A Practical VM-State Model for Quantum Hyper-Aether: Final-Layer Semantic States, Layer-Wise Weight States, Semantic Routing, Route Entropy, and Minimal Prototype Definition

Hirofumi Inomata Hyper-Aether Laboratory inomata@acm.org

Abstract

This paper presents a practical implementation-oriented formulation of the Quantum Hyper-Aether VM-state model. Previous formulations introduced semantic-cognitive virtual machines, semantic geometry, quantum-inspired semantic computation, semantic entropy, and semantic temperature. However, several elements remained abstract, including the concrete definition of a VM semantic state, the task semantic state, the layer-wise weight state, the VM distance function, and the initial implementation of quantum-like control.

This paper resolves these undefined elements by introducing a minimal working definition. The semantic state of a VM is defined as the final-layer output vector of the VM's internal neural network. The task semantic state is obtained by passing the task input through the same internal neural network of each VM. Thus, each VM evaluates a task within its own learned semantic space. Semantic routing is then defined by cosine similarity between the task semantic state and the current VM semantic state.

The layer-wise weight state is defined as an input-importance vector. Instead of representing the entire weight matrix as a high-dimensional object, the weight state of each layer is compressed into a vector whose components are the norms of the input columns of the layer weight matrix. This representation captures which input dimensions are emphasized by each layer and provides a practical approximation of weight-state evolution.

For the first implementation, the VM distance is simplified to semantic distance only. Quantum-like superposition is approximated by route probability distributions computed from routing scores using a softmax function. Semantic entropy is defined as the entropy of the route probability distribution, and semantic temperature is initially approximated by semantic entropy.

The result is a minimal, implementable version of Quantum Hyper-Aether that does not require physical quantum hardware. Instead, it can be realized as a quantum-inspired semantic routing architecture using ordinary neural networks, cosine similarity, softmax route probabilities, entropy-based uncertainty, and verifier-based safety checking.

1 Introduction

Quantum Hyper-Aether is a semantic-native computational framework in which virtual machines are treated not merely as execution containers, but as semantic-cognitive entities. In this framework, computation is interpreted as semantic state evolution, communication becomes semantic interaction, memory becomes semantic topology, and routing becomes semantic affinity propagation.

Earlier formulations introduced a rich theoretical structure involving semantic Hilbert spaces, semantic graph evolution, quantum-like superposition, semantic entropy, and semantic temperature. However, for a practical prototype, several important elements must be concretely defined.

The main undefined elements were:

- the semantic state of a VM,
- the task semantic state,

- the layer-wise weight state,
- the VM distance function,
- the initial form of the quantum-control state,
- the definitions of semantic entropy and semantic temperature,
- the VM lifecycle rules.

This paper proposes a minimal practical definition that resolves these elements in a way suitable for implementation. The goal is not to implement physical quantum computation, but to construct a quantum-inspired semantic routing architecture using standard neural-network components.

2 Basic VM-State Model

A VM in Quantum Hyper-Aether is represented as a joint state:

$$|\Psi_{\mathrm{VM}}\rangle = |\psi_{\mathrm{VM}}\rangle \otimes |W_1\rangle \otimes \cdots \otimes |W_N\rangle \otimes |M\rangle \otimes |C\rangle \otimes |P\rangle. \tag{1}$$

Here:

- $|\psi_{\mathrm{VM}}\rangle$ is the VM semantic state.
- $|W_l\rangle$ is the weight state of layer l.
- $|M\rangle$ is the memory state.
- $|C\rangle$ is the context state.
- $|P\rangle$ is the purpose or policy state.

For Quantum Hyper-Aether, an additional quantum-control state may be included:

$$|\Psi_{\mathrm{QVM}}\rangle = |\Psi_{\mathrm{VM}}\rangle \otimes |Q\rangle. \tag{2}$$

The purpose of this paper is to provide practical definitions for these components, especially $|\psi_{\mathrm{VM}}\rangle$, $|W_l\rangle$, and $|Q\rangle$.

