Mantzaris *EPJ Data Science* 2014, **3**:26
http://www.epjdatascience.com/content/3/1/26

EPJ Data Science
a SpringerOpen Journal

REGULAR ARTICLE Open Access

Uncovering nodes that spread information between communities in social networks

Alexander V Mantzaris*

*Correspondence:
alexander.mantzaris@strath.ac.uk
Department of Mathematics and Statistics, University of Strathclyde, 26 Richmond Street, Glasgow, G1 1XH, UK

Abstract

From many datasets gathered in online social networks, well defined community structures have been observed. A large number of users participate in these networks and the size of the resulting graphs poses computational challenges. There is a particular demand in identifying the nodes responsible for information flow between communities; for example, in temporal Twitter networks edges between communities play a key role in propagating spikes of activity when the connectivity between communities is sparse and few edges exist between different clusters of nodes. The new algorithm proposed here is aimed at revealing these key connections by measuring a node's vicinity to nodes of another community. We look at the nodes which have edges in more than one community and the locality of nodes around them which influence the information received and broadcasted to them. The method relies on independent random walks of a chosen fixed number of steps, originating from nodes with edges in more than one community. For the large networks that we have in mind, existing measures such as betweenness centrality are difficult to compute, even with recent methods that approximate the large number of operations required. We therefore design an algorithm that scales up to the demand of current big data requirements and has the ability to harness parallel processing capabilities. The new algorithm is illustrated on synthetic data, where results can be judged carefully, and also on a real, large scale Twitter activity data, where new insights can be gained.

Keywords: social network analysis; community analysis; Twitter; viral content; community connectivity; betweenness; information diffusion

1 Introduction

Online social networks (OSNs) such as Facebook, LinkedIn and Twitter have inspired a great amount of research. Whether it is regarding their uses [1] in different aspects of our daily lives or on how a important scientific breakthrough can spread around the world [2]. These networks can be very large, for example Facebook currently holds around 1 billion user accounts. Despite the obvious computational challenges, analysis of these large datasets provides the opportunity to test hypothesis about human social behavior on an unprecedented scale, and hence to reveal deeper understandings of human social behavior [3]. Furthermore, commercial, government and charitable enterprises can utilize the networks to inform campaigning, advertising and promotion. Hence, there is great potential impact for improvements in the analytical tools designed for analysing social networks.

Within the OSNs generated by users, community structures form naturally, and research into their detection is very active [4, 5]. These developments in community detection have produced a diverse set of methods which are at our disposal. Run times of the algorithms are a major concern, and current datasets can be too large for many of the algorithms available. One approach to deal with the size is by using network samples; for example, [6] analyzes community structure in a subset of millions of nodes taken from Facebook. However, for the types of effects that span over the entirety of the networks, we wish to avoid sampling and deal with complete networks.

Communities in OSNs can emerge for many reasons. A key driver can be homophily [7], where some underlying similarity between users in a community leads to a higher number of edges between these users than with users in a different community. [8] investigates homophily formation and evolution in a online social buyers setting. Here, a community builds trust and supports the activity of online purchases, which is the motivation for more in depth research into the nature of the inter-community connections. Companies have an interest in their brand identity within OSN communities, as users now have the ability to broadcast brand information to many other users within their social reach. Although not based on data from OSNs, [3] discusses the attributes that users exhibit to utilize their business associations, and how companies should work to cultivate their brand presence with customers. The authors also raise many interesting questions concerning the dynamic elements of brand presence which are relevant to this work.

It is also interesting to elaborate on how distinct communities are brought together to create large connected graphs. Without the connectivity between dense communities, isolated components would not support many of the fascinating phenomena that have been observed, notably the hugely influential small world effect [9], where there exist surprisingly short paths between members of the network located in different communities. By definition, the density of those edges connecting communities is less than the density of edges within communities. The sparsity of the between-community connectivity is the basis for community separation quality measures such as the modularity index, [10]. The relatively low number of these edges connecting communities together gives them special importance as they are critical for the graph's connectivity. A recent study explores this network feature using examples from brain connectivity, [11], concluding that connection costs can explain these modular networks. For the applications in OSNs, where companies seek to harness the power of Internet advertising, nodes which offer community traversal connections are critical targets [12]. The aim of this work is therefore to give a simple and scalable methodology for defining and discovering this type of key structural component. For the remainder of this paper an edge connecting two different communities will be referred to as a *boundary edge* and the nodes on either side of these edges as *boundary nodes*.

It is important to have in mind that the edges created facilitate an information exchange but when a node receives content, independently it decides on whether to repeat this received information to its follower node set. In future time steps this can include nodes that were previously not included in the sharing of this content for whatever reasons or constraints might exist. For content to spread throughout the network this decision to repeat the content must be consistently agreed on independently. The number of times this must occur is increased when there is a large number of distinct communities and only a few boundary nodes acting as regulators for the content to cross communities and

become *viral*. The term viral usually assumes that a large portion of the nodes in network are aware of a piece of information or content regardless of the specific niche community they may belong to. Viral activity can be identified through conversation volume spikes, or cascades, as users share a common piece of content in a short amount of time.

In general the content users would classify as *news* has many examples of viral spreading of content. Twitter is sometimes considered to be a news source, with [13] counting at least 85% of Tweets being related to headline news. Much of which includes news of commercial interest and opens possibilities for real time engagement. Real time monitoring of these events being discussed is therefore essential for automated engagement. With spikes in topics lasting in the order of minutes, the run time of an algorithm should be reduced as much as possible and the ability of the algorithm to utilize the hardware of multiple processors is highly desirable. The works [14, 15] discuss this real time monitoring of events and gives a number of case studies, comparing techniques for spike detection. Our work has a slightly different emphasis, since we aim to detect nodes and edges that facilitate propagation of information, and hence would be natural candidates for monitoring and intervention.

To introduce notation and background, we consider a graph $G = (V, E)$, with $N = |V|$ number of nodes and $M = |E|$ as the number of edges. The standard centrality measure most relevant to our work is betweenness (shortest path betweenness), [16]. For a node v, this measure is defined as

$$b_v = \sum_{i \neq j \neq v} \frac{\sigma_{i,j}(v)}{\sigma_{i,j}}, \tag{1}$$

where $\sigma_{i,j}$ counts the total number of shortest paths between i and j, and $\sigma_{i,j}(v)$ counts how many of these pass through node v. Hence, b_v gives an indication for the amount of potential control or influence node v has on the information flow between all other nodes in the network. Computing this measure straightforwardly for each node requires a large number of operations, $\Theta(N^3)$, leading to a run time that is impractical for large networks. Using Brandes algorithm [17] a complexity of $\Omega(M \times N)$ is possible, which is still time consuming for the networks with millions of nodes.

