Formalising Sharkovsky's Theorem (Proof Pearl)

Bhavik Mehta
Department of Pure Mathematics and Mathematical Statistics
Cambridge, United Kingdom
bm489@cam.ac.uk

Abstract

Sharkovsky's theorem is a celebrated result by Ukrainian mathematician Oleksandr Sharkovsky in the theory of discrete dynamical systems, including the fact that if a continuous function of reals has a point of period 3, it must have points of any period. We formalise the proof in the Lean theorem prover, giving a characterisation of the possible sets of periods a continuous function on the real numbers may have. We further include the converse of the theorem, showing that the aforementioned sets are achievable under mild conditions.

CCS Concepts: • **Theory of computation** → *Type theory*; **Logic and verification**; • **Mathematics of computing** → **Continuous functions**.

Keywords: formal math, real analysis, Lean, proof assistant

ACM Reference Format:
Bhavik Mehta. 2023. Formalising Sharkovsky's Theorem (Proof Pearl). In *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP '23), January 16–17, 2023, Boston, MA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3573105.3575689

1 Introduction

Given a continuous function $f : \mathbb{R} \to \mathbb{R}$, a point $x \in \mathbb{R}$ has *least period* $n \in \mathbb{N}_{>0}$ if $f^n(x) = x$, but $f^m(x) \neq x$ for any $0 < m < n$. It is a remarkable fact that the existence of a point of least period three implies the existence of points of any least period, a property sometimes known as 'period three implies chaos', after the 1975 paper [6] of this title which proved this result. However unbeknownst to the authors of that paper, a stronger result was shown by Ukrainian mathematician Sharkovsky in 1964 [9] who described the structure of the set of least periods.

Specifically, the Sharkovsky ordering is a linear order (here denoted $\preceq$ to distinguish from the usual order $\leq$) on positive integers, and he showed that the set s of least periods of any such function must be upwards-closed in this order, i.e. for any $n \in s$, if $n \preceq m$ then $m \in s$ also. Thus, the result that 'period three implies chaos' follows immediately by observing that $3 \preceq n$ in the Sharkovsky order. In fact, Sharkovsky also showed the converse to this theorem: any upwards-closed set of the Sharkovsky order can be realised as the set of periods of some continuous real-valued function. For clarity, throughout this article we will call these two results the *Sharkovsky forcing theorem* and *Sharkovsky realisation theorem* respectively, and their combination the *Sharkovsky theorem* following the descriptions by Burns and Hasselblatt [1].

In this article we describe our formalisation of both of these in the Lean theorem prover, following a more direct version of Sharkovsky's original argument for the forcing theorem by Du [5], as well as the construction for the realisation theorem as presented by Burns and Hasselblatt [1]. To the best of our knowledge, none of these theorems have been formalised previously.

We use results from the Lean mathematical library mathlib [7], which is characterised by a distributed and decentralised community of contributors, and ubiquitous classical reasoning. Throughout the paper we provide code blocks of Lean from our formalisation, with some adjustments for the sake of readability. The full formalisation is available in the supplementary material, as well as at https://github.com/b-mehta/sharkovsky.

We highlight that the formalisation of the forcing theorem very closely resembles the informal argument while being fully rigorous, though the same cannot be said for the realisation theorem, where reliance on 'geometrically obvious' facts and some explicit numeric calculations make the converse more difficult to formalise. The informal proofs do not require a large amount of background knowledge, simply undergraduate real analysis, though the arguments are still somewhat subtle. In this way, the Sharkovsky theorem can be seen as an introduction to dynamical systems theory and chaos theory, while not requiring any serious understanding of the measure theory usually required in these fields.

Our main contributions are thus to exposit a new formalisation of an elegant yet elementary piece of mathematics, and hence to show some of the joys and pains of formalising modern mathematics.

Outline. In Section 2 we more precisely state the definitions and theorems which have been formalised, and give an idea of the proof techniques involved. Section 3 explains the formalisation of the key prerequisites for Sharkovsky's theorem (such as the Sharkovsky order), and describes those prerequisites which were already present in mathlib. Next, sections 4 and 5 illustrate the formalisations of the forcing and realisation theorems respectively, explaining the design decisions made and difficulties along the way, and section 6 concludes.

2 Mathematical Background

Definition 2.1 (Period, least period)**.** For a function $f : A \to A$ from some set A to itself, the point $x \in A$ is said to have *period* n if $f^n(x) = x$, where f^n denotes the nth iterate of f. It has *least period* (sometimes, *minimal period*) n if n is the smallest positive integer which is the period of x. In this case, we also say n is a least period for f.

