A Dimensionality-Reduction Strategy to Compute Shortest Paths in Urban Water Networks

Carlo Giudicianni\textsuperscript{a*,} Armando Di Nardo\textsuperscript{a,b}, Gabriele Oliva\textsuperscript{c}, Antonio Scala\textsuperscript{b}, Manuel Herrera\textsuperscript{d}

\textsuperscript{a}Dipartimento di Ingegneria, Universit`a degli Studi della Campania ‘L. Vanvitelli’, via Roma 29, Aversa 81031, Italy
\textsuperscript{b}Institute for Complex Systems – Italian National Research Council, via dei Taurini 19, Roma 00185, Italy
\textsuperscript{c}Dipartimento di Ingegneria, Universit`a Campus Bio-Medico di Roma, via levaro del Portillo 21, Roma 00128, Italy
\textsuperscript{d}Institute for Manufacturing – Dept. of Engineering, University of Cambridge, 17 Charles Babbage Rd., CB3 0FS Cambridge, United Kingdom

Abstract

The efficient computation of shortest paths in complex networks is essential to face new challenges related to critical infrastructures such as a near real-time monitoring and control and the management of big size systems. In particular, using information on the minimum paths in water distribution networks (WDNs) allows to track the diffusion of contaminants and to quantify the resilience and criticality of the system. This is, ultimately, approached by considering dynamically changing path-weights that depend on the flow or on other information available at run-time. These analyses typically include all the WDN assets but reducing the high degree of physical details with a minimum lost of key information for their performance assessment. This paper proposes a strategy to compute minimum paths that is based on a dimensionality-reduction process. Specifically, the network is partitioned into communities and suitably modified to obtain a reduced complexity representation (e.g., in terms of number of nodes and links). The paper shows how this novel, reduced representation is equivalent to the traditional network on computing the shortest paths. The proposed approach is validated considering two utility networks as case studies.

*Corresponding author: Tel.: +39-081-000-0000; Email address: carlo.giudicianni@unicampania.it (Carlo Giudicianni)
The results show that the proposed method provides the exact solution for the shortest path with a computational-time reduction consistently over 50% and up to 90% for some cases. Furthermore, the application of the proposal on WDNs partitioning shows both hydraulic and economic advantages thanks to their monitoring and controlling at near real-time.

*Keywords:* Utility networks, Critical infrastructures, Water distribution systems, Shortest Paths, Dimensionality Reduction, Clustering, Complex networks

1. Introduction

Modern society is strongly dependent on infrastructure systems (i.e. transportation, power grids, telecommunications, water systems), which support city growth and economic prosperity. These infrastructures continually face natural and man-made threats that cause economic and social disruption, leading their operators to continuously work on improving safety and security and on speeding up mitigation actions. Today, the reliability and performance assessment, continuous operation, monitoring, and protection of critical infrastructures are national priorities for countries worldwide [1]. Furthermore, as cities increase in their size, these infrastructures are getting larger and tangled, showing a complex behaviour due to the high degree of interdependency among them. As a consequence, the management of such infrastructures is becoming an arduous task to address, and there is a need to develop new, agile tools and methodologies to analyse and control them. This is especially true for water distribution networks (WDNs), which are among the most important civil networks as they deliver drinking and industrial water to metropolitan areas. WDNs constitute essential and critical infrastructures, as systems whose operability is of crucial importance to ensure social survival and welfare. They must to face two major threats:

- *Contamination:* water infrastructures are strongly vulnerable to malicious and intentional attacks since they are made up of thousands of ex-
posed elements; as a consequence, water can be easily polluted by chemical or biological contaminants, which spread all over the system by flowing, with dramatic effects on the citizen health;

- **Leakages**: water infrastructures are constituted by aged buried pipelines, and as a consequence, pipelines are easily eroded by the moist environment. Furthermore, the daily pressure variability strongly stresses the pipes. These factors lead to the failure and burst of the pipes, causing leakages and wasting huge amount of water.

The downside in the management of such infrastructures is that the underlying details of the physics involved in their functioning complicates the analysis to a relevant extent, making it difficult to achieve useful insights in reasonable time. Complexity Science has proven to be a particularly adequate tool for the timely and agile analysis and management of WDNs (and more in general for critical infrastructures), especially in the case of limited information about the system. In particular, a complex network representation of infrastructures allows to abstract away from the high degree of physical details of the systems and focus only on a few crucial aspects in a manageable way. At this end, knowledge on the minimum paths (eventually considering dynamically changing weights that depend on the flow or on other information available at run-time) allows to track the diffusion of contaminants and quantify the resilience and criticality of the system and its composing elements.

The work of encompasses an extensive survey of various heuristic shortest path (SP) algorithms developed in the last years. It is worth to mention the interesting strategy adopted for practitioners and applied researchers to exploit network’s domain-specific information. This is the case of traffic systems researchers adopting the natural hierarchies of the roads to significantly speed up the SP computational time. Overall, there are two widely investigated strategies for approximate the SP computation in large-scale complex networks. One of them is the landmark–based method. This requires to pre-compute
the shortest paths between special nodes (landmark nodes) and all the other nodes in the network, saving these distances in a database. The shortest-path between two nodes is, then, approximated by combining those distances stored in the database [19, 20, 21]. The other one is a topology-based approach. This strategy lies on the structure of networks through their partition into discrete areas [22, 23, 24, 25]. In this regard, [26] propose an approximated landmark-based method for point-to-point distance estimation in large-scale networks, also adding the partitioning variant. The landmark set is selected for each network area and the shortest paths consequently saved in a database. The Authors also demonstrate that selecting the optimal set of landmark nodes is an NP-hard problem.

