Global Learning Network Analytics to Enhance PLN Understanding

Bernhard Göschlberger
Research Studios Austria FG
Linz, Austria
goeschlberger@researchstudio.at

Matthias Steinbauer
Johannes Kepler University Linz
Linz, Austria
matthias@steinbauer.org

ABSTRACT

In the 21st century learning increasingly happens on the social web. Learning has evolved into an interactive social process producing large amounts of data across a multitude of inhomogeneous systems. To identify the role of individual actors and groups of actors the whole global learning network needs to be analyzed. The work presented in this paper ingests learning data into a cloud hosted distributed temporal graph model with a supporting distributed processing framework to calculate global graph metrics. The presented simple architectural approach builds upon the xAPI specification to ensure compatibility and flexibility. Based on the global graph metrics we can detect communities and identify information brokers. This information enhances the understanding of the learner's personal learning network and its development over time. It contributes to ongoing efforts to guide learners' through the tangled undergrowth of the global social learning network towards individuals and communities relevant to their interests, skills and aptitudes.

CCS CONCEPTS

• **Applied computing → Collaborative learning**; • **Computer systems organization** → *Cloud computing*;

KEYWORDS

learning analytics, social network analysis, personal learning network (PLN), learning network graph, Experience API (xAPI), large scale distributed temporal graphs

ACM Reference Format:
Bernhard Göschlberger and Matthias Steinbauer. 2017. Global Learning Network Analytics to Enhance PLN Understanding. In *iiWAS '17: The 19th International Conference on Information Integration and Web-based Applications & Services, December 4–6, 2017, Salzburg, Austria.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3151759.3151843

1 INTRODUCTION

Social interactions play a key role in the process of learning. Connectivism — a learning theory proposed by Siemens [13] — assumes that knowledge is acquired through the connections between the learner and his personal learning network (PLN). A PLN is the set of interpersonal connections to other people related to the personal learning process. Connectivist Massive Open Online Courses (cMOOCs) build upon connectivism. Instead of using a single platform driven approach like traditional MOOCs, cMOOCs use the social web (Facebook, Twitter, YouTube, GoogleDocs) to connect course participants, to host and share learning artifacts, and to collaboratively learn. Their non-hierarchical structure, constant evolvement over time, and massive amount of participants makes them large-scale temporal social networks. These networks are particularly hard to analyze as interaction data is inhomogeneous and spread across multiple systems and platforms. The ExperienceAPI (xAPI) was designed to overcome exactly these problems. A wide range of systems that adopt the standard as well as data scraping tools that import data from non-adopting systems (see e.g. [6]) help to collect all relevant information.

While the term PLN emphasizes the micro-scale of learning networks i.e. the local network of an individual, macro-scale analytics of the network provide essential information about roles of individuals and affiliation to communities. We strive to derive global information from the network (communities, trends, etc.) to enrich the results in local PLN analytics.

Pentland et.al. [1, 10] demonstrated that analyzing social networks does not only lead to a better understanding of the network, but also of its nodes. Preferences, interests and skills can be deducted from personal social networks. Personal learning networks evolve over time and changes in the personal learning network reflect information about the development of the learners' focus, preferences and aptitudes. Mucha et.al. [9] introduced an approach that accounts for the temporal dynamics of networks.

In this paper, we use the work of Pentland et.al. and Mucha et.al. as the theoretical base model for novel ways of analyzing learner interaction. We claim that, social interaction data generated by any learning system supporting the xAPI, can be represented as a temporal, dynamic, directed, weighted graph. We use a novel platform for distributed temporal graph processing in our prototypical implementation of a learning dynamics analytics platform. The tools originating from this work are designed to be scalable and thus are able to accommodate to very large temporal graphs that occur in real world learning networks as in cMOOCs.

The paper at hand documents the current progress in designing the architecture for this system, the status and experiences with the already implemented parts of the system, and gives an outlook on prospective interesting metrics that can be computed with the proposed architectural approach.

iiWAS '17, December 4–6, 2017, Salzburg, Austria

ACM ISBN 978-1-4503-5299-4/17/12…$15.00
https://doi.org/10.1145/3151759.3151843

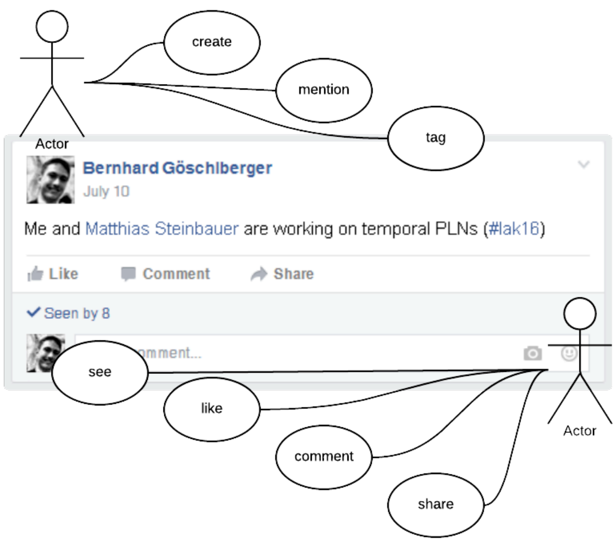

Figure 1: Social Interaction in social network applications

The remainder of this paper is structured as follows: In section 2 we elaborate on social interaction data in the context of learning. In section 3 we briefly describe our distributed temporal graph processing system and its concepts. In section 4 we present our architectural solution and our graph model for inferring and analyzing temporal personal learning networks from learning data. In section 5 we list our results and give an outlook on future work before we conclude in section 6.

2 SOCIAL LEARNING EXPERIENCE

While previous standardization approaches in technology enhanced learning focused mainly on content interoperability, current initiatives have different foci. One of the most notable recent developments was the specification of the xAPI as an essential part of the Training and Learning Architecture (TLA) envisioned by the Advanced Distributed Learning Initiative (ADL). The xAPI specification defines the structure of xAPI statements and a web service interface. By design, it does not constrain the available vocabulary. The ADL expects the respective Communities of Practice (CoP) to agree on a controlled vocabulary for their respective domain. The controlled vocabulary consists of verbs and activity types. For our approach, we build on the work of the social collaboration CoP, which focuses on various kinds of collaborative systems like social network applications (e.g. Facebook), wikis or Sharepoint. It strives to agree on a list of verbs and activities expressing (most of) the interactions within such systems. Whilst discussion is still ongoing, some essential actions in social network applications have been pointed out by Kitto et.al. [6].

Figure 1 illustrates a set of typical social interactions in social media applications. In this example, some potential social interactions on the social web platform Facebook are displayed. The simple Facebook post already illustrates the many different types of social actions possible. Actions initiating social interaction such as creating a post, referring to something (tag) or somebody (mention) can trigger reactions of other actors in the system. For communication the act of receiving is essential (Shannon-Weaver model [12]). Within typical social network application actions like see, read, watch, click or open represent receiving acts. Social interaction is established by reactions to social actions. The action of creating an artifact (text, photo, video, etc.) becomes part of a social interaction when others read, like or share it.

Social interactions connect people. Dependent on the users' intent, different grades of involvement are shown. This involvement is expressed through the use of different actions (see, like, comment, share). Related work [2, 4] suggests that frequency, quality and importance of social interactions between two persons can be used as metrics for social bonds. Our model uses similar dimensions: the aforementioned social involvement [ι] and the directedness [δ]. The social involvement is inherