ABSTRACT

Title of Thesis: Learning Hierarchical Workflows Using Community Detection

Akshay Peshave, Masters in Computer Science, 2014

Thesis directed by: Dr. Tim Oates, Professor
Department of Computer Science and
Electrical Engineering

Workflows identified from user event logs and click-stream data are useful as knowledge bases for behavioral analysis and recommendation systems. In this study we identify abstractions or summaries of event logs modeled as user activity flow networks. The abstractions are identified based on structural properties as well as user activity flow dynamics over the network using community detection methods. We apply a fast modularity optimization and multi-level resolution approach to detect hierarchical community structure in user activity flow networks. The detected communities are compared to those detected by the information-theoretic map equation minimization approach to weigh pros and cons of the fast modularity optimization approach in the workflows context. We further attempt to identify the most probable sources and sinks of user activity in individual communities and trim the network accordingly to reduce entropy of the workflow abstractions.

LEARNING HIERARCHICAL WORKFLOWS USING COMMUNITY DETECTION

by

AKSHAY PESHAVE

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
MASTER'S IN COMPUTER SCIENCE
2014

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

Dedicated to my mother...

look mom, I almost made computer scientist!!!

ACKNOWLEDGMENTS

Sincere thanks to Dr. Tim Oates and Dr. Matt Schmill for their guidance and encouragement throughout my research. And to all my loved ones for their support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Web application clickstreams and event logs have been historically mined for behavioral patterns which serve as knowledge bases for analytics and personalization systems (Baglioni *et al.* 2003; Eirinaki & Vazirgiannis 2003; Lu, Dunham, & Meng 2006). The e-commerce and online advertising domains have been applying clickstreams and event logs based personalization and analytics methods for advertisement targeting (De Bock & Van den Poel 2010) and campaign analytics (Chatterjee, Hoffman, & Novak 2003). Research on applying these methods to learn user activity models has also been maturing since.

Workflow learning deals with discovering user activity models which may fit the activity flow present in event logs while generalizing enough to suggest unseen yet valid possible activity flows. Several apporaches have been discussed in the past to discover and learn workflows from logs and traces: machine learning approaches (Herbst 2000), WF-net induction (Van der Aalst, Weijters, & Maruster 2004) and grammar induction (Yaman, Oates, & Burstein 2009; Jones & Oates 2010). All approaches discover certain classes of workflows from their corresponding process event logs and are susceptible to adverse effects of noisy traces.

Web application clickstreams and event logs are known to be of significantly fine

granularity to enable production bugs triaging. Such fine granularity implies a sufficiently large number of events and transitions all of which may not necessarily be relevant to learning a workflow model. Our interest lies in discovering an abstract workflow at a much coarser granularity than that of the raw clickstreams and event logs. For example, clicks on various widgets of a web page may be logged by several web applications. But adapting a process model to events inside a web page may seldom be a requirement. Such events occur non-deterministically and are often required to be so. This extent of fineness in granularity will impose arbitrary noise if raw clickstreams are subjected to the approaches mentioned above.

The objective of this work is to mine structural patterns which are abstractions of user activity flow represented by event logs or clickstreams at a relatively coarser granularity. The abstractions should represent the macro activity flow in the system. As with any summarization process, we expect a potential loss of micro flow information. The abstractions will be valid if this loss of information is limited enough so that the correct macro flow is represented by the abstractions. That is, the noise in clikstreams and event logs should constitute the majority portion of the lost activity flow information. Such abstractions will be good inputs for the approaches mentioned above in order to discover macro workflows for software aided processes.

A workflow is a collection of tasks. Each task may be composed of several sub-tasks which in turn are a collection of activities. Discovering this hierarchy is central to our work. The raw clickstreams and event logs are formed by individual activities and form the lowest level of this workflow hierarchy. We pursue the idea of clustering events in event logs based on a measure of cohesiveness among them. The intuition is that high cohesiveness among events in a software aided process should imply commonality in context of these events. The context here is the task (or sub-tasks) to which the activities belong. (Facca & Lanzi 2005) surveys various classes of web-log mining approaches including clustering based on

similarity metrics. These clustering methods require prior knowledge of the tasks to which each event belongs. Our method yields clusters without requiring prior knowledge of such a mapping.

1.1 The Activity Flow Graph

In graphs, cohesiveness and inter-dependency among its nodes are quantified in terms of edge density and flow density distributions among other measures. Community detection algorithms exist which partition graphs optimally to resolve inter-dependent sub-graphs based on measures of cohesiveness. Each resulting community is composed of a cohesive subset of the graph's vertex set and and the edges among them. Thus, we represent the activity flow information in event logs in the form of a graph. We parse event logs and construct a directed, weighted graph G=(V, E) representing the user activity flow:

$$V = \{\, v : v \in \{activities\}\},$$

$$E = \{\, (v_1, v_2, w) : v_1, v_2 \in V,\ w \in \mathbb{Z}^+,\ (v_1, v_2, w) \neq (v_2, v_1, w)\},$$

$$and$$

$$edge\ e = (v_1, v_2, w)\ represents\ a\ next-step\ relationship\ between\ v_1\ and\ v_2$$
$$with\ w = observed\ frequency\ of\ the\ relationship\ in\ the\ event\ logs.$$

Throughout this work, we will refer to G as the activity flow graph because it represents the flow of user activity based on the event logs for a system. The activity flow graph will be used to discover a high likelihood hierarchical workflow using a community detection approach. Figure 1.1 shows a sample user activity log (1.1(a)) and the corresponding activity flow graph (1.1(c)) constructed using the logs. We can construct a activity workflow graph for the activity logs using activity sequences per user shown in Figure 1.1(b).

We see that activity A_1 has a self-loop transition. This transition is observed 4 times in the user activity log, hence the frequency for the transition is 4.

1.2 Experiment Data Source

GLOBE (Magliocca *et al.* 2014) or Global Collaboration Engine is an "online collaborative environment" for land change researchers. We will use the event logs from GLOBE for our experiments. Its event logs are comprehensive and well structured. It has 43 distinct user activities (atomic system events) which are logged per user. Every activity is mapped to one of 9 logical activity groups defined in GLOBE for the purposes of logging. Every activity has a unique activity id for event logging purposes and its logical activity group is indicated by a prefix to its activity id. For example, the login and logout activities belong to the "user account" activity group and their activity ids bear the prefix "ua".

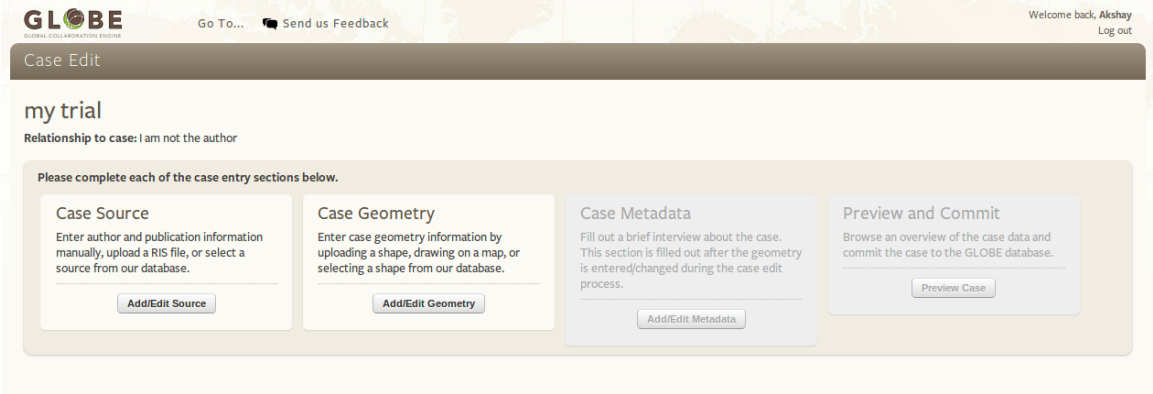


FIG. 1.2. GLOBE Case Creation Workflow

Basic users of GLOBE follow a predefined workflow to contribute a case (a basic artifact in GLOBE) based on a study of their choice to the GLOBE user community. The workflow is shown in the screenshot in Figure 1.2. Users are allowed to edit source in-

formation as well as the geometry for the case concurrently or in any sequential order. The user may edit work on the metadata for the case only after completing the source and geometry information for the case once. The last step involves confirming all data and committing the case.

