A UNNS Note on NP-Hardness: Substrate-Relative Complexity and Recursive Operators

—

Abstract

We consider the notion of NP-hardness in the context of the Unbounded Nested Number Sequences (UNNS) Substrate. While NP-hard problems are believed to lack polynomial-time algorithms under the standard Turing framework, the recursive operators of UNNS (Inletting, Inlaying, Trans-Sentifying, and Repair/Normalization) suggest an alternative view. We propose that complexity is not absolute, but substrate-relative, and that UNNS may transform apparently exponential problems into tractable polynomial ones.

1 Classical Complexity

In classical computational complexity theory:

- A decision problem is in **P** if it can be solved in polynomial time on a deterministic Turing machine.
- A problem is **NP**-hard if every problem in **NP** can be reduced to it in polynomial time.

It is widely conjectured that $\mathbf{P} \neq \mathbf{NP}$.

2 UNNS Substrate View

The UNNS substrate operates not by sequential symbol manipulation, but through recursive operators acting on nested lattices:

- **Inletting**: growth injection, introducing new recursive levels.

- **Inlaying**: structural embedding, compressing complexity into nested forms.
- **Trans-Sentifying**: projection of recursive states into perceivable invariants.
- **Repair/Normalization**: pruning redundant states and collapsing equivalent recursive branches.

Definition 1 (Substrate Complexity)**.** *The* substrate complexity *of a problem is the growth rate of resources required when it is embedded into the UNNS recursive grammar, measured in terms of recursion depth and lattice dimension, rather than Turing steps.*

3 Formal Statement

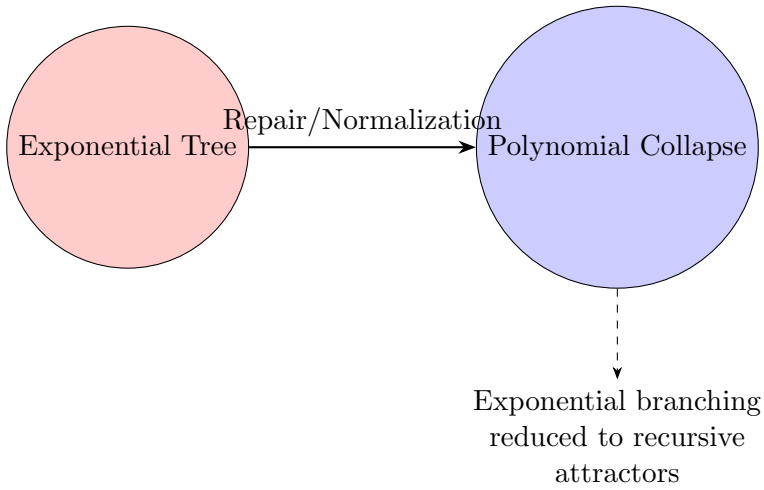
Lemma 1 (Collapse by Normalization)**.** *Any recursive process generating exponential state growth under flat enumeration may collapse to polynomial effective growth under UNNS Repair/Normalization, provided equivalence classes of recursive branches exist.*

Proposition 1 (NP-hardness is not substrate-invariant)**.** *Let P be a problem that is NP-hard in the Turing sense. If P can be represented as a recursive embedding in the UNNS substrate such that Repair/Normalization collapses exponential branching into polynomial equivalence classes, then P admits a polynomial substrate algorithm. Thus, NP-hardness is relative to the computational grammar.*

4 Consequences

- **Complexity Theory:** NP-hardness is reframed as Turing-relative, not universal.
- **Cryptography:** Hardness assumptions may fail in UNNS; new substrate-resistant cryptosystems are required.
- **Optimization:** Scheduling, logistics, and design become tractable under recursive attractor compressions.
- **Philosophy:** Complexity becomes a property of representational substrate, not of the problem itself.

5 Visualization



6 Remarks

This does not constitute a proof that **P** = **NP** in the classical sense. Rather, it highlights that computational hardness is dependent on the formal substrate. If the UNNS substrate can be realized physically or in simulation, it provides a new paradigm of tractable computation.