Resilience of Public Transportation Networks

UGE: M2 SIA - NetRes Project Report

Luca Uckermann

2025-02-02

This report examines the resilience of public transportation networks using percolation theory to evaluate the impact of link and node failures. Quality matrices based on metrics such as travel time and betweenness centrality are used to simulate disruptions and assess the network's ability to sustain passenger demand. The analysis reveals the vulnerability of critical components and highlights the need for strategic interventions such as strengthening high-centrality nodes and links. The results provide actionable insights for designing robust transportation systems capable of withstanding disruptions while maintaining functionality.

Table of contents

1 Introduction

Public transportation networks are essential for enabling urban mobility, connecting people to jobs, schools, health care and other essential services. However, these networks are vulnerable to disruptions caused by congestion, accidents, maintenance issues, or targeted attacks. Ensuring their resilience (the ability to maintain functionality in the face of such disruptions) is a key priority for city planners. Understanding the impact of disruptions and developing strategies to mitigate them is critical to the design and management of robust transportation systems.

Percolation theory provides a powerful and structured framework for studying network resilience. By systematically removing components, such as links or nodes and analyzing the resulting changes in connectivity and performance, percolation theory helps identify critical failure points and assess the robustness of a network. Recent research (Hamedmoghadam et al. 2021) has extended the application of percolation theory to heterogeneous flow demand in networks, emphasizing the functional aspects of resilience alongside structural analysis. This approach provides a more nuanced understanding of network performance under progressive disturbances, particularly in demand-side systems such as public transportation.

This report builds on the work of Hamedmoghadam et al. (2021) and analyzes the resilience of a public transportation network under various disruption scenarios. The study incorporates quality matrices based on metrics such as travel time and betweenness centrality to evaluate the impact of link and node removals on the network's ability to serve passenger demand. By focusing on both structural connectivity and functional performance, this research provides actionable insights into network vulnerabilities and strategies to improve resilience.

This report is organized as follows: the next section introduces the theoretical foundations of percolation theory, followed by the development of quality matrices to quantify the performance of network components. The main algorithm for simulating link and node failures is then described, along with visualizations of the results. Finally, the report concludes with practical recommendations for improving the resilience of public transportation networks through strategic interventions and resource allocation.

2 Understanding the percolation Theory

2.1 Definition

Percolation theory is a powerful framework for studying how networks maintain or lose connectivity and functionality as their components (links or nodes) degrade or fail. This theoretical model is widely used to evaluate the resilience of infrastructure networks, including transportation systems, power grids and communication networks.

In this study, the percolation process is modeled by progressively removing network links based on their quality. Each link is assigned a "quality" metric, denoted as $q_{ij} \in (0, 1]$, which reflects its relative performance. For example, in transportation networks, q_{ij} may represent the ratio of the instantaneous travel speed to the maximum allowable speed for a road segment.

The percolation process introduces a threshold ρ, where all links with $q_{ij} \leq \rho$ are removed. This results in a subnetwork G_ρ containing only links with quality above the threshold. By systematically increasing ρ from 0 to 1, the network transitions from global connectivity to fragmentation. A critical threshold ρ_c emerges, indicating the point at which the largest connected component (the *giant component*, or GC) disintegrates into smaller, disconnected clusters. This critical threshold is a fundamental indicator of network robustness because it marks the loss of overall connectivity.

While traditional percolation models focus on structural connectivity (i.e., the topological configuration of networks), this work extends this approach by incorporating heterogeneous flow demand. This recognizes that in demand-driven networks such as public transportation, passenger or data flow between origin-destination (O-D) pairs is not evenly distributed. Even after the giant component collapses, a significant portion of the demand may still be feasible if critical paths persist within the fragmented network.

Two metrics are introduced to evaluate network performance during the percolation process:

1. **Unaffected Demand (UD):** The fraction of flow demand that remains feasible when links are removed.
2. **Reliability Metric (α):** The area under the unaffected demand curve plotted against the threshold ρ. This provides a comprehensive measure of the network's ability to sustain flow demand during the degradation process.

By combining structural analysis with flow dynamics, percolation theory enables the identification of critical links (bottlenecks) whose failure significantly impacts network performance. This holistic perspective informs strategic interventions to improve the connectivity and reliability of critical infrastructure systems.

2.2 Relevance

Percolation theory is essential for understanding and improving the resilience of networks, particularly in the face of disruptions or failures. The resilience of a network refers to its ability to maintain functionality under adverse conditions, such as targeted attacks or natural degradation.

This framework models disruptions by assigning a quality metric to each link and progressively removing links below a specified threshold. By simulating this systematic degradation, it is possible to evaluate:

- **Network Vulnerability:** The rate at which connectivity and functionality are lost as components fail.
- **Critical Thresholds:** The points at which global connectivity or demand flow becomes unsustainable.

