Information Sciences 556 (2021) 341–360

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Randomized shortest paths with net flows and capacity constraints

Sylvain Courtain [a,*], Pierre Leleux [a], Ilkka Kivimäki [e], Guillaume Guex [b], Marco Saerens [c]

[a] LOURIM, Université catholique de Louvain, Belgium
[b] SLI, Université de Lausanne, Switzerland
[c] ICTEAM, Université catholique de Louvain, Belgium
[e] Department of Computer Science, Aalto University, Finland

ARTICLE INFO

Article history:
Received 4 October 2019
Received in revised form 1 October 2020
Accepted 4 October 2020
Available online 26 October 2020

Keywords:
Link analysis
Graph mining
Network data analysis
Complex networks
Network science
Distances between nodes

ABSTRACT

This work extends the randomized shortest paths (RSP) model by investigating the net flow RSP and adding capacity constraints on edge flows. The standard RSP is a model of movement, or spread, through a network interpolating between a random-walk and a shortest-path behavior (Kivimäki et al., 2014; Saerens et al., 2009; Yen et al., 2008). The framework assumes a unit flow injected into a source node and collected from a target node with flows minimizing the expected transportation cost, together with a relative entropy regularization term. In this context, the present work first develops the net flow RSP model considering that edge flows in opposite directions neutralize each other (as in electric networks), and proposes an algorithm for computing the expected routing costs between all pairs of nodes. This quantity is called the net flow RSP dissimilarity measure between nodes. Experimental comparisons on node clustering tasks indicate that the net flow RSP dissimilarity is competitive with other state-of-the-art dissimilarities. In the second part of the paper, it is shown how to introduce capacity constraints on edge flows, and a procedure is developed to solve this constrained problem by exploiting Lagrangian duality. These two extensions should improve significantly the scope of applications of the RSP framework.

1. Introduction

Link analysis and network science are dedicated to the analysis of network data and are currently studied in a large number of different fields (see, e.g., [37]). One recurring problem in this context is the definition of meaningful measures for capturing interesting properties of the network and its nodes, like distance measures between nodes or centrality measures. These quantities usually take the structure of the whole network into account.

However, many such measures are based on strong assumptions about the movement, or communication, occurring in the network whose two extreme cases are the optimal behavior and the random behavior. Indeed, the two most popular distance measures in this context are the least-cost distance (also called shortest-path distance) and the resistance distance [31] (equal to the effective resistance and proportional to the commute-time distance when considering a random walk on the

* Corresponding author.
E-mail addresses: sylvain.courtain@uclouvain.be (S. Courtain), p.leleux@uclouvain.be (P. Leleux), ilkka.s.kivimaki@gmail.com (I. Kivimäki), guillaume.guex@unil.ch (G. Guex), marco.saerens@uclouvain.be (M. Saerens).

https://doi.org/10.1016/j.ins.2020.10.005
0020-0255/

graph; see [16] and references therein). The same holds with the betweenness centrality: popular measures are the shortest-path betweenness introduced by Freeman [18] and the random-walk betweenness (also called the current-flow betweenness) independently introduced by Newman [36] and by Brandes and Fleischer [6]. Still another example is provided by the measures of compactness of a network, such as the Wiener index (based on shortest paths) and the Kirchhoff index (based on random walks or electric networks).

In reality, however, behavior is seldom based on complete randomness or optimality. Therefore, a large effort has been invested in defining models interpolating between a shortest-path and a random-walk behavior [17], especially in the context of distance measures between nodes where both proximity and high connectivity are taken into account (the concept of relative accessibility [9]). These models are based on extensions of electric networks [3,24,38], on combinatorial analysis arguments [7,8,9], on mixed L1-L2 regularization [34], on entropy-regularized network flow models [4,21,22], or on entropy-regularized least-cost paths ([30,42,49], directly inspired by a transportation model, denoted as the Markovian traffic assignment in this field [2], and also related to [46]). The latter (i.e., the entropy-regularized least-cost paths) constitutes the **randomized shortest paths** (RSP) framework which is the main subject of this paper. For a more thorough discussion of families of distances between nodes, see, for example, [17,30].

This effort is pursued here by proposing two extensions to this RSP model, considering:

- A new dissimilarity measure extending the RSP dissimilarity [30,42,49], namely the **net-flow RSP dissimilarity**. Like the standard RSP dissimilarity, this new dissimilarity is the expected cost needed for reaching the target node from the source node, but now considering that edge flows in two opposite directions cancel each other out, as for the electric current [13]. Therefore, this model of movement based on net flows more resembles electric flows when the temperature of the system is high. An algorithm is proposed for computing the net-flow RSP dissimilarity matrix between all pairs of nodes.
- The introduction of **capacity constraints** in the model. Capacity constraints on edge flows are very common in practice [1], and the applicability of the RSP model would certainly be increased if such constraints could be integrated. Therefore, the main contributions related to capacity constraints are (1) to show how the model can accommodate such constraints, for both raw edge flows and net flows, and (2) to provide an algorithm for solving the constrained RSP model in the case of a single pair of source/target nodes.

Capacity constraints appear frequently in network flow problems, and there is a vast literature on the subject, especially in the operations research field (see, e.g., [1,32]). They arise, for instance, in the standard *minimum cost flow* problem that aims to find the source-target flow with minimal cost and subject to some capacity constraints. As discussed in [1], this task often appears in real-world problems. In short, capacity constraints are mainly present in order to, for example, avoid congestion, spreading the traffic, or simply because the flow is restricted. In this context, various algorithms for solving the standard minimum cost flow problem have been proposed: minimum mean cycle-cancelling, successive shortest paths, network simplex, and primal-dual, to name a few (see the above citations and references therein). The difference with our work is that, in the RSP, (1) the model is expressed in terms of full paths and (2) a Kullback-Leibler regularization term is introduced, inducing randomized routing policies that can be computed by standard matrix operations.

In the transportation networks field that inspired the RSP [2,12], capacity constraints were also considered in various transportation models, such as the standard traffic assignment problem [23], the deterministic static equilibrium model [15], the stochastic user equilibrium model [5], or the random utility theory [41] (see also the references in these papers).

Among them, the closest to our work is the stochastic user equilibrium model. However, to the best of our knowledge, we are not aware of such models integrating capacity constraints based on an RSP-type full paths formalism. Therefore, because of their practical usefulness, we found that it would be a valuable contribution to introduce such constraints within the framework, which is developed in Section 4. Indeed, we will exploit the fact that, as the RSP model can be derived using maximum entropy arguments [10,27], linear inequality constraints can be handled by Lagrangian duality (see, e.g., [28]).

The content of this paper is structured as follows. Section 2 summarizes the standard RSP framework. Section 3 introduces the net flow RSP dissimilarity. Then, Section 4 develops new algorithms for constraining the flow capacity on edges, while Section 5 deals with net flow capacity constraints. Illustrative examples and experiments on node clustering tasks are described in Section 6. Finally, Section 7 presents the conclusion.

2. The standard randomized shortest paths framework

As already discussed, the main contributions of the paper are based on the RSP framework, interpolating between a least-cost and a random-walk behavior, and allowing dissimilarity measures to be defined between nodes [30,42,49]. The formalism, inspired by models developed in transportation science [2,12], is based on full paths instead of standard "local" edge flows [1,4,22] and is briefly described in this section for completeness. We start by providing a short account of the RSP before introducing, in the following sections, the net flow RSP dissimilarity as well as the algorithm for solving the flow-capacity constrained RSP problem on a directed graph.

2.1. Background and notation

Let us begin by introducing some necessary notation [17,30]. First, notice that column vectors are written in bold lowercase and matrices are in bold uppercase. Moreover, in this work, we consider a weighted directed,[1] strongly connected graph or network, $G = (\mathcal{V}, \mathcal{E})$, with a set $\mathcal{V}$ of n nodes and a set $\mathcal{E}$ of m directed edges. An edge connecting node i to node j is denoted by (i,j) or $i \to j$. Furthermore, we are given an **adjacency matrix** $\mathbf{A} = (a_{ij}) \geqslant 0$ quantifying the directed, local, positive affinity between pairs of adjacent nodes i, j. As usual, a zero value, $a_{ij} = 0$, indicates that there is no edge between nodes i and j. In addition, we assume that there are no self-loops in the network, that is, $a_{ii} = 0$ for all i. From this adjacency matrix, the standard **reference random walk** on the graph is defined in the usual way: the **transition probabilities** associated with each node are set proportionally to the affinities and then normalized in order to sum to one,

$$p_{ij}^{\mathrm{ref}} = \frac{a_{ij}}{\sum_{j'=1}^{n} a_{ij'}} = \frac{a_{ij}}{d_i} \tag{1}$$

where d_i is the (out) degree of node i. The matrix $\mathbf{P}_{\mathrm{ref}} = \left(p_{ij}^{\mathrm{ref}}\right)$ is row-stochastic and is called the transition matrix of the natural, reference, random walk on the graph.

In addition, a transition **cost**, $c_{ij} \geqslant 0$, is associated with each edge (i,j) of G. If there is no edge linking i to j, the cost is assumed to take an infinite value, $c_{ij} = \infty$. We assume for consistency that $c_{ij} = \infty$ if $a_{ij} = 0$, and the cost matrix is defined accordingly, $\mathbf{C} = (c_{ij})$. Costs are usually set independently of the adjacency matrix; they quantify the immediate penalty associated with a transition, depending on the application at hand. This is, however, not always the case. For example, in electric networks, the costs are resistances and the affinities are conductances: in this context, they are linked by $a_{ij} = 1/c_{ij}$.

A **path** or **walk** $\wp$ is a finite sequence of hops to adjacent nodes on G (including cycles), initiated from a source node s and stopping at some ending target node t with $s \neq t$. A **hitting path** is a path where the last node t does not appear as an intermediate node. In other words, a hitting path to node t stops when it reaches t for the first time. In practice, we consider hitting paths to the fixed target node t by setting this target node as absorbing (or "killing"). Computationally, this is achieved by putting the corresponding row t of the transition matrix to zero. The node at position τ along path $\wp$ is denoted by $\wp(\tau)$. The **total cost** of a path, $\tilde{c}(\wp)$, is simply the sum of the edge costs c_{ij} along $\wp$, while its **length** $\ell(\wp)$ is the number of steps, or hops, needed for following that entire path.

2.2. The standard randomized shortest paths formalism

The main idea behind the RSP model is closely related to maximum entropy problems [10,27]. Let us consider the set of all hitting paths, or walks, $\wp \in \mathcal{P}_{st}$ from a node $s \in \mathcal{V}$ (source) to an absorbing, killing node $t \in \mathcal{V}$ (target) on G. Then, we assign a probability distribution $\mathrm{P}(\cdot)$ on the discrete set of paths $\mathcal{P}_{st}$ [4,30] by minimizing the **free energy** of statistical physics [27], [2]

$$\left| \begin{array}{ll} \underset{\{\mathrm{P}(\wp)\}_{\wp \in \mathcal{P}_{st}}}{\text{minimize}} & \phi(\mathrm{P}) = \sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp)\tilde{c}(\wp) + T \sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp) \log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) \\ \text{subject to} & \sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp) = 1 \end{array} \right. \tag{2}$$

where $\tilde{c}(\wp) = \sum_{\tau=1}^{\ell(\wp)} c_{\wp(\tau-1)\wp(\tau)}$ is the total cumulated cost along path $\wp$ when visiting the nodes $(\wp(\tau))_{\tau=0}^{\ell(\wp)}$ in the sequential order. Furthermore, $\tilde{\pi}(\wp) = \prod_{\tau=1}^{\ell(\wp)} p_{\wp(\tau-1)\wp(\tau)}^{\mathrm{ref}}$ is the product of the reference transition probabilities along path $\wp$, i.e., the random walk probability of path $\wp$.

The objective function (Eq. (2)) is a mixture of two dissimilarity terms, with the temperature $T > 0$ balancing the trade-off between them. The first term is the expected cost for reaching the target node from the source node (favoring shorter paths – *exploitation*). The second term corresponds to the relative entropy [10], or Kullback-Leibler divergence, between the path probability distribution and the reference (random-walk) paths probability distribution (introducing randomness and diversity – *exploration*). For a low temperature T, low-cost paths are favored, whereas when T is large, paths are chosen according to their likelihood in the reference random walk on G.