This paper proposes a strategy to compute minimum paths, namely Multi-scale Shortest Path (MS-SP), that is based on a dimensionality-reduction process. Specifically, we make a network partition into communities and suitably modify such network to obtain a representation with reduced complexity (e.g., in terms of the number of nodes and edges), namely Multiscale Network (MS), but equivalent in terms of computing the shortest paths. The proposed approach is validated considering two real-world WDNs as case studies.

An antecedent of this paper can be found on the work of [27] which provides a valid approximation to the shortest path problem for social networks. In such work, the authors propose a combined process for community detection and network reduction, by collapsing communities into nodes of a new network. The proposed algorithm has a similar scope, but the main innovation proposed herein is that the network reduction process identifies a subset of key nodes that lie at the boundary of the communities and transforms the community into hyper-links connecting such boundary nodes, rather than collapsing the communities in single nodes. As a result, the network collapses into a reduced-size graph were boundary nodes are interconnected by edges that are weighted in a suitable manner to guarantee that the minimum path between two nodes in the original network can be computed in terms of the minimum path between the boundary nodes that are closest to the source and destination, respectively.
Two utility networks validate the proposal. One is the well-known network of Colorado Springs at Colorado, US. This network is investigated to facilitate repeating further the proposed MS-SP algorithm. The second case-study corresponds to the operational network supplying water to Alcalá de Henares (Madrid, Spain). In this network is more evident the usefulness of the proposal to speed-up the SP computation in large-scale, urban infrastructure networks. In addition to the advantages of the computation, the dimensionality-reduction process leads to a novel representation of WDNs where it is even possible to obtain a visualisation of the shortest paths.

The outline of the paper is as follows: in Section 2 the proposed algorithm is presented from a mathematical point of view. Section 3 shows the two case studies. Section 4 reports the simulation results with special emphasis on their associated computational efforts. Section 5 presents a further application of the algorithm for the management of water networks. The paper closes with a conclusions section which also points out future research directions.

2. Methods

2.1. Graphs

Let’s \( G = \{ V, E, W \} \) denote a weighted graph with a finite number \( n \) of nodes \( v_i \in V \) with \( i \in \{ 1, \ldots, n \} \) and edges \( (v_i, v_j) \in E \subset V \times V \) from node \( v_i \) to node \( v_j \). For each edge \( (v_i, v_j) \in E \) we denote by \( w_{ij} \in W \) the associated weight. A graph is said to be undirected if \( (v_i, v_j) \in E \) whenever \( (v_j, v_i) \in E \), and it is said to be directed otherwise. In the following we will consider undirected graphs. For undirected graphs, we assume the weights satisfy \( w_{ij} = w_{ji} \) for all \( (v_i, v_j) \in E \). Let’s the weighted adjacency matrix of a graph \( G = \{ V, E, W \} \) be the \( n \times n \) matrix \( A \) with the same structure as \( G \), i.e., such that \( A_{ij} = w_{ij} \) if \( (v_i, v_j) \in E \) and \( A_{ij} = 0 \), otherwise. In the case of undirected graphs, matrix \( A \) is symmetric. A path over a graph \( G = \{ V, E, W \} \), starting from a node \( v_i \in V \) and ending in a node \( v_j \in V \), is a subset of links in \( E \) that connect \( v_i \) and \( v_j \); the length of the path is the sum of the weights associated to the links in the path.
A minimum path that connects \( v_i \) and \( v_j \) is the path from \( v_i \) to \( v_j \) of minimum length. An undirected graph is connected if for each pair of nodes \( v_i, v_j \in V \) there is a path over \( G \) that connects them.

2.2. Shortest path algorithm

One of the most well known algorithms to compute the shortest path between two nodes in a weighted graph is Dijkstra’s Algorithm (D-SP) [29], which can be summarized as follows. Given a weighted graph \( G = \{V, E, W\} \) with \( |V| = n \) nodes, a start node \( v_s \) and a goal node \( v_g \), the algorithm keeps track of three variables for each node:

- \( \text{visited}(v_i) \) which is equal to one if the node has already been visited during the algorithm and is zero otherwise;
- \( \text{distance}(v_i) \) which is the current estimate for the distance of node \( v_i \) from the start node \( v_s \);
- \( \text{parent}(v_i) \) which is the identifier of the node immediately before node \( v_i \) in the path connecting \( v_s \) and \( v_i \).

Moreover, the algorithm keeps track of the node currently being examined, which is referred to as \( v_* \).

During the initialisation phase, the algorithm sets \( \text{visited}(v_s) = 1 \) and \( \text{visited}(v_i) = 0 \), for all \( v_i \in V \setminus \{v_s\} \). Moreover, it sets \( \text{distance}(v_s) = 0 \) and \( \text{distance}(v_i) = \infty \), for all \( v_i \in V \setminus \{v_s\} \). Finally, the algorithm selects \( \text{parent}(v_i) = \emptyset \) for all \( v_i \in V \) and sets \( v_s = v_s \). Then, the main cycle of the algorithm is executed; such a main cycle is composed of the following conceptual steps:

Step 1 For all neighbours \( v_i \) of \( v_s \) such that \( \text{visited}(v_i) = 0 \) set the distance of node \( v_i \) from \( v_s \) as the minimum between the previous estimate and the sum of the distance of \( v_s \) from \( v_s \) and the weight of the link \( w_{s,i} \) connecting \( v_s \) and \( v_i \), i.e.,

\[
\text{distance}(v_i) = \min \{\text{distance}(v_i), \text{distance}(v_s) + w_{s,i}\};
\]
moreover, if the distance is updated for node \( v_i \) the algorithm keeps track of the fact that the minimum path from \( v_s \) to \( v_i \) features the edge \((v_s, v_i)\) by setting

\[
\text{parent}(v_i) = v_s.
\]

Step 2 Set \( \text{visited}(v_s) = 1 \)

Step 3 If \( \text{visited}(v_t) = 1 \) then stop, the algorithm is terminated.

Step 4 Otherwise, select the node with minimum current distance among the not visited ones as the new current node, i.e.,

\[
v_* = v_j, \quad \text{where } j = \arg \min_{i \mid \text{visited}(v_i) = 0} \{\text{distance}(v_i)\}
\]

and go back to Step 3.

Notice that a straightforward application of the above algorithm yields a computational complexity \( \mathcal{O}(|V|^2) \) where \( |V| \) is the number of nodes in the graph; moreover, when the graph is particularly sparse, i.e., when \(|E| \ll |V|(|V|−1)/2\), where \(|E|\) is the number of edges, it is possible to reduce complexity by using an implementation that relies on data structures such as the so-called Fibonacci heaps \[30\].

2.3. Clustering approach

A network community detection algorithm \[31\] can be used in case the initial partition of the network is not available. These communities (clusters) are formed by grouping elements with similar characteristics or with a higher connection density than that external to the community. There is a large set of community detection algorithms proposed in literature. In this paper the Louvain method \[32\] is adopted, which uses an iterative process to improve the scalability of the overall community detection. Louvain algorithm is a heuristic method based on modularity optimisation \[33\]. In particular, it is divided in two iteratively repeated phases.
• The algorithm starts assigning a different community to each node of the network, then, for each node $i$ the neighbour $j$ is considered and the gain of modularity that would take place by removing $i$ from its community and by placing it in the community of $j$ is evaluated. After that, the node $i$ is placed in the community for which this gain is maximum. The process is applied for all nodes until no further improvement can be achieved. The first phase stops when a local maxima of the modularity is attained, and no individual move can improve the modularity.

• The second phase of the algorithm builds a new network whose nodes are the communities found during the first phase; once it is completed, the first phase of the algorithm is reapplied to the resulting network. The passes are iterated until the maximum of modularity is attained.

It is known that, the Louvain algorithm appears to run in time $O(|E|)$, where $|E|$ is the number of edges in the graph.

2.4. Multiscale Shortest Path (MS-SP) algorithm

Given a graph $G = \{V, E, W\}$, the proposed approach to calculate the shortest path from a node $v_s$ to a node $v_t$ is based on a dimensionality reduction procedure, where the network is decomposed into clusters and the nodes/edges in each clusters are collapsed in a suitable way that guarantees that the shortest path computed over the resulting graph corresponds to the one on the original graph.

Specifically, we apply the Louvain clustering algorithm to $G$, decomposing the set of nodes $V$ into $q$ disjoint sets $V_1, \ldots, V_q$, each corresponding to a cluster. In the following, we denote by $E_i$ the set of edges in the original edge set $E$ that connect nodes in the same cluster, i.e.,

$$E_i = \{(v_a, v_b) \in E \mid v_a, v_b \in V_i\};$$

moreover, we define

$$\hat{E}_{ij} = \{(v_a, v_b) \in E \mid v_a \in V_i \text{ and } v_b \in V_j\}.$$
and

\[ E_{ij} = \hat{E}_{ij} \cup \hat{E}_{ji}. \]

Finally, we define the set of boundary nodes \( V^b_i \subseteq V_i \) as the set of nodes in \( V_i \) that belong to at least one edge in \( E_{ij} \) for some \( j \in \{1, \ldots, q\} \setminus \{i\} \), i.e.

\[ V^b_i = \{v_a \in V_i \mid \exists (v_a, v_b) \in E, v_b \notin V_i\}. \]

In other words, \( E_{ij} \) is the set of edges that connect nodes in \( V_i \) and nodes in \( V_j \), and it holds \( E_{ij} = E_{ji} \). Specifically, by running the clustering procedure described above, the network is decomposed into \( q \) clusters.

The dimensionality reduction strategy consists in the construction of a graph

\[ \tilde{G} = \{\tilde{V}, \tilde{E}, \tilde{W}\}, \]

where \( \tilde{V} \) includes the set of boundary nodes and the start and goal nodes, i.e.,

\[ \tilde{V} = \{v_s, v_t\} \cup_{i=1}^{q} V^b_i. \]

As for the edge set \( \tilde{E} \), we have that

\[ \tilde{E} = \tilde{E}_{in} \cup \tilde{E}_{out}, \]

where \( \tilde{E}_{out} \) is the union of the edges connecting boundary nodes, i.e.,

\[ \tilde{E}_{out} = \bigcup_{i,j \in \{1,\ldots,q\}} E_{ij} \]

and \( \tilde{E}_{in} \) is the union of sets \( \tilde{E}^b_i \) of edges that directly connect the boundary nodes in the \( i \)-th cluster. Note that, if the start or goal nodes are in the \( i \)-th cluster, then the start or goal nodes are considered as a boundary node.

As for the weights, we select \( \tilde{w}_{ab} = w_{ab} \) whenever \((v_a, v_b) \in \tilde{E}_{out}\), while for each pair of boundary nodes \( v_a, v_b \) that belong to the same cluster \( i \) (including the start or goal node if they belong to cluster \( i \)), we compute the minimum path \( p_{ab}^i \) between \( v_a \) and \( v_b \) over the subgraph of \( G \) induced by considering just the nodes \( V_i \) in the \( i \)-th cluster and we set the weight as the length of the path.
\[ p^i_{ab}, \text{i.e.,} \]
\[ w_{ab} = \sum_{(v_h,v_k) \in p^i_{ab}} w_{hk}. \]

At this point, the algorithm finds the minimum path between nodes \( v_s \) and \( v_t \) by computing the minimum path between \( v_s \) and \( v_t \) over \( \tilde{G} \). Note that, by keeping track of the minimum paths involving boundary nodes in each cluster (treating \( v_s \) and \( v_t \) as boundary nodes), we are able to reconstruct the minimum path over \( G \) in terms of the minimum path over \( \tilde{G} \).

The algorithm is graphically explained by Figure 1 in which there are 3 groups: the upper-left cluster contains three boundary nodes (and the start node), the right cluster has three boundary nodes and the lower cluster has two boundary nodes (plus the target node). As a result of the decomposition, we obtain a network with \( |\tilde{V}| = 10 \) nodes (i.e., the boundary nodes plus the start and goal) and \( |\tilde{E}| = 16 \) edges; in particular, the four edges connecting nodes in different clusters are kept, while for each pair of boundary (or start/goal) nodes in each cluster a new link is added, whose weight corresponds to the length of the minimum path, computed over the subgraph induced by the nodes in the cluster. At this point, the minimum path is computed by computing the minimum path over \( \tilde{G} \).

### 2.5. Correctness of the algorithm

The following theorem establishes that the path found via the proposed algorithm is, indeed, a minimum path.

**Theorem 1.** The minimum path between nodes \( v_s \) and \( v_t \) over \( \tilde{G} \) is equivalent to the one connecting \( v_s \) and \( v_t \) over the original graph \( G \).

**Proof 1.** Let’s \( v_s \) and \( v_t \) belonging to the same cluster in \( G \). Then, by construction, the minimum path found over \( \tilde{G} \) corresponds to the one over \( G \). Let’s assume now that \( v_s \) and \( v_t \) belong to different clusters with node sets \( V_s \) and \( V_t \). By construction, since the clusters are connected only via edges joining boundary nodes belonging to different clusters, the minimum path joining \( v_s \) and \( v_t \) in \( G \)
settings a path from $v_s$ to a node $v_s' \in V_s$, a path from $v_s'$ to a node $v_t' \in V_t$ and path from a node $v_t'$ to $v_t$ (note that if $v_s = v_s'$ or $v_t = v_t'$ the path joining them is the empty set).

At this point, we observe that the path connecting $v_s$ to any $v_{s'} \in V_s$ and the path connecting $v_t$ to any $v_{t'} \in V_t$ are, by construction, minimum paths; similarly, the path connecting any $v_{s'} \in V_s$ and $v_{t'} \in V_t$ with (recall that we assumed $s \neq t$) is a minimum path. Hence, by construction, the minimum path found over $\tilde{G}$ corresponds to a minimum path $p_{st} = p_{ss'} \cup p_{s't'} \cup p_{tt'}$ over the original graph, for some $v_{s'} \in V_s$ and $v_{t'} \in V_t$. The proof is complete.

2.6. Time complexity of the MS-SP algorithm

In the following, it is shown the computational cost of the proposed algorithm. It is important to point out that, the core idea of working on a size-reduced graph does not depending on the chosen clustering algorithm. As a consequence, a faster method can be adopted making the proposed algorithm even more convenient from a computational point of view.

Proposition 1. The computational complexity of the proposed approach, including the clustering procedure and the construction of the reduced graph $\tilde{G}$, is
equal to

$$\max \left\{ \mathcal{O}(|E|), \mathcal{O}(n_b^2), \mathcal{O} \left( \sum_{i=1}^{q} |V_i|^2 |V_i^b|^2 \right) \right\},$$

where $V_i$ is the set of nodes in the $i$-th cluster and $V_i^b$ is the set of boundary nodes in the $i$-th cluster and $n_b = \sum_{i=1}^{q} |V_i^b|$ is the cardinality of the set of all boundary nodes identified by applying Louvain algorithm.

**Proof 2.** The computational complexity of the Louvain method is $\mathcal{O}(|E|)$. Moreover, once the clusters are formed, we need to scan all the edges to identify the set of boundary nodes, a procedure that requires $\mathcal{O}(|E|)$.

At this point, the proposed algorithm computes the shortest path over the subgraph induced by each cluster among each pair of boundary nodes in that cluster; each cluster has $|V_i|$ nodes, hence the computation of the shortest path from one node in the cluster to all other nodes in the cluster requires $\mathcal{O}(|V_i|^2)$, since each cluster has $\mathcal{O}(|V_i^b|)$ distinct pairs of boundary nodes, we have that the computational complexity for each cluster is $\mathcal{O}(|V_i|^2 |V_i^b|^2)$. Since the clusters are $q$ we get $\mathcal{O}(\sum_{i=1}^{q} |V_i|^2 |V_i^b|^2)$.

To conclude, the application of Dijkstra’s algorithm on the reduced-size network has a complexity $\mathcal{O}(n_b^2)$; the proof follows since the two operations are done in series, hence the computational complexity is equal to the largest among the computational complexities of the above procedures.

Note that the computational complexity of the computation of the minimum path, after the graph $\tilde{G}$ has been created is remarkably smaller, being $n_b \ll |V|$ for real world networks. Similarly, the complexity of the clustering procedure, although being theoretically upper bounded by $\mathcal{O}(|V|^2)$, is likely to be remarkably smaller, especially when the graph is sparse and $|E| \ll |V|((|V| - 1)/2, |V|(|V| - 1)/2$ being the number of edges in a complete graph.

As for the calculation of the minimum paths among the boundary nodes in the same cluster, we observe that there may be instances where complexity is above Dijkstra’s algorithm\(^1\); however the likelihood of facing such instances

\(^1\)Consider for instance the case where the graph is full and is arbitrarily divided into 4
is nearly zero in the case of WDNs and, in general, for graphs that have high
sparsity and modularity. In fact, as discussed in the next remark, for those
graphs the complexity of the construction of \( \tilde{G} \) is likely to be well below the one
of Dijsktra’s Algorithm. This fact is experimentally demonstrated in the next
section.