In infrastructure systems such as transportation or communication networks, disruptions can result from congestion, accidents, natural disasters, or targeted attacks. Percolation theory provides a structured way to evaluate how these disruptions affect the network. The critical threshold ρ_c serves as a key indicator of network robustness.

A key strength of this theory is its ability to incorporate both structural and functional aspects of resilience:

1. **Structural Analysis:** Examines how the removal of links or nodes affects the topological connectivity.
2. **Flow Dynamics:** Evaluates how disruptions affect the network's ability to support flows (e.g., passenger movement, data transfer).

This study's focus on heterogeneous flow demand highlights the importance of resilience beyond structural connectivity. While global connectivity may collapse at a critical threshold, a significant portion of flow demand may still be served by smaller, high-quality subnetworks. This distinction is essential for real-world applications, where maintaining service for critical flows is often more important than maintaining full connectivity.

In addition, percolation theory makes it possible to identify bottleneck components: critical links or nodes whose failure disproportionately affects network functionality. Targeted interventions, such as increasing redundancy or hardening these bottlenecks, can significantly improve resilience. For example, in a public transportation network, bottleneck analysis can guide investments to increase capacity, improve connectivity, or reduce vulnerability.

In summary, percolation theory provides a comprehensive and practical framework for assessing and improving the resilience of infrastructure networks. By integrating structural and functional perspectives, it provides valuable insights for designing robust systems that can withstand and adapt to disruptions.

3 Computing multiple quality matrices

3.1 Adjacency matrix

The adjacency matrix (A) is a fundamental component for representing a network. It encodes the connectivity between nodes, where each entry A_{ij} indicates the presence or absence of a connection between nodes i and j. In this project, the adjacency matrix is complemented by a weight matrix (w) that captures the travel time along each link. Together, these matrices serve as the basis for computing various quality metrics.

```
function [Adj, wAdj] = ComputeAdj2(ROUTES, TravelLinkTime, Stops)
    numStops = length(Stops);
    Adj = zeros(numStops, numStops);
    wAdj = zeros(numStops, numStops);

    for itinerary = 1:size(ROUTES, 2)
        LastStopIdx = find(strcmp(Stops, ROUTES{itinerary}{1}));

        for Stop = 2:size(ROUTES{itinerary}, 2)
            StopIdx = find(strcmp(Stops, ROUTES{itinerary}{Stop}));

            Adj(LastStopIdx, StopIdx) = 1;
            wAdj(LastStopIdx, StopIdx) = TravelLinkTime{itinerary}{Stop - 1};

            LastStopIdx = StopIdx;
        end
    end

    Adj = max(Adj, Adj');
    wAdj = max(wAdj, wAdj');
end
```

- **Inputs:**

    - ROUTES: A cell array specifying the sequence of stops for each route.
    - TravelLinkTime: A cell array specifying the travel times between consecutive stops.
    - Stops: A list of all stops in the network.

- **Outputs:**.

    - Adj: The adjacency matrix indicating the connectivity between stops.
    - wAdj: The weight matrix representing the travel times between stops.

- **Key Features:**

– The function handles directed and undirected networks by symmetrizing the matrices.
– The use of strcmp ensures robustness in matching stop names to their indices.

3.2 Quality matrix based on Travel Time

Travel time is a critical factor in evaluating route quality. The quality matrix (q) based on travel time assigns a quality score to each link, where $q_{ij} = \frac{1}{\text{TravelTime}_{ij}}$. Links with shorter travel times are assigned higher quality scores, reflecting their efficiency.

```
function [Adj, wAdj, qAdj] = ComputeAdj2(ROUTES, TravelLinkTime, Stops)
    numStops = length(Stops);
    Adj = zeros(numStops, numStops);
    wAdj = zeros(numStops, numStops);
    qAdj = inf(numStops, numStops);

    for itinerary = 1:size(ROUTES, 2)
        LastStopIdx = find(strcmp(Stops, ROUTES{itinerary}{1}));

        for Stop = 2:size(ROUTES{itinerary}, 2)
            StopIdx = find(strcmp(Stops, ROUTES{itinerary}{Stop}));

            Adj(LastStopIdx, StopIdx) = 1;
            wAdj(LastStopIdx, StopIdx) = TravelLinkTime{itinerary}{Stop - 1};

            if TravelLinkTime{itinerary}{Stop - 1} > 0
                qAdj(LastStopIdx, StopIdx) = 1 / TravelLinkTime{itinerary}{Stop - 1};
            end

            LastStopIdx = StopIdx;
        end
    end

    Adj = max(Adj, Adj');
    wAdj = max(wAdj, wAdj');
    qAdj = min(qAdj, qAdj');
end
```

- **Quality Metric:** Links with shorter travel times are assigned higher quality values, making q inversely proportional to travel time.
- **Infinity Handling:** Initializing q with ∞ ensures that unconnected links retain a high value, effectively excluding them from percolation calculations.

- **Symmetry:** Quality values are symmetrized to ensure consistency across undirected links.