Definition 2.2 (Sharkovsky ordering)**.** The Sharkovsky ordering $\prec$ on positive integers is defined[1] as

$$\begin{aligned} & 3 \prec 5 \prec 7 \prec 9 \prec \cdots \\ & \prec 2 \cdot 3 \prec 2 \cdot 5 \prec 2 \cdot 7 \prec \cdots \\ & \prec 2^2 \cdot 3 \prec 2^2 \cdot 5 \prec 2^2 \cdot 7 \prec \cdots \\ & \vdots \\ & \prec 2^n \cdot 3 \prec 2^n \cdot 5 \prec 2^n \cdot 7 \prec \cdots \\ & \vdots \\ \cdots \prec 2^n \prec \cdots \prec 16 \prec 8 \prec 4 \prec 2 \prec 1. \end{aligned}$$

That is, the odds excluding 1 in increasing order, then their doubles, and so on; and finally the powers of two in decreasing order. We may then say $n \preceq m$ iff $n = m$ or $n \prec m$.

Observe that we indeed have $3 \preceq n$ for any n.

Theorem 2.3 (Sharkovsky forcing theorem)**.** *Suppose $I \subseteq \mathbb{R}$ is an interval (not necessarily closed or bounded), and $f : I \to I$ is continuous. If m is a least period for f and $m \preceq n$, then n is a least period for f. Equivalently, the set of least periods of f is upwards closed in the Sharkovsky order.*

Note that this is stronger than the statement given in the introduction, since we are now talking about functions which need not be defined on all of $\mathbb{R}$.

Theorem 2.4 (Sharkovsky realisation theorem)**.** *Every upwards closed set of the Sharkovsky order is the set of least periods of some continuous function $f : \mathbb{R} \to \mathbb{R}$.*

Some authors [1] state this result for continuous functions only on compact intervals, despite the original version being for the whole real line, but if we restrict to compact intervals only the result is no longer true.

Both of these theorems have many possible proofs, Du [4] presents eight different proofs of the forcing theorem, and three proofs of the realisation theorem: we have just chosen the proof which appears simplest. For instance, the other seven proofs in [4] of the forcing theorem use directed graphs and some additional definitions from dynamical systems theory, while the proof in [1] essentially uses a construction known as Štefan cycles. Certainly these notions can be formalised in Lean, but we instead aim for the more direct proofs for clarity. Thus, our proof of the forcing theorem is a very slight adaptation of the first proof presented in [5], and for the realisation theorem we use [1].

2.1 Proving the Forcing Theorem

Throughout this section I will be a real interval (not necessarily closed or bounded), and $f : I \to I$ will be a continuous function.

The key results which will be used repeatedly are the intermediate value theorem, and its direct consequence on fixed points whose proof we omit here.

Lemma 2.5. *If J is a closed subinterval of I and $f(J) \supseteq J$, then f has a fixed point (i.e. a point of period 1) in J.*

The proof of the forcing theorem splits into five parts:

(a) If 2 is a least period for f, then 1 is a least period for f.
(b) If $m \geq 3$ is a least period for f, then 2 is a least period for f.
(c) If $m \geq 3$ is a least period for f with m odd, then $m + 2$ is a least period for f.
(d) If $m \geq 3$ is a least period for f with m odd, then 6 and $2m$ are both least periods for f.
(e) The previous four imply the forcing theorem.

We present first the proof of (b) to illustrate the key ideas required, and suggest the modifications required for (a), (c), (d).

Proof sketch for (b). Say x has least period m, and write P for the orbit of x under f (i.e. $P = \{f^i(x) : 0 \leq i < m\}$). Set $b = f^{m-1}(\min P)$.

First claim the existence of a point $a \in [\min P, b]$ with $f(a) \geq b$: if not, for all $i \geq 1$, $f^i(\min P) < b$, contradicting the definition of b. Then Lemma 2.5 asserts the existence of a fixed point z in $[a, b]$. By the intermediate value theorem, find a point $v \in [a, z]$ with $f(v) = b$.

We now have $f^2(\min P) > \min P$ (since $m \geq 3$) and $f^2(v) = \min P < v$, so again Lemma 2.5 says a fixed point of f^2 exists: suppose y is the *largest* such.

Now certainly y has period 2, but we must show it has *least* period 2, i.e. that $f(y) \neq y$. But suppose $f(y) = y$, then the intermediate value theorem gives a point $u \in (y, v]$ with $f(u) = z$. Then Lemma 2.5 applied to f^2 and $[u, v]$ contradicts our choice of y. □

[1]Note that some authors [1, 5] define this order in the opposite direction from this, however we follow Sharkovsky's original [9].

Observe that we have shown (a) also: the construction of z only required $m \geq 2$.