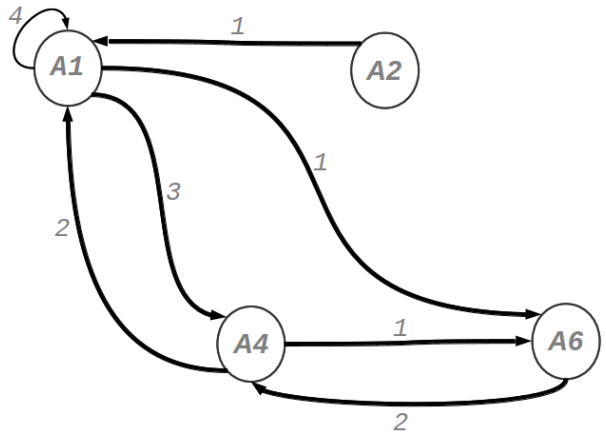
Other implicit workflows include case collection management, case editing and analyses of representativeness and similarity of cases. These workflows along with the activity to activity groups mapping will serve as a crude measure of accuracy for the activity communities and workflow hierarchy detected by our method.

Time Stamp	User	Activity ID
t_{δ}	U_1	A_1
$t_{\delta+1}$	U_2	A_2
$t_{\delta+2}$	U_1	A_1
$t_{\delta+3}$	U_3	A_1
$t_{\delta+4}$	U_1	A_4
$t_{\delta+5}$	U_1	A_6
$t_{\delta+6}$	U_2	A_1
$t_{\delta+7}$	U_2	A_1
$t_{\delta+8}$	U_1	A_4
$t_{\delta+9}$	U_3	A_4
$t_{\delta+10}$	U_3	A_1
$t_{\delta+11}$	U_3	A_6
$t_{\delta+12}$	U_3	A_4
$t_{\delta+13}$	U_2	A_1
$t_{\delta+14}$	U_2	A_4
$t_{\delta+15}$	U_1	A_1
$t_{\delta+16}$	U_1	A_1

(a) User Activity Log

User	Activity Sequence
U_1	$A_1, A_1, A_4, A_6, A_4, A_1, A_1$
U_2	A_2, A_1, A_1, A_1, A_4
U_1	A_1, A_4, A_1, A_6, A_4

(b) User Activity Sequence



(c) User Activity Flow Graph

FIG. 1.1. Example of Activity Flow Graph Creation

Chapter 2

COMMUNITY DETECTION

In this section we briefly discuss three community detection approaches: (i) Modularity optimization: a relatively fast and popular approach which optimizes an edge density based objective function called modularity. (ii) Infomap: an information-theoretic approach which uses containment of flow in modules as a criterion. (iii) Stability augmented modularity optimization: a flexible, multi-level community detection approach which adopts both quantifications of cohesiveness used in the previous two approaches.

We will weigh merits and demerits of each approach to resolve hierarchical workflows from event logs. Essential factors to be considered for our problem become clearer here allowing for an informed choice of community detection method for our needs.

2.1 Modularity Optimization

Modularity (Newman & Girvan 2004) is a measure of the quality of a partition of a given network which may or may not exhibit community structure. Given a partition, modularity is quantified as the difference between the fraction of intra-community edges considering original edges in the network and the fraction when edges are distributed across the network at random. It measures how different the network exhibiting community structure is, in terms of localized edge density, from the same network with a random distribution of

edges based on uniform degree.

Although modularity of 1.0 is theoretically indicative of strongest community structure, Newman et al. (Newman & Girvan 2004) state the typical range of modularity for networks with strong community structure to be [0.3,0.7], higher modularity being rare. Their formulation of modularity holds for unweighted and undirected networks. Although modifying the modularity definition for weighted networks is fairly simple, doing so for directed networks is not. A survey of works addressing modularity optimization for directed networks, challenges and their respective merits and short-comings is found in (Malliaros & Vazirgiannis 2013).

Modularity optimization approaches, in addition, are faced with a limit on the minimum size of detectable communities termed the resolution limit (Fortunato & Barthlemy 2007). Modularity, being a measure of pairwise interactions among network nodes, implies this approach misses out on information flows over longer paths and may detect communities which do not accurately represent the dynamics of the network (detailed discussion in (Rosvall & Bergstrom 2008)).

2.2 Infomap and the Map Equation

Infomap is an information-theoretic approach to community detection proposed by Rosvall & Bergstrom (Rosvall & Bergstrom 2008). It seeks to understand patterns of flow of information (by way of movement) over a network. In doing so it identifies local interactions which tend to induce flow across the system. These local flow patterns are characteristic of communities of nodes in the network. This is in contrast to modularity optimization which relies on pairwise associations among nodes of a network.

Rosvall and Bergstrom reduce the community detection problem to that of efficient codification of a weighted random surf over the network. Rosvall et al. (Rosvall, Axels-

son, & Bergstrom 2009) further discuss this ”inference-compression duality” and the map equation which provides a lower bound on the code length for describing a random surf on the network for a given partition.

Infomap is based on two fundamental premises: (i) flow persists for longer durations in modules of the network. (ii) modules induce a pattern of flow over the network. Since the ”inference-compression duality” is being exploited, two code books are maintained:

1. Module codebook: contains codewords to describe walks within modules

2. Index codebook: contains codewords to describe inter-module walks

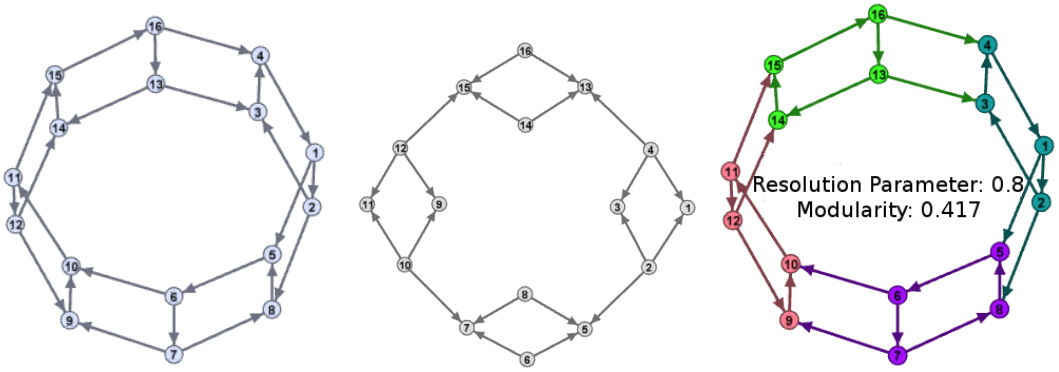
The intuition is that the optimal network partition will require minimum code length to describe flow on the network which is composed of intra-community and inter-community movements. The map equation is a manifestation of both these movements weighted by frequency of corresponding codebook usage i.e. frequency of intra-community (module codebook usage) and inter-community (module codebook usage) movement. In effect, Infomap detected communities are representative of network dynamics over longer distances based on flow (or movement), which is an attribute central to characterization of modular networks.

