Identifying Sensitive Components in Infrastructure Networks via Critical Flows

James Williams, Department of Computer Science, University of Toronto

September 2019

Abstract

This paper introduces a novel set of component importance measures that are based on the concept of critical flow. Various research communities have developed techniques for identifying critical components of networks. The methods in this paper extend previous work on flow-based centrality measures by incorporating insights from both critical infrastructure protection and urban planning. The motivation is to provide municipalities with a means of reasoning about the impact of urban interventions. An infrastructure system is represented as a flow network in which demand nodes are assigned both demand values and criticality ratings. Sensitive elements in the network are those that carry critical flows, where a flow is deemed critical to the extent that it satisfies critical demand. A method for computing these flows is presented, and its utility is demonstrated by comparing the new measures to existing flow centrality measures. The paper also shows how the method may be combined with a simple form of reliability analysis.

1 Introduction

This paper describes a set of *component importance measures* ("CIMs") that allow system components to be ranked according to their role in the delivery of resources to critical locations. These measures can be viewed as variants of *flow centrality* [19] that are customized for flow networks in which demand nodes are assigned both demand values and criticality ratings. The paper describes a general method for computing critical flow CIMs, as well as a simple embodiment that demonstrates the various stages of computation.

The perspective on infrastructure criticality adopted in this work is demand-based, motivated by an interest in *decision support systems* ("DSS") [25] for use by municipal governments (see [21, 9]). As urbanization continues at a rapid pace [52], cities are experiencing significant increases in population and concomitant increases in demand for goods and services. The resulting capacity issues are a major source of risk for legacy infrastructure systems due to the fact that they are typically extremely expensive to alter (e.g., the Thames Tideway tunnel project [51]).

In addition to population growth, various interventions have the potential to alter patterns of demand, and therefore to change the distribution of flows within infrastructure systems. For example, the addition of a new demand node (e.g., subdivision) to an existing infrastructure network will change existing flow patterns and potentially alter the set of components that are relied upon to deliver resources to existing critical locations. As a second example, maintenance activities can reduce the capacity of existing network components, leading to similar consequences. In the worst case, critical locations may experience reduced access to resources.

This paper provides a method for identifying sensitive portions of an infrastructure system according to the degree to which those portions carry flow to critical locations (i.e., critical flows). The idea of identifying locations or services as critical is not novel; prior art exists, for instance, in the critical infrastructure protection literature (e.g., [4]). Unlike previous work, however, the method presented in this paper is intended for use in urban decision support systems (e.g., for maintenance scheduling). A satisfactory technique for this application must allow for the analysis of highly detailed networks that represent infrastructures at a parcel/lot level.

Of course, infrastructures do not exist in isolation. Interdependencies between infrastructure systems are ubiquitous, leading to complex failure patterns that are much more difficult to predict [3]. The resulting 'systems-of-systems' [15] are subject to additional risks that arise from the complex interaction of their component systems, including cascading failures [46]. Nevertheless, in the interests of brevity this paper focuses on a single infrastructure system at a single moment in time.

The method presented in this paper consists of four elements: (1) a representation of an infrastructure system in the form of a (capacitated, potentially cyclic, spatial) network that contains supply and demand nodes; (2) demand values and criticality ratings for each demand node in the network; (3) a means of calculating flows on the network, and; (4) a means of determining the probability that a given network component carries flow to a given demand node. As an optional fifth element, a modeler may also provide a *performance measure* — a means of calculating system performance. There are many different ways of providing these elements, and the goal of this paper is to demonstrate the most simple means of doing so.

The structure of this paper is as follows. Section 2 provides a concise review of background information on network component importance measures, as well as a discussion of the most relevant applications to infrastructure systems. Section 3 discusses the rationale for introducing flow criticality. Section 4 provides a simple instantiation of the method described above, while Section 5 presents a comparison with other centrality measures and a demonstration of how reliability analysis might be integrated with critical flow modeling. The paper closes with numerous suggestions for future work.

2 Background and Previous Work

As mentioned above, many research communities have proposed *component importance measures* ("CIMs") [2] for infrastructure systems. Examples include network efficiency [35], flow vulnerability [43], and flow capacity rate [39]. The CIMS of interest in this paper are the *centrality measures*, which have been developed to identify the most central components in a network (see, for example, [48, 7, 8, 31, 38]). Since numerous centrality measures have been proposed, they are typically categorized (e.g., [47, 43]):

1. *Nearness measures*: which calculate a component's centrality by means of its proximity to other components. Examples include: degree centrality [38], closeness centrality [6], residual closeness centrality [14], information centrality [29], and evidential centrality [54].

2. *Betweenness measures*: which consider components to be central to the extent to which they stand between other components as intermediaries. Examples include: (shortest path) betweenness centrality [19], (general path) betweenness centrality [26], load centrality [22], and random walk betweenness centrality [37].

3. *Dynamical measures*: which examine both topology and dynamical processes situated on the network. Examples include: flow centrality [20], traffic load centrality [36], random walk centrality [40, 37], routing betweenness centrality [16], dynamical influence [28], efficiency centrality [53], and percolation centrality [45].

Of particular importance to this paper is the notion of *flow centrality* [20]. Consider a simple network with nodes/vertices V and links/edges E. A node v is considered to be *between* other nodes u and w to the extent that the maximum flow between u and w depends on v. For $u, v, w \in V$, let $m_{u,w}$ be the maximum flow between u and w, and let $m_{u,w}(v)$ be the maximum flow between u and w that depends on v.

Then the **flow centrality** ("FC") of a node $v \in V$ is defined as the degree to which the maximum flow between all unordered pairs of nodes depends on v:

$$C^F(v) = \sum_{u \neq w \neq v} m_{u,w}(v)$$

This can be normalized by dividing the flow that passes through v by the total flow between all other pairs of nodes, yielding the percentage of flow that depends on v:

$$C'^F(v) = \frac{C^F(v)}{\sum_{u \neq w \neq v} m_{u,w}}$$

2.1 Flow-based CIMs Applied to Infrastructure

Various research communities have applied centrality measures to the study of infrastructure systems (see, for example, [34, 55, 12, 32]). Recent work has focused on dynamical CIMs, including those based on network flows. Comparisons between topological and flow-based methodologies have appeared in several recent studies (e.g., [57, 42, 43]).

Probabilistic estimates of reliability in a flow network were provided by Kansal and Kumar [27]. They deemed a network to be *reliable* to the extent that demand nodes have at least one path to the source, given component failures. To determine reliability, an *appended spanning tree* data structure was used to identify a set of disjoint spanning trees. For a given failure scenario, each of these spanning trees T_i was inspected to determine if demand nodes have at least one path to the source. The results were then aggregated to create a measure of reliability for the scenario.

A method for ranking vulnerabilities in a directed, capacitated, flow network was introduced by Michaud and Apostolakis [33]. In their approach, damage scenarios are created by disabling one of the edges. Each scenario is evaluated by determining its impact on the *supply coverage* (supply/demand ratio) at each sink. A greedy, path-based algorithm finds the geodesic with maximum capacity between a sink and a source, and reserves the demand from the sink against the capacity of each edge and vertex of the geodesic. (If capacity constraints are violated, there is a supply problem for the given sink). Multi-attribute decision theory is used to calculate the *disutility* of the scenario.

Dunn and Wilkison [17] defined a CIM for a directed, uncapacitated, flow network. To identify critical nodes: (1) a synthetic baseline network G_B was generated, using nodal degree to identify a unique supply node and to assign demand values; (2) hydraulic simulation was used to compute flows; (3) a 'roving supply node' simulation was performed. In each iteration, a single demand node d was removed from G_B, yielding $G_B - \{d\}$, and hydraulic simulation is used again. After iterating through all demand nodes, a new supply node was chosen and the process was repeated until all combinations were covered. The impact of each scenario j was $\sqrt{(\Delta_{1j})^2 + (\Delta_{2j})^2 + \cdots + (\Delta_{nj})^2}$, where Δ_{ij} is the difference in flow for node i between G_B and $G_B - \{i\}$.

A hierarchical approach to evaluating the robustness of capacitated flow networks was provided by Ferrario et al [18]. In their method, a network is represented as a multi-level flow model in which each demand node is assigned a *priority level*. Reliability evaluation is carried out through a Monte Carlo procedure [56] in which a system configuration is created by sampling the link capacities from their probability distributions. For each configuration, a greedy, surplus-based algorithm pushes flow through the network, satisfying higher priority nodes first. The results from numerous such configurations are used to estimate the probability distribution of the amount of flow delivered to each demand node at equilibrium. The network is *robust* to the extent that sufficient levels of flow reach the demand nodes despite capacity degradation of the links.

Finally, flow-based vulnerability measures for a capacitated and directed network were developed by Nicholson et al [39]. The goal of their analysis was to identify a set of list of network edges that can be hardened against damage from geographically-situated disruptions. In their approach, performance of the network is measured by computing an *all-pairs average network flow* by means of a minimum cost network flow algorithm. Several edge importance measures are considered, including: (1) *all-pairs max-flow edge count*, the total number of times a given edge is utilized in all s-t pairs max-flow problems, and; (2) *edge flow centrality*: sum of flow on the edge for all possible s-t pair max-flow problems, normalized by the sum of all-pairs max-flows.

3 Critical Flows

One of the key claims of this paper is that changes in demand can lead to significant changes in the distribution of flows. **Figure 1** illustrates this claim by means of a simple example. In network (A), demand node d_1 requires 10 resources from the sink S. There are two paths from d_1 to S, and each edge on each path has a capacity of 6 units. A maximum flow satisfying the demand at d_1 can be achieved by sending 5 units down each path. (In the example, S is assumed to have sufficient capacity, and the flow is assumed to balance when exiting a node).

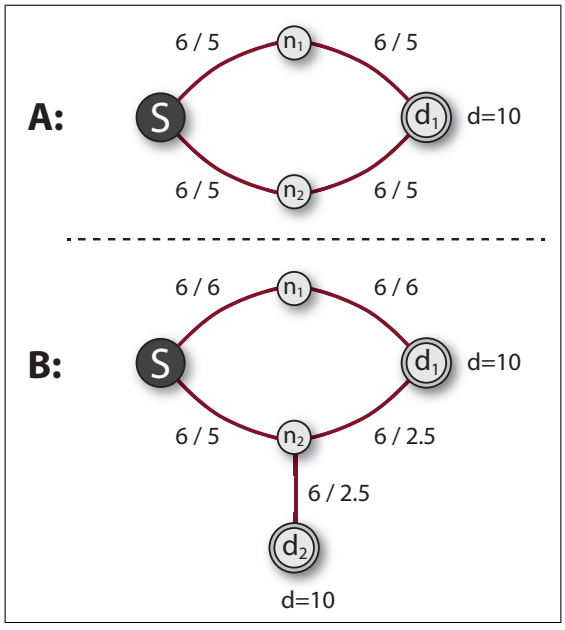


Fig. 1. Introducing a new node d_2 to an existing network (A) can change the flow to existing locations. Capacities and flow values are given as X/Y where X is the capacity and Y is the flow value.

Referring again to **Figure 1**, network (B) is created from network (A) by adding an additional demand node d_2 that seeks 10 units from the source S. The maximum flow solution supplies 2.5 units to d_2 and $6 + 2.5 = 8.5$ units to d_1. In contrast to (A), the upper path through n_1 is now more important for d_1 (i.e., it delivers 6 units, compared to 2.5 units for the path through n_2).

Admittedly, capacity constraints played a key role in this contrived example. One can, however, construct networks in which: (1) capacity constraints are not limiting, and; (2) re-routing of flows occurs in a similar fashion. In **Figure 2**, each link in the network has sufficient capacity to satisfy demand. With the introduction of d_2, the total demand is 20, which is no greater than the capacities of the edges. Two maximum flow solutions are shown in (B) and (C). In the latter, flow from d_1 is routed as in (A), while in the former the same flow is routed exclusively through n_1.

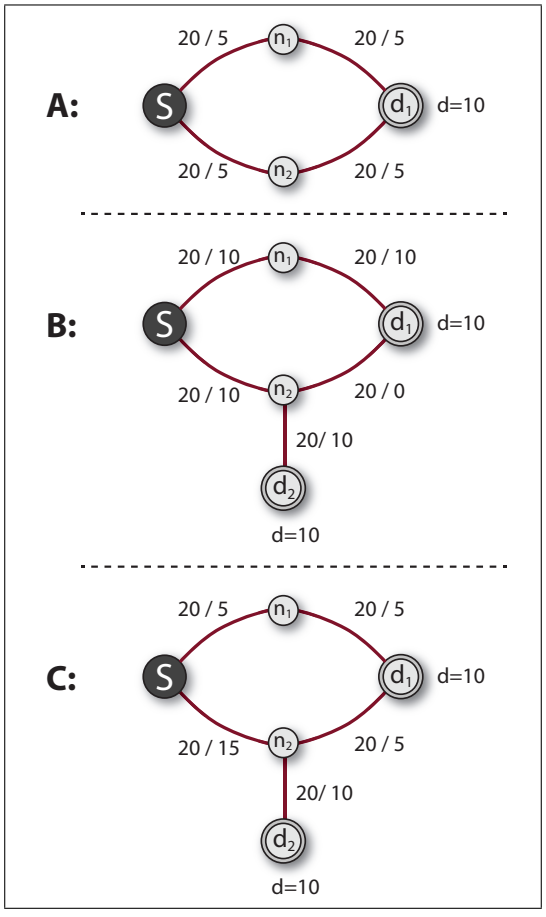


Fig. 2. Two maximum flow solutions (B) and (C) obtained after adding a new node d_2 to an existing network (A). Capacities and flow values are given as X/Y where X is the capacity and Y is the flow value.

Changes in demand also have the potential to redistribute *critical flows* (i.e., the flows in the network that serve critical locations). Part A of **Figure 3** shows a capacitated network in which demand node d_1 has been deemed to be critical. Part B shows a maximum flow solution for the network, with edges shaded to the degree to which they carry flow to d_1. The flow to d_1 is balanced between the paths (S, n_1, d_1) and (S, n_2, d_1), with each path carrying 7.5 units of flow. Furthermore, the flow to all demand nodes is approximately balanced, and the internal edge (n_1, n_2) is not used.

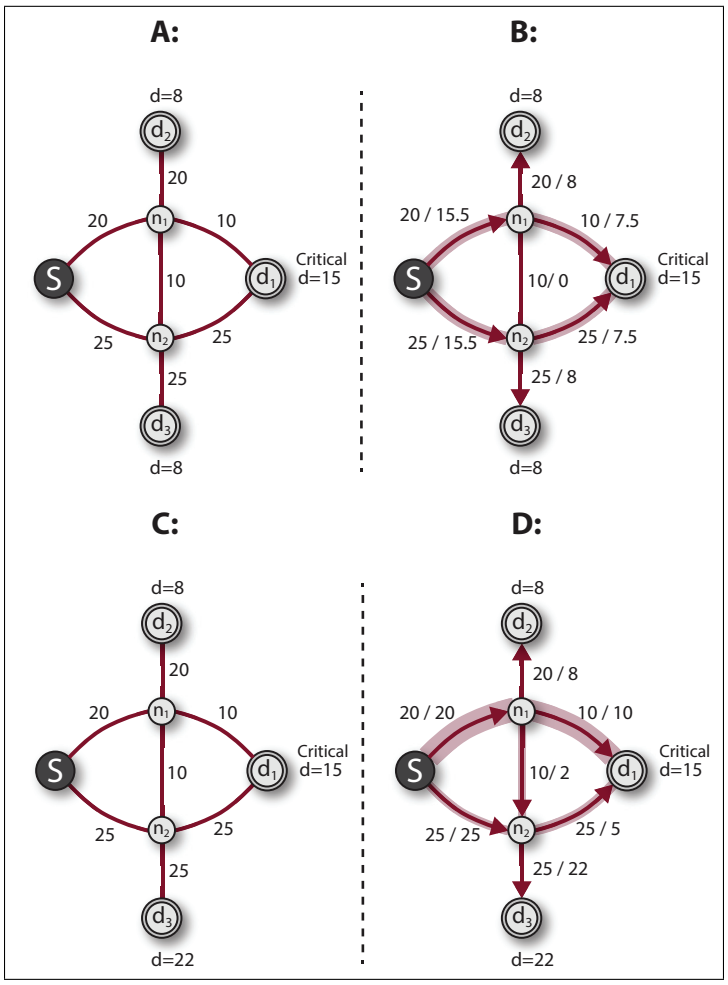


Fig. 3. A capacitated network (A) and a maximum flow solution (B) showing critical flows. In the network, demand node d_1 is deemed a critical location. Capacities and flow values are given as X/Y where X is the capacity and Y is the flow value, and demands are indicated as $d = x$. (C) shows the same network as (A), but node d_3 has greatly increased demand. (D) shows how the upper path now carries the bulk of critical flow to d_1.

Changing the demand pattern drastically alters the distribution of critical flows. Part C of **Figure 3** shows a modified network in which the demand at d_3 has been increased to 22. Part D shows the resulting change in critical flow patterns: the majority of flow to critical location d_1 is now routed through the upper path (S, n_1, d_1). If, for instance, those network elements were unreliable or scheduled for maintenance (i.e., potentially unavailable), the introduction of new demand at d_3 might be far more risky than initially anticipated.

3.1 Representing Criticality

In order to quantify the effect of flow redistribution on critical demand, it is useful to introduce appropriate measures. There are many possibilities for defining CIMs based on critical flows. One major design decision is the *representation of criticality*. There are three main choices:

1. **Binary representation**: in which a demand node $d \in V$ is deemed to be critical or not according to a predicate that takes on 'true' or 'false' values. This is the most simple option, but it leaves an analyst with no ability to model degrees of criticality.

2. **Categorical representation**: in which a demand node d's criticality is selected from a finite set of categories (e.g., $\{low, med, high\}$). This is probably the most intuitive option, although the criticality levels require transformation for use in equations or for certain types of statistical inference (e.g., regression).

3. **Continuous representation**: in which a demand node d's criticality is selected from a subset of the real numbers $\mathbb{R}$. A simple approach is to normalize each value (e.g., $Cr(d) \in [0, 1]$). An alternative representation, inspired by Location Theory (see [24]), is the following: the entire set of demand nodes may be placed into a vector $\vec{d} = (d_1, d_2, d_3, \ldots, d_k)$. A corresponding (stochastic) criticality vector $Cr(\vec{d}) = (Cr(d_1), Cr(d_2), Cr(d_3), \ldots, Cr(d_k))$ may then be constructed in which $\sum_{i=1}^{k} Cr(d_i) = 1$.

A second design decision concerns the manner in which component importance should be calculated. Again, there are several obvious candidates:

1. **Absolute critical flow**: a component's importance is measured by the aggregate amount of critical flows that pass through it. For instance, if component c_i carries 5 units of critical flow and c_j carries 6 units of critical flow, the latter is deemed to be more important. No distinction is made as to where the critical flow is routed (i.e., all critical locations are considered equally important).

2. **Percentage of total flow**: a component's importance is measured by the percentage of its flow that is critical. If component c_i carries 100 units of flow (of which 5 is critical) and component c_j carries 50 units of flow (of which 2 is critical), the latter is deemed to be more important. Again, no distinction is made as to where the critical flow is routed.