The strict assumption that information flows along shortest paths (geodesics) is not always appropriate, as discussed, by Newman [18], who proposes a *random walk betweenness* measure computed using matrix methods. An important criticism of the geodesic viewpoint, which also motivates the random walk alternative, is that when passing messages to target nodes, typical users do not have the global network information and hence may not be aware of the shortest paths between pairs of nodes to be able to place them along the correct route. The runtime for this *random walk betweenness* measure is $\Omega((M + N)N^2)$ and the algorithm requires matrix inversions. We also note that these two betweenness measures above are designed for static networks, and changes in the size of communities over time can affect the distribution of the betweenness values amongst the nodes.

2 Methodology

Given networks arising from online social media, there are many cases where rich community structure is observed. The edges connecting these separately clustered groups

Table 1 Outline of the boundary node vicinity algorithm

1	Extract the set of connected graphs from the original graph
2	For each connected component obtain the community labels for the graph
3	Obtain the set of boundary nodes
4	Measure the local vicinity of each boundary node using the fixed length random walk method and aggregate all of the values in the graph into a normalized score

of nodes are referred to as boundary nodes here, and those edges connecting them are boundary edges. In this section our algorithm for measuring the boundary node proximity is described. The goal is to be able to rank nodes in a network according to their ability to influence nodes across different communities by the information (content) they exchange. This will reveal the boundary nodes, which play a key role in exchanging information between different communities, and those nodes surrounding them in their local vicinity. The algorithm is based on the premise that information travels via a random walk rather than through a shortest path route.

An adjacency matrix A of dimension N will be used to represent the original network, where $A_{i,j} = 1$ when there is an edge between nodes i and j. Once the network has been decomposed into its connected subcomponents and the community labelling has been assigned, the set of boundary edges, connecting two nodes (i,j) belonging to different communities, can then be defined as:

$$\mathbf{W}_{i,j} = \left\{(i,j) : i \in C_1, j \notin C_1, i \notin C_2, j \in C_2\right\}. \tag{2}$$

Here (C_1, C_2) are two communities belonging to the list of community labels, $\mathbf{C}$, in the graph. We assume that the number of community labels will be much less than the number of nodes, $|\mathbf{C}| \ll N$. From the boundary edge set $\mathbf{W}$, the boundary nodes $\mathbf{B}$, can be found. Due to the typical sparsity of the community connectivity, the number of boundary nodes will be much less than the total number of nodes, $|\mathbf{B}| \ll N$.

The algorithm proposed here iterates through the boundary node set and performs a set of independent truncated random walkers originating at each boundary node, until convergence is reached in the distribution of visits to the nodes in the vicinity of each boundary node. It is the counts of the visits from the random walkers to the boundary nodes and nodes of the same community in their vicinity which allows a ranking in terms of being able to influence another community by spread of content. The description of the algorithm is summarized in steps 1 to 4 of the outline given in Table 1. In the first step of the algorithm the set of connected graphs is extracted from the original graph of the network, $\mathbf{G}$, using breadth first search (BFS). BFS can be performed with run time linear in terms of the number of edges and vertices, $O(N + M)$, and space complexity linear in terms of N. Given that most social networks will be small-world and scale-free, the number of edges will not grow too fast with the number of vertices. A relatively small subset of high degree nodes are responsible for the connectivity. The degree distribution following a power law [19] means that the majority of nodes will have a small number of edges. The second step is to label each node according to the community it belongs to and can be computed in $O(N)$. There is a wide selection of algorithms for obtaining the community structure, [5, 20]. In our work, we use the Louvain method of [21]. The run time of the Louvain algorithm is $\Omega(N \log N)$, and there is an efficient implementation available. Tests run with this method report working with millions of nodes under 2 minutes on a standard PC. It is

a greedy algorithm using the modularity index, [22], as an optimization criteria, which has the benefit that the number of iterations taken by the algorithm can be controlled to some extent by examining the value of change in the index per iteration. Other algorithms for community detection were experimented with and produced similar results in datasets where the community separation was clear. In situations where different approaches produce different community labellings the algorithm proceeds to focus on a different set of nodes which is likely to have a large amount of overlap. The third step extracts the set of boundary nodes which controls the information exchange between different communities as they are the only nodes that connect directly to nodes in a different community.

The final step in Table 1 runs a number of i.i.d. random walkers from each boundary node until a convergence criterion is satisfied based on the number of visits to the nodes in the network. The number of visits to each node is counted and is a measure of ability to disseminate information across boundary edges and influence different communities. Steps 3 and 4 are described in more detail in the next subsection. This algorithm can be referred to as the boundary vicinity algorithm (BVA).

2.1 Boundary node analysis

To obtain the boundary nodes/edges of a connected graph as defined in (2), we use the vector of community labellings for the set of nodes in the network **C** and look for adjacent nodes with different community labellings. With an edge list of the adjacency matrix of the graph, L, each edge is represented as a row number and a column number in this two column matrix. Where the two labels differ on a row in this edge list, a boundary edge has been detected;

$$\mathbf{W} = \left\{\mathbf{C}\big(L(s,1)\big) \neq \mathbf{C}\big(L(s,2)\big)\right\}. \tag{3}$$

The adjacency matrix of the community specific graph is the matrix $A_C(i,j)$, where

$$A_{C_l:i,j} = \begin{cases} 1, & A_{i,j} \times \delta(C_l(i), C_l(j)) = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Here the $\delta(\cdot,\cdot)$ is the Kronecker delta where the value of one is given when both inputs are equal. To obtain these matrices it is not a requirement to iterate through each element. This will be clarified below. With the community adjacency matrices for each community label A_C and the boundary nodes belonging to the network **B** we can iterate through the nodes of **B** and run the series of random walkers localized on each boundary node and confined to each A_C.

The random walks used to measure the ability for nodes to influence and affect the boundary nodes have a fixed number of steps. For the walks to represent the localized region of these nodes, **B**, the walks cannot be given an excessively large length as this would dilute the importance of nodes closer to the boundary nodes. The Barabási-Albert model [19, 23] uses the mechanism of preferential attachment to reproduce the growth characteristics of many networks. The average path length for these networks is $\log(N)/\log(\log(N))$, where we assume that the ceiling of the value is taken. We use this value as a baseline in deciding the number of random walk steps that must be taken before a piece of information loses the consistency and relevance of the original content. Various other values can

```
procedure BNV(G,walknum,stepnum)
    visitCounts = ← zeros(1,N)
    ids ← connected_components(G)
    for all id ∈ ids do
        G_i ← G(id, id)                                          ▷ G_i is a connected graph
    end for
    for all g ∈ G do
        (C, Q) ← community(g)                                    ▷ infer community structure
        if Q < threshold then
            continue
        end if
        W = g(i, j) × (1 − δ(C_i ≠ C_j))                         ▷ boundary edge list
        A_C =community_mask(g, C)
        for all node ∈ W do
            A = A_{node∈C}
            present_node = node
            w = 0
            walks = null
            while w < walknum do                                 ▷ i.i.d walkers
                pWeightsVec(node) = 1
                s = 0
                while s < stepnum do                             ▷ random walker
                    node = randomStateTransition(A,node)
                    pWeightsVec(node) = 1 + pWeightsVec(node)
                    s = s + 1
                end whilewalks = append(walks, pWeightsVec];
                w = w + 1
                if w =walknum then
                    psrf = GelmanRubinDiagnostic(walks);
                    if psrf < 0.95 — psrf> 1.05 then
                        w = 0                                    ▷ more walks for convergence
                    else
                        pWeights = scaleCommunityWeights(pWeightsVec)
                        visitCounts = append(visitCounts,pWeightsVec)
                    end if
                end if
            end while
        end for
    end for
    return visitCounts
end procedure
```

Figure 1 Boundary vicinity algorithm (BVA).

be used for the number of steps taken in the random walks. The method is not sensitive towards this value as long as the chains do not reach the stationary distribution where the initial state of the chain has not affected the final results, since we are interested in the locality of nodes surrounding the initial state which is one of the boundary nodes. Alternatives which worked well are the average path length, and the longest path length from the boundary node to another node in the same community.

Based on this idea, Figure 1 displays the pseudocode of the algorithm for measuring the boundary node proximities. The first input is the data structure for the connectivity of the nodes in the network. The second input is *walknum*, which is the number of random walkers that are started from each boundary node before convergence is tested. The third input, *stepnum*, is the number of steps/node traversals taken by each random walker. The vector *visitCounts* holds the number of visits by random walkers which are made to each node in the network throughout the algorithm. As a variation this vector could be made into a sparse matrix where each row is the contribution of an i.i.d. walker, so that more information from the walks can be found. The function call here, connected_components(G), is to the breadth-first-search algorithm which produces a list of the elements in G which are connected to each other. Using the identifiers of connected graph membership, a list of the

connected components is produced, **G**. In the loop of the elements of **G**, g is a connected network. Here the function call community(g), is to the community detection algorithm of choice (in this work the Louvain algorithm is used). This returns the community membership labels **C** for the networks and modularity index Q. If the modularity index is not larger than a certain threshold of choice, then it is considered that g has no evident community structure and therefore no boundary nodes and the loop continues to the next connected graph component g. The boundary edges, **W**, can be extracted by including only edges which connect different community labels. We obtain the adjacency matrices for each community label by including only the nodes in each community label in a separate adjacency matrix. For each unique *node* in the list of **W** the algorithm then proceeds through the standard method of a set of random walks on the adjacency matrix. After the completion of *walknum* number of walks, the trajectories are tested for convergence using the potential scale reduction factor (PSRF) of [24]. If the set of trajectories have not converged, more walkers are computed until the required amount of convergence is achieved. Upon convergence, the values counts of visits to each node by the walkers is normalized to remove the effects of more walks due to lack of convergence. The results are scaled according to the relative size of the community in relation to the whole graph because of the impact it may have on the large scale of the information flow, so that the nodes in larger communities deliver a larger impact than small ones.

The run time of this algorithm is dominated by the community detection phase. Due to the boundary node set being much smaller in size than the number of nodes, the loops required to iterate through them and perform the random walks will typically cost less than N.

Each component of the BVA algorithm, Table 1, can be made more efficient using parallelization methods. The first step requiring breadth first search can be parallelized using shared or distributed memory, following the works [25, 26], where the number of edges visited is significantly reduced. Another approach to BFS is [27] that utilizes the Nvidia GPUs, but the authors note the memory restrictions for large graphs that take up more than the graphics card memory of around 1 GB. When the number of nodes goes beyond a few million nodes and tens of millions of edges, memory becomes a concern and the method of [28] shows that the step of acquiring the set of connected components can be performed in log space. The community detection component can also be parallelized by using the method of [29] resulting in a completely parallelizable algorithm. The last steps of the algorithm can naturally be parallelized by running the i.i.d. random walkers on separate processors at the same time. After they have completed their walks the trajectories can then be monitored for convergence.

3 Results

Here the results of using the algorithm on synthetic datasets and a real dataset are shown. One synthetic network used for testing is a set of random Erdös-Rényi (ER) graphs produced and connected together to from a connected network by choosing randomly members to act as boundary nodes, shown in Figure 2. The other synthetic network is produced from connecting independent communities graphs produced using preferential attachment, shown in Figure 3. These synthetic datasets are used because the results of using them are easy interpret and compare when using BVA and betweenness. The well known Zachary Karate club dataset [30] is analyzed and presented in Figure 4. The Enron email

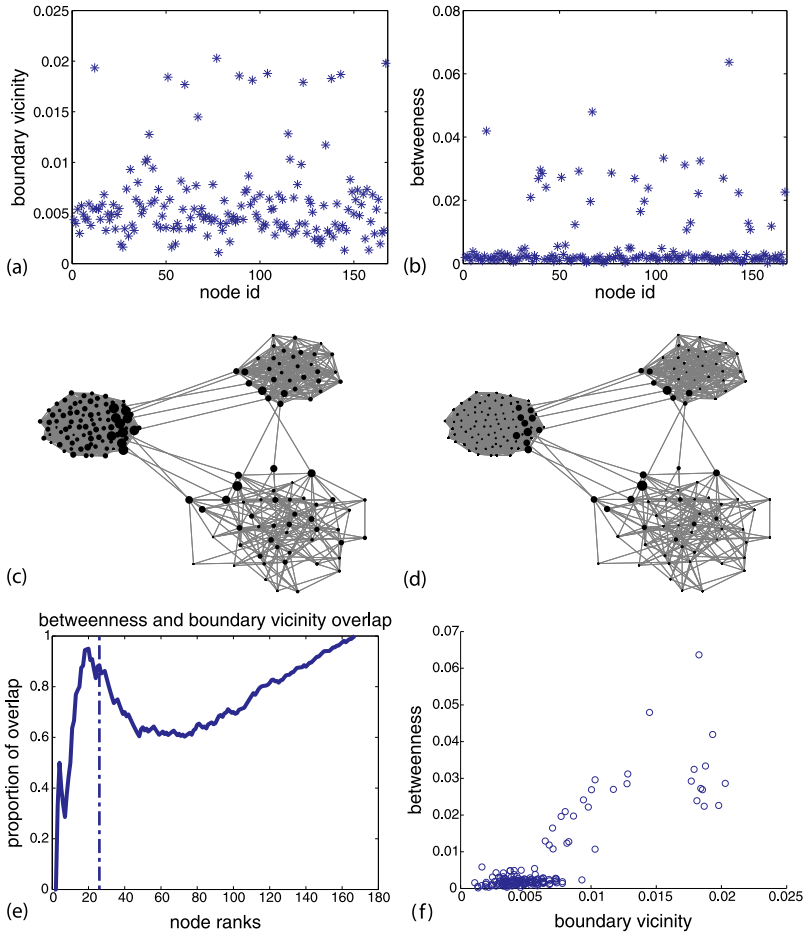

Figure 2 3 ER communities connected and analysed with BVA and betweenness. The first row of plots show the scores given to the node IDs with the boundary vicinity algorithm and betweenness respectively. The second row of plots is the network visualizations with the nodes scaled in size according the normalized scores from the boundary vicinity algorithm and betweenness respectively. Subplot **(e)** shows the proportion of overlap between the top ranking nodes from BVA and betweenness for different sizes of the ranking size. Subplot **(f)** is a scatter plot of the BVA and betweennes values for each node.

dataset [31] is also analysed utilizing the valuable semantic data associated with the nodes to show the qualitative validity of the algorithm. Lastly two new datasets collected from monitoring Twitter hashtags are presented where the volume of Tweets and the volume of boundary nodes are presented against a random set.

Figure 2 shows the results of using the boundary vicinity algorithm (BVA) and calculating betweenness on a synthetically produced network. Three communities were generated independently with the ER model and then a set of random nodes (26 here) were selected from these communities to be connected to a different community. These selected nodes become the *boundary nodes* in the network. There are 167 nodes, and the three communities have 87, 47 and 33 nodes with a total of 13 bridges between them. The chains of random walkers used were run until the convergence diagnostic of PSRF was below 1.2. There are six subfigures labelled (a)-(f), where (a) and (b) show the normalized values from the algorithms (y-axis) given to each node in the network (x-axis). Subfigure (a) for the boundary vicinity algorithm has a more evenly spread distribution across the nodes

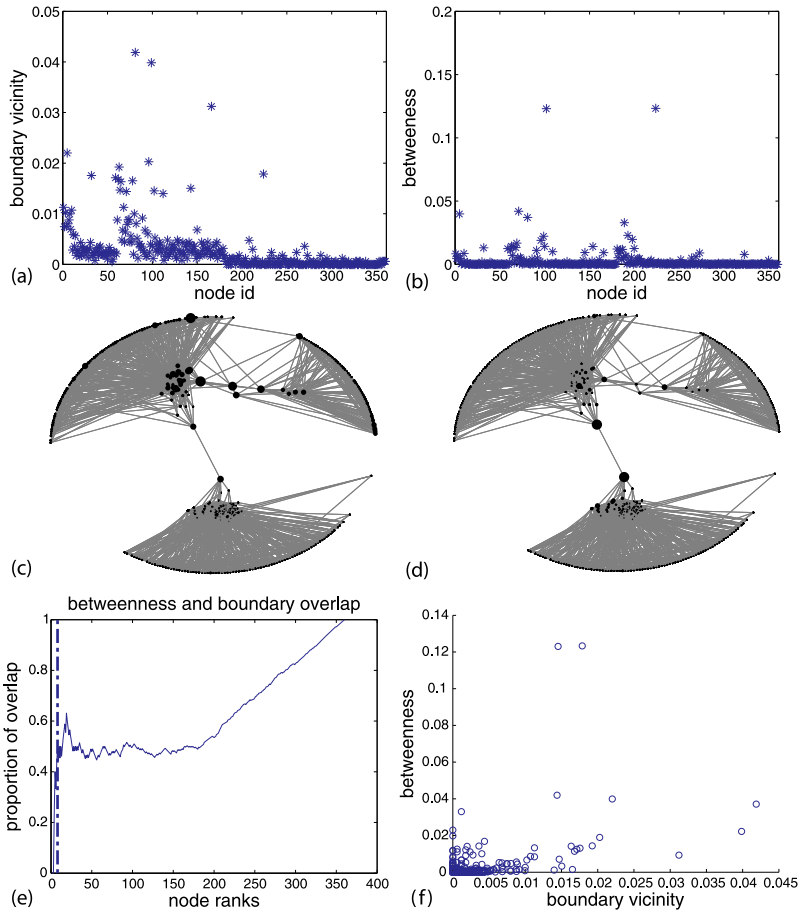

Figure 3 3 communities produced with preferential attachment connected and analysed with BVA and betweenness. The first row of plots show the scores given to the node IDs with the boundary vicinity algorithm and betweenness respectively. The second row of plots is the network visualizations with the nodes scaled in size according the normalized scores from the boundary vicinity algorithm and betweenness respectively. Subplot **(e)** shows the proportion of overlap between the top ranking nodes from BVA and betweenness for different sizes of the ranking size. Subplot **(f)** is a scatter plot of the BVA and betweennes values for each node.

than what betweenness produces in subfigure (b). We can see that betweenness gives almost absolute importance to the nodes on the boundary with little emphasis for the nodes in the vicinity of those boundary nodes. Subfigures (c) and (d) display the networks with the vertices scaled according to the boundary vicinity measure and betweenness respectively. In (c) we can see the neighboring nodes of the boundary scaled as well. Subfigure (e) counts the proportion of overlap in the ranking between BVA and betweenness for an increasing number of nodes. We can see that both algorithms have almost complete overlap in choosing the top 26 nodes but differ in the order for the subsequent nodes. Subfigure (f) shows a scatter plot of the values for all the nodes with both algorithms. We can see how the top ranking nodes are clearly distinct from the bulk of the network and how BVA produces a greater variance for nodes not in the boundary set. These results are consistent with multiple runs, and alternative networks which varied the number of bridges connecting communities and the density of edges between nodes in a community.

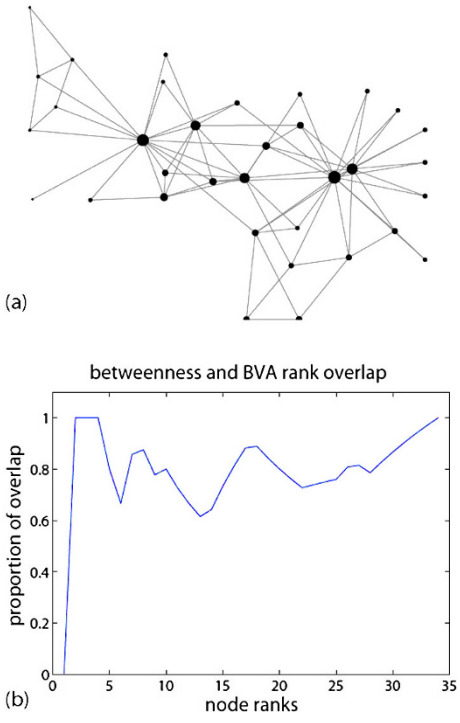

Figure 4 Using the boundary vicinity algorithm on the Zachary Karate club dataset. Subfigure **(a)** shows the network vertices scaled according to the normalized values given by BVA. Subfigure **(b)** shows the rank proportion overlap of BVA and betweenness for a number of different rank sizes.

