Tracking Local Community Evolution

Master Thesis Computer Science
December 2009

Institute for Computing and Information Sciences
Radboud University
Nijmegen, The Netherlands

Author: F.T.W. (Frank) Koopmans
Email: ftwkoopmans@gmail.com
Thesis Number: 617
Supervisor: Prof. dr. ir. Th.P. (Theo) van der Weide
Second Corrector: Prof. dr. ir. W. (Wessel) Kraaij

Abstract. We are interested in observing changes to a particular web graph community over time. A small subset of the entire web graph can provide sufficient information about our community of interest. Therefore we use a local approach where knowledge of the entire web graph is not needed. In this paper we will consider how we may track the evolution of a community from a local point of view and analyze the quality of our model. We use local modularity as our community definition of choice and propose an improvement thereof. An incremental definition is coined such that we may analyze the consequences of elementary modification of a community. We analyze the quality of a local community as constructed using this incremental definition. Therefore we propose a probability model that we demonstrate on omission errors for the incremental algorithm. **Keywords:** Web graph; community; tracking; evolution; local modularity

1 Introduction

The internet is a topic of interest to many. It is ever growing, evolving and the number of users is increasing.

Advertisers may try to reach their target audience on the internet in order to sell a product. But they may also try to observe their target audience and observe their interest. A shift of interest may indicate a trend amongst a group of people, opportunities arise if one can spot these in an early stage.

Finding communities within a graph is receiving much attention from various academic disciplines. Physicists may be interested in representing the patterns of interactions in complex systems as networks [11]. In the field of biology, community structure in graphs is used for research on the structure of bio-molecular networks [12]. And computer scientists study internet phenomena using community structure within the web graph [10].

We are interested in observing and tracking a target audience on the internet. Assuming we can identify this group of people as the visitors of websites in a web graph community, we would like to observe their shifting interest by observing changes to a community. For example, such mutations may be used for trend analysis.

The entire web graph is huge [1] and if we are only interested in a particular community we would rather use a local approach where we only have knowledge about the nodes in our community of interest.

In order to do an analysis of local community evolution we first consider the definition of a local community. Then we consider two techniques for modeling community evolution in section 3. In section 4 we use local modularity as introduced by Clauset [3] as our community definition of choice and propose an improvement thereof. This will provide understanding and insight in the structure and measured quality of a local community.

In section 5 we coin an incremental definition that allows us to consider the impact of the addition of a single node to a community from a local point of view. This allows us to observe small step evolution of a local community and reason about elementary modifications.

Using a local view on the web graph to observe local communities may lead to a lack of information about nearby nodes that may be, or become, relevant to that community. In section 6, we propose two methods for grouping nodes outside a community by their relevance to that community in order to gain more relevant contextual information.

In section 6 we also analyze the presence of errors in a local community model compared to the actual web graph community. This is done by considering the probability that a node outside a local community should be a member of that community. In the last section we give some conclusions.

2 Community Definition

The internet contains a lot of websites that are somehow related. Websites can have a relation by a common topic or interest. Examples of such a relation are a direct link to each other, being a member of the same web ring or both be referenced to by another site. The web graph preserves these relations, so any community on the internet should be in the web graph too.

Web graph communities can be loosely defined as a collection of nodes that are somehow related. A community in the web graph is a subsets of nodes which are more densely linked when compared to the rest of the graph. Despite the lack of information about the contents of a node within the web graph, communities are quite accurately grouped by a particular topic or theme [6].

Visitors of any website within a community are very likely to be interested in other websites of the same community due to the shared topic. The interest of a visitor in a certain topic relates it to a community. While there is no explicit link between the visitor of a website and a node in the web graph, we can say that each community in the web graph has an *implicit target audience*. A community within the web graph does not only project websites and links, it implicitly reflects the interest of people on a topic.

The implicit relation between a website and its visitors is also expressed by the way they influence each other. One can question if it is the website who influences the visitors and thereby changes their interest, or do the visitors have a somewhat rigid interest such that websites adapt accordingly to be more successful? We assume the answer lies in the middle since both actors will have at least some influence on each other.

The web graph is exceedingly sparse [13]. Being mostly read and written by humans, typical vertices have a relative low degree when compared to the amount possible if the web graph as a whole was dense. There are, of course, variations in the density within the web graph and some specific nodes may have a huge degree.

2.1 Community properties

We consider the web graph as a directed graph and assume that each node knows its outgoing edges. There is an edge from node u to node v if and only if the corresponding web pages are not the same and there is a hyperlink from the website corresponding to u to that of v. A node may have incoming edges but it is not aware of them, just like web pages do not know by what hyperlinks they are referred to. Consequently, there are no point cycles. Also there are no multiple edges between nodes.

We will refer to G as the entire web graph from now on. We define $G.N$ as its set of nodes and $G.E$ as its set of edges. We use $arcs(X, Y)$ for the set of edges from some node in X to some node in Y.

A graph can also be regarded as an *adjacency matrix*. This is a $n \times n$ matrix where n is the number of nodes in the graph. In this matrix we can map the edges between nodes such that the value A_{vu} is the number of edges from node u to node v. On the diagonal axis of the matrix one can see the number of self references, A_{vv} illustrates the number of edges from node v to node v. In our model, the latter will always be 0 because point cycles are not allowed.

2.2 Community identification

From a graph theoretic point of view a community can be regarded as a cluster within the web graph with dense linkage between nodes within the community and sparse density outside the community. This notion builds a bridge to a vast resource of knowledge on the topic of graph theory. The process of community identification can be regarded as a simplification of general graph clustering since one only needs to identify a group of nodes which is similar to a given seed node [6].

There are many approaches to defining and identifying graph communities. There is much interest in the topic from various scientific disciplines and many different approaches to community identification have been introduced [13] [7].

Because there are many different applications for community identification, a lot of algorithms have been developed that may suit specific properties. Some of these require a priori knowledge of some expected community properties in the graph in order to function well, such as the algorithm proposed by Wu & Huberman [14]. However, in most complex real world networks we have no knowledge about how many communities we wish to discover.

In general, most methods for community identification have their own niche in which they operate. For example, the specialty of an algorithm may be a specific type of network or a focus on computational complexity.

More specifically, finding a community is the same as looking for all members of the cluster that also contains a given seed node. Community identification algorithms are easily converted to clustering algorithms and vice-versa, so there is no significant theoretical difference. Research has shown that regardless of the algorithm and data source, the quality of clustering can be measured in various ways [7] and no single measure for cluster quality is perfect [8].

The computational complexity of community identification is vast because clustering algorithms generally require the analysis of every node in the web graph. Instead of trying to identify the optimal community one could make a tradeoff between quality and computational complexity in order to obtain an acceptable result. While searching for clusters in the web graph one can use distinct web graph properties that may not be present in a generic graph. For an overview of different approaches to community identification considering both quality and computational complexity refer to [15].

Modularity *Graph modularity* defines a community as the collection of nodes that have more links between them than to nodes outside the community [9]. Thus such a community is separated, distinct, from the rest of the graph but (most likely) not entirely disconnected.

Newman & Girvan proposed a modularity optimization method that indicates a good division of an undirected graph into communities [4]. Their approach to finding communities requires total knowledge of a graph. The modularity measure defines a good graph partitioning such that there is a maximum of edges within communities and a minimum of edges between nodes from different communities. This measure also compares the division of the graph into communities with a random graph in order to measure if the partitioning is statistically surprising. The modularity measure is intuitively defined as:

$$\begin{aligned} Q = {} & \text{(fraction of edges within communities)} \\ & \text{- (expected fraction of such edges)} \end{aligned}$$

where higher values for Q indicate when a statistically surprising fraction of the edges in a network fall within the chosen communities.

Several notions of random graphs have been introduced, see for example [16] for a brief overview. In the context of social networking the *configuration model* has obtained a central position. In this model random graphs are generated that are like a given graph such that they have the same set of nodes and for each node the same degree as the given graph [16]. Two variants are used, the undirected and the directed variant.

Let graph G be represented by an adjacency matrix where A_{ij} yields 1 if there is an edge from j to i and zero otherwise. The total amount of edges in the graph is defined by m. Let k_i^{in} be the indegree of node i in this graph, k_i^{out} the outdegree and the degree $k_i = k_i^{in} + k_i^{out}$.

Then, according to the configuration model, the probability of a connection between nodes i and j in an undirected graph is proportional to both k_i and k_j such that $p_{ij} = a.k_i.k_j$. For an undirected graph, we consider each edge twice since both the begin- and end-node of each edge add the edge to their degree. Thus the sum of the degree of all nodes equals $2m$. Then the total number of $2m$ connections in the graph will be equal to:

Lemma 1.

$$\sum_{ij} p_{ij} = \sum_{ij} a.k_i.k_j = a(2m)^2$$

Consequently, $a = \frac{1}{2m}$.

We can now derive the probability of an edge between nodes i and j as:

Lemma 2.

$$p_{ij} = \frac{k_i k_j}{2m}$$

where one considers the chance for these two nodes to be connected given a random distribution of their edges over m nodes.

Since graph modularity considers the amount of inter community connections the Kronecker delta δ_{ij} may be used to define if both nodes are in the same community. Let ζ be a clustering of G and let $[i] \in \zeta$ be the cluster assigned to node i. Then we introduce:

Lemma 3.

$$\delta_{ij} := [i] == [j]$$

So if nodes i and j are in the same community δ_{ij} yields 1 and zero otherwise.

Using the intuitive definition, probability p_{ij} and the Kronecker delta δ_{ij} we may derive Newman's modularity measure for undirected graphs:

Lemma 4. *Let ζ be a clustering of G.*

$$\begin{aligned} Q(\zeta) &= \textit{(fraction of edges within communities)} \\ &\quad \textit{- (expected fraction of such edges)} \\ &= \textstyle\sum_{C \in \zeta} [\textit{edges within } C \textit{ - expected amount}] \\ &= \tfrac{1}{2|G.N|} \textstyle\sum_{C \in \zeta} \left[|arcs(C,C)| - \sum_{ij \in C} p_{ij} \right] \\ &= \tfrac{1}{2m} \textstyle\sum_{C \in \zeta} \sum_{ij \in C} [A_{ij} - p_{ij}] \\ &= \tfrac{1}{2m} \textstyle\sum_{ij \in G.N} [A_{ij} - p_{ij}]\, \delta_{ij} \\ &= \tfrac{1}{2m} \textstyle\sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta_{ij} \end{aligned}$$

This measure can be used for identifying graph communities by finding the optimized value for Q for any possible graph partitioning. This process is quite exhaustive, local and greedy implementations optimize this process. Besides the identification of communities this measure may also be used as an evaluation for the quality of a graph partitioning while comparing various community identification algorithms [15].

However, this approach does not apply to directed graphs. An adapted measure of Q was later introduced by Leicht & Newman [5] such that Q can be used to consider the optimal community diversion of a directed graph using modularity.

For a directed graph one must consider that the edge direction has great influence on the expected degree distribution. Suppose there are two nodes, u and v. Node u has a high outdegree and a low indegree while v has the opposite degree distribution. So a given edge is more likely to run from u to v than vice versa. Thus if we observe such a degree distribution in a directed graph we should consider an edge from v to u statistically surprising and we should value its contribution to the modularity as such, according to Newman [5].

For directed graphs, as opposed to undirected graphs, we do not consider each edge twice. For each node we consider whether the connected edges are incoming or outgoing. We can now derive the probability of an edge between nodes i and j in a directed graph as:

Lemma 5.

$$p_{ij} = \frac{k_i^{in} k_j^{out}}{m}$$

Note that the degree distribution is normalized by m here, whereas it was normalized by $2m$ for undirected graphs as we have seen earlier.

Consequentially, the modularity measure for directed graphs by Newman amounts to:

Lemma 6.

$$Q = \frac{1}{m} \sum_{ij} \left[A_{ij} - \frac{k_i^{in} k_j^{out}}{m} \right] \delta_{ij}$$

where we can see that edges from j to i indeed make a larger contribution if k_i^{in} and/or k_j^{out} is small.

Local modularity Instead of finding an optimized division of the entire graph into communities, one may rather be interested in just one community that contains a node of special interest. While processing the entire web graph is quite exhaustive because of its size, a local approach that does not require knowledge and processing of the entire graph is desirable.

A *local modularity* measure was introduced by Clauset in the context of restricted knowledge of the graph [3]. Since this model only has local knowledge of the graph, the configuration model as used by Q cannot be used since the total amount of edges (m) is unknown.

