Applied Network Science

RESEARCH Open Access

Making communities show respect for order

Vaiva Vasiliauskaite* and Tim S. Evans

*Correspondence:
VV1615@imperial.ac.uk
Centre for Complexity Science and Theoretical Physics Group, Imperial College London, SW7 2AZ London, UK

Abstract

In this work we give a community detection algorithm in which the communities both respects the intrinsic order of a directed acyclic graph and also finds similar nodes. We take inspiration from classic similarity measures of bibliometrics, used to assess how similar two publications are, based on their relative citation patterns. We study the algorithm's performance and antichain properties in artificial models and in real networks, such as citation graphs and food webs. We show how well this partitioning algorithm distinguishes and groups together nodes of the same origin (in a citation network, the origin is a topic or a research field). We make the comparison between our partitioning algorithm and standard hierarchical layering tools as well as community detection methods. We show that our algorithm produces different communities from standard layering algorithms.

Keywords: Directed acyclic graph, Antichain, Community detection, Citation network, Food web

Introduction

Nodes in networks have many natural orders. Every centrality measure allows us to say if one node has a higher centrality value than another. However, some systems can have important constraint leading to a characteristic order in a network; examples include the publication dates of papers in a citation network, dependency of packages in computer software, and predator-prey relationships in a food web. If edges respect this order, they exist only if they link a high value node to a lower value node, from an earlier paper to a later paper, then edges are directed and there can be no cycles — a Directed Acyclic Graph (DAG).

A link between two nodes in a DAG must always encode the *order* of the pair. It is the converse which is important here: certain links can not exist because of the inherent order. This means that two nodes can be very similar but they can not be connected while in standard network analysis such an edge would be assumed to exist most of the time in a DAG. For example, two papers can be produced independently at the same time with similar new results yet by definition they can not cite each other. The development of the relativistic model for the Higgs mechanism is a good illustration of this as it is attributed to three independent groups: Brout and Englert (August 1964), Higgs (October 1964), and Guralnik, Hagen and Kibble (November 1964). Only the last of these three papers cites the two earlier publications but there is no citation between the first two.

So in order to understand DAGs, we need to adapt our standard network analysis tools to make them respect the order implicit in these common network topologies. In this paper we will focus on the important topic of community detection in networks, clustering in the language of data science, in which the aim is to find sets of similar nodes (Fortunato 2010; Jain et al. 1999). Since most network methods assume that a link between nodes indicates similarity, network communities translate the similarity of nodes into the requirement that communities are subgraphs where there are more links within the community than there are to the rest of the network (Fortunato 2010). This is justified when links indicate node similarity that is not necessarily true for DAGs so we need to find new ways to define clusters otherwise the order in a DAG may obscure some of the natural clustering.

To see how we will find communities in a DAG, we start by noting that one expression of the order in a DAG is that there is a natural hierarchy in the system. Two computer packages which fill a very similar role will not depend on each other but they will draw on similar "lower level" packages reflecting a hierarchy and an order in packages. For instance, the python network package `networkx` is a prerequisite for two python community detection algorithms, `python-louvain` and `demon`. In the Florida-bay foodweb we study in "Florida bay food web" section, pinfish, parrotfish and manatee are grouped together because they feed on similar species, such as detritivorous polychaetes, sponges, bivalves. So in a DAG a natural property of any nodes at the same level in the hierarchy is that they are *not* connected, directly by an edge or even indirectly via a longer path. So, in the context of a network with an order, with a hierarchy, it is very natural to cluster nodes which are not connected by any path, and these are called ANTICHAINS, see Fig. 1. So in our approach, in order to respect the order inherent in a DAG we will create communities which are antichains.

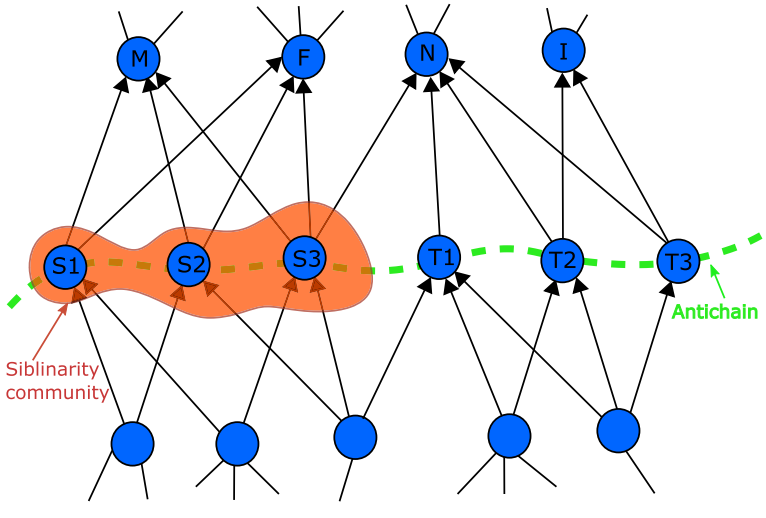

Fig. 1 An antichain is a subset of nodes in the graph, such that none of the nodes are pairwise connected with edges or paths. In this illustration, an antichain is composed of nodes $\{S_1, S_2, S_3, T_1, T_2, T_3\}$ (the green dotted line is drawn as a guide of eye). Although in the same antichain, not all nodes share many neighbours. Nodes $\{S_1, S_2, S_3\}$ share many successors (they are all connected to nodes *M*, *F*), so we join them in a Siblinarity community

The next problem is that there are many possible antichains and each node can be in many different antichains. Also nodes in useful communities will need to be similar in some sense. In a set of computer packages, we don't want to cluster a package on games with one on networks, rather we want to collect all the different network packages together in one community. Likewise, we don't want to cluster species at the same level in food webs if they are from completely different environments. So we will aim to find the antichains which contain nodes which are similar by some appropriate measure. We take our inspiration from classic measures used to assess the similarity of two documents from their citation network alone (for example see Boyack and Klavans (2010) for more recent application). In *bibliographic coupling* the similarity of two documents is measured using the overlap of their bibliographies (Kessler 1963; Martyn 1964). The *co-citation* similarity of two documents (Small 1973; Small and Griffith 1974) uses the overlap in the citing documents. So by looking into similarities of neighbourhoods of two nodes, we can say something about how similar they are themselves. Though defined in terms of a citation network, these similarity measures are widely used in network analysis, for example see Satuluri and Parthasarathy (2011).

Our aim in this paper is to produce a method to find communities of similar nodes which take account of the sense of order in a DAG, the hierarchy implicit in a DAG. We partition the nodes of a DAG into antichains which have large neighbourhood overlaps. We will also look at the properties of our DAG clusters and we will compare them against properties of other types of DAG layering and clustering methods. Lastly we will investigate the potential uses of this approach to study citation networks and food webs.

Our method is related to two different types of network algorithms: community detection and DAG layering. Community detection does not include a notion of "layers" but aims to find a collection of nodes with strong intra-community links. In this sense a community of similar nodes is being defined by the network topology on mesoscopic scales, not just on the local scale of nearest neighbours. Since an edge connecting two nodes is normally used as the strongest indicator of a relationship, antichains seem at first to be the exact opposite of traditional network communities. As a consequence, a conventional community detection method applied to a DAG (especially if the edge direction is ignored) can be composed of nodes from many different layers. DAG layering, on the other hand, does not have a notion of similarity but typically it aims to minimise the number of layers used so maximising the size of the layers. Such layers are composed of hierarchically equivalent nodes but a layer does not represent a community of similar nodes. Our algorithm bridges this gap and finds layer decomposition of a DAG with a notion of similarity.

Community detection is an active research area in network science. This field is wide and many different approaches to finding groups of similar nodes were proposed. An extensive discussion of the main approaches to community detection in networks is given in Fortunato (2010). Some works have studied the stability and performance of conventional community detection algorithms in DAGs such as Leicht et al. (2007); II et al. (2010). There is also some work on specialised approaches to community detection in directed acyclic graphs, for instance (Speidel et al. 2015), in addition to clustering algorithms developed for analysis of task scheduling graphs (which are also DAGs), for example see Gerasoulis and Yang (1992) and references therein.

Graph layering is another topic related to our work. It is used for drawing hierarchical graphs, such as trees and DAGs. Layering is a problem of partitioning a DAG into layers, such that edges in the visualisation always point in one direction on the page, thus it is a form of partitioning a DAG into antichains. As part of Sugyiama graph drawing algorithm, various layering techniques are used to remove edge overlap as much as possible, as well as minimise the number of edges that span many layers (Sugiyama et al. 1981). Perhaps the simplest way to partition a DAG into layers (or antichains) is by using the so-called longest path algorithm, which yields the smallest number of layers (Mirsky 1971). We will discuss this algorithm in more detail in "Height and depth antichain partitions" section and utilise partitions, obtained using the longest paths, to evaluate the performance of our proposed method. More sophisticated graph layering algorithms limit the number of layers in the graph, so the size of most layers is large, for instance to minimise the number of layers with crossing edges (Tang and Hu 2013; Gansner et al. 1993; Nikolov and Tarassov 2006) or to ensure the maximum width of any of the layers (Healy and Nikolov 2002; 2013).

Our aim here is to find a clustering algorithm which both respects the topology and ordering of a DAG while using that same topology to ensure the clusters contain only similar nodes. We start by discussing the methods we use in "Methods" section. We look at the relevant properties of DAGs in "Basic DAG properties" section. We review a standard way to partition a DAG into antichains by considering the heights and depths of nodes in "Height and depth antichain partitions" section. We then describe the algorithm we used to find our siblinarity communities in "Siblinarity antichain partitions" and "Optimisation" sections. In "Data and models" section we describe the models and data used to test our algorithm. The results of our analysis are given in "Results" section where we study the properties of siblinarity-based communities, comparing them to alternative methods and exploring variations of our algorithm. We conclude with a brief overview of our work in "Discussion" section. Further details and additional examples are contained in Additional file 1. We provide data for many of our examples online (Vasiliauskaite and Evans) while the source of data for any remaining examples is given in our bibliography.

