BOUNDED ARITHMETIC

Induction, Recursion, and the Self-Grounding of Finite Mathematics

Working Paper — 2026

Abstract

This paper constructs bounded arithmetic from Bounded First-Order Logic (BFOL) and Bounded Set Theory (BST). Starting from the standard models and the Bounded Fundamental Theorems, the paper derives bounded induction (BI-BST) and bounded recursion (BR-BST) as the proof engine and computational engine of finite mathematics, defines the arithmetic operations (addition, multiplication, exponentiation) by bounded recursion, proves the complete set of algebraic identities (commutativity, associativity, distributivity) by bounded induction, and develops elementary number theory (divisibility, primality, GCD, Bezout's identity, unique factorisation) within the bounded naturals. Every result is Type I (exact).

The paper then addresses the self-grounding question: whether the bounded arithmetic constructed here can evaluate any BFOL sentence in any standard model by direct computation. The answer is yes (Theorem 8.1). The arithmetic evaluates truth by recursion on formula structure, with every quantifier bounded and every domain finite. The loop closes: the logic defines the set theory, the set theory constructs the arithmetic, and the arithmetic evaluates the logic. No external system is consulted.

1. Introduction

1.1 The engine of everything

Every construction in the bounded framework traces back to two tools: bounded induction (the proof engine) and bounded recursion (the computational engine). Every algebraic identity, every recursive definition, every algorithm in every companion paper is an application of BI-BST or BR-BST. This paper constructs those tools and the arithmetic they produce.

The paper also addresses the deepest question in the framework: can the arithmetic ground itself? If the bounded arithmetic constructed here can evaluate any BFOL sentence in any standard model by direct computation, then the framework is self-supporting. The logic defines the set theory, the set theory constructs the arithmetic, and the arithmetic evaluates the logic. No external system is consulted. The loop closes.

1.2 Prerequisites from BST

The following is assumed from Bounded Set Theory and is summarised here so that the reader can follow the constructions without consulting the predecessor paper.

The Axiom of Finite Bounds (AFB). AFB negates infinity and asserts a finite upper bound. The standard models are the finite levels $\mathcal{V}$_n = V_n of the cumulative hierarchy. BST's theorems are exactly the sentences true in every standard model $\mathcal{V}$_n.

The standard models. V_0 = {∅}, V_n = P(V_{n-1}). Every subset of V_{n-1} is an element of V_n.

The interior/ceiling partition. An element x ∈ V_n is interior if x ∈ V_{n-1} (equivalently, x is a member of something in V_n). Ceiling elements are in V_n \{} V_{n-1} and are constructively inert. Interior(V_n) = V_{n-1}.

The BFTs used in this paper:

BFT	Name	Content	Used for
1	Foundation	Every nonempty interior set has an ∈-minimal element.	Theorem 4.13 (well-ordering)
4	Pairing	If a, b are interior, {a, b} exists.	Definition 3.2 (ordered pairs, requires V_{n-2}), successor, Theorem 3.4 (BR-BST derivation)
5	Union	If F is interior, ⋃F exists.	Successor, Theorem 3.4 (BR-BST derivation)
7	Separation	If A is interior, {x ∈ A : φ(x)} exists for any BFOL formula φ.	Definition 6.1 (divisibility), Remark 6.9 (sieve of Eratosthenes)

The ordinal theory. Bounded ordinals: $0 = \emptyset$, $1 = \{\emptyset\}$, $2 = \{\emptyset, \{\emptyset\}\}$, ..., up to the bound. No limit ordinals exist: every nonzero interior ordinal is a successor. Successor $S(\alpha) = \alpha \cup \{\alpha\}$ requires α to be interior (BFT 4 and BFT 5).

BFOL. The logical substrate. Every quantifier carries an explicit bounding term. The only quantifier forms are $\forall x \leq t\ \varphi(x)$ and $\exists x \leq t\ \varphi(x)$.

1.3 Conventions and notation

The parameter k is the cardinality bound (n_M in BST notation). $\mathbb{N}$_B(k) = {0, 1, ..., k}. Successor: $S(\alpha) = \alpha \cup \{\alpha\}$. The interiority convention: all quantification over interior elements unless stated otherwise. When a theorem states "for all m, n" the quantification ranges over interior elements of $\mathbb{N}$_B(k).

This paper works entirely within $\mathbb{N}$_B(k). No integers, rationals, reals, or complex numbers appear. The extension of arithmetic to those systems belongs to subsequent papers.

BFOL formulas: φ, ψ, etc. Function symbols: f, g, h. The syntactic approach to functions (BFOL formulas satisfying totality and uniqueness) is used throughout; function graphs as sets (FA-BST) are noted where relevant but not required for the arithmetic.

1.4 Plan of the paper

Section 2 develops bounded induction. Section 3 derives bounded recursion from bounded induction. Section 4 defines the arithmetic operations, the order structure, and the interiority condition. Section 5 proves the algebraic identities by bounded induction. Section 6 develops elementary number theory. Section 7 analyses the scope and limits of bounded arithmetic. Section 8 addresses the self-grounding question. Section 9 concludes.

2. Bounded Induction

2.1 Why unbounded induction fails in BST

Proposition 2.1 (unbounded induction fails in BST):

```
The unbounded induction conclusion ∀n φ(n) is false
in every standard model of BST for the property
φ(n) = "there exists a set of cardinality n."

Proof:
  Let φ(n) ≡ ∃S ∈ V_n (|S| = n).

  Base: φ(0) holds. ∅ has cardinality 0, and ∅ ∈ V_n
    for all n ≥ 1.

  Step: if S is an interior set of cardinality c, then
    S ∪ {x} for any interior x ∉ S has cardinality c+1.
    {x} exists by BFT 4 (x is interior).
    S ∪ {x} = ⋃{S, {x}} exists by BFT 5
    (both S and {x} are interior, so {S, {x}} exists
    by BFT 4, and ⋃{S, {x}} exists by BFT 5).
    So φ(c) → φ(c+1) holds whenever a set of
    cardinality c exists as an interior element.

  The failure: every set in V_n is a subset of
    V_{n-1} (since V_n = P(V_{n-1})). The largest
    set in V_n is V_{n-1} itself, with cardinality
    |V_{n-1}|. No set in V_n has cardinality
    |V_{n-1}| + 1, because no subset of V_{n-1}
    has more than |V_{n-1}| elements.

  So φ(|V_{n-1}| + 1) is false. The unbounded
  conclusion ∀n φ(n) is refuted by the model.
```

The failure is structural: the BFT preconditions enforce a ceiling on inductive constructions, and this is the mechanism by which the Axiom of Finite Bounds operates. Bounded induction resolves this by restricting the conclusion to $\forall\alpha \leq$ k $\varphi(\alpha)$ for an explicit interior k.

2.2 The bounded induction schema (BI-BST)

Schema 2.2 (BI-BST, full form):

```
For any BFOL property φ, and any interior bound k:

φ(0) ∧ ∀α < k (Interior(α) ∧ φ(α) → φ(S(α)))
  → ∀α ≤ k φ(α)
```

The explicit Interior(α) in the hypothesis matches BST's formulation (BST Section 5.3). The following lemma shows it is automatically satisfied, so in practice it can be dropped.

