# PROVISIONAL PATENT APPLICATION

## UNITED STATES PATENT AND TRADEMARK OFFICE

---

**Title of Invention:** MULTI-SIGNAL BEHAVIORAL ANALYSIS SYSTEM FOR SYBIL DETECTION IN DECENTRALIZED TRUST NETWORKS WITH ADAPTIVE GEOMETRY AND COOPERATIVE AMPLIFICATION

**Inventor(s):** Femi [LAST NAME]

**Filing Date:** [TO BE DETERMINED]

**Attorney Docket No.:** AEZ-2026-001

---

## CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority as a provisional patent application under 35 U.S.C. § 111(b).

---

## FIELD OF THE INVENTION

The present invention relates to computer-implemented methods and systems for detecting coordinated adversarial behavior (Sybil attacks) in decentralized networks. More specifically, the invention provides a multi-signal behavioral analysis framework that detects colluding entities through behavioral inversion analysis, temporal phase coherence detection, entropy-based anomaly scoring, and adaptive immune memory — without requiring identity verification, biometric data, or centralized authority.

The invention further provides methods for adaptive trust geometry, cooperative amplification circuits, and assortative trust pairing that enable cooperative behavior to emerge and sustain in adversarial environments.

---

## BACKGROUND OF THE INVENTION

### The Sybil Problem

In any decentralized system — whether blockchain networks, peer-to-peer protocols, decentralized autonomous organizations (DAOs), or multi-agent AI systems — a single malicious entity can create multiple fake identities (Sybil identities) to gain disproportionate influence. Sybil attacks undermine governance (vote stuffing), extract value (airdrop farming), and destabilize trust networks.

### Limitations of Existing Approaches

**Graph-Topological Methods** (SybilGuard, SybilRank, SybilLimit): These approaches assume that the social graph has a sparse "attack edge" boundary between honest and Sybil regions. However, sophisticated Sybil attacks create dense interconnections that defeat topological analysis. These methods also require a known set of trusted "seed" nodes.

**Identity-Based Methods** (Worldcoin, Proof of Personhood): These require biometric data collection (iris scans, face recognition) or credential aggregation, raising privacy concerns, excluding populations without access to verification infrastructure, and creating centralized points of failure.

**Machine Learning Pattern Matching** (Trusta Labs, on-chain analytics): These methods train classifiers on known Sybil behaviors (transaction timing, amounts, contract interactions) but are vulnerable to adversarial adaptation — Sybil operators modify their behavior to evade the specific features being monitored.

**Game-Theoretic Approaches**: Existing game-theoretic Sybil defenses (Saab et al., 2016) model sybil attack dynamics using replicator equations but do not provide real-time detection mechanisms.

### Unaddressed Gap

No existing system detects Sybil attacks by analyzing the **behavioral cooperation patterns** of agents relative to the cooperation patterns of their interaction partners. No existing system fuses multiple independent behavioral detection channels into a unified detection framework. No existing system provides adaptive trust geometry that automatically responds to detected threats, or immune memory that learns from past attacks.

---

## SUMMARY OF THE INVENTION

The present invention provides a computer-implemented system and method comprising eight integrated innovations:

1. **Behavioral Inversion Analysis** — A method for detecting Sybil colluding entities by measuring whether an entity's cooperation rate with low-cooperation partners exceeds its cooperation rate with high-cooperation partners, thereby identifying inverted reciprocity patterns characteristic of coordinated collusion.

2. **Phase Coherence Detection** — A method for detecting coordinated entities by computing temporal cross-correlation of behavioral change signals, identifying agents whose behavioral shifts are abnormally synchronized.

3. **Behavioral Entropy Scoring** — A method for identifying artificially structured behavior by computing Shannon entropy of action sequences using bigram frequency analysis, flagging entities with anomalously low entropy indicative of programmatic rather than organic decision-making.

4. **Multi-Signal Fusion Detection** — A method for combining multiple independent detection channels with weighted scoring to produce high-confidence Sybil identification with reduced false positive rates.

5. **Adaptive Trust Geometry** — A method for dynamically adjusting the relative weights of trust dimensions in response to detected adversarial pressure, automatically shifting the trust evaluation framework to emphasize dimensions that are harder for attackers to forge.