3. **Critical Location Count**: a component's importance is measured by the number of critical locations that receive flow through paths in which that component appears. No weighting is made as to the amount of flow directed to each location; it is enough that even a tiny bit of critical flow passes through the component.

4. **Expected Critical Flow**: a component's importance is measured by the expected amount of resource that it supplies to any of the k critical locations. In this case, the boolean representation of criticality is used, with no differentiation between degrees of criticality.

5. **Weighted Expected Critical Flow**: a component's importance is measured by the expected amount of resource that it supplies to critical locations, where locations are weighted by their importance. In this case, either the categorical or continuous representations of criticality must be used (with an appropriate conversion for the categorical variant).

These concepts can also be used to provide overall measures of network performance in a similar manner to various centrality measures, such as efficiency [30], flow centrality [20], edge reliability [55], and reliability centrality [12].

4 Component Importance Measures based on Critical Flows

This section demonstrates how the concept of critical flow may be used to define novel *component importance measures* ("CIMs"). In the interests of brevity, two examples are presented: (1) CIMs that are elaborations of classical flow centrality, and; (2) probabilistic CIMs that are applicable to systems that are not in equilibrium.

4.1 Variants of Flow Centrality

The most obvious means of applying the concept of critical flow to CIMs is to formulate a new measure of *flow centrality* [19]. The underlying idea is that a node is deemed central to the extent that it is involved in the flow of resources to critical locations. Examples of such measures include:

1. **(Binary) Critical Flow Centrality**: let $D_C \subseteq V = \{d_1, d_2, \ldots, d_k\}$ be the set of demand nodes in G that are deemed critical. and let $S = \{s_1, s_2, \ldots, s_p\}$ be the set of source nodes in G. Let $m_{s,d}$ be the maximum flow between $s \in S$ and $d \in D_C$. (In the case where there is a single source, this is the same as the flow entering d). Furthermore, given node $v \in V - D_c$, let $m_{s,d}(v)$ be the maximum flow between s and d that depends on v. Then the **(binary) critical flow centrality** of a node $v \in V - D_c$ is the degree to which the maximum flow between critical demand nodes and source nodes depends on v:

$$C^{BCF}(v) = \sum_{s \in S, d \in D_C} m_{s,d}(v)$$

This can be normalized by considering the total critical flow on the network:

$$C'^{BCF}(v) = \frac{C^{BCF}(v)}{\sum_{s \in S, d \in D_C} m_{s,d}}$$

A similar definition can also be formulated for edges. While non-critical demand nodes are ignored in the summations. they are still used implicitly to compute the overall flow.

2. **Weighted Critical Flow Centrality**: let the set of demand nodes in the network be $D = \{d_1, d_2, \ldots, d_m\}$, and consider function $c_r : D \rightarrow \mathbb{R}$ that maps a demand node $d \in D$ to a *criticality value* $c_r(d)$. Then the **weighted critical flow centrality** of a node $v \in V$ is:

$$C^{WCF}(v) = \sum_{s \in S, d \in D} c_r(d) m_{s,d}(v)$$

This can be normalized as follows:

$$C'^{WCF}(v) = \frac{C^{WCF}(v)}{\sum_{s \in S, d \in D_C} c_r(d) m_{s,d}}$$

A particularly useful case is where $c(d) \in [0, 1]$, with $\sum_{d \in D} c(d) = 1$. Any assignment of criticality values $c(d) \in \mathbb{R}^+$ can be converted to this representation by normalizing each criticality value $c(d)$ by the total criticality of the network $\sum_{d \in D} c(d)$. Note also that this centrality measure will work with categorical values that have been encoded as numbers.

Computing these centrality measures is not as intuitive as computing other network centrality measures such as betweenness [38] and flow centrality [20]. Note also that algorithms for computing basic centrality measures can be found in a comparatively small number of sources [e.g., [36, 11, 37, 10, 23]].

4.2 Probabilistic Critical Flow Measures

The critical flow centrality measures discussed in the previous section are based upon maximum flows. This section presents a probabilistic CIM that relaxes this assumption, and also allows demand nodes to have both demand values and criticality levels. As above, let the set of demand nodes in the network be $D = \{d_1, d_2, \ldots, d_m\}$, and consider function $c : D \rightarrow \mathbb{R}^+$ that maps a demand node $d \in D$ to a **criticality value** $c(d)$. Let $\mathcal{D}$ be a family of functions $\delta_i : D \rightarrow \mathbb{R}^+$, each of which assigns a **demand** $\delta(d) \in \mathbb{R}$ for each demand node $d \in D$. Finally, let an **assignment** A be a concrete choice of function δ_i from $\mathcal{D}$, and let $f_A(d)$ be the **flow received** by $d \in D$ under A.

The **flow** in network G given assignment A is the aggregate of all flows reaching the demand nodes:

$$F_A(G) = \sum_{d \in D} f_A(d)$$

The **critical flow** in network G given assignment A is the set of flows reaching the demand nodes, weighted by criticality:

$$F_A^C(G) = \sum_{d \in D} f_A(d) c_r(d)$$

Let $f_A(v_i, d_j)$ be the flow that reaches $d_j \in D$ from node $v_i \in V$ under assignment A. In a probabilistic model, what we wish to compute is the expected amount of flow from v_i that reaches d_j:

$$E[f_A(v_i, d_j)]$$

Referring to Figure 4, a flow network (A) with a single critical demand node d_4 is shown. Given a flow (B), the expected amount of flow delivered by a network element can be calculated by inspection. For example, $f_A(n_2, d_2) = 12$, since there is a single path between these elements. In contrast, $f_A(n_2, d_1) = 0$, since there is no path between these elements.

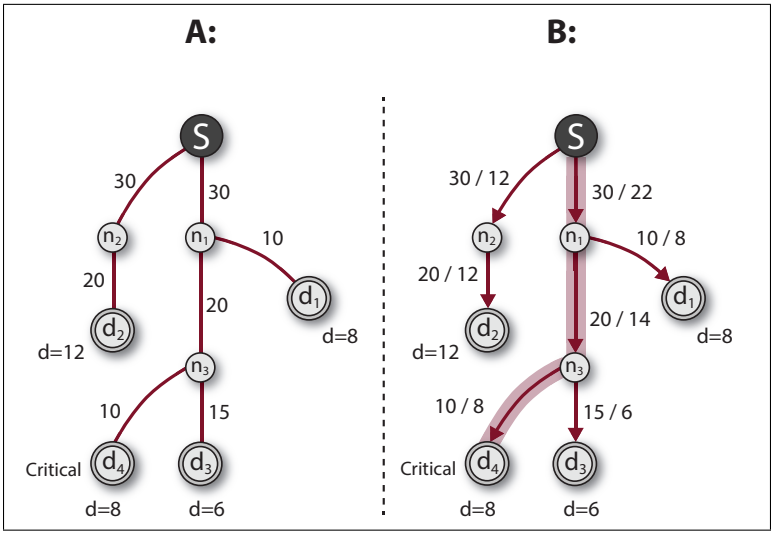


Fig. 4. A flow network (A) with a single critical demand node. Given a flow (B), the expected amount of flow to each demand node can be computed. For instance, the expected amount of flow from n_1 to d_4 is 0, since there is no path between those elements.

Assuming that there is a method to compute these expectations, one can define a relevant CIM. For instance, the **(probabilistic) critical flow centrality** ("CFC") of a vertex $v \in V$ is:

$$C^{CF}(v) = \sum_{d \in D} c_r(d) E[f_A(v, d)]$$

This is the sum of all expected amounts of flow being delivered via v to the demand nodes, weighted by their criticality. This quantity may be normalized by an appropriate expression, such as the total flow $F_A(G)$ or the total flow weighted by criticality:

$$C'^{CF}(v) = \frac{C^{CF}(v)}{F_A^C(G)} = \frac{\sum_{d \in D} c_r(d) E[f_A(v,d)]}{\sum_{d \in D} c_r(d) f_A(d)}$$

Variations on these formulas are easy to construct. First, identical definitions can be formulated for edges. Second, one can concoct unweighted versions in order to accommodate the boolean representation of criticality. Third, additional components may be incorporated (e.g., reliability estimates for components, as discussed in Section 6.2). Lastly, the general approach exemplified by these definitions is compatible with the use of *utility functions*, a topic that is not pursued in this paper for reasons of brevity.

5 Computing the Critical Flow Measures

The previous section introduced a set of new *component importance measures* ("CIMs") based on critical flows, providing definitions but no guidance on how they are to be computed. This section presents a method for doing so, focusing on probabilistic *critical flow centrality* ("CFC").

5.1 A Method for Critical Flow Analysis

Since there are multiple choices of critical flow CIMs, and also multiple ways to compute them, the method presented in this paper can be viewed as a general *framework*. The framework is instantiated by supplying means of accomplishing key tasks. The paper demonstrates one such instantiation (the 'present instantiation') in order to support evaluation and testing. The framework for computing critical flow CIMs consists of four elements:

1. a representation of an infrastructure system as a flow network;
2. criticality metadata for nodes (i.e., demand values, criticality ratings);
3. a means of calculating flows;
4. a means of determining the probability that a given network component (node, link) carries flow to a given demand node;

Since these elements can be supplied in numerous ways, there are many possible instantiations ranging from simple to highly complex. More sophisticated variants can be constructed according to the needs of a particular application.

5.2 Network Representation

In the present instantiation, an infrastructure system is represented as a weighted, capacitated, flow network $G = \langle V, E \rangle$ where G is a set of vertices, and $E \subseteq V \times V$ is a set of edges. Each node $v \in V$ is situated in Euclidean space, having **world coordinate** $\vec{w}(v) = (v_x, v_y, v_z) \in \mathbb{R}^3$. (Note: vertices do not have capacities). Each edge $e = (v_i, v_j) \in E$ has a **capacity** $c(e) \in \mathbb{N}$, a **flow** $f(e) \in \mathbb{N}$, and a **length** $l(e) \in \mathbb{R}$ defined as $\|\vec{w}(v_i) - \vec{w}(v_j)\|_2$. Bi-directional relationships, cycles, and self-loops are all permitted.

The network G contains both source (supply) and sink (demand) nodes. The set of **source nodes** is $S = \{s_1, s_2, \ldots, s_p\} \subseteq V$, and the set of **demand nodes** is $D = \{d_1, d_2, \ldots d_k\} \in V$. All other nodes are called *transmission nodes*. A **flow** on G is a real-valued function $f : E \to \mathbb{R}$ on G's edges that obeys three flow properties:

1. **Capacity Constraints**: for all $e = (v_i, v_j) \in E$, we have $f(e) \leq c(e)$.
2. **Skew Symmetry**: for all $e = (v_i, v_j) \in E$, we have $f((v_i, v_j)) = -f((v_j, v_i))$.
3. **Flow Conservation**: for all transmission nodes $v_t \in V - (D \cup S)$, we have $\sum_{v \in V} f((v_t, v)) = 0$.

The **value of a flow** is defined as the flow exiting the source nodes: $|f| = \sum_{v \in V} \sum_{s \in S} f(s, v)$. While a network with multiple source and sink nodes may be reduced to a network with a single sink and source (see [13]), the explicit representation is used throughout this paper.

5.3 Demand Distributions and Criticality

The basic network flow model is augmented with additional metadata. First, a **demand function** $\delta : D \to \mathbb{N}$ maps a demand node $d \in D$ to a demand $\delta(d)$. Second, a **supply constraint function** $f_s : S \to \mathbb{N}$ assigns each source node $s \in S$ a maximum flow $c_f(s)$. Third, a **criticality function** $cr : D \to \mathbb{R}$ maps $d \in D$ to a criticality rating $cr(d)$. There are at least three options for the latter:

1. For a *binary representation* of criticality, $cr(d) \in \{0, 1\}$.
2. For a *categorical representation* of criticality, a suitable encoding is used to map values to real numbers (e.g., 'high' $\to$ 1, 'med' $\to$ 1/2, and 'low' $\to$ 0).
3. For a *continuous representation* of criticality, any mapping may be chosen. In many cases it may be useful to normalize the values so $cr(d) \in [0, 1]$ and $\sum_{d \in D} cr(d) = 1$.

An **assignment** to a network involves assigning supply constraints to all source nodes and demand values to all demand nodes. Theses values may be provided by probability distributions (e.g., those arising from stochastic processes), or via external data (e.g., real world data obtained from smart meters). In this paper, demands are represented as natural numbers.

5.4 Calculating Network Flow

The choice of network flow algorithm depends upon the representation of criticality. For instance, the binary representation of criticality assigns 1 or 0 to each demand node. Perhaps the most intuitive solution strategy for this representation is to use a *2-commodity network flow algorithm* (see [1]) in which one commodity represents critical demand. Each edge in the network will then have an explicit representation of both the critical and non-critical flow passing through it, making it relatively easy to reason about the distribution of critical flows.

A similar approach can be used for the *categorical representation*, in which each demand node is assigned a criticality rating from a (potentially ordered) set of k categories. A k-ary *multi-commodity network flow* can be computed, yielding an explicit representation of the amount of each type of critical flow at each edge. One must, however, be cautious of the computational requirements of this approach.

This paper focuses on the *continuous representation* of criticality, in which each demand node is assigned a real-valued criticality rating. This approach to modeling criticality makes it impossible to represent critical flows through a finite set of commodities. Feasible solution techniques include *network flow algorithms* and *domain-specific simulation techniques* (e.g., hydraulic simulation [41]).

The most intuitive way to create a flow on an infrastructure model is to utilize an algorithm for the MAXIMUM FLOW PROBLEM [1]. A major drawback to this approach is that popular maxflow algorithms do not yield network flows that are *balanced* across competing paths. Figure 5 illustrates this situation. Part (A) shows a maximum flow on a network where the total demand at d_1 is satisfied entirely by the value of the flow. However, all of the flow passes through the uppermost path between the source and sink vertices, leaving the alternative paths unused. In contrast, Part (B) shows a maximum flow of the same value, but with balancing between the three alternative paths from source to sink.

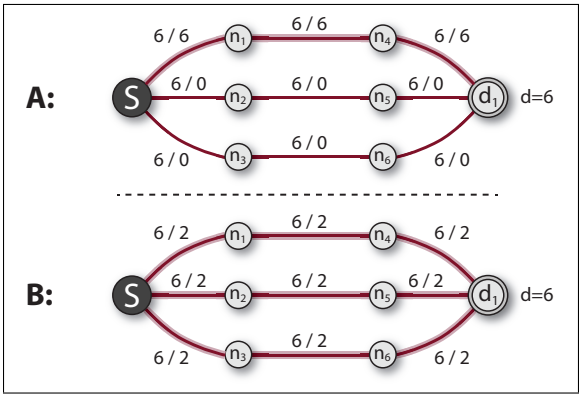


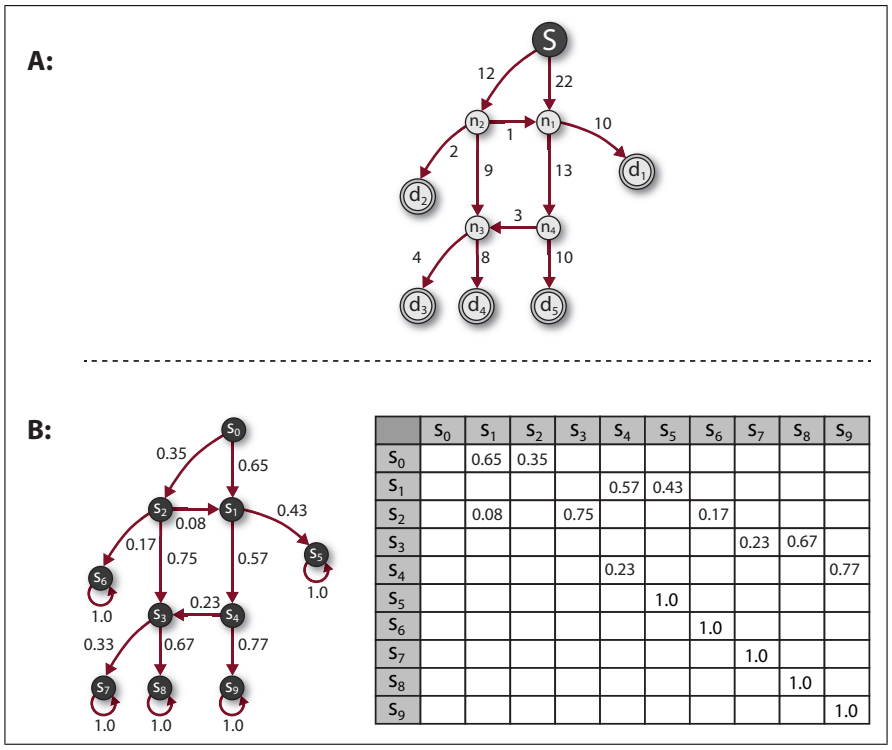
Fig. 5. Two maximum flow solutions on the same network. The first number on each edge is the capacity, while the second number is the assigned flow.

Algorithms that allow for disjunctive constraints exist, as in [44], but they have some drawbacks. Domain-specific methods such as hydraulic simulation [41] are preferable in cases where a realistic flow must be obtained. For simplicity, this paper uses a common maximum flow algorithm that does not provide balancing.

5.5 Calculating Critical Flow from Flow Patterns

Once a flow solution has been computed for the current assignment — in the case of this paper, through the Edmonds-Karp algorithm (see [13]) — CIMs must be computed. In the case of probabilistic CIMs, this involves determining the probability that a unit of flow from component x ends up in demand node $d \in D$. There are numerous ways to accomplish this task.

The most intuitive approach is to reduce the problem to *Markov chain* computation [49, 50]. Figure 6 shows a flow network (A) represented as a Markov chain (B). Each network node appears as a state in the chain. Edges are represented implicitly by Markov chain transitions, while demand nodes are absorbing states in the chain. This representation incurs $O(|V|^2)$ space.



	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
S_0		0.65	0.35							
S_1					0.57	0.43				
S_2		0.08		0.75			0.17			
S_3								0.23	0.67	
S_4					0.23					0.77
S_5						1.0				
S_6							1.0			
S_7								1.0		
S_8									1.0	
S_9										1.0

Fig. 6. A flow network (A) represented as a Markov chain (B) in which network nodes are states in the chain. The sparseness of the transition matrix is typical of infrastructure networks.

An alternative Markov chain is shown in Figure 7. In this variant, both network nodes and network edges are represented explicitly as states in the chain. This representation permits inferences about both node and edge criticality, but the full transition matrix now requires $O((|V| + |E|)^2)$ space.

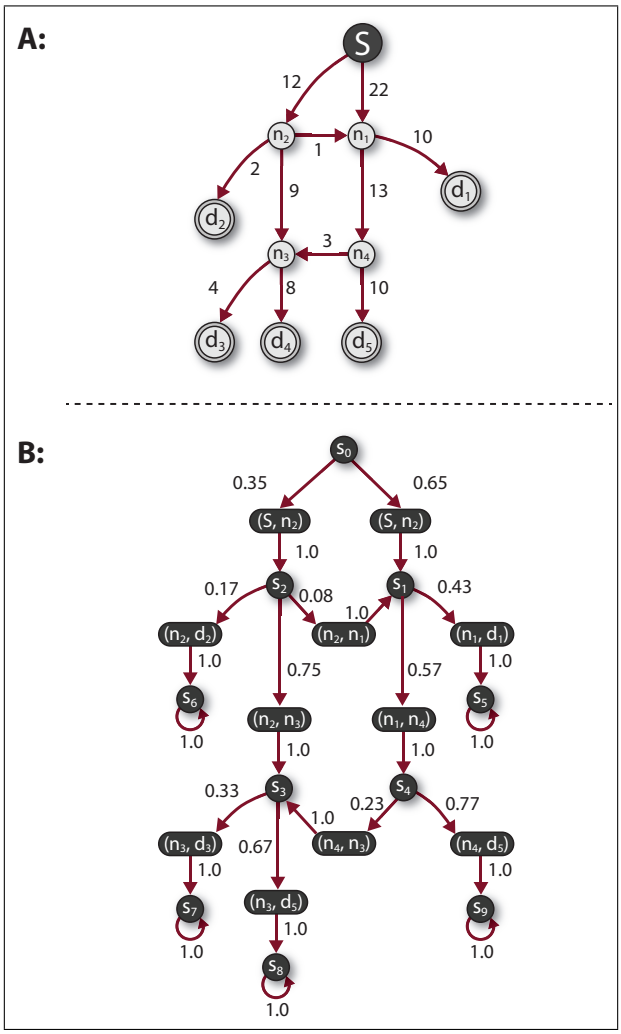


Fig. 7. A flow network (A) represented as a Markov chain (B) in which both vertices and edges are states in the chain. Markov states representing edges are labeled with the edge coordinates from the original network.

No matter which representation is chosen, it is clear that the resulting Markov chain transition matrix will be extremely sparse. In most infrastructure systems, the maximum degree of a vertex is quite small (e.g., in our real-world network, $\max_{v \in V} \deg(v) = 5$, but the average degree is 2.3). In this case, the number of non-zero entries of the transition matrix is $\sum_{v \in V} \deg(v) = O(|V|)$, while the space required will be $O((|V|+|E|)^2) = O(|V|^2)$ for the Markov chain in which both nodes and edges are represented.

Another option is to estimate probabilities from the flow network using a *Time Forward Random Walk* [5]. In this approach, particles are placed on the network at source nodes, following flows until: (1) they reach a demand node, or; (2) until a time threshold is exceeded. This type of solution is a sampling procedure [56], albeit one that permits the explicit paths through the network to be retained.

5.5.1 A Deterministic Algorithm for Computing Probabilities

For simplicity, this paper uses a deterministic variant of a *depth-first search* ("DFS") to directly compute the probability that a unit of resource passing through component c lands in a given demand node d_i. From this probability, together with the total flow passing through c, allows one to compute the expected amount of c's flow that ends up in d_i. Figure 8 illustrates the expected amount of flow delivered to a demand node.

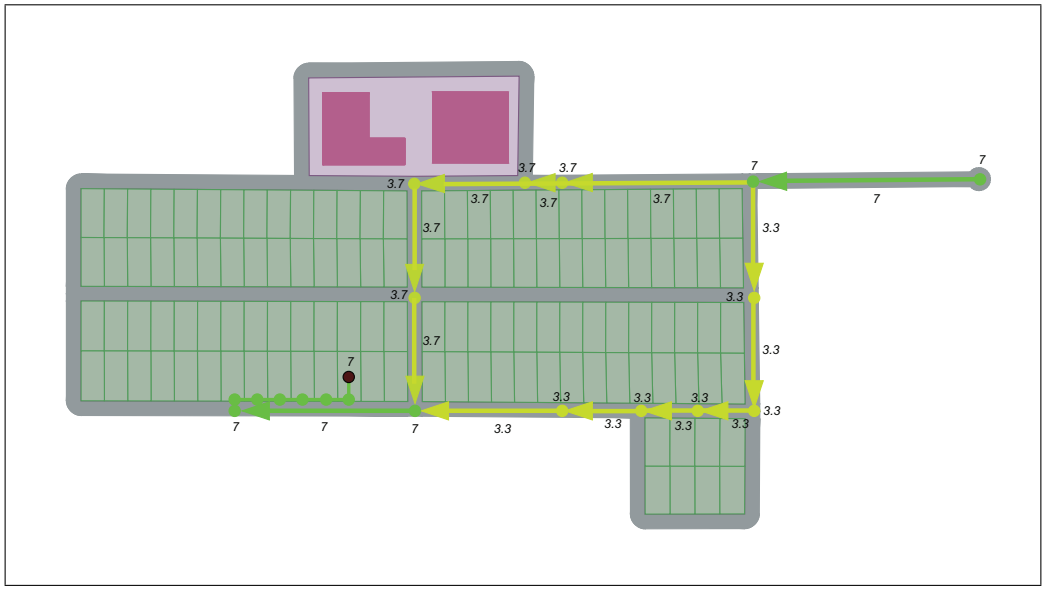


Fig. 8. Calculation of expected amount of flow delivered from source (green) to a demand node (black). Nodes and edges that do not supply resources to the demand node are not shown.

The flow network obtained in the previous step is transformed to a **probability graph**; each vertex in the graph maintains an outgoing edge list whose entries are labeled with the probability that a unit of flow travels down that edge. (If an edge in the flow network has a flow of 0, it does not appear in the probability graph). Demand nodes $d \in D$ in the flow network are referred to as *absorbing nodes* in the induced graph.

Given such a probability graph, the procedure associates with each non-absorbing node v (and each edge e) a **data structure** (e.g., map) that holds the entire set of demand nodes reachable from v (or e). Each entry in this map contains an identifier of a demand node, together with the probability that a unit of flow will reach the demand node from the node or edge to which the map belongs. Metrics can be computed for each node or edge by examining the contents of the map and the attributes of the demand nodes.

```
Function ComputeProbabilities(G)
    Data: G, a probability graph with components (V, E) and absorbing nodes
          D ⊆ V.
    foreach d ∈ D do
        ReverseSearch(G, d)
    end
Function ReverseSearch(d)
    Data: G, as above.
    Data: d, an absorbing node.
    Var excess[]                  // array of numbers ∈ [0, 1] of size |V|
    Var stack                                          // stack of nodes
    excess[d.ID] = 1
    stack.push(d)
    while stack not empty do
        Var v = S.pop()                                  // current node
        Var amt = excess[v.ID]          // amount of probability to push
        foreach incoming edge e of v do
            e.map.IncrementOrAddProbability(d.ID, amt)
            excess[e.src.ID] = amt * e.probability
            stack.push(e.src)
        end
        v.map.IncrementOrAddProbability(d.ID, amt)
        excess[v.ID] = 0
    end
```

Algorithm 1: A Deterministic Algorithm for Computing Flow Probabilities.

Algorithm 1 proceeds by performing a reverse DFS for each demand node $d \in D$, computing the probability that each edge or non-demand node sends flow to d. A given component may be visited multiple times in the course of the search, requiring care to avoid pushing superfluous probability.

The deterministic algorithm presented above is reasonably efficient both in time and space compared to CIMs such as betweenness or flow centrality. In the worst case, the map at each node and edge stores $|D|$ entries, one for each demand node in the network, leading to $O((V + E)D)$ in storage space. As a variant of depth first search, the time required to compute probabilities for each demand node is $O(V + E)$, so the total time required for all demand nodes is again $O((V + E)D)$. In the networks that we study, $V \approx E$ and $D \approx \frac{1}{2}V$, yielding time and and space complexity of $O(V^2)$.

It is important to note that this algorithm does not work for a directed graph that contains *cycles*. The augmenting-path flow algorithm that was used in this paper does not create cycles, but this is not necessarily the case for domain-specific methods (e.g., hydraulic simulation). For flow networks with cycles, alternative techniques must be used.

Once the algorithm has been executed and the probabilities computed for each demand node in the network, the computation of criticality for each network component is simple. The **critical flow centrality** ("CFC") (see Section 4.2) of a vertex is computed as:

$$C^{CF}(v) = \sum_{d_i \in v.map} p_v(d_i) c_r(d_i) f(v)$$

where d_i is a demand node, $p_v(d_i) > 0$ is the probability that flow passing through v reaches d_i, $f(v)$ is the flow passing through v, and $c_r(d_i)$ is the criticality of demand node d_i. (A similar formula holds for edges). Variations on this basic approach are possible, including normalization of the centrality values by the maximum centrality value for the network:

$$C'^{CF}(v) = \frac{\sum_{d_i \in v.map} p_v(d_i) c_r(d_i) f(v)}{\max_{v \in V}(C^{CF}(v))}$$

5.5.2 Summary of Present Instantiation

The present instantiation involves four steps: (1) representing the infrastructure system as a capacitated flow network with demand and source nodes; (2) suppying demand and criticality values for demand nodes (and optionally resource limits for source nodes); (3) computing the network flow for a given configuration of demand values, and; (4) obtaining the induced probability graph from the network flow, computing probabilities for demand-satisfying paths, and calculating criticality measures.

Figure 9 shows these four stages of computation. In the upper left, the basic network model is shown along with demands for each lot and each building. (Criticality values are not shown, but the reader may assume the school and hospital are more critical than other buildings). The upper right shows a flow solution computed on the basic network, while the lower left shows the induced probability graph arising from this flow. (Network edges with 0 flow are elided from the probability graph). Finally, the lower right shows the critical flow metrics, ranging from green (low criticality) to red (high criticality).

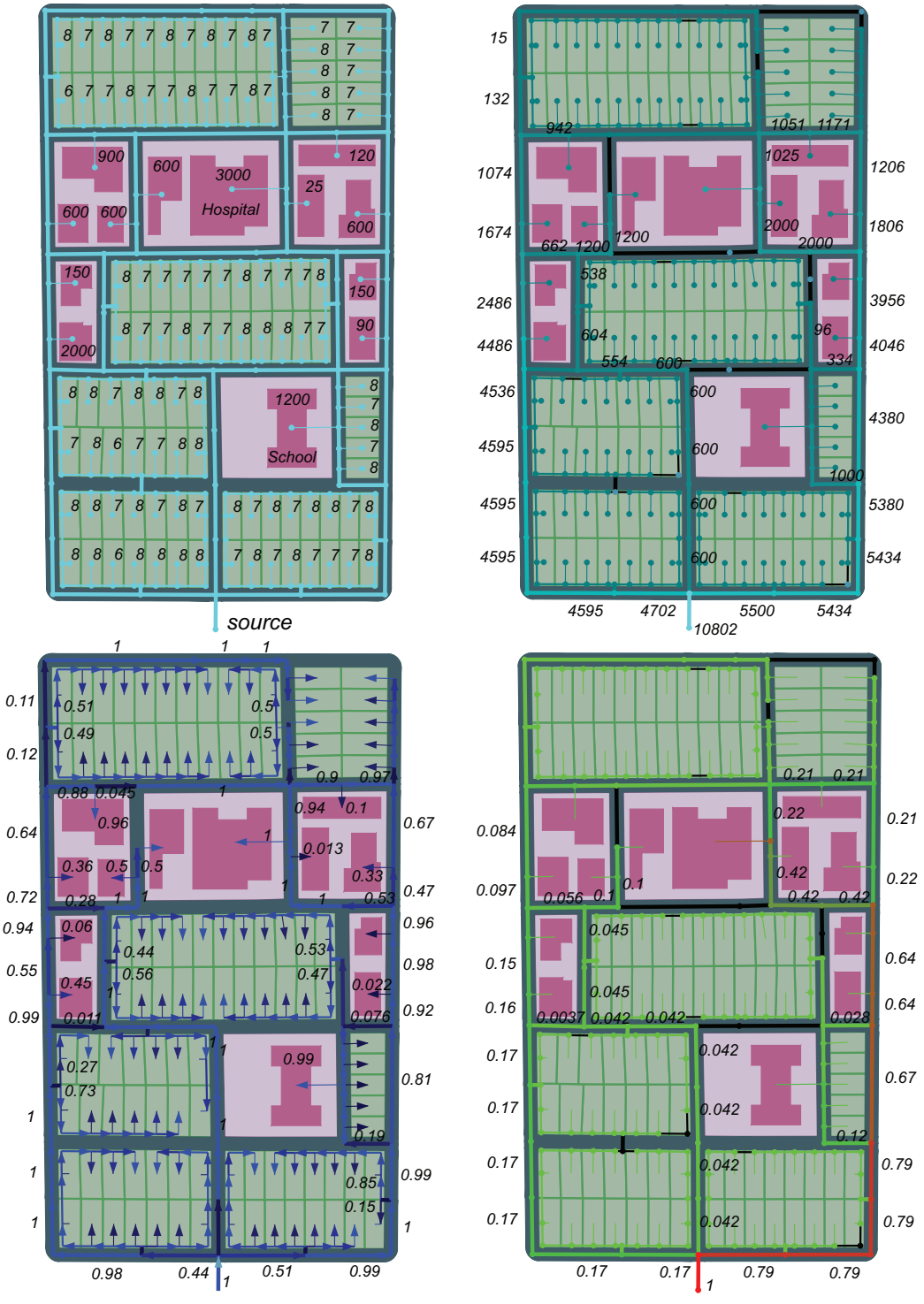


Fig. 9. Four steps in critical flow computation. Upper left: network topology with demands. Upper right: maximum flow. Lower left: induced probability graph. Lower right: critical flow.

6 Evaluation

The utility of the critical flow method presented in this paper is demonstrated in two ways. First, the critical flow method is compared against standard centrality measures using a synthetic model of a size feasible for use in diagrams. Second, the critical flow method is combined with an edge reliability metric to show that it allows criticality and reliability analysis to be combined.

6.1 Comparison with other Metrics

This section compares the CFC measure with two variants of the flow centrality measure: (1) the traditional version of flow centrality ("FC") (discussed above), and (2) a narrower version that is more appropriate for infrastructure networks. The latter is described below.

Flow centrality was originally proposed in the context of social network analysis. In an infrastructure system, it is not clear that it is useful to take account of maximum flows between demand or transmission nodes when the key issues typically concern the relation between sources and sinks. As a result, this paper provides a narrower formulation of flow centrality [and of edge-flow centrality [39]] that is designed for use on infrastructure networks. The **source-sink flow centrality** ("SSFC") of a node v is the degree to which the maximum flow between all source and demand nodes depends on v:

$$C^{SSF}(v) = \sum_{s \in S, d \in D, s \neq d \neq v} m_{s,d}(v)$$

where $m_{s,d}$ is the maximum flow between s and d, and $m_{s,d}(v)$ is the maximum flow between s and d that depends on v. This can be normalized by dividing the flow that passes through v by the total flow between all pairs of source/sink nodes, yielding the percentage of flow that depends on v:

$$C'^{SSF}(v) = \frac{C^F(v)}{\sum_{s \in S, d \in D} m_{s,d}}$$

Both of the flow centrality measures presented above can be defined for edges.

6.1.1 Comparison of Measures

The FC, SSFC, and CFC measures were computed for a small model that was patterned after high density neighbourhoods in Toronto, Canada. Figure 10 shows four views of this basic model. In the upper left, the network is shown along with the demands at each building and lot. The flow resulting from this configuration is shown in the upper right, where black network edges indicate that no flow is present. In the lower left, the SSFC centrality measure has been computed for each edge and vertex. The lower right shows the FC centrality measure.

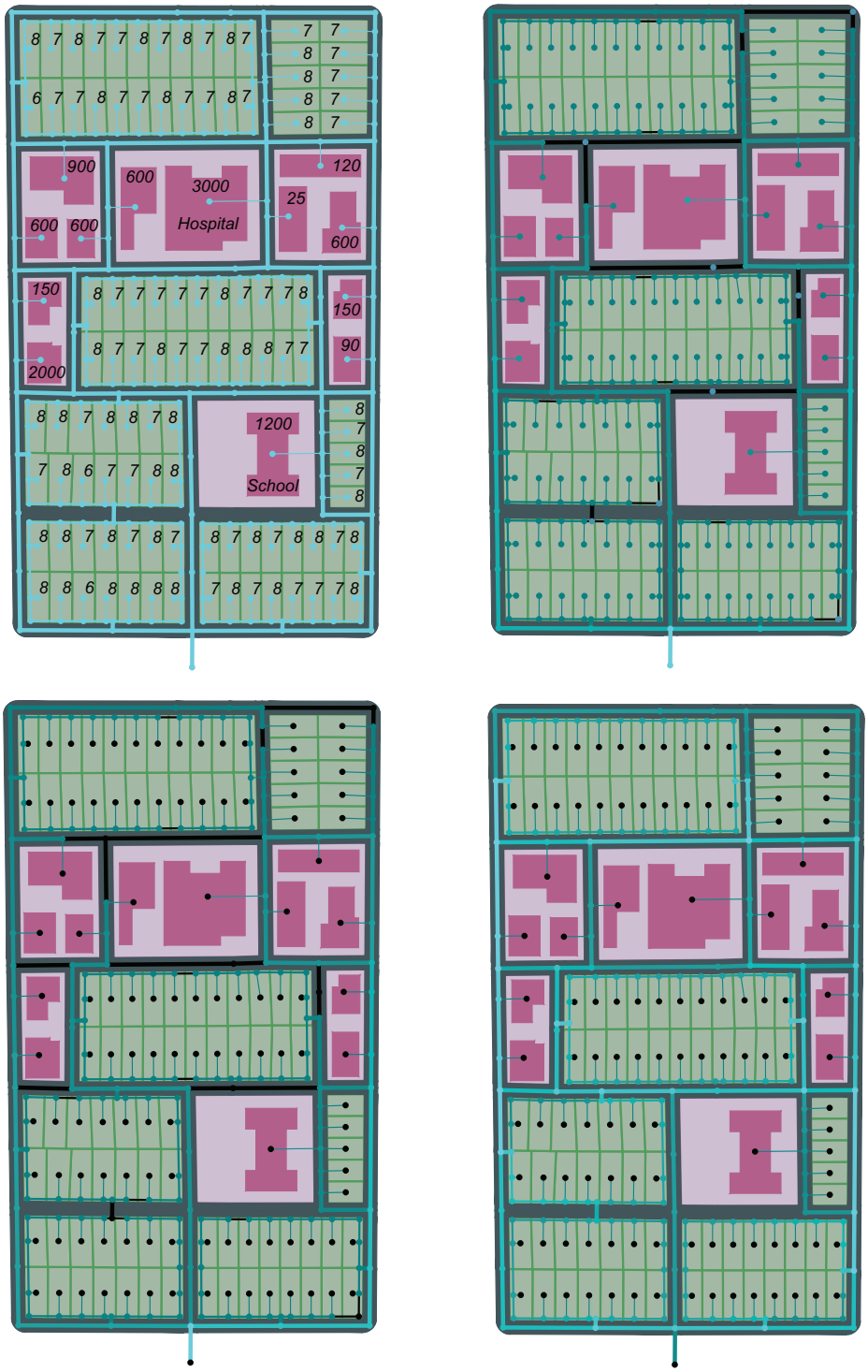


Fig. 10. Comparison of centrality measures. Upper left: empty network showing demands. Upper right: flow upon the same network. Lower left: source-sink flow centrality. Lower right: Freeman's flow centrality.

The FC measure places emphasis on the central nodes in the graph, leaving the sole edge emanating from the source (i.e., the most critical) relatively dark. In contrast, the SSFC is highly correlated with the flow, as shown in Figure 11. The fact that both the SSFC and the flow leave black edges is likely an artefact of the Edmonds-Karp algorithm that was used to produce flows, and something that would be rectified with domain-specific methods.

