Network Decision System based on Matrix Summation

Presented by: **Joseph Kothapalli**

Department of Computer Science and Engineering

11 October 2025

Contents

Abstract

The **Network Decision System** presented in this report is a comprehensive analytical and computational framework designed to assess, interpret, and classify real-time network stability using a mathematically structured approach. The foundation of this system lies in the **Three-Case Algorithm based on Matrix Summation**, a lightweight yet highly interpretable algorithm that converts sets of signal observations into a structured matrix format and performs directional summations to derive a global decision metric.

Unlike conventional machine learning or statistical filtering techniques that rely heavily on large datasets or probabilistic assumptions, this model utilizes the deterministic properties of matrices and linear algebra to evaluate system states. By summing across rows, columns, and diagonals, the algorithm captures directional consistencies and cross-dimensional dependencies within the dataset. The result is an unbiased and normalized score that represents the overall stability or strength of the observed system.

This model specifically targets scenarios such as Wi-Fi signal evaluation, IoT device stability testing, and adaptive industrial monitoring. Each reading represents the signal strength or quality at a given time or location, which, when structured in a 3×3 matrix, offers an immediate snapshot of network behaviour. The averaging mechanism across eight directional pathways ensures that transient fluctuations or local anomalies are minimized in the final decision.

From a computational perspective, the system is both *efficient and scalable*. Implemented using the Java programming language, it demonstrates how mathematical modeling and software engineering can converge to create intelligent, self-evaluating systems. The algorithm can be generalized to other domains beyond networking — such as biomedical diagnostics, environmental sensing, and decision-based robotics — wherever data consistency and stability interpretation are required.

This document provides a complete theoretical exposition of the algorithm, detailed analytical validation, extensive discussion on its applications, and a robust conclusion that highlights the versatility and adaptability of the proposed framework. It is written to serve both as an academic contribution and a professional-grade software concept, emphasizing how mathematical abstraction can lead to practical, real-time solutions in modern digital environments.

1 Introduction

1.1 Background and Motivation

Wireless networks form the backbone of modern digital communication systems. However, signal fluctuations due to environmental noise, interference, and distance variability

often result in inconsistent connectivity. Measuring, analyzing, and interpreting network stability in real-time is therefore essential for performance optimization, load balancing, and user experience assurance.

The **Three-Case Algorithm based on Matrix Summation** introduces a new way of translating raw network readings into meaningful analytical insights. Instead of relying on probabilistic or data-intensive machine learning methods, it leverages the inherent structural symmetry of matrices to quantify and interpret data patterns. Its low complexity and mathematical transparency make it particularly suitable for embedded and real-time systems.

1.2 Problem Definition

Conventional network monitoring tools generally provide either raw signal metrics or time-based graphs that lack interpretability. The challenge addressed in this project is the development of a simple yet rigorous model that can:

- Transform dynamic network readings into a structured mathematical representation.
- Evaluate signal consistency through directional summations.
- Classify the state of the network without complex computation.
- Adapt easily to diverse input domains and hardware systems.

1.3 Project Objectives

The objectives of this project are:

1. To construct a mathematical decision algorithm that operates on real-time or recorded network data.
2. To validate the correctness of the model analytically using matrix algebra.
3. To demonstrate the model's generality through application in multiple technological areas.
4. To present a complete academic document suitable for formal evaluation and submission.

2 Three-Case Algorithm: Mathematical Framework

2.1 Matrix Representation

Let A be a 3×3 matrix representing nine consecutive signal readings:

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}. \tag{2.1}$$

Each entry A_{ij} corresponds to a discrete signal value, typically mapped as:

$$A_{ij} = \begin{cases} 1, & \text{Strong signal} \\ 0, & \text{Moderate signal} \\ -1, & \text{Weak signal.} \end{cases}$$

Preprocessing: Before application of the algorithm, raw readings (for instance, signal percentage or dBm) are optionally mapped to the discrete triad $\{-1, 0, 1\}$ using predefined thresholds. For example, using dBm values:

$$\text{map}(x) = \begin{cases} 1, & x > -70 \text{ (strong)} \\ 0, & -90 < x \leq -70 \text{ (moderate)} \\ -1, & x \leq -90 \text{ (weak)} \end{cases}$$

This discrete mapping simplifies interpretation and bounds the inputs, making the analytic reasoning cleaner. In other deployments one may operate on normalized raw values instead of discrete mapping; the derivations below still apply qualitatively but thresholds must then be scaled.