Suppose we consider graph G where we have knowledge of a set of nodes and their edges that we consider to be a community which we denote as C. Since we are aware of all edges from C, we are aware of a set of nodes outside of C about which we know only their adjacencies to C. We will refer to this as the *universe* of C, denoted as $U(C)$. If we only consider the nodes in C that have at least one neighbor in $U(C)$, thus each node in this set contains at least one edge towards a node in $U(C)$, we obtain the *boundary* of C denoted as $B(C)$. See Figure 1.

In this local model we can only gain additional knowledge of the graph by visiting neighboring nodes of C, thus nodes in $U(C)$. Processing these nodes one at a time, we may add nodes to C and thereby obtain previously undiscovered nodes in the universe.

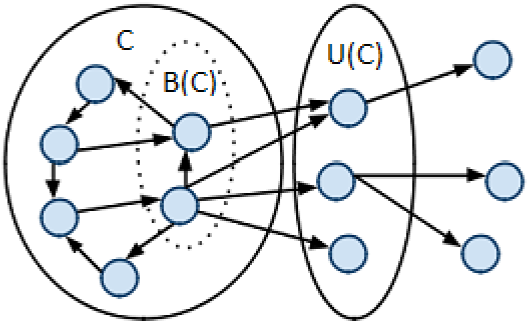

Fig. 1. An abstract graph with a community, its boundary and universe.

Clauset defined local modularity by considering all edges originating from the boundary and then calculating the fraction that is pointed at C internally [3]. This measure defines the sharpness of the boundary of a community and is independent of the size of the community, whereas the graph modularity measure Q is not. Intuitively, one would expect that a community with a sharp boundary will have few connections from its boundary to the unknown portion of the graph ($U(C)$), while having a greater proportion of connections from the boundary back into the local community (C).

As a result, the local modularity for community C is defined as:

$$R(C) = \frac{|arcs(B(C), C)|}{|arcs(B(C), G.N)|}$$

where the resulting local modularity, $R(C)$, lies on the interval $0 \leq R(C) \leq 1$. The local modularity in Figure 1 is $\frac{3}{7}$.

3 Modeling Community Evolution

We consider tracking a web graph community over time as observing mutations to a community as it evolves. Since the web graph and its communities evolve fast, we need to define a way to observe one and the same community and all changes to it over time. In this chapter we consider two models for observing the evolution of a community.

3.1 Snapshot model

The first model takes snapshots of the web graph at several moments in time and performs community identification within each dataset. We will refer to this as the *snapshot model*. The datasets are not explicitly linked or related in any way using this model. After acquiring snapshots of the web graph of the moments we are interested in, the next step is to identify the community we wish to monitor from the first dataset and try to find the same community in the other datasets. If we can successfully find this community in every snapshot we can compare the resulting subgraphs and analyze their differences.

Using this approach it is fairly straightforward to gather and format necessary data since all it takes is making a copy of the web graph in the time frames we are interested in. Identifying communities is done using any community identification algorithm. These basic steps required for the snapshot model indicate that it is a very practical approach which suits widely used web graph analysis techniques.

The next step to the snapshot model, finding a community that is similar to the initial community, is more challenging. Suppose we have identified community C in the first snapshot and want to find the same community in a collection of communities identified within another snapshot for the purpose of observing mutations. One will require a characteristic which uniquely defines community C such that this characteristic still defines C after (moderate) evolution of C. If this characteristic can be converted to practical metrics it is possible to find C in a collection of web graph communities.

The main problem while defining such characteristics (and metrics) is the lack of connections between different datasets. The most reliable metric available is that one can see if nodes and edges from C are in another community as well. A common approach is to find similar communities by pattern matching $x\%$ of the nodes or edges from C with nodes or edges from another community. However, there is no known measure based upon this principle which guarantees that for each possible mutation to the original C, the mutated C in another dataset is found. Usually, such algorithms depend on a specific threshold (eg. 80% of the nodes from C must be in the other community as well) which is trained by running the algorithm on several datasets where the results are known in advance.

The problems that arise while finding similar communities are increased when the snapshots are taken far in between such that there are a lot of mutations in the web graph. The consequence is an even greater difference between community C and communities in other datasets which makes algorithms based on an overlap of similar nodes or edges less effective, assuming C is an active and evolving community.

Splitting communities prove to be a hard problem for snapshot models as well. Suppose we are tracking community C and this community splits up and evolves as two separate communities. It is likely that we can still consider C to be the same community according to some community definition even if a lot of nodes are lost due to the split. For instance, if 30% of the total amount of nodes left C but these were all unimportant nodes, one could consider this as an insignificant mutation because all the important nodes in C are still there. This example illustrates that a hard criteria for community similarity, such as an overlap of $x\%$ of the nodes, is not able to find similar communities for certain community mutations.

We conclude that the snapshot model as presented here does not suit our needs for tracking web graph community evolution. The model seems too imprecise because it can only show all mutations between one snapshot and another instead of individual changes. This problem cannot be overcome by taking snapshots more frequent because it is not feasible to make a snapshot of the web graph for each mutation. The problem of finding similar communities in different snapshots attributes to further imprecision in tracking one and the same community over time. Taking these defining properties of the snapshot model into account we consider the model too imprecise for tracking mutations within a web graph community over time.

3.2 Incremental model

Another approach is to take a single snapshot of the web graph as a base dataset, identify the community we are interested in and then incrementally record all mutations to this community. We shall refer to this as an *incremental model.* In this model we register changes to nodes in community C from the base snapshot in such a way that we can see each new node or edge as a mutation to C. The model can only observe mutations that consist of references, edges, from nodes within this community because the model initially is only aware of all nodes and edges that it contains. So new outdegree edges can be observed because these are mutations to already known nodes while new indegree edges cannot since they origin in unknown nodes. Indegree edges can only be observed and analyzed in retrospective, after the originating node has been discovered by outdegree from a known node.

An obvious advantage for the incremental model is the ease in which we can both register and observe changes to nodes and edges in a community. This property leads us to the prime advantage of the incremental model, it enables us to very accurately observe the consequences of any change to a node or edge to specific nodes or to the community as a whole. This model allows for small step analysis that is not possible using the snapshot model presented earlier.