3 Internal Neural Network of a VM

Let VM j contain an internal neural network F_j with N layers. The layer-wise computation is:

$$\mathbf{x}_j^{(l)} = \sigma_l \left(W_{j,l}\mathbf{x} * j^{(l-1)} + \mathbf{b} * j, l \right), \qquad l = 1, \ldots, N. \tag{3}$$

Here:

- $\mathbf{x} * j^{(0)}$ is the input vector.
- $W * j, l$ is the weight matrix of layer l in VM j.
- $\mathbf{b}_{j,l}$ is the bias vector.
- σ_l is the activation function.
- $\mathbf{x}_j^{(N)}$ is the final-layer output.

The internal neural network F_j is therefore:

$$F_j(\mathbf{x}) = f_{j,N} \circ f_{j,N-1} \circ \cdots \circ f_{j,1}(\mathbf{x}). \tag{4}$$

4 Definition of VM Semantic State

The first key definition is that the semantic state of a VM is the final-layer output of its internal neural network:

$$\mathbf{s}_{\mathrm{VM},j} = \mathbf{x}_j^{(N)}. \tag{5}$$

In quantum-style notation, this is written as:

$$|\psi_{\mathrm{VM},j}\rangle = |\psi_{\mathrm{sem},j}^{(N)}\rangle. \tag{6}$$

Thus, intermediate layers are interpreted as internal semantic transformation stages, while the final layer represents the observable semantic state of the VM.

If the final layer has dimension d_N, then:

$$\mathbf{s}_{\mathrm{VM},j} \in \mathbb{R}^{d_N}. \tag{7}$$

This definition makes the VM semantic state directly measurable and implementable.

5 Definition of Task Semantic State

The task semantic state is obtained by passing the task input through the same internal neural network of each VM. For VM j, let the task input be $\mathbf{x} * \mathrm{task}$. Then:

$$\mathbf{s} * \mathrm{task}^{(j)} = E_{\mathrm{task}}^{(j)}(\mathrm{task}) F_j(\mathbf{x}_{\mathrm{task}}) \tag{8}$$

. This means that each VM evaluates the task in its own learned semantic space.

Since both $\mathbf{s} * \mathrm{task}^{(j)}$ and $\mathbf{s} * \mathrm{VM}, j$ are produced by the same internal network F_j, they belong to the same semantic space:

$$\mathbf{s} * \mathrm{task}^{(j)} \in \mathbb{R}^{d_N}, \qquad \mathbf{s} * \mathrm{VM}, j \in \mathbb{R}^{d_N}. \tag{9}$$

This resolves the problem of comparing a task and a VM state. Each VM locally evaluates whether the task belongs to its own semantic region.

6 Layer-Wise Weight State

For layer l in VM j, assume the weight matrix is:

$$W_{j,l} \in \mathbb{R}^{d_l \times d_{l-1}}, \tag{10}$$

where d_{l-1} is the input dimension of the layer and d_l is the output dimension.

Let $W_{j,l}^{(:,i)}$ denote the i-th input column of the layer weight matrix $W_{j,l}$. Instead of representing the entire weight matrix as the weight state, we define a compressed layer-wise weight state as an input-importance vector:

$$\mathbf{w}_{j,l} = \left[\left| W_{j,l}^{(:,1)} \right| * 2, \left| W * j, l^{(:,2)} \right| * 2, \ldots, \left| W * j, l^{(:,d_{l-1})} \right|_2 \right]. \tag{11}$$

Thus:

$$|W_{j,l}\rangle \equiv \mathbf{w} * j, l \in \mathbb{R}^{d*l-1}. \tag{12}$$

Each component of $\mathbf{w}_{j,l}$ indicates how strongly the layer uses the corresponding input dimension.

A normalized version may be defined as:

$$\tilde{\mathbf{w}}_{j,l} = \frac{\mathbf{w} * j, l}{|\mathbf{w} * j, l|_2 + \epsilon}, \tag{13}$$

where $\epsilon > 0$ prevents division by zero.

This makes the weight state practical, interpretable, and much lower-dimensional than the full weight matrix.

7 Semantic Routing

Semantic routing is based on cosine similarity between the task semantic state and the VM semantic state.

For VM j, define the routing score:

$$R_j = \cos\left(\mathbf{s} * \text{task}^{(j)}, \mathbf{s} * \text{VM}, j\right). \tag{14}$$

Expanding this:

$$R_j = \frac{\mathbf{s} * \text{task}^{(j)} \cdot \mathbf{s} * \text{VM}, j}{\left|\mathbf{s}_{\text{task}}^{(j)}\right| * 2\left|\mathbf{s} * \text{VM}, j\right|_2}. \tag{15}$$

The selected VM is:

$$\text{VM}^* = \arg\max_j R_j. \tag{16}$$

Thus, the first practical version of semantic routing selects the VM whose final-layer semantic state is most similar to the task semantic state as interpreted by that VM.

8 Simplified VM Distance

A general VM distance may be written as:

$$D_{\text{VM}} = aD_{\text{sem}} + bD_{\text{weight}} + cD_{\text{memory}} + dD_{\text{context}} + eD_{\text{purpose}}. \tag{17}$$

However, for the first prototype, we choose:

$$a = 1, \qquad b = c = d = e = 0. \tag{18}$$

Therefore:

$$D_{\text{VM}} = D_{\text{sem}}. \tag{19}$$

The semantic distance is:

$$D_{\text{sem}} = 1 - \cos\left(\mathbf{s} * \text{task}^{(j)}, \mathbf{s} * \text{VM}, j\right). \tag{20}$$

This simplified distance provides a baseline semantic routing model. Weight-state distance, memory distance, context distance, and purpose distance may be introduced later as higher-order correction terms.

9 Route Probability as Quantum-Like Superposition

In the full Quantum Hyper-Aether model, a quantum-control state $|Q\rangle$ may represent superposition, interference, entanglement, and measurement-like selection.

For the first implementation, we approximate $|Q\rangle$ by the route probability distribution:

$$|Q\rangle \approx P_1, P_2, \ldots, P_K, \tag{21}$$

where K is the number of candidate VMs.

The route probability is computed from routing scores using a softmax function:

$$P_j = \frac{\exp(\tau R_j)}{\sum_{k=1}^{K} \exp(\tau R_k)}. \tag{22}$$

Here τ is a sharpness parameter. A large τ makes the route selection more deterministic, while a small τ makes the distribution more uniform.

This gives a practical interpretation:

$$\text{quantum-like route superposition} \approx \text{route probability distribution}. \tag{23}$$

10 Semantic Entropy

Since route probabilities are defined, semantic entropy can be implemented as route entropy:

$$H_{\text{sem}} = -\sum_{j=1}^{K} P_j \log P_j. \tag{24}$$

This entropy measures the uncertainty of VM selection.

If H_{sem} is low, the correct VM is clear. If H_{sem} is high, the system is uncertain about which VM should handle the task.

Thus:

$$H_{\text{sem}} \approx \text{routing uncertainty}. \tag{25}$$

11 Semantic Temperature

In the complete theory, semantic temperature may depend on fluctuations in semantic distance, routing probability, VM states, collapse events, and VM birth/deletion events.

For the first prototype, we use the simplest approximation:

$$T_{\text{sem}} = H_{\text{sem}}. \tag{26}$$

This means:

$$\text{semantic temperature} \approx \text{routing uncertainty}. \tag{27}$$

The interpretation is as follows:

- Low T_{sem} means that a suitable VM is clearly identified.
- Medium T_{sem} means that several VMs are plausible.
- High T_{sem} means that routing is uncertain and the task may not fit existing VMs.

This provides a practical measurement of semantic fluctuation intensity.

12 VM Spawn Rule

A new VM is spawned when no existing VM is sufficiently similar to the task:

$$\max_j R_j < \theta_{\text{spawn}} \quad \Rightarrow \quad \text{spawn a new VM}. \tag{28}$$

Here θ_{spawn} is a threshold.

The interpretation is:

$$\text{if no existing VM matches the task, create a new semantic region}. \tag{29}$$

In the first experiment, θ_{spawn} may be selected empirically. For example, one may start with:

$$0.5 \leq \theta_{\text{spawn}} \leq 0.7. \tag{30}$$

13 VM Deletion or Sleep Rule

VM deletion should be conservative. In the first prototype, unused VMs should not be immediately removed. Instead, they should be moved to a sleep or deletion-candidate state.

Let $U_j(T)$ be the number of times VM j has been selected during the last T cycles. Then:

$$U_j(T) < \theta_{\text{delete}} \quad \Rightarrow \quad \text{mark VM}_j \text{ as a sleep or deletion candidate}. \tag{31}$$

This avoids premature deletion and allows recovery if a temporarily unused VM becomes useful again.