Methods

Basic DAG properties

Suppose we have a DAG (directed acyclic graph) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the set of nodes and edges in the graph are $\mathcal{V}$ and $\mathcal{E}$ respectively. We will use $N = |\mathcal{V}|$ for the number of nodes in the DAG. We will denote an edge from node n to node m as (n, m). Edges can be weighted, such that a larger weight of an edge represents a stronger connection.

A PATH in any directed graph, including a DAG, is a sequence of nodes in which consecutive nodes are linked by an edge in the correct direction (Newman 2010). The length of the path is the number of edges in that path, one less than the number of nodes in the path. This allows us to define an ANTICHAIN $\mathcal{A}$ as a set of nodes in which there is no path between any of the nodes in the antichain. We will later use the antichains in both the original DAG and in directed graphs we shall derive.

The lack of cycles in a DAG leads to a special property, namely that every DAG is linked to a unique PARTIAL ORDER on the set of nodes. That is we can say $n \prec m$ if there is a path *from n to m*. Note there need not be any sense of order between two nodes, i.e. when there is no path between the nodes, and it is this partial order which we aim to respect in our community detection. This mathematical property of the DAG's partial

order is often seen in key properties of a real data set. For instance, in a perfect citation network, the publication dates of documents express the partial order. That is if $n \prec m$ for two documents n and m in the citation network then we know that document n must be published before document m where we take the edges in the DAG to run *from* the older document being cited *to* the bibliographies of newer documents. The partial order in this case reflects the "arrow of time" inherent in the publication of documents. In computer libraries, each package tends to require less specialised packages. For a food web, the mathematical partial order may reflect the tendency of larger animals to eat only smaller ones.

In general, this sense of order in a DAG is often seen as a natural hierarchy. For a library of computer packages, we might talk about a low-level package. In a food web, we have "top-level predators". Typically, elements at the same level in the hierarchy are going to have no connections between them since they do the same job at this level, they are alternatives to each other. That is there is a good chance that such elements with no connections are similar and it is the order in the DAG, the level of the hierarchy is telling us about this similarity. In terms of the formal properties of the DAG, these elements form an antichain.

For example, predators of a similar size will not generally eat each other and so are often found at the same level in a food web. Yet such animals could be similar in the sense that they compete for similar (smaller) prey. In the food web example we will consider in "Florida bay food web" section, sharks, tarpon and grouper will be found in the same siblinarity community, because they share many common prey, such as killifish and crabs.

Height and depth antichain partitions

So the approach we take is to look for communities of similar nodes in a DAG which are antichains, denoted as $\mathcal{A}$. In that way, none of the nodes in one of our antichain communities are ordered before or after any of the others in the same community. There is no direct connection between nodes in our antichain communities by definition so these are very different communities from those produced in general networks such as discussed in Fortunato (2010).

For simplicity, we will also restrict ourselves to the case where each community is an antichain, and the set of all communities is a partition $\mathfrak{A}$, which we call an ANTICHAIN PARTITION. That is every node is in exactly one element (one community) of our antichain partition $\mathfrak{A}$. This is hard or non-fuzzy clustering in the language of data science.

Our first two examples of antichain partitions are well known features of any DAG. We start by noting that another property of any DAG is that we can always assign a HEIGHT to every node. The height $h(n)$ of node n is the length of the longest path to n from any node with zero in-degree. It is straightforward to see that each node on a path must have different heights with the height increasing as you move along the path. Conversely, nodes of the same height cannot have any path between them so that nodes of the same height form an antichain. Thus we can define the HEIGHT PARTITION $\mathfrak{A}^{(\mathrm{h})}$ to be the set of antichains $\{\mathcal{A}_a^{(\mathrm{h})}\}$, each of which contains all the nodes of a give height

$$\mathfrak{A}^{(\mathrm{h})} = \{\mathcal{A}_a^{(\mathrm{h})}\}, \quad \mathcal{A}_a^{(\mathrm{h})} = \{n | n \in \mathcal{V}, h(n) = a\}. \tag{1}$$

Similarly, we can define the DEPTH $d(n)$ of a node n to be the length of the longest path from n to a node with zero out-degree. Nodes with the same depth are guaranteed to

form an antichain so we can define the DEPTH PARTITION $\mathfrak{A}^{(\mathrm{d})}$ to be partition of the set of nodes by their depth,

$$\mathfrak{A}^{(\mathrm{d})} = \{\mathcal{A}_a^{(\mathrm{d})}\}, \quad \mathcal{A}_a^{(\mathrm{d})} = \{n | n \in \mathcal{V},\, d(n) = a\}. \tag{2}$$

Note that these height and depth antichains are examples of maximal antichains, each antichain is not a proper subset of any other antichain including those not in our partitions. Many of the layering algorithms are designed to produce the same or similar numbers of antichains, so again those are often maximal antichains or something close to that. For this reason, the height and depth antichains are good representatives of the type of antichain produced by traditional layering algorithms, so they will be used to illustrate how our siblinarity antichain partitions are very different.

Siblinarity antichain partitions

Often the *elements* of either the height or depth antichain partitions (an element here is one community, one antichain in a partition) provide one definition of a level in the hierarchy. However, while these antichains respect the order of the DAG, we want to highlight much smaller groups which contain nodes which are much more similar than just the similarity imposed by the order as encoded by an antichain. All the nodes at one height, or those at one depth need not be very similar in general.

To add similarity to the hierarchy constraint encoded through our restriction to antichains, we can take inspiration from classic similarity measures used in bibliometrics. In that context, one way to assess the similarity of two publications is to look at overlap of their neighbours in the citation network. The more two publications share the same neighbours, the more similar they are said to be. The size of an intersection between two paper's bibliographies is called BIBLIOGRAPHIC COUPLING (Kessler 1963; Martyn 1964), whereas the size of an overlap between articles that they were referenced by is called CO-CITATIONS (Small 1973; Small and Griffith 1974). So by looking into similarities of neighbourhoods of two nodes, we can say something about how similar they are themselves.

A family tree provides a good example where people are the nodes and edges are from a parent to a child and so point forward in time in terms of birth date. There will be many people in a single generation but, by definition, none will be a parent or a child of any other person in the same generation so each generation forms an antichain. Generations are layers in a natural hierarchy for this DAG. However, almost all people in one generation will have little genetic biological relationship to each other so this large antichain, a single generation, may not be very interesting in many problems. However, if we also look for clusters of people within this generation, people who have common predecessors and so share one or two parents, then these smaller communities may be of more interest. Using such a predecessor similarity measure on top of an antichain constraint would mean a community detection method would be producing communities of siblings.

These ideas of antichain and neighbour similarity are encoded in a function which measures the quality of a given partition $\mathfrak{A}$ of our DAG into antichains, $\mathcal{A}$. Motivated by the family tree example, we call our function SIBLINARITY $S(\mathfrak{A})$ and we define it to be

$$S(\mathfrak{A}) = \sum_{\mathcal{A}\in\mathfrak{A}} \sum_{n\in\mathcal{A}} \sum_{m\in\mathcal{A}\setminus n} (\mathrm{sim}(n,m) - \mathrm{sim}_{\mathrm{null}}(n,m)) \,. \tag{3}$$

Here the first term $\mathrm{sim}(n,m)$ is some measure of the similarity of two nodes n and m. The second term, $\mathrm{sim}_{\mathrm{null}}(n,m)$, is the expected value of similarity of these two nodes in some suitable null model. There is a lot of freedom in choosing a null model but in general it is some randomised version of the DAG. As m and n are in the same antichain $\mathcal{A}$ there is no path between nodes contributing to siblinarity. We have excluded the case $m = n$ so any node in a community by itself contributes zero and $S(\mathfrak{A}) = 0$ for the case where every community is a single node. Including the $m = n$ terms only adds an irrelevant overall constant.

Any similarity measure could be used but a logical choice for the similarity function in our context is the number of neighbours that n and m have in common, so we will use $\mathrm{sim}(n,m) = |\mathcal{N}(n) \cap \mathcal{N}(m)|$ where $\mathcal{N}(n)$ is the neighbourhood of node n. Of course, we could use any other similarity measure between two sets, such as Jaccard index (Jaccard 1912) or cosine similarity. However, we chose to use the neighbourhood overlap, because this similarity measure is analogous to the similarity, assumed in the classical modularity (Newman 2006) and has been shown useful in studying modular structures in multiple types of networks.

There are two obvious choices for this neighbourhood: one in terms of its predecessors $\mathcal{N}^{(\mathrm{pre})}(n)$, as used in the family tree example above, and another in terms of the successors, $\mathcal{N}^{(\mathrm{suc})}(n)$. More formally

$$\mathcal{N}^{(\mathrm{pre})}(n) = \{m|(m,n) \in \mathcal{E}\}\,, \qquad \mathcal{N}^{(\mathrm{suc})}(n) = \{m|(n,m) \in \mathcal{E}\}\,. \tag{4}$$

We could also use both, and use $\mathcal{N}^{(\mathrm{both})}(n) = \mathcal{N}^{(\mathrm{pre})}(n) \cup \mathcal{N}^{(\mathrm{suc})}(n)$.

It is useful to express this in a matrix form as follows (Saturluri and Parthasarathy 2011)

$$S(\mathfrak{A}) = \sum_{\mathcal{A}\in\mathfrak{A}} \sum_{n\in\mathcal{A}} \sum_{m\in\mathcal{A}\setminus n} \left(\tilde{A}_{nm} - \frac{\kappa_n \kappa_m}{W}\right), \tag{5}$$

$$\kappa_n := \sum_m \tilde{A}_{nm}\,, \quad W = \sum_{n,m} \tilde{A}_{nm}\,.$$

Here the adjacency matrix $\mathbf{A}$ for our DAG is defined so that A_{nm} is the weight of the edge from n to m. The neighbourhood overlap is captured by the matrix $\tilde{\mathbf{A}}$ which is the product of the adjacency matrix $\mathbf{A}$ of the DAG and its transpose. The $\tilde{\mathbf{A}}$ matrix can be regarded as the adjacency matrix for a derived graph $\tilde{\mathcal{G}}$ which is a directed weighted graph with the same node set as our DAG. In the case where we have an unweighted DAG we define this to be either $\tilde{\mathbf{A}}^{(\mathrm{suc})}$, our successors-based similarity matrix, or $\tilde{\mathbf{A}}^{(\mathrm{pre})}$ is a similarity matrix based on predecessors[1] where

$$\tilde{\mathbf{A}}^{(\mathrm{suc})} = \mathbf{A}.\mathbf{A}^{\mathrm{T}}\,, \quad \tilde{\mathbf{A}}^{(\mathrm{pre})} = \mathbf{A}^{\mathrm{T}}.\mathbf{A}\,. \tag{6}$$

The effective similarity matrix $\tilde{\mathbf{A}}$ can be seen as a similarity matrix of the "second-order" neighbours[2] For instance, the value of an entry $\tilde{A}^{(\mathrm{suc})}_{nm}$ is equal to the number of different walks, consisting of one step forward (with respect to the edge direction) from a node n, followed by one step backward (against the edge direction) such that the end node of this walk is m. Similarly, in $\tilde{A}^{(\mathrm{pre})}_{nm}$, the number of paths which begin with a step against the edge direction, followed by a step with respect to the edge direction, are counted between

[1] Should we choose to use both sets of neighbours then we simply use the sum of these two matrices $\tilde{\mathbf{A}}^{(\mathrm{both})} = \tilde{\mathbf{A}}^{(\mathrm{suc})} + \tilde{\mathbf{A}}^{(\mathrm{pre})}$ (Saturluri and Parthasarathy 2011).

[2] It is worth noting that every node is its own second neighbour so that $\tilde{\mathbf{A}}$ has diagonal entries (self-loops). It is possible to exclude this effect and to replace $\tilde{\mathbf{A}}$ by a non-backtracking form of the adjacency matrix. For instance we could use $\tilde{A}^{(\mathrm{NBT,suc})}_{mn} = \tilde{A}^{(\mathrm{suc})})_{mn} - \mathrm{out}_n \delta_{mn}$ instead of $\tilde{\mathbf{A}}^{(\mathrm{suc})}$. We see no strong reason to do this so we will use our simpler form.

the two nodes n, m. By looking at this higher order structure in our DAG (Xu et al. 2016) we can get round the problem that there is no direct connection between nodes in our antichain communities.

The κ_n is the strength of edges attached to a node n in a graph with similarity matrix $\tilde{\mathbf{A}}$ and W is the total strength of edges in that graph. The second term in (5) also defines the null model we use. This is a configuration model applied to the derived graph $\tilde{\mathcal{G}}$.

The form we use in (5) emulates the definition of modularity (Newman and Girvan 2004), a similarity measure minus expected value of that similarity measure in some null model. The biggest difference between modularity and siblinarity is that we impose the antichain constraint in the communities we study.

This comparison with modularity (see Fortunato (2010) for a review) suggests that we can modify sibllinarity to adjust the typical number of antichains found, the resolution of our method. One simple method is to scale the null model term (Reichardt and Bornholdt 2006) so that

$$S(\mathfrak{A}, \lambda) = \sum_{\mathcal{A}\in\mathfrak{A}} \sum_{n\in\mathcal{A}} \sum_{m\in\mathcal{A}\setminus n} \left(\tilde{A}_{nm} - \lambda \frac{\kappa_n \kappa_m}{W}\right). \tag{7}$$

We will show some examples of this in the Additional file 1. However it is clear that for large λ we expect many small antichain communities. In particular, for $\lambda \gtrsim W$ adding any node in a community by itself to any other antichain will reduce siblinarity so we expect all the antichains to be the trivial case where each antichain has just one node. Conversely, we expect $\lambda = 0$ to produce large antichain communities.

Optimisation

We can use any numerical optimisation scheme to find an antichain partition that gives a value for siblinarity $S(\mathfrak{A})$ of (5) which is close to a maximum value for siblinarity. In this paper we choose to adopt the Louvain approach (Blondel et al. 2008) to community detection in networks. We chose this because it is fast and successful at finding communities in networks and because it proved easy to adapt to our context. In this method we first use a greedy optimisation phase. In this we try moving single nodes into a different antichain community, choosing the community structure which gives the biggest siblinarity value, even if that means leaving the structure unchanged, We sweep through all the nodes many times until there are no more changes in siblinarity. We then initiate the second stage in which we produce a new weighted directed network $\mathcal{H}$ in which each antichain community in the original network is now represented by a single node in the new derived network. Edges are merged to match this new vertex set, keeping the total weight of the graph unchanged. Note that this derived network is not necessarily a DAG, see Additional file 1: Fig. B4 for an example. However, while our discussion has been framed in terms of a DAG, the concept of an antichain is useful in any directed network with few small cycles and all our expressions, e.g. for siblinarity, remain valid. One then repeats the greedy optimisation with this new derived graph, first a greedy step and second a projection onto a smaller derived graph. One can stop the process with any of the derived graphs but it will terminate when there are no pairs of nodes (antichains in the original graph) which can be merged into a new antichain such that siblinarity is increased.

Antichain measures

Once we have found our antichains, we will need to analyse their properties and we will need some additional tools to do this for large examples.

An interesting question to ask is whether nodes in an antichain are similar to other nodes in the antichain. Many traditional methods for measuring the strength of a community are based on the number of edges between members of the community compared with edges to those outside. Such measures fail for our antichain communities where there are no edges between community members. However, in constructing siblinarity, we have used a similarity measure based on the number of common neighbours in a suitable set of neighbours $\mathcal{N}(n)$, such as the set of successors $\mathcal{N}^{(\mathrm{suc})}(n)$ or predecessors $\mathcal{N}^{(\mathrm{pre})}(n)$. Any similarity measure could be used in principle.

So for each antichain community $\mathcal{A}$ we define a similarity matrix $\mathbf{W}(\mathcal{A})$ where

$$W_{nm}(\mathcal{A}) = |\mathcal{N}(n) \cap \mathcal{N}(m)|\,, \qquad n, m \in \mathcal{A} \tag{8}$$

is equal to the number of common neighbours of nodes m and n. There are many ways to use this. We can then use multiple metrics to give insights about how interconnected $\mathcal{A}$ is. For instance, we can define $W(\mathcal{A})$ to be the average weight per node in an antichain $\mathcal{A}$,

$$\frac{W(\mathcal{A})}{|\mathcal{A}|} = \frac{1}{2|\mathcal{A}|} \sum_{n,m\in\mathcal{A}} W_{nm} \tag{9}$$

which reflects on the mean similarity of nodes in the antichain. If this value is small, and the siblinarity of a given antichain,

$$S(\mathcal{A}, \lambda) = \sum_{n\in\mathcal{A}} \sum_{m\in\mathcal{A}\setminus n} \left(\tilde{A}_{nm} - \lambda \frac{\kappa_n \kappa_m}{W}\right) \tag{10}$$

is large, we can expect that the overlap of neighbourhoods of nodes in the antichain is sparse. In "Citation networks" section we will use these metrics as well as summary statistics to study antichains, obtained in a citation network.

Data and models

We used a variety of models and data sets which have natural representations as a DAG in order to compare different community structures found by different methods.

Space-Time lattice model

Our first test model is a simple DAG where the nodes are placed on a square lattice at (t, x) where tL and xL are integers between 0 and $(L-1)$. To place edges between the node at the points of our lattice, we use the Manhattan distance $d(n, m) = |t_n - t_m| + |x_n - x_m|$ for the distance between nodes n and m of coordinates (t_n, x_n) and (t_m, x_m) respectively. We add a directed edge from n to m with probability $p(n, m)$ where

$$p(n,m) = \begin{cases} 0 & \text{if } t_n \geq t_m \\ 1 - \frac{d(n,m)}{D} & \text{if } t_n < t_m. \end{cases} \tag{11}$$

The edges from n to m only exist if $t_n < t_m$ and it is this arrow-of-time which ensures that we will always have a DAG.

The idea behind this DAG model is that there is a natural hierarchy given by the t-coordinate which guarantees acyclicity. In addition, nodes which are close in their space

and time coordinates will often have large numbers of common neighbours, both successors and predecessors, whereas nodes that are distant in space or time will have few neighbours in common. So in the visualisations we expect to see nodes of the same time coordinate and close in their space coordinates, to be placed in the same antichain community. However, as the links are placed with a stochastic mechanism, we will sometimes see nodes from neighbouring layers grouped together. That is the antichain structure in a given realisation of our model is not a perfect match to the natural layers defined by the time coordinate. This simulates what one finds in real data sets. For instance, two papers written on a similar topic would be represented by nodes with a similar x coordinate in this model. If they were written independently at the same time then they could not be connected but if published at slightly different times it might be possible for the earlier paper to be cited by the later paper.

Price model with subject fields

One of the oldest models of citation networks is the Price model of cumulative advantage (Price 1976) and this defines a DAG which has the fat-tailed (power-law) distribution for the number of papers with a given citation count[3]. Here we consider a modified version of the Price model in which we assign nodes (representing papers) to different 'fields' and we create edges (citations between papers) such that they are usually between papers in the same field. This model is too simple to capture many aspects of real citation network though it does emulate three fundamental aspects: the order of papers imposed by time, the fat-tailed citation count distribution, and the preference of most papers to cite papers within a similar field. For our purposes, all we use here is that this gives a DAG with a well known structure and now with a planted partition. Our expectation is that our siblinarity antichain partitions will tend to cluster papers published around the same time and in the same field.

The model is illustrated in Fig. 2. To define this model, consider a sequence of networks $\mathcal{G}(t)$ where t is a positive integer playing the role of time and which gives us an order to the nodes in our networks. Each graph $\mathcal{G}(t)$ has t nodes. In our notation, the node $u(s)$ is always the node added at step s in the process so it exists in all networks $\mathcal{G}(t)$ provided $0 < s \leq t$. The nodes in these networks are also partitioned into different fields, that is each node $u(t)$ is in field $f(t)$, one of F distinct fields.

To create the next graph in the sequence, $\mathcal{G}(t+1)$, we first add a new node $v(t+1)$ to the vertex set. This new node also is assigned to a field, $f(t+1)$, chosen uniformly at random from the set of F possible fields.

We now add m directed edges to this new node $v(t+1)$ from existing nodes $u(s)$ where $s \leq t$. Following Price, we chose the source nodes s for the new edges from the existing network $\mathcal{G}(t)$ with probability $\Pi(t,s)$ defined to encode "cumulative advantage". That is the higher the current citation count of a paper, the more likely it is to be cited. Since we define our directed edges to run from early to late times, this means our probability $\Pi(t,s)$ is proportional to a linear function of the current out degree of existing nodes, say $\text{out}(t,s)$ for the citation count of node s in the graph $\mathcal{G}(t)$. For simplicity we choose Price's original form $\Pi(t,s) \propto (\text{out}(t,s)+1)$.

[3]The undirected version of this is the Barabási-Albert model, see Newman (2010) for a discussion.

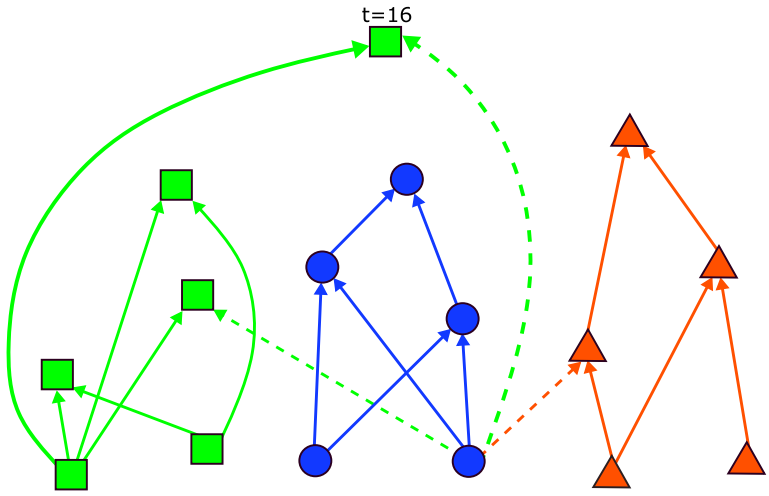

Fig. 2 An illustration of the Price model with two edges added per node ($m = 2$) and three fields ($F = 3$) as indicated by the three types of node. Time increases as you move up the diagram so the vertical ordering of the nodes represents a total order in the DAG. Solid (dashed) lines represent citations between papers of identical (different) fields. A new node labelled $16 = (t + 1)$ in field 0 (green squares) is added an existing network of fifteen nodes. Here we suggest the two edges added to node 16 are to the two nodes of highest degree (the cumulative advantage bias) with one in the same field (continuous edge) and one in a different field (dashed edge)

To impose a modular structure reflecting the preference of papers to cite others papers within the same field, we use a parameter ϕ which is the probability that a new paper $v(t + 1)$ cites another paper $u(s)$ in its own field i.e. $f(s) = f(t)$ but where the connection is made to node $u(s)$ chosen from the papers in the same field using our cumulative advantage $\Pi(t, s)$. With probability $(1 - \phi)$, the source node $u(s)$ of a new edge is chosen from within the papers not in the same field i.e. $f(t + 1) \neq f(s)$.

This leaves us with a stochastic model of three parameters: m, ϕ and the total number of nodes in any network we use. We considered networks with 5,000 nodes and $m = 3$, $m = 5$ edges added to each new node. Some aspects of the model can depend on the initial graph used to start the simulation but this was unimportant for our studies. The model is described in more detail in Additional file 1: Appendix F.

Real world data sets

To test our approach on actual data, we use three examples.

First we use the Florida Bay food-web dataset (Ulanowicz et al. 1998). In this network, the nodes are compartments and edges represent directed carbon exchange in the Florida Bay. There is an edge from i to j if compartment j consumes carbon from compartment i (often, this means species j eats species i). The compartments are mostly organisms but also encompass special nodes such as "input" and "output". We also had group classification labels. Examples groups include "Zooplankton Microfauna" and "Pelagic Fishes" (Benson et al. 2016). The original network consists of 128 nodes and 2106 edges but this contains cycles. We used the breadth first search approach, described in Sun et al. (2017) to recover a DAG, removing 176 edges (less than 9%).

We also used two examples of a citation networks in which documents are nodes and we draw an edge from a newer document when it cites an older document. We used two

datasets: citations between papers in the arXiv High Energy Physics-Theory repository, for papers published between 1992 and 2003 (referred to here as `hep-th`) (KDD cup 2003) and a selection of Cora (Lu and Getoor 2003; Sen et al. 2008) papers, referred below as `cora`. The `hep-th` dataset contains a set of 27,770 papers and 351,500 edges, that represent citations between the papers. The `cora` dataset consists of a selection of 2,708 scientific publications on Machine Learning, classified into one of seven classes, based on their topics. The papers were selected in a way such that in the final network every paper cites or is cited by at least one other paper. This citation network consists of 5,429 links[4]. We use a convention that a citation of a paper u by a paper v is represented by an edge (u, v).

Results

Space-Time lattice

We use the space-time lattice model of "Space-Time lattice model" section on a small scale because we can make effective visualisations which illustrate the key ideas of communities in DAGs. In Fig. 3 we show the same instance of the space-time lattice model in which nodes are placed on an eight-by-eight square lattice. We then use colour to show the different partitions produced using different algorithms.

The bottom two examples of Fig. 3 illustrate standard community detection algorithms applied to our network. Here these are found by optimising modularity with two different resolutions (obtained by scaling the null model term). The communities found tend to be located in one region of space reflecting the spatial constraint in the model. However, they extend over several values of time so these communities do not respect the order inherent in the DAG. That is, the nodes in the these modularity communities are not similar in terms of their time coordinates.

If this was a citation network, we would be grouping papers together with different publication dates. If we were interested in comparing the impact of each paper through citation counts, the older papers would have an advantage making the comparison unfair. If this toy model represented a food web, these communities derived from the undirected network might capture aspects such as distinct parts of the ecosystem. However it fails to find the hierarchy, each group would contain predators and their prey.

So these communities from the undirected graph may be of use in some contexts, but they are not particularly sensitive to the inherent structure of the DAG coming from the time direction.

The middle two examples in Fig. 3 use the height- and depth-antichain partitions. Now the hierarchical structure coming from the time direction is clearly exposed, with many communities running across one row, perhaps two. Where two rows are involved, it is showing how the these community detection methods are highlighting where the placement of the nodes in the visualisation does not reflect the topological reality because of

[4]In practice there are often "bad" links which are in the "wrong" direction, from a newer document to an older document. This is because documents are published in different versions and the text available may not have been created at the time associated with the document in the data set. For instance, a revised version of an arXiv paper carry the same index as the first version. A journal article has several associated dates: first submitted, date accepted, published online, formal publication date and so forth. Such bad links can introduce cycles and these must be dealt with. For our arXiv data we simply dropped the links that do not respect the agreed order of time; they account for less than 1% of the data. For the `cora` data we used the approach, discussed in Sun et al. (2017) to recover a DAG from a directed network, leaving us with a network, composed of 5,255 edges.

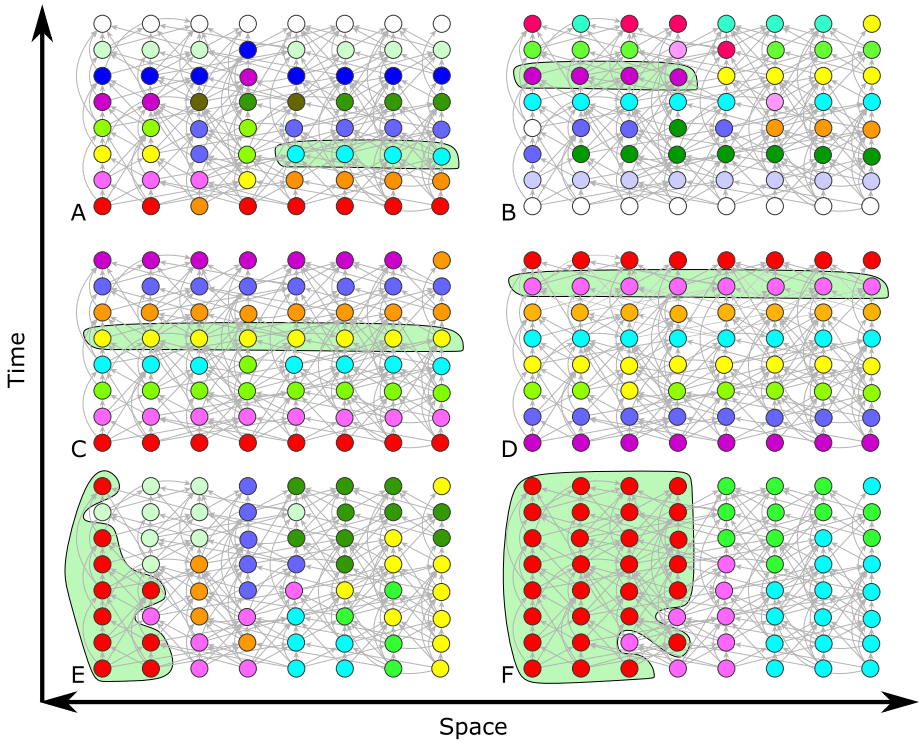

Fig. 3 Examples of various partitions of DAG from a simple space-time model. In this DAG, edges are more probable between two nodes if they are a shorter Manhattan distance apart (in the space-time). The time coordinate, which induces the acyclicity in the network, is vertical. Colour indicates community of a node. The top row shows siblinarity communities based on common successors (**a**), and predecessors (**b**). The central row shows layers based on height (**c**) and depth (**d**). The communities in the bottom row are from modularity community detection using resolution parameter value of one (**e**) and two (**f**). Nodes coloured white are in a community of size one. Siblinarity partitioning tends not to form communities (top row) which stretch across the whole network unlike those communities based on height or depth shown in the centre row. The communities found using traditional methods respect the spatial (horizontal) constraint but show no respect for the order in the DAG as they are spread over several times vertically

the stochastic aspect of edge placement. This shows that these antichain partitions can do a useful job, picking out the true topological hierarchy as defined by the data.

However the height- and depth-antichain communities show the opposite problem from the undirected graph communities. That is they respect the order coming from the arrow of time but they fail to pick up in any way the spatial clustering in the data. If this was a citation network, these would cluster papers published at a similar time regardless of academic field. For a food web, all predators at the same level would be grouped together regardless of whether or not they were competing in the same ecological niche.

The top two networks in Fig. 3 show the siblinarity antichain communities, one based on predecessor neighbours and the other using successors. This shows that the communities respect both the time and the spatial clustering in the data. The nodes tend to be in the same row and only contain nodes which are close by reflecting the spatial clustering in the model. This suggests that when applied to a citation network, this method will cluster papers which are directly comparable, similar publication date and similar field. Applied to food webs, siblinarity will only group predators at the same level competing in the same niche.

Diversity of antichains: the Price model with fields

The space-time model illustrated that the communities formed by our methods respect the order implicit in a DAG. Our method also aims to create groups which are similar in other ways as reflected in the network structure. To test this we use our modified version of the Price model of "Price model with subject fields" section. We will use the language of a citation model, reflecting Price's original context. So here we say we are aiming to group papers (nodes) of a similar age (in an antichain) published in the same academic field based on the network structure alone (using co-citation or bibliometric coupling).

Here we consider the case when a field is assigned stochastically using a uniform distribution so that it is equally likely to assign a field $f = 1$ and $f = 5$. Other variations are possible, for instance, one could assign the fields sequentially deterministically or create non-uniformity in field sizes. Some of these variations will be discussed in Additional file 1: Appendix F.

Our aim is to show that our antichain partitions are largely composed of papers from the same field which we measure using Shannon's diversity metric D (Jost 2006). That is given an antichain $\mathcal{A}$ we find p_f, the fraction of nodes in the antichain which are in field f. The diversity of this antichain is then given by

$$D(\mathcal{A}) = \exp\left(-\sum_f p_f \ln(p_f)\right) \tag{12}$$

so that $1 \leq D(\mathcal{A}) \leq |\mathcal{F}|$.

The results for the average diversity of a partition, $\sum_{\mathcal{A}\in\mathfrak{A}} D(\mathcal{A})/|\mathfrak{A}|$ are shown in Fig. 4, results for the distribution of $D(\mathcal{A})$ values within $\mathfrak{A}$ are shown in Fig. 5. Not surprisingly the diversity of height and depth antichains is almost always larger than that of communities found from a siblinarity antichain partition. This is expected because the height and depth antichains typically contain more nodes, reflected by the low number of points on their plots, as their construction is not affected by the field labels. Since they are constructed without regard to the field labels, we would expect, and find, that the diversity of the height and depth partitions is close to the maximum value where $p_f \approx 1/|\mathcal{F}|$ and so $D \approx |\mathcal{F}|$.

We also noted that the diversity of antichains that are height or depth based does not approach maximum if the sizes of fields are not uniform. This is expected as Shannon's entropy not only accounts for number of species in the ecosystem but also their relative abundances. Uneven sizes of fields result in expected diversity score lower than maximum, as the maximum would be obtained if we had equally abundant species, as seen in the other two figures.

Florida bay food web

We also studied our antichain partitions in the DAG version of the Florida Bay food web data set (Ulanowicz et al. 1998). Results are shown in Table 1.

We see that many of our siblinarity antichains consist of similar species as indicated by their similar names. For instance community 9 consists of, amongst other crustaceans, several types of amphipods. Another example of how our siblinarity antichains work is the example of the green turtle in community 25. The "Green Turtle" node has been placed in a community along with much smaller species, seemingly very different from turtles, such

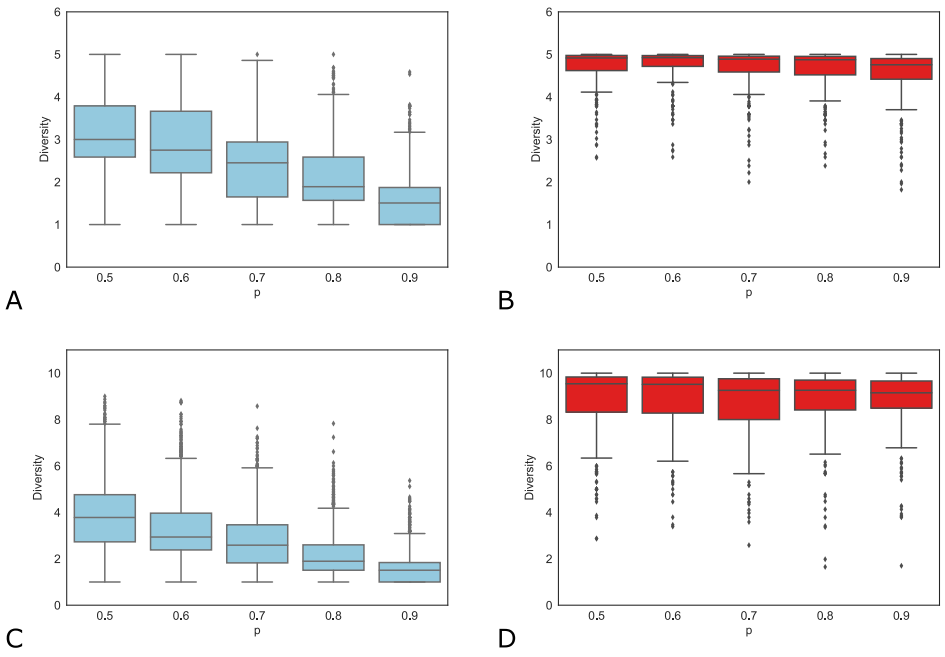

Fig. 4 Comparison of diversity in height-based antichain partitions (red boxes on right) versus siblinarity communities, based on common successors (blue boxes on left) in the Price model with five fields (top row, **a** and **b**) and ten fields (bottom row, **c** and **d**) and different intraconnectivity probability *p*. We studied ten networks of 5,000 nodes for each set of parameters. Figures show that the diversity of siblinarity antichains is clearly significantly smaller than that of heights. The box extends from the lower to upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Flier points are those past the end of the whiskers

as shrimp and isopods. However, green turtles feed on thalassia (a type of seagrass, commonly known as turtlegrass), which is also a food source for organisms, represented with "DOC" (Dissolved Organic Carbon) node and isopods. Furthermore, all of the species in this green turtle antichain community feed on epiphytes. Nodes in this antichain all have the same height of two, but their depths range from 22 for "Epiphytic Gastropods" to 30 for "Isopods", "Herbivorous Shrimp", and "Thor Floridanus". Another example of an antichain, in which nodes of a variety of heights and depths were collected into a siblinarity community is the community 27, consisting of "Other Snapper" (height 27, depth 4), "Other Pelagic Fishes" (height 28, depth 4) and "Spotted Seatrout" (height 29, depth 3).

Citation networks

The Science Citation Index, introduced by Garfield in 1964, has a functionality to search for "Related Records" (Newman 2010). A Related Record is any record which shares at least one cited reference with the original source record. The more shared references, the more closely related the records are — an extension of the notion of citation searching to track a subject area. The related records are further ranked based on the number of shared references.

So in a bibliometric context our approach is an extension for "Related Records" functionality in the SCI. As pointed out by Newman, simply using bibliographic coupling or co-citations is flawed: strong bibliographic coupling or co-citations only occur between papers that have either large bibliographies or are highly cited, respectively (Newman

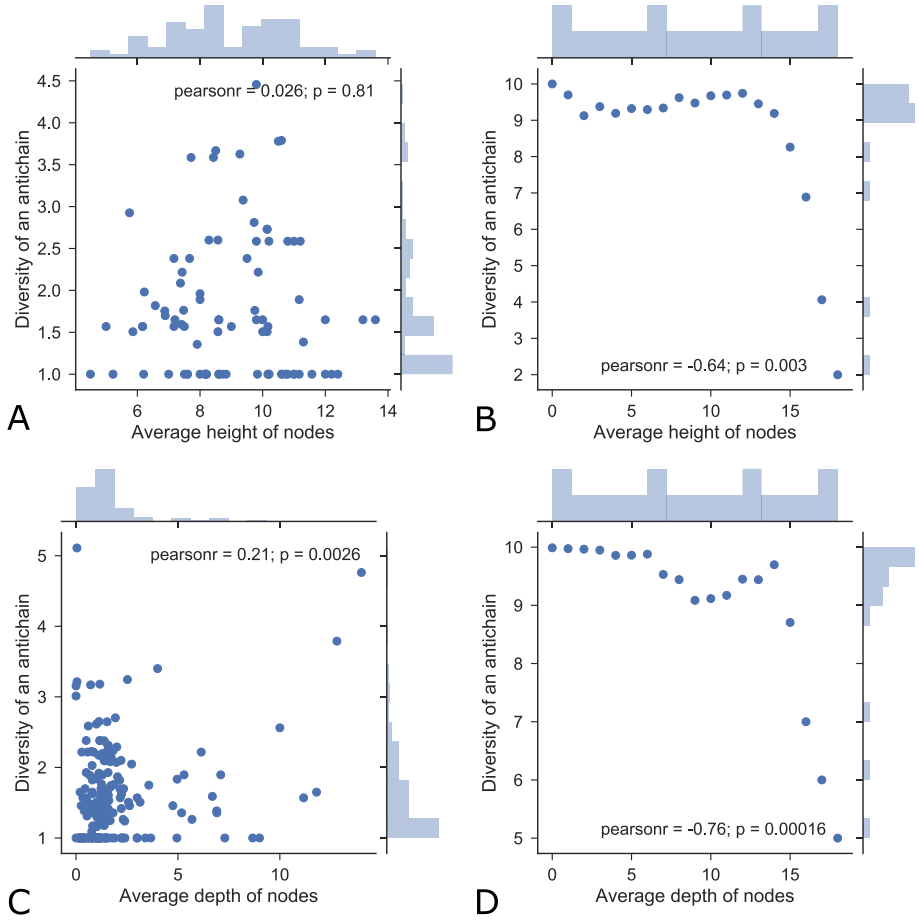

Fig. 5 Shannon's diversity of antichains in a Price model with planned partition of nodes into five fields for networks of 5,000 nodes. Each node is assigned to a field sequentially and nine times out of ten a new node chooses to connect to an existing node of the same field, with a preference for large degree nodes, $\phi = 0.9$. Each new node attaches to $m = 3$ older nodes. Each point represents an antichain in a given partition, with the average height or depth of nodes in the antichain and Diversity. Four different types of antichain partition are shown: **a** - successors-based siblinarity antichains, **b** - height-based antichains, **c** - predecessors-based siblinarity antichains, **d** - depth-based antichains. The height and depth antichains are clearly much bigger (few points on their plots) but much less diverse, with their diversity close to the theoretical maximum 10.0. On the other hand, the siblinarity antichains based on co-citation similarity (**a**) or bibliometric coupling (**b**) are drawn largely from the same field with diversities close to the minimum value of 1.0

2010). Our approach naturally eliminates this flaw: by comparing the observed neighbourhood overlap to a null model, we can evaluate the significance of the similarity regardless of the degrees of two nodes.