The problem (2) corresponds to a standard minimum cost flow problem, as discussed in the introduction,[3] with a Kullback-Leibler regularization term expressed in terms of full paths in the RSP formalism. There are, however, some subtle differences, such as the fact that in the standard minimum cost flow problem the flows are unidirectional, whereas the RSP defines a Markov chain for which flows are generally bi-directional.

This argument, akin to maximum entropy [10,27], leads to a **Gibbs-Boltzmann distribution** on the set of paths (see, e.g., [30,42]),

[1] When the graph is undirected, we consider that it is made of two reciprocal, directed, edges with the same affinities and costs.

[2] More precisely, it corresponds to a generalized free energy based on the relative entropy instead of the Shannon entropy.

[3] Here, without capacity constraints, which will be introduced in Section 4.

$$\mathrm{P}^*(\wp) = \frac{\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]}{\sum_{\wp'\in\mathcal{P}_{st}}\tilde{\pi}(\wp')\exp[-\theta\tilde{c}(\wp')]} = \frac{\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]}{\mathcal{Z}} \tag{3}$$

where $\theta = 1/T$ is the inverse temperature and the denominator

$$\mathcal{Z} = \sum_{\wp\in\mathcal{P}_{st}}\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)] \tag{4}$$

is the **partition function** of the system. Eq. (3) provides the randomized routing policy in terms of full paths from s to t.

Interestingly,[4] if we replace the probability distribution $\mathrm{P}(\cdot)$ by the optimal distribution $\mathrm{P}^*(\cdot)$ provided by Eq. (3) in the objective function (Eq. (2)), we obtain

$$\begin{aligned}\phi^* = \phi(\mathrm{P}^*) &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}^*(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}^*(\wp)\log\left(\tfrac{\mathrm{P}^*(\wp)}{\tilde{\pi}(\wp)}\right)\\ &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}^*(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}^*(\wp)\left(-\tfrac{1}{T}\tilde{c}(\wp) - \log\mathcal{Z}\right)\\ &= -T\log\mathcal{Z}\end{aligned} \tag{5}$$

which has been termed the directed **free energy distance** [30], and plays the role of a potential.

2.3. Computing quantities of interest

The quantities of interest can be computed by taking the partial derivative of the optimal free energy provided by Eq. (5) [30,42,49]. Here, we introduce only the quantities that are necessary to deriving the algorithms developed later.

Fundamental matrix and partition function. It turns out that the partition function can be computed in closed form from an auxiliary matrix,[5] $\mathbf{W} = \left(p_{ij}^{\mathrm{ref}}\exp[-\theta c_{ij}]\right)$. First, the **fundamental matrix** of the RSP system is defined as

$$\mathbf{Z} = \mathbf{I} + \mathbf{W} + \mathbf{W}^2 + \cdots = (\mathbf{I}-\mathbf{W})^{-1} \quad \text{with} \quad \mathbf{W} = \mathbf{P}_{\mathrm{ref}} \circ \exp[-\theta\mathbf{C}] \tag{6}$$

where $\mathbf{C}$ is the cost matrix and $\circ$ is the elementwise (Hadamard) product. The equation sums up contributions of different paths lengths, starting from zero-length paths (identity matrix $\mathbf{I}$). The partition function is then provided by $\mathcal{Z} = [\mathbf{Z}]_{st} = z_{st}$ [30,42,49].

Computation of flows and numbers of visits. The directed **flow** in edge $(i,j)\in\mathcal{E}$ (the expected number of passages through (i,j) when going from s to t) can be obtained from Eq. (5) and Eq. (6) for a given inverse temperature $\theta = 1/T$ (see [29,42] for details) by

$$\bar{n}_{ij} \triangleq \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\,\eta((i,j)\in\wp) = -\frac{1}{\theta}\frac{\partial\log\mathcal{Z}}{\partial c_{ij}} = \frac{z_{si}w_{ij}z_{jt}}{z_{st}} \tag{7}$$

where $\eta((i,j)\in\wp)$ counts the number of times edge (i,j) appears on path $\wp$. Now, as only the row s and the column t of fundamental matrix $\mathbf{Z}$ are needed, two systems of linear equations can be solved instead of the matrix inversion in Eq. (6). Note also that it can be readily shown from (7) that flows are conserved on intermediate nodes $\mathcal{V}\setminus\{s,t\}$; indeed, $\sum_{i=1}^{n}(\bar{n}_{ij}-\bar{n}_{ji}) = 0$ (input flow to j is equal to output flow from j). Let us further define the matrix containing the expected number of passages through edges (i,j) by $\mathbf{N} = (\bar{n}_{ij})$. From Eq. (7), the expected **number of visits** to a node j can also be defined and easily computed as

$$\bar{n}_j = \sum_{i\in\mathcal{P}red(j)}\bar{n}_{ij} + \delta_{sj} = \sum_{i=1}^{n}\bar{n}_{ij} + \delta_{sj} = \frac{z_{sj}z_{jt}}{z_{st}} \tag{8}$$

because a unit source flow of $+1$ is injected in node s. Note that $\mathcal{P}red(j)$ is the set of predecessor nodes of node j and that we used $\sum_{i=1}^{n}z_{si}w_{ij} = z_{sj} - \delta_{sj}$, coming from $\mathbf{Z}(\mathbf{I}-\mathbf{W}) = \mathbf{I}$, with δ_{sj} being a Kronecker delta.

Optimal transition probabilities. Finally, the optimal transition probabilities of following any edge $(i,j)\in\mathcal{E}$ (the policy) induced by the set of paths $\mathcal{P}_{st}$ and their probability mass (Eq. (3)) are [42]

$$p_{ij}^* = \frac{\bar{n}_{ij}}{\bar{n}_i} = \frac{z_{jt}}{z_{it}}w_{ij} \quad \text{for all } i\neq t \tag{9}$$

and $p_{tj}^* = 0$ for the target node and all j. These transition probabilities define a **biased random walk** (an absorbing Markov chain) on G – the random walker is "attracted" by the target node t. The lower the temperature, the larger the attraction.

[4] This is also a standard result from statistical physics.

[5] Recall that the target node t is made to be killing and absorbing by setting the corresponding row of the reference transition matrix to zero, which implies that row t of $\mathbf{W}$ is also zero. Note also that the framework can easily be extended to any sub-stochastic matrix $\mathbf{W}$.

Interestingly, these transition probabilities do not depend on the source node, and they correspond to the optimal randomized strategy, or policy, thereby minimizing free energy (Eq. (2)) for reaching the target node. Notice further that $\bar{n}_{ij} = \bar{n}_i p^*_{ij}$.

3. The net flow randomized shortest paths dissimilarity

In this section, we introduce the **net flow RSP** dissimilarity to extend the standard RSP dissimilarity developed in [30,42,49]. Similarly to the standard RSP, the **net flow RSP** corresponds to the expected cost for reaching target node t from source node s, but with the important difference that *net flows* are considered instead of raw flows. These measures are now introduced in this section.

3.1. Definition of the net edge flow

In some situations, such as electric networks [13], only net flows matter. Intuitively, this means that the edge flows in opposite directions $i \to j$ and $j \to i$ compensate each other so that only the positive net flow, provided by $|\bar{n}_{ij} - \bar{n}_{ji}|$, is taken into account, where edge flows are given by Eq. (7). In many situations, net flows look intuitively more natural because of the argument of flow compensation common to electricity. Net flows have already been used in the RSP framework in order to define node betweenness measures [29], generalizing two previous models based on electric currents [6,36]. They are further investigated in this section in order to define a new dissimilarity measure between nodes of a graph.

Inspired by electric networks [13], the non-negative net flow in each edge (i,j), denoted here as j_{ij}, is defined from Eq. (7) by

$$j_{ij} = \max\left((\bar{n}_{ij} - \bar{n}_{ji}), 0\right) = \delta(\bar{n}_{ij} > \bar{n}_{ji})\,(\bar{n}_{ij} - \bar{n}_{ji}) \tag{10}$$

where $\delta(\mathrm{p}) = 1$ if proposition p is true and 0 otherwise. In matrix form,

$$\mathbf{J} = \max\left(\left(\mathbf{N} - \mathbf{N}^{\mathrm{T}}\right), \mathbf{0}\right) \tag{11}$$

where the maximum is taken elementwise. This means that, for each edge, the net flow is defined (that is, positive) in only one direction,[6] and is equal to zero in the other direction. From their definition, net flows are also conserved on intermediate nodes $\mathcal{V} \setminus \{s,t\}, \sum_{i=1}^{n}(j_{ij} - j_{ji}) = 0$ (net input flow to j is equal to net output flow from j). Interestingly, because the flow is equal to zero in one of the two edge directions, the net flow defines a directed graph from the source to the destination node, even if the original graph is undirected.

3.2. Expected net cost and net RSP dissimilarity measure

The **expected cost** until absorption by target node t at temperature T can easily be computed in closed form from the RSP formalism [42]. This expected cost spread in the network has been used as a dissimilarity measure between nodes [30,49] and has been termed the directed **RSP dissimilarity**. More formally, the expected cost spread in the network is given by

$$\langle \tilde{c} \rangle = \sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp)\tilde{c}(\wp) \tag{12}$$

Let us now express the cost along path $\wp$ as $\tilde{c}(\wp) = \sum_{(i,j)\in\mathcal{E}} \eta((i,j) \in \wp)\, c_{ij}$, where we saw that $\eta((i,j) \in \wp)$ is the number of times edge (i,j) appears on path $\wp$ and $\mathcal{E}$ is the set of edges. Injecting this last result in Eq. (12) provides

$$\langle \tilde{c} \rangle = \sum_{(i,j)\in\mathcal{E}} \Bigg(\underbrace{\sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp)\, \eta\big((i,j) \in \wp\big)}_{\text{expected number of passages } \bar{n}_{ij}} \Bigg) c_{ij} = \sum_{(i,j)\in\mathcal{E}} \bar{n}_{ij} c_{ij} \tag{13}$$

or, in matrix form,

$$\langle \tilde{c} \rangle = \mathbf{e}^{\mathrm{T}}(\mathbf{N} \circ \mathbf{C})\mathbf{e} \tag{14}$$

where $\circ$ is the elementwise (Hadamard) matrix product, $\mathbf{e}$ is a column vector of 1s, and $\mathbf{N}$ is the matrix of expected number of passages through edges defined in Eq. (7). Intuitively, this quantity is just the cumulative sum of the expected number of passages through each edge times the cost of following the edge.

When dealing with net flows instead, Eq. (14), now computing the expected *net cost*, becomes

$$\langle \tilde{c}_{\mathrm{net}} \rangle = \mathbf{e}^{\mathrm{T}}\left[\max\left(\left(\mathbf{N} - \mathbf{N}^{\mathrm{T}}\right), \mathbf{0}\right) \circ \mathbf{C}\right]\mathbf{e} = \mathbf{e}^{\mathrm{T}}(\mathbf{J} \circ \mathbf{C})\mathbf{e} \tag{15}$$

This quantity can be interpreted as the net cost needed to reach the target node t from the source node s in a biased random walk (defined by Eq. (3) or Eq. (9)) attracting the walker toward the target node t. It is, therefore, the equivalent of the

[6] Another common convention is to consider $j_{ij} = (\bar{n}_{ij} - \bar{n}_{ji})$, and thus $j_{ji} = -j_{ij}$.

expected first passage cost defined in Markov chain theory, translated into the RSP formalism and for net flows. It can be seen as a directed dissimilarity between node s and node t, taking both proximity and amount of connectivity between s and t into account.

When the temperature is low, $T \to 0^+$, the directed dissimilarity $\langle \tilde{c}_{\text{net}} \rangle_{st}$ reduces to the least-cost dissimilarity between s and t, while when $T \to \infty$, it tends to the expected net cost for a random walker moving according to the reference random walk (and thus electric current). This quantity is in fact equivalent to the so-called R_p distance introduced in [38] for $p = 1$, that is, the weighted-by-costs sum of the net flows in the case of a pure random walk (electric current).