The proofs of (c) and (d) are more complex however, requiring more careful inductive arguments. As somewhat indicated by this proof, finding a point of given period is usually fairly straightforward, but we must take extra care to show it has the desired least period. Indeed, in the proof of (b), if we were only required to show that 2 is a period for f, any choice of y would have worked, and the last paragraph would be unnecessary.

To illustrate the ideas behind the proof of (e), first observe that the four given results already give a number of special cases of the forcing theorem. For instance, if 3 is a least period for f then induction with (c) shows every larger odd is a least period for f, and (a), (b) together cover 1. However, if 6 is a least period for f, then a priori we may only deduce that 2 and 1 are least periods.

Nonetheless, by instead applying (a)-(d) to iterates of f we may indeed complete the theorem - we show here a special case, leaving a detailed presentation of the remaining cases to [4].

Proposition 2.6. *Assuming (c) and (d), if* 6 *is a least period for* f*, then* 10 *is a least period for* f.

Proof. If 6 is a least period for f, then 3 is a least period for f^2. Then by (c), 5 is a least period for f^2.

Now either 5 is a least period for f, or 10 is a least period for f; but in the former case we may use (d) to deduce that 10 is a least period for f. □

Putting the five parts together, we thus obtain a proof of Sharkovsky's forcing theorem.

2.2 Proving the Realisation Theorem

While we remarked earlier that 2.4 fails for compact intervals, a minor modification does still hold.

Theorem 2.7 (Compact Sharkovsky realisation theorem)**.** *For a compact interval $I \subseteq \mathbb{R}$, every nonempty upwards closed set of the Sharkovsky order is the set of least periods of some continuous function $f : I \to I$.*

It is easy to see that this theorem implies our original version by simply noting that the map $f(x) = x + 1$ has no periodic points, and we may trivially extend our function on I to all of $\mathbb{R}$ continuously. Thus, we content ourselves with sketching the proof of the compact version, additionally assuming that $I = [0, 1]$ for simplicity of presentation.

Observe first that there are two types of nonempty upward closed set of the Sharkovsky order: those of the form $\{n : m \preceq n\}$ for a given m, and the powers of two.

Definition 2.8 (Truncated tent map [1])**.** For $h \in [0, 1]$, the truncated tent map $T_h : [0, 1] \to [0, 1]$ is given by

$$T_h(x) = \min\left(h, 1 - |2x - 1|\right).$$

A diagram of an example T_h is given in 1, together with the 'untruncated' version T_1.

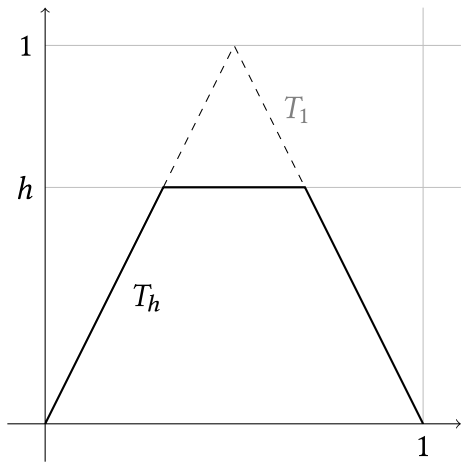

Figure 1. An illustration of T_h

We make a few key remarks about the truncated tent maps[1].

(1) T_1 has a 3-cycle $\{2/7, 4/7, 6/7\}$, and hence by the forcing theorem has all positive integers as periods.
(2) T_1 has at most 2^n points of period n, hence has finitely many points of least period n.
(3) If $h \leq k$, any cycle $O \subseteq [0, h)$ of T_h is a cycle for T_k, and any cycle $O \subseteq [0, h]$ of T_k is a cycle for T_h.

Then we define

$$h(m) := \min\{\max O : O \text{ is an } m\text{-cycle of } T_1\},$$

which is well-defined by (1) and (2), and may show from (3) that the least periods of $T_{h(m)}$ are exactly $n : m \preceq n$, as required. Setting $h^\infty := \sup_k h(2^k)$, the least periods of T_{h^∞} are the powers of two, completing the sketch.

It is important to notice here that while (1) is relatively clear, (2) is less immediate formally: indeed this is proved by Burns-Hasselblatt[1] by 'inspection of the graph'. Similarly (3) is not obvious, but a little work reveals the idea, and the remainder of the proof requires some effort; we omit it here for brevity.

3 Prerequisites to the Proofs

We begin by describing the basic data types, definitions and lemmas used for the formalisation, then move on to talk about the implementation of the proofs given in Section 2.

