Vector Protocol v0.6.1 – Relational Recursion Addendum

Date: 2025-05-22

This addendum introduces and formalizes the principle of relational recursion as an emergent property of the Vector semantic space.

# 1. Principle of Relational Recursion

Vector does not define recursion as a procedural function. Instead, recursion emerges naturally from the ledgered, referential, and compositional properties of the semantic system. Nodes, sets, traversals, and agents are all first-class entities. When any of these refer to themselves or to other referring structures, recursion arises.

# 2. How Recursion Emerges

Recursion in Vector emerges when:

* • Concepts refer to other concepts
* • Sets contain nodes that refer to sets
* • Traversals become ledgered and referenced
* • Agents reason over structures that themselves contain traversals

These recursive chains form semantic loops, feedback structures, or reflective operations that do not require special syntax—only the permission to reference and the discipline to traverse.

# 3. Recursion Clause

All entities in Vector (nodes, sets, traversals, beliefs, permissions, and agents) are ledgerable and may refer to each other. Recursion is allowed wherever such references exist. There is no enforced recursion depth or syntax constraint. Traversal operates recursively by default, unless blocked by agent permissions.

# 4. Illustrative Example

set("ethical\_loops") := {  
 compare("value.freedom", "value.harm"),  
 traverse(agent("Robin")::beliefs where belief.relates\_to("freedom")),  
 compress(above\_result)  
}  
  
This demonstrates an emergent recursive loop: evaluating, traversing, and transforming structures that refer to themselves.