Therefore, the **net flow RSP dissimilarity** (nRSP, the counterpart of the standard RSP dissimilarity [30,42,49]) between node s and node t is defined as the symmetrized quantity

$$\Delta_{st}^{\text{nRSP}} = \langle \tilde{c}_{\text{net}} \rangle_{st} + \langle \tilde{c}_{\text{net}} \rangle_{ts} \tag{16}$$

where the starting and ending nodes are specified again. This is similar to the symmetric commute-cost quantity appearing in Markov chains [16], characterizing their relative accessibility [9].

Notice the difference between this quantity and the energy spread in an electric network. Indeed, if the costs are viewed as resistances, then, in the context of a resistive network, the energy weights the costs by the *squared* net flow, instead of by the simple net flow in Eq. (15) [13].

3.3. Net flows define a directed acyclic graph

Let us now show that the RSP net flows to a fixed target node t define a directed acyclic graph (DAG) when the reference probabilities are defined by Eq. (1) on a weighted undirected graph G. This comes from the fact that the net flows provided by Eq. (10) can be considered as an electric current generated from a new graph $\widehat{G}_t$ derived from G by redefining its edge weights. In addition, electric currents define a DAG because current always follows edges in the direction of decreasing potential (voltage). This potential therefore defines a topological ordering of the nodes, from a higher potential to a lower one (and lowest on the target node).

More precisely, for a fixed target t, let us define the graph $\widehat{G}_t$ by considering the following weights on edges (i,j) (conductances in electric circuits)

$$\hat{a}_{ij} \triangleq z_{it} w_{ij} z_{jt} d_i = z_{it} p_{ij}^{\text{ref}} \exp\left[-\theta c_{ij}\right] z_{jt} d_i = z_{it} a_{ij} \exp\left[-\theta c_{ij}\right] z_{jt} \tag{17}$$

which is symmetric when $\mathbf{A}$ and $\mathbf{C}$ are symmetric (undirected graph). In this derivation, we used Eqs. (1) and (6). In matrix form, this reads $\hat{\mathbf{A}} = \mathbf{Diag}(\mathbf{col}_t(\mathbf{Z}))(\mathbf{A} \circ \exp[-\theta \mathbf{C}])\mathbf{Diag}(\mathbf{col}_t(\mathbf{Z}))$ where $\mathbf{z}_t = \mathbf{col}_t(\mathbf{Z})$ extracts column t (containing elements z_{it}) of matrix $\mathbf{Z}$ and $\mathbf{Diag}(\mathbf{z}_t)$ defines a diagonal matrix from vector $\mathbf{z}_t$.

The natural transition probabilities on this new graph $\widehat{G}_t$ are provided by Eq. (1) where we replace a_{ij} by $\hat{a}_{ij}$,

$$\hat{p}_{ij} = \frac{\hat{a}_{ij}}{\sum_{j'=1}^{n} \hat{a}_{ij'}} = \frac{w_{ij} z_{jt}}{\sum_{j'=1}^{n} w_{ij'} z_{j't}} = \frac{z_{jt}}{z_{it}} w_{ij} \quad \text{for all } i \neq t \tag{18}$$

which, from Eq. (9), are exactly the optimal RSP transition probabilities. Note that we used the relation $z_{it} = \sum_{j'=1}^{n} w_{ij'} z_{j't} + \delta_{it}$, which can be easily derived from the definition of the fundamental matrix, $(\mathbf{I} - \mathbf{W})\mathbf{Z} = \mathbf{I}$ (Eq. (6); see also, e.g., [30,42,49]).

This shows that the net flows resulting from the optimal biased random walk provided by Eq. (9) are generated by a natural random walk on $\widehat{G}_t$ where target node t is made absorbing (an absorbing Markov chain). From the close relationship[7] between random walks and electric current on an undirected graph [13], this current defines a DAG on $\widehat{G}_t$. From Eq. (10), the corresponding **net flow transition probabilities** on the DAG are

$$p_{ij}^{\text{net}*} = \frac{j_{ij}}{\sum_{j'=1}^{n} j_{ij'}} \tag{19}$$

Let us now turn to the description of an algorithm that enables all pairs of net flow distances to be computed on a graph.

3.4. Computation of the net flow randomized shortest paths dissimilarity

This subsection shows how the net flow RSP dissimilarity between all pairs of nodes (Eq. (16)) can be computed in matrix form on an undirected graph. Unfortunately, the computation of these net flow RSP dissimilarities is more time-consuming than computing the standard RSP dissimilarities, for which it suffices to perform a matrix inversion [30]. This is because, before being able to compute the dissimilarities, we need to find the net flows, which involves a non-linear function (max). It is, however, still feasible for small- to medium-size networks.

The algorithm for computing the net flow RSP dissimilarity is detailed in Algorithm 1. It uses a trick introduced in [29] for calculating the net flows between all source-destination pairs s,t in a particular edge (i,j) without having to explicitly turn

[7] Net flows defined by a random walk on an undirected graph $\widehat{G}_t$, where node t is made absorbing, correspond to the electric currents (see [13], p. 50).

node t into a killing, absorbing node. More specifically, the procedure is a simple adaptation of Algorithm 2 in [29] (following Eq. (12) in this work, providing net flows) to the case of an undirected graph and the computation of net flow dissimilarities, rather than betweenness centrality. It is also optimized in order to loop over (undirected) edges only once: on line 10, the contributions of the two directions of edge (i,j) (one is necessarily equal to 0 and the other is equal to $|\bar{n}_{ij} - \bar{n}_{ji}|$) are summed together.

Following [29], its time complexity is $\mathcal{O}(n^3 + mn^2)$, where m is the number of edges and n the number of nodes, or $\mathcal{O}(mn^2)$ overall because $m \geqslant n$ for an undirected, connected, graph. Indeed, the algorithm contains a matrix inversion, which is $\mathcal{O}(n^3)$. Thereafter, there is a loop over all edges that repeats some standard matrix operations of order $\mathcal{O}(n^2)$, which finally provides $\mathcal{O}(n^3 + mn^2)$. Therefore, the algorithm does not scale well on large graphs; in its present form, it can only be applied on medium-size graphs.

Algoithm 1: Computing the net flow randomized shortest paths dissimilarity matrix (inspired by [29]).

Input:

- A weighted, undirected, connected graph G containing n nodes.
- The $n \times n$ symmetric adjacency matrix $\mathbf{A}$ associated to G, containing non-negative affinities.
- The $n \times n$ reference transition probabilities matrix $\mathbf{P}_{\text{ref}}$ associated to G (usually, the transition probabilities associated to the natural random walk on the graph, $\mathbf{P}_{\text{ref}} = \mathbf{D}^{-1}\mathbf{A}$ where $\mathbf{D}$ is the outdegree diagonal matrix).
- The $n \times n$ symmetric cost matrix $\mathbf{C}$ associated to G, defining non-negative costs of transitions.
- The inverse temperature parameter $\theta > 0$.

Output:

- The $n \times n$ randomized shortest paths net flow dissimilarity matrix $\boldsymbol{\Delta}$ defined on all source-destination pairs.

1. $\mathbf{W} \leftarrow \mathbf{P}_{\text{ref}} \circ \exp[-\theta \mathbf{C}]$ ▷elementwise exponential and multiplication $\circ$
2. $\mathbf{Z} \leftarrow (\mathbf{I} - \mathbf{W})^{-1}$ ▷the fundamental matrix
3. $\boldsymbol{\Delta} \leftarrow \mathbf{0}$ ▷initialize the $n \times n$ net RSP flow dissimilarity matrix
4. **for** $i = 1$ to n **do** ▷compute contribution of each node i
5. $\quad \mathbf{z}_i^{\text{c}} \leftarrow \mathbf{col}_i(\mathbf{Z}), \mathbf{z}_i^{\text{r}} \leftarrow \mathbf{row}_i(\mathbf{Z})$ ▷copy column i and row i of $\mathbf{Z}$ transformed into a column vector
6. $\quad$ **for** $j \in \mathcal{N}(i)$ with $j > i$ **do** ▷loop on neighboring nodes j, considering each (undirected) edge only once
7. $\quad\quad \mathbf{z}_j^{\text{c}} \leftarrow \mathbf{col}_j(\mathbf{Z}), \mathbf{z}_j^{\text{r}} \leftarrow \mathbf{row}_j(\mathbf{Z})$ ▷copy column j and row j of $\mathbf{Z}$ transformed into a column vector
8. $\quad\quad \mathbf{N}_{ij} \leftarrow w_{ij}\left[\left(\mathbf{z}_i^{\text{c}}\left(\mathbf{z}_j^{\text{r}}\right)^{\text{T}} \div \mathbf{Z}\right) - \mathbf{e}\left(\left(\mathbf{z}_i^{\text{c}} \circ \mathbf{z}_j^{\text{r}}\right) \div \mathbf{diag}(\mathbf{Z})\right)^{\text{T}}\right]$ ▷matrix of flow in edge $i \to j$ for all source-destination pairs (see [29], Eq. (17))
9. $\quad\quad \mathbf{N}_{ji} \leftarrow w_{ji}\left[\left(\mathbf{z}_j^{\text{c}}\left(\mathbf{z}_i^{\text{r}}\right)^{\text{T}} \div \mathbf{Z}\right) - \mathbf{e}\left(\left(\mathbf{z}_j^{\text{c}} \circ \mathbf{z}_i^{\text{r}}\right) \div \mathbf{diag}(\mathbf{Z})\right)^{\text{T}}\right]$ ▷matrix of flow in edge $j \to i$ for all source-destination pairs (see [29], Eq. (17))
10. $\quad\quad \mathbf{Net}_{ij} \leftarrow \text{abs}(\mathbf{N}_{ij} - \mathbf{N}_{ji})$ ▷net flow contribution from edge $i \leftrightarrow j$
11. $\quad\quad \boldsymbol{\Delta} \leftarrow \boldsymbol{\Delta} + c_{ij}\mathbf{Net}_{ij}$ ▷update dissimilarity matrix with the contribution of edge $i \leftrightarrow j$
12. $\quad$ **end for**
13. **end for**
14. $\boldsymbol{\Delta} \leftarrow \boldsymbol{\Delta} + \boldsymbol{\Delta}^{\text{T}}$ ▷the resulting net RSP dissimilarities matrix
15. **return** $\boldsymbol{\Delta}$

4. Considering edge flow capacity constraints

In this section, an algorithm computing the optimal policy (the equivalent of Eq. (9)) under flow capacity constraints on edges is derived. It is based on the fact that linear inequality constraints can easily be integrated in maximum entropy problems by considering Lagrangian duality (see, e.g., [28] and references therein).

For convenience, we assume a weighted, undirected, connected graph G with a single source node (node s) and a single target node (node $t \neq s$). An input flow is injected into node s and absorbed by node t, but the model can easily be generalized for multiple sources and destinations. As before, it is assumed that the target node t is killing and absorbing, meaning that the transition probabilities $p_{tj}^{\text{ref}} = 0$ for all nodes j, including node $j = t$.

The idea now is to constrain the flow visiting some edges belonging to a set of constrained edges $\mathcal{C}$. The expected number of passages through these edges (see Eq. (7)) is therefore forced not to exceed some predefined values (upper bound),

$$\bar{n}_{ij} \leqslant \sigma_{ij} \quad \text{for edges } (i,j) \in \mathcal{C} \tag{20}$$

which ensures that the flows on edges in $\mathcal{C}$ are limited by the capacities $\sigma_{ij} > 0$ and thus must remain in the interval $[0, \sigma_{ij}]$. In this section, we consider that, although the graph G is assumed undirected here, each capacity constraint is directed and

thus active in only one direction of an edge. Therefore, each undirected edge $i \leftrightarrow j$ of G possibly leads to two directed edges (i,j) and (j,i) in $\mathcal{C}$, reflecting a possibly different capacity constraint in each of the two directions. Thus, the set of constrained edges $\mathcal{C}$ contains directed edges, limiting the directional flow through them.

Moreover, we assume that the constraints are feasible, which is discussed later.[8] The problem aims to minimize the free energy objective function (Eq. (2)) while satisfying these inequality constraints.