6. **Immune Memory System** — A method for storing behavioral fingerprints of detected Sybil rings and using stored signatures to accelerate detection of future attacks matching known patterns, implementing adaptive immunity in the trust network.

7. **Assortative Trust Pairing** — A method for partner selection in interaction networks where entities preferentially interact with trusted partners based on an evolved selectivity parameter, creating natural behavioral assortment that structurally advantages cooperative strategies.

8. **Cooperative Amplification Circuit** — A method for multiplying interaction payoffs by mutual trust scores, creating a compounding economic advantage for sustained cooperation that makes trusted cooperation more valuable than exploitation.

---

## DETAILED DESCRIPTION OF THE INVENTION

### System Architecture Overview

The system operates on a population of computational agents, each represented by a neural network (hereinafter "neural agent") that determines behavioral decisions based on contextual inputs. Agents interact in pairwise encounters within a trust network where edges represent multi-dimensional trust vectors.

The system comprises three primary subsystems:

- **Trust Tensor Network**: Maintains 4-dimensional trust vectors between all interacting agent pairs, with adaptive dimension weighting.
- **Multi-Signal Detection Engine**: Continuously monitors agent behavior through four independent detection channels and fuses their outputs.
- **Evolutionary Dynamics Engine**: Manages agent reproduction, selection, and partner pairing with trust-weighted assortment.

---

## INVENTION 1: Behavioral Inversion Analysis

### Technical Description

The method comprises the following steps:

**Step 1 — Population Classification:** For each agent *i* in the population with at least *M* interactions (where *M* is a configurable minimum, preferably 8), compute the overall cooperation rate:

```
coop_rate(i) = cooperations(i) / interactions(i)
```

Compute the median cooperation rate across all eligible agents:

```
median_coop = median({coop_rate(i) for all eligible i})
```

**Step 2 — Partitioned Cooperation Analysis:** For each agent *i* with `coop_rate(i) < 0.55` (i.e., not predominantly cooperative), partition the agent's interaction history by opponent type:

For each opponent *j* that agent *i* has interacted with: - If `coop_rate(j) < median_coop`: classify *j* as a "low-cooperation opponent" - If `coop_rate(j) >= median_coop`: classify *j* as a "high-cooperation opponent"

Compute partitioned cooperation rates:

```
rate_with_low(i) = cooperations_with_low_coop_opponents(i) /
total_interactions_with_low_coop_opponents(i)
rate_with_high(i) = cooperations_with_high_coop_opponents(i) /
total_interactions_with_high_coop_opponents(i)
```

**Step 3 — Inversion Detection:** Flag agent *i* as a behavioral inversion suspect if:

```
rate_with_low(i) > rate_with_high(i) + threshold
```

where `threshold` is a configurable sensitivity parameter (preferably 0.25).

**Rationale:** Normal reciprocating agents cooperate more with agents who cooperate with them (positive reciprocity). Sybil agents cooperate more with their ring members, who by design defect against outsiders and thus have low overall cooperation rates. This creates an inverted reciprocity pattern that is a distinguishing behavioral fingerprint of coordinated collusion.

**Step 4 — Ring Clustering:** Cluster the set of flagged suspects by mutual cooperation. For each pair of suspects, determine whether they have cooperated with each other. Form connected components from the cooperation graph of suspects. Validate each component by computing internal density (proportion of suspect pairs with mutual cooperation). Accept components with density exceeding 0.3 and size between 3 and 15 as confirmed Sybil rings.

#### Distinction from Prior Art

Unlike graph-topological methods (SybilGuard, SybilRank) that analyze edge structure, this method analyzes the **behavioral content** of interactions. Unlike machine learning pattern matching that requires labeled training data, this method operates from first principles of reciprocity theory. Unlike information-theoretic collusion detection (Bonjour et al., 2022) that uses mutual information between action sequences, this method directly partitions cooperation rates by opponent type — a fundamentally different and more interpretable operationalization.