While mathlib does not have a large development of dynamical systems, it does have existing definitions of periodic points and minimal periods: the predicate `is_periodic_pt f n x` asserts that $f^n(x) = x$. However Lean's definition of the natural numbers includes zero, which is somewhat problematic in our context as all of our results are concerned with positive naturals. While we can solve this by using

Lean's pnat, defined as the positive naturals, it is much more convenient instead to use the type of naturals and add the non-zero condition where appropriate. Similarly, mathlib already has notions of intervals, real numbers, continuous functions and upward-closed sets.

However instead of continuous functions $f : I \to I$, we may equivalently talk about functions $f : \mathbb{R} \to \mathbb{R}$ which are continuous on I and have $f(I) \subseteq I$, and we take this approach to more closely align with mathlib. Thus, unless otherwise specified, f will be a function on the real numbers, with no additional assumptions for the moment.

To state Sharkovsky's theorem it therefore only remains to define the Sharkovsky ordering as in Definition 2.2. Noting the key role that odds and powers of two have in the ordering, we first define the auxiliary Sharkovsky ordering on $\mathbb{N} \times \mathbb{N}$, where the intended meaning is that (k, m) is less than (l, n) in this ordering iff $2^k m \leq 2^l n$ for odd m, n. Note this is not the only possible way of defining the Sharkovsky ordering, but it is chosen to reflect the important nature of powers of two.

We call this ordering SOA (for Sharkovsky ordering auxiliary), allowing the somewhat obscure name since it is not intended for use by any end users: we will define the actual Sharkovsky ordering by decomposing into the 'odd' and 'power of two' part, and plugging this into SOA.

```
def to_pair (n : ℕ) : ℕ × ℕ := ...
def un_pair (i : ℕ × ℕ) : ℕ := 2 ^ i.1 * i.2
lemma un_pair_to_pair (n : ℕ) :
  un_pair (to_pair n) = n := ...

inductive SOA : ℕ × ℕ → ℕ × ℕ → Prop
| zero_le {k : ℕ} {l : ℕ × ℕ} : SOA (k, 0) l
| powers {k l : ℕ} : l ≤ k → SOA (k, 1) (l, 1)
| to_pow_two {k l m : ℕ} :
    SOA (k, m + 2) (l, 1)
| same_level {k m n : ℕ} : m ≤ n →
    SOA (k, m + 2) (k, n + 2)
| next_level {k l m n : ℕ} : k < l →
    SOA (k, m + 2) (l, n + 2)
```

The function to_pair decomposes, and un_pair reconstructs, a natural number, with un_pair_to_pair showing we do indeed have a reconstruction, and thus define the Sharkovsky ordering as SOA (to_pair n) (to_pair m). With this in mind, the constructors of SOA clearly match the Sharkovsky ordering - save for the first - powers says powers of two are in decreasing order, to_pow_two says every non power of two is less than any power of two, etc. However the first constructor zero_le deserves special attention, recall that Lean's natural numbers include 0, so we have 0 in the ordering.

There are a number of possible natural solutions here: we can make 0 incomparable with everything, make it less than everything, make it more than everything, or disallow it from the type entirely. We choose to make 0 less than everything else in our order: disallowing it from the type presents similar issues to pnat as discussed previously, and any other solution introduces new upward closed sets in the Sharkovsky order, so this is the most faithful approach.

Notice also that the constructor to_pow_two asks for m + 2, rather than m with the hypothesis that $2 \leq m$:

```
| to_pow_two {k l m : ℕ} : 2 ≤ m →
    SOA (k, m) (l, 1)
```

This version would be logically equivalent, however our original version has the advantage of allowing Lean's equation compiler to automatically discard impossible cases. For instance to prove transitivity of the SOA relation, a naive approach would require 5×5 cases as we have 5 constructors, but our proof only uses 9 cases, vastly simplifying the simple order results.

Finally we define the type sharkovsky as a type alias for the natural numbers, providing explicit functions to convert between $\mathbb{N}$ and sharkovsky, and place our linear order on the new type. Then we may define notation $\preceq$ and $\prec$ on naturals to recover our familiar expressions.

```
def sharkovsky := ℕ

def of_nat (n : ℕ) : sharkovsky := n
def to_nat (n : sharkovsky) : ℕ := n

lemma to_nat_of_nat (n : ℕ) :
  to_nat (of_nat n) = n := rfl
lemma of_nat_to_nat (n : sharkovsky) :
  of_nat (to_nat n) = n := rfl

instance : linear_order sharkovsky :=
{ le := λ n m,
    SOA (to_pair n.to_nat) (to_pair m.to_nat),
  ... /- omit proofs of linearity here -/ }

notation n ` ≼ ` m := of_nat n ≤ of_nat m
notation n ` ≺ ` m := of_nat n < of_nat m
```