Lemma 2.3 (interior inheritance):

```
For α < k with k interior, α is interior.

Proof:
  α < k means α ∈ k.                (bounded ordinal membership)
  k interior means k ∈ V_{n-1}.
  V_n is transitive: if x ∈ y ∈ V_n
    then x ∈ V_n.
    Proof of transitivity: V_n = P(V_{n-1}).
    y ∈ V_n means y ⊆ V_{n-1}.
    x ∈ y and y ⊆ V_{n-1} gives x ∈ V_{n-1}.
    x ∈ V_{n-1} ⊆ V_n, so x ∈ V_n.
  Applying transitivity: α ∈ k and k ∈ V_{n-1}
    gives α ∈ V_{n-1}.
  So α is interior.
```

Consequence: the schema simplifies to

Schema 2.4 (BI-BST, simplified form):

```
φ(0) ∧ ∀α < k (φ(α) → φ(S(α))) → ∀α ≤ k φ(α)
```

with no explicit interiority check at each step. The interiority is structurally guaranteed. This is what makes BI-BST usable: one does not verify interiority at each step of an inductive proof. One verifies that k is interior (once), and interior inheritance does the rest.

Remark 2.5 (BI-BST is finite iteration):

```
In any model V_n, the ordinals are {0, 1, ..., n_M}
for some finite n_M. Any k ≤ n_M is a specific finite
number. If φ(0) holds and the step φ(α) → φ(S(α))
holds for all α < k, then φ holds at 0, then at 1,
then at 2, ..., terminating after exactly k steps.

BI-BST is finite iteration: apply a step function k
times, starting from a base case.
```

BI-BST is not an axiom. It is a theorem schema of BST, valid in every standard model. The validity rests on two facts: the domain is finite, and interior inheritance (Lemma 2.3) guarantees that every step in the iteration operates on interior elements.

2.3 Strong induction (SBI-BST)

Schema 2.6 (SBI-BST):

```
∀α ≤ k (∀β < α φ(β) → φ(α)) → ∀α ≤ k φ(α)

Derivation from BI-BST:
  Define ψ(α) ≡ ∀β ≤ α φ(β).
  The hypothesis ∀β < α φ(β) → φ(α)
  gives the step ψ(α) → ψ(S(α)).
  BI-BST yields ψ(k), which implies
  φ(α) for all α ≤ k.
```

SBI-BST is used for proofs where the induction hypothesis needs ”all predecessors” rather than just ”the immediate predecessor.” It is derivable from BI-BST, not an independent schema.

2.4 The ceiling boundary

BI-BST proofs are valid in every model simultaneously. If BI-BST establishes $\varphi(\alpha)$ for all $\alpha \leq$ k for some explicit k, then in any model where k is interior (n_M > k), the conclusion holds. The proof does not need to know the model’s ceiling.

At the ceiling itself, S(k) = k $\cup$ {k} requires k to be interior (BFT 4). Ceiling elements are not interior. The successor construction is blocked. This is the structural reason induction stops: not because the property fails, but because the construction cannot be performed. Every inductive proof in BST has a definite stopping point, determined by the bound.

3. Bounded Recursion

3.1 The schema (BR-BST)

Schema 3.1 (BR-BST):

```
For any BFOL term g and any BFOL formula h defining
a function from pairs to single values, and any
bound k, there exists a unique function f with
domain {0, ..., k} such that:

f(0) = g
f(S(α)) = h(α, f(α))   for all α < k

The function exists as a computation (Remark 3.7).
When the function graph is needed as a set of
ordered pairs, Kuratowski encoding (Definition 3.2)
requires all ordinals and values to be in V_{n-2}.
```

The interiority condition on α is satisfied by interior inheritance (Lemma 2.3). The domain of f is bounded by k. BST cannot assert a recursively defined function on all natural numbers, only on ordinals up to a specific bound.

3.2 Ordered pairs

The derivation of BR-BST constructs function graphs as sets. This requires ordered pairs.

Definition 3.2 (Kuratowski encoding):

```
(a, b) := {{a}, {a, b}}

Construction requires three applications of BFT 4:
  {a} exists by BFT 4                  (a interior: a ∈ V_{n-1})
  {a, b} exists by BFT 4               (a, b interior: a, b ∈ V_{n-1})
  {{a}, {a, b}} exists by BFT 4        (requires {a} and {a, b}
                                         to be interior: {a}, {a, b} ∈ V_{n-1})

The third step requires {a} ∈ V_{n-1} and {a, b} ∈ V_{n-1}.
{a} ⊆ V_{n-1} gives {a} ∈ V_n, but {a} ∈ V_{n-1} requires
{a} ⊆ V_{n-2}, i.e., a ∈ V_{n-2}.
Similarly {a, b} ∈ V_{n-1} requires a, b ∈ V_{n-2}.

So Kuratowski pairs are constructible when a, b ∈ V_{n-2}
(two levels below the ceiling). For the bounded ordinals
in ℕ_B(k), this constraint is satisfied in any model
V_n with n ≥ k + 3.
```

Lemma 3.3 (ordered pair correctness):

```
(a, b) = (c, d) iff a = c and b = d.

Proof:
  (⇐) If a = c and b = d then {{a}, {a, b}} = {{c}, {c, d}}
  by substitution.

  (⇒) Suppose {{a}, {a, b}} = {{c}, {c, d}}.

  Case 1: a = b.
    Then (a, b) = {{a}, {a, a}} = {{a}}.
    So {{a}} = {{c}, {c, d}}.
    This set has one element, so {c} = {c, d} = {a}.
    {c} = {a} gives c = a.
    {c, d} = {a} gives d = a.
    So a = c and b = a = d.

  Case 2: a ≠ b.
    Then (a, b) = {{a}, {a, b}} has two distinct
    elements ({a} ≠ {a, b} since a ≠ b).
    So {{a}, {a, b}} = {{c}, {c, d}} as two-element sets.
    {a} is the only singleton in {{a}, {a, b}}.
    {c} is the only singleton in {{c}, {c, d}}
    (if c = d then {c} = {c, d} and the set has one
    element, contradicting two elements; so c ≠ d).
    Therefore {a} = {c}, giving a = c.
    Then {a, b} = {c, d} = {a, d}, giving b = d.
```

This is the minimal ordered pair construction needed for BR-BST. The full development of Cartesian products, relations, and quotient sets belongs to BNT.

3.3 Derivation from BI-BST

BR-BST is not assumed independently but derived from BI-BST. This is the key derivation.

Theorem 3.4 (bounded recursion):

```
BR-BST is derivable from BI-BST.
```

```
Proof: induct on the property
P(m) = "there exists a unique function f_m
        with domain {0, ..., m} satisfying
        the recursion."

The function graph is a set of Kuratowski pairs.
By Definition 3.2, this requires all ordinals
and all values to be in V_{n-2} so that the
pairs land in V_n.

Base (m = 0):
  f_0 = {(0, g)}.
  The ordered pair (0, g) exists          (Definition 3.2:
                                           0, g ∈ V_{n-2})
  The singleton {(0, g)} exists           (BFT 4)
  f_0 is the unique function with
  domain {0} satisfying f_0(0) = g.
  P(0) holds.

Step: assume f_m exists uniquely (P(m) holds,
  m ∈ V_{n-2} since m < k and the model is
  large enough).
  Define v = h(m, f_m(m)).
  v ∈ V_{n-2}                             (h maps to V_{n-2})
  (S(m), v) exists                        (Definition 3.2)
  {(S(m), v)} exists                      (BFT 4)
  {f_m, {(S(m), v)}} exists               (BFT 4)
  f_{S(m)} = ⋃{f_m, {(S(m), v)}}
           = f_m ∪ {(S(m), v)}            (BFT 5)
  This extension is unique given f_m.
  P(S(m)) holds.

BI-BST gives P(m) for all m ≤ k.
Take f = f_k.
```

The BFTs used: BFT 4 (Pairing) for ordered pairs, BFT 5 (Union) for extending the function. These are the construction tools. BI-BST is the proof tool. Together they give BR-BST. The construction requires the model $\mathcal{V}$_n to be large enough that all ordinals up to k and all values

produced by g and h are in V_{n-2}.