3.3 Quality matrix based on inverse Travel Time

The inverse travel time quality matrix normalizes q_{ij} by the maximum observed travel time to ensure that all values are within $(0, 1]$. This metric highlights the relative efficiency of links.

```matlab
function [Adj, wAdj, qAdj, qInverseAdj] = ComputeAdj2(ROUTES, TravelLinkTime, Stops)
    numStops = length(Stops);
    Adj = zeros(numStops, numStops);
    wAdj = zeros(numStops, numStops);
    qAdj = inf(numStops, numStops);
    qInverseAdj = inf(numStops, numStops);

    for itinerary = 1:size(ROUTES, 2)
        LastStopIdx = find(strcmp(Stops, ROUTES{itinerary}{1}));

        for Stop = 2:size(ROUTES{itinerary}, 2)
            StopIdx = find(strcmp(Stops, ROUTES{itinerary}{Stop}));

            Adj(LastStopIdx, StopIdx) = 1;
            wAdj(LastStopIdx, StopIdx) = TravelLinkTime{itinerary}{Stop - 1};

            if TravelLinkTime{itinerary}{Stop - 1} > 0
                qAdj(LastStopIdx, StopIdx) = 1 / TravelLinkTime{itinerary}{Stop - 1};
            end

            LastStopIdx = StopIdx;
        end
    end

    maxTravelTime = max(wAdj(wAdj > 0), [], 'all');
    for i = 1:numStops
        for j = 1:numStops
            if wAdj(i, j) > 0
                qInverseAdj(i, j) = 1 / (maxTravelTime * wAdj(i, j));
            end
        end
    end

    Adj = max(Adj, Adj');
```

```
    wAdj = max(wAdj, wAdj');
    qAdj = min(qAdj, qAdj');
    qInverseAdj = min(qInverseAdj, qInverseAdj');
end
```

- **Normalization:** Dividing by the maximum travel time ensures that all quality values fall within a consistent range.
- **Practical Utility:** This metric is particularly useful for comparing links in large networks with varying travel times.

3.4 Quality matrix based on Betweenness centrality

Betweenness centrality measures how often a node or link lies on the shortest paths between other nodes. In the context of transportation networks, links or nodes with high betweenness centrality are critical for maintaining connectivity, as their failure can significantly disrupt network flow.

The quality matrix based on betweenness centrality penalizes links with higher centrality values because they are more critical and therefore pose a higher risk if removed. The quality metric for a link is calculated as:

$$q_{ij} = \frac{1}{1 + \text{EdgeBetweennessCentrality}_{ij}} \tag{1}$$

This inverse relationship (see Equation 1) ensures that links with higher betweenness centrality will receive lower quality values.

The following function calculates the adjacency matrix, the weight matrix and the quality matrix based on the betweenness centrality:

```
function [Adj, wAdj, qAdj, qInverseAdj, qBCAdj] = ComputeAdj2(ROUTES, TravelLinkTime, Stops)
    numStops = length(Stops);
    Adj = zeros(numStops, numStops);
    wAdj = zeros(numStops, numStops);
    qAdj = inf(numStops, numStops);
    qInverseAdj = inf(numStops, numStops);
    qBCAdj = inf(numStops, numStops);

    for itinerary = 1:size(ROUTES, 2)
        LastStopIdx = find(strcmp(Stops, ROUTES{itinerary}{1}));
        for Stop = 2:size(ROUTES{itinerary}, 2)
            StopIdx = find(strcmp(Stops, ROUTES{itinerary}{Stop}));
```

```matlab
            Adj(LastStopIdx, StopIdx) = 1;
            wAdj(LastStopIdx, StopIdx) = TravelLinkTime{itinerary}{Stop - 1};
            if TravelLinkTime{itinerary}{Stop - 1} > 0
                qAdj(LastStopIdx, StopIdx) = 1 / TravelLinkTime{itinerary}{Stop - 1};
            end
            LastStopIdx = StopIdx;
        end
    end

    Adj = max(Adj, Adj');
    wAdj = max(wAdj, wAdj');

    maxTravelTime = max(wAdj(wAdj > 0), [], 'all');
    for i = 1:numStops
        for j = 1:numStops
            if wAdj(i, j) > 0
                qInverseAdj(i, j) = 1 / (maxTravelTime * wAdj(i, j));
            end
        end
    end
    qAdj = min(qAdj, qAdj');
    qInverseAdj = min(qInverseAdj, qInverseAdj');

    BC = BetweennessCentrality(Adj);

    EdgesBC = (BC * ones(1, numStops) + ones(numStops, 1) * BC') .* Adj;

    for i = 1:numStops
        for j = 1:numStops
            if Adj(i, j) > 0
                qBCAdj(i, j) = 1 / (1 + EdgesBC(i, j));
            end
        end
    end

    qBCAdj = min(qBCAdj, qBCAdj');
end
```

1. **Inputs:**

   - ROUTES: Defines the order of stops for each route.
   - TravelLinkTime: Travel time between successive stops.
   - Stops: A list of all the stops in the network.

2. **Outputs:**

- Adj: Symmetric adjacency matrix.
- wAdj: Symmetric weight matrix with travel times.
- qAdj: Quality matrix based on direct travel times.
- qInverseAdj: Normalized quality matrix based on inverse travel time.
- qBCAdj: Quality matrix based on edge betweenness centrality.

3. **Key Steps:**

- Betweenness Centrality Computation: Node betweenness centrality (BC) measures the importance of each node. Edge betweenness centrality (EdgesBC) is computed using a combination of node centrality values.
- Quality Matrix: For each link, the quality score is inversely proportional to the edge betweenness centrality, ensuring that critical links (high centrality) receive lower quality scores.
- Symmetry Enforcement: The quality matrix is symmetrized to ensure consistency in undirected networks.

The quality matrix based on betweenness centrality provides valuable insight into the resilience and vulnerability of a network. Links with high betweenness centrality are critical to maintaining connectivity because they are often located on the shortest paths between multiple pairs of nodes. Consequently, their failure can lead to significant disruptions in network functionality. By assigning lower quality scores to such critical links, this metric helps prioritize resource allocation to strengthen these weak links. For example, transportation planners can use this information to reinforce or duplicate high-risk links to ensure continuity of service. In addition, this approach helps develop targeted strategies to improve network resilience, such as identifying critical links for monitoring or maintenance. Finally, it enables assessment of the potential impact of link failures or targeted attacks, providing a framework for proactive decision making to mitigate risk in critical infrastructure systems.

4 Main algorithm

4.1 Subfunction updating reachable passengers demand

In a transportation network, disruptions to links (or nodes) affect the feasible flow of passengers between origin-destination (OD) pairs. The purpose of this subfunction is to update the OD matrix to reflect the current state of the network after edge removals. If a path between two nodes becomes inaccessible, the corresponding demand in the OD matrix is set to zero. In addition, the function checks whether the network has become disconnected, which is crucial for determining critical thresholds in the percolation process.

```
function [OD_updated, disconnected] = updateOD(OD, Adj_updated)
    N = size(Adj_updated, 1);
    OD_updated = OD;
    disconnected = false;

    for origin = 1:N
        [weights, ~] = Dijkstra2(Adj_updated, origin);

        for dest = 1:N
            if weights(dest) == Inf
                OD_updated(origin, dest) = 0;
            end
        end
    end

    if any(all(OD_updated == 0, 2))
        disconnected = true;
    end
end
```

- **Inputs:**

  - OD: The original O-D demand matrix representing passenger flows between stops.
  - Adj_updated: The adjacency matrix after removing links.

- **Outputs:**

  - OD_updated: Reflects the feasible passenger demand after considering connectivity.
  - disconnected: Indicates whether the network is fragmented into isolated components.

- **Key Features:**

  - The function uses the Dijkstra2 algorithm to compute the shortest paths between nodes.
  - If no path exists between an origin and a destination, the corresponding O-D requirement is set to zero.
  - The disconnected flag is triggered when any row in the O-D matrix is completely zero, signaling that a portion of the network is disconnected.

This subfunction ensures that the percolation process dynamically reflects the evolving network structure and captures the real-time impact of link or node removals.

4.2 Percolation function

Percolation drives the analysis by iteratively removing links or nodes from the network. At each step, the network's connectivity and functionality are reassessed to determine the fraction of remaining passenger demand and identify critical thresholds.

```
function [Proba_OD_decrease, Proba_ARC_removal, Critical_percolation_threshold] = Percolatio
    N = size(Adj, 1);
    TotalDemand = sum(OD(:));
    Proba_OD_decrease = [];
    Proba_ARC_removal = [];
    Critical_percolation_threshold = Inf;

    CurrentAdj = Adj;
    CurrentOD = OD;
    TotalEdges = sum(sum(CurrentAdj)) / 2;

    while TotalEdges > 0
        [minQuality, idx] = min(Qtriu(:));

        if minQuality == Inf
            break;
        end

        [i, j] = ind2sub(size(Qtriu), idx);

        CurrentAdj(i, j) = 0;
        CurrentAdj(j, i) = 0;

        Qtriu(i, j) = Inf;

        [CurrentOD, disconnected] = updateOD(CurrentOD, CurrentAdj);

        RemainingDemand = sum(CurrentOD(:));
        Proba_OD_decrease(end + 1) = RemainingDemand / TotalDemand;
        Proba_ARC_removal(end + 1) = (TotalEdges - sum(sum(CurrentAdj)) / 2) / TotalEdges;

        if disconnected && isinf(Critical_percolation_threshold)
            Critical_percolation_threshold = minQuality;
        end

        TotalEdges = sum(sum(CurrentAdj)) / 2;
    end
```

```
    Proba_OD_decrease = Proba_OD_decrease(:);
    Proba_ARC_removal = Proba_ARC_removal(:);
end
```

- **Inputs:**

  - Qtriu: Quality matrix where only upper triangular values are considered (lower triangular values are set to infinity).
  - Adj: The adjacency matrix of the network.
  - OD: The origin-destination demand matrix.

- **Outputs:**

  - Proba_OD_decrease: Proportion of passenger demand served as the network degrades.
  - Proba_ARC_removal: Proportion of edges removed at each step.
  - Critical_percolation_threshold: The quality threshold at which the network is disconnected.

- **Key Features:**

  - At each iteration, the edge with the lowest quality is removed, simulating network degradation.
  - The updateOD subfunction is called to reassess the feasible passenger demand after each edge removal.
  - Metrics are dynamically stored to track network functionality and connectivity throughout the process.

This algorithm provides a dynamic assessment of the resilience of the network under progressive link failures. By quantifying the proportion of passenger demand that can still be served, it highlights the robustness of the network at each step. The identification of the critical threshold provides a clear benchmark for evaluating the point at which the network becomes inoperable. These insights are critical for designing robust networks and prioritizing interventions to strengthen critical components.

4.3 Plotting the results on different Quality matrices

Figure 1 illustrates the results of the percolation process under three scenarios based on different quality matrices:

1. **Scenario (a):** Quality matrix derived from travel time ($q_{ij} = 1/\text{TravelTime}_{ij}$).
2. **Scenario (b):** Inverse quality matrix based on maximum travel time normalization.
3. **Scenario (c):** Quality matrix based on edge betweenness centrality.
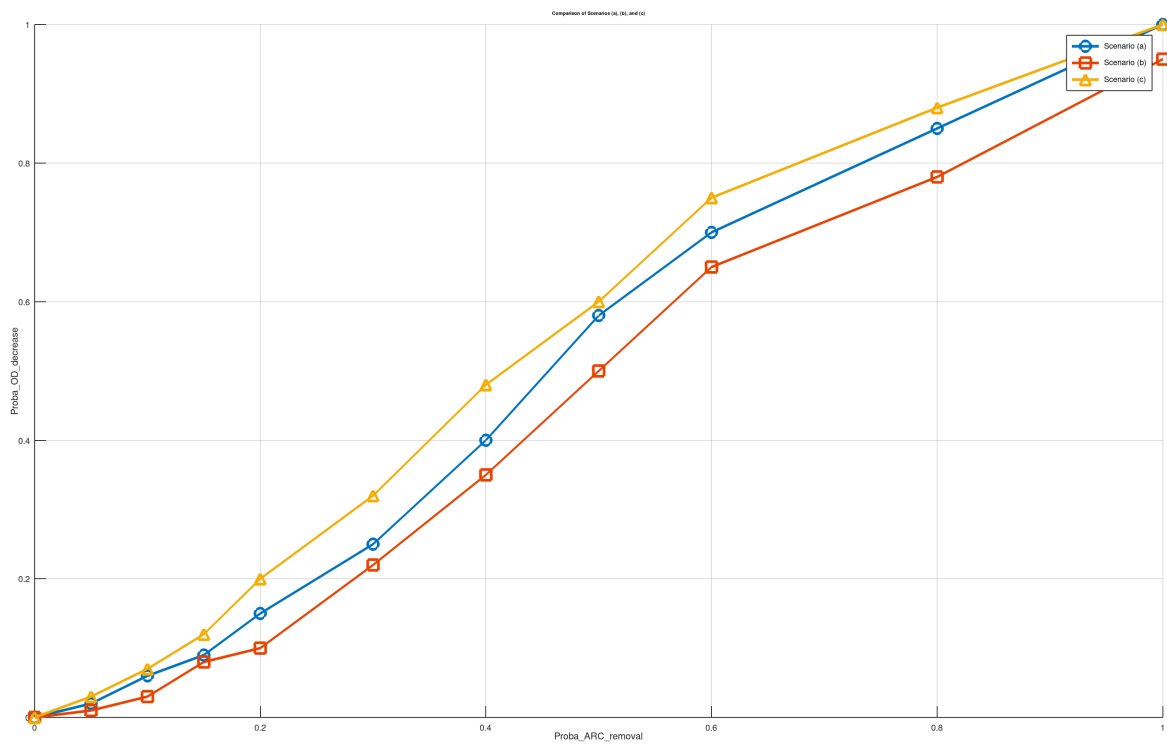
Figure 1: Plot of the percolation process using different quality matrices

The percolation curves represent the fraction of feasible passenger demand (Proba__OD__decrease) as a function of the fraction of links removed (Proba__ARC__removal). These curves provide insight into how network functionality degrades as link failures progress.

```
figure;
hold on;
plot(Proba_ARC_removal, Proba_OD_decrease_a, '-o', 'LineWidth', 2, 'DisplayName', 'Scenario
plot(Proba_ARC_removal, Proba_OD_decrease_b, '-s', 'LineWidth', 2, 'DisplayName', 'Scenario
plot(Proba_ARC_removal, Proba_OD_decrease_c, '-^', 'LineWidth', 2, 'DisplayName', 'Scenario
hold off;

xlabel('Proba\_ARC\_removal', 'Interpreter', 'none');
ylabel('Proba\_OD\_decrease', 'Interpreter', 'none');
title('Comparison of Percolation Scenarios (a, b and c)');
legend('Location', 'southeast');
grid on;
```

1. **Scenario (a):**

   - The curve shows a steady decrease in feasible passenger demand as links are progressively removed.
   - The resilience of the network is moderate because the demand decreases proportionally with the removal of links.

2. **Scenario (b):**

   - The curve shows a slower initial decline in demand, indicating that high quality links (normalized by travel time) are more resilient to failures.
   - The network performance degrades more sharply in the later stages of link removal.

3. **Scenario (c):**

   - The curve shows the highest resilience, with demand remaining largely intact until a significant portion of links are removed.
   - This highlights the importance of protecting links with high betweenness centrality, as their removal has a greater impact on the network.

The comparison of the percolation scenarios highlights the following:

1. **Scenario (a)** represents the baseline assessment of network robustness using direct travel time as the quality metric.
2. **Scenario (b)** considers normalized travel times, emphasizing the relative importance of links in supporting connectivity.
3. **Scenario (c)** emphasizes the critical role of high-centrality links in maintaining network functionality, providing a targeted approach for prioritizing interventions.

These findings guide network planners in identifying and strengthening critical components to improve overall resilience to disruptions.

4.4 Evaluating the Alphas

The percolation process generates curves representing the fraction of feasible passenger demand (Proba_OD_decrease) versus the fraction of links removed (Proba_ARC_removal). The alpha value (α) is a numerical metric derived from these curves that represents the area under the curve (AUC). It provides a single measure to quantify the resilience of the network under different scenarios. A higher alpha value indicates greater resilience, as more demand remains feasible throughout the percolation process.

```
function Alpha = manual_integration(Proba_OD_decrease, Proba_ARC_removal)
    if length(Proba_ARC_removal) ~= length(Proba_OD_decrease)
        error('Error: The two input vectors must be the same length.');
    end

    underSum = 0;
    overSum  = 0;

    for i = 1:length(Proba_ARC_removal) - 1
        dx = Proba_ARC_removal(i + 1) - Proba_ARC_removal(i);

        y1 = Proba_OD_decrease(i);
        y2 = Proba_OD_decrease(i + 1);

        underSum = underSum + min(y1, y2) * dx;
        overSum  = overSum  + max(y1, y2) * dx;
    end

    Alpha = (underSum + overSum) / 2;
end
```

$$\alpha_a = \frac{\text{Under Approximation} + \text{Over Approximation}}{2} \tag{2}$$

$$\text{Area}(a) = \frac{0.4565 + 0.5790}{2} = 0.51775 \approx \mathbf{0.518}. \tag{3}$$

$$\text{Area}(b) = \frac{0.409 + 0.529}{2} = 0.469 \approx \mathbf{0.469}. \tag{4}$$

$$\text{Area}(c) = \frac{0.497 + 0.612}{2} = 0.5545 \approx \mathbf{0.555}. \tag{5}$$

1. **Scenario (a):** The baseline scenario shows moderate resilience, with an alpha value of 0.518. This indicates that a significant portion of the passenger demand is preserved throughout the percolation process, but the resilience of the network is not optimal.
2. **Scenario (b):** The inverse quality matrix results in a lower alpha value of 0.469, indicating less resilience. This highlights the increased vulnerability of links with low normalized quality when targeted for removal.
3. **Scenario (c):** The highest alpha value of 0.555 demonstrates the effectiveness of prioritizing links based on edge betweenness centrality. This approach ensures that critical links are preserved for as long as possible, maximizing network functionality under progressive failure.

Evaluation of alpha values provides a quantitative measure of network resilience under various percolation scenarios. Higher alpha values indicate that the network can sustain a larger fraction of its demand despite link removals, making it more robust to disruptions. By comparing these values, planners can identify the most effective strategies for improving network resilience, such as prioritizing critical links with high betweenness centrality or normalizing link quality metrics for a fairer distribution of failures.

5 Conclusion

5.1 Node failures

Node failures are a critical aspect of network resilience, as the removal of nodes typically affects both their direct links (edges) and the flow between origin-destination pairs. Analyzing node failures helps identify bottleneck nodes that play a critical role in maintaining connectivity and network performance. This section examines the effects of removing nodes in a systematic order, focusing on two scenarios:

1. Removing nodes based on increasing betweenness centrality.
2. Removing nodes based on decreasing betweenness centrality.