2.3 Hierarchical Community Detection

2.3.1 Hierarchical Modularity Optimization

The hierarchical modularity optimization approach described by Blondel et al. in (Blondel *et al.* 2008) uses a pair-wise agglomerative strategy for optimizing modularity. It significantly reduces the effect of resolution limit and is demonstrated to detect communities accurately and with more efficiency than most other community detection algorithms on large networks.

The speed and hierarchical nature of this approach is attributed to its utilization of the method of Arenas et al. (Arenas *et al.* 2007). Arenas et al. reduce network size by grouping nodes and collapsing them into community nodes while preserving modularity of the network. When done recurrently the approach identifies successively smaller macro-networks of super-communities having the same modularity as their predecessor macro-networks. Successive network size reduction allows modularity optimization described in (Blondel *et al.* 2008) to iteratively handle smaller macro-networks resulting in a steady speedup. Iterative reduction and community resolution enables hierarchical modularity optimization. Arenas et al. (Arenas *et al.* 2007) formulate the modularity expression and reduction mechanism for directed graphs too, thus extending applicability to directed, weighted graphs.



(a) N/W 1 (b) N/W 2 (c) Communities in N/W 1 using Multi-level Community Resolution

FIG. 2.1. Example N/Ws for Community Detection Evaluation (rendition of Fig 2 in (Rosvall & Bergstrom 2008))

(Rosvall & Bergstrom 2008) shows that modularity optimization may detect communities, based on edge densities, contrary to the true flow dynamics of networks. This is demonstrated using two example networks. Refer to Fig. 2.1(a) & 2.1(b) for these two

networks. It is also noted in (Malliaros & Vazirgiannis 2013) that the method of Arenas et al. (Arenas *et al.* 2007) confuses directionality of edges making it susceptible to the issue demonstrated in (Rosvall & Bergstrom 2008). This suggests that the approach of Blondel et al.(Blondel *et al.* 2008) does not satisfy the augmenting requirement of consideration of localized persistence of flows in cohesive sub-graphs on the network.

2.3.2 Multi-level Community Resolution Using Stability

Lambiotte et al. (Lambiotte, Delvenne, & Barahona 2008) describe a novel approach for multi-level resolution of the modular structure of networks. This method utilizes the stability of network partitions as a quality function. The correlation between stability and modularity is established through symmetric graph counterparts for the directed, weighted case. This implies modularity optimization on a symmetric graph translates into stability optimization for the original graph. Stability optimization favors communities which make it hard for a random walker to exit them implying they contain more than expected information flow within them. Time is used as the resolution parameter. Larger values of the resolution parameter result in larger and few communities. The approach circumvents the resolution limit issue inspite of using modularity optimization as the base process. Hence, it has the efficiency of modularity optimization methods and produces results coherent with information flow dynamics of the network.

Chapter 3

METHOD

As described in the introductory chapter, our objective is to mine the activity flow graph for a hierarchical workflow. In the survey of three community detection approaches we saw that discovering quality workflows will require consideration of structural patterns as well as activity flow patterns in the activity flow graph. An amalgamation of two, not necessarily mutually exclusive, insights into the activity flow graph are required:

1. Static Structural Patterns: These patterns are the micro-structures and inter-dependencies which lent themselves to the formation of the network. These are mined based on localized cohesion.

2. Dynamic Flow Patterns: These patterns help us understand the dynamics of the graph. The flow in a graph is bounded by the static structural properties of the graph. These patterns exist for longer distances and are suggestive of coupling of localized cohesive sub-graphs.

Further discovering hierarchical structure of a workflow requires multi-level resolution of the above patterns. The method presented by Lambiotte et al. in (Lambiotte, Delvenne, & Barahona 2008) incorporating community detection using (Blondel *et al.* 2008) and (Arenas *et al.* 2007) is a fitting choice for these requirements and scales well for large graphs too.

The algorithm has been implemented as a plugin for Gephi(Bastian, Heymann, & Jacomy 2009), an open source network visualization and exploration tool. The experiments in this work have been performed using the Gephi implementation. It is observed that for the example network considered in (Rosvall & Bergstrom 2008) (see Fig. 2.1(a)), the plugin produces a network partition identical to that produced by Infomap (see Fig. 2.1(c)), for the resolution parameter value of 0.8. In the example network in (Rosvall & Bergstrom 2008) (see Fig. 2.1(b)) every node is either a global source or sink. This structure is highly improbable in the context of workflows.

Our method can broadly be laid out into three phases described in the sub-sections to follow. The three phases are that of community detection, trimming low confidence structures and knitting the abstraction with higher level abstractions in the hierarchy if any.

3.1 Identifying Workflow Modules

This phase deals with community detection in a sub-graph of the activity flow log. The output of this phase is an optimal partition of the subgraph into communities. We begin by applying Lambiotte et al.'s method on the subgraph with resolution parameter 1.0. Resolution parameter of 1.0 yields a partition with optimal modularity. Smaller (or larger) value of the resolution parameter will yield finer (or coarser) granularity partitions i.e. more (or less) number of communities. The modularity of the partitions decreases in either case.

Resolution parameter values greater than 1.0 yield coarser granularity partitions which are formed by merging communities detected with resolution parameter 1.0. But, the partition obtained using resolution parameter 1.0 has optimal modularity. As a rule, we will begin with a value of 1.0 and successively decrease the value until a partition with more than one community is found.

The GLOBE activity flow graph yields 4 communities in this phase for resolution parameter value of 1.0. The observed optimal modularity is 0.434 which is indicative of strong community structure. Figure 3.2(a) shows a bird's eye view of the partitioned activity flow graph. Figure 3.1 shows the community membership for all activities after this phase. We see that members of each community have consistent prefixes. This indicates that communities represent the logical grouping of activities in GLOBE.

Community 2 contains activities with prefix "cc". These activities represent the core case creation workflow activities involved in the series of steps shown in Figure 1.2. These include the case geometry, source and other metadata needed to be completed before committing a case. This community forms the Case Creation Ecosystem in GLOBE.

Community 3 represents the Case Editing Ecosystem. It is formed by activities related to the editing and deleting of committed cases. Hence, it is justified that these activities form a separate community even though they have the "cm" prefix similar to activities in Community 1.

Community 1 is the largest of all detected communities and contains activities pertinent to the following user activity ecosystems :

1. Case Collections Ecosystem (e.g. collection creation, deletion, viewing/listing etc.)
2. User Account Ecosystem (e.g. change password, user profile updates, login/logout etc.)
3. Representativeness and Similarity Analyses Ecosystem (e.g. analyze, save/resume analysis etc.).

The activities in Community 1, when subjected to the second iteration of community detection, form sub-communities consistent with the logical grouping. Figure 4.3(a) in Chapter 4 shows the sub-communities of Community 1. All the analyses activities form a separate sub-community.

FIG. 3.1. Level 1 Community Membership

Community 4 represents a new optional feature-set relating to data sheets which was a recent introduction to the system. This feature is part of one of the three steps in the case creation workflow but it's use is optional. Given that this is a new and optional feature the activity flow into and out of the feature is low relative to older activities in GLOBE. Hence the community detection method classifies all activities related to this new feature (bearing prefix "dt") into a separate cohesive community since they have less than expected links with activity nodes in other communities. This follows from the intuition for applying flow-based community detection to mine abstract units of a workflow. The Infomap approach produces the same network partition for our network thus validating this partition from a

flow oriented perspective.

Further drill down, to sub-communities within each community, is possible by isolating the sub-graphs representing individual communities and subjecting them to community detection. This is outlined in Blondel et al..(Blondel *et al.* 2008) for finding out community structure of communities. The drill down may continue until some exit criteria are satisfied. In the context of workflow mining, an intuitive exit criteria exercisable in any combination or singly are:

1. Modularity Threshold: Modularity greater than 0.3, indicative of strong community structure, may be sufficient in most cases. For finer granularity this threshold may be relaxed further.