4.1. The Lagrange function in case of capacity constraints

From nonlinear optimization theory (see, e.g., [20]), the Lagrange function (following Eq. (2)) is

$$\begin{aligned}
\mathscr{L}(\mathrm{P},\boldsymbol{\lambda}) &= \sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) + \mu\left(\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) - 1\right) \\
&\quad + \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\left(\bar{n}_{ij} - \sigma_{ij}\right) \\
&= \underbrace{\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)}_{\text{free energy, } \phi(\mathrm{P})} + \mu\left(\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) - 1\right) \\
&\quad + \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\Bigg(\underbrace{\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\,\eta\big((i,j)\in\wp\big)}_{\bar{n}_{ij}\text{(see Eq. 7)}} - \sigma_{ij}\Bigg)
\end{aligned} \tag{21}$$

where there is a Lagrange parameter μ associated with the sum-to-one constraint and a Lagrange parameter λ_{ij} associated with each constrained edge. The Lagrange parameters $\{\lambda_{ij}\}, (i,j)\in\mathcal{C}$, are all non-negative in the case of inequality constraints [20] and are stacked into the parameter vector $\boldsymbol{\lambda}$.

Note that, by inspecting (21) and from the convexity of the Kullback-Leibler divergence, the primal objective function to be minimized with respect to the discrete probabilities (which is similar to Eq. (2)) is convex, the equality constraints are all linear, and the inequality constraints form a convex set. Therefore, the duality gap between the primal and dual problems is zero, which will be exploited for solving the problem [20].

The Lagrange function in Eq. (21) can be rearranged as

$$\begin{aligned}
\mathscr{L}(\mathrm{P},\boldsymbol{\lambda}) &= \sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\underbrace{\left(\tilde{c}(\wp) + \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\,\eta\big((i,j)\in\wp\big)\right)}_{\text{augmented cost } \tilde{c}'(\wp) \text{ cumulated on path } \wp} \\
&\quad + T\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) + \mu\left(\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) - 1\right) - \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\sigma_{ij} \\
&= \sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) \sum_{(i,j)\in\mathcal{E}} \eta\big((i,j)\in\wp\big)\underbrace{\left(c_{ij} + \delta\big((i,j)\in\mathcal{C}\big)\,\lambda_{ij}\right)}_{\text{augmented costs } c'_{ij}} \\
&\quad + T\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) + \mu\left(\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) - 1\right) - \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\sigma_{ij} \\
&= \underbrace{\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\tilde{c}'(\wp) + T\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)}_{\text{free energy based on augmented costs, } \phi'(\mathrm{P})} + \mu\left(\sum_{\wp\in\mathcal{P}_{st}} \mathrm{P}(\wp) - 1\right) \\
&\quad - \sum_{(i,j)\in\mathcal{C}} \lambda_{ij}\sigma_{ij}
\end{aligned} \tag{22}$$

[8] If not, the duality gap in our algorithm will not converge to zero.

where the symbol $\delta((i,j) \in \mathcal{C})$ is defined as 1 when edge $(i,j) \in \mathcal{C}$ and 0 otherwise. Note that we used $\tilde{c}(\wp) = \sum_{(i,j)\in\mathcal{E}} \eta((i,j) \in \wp)\, c_{ij}$ (Eq. (13)) to compute the total cost along path $\wp$.

During this derivation, we observed that the costs c_{ij} can be redefined into **augmented costs** that integrate the additional "virtual" costs (the Lagrange parameters) needed for satisfying the constraints,

$$c'_{ij} = \begin{cases} c_{ij} + \lambda_{ij} & \text{when edge } (i,j) \in \mathcal{C} \\ c_{ij} & \text{otherwise} \end{cases} \tag{23}$$

where $\mathbf{C}' = \left(c'_{ij}\right)$ is the matrix containing these augmented costs. Thus, the Lagrange parameters have an interpretation similar to the dual variables in linear programming: they represent the extra cost to pay, associated with each edge, in order to satisfy the constraints [20]. This is also common in many network flow problems [1].

Let $\phi'(\mathrm{P})$ be the free energy obtained in Eq. (22) from these augmented costs (Eq. (23)). We now turn to the problem of finding the Lagrange parameters λ_{ij} by exploiting Lagrangian duality.

4.2. Exploiting Lagrangian duality

In this subsection, we will take advantage of the fact that, in the formulation of the problem (close to maximum entropy arguments), the Lagrange dual function and its gradient are relatively easy to compute; see, for instance, [28] for similar arguments in the context of supervised classification. Indeed, as the objective function is strictly convex, the equality constraints are linear and the support set for the path probabilities is convex, it is known that, provided that the problem is feasible,[9] there is only one global minimum and the duality gap is zero [20]. The optimum is a saddle point of the Lagrange function, and a common optimization procedure (sometimes called the Arrow-Hurwicz-Uzawa algorithm) consists of sequentially (i) solving the primal (finding the optimal probability distribution) while considering the Lagrange parameters as fixed, and then (ii) maximizing the dual (which is concave) with respect to the Lagrange parameters, until convergence.

In our context, this provides the following steps [20] which are iterated,

$$\begin{cases} \mathcal{L}(\boldsymbol{\lambda}) = \mathcal{L}(\mathrm{P}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) = \underset{\{\mathrm{P}(\wp)\}_{\wp\in\mathcal{P}_{st}}}{\text{minimize}} \mathcal{L}(\mathrm{P}, \boldsymbol{\lambda}) & \text{(compute the dual function)} \\ \boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}) & \text{(maximize the dual function)} \\ \boldsymbol{\lambda} = \boldsymbol{\lambda}^* & \text{(update } \boldsymbol{\lambda}\text{)} \end{cases} \tag{24}$$

This is the procedure that will be followed whereby the dual function will be maximized through a simple gradient ascent procedure.

4.2.1. Computing the dual function

To compute the dual function $\mathcal{L}(\mathrm{P}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda})$ in Eq. (24), we first have to find the optimal probability distribution P^* in terms of the Lagrange parameters. We thus have to compute the minimum of Eq. (22) for a constant $\boldsymbol{\lambda}$. But this Lagrange function (Eq. (22)) is identical to the Lagrange function associated with the standard RSP optimization problem (Eq. (2)), except that the costs c_{ij} are replaced by the augmented costs c'_{ij}, and that the introduction of the final additional term does not depend on the probability distribution. Therefore, the probability distribution $\mathrm{P}(\cdot)$ minimizing Eq. (22) is a Gibbs-Boltzmann distribution of the form of Eq. (3), but it now depends on the augmented costs instead of the original costs.

Next, we replace the probability distribution $\mathrm{P}(\cdot)$ in $\mathcal{L}(\mathrm{P}, \boldsymbol{\lambda})$ by the optimal Gibbs-Boltzmann distribution $\mathrm{P}^*(\cdot)$, which depends on the augmented costs and thus also on the Lagrange parameters. From the result of Eq. (5), the obtained dual function[10] (Eq. (24)) is

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathcal{L}(\mathrm{P}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) = -T \log \mathcal{Z}' - \sum_{(i,j)\in\mathcal{C}} \lambda_{ij} \sigma_{ij} \tag{25}$$

In this equation, $\mathcal{Z}'$ is the partition function (see Eq. (4)), computed from the augmented costs $\mathbf{C}'$, which depends on $\boldsymbol{\lambda}$. We now need to maximize this dual function with respect to these Lagrange parameters.

[9] Recall that we assume that the problem is feasible; see the discussion at the end of SubSection 4.3.

[10] We leave out the sum-to-one constraint term which does not depend on $\boldsymbol{\lambda}$.

4.2.2. Maximizing the dual function

The maximization of the dual function can be done, by, for example, using the simple method developed by Rockafellar (see [40], Eqs. (10) and (12)).

But let us first compute the gradient of the dual function (Eq. (25)) with respect to the non-negative λ_{ij} defined on edges $(i,j) \in \mathcal{C}$. From $-T\,\partial \log \mathcal{Z}/\partial c_{ij} = \bar{n}_{ij}$ (Eq. (7)) and $\partial c'_{ij}/\partial \lambda_{ij} = \delta((i,j) \in \mathcal{C})$ (Eq. (23)), this gradient is simply $\partial \mathcal{L}(\boldsymbol{\lambda})/\partial \lambda_{ij} = \partial\left(-T \log \mathcal{Z}' - \sum_{(k,l)\in\mathcal{C}} \lambda_{kl}\sigma_{kl}\right)/\partial \lambda_{ij} = \bar{n}_{ij} - \sigma_{ij}$, where $\bar{n}_{ij}$ is computed from the augmented costs. It can be observed that we simply recover the expressions for the capacity constraints; this is actually a standard result when dealing with maximum entropy problems (see, e.g., [10,28]).

For computing the Lagrange parameters, we thus follow [40] who proposed the following (gradient-based) updating rule,[11]

$$\lambda_{ij} \leftarrow \max\left(\lambda_{ij} + \alpha(\bar{n}_{ij} - \sigma_{ij}), 0\right) \quad \text{for all } (i,j) \in \mathcal{C} \tag{26}$$

which is guaranteed to converge in the concave case, as long as α is positive, is not too large, and the problem is feasible [40]. This expression states that, if the flow in an edge (i,j) is too large (that is, $\bar{n}_{ij} - \sigma_{ij} > 0$), the augmented cost should increase in order to reduce this flow. On the contrary, if the flow is below the capacity threshold, the augmented cost should decrease until eventually reach the normal cost value c_{ij} (when $\lambda_{ij} = 0$). Notice also that, because costs are multiplied by θ in the model (see Eq. (6)), it becomes insensitive to cost variations when θ is small (close to 0, an almost random walk behavior). We therefore set the α parameter proportional to $1/\theta$ in Eq. (26) during our experiments.

Of course, we could use other, more sophisticated and more efficient, optimization techniques (see, e.g., [20]), but this simple procedure worked satisfactorily for our tests on small- to medium-size graphs. The parameter α has to be tuned manually for each different dataset, but this was not a problem. However, in its present form, the algorithm does not scale to larger graphs.

4.3. The resulting algorithm

The resulting algorithm is presented in Algorithm 2.[12] It computes the optimal policy (Eq. (9)), minimizing the objective function (Eq. (2)) while satisfying the inequality constraints (Eq. (20)). This optimal policy guides the random walker to the target state with a trade-off between exploitation and exploration that is monitored by the inverse temperature parameter $\theta = 1/T$. The different steps of the procedure are as follows:

- Initialize the Lagrange parameters to 0 and the augmented costs to the original edge costs $\mathbf{C}$.
- Iterate the following steps until convergence:
  - The required elements of the fundamental matrix are recomputed from the current augmented costs (Eq. (6)) by solving two systems of linear equations.
  - The expected number of passages through each edge (edge flows) is computed (Eq. (7)).
  - The Lagrange parameters and the augmented costs are updated (Eqs. (26) and (23)).
- Compute and return the optimal policy (transition probabilities) according to Eq. (9).

Because two systems of n linear equations need to be solved at each iteration, its complexity is of the order $\mathcal{O}(kn^3)$, depending on the number of iterations k needed for convergence. This number of iterations is unknown in advance but could become large depending on the problem and the gradient step. However, for sparse graphs, the complexity could be reduced by taking advantage of special numerical methods for solving sparse systems of linear equations.

Note also that lower bound (instead of upper bound) constraints on the expected flows have also been considered – in this case the flow is constrained to be greater or equal (and not lesser or equal) to a threshold value. However, the resulting virtual costs (Lagrange parameters) in this case can become negative in order to augment the flow in the edge. This sometimes leads to numerical instabilities when some costs become negative and negative cycles appear. One quick fix is to prohibit negative costs; however, by doing this, the problem becomes unfeasible in some situations. Therefore, we decided to leave the study of lower-bounded capacities for further work.

As a last remark, note that it was assumed that the problem is feasible, which can be difficult to check in practice. Indeed, the model injects a unit flow into the network so that the maximum flow through the network must be at least equal to one. This means that we cannot blindly assign capacities because, in the case where the problem is not feasible, the algorithm does not converge. One way to check the overall capacity of the network is to run a standard max-flow algorithm [1,32].

[11] Note that [40] uses 2α for the error correction term (instead of α here).

[12] In this pseudocode, $\mathbf{e}_i$ is a column (basis) vector of 0s except on row i where it contains a 1.

Algoirthm 2: Randomized shortest paths with capacity constraints.