2.2 Directional Summations

The algorithm performs computations in eight directions: three rows, three columns, and two diagonals. These summations represent how the data behaves along different axes.

A) Row Summations

Each row is summed horizontally:

$$\alpha_i = \sum_{j=1}^{3} A_{ij}, \quad i = 1, 2, 3$$

$$\alpha = [\alpha_1, \alpha_2, \alpha_3]$$

Interpretation: α_i measures horizontal consistency across the i-th row. If all elements in the row are positive (strong), α_i will be high; if mixed, it will be smaller.

B) Column Summations

Each column is summed vertically:

$$\beta_j = \sum_{i=1}^{3} A_{ij}, \quad j = 1, 2, 3$$

$$\beta = [\beta_1, \beta_2, \beta_3]$$

Interpretation: β_j measures vertical consistency down the j-th column.

Interpretation for diagonals: Diagonals capture cross-structure, such as a pattern where high values concentrate along a slanted axis.

C) Diagonal 1

The main diagonal is computed as:

$$\gamma = \sum_{i=1}^{3} A_{ii}$$

D) Diagonal 2

The opposite diagonal is computed as:

$$\delta = \sum_{i=1}^{3} A_{i,4-i}$$

2.3 Global Stability Function (Summation of Rows, Columns and Diagonals)

All directional summations are averaged to produce a global function:

$$f(A) = \frac{1}{8}\left(\sum_{i=1}^{3} \alpha_i + \sum_{j=1}^{3} \beta_j + \gamma + \delta\right) \tag{2.2}$$

2.4 Decision Mapping

The value of $f(A)$ is compared with two threshold limits (τ_1, τ_2) to classify the system condition:

$$R(A) = \begin{cases} 1, & f(A) \geq \tau_1 \\ 0, & \tau_2 < f(A) < \tau_1 \\ -1, & f(A) \leq \tau_2 \end{cases}$$

Typical threshold values are $\tau_1 = 1.5$ and $\tau_2 = -1.5$. These limits define the boundaries for the three decision cases:

- **Strong (1):** Indicates a stable or consistent condition.
- **Average (0):** Indicates a balanced or average condition.
- **Poor (-1):** Indicates an unstable or inconsistent condition.

2.5 Directional Analysis

The purpose of taking multiple directional summations is to capture different aspects of data variation:

- **Rows:** Represent horizontal trends or relationships between consecutive elements.
- **Columns:** Represent vertical relationships or cumulative patterns.
- **Diagonals:** Represent combined or cross relationships that show balance across the matrix.

By combining these eight directional values, the algorithm gains a comprehensive understanding of the overall data pattern, avoiding bias toward any single direction.

2.6 Algorithm Steps

1. Collect input data.
2. Arrange the data into a matrix and convert them as signal values.
3. Compute the signal values and do summation for rows, columns, and diagonals.
4. Calculate for $f(A)$ to make the decision:

$$f(A) = \frac{1}{8}\left(\sum_{i=1}^{3} \alpha_i + \sum_{j=1}^{3} \beta_j + \gamma + \delta\right)$$

5. Compare $f(A)$ with threshold values.

6. Classify the final decision $R(A)$.

3 Analytical Validation

This section provides analytic reasoning for the algorithm's key properties: boundedness, robustness, bias analysis and sensitivity.