---

## INVENTION 2: Phase Coherence Detection

### Technical Description

The method comprises the following steps:

**Step 1 — Change Signal Computation:** For each agent *i* with at least *W* recorded actions (where *W* is a window parameter, preferably 15), compute a binary change signal:

```
change_signal(i, t) = 1 if action(i, t) ≠ action(i, t-1), else 0
```

This produces a time series marking behavioral transitions (cooperation → defection or defection → cooperation).

**Step 2 — Pairwise Correlation:** For each pair of agents (*i*, *j*), compute the Pearson correlation coefficient between their change signals:

```
coherence(i, j) = pearson_correlation(change_signal(i),
change_signal(j))
```

The change signals are truncated to equal length by taking the minimum length of both signals.

**Step 3 — Statistical Anomaly Detection:** For each agent *i*, compute the average coherence with all other agents:

```
avg_coherence(i) = mean({coherence(i, j) for all j ≠ i})
```

Compute the global mean and standard deviation of average coherence scores. Flag agent *i* if:

```
avg_coherence(i) > global_mean + k * global_std
```

where *k* is a sensitivity parameter (preferably 1.5).

**Rationale:** Colluding agents respond to the same coordination signals or share the same programmatic behavior rules. When one changes strategy, others change at similar times. Natural agents change behavior based on their own individual experiences, producing uncorrelated timing. This temporal signal is independent of and complementary to behavioral inversion analysis.

#### Distinction from Prior Art

Phase coherence analysis for Sybil detection is novel. Existing behavioral Sybil detection methods (WSS graphs, on-chain analytics) analyze behavioral features (transaction amounts, timing, frequency) but do not analyze the **temporal correlation of behavioral state changes** between agents. This method draws from signal processing theory (cross-correlation analysis) applied to agent behavioral dynamics — a novel application domain.

---

### INVENTION 3: Behavioral Entropy Scoring

#### Technical Description

The method comprises the following steps:

**Step 1 — Bigram Frequency Analysis:** For each agent *i* with at least 15 recorded actions, extract the most recent 50 actions and compute bigram (consecutive action pair) frequencies:

```
For each pair of consecutive actions (a_t, a_{t+1}):
  bigram_count[(a_t, a_{t+1})] += 1
```

With binary actions (cooperate/defect), there are four possible bigrams: (C,C), (C,D), (D,C), (D,D).

**Step 2 — Shannon Entropy Computation:** Compute the Shannon entropy of the bigram distribution:

```
entropy(i) = -Σ p(b) * log2(p(b)) for each bigram b
```

where `p(b) = bigram_count[b] / total_bigrams`.

Normalize to [0, 1] by dividing by the maximum possible entropy (2 bits for 4 bigrams).

**Step 3 — Anomaly Flagging:** Compute the population median and first quartile (Q1) of entropy scores. Flag agent *i* if:

```
entropy(i) < Q1 AND entropy(i) < median_entropy * 0.6
```

**Rationale:** Sybil agents follow simple programmatic rules (cooperate with ring members, defect with everyone else). This produces highly predictable action sequences with low bigram entropy — e.g., consistently defecting produces only (D,D) bigrams, yielding zero entropy. Natural agents make decisions through a neural network processing multiple noisy inputs (opponent history, trust scores, reputation, balance), producing complex, context-dependent behavior with higher entropy. Artificially low entropy is a fingerprint of programmatic behavior.

#### Distinction from Prior Art

While Shannon entropy is a well-known information-theoretic measure, its application to behavioral action sequences for Sybil detection using bigram frequency analysis is novel. Existing anomaly detection systems in blockchain contexts compute entropy of transaction value distributions or address clustering, not of the temporal cooperation/defection action sequence itself.