Input:

– A weighted, undirected, connected graph G containing n nodes. Node s is the source node and node t the target node.
– The $n \times n$ reference transition probabilities matrix $\mathbf{P}_{\text{ref}}$ associated to G.
– The $n \times n$ symmetric cost matrix $\mathbf{C}$ associated to G, defining non-negative costs of transitions.
– The set of constrained edges $\mathcal{C}$ (see the text for details).
– The set of non-negative capacities on flows, $\{\sigma_{ij}\}$, defined on the set of constrained edges, $(i,j) \in \mathcal{C}$.
– The inverse temperature parameter $\theta > 0$.
– The gradient ascent step $\alpha > 0$.

Output:

– The $n \times n$ randomized policy provided by the transition matrix $\mathbf{P}^*$, defining a biased random walk on G satisfying the capacity constraints.

$\boldsymbol{\lambda} \leftarrow \mathbf{0}$ ▷initialize the $|\mathcal{C}| \times 1$ Lagrange parameters vector
$\mathbf{C}' \leftarrow \mathbf{C}$ ▷initialize the augmented costs matrix
Set row t of matrix $\mathbf{P}_{\text{ref}}$ to $\mathbf{0}^{\mathrm{T}}$ ▷target node t is made absorbing and killing
repeat ▷main iteration loop
  $\mathbf{W} \leftarrow \mathbf{P}_{\text{ref}} \circ \exp\left[-\theta \mathbf{C}'\right]$ ▷update $\mathbf{W}$ matrix (elementwise exponential and multiplication $\circ$)
  Solve $(\mathbf{I} - \mathbf{W})\mathbf{z}_t = \mathbf{e}_t$ ▷backward variables $\mathbf{z}_t$ (column t of the fundamental matrix $\mathbf{Z}$) with elements z_{it}
  Solve $(\mathbf{I} - \mathbf{W})^{\mathrm{T}}\mathbf{z}_s = \mathbf{e}_s$ ▷forward variables $\mathbf{z}_s$ (row s of the fundamental matrix $\mathbf{Z}$ viewed as a column vector) with elements z_{si}
  $\mathbf{N} \leftarrow \frac{\mathbf{Diag}(\mathbf{z}_s)\mathbf{W}\mathbf{Diag}(\mathbf{z}_t)}{z_{st}}$ ▷compute the expected number of passages in each edge (see Eq. (7))
  for all $(i,j) \in \mathcal{C}$ **do** ▷gradient ascent: update all quantities associated to constrained edges
    $\lambda_{ij} \leftarrow \max\left(\lambda_{ij} + \alpha(\bar{n}_{ij} - \sigma_{ij}), 0\right)$ ▷update Lagrange parameters
    $c'_{ij} \leftarrow c_{ij} + \lambda_{ij}$ ▷update augmented costs
  end for
until convergence
$\mathbf{P}^* \leftarrow (\mathbf{Diag}(\mathbf{z}_t))^{-1}\mathbf{W}\mathbf{Diag}(\mathbf{z}_t)$ ▷compute optimal policy
return $\mathbf{P}^*$

5. Dealing with net flow capacity constraints

Let us now consider the case where the capacities $\sigma_{ij} > 0$ with $(i,j) \in \mathcal{C}$ are defined on *net flows* instead of raw flows. As before, it is assumed in this section that the original graph is undirected and that the adjacency matrix as well as the cost matrix are symmetric. In this situation (see Eq. (10) and its discussion), we now consider that the constraints operate on the net flows instead of on the raw flows,

$$\begin{aligned} & j_{ij} = \max\left((\bar{n}_{ij} - \bar{n}_{ji}), 0\right) \leqslant \sigma_{ij}, \\ & \text{or equivalently both} \begin{cases} \bar{n}_{ij} - \bar{n}_{ji} \leqslant \sigma_{ij} \\ \bar{n}_{ji} - \bar{n}_{ij} \leqslant \sigma_{ij} \end{cases} \text{and for each edge } (i,j) \in \mathcal{C} \end{aligned} \tag{27}$$

In the net flows setting, we further assume that σ_{ji} is always defined when there exists a capacity constraint σ_{ij} and that $\sigma_{ji} = \sigma_{ij}$ (symmetry). In Eq. (27), only one among the two flow differences is positive so that the constraint only operates in this direction of the flow: the second constraint is automatically satisfied. This also means that only one of the two constraints can become active.

To summarize, if the set of constraint nodes $\mathcal{C}$ contains edge (i,j), it also necessarily contains its reciprocal (j,i) (they come as a pair) with the same capacity value, $\sigma_{ji} = \sigma_{ij}$. We now turn to the definition of the Lagrange function and the derivation of the algorithm.

5.1. The Lagrange function in case of net flow capacity constraints

After rearranging the terms as in Eq. (22) and, once again, using $\bar{n}_{ij} = \sum_{\wp \in \mathcal{P}_{st}} \mathrm{P}(\wp)\, \eta((i,j) \in \wp)$, the Lagrange function then becomes

$$\begin{aligned}\mathcal{L}(\mathrm{P},\boldsymbol{\lambda}) &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\tilde{c}(\wp)+T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\log\left(\tfrac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)+\mu\left(\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)-1\right)\\ &\quad+\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\left[\left(\bar{n}_{ij}-\bar{n}_{ji}\right)-\sigma_{ij}\right]\\ &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\left(\tilde{c}(\wp)+\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\left(\eta((i,j)\in\wp)-\eta((j,i)\in\wp)\right)\right)\\ &\quad+T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\log\left(\tfrac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)+\mu\left(\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)-1\right)-\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\sigma_{ij}\end{aligned}\tag{28}$$

Now, from the symmetry of edges (edges are present in pairs; for each $(i,j)\in\mathcal{C}:(j,i)\in\mathcal{C}$ and $\sigma_{ij}=\sigma_{ji}$), we deduce $\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\eta((j,i)\in\wp)=\sum_{(j,i)\in\mathcal{C}}\lambda_{ij}\eta((j,i)\in\wp)=\sum_{(i,j)\in\mathcal{C}}\lambda_{ji}\eta((i,j)\in\wp)$. Injecting this result into Eq. (28) and proceeding in the same way as in Eq. (22) provides

$$\begin{aligned}\mathscr{L}(\mathrm{P},\boldsymbol{\lambda}) &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\underbrace{\left(\tilde{c}(\wp)+\sum_{(i,j)\in\mathcal{C}}\left(\lambda_{ij}-\lambda_{ji}\right)\eta\big((i,j)\in\wp\big)\right)}_{\text{augmented cost }\tilde{c}'(\wp)\text{ cumulated on path }\wp}\\ &\quad+T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)+\mu\left(\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)-1\right)-\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\sigma_{ij}\\ &= \sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\sum_{(i,j)\in\mathcal{E}}\eta\big((i,j)\in\wp\big)\underbrace{\left(c_{ij}+\delta\big((i,j)\in\mathcal{C}\big)\left(\lambda_{ij}-\lambda_{ji}\right)\right)}_{\text{augmented costs }c'_{ij}}\\ &\quad+T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)+\mu\left(\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)-1\right)-\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\sigma_{ij}\\ &= \underbrace{\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\tilde{c}'(\wp)+T\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)}_{\text{free energy based on augmented costs, }\phi'(\mathrm{P})}+\mu\left(\sum_{\wp\in\mathcal{P}_{st}}\mathrm{P}(\wp)-1\right)\\ &\quad-\sum_{(i,j)\in\mathcal{C}}\lambda_{ij}\sigma_{ij}\end{aligned}\tag{29}$$

which has exactly the same form as in the raw flow case (see Eq. (22)) with the exception that the definition of the augmented costs differs in the two expressions. Indeed, as before, the costs c_{ij} can be redefined into *augmented* costs,

$$c'_{ij}=\begin{cases}c_{ij}+\lambda_{ij}-\lambda_{ji} & \text{when edge } (i,j)\in\mathcal{C}\\ c_{ij} & \text{otherwise}\end{cases}\tag{30}$$

We must stress the requirement that the constraints in Eq. (27) be symmetric and come by pair. In addition, as discussed before, only one constraint can become active among the two directions (i,j) and (j,i), which implies that one of the two Lagrange multipliers $\{\lambda_{ij},\lambda_{ji}\}$ must be equal to 0: the λ_{ij} for which $(\bar{n}_{ij}-\bar{n}_{ji})<0$.

Moreover, by following the same reasoning as in the previous section (see Eqs. (25) and (26)), it can immediately be observed that the dual function has the same form as before and is provided by Eq. (25).

5.2. The resulting algorithm

In the net flow context, by proceeding in the same way as in the previous section (see SubSection 4.2.2), the gradient of the dual Lagrange function (Eq. (25)) for augmented costs provided by Eq. (30) is $d\mathcal{L}(\boldsymbol{\lambda})/d\lambda_{ij}=\left(\partial\mathcal{L}(\boldsymbol{\lambda})/\partial c'_{ij}\right)\left(\partial c'_{ij}/\partial\lambda_{ij}\right)+\left(\partial\mathcal{L}(\boldsymbol{\lambda})/\partial c'_{ji}\right)\left(\partial c'_{ji}/\partial\lambda_{ij}\right)+\partial\mathcal{L}(\boldsymbol{\lambda})/\partial\lambda_{ij}=\bar{n}_{ij}-\bar{n}_{ji}-\sigma_{ij}$. From this last result, we can derive the update of the Lagrange parameters λ_{ij} with $(i,j)\in\mathcal{C}$,

$$\lambda_{ij}\leftarrow\begin{cases}\max\left(\lambda_{ij}+\alpha\left(\bar{n}_{ij}-\bar{n}_{ji}-\sigma_{ij}\right),0\right) & \text{when } (\bar{n}_{ij}-\bar{n}_{ji})\geqslant 0\\ 0 & \text{when } (\bar{n}_{ij}-\bar{n}_{ji})<0\end{cases}\tag{31}$$

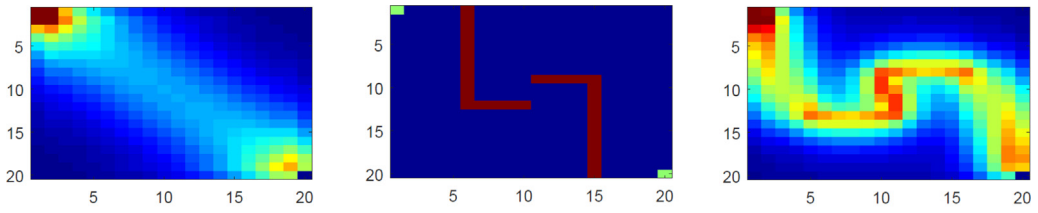

Fig. 1. Illustrative example of the capacity-constrained RSP on a 2D grid. (a) Expected number of visits to nodes (see Eq. (7)), with $\theta = 0.05$ and $\alpha = 40$. Red represents a higher expected number of visits and blue a lower number of visits. (b) Introduction of capacity constraints on edges belonging to the obstacles (walls in red). Source node s (upper left corner) and target node t (lower right corner) are in green. (c) Expected number of visits to nodes (see Eq. (7)) after introducing capacity constraints on the walls, for an intermediate value of $\theta = 0.05$ and gradient step $\alpha = 40$.

Algorithm 2 is easy to adapt in order to consider net flow capacity constraints (with $\sigma_{ij} = \sigma_{ji}$); only lines 10 and 11 must be modified according to Eqs. (30) and (31).

Notice that still another procedure could be derived when considering the edges as undirected and unique; that is, there is a unique affinity and cost value associated to each edge $i \leftrightarrow j$. In that situation, the Lagrange function is defined in terms of m costs (instead of $2m$ in this work) and augmented costs cannot differ along the edge direction. A similar algorithm updating only one Lagrange parameter per edge could be derived for this case.

In practice, we however observed that the net flow constraint algorithm is slower to converge and more sensitive to the gradient step than simple flow constraints; in other words, the problem looks harder to solve. This is especially the case for small values of θ, where the process behaves more like a random walker. Indeed, in that situation, the addition of capacity constraints seems to be partly in conflict with the objective of walking randomly and moving back and forth. One way to handle this issue [12] would be first to define a DAG (deduced, e.g., from the electric potential on nodes when imposing a + 1 potential at the source node and a 0 potential at the target node, followed by a topological sort). This will enable an RSP problem with simple capacity constraints to be solved on this DAG. This has the additional advantage that it should be more efficient (we avoid cycles and can use dynamic programming techniques to compute the RSP solution) and scale to larger graphs. This extension is left for further work.