We additionally add notation $\bot$ and $\top$ for the bottom (0) and top (1) of the ordering using mathlib's order hierarchy, and may now show some expected specific properties of the order.

```
lemma three_le_of_ne_zero (hn : n ≠ 0) :
  3 ≼ n := ...

lemma two_pow_le_two_pow_iff {k l : ℕ} :
  2 ^ l ≼ 2 ^ k ↔ k ≤ l := ...

lemma doubling_iff {n m : ℕ} :
  2 * n ≼ 2 * m ↔ n ≼ m := ...
```

To further confirm our order is as expected, we set up lemmas for the simplifier to prove or refute $n \preceq m$ for explicit numerals, and can thus verify the inequalities indicated in

Definition 2.2. The mathlib definition list.chain′ asserts that the given relation holds between consecutive elements of the list, so the following is equivalent to $3 \preceq 5 \preceq 7 \preceq \cdots \preceq 2 \preceq 1$.

```
example : list.chain′ (≼)
  [3, 5, 7, 9, 6, 10, 14, 12, 20, 28, 16, 8, 4,
    2, 1] :=
by simp
```

Before terminating this section, we discuss the extra prerequisite lemmas which were developed for the proof, rather than just the statement. While mathlib has versions of the intermediate value theorem, Lemma 2.5 will be used extensively, so we prove a version of it, recalling that Icc a₁ a₂ refers to the closed interval $[a_1, a_2]$.

```
lemma exists_fixed_point_Icc {a₁ a₂ : ℝ}
  (h : a₁ ≤ a₂)
  (hf : continuous_on f (Icc a₁ a₂))
  (ha : a₁ ≤ f a₁ ∧ f a₂ ≤ a₂ ∨
        f a₁ ≤ a₁ ∧ a₂ ≤ f a₂) :
  ∃ c ∈ Icc a₁ a₂, is_fixed_pt f c := ...
```

This proof is remarkably short, taking only 5 lines of Lean, and its statement is readily seen to imply 2.5. Indeed it is this form that we will find most helpful, as showing $f(J) \supseteq J$ typically involves reasoning about endpoints, so we simply have the lemma stated in terms of endpoints. Intuitively, the condition ha says that if the endpoints of a closed interval 'go in different directions', then the interval contains a fixed point.

In the proof sketches we have given for both the forcing and realisation theorems, the notion of orbit has been helpful, so we define a version of this as well. Namely, it will be useful to convert between points of minimal period n and (finite) cycles of size n.

```
def finset.is_cycle {α : Type*} (P : finset α)
    (f : α → α) : Prop :=
∀ x ∈ P,
  f x ∈ P ∧ minimal_period f x = P.card

lemma exists_cycle_of_minimal_period {n : ℕ}
  {x : ℝ} (hf : maps_to f I I) (hx : x ∈ I)
  (hn : n ≠ 0) (hx′ : minimal_period f x = n) :
  ∃ (P : finset ℝ),
    P.is_cycle f ∧ x ∈ P ∧
    P.card = n ∧ P ⊆ I := ...
```

Thus we say a finite set P is a cycle for f if every point of P remains in P, and has minimal period the size of P. This implies that the orbit of any point in P is P itself. Given a point x of minimal period n with $x \in I$ and $f(I) \subseteq I$, there is a cycle P which contains x, is contained in I, and has cardinality n. The opposite direction is immediate: simply take any point of our cycle and the second part of the condition shows it has the correct minimal period. Thus we may smoothly switch between the two representations.

4 Formalising the Forcing Theorem

We are now in good shape to prove the five parts of the forcing theorem. An unfortunate complication here is that the informal proof for each of (b)-(d) starts in the same way, defining y as shown in our proof sketch for (b), but then each part diverges and finishes the proof in a different way. It is undesirable to have a single long proof for all four cases, and similarly undesirable to repeat the same proof steps, so we instead extract this 'core' into a separate lemma, and use this multiple times. This has the downside of producing a lemma with a particularly cryptic statement, which we will not reproduce here, but the cost seems to be worth it.

Instead, we provide the statements of (a)-(d): it is most convenient for these proofs to have assumptions in 'cycle' form, and consequences in 'minimal period' form.

```
variables [ord_connected I]
  (hf : continuous_on f I) (hf′ : maps_to f I I)
  (hP : P.is_cycle f) (hPI : P ⊆ I)

lemma part_a (hP₂ : 2 ≤ P.card) :
  ∃ c ∈ I, is_fixed_pt f c := ...

lemma part_b (hP₂ : 3 ≤ P.card) :
  ∃ c ∈ I, minimal_period f c = 2 := ...

lemma part_c (hP₂ : 3 ≤ P.card)
  (hP₃ : odd P.card) :
∃ c ∈ I, minimal_period f c = P.card + 2 := ...

lemma part_d (hP₂ : 3 ≤ P.card)
  (hP₃ : odd P.card) :
  ∀ i : ℕ, ∃ c ∈ I,
    minimal_period f c = 2 * (i + 1) := ...
```

The Lean keyword variables asserts assumptions shared across the following statements. Recall that maps_to f I I is the claim that $f(I) \subseteq I$. In addition, ord_connected I says that if $x, y \in I$, then $[x, y] \subseteq I$, i.e. in our context that I is an interval, so the first three statements are the translations of the claims of 4, and the fourth is an immediate generalisation. As noted in by Du[4], we actually prove the more general version despite only needing two of its special cases: it is ultimately a proof by induction, so showing the special cases is no easier than the general version.