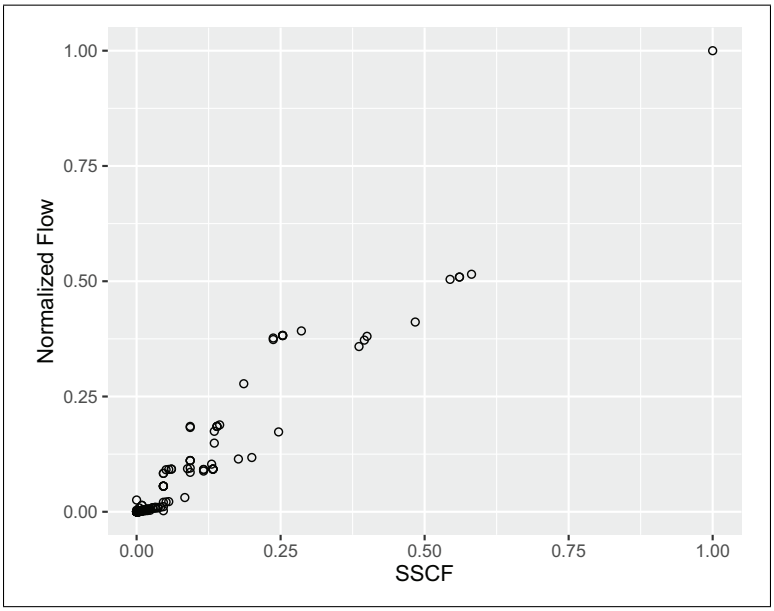


Fig. 11. Correlation between SSFC and flow on the network in Figure 10.

Figure 12 (A) shows the critical flow for the same model and demand/criticality configuration. Criticality levels are displayed in white font for buildings, while lot criticality is elided for simplicity. The critical flow comes from the hospital, school and public buildings down the right of the network. Figure 12 (B) arises from a single modification: the demand on one of the buildings is raised from 150 to 2000. This forces a redistribution of flows on the network due to capacity constraints on the right hand path. As a result, critical flow is more evenly balanced across all paths from the source.

The SSFC and FC measures are unable to capture the flow redistribution involved in Figure 12. (For reasons of brevity the diagrams are not shown, but the edge and vertex values differ only slightly across the two scenarios). To a large extent this is a result of SSFC and FC being aggregated sums of maximum flows between pairs of vertices. Since the underlying graph is only changing slightly between scenario (A) and (B) of Figure 12, the measures change only slightly. In contrast, CFC is computed on a per-flow basis, where the flow in question is the result of competition between demand nodes for supply. Thus, the CFC is a snapshot in time of a particular flow, whereas SSFC and FC are a product of the network topology as it pertains to inter-pair communication.

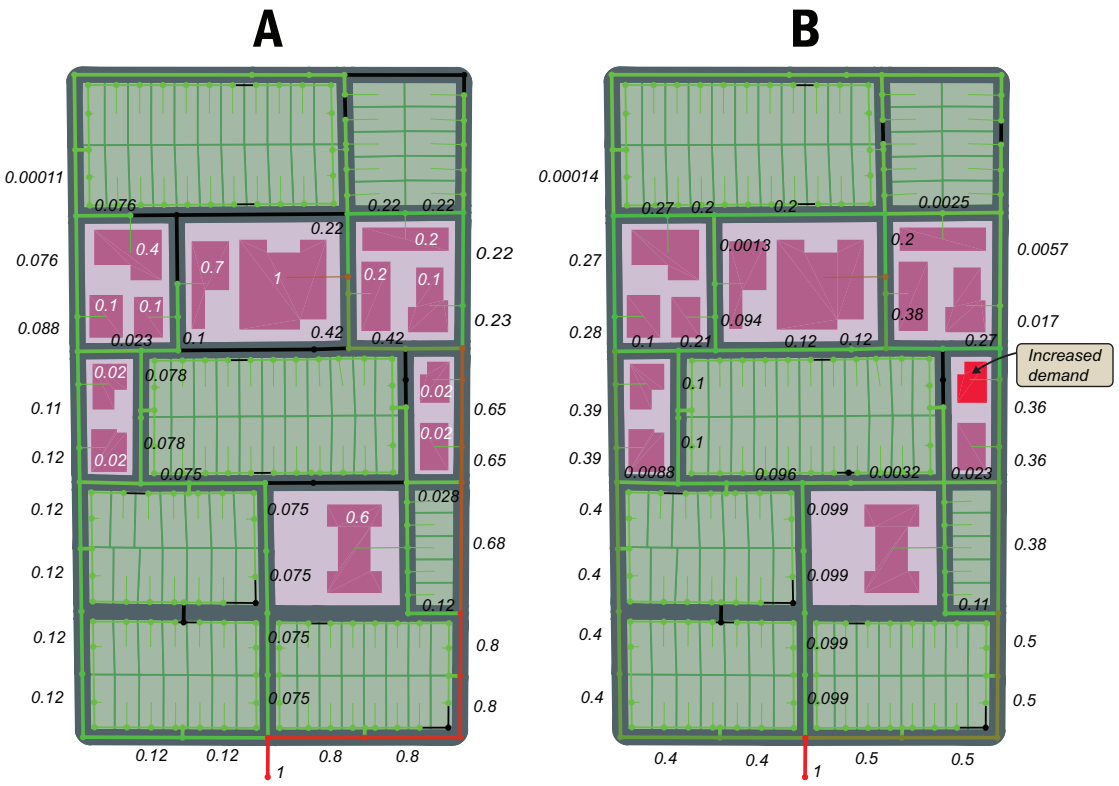


Fig. 12. CCF measures. **A**: identical model and demand/criticality configuration to that in Figure 10. Criticality levels for buildings are show in white, while lot criticality is fixed at 0.05. Critical flow favors the right side of the network. **B**: a variation in which a single building has had its demand raised from 150 to 2000. Flow patterns are changed drastically, with critical flow being redistributed across the network.

6.2 Reliability Analysis

The critical flow measures introduced in this paper may be combined with reliability estimation procedures (e.g., edge reliability [55]). As a simple example, a **reliability function** $r : E \rightarrow [0, 1]$ can be used to assign edges $e \in E$ a reliability rating $r(e)$. Edges with high criticality and low reliability can be identified by using the following **Unreliable Critical Flow** ("UCF") component importance measure:

$$C'^{UCF}(e) = C'^{CF}(e)(1 - r(e))$$

Since both the normalized CCF and $1 - r(e)$ lie in the interval $[0, 1]$, the same is true for $C'^{UCF}(e)$. (As usual, a similar measure can be defined on vertices). Variations on this basic approach can easily be introduced. While criticality values arise from the configuration of the network, reliability values muse be supplied through other means, including surveys, inspections, or inferences from the physical characteristics (e.g., age) of the relevant components.

To demonstrate the use of UCF to analyze edges, a simple reliability function was defined on the infrastructure model shown in Figure 10. A sampling procedure produced reliability estimates for each edge uniformly in the interval $[0.7, 1.0]$. Figure 13 shows a plot of edge unreliability [i.e., $1 - r(e)$] versus edge criticality for the model. The majority of the edges have negligible criticality (e.g., those edges that feed individual lots). The solitary edge with a criticality of 1.0 is the edge from the reservoir that carries flow for the entire model. Of particular interest are those edges that have higher unreliability ratings and moderate criticality.

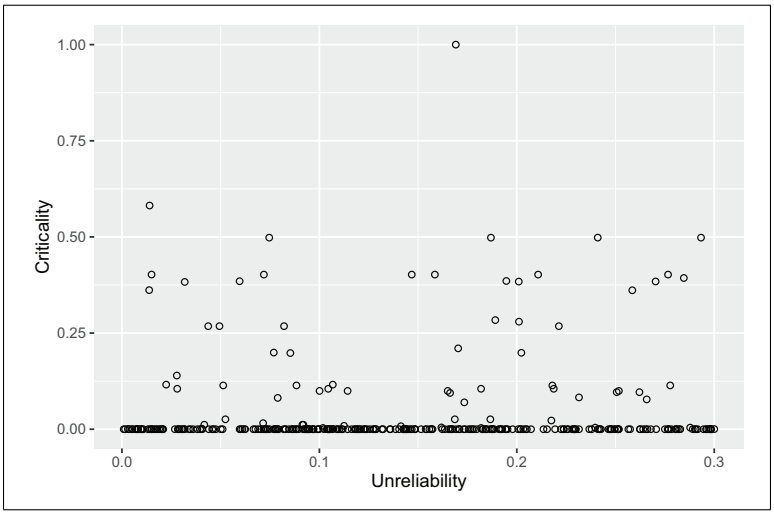


Fig. 13. Edge unreliability plotted against edge criticality.

7 Conclusion

The goal of this paper was to define a novel set of *component importance measures* ("CIMs") based on critical flows and to provide a method for their computation. In the ensuing discussion, the critical flow measures were compared against two versions of the traditional flow centrality measure. A simple example of how the critical flow measures could be combined with reliability analysis was also provided.

Unlike flow centrality, critical flow measures are able to track shifts in flow distribution that arise from changes in both demand and network topology. Flow centrality aggregates the distribution of maximal flows generated between pairs of vertices, resulting in a measure that largely tracks the topology of the network, and one that is not necessarily suitable for networks with defined sets of sources and sinks. In contrast, critical flow measures are specific to a particular configuration of source capacities and sink demands. This means that they give an ephemeral glimpse of a system under load, as opposed to a more general evaluation based on network topology alone.