2. Resolution Parameter Threshold: The resolution parameter is indicative of the granularity of the resultant graph partition in community detection. Based on the application domain and the objective a limit on the granularity may be set. This limit translates into a limit on the resolution parameter in addition to that on the modularity.

3. Limit on Graph Properties: This may include relative proportions of community sizes and ratio of inter-community and intra-community flow.

A comprehensive discussion on and analysis of exit criteria is beyond the scope of this work and is left for future work. Nevertheless, this brief discussion does highlight the extensibility for diverse requirements.

3.2 Inter-module Linkages and Trimming the Flow Network

The goal of this phase is to find and retain high confidence directed edges among nodes in different communities while trimming low confidence ones. This enables identification

of high probability pairwise community interactions as well as community entry (source) and exit (sink) activity nodes for inter-community interactions. Intuitively, nodes with comparatively greater activity inflow (outflow) for the aggregation of all pairwise community interactions in which their communities participate should be the most probable source (sink) nodes for each community.

We use the modularity preserving graph reduction approach described in (Arenas *et al.* 2007) for this phase. Post community detection the partitioned graph may be reduced to a macro-graph, G'=(V', E'), using modularity preserving graph reduction, with:

$$V' = \{\, C \cup A : C \subseteq \{Communities\ at\ current\ level\ of\ hierarchy\},$$
$$A = V - \{\forall c \in C, activities\ which\ are\ members\ of\ c\}\},$$
$$E' = \{\, (v'_1, v'_2, w') : v'_1, c_2 \in V',\ w \in \mathbb{Z}^+,\ (v'_1, v'_2, w) \neq (v'_2, v'_1, w)\},$$
$$and$$
$$edge\ e = (v'_1, v'_2, w')\ represents\ a\ next-step\ relationship\ between\ v'_1\ and\ v'_2$$
$$with\ w' = modularity\ preserving\ weight\ for\ the\ relationship.$$

This reduction approach enables us to collapse and expand individual communities while retaining other communities in their collapsed or expanded form. Thus, weights for community-community and node-community next-step relationships can be computed.

The process for identifying source and sink nodes in each community is as follows:

1. Compute the sum of inflow (outflow) over all inter-community edges directed into (out of) the community.

2. Set a flow coverage target: A flow coverage target is the percentage of inter-community activity flow that the workflow abstraction should capture for it to be a valid representation of user behavior. This is set for both the inter-community in-

flow and outflow to enable discovery of source and sink nodes in the community. Two cases arise when setting a flow coverage target for a node's inter-community inflow (outflow):

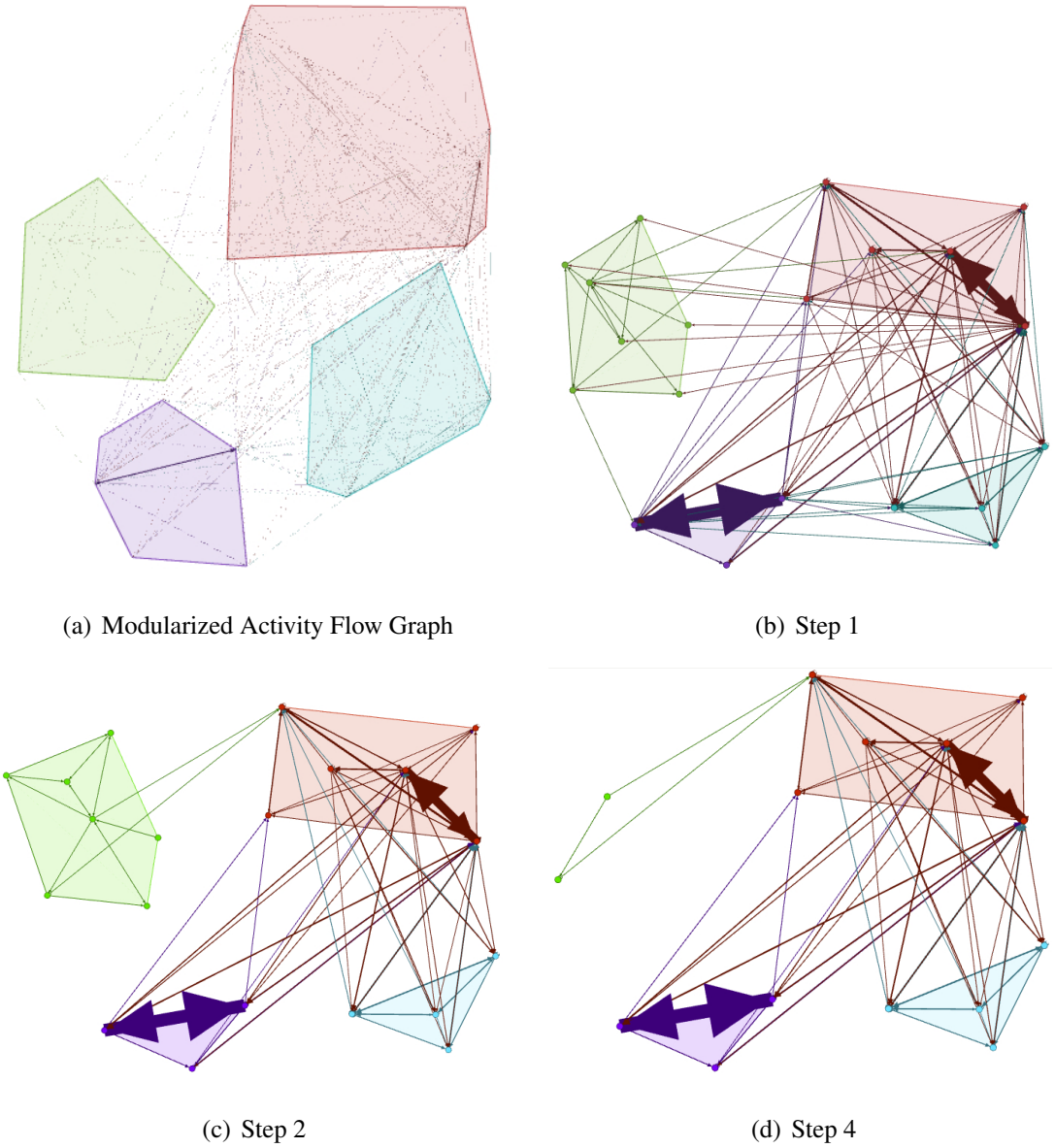
(a) Non-uniform flow distribution: We apply the Pareto principle (Newman 2005) and set the flow coverage target at 80% of the total inter-community flow. This helps discover 20% of the community's nodes which are vital contributors to that community's inter-community activity flow.

(b) Near-uniform flow distribution over edges: In this case applying the Pareto principle to the activity-wise inter-community flow would result in information loss in the context of a recommendation knowledge base. Thus, we set the inter-community flow coverage target at 100%.

Uniformity of flow distribution can be assessed using the chi-square goodness of fit test (D'Agostino & Stephens 1986).

3. We sort the nodes in decreasing order of their inflow (outflow) and select source (sink) nodes one by one till the flow coverage target is achieved.

All activity flow graph elements other than those required for achieving the flow coverage target above may be trimmed to get a relatively more deterministic activity flow network. This trimmed network represents the most frequent user activity transitions between communities.



(a) Modularized Activity Flow Graph

(b) Step 1

(c) Step 2

(d) Step 4

FIG. 3.2. Level 1 Workflow Abstraction: Trimming Stages

The following activity flow network elements, which are not vital to the inter-community activity flow, are trimmed in a sequence of steps in this stage:

Step 1: Community Internal Activities: Nodes which do not have any incident inter-community edges.

