha/\beta \). If the period is \( l \), we have \( p_i^n | k^l - 1 \) \( \forall 0 \leq i \leq t \). By Lemma 4, for each odd \( p_i \), the minimum value of \( l \) is \( l,p_i^n - m_i \) and by Lemma 5, for \( p_0 = 2 \), the minimum value of \( l \) is \( 2^{\infty} - m_0 \). Therefore, the minimum value, \( L \), of \( l \) such that \( \beta | k^l - 1 \) is given by

\[
L = \text{lcm}\{l,p_i^n - m_i|0 \leq i \leq t\}.
\]

We have verified that the length of the period starting at \( x = \alpha/\beta \) is greater than or equal to \( L \).

To prove that the period is exactly \( L \), we calculate

\[
\tau_k^L(x) = \tau_k^L \alpha \beta = k^L \alpha \beta - \left[ k^L \frac{\alpha}{\beta} \right] = k^L \frac{\alpha}{\beta} - \left( \frac{(k^L - 1)\alpha}{\beta} + \frac{\alpha}{\beta} \right).
\]

By the definition of \( L \), \((k^L - 1)\alpha/\beta \) is an integer. Therefore,

\[
\tau_k^L(x) = k^L \frac{\alpha}{\beta} - (k^L - 1) \frac{\alpha}{\beta} = \frac{\alpha}{\beta} = x.
\]

We have proved the following theorem.

**Theorem 3**

Let \( \tau_k(x) = kx(\text{mod } 1) \) and let

\[
x = \frac{\alpha}{p_0^n \ldots p_i^n} \in [0, 1],
\]

where \( p_0 = 2, n_i > 0 \) for \( 1 \leq i \leq t \) and \((\alpha, p_i) = 1 \forall i \) with \( n_i > 0 \). Then \( x \) is a periodic point and the length of the period, \( L = \text{lcm}\{l,p_i^n - m_i|0 \leq i \leq t\} \), where \( l_i \) is the minimum positive integer such that \( p_i^l | k^l - 1 \); \( m_i \) is defined by \( p_i^n | k^l - 1 \) for \( 1 \leq i \leq t \) and \( m_0 \) is defined by \( 2^{\infty} | k - 1 \) if \( 2^2|k - 1 \) or \( 4^{m_0} | k^2 - 1 \) if \( 2|k - 1 \).

Note that we have implicitly assumed \( n_i \geq m_i \). These are not major restrictions and \( L \) can be calculated without difficulty when these do not hold. For example, if \( n_i < m_i \) for \( c > 0 \), we can replace \( l,p_i^n - m_i \) by \( l_i \). Similarly, if \( n_0 < m_0 \), we replace \( 2^{\infty} - m_0 \) by \( 1 \). That is, in the definition of \( L \), we define \( p^2 \) to be \( 1 \) when \( \alpha < 0 \).

**Example 1**

Consider the length, \( L \), of the orbit of the point \( \alpha/2^n \), where \((\alpha, 2) = 1 \). \( L \) is given by \( 2^{\infty} - m_0 \) where \( m_0 \) depends only on \( k \). (Note: \((k, 2) = 1 \).) Some values of \( m_0 \) as a function of \( k \) are given below:

<table>
<thead>
<tr>
<th>( k )</th>
<th>( m_0 )</th>
<th>( k )</th>
<th>( m_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>17</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>25</td>
<td>2</td>
</tr>
</tbody>
</table>

From the results of Ref. [6], upper and lower bounds for the length of the periodic orbit starting at rational points can be derived.
4. DISTRIBUTION OF PERIODIC ORBITS

Let \( \tau_b(x) = kx \mod 1 \), where \( k \) is an integer > 1. In this section we shall restrict our attention to base \( b = 2 \). Let \( x = \alpha/2^N \) with \( (\alpha, 2) = 1 \). In the notation of Section 3, \( N = n_0 \). We also assume that \( (k, 2) = 1 \). Under these conditions \( x \) is periodic and the period is given by \( 2^{m_0}k - 1 \), where \( m_0 \) is defined by: \( 2^{m_0}k \equiv 1 \mod 2^m \) if \( 2 \nmid k - 1 \). If the exponents \( N - m_0 \) are negative, the expression is defined to be equal to 1.