To understand what type of nodes are joined together in antichains and why, we looked at various statistics for successor siblinarity antichains in the `hep-th` dataset. These neighbours are newer papers that cite the papers in the antichain communities. We found 10,806 successor based antichain communities, 7,352 of which are composed of singular nodes and 1,225 are composed of at least five nodes. Several of the measures we found useful can be defined in terms of a bipartite network constructed for each antichain community. This is simply the undirected subgraph of the full network, containing the nodes in the antichain of interest, the nodes of all their neighbours (here successor neighbours) and then all the edges between these nodes in the original graph. This is a bipartite

Table 1 Siblinarity communities, based on common prey and common predators in the Florida Bay food web

No.	Species (type)
1	$2\mu m$ Spherical Phytoplankt, Synedococcus, Oscillatoria, Small Diatoms ($< 20\mu m$), Big Diatoms($> 20\mu m$), Dinoflagellates, Other Phytoplankton, Benthic Phytoplankton, Thalassia, Halodule, Syringodium, Roots, Drift Algae, Epiphytes.
3	Acartia Tonsa, Oithona nana, Paracalanus, Other Copepoda, Other Zooplankton, Sponges, Bivalves.
7	Coral, Other Cnidaridae.
9	Benthic Crustaceans, Detritivorous Amphipods, Herbivorous Amphipods, Detritivorous Gastropods, Detritivorous Polychaetes, Suspension Feeding Polych, Macrobenthos, Detritivorous Crabs.
18	Toadfish, Brotalus.
19	Other Killifish, Goldspotted killifish, Blennies, Clown Goby, Silverside, Lobster, Predatory Crabs, Callinectus sapidus, Bay Anchovy, Rainwater killifish, Mullet, Other Horsefish, Gulf Pipefish, Dwarf Seahorse, Code Goby, Halfbeaks.
22	Flatfish, Grunt, Pinfish, Rays, Porgy, Scianids, Parrotfish, Bonefish, Needlefish, Snook, Puffer, Manatee.
23	Omnivorous Crabs, Pink Shrimp.
25	DOC, Isopods, Herbivorous Shrimp, Thor Floridanus, Sailfin Molly, Green Turtle.
26	Sharks, Tarpon, Lizardfish, Grouper, Jacks, Pompano, Gray Snapper, Red Drum, Mackerel, Small Herons & Egrets, Ibis, Roseate Spoonbill, Herbivorous Ducks, Omnivorous Ducks, Gruiformes, Small Shorebirds, Gulls & Terns, Kingfisher, Loggerhead Turtle, Hawksbill Turtle.
27	Other Snapper, Other Pelagic Fishes, Spotted Seatrout.
28	Stone Crab, Sardines, Anchovy, Other Demersal Fishes, Filefishes.
29	Barracuda, Loon, Greeb, Pelican, Comorant, Big Herons & Egrets, Predatory Ducks.
33	Free Bacteria, Dolphin.
35	Raptors, Crocodiles.
36	Output, Respiration.
Ind.	Input (0), Water Flagellates (2), Water Cilitaes (4), Benthic Ciliates (5), Meroplankton (6), Meiofauna (8), Benthic Flagellates (10), Water POC (12), Predatory Gastropods (13), Echinoderma (14), Predatory Shrimp (15), Predatory Polychaetes (16), Mojarra (20), Benthic POC (24), Spadefish (31), Catfish (32), Eels (34).

Some communities consist of a single species and these are listed together under community number "Ind." to reduce the size of the table. The numbers in brackets indicate the community index in the induced graph. Note the green turtle in community 25 appears to be out of place with the other smaller species in the same community but they all tend to feed on similar species

network given the two types of nodes (see Additional file 1: Appendix D for a formal definition). To remove noise from weakly connected nodes in antichain communities, we also found it was useful to reduce the bipartite subgraphs by dropping any neighbour node that is connected to just one node in the antichain.

We then applied a number of simple network statistics to these bipartite subgraphs in order to understand the nature of our antichain communities. Our results are summarised in Fig. 6. Several simple network statistics, formally defined in Additional file 1: Appendix D, proved useful. The most basic measures are the size of each antichain size $|\mathcal{A}|$, and $|\mathcal{N}|$ the total number of neighbours of each community. The ratio of these gives us the average number of neighbours of an antichain node in each community, $\langle k \rangle$, and we also use the standard deviation of that measure, $\sigma(k)$. A low $\langle k \rangle$ indicates a sparse "zig-zag" pattern with very weak overlaps between members of the antichain. A large $\sigma(k)$ indicates that we may have one well connected node in the antichain plus many connected to only a couple of neighbours. The last network statistic shown in Fig. 6 is the density of the bipartite graph, the total number of edges divided by the maximum number possible which is equal to $\langle k \rangle / |\mathcal{N}|$. This final network measure is close to one only if almost every member of our antichain community has the same set of neighbours.

The majority of the successor antichain communities we find in `hep-th` are small, composed of around 10 to 20 nodes. They typically share around 40 neighbours though a

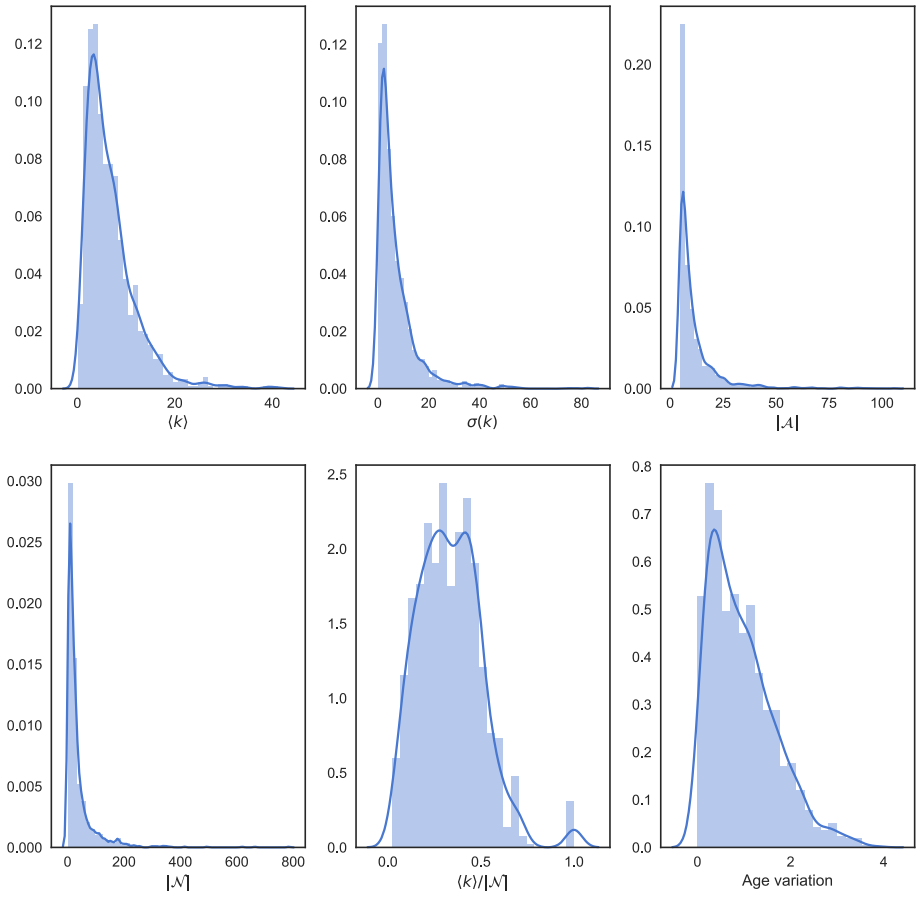

Fig. 6 Summary statistics of antichains and shared successors of those antichains, based on which they were obtained in `hep-th` citation network. In each case the relative frequency distribution is plotted over all the antichain communities found using successor neighbours, older neighbours who are cited by the newer papers in the antichain community. The statistics used are as follows: the average number of neighbours of each antichain node $\langle k \rangle$, the standard deviation of the number of neighbours of each antichain node $\sigma(k)$, the size of each antichain community $|\boldsymbol{\mathcal{A}}|$, the density of each bipartite graph $\langle k \rangle / |\boldsymbol{\mathcal{N}}|$. The final panel shows the age variation of papers in each antichain. That is the standard deviation in the publication date of papers in each community, known in terms of months but we give the plot in terms of years

few antichain communities can have close to a 1,000 shared neighbours. The distribution of $\langle k \rangle$ peaks around 6 but there is often a sizeable variation in the degree as shown by similarly large values of $\sigma(k)$.

The density of the bipartite graphs, the ratio between $\langle k \rangle$ and $|\mathcal{N}|$ in Fig. 6, shows that the antichain communities in `hep-th` possess a wide range of values. The distribution of the density peaks at values close to $0.25 - 0.5$ for many antichains, but the number of antichains with minimal and maximal values of the ratio is non-negligible.

Our last statistic is not a network measure but exploits information we have on the month of publication for each paper in our `hep-th` dataset. We looked at the standard deviation in the ages of papers in each antichain. We see that nodes that feature in the same antichain are also nodes of similar age, which varies by less than one year for most antichains.