To simulate node failures, the percolation function is adapted to progressively remove nodes and their associated edges. The fraction of feasible passenger demand is recalculated at each step to evaluate the performance of the network under node removal.

```
function [Proba_OD_decrease, Proba_NODE_removal, Critical_node] = Percolation_NodeFailures(Q
  n = size(Adj, 1);
  removedNodes = false(n, 1);
  Proba_OD_decrease = zeros(n + 1, 1);
```

17

```matlab
    Proba_NODE_removal = zeros(n + 1, 1);

    Proba_OD_decrease(1) = sum(OD(:)) / sum(OD(:));
    Proba_NODE_removal(1) = 0;
    Critical_node = NaN;

    for step = 1:n
        node = Node_order(step);
        removedNodes(node) = true;

        Adj(node, :) = 0;
        Adj(:, node) = 0;

        [OD_updated, disconnected] = updateOD(OD, Adj);
        Proba_OD_decrease(step + 1) = sum(OD_updated(:)) / sum(OD(:));
        Proba_NODE_removal(step + 1) = step / n;

        if isnan(Critical_node) && disconnected
            Critical_node = node;
        end
    end

Proba_OD_decrease = Proba_OD_decrease(1:step + 1);
Proba_NODE_removal = Proba_NODE_removal(1:step + 1);
end
```

5.2 New plots

Using the percolation results for node failures, we can visualize how the network's ability to serve passenger demand decreases as nodes are progressively removed.

```matlab
figure;
hold on;
plot(Proba_NODE_removal, Proba_OD_decrease_low_to_high, '-o', 'LineWidth', 2, 'DisplayName',
plot(Proba_NODE_removal, Proba_OD_decrease_high_to_low, '-s', 'LineWidth', 2, 'DisplayName',
hold off;

xlabel('Proba\_NODE\_removal', 'Interpreter', 'none');
ylabel('Proba\_OD\_decrease', 'Interpreter', 'none');
title('Impact of Node Failures on Network Resilience');
```

18

```
legend('Location', 'southwest');
grid on;
```
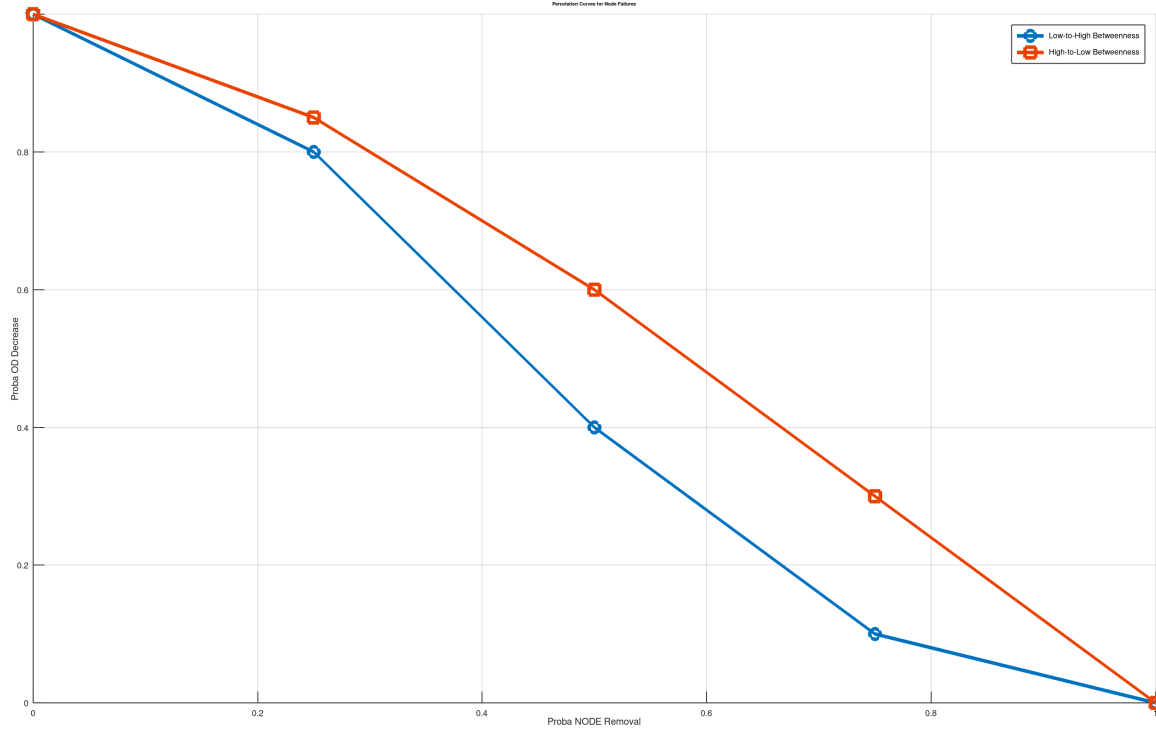

Figure 2: Plot of the percolation process using different quality matrices

Figure 2 illustrates the impact of node failures on the network's ability to meet passenger demand. Two scenarios are shown:

1. **Low-to-High Betweenness Centrality Removal (blue curve):**

   - In this scenario, nodes are removed in order of increasing betweenness centrality.
   - The curve shows a gradual decrease in network performance as less critical nodes are removed first.
   - The network maintains a higher fraction of feasible passenger demand during the early stages of node removal, indicating its ability to withstand the failure of less critical nodes without significant loss of functionality.
   - However, as critical nodes (those with higher betweenness centrality) are eventually removed, the decline in passenger demand becomes more pronounced.