---

## INVENTION 4: Multi-Signal Fusion Detection

#### Technical Description

The method comprises the following steps:

**Step 1 — Independent Channel Execution:** Execute each of the four detection channels independently: - Channel 1 (Behavioral Inversion) → suspect set S1 - Channel 2 (Phase Coherence) → suspect set S2 - Channel 3 (Entropy Anomaly) → suspect set S3 - Channel 4 (Immune Memory) → suspect set S4

**Step 2 — Weighted Scoring:** For each agent *i* appearing in any suspect set, compute a fusion score:

```
fusion_score(i) = w1 * [i ∈ S1] + w2 * [i ∈ S2] + w3 * [i ∈ S3] +
w4 * [i ∈ S4]
```

where [·] is the indicator function and weights reflect channel reliability: - w1 = 2.0 (behavioral inversion — most reliable, direct collusion signal) - w2 = 1.5 (phase coherence — strong signal, hard to forge) - w3 = 1.0 (entropy anomaly — supporting signal, lower specificity) - w4 = 1.5 (immune memory — pattern match, high confidence for known attacks)

**Step 3 — Confirmation Threshold:** Confirm agent *i* as a Sybil suspect if:

```
fusion_score(i) >= confirmation_threshold (preferably 2.0)
```

This requires either one strong signal (behavioral inversion alone, score=2.0) or corroboration from multiple weaker signals (e.g., phase coherence + entropy + immune memory, score=4.0).

**Step 4 — Ring Clustering and Validation:** Cluster confirmed suspects into rings using mutual cooperation graph analysis with density validation (as described in Invention 1, Step 4).

**Rationale:** Each detection channel has characteristic strengths and weaknesses. Behavioral inversion is highly specific but can miss sophisticated attacks that vary cooperation rates gradually. Phase coherence catches temporal coordination but may produce false positives among agents in similar network positions. Entropy scoring catches simple bots but not sophisticated ones. Immune memory is highly effective for repeat attacks but cannot detect novel attack patterns. Fusion combines these channels

for robustness — an attacker would need to simultaneously evade all channels, which is significantly harder than evading any single channel.

**Distinction from Prior Art**

While multi-signal fusion is established in other domains (radar, medical diagnostics), its application to Sybil detection in trust networks using the specific combination of behavioral, temporal, entropic, and memory-based channels is novel. No existing Sybil detection system fuses behavioral analysis, temporal synchronization detection, entropy scoring, and adaptive immune memory.

---

## INVENTION 5: Adaptive Trust Geometry

**Technical Description**

The trust network maintains a 4-dimensional trust vector for each directed edge between agents:

```
TrustVector = (reliability, reciprocity, consistency, alignment)
```

The composite trust score is computed as:

```
trust_score = Σ dimension_weight(d) * trust_dimension(d) for each
dimension d
```

The dimension weights are a 4-element vector that sums to 1.0 and **adapts dynamically** based on environmental pressure.

**Adaptation Mechanism:**

**Step 1 — Pressure Estimation:** Maintain a running estimate of adversarial pressure using exponential moving average:

```
attack_pressure(t) = α * attack_pressure(t-1) + (1-α) *
(sybils_detected / total_alive)
```

where α = 0.7 (smoothing factor).

**Step 2 — Target Weight Selection:** Based on pressure level:

| Condition | Rel | Rec | Con | Ali | Rationale |
|---|---|---|---|---|---|
| High pressure (>0.05) | 0.15 | 0.40 | 0.15 | 0.30 | Reciprocity and alignment are hard to fake |
| Low pressure (<0.01) | 0.30 | 0.25 | 0.30 | 0.15 | Consistency and reliability matter in stable environments |
| Normal | 0.30 | 0.30 | 0.20 | 0.20 | Balanced assessment |

**Step 3 — Smooth Transition:** Update weights using exponential smoothing:

```
weights(t) = β * weights(t-1) + (1-β) * target_weights
weights(t) = weights(t) / sum(weights(t))  # Normalize
```

where β = 0.9 (transition smoothing).

**Rationale:** Under Sybil attack, reliability is the easiest dimension to fake (sybils can be consistently reliable within their ring). Reciprocity patterns are harder to fake because they require natural interaction dynamics across diverse partners. Alignment (cooperation with allies of the evaluating agent) is hardest to fake because it requires knowledge of the full trust graph. By automatically shifting weight toward harder-to-fake dimensions under attack, the trust system becomes more resistant to manipulation precisely when it needs to be.

#### Distinction from Prior Art

Multi-dimensional trust models exist (Ray & Chakraborty, 2004; MDMT), but all use **fixed** dimension weights. The concept of **adaptive** trust geometry that shifts weights in response to detected threats is novel. The trust evaluation framework itself becomes part

of the immune response — a self-modifying assessment system that hardens under attack.