**Remark 1.** Note that the complexity of computing the minimum paths locally
at every cluster has a complexity \( \mathcal{O} \left( \sum_{i=1}^{q} |V_i|^2 |V_i^b|^2 \right) \). However, especially when
the network has a clear modular structure, the number \( q \) of clusters is likely to
be sublinear \( \leq |V| \) (e.g., \( q = |V|^{\gamma} \) with \( \gamma \in (0, 1) \)). Hence, on average, also the
cardinality \( |V_i| \) of the node set of the clusters is likely to be sublinear, i.e.,

\[ |V_i| \approx n/q = |V|^{1-\gamma}. \]

Moreover, the cardinality of \( V_i^b \) is likely to satisfy \( |V_i^b| \ll |V_i| \) and, in several
practical cases, can be assumed to be constant for planar graphs and WDNs (see
\([9]\))\footnote{For instance, in \([9]\) it is shown that for real WDNs the optimal number of clusters is \( q \approx n^{0.3} \).}, i.e., \( |V_i^b| \approx \mathcal{O}(1) \). Hence, in practical cases of interest for this paper, we
have

\[ \mathcal{O} \left( \sum_{i=1}^{q} |V_i|^2 |V_i^b|^2 \right) = \mathcal{O}(|V|^{1+\gamma}) < \mathcal{O}(|V|^2). \]

**Remark 2.** Note that the construction of \( \tilde{G} \) can be slightly modified in order
to be the base for the calculation of all shortest paths. In fact, it is sufficient
to compute all shortest paths among every node in each cluster (i.e., requiring
a computational complexity \( \mathcal{O}(\sum_{i=1}^{q} |V_i|^2 |V_i|) = \mathcal{O}(\sum_{i=1}^{q} |V_i|^4) \) and storing
information on the paths within each cluster. In this way, the graph \( \tilde{G} \) for
calculating a path from any node \( v_s \) to any node \( v_t \) can be constructed by con-
sidering the links connecting boundary nodes and those connecting the start and
goal nodes to the boundary nodes, an operation that requires at most \( \mathcal{O}(|V|) \) in
the worst case).
3. Experimental study

Urban utilities such as water, gas, or electric power networks can be modelled as quasi-planar graphs (e.g., networks forming vertices wherever two edges cross) with spatially organised weighted graphs $G = (V, E, w)$. In the case of water distribution systems the set $V$ of $n$ vertices/nodes encompasses junctions, water sources and demand points. The set $E$ of $m$ edges/links includes pipes, pump stations, and valves. Eventually, $w$ is a function that assigns a weight to each edge quantifying the physical characteristics (diameter, length, roughness) for each pipe. In particular, WDNs are strongly constrained by their geographical embedding in that connections between distant nodes are unlikely to be found, due to physical and economic constraints.

3.1. Benchmarking water distribution system

MS-SP is firstly tested on the real medium-size Colorado Springs (US) water utility - which currently serves a population of about 370,000 inhabitants. Figure 2(a) shows its network layout. This encompasses 1,782 junctions and 4 reservoirs ($n = 1,786$ nodes), 1,985 pipes, 6 pumps and 4 valves ($m = 1995$ links). Figure 2(b) clearly demonstrates the size reduction of the Colorado water network after its transformation into a MS network. From a visual analytics point of view, Figure 2(b) naturally highlights both the more inter-connected network areas and bottleneck links likely related to most vulnerable parts of the system.

3.2. Large scale water distribution system

The second case-study corresponds to the large-scale water utility which serves the Spanish city of Alcalá de Henares. It is located 22 miles northeast of the country’s capital, Madrid, and it counts on a population of 201,000 inhabitants. The water distribution network model (see Figure 3(a)) encompasses 11,473 junctions, 3 reservoirs ($n = 11,476$ nodes), and 12,454 pipes, ($m = 12,454$ links). Figure 3(b) shows the corresponding MS network layout.
4. Results

The first results this section introduces are those corresponding to the topological analysis of the original water network and the MS network, for both case studies. A comparison between the original layout and the collapsed one
Figure 3: Multiscale dimensionality reduction for Alcalá water network

is carried out in terms of:

- *Links Density* $q$ which is the ratio between the total number $m$ of network edges and the maximum number of edges $m^* = n(n - 1)/2$ of a network.
with \( n \) nodes:

\[
q = \frac{2m}{n(n-1)} \quad (1)
\]

- **Average Node Degree** \( \overline{K} \) is the average value of the node degree \( k_i \) (number of edges concurring in the node) over all nodes \( n \):

\[
\overline{K} = \frac{2m}{n} \quad (2)
\]

- **Diameter** \( D \) \[36\] is defined as the maximum shortest distance (the maximum geodesic length) \( d_{ij} \) between any pair of vertices \( i \) to node \( j \) (computed as the number of edges along the shortest path connecting them):

\[
D = \max d_{ij} \quad (3)
\]

- **Average Path Length** \( l \) \[36\] is the average number of steps along the shortest paths for all possible pairs of nodes in the network:

\[
l = \frac{2 \sum d_{ij}}{n(n-1)} \quad (4)
\]

- **Algebraic Connectivity** \( \lambda_2 \) \[37\] corresponds to the second smallest eigenvalue of graph Laplacian matrix \( L \)

- **Spectral Gap** \( \Delta \lambda \) \[38\] is the difference between the first and second eigenvalue of the adjacency matrix \( A \).

Table 1 enumerates the main topological metrics computed for both case studies on the original and the MS network. The total number of links \( m_b \) for the MS network is equal to the sum of the boundary links \( m_{ex} \) and the internal hyper-links \( m_{in} \). The size problem reduction is evident on nodes (from \( n = 1,786 \) to \( n_b = 33 \) for Colorado and from \( n = 11,476 \) to \( n_b = 114 \) for Alcalá) and also on links (from \( m = 1,992 \) to \( m_b = 83 \) for Colorado and from \( m = 12,454 \) to \( m_b = 596 \) for Alcalá). The connectivity \( \overline{K} \) strongly increases for the both
the MS network (from $K = 2.23$ to $K = 5.03$ for Colorado and from $K = 2.17$ to $K = 10.46$ for Alcalá).