3.4 Iterated and course-of-values recursion

Remark 3.5 (iterated recursion):

```
Define a sequence of functions f_1, f_2, ..., f_d
where each f_{i+1} is defined by BR-BST using f_i.

This is itself a BR-BST construction:
  Outer recursion on i with bound d.
  g = f_1 (the first function, defined by BR-BST).
  h(i, f_i) = the function f_{i+1} defined by
    applying BR-BST with step function using f_i.

At each outer step, the inner BR-BST defines a
complete function on {0, ..., k}. The outer
recursion produces d such functions in sequence.
```

Remark 3.6 (course-of-values recursion):

```
f(n) defined in terms of f(0), f(1), ..., f(n-1) (not just f(n-1)).

Derivable from BR-BST: define
F(n) = the sequence (f(0), ..., f(n)),
then f(n) is the last element of F(n).
The step function computes F(n+1) from F(n)
by appending one value.
```

Each variant is derived from basic BR-BST, not assumed independently.

3.5 The function existence guarantee

BR-BST guarantees that the recursively defined function exists as a specific finite object in the model. The function has domain {0, ..., k} and codomain contained in the model's domain. Every value f(n) is a definite interior element.

BR-BST does not merely assert existence. It provides a construction. The function is built step by step, and each step uses only BFTs applied to interior elements. The construction is effective: given specific g, h, and k, the function f can be computed by iterating the step function k times.

Remark 3.7 (BR-BST as algorithm):

```
Every BR-BST derivation is an algorithm. The schema
specifies the base value (g), the step function (h),
and the bound (k). The computation proceeds:
  f(0) = g
  f(1) = h(0, g)
  f(2) = h(1, h(0, g))
  ...
  f(k) = h(k-1, f(k-1))
Each step is a single application of h to interior
elements, producing an interior element. The
computation terminates after exactly k steps.
```

4. Arithmetic Operations

4.1 The bounded naturals

Definition 4.1 (bounded naturals):

```
ℕ_B(k) = {0, 1, 2, ..., k}

Construction: ℕ_B(k) = S(k) = k ∪ {k}
  {k} exists by BFT 4                    (k is interior)
  k ∪ {k} exists by BFT 5                (both are interior)
```

Cardinality: |ℕ_B(k)| = S(k). The identification S(k) = k + 1 follows from the definition of addition (Section 4.2).

Interiority: when k is interior (n_M > k), all elements of ℕ_B(k) are interior.

There is no set ℕ in BST. There is no completed infinite set of all natural numbers. ℕ_B(k) is the natural number set for bound k.

4.2 Addition

Definition 4.2 (addition):

```
Defined by BR-BST on n, holding m fixed.

m + 0 := m
m + S(n) := S(m + n)

BR-BST instantiation: g = m, h(n, v) = S(v).

Each step computes S(m + n) = (m + n) ∪ {m + n},
which requires m + n to be interior (BFT 4, BFT 5).
When m + n is interior, the step is exact.
```

4.3 Multiplication

Definition 4.3 (multiplication):

```
Defined by BR-BST on n, holding m fixed.

m × 0 := 0
m × S(n) := (m × n) + m

BR-BST instantiation: g = 0, h(n, v) = v + m.

Each step uses addition (Definition 4.2). This is the
first instance of iterated recursion (Remark 3.5):
the step function h uses addition, which was itself
defined by BR-BST.
```

4.4 Exponentiation

Definition 4.4 (exponentiation):

```
Defined by BR-BST on n, holding m fixed.

m^0 := 1
m^{S(n)} := (m^n) × m

BR-BST instantiation: g = 1, h(n, v) = v × m.

Each step uses multiplication (Definition 4.3). This
is the second layer of iterated recursion:
exponentiation uses multiplication, which uses addition.
```

4.5 Predecessor and bounded subtraction

Definition 4.5 (predecessor):

```
P(0) = 0
P(S(n)) = n

BR-BST instantiation: g = 0, h(n, v) = n.

The step function h returns its first argument,
ignoring the accumulated value v. This is permitted
by Schema 3.1: h is any BFOL formula defining a
function from pairs of interior elements to interior
elements. The formula h(n, v) = n satisfies this
(it ignores v and returns n, which is interior by
Lemma 2.3).

Computation:
  P(0) = 0
  P(1) = h(0, 0) = 0
  P(2) = h(1, P(1)) = h(1, 0) = 1
  P(3) = h(2, P(2)) = h(2, 1) = 2
  In general, P(S(n)) = n.
```

Definition 4.6 (monus):

```
m ∸ 0 := m
m ∸ S(n) := P(m ∸ n)

BR-BST instantiation: g = m, h(n, v) = P(v).
```

The monus $\dot{-}$ is the standard truncated subtraction on natural numbers: the result is m - n when m $\geq$ n, and 0 when m < n. It is a total function on $\mathbb{N}$_B(k), unlike integer subtraction (which produces negative values and belongs to BNT's $\mathbb{Z}$_B(k)).

Lemma 4.7 (successor cancellation for monus):

```
S(a) ∸ S(n) = a ∸ n.

Proof: by BI-BST on n.
Base: S(a) ∸ S(0) = P(S(a) ∸ 0)          (definition of ∸)
                  = P(S(a))               (definition of ∸)
                  = a                     (Definition 4.5)
                  = a ∸ 0.                (definition of ∸)
Step: assume S(a) ∸ S(n) = a ∸ n.
  S(a) ∸ S(S(n)) = P(S(a) ∸ S(n))         (definition of ∸)
                 = P(a ∸ n).              (induction hypothesis)
  a ∸ S(n) = P(a ∸ n).                    (definition of ∸)
  So S(a) ∸ S(S(n)) = a ∸ S(n).
```

Lemma 4.8 (monus properties):

```
(i) m ∸ 0 = m.
  Proof: by definition of ∸.

(ii) 0 ∸ n = 0.
  Proof: by BI-BST on n.
  Base: 0 ∸ 0 = 0.                        (definition of ∸)
  Step: assume 0 ∸ n = 0.
    0 ∸ S(n) = P(0 ∸ n)                   (definition of ∸)
             = P(0)                        (induction hypothesis)
             = 0.                          (Definition 4.5)

(iii) (m + n) ∸ n = m.
  Proof: by BI-BST on n.
  Base: (m + 0) ∸ 0 = m ∸ 0 = m.          (definition of +, property (i))
  Step: assume (m + n) ∸ n = m.
    (m + S(n)) ∸ S(n)
      = S(m + n) ∸ S(n)                   (definition of +)
      = (m + n) ∸ n                       (Lemma 4.7)
      = m.                                (induction hypothesis)

(iv) m ∸ n = 0 iff m ≤ n.
  Proof (⇒): by BI-BST on m.
```

```
    Base: 0 ∸ n = 0 and 0 ≤ n.          (property (ii), Theorem 4.12)
    Step: if S(m) ∸ n = 0:
      If n = 0: S(m) ∸ 0 = S(m) ≠ 0,
        contradiction.
      So n = S(n') for some n'.
      S(m) ∸ S(n') = m ∸ n' = 0.        (Lemma 4.7)
      By induction hypothesis, m ≤ n'.
      So S(m) ≤ S(n') = n.
  Proof (⇐): by BI-BST on m.
    Base: 0 ∸ n = 0.                    (property (ii))
    Step: if S(m) ≤ n, then n ≥ 1,
      so n = S(n') with m ≤ n'.
      S(m) ∸ n = S(m) ∸ S(n')
               = m ∸ n'                 (Lemma 4.7)
               = 0.                      (induction hypothesis)
```

4.6 The interiority condition on arithmetic

Arithmetic on $\mathbb{N}$_B(k) is exact for interior results. The interiority condition determines the precise range of available computation:

```
m + n is interior when m + n ≤ k.
m × n is interior when m × n ≤ k.
m^n   is interior when m^n   ≤ k.
```

When a result would exceed k, the successor construction cannot be performed (BFT 4 requires interiority). The operation is not ”wrong” or ”truncated”; it is simply not available.

This paper works within stated bounds: arithmetic is performed on the domain where results are interior, with explicit provisos when an operation might exceed the bound.

The alternative (truncated arithmetic: m +_k n = min(m+n, k)) is unconditionally closed but sacrifices the cancellation law. This paper uses domain restriction throughout.

4.7 The natural order

Every bounded ordinal in $\mathbb{N}$_B(k) is a transitive set. This fact, used throughout the order theory, is proved by BI-BST.

Lemma 4.9 (ordinal transitivity):

```
For all α in ℕ_B(k): if β ∈ α then β ⊆ α.
(Equivalently: every member of α is a subset of α.)

Proof: by BI-BST on α.
Base (α = 0 = ∅): vacuously true
  (∅ has no members).
Step: assume every member of α is a subset of α.
  Let γ ∈ S(α) = α ∪ {α}.
  Either γ ∈ α or γ = α.
  If γ ∈ α: by the induction hypothesis,
    γ ⊆ α. Since α ⊆ S(α), we have γ ⊆ S(α).
  If γ = α: α ⊆ α ∪ {α} = S(α).
  In both cases, γ ⊆ S(α).
```

Definition 4.10 (natural order):

```
m ≤ n iff m ⊆ n (bounded ordinals).
Equivalently: m ≤ n iff m ∈ n or m = n.

m < n iff m ∈ n.
Equivalently: m ≤ n and m ≠ n.
```

The equivalence of $m \subseteq n$ and ($m \in n$ or $m = n$) follows from ordinal transitivity (Lemma 4.9): if $m \in n$ then $m \subseteq n$ (by Lemma 4.9), and $m \subseteq n$ together with $m \neq n$ gives $m \in n$ (by Lemma 4.11 below).

Lemma 4.11 (proper subsets of ordinals are members):

```
For bounded ordinals m, n in ℕ_B(k):
if m ⊆ n and m ≠ n, then m ∈ n.

Proof: by BI-BST on n.
Base (n = 0 = ∅): if m ⊆ ∅ then m = ∅.
  So m ≠ n is impossible. Vacuously true.
Step: assume the result for n.
  Suppose m ⊆ S(n) = n ∪ {n} and m ≠ S(n).
  Case 1: m ⊆ n.
    If m = n: then m ∈ S(n)              (since n ∈ n ∪ {n} = S(n)).
    If m ≠ n: then m ∈ n                 (induction hypothesis).
    In either case, m ∈ S(n)             (since n ⊆ S(n), Lemma 4.9).
  Case 2: m ⊄ n.
    Then there exists x ∈ m with x ∉ n.
    Since m ⊆ n ∪ {n}, we have x ∈ {n},
    so x = n. Hence n ∈ m.
    By Lemma 4.9, n ∈ m gives n ⊆ m.
    Combined with m ⊆ S(n) = n ∪ {n}:
    every member of m is in n ∪ {n}, and
    n ∈ m, so m contains n and possibly
    members of n.
    But m ≠ S(n) = n ∪ {n}, so there exists
    y ∈ S(n) with y ∉ m. Since n ∈ m,
    y ≠ n, so y ∈ n. But y ∈ n and
    n ⊆ m gives y ∈ m, contradiction.
    So Case 2 is impossible.
```

Theorem 4.12 (total order):

```
≤ is a total order on ℕ_B(k).

Reflexivity: m ≤ m.                      (m = m)

Antisymmetry: m ≤ n and n ≤ m implies m = n.
  Proof: if m ≠ n then m ∈ n and n ∈ m.
  Then {m, n} is a nonempty set with no
  ∈-minimal element (m contains n and n
```

```
  contains m), contradicting Foundation
  (BFT 1). So m = n.

Transitivity: m ≤ n and n ≤ p implies m ≤ p.
  If m = n or n = p, immediate. If m ∈ n
  and n ∈ p, then m ∈ p: by Lemma 4.9,
  n ∈ p gives n ⊆ p, so m ∈ n ⊆ p gives
  m ∈ p.

Trichotomy: for any m, n in ℕ_B(k), either
  m ∈ n, m = n, or n ∈ m.
  Proof: by BI-BST on m.
  Base (m = 0): either n = 0 (so m = n)
    or n ≠ 0 (then ∅ ⊆ n and ∅ ≠ n,
    so 0 ∈ n by Lemma 4.11).
  Step: assume trichotomy holds for m
    against all n. For S(m) and n:
    If n ≤ m then n ∈ S(m) (since
    S(m) = m ∪ {m} and n ∈ m or n = m).
    If m < n then either S(m) = n or
    S(m) < n. Proof: m ∈ n, so by
    Lemma 4.9, m ⊆ n. Also {m} ⊆ n
    (since m ∈ n). So S(m) = m ∪ {m} ⊆ n.
    If S(m) = n, done. If S(m) ≠ n, then
    S(m) ∈ n by Lemma 4.11.
```

Theorem 4.13 (well-ordering):

```
Every nonempty subset of ℕ_B(k) has a minimum.

Proof: let S be nonempty. By BFT 1
  (Foundation), S has an ∈-minimal element
  m: no member of m is in S. By Lemma 4.9,
  the members of m are exactly the ordinals
  less than m (since β ∈ m iff β < m). So no
  ordinal less than m is in S. So m is the
  minimum of S under the ordinal order.
```

Minimum element of $\mathbb{N}_B(k)$: 0. Maximum element: k.

4.8 Compatibility of order with arithmetic

Theorem 4.14 (addition preserves order):

```
If m ≤ n then m + p ≤ n + p.

Proof: by BI-BST on p.
Base: m + 0 = m ≤ n = n + 0.
Step: assume m + p ≤ n + p.
  m + S(p) = S(m + p)                    (definition of +)
  n + S(p) = S(n + p)                    (definition of +)
  m + p ≤ n + p                          (induction hypothesis)
  If m + p = n + p: S(m + p) = S(n + p).
  If m + p ∈ n + p: by Lemma 4.9,
    m + p ∈ n + p gives m + p ⊆ n + p.
    Also m + p ∈ S(n + p) (since
    n + p ∈ S(n + p) and m + p ∈ n + p ⊆ S(n + p)
    by Lemma 4.9 applied to S(n + p)).
    And {m + p} ⊆ S(n + p).
    So S(m + p) = (m + p) ∪ {m + p} ⊆ S(n + p),
    hence S(m + p) ≤ S(n + p).
```

Theorem 4.15 (multiplication preserves order):

```
If m ≤ n then m × p ≤ n × p.

Proof: by BI-BST on p.
Base: m × 0 = 0 ≤ 0 = n × 0.
Step: assume m × p ≤ n × p.
  m × S(p) = m × p + m                   (definition of ×)
  n × S(p) = n × p + n                   (definition of ×)
  m × p ≤ n × p                          (induction hypothesis)
  m × p + m ≤ n × p + m                  (Theorem 4.14)
  m ≤ n, so n × p + m ≤ n × p + n        (Theorem 4.14)
  m × p + m ≤ n × p + n.                 (transitivity of ≤, Theorem 4.12)
```

4.9 Minimum and maximum

Theorem 4.16 (bounded extremal principle):

```
Every nonempty interior subset S of ℕ_B(k) has
both a minimum and a maximum.

min(S) exists by well-ordering (Theorem 4.13).

max(S) exists by finiteness: S is a finite
totally ordered set. Constructed by BR-BST:
start with an arbitrary element s_0 of S,
iterate through S comparing each element to
the current maximum, update if larger.
Terminates in |S| steps.
```

5. Algebraic Identities

All proved by BI-BST. Each proof is a finite derivation, valid in every standard model where the bound is interior.

5.1 Additive identities

Lemma 5.1 (left identity):

```
0 + m = m.

Proof: by BI-BST on m.
Base: 0 + 0 = 0.                        (definition of +)
Step: assume 0 + m = m.
  0 + S(m) = S(0 + m)                   (definition of +)
           = S(m).                       (induction hypothesis)
```

Lemma 5.2 (successor shifts right):

```
n + S(m) = S(n + m).

Proof: by BI-BST on n.
Base: 0 + S(m) = S(m)                   (Lemma 5.1)
               = S(0 + m).              (Lemma 5.1)
Step: assume n + S(m) = S(n + m).
  S(n) + S(m) = S(S(n) + m)             (definition of +)
              = S(S(n + m))             (definition of +: S(n) + m = S(n + m))
  n + S(m) = S(n + m)                   (induction hypothesis)
  S(n + S(m)) = S(S(n + m))             (applying S to both sides)
  S(n) + S(m) = S(n + S(m)).            (combining)
```

Lemma 5.3 (successor is injective):

```
If S(a) = S(b) then a = b.

Proof: S(a) = a union {a} and S(b) = b union {b}.
If a union {a} = b union {b}, then a is the largest
member of S(a) (a is in S(a), and no member of a
contains a, since bounded ordinals are transitive
by Lemma 4.9).
Similarly b is the largest member of S(b). Since
S(a) = S(b), their largest members are equal: a = b.
```

Theorem 5.4 (commutativity of addition):

```
m + n = n + m.

Proof: by BI-BST on n.
Base: m + 0 = m                          (definition of +)
            = 0 + m.                     (Lemma 5.1)
Step: assume m + n = n + m.
  m + S(n) = S(m + n)                    (definition of +)
           = S(n + m)                    (induction hypothesis)
           = n + S(m).                   (Lemma 5.2)
```

Theorem 5.5 (associativity of addition):

```
(m + n) + p = m + (n + p).

Proof: by BI-BST on p.
Base: (m + n) + 0 = m + n                (definition of +)
                  = m + (n + 0).         (definition of +)
Step: assume (m + n) + p = m + (n + p).
  (m + n) + S(p) = S((m + n) + p)        (definition of +)
                 = S(m + (n + p))        (induction hypothesis)
                 = m + S(n + p)          (definition of +)
                 = m + (n + S(p)).       (definition of +)
```

Theorem 5.6 (additive cancellation):

```
If m + p = n + p then m = n.

Proof: by BI-BST on p.
Base: m + 0 = n + 0 gives m = n.        (definition of +)
Step: assume the result for p.
  m + S(p) = n + S(p)
  S(m + p) = S(n + p)                   (definition of +)
  m + p = n + p                         (Lemma 5.3, injectivity)
  m = n.                                (induction hypothesis)
```

5.2 Multiplicative identities

Lemma 5.7 (left zero):

```
0 × m = 0.

Proof: by BI-BST on m.
Base: 0 × 0 = 0.                        (definition of ×)
Step: assume 0 × m = 0.
  0 × S(m) = 0 × m + 0                  (definition of ×)
           = 0 + 0                      (induction hypothesis)
           = 0.                         (definition of +)
```

Lemma 5.8 (left multiplicative identity):

```
1 × m = m.

Proof: by BI-BST on m.
Base: 1 × 0 = 0.                        (definition of ×)
Step: assume 1 × m = m.
  1 × S(m) = 1 × m + 1                  (definition of ×)
           = m + 1                      (induction hypothesis)
           = S(m).                      (definition of +: m + S(0) = S(m + 0) = S(m))
```

Lemma 5.9 (left successor):

```
S(n) × m = n × m + m.

Proof: by BI-BST on m.
Base: S(n) × 0 = 0                      (definition of ×)
               = 0 + 0                 (Lemma 5.1)
               = n × 0 + 0.            (definition of ×)
Step: assume S(n) × m = n × m + m.
  S(n) × S(m) = S(n) × m + S(n)         (definition of ×)
              = (n × m + m) + S(n)      (induction hypothesis)
              = (n × m + m) + (n + 1)   (definition of +: S(n) = n + 1)
              = n × m + (m + (n + 1))   (Theorem 5.5, associativity of +)
              = n × m + ((m + n) + 1)   (Theorem 5.5, associativity of +)
              = n × m + ((n + m) + 1)   (Theorem 5.4, commutativity of +)
              = n × m + (n + (m + 1))   (Theorem 5.5, associativity of +)
              = (n × m + n) + (m + 1)   (Theorem 5.5, associativity of +)
              = (n × m + n) + S(m)      (definition of +: m + 1 = S(m))
              = n × S(m) + S(m).        (definition of ×: n × S(m) = n × m + n)
```

The definition of multiplication (Section 4.3) gives $m \times S(n) = m \times n + m$ directly. Lemma 5.9 requires proof because multiplication is defined by recursion on the second argument, not the first.

Theorem 5.10 (distributivity):

```
m × (n + p) = m × n + m × p.

Proof: by BI-BST on p.
Base: m × (n + 0) = m × n               (definition of +)
                  = m × n + 0           (definition of +)
                  = m × n + m × 0.      (definition of ×)
Step: assume m × (n + p) = m × n + m × p.
  m × (n + S(p)) = m × S(n + p)         (definition of +)
                 = m × (n + p) + m      (definition of ×)
                 = (m × n + m × p) + m  (induction hypothesis)
                 = m × n + (m × p + m)  (Theorem 5.5, associativity of +)
                 = m × n + m × S(p).    (definition of ×)
```

Theorem 5.11 (commutativity of multiplication):

```
m × n = n × m.

Proof: by BI-BST on n.
Base: m × 0 = 0                          (definition of ×)
           = 0 × m.                     (Lemma 5.7)
Step: assume m × n = n × m.
  m × S(n) = m × n + m                   (definition of ×)
           = n × m + m                   (induction hypothesis)
           = S(n) × m.                   (Lemma 5.9)
```

Theorem 5.12 (associativity of multiplication):

```
(m × n) × p = m × (n × p).

Proof: by BI-BST on p.
Base: (m × n) × 0 = 0                    (definition of ×)
                  = m × 0                (definition of ×)
                  = m × (n × 0).         (definition of ×)
Step: assume (m × n) × p = m × (n × p).
  (m × n) × S(p) = (m × n) × p + (m × n)   (definition of ×)
                 = m × (n × p) + m × n     (induction hypothesis)
                 = m × (n × p + n)         (Theorem 5.10, distributivity)
                 = m × (n × S(p)).         (definition of ×)
```

The associativity proof uses distributivity (Theorem 5.10). In the paper, distributivity is proved first and associativity second, matching the order in which they can be verified.

5.3 Exponentiation identities

All identities require all intermediate results to be interior (i.e., $\leq$ k). Exponentiation grows fastest of the three operations, so the interiority provisos are most restrictive here.

Theorem 5.13 (exponentiation base case):

```
m^0 = 1.

Proof: by definition of exponentiation (Section 4.4).
```

Theorem 5.14 (exponentiation identity):

```
m^1 = m.

Proof:
  m^1 = m^{S(0)}
      = m^0 × m                          (definition of ^)
      = 1 × m                            (Theorem 5.13)
      = m.                               (Lemma 5.8)
```

Theorem 5.15 (exponentiation distributes over addition of exponents):

```
m^{a+b} = m^a × m^b.

Proviso: m^a, m^b, m^{a+b}, m^a × m^b all ≤ k.

Proof: by BI-BST on b.
Base: m^{a+0} = m^a                      (definition of +)
              = m^a × 1                  (definition of +: m^a + 0 = m^a,
                                          and m^a × 1 = m^a by Lemma 5.8
                                          and Theorem 5.11)
              = m^a × m^0.              (Theorem 5.13)
Step: assume m^{a+b} = m^a × m^b.
  m^{a+S(b)} = m^{S(a+b)}               (Lemma 5.2: a + S(b) = S(a + b))
             = m^{a+b} × m              (definition of ^)
             = (m^a × m^b) × m          (induction hypothesis)
             = m^a × (m^b × m)          (Theorem 5.12, associativity of ×)
             = m^a × m^{S(b)}.          (definition of ^)
```

Theorem 5.16 (exponentiation distributes over multiplication of bases):

```
(m × n)^a = m^a × n^a.

Proviso: m^a, n^a, (m×n)^a, m^a × n^a all ≤ k.

Proof: by BI-BST on a.
Base: (m × n)^0 = 1                      (definition of ^)
              = 1 × 1                   (Lemma 5.8)
              = m^0 × n^0.             (definition of ^)
Step: assume (m × n)^a = m^a × n^a.
  (m × n)^{S(a)} = (m × n)^a × (m × n)  (definition of ^)
                 = (m^a × n^a) × (m × n) (induction hypothesis)
                 = m^a × n^a × m × n
                 = m^a × m × n^a × n     (Theorem 5.11, commutativity of ×)
                 = (m^a × m) × (n^a × n) (Theorem 5.12, associativity of ×)
                 = m^{S(a)} × n^{S(a)}.  (definition of ^)
```

Theorem 5.17 (power of a power):

```
(m^a)^b = m^{a×b}.

Proviso: m^a, m^{a×b}, (m^a)^b all ≤ k.

Proof: by BI-BST on b.
Base: (m^a)^0 = 1                        (definition of ^)
             = m^0                      (definition of ^)
             = m^{a*0}.                (definition of ×)
Step: assume (m^a)^b = m^{a×b}.
  (m^a)^{S(b)} = (m^a)^b × m^a           (definition of ^)
               = m^{a×b} × m^a           (induction hypothesis)
               = m^{a×b + a}             (Theorem 5.15)
               = m^{a*S(b)}.             (definition of *: a × S(b) = a × b + a)
```

5.4 The semiring structure

The algebraic identities of Sections 5.1-5.3 establish that (ℕ_B(k), +, ×, 0, 1) is a commutative semiring on the domain where operations produce interior results. It satisfies commutativity, associativity, and distributivity of both operations, with additive identity 0 and multiplicative identity 1. It lacks additive inverses (those require ℤ_B(k), which belongs to subsequent work).

6. Elementary Number Theory

6.1 Divisibility

Divisibility is decidable in $\mathbb{N}_B(k)$: the quantifier is bounded, so the property can be checked by finite search over $\{0, ..., k\}$.

Definition 6.1 (divisibility):

```
m | n (m divides n) iff ∃q ≤ k (n = m × q).
```

Lemma 6.2 (multiplicative cancellation):

```
If m × a = m × b and m > 0, then a = b.

Proof: by BI-BST on a.
  Base (a = 0): m × 0 = m × b gives
    0 = m × b.                          (definition of ×)
    If b > 0 then m × b ≥ m > 0         (Theorem 4.15)
    contradiction. So b = 0.
  Step: assume the result for a.
    m × S(a) = m × b.
    m × a + m = m × b.                  (definition of ×)
    If b = 0: m × a + m = 0, but
    m > 0 so m × a + m ≥ m > 0,         (Theorem 4.14)
    contradiction.
    So b = S(b') for some b'.
    m × a + m = m × b' + m.             (definition of ×)
    m × a = m × b'.                     (Theorem 5.6, additive cancellation)
    a = b'.                             (induction hypothesis)
    S(a) = S(b') = b.
```

Lemma 6.3 (divisibility properties):

```
Provable by BI-BST:

Reflexivity:  m | m                        (take q = 1)

Transitivity: m | n and n | p → m | p
  Proof: n = m×q and p = n×r gives
  p = m×(q×r).                             (Theorem 5.12, associativity of ×)

Antisymmetry: m | n and n | m and
              m, n > 0 → m = n
  Proof: m×q = n and n×r = m gives
  m×q×r = m, so m×(q×r) = m×1.          (Theorem 5.12, Lemma 5.8)
  By Lemma 6.2, q×r = 1.
  Since q, r ≥ 1 and q×r = 1:
  q ≤ 1 (if q > 1 then q×r ≥ q > 1,     (Theorem 4.15)
  contradiction). So q = 1, hence r = 1,
  hence m = n.

Compatibility: m | a and m | b → m | (a + b)
  Proof: a = m×q₁ and b = m×q₂ gives
  a + b = m×q₁ + m×q₂ = m×(q₁ + q₂).   (Theorem 5.10, distributivity)
```

6.2 Division with remainder

Theorem 6.4 (division with remainder):

```
For any m and n > 0 in ℕ_B(k), there exist unique
q, r with m = n × q + r and 0 ≤ r < n.

Existence: by BI-BST on m.
  Base (m = 0): q = 0, r = 0.
    0 = n × 0 + 0 and 0 < n.            (n > 0 by hypothesis)
  Step: if m = n×q + r with r < n, then
    S(m) = n×q + S(r).
    If S(r) < n: take q' = q, r' = S(r).
      S(m) = n×q + S(r) and 0 ≤ S(r) < n.
    If S(r) = n: take q' = S(q), r' = 0.
```

```
       S(m) = n×q + n = n×S(q) + 0       (definition of ×)
       and 0 < n.
     (S(r) > n is impossible: r < n
      implies S(r) ≤ n by Theorem 4.12.)

Uniqueness: suppose m = n×q₁ + r₁ = n×q₂ + r₂
  with 0 ≤ r₁, r₂ < n.

  Suppose q₁ > q₂. Then q₁ ≥ S(q₂), so
    n × q₁ ≥ n × S(q₂)                  (Theorem 4.15)
           = n × q₂ + n.                (definition of ×)
  So n × q₁ + r₁ ≥ n × q₂ + n           (Theorem 4.14)
                 > n × q₂ + r₂.          (since n > r₂)
  But n × q₁ + r₁ = n × q₂ + r₂,
  contradiction.

  By symmetry (swapping q₁ and q₂), q₂ > q₁
  also leads to contradiction.

  So q₁ = q₂. Then n × q₁ + r₁ = n × q₁ + r₂,
  so r₁ = r₂.                           (Theorem 5.6, additive cancellation)
```

6.3 GCD and the Euclidean algorithm

Definition 6.5 (GCD):

```
For any m, n in ℕ_B(k) not both zero, gcd(m, n)
is the largest d dividing both m and n.
```

The Euclidean algorithm computes gcd(m, n) by course-of-values recursion (Remark 3.6). We write m mod n for the remainder r from Theorem 6.4, and $\lfloor m/n \rfloor$ for the quotient q.