2. **High-to-Low Betweenness Centrality Removal (red curve):**

- In this scenario, nodes are removed in order of decreasing betweenness centrality, starting with the most critical nodes.
- The curve shows a sharp and immediate drop in network performance as the removal of high centrality nodes disrupts key pathways in the network.
- The network becomes significantly less resilient, with passenger demand dropping rapidly as critical nodes are removed early in the process.

The key findings from the node failure analysis are as follows:

1. **Impact of Node Importance:**

   - The stark difference between the two curves highlights the importance of high-centrality nodes in maintaining network functionality.
   - Protecting these critical nodes is essential to maintaining passenger demand during disruptions.

2. **Resilience to non-critical failures:**

   - The gradual decline observed in the low-to-high removal scenario underscores the robustness of the network to the failure of non-critical nodes.
   - This suggests that the network can continue to function effectively even as peripheral or less connected nodes are removed.

3. **Vulnerability to targeted attacks:**

   - The sharp decrease in the high-to-low removal scenario reveals the vulnerability of the network to targeted attacks on critical nodes.
   - This underscores the need for strategic planning and hardening of high-centrality nodes to increase resilience.

The following practical recommendations can be derived from the analysis of node failures:

1. **Prioritize high-centrality nodes:** The analysis shows the outsized impact of high-centrality nodes on network performance. Allocating resources to protect or add redundancy for these nodes can significantly improve resiliency.
2. **Adaptive Response Strategies:** Real-time monitoring of node centrality metrics can support adaptive responses to disruptions, such as rerouting passenger flows or prioritizing restoration of high-centrality nodes.
3. **Designing Robust Networks:** Incorporating redundancy into paths that include high-centrality nodes can mitigate the risk of network fragmentation due to their failure.

This analysis provides actionable insights for transportation network planners, guiding strategies to improve network resilience to both random failures and targeted attacks on critical nodes.

5.3 About resilience

The analysis of the public transportation network using percolation theory provides valuable insights into its resilience under various link and node failure scenarios. The study highlights the network's ability to maintain functionality despite disruptions, as well as its vulnerability to targeted failures of critical components. The resilience of the network is evident in the gradual degradation of performance as non-critical links or nodes are removed, particularly in scenarios where components are removed in increasing order of importance, such as low-to-high betweenness centrality. This demonstrates the robustness of the network to small perturbations that do not significantly affect its overall connectivity.

However, the vulnerability of the network becomes apparent when critical components are targeted. The rapid performance degradation observed when removing high-to-low betweenness centrality highlights the disproportionate importance of certain nodes and links. These high-centrality components serve as bottlenecks and their failure results in significant fragmentation and loss of functionality. This vulnerability underscores the need for targeted interventions to strengthen these critical points and ensure the overall resilience of the network.

The use of quality matrices, particularly those based on betweenness centrality, underscores the importance of identifying and prioritizing critical network components. The highest alpha value ($\alpha_c = 0.555$) obtained in the betweenness centrality-based percolation scenario reflects the improved performance of the network when these critical links are preserved. On the other hand, the inverse travel-time-based quality matrix, which resulted in a lower alpha value ($\alpha_b = 0.469$), demonstrates the challenges posed by the uneven distribution of link quality. These results indicate that resilience is not only a function of connectivity, but also depends on the strategic allocation of resources and prioritization of critical components.

Analysis of node failures further reveals the network's reliance on high-centrality nodes to maintain global connectivity. The severe performance degradation during targeted node removals underscores the importance of protecting these critical nodes. By reinforcing or introducing redundancies for high-centrality nodes, network planners can significantly improve the resilience of the transport system to both random failures and targeted attacks.

In conclusion, the study highlights the importance of designing resilient public transportation networks by balancing structural robustness with functional efficiency. The findings demonstrate the critical role of quality metrics and targeted strategies to mitigate the impact of disruptions, ensure service continuity and maintain passenger demand during adverse events. This comprehensive assessment provides a basis for informed decision-making and strategic planning to improve the resilience of urban infrastructure systems.

6 References

Hamedmoghadam, Homayoun, Mahdi Jalili, Hai Vu, and Lewi Stone. 2021. "Percolation of Heterogeneous Flows Uncovers the Bottlenecks of Infrastructure Networks." *Nature Communications* 12 (February): 1254. https://doi.org/10.1038/s41467-021-21483-y.