Let \( \text{per}(x) \) denote the period of \( x \) and let \( m \) denote Lebesgue measure on \([0, 1] \). Let \( \chi_i(x) \) be the characteristic function of the set \( I \), i.e. \( \chi_i(x) = 1 \) if \( x \in I \) and 0, otherwise.

**Theorem 4**

The proportion of points in a periodic orbit \( \{\tau_i(x_0)\}_{i=0}^{\text{per}(x_0)} \), where \( x_0 = \alpha/2^N \), which lies in an interval \( I = (c, d) \subset [0, 1] \) approaches \( m(I) = d - c \) as \( N \to \infty \), i.e.

\[
\frac{1}{\text{per}(x)} \sum \chi_I(\tau_i(x)) \to m(I),
\]

as \( N \to \infty \), where the summation is over a complete period.

**Proof.** Let \( l = \text{per}(x) \). By the division algorithm, there exist integers \( q_i \) and \( r_i \) such that

\[
k_i \cdot \frac{\alpha}{2^N} = q_i + \frac{r_i}{2^N}, \quad i = 0, 1, \ldots, l - 1,
\]

where \( 0 < r_i < 2^N \) and \( q_i \geq 0 \). Then \( \tau_i(x) = r_i/2^N \) and, because no value is repeated in the period of \( x_0 \), the \( r_i \)'s are distinct. It follows that the sum in the l.h.s. of expression (3) is

\[
\frac{1}{l} \sum \chi_i(\tau_i(x)) = \frac{1}{l} \left( \# \left\{ i : c < \frac{r_i}{2^N} < d \right\} \right)
\]

\[
= \frac{1}{l} \left( \# \left\{ i : 2^Nc < r_i < 2^Nd \right\} \right),
\]

where \( \# \{A\} \) denotes the number of points in the set \( A \). We now claim that \( r_i = 2^m i' + r', \) with \( 0 \leq i' < 2^{N-m_0} \) and \( 0 \leq r' < 2^m \), where \( r' \) is just \( r_i \) for the case \( N = m_0 \) (case (i)), and the \( i \) and \( i' \) are in 1-1 correspondence. To show this, note that if we replace \( i \) by \( i + 2^{m_0} \) we have

\[
k_i + 2^{m_0} \cdot \frac{\alpha}{2^N} = q_i + 2^{m_0} + \frac{r_i + 2^{m_0}}{2^N},
\]

or, multiplying by \( 2^N \) and using the fact that \( k^{2^{m_0}} = 1 \mod 2^{m_0} \), we obtain:

\[
k_i \cdot \frac{\alpha}{2^N} \equiv r_i + 2^{m_0} \cdot r' \mod 2^{m_0}.
\]

However, the l.h.s. equals \( 2^N q_i + r_i \equiv r_i \mod 2^{m_0} \). Thus, we have shown that \( r_i + 2^{m_0} \equiv r_i \mod 2^{m_0} \), i.e. the \( r_1 \) have a period of \( 2^{m_0} \) as a function of \( i \). Hence we can write \( r_i = 2^{m_0} i' + r' \), where for each \( i, r' \) is the residue of \( r_i \mod 2^{m_0} \) and the \( i' \) are distinct. This completes the proof of the claim. Therefore,

\[
\frac{1}{l} \sum \chi_i \left( \frac{r_i}{2^N} \right) = \frac{1}{l} \left( \# \left\{ i' : 2^N c < 2^{m_0} i' + r' < 2^N d \right\} \right)
\]

\[
= \frac{1}{l} \left( \# \left\{ i' : 2^N c - r' < 2^{m_0} i' < 2^N d - r' \right\} \right).
\]

However, the number of integers in an interval \((c, d)\) is equal to \(|d - c| + o(1)\). Thus,

\[
\frac{1}{l} \sum \chi_i \left( \frac{r_i}{2^N} \right) = \frac{1}{l} \left( 2^{N-m_0} (d - c) + o(1) \right)
\]

\[
= \frac{1}{2^{N-m_0}} \left( 2^{N-m_0} (d - c) + o(1) \right)
\]
Ergodic properties of computer orbits

\[ = d - c + o \left( \frac{1}{2^{N - n}} \right) \]
\[ \rightarrow d - c = m(c, d) \quad \text{as} \quad N \rightarrow \infty \]

Q.E.D.

Remark. This proof goes through for \( x = \alpha/p^n \), where \((\alpha, p) = 1, p \) is prime and \((k, p) = 1\).