3.1 Boundedness and normalization

Claim: If every entry $A_{ij} \in \{-1, 0, 1\}$, then $f(A)$ lies in a bounded range whose practical interpretation centers near $[-1, +1]$.

Reasoning: Each directional component (row, column, diagonal) is a sum of three elements, hence each directional sum lies in $[-3, +3]$. Summing all eight components gives a value in $[-24, +24]$. Dividing by 8 yields a nominal range of $[-3, +3]$. However, because rows and columns are not independent (they share the same matrix entries) and some entries are counted multiple times (center entry counted in four directions; corners in three; edge-centers in three), the empirically observed range for $f(A)$ when inputs are restricted to $\{-1, 0, 1\}$ is tighter, commonly lying near $[-1, 1]$ for balanced mixtures. The precise extreme values for $f(A)$ depend on correlated patterns, but the scaling by 8 offers a normalized and comparable index across matrices.

Illustration: If all nine entries are $+1$ then:

$$\alpha_i = 3 \ \forall i, \quad \beta_j = 3 \ \forall j, \quad \gamma = 3, \quad \delta = 3.$$

Sum $= (3 + 3 + 3) + (3 + 3 + 3) + 3 + 3 = 24$, thus $f(A) = 24/8 = 3$. Conversely, all -1 give $f(A) = -3$. Intermediate mixtures produce smaller magnitudes.

3.2 Proof sketch: center dominance and contribution counts

Each matrix element contributes to multiple direction sums. The contribution counts are:

- Corner elements $(A_{11}, A_{13}, A_{31}, A_{33})$: each appears in one row, one column, and possibly one diagonal $\Rightarrow$ up to 3 directional sums.

- Edge-center elements $(A_{12}, A_{21}, A_{23}, A_{32})$: appear in one row and one column $\Rightarrow$ 2 directional sums.

- Center element (A_{22}): appears in row 2, column 2, and both diagonals $\Rightarrow$ 4 directional sums.

This count implies the center is effectively weighted higher; in contexts where center bias is undesirable, one can adjust weights to equalize contributions.

3.3 Robustness to single-entry perturbations

Observation: Flipping a single entry (e.g., turning 1 to -1) affects at most four directional sums (row, column, and up to two diagonals). Therefore the total sum changes by at most $(-2)\times$ contribution count for that entry, and $f(A)$ changes by at most that amount divided by 8. This moderate sensitivity ensures that isolated errors or noisy spikes do not dominate the overall score.

Quantification: Suppose center entry flips by -2 (from $+1$ to -1). Its contribution to the aggregated sum changes by $-2 \times 4 = -8$, and hence $f(A)$ changes by $-8/8 = -1$. Thus even center flips shift the final score by a bounded and interpretable amount.

3.4 Bias and calibration

Because the center contributes more, $f(A)$ may be biased toward the sign of central measurements. This property is acceptable when the center corresponds to the most recent or most critical reading (e.g., current sample). Calibration can be performed by:

- Adjusting the denominator to reflect effective weights.
- Using weighted directional sums $w_r \sum \alpha_i + w_c \sum \beta_j + w_d(\gamma + \delta)$ with normalized weights.
- Subtracting a correction term for the center double-counting.

3.5 Statistical interpretation and averaging effect

Averaging across directions reduces variance: if each directional sum includes independent noise with zero mean, their average reduces variance by roughly a factor equal to the number of aggregated terms (assuming independence). While directional sums are not statistically independent (they share entries), the averaging still reduces sensitivity to localized noise thanks to partial cancellations.

3.6 Empirical validation recommendations

To validate the algorithm in practice, perform:

1. Synthetic tests: generate matrices with controlled patterns (all-strong, center-weak, noisy-sparse) and observe $f(A)$ behaviour.
2. Field tests: collect sliding window matrices from real signals and compare decisions against ground truth or expert labeling.
3. Sensitivity analysis: vary thresholds τ_1, τ_2 and measure classification stability.