Subfigure (a) in Figure 5 shows the results of simulating an S-I epidemic on the network of three connected ER communities. Each simulation begins where a single node is put into the infected state and each infected node can infect the nodes in its locality of a single edge according to the adjacency matrix. Three hundred independent simulations are run for each different configuration of the transmission probability and the average percentage of the network that is infected at each iteration is shown. In the first plot the black line shows the results where the transmission probability is uniform across all nodes, and is 0.2 in this case. The blue and red lines show where the top ten BVA and betweenness scoring nodes are removed/immunized from the spread of the infection. The rate of network infection is reduced in both cases showing that both scores provide useful targets for limiting spread. The second plot shows a slightly different strategy where for the blue and red lines, instead of removing the top ten nodes based on BVA and betweenness, their probability to move from susceptible to the infected state is 0.01 compared to the rest of the nodes with probability 0.2. The black line here is still the case of the uniform 0.2 probability used across the network.

In Figure 3 3 communities are produced using the Barabási-Albert model [19, 23] algorithm of preferential attachment and then these communities are connected by choosing nodes uniformly from each group. There are 360 nodes, 3 communities of 60, 120, and 180 nodes with a total of 13 bridges. When BVA is run the chains of random walkers that begin from the boundary nodes were run until the convergence diagnostic of PSRF was below 1.2. The same format as with the previous figure is used. In the first row of subplots, (a) and (b), we can see again that there is a wider distribution in the scores for the nodes with the BVA algorithm on non-boundary nodes. In subfigures (c) and (d) we visualize the networks with the nodes scaled according to the BVA and betweenness respectively. We can see that the highest degree nodes which are central to the community they belong to

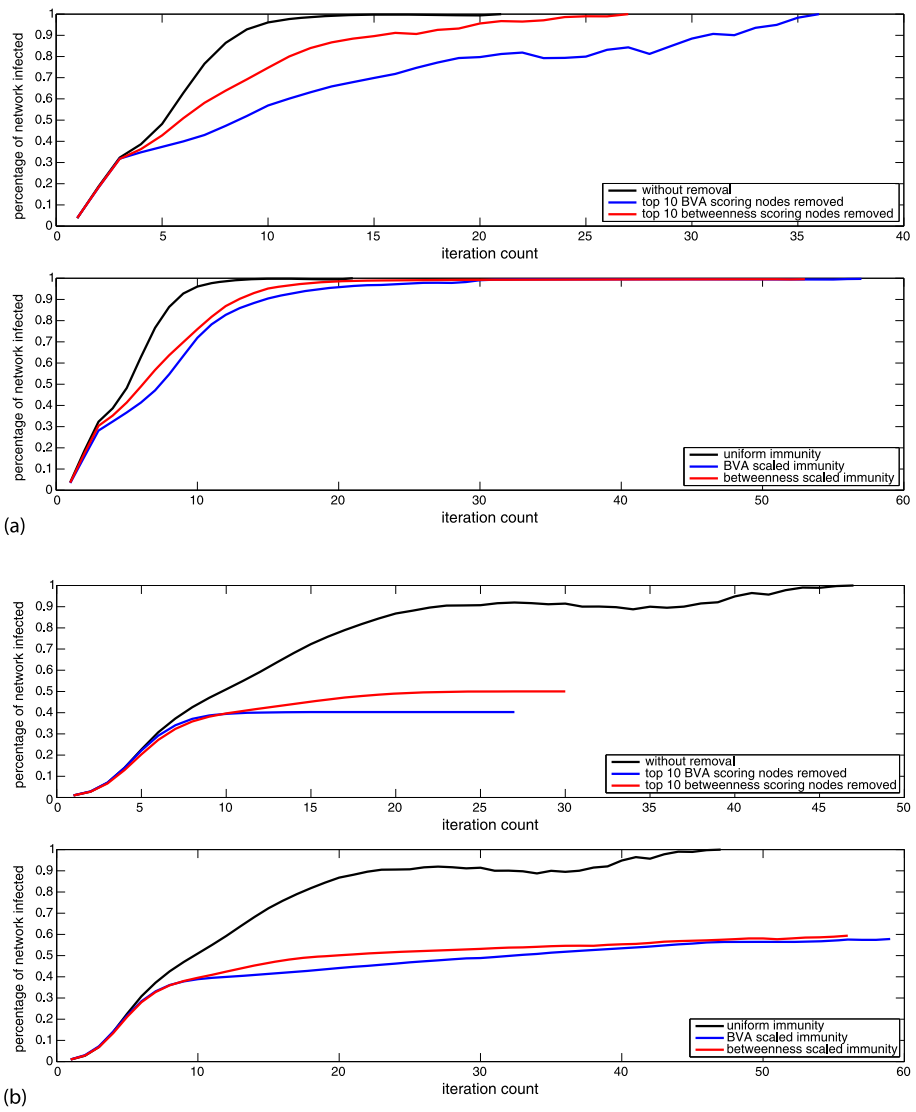

Figure 5 An S-I epidemic simulated on the networks of 3 connected communities and the Zachary Karate club. We test the ability of BVA and betweenness to prevent spread in an S-I epidemic. A single node at random is chosen as the initial infected node and the simulation is run until all nodes in the network are infected or there is no further spread after 100 iterations. The simulations are run 300 times and the average percentage of the network infected at each iteration is plotted. The results for each situation are shown up to the iteration number there was on average a increase in the network infected where the maximum permitted was 60. In subfigure **(a)** the graph of Figure 2 is used for the S-I simulation. The first subplot shows the results of the simulations using a uniform transmission probability of 0.2 in a black line, in blue the results of removing the top 10 nodes which BVA gave the highest score to, and in red the results of removing the top 10 nodes which betweenness produced. In the second subplot in black is the results of using a uniform transmission probability of 0.2, the blue line the results of giving only the top 10 BVA nodes a probability of 0.01 of becoming infected, and the red line where the top 10 betweenness nodes become infected with 0.01. We can see in both plots that BVA and betweenness target nodes which assist slowing down the spread. In subfigure **(b)** we see the results of the same simulations using the network of communities shown in Figure 3 produced using preferential attachment. With the case of removal both BVA and betweenness significantly reduce the spread and restrict the ability of the spread through the whole network. In the last subplot the rate of infecting the whole network is also significantly reduced although the simulation do not terminate as with the situation of removal. Subfigure **(c)** shows the epidemic simulation using the Zachary Karate club network presented in Figure 4 with nodes scaled according to the BVA score. The plot shows the results of giving the top 3 BVA and betweenness scored nodes the same reduced probability of infection. The reduced rate of infection is visible for this network as well.