5. ERGODIC PROPERTY OF COMPUTER ORBITS

Consider the map \( \tau_k(x) = kx \mod 1 \) with \((k, 2) = 1\) and the starting point \( Nx_0 = \alpha/2^N \), where \((\alpha, 2) = 1\) as in Section 4. By Theorem 1 we know that the computer orbit starting at \( Nx_0 \) will be exact if the precision of computation is sufficiently high. By Theorem 3, the length of the periodic orbit and, therefore, the computer orbit starting at \( Nx_0 \) increases with \( N \), the number of digits in the representation of \( Nx_0 \). Finally, Theorem 4 proves that the sequences of these periodic orbits starting at \( \alpha/2^n \) approaches Lebesgue measure on \([0, 1]\). We have therefore established the following theorem.

Theorem 5

Let \( \tau_k(x) = kx \mod 1 \), where \( k \) is an integer \( > 1 \), \((k, 2) = 1\). Let \( \{Nx_0 = \alpha/2^N\} N \geq 1 \) be a sequence of starting points, where \((a, 2) = 1\). Then for sufficiently high precision of computation \( p \) the computer orbit starting at \( Nx_0 \), \( N \leq p - r \), will have a distribution which is approximately Lebesgue measure, where \( r \) is the number of digits in the floating point representation of \( k \) in base 2.

Acknowledgment—This research was supported by NSERC grants.

REFERENCES


APPENDIX

Proof of Lemma 4

The proof is by induction on \( n_i \). Suppose \( n_i \leq m_i \), but \( l \) is not a multiple of \( l_i \). Then \( l = cl_i + d \) for some \( d < l_i \). Now \( k^i - 1 = sp^n_i \) for some \( s \) and

\[ k^i - 1 = k^d (sp^n_i + 1)^{1 - 1} = (k^d - 1)(sp^n_i + 1)^{1 - 1} = (sp^n_i + 1)^{1 - 1} - 1. \]

Now \( p^n_i [ (sp^n_i + 1)^{1 - 1} - 1 ] \) but \( p^n_i [ (sp^n_i + 1)^{1 - 1} - 1 ] \). Since \( p^n_i [ k^i - 1 \), we must have \( p^n_i [ k^d - 1 \), contradicting the minimality of \( l_i \). Now we assume the lemma is true for \( n_i \geq n_i \) and let \( N = m_i + 1 \). If \( p^n_i [ k^i - 1 \), then \( p^n_i [ k^i - 1 \) and thus, by the hypothesis \( l = cl/p^n_i - 1 \) for some integer \( c \geq 0 \). Thus,

\[ k^i - 1 = k^0 p^n_i - 1 = (sp^n_i + 1)^{1 - 1} - 1, \]

where \((s, p) = 1\)

\[ = csp^n_i + p^n_i + 1 \] [other terms].

Since \( p^n_i [ k^i - 1, p_i | c \) and hence \( l = (c/p_i)l/p^n_i - 1 \). It is not difficult to show that \( p^n_i [ k^i - 1, l/p^n_i - 1 \). Q.E.D.

Proof of Lemma 5

Case (i)

The proof is by induction. The result is true for \( n_0 = m_0 \). Assume it is true for \( n_0 = N \). Now if \( 2^{N+1} [ k^i - 1 \), then since
2^N | k^l - 1 we have \( l = m 2^N \), where \( m \) is a positive integer. Thus,
\[
k^l - 1 = k^{m2^N} - 1 = (k^{2^N})^m - 1,
\]
where \((c, 2) \) and \( N \geq 2 \). The last expression can be written as \( mc 2^N + \) multiple of \( 2^{2N} \), and therefore, \( 2|m \). Also, for \( m = 2 \) this expression is exactly divisible by \( 2^{N+1} \), and thus Case (i) is proved.

**Case (ii)**

We assume that \( 2 \nmid k - 1 \). Thus, \( 2^{m+1} \| k^l - 1 \), where \( m_0 \geq 2 \). Now if \( 2^N \| k^l - 1 \) and \( N = 1 \), then \( l \) is arbitrary. If \( N \geq 2 \), then \( l \) is even and we may now apply Case (i) to \( 2^N | k^l - 1 \) or \( 2^N | (k^2)^l - 1 \) to obtain \( \frac{l}{2} = \) multiple of \( 2^{N-m_0+1} \) and \( l = \) multiple of \( 2^{N-m_0} \). If \( N = 1 \), we interpret the last expression as 1.

Q.E.D.