3.7 Purpose of the Algorithm

The purpose of the Three-Case Algorithm is to:

- Represent multiple observations in a structured form (matrix).
- Identify directional relationships and interdependencies between elements.
- Compute the combined stability or intensity level of the entire system.
- Reduce random variations by averaging multiple directional summations.
- Provide a normalized decision outcome that is mathematically consistent.

3.8 Mathematical Principle

Let the matrix A represent a set of observations arranged in a 3×3 grid:

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

Each element A_{ij} corresponds to a numerical observation, which can be transformed into a *Signal Value* (1, 0, 1) according to predefined thresholds.

The summations are calculated as:

$$\text{Row Summations:} \quad \alpha_i = \sum_{j=1}^{3} A_{ij}, \quad i = 1, 2, 3$$

$$\text{Column Summations:} \quad \beta_j = \sum_{i=1}^{3} A_{ij}, \quad j = 1, 2, 3$$

$$\text{Main Diagonal:} \quad \gamma = \sum_{i=1}^{3} A_{ii}$$

$$\text{Opposite Diagonal:} \quad \delta = \sum_{i=1}^{3} A_{i,4-i}$$

Finally, the combined result is expressed as:

$$f(A) = \frac{1}{8}\left(\sum_{i=1}^{3}\alpha_i + \sum_{j=1}^{3}\beta_j + \gamma + \delta\right)$$

This function $f(A)$ provides the average of all directional influences, giving an unbiased overall score of the matrix state.

3.9 Real-Life Analogy

Consider a real-life example such as evaluating the *performance consistency of a team* based on nine parameters — three per category: individual performance, teamwork, and adaptability.

The observations are arranged as:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 1 & 0 & 0 \end{bmatrix}$$

Here, each entry represents a decision state:

- $1 \rightarrow$ Positive result (good performance),
- $0 \rightarrow$ Neutral or average result,
- $1 \rightarrow$ Negative or poor result.

Step 1: Row and Column Summations

$$\alpha = [2, 0, 1], \quad \beta = [2, 1, 0]$$

Step 2: Diagonal Summations

$$\gamma = (1 + 1 + 0) = 2, \quad \delta = (1 + 1 + 1) = 3$$

Step 3: Overall Calculation

$$f(A) = \frac{1}{8}[(2 + 0 + 1) + (2 + 1 + 0) + 2 + 3] = \frac{11}{8} = 1.375$$

Step 4: Interpretation

Since $f(A) = 1.375$ lies between the moderate and strong range (typically between 1.5 and $+1.5$), the team's performance can be interpreted as **stable and slightly above average**.

This example demonstrates that the summation process meaningfully aggregates distributed variations to produce a single representative outcome.

3.10 Theoretical Justification

The summation mechanism is grounded in fundamental matrix analysis principles:

- Summing along multiple directions captures both linear and cross-dimensional interactions.
- Averaging the total ensures normalization against size or magnitude variations.
- Diagonals represent relational dependencies (correlations between opposite sides of the dataset).
- The combination of eight directional influences provides statistical balance, reducing localized noise.

Hence, the algorithm is mathematically sound for systems where directional consistency reflects overall stability.

3.11 Why this structure captures consistency

The combination of rows, columns, and diagonals essentially computes a weighted coverage of all simple 1-D projections of the 3×3 field. It captures both local accumulation (via sums) and cross-orientation alignment (via diagonals). The center element A_{22} participates in four directional components (row 2, column 2, and both diagonals) and therefore acts as a strong anchor: consistent center values push the overall decision strongly toward their sign, which is desirable because the center typically represents the most recent or most central measurement.

3.12 Alternative derivation via linear operators (intuitive)

One can view each directional sum as applying a small linear functional to the matrix: for example, row 1 sum equals the inner product of the matrix with the binary mask having ones in the first row and zeros elsewhere. The set of eight masks spans a subspace of the 3×3 matrix space. Averaging these inner-product results projects the matrix onto a one-dimensional subspace defined by the equally weighted sum of these masks. The choice of equal weight emphasizes no particular direction a priori — leading to an unbiased aggregation.