14 Learning Objective

The full autonomous learning objective may be:

$$\mathcal{J} = L_{\text{task}} + \lambda_1 L_{\text{self}} + \lambda_2 L_{\text{consistency}} + \lambda_3 L_{\text{novelty}} + \lambda_4 L_{\text{stability}}. \tag{32}$$

However, for the first prototype, we simplify this to:

$$\mathcal{J} = L_{\text{task}}. \tag{33}$$

Thus, the first implementation uses task loss only. Self-evaluation, consistency, novelty, and stability terms can be added later.

15 Semantic Verification

The first prototype should include a simple verification pipeline:

$$\text{VM output} \rightarrow \text{LLM verifier} \rightarrow \text{Constraint DB} \rightarrow \text{Accept, Reject, or Revise.} \tag{34}$$

The LLM verifier checks whether the output is semantically consistent with the task. The constraint database checks whether the output violates known rules, policies, or system constraints.

This provides a practical initial implementation of semantic verification without requiring a full multi-VM consensus system.

16 Minimal Prototype Algorithm

The initial prototype can be described as follows.

1. Receive task input $\mathbf{x}_{\text{task}}$.
   ‘‘‘
2. For each VM j, compute:
$$\mathbf{s}_{\text{task}}^{(j)} = F_j(\mathbf{x}_{\text{task}}). \tag{35}$$
3. Compute the routing score:
$$R_j = \cos\left(\mathbf{s}_{\text{task}}^{(j)}, \mathbf{s}_{\text{VM},j}\right). \tag{36}$$
4. Convert routing scores into route probabilities:
$$P_j = \frac{\exp(\tau R_j)}{\sum_{k=1}^{K} \exp(\tau R_k)}. \tag{37}$$
5. Compute semantic entropy:
$$H_{\text{sem}} = -\sum_j P_j \log P_j. \tag{38}$$
6. Set semantic temperature:
$$T_{\text{sem}} = H_{\text{sem}}. \tag{39}$$
7. If $\max_j R_j < \theta_{\text{spawn}}$, spawn a new VM.
8. Otherwise select:
$$\text{VM}^* = \arg\max_j R_j. \tag{40}$$

9. Execute the task on VM*.
10. Verify the result using an LLM verifier and a constraint database.
11. Update the selected VM state and usage statistics.
12. Mark unused VMs as sleep or deletion candidates when necessary. ‘‘‘

17 Interpretation

The above definitions show that Quantum Hyper-Aether can be implemented initially without physical quantum hardware.

The initial practical interpretation of Quantum Hyper-Aether concepts is as follows:

- Semantic state is implemented as the final-layer output vector of the VM internal neural network.
  ‘‘‘
- Task semantic state is implemented as the output obtained by passing the task input through the same VM internal neural network.
- Weight state is implemented as the layer-wise input-importance vector computed from the column norms of the layer weight matrix.
- Superposition is approximated by the route probability distribution over candidate VMs.
- Measurement is implemented as the selection of the VM with the highest routing score.
- Collapse is interpreted as the final route decision and the verification result.
- Semantic entropy is implemented as the entropy of the route probability distribution.
- Semantic temperature is approximated by route entropy.
- VM birth occurs when no existing VM sufficiently matches the task.
- VM deletion is implemented conservatively as transition to a sleep or deletion-candidate state after long-term non-selection. ‘‘‘

Thus, the first implementation of Quantum Hyper-Aether can be interpreted as:

$$\text{quantum-inspired semantic routing architecture} \tag{41}$$

rather than a physical quantum computer.

18 New Results from Resolving Undefined Elements

By resolving the previously undefined elements, several results become clear.