Step 2: Low Confidence Edge Activities: Nodes, all of whose incident inter-community edges do not contribute to the flow coverage target set for the community.

Step 3: Low Confidence Inter-community Transitions: A portion of edges incident on community edge nodes which do not contribute to the flow coverage target.

Step 4: Community Internal Activities: Upon completion of Step 1 through 3 some edge nodes may be rendered internal nodes due to their adjacent nodes from other communities being trimmed in earlier trimming steps.

Step 5: Intra-community Transitions: Edges internal to the community should be trimmed as they are irrelevant to the current level of abstraction hierarchy.

Figure 3.2 shows the level 1 community partition followed by the partition state post trimming steps 1 through 4. It helps visualize the reduction in complexity of the workflow abstraction at any level due to trimming. Trimming applied after community detection at each finer granularity will ensure entropy reduction of the resultant workflow model. Identifying source and sink nodes in each community brings the community subgraphs closer to standard workflows in structure. Simplifying the model, the knowledge base for recommendation systems, implies relatively more determinism and faster recommendation logic.

Table 3.1 shows decrease in complexity of the workflow abstraction at level 1 as the trimming phase progresses. The node diversity (Eagle, Macy, & Claxton 2010), (Bonchev & Buck 2005) is the Shannon entropy of a node normalized by the log of its degree. Edge directionality is not considered when computing this. Hence, for each node both its inflow and outflow is considered in the computation. That is,

$$NodeDiversity_i = \frac{-\sum_{j \epsilon Neighbors(i)} [p_{i,j} \ * \ lg(p_{i,j})]}{lg(indegree_i + outdegree_i)}$$

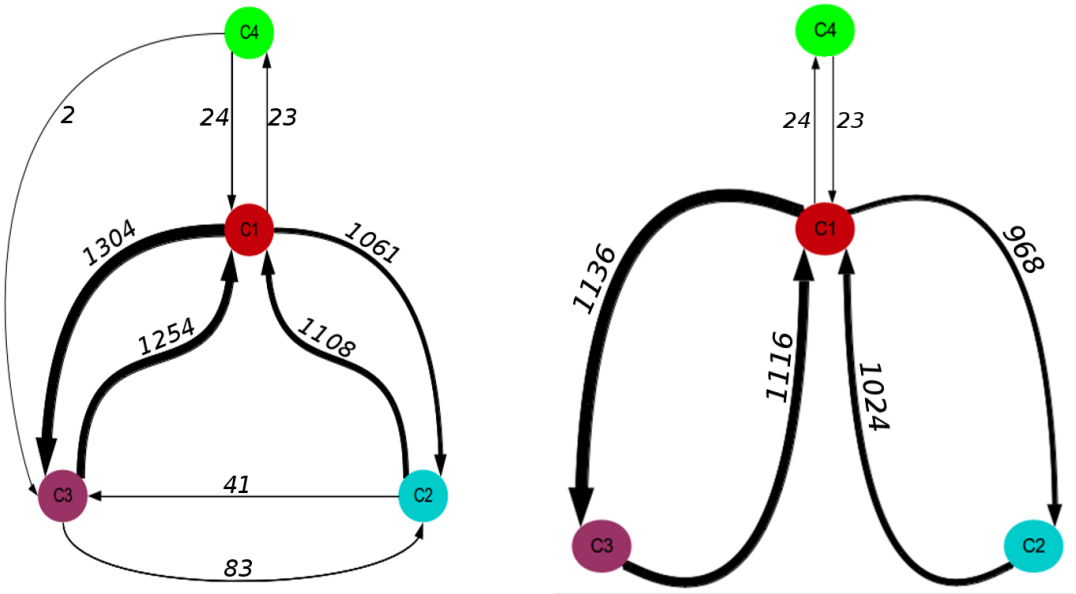
The node diversity compares the entropy of a node with weighted edges to that of the node if its edges were equally probable. Thus, it is a measure of the reduction in uncertainty due to the probability distribution over a node's edges. We compute the average of the diversity of all nodes in the workflow abstraction after every trimming step (column 4 of Table 3.1). The complexity (in terms of edges and nodes) and average uncertainty per node of the workflow abstraction decreases with every trimming step.

Trimming Step	# Nodes	# Edges	Avg. Node Diversity
1	40	367	0.7277
2	20	148	0.7161
3	20	104	0.7138
4	15	91	0.7034

Table 3.1. Level 1 Abstraction: Stage-wise Network Size and Diversity

Community 4 is seen to have lost most of its activity flow as it manages to retain only 2 of its activities post trimming phase (see Figure 3.2(d)). We have commented on the detection of community 4 and its membership in Section 3.1. This community is a case of near-uniform flow distribution and has a 100% flow coverage target. Additionally, the flow into and out of this community is relatively very low. Hence, inter-community edges in which activities of community 4 participate do not contribute to the flow coverage target of other communities and hence are trimmed. To prevent this, inter-community edges in which members of a community with near-uniform activity flow participate should not be trimmed at any stage. This ensures that 100% flow coverage holds. We will explore an alternative to 100% flow coverage for communities with near-uniform flow distribution in Chapter 4 with an example.

The post-trimming macro-workflow in Figure 3.3 shows we have mined a high con-



(a) Pre-Trimming (b) Post-Trimming

FIG. 3.3. Level 1 Workflow Abstraction (Communities Collapsed)

fidence approximate workflow model representing user interaction while meeting the flow coverage target. The nodes are collapsed communities detected in the community detection phase with their components trimmed in this phase. The edges indicate macro-activity flow from one community to another.

3.3 Constructing the Workflow Hierarchy

We define an optimal workflow abstraction, at any level in the hierarchical workflow, as the set of activity nodes and edges in the subgraph under consideration which characterizes the most significant inter-community activity flow. This optimal abstraction is the output of one community detection and trimming iteration. Successive iterations, applied to individual communities, will yield abstractions at successive levels of the top-down hierarchy.

After each iteration, the abstraction should replace the community which it represents in the preceeding level of the hierarchy. When replacing we may have to eliminate inter-community edges which involve an activity in the community which was trimmed in its succeeding abstraction. Although rare, this occurence will only better the quality of the hierarchical workflow in flow confidence terms.

Chapter 4

AN EXPERIMENT: 2-LEVEL HIERARCHICAL WORKFLOW WITH PRE-DEFINED SOURCE AND SINK

We now apply our method to the data set to construct a 2-level hierarchical workflow. As with any real world case, our interest lies in a specific workflow, that of a GLOBE case creation. For this we identify two activities from the event logs:

1. Source Activity: This activity, with all certainty, indicates the beginning of the workflow at level 1. In GLOBE this is the cc_startcase activity. All incoming transitions for this node are ignored and only outgoing transitions are considered.

2. Sink Activity: This activity, with all certainty, indicates the termination or completion of the workflow at level 1. In GLOBE this is the cc_commitcase activity. All outgoing transitions for this node are ignored and only incoming transitions are considered.

Both source and sink activities are considered as single node communities and the rest of the activities will progress through various phases of our method as described in Chapter 3.

The first iteration of community detection is applied with a resolution parameter value of 1. We obtain 4 communities with the modularity of the partition as 0.429, indicative

of strongly community structure. In Section 3.1 we briefly discussed the case of activities relating to the data sheets feature and the relatively low flow through the community they are members of. Community 4, detected in the current experiment, represents that feature. As per the trimming process described in Section 3.2, we avoid trimming any inter-community transitions in which members of community 4 participate. This ensures 100% inter-community flow coverage for community 4 post trimming phase.