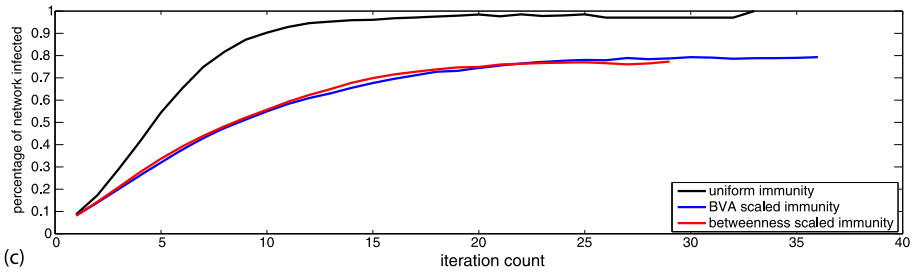

Figure 5 Continued

are scaled and highlighted in both cases. A critical difference is that the boundary nodes at the top which receive a large score with BVA but are given minimal importance with betweenness. With betweenness the role of these nodes is redundant given alternative routes through nodes with higher degree and direct connections to many nodes in the community. In the effort to inspire cross pollination of communities with promoted content, the ability to saturate a user with fewer connections may be advantageous, and worth considering because they may be influenced more easily. In (e) we look at the overlap proportion of the ranking between nodes for a number of nodes in both algorithms. We can see the local peak of the number of overlaps for more nodes than the number of boundary nodes. This is because the structure of the network includes nodes in the vicinity of the boundary which lay on the shortest paths to other nodes in the community. In the last subfigure we can see the scatter plot of the BVA values and betweenness. The ranking of the algorithms may be more similar to each other than with the ER communities connected but the distribution is much more narrow for betweenness in this case, highlighting the few boundary nodes that are also core to the communities.

Subfigure (b) in Figure 5 shows the results of simulating an S-I epidemic on the network of three connected ER communities. Each simulation begins where a single node is put into the infected state and each infected node can infect the nodes in its locality of a single edge according to the adjacency matrix. Three hundred independent simulations are run for each different configuration of the transmission probability and the average percentage of the network that is infected at each iteration is shown. The percentage of the infected network is stops where simulations no longer continued infecting new nodes on average. The maximum permitted iteration number for each simulation was set at 60. In the first plot the black line shows the results where the transmission probability is uniform across all nodes, and is 0.2 in this case. The blue and red lines show where the top ten BVA and betweenness scoring nodes are removed/immunized from the spread of the infection. The rate of network infection is reduced in both cases showing that both scores provide useful targets for limiting the spread. Since the communities had a very sparse interconnectivity the removal restricted the between community spread limiting the number of infected nodes. The second plot shows a slightly different strategy where for the blue and red lines, instead of removing the top ten nodes based on BVA and betweenness, their probability to move from susceptible to the infected state is 0.01 compared to the rest of the nodes with probability 0.2. The black line here is still the case of the uniform 0.2 probability used across the network. The rate of transmission is significantly reduced from the uniform case and even more than the results on the ER graph in the first subfigure since the community connectivity relies on few edges.

Figure 4 shows the results of using BVA on the Zachary Karate club dataset. In subplot (a) the network is visualized and the vertices are scaled according to normalized scores given by the BVA algorithm. The central members of the communities are given large values as are the boundary nodes since they are within the vicinity of the boundary. In subfigure (b) the overlap of the rankings with BVA and betweenness is shown for the number of nodes included, and as with the previous two figures the overlap for both methods peaks when including the top number of nodes which corresponds to the number of boundary nodes. In Figure 5, subfigure (c) shows the results of simulating an S-I epidemic on this network. The case of removal of top scoring BVA and betweenness nodes is not presented due to the size of the network. Here only the top 3 scoring nodes for BVA and betweenness are given the reduced probability of infection 0.01. As with the simulations on the other networks, BVA and betweenness target nodes which reduce the rate of infection.

When analyzing the Enron email dataset, a subset of the nodes are included where the position in the company is known. BVA and betweenness scores are calculated for each of the nodes in the network and the top ten nodes for which their roles are known are compared. BVA selects 3 vice presidents, 1 CEO, 2 managers, 2 traders, and 2 employees to be in the top ten. Betweenness selects 1 vice president, 1 managing director, 2 managers, 1 director of trading, 2 traders, 1 secretary and 2 employees. The list provided by BVA contains more company members with higher positions than by betweenness. This may not be always the case, but it does show that the features of the network extracted by BVA captures importance in the node placements.

Figure 6 shows the Twitter activity of a TV show *FearneHolly* which is monitored in real time using the paid Twitter API service that does not deliver such a limited subset of the Tweets being sent regarding the hashtag, as does the free service. We look at the Tweet volumes over time for this topic and plot them in the bottom subplot of the figure. We can see a single dominating conversation intensity spike. We wish to see the activity of the boundary node set, $\mathbf{W}$, according to the large conversation volumes. BVA focuses on the boundary nodes since it supposes that they are key in facilitating the productions of these large spikes of conversation activity in Twitter. Since these dominant spikes observed in Twitter stand out so much from smaller oscillations it can be assumed that it is due to the conversation taking place across the entire network of different communities and not confined to the locality of clustered nodes in a single community. The boundary nodes of the network are found, and over time the number is counted at each time point, shown in the blue line in the first subplot. We see a single dominant spike for the number of boundary nodes in the same region as that for the total conversation intensity in the bottom subplot. There is a need to test whether the boundary node increase at the same time as the total volume indicates that they provided vital routes for content to spread or is their number only a consequence of the overall activity of the nodes uniformly over the network and not in any way dependent on the presence of the boundary nodes. To investigate this, we select a random set of nodes in the network of equal size to that of the boundary node set and look for spikes in the volume of communication in this random set. A threshold is set at a standard deviation above the mean Tweet count per minute for each subplot and is shown as a dashed black line. Using a confusion matrix gives us a table of values for the false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN) for a predictor. A confusion matrix for the boundary node set activity as an indicator of the

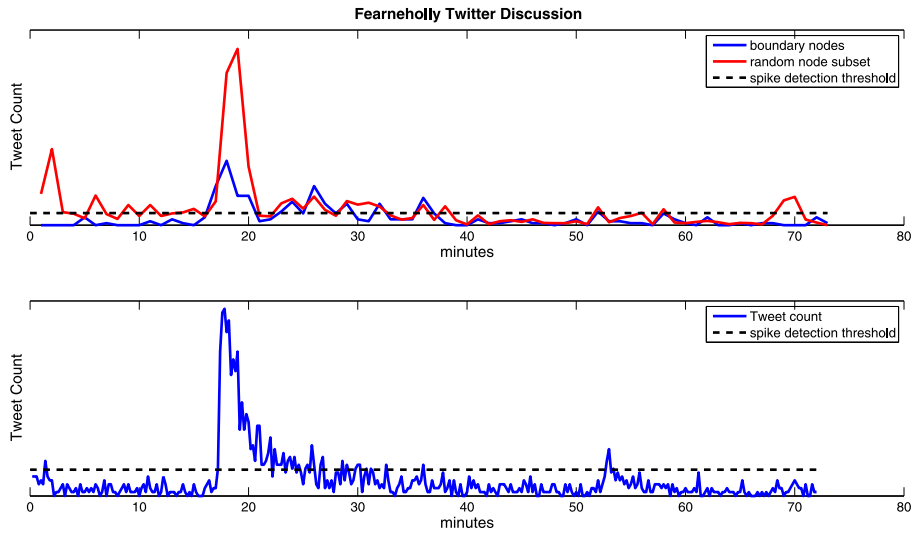

Figure 6 Boundary node detection over time from real time monitoring of events in FearneHolly Twitter discussion. The Twitter activity of a TV show, FearneHolly, is monitored and the Tweets are gathered. The top plot shows the number of boundary nodes that produce a Tweet between time points in blue, and in red is shown the number of Tweets from a random set of nodes of equal size to the boundary node set of the Twitter network. In the bottom plot is the total volume of Tweets over time and we see a single dominant spike in the conversation activity with a decay trailing afterwards. The black dashed line in each subplot is a threshold for spike detection which is one standard deviation from the average Tweet count over time. We can see that the boundary nodes produce a single dominant spike mirroring that of the total conversation activity. The random set of nodes is selected to see whether the boundary node activity simply reflects the total number of Tweets, but we can see a spike in the activity occurs at the start of the monitoring which is not present in either of the other trajectories.

total Tweet volume is calculated:

$$\begin{array}{lcc} & \text{Boundary Spike} & \text{No Boundary Spike} \\ \text{Volume Spike} & 7 & 3 \\ \text{No Volume Spike} & 5 & 58 \end{array}. \tag{5}$$

For the randomly selected group of nodes of equal size to the boundary nodes set the confusion matrix is:

$$\begin{array}{lcc} & \text{Random Group Spike} & \text{No Random Group Spike} \\ \text{Volume Spike} & 8 & 2 \\ \text{No Volume Spike} & 16 & 47 \end{array}. \tag{6}$$

Given the scope of the application BVA the precision of the indicator is the most relevant measure to compare from the confusion matrices. For the boundary node set the precision ($TP/(TP+FP)$) for this dataset is 0.58 and for the randomly selected group of nodes 0.33 (two significant figures given). The boundary node vicinity algorithm focusing on the boundary node activity is working with a subset of the network which is appears to have value during the intense information exchange events (the dataset can be provided by contacting the author).

We look at the Tweets gathered from another TV show *Got to Dance*, shown in Figure 7, in a similar way to the previous *FearneHolly* example. A confusion matrix for the boundary

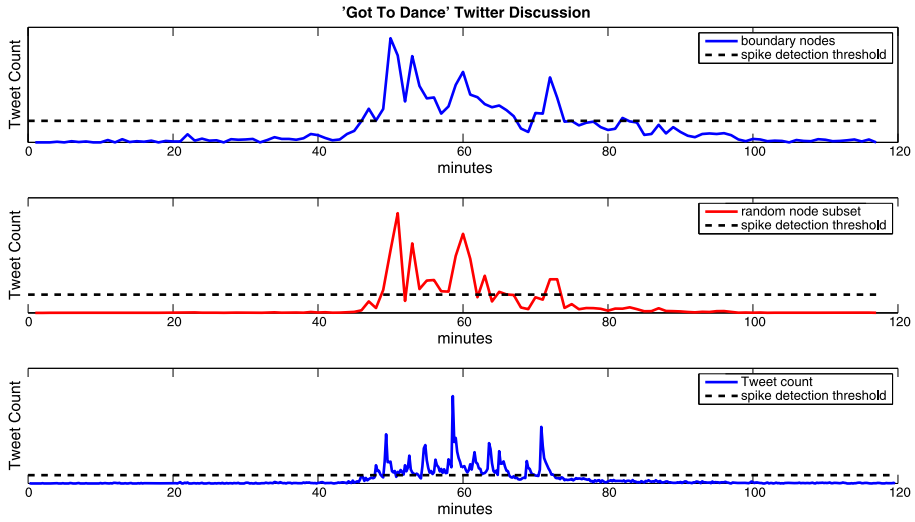

Figure 7 Boundary node detection over time from real time monitoring of events in the 'Got to Dance' Twitter discussion. The Twitter activity of a TV show is monitored and the Tweets are gathered. The first subplot shows the number of boundary nodes that produce a Tweet, and in the second subplot is shown the number of Tweets from a random set of nodes of equal size to the boundary node set taken from this Twitter network. In the bottom plot is the total volume of Tweets over time, and we see a time frame of intense conversation activity. The dashed lines in black are a threshold for the number of Tweets in a minute that are considered to be a spike and is a standard deviation above the mean Tweet count. The boundary nodes as well as the rest of the network are taking part in the conversation with the ability for information exchange between communities.

node set activity as an indicator for the total Tweet volume is calculated:

$$\begin{array}{l} \\ \text{Volume Spike} \\ \text{No Volume Spike} \end{array} \begin{array}{c} \begin{array}{cc} \text{Boundary Spike} & \text{No Boundary Spike} \end{array} \\ \begin{pmatrix} 18 & 1 \\ 8 & 90 \end{pmatrix}. \end{array} \tag{7}$$

For the randomly selected group of nodes of equal size to the boundary nodes set the confusion matrix is:

$$\begin{array}{l} \\ \text{Volume Spike} \\ \text{No Volume Spike} \end{array} \begin{array}{c} \begin{array}{cc} \text{Random Group Spike} & \text{No Random Group Spike} \end{array} \\ \begin{pmatrix} 12 & 7 \\ 6 & 92 \end{pmatrix}. \end{array} \tag{8}$$

The precision values are 0.69 and 0.67 for the boundary nodes and random node subset respectively. Both network subsets show substantial alignment with the conversation spikes in the total volume count. The added value of looking at the boundary nodes in this type of situation is to exploit the unique positioning for efficient spreading between communities.