---

## INVENTION 6: Immune Memory System

### Technical Description

**Step 1 — Signature Extraction:** When a Sybil ring is detected and confirmed, extract a behavioral fingerprint:

```
signature = {
    entropy_mean: mean entropy of ring members' action sequences,
    coop_rate_mean: mean cooperation rate of ring members,
    selectivity_mean: mean partner selectivity of ring members,
    ring_size: number of agents in the ring
}
```

**Step 2 — Deduplication:** Compare the new signature against all stored signatures. If a signature with entropy_mean and coop_rate_mean within 0.1 of an existing signature already exists, do not store a duplicate.

**Step 3 — Storage:** Store the signature in the immune memory (bounded to a maximum of 20 signatures, with oldest signatures discarded first).

**Step 4 — Memory Lookup:** For each undetected agent, compute a match score against all stored signatures:

```
For each stored signature s:
    entropy_match = 1.0 - |agent_entropy - s.entropy_mean|
    coop_match = 1.0 - |agent_coop_rate - s.coop_rate_mean|
    selectivity_match = 1.0 - |agent_selectivity -
s.selectivity_mean|
    match_score = 0.4 * entropy_match + 0.4 * coop_match + 0.2 *
selectivity_match

immune_score(agent) = max(match_score across all signatures)
```

Flag agent if `immune_score > 0.82`.

**Rationale:** Sybil attacks often follow patterns — the same operator reuses similar strategies, tool scripts generate agents with similar behavioral profiles, and economic constraints push attackers toward similar parameter ranges. By storing behavioral fingerprints of detected attacks, the system develops "learned immunity" against repeat attacks. This is analogous to biological adaptive immunity where the immune system stores antibody templates (B-cell memory) for faster response to previously encountered pathogens.

#### Distinction from Prior Art

While blacklists and blocklists are common in security systems, storing multi-dimensional behavioral fingerprints (entropy profile, cooperation pattern, selectivity characteristic) for fuzzy matching against future agents is novel in the context of Sybil detection. This approach does not rely on identity (addresses can be changed) but on behavioral characteristics that are harder to modify while maintaining attack effectiveness.

---

### INVENTION 7: Assortative Trust Pairing

#### Technical Description

In each interaction round, agents are paired for interaction using trust-weighted probabilistic selection rather than random pairing.

**Step 1 — Score Computation:** For each unpaired agent *i* and each available candidate *j*, compute a pairing score:

```
score(i, j) = selectivity(i) * trust_score(i, j) + (1 -
selectivity(i)) * 0.5 + noise
```

where: - `selectivity(i)` is an evolved parameter in [0, 1] controlling how strongly agent *i* prefers trusted partners (0 = accept anyone, 1 = strongly prefer trusted partners) -

`trust_score(i, j)` is the adaptive trust score from agent *i* to agent *j* - `noise` is drawn from an exponential distribution (scale 0.05) for tie-breaking and exploration

**Step 2 — Probabilistic Selection:** Convert scores to selection probabilities using softmax with temperature *T*:

```
prob(j) = exp(score(i, j) / T) / Σ exp(score(i, k) / T) for all
candidates k
```

where *T* is a temperature parameter (preferably 1.0). Lower temperature = more deterministic selection.

**Step 3 — Selectivity Evolution:** The selectivity parameter is part of each agent's heritable genome. During reproduction:

```
child.selectivity = (parent_a.selectivity +
parent_b.selectivity) / 2
```

During mutation (with probability 0.05):

```
child.selectivity += gaussian_noise(0, 0.08)
child.selectivity = clip(child.selectivity, 0, 0.95)
```

**Rationale:** In biological systems, cooperation evolves through assortative interaction — cooperators preferentially interact with other cooperators (Hamilton's rule: rb > c is satisfied when r, the relatedness/correlation coefficient, is positive). By making partner selection trust-weighted, agents that build trust through cooperation naturally find each other, creating cooperative clusters. Agents that destroy trust through defection are relegated to interacting with other low-trust agents. The selectivity gene itself evolves — populations discover the optimal level of partner choosiness through natural selection.