We augment the treatment of this case slightly here. Figure 4.1 shows macro-level inter-community activity flow in which community 4 participates. We see that macro-level outflow distribution for community 4 is non-uniform. If we impose 80% outflow coverage target on the macro-outflow of community 4 we can trim inter-community transitions from activities in C4 to activities in C3, which are low-confidence and possibly noisy transitions. This additional step follows from the Pareto principle applied to the macro-level activity flow and can be generalized to all cases of near-uniform node-level flow distribution. After the trimming phase, including the additional C4 flow trimming step, we obtain the level 1 workflow abstraction shown in Figure 4.2 in micro- and macro-view.

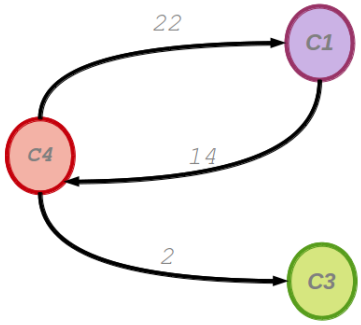
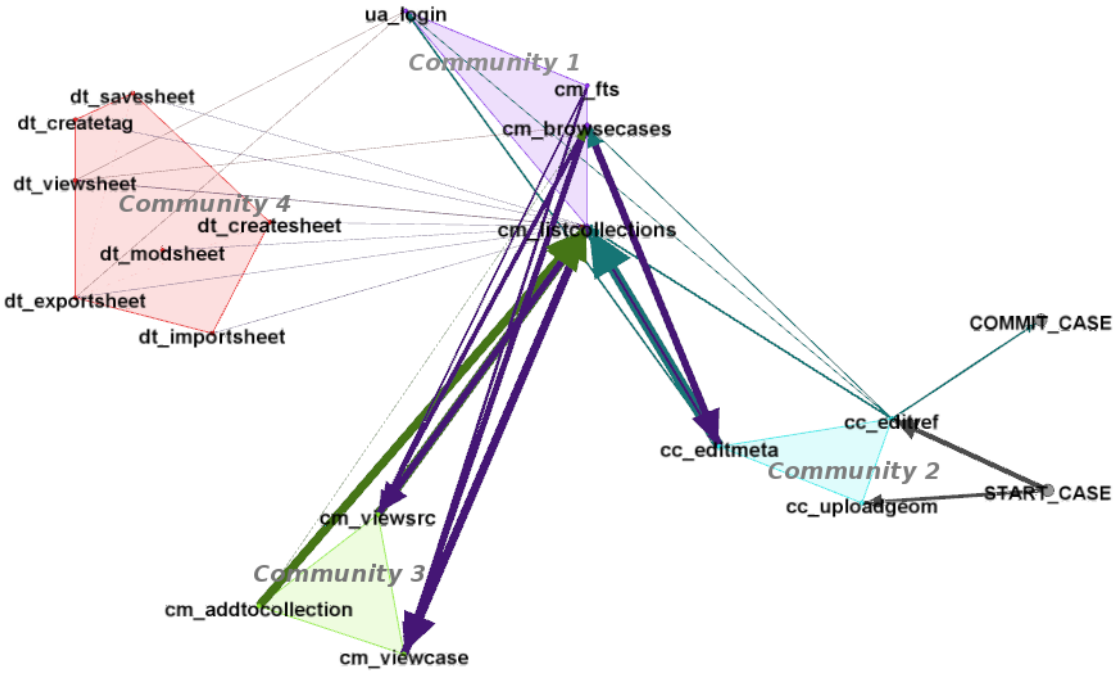
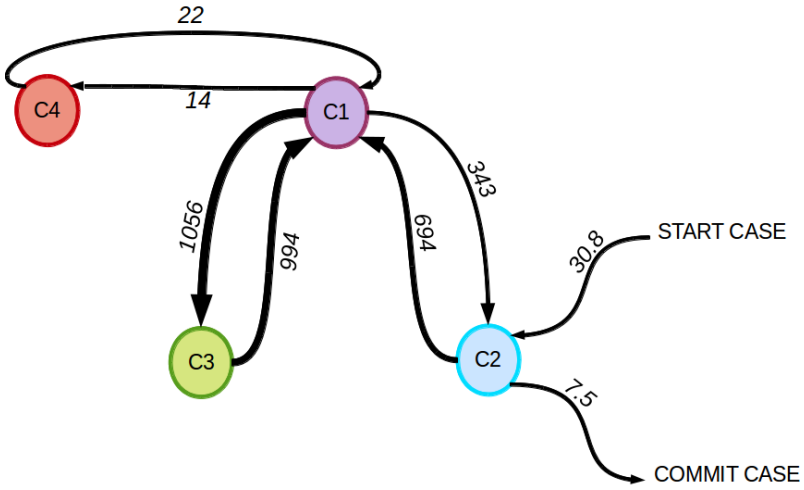


FIG. 4.1. Community 4 : Inter-Community Transitions



(a) Communities Expanded

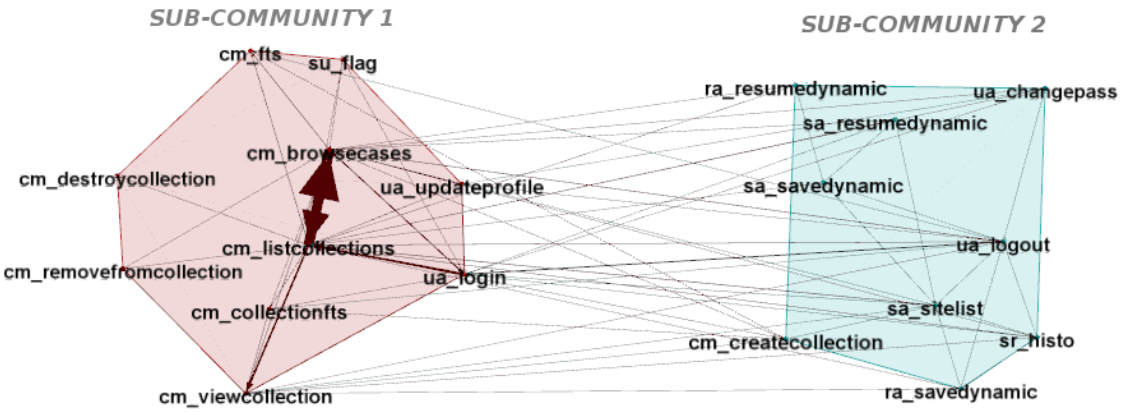


(b) Communities Collapsed

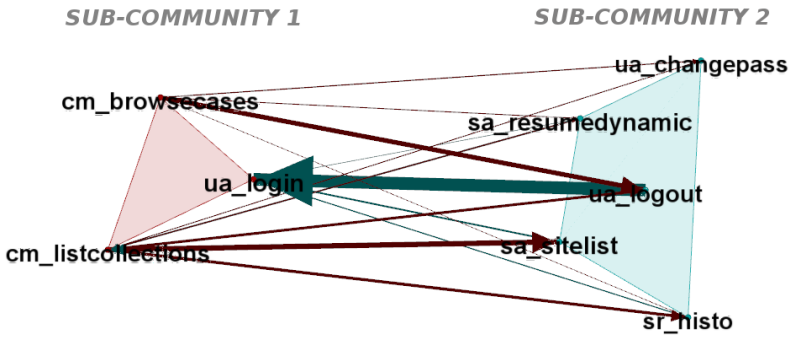
FIG. 4.2. 1-Level Workflow Abstraction with Source and Sink Defined

We choose community 1 for another iteration of our method. Community 1 is the largest of the four Level-1 communities and we seek to find any finer granularity workflow which may be detected. Communities are not detected at resolution parameter=1.0. This suggests lack of adequately strong community structure based on activity flow. Reducing the value of the resolution parameter till we detect communities with modularity greater than zero for the sub-network will reveal micro-structures in the flow. These micro-structures may signify localized clusters of activities on a web-page which may be hard to detect. These micro-structures may also show association between user triggered activities and automated back-end activities in the software. These micro-structures may be of help in productivity analysis and usability profiling.