Nonetheless, (d) is certainly the most involved of the four proofs. As may be evident from the sketch in 4, (a) is very easy while (b) requires some more work, but can be accomplished in tandem. These two take up around 120 lines of code, including the shared 'core'. The proof of part (c) is around 90 lines, requiring some parity calculations which are drastically simplified by the extension parity_simps[3].

Meanwhile, part (d) uses 250 lines: more than the previous three combined. This is not too surprising when comparing to the proofs on paper however, especially considering that the proof of (d) uses a subtle double induction, where the induction hypothesis is not explicitly written down.

To complete we show (e), i.e. that $n \leq m$ with $n \neq 0$ and a point of minimal period n implies the existence of minimal period m. We may do this by cases on $n \preceq m$, working through each of the five constructors in turn (the first is trivial since we have the non-degeneracy condition $n \neq 0$).

The proofs for most of these follow a very similar template to the outline given for Proposition 2.6 however there is an extra lemma required. Specifically, mathlib already has a lemma for least periods of iterated functions: if x is periodic for f with least period m, then it is periodic for f^n with least period $m/\gcd(m, n)$. However we also need to move in the opposite direction.

Lemma 4.1. *If x is periodic for f^n with least period k, then it is periodic for f with least period kn/s for some s which divides n and is coprime to k.*

The proof is not particularly complex, but contained a surprisingly tricky part, encapsulated by the following example.

```
example {n s k : ℕ} (hn : n ≠ 0) (hk : k ≠ 0)
  (hs : s ≠ 0) (hskn : s ∣ k * n)
  (h : (k * n / s) / (k * n / s).gcd n = k) :
  s ∣ n ∧ coprime s k := ...
```

Despite being purely about natural numbers and only elementary operations on them, this proof takes 14 lines and very little support from automation, meaning a nontrivial amount of human-effort. While this may appear to be only a minor annoyance, this was the most significant detour in the formalisation, and is especially notable since it appears trivial on paper.

Once this roadblock has been overcome, however, we can smoothly complete our proof. The combination of our inductive definition of the ordering, and the splitting of the cases (a)-(d) makes the overall argument extremely clean to formalise.

```
theorem sharkovsky_forcing {I : set ℝ}
  [ord_connected I] (hf : continuous_on f I)
  (hf' : maps_to f I I) {n m : ℕ} (h : n ≼ m)
  (hn : n ≠ 0)
  (hx : ∃ x ∈ I, minimal_period f x = n) :
  ∃ x ∈ I, minimal_period f x = m := ...
```

5 Formalising the Realisation Theorem

As indicated in Section 2, the main case of interest is Theorem 2.7 for compact intervals, and for these it suffices to work with the unit interval. We formalise Theorem 2.7 for the unit interval only, as well as Theorem 2.4 for completeness. We see no particular obstruction to generalising to any compact interval (indeed these are all homeomorphic to the unit interval), but do not discuss these here for the sake of simplicity. Similarly, while 2.4 should in fact hold for any non-compact interval, we do not discuss it here, preferring to exposit the cleaner and more familiar versions of these results.

Defining the truncated tent map and showing some elementary properties is very straightforward: each of the omitted proofs is just one line, and the tactic `continuity` makes the proof of continuity merely two words! From here on, we refer to this as simply the tent map for brevity.

```
noncomputable def tent_map (h x : ℝ) : ℝ :=
    min h (1 - |2 * x - 1|)

lemma tent_map_continuous :
  continuous (tent_map h) := by continuity

lemma tent_map_one_eq :
  tent_map 1 x = 1 - |2 * x - 1| := ...
lemma tent_map_one_of_le_half (hx : x ≤ 1 / 2):
  tent_map 1 x = 2 * x := ...
lemma tent_map_one_of_half_le (hx : 1 / 2 ≤ x):
  tent_map 1 x = 2 - 2 * x := ...
lemma tent_map_flip :
  tent_map h (1 - x) = tent_map h x := ...
lemma tent_map_right_zero (h₀ : 0 ≤ h) :
  tent_map h 0 = 0 := ...
lemma tent_map_right_one (h₀ : 0 ≤ h) :
  tent_map h 1 = 0 := ...
lemma minimal_period_zero_tent_map (h₀ : 0 ≤ h):
  minimal_period (tent_map h) 0 = 1 := ...
```

Notice that we in fact showed that the tent map is continuous everywhere, not just when restricted to $[0, 1]$, but this is clearly a stronger statement. Further, the definition of the tent map does not require that $h \in [0, 1]$, though some lemmas do.