6. Experiments

In this section, we first present two illustrative examples of the use of capacity constraints on the edge flows of a graph as well as a brief study of the scalability of the net flow Algorithm 1. Following this, we evaluate the net flow RSP on unsupervised classification tasks and compare its results to other state-of-the-art graph node distances. It is important to emphasize that our goal here was not to propose that new node clustering algorithms outperform state-of-the-art techniques. Rather, the aim was to investigate if the net flow RSP model is able to capture the community structure of networks in an accurate way, compared to other dissimilarity measures between nodes.

6.1. Illustrative examples

First example. The first example illustrates the expected number of visits to nodes (see Eq. (8)) over the RSP paths distribution between one source node s and one target node t on a 20×20 grid, in two different situations obtained after running Algorithm 2. Nodes were linked to their neighbors[13] with a unit affinity and a unit cost. In the first situation, we did not set any capacity constraint, and the expected number of visits is represented in Fig. 1a. As expected, walkers followed a trajectory close to the diagonal of the grid, representing the shortest paths between the source node and the target node.

In the second situation, we placed two obstacles (porous walls) by constraining the capacities of all the edges linking the nodes represented in red in Fig. 1b to 0.01. As can be observed in 1c, in this situation, the expected number of visits to nodes no longer concentrated around the diagonal, but instead closely followed the obstacles. This trajectory reflects the least-cost paths between the source node and the target node, avoiding the low-capacity obstacles.

Second example. Our second illustrative example was taken from [39] and makes a link between the RSP with net flow capacity constraints and the maximum flow problem. Its aim was to show that Algorithm 2 could be used to compute the maximum flow (which takes a value of 12 in this example) between the source node s and the target node t in the undirected graph G presented in Fig. 2. Each edge of this graph has a unit cost and affinity, as well as capacities shown on the drawing. Note that, in practice, because the RSP model assumes a unit input flow, all capacities are scaled[14] so that the value of the

[13] Only horizontal and vertical neighbors are considered (no diagonal edge).

[14] We divide capacities by a graph cut value (an upper bound on the min-cut), such as the cut between nodes $\{f, g, h\}$ and t: 22 in our example.

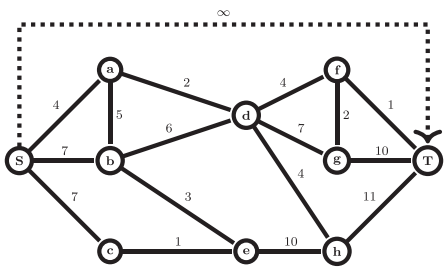

Fig. 2. A small undirected graph composed of eight nodes [39]. The values on the edges represent capacities; moreover, all edge costs are set to 1, except the added shortcut edge (dashed line) whose cost is 10. The maximum possible flow between the source node s and the target node t is 12, and is equal to the min-cut between nodes $\{a, b, c\}$ and $\{d, e\}$.

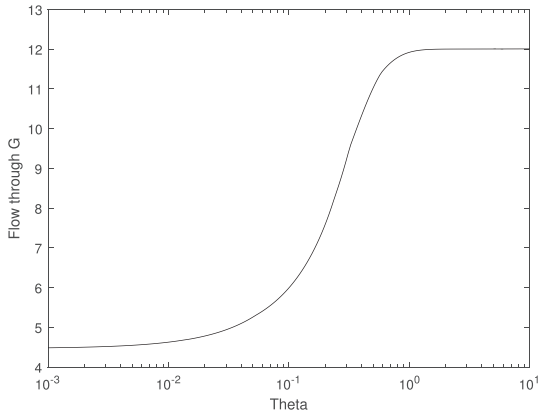

Fig. 3. Evolution of the flow between nodes $\{f, g, h\}$ and t satisfying the capacity constraints (total flow in the original graph G, without the shortcut edge), provided by Algorithm 2, in terms of the θ parameter, with a gradient step $\alpha = 1/\theta$. The intersection between the curve and the y-axis when $\theta \to 0^+$ is 4.699. Note that the x-axis is scaled logarithmically.

computed maximum flow after scaling lies between 0 and 1. The maximum flow of G is then obtained by the reverse transformation.

To find the max-flow, we added a directed edge from s to t (dashed line) with infinite capacity and an edge cost of 10 to the original graph. This new edge introduced a "shortcut" allowing the passage of all the overflow of the graph, but with a higher cost. Theoretically, our algorithm will avoid going through this shortcut as much as possible because it has a high cost compared to the other trajectories in the graph. Therefore, it should try to maximize the flow that travels through the original graph before using this shortcut edge.

Fig. 3 shows the evolution of the flow between the nodes $\{f, g, h\}$ and t (the flow through the original graph), provided by Algorithm 2, and thus satisfying the capacity constraints in terms of the value of the θ parameter. As observed in Fig. 3 and Fig. 4, this flow through the original graph reaches almost exactly the maximum flow value of 12 for all values of θ larger than 1. However, when θ is low (close to zero), the walks became increasingly random, and no longer consider costs (see Eq. (2)). For that reason, part of the total flow went through the shortcut, even if this was not optimal in terms of cost. This explains the reduction of the flow through the original graph when θ was close to zero.

6.2. Scalability experiment

In this subsection, we study the scalability of Algorithm 1, which computes the full net flow RSP dissimilarity matrix (see Eq. (16)), through a small experiment. To evaluate the time complexity, we generated 120 graphs with the benchmark algorithm of Lancichinetti, Fortunato and Radicchi (LFR) [33]. More precisely, we created 10 graphs for each of the following sizes of {50, 100, 150, 200, 250, 300, 500, 1,000, 1,500, 2,000, 2,500, 3,000} nodes. Moreover, for each size among the 10 graphs, we changed the mixing parameter value (μ) each 2 graphs between all these values {0.1, 0.2, 0.3, 0.4, 0.5}.

Algorithm 1, computing the dissimilarity matrix between all pairs of nodes, was then run on each of these graphs. We report the average computation time in seconds over the 10 graphs for each different size (in terms of number of nodes

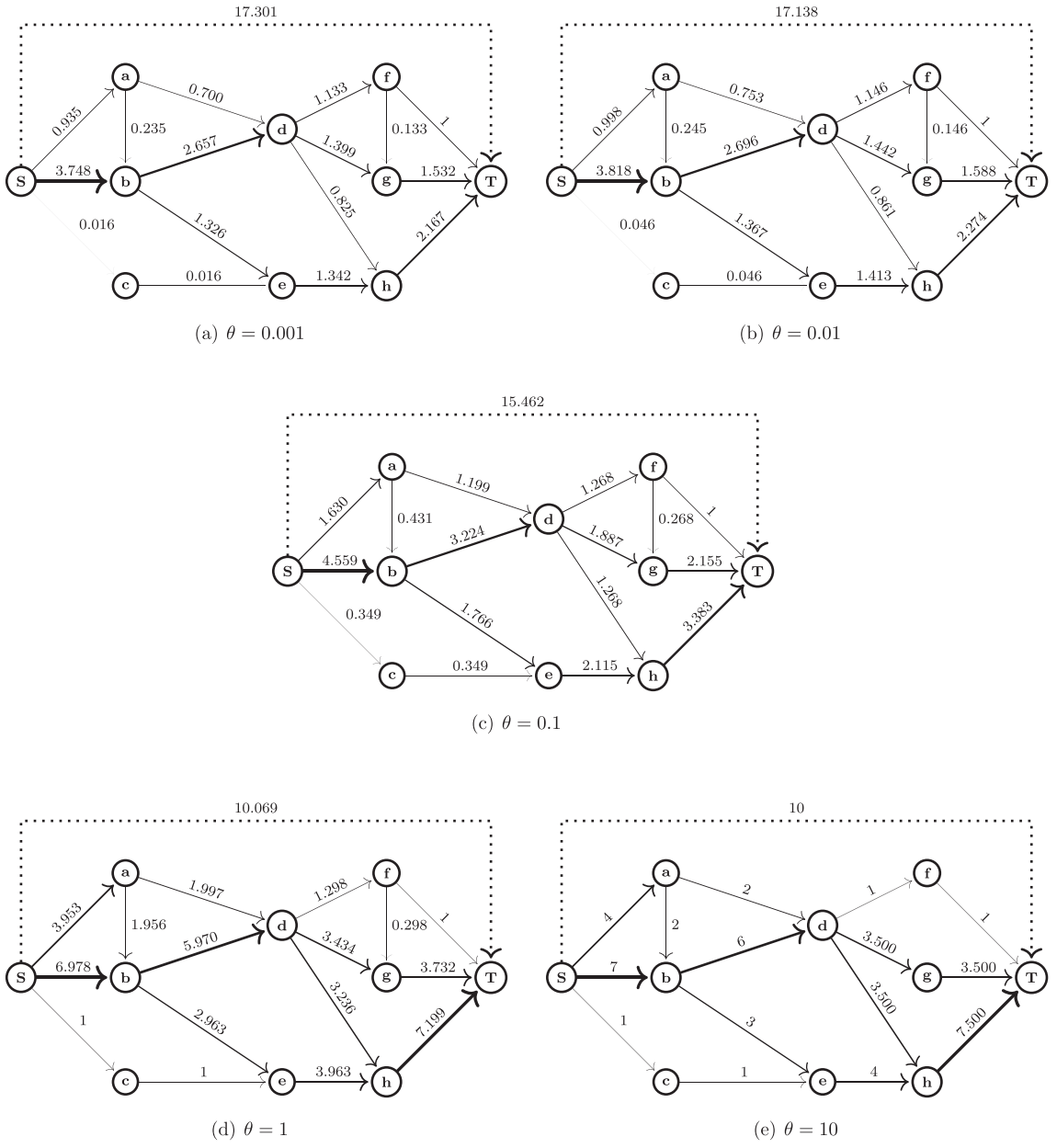

Fig. 4. Representation of the net flows from s to t depending on the value of θ, for the capacity-constrained RSP and the graph appearing in Fig. 2. The thickness of the arrows is scaled with respect to the largest net flow present in the graph.

and number of computed dissimilarities) in Fig. 5. All results were obtained with Matlab (version R2017a) running on an Intel Core i7-8750H CPU@4.10 GHz with 32 GB of RAM.

As previously mentioned, the time-complexity of this algorithm is $\mathcal{O}(n^3 + mn^2)$ with n being the number of nodes and m being the number of edges. Although applicable to medium-size graphs, it can clearly be observed that the algorithm does not scale well on large graphs in its present form, at least if the full dissimilarity matrix is needed.

6.3. Nodes clustering experiment

In this subsection, we present an application of the **Net Flow Randomized Shortest Paths** (nRSP) in a graph nodes clustering context. Note that a methodology close to [44] is used (for further details, see the cited paper).

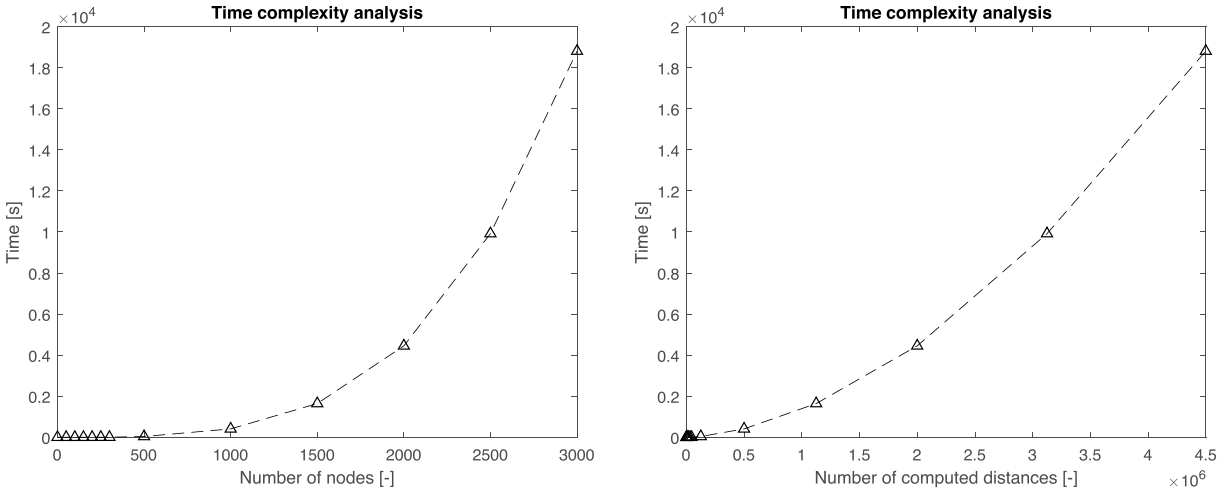

Fig. 5. Average computation time in seconds of the net flow RSP dissimilarity matrix over the 10 LFR graphs for each different network size, in terms of number of nodes (left) and number of computed dissimilarities (right).