4 Discussion

The work presented here gives an efficient algorithm for ranking the ability of nodes in a network, with community structure, to spread information between clusters. Previously proposed methods impose large computational difficulties or are not based on principles which realistically model how information across the communities can spread. Focusing

attention on these boundary nodes in a network can be critical for monitoring whether content may reach the point of becoming *viral*. In practice not all of the nodes in the network may be directly influenceable. An alternative approach can be to indirectly influence a chosen node by targeting the local vicinity of the node in the network. The boundary vicinity algorithm (BVA) acknowledges nodes that may be placed in such a position to have more or less influence on content leaving or entering a community of nodes in network.

A strength of this boundary vicinity algorithm is that it combines the power of community detection algorithms with the use of random walkers to assist in the process of investigating the range of influence of the boundary nodes. The results show that this algorithm is comparable with betweenness centrality without the requirement for full the maturity of a network to be visible. In situations where the observed connectivity is changing, analysing the network in sections based on a community structure is an approach to provide more consistent results over time. The algorithm has a single tuning parameter which determines the number of steps a random walker takes from the boundary nodes. Using a fraction of the average path length for networks constructed with the Barabási-Albert model has given stable results in our experiments.

Measures such as betweenness can provide a set of optimal targets for spreading content along shortest path routes throughout the complete connected network. This task ignores the challenges that might be faced which attempting to promote activity in the critical set of nodes which lay on the boundary of the communities making up the complete network. A list of the nodes which are best positioned to quickly spread a piece of content does not address many of the practical challenges in inspiring activity as a non-invasive influencer. Assessing the vicinity of the influencers for the boundary nodes gives a reasonable subset for which attention must be given to ensure that cross pollination between clustered sets of nodes can occur.

Overall, the proposed algorithm has the potential to quickly handle the task of analysis with an online stream of large datasets. In particular real time event monitoring in environments such as Twitter where topic discussions can grow and decay rapidly, this is especially important. With the goal of spreading the content as far as possible the boundary nodes, and those nodes in its close vicinity, in a community must be targeted, which is at the core of this method proposed here.

Competing interests
The author declares that they have no competing interests.

Acknowledgements
Thanks is given to Peter Laflin for providing feedback over the course of this work, and to Bloom Agency, Leeds, for supplying anonymised Twitter data. This work was performed as part of the Mathematics of Large Technological Evolving Networks (MOLTEN) project, which is supported by the Engineering and Physical Sciences Research Council and the Research Councils UK Digital Economy programme, with grant EP/I016058/1, and the support of the University of Strathclyde with Bloom Agency for the follow-on support from the Impact Acceleration Account.

Received: 13 January 2014 Accepted: 17 September 2014 Published online: 22 October 2014

References

1. Skeels MM, Grudin J (2009) When social networks cross boundaries: a case study of workplace use of Facebook and LinkedIn. In: Proceedings of the ACM 2009 international conference on supporting group work. ACM, New York, pp 95-104
2. De Domenico M, Lima A, Mougel P, Musolesi M (2013) The anatomy of a scientific gossip. arXiv:1301.2952
3. McAlexander JH, Schouten JW, Koenig HF (2002) Building brand community. J Mark 66:38-54
4. Leskovec J, Lang KJ, Dasgupta A, Mahoney MW (2009) Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math 6(1):29-123

5. Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75-174
6. Ferrara E (2012) A large-scale community structure analysis in Facebook. EPJ Data Sci 1(1):1-30
7. McPherson M, Smith-Lovin L, Cook JM (2001) Birds of a feather: homophily in social networks. Annu Rev Sociol 27:415-444
8. Matsuo Y, Yamamoto H (2009) Community gravity: measuring bidirectional effects by trust and rating on online social networks. In: Proceedings of the 18th international conference on World Wide Web. ACM, New York, pp 751-760
9. Travers J, Milgram S (1969) An experimental study of the small world problem. Sociometry 32:425-443
10. Newman ME, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113
11. Clune J, Mouret J-B, Lipson H (2013) The evolutionary origins of modularity. Proc R Soc Lond B, Biol Sci 280(1755):20122863
12. Subramani MR, Rajagopalan B (2003) Knowledge-sharing and influence in online social networks via viral marketing. Commun ACM 46(12):300-307
13. Kwak H, Lee C, Park H, Moon S (2010) What is Twitter, a social network or a news media? In: Proceedings of the 19th international conference on World Wide Web. ACM, New York, pp 591-600
14. Weng J, Lee B-S (2011) Event detection in Twitter. In: ICWSM
15. Nichols J, Mahmud J, Drews C (2012) Summarizing sporting events using Twitter. In: Proceedings of the 2012 ACM international conference on intelligent user interfaces. ACM, New York, pp 189-198
16. Freeman LC (1977) A set of measures of centrality based on betweenness. Sociometry 40:35-41
17. Brandes U (2001) A faster algorithm for betweenness centrality. J Math Sociol 25(2):163-177
18. Newman ME (2005) A measure of betweenness centrality based on random walks. Soc Netw 27(1):39-54
19. Barabási A-L, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509-512
20. Lancichinetti A, Fortunato S (2009) Community detection algorithms: a comparative analysis. Phys Rev E 80(5):056117
21. Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):P10008
22. Clauset A, Newman ME, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70(6):066111
23. Albert R, Barabási A-L (2002) Statistical mechanics of complex networks. Rev Mod Phys 74(1):47
24. Gelman A, Rubin DB (1992) Inference from iterative simulation using multiple sequences. Stat Sci 7:457-472
25. Beamer S, Asanović K, Patterson D (2013) Direction-optimizing breadth-first search. Sci Program 21(3):137-148
26. Beamer S, Buluc A, Asanovi K, Patterson DA (2013) Distributed memory breadth-first search revisited: enabling bottom-up search. Technical report, DTIC document
27. Harish P, Narayanan PJ (2007) Accelerating large graph algorithms on the GPU using CUDA. In: High performance computing—HiPC 2007. Springer, Berlin, pp 197-208
28. Reingold O (2008) Undirected connectivity in log-space. J ACM 55(4):17
29. Martelot El, Hankin C (2013) Fast multi-scale community detection based on local criteria within a multi-threaded algorithm. arXiv:1301.0955
30. Zachary WW (1977) An information flow model for conflict and fission in small groups. J Anthropol Res 33:452-473
31. Chapanond A, Krishnamoorthy MS, Yener B (2005) Graph theoretic and spectral analysis of Enron email data. Comput Math Organ Theory 11(3):265-281

doi:10.1140/epjds/s13688-014-0026-9
Cite this article as: Mantzaris: **Uncovering nodes that spread information between communities in social networks.** *EPJ Data Science* 2014 **3**:26.