The correctness proof requires distributivity of $\times$ over $\dot{-}$:

Lemma 6.6 (distributivity of $\times$ over $\dot{-}$):

```
m × (a ∸ b) = m × a ∸ m × b.

Proof: two cases.

Case a ≥ b: then a = b + (a ∸ b)          (Lemma 4.8(iii), reversed)
  so m × a = m × (b + (a ∸ b))
           = m × b + m × (a ∸ b).         (Theorem 5.10, distributivity)
  Therefore m × a ∸ m × b
           = (m × b + m × (a ∸ b)) ∸ (m × b)
           = m × (a ∸ b).                 (Lemma 4.8(iii))

Case a < b: then a ∸ b = 0                (Lemma 4.8(iv))
  so m × (a ∸ b) = m × 0 = 0.            (definition of ×)
  Also m × a ≤ m × b                      (Theorem 4.15, since a ≤ b)
  so m × a ∸ m × b = 0.                   (Lemma 4.8(iv))
  Both sides equal 0.
```

Theorem 6.7 (Euclidean algorithm):

```
gcd(m, 0) = m.
gcd(m, n) = gcd(n, m mod n) for n > 0.

Termination: the second argument strictly
  decreases at each step (m mod n < n,
  by Theorem 6.4). Since the second argument
  is in ℕ_B(k) and decreases by at least 1
  at each step, the algorithm terminates in
  at most n steps.

Correctness: by SBI-BST on the second argument.
  At each step, m = n × q + r where
  r = m mod n and q = ⌊m/n⌋.             (Theorem 6.4)

  Any d | m and d | n: write m = d×s, n = d×t.
    Then n×q = d×t×q, and
    r = m ∸ n×q = d×s ∸ d×(t×q)
      = d × (s ∸ t×q).                   (Lemma 6.6)
    So d | r.

  Any d | n and d | r: write n = d×t, r = d×u.
    Then m = n×q + r = d×t×q + d×u
           = d × (t×q + u).              (Theorem 5.10)
    So d | m.

  The common divisors of (m, n) and (n, r) are
  identical. The greatest common divisor is
  therefore the same.
```

6.4 Bezout's identity

Theorem 6.8 (Bezout's identity, natural number form):

```
For any m, n in ℕ_B(k) not both zero, the extended
Euclidean algorithm produces p, q in ℕ_B(k) such
that either:
  p × m = q × n + gcd(m, n)
or:
```

```
  q × n = p × m + gcd(m, n)

Proof: by SBI-BST on the second argument of the
  Euclidean algorithm (same termination measure
  as Theorem 6.7).

  Base (n = 0): gcd(m, 0) = m.
    Take p = 1, q = 0.
    1 × m = 0 × 0 + m.                  (Lemma 5.8, definition of ×)

  Step: assume the result for gcd(n, m mod n).
    Let q₀ = ⌊m/n⌋ and r = m mod n,
    so m = n × q₀ + r.                  (Theorem 6.4)

    By the induction hypothesis, there exist
    p', q' such that (taking the first form):
      p' × n = q' × r + gcd(n, r).

    Since gcd(n, r) = gcd(m, n)          (Theorem 6.7)
    and r = m ∸ n × q₀:

      p' × n = q' × (m ∸ n × q₀) + gcd(m, n)
             = (q' × m ∸ q' × n × q₀) + gcd(m, n).
                                        (Lemma 6.6)

    Rearranging (adding q' × n × q₀ to both
    sides, using Theorem 5.5 and Theorem 5.4):

      p' × n + q' × n × q₀ = q' × m + gcd(m, n)
      (p' + q' × q₀) × n = q' × m + gcd(m, n).
                                        (Theorem 5.10)

    Take p = q', q = p' + q' × q₀.
    Then q × n = p × m + gcd(m, n).

    (The symmetric case, where the induction
    hypothesis gives q' × r = p' × n + gcd(n, r),
```

```
    is handled analogously, producing the
    other form of the identity.)
```

This natural number form avoids negative coefficients (negative integers belong to ℤ_B(k), not constructed in this paper).

6.5 Primality

Primality is decidable in ℕ_B(k): the quantifier is bounded, so the property can be checked by finite search over {1, ..., n}.

Definition 6.9 (prime):

```
n is prime iff n > 1 ∧ ∀m ≤ n (m | n → m = 1 ∨ m = n).
```

The sieve of Eratosthenes computes the set of all primes up to a given bound.

Remark 6.10 (sieve of Eratosthenes):

```
For a given bound B ≤ k, the set of primes up to B
is computed by BR-BST:
  Start with {2, 3, ..., B}              (by BFT 7, Separation)
  For each p from 2 to ⌊√B⌋:
    remove all multiples of p greater than p
    from the set                          (by BFT 7, Separation)
  The remaining elements are the primes up to B.

Termination: the outer loop runs at most
  ⌊√B⌋ - 1 steps. Each inner loop removes
  at least one element. All exact.
```

6.6 Unique factorisation

Lemma 6.11 (Euclid's lemma):

```
If prime p divides a × b, then p | a or p | b.

Proof: suppose p /| a. Then gcd(p, a) = 1
  (since p is prime, its only divisors are
  1 and p; since p /| a, gcd(p, a) ≠ p;
  so gcd(p, a) = 1).
```

```
By Bezout (Theorem 6.8), there exist p', a'
such that either:
  p' × p = a' × a + 1        ... (Form 1)
or:
  a' × a = p' × p + 1        ... (Form 2)

Consider Form 1: p' × p = a' × a + 1.
Multiply both sides by b:
  p' × p × b = (a' × a + 1) × b
             = a' × a × b + 1 × b      (Theorem 5.10)
             = a' × a × b + b.          (Lemma 5.8)

Since p | a × b, write a × b = p × t.
Then a' × a × b = a' × (p × t)
                = (a' × t) × p.         (Theorems 5.12, 5.11)
Substituting:
  (p' × b) × p = (a' × t) × p + b.     (Theorems 5.12, 5.11)

So b = (p' × b) × p ∸ (a' × t) × p
     = (p' × b ∸ a' × t) × p.           (Lemma 6.6)
Therefore p | b.

Form 2: a' × a = p' × p + 1.
Multiply both sides by b:
  a' × a × b = p' × p × b + b.
  (a' × t) × p = (p' × b) × p + b.     (same rearrangement)
So b = (a' × t) × p ∸ (p' × b) × p
     = (a' × t ∸ p' × b) × p.           (Lemma 6.6)
Therefore p | b.
```

Theorem 6.12 (unique factorisation):

```
For any n in ℕ_B(k) with n > 1, n has a unique
prime factorisation.

Existence: by SBI-BST on n.
  If n is prime, the factorisation is (n) itself.
  If n is composite, n = a × b with 1 < a, b < n.
  By the strong induction hypothesis, a and b
  have prime factorisations. Their concatenation
  is a prime factorisation of n.

Uniqueness: by SBI-BST on n.
  Suppose n = p₁ × ... × p_r = q₁ × ... × q_s
  are two prime factorisations.
  p₁ | q₁ × ... × q_s = n.
  By repeated application of Euclid's lemma
  (Lemma 6.11), p₁ | q_j for some j.
  Since p₁ and q_j are both prime, p₁ = q_j.
  Cancel p₁ from both sides:
    p₂ × ... × p_r = (q₁ × ... × q_{j-1} ×
                      q_{j+1} × ... × q_s)
  by multiplicative cancellation (Lemma 6.2).
  Apply the induction hypothesis to n/p₁ < n.
```

7. Scope and Limits

7.1 What bounded arithmetic proves

Every specific numerical identity (e.g., 7 + 5 = 12) is provable by direct computation within $\mathbb{N}$_B(k) for any k $\geq$ 12. The computation is a BR-BST evaluation: iterate the successor 5 times starting from 7, producing S(S(S(S(S(7))))) = 12.

Every universal identity over a bounded domain (e.g., $\forall$m,n $\leq$ k, m + n = n + m) is provable by BI-BST: the base case and step case are verified, and BI-BST (Remark 2.5, finite iteration) yields the conclusion in exactly k steps.

Elementary number theory up to unique factorisation (Theorem 6.12) is fully proved.

7.2 Recursion depth

BST can define functions by iterated recursion (Remark 3.5) to any fixed depth. Addition is one level of BR-BST. Multiplication is one level of BR-BST whose step function uses addition (two levels total). Exponentiation is one level of BR-BST whose step function uses multiplication (three levels total). d-fold iterated recursion defines d-level towers. For any specific d, the totality of d-fold iterated recursion is provable by BI-BST.