First, the VM semantic state is measurable because it is defined as the final-layer output:

$$\mathbf{s}_{\mathrm{VM},j} = \mathbf{x}_j^{(N)}. \tag{42}$$

Second, task and VM states are comparable because the task is encoded using the same internal VM network:

$$\mathbf{s}_{\mathrm{task}}^{(j)} = F_j(\mathbf{x}_{\mathrm{task}}). \tag{43}$$

Third, semantic routing becomes a concrete computation:

$$R_j = \cos\left(F_j(\mathbf{x} * \text{task}), \mathbf{s} * \text{VM}, j\right). \tag{44}$$

Fourth, the quantum-like control state can be approximated by a route probability distribution:

$$|Q\rangle \approx P_1, \ldots, P_K. \tag{45}$$

Fifth, semantic entropy and semantic temperature become measurable through route uncertainty:

$$H_{\text{sem}} = -\sum_j P_j \log P_j, \qquad T_{\text{sem}} = H_{\text{sem}}. \tag{46}$$

Therefore, the proposed model moves Quantum Hyper-Aether from an abstract semantic-quantum theory toward a minimal implementable architecture.

19 Advantages

The proposed minimal model provides the following advantages:

- It defines the VM semantic state as a measurable vector.
  “‘
- It avoids the need for a separate task encoder.
- It allows each VM to evaluate tasks in its own learned semantic space.
- It defines semantic routing using cosine similarity.
- It compresses weight states into interpretable input-importance vectors.
- It approximates quantum-like superposition using route probabilities.
- It defines semantic entropy using route uncertainty.
- It approximates semantic temperature using semantic entropy.
- It provides simple VM spawn and sleep or deletion rules.
- It is implementable using ordinary neural networks and standard software. “‘

20 Limitations

The first prototype is intentionally simplified. Several limitations remain:

- The VM distance uses only semantic distance.
  “‘
- Memory, context, and purpose distances are ignored in the first version.
- The quantum-control state is approximated by route probabilities only.
- Semantic temperature is approximated by entropy only.
- The learning objective uses task loss only.
- Verification relies on an LLM verifier and a constraint database.
- Thresholds such as θ_{spawn} and θ_{delete} must be empirically calibrated. “‘

These limitations are acceptable for the first implementation because the purpose is to establish a minimal working system.

21 Future Work

Future work should extend the model in the following directions:

- Include weight-state distance in D_{VM}.
  ‘‘‘
- Include memory distance and context distance.
- Add purpose alignment to routing.
- Introduce a richer quantum-control state $|Q\rangle$.
- Implement interference-like candidate score adjustment.
- Add multi-VM consensus verification.
- Calibrate semantic temperature regions.
- Evaluate VM spawn and deletion dynamics.
- Build simulation benchmarks.
- Compare against conventional routing and agent-selection methods. ‘‘‘

22 Conclusion

This paper proposed a practical VM-state model for Quantum Hyper-Aether. The main result is that the previously abstract semantic state can be concretely defined as the final-layer output of a VM's internal neural network. The task semantic state is obtained by passing the task through the same VM internal neural network. Semantic routing is then implemented by cosine similarity between the task semantic state and the VM semantic state.

The layer-wise weight state is defined as an input-importance vector computed from the column norms of the layer weight matrix. The VM distance is initially simplified to semantic distance only. The quantum-control state is approximated by route probabilities. Semantic entropy is defined as route entropy, and semantic temperature is approximated by semantic entropy.

These definitions show that the first implementation of Quantum Hyper-Aether does not require physical quantum hardware. It can be implemented as a quantum-inspired semantic routing architecture based on ordinary neural networks, vector similarity, softmax route probabilities, entropy-based uncertainty, simple VM lifecycle rules, and verifier-based safety checking.

The resulting framework provides a minimal working foundation for future AI-native semantic computing systems, distributed cognitive VMs, adaptive semantic routing, and practical Quantum Hyper-Aether prototypes.

References

[1] John von Neumann, *Mathematical Foundations of Quantum Mechanics*, Princeton University Press, 1955.

[2] Richard P. Feynman, “Space-Time Approach to Non-Relativistic Quantum Mechanics,” *Reviews of Modern Physics*, 1948.

[3] Claude E. Shannon, “A Mathematical Theory of Communication,” *Bell System Technical Journal*, 1948.

[4] E. T. Jaynes, "Information Theory and Statistical Mechanics," *Physical Review*, 1957.

[5] Geoffrey Hinton, "Learning Distributed Representations of Concepts," *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, 1986.

[6] Ashish Vaswani et al., "Attention Is All You Need," *NeurIPS*, 2017.

[7] Michael A. Nielsen and Isaac L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2000.

[8] Ilya Prigogine and Isabelle Stengers, *Order Out of Chaos*, Bantam Books, 1984.