The dimensionality reduction working with the MS network makes the network density increases up to 2 orders of magnitude (from $q = 0.001$ to $q = 0.157$ for Colorado and from $q = 0.0002$ to $q = 0.0933$ for Alcalá). This augmented inter-connectivity is also reflected by the two spectral metrics measuring the robustness of network. The algebraic connectivity and the spectral gap also increase when moving from the original to the MS network, as it is shown in Table 1. These changes in the topological metrics reflect the shift in the structure of the MS network which can be regarded now as low interconnected small-world clusters (whose links are the internal hyper-links). In fact, after the size reduction due to the application of the MS-SP algorithm, each cluster of the MS network becomes into a fully connected layout, weakly linked to other clusters by the boundary links. The typical small-world behaviour is also confirmed by the low value of the communication metrics as they are the diameter and the average path length which scale approximately with the log($n$), as happens for the small world networks (Table 1).

Table 1: Topological characteristics of the original water network and the MS network layout for Colorado Springs and Alcalá de Henares

| Metric | Colorado | Colorado-MS | Alcalá | Alcalá-MS |
|--------|----------|-------------|--------|-----------|
| $n$ or $n_b$ | 1786 | 33 | 11,476 | 114 |
| $m$ or $m_b$ | 1995 | 83 | 12,454 | 596 |
| $K$ | 2.23 | 5.03 | 2.17 | 10.46 |
| $q$ | 0.0012 | 0.157 | 0.0002 | 0.0933 |
| $D$ | 69 | 8 | 163 | 9 |
| $l$ | 25.94 | 3.15 | 64.88 | 3.87 |
| $\lambda_2$ | 0.00053 | 0.23512 | 0.00009 | 0.15884 |
| $\Delta \lambda$ | 0.1293 | 0.1735 | 0.0957 | 0.0587 |

The simulation results for Colorado and Alcalá water utilities are reported in Table 2 and Table 3, respectively. A suitable number of clusters $C$ is taken on both cases to optimise the overall connectivity of the partitioned network, according to the relationship $C_{opt} \propto n^{0.28}$ reported in [9], where $C_{opt}$ is the
optimal number of clusters from a topological point of view. As a result, the number of clusters for Colorado is set to $C = 8$, while $C = 13$ for Alcalá’s network. Up to 10 paths are generated by connecting random pairs of source and target to validate the proposed MS-SP algorithm. For each pair, the shortest path is computed by running the code 10 times and averaging the computational time.

Table 2: Simulation results for the Colorado Springs water network

| Pairs | D-SP value [-] | MS-SP value [-] | D-SP time [s] | MS-SP time [s] | Red. time [%] |
|-------|----------------|-----------------|---------------|----------------|--------------|
| 1     | 13             | 13              | 0.0010        | 0.0001         | 90.0         |
| 2     | 21             | 21              | 0.0015        | 0.0005         | 66.6         |
| 3     | 29             | 29              | 0.0022        | 0.0006         | 72.6         |
| 4     | 33             | 33              | 0.0026        | 0.0007         | 72.9         |
| 5     | 38             | 38              | 0.0032        | 0.0004         | 87.4         |
| 6     | 41             | 41              | 0.0033        | 0.0006         | 81.7         |
| 7     | 52             | 52              | 0.0043        | 0.0007         | 83.6         |
| 8     | 56             | 56              | 0.0036        | 0.0005         | 85.8         |
| 9     | 60             | 60              | 0.0041        | 0.0006         | 85.2         |
| 10    | 66             | 66              | 0.0039        | 0.0004         | 89.6         |

MS-SP algorithm provides the exact value of the shortest path between each pairs of randomly generated source and target nodes. This represents a clear advantage with respect to previous methodologies whom provide approximated results. Table 2 and Table 3 clearly state the D-SP and the MS-SP provide the same results (difference is equal to zero).

The computational time for D-SP algorithm grows with the distance between source and target nodes. However, computational times for MS-SP show to be a plateau value of an order of magnitude smaller than that D-SP method. This is clearly shown in figures 4 and 5 (with the results on Colorado and Alcalá utility networks).

Table 2 shows the difference in percentage between the D-SP and the proposed MS-SP computational time for Colorado. This difference on time ranges from 66% to 90%. Table 3 shows the difference in percentage between the D-SP and the proposed MS-SP computational time for Alcalá. This difference on time
Table 3: Simulation results for the Alcalá water network

| Pairs | D-SP value | MS-SP value | D-SP time | MS-SP time | Red. time |
|-------|------------|-------------|-----------|------------|-----------|
| [ ]   | 32         | 32          | 0.0007    | 0.0003     | 50.2      |
| 2     | 40         | 40          | 0.0019    | 0.0004     | 80.5      |
| 3     | 53         | 53          | 0.0032    | 0.0005     | 84.7      |
| 4     | 60         | 60          | 0.0041    | 0.0006     | 85.1      |
| 5     | 72         | 72          | 0.0027    | 0.0004     | 84.4      |
| 6     | 88         | 88          | 0.0098    | 0.0010     | 90.1      |
| 7     | 94         | 94          | 0.0102    | 0.0015     | 85.3      |
| 8     | 102        | 102         | 0.0129    | 0.0011     | 91.6      |
| 9     | 115        | 115         | 0.0094    | 0.0015     | 82.5      |
| 10    | 116        | 116         | 0.0097    | 0.0017     | 91.9      |

Figure 4: Computational time for D-SP and MS-SP algorithms, for Colorado water network
ranges from 50% to 92%. Both differences on computational time stand as a conspicuous time reduction for computing the shortest path.

The MS-SP algorithm is implemented in Python 3.6. All the simulations were run on a Linux Xubuntu 16.04 PC with 2.13 GHz Intel® Core™ i3 CPU m330 64 GB of memory and 4.00 GB of RAM.
5. Discussion

An appealing by-product of the current research is the Small-World properties of the collapsed layout of the original network which preserves essential information about the system, but it is simultaneously characterised by a dramatically size reduction. Indeed, one of the most useful management strategies for WDNs is the sectorisation, which consists in dividing the system in smaller and monitored districts, connected each other by a few number of pipes. Sectorisation helps the pressure management, the leakage detection and reduction, the spreading reduction of contaminants. For this reason, clustering algorithms have been largely applied in hydraulic engineering to define the optimal configuration of districts that balance the aforementioned positive aspects to the resilience reduction that could happen as consequence of the closure of some boundary pipes (boundary links $m_{ex}$ in the model). The spatial-temporal variability of the functioning conditions, such as the change in water request all over the system, could invalidate the designed sectorisation, by compromising the performance of the WDN. An adaptive/dynamic approach could be a valid
solution, providing the aggregation/desegregation of districts (clusters) according to specific conditions. In this regard, the following constraints should be respected:

- the new aggregate districts include the former ones without splitting them;
- the previous clustering layout and the devices already installed have to be exploited;
- the set of new boundary links is included in the set of boundary links of the original partitioning;

As a consequence of working with the MS network layout the management of the system is simplified. In addition, the new investment costs is minimised, even nullified, and the computational burden of the whole procedure is reduced. This structural knowledge comes in the form of pairwise must-link (boundary links) and cannot link (internal links) constraints to be respected at each step of clustering by means a semi-supervised approach (which is particularly suitable for working with real-world systems if background knowledge about the structure are available).

The MS network implicitly takes into account the aforementioned structural knowledge, thanks to the shift in the structure to a low interconnected small-world clusters. In this way, it is ensured that any clustering algorithm provides a solution in which the novel set of boundary links constitutes a sub-set of the boundary links of the original cluster layout. On top of this, a network community detection algorithm splits a network in such way that each cluster is formed by elements having a high density connection between each other and a lower probability to be connected to items belonging to other clusters.

The steps of the procedure are shown in Figure 6. First, the actual clustering layout of the WDN is detected (see Figure 6(a)) and then the corresponding MS network is built (see Figure 6(b)) in which it is evident the size reduction of the WDN. This figure also shows the key elements, such as the boundary nodes of each cluster (highlighted by their corresponding district colour at Figure 222.
Figure 6: Graphical explanation for the dynamic aggregation / disaggregation process for the partitioning of a WDN through the application of the MS layout.

Figure 6(a), the boundary links (bold black line), and the internal links (thin dashed grey line). The set of hyper-links (boundary links and internal links) is
crucial for automatically implementing the semi-supervised clustering during the aggregation phase. Boundary links represent the connectivity between different clusters, while the internal links constitute the internal connectivity of each cluster (according to the shortest path linking each pairs of boundary nodes belonging to the same cluster). After that, the previous districts are aggregated in the MS network (see Figure 6(c)), by applying a clustering algorithm to provide the new clustering layout. Finally, in Figure 6(d) the new district configuration is shown, with the new bigger clusters that perfectly include the former ones (avoiding to split them), and the boundary set which constitutes a sub-set of the previous one. This dynamic aggregation / disaggregation process ensures that the districts in each phase are kept in control, exploiting the devices already installed in the WDN.

6. Conclusions

This paper proposes a novel method to efficiently solve the shortest path problem for large-scale water networks (and other utility networks), showing the potentialities on its application to their management and protection. The algorithm is based on a community structure principle, which aids to collapse the original network into a limited set of key elements. This is made through the novel concept of a multiscale (MS) network that reduces the original system into a series of interconnected, landmark, nodes. These landmark nodes are connected by a family of links coming both from the original network and the scale process where links and nodes are aggregated into hyper-links. The hyper-links are weighted by the shortest-path distances between their corresponding extreme nodes. The MS network structure easier the shortest path computation as the network’s dimensionality is significantly reduced. Computing the shortest path is broken down to be done in several but smaller areas instead of directly using the whole original network. The simulation results over two urban water utilities confirm the efficiency of the proposed multiscale shortest path algorithm. This provides an exact solution for the problem in a significantly lower
computational time than using Dijkstra’s algorithm. The approach conveniently scales to big-size networks. The reduced size of the multiscale network coming from the application of the proposed algorithm results in a dramatic advantage for speeding up and simplifying the management of water systems by means the definition of optimal metered districts.

Further works will benefit of this process since, in general, links connecting key nodes are often closed (i.e. most of the boundary pipes connecting nodes belonging to different network areas). In addition to utility networks applications, the multiscale shortest path usefulness extends to critical infrastructures such as transportation, communication, logistic and supply networks. The proposal will also improve the communication speed in general big-size networked systems and will back-up further near real-time operations.

References

[1] C. Alcaraz, S. Zeadally, Critical infrastructure protection: Requirements and challenges for the 21st century, International Journal of Critical Infrastructure Protection 8 (2015) 53–66.

[2] S. Chianese, A. Di Nardo, M. Di Natale, C. Giudicianni, D. Musmarra, G. F. Santonastaso, Dma optimal layout for protection of water distribution networks from malicious attack, in: International Conference on Critical Information Infrastructures Security, Springer, 2017, pp. 84–96.

[3] D. J. Kroll, Securing our water supply: protecting a vulnerable resource, PennWell Books, 2006.

[4] F. Wang, X.-z. Zheng, N. Li, X. Shen, Systemic vulnerability assessment of urban water distribution networks considering failure scenario uncertainty, International Journal of Critical Infrastructure Protection 26 (2019) 100299.

[5] U. EPA, Control and mitigation of drinking water losses in distribution systems (2010).
[6] A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, C. Faloutsos, Efficient sensor placement optimization for securing large water distribution networks, Journal of Water Resources Planning and Management 134 (6) (2008) 516–526.

[7] G. Stergiopoulos, P. Kotzanikolaou, M. Theocharidou, D. Gritzalis, Risk mitigation strategies for critical infrastructures based on graph centrality analysis, International Journal of Critical Infrastructure Protection 10 (2015) 34–44.

[8] J. M. Torres, L. Duenas-Osorio, Q. Li, A. Yazdani, Exploring topological effects on water distribution system performance using graph theory and statistical models, Journal of Water Resources Planning and Management 143 (1) (2016) 04016068.

[9] C. Giudicianni, A. Di Nardo, M. Di Natale, R. Greco, G. F. Santonastaso, A. Scala, Topological taxonomy of water distribution networks, Water 10 (4) (2018) 444.

[10] A. Di Nardo, C. Giudicianni, R. Greco, M. Herrera, G. F. Santonastaso, Applications of graph spectral techniques to water distribution network management, Water 10 (1) (2018) 45.

[11] Z. Wang, A. Scaglione, R. J. Thomas, Generating statistically correct random topologies for testing smart grid communication and control networks, IEEE transactions on Smart Grid 1 (1) (2010) 28–39.

[12] J. Beyza, E. Garcia-Paricio, J. M. Yusta, Applying complex network theory to the vulnerability assessment of interdependent energy infrastructures, Energies 12 (3) (2019) 421.

[13] A. Házsnagy, I. Fi, A. London, T. Németh, Complex network analysis of public transportation networks: A comprehensive study, in: 2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), IEEE, 2015, pp. 371–378.
[14] A. Di Nardo, M. Di Natale, C. Giudicianni, G. Santonastaso, D. Savic, Simplified approach to water distribution system management via identification of a primary network, Journal of Water Resources Planning and Management 144 (2) (2017) 04017089.

[15] S. Tinelli, E. Creaco, C. Ciaponi, Sampling significant contamination events for optimal sensor placement in water distribution systems, Journal of Water Resources Planning and Management 143 (9) (2017) 04017058.

[16] L. Fu, D. Sun, L. Rilett, Heuristic shortest path algorithms for transportation applications: State of the art, Computers & Operations Research 33 (2006) 3324–3343.

[17] G. Jagadeesh, T. Srikanthan, K. Quek, Heuristic techniques for accelerating hierarchical routing on road networks, IEEE Transactions on Intelligent Transportation Systems 3 (4) (2002) 301–309.

[18] S. Jung, S. Pramanik, An efficient path computation model for hierarchically structured topographical road maps, IEEE Transactions on Knowledge and Data Engineering 14 (5) (2002) 1029–1046.

[19] L. Tang, M. Crovella, Virtual landmarks for the internet, In IMC 2003 (2003).

[20] J. Kleinberg, A. Slivkins, T. Wexler, Triangulation and embedding using small sets of beacons, in: Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on, IEEE, 2004, pp. 444–453.

[21] M. V. Vieira, B. M. Fonseca, R. Damazio, P. B. Golgher, D. d. C. Reis, B. Ribeiro-Neto, Efficient search ranking in social networks, in: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, ACM, 2007, pp. 563–572.

[22] H. Djidjev, Efficient algorithms for shortest path queries in planar digraphs., In Proceedings of the 22nd Workshop on Graph Theoretic Concepts in
Computer Science, Lecture Notes in Computer Science Springer Verlag (1996) 151–165.

[23] M. Henzinger, P. Klein, S. Rao, M. Rauch, S. Subramanian, Faster shortest-path algorithms for planar graph., Special Issue of Journal of Computer and System Science on selected papers of STOC 1994 55 (1) (1997) 3–23.

[24] N. Jing, Y. Huang, E. Rundensteiner, Hierarchical encoded path views for path query processing: an optimal model and its performance evaluation., IEEE Transactions on Knowledge and Data Engineering 10 (3) (1998) 409–431.

[25] R. Gutman, Reach-based routing: A new approach to shortest path algorithms optimized for road networks., In Proc. Algorithm engineering and experimentation: sixth annual international workshop (2004).

[26] M. Potamias, F. Bonchi, C. Castillo, A. Gionis, Fast shortest path distance estimation in large networks, CIKM 09 Proceedings of the 18th ACM conference on Information and knowledge management, Hong Kong, China, November 02-06, 2009 (2009) 867–876.

[27] M. Gong, G. Li, Z. Wang, L. Ma, D. Tian, An efficient shortest path approach for social networks based on community structure, CAAI Transactions on Intelligence Technology 1 (1) (2016) 114–123.

[28] I. Lippai, Colorado springs utilities case study: Water system calibration/optimization, Pipelines 2005: Optimizing Pipeline Design, Operations, and Maintenance in Todays Economy, American Society of Civil Engineers: Reston, VA, USA (2005).

[29] E. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1959) 269–271.

[30] M. L. Fredman, R. E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, Journal of the ACM (JACM) 34 (3) (1987) 596–615.
[31] S. Fortunato, D. Hric, Community detection in networks: A user guide, Physics Reports 659 (2016) 1–44.

[32] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, Journal of Statistical Mechanics: Theory and Experiment 2008 (10) (2008) P10008.

[33] M. E. Newman, Fast algorithm for detecting community structure in networks, Physical review E 69 (6) (2004) 066133.

[34] V. A. Traag, Faster unfolding of communities: Speeding up the louvain algorithm, Physical Review E 92 (3) (2015) 032801.

[35] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, D.-U. Hwang, Complex networks: Structure and dynamics, Physics reports 424 (4-5) (2006) 175–308.

[36] D. J. Watts, S. H. Strogatz, Collective dynamics of small-world networks, Nature 393 (1998) 440–442.

[37] M. Fiedler, Algebraic connectivity of graphs., Czech. Math. J. 23 (2) (1973) 298–305.

[38] E. Estrada, Network robustness to targeted attacks. the interplay of expansibility and degree distribution., Eur. Phys. J. B - Condens. Matter Complex Syst. 52 (4) (2006) 563–574.