What BST cannot prove total: functions whose recursion depth is itself a variable. The Ackermann function, where the recursion depth grows with the input, is the canonical example. Every specific value A(m, n) is computable by BR-BST for large enough k (it is a specific finite number, and the computation terminates in finitely many steps). But the universal statement "A is total" (for all m, n there exists a result) is not provable in BST, because it requires quantifying over unbounded recursion depth.

Theorem 7.1 (Ackermann computability):

```
For every specific m, n in ℕ_B(k), the value A(m, n)
is computable by BR-BST, provided k is large enough
to contain the result.

Proof: A(m, n) is defined by m-fold iterated
  recursion (Remark 3.5). For specific m, this
  is a fixed number of BR-BST applications:
    A(0, n) = n + 1                    (one level)
    A(1, n) uses A(0, -)               (two levels)
    A(2, n) uses A(1, -)               (three levels)
    ...
    A(m, n) uses A(m-1, -)            (m+1 levels)
  Each level is a BR-BST construction.
  The result is a specific interior element
  of ℕ_B(k).
```

The distinction is precise: ”computable at each instance” versus ”provably total.” BST proves the former for every instance but not the latter as a universal statement.

7.3 The boundary of provability

The boundary is real but narrow. Goodstein’s theorem, Paris-Harrington, and Ackermann totality sit at this boundary: every specific finite instance is provable in BST, but the universal quantification across all naturals is not.

For all of elementary number theory, all of finite algebra, all of combinatorics, and all of experimental physics, this boundary is irrelevant: no result in these fields requires recursion of variable depth. The self-grounding (Section 8) does not reach the boundary either: evaluating a BFOL sentence of quantifier depth d is recursion of depth d, which is fixed for each specific sentence.

8. Self-Grounding

8.1 The question

The bounded framework has a dependency chain: BFOL provides the logic, BST provides the set theory, and this paper provides the arithmetic. The question is whether this chain is self-supporting.

Self-supporting means: the arithmetic can evaluate the sentences of the logic that defines the set theory that constructs the arithmetic. If so, nothing outside the chain is needed. The chain validates itself by direct computation.

8.2 Direct evaluation

The answer is yes. For any specific BFOL sentence φ and any specific standard model $\mathcal{V}$_n, the bounded arithmetic can determine whether $\mathcal{V}$_n $\models \varphi$ by direct computation.

Theorem 8.1 (evaluability):

```
For any BFOL sentence φ of quantifier depth d and
any standard model V_n, the truth value of V_n |= φ
is computable by BR-BST.

Proof: by recursion on the structure of φ.

Atomic formulas:
  x ∈ y: the set y is a finite set in V_n.
    Enumerate the members of y (finitely many).
    Check whether x is among them (by equality
    checks). This is a finite search, terminating
    in at most |y| steps.
  x = y: by extensionality (BFT 2), x = y iff
    every member of x is a member of y and every
    member of y is a member of x. Enumerate the
    members of x and of y (both finite). For each
    member of x, check membership in y. For each
    member of y, check membership in x. This is
    a finite double loop, terminating in at most
    |x| × |y| steps.
```

```
Logical connectives (¬, ∧, ∨, →): computed from
  the truth values of the subformulas by Boolean
  operations (one step each).

Bounded quantifiers: in BFOL, ∀x ≤ t ψ(x) means
  "for all x that are members of S(t), ψ(x) holds,"
  where S(t) = t ∪ {t}. The set S(t) is finite
  (it has at most |t| + 1 members, all in V_n).
  Evaluate ψ(x) for each x ∈ S(t) by BR-BST.
  ∀x ≤ t ψ(x) is the conjunction of finitely
  many truth values. ∃x ≤ t ψ(x) is the
  disjunction.

Each quantifier elimination reduces the depth by 1.
After d steps, only atomic formulas remain.
The computation terminates in finitely many steps,
all within ℕ_B(k) for large enough k.
```

This is not a soundness theorem. It is a computation. The arithmetic evaluates specific sentences in specific models. A soundness theorem would be a universal statement about all sentences and all models, requiring unbounded quantification. Direct evaluation is a specific computation for each specific instance.

8.3 The loop

The dependency chain is:

BFOL defines the language in which BST's axiom (AFB) and the BFTs are stated.

BST provides the standard models $\mathcal{V}$_n and the bounded ordinals from which ℕ_B(k) is constructed.

Bounded Arithmetic (this paper) provides BI-BST, BR-BST, and the arithmetic operations.

The loop closes because the bounded arithmetic can evaluate any BFOL sentence in any $\mathcal{V}$_n (Theorem 8.1). In particular, it can evaluate the BFTs themselves.

Example (BFT 4, Pairing, in $\mathcal{V}$_2):

```
BFT 4 states: if a, b are interior, {a, b} exists.

In V_2 = P(P({ø})) = {ø, {ø}, {{ø}}, {ø, {ø}}},
the interior is V_1 = {ø, {ø}}.

To evaluate BFT 4 in V_2: enumerate all pairs (a, b)
  of interior elements (a, b ∈ V_1, giving 4 pairs).
  For each pair, check whether {a, b} exists in V_2.

  (ø, ø):     {ø} ∈ V_2.                 ✓
  (ø, {ø}):   {ø, {ø}} ∈ V_2.             ✓
  ({ø}, ø):   {ø, {ø}} ∈ V_2.             ✓
  ({ø}, {ø}): {{ø}} ∈ V_2.                ✓

All cases hold. BFT 4 is true in V_2.

The computation enumerates a finite set (V_1 × V_1,
four pairs) and checks membership in a finite set
(V_2, four elements). Each check is a finite
comparison. The entire evaluation terminates in
finitely many steps using only BR-BST.
```

For any specific BFT and any specific $\mathcal{V}$_n, the arithmetic computes that the BFT holds. This is not a proof of the BFTs (the BFTs are proved in BST). It is the verification that the arithmetic is sufficient to carry out the reasoning that produces it.

No external system is consulted. The arithmetic evaluates the sentences of the logic that defines the set theory that constructs the arithmetic.

8.4 Consequences

The framework is self-supporting. The logic, the set theory, and the arithmetic form a closed loop with no external dependencies.

Every result in this paper is Type I (exact). The precision parameters on $\mathbb{N}$_B(k) are not approximations to values in some external framework of higher precision. There is no such framework. The bound k determines the domain, and within that domain every computation is exact, every truth value is determined, and every result is final.

9. Conclusion

This paper derives bounded induction, bounded recursion, the arithmetic operations, the algebraic identities, the order structure, and elementary number theory through unique factorisation from Bounded Set Theory as the sole prerequisite, using only BFTs 1, 4, 5, and 7. The construction proceeds in strict dependency order: bounded induction (Section 2), bounded recursion (Section 3), arithmetic operations and order (Section 4), algebraic identities (Section 5), and elementary number theory (Section 6).

The self-grounding loop closes by Theorem 8.1: the bounded arithmetic evaluates any BFOL sentence in any standard model by direct computation. The logic defines the set theory, the set theory constructs the arithmetic, and the arithmetic evaluates the logic. No external system is consulted. Every result is Type I.

References

Bounded First-Order Logic (BFOL). Working Paper, 2026. The logical substrate.

Bounded Set Theory (BST). Working Paper, 2026. The set theory from which the arithmetic is constructed. Together with BFOL and this paper, forms a self-supporting loop (Section 8).

Euclid. *Elements*, Book VII, Proposition 30 (c. 300 BCE). The result that if a prime divides a product then it divides one of the factors. Proved independently within BST as Lemma 6.11.