6.3.1. Experimental setup

Baselines. As part of the node clustering experiment, four dissimilarity matrices between nodes as well as five kernels on a graph were used as baselines to assess our nRSP method.

Baseline dissimilarities

- The **Free Energy** (FE) **distance** and the **RSP dissimilarity**, depending on an inverse temperature parameter $\theta = 1/T$. As presented earlier, these methods have been shown to perform well in a node clustering context [44] as well as in semi-supervised node classification tasks.
- The **Logarithmic Forest** (LF) **distance**. Introduced in [7], the LF distance relies on the matrix forest theorem [9] and defines a family of distances interpolating (up to a scaling factor) between the shortest path distance and the resistance distance [31], depending on a parameter α.
- The **Shortest Path** (SP) **distance**. This well-known, standard, distance corresponds to the cost along the least-cost path between two nodes i and j, derived from the cost matrix $\mathbf{C}$.

These dissimilarity matrices were transformed into inner products (kernel matrices) by classical multidimensional scaling (see below).

Baseline kernels on a graph

- The **Neumann kernel** [17,43] (Katz) is defined as $\mathbf{K} = (\mathbf{I} - \alpha\mathbf{A})^{-1} - \mathbf{I}$. The α parameter has to be positive and smaller than the inverse of the spectral radium of $\mathbf{A}$, $\rho(\mathbf{A}) = \max_i(|\lambda_i|)$.
- The **Logarithmic Communicability** (lCom) **kernel**, proposed in [26]. The lCom kernel corresponds to the logarithmic version of the communicability measure [14], computed as $\mathbf{K} = \ln(\exp(t\mathbf{A}))$, $t > 0$, where expm is the matrix exponential and ln the natural elementwise logarithm (see [26] for details).
- The **Sigmoid Commute Time** (SCT) **kernel**. Proposed in [48], the SCT kernel is obtained by applying a sigmoid transform [43] on the commute-time kernel [16]. An alternative version, the **Sigmoid Corrected Commute Time** (SCCT) **kernel**, based on the correction of the commute time suggested in [47], was also used as part of the experiments. For both methods, the parameter α controls the sharpness of the sigmoid.

In addition, the **Modularity matrix Q** (Q) was used as the final baseline, and was computed by $\mathbf{Q} = \mathbf{A} - \frac{\mathbf{d}\mathbf{d}^{\mathrm{T}}}{\mathrm{vol}}$ where $\mathbf{d}$ contains the node degrees and vol is a constant denoting the volume of the graph (see, e.g., [37] and references therein). Recall that modularity is an unsupervised measure of the quality of a partition of the nodes (a set of communities). Here, the kernel k-means is executed directly on matrix $\mathbf{Q}$.

Table 1
Datasets used in our experiments.

Dataset Name	Clusters	Nodes	Edges
Dolphin_2	2	62	159
Dolphin_4	4	62	159
Football	12	115	613
LFR1	3	600	6142
LFR2	6	600	4807
LFR3	6	600	5233
Newsgroup_2_1	2	400	33854
Newsgroup_2_2	2	398	21480
Newsgroup_2_3	2	399	36527
Newsgroup_3_1	3	600	70591
Newsgroup_3_2	3	598	68201
Newsgroup_3_3	3	595	64169
Newsgroup_5_1	5	998	176962
Newsgroup_5_2	5	999	164452
Newsgroup_5_3	5	997	155618
Political books	3	105	441
Zachary	2	34	78

Table 2
Parameter range for the investigated methods.

Algorithm	Parameter values
FE	
RSP	$\theta = (0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 3, 5, 10, 15, 20)$
nRSP	
LF	$\alpha = (0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 3, 5, 10, 15, 20)$
Katz	$\alpha = (0.05, 0.10, \ldots, 0.95) \times (\rho(\mathbf{A}))^{-1}$
lCom	$t = (0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10)$
SCT	
SCCT	$\alpha = (5, 10, 15, \ldots, 50)$

Datasets. A collection of 17 datasets was investigated for the experimental comparisons of the dissimilarity measures. For each dataset, costs were computed as the reciprocal of affinities, $c_{ij} = 1/a_{ij}$, as in electric networks. The collection included Zachary's karate club [50], the Dolphin datasets [35], the Football dataset [19], the Political books,[15] three LFR benchmarks [33] and nine Newsgroup datasets processed from the original Newsgroup data[16] (see [48] for details). The list of datasets along with their main characteristics are presented in Table 1.

Evaluation metrics. Each partition provided by an investigated clustering technique was assessed by comparing it with the "observed partition" of the dataset. Two criteria were used to evaluate the similarity between both partitions.

- The **Normalized Mutual Information (NMI)** [45] between two partitions $\mathcal{U}$ and $\mathcal{V}$ was computed by dividing the mutual information [10] between the two partitions by the average of the respective entropy of $\mathcal{U}$ and $\mathcal{V}$.
- The **Adjusted Rand Index (ARI)** [25] is an extension of the Rand Index (RI), which measures the degree of matching between two partitions. The RI has the drawback of not showing a constant expected value when working with random partitions. In contrast, the ARI has an expected value of 0 and a maximum value of 1.

6.3.2. Experimental methodology

The experiments relied on the kernel k-means introduced in [48]. For the dissimilarities, the experimental methodology was similar to the one used in [44]. For each given dataset, the dissimilarity matrix **D** obtained by the different methods providing dissimilarities[17] was transformed into a kernel **K** (a inner product matrix) using classical multidimensional scaling [17]. If the resulting kernel was not positive semi-definite, we simply set the negative eigenvalues to zero when computing the kernel.

The kernel k-means was run 30 times (trials) on **K** with different initializations. The NMI and ARI were then computed for the partition to maximize the modularity among these 30 trials. This operation was repeated 30 times (leading to a total of 900 runs of the k-means) to obtain the average modularity, and the NMI and ARI scores over these 30 repetitions for a given method (dissimilarity/similarity matrix), with a given value of its parameter (for instance, θ in the case of methods based on

[15] Collected by V. Krebs and labelled by M. Newman, this database is not, to the best of our knowledge, published but it is available for download at http://www-personal.umich.edu/mejn/netdata/.
[16] Available from the UCI Machine Learning Repository.
[17] For methods that directly provide a kernel, the obtained kernel matrix was directly used in the kernel k-means.

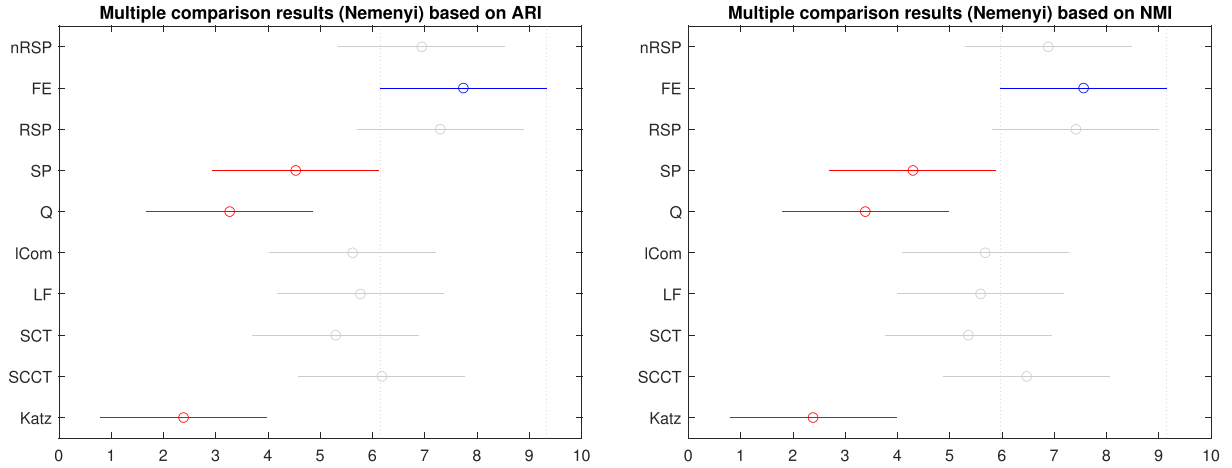

Fig. 6. Mean ranks and 95% Nemenyi confidence intervals of the tested methods across the 17 datasets for the ARI (left) and the NMI (right) measures. The larger the rank, the better.

Table 3
The *p*-values provided by a pairwise Wilcoxon signed-rank test, for the NMI in the upper right triangle and the ARI in the lower left.

Method	nRSP	FE	RSP	SP	Q	lCom	LF	SCT	SCCT	Katz
nRSP	-	0.389	0.497	**0.017**	**0.004**	0.241	0.188	0.151	0.561	**0.001**
FE	0.277	-	0.626	**0.015**	**0.002**	0.068	**0.025**	**0.015**	0.127	**0.001**
RSP	0.588	0.153	-	**0.011**	**0.002**	**0.025**	**0.020**	**0.026**	0.194	**0.001**
SP	**0.019**	**0.015**	**0.031**	-	0.246	**0.028**	0.093	0.062	**0.011**	**0.004**
Q	**0.002**	**0.001**	**0.001**	0.055	-	**0.006**	**0.005**	**0.004**	**0.002**	0.076
lCom	0.241	**0.042**	0.078	0.068	**0.004**	-	0.855	1.000	0.241	**0.001**
LF	0.151	**0.011**	0.078	0.062	**0.003**	0.952	-	0.934	0.217	**0.001**
SCT	0.055	**0.012**	**0.030**	0.163	**0.004**	0.359	0.359	-	**0.009**	**0.001**
SCCT	0.252	**0.048**	0.153	**0.044**	**0.002**	0.808	0.426	**0.004**	-	**0.001**
Katz	**0.001**	**0.001**	**0.001**	**0.002**	0.062	**0.001**	**0.001**	**0.001**	**0.001**	-

RSP), on a specific dataset. Finally, the reported NMI and ARI scores for each method and dataset were the average (over the 30 repetitions) for the parameter value showing the largest modularity. Thus, modularity (instead of a separate dataset in [44]) was used as a metric to tune the parameters of the algorithms. These parameters were the θ for the nRSP, the FE distance and the RSP dissimilarity, the α for the LF distance and Katz, the t for the lCom kernel, and the α for the sigmoid transform of the SCT and SCCT kernels. The values tested for the parameter are listed in Table 2.

6.3.3. Experimental results

The different methods were globally assessed using the same method as in [44] based on a non-parametric Friedman-Nemenyi test [11]. Additionally, a non-parametric Wilcoxon signed-rank test was performed pairwise to measure the significance of the difference in the algorithms' performance.

The results of the Friedman-Nemenyi test are summarized in Fig. 6. Additionally, Table 3 contains the pairwise *p*-values for the Wilcoxon signed-rank test performed on the results obtained from the 17 datasets. More specifically, the upper-right side of the diagonal of the matrix contains the *p*-values when considering NMI and the lower-left side contains the *p*-values when using the ARI.

We can observe on the Nemenyi plot that the three leading methods appear to be the FE, the RSP and the nRSP. More specifically, the FE was the best method on the investigated datasets and in this setup. Based on the Wilcoxon test using ARI, the FE performed significantly better than all the other methods, at a $\alpha = 0.05$ level, except for the RSP and nRSP. However, note that, for the NMI score, the difference between the FE and the lCom and between the FE and the SCCT were not significant.