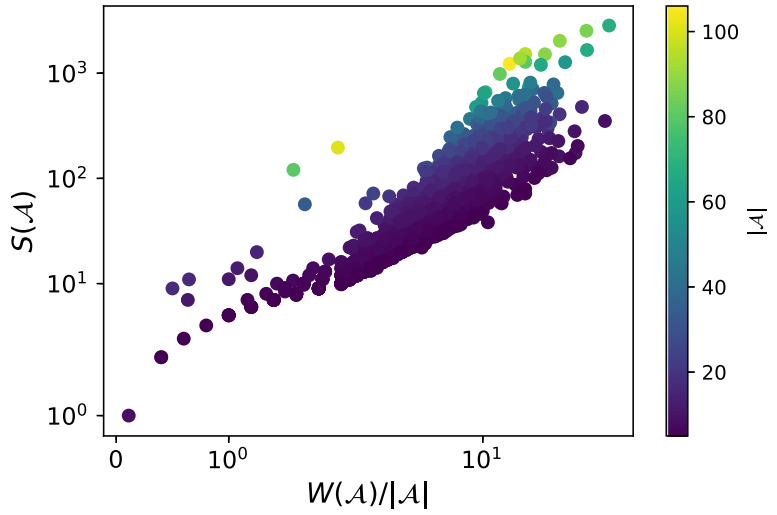

Fig. 7 A colour plot showing a relation between siblinarity score of an antichain $\boldsymbol{\mathcal{A}}$, $S(\boldsymbol{\mathcal{A}})$, $W(\boldsymbol{\mathcal{A}})/|\boldsymbol{\mathcal{A}}|$ and $|\boldsymbol{\mathcal{A}}|$ for an antichain partition obtained in `hep-th` citation network. Antichains were obtained considering co-citations. Figure shows that siblinarity scores, obtained by antichains relate to $W(\boldsymbol{\mathcal{A}})/|\boldsymbol{\mathcal{A}}|$ but not to the size of an antichain

We also looked at siblinarity values obtained by antichains in `hep-th` and how they relate to $W(\mathcal{A})/|\mathcal{A}|$, the mean similarity of nodes in an antichain as defined in 1, as well as the size of antichains $|\mathcal{A}|$.

Figure 7 shows that for a given size of an antichain community, there are a wide range of values for the average neighbour overlap $W(\mathcal{A})/|\mathcal{A}|$ and the siblinarity measure. Higher overlap tends to produce a better quality antichain, that is a higher siblinarity value, but there is still a large variation, reflecting the role of the null model.

For instance some large antichains have small siblinarity scores. Conversely, the largest siblinarity score is obtained from an antichain community which is composed of 68 nodes (antichain with an index 2175 in our data), large but not the largest we found. This antichain is composed of papers, published in 1992–1994, some of which are relatively highly cited (large out-degree). The paper with the largest out-degree of those in this community has an out-degree of 112 (paper index 9201061).

We also studied a second citation network, the `cora` dataset, in which each node is labelled by one of seven different topics. We use this label to quantify the diversity of the nodes in the antichain communities we find, similar to the analysis in "Diversity of antichains: the Price model with fields" section. Figure 8 shows that the diversity of 134 siblinarity antichains is much smaller than the partitions based on heights. Furthermore, most of the time the diversity values for siblinarity antichains are equal to one, which indicates that all nodes in the antichain have the same label. This result confirms that the siblinarity communities are communities of similar papers, not just ones published around the same time.

Discussion

In network analysis an edge between nodes generally indicates a close relation and a similarity between the two connected nodes. This is exploited to understand local features, such as centrality of individual nodes, as well as gain insights on a macroscopic structure, such as the extent of community structure.

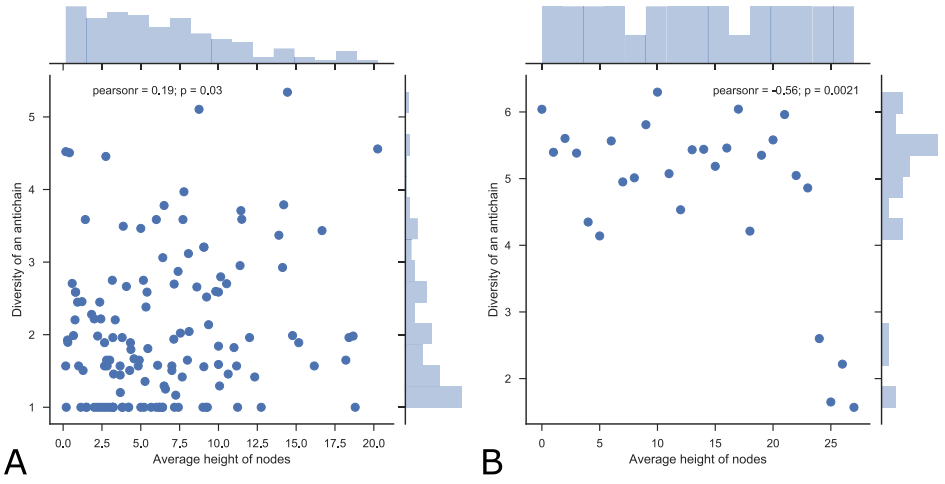

Fig. 8 Shannon's diversity of nodes in antichains of the `cora` citation network. Each node has a label assigned to it, based on the topic the paper is on. There are seven topic in total. Two different antichain partitions are shown: **a** — successors and predecessors-based siblinarity antichains, **b** — height-based antichains (diversity calculated only for antichains, that are composed of at least five nodes). The horizontal axis is average heights of nodes in the antichain. The height antichains, with diverity values often close to the maximum value of seven, are much more diverse than the siblinarity comminties

In this work we have highlighted that in some circumstances, the absence of a pairwise relationship can be a just as useful a signal as the presence of it. Our focus has been on Directed Acyclic Graphs (DAGs) where the implicit order in such networks prohibits the direct connection of many similar nodes. As this order is intrinsic to the very nature of a DAG, our response has been to embrace this order as it reflects important features of the data. To find meaningful communities, we have turned to antichains, sets of disconnected nodes which often play a useful role in the study of DAGs and which reflect the topological properties on the networks we consider.

However, communities in networks can be defined as nodes which share strong intra-community connections (Fortunato 2010). Since the strongest connection is a direct edge between members of nodes in the same community, the antichain seems to be the precise opposite of a network community. However, even in tightly connected network communities, not all nodes are connected to each other. So network communities reflect on a larger mesoscopic scale of intra-connection in a network. So to find antichain communities we look beyond one edge and use node-node relationships as defined by pairs of edges in order to find closely related nodes. That is to ensure our antichains contain similar nodes, we have used neighbourhood overlap to define the siblinarity of a partition of a DAG into antichain communities. Neighbourhood overlap has long been used to define node similarity based solely on the network topology and it does not require a direct connection between nodes.

We have shown using many examples how the antichains which maximise siblinarity are interesting and relevant. Our communities are extremely different from usual communities found in network analysis as the simple examples in Fig. 3 show. The communities in traditional network analysis consist of strongly interconnected nodes while siblinarity communities consist of nodes that are strictly not connected, but which share similar neighbours.

Likewise, it is important to note that our siblinarity antichains are very different from those produced by existing antichain methods such as the graph layering method often used in visualisation (Sugiyama et al. 1981; Mirsky 1971; Tang and Hu 2013; Gansner et al. 1993; Nikolov and Tarassov 2006; Healy and Nikolov 2002; 2013). The height or depth antichains used in our work represent such layering methods and are used to illustrate the difference between those methods and our approach. These graph layering methods have different goals which, directly or indirectly, produce as few antichains as possible, often "maximal antichains" which are not proper subsets of any other antichain. As a result such layering algorithms produce communities in which the nodes have very different properties, the antithesis of the goal in community detection (Fortunato 2010) which is to find clusters of similar nodes.

However, as with most community detection methods, we have a resolution parameter, our λ in (7), which can change the size of typical communities. In our case, lowering the value of λ decreases the number of antichains found so we can interpolate from antichains of very similar nodes of primary interest to us, to a DAG broken into a small number of layers regardless of the similarity of nodes within those layers. In that limit we expect to produce something similar to traditional layering methods.

The resolution parameter highlights another issue. Our main aim was to illustrate how we could reconcile the order constraint implicit DAG with the strong intra-community connections typical of network clustering methods. However, we have illustrated our concept using a modularity-like construction, but modularity brings its own baggage such as a well known resolution problem (Fortunato and Barthelemy 2007). Our resolution parameter, or similar resolution parameters (Schaub et al. 2012; Lambiotte et al. 2014), can be exploited to find "better" communities as defined by suitable measures, but we are also mindful that there is some debate about how to define a "good" community structure e.g. (Peel et al. 2017). To see if the ideas and solutions that worked for undirected graphs transfer well to the antichain communities we have suggested for DAGs, it is probably best to work within one area. Citation networks often come with rich metadata necessary for such studies, perhaps complementing traditional approaches such as those discussed in Gläser et al. (2017).

This novel type of node partitioning has several potential uses whenever data is well represented by a DAG. In the context of bibliometrics, it is extremely difficult to measure the impact of papers published at different times in different fields. Different research fields publish papers at different rates and cite at very different frequencies, for instance pure maths and astrophysics have very different citation patterns. Assigning a field to a set of papers is non-trivial, for example see Boyack et al. (2017). Even the publication date is not unique (Haustein et al. 2015) as an article has a range of dates: the appearance of various electronic versions, the date it was assigned an index number, formal publication date, and so forth. What our approach offers is the ability to group truly similar papers together, in terms of both topic and age, using the topology of the citation network alone. For instance it is sensible to compare the impact of papers in our antichains as the effects of time and field have been accounted for and the papers should have a similar opportunity to make an impact. Other approaches tend to rely on metadata such as "publication dates" (unreliable as noted) so our approach using the citation network alone complements and enhances the tools already available. Another use could be as a recommender system for publication search: given an input paper,

one could find alternative similar publications by looking into the antichain of the input paper.

Our siblinarity algorithm, like any clustering algorithm, can also be used to coarse-grain a DAG which can help navigate a large amount of data. In our case we would contend the order of a DAG is important. Having a predator and its prey in the same community may not be a useful classification when examining food webs. Indeed we exploit this coarse graining in our implementation of the Louvain algorithm to find antichain partitions which have approximate maximal values of siblinarity.

More broadly, we see our work as emphasising that the order found in some data is an important feature which should not be ignored. However, in the case of DAGs, this constraint is not reflected in most traditional network measures suggesting that network analysis methods need to be reexamined carefully in such cases. This has been acknowledged in some contexts, the use of co-citation and bibliometric coupling, our neighbour overlap similarity measure, is well known in scientometric analysis and again it is a response to the idea that many similar papers are not always directly linked by a citation. However, enforcing the constraint has not always been taken to its logical conclusion as it requires further adaptation of existing methods.

We have developed our work in terms of a directed acyclic graph. However, the natural network representation of many data sets where there is a strong order or hierarchy is a directed graph with a few cycles. For instance in citation data, a document can cite a document with a later publication date. In part this is because documents have several different publication dates (Haustein et al. 2015). Such backwards citation links are regularly found, with between 0.005% and 0.4% of links reported to be in the wrong direction in various citation networks (Clough et al. 2014).

There are, however, several methods to produce an exact DAG from such data which then allows our methods to be applied. One could just delete the few links which do not respect the dominant order (Clough et al. 2014) or one could use more sophisticated approaches such as "agony" (Gupte et al. 2011; Tatti 2017; Letizia et al. 2018). Our antichain method suggests another approach. Our method works in principle on directed graphs, not just those with no cycles. We exploit this in the coarse graining step of our Louvain-style numerical optimsation, which can produce cycles in the derived graphs used at later stages of the method, see Additional file 1: Appendix C. Our approach should work well if there are few cylces. In this case, one can look at "bad links", cases where citations go from an older to a newer document or perhaps even go in both directions. The two documents connected by such a bad link will be placed in two different antichain communities. By looking at the properties of the two communities, we might be able to find the best direction for the bad link. We could even see if deleting the bad link makes most sense as indicated by a higher siblinarity score for the merger of the two antichain communities.

It is worth pointing out that although we developed our methodology in terms of DAGs, this type of partitioning can be applied to other types of networks, such as multilayer networks, bipartite networks, directed networks and even undirected networks (although in the case of the last type, antichains would most likely be composed of individual nodes). Higher order structures, such as our two-step network used in our version of modularity, are important in many areas of research so this could be a way to tackle more general network problems such as those highlighted in Traag et al. (2019).

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1007/s41109-020-00255-5.

Additional file 1: Supplementary information: appendix.

Abbreviations

DAG: Directed acyclic graph

Acknowledgements

V.V. acknowledges financial support from EPSRC, grant EP-R512540-1.

Authors' contributions

Both authors developed the theoretical concepts and designed the experiments discussed in the paper. VV created the software used and performed the data analysis. Both authors wrote the manuscript. TSE supervised the work. Both authors read and approved the final manuscript.

Funding

This work was funded by EPSRC, grant EP-R512540-1.

Availability of data and materials

Our code implementing these methods, the network data analysed during this study, and the datasets supporting the results discussed in this article are all available in a Figshare repository 10.6084/m9.figshare.9725159 (Vasiliauskaite and Evans).

Competing interests

The authors declare that they have no competing interests.

Received: 18 September 2019 Accepted: 6 February 2020
Published online: 21 February 2020

References

Benson AR, Gleich DF, Leskovec J (2016) Higher-order organization of complex networks. Science 353(6295):163–166. 1612.08447

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):10008

Boyack K, Glänzel W, Gläser J, Havemann F, Scharnhorst A, Thijs B, van Eck NJ, Velden T, Waltmann L (2017) Topic identification challenge. Scientometrics 111(2):1223–1224

Boyack KW, Klavans R (2010) Co-citation analysis, bibliographic coupling, and direct citation: Which citation approach represents the research front most accurately? J Am Soc Inf Sci Technol 61(12):2389–2404. https://doi.org/10.1002/asi.21419

Clough JR, Gollings J, Loach TV, Evans TS (2014) Transitive reduction of citation networks. J Complex Netw 3(2):189–203

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174. 0906.0612v2

Fortunato S, Barthelemy M (2007) Resolution limit in community detection. PNAS 104:36–41. https://doi.org/10.1073/pnas.0605965104

Gansner ER, Koutsofios E, North SC, Vo K-P (1993) A technique for drawing directed graphs. IEEE Trans Softw Eng 19(3):214–230

Gerasoulis A, Yang T (1992) A comparison of clustering heuristics for scheduling directed acyclic graphs on multiprocessors. J Parallel Distrib Comput 16(4):276–291

Gläser J, Glänzel W, Scharnhorst A (2017) Same data — different results? towards a comparative approach to the identification of thematic structures in science. Scientometrics 111(2):981–998. https://doi.org/10.1007/s11192-017-2296-z

Gupte M, Shankar P, Li J, Muthukrishnan S, Iftode L (2011) Finding hierarchy in directed online social networks. In: Proceedings of the 20th International Conference on World Wide Web. ACM. pp 557–566. https://doi.org/10.1145/1963405.1963484

Haustein S, Bowman TD, Costas R (2015) When is an article actually published? an analysis of online availability, publication, and indexation dates. In: Proceedings of the 15th International Society of Scientometrics and Informetrics Conference. pp 1170–1179. 1505.00796

Healy P, Nikolov NS (2002) A branch-and-cut approach to the directed acyclic graph layering problem. In: Goodrich MT, Kobourov SG (eds). Graph Drawing. Springer, Berlin. pp 98–109

Healy P, Nikolov N (2013) Hierarchical drawing algorithms. In: Tamassia R (ed). Handbook of Graph Drawing and Visualization, Chap. 13. CRC Press, Florida. pp 409–454

Il MJB, Katz DM, Zelner JL (2010) On the stability of community detection algorithms on longitudinal citation data. Procedia Soc Behav Sci 4:26–37

Jaccard P (1912) The distribution of the flora in the alpine zone. New Phytol 11(2):37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x

Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surv (CSUR) 31(3):264–323

Jost L (2006) Entropy and diversity. Oikos 113(2):363–375

KDD cup (2003) Datasets. https://www.cs.cornell.edu/projects/kddcup/datasets.html. Accessed 10 Oct 2017

Kessler MM (1963) Bibliographic coupling between scientific papers. Am Doc 14(1):10–25

Lambiotte R, Delvenne J-C, Barahona M (2014) Random walks, markov processes and the multiscale modular organization of complex networks. IEEE Trans Netw Sci Eng 1(2):76–90. https://doi.org/10.1109/tnse.2015.2391998
Leicht EA, Clarkson G, Shedden K, Newman MEJ (2007) Large-scale structure of time evolving citation networks. Eur Phys J B 59(1):75–83
Letizia E, Barucca P, Lillo F (2018) Resolution of ranking hierarchies in directed networks. PLoS ONE 13(2):1–25
Lu Q, Getoor L (2003) Link-based classification. In: Proceedings of the 20th International Conference on Machine Learning. Springer-Verlag. pp 496–503. https://doi.org/10.1007/1-84628-284-5_7
Martyn J (1964) Bibliographic coupling. J Doc 20(4):236
Mirsky L (1971) A dual of dilworth's decomposition theorem. Am Math Monthly 78(8):876–877
Newman MEJ (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103(23):8577–8582. https://doi.org/10.1073/pnas.0601602103
Newman M (2010) Networks: An Introduction. Oxford University Press, Oxford
Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113
Nikolov NS, Tarassov A (2006) Graph layering by promotion of nodes. Discrete Appl Math 154(5):848–860
Peel L, Larremore DB, Clauset A (2017) The ground truth about metadata and community detection in networks. Sci Adv 3(5):1602548. https://doi.org/10.1126/sciadv.1602548
Price DJdS (1976) A general theory of bibliometric and other cumulative advantage processes. J Am Soc Inform Sci 27:292–306
Reichardt J, Bornholdt S (2006) Statistical mechanics of community detection. Phys Rev E 74(1):016110
Satuluri V, Parthasarathy S (2011) Symmetrizations for clustering directed graphs. In: Proceedings of the 14th International Conference on Extending Database Technology - EDBT/ICDT '11. ACM Press, New York. https://doi.org/10.1145/1951365.1951407
Schaub MT, Delvenne J-C, Yaliraki SN, Barahona M (2012) Markov dynamics as a zooming lens for multiscale community detection: non clique-like communities and the field-of-view limit. PloS ONE 7(2):32210. https://doi.org/10.1371/journal.pone.0032210
Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data. AI Magaz 29(3):93
Small H (1973) Co-citation in the scientific literature: A new measure of the relationship between two documents. J Am Soc Inf Sci 24(4):265–269
Small H, Griffith BC (1974) The structure of scientific literatures i: Identifying and graphing specialties. Sci Stud 4(1):17–40
Speidel L, Takaguchi T, Masuda N (2015) Community detection in directed acyclic graphs. Eur Phys J B 88(8). https://doi.org/10.1140/epjb/e2015-60226-y
Sugiyama K, Tagawa S, Toda M (1981) Methods for visual understanding of hierarchical system structures. IEEE Trans Syst Man Cybernet 11(2):109–125
Sun J, Ajwani D, Nicholson PK, Sala A, Parthasarathy S (2017) Breaking cycles in noisy hierarchies. In: Proceedings of the 2017 ACM on Web Science Conference - WebSci '17. ACM Press, New York
Tang H, Hu Z (2013) Network simplex algorithm for DAG layering. In: 2013 International Conference on Computational and Information Sciences. pp 1525–1528. https://doi.org/10.1109/iccis.2013.401
Tatti N (2017) Tiers for peers: a practical algorithm for discovering hierarchy in weighted networks. Data Min Knowl Discov 31(3):702–738
Traag VA, Waltman L, van Eck NJ (2019) From louvain to leiden: guaranteeing well-connected communities. Sci Rep 9(1). https://doi.org/10.1038/s41598-019-41695-z
Ulanowicz R, Bondavalli C, Egnotovich MS (1998) Network analysis of trophic dynamics in South Florida ecosystem, fy 97: The florida bay ecosystem. Annual Report to the United States Geological Service Biological Resources Division. Ref. No. [UMCES]CBL. https://www.researchgate.net/publication/237005294_Network_Analysis_of_Trophic_Dynamics_in_South_Florida_Ecosystem_FY_97_The_Florida_Bay_Ecosystem
Vasiliauskaite V, Evans TS Data for "Making Communities Show Respect for Order" Paper. https://doi.org/10.6084/m9.figshare.9725159. https://figshare.com/s/3ecc2bd6919a64916f44
Xu J, Wickramarathne TL, Chawla NV (2016) Representing higher-order dependencies in networks. Sci Adv 2(5):1600028. https://doi.org/10.1126/sciadv.1600028

Publisher's Note