Whereas it was easy to gather data for the snapshot model it can be hard to do so for the incremental model. It is not common for websites, nodes in the web graph, to record and distribute all their mutations. This model would either requires them to do so or the model could depend on an accurate crawler that monitors specific nodes in the web graph for mutations. Having nodes report their own mutations could result in a system with asynchronous information flow because it is possible for nodes to be slower then others in signalling changes. Reliability is another apparent issue if there is no system enforcing nodes to regularly update their status. Compared to the conventional data acquisition techniques of the snapshot model, gathering the necessary data for this model seems overly complicated.

We conclude that the incremental model is better suited then the snapshot model because it enables accurate analysis of small step mutations and their consequences whereas the snapshot model does not. The increased difficulty in gathering necessary data is considered a solvable practical problem that does not outweigh the said advantages.

In order to track mutation in communities using the incremental model we will need to use a definition for web graph communities that is compatible with the incremental mode. The candidate local community definition will need to enable small step analysis such that we can reason about the impact of single node mutations. Local modularity, as introduced in the previous chapter, meets these requirements and is our community definition of choice.

4 Analyzing Local Modularity

In this section we will focus on local modularity as introduced in section 2 which is our community definition of choice. It does not require total knowledge of the graph and is capable of operating in an incremental model, as thoroughly reviewed in the next section, so it satisfies all our criteria for a community definition.

A community is identified as a set of nodes. Let C be a community as encountered during the tracking algorithm. Since we are aware of all edges from nodes in C, we also are aware of the set of nodes outside of C that are adjacent to C. We will refer to this as the *universe* of C, denoted as $U(C)$. The *boundary* of C is formed by those nodes from C with an edge to a neighbor in $U(C)$. The boundary of C is denoted as $B(C)$.

Let D be defined as all nodes outside of community C:

Lemma 7.

$$D(C) = G.N - C$$

We will denote $D(C)$ as D and we may refer to this set as the *external nodes* of community C.

The boundary of a subgraph C from G, noted as $B(C)$, is defined as the set of all nodes within C that have an edge towards a node outside of C:

Lemma 8.

$$B(C) = \big\{u \in C \;\big|\; \exists_{v \in D}\,[(u,v) \in G.E]\big\}$$

Note that the boundary of C is a part of C, thus $B(C) \subseteq C$.

The universe of C is defined as the set of all nodes outside C that are referred to by some node from C:

Lemma 9.

$$U(C) = \big\{v \in D \;\big|\; \exists_{u \in C}\,[(u,v) \in G.E]\big\}$$

Here we can see that the universe of C is a part of D, thus $U(C) \subseteq D$.

We can divide the set of edges from any cluster in the graph into the edges towards C and the edges towards D:

Lemma 10. *Let Z be a set of nodes, then:*

$$arcs(Z, G.N) = arcs(Z, C) \;\cup\; arcs(Z, D)$$

We denote $i(C) = |arcs(B(C), C)|$ as the indegree of the boundary of C and $u(C) = |arcs(B(C), G.N)|$ as its total degree in accordance to Hinne [2].

Lemma 11.

$$arcs(B(C), C) \subseteq arcs(B(C), G.N)$$

thus

$$0 \leq i(C) \leq u(C)$$

Local modularity as defined by Clauset [3], a local variant of Newman's well known modularity measure [4], considers the fraction of boundary edges that are directed at nodes inside the community. The local modularity of a community, $R(C)$, is consequently defined as:

$$R(C) = \frac{i(C)}{u(C)}$$

Note that $0 \leq R(C) \leq 1$ by lemma 11.

We define the *mutual interest* of v and C as a fraction of the edges between C and v by all interest of v (its total outdegree). This notion is used to express how strong C and v are committed to each other in terms of edge overlap.

$$MI(C, v) = \frac{|arcs(B(C), v)| + |arcs(v, C)|}{|arcs(v, G.N)|}$$

4.1 Improved Definition

Using the given definition of $R(C)$ may lead to some problems with specific communities. We will show some limitations of this definition and propose an improvement to solve the demonstrated problems.

Example 1. Let C be the set of all nodes, then $C = G.N$, $B(C) = \emptyset$, $U(C) = \emptyset$ and thus $R(C)$ is undefined.

Example 2. Let v be a node in $G.N$ and $C = \{v\}$.
1. If v has no outgoing link, then $B(C)$ and $U(C)$ are empty and thus $R(C)$ is undefined.
2. Suppose v has outgoing links, then:

$$\begin{aligned} &B(C) = \{v\} \\ &U(C) = \left\{u \neq v \;\middle|\; (v, u) \in G.E\right\} \\ &R(C) = \tfrac{0}{u(C)} = 0 \end{aligned}$$

As Example 1 illustrates, a single node can not be a local community for any amount of outdegree edges because there are no edges from the boundary into the community if there is only one node in C, thus $|arcs(B(C), C)|$ will always be zero and so will the resulting modularity $R(C)$.

However, we regard each set of nodes as a community, its quality is measured as $R(C)$. Low values mean a low community structure. Furthermore, we consider a single node with only one edge to be a stronger community then a single node with multiple edges because if a community has less references to the outside world it is more introvert, has a higher inbound modularity, thus is stronger by definition of local modularity. Since we have shown that the current definition of $R(C)$ does not reflect this, we adapt $R(C)$ such that:

$$R(C) = \frac{i(C) + \epsilon}{u(C) + \epsilon}$$

for some yet undetermined $\epsilon > 0$. The addition of ϵ prevents the division by zero problems that arise in the original definition while maintaining the desired properties mentioned before. For this improved definition, $0 \leq R(C) \leq 1$ still applies.

Example 3. Let C be the set of all nodes, then $C = G.N$, $B(C) = \emptyset$, $U(C) = \emptyset$ and thus $R(C) = \frac{\epsilon}{\epsilon} = 1$.

Example 4. Let v be a node in $G.N$ and $C = \{v\}$.
1. If v has no outgoing link, then $B(C)$ and $U(C)$ are empty and thus $R(C) = \frac{\epsilon}{\epsilon} = 1$ as we have seen in the previous example.
2. Suppose v has outgoing links, then:

$$\begin{aligned} &B(C) = \{v\} \\ &U(C) = \left\{ u \neq v \;\middle|\; (v,u) \in G.E \right\} \\ &R(C) = \tfrac{\epsilon}{u(C)+\epsilon} \neq 0 \end{aligned}$$

Now that we have considered how ϵ improves $R(C)$ we regard its influence on the result. We find that choosing a small value for ϵ will cause no significant influence on $R(C)$.

Lemma 12. *As the community boundary grows, the influence of ϵ on the outcome of $R(C)$ decreases rapidly. A smaller ϵ will cause less deviation from $\frac{i(C)}{u(C)}$ as well.*

Proof.

$$\begin{aligned} \text{Let}\quad & \epsilon < u(C),\ a = i(C),\ b = u(C),\ c = \epsilon \\ R(C) &= \tfrac{a+c}{b+c} \\ &= \tfrac{a}{b+c} + \tfrac{c}{b+c} \\ &= \tfrac{a}{b} + \tfrac{-ac}{b(b+c)} + \tfrac{bc}{b(b+c)} \\ &= \tfrac{a}{b} + \tfrac{bc-ac}{b^2+bc} \end{aligned}$$

Here we can see that the influence of ϵ is expressed by $\frac{bc-ac}{b^2+bc}$. The deviation from $\frac{i(C)}{u(C)}$ decreases as we use a smaller value for ϵ (c) or when $u(C)$ (b) grows.

Note that if either $i(C)$ or $u(C)$ is zero, ϵ corrects problems from the original definition we mentioned earlier so we do not consider ϵ to negatively influence the result in that case.

Lemma 13. *$U(C) = \emptyset \Leftrightarrow R(C) = 1$.*

Proof. $\Rightarrow$ If C has no universe there are no outbound edges in C thus the boundary of C is empty as well. Therefore $i(C) = u(C)$ thus $R(C) = 1$.
$\Leftarrow$ If $R(C) = 1$ then $i(C) = u(C)$ and therefore $|arcs(B(C), D)| = 0$. Thus there are no outbound edges in C and therefore C has no boundary nor a universe.

5 Community Tracking and Evolution

So far we have analyzed and extended the definition of local modularity and now we propose an incremental definition such that we can regard the impact of the addition of one universe node to a community. This definition may represent the next step in a (local) community building algorithm or a step in the evolution of a predetermined community.

Given a community C under construction, the question is what nodes are the best candidates to extend C. Nodes in $U(C)$ are candidates as they are at least connected to C. Other nodes in the rest of the graph may have a link to C but there is no direct link from C to these nodes , otherwise they would have been a member of $U(C)$. Since the local view of the web graph implies limited knowledge of the entire graph we may only consider known nodes as possible community candidates.

Only nodes that have an indegree can be detected by any algorithm that follows the principle of local modularity since they are unknown to C otherwise. A node v that has no indegree cannot be a member of $U(C)$ since there is no node from C that has an edge to v. Thus our incremental model only considers nodes from the universe of C and these all have an indegree of at least one. We will use the conventional notation $C + v$ to denote $C \cup \{v\}$.

5.1 Node Position Within a Community

By definition of $R(C)$ a member of a community has a different impact on the local modularity depending on its position in the community since the outdegree of a node in $B(C)$ affects the modularity of C while the outdegree of a node in $C - B(C)$ does not. So in order to determine the impact of the addition of any node $v \in U(C)$ to C, we distinct two consequences for v:

1. v will not become a member of the boundary because all edges of v are inbound to C, thus $v \notin B(C + v)$
2. v will end up in the boundary because v has edges outbound to C, thus $v \in B(C + v)$

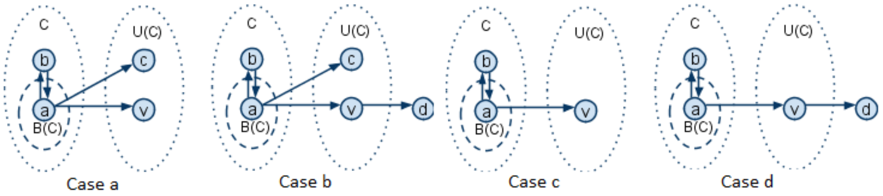

Fig. 2. Four graphs with different consequences of adding node $v \in U(C)$ to C.

Besides the consequences for v we must also consider the consequences for C after adding v to C. Adding a node from the universe to a community may, or may not, affect the position of nodes from $B(C)$ such that they are not a member of the boundary of $C + v$. All nodes from $B(C)$ for which their edge to v is their only edge towards a node outside of C are not a member of $B(C + v)$ because they have no edges towards any node outside of the new community. So we distinct two cases for the consequences for nodes in $B(C)$ after adding v to C:

1. All nodes from $B(C)$ are a member of $B(C+v)$ as well, thus $B(C) \subseteq B(C+v)$
2. At least one node from $B(C)$ is repositioned such that it is not a member of $B(C + v)$, thus $B(C) \nsubseteq B(C + v)$

If we combine these cases, there are four situations where we must consider the differences between $R(C)$ and $R(C + v)$:

$$\begin{aligned}
&\text{a: } v \notin B(C+v), B(C) \subseteq B(C+v)\\
&\text{b: } v \in B(C+v), B(C) \subseteq B(C+v)\\
&\text{c: } v \notin B(C+v), B(C) \nsubseteq B(C+v)\\
&\text{d: } v \in B(C+v), B(C) \nsubseteq B(C+v)
\end{aligned}$$

Figure 2 illustrates all four situations listed above situated in example graphs.

If we consider the set of nodes that is $B(C)$ and add node v to community C, we observe that the boundary of the new community $C+v$ is either the same as $B(C)$ or a subset thereof.

Lemma 14.

$$B(C + v) \subseteq B(C) + v$$

Proof. As we have seen earlier, the addition of a node to a community may cause previous members of $B(C)$ to leave the boundary such that they are not a member of $B(C+v)$. In this case, $B(C) \nsubseteq B(C+v)$. Also, we know that node v does not have to become a member of the boundary when it joins community C such that $v \notin B(C + v)$.

When combined, these two observations show that the boundary of $C + v$ is a subset of $B(C) + v$.

Lemma 15.

$$v \notin B(C+v) \wedge B(C) \subseteq B(C+v) \Rightarrow B(C+v) = B(C)$$

Proof. As $v \notin B(C+v)$ we have $B(C+v) \subseteq B(C)$. Because v is not a member of $B(C+v)$ there are no nodes from $B(C)$ that will leave the boundary and thus by Lemma 14 $B(C+v) = B(C)$.

5.2 Case a

First we will consider cases a and b. Because $B(C) \subseteq B(C+v)$ applies we only need to consider the impact v has on the modularity. For case a we place v in $C - B(C)$ because all edges from v are directed at nodes in C. We also know that there is at least one edge from a node in $B(C)$ to v because v used to be a member of $U(C)$. Thus moving v from $U(C)$ to C increases the amount of inbound edges for the boundary. And since v will not be a member of the boundary we observe that the amount of edges in the boundary does not change.

Lemma 16. *Thus case a amounts to:*

$$R(C+v) = \frac{i(C) + \epsilon + |arcs(B(C), v)|}{u(C) + \epsilon}$$

Corollary 1. *Suppose $v \notin B(C+v)$ and $B(C) \subseteq B(C+v)$, then we observe that v always improves the community:*

$$\begin{aligned} i(C+v) &= i(C) + |arcs(B(C), v)| \\ u(C+v) &= u(C) \\ R(C+v) &= R(C) + \tfrac{|arcs(B(C),v)|}{u(C)+\epsilon} > R(C) \end{aligned}$$

5.3 Case b

Case b places v in $B(C+v)$ because there is at least one edge from v to D. As seen in condition a, the edges from $B(C)$ to v are considered an increase to the amount of inbound edges in the boundary because v is a (new) member of C. But since v also is a member of $B(C+v)$, all edges from v towards C are contributing to the amount of inbound edges as well. Finally, the total amount of edges from the boundary is increased by the amount of edges from v towards any node in the graph.

Lemma 17. *Thus case b amounts to:*

$$R(C+v) = \frac{i(C) + \epsilon + |arcs(B(C), v)| + |arcs(v, C)|}{u(C) + \epsilon + |arcs(v, G.N)|}$$

Corollary 2. *By definition of $R(C{+}v)$ for case b we may use the mutual interest in this case to consider if an external node will improve a community.*

$$R(C+v) > R(C) \Leftrightarrow R(C) < MI(C,v) \wedge v \in U(C)$$

Lemma 18.

$$v \in B(C+v) \wedge B(C) \subseteq B(C+v) \Rightarrow B(C+v) = B(C) + v$$

Proof. According to Lemma 14, $B(C+v) \subseteq B(C) + v$. Let $x \in B(C+v)$ and $x \neq v$ then $x \in B(C)$. So $B(C+v) - v = B(C)$ thus $B(C+v) = B(C) + v$

Suppose node $v \in U(C)$ has one edge towards a node outside of C, $w \in D$, then the addition of v to C will be an improvement to C. Moving $v \in U(C)$ to C will only add 1 edge to the boundary that does not lead to a node in C in this case, that edge is (v, w), thus $u(C+v) = u(C) + 1$. Since any edge from $B(C)$ to v has now become an internal edge from the boundary, there is at least one such edge by definition of $U(C)$, and the modularity was less then 1 by Lemma 13 the local modularity of C has improved.

Lemma 19. *If case b applies and* $|arcs(v, D)| = 1$*, then the addition of* v *improves the community.*

Proof. Note that $|arcs(v, D)| = 1$ is equivalent with $|arcs(v, G.N)| = |arcs(v, C)| + 1$. Let $|arcs(B(C), v)| > 0$ since $v \in U(C)$ and $R(C) < 1$ by Lemma 13. Then $MI(C, v) = \frac{|arcs(B(C),v)|+|arcs(v,C)|}{|arcs(v,C)|+1}$ and thus $MI(C, v) \geq 1 > R(C)$. Therefore $R(C+v) > R(C)$.

5.4 Case c

For cases c and d we will consider which nodes from $B(C)$ are not a member of $B(C+v)$. If we consider $B(C) \not\subseteq B(C+v)$, we define the set of nodes that will leave the boundary as a consequence of adding v to C as:

$$\begin{aligned} Z(C, v) &= B(C) - B(C+v) \\ &= \left\{ u \in B(C) \;\middle|\; u \notin B(C+v) \right\} \end{aligned}$$

Intuitively, case c is the same as case a except for the loss of a set of boundary nodes Z as a consequence of adding v to C. From the inbound boundary edges $i(C)$ we will need to subtract the edges from Z to C and from the boundary degree $u(C)$ we must subtract the edges from Z to $G.N$.

Lemma 20. *Thus case c amounts to:*

$$R(C+v) = \frac{i(C) + \epsilon + |arcs(B(C+v), v)| - |arcs(Z, v)|}{u(C) + \epsilon - |arcs(Z, G.N)|}$$

5.5 Case d

The last case, d, is similar to case b except for the loss of boundary nodes as we have seen in case c.

Lemma 21. *Thus case d amounts to:*

$$R(C+v) = \frac{i(C)+\epsilon+|arcs(B(C+v),v)|+|arcs(v,C)|-|arcs(Z,v)|}{u(C)+\epsilon+|arcs(v,G.N)|-|arcs(Z,G.N)|}$$

Note that cases c and d can also define cases a and b respectively, since Z will be the empty set in cases a and b. The fraction of inbound edges from Z may be used to reason about cases c and d just as we reason about cases a and b. Suppose we are considering case c for some $C+v$. Then we may apply Corollary 1 if $|arcs(Z,v)|\backslash|arcs(Z,G.N)| < R(C+v)$ if we use the $R(C+v)$ definition from case a. The use of case b while considering case d is analogous.

6 Error Analysis

After each step in the algorithm the community found is an approximation of the actual community. We denote the actual community around a given source node v as $[v]$.

There are two kinds of errors in this approximation. The first is an error of omission where we may exclude a node from the actual community, $u \in [v] \wedge u \notin C$. The second is an error of commission where we may include a node that is not within the actual community, $u \in C \wedge u \notin [v]$. In this paper we only consider the first kind.

To analyze such errors we require additional information about nodes outside of a local community in order to consider their relevance to that community. Therefore we group such external nodes in multiple boundaries around a community such that we are able to reason about their relevance to that community in term of their distance to that community. In this section we consider two distinct ways to determine membership of a boundary, by path distance from the community or by their optimal local modularity value.

Regardless of how membership of a boundary is determined, we assume that the nodes that are in the boundary closest to the community are the most important nodes. Thus nodes closer to the community are more likely to be a member of this community.