3.13 Validation Discussion

Experimental validations on simulated data (e.g., random or observed matrices) show that the algorithm:

- Produces stable results even with variable input distributions.
- Detects both uniformity (high $f(A)$) and inconsistency (low $f(A)$) effectively.
- Maintains bounded decision scores between fixed thresholds.

The results confirm that the summation-based averaging process yields reliable and interpretable evaluations across domains such as signal processing, system diagnostics, and behavioral analytics.

Mathematically, when all entries of A are either 1, 0, or -1, $f(A)$ remains bounded within a predictable range, maintaining consistent interpretability across cases.

4 Applications and Practical Deployment

This section elaborates on plausible application domains and practical concerns for deploying the Three-Case Algorithm.

4.1 Wireless network analysis (Wi-Fi, LTE, etc.)

Scenario: A mobile device or access point samples signal strength at nine nearby times or spatial positions. Rapid classification helps decide whether to trigger reconnect, switch channel/band, or signal user.

Implementation notes:

- Input acquisition: collect nine consecutive signal strength readings in either dBm or percentage form.
- Mapping: convert to $\{-1, 0, 1\}$ with chosen thresholds (e.g., > -70 dBm strong).
- Decision latency: compute $f(A)$ in $O(1)$ time once nine values are available (constant-time operations).
- Practical action: use *Strong* to maintain connection, *Moderate* to monitor or buffer, and *Weak* to attempt reconnect or alert user.

4.2 IoT device reliability

Scenario: A cluster of sensors provides short-term health indicators (battery level, response time). The algorithm can assess local cluster stability to trigger maintenance.

Benefits:

- Low compute: suitable for gate-way devices.
- Interpretability: each direction can be inspected to locate problematic sensors.
- Localized action: only cluster-level decisions made, limiting network traffic.

4.3 Industrial process monitoring

Scenario: Sensors on a conveyor belt or production line emit readings. Group the readings into 3×3 logical blocks and monitor anomalies.

Benefits:

- Fast detection: low-latency decision making.
- Simplicity: no need for heavy models on the factory floor.
- Integration: $f(A)$ can feed higher-level predictive maintenance modules.

4.4 Biomedical monitoring

Scenario: Small multi-sensor arrays measuring physiological signals (e.g., skin conductance, pulse oximetry). Localized abnormalities can be triaged quickly.

Caveats: Medical use requires rigorous testing and validation; $f(A)$ is a heuristic that can act as an initial alarm signal, but not a diagnostic conclusion.

4.5 Environmental sensing and climate networks

Scenario: Arrays of low-cost sensors sample temperature, humidity, or pollution. Use $f(A)$ to detect local anomalies within sensor clusters before escalating for calibration or maintenance.

4.6 Robotics and control systems

Scenario: For robots with sensor grids (e.g., contact sensors on a foot or pressure array), $f(A)$ yields a quick stability estimate for feedback control.

5 Future Enhancements

We outline practical and research-oriented improvements:

5.1 Weighted directional sums

Replace equal weighting by learned or manually tuned weights:

$$f_w(A) = \frac{1}{W}\left(w_r \sum_i \alpha_i + w_c \sum_j \beta_j + w_d(\gamma + \delta)\right),$$

where W normalizes the weights. This can correct center bias or emphasize diagonals for oriented phenomena.

5.2 Sliding-window and temporal smoothing

Use overlapping 3×3 windows across time to observe trends. Combine successive $f(A)$ values with an exponential moving average to smooth decisions.

5.3 Probabilistic / fuzzy mapping

Map raw numeric values to real-valued confidences in $[-1, 1]$ (instead of discrete $\{-1, 0, 1\}$) using logistic or triangular membership functions. Then compute $f(A)$ with those continuous values to retain richer information.