The approach shown in this paper is only a simple instantiation of the general method, providing simple means of generating flows and computing probabilities. Future work should investigate the use of domain-specific methods, as well as approaches for computing probabilities that work with cyclic graphs. Perhaps the most urgent need is to discuss the use of the critical flow method to analyze a system as it evolves over time. The simple algorithm presented in this paper is somewhat efficient both in terms of time and space complexity, making it feasible for such analysis.

There are additional avenues of future work that were mentioned in passing. First, performance measures can be introduced to give a metric that enables comparison and optimization. The use of utility functions was suggested, but not developed within the paper. Second, vertices and sources could be given capacity constraints, and transmission nodes/edges could be augmented to act simultaneously as supply or demand nodes (e.g., edges could have a small loss to model malfunction or component degradation). Third, network clustering algorithms (see [11]) could be introduced to reduce the number of demand nodes. Lastly, no attempt was made to identify groups of critical components (e.g., critical paths or clusters).

References

[1] Ravindra Ahuja, Thomas Magnanti, and James Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Upper Saddle River, NJ, 1993.

[2] Yasser Almoghathawi and Kash Barker. Component importance measures for interdependent infrastructure network resilience. *Computers and Industrial Engineering*, 133(August 2017):153–164, 2019.

[3] Massoud Amin. Toward Secure and Resilient Interdependent Infrastructures. *Journal of Infrastructure Systems*, 8(3):67–75, 2007.

[4] George E Apostolakis and Douglas M Lemon. A screening methodology for the identification and ranking of infrastructure vulnerabilities due to terrorism. *Risk Analysis*, 25(2):361–376, 2005.

[5] Philippe Blanchard and Dimitri Volchenkov. *Random Walks and Diffusions on Graphs and Databases: An Introduction*. Springer, 2011.

[6] Stephen P Borgatti. Centrality and AIDS. *Connections*, 18(1):112–114, 1995.

[7] Stephen P. Borgatti. Centrality and network flow. *Social Networks*, 27(1):55–71, 2005.

[8] Stephen P. Borgatti and Martin G. Everett. A Graph-theoretic Perspective on Centrality. *Social Networks*, 28(4):466–484, 2006.

[9] Richard K. Brail and Richard E. Klosterman. *Planning Support Systems: Integrating Geographic Information Systems, Models,and Visualization Tools*. Esri Press, 2001.

[10] Ulrik Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, 2008.

[11] Ulrik Brandes and Thomas Erlebach. *Network Analysis: Methodological Foundations*. Springer, 2005.

[12] Francesco Cadini, Enrico Zio, and Cristina Andreea Petrescu. Using centrality measures to rank the importance of the components of a complex network infrastructure. *Lecture Notes in Computer Science*, 5508:155–167, 2009.

[13] Thomas Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 3 edition, 2009.

[14] Chavdar Dangalchev. Residual closeness in networks. *Physica A: Statistical Mechanics and its Applications*, 365(2):556–564, 2006.

[15] D. DeLaurentis and W. Crossley. A taxonomy-based perspective for systems of systems design methods. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 86–91, 2005.

[16] Shlomi Dolev, Yuval Elovici, and Rami Puzis. Routing betweenness centrality. *Journal of the ACM*, 57(4):1–27, 2010.

[17] Sarah Dunn and Sean M. Wilkinson. Identifying Critical Components in Infrastructure Networks Using Network Topology. *Journal of Infrastructure Systems*, 19(2):157–165, 2012.

[18] E. Ferrario, N. Pedroni, and E. Zio. Evaluation of the robustness of critical infrastructures by Hierarchical Graph representation, clustering and Monte Carlo simulation. *Reliability Engineering and System Safety*, 155:78–96, 2016.

[19] L.C Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, 1977.

[20] Linton C. Freeman, Stephen P. Borgatti, and Douglas R. White. Centrality in valued graphs: A measure of betweenness based on network flow. *Social Networks*, 13(2):141–154, 1991.

[21] Stan Geertman, Joseph (Jr) Ferreira, Robert Goodspeed, and John Stillwell. *Planning Support Systems and Smart Cities*. Springer, 2015.

[22] K. I. Goh, B. Kahng, and D. Kim. Universal Behavior of Load Distribution in Scale-Free Networks. *Physical Review Letters*, 87(27):278701–278701–4, 2001.

[23] Oded Green and David A. Bader. Faster betweenness centrality based on data structure experimentation. *Procedia Computer Science*, 18:399–408, 2013.

[24] Gabriel Y Handler and Pitu B. Mirchandani. Location on Networks, 1979.

[25] Clyde Holsapple. Decisions and Knowledge. In Frada Burstein and Clyde Holsapple, editors, *Handbook on Decision Support Systems 1: Basic Themes*, chapter 1. Springer, 2008.

[26] Luis R. Izquierdo and Robert A. Hanneman. *Introduction to the formal analysis of social networks using Mathematica*. University of California, Riverside, California, 2006.

[27] M L Kansal and Arun Kumar. Reliability analysis of water distribution systems under uncertainty. *Reliability Engineering and System Safety*, 50:51–59, 1995.

[28] Konstantin Klemm, M. Ángeles Serrano, Víctor M. Eguíluz, and Maxi San Miguel. A measure of individual role in collective dynamics. *Scientific Reports*, 2:1–8, 2012.

[29] V. Latora and M. Marchiori. A measure of centrality based on network efficiency. *New Journal of Physics*, 9, 2007.

[30] Vito Latora and Massimo Marchiori. Efficient Behavior of Small-World Networks. *Physical Review Letters*, 87(19):3–6, 2001.

[31] Vito Latora, Vincenzo Nicosia, and Giovanni Russo. *Complex Networks Principles , Methods and Applications*. Cambridge University Press, Cambridge (UK), 2017.

[32] Linyuan Lü, Duanbing Chen, Xiao-long Ren, and Qian-ming Zhang. Vital nodes identification in complex networks. *Physics Reports*, 650:1–63, 2016.

[33] David Michaud and George E. Apostolakis. Methodology for Ranking the Elements of Water-Supply Networks. *Journal of Infrastructure Systems*, 12(4):230–242, 2006.

[34] Alan T. Murray, Timothy C. Matisziw, and Tony H. Grebesic. A Methodological Overview of Network Vulnerability Analysis. *Growth and Change*, 39(4):573–592, 2003.

[35] Anna Nagurney and Qiang Qiang. *Fragile Networks: Identifying Vulnerabilities and Synergies in an Uncertain World*. John Wiley & Sons, Hoboken, NJ, 2009.

[36] M. E.J. Newman. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 64(1):7, 2001.

[37] M. E.J. Newman. A Measure of Betweenness Centrality based on Random Walks. *Social Networks*, 27(1):39–54, 2005.

[38] Mark Newman. *Networks: An Introduction*. Oxford University Press, 2 edition, 2018.

[39] Charles D. Nicholson, Kash Barker, and Jose E. Ramirez-Marquez. Flow-based vulnerability measures for network component importance: Experimentation with preparedness planning. *Reliability Engineering and System Safety*, 145:62–73, 2016.

[40] Jae Dong Noh and Heiko Rieger. Random Walks on Complex Networks. *Physical Review Letters*, 92(11):118701, 2004.

[41] Paul Novak, Vincent Guinot, Alan Jeffrey, and Dominic E Reeve. *Hydraulic Modelling – an Introduction: Principles, Methods and Applications*. Spon Press, 2010.

[42] Min Ouyang. Comparisons of purely topological model, betweenness based model and direct current power flow model to analyze power grid vulnerability. *Chaos*, 23(2), 2013.

[43] Min Ouyang, Lijing Zhao, Liu Hong, and Zhezhe Pan. Comparisons of complex network based models and real train flow model to analyze Chinese railway vulnerability. *Reliability Engineering and System Safety*, 123:38–46, 2014.

[44] Ulrich Pferschy and Joachim Schauer. The maximum flow problem with disjunctive constraints. *Journal of Combinatorial Optimization*, 26(1):109–119, 2013.

[45] Mahendra Piraveenan, Mikhail Prokopenko, and Liaquat Hossain. Percolation Centrality: Quantifying Graph-Theoretic Impact of Nodes during Percolation in Networks. *PLoS ONE*, 8(1), 2013.

[46] Steven M. Rinaldi, James P. Peerenboom, and Terrence K. Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE Control Systems Magazine*, 21(6):11–25, 2001.

[47] John P Scott. *Social Network Analysis: A Handbook*. SAGE Publications, London, UK, 2000.

[48] Karen Stephenson and Marvin Zelen. Rethinking centrality: Methods and examples. *Social Networks*, 11(1):1–37, 1989.

[49] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[50] William J. Stewart. *Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Monitoring*. Princeton University Press, 2009.

[51] Phil Stride. Super sewer: an introduction to the Thames Tideway tunnel project in London. *Proceedings of the Institution of Civil Engineers - Civil Engineering*, 169(2):51–51, 2016.

[52] UN-Habitat. Planning Sustainable Cities—Global Report on Human Settlements 2009, 2009.

[53] Dingjie Wang, Haitao Wang, and Xiufen Zou. Identifying key nodes in multilayer networks based on tensor decomposition. *Chaos*, 27(063108), 2017.

[54] Daijun Wei, Xinyang Deng, Xiaoge Zhang, Yong Deng, and Sankaran Mahadevan. Identifying influential nodes in weighted networks based on evidence theory. *Physica A: Statistical Mechanics and its Applications*, 392(10):2564–2575, 2013.

[55] E. Zio. From complexity science to reliability efficiency: a new way of looking at complex network systems and critical infrastructures. *International Journal of Critical Infrastructures*, 3(3/4):488, 2007.

[56] Enrico Zio. *The Monte Carlo Simulation Method for System Reliability and Risk Analysis*. Springer, London, UK, 2013.

[57] Enrico Zio and Wolfgang Kroger. Vulnerability assessment of critical infrastructures. *IEEE Reliability Society*, pages 1–7, 2009.