Ideally, we would like to see new nodes in the web graph that become increasingly important to a community move closer to that community over time instead of appearing in the community out of the blue. If the node first appeared at boundary k, it should gradually move to a boundary with a lower value for k until finally merging with the community.

Suppose there is a node u that is a part of the actual community $[v]$ but it is not within community C. So $C+u$ would represent community $[v]$ better then C does. However, we may not know much about node u and therefore we may need to assume information about the edges, and perhaps nodes, between u and C.

This node u may be at a k step distance from community C, meaning there are $k-1$ nodes in the path from C to u. In this paper we only consider the probability that u is 1 step away from C, thus $u \in U(C)$. If we find this probability, distance k may be determined by induction.

In this section we propose a probability function for omission errors for a local community defined by the incremental definition we coined earlier.

6.1 Boundaries by Distance

First we consider community *boundaries by distance*. In order to group external nodes we express their importance to C by path distance. We define community boundaries as clusters of nodes that are referred to by nodes from the community in k steps.

One can imagine that an external node can be reached through various paths with a different length. We will place a node in the boundary according to its shortest path from the community, thus a node can only be a member of one boundary. We define boundaries by distance as $B_0 = C$ and $B_{k+1}(C) = B(B_k(C))$. Note that $B_1(C) = B(C)$ and $B_2(C) = U(C)$. We may also define boundary k for community C as:

$$B_k(C) = \big\{ v \in G.N \;\big|\; v \notin B_{k-1}(C) \wedge \exists_{u \in B_{k-1}(C)} \, [u \rightarrow v] \,\big\}$$

6.2 Boundaries by Modularity Value

We may also define *boundaries by modularity value*. Instead of using node distance, we define importance of a node in terms of the modularity value of a subset that both this node and the community source node are in.

A local community is defined as a set of nodes that has a coherence that corresponds to a modularity threshold. The set of local communities in G defined by modularity threshold d is:

$$LC(G, d) = \big\{ C \subseteq G.N \;\big|\; R(C) \geq d \big\}$$

We define a layer as a cluster of nodes from the graph that contains some source node and, as a whole, corresponds to a given modularity value indicating the coherence of this cluster.

$$Layer(v, d) = \big\{ C \in LC(g, d) \;\big|\; v \in C \big\}$$

where d is the defining modularity threshold for a layer. For a graph that contains several layers around some source node there will be a series of such layers.

The layers are sorted by their modularity thresholds such that the most coherent layer, that has the highest value for d, is L_1. In contrast to the boundaries defined by node distance, the use of layers allows room for overlap. This approach may be more computationally intensive then the usage of boundaries by distance because it requires the community identification algorithm to be used a number of times, the exact amount depends on the desired amount of layers.

The approach is quite versatile and may be used for other community definitions if one would substitute the modularity criterium in $LC(G, d)$ with the criterium for any other community definition.

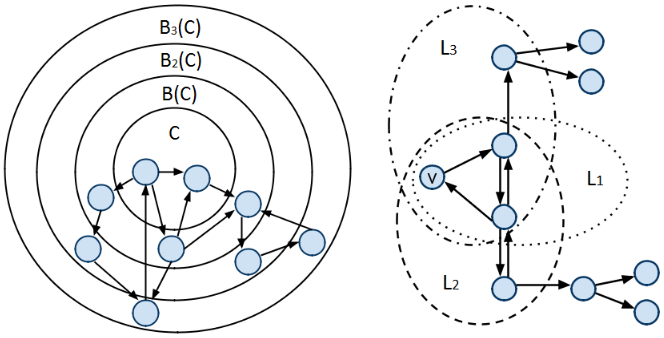

Fig. 3. Left: Multiple boundaries for community C defined by node distance. Right: Multiple layers around source node v. $R(L_1) = \frac{3+\epsilon}{5+\epsilon}$, $R(L_2) = \frac{2+\epsilon}{4+\epsilon}$, $R(L_3) = \frac{2+\epsilon}{5+\epsilon}$.

6.3 Omission Error Probability

In this subsection we consider the probability of missing a relevant node such that $u \in [v] \wedge u \notin C$. Let k be the distance from C to node u. Then there is a (shortest) path of lenght k from C to u. The addition of each of the $k-1$ nodes on this path starting with the one closest to C will have an impact on the modularity of C. Since we cannot predict the impact of the addition of more than one node at a time we will require each node in this path to improve the community.

The probability function considers how likely it is that a node from the actual community is not in C but at k steps away from C instead. This distance is expressed by being a member of boundary (by distance) k. Thus the probability function is defined as:

$$Pr\left(u \in [v] \;\middle|\; u \in B_k\right)$$

We use the *configuration model* for directed graphs [5] which represents a random (directed) graph with a given degree sequence to reason about unknown edges. In the configuration model random graphs are generated that are like a given graph such that they have the same set of nodes and for each node the same degree as the given graph. The edges are randomly distributed in accordance to the predetermined in and out degree of each node.

For a directed graph one must consider that the edge direction has great influence on the expected degree distribution. Suppose there are two nodes, u and v. Node u has a high outdegree and a low indegree while v has the opposite degree distribution. So a given edge is more likely to run from u to v than vice versa. Thus if we observe such a degree distribution in a directed graph we should consider an edge from v to u as statistically surprising.

For each node we consider whether the connected edges are incoming or outgoing. Let m be the total amount of edges in the graph, k_i^{in} is the indegree of node i in the graph and k_i^{out} the outdegree. The configuration model leads to the probability of an edge from j to i in a directed graph:

$$Pr\left(j \to i\right) = \frac{k_i^{in} k_j^{out}}{m}$$

We may use the power law to reason about the degree distribution of unknown nodes. As mentioned in the introduction of this section, we reduce the k step path to $k = 1$ and consider how likely it is for a random node at distance 1 to become a member of C. Later we may determine the probability at distance k by induction. So we are adding a node from the universe to the community and we are interested in the probability that this will improve the community.

$$Pr\left(R(C+v) > R(C) \;\middle|\; v \in U(C)\right)$$

The incremental definition for the local modularity of a web graph community considers four cases. So we can break this probability function down to finding the probability of $R(C+v) > R(C)$ for each case and the probability that each case may occur. In this paper we elaborate case a, the other cases are analogous.