In the case of community 1, two sub-communities are obtained at resolution parameter = 0.90 with modularity=0.012. The sub-communities and the optimal level 2 workflow abstraction for community 1 are shown in Figure 4.3. The sub-community membership is consistent with the activity groups to which the activities in both sub-communities are mapped. We replace community 1 in the level 1 workflow abstraction with its level 2 abstraction and obtain the 2-level workflow as shown in Figure 4.4

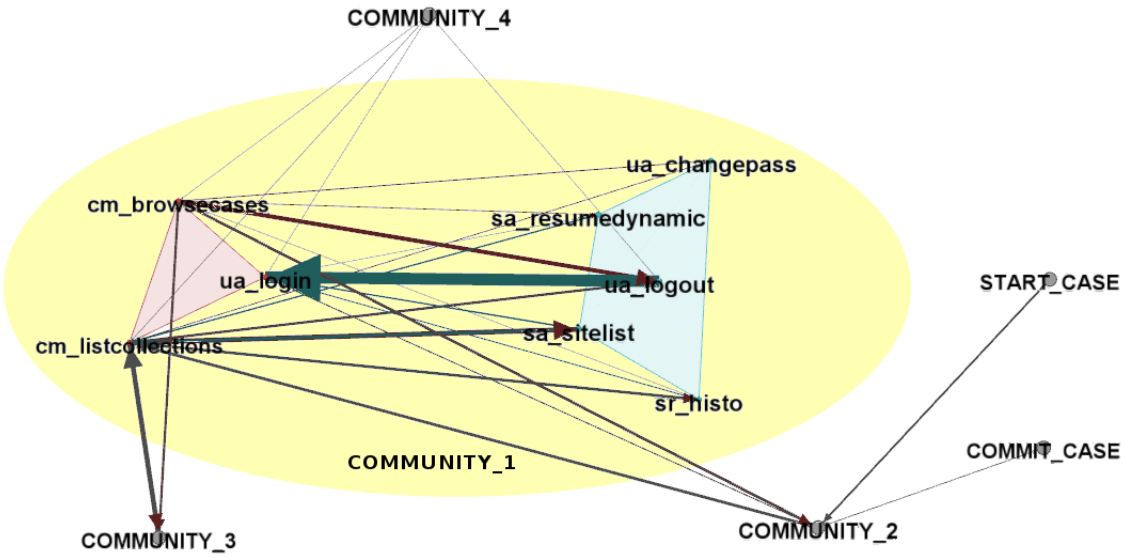


(a) Sub Communities Pre-Trimming

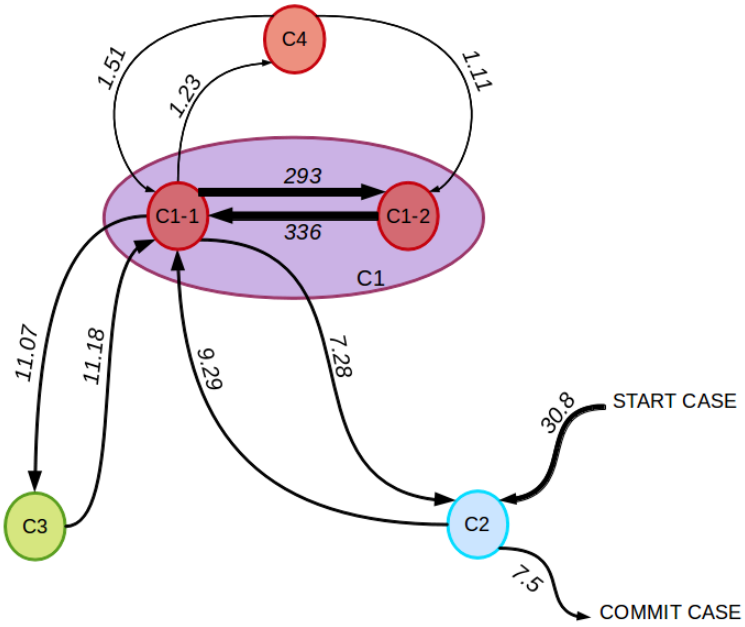


(b) Sub Communities Post-Trimming

FIG. 4.3. Level 2 Workflow Abstraction for Community 1



(a) Level 2 WF Expanded



(b) 2-Level Macro Hierarchy

FIG. 4.4. 2-Level Workflow Abstraction Hierarchy

Chapter 5

CONCLUSION

We have described an intuitive and novel approach to identify abstract hierarchical workflows from raw, fine grained software event logs and clickstreams. It presents a way to obtain low complexity, high likelihood workflow abstractions which may form quality recommendation engine knowledge bases. The method provides significantly accurate workflow modules without requiring activity context data. Communities of activities produced are comparable to those produces by information theoretic community detection methods while performing at par with community optimization approaches. When applied to GLOBE, the community membership is significantly consistent with the logical grouping of activities and the case creation workflow already in place.

We also demonstrated the flexibility provided by the resolution parameter to change the granularity of community detection which allows construction of hierarchical workflows at varying levels. This also facilitates tiered workflow recommendation logic: Inter-community recommendation logic and intra-community recommendation logic. Intra-community logic includes any sub-community transition logic that is discovered at lower levels of granularity. Through all the steps of our method the average node diversity of the workflow abstractions decreases while preserving the vital graph elements which are representative of the high likelihood, true workflow abstraction.

Overlapping communities, i.e. activities common to two or more communities, occur in the context of workflows. The community detection approach discussed here does not handle such cases. For example, activities such as login and logout may occur in two or more communities. These overlap activities add to noise in the workflow mining process and their identification and appropriate handling may better the workflow abstractions. Community detection approaches to handle overlapping nodes are a subject of research. Incorporation of this capability is one of the next logical steps of study and will ensure wide-spread applicability of this method in analytics.

Adapting our approach to handle real-time activity flow updates is a practical requirement for present day recommendation systems too. To achieve this we require efficiency and preservation of original flow information. In the trimming process we lose part of the flow information present in the original network. For large and deep networks, which are common place today, reconstruction of the activity network and obtaining workflow abstractions repeatedly is not feasible either. Thus, alteration and evaluation of this method to enable reinforcement of the hierarchical workflow abstractions over time needs to be explored.

Appendix A

COMMUNITY DETECTION QUALITY FUNCTIONS

We discussed community detection based on structural features and dynamics of a network in Chapter 2. All three approaches optimize a partition quality function to approximate communities in a graph. Here we briefly discuss each quality function for additional background for the interested reader.

A.1 Modularity

Modularity (Newman & Girvan 2004) of a graph partition is a function of edge densities in communities. It compares the edge density in each community with the expected edge density in that community of nodes for the null graph. That is,

$$Q = (fraction\ of\ intra-community\ edges) - (the\ expected\ fraction)$$

For the directed and weighted graph case, (Rosvall, Axelsson, & Bergstrom 2009) states modularity as :

$$Q = \sum_{i=1}^{m} \frac{w_{ii}}{w} - \frac{w_i^{in} \cdot w_i^{out}}{w^2}$$

where,

m is the number of communities detected

w is the sum of weights of all edges in the graph

w_{ii} is the weights of intra-community edges in community i

w_i^{in} is the inflow of edges in community i

w_i^{out} is the outflow of edges in community i

The first fraction in the summation represents the observed fraction of flow in the community while the second fraction represents the expected fraction of flow given a null graph. Modularity quantifies the deviation of localized edge density distribution over communities from the expected distribution. The larger this deviation the stronger is a graph's community structure. An optimal partition would, therefore, imply maximal modularity for the graph.