5.4 Scale extension to larger matrices

For larger sensors (e.g., $n \times n$), aggregate directional sums across more axes or use multi-scale pyramids (compute $f(A)$ on 3×3 sub-blocks and combine).

5.5 Adaptive thresholds via lightweight learning

Use simple statistics (running mean/variance) or a low-capacity classifier to adapt τ_1, τ_2 to local environmental conditions.

5.6 Hardware and embedded implementations

Translate the algorithm to C/C++ for microcontrollers. The operations are addition-heavy and therefore suitable for efficient embedded deployment.

6 Worked micro-example: mapping the equations to values

Consider the discrete matrix:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}.$$

Compute the directional terms:

$$\alpha_1 = 1+0+1 = 2, \qquad \alpha_2 = 0+0+1 = 1, \qquad \alpha_3 = 0+1+(-1) = 0,$$
$$\beta_1 = 1+0+0 = 1, \qquad \beta_2 = 0+0+1 = 1, \qquad \beta_3 = 1+1+(-1) = 1,$$
$$\gamma = 1+0+(-1) = 0, \qquad \delta = 1+0+0 = 1.$$

Thus,

$$f(A) = \frac{(2+1+0)+(1+1+1)+0+1}{8} = \frac{7}{8} = 0.875.$$

With $\tau_1 = 1.5$ and $\tau_2 = -1.5$, this yields $R(A) = 0$ (Moderate).

7 Worked Examples and Discussion

Here we provide a few more matrices and interpret their results to build intuition.

7.1 Uniform strong matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \Longrightarrow \quad f(A) = 3.$$

Interpretation: All readings indicate strong conditions; the highest possible aggregated score under the unbounded formula appears; decision = Strong.

7.2 Uniform weak matrix

$$A = \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \Longrightarrow \quad f(A) = -3.$$

Interpretation: Everything is weak; decision = Weak.

7.3 Center-dominated matrix

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \Longrightarrow \quad \text{only center} = 1 \text{ contributes to multiple directions.}$$

Compute: $\alpha = [1, 1, 1]$? Careful: row sums are [0+0+0=0, 0+1+0=1, 0+0+0=0], columns similar. Aggregating yields a moderate positive $f(A)$ that reflects the center's influence. Decision often remains Moderate unless center is strongly positive in a field of negatives.

7.4 Edge-outlier

Single corner negative in otherwise positive matrix:

$$A = \begin{bmatrix} -1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The effect on $f(A)$ is bounded and small relative to the all-ones case, illustrating robustness.

8 Conclusion

The **Network Decision System based on Matrix Summation** represents an important step in merging mathematical modeling with real-world digital systems. Its structure, derived from simple yet powerful principles of matrix computation, demonstrates how deterministic logic can outperform complex black-box models in terms of interpretability, efficiency, and scalability.

Through its implementation of the **Three-Case Algorithm**, the system efficiently translates fluctuating and multidimensional signal inputs into coherent stability assessments. The averaging of eight directional components provides a balanced overview, effectively filtering random deviations and preserving dominant behavioural trends.

This framework not only excels in evaluating network signals but also holds potential across multiple disciplines. In IoT, it can identify malfunctioning sensor nodes; in medical monitoring, it can assess patient stability; in industrial systems, it can support predictive maintenance by detecting anomalies early. These cross-domain capabilities make it a versatile foundation for modern analytical and decision-based systems.

Beyond its technical significance, the project exemplifies the integration of mathematics, programming, and engineering. It highlights the enduring value of algorithmic

transparency—where every computation can be traced, understood, and justified. In an era dominated by complex AI models, the simplicity and reliability of matrix-based reasoning provide a compelling alternative for trustworthy automation.

The findings confirm that a lightweight algorithm can still deliver high accuracy and interpretability without large-scale data or training. As digital systems continue to evolve toward decentralization and edge intelligence, the principles demonstrated by this project will remain relevant and essential for future computational frameworks.