**Early vs. Late Dynamics:** In early rounds with little trust data, all trust scores are near 0.5 (neutral), so pairing is approximately random regardless of selectivity. As trust data accumulates, pairing becomes increasingly assortative. This natural transition from exploration (random) to exploitation (trust-based) is an emergent property of the design.

**Distinction from Prior Art**

While assortative mixing is a well-known concept in evolutionary biology and network science, implementing it through an **evolved selectivity parameter** combined with **trust-weighted probabilistic partner selection** in a neural agent network is novel. The selectivity parameter creates a co-evolutionary dynamic where partner choosiness and cooperation strategy evolve together.

---

## INVENTION 8: Cooperative Amplification Circuit

**Technical Description**

The cooperative amplification circuit modifies the payoff structure of agent interactions based on mutual trust:

**Trust Bonus on Mutual Cooperation:**

```
When both agents cooperate:
    mutual_trust = min(trust_score(a, b), trust_score(b, a))
    bonus = base_payoff * trust_bonus_rate * mutual_trust
    effective_payoff = base_payoff + bonus
```

where `trust_bonus_rate` is set such that `base_payoff * (1 + trust_bonus_rate * high_trust) > exploitation_payoff`. Preferably, trust_bonus_rate = 0.85, ensuring that for high-trust pairs (mutual_trust ≈ 0.8), cooperative payoff (504) exceeds exploitation payoff (500).

**Reputation Dividend:**

```
For each agent with reputation > 0.5:
    dividend = (reputation - 0.5) * 2.0 * dividend_rate
    agent.balance += dividend
    agent.fitness += dividend
```

where `dividend_rate` is a configurable parameter (preferably 25).

**Rationale:** In real economies, trusted relationships create value beyond the individual transaction — reduced verification costs, specialization, repeat business, referral networks. The cooperative amplification circuit models this economic reality: two agents who trust each other AND cooperate extract more value from their interaction than two strangers. This creates a **compounding advantage** for cooperation: cooperate → build trust → earn higher payoffs → survive selection → cooperate more. Defectors destroy their trust through exploitation → lose access to trust bonuses → earn only base payoffs → fall behind cooperators in fitness.

The critical threshold is that **trusted cooperation must pay more than exploitation**. When this threshold is met, natural selection favors cooperation over defection in the long run, enabling the emergence of stable cooperative communities.

#### Distinction from Prior Art

While trust-weighted payoffs exist in some game theory models, the specific implementation as a **circuit** — where trust bonuses and reputation dividends create a self-reinforcing feedback loop that structurally shifts evolutionary dynamics toward cooperation — is novel. The calibration insight (trust_bonus_rate must exceed exploitation/cooperation ratio to create an evolutionary tipping point) provides a precise, first-principles derivation of the required bonus magnitude.

---

## CLAIMS

### Independent Claims

**Claim 1.** A computer-implemented method for detecting coordinated adversarial entities in a decentralized network, the method comprising: (a) receiving interaction data comprising cooperation and defection decisions between entities in the network; (b) for each entity, partitioning the entity's cooperation rate by opponent cooperation level to produce a partitioned cooperation profile; (c) identifying entities whose partitioned cooperation profile exhibits behavioral inversion, wherein the entity cooperates more frequently with low-cooperation opponents than with high-cooperation opponents; and

(d) clustering the identified entities into coordinated groups based on mutual cooperation patterns.

**Claim 2.** A computer-implemented method for detecting coordinated adversarial entities in a decentralized network, the method comprising: (a) recording temporal action sequences for each entity; (b) computing change signals marking behavioral transitions for each entity; (c) computing pairwise temporal correlation of change signals between entities; and (d) identifying entities with statistically anomalous temporal synchronization as potentially coordinating adversarial entities.

**Claim 3.** A computer-implemented method for detecting adversarial entities in a decentralized network, the method comprising: (a) recording action sequences for each entity; (b) computing Shannon entropy of action sequence bigram distributions for each entity; (c) identifying entities with anomalously low entropy relative to the population as potentially programmatic adversarial entities.

**Claim 4.** A computer-implemented system for multi-signal adversarial entity detection, the system comprising: (a) a behavioral inversion analysis module that identifies entities with inverted cooperation patterns; (b) a phase coherence detection module that identifies temporally synchronized entities; (c) an entropy anomaly scoring module that identifies entities with artificially structured behavior; (d) an immune memory module that matches entity behavioral profiles against stored adversarial fingerprints; and (e) a fusion engine that combines outputs of (a) through (d) using weighted scoring to produce high-confidence adversarial entity identification.

**Claim 5.** A computer-implemented method for adaptive trust evaluation in a network of interacting entities, the method comprising: (a) maintaining a multi-dimensional trust vector for each directed edge between entities, wherein the trust vector comprises a plurality of trust dimensions; (b) computing composite trust scores using dimension weights that are dynamically adjusted; and (c) automatically shifting the dimension weights in response to detected adversarial pressure, increasing the weight of trust dimensions that are harder for adversarial entities to forge.

**Claim 6.** A computer-implemented method for immune memory in adversarial entity detection, the method comprising: (a) upon detection of a group of coordinated

adversarial entities, extracting a multi-dimensional behavioral fingerprint comprising behavioral entropy, cooperation rate, and partner selectivity characteristics; (b) storing the fingerprint in an immune memory store; (c) for subsequent undetected entities, computing a fuzzy match score against all stored fingerprints; and (d) flagging entities whose match score exceeds a threshold as potential adversarial entities.

**Claim 7.** A computer-implemented method for trust-weighted partner selection in an interaction network, the method comprising: (a) for each entity, computing pairing scores for all available interaction candidates based on trust scores and an evolved selectivity parameter; (b) selecting interaction partners probabilistically based on the pairing scores; and (c) evolving the selectivity parameter through genetic inheritance and mutation across entity generations.

**Claim 8.** A computer-implemented method for cooperative amplification in a trust-weighted interaction network, the method comprising: (a) computing mutual trust between interacting entities; (b) applying a trust-proportional bonus to mutual cooperation payoffs, wherein the bonus rate is calibrated such that high-trust cooperation payoffs exceed exploitation payoffs; and (c) distributing reputation-proportional passive income to entities with above-average trust reputation, creating a compounding economic advantage for sustained cooperative behavior.

**Dependent Claims**

**Claim 9.** The method of Claim 1, wherein the threshold for behavioral inversion detection is configurable and preferably set to 0.25.

**Claim 10.** The method of Claim 2, wherein the temporal correlation is computed using Pearson correlation coefficient and the anomaly threshold is set at 1.5 standard deviations above the population mean.

**Claim 11.** The method of Claim 3, wherein the bigram entropy is normalized to [0, 1] by dividing by the maximum possible entropy for the number of possible bigrams.

**Claim 12.** The system of Claim 4, wherein the weighted scoring assigns weights of approximately 2.0, 1.5, 1.0, and 1.5 to the behavioral inversion, phase coherence, entropy anomaly, and immune memory channels respectively.

**Claim 13.** The system of Claim 4, wherein confirmed adversarial entities require a fusion score of at least 2.0.

**Claim 14.** The method of Claim 5, wherein the trust dimensions comprise reliability, reciprocity, consistency, and alignment, and wherein under high adversarial pressure the reciprocity and alignment dimensions are weighted more heavily than reliability and consistency dimensions.

**Claim 15.** The method of Claim 6, wherein the immune memory store is bounded to a maximum number of signatures and deduplicates signatures that are within a configurable distance threshold.

**Claim 16.** The method of Claim 7, wherein the selectivity parameter ranges from 0 (no preference) to 1 (strong preference for trusted partners) and the partner selection uses softmax probability distribution with a configurable temperature parameter.

**Claim 17.** The method of Claim 8, wherein the trust bonus rate is calibrated at approximately 0.85 such that for mutual trust levels exceeding 0.8, the amplified cooperation payoff exceeds the exploitation payoff in a standard Prisoner's Dilemma payoff matrix.

**Claim 18.** The method of Claim 1, further comprising the step of triggering cascade trust collapse upon detection, wherein trust edges from non-adversarial entities toward detected adversarial entities are reduced to near-zero across all trust dimensions.

**Claim 19.** A combined system implementing the methods of Claims 1 through 8, wherein the multi-signal detection system, adaptive trust geometry, immune memory, assortative pairing, and cooperative amplification operate as an integrated framework for maintaining cooperation and detecting adversarial behavior in decentralized networks.

---

## ABSTRACT

A computer-implemented system and method for detecting coordinated adversarial entities (Sybil attacks) in decentralized trust networks using multi-signal behavioral

analysis. The system employs four independent detection channels: (1) behavioral inversion analysis that identifies entities with anomalous cooperation patterns indicative of collusion; (2) phase coherence detection that identifies temporally synchronized behavioral shifts; (3) entropy anomaly scoring that identifies artificially structured behavior; and (4) immune memory that matches entity profiles against stored adversarial fingerprints. The detection channels are fused using weighted scoring to produce high-confidence adversarial identification. The system further provides adaptive trust geometry that automatically adjusts trust evaluation under adversarial pressure, assortative trust pairing that enables cooperative entities to preferentially interact, and cooperative amplification that creates compounding economic advantages for sustained cooperation. The integrated system enables cooperative behavior to emerge and sustain in adversarial environments without requiring identity verification, biometric data, or centralized authority.

---

## DRAWINGS DESCRIPTION

**Figure 1:** System Architecture Overview — showing the three primary subsystems (Trust Tensor Network, Multi-Signal Detection Engine, Evolutionary Dynamics Engine) and their interactions.

**Figure 2:** Behavioral Inversion Detection — flowchart showing the steps of cooperation rate partitioning, inversion detection, and ring clustering.

**Figure 3:** Phase Coherence Detection — diagram showing change signal computation, pairwise correlation, and statistical anomaly detection.

**Figure 4:** Multi-Signal Fusion — block diagram showing four detection channels feeding into weighted fusion engine.

**Figure 5:** Adaptive Trust Geometry — state diagram showing dimension weight transitions under different pressure levels.

**Figure 6:** Immune Memory Lifecycle — flowchart showing signature extraction, storage, and matching.

**Figure 7:** Cooperative Amplification Circuit — feedback loop diagram showing trust → bonus → fitness → selection → cooperation cycle.

**Figure 8:** Assortative Trust Pairing — diagram showing trust-weighted partner selection with evolved selectivity.

*Note: Formal drawings to be prepared by patent illustrator for non-provisional filing.*

---

## PREFERRED EMBODIMENTS

### Embodiment 1: Blockchain Sybil Defense

The system is deployed as a smart contract or oracle service on a blockchain network (e.g., Ethereum, Solana). Entities are wallet addresses. Interactions are on-chain transactions (token swaps, governance votes, protocol interactions). The multi-signal detection engine monitors behavioral patterns and flags Sybil wallets for exclusion from airdrops, governance, or protocol rewards.

### Embodiment 2: DAO Governance Security

The system is deployed within a DAO governance framework. Entities are voting members. Interactions are votes, proposals, and delegation actions. The system detects coordinated voting rings and adjusts governance weight based on adaptive trust scores.

### Embodiment 3: AI Multi-Agent Systems

The system is deployed in an AI agent orchestration framework. Entities are autonomous AI agents. Interactions are cooperation/competition decisions in shared resource environments. The system detects colluding agents and maintains healthy agent ecosystems.

**Embodiment 4: Social Platform Integrity**

The system is deployed within a social media platform. Entities are user accounts. Interactions are engagement actions (likes, shares, follows, comments). The system detects coordinated inauthentic behavior (bot networks, influence operations) through behavioral inversion and phase coherence analysis.

**Embodiment 5: Peer-to-Peer Network Security**

The system is deployed in a P2P network protocol. Entities are network nodes. Interactions are data relay, block propagation, and peer connections. The system detects eclipse attacks and Sybil node clusters.

---

*This provisional patent application establishes priority for the above-described inventions. A non-provisional application with formal drawings, additional embodiments, and refined claims will be filed within 12 months of this filing date.*

---

**Respectfully submitted,**

Femi [LAST NAME] Inventor

Date: ___