A.2 Map Equation

The map equation (Rosvall, Axelsson, & Bergstrom 2009) is used as a quality function in the context of information theoretic community detection approaches. These approaches detect communities by utilising implications of a graph's structure and dynamics on its codification. The map equation gives the average length of the code that describes a step of the random walker (or surfer) over the network.

$$L(M) = q_{\curvearrowright} H(Q) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(P^i)$$

where,

m is the number of communities detected

$H(Q)$ is the frequency weighted average length of codewords in the index codebook

$H(P^i)$ is the frequency weighted average length of codewords in the i^{th} community's module codebook

$q_{\curvearrowright}$ is the probability that the random walker transitions from one community to another i.e. the rate of use of the index codebook

$p^i_{\circlearrowright}$ is the time for which the random walker is trapped in community i before exiting the community i.e. the rate of use of the i^{th} community's module codebook

The map equation gives the usage frequency weighted sum of the codebook entropies. Given this definition, an optimal partition would imply minimal average code length given by the map equation.

A.3 Stability

Stability(Lambiotte, Delvenne, & Barahona 2008) evaluates partitions based on flow dynamics and is a statistical evaluation of a dynamic continuous-time process applied to a graph. Unlike the map equation, which uses an information theoretic perspective. The map equation quantifies the average duration for which a random walker may be trapped in a community. Stability, on the other hand, quantifies the probability that a random walker may revisit a community in a given time window. Time is the resolution parameter in this case and allows control over the granularity of community detection. The stability of a partition of a graph to which a Markov process is applied is

$$R_{\mathcal{M}}(t) = \sum_{\mathcal{C}\epsilon\mathcal{P}} P(\mathcal{C}, t) \; - \; P(\mathcal{C}, \infty)$$

where,

$\mathcal{M}$ is the Markov process

t is the instance in time (resolution parameter) when a partition is retrieved and it's stability computed

$\mathcal{P}$ is the set of all communities detected until time t

$P(\mathcal{C}, t)$ is the probability that the random walker started in community $\mathcal{C}$ and revisits

it at time t

$P(\mathcal{C}, \infty)$ is the probability that the process loses memory of the initial conditions, i.e. of the walker having started in community $\mathcal{C}$

An optimal partition would imply maximal stability. There exists a correlation between stability and modularity. In the case of directed and weighted graphs, stability optimization of a given graph is equivalent to modularity optimisation of another graph which is "manifestly symmetric" to the given graph. Hence, the problem of stability optimization may be transformed into one of modularity optimization. This is done in the interest of computation efficiency without loss of effectiveness of community detection as described using stability as a quality function.

REFERENCES

[Arenas *et al.* 2007] Arenas, A.; Duch, J.; Fernández, A.; and Gómez, S. 2007. Size reduction of complex networks preserving modularity. *New Journal of Physics* 9:176.

[Baglioni *et al.* 2003] Baglioni, M.; Ferrara, U.; Romei, A.; Ruggieri, S.; and Turini, F. 2003. Preprocessing and mining web log data for web personalization. In Cappelli, A., and Turini, F., eds., *AI*IA 2003: Advances in Artificial Intelligence*, volume 2829 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 237–249.

[Bastian, Heymann, & Jacomy 2009] Bastian, M.; Heymann, S.; and Jacomy, M. 2009. Gephi: An open source software for exploring and manipulating networks.

[Blondel *et al.* 2008] Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 10:8.

[Bonchev & Buck 2005] Bonchev, D., and Buck, G. 2005. Quantitative measures of network complexity. In Bonchev, D., and Rouvray, D., eds., *Complexity in Chemistry, Biology, and Ecology*. Springer US. 191–235.

[Chatterjee, Hoffman, & Novak 2003] Chatterjee, P.; Hoffman, D. L.; and Novak, T. P. 2003. Modeling the clickstream: Implications for web-based advertising efforts. *Marketing Science* 22(4):520–541.

[D'Agostino & Stephens 1986] D'Agostino, R. B., and Stephens, M. A. 1986. *Goodness-of-fit techniques*. New York, USA: Marcel Dekker Inc.

[De Bock & Van den Poel 2010] De Bock, K., and Van den Poel, D. 2010. Predicting

website audience demographics forweb advertising targeting using multi-website clickstream data. *Fundam. Inf.* 98(1):49–70.

[Eagle, Macy, & Claxton 2010] Eagle, N.; Macy, M.; and Claxton, R. 2010. Network diversity and economic development. *Science* 328(5981):1029–1031.

[Eirinaki & Vazirgiannis 2003] Eirinaki, M., and Vazirgiannis, M. 2003. Web mining for web personalization. *ACM Trans. Internet Technol.* 3(1):1–27.

[Facca & Lanzi 2005] Facca, F. M., and Lanzi, P. L. 2005. Mining interesting knowledge from weblogs: A survey. *Data Knowl. Eng.* 53(3):225–241.

[Fortunato & Barthlemy 2007] Fortunato, S., and Barthlemy, M. 2007. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104(1):36–41.

[Herbst 2000] Herbst, J. 2000. A machine learning approach to workflow management. In Lpez de Mntaras, R., and Plaza, E., eds., *Machine Learning: ECML 2000*, volume 1810 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 183–194.

[Jones & Oates 2010] Jones, J., and Oates, T. 2010. Learning deterministic finite automata from interleaved strings. In Sempere, J., and Garca, P., eds., *Grammatical Inference: Theoretical Results and Applications*, volume 6339 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 80–93.

[Lambiotte, Delvenne, & Barahona 2008] Lambiotte, R.; Delvenne, J. .; and Barahona, M. 2008. Laplacian Dynamics and Multiscale Modular Structure in Networks. *ArXiv e-prints*.

[Lu, Dunham, & Meng 2006] Lu, L.; Dunham, M.; and Meng, Y. 2006. Mining significant usage patterns from clickstream data. In *Proceedings of the 7th International Conference*

on Knowledge Discovery on the Web: Advances in Web Mining and Web Usage Analysis, WebKDD'05, 1–17. Berlin, Heidelberg: Springer-Verlag.

[Magliocca *et al.* 2014] Magliocca, N. R.; Ellis, E. C.; Oates, T.; and Schmill, M. 2014. Contextualizing the global relevance of local land change observations. *IOP Conference Series: Earth and Environmental Science* 18(1):012107.

[Malliaros & Vazirgiannis 2013] Malliaros, F. D., and Vazirgiannis, M. 2013. Clustering and community detection in directed networks: A survey. *Physics Reports* 533(4):95 – 142. Clustering and Community Detection in Directed Networks: A Survey.

[Newman & Girvan 2004] Newman, M. E. J., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Phys. Rev. E* 69:026113.

[Newman 2005] Newman, M. 2005. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46:323–351.

[Rosvall & Bergstrom 2008] Rosvall, M., and Bergstrom, C. T. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105(4):1118–1123.

[Rosvall, Axelsson, & Bergstrom 2009] Rosvall, M.; Axelsson, D.; and Bergstrom, C. T. 2009. The map equation. *The European Physical Journal Special Topics* 178(1):13–23.

[Van der Aalst, Weijters, & Maruster 2004] Van der Aalst, W.; Weijters, T.; and Maruster, L. 2004. Workflow mining: discovering process models from event logs. *Knowledge and Data Engineering, IEEE Transactions on* 16(9):1128–1142.

[Yaman, Oates, & Burstein 2009] Yaman, F.; Oates, T.; and Burstein, M. 2009. A context driven approach for workflow mining. In *Proceedings of the 21st International Jont*

Conference on Artifical Intelligence, IJCAI'09, 1798–1803. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.