Claim (1) from 2 about the special tent map T_1 is not too difficult to show: it is enough to show that $2/7$ has minimal period 3, equivalently that it has period 3 and does not have period 1, and mathlib's numeral normaliser `norm_num` can evaluate $T_1(2/7)$ fairly smoothly. Then our statement of the forcing theorem together with `three_le_of_ne_zero` shown in 3 allows us to show that T_1 has a point of every positive minimal period.

```
lemma tent_map_one_three :
  minimal_period (tent_map 1) (2 / 7) = 3 := ...
lemma tent_map_one_minimal_period
  {n : ℕ} (hn : n ≠ 0) :
  ∃ x ∈ Icc (0 : ℝ) 1,
  minimal_period (tent_map 1) x = n := ...
```

In contrast, claim (2) is more difficult: we must show that T_1^n has at most 2^n fixed points. The informal proof idea here is ultimately to inspect the graph as in 2, and observe that it has exactly 2^n intersections with $f(x) = x$.

A formal version of this argument clearly requires some more work. Our approach was to split the range $[0, 1]$ into 2^n regions, such that T_1^n is linear on each, and give an explicit

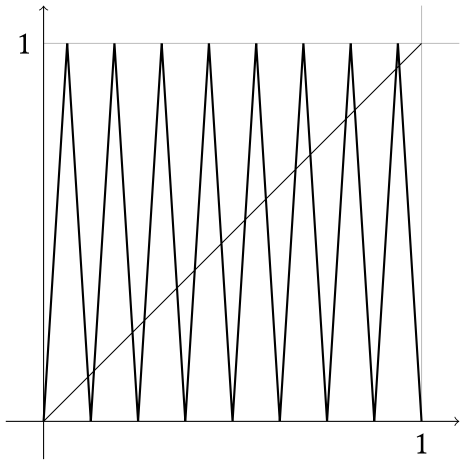

Figure 2. An illustration of T_1^4 and the location of its fixed points

expression for T_1^n on each region. Then, we can explicitly enumerate the fixed points of T_1^n, since the existence of a fixed point in such a region is equivalent to a (rational) linear equation, and the regions are a disjoint cover of $[0, 1]$.

These proofs are somewhat tedious, needing many floor and parity calculations, though as a consequence we in fact show a stronger statement than was required: that there are exactly 2^n points of period n for T_1, and furthermore that every periodic point of T_1 is rational. This latter statement isn't directly useful for our purposes, but it is a nice property of the tent map nonetheless (indeed, one can further show that every rational is pre-periodic, meaning after finitely many iterations it hits a periodic point).

We can express the set of periodic points explicitly as a consequence

```
def tent_map_periodic_pts_rat (n : ℕ) :
  finset ℚ :=
(finset.range (2 ^ n)).image
  (λ k, if even k
          then k / (2 ^ n - 1)
          else (k + 1) / (2 ^ n + 1))
```

However it is more useful to view this as a finite set of reals, and we can then show its key properties.

```
def tent_map_periodic_pts (n : ℕ) :
  finset ℝ := ...

lemma tent_map_periodic_pts_subset :
  tent_map_periodic_pts n ⊆ Icc (0 : ℝ) 1 := ...

lemma tent_map_periodic_pt_iff (hn : n ≠ 0) {x :
    ℝ}
  (hx : x ∈ Icc (0 : ℝ) 1) :
  is_periodic_pt (tent_map 1) n x ↔
    x ∈ tent_map_periodic_pts n := ...

lemma tent_map_periodic_pts_card (hn : n ≠ 0) :
  (tent_map_periodic_pts n).card = 2 ^ n := ...

lemma finitely_many_periodic_pts (hn : n ≠ 0) :
  set.finite {x ∈ Icc (0 : ℝ) 1 |
    is_periodic_pt (tent_map 1) n x} :=
```

The condition $n \neq 0$ in the last three lemmas is unfortunately necessary, but entirely unproblematic as the only 'mathematical' cases we are interested in are for positive integers, and the final lemma is the result we were looking for. This section of the formalisation (i.e. showing (2)) took approximately 260 lines, and was arguably the part with the most originality required, since our references didn't really suggest a formal proof.