For case a we proved that adding node v to the community C always improves C in Corollary 1. So we only need to consider the probability that this case may occur, the requirements being $v \notin B(C+v) \wedge B(C) = B(C+v)$. The first condition is met when all edges from v are directed at nodes within C, the latter requires all nodes that refer to v from C to remain within the boundary. Thus these nodes need to have another reference outside of C. We denote this probability as:

$$Pr\left(v \not\to u \;\middle|\; u \in D\right) * \sum_{u \in C} Pr\left(u \to v \;\middle|\; u \to w, w \in D - v\right)$$

The probability of no edges from v referring outside of C may also be denoted as $1 - Pr\left(v \to D\right)$. So we can resolve both probabilities using $Pr\left(x \to A\right)$ where $x \in G.N$ and $A \subseteq G.N$. We will denote the combined indegree of all nodes in set A as k_A^{in}. By using the configuration model and the power law we find this probability:

$$\begin{aligned} Pr\left(x \to A\right) &= \textstyle\sum_{d=0}^{\infty} Pr\left(x \to A \;\middle|\; k_x^{out} = d\right) * \\ &\quad\; Pr\left(k_x^{out} = d\right) \\ &\sim \textstyle\sum_{d=0}^{\infty} \frac{d.k_A^{in}}{m}\, e^{-d} \\ &= \textstyle\frac{k_A^{in}}{m} \sum_{d=0}^{\infty} d\, e^{-d} \\ &\sim \textstyle\frac{k_A^{in}}{m} \int_d^{\infty} x\, e^{-x}\, dx \end{aligned}$$

Using the omission error probability coined in this section we illustrate how to estimate the quality of a local community defined by our incremental definition. Our analysis has shown how this can be achieved and the results look promising.

7 Conclusions and Future Work

We have considered relevant web graph community properties and analyzed how to identify such communities. We discussed two models for observing community evolution and found the snapshot model too coarse and proposed the use of an incremental model for tracking web graph communities.

We chose to use local modularity for defining a local community in the web graph without having knowledge of the entire graph. After observing specific local community cases we have proposed an extension to the definition of local modularity and subsequently we have introduced an incremental definition thereof. Our incremental definition considers four distinct cases to explain the impact of the addition of a new node to a community. So this technique allows us to observe the small step evolution of the structure of a community and its boundary in terms of local modularity.

While observing a community in a local model there may be a need for additional contextual information of a community. Using a local approach there usually is only knowledge of all nodes within a community and its universe. We have proposed the use of multiple boundaries to group external nodes by their relevance to a community. We proposed two measures for defining such boundaries, grouping nodes by distance to a community or by the highest modularity value of all possible clusters that relate a node to a community.

The error analysis for local communities looks promising. We are able to reason about the quality of a local community by considering the probability of omission errors for a local community. For future research we are interested in continuing the error analysis. Besides considering the other cases of the incremental definition for omission error analysis we would like to validate the omission error probability.

References

1. The Size of the World Wide Web, http://www.worldwidewebsize.com

2. Hinne, M.: Local Identification of Web Graph Communities. Proceedings of ICTIR 2007, Budapest, Hungary, 261-278 (2007)

3. Clauset, A.: Finding Local Community Structure in Networks. Physical Review E 72(2), 026132 (2005)

4. Newman, M.E.J., Girvan M.: Finding and Evaluating Community Structure in Networks. Physical Review E 69(2), 026113 (2004)

5. Leicht, E.A., Newman, M.E.J.: Community Structure in Directed Networks. Physical Review Letters 100, 118703 (2008)

6. Kleinberg, J.M.: Authoritative Sources in a Hyperlinked Environment. Proc. ACM-SIAM Symposium on Discrete Algorithms (1998)

7. Fasulo, D.: An Analysis of Recent Work on Clustering Algorithms. Univ. of Washington (1999)

8. Kleinberg, J.M.: An Impossibility Theorem for Clustering. Advances of Neural Information Processing Systems 15, MIT Press (2003)

9. Flake, G.W. and Lawrence, S. and Giles, C.L.: Efficient Identification of Web Communities. 6th Int. Conf. on Knowledge Discovery and Data Mining (2000)

10. A. Broder, et al.: Graph Structure in the Web. Computer Networks 33, 309 (2000)

11. Albert, R. and Barabsi, A.L.: Statistical Mechanics of Complex Networks. Reviews of Modern Physics, vol. 74, no. 1, pp. 47-97 (2002)

12. Mason, O. and Verwoerd, M.: Graph Theory and Networks in Biology. IET Systems Biology (2006)

13. Flake, G.W. and Tsioutsiouliklis, K. and Zhukov, L.: Methods for Mining Web Communities: Bibliometric, Spectral, and Flow. Web Dynamics Springer Verlag, ch. 4, pp. 45-68 (2004)

14. Wu, F and Huberman, B.A.: Finding Communities in Linear Time: A Physics Approach. The European Physical Journal B - Condensed Matter and Complex Systems, vol. 38, no. 2, pp. 331-338 (2004)

15. Danon, L. and Duch, J. and Diaz-Guilera, A. and Arenas A.: Comparing Community Structure Identification. Journal of Statistical Mechanics: Theory and Experiment, vol. 09, P09008 (2005)

16. Xin-Ping, X and Feng, L: A Novel Configuration Model for Random Graphs With Given Degree Sequence. Chinese Physics, vol. 16, 282-286 (2007)

8 Appendix: list of symbols

A_{ij}	An edge j to i in an adjacency matrix
$arcs(X, Y)$	Set of edges from X to Y
$B(C)$	The boundary of community C
$B_k(C)$	Same as $B(C)$ but at distance k
C	A web graph community
D	All nodes outside of community C
G	A graph
$G.E$	All edges in graph G
$G.N$	All nodes in graph G
$i(C)$	Edge count from $B(C)$ towards C
k_i^{in}	Indegree of node i
k_i^{out}	Outdegree of node i
k_i	Degree of node i
$LC(G, d)$	Set of local communities with $R(C) = d$
$Layer(v, d)$	Element of $LC(G, d)$ containing node v
$MI(C, v)$	The mutual interest of C and node v
Pr	Probability function
Q	Newman's modularity measure
$R(C)$	The modularity of C
$u(C)$	Edge count from $B(C)$ to any node
$U(C)$	The universe of community C
$[v]$	Actual community for source node v