The introduced method (nRSP) obtained results comparable to the RSP and the FE for both the ARI and the NMI measures. None of these differences were significant after performing a Wilcoxon signed rank test ($\alpha = 0.05$). Although not the best overall, the introduced nRSP method proved to be competitive with respect to the FE and the RSP, which, in turn, performed best in a similar but more extensive node clustering comparison [44]. It can also be observed that the nRSP and the standard RSP performed very similarly, which was somewhat expected because both dissimilarities are based on the same framework.

7. Conclusion

In this work, two extensions of the RSP formalism were developed. The first extension introduces an algorithm for computing the expected net costs between all pairs of nodes by considering the *net flows* between nodes instead of the raw flows. This quantity is called the net flow RSP dissimilarity; it quantifies the level of accessibility (proximity and ease of access) between nodes [9] and serves as a dissimilarity measure. The second extension deals with *capacity constraints* on edges for both raw and net flows within the RSP formalism. An algorithm solving the constrained problem has been developed.

These contributions extend the scope of the RSP formalism, which essentially defines a model of movement, or spread, through the network. Indeed, as already discussed in the introduction, many of the traditional models are based on two common paradigms about the transfer of information, or more generally the movement, occurring in the network: an optimal behavior based on least-cost paths and a random behavior based on a random walk on the graph. In contrast to these standard models, the RSP, and other families of distances (see the related work in the introductory Section 1), interpolate between a pure random walk on the graph and an optimal behavior based on shortest paths. They depend on a parameter that allows the amount of randomness of the trajectories to be monitored. This acknowledges the fact that in many practical cases the (random) walker on the graph is neither completely rational nor completely stochastic.

Another peculiarity of the RSP model is that it adopts a statistical physics framework that considers the system of all paths (or walks) connecting pairs of nodes in the network. The path-based formalism assigns a Gibbs-Boltzmann probability distribution on the set of paths by minimizing expected cost with relative entropy regularization weighted by temperature – the free energy objective function.

Different quantities capturing the degree of relative accessibility between the nodes can then be derived within this formalism, showing different properties depending on the temperature controlling randomness. Another interesting feature is that most quantities of interest can be computed in closed form by using standard matrix operations.

Experimental comparisons on clustering tasks demonstrated that the net flow RSP dissimilarity is competitive in comparison with other state-of-the-art baseline methods. This indicates that the model is able to capture the cluster structure of networks in an accurate way on the investigated datasets. Indeed, based on the same framework, the net flow RSP obtained results comparable to the simple RSP and the free energy dissimilarities.

In conclusion, the contributions of this paper should enlarge the range of possible applications of the RSP formalism. Indeed, many problems related to the spread of information in a network involve capacity constraints on edges. Moreover, in many real cases, it can be argued that a model based on unidirectional net flows is more realistic than raw flows going back and forth.

Concerning further work, we are also interested in applying the proposed models to operations research problems. Indeed, it would certainly be interesting to compare the RSP solution (with entropy regularization) to more standard algorithms for solving minimum cost flow problems and minimum cost flow with capacity constraints problems [1,32].

In addition, more sophisticated optimization techniques, going beyond the simple gradient technique used for solving the capacity-constrained RSP problem in Section 4, should be investigated. Still another interesting idea would be to try to reformulate the optimization problem in terms of expected net costs instead of expected costs in Eq. (28).

We also plan to explore a route that could improve the scalability of the proposed algorithms on sparse graphs. Following [12], the idea would be to extract a DAG from the original *s-t* graph by, for instance, performing a breath-first traversal or computing the electric current flow between the source node *s* and the target node *t*. Recall that we saw in SubSection 3.3 that the electric current defines a DAG. It should, therefore, be possible to compute efficiently the optimal policy (and, consequently, the directed flows) on this DAG by using the Bellman-Ford-like expression that computes the free energy directed distance (see [17]). It would also be interesting to explore the introduction of capacity constraints on a DAG, as already discussed in SubSection 5.2.

CRediT authorship contribution statement

Sylvain Courtain: Software, Formal analysis, Investigation, Writing - original draft, Writing - review & editing, Supervision. **Pierre Leleux:** Software, Formal analysis, Investigation, Writing - original draft, Writing - review & editing. **Ilkka Kivimäki:** Conceptualization, Methodology, Software, Writing - original draft, Funding acquisition. **Guillaume Guex:** Conceptualization, Methodology. **Marco Saerens:** Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially supported by the Immediate and the Brufence projects funded by InnovIris (Brussels region), as well as former projects funded by the Walloon region, Belgium. We thank these institutions for giving us the opportunity to

conduct both fundamental and applied research. Moreover, we thank Professor Bernard Fortz (Université Libre de Bruxelles) for his remarks on the RSP formalism. Finally, we are also grateful to the anonymous reviewers whose suggestions have helped us significantly to improve the manuscript. Marco Saerens is also 'Collaborateur scientifique' at the IRIDIA lab, Université Libre de Bruxelles (ULB).

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.
[2] T. Akamatsu, Cyclic flows, Markov process and stochastic traffic assignment, Transp. Res. B 30 (1996) 369–386.
[3] M. Alamgir, U. von Luxburg, Phase transition in the family of p-resistances, in: Advances in Neural Information Processing Systems 24 Proceedings of the NIPS 2011 Conference, MIT Press, 2011, pp. 379–387.
[4] F. Bavaud, G. Guex, Interpolating between random walks and shortest paths: a path functional approach, in: K. Aberer, A. Flache, W. Jager, L. Liu, J. Tang, C. Guéret (Eds.), Proceedings of the 4th International Conference on Social Informatics (SocInfo '12), Springer, 2012, pp. 68–81.
[5] M. Bell, Stochastic user equilibrium assignment in networks with queues, Transp. Res. Part B: Methodol. 29 (1995) 125–137.
[6] U. Brandes, D. Fleischer, Centrality measures based on current flow, in: Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS '05), 2005, pp. 533–544.
[7] P. Chebotarev, A class of graph-geodetic distances generalizing the shortest-path and the resistance distances, Discrete Appl. Math. 159 (2011) 295–302.
[8] P. Chebotarev, The walk distances in graphs, Discrete Appl. Math. 160 (2012) 1484–1500.
[9] P. Chebotarev, E. Shamis, The matrix-forest theorem and measuring relations in small social groups, Autom. Remote Control 58 (1997) 1505–1514.
[10] T.M. Cover, J.A. Thomas, Elements of Information Theory, second ed., Wiley, 2006.
[11] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[12] R. Dial, A probabilistic multipath assignment model that obviates path enumeration, Transp. Res. 5 (1971) 83–111.
[13] P.G. Doyle, J.L. Snell, Random Walks and Electric Networks, The Mathematical Association of America, 1984.
[14] E. Estrada, N. Hatano, Communicability in complex networks, Phys. Rev. E 77 (2008) 036111.
[15] P. Ferrari, Road pricing and network equilibrium, Transp. Res. Part B: Methodol. 29 (1995) 357–372.
[16] F. Fouss, A. Pirotte, J.M. Renders, M. Saerens, Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation, IEEE Trans. Knowl. Data Eng. 19 (2007) 355–369.
[17] F. Fouss, M. Saerens, M. Shimbo, Algorithms and Models for Network Data and Link Analysis, Cambridge University Press, 2016.
[18] L.C. Freeman, A set of measures of centrality based on betweenness, Sociometry 40 (1977) 35–41.
[19] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. U.S.A. 99 (2002) 7821–7826.
[20] I. Griva, S. Nash, A. Sofer, Linear and Nonlinear Optimization, 2nd ed.,., SIAM, 2008.
[21] G. Guex, Interpolating between random walks and optimal transportation routes: flow with multiple sources and targets, Physica A 450 (2016) 264–277.
[22] G. Guex, F. Bavaud, Flow-based dissimilarities: shortest path, commute time, max-flow and free energy, in: B. Lausen, S. Krolak-Schwerdt, M. Bohmer (Eds.), Data Science, Learning by Latent Structures, and Knowledge Discovery, Springer, 2015, pp. 101–111.
[23] D.W. Hearn, S. Lawphongpanich, A dual ascent algorithm for traffic assignment problems, Transp. Res. Part B: Methodol. 24 (1990) 423–430.
[24] M. Herbster, G. Lever, Predicting the labelling of a graph via minimum p-seminorm interpolation, in: Proceedings of the 22nd Conference on Learning Theory (COLT '09), 2009, pp. 18–21.
[25] L. Hubert, P. Arabie, Comparing partitions, J. Classif. 2 (1985) 193–218.
[26] V. Ivashkin, P. Chebotarev, Do logarithmic proximity measures outperform plain ones in graph clustering?, in: Proceedings of the International Conference on Network Analysis (NET 2016), Springer International Publishing, 2016, pp 87–105.
[27] E.T. Jaynes, Information theory and statistical mechanics, Phys. Rev. 106 (1957) 620–630.
[28] T. Jebara, Machine Learning: Discriminative and Generative, Kluwer Academic Publishers, 2004.
[29] I. Kivimäki, B. Lebichot, J. Saramäki, M. Saerens, Two betweenness centrality measures based on randomized shortest paths, Scientific Rep. J. 6 (2016) srep19668.
[30] I. Kivimäki, M. Shimbo, M. Saerens, Developments in the theory of randomized shortest paths with a comparison of graph node distances, Physica A 393 (2014) 600–616.
[31] D.J. Klein, M. Randic, Resistance distance, J. Math. Chem. 12 (1993) 81–95.
[32] B. Korte, J. Vygen, Combinatorial Optimization. Theory and Algorithms, sixth ed., Springer, 2018.
[33] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (2008) 046110.
[34] Y. Li, Z.L. Zhang, D. Boley, From shortest-path to all-path: the routing continuum theory and its applications, IEEE Trans. Parallel Distrib. Syst. 25 (2013) 1745–1755.
[35] D. Lusseau, The emergent properties of a dolphin social network, Proc. R. Soc. Lond. B Biol. Sci. 270 (2003) S186–S188.
[36] M.E.J. Newman, A measure of betweenness centrality based on random walks, Soc. Netw. 27 (2005) 39–54.
[37] M.E.J. Newman, Networks: An Introduction, second ed., Oxford University Press, 2018.
[38] C. Nguyen, H. Mamitsuka, New resistance distances with global information on large graphs, in: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS '16), 2016, pp. 639–647.
[39] W.L. Price, Graphs and Networks: An Introduction, Butterworths, London, 1971.
[40] R. Rockafellar, The multiplier method of Hestenes and Powell applied to convex programming, J. Optim. Theory Appl. 12 (1973) 555–562.
[41] S. Ryu, A. Chen, X. Xu, K. Choi, A dual approach for solving the combined distribution and assignment problem with link capacity constraints, Netw. Spatial Econ. 14 (2014) 245–270.
[42] M. Saerens, Y. Achbany, F. Fouss, L. Yen, Randomized shortest-path problems: two related models, Neural Comput. 21 (2009) 2363–2404.
[43] B. Schölkopf, A. Smola, Learning with Kernels, MIT Press, 2002.
[44] F. Sommer, F. Fouss, M. Saerens, Comparison of graph node distances on clustering tasks, in: Proceedings of the International Conference on Artificial Neural Networks (ICANN 2016). Lecture Notes in Computer Science, Springer, 2016, pp. 192–201.
[45] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2002) 583–617.
[46] E. Todorov, Linearly-solvable Markov decision problems, in: Advances in Neural Information Processing Systems 19 (NIPS 2006), MIT Press, 2007, pp. 1369–1375.
[47] U. von Luxburg, A. Radl, M. Hein, Getting lost in space: large sample analysis of the commute distance, in: Advances in Neural Information Processing Systems 23: Proceedings of the NIPS '10 Conference, 2010, pp. 2622–2630.
[48] L. Yen, F. Fouss, C. Decaestecker, P. Francq, M. Saerens, Graph nodes clustering with the sigmoid commute-time kernel: a comparative study, Data Knowl. Eng. 68 (2009) 338–361.
[49] L. Yen, A. Mantrach, M. Shimbo, M. Saerens, A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008), 2008, pp. 785–793.
[50] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (1977) 452–473.