In stark contrast, our proof of (3) follows [1] almost exactly, taking barely 30 lines for both parts.

```
variables {O : finset ℝ} (hk : h ≤ k)
lemma cycle_up (hO : O.is_cycle (tent_map h))
  (hO' : O ⊆ Ico 0 h) :
  O.is_cycle (tent_map k) := ...

lemma cycle_down (hO : O.is_cycle (tent_map k))
  (hO' : O ⊆ Icc 0 h) :
  O.is_cycle (tent_map h) := ...
```

Our definition of the set of periodic points as a finite set allows an almost automatic proof that h is well-defined, and hence putting the results together, we arrive at our final three theorems.

```
lemma sharkovsky_lt_iff {n m : ℕ} (hn : n ≠ 0)
    (hm : m ≠ 0) : n ≺ m ↔
  ∃ f : ℝ → ℝ,
  continuous_on f (Icc 0 1) ∧
  maps_to f (Icc 0 1) (Icc 0 1) ∧
  (∃ x ∈ Icc 0 1, minimal_period f x = m) ∧
  ∀ x ∈ Icc 0 1, minimal_period f x ≠ n
:= ...

variables {s : set sharkovsky} (hs : ⊥ ∉ s)
theorem sharkovsky_unit_interval  :
  (∃ f : ℝ → ℝ, continuous_on f (Icc 0 1) ∧
    maps_to f (Icc 0 1) (Icc 0 1)
    ∧ s = sharkovsky.of_nat ''
    minimal_periods_on f (Icc 0 1)) ↔
  is_upper_set s ∧ s.nonempty
:= ...

theorem sharkovsky_real_line :
  (∃ f : ℝ → ℝ, continuous f ∧
    s = sharkovsky.of_nat '' minimal_periods_on
    f set.univ) ↔
  is_upper_set s := ...
```

The first theorem `sharkovsky_lt_iff` shows the fundamental property of the Sharkovsky relation, asserting precisely which minimal periods may coexist for maps on the unit interval. Our second and third theorems are stated in terms of sets of Sharkovsky points, so that the ordering inferred by `is_upper_set` is the correct one, and we use `sharkovsky.of_nat ''` to convert from naturals. Finally, `minimal_periods_on` gives the set of minimal periods for any point in the given set, hence these two theorems state precisely what was claimed.

6 Conclusion

We have demonstrated the formalisation of a piece of mathematics which has not been verified previously, illustrating the beginnings of chaos theory and simple dynamical systems in Lean. We hope to have shown the sources of difficulties in formalisation, as well as particularly elegant sub-problems through a single theorem.

Our formalisation uses mathlib heavily, in particular drawing on its basic theory of dynamics, as well as finite set combinatorics[2, 8] and real analysis[7]. In addition, the automation provided for topology and simplification has been invaluable, while lack of automation for natural numbers has been both notable and painful.

We hope that future work can develop on these ideas further. Of particular interest is the remainder of Li and Yorke's paper[6], showing the existence of chaotic points from period 3, not just points of arbitrary period, and generalisations of Sharkovsky's theorem to other spaces.

References

[1] Keith Burns and Boris Hasselblatt. 2011. The Sharkovsky theorem: A natural direct proof. *The American Mathematical Monthly* 118, 3 (2011), 229–244.

[2] Yaël Dillies and Bhavik Mehta. 2022. Formalising Szemerédi's Regularity Lemma in Lean. In *13th International Conference on Interactive Theorem Proving (ITP 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[3] Floris van Doorn, Gabriel Ebner, and Robert Y Lewis. 2020. Maintaining a library of formal mathematics. In *International Conference on Intelligent Computer Mathematics*. Springer, 251–267.

[4] Bau-Sen Du. 2007. A Simple Proof of Sharkovsky's Theorem Rerevisited. *arXiv preprint arXiv:0711.3892* (2007).

[5] Bau-Sen Du. 2007. A simple proof of Sharkovsky's theorem revisited. *The American Mathematical Monthly* 114, 2 (2007), 152–155.

[6] Tien-Yien Li and James A Yorke. 1975. Period Three Implies Chaos. *The American Mathematical Monthly* 82, 10 (1975), 985–992.

[7] The mathlib Community. 2020. The Lean Mathematical Library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs* (New Orleans, LA, USA) *(CPP 2020)*. Association for Computing Machinery, New York, NY, USA, 367–381.

[8] Bhavik Mehta. 2022. Formalising the Kruskal-Katona Theorem in Lean. In *International Conference on Intelligent Computer Mathematics*. Springer, 75–91.

[9] A Sarkovskii. 1964. Coexistence of cycles of a continuous map of a line to itself. *Ukr. Mat. Z.* 16 (1964), 61–71.

Received 2022-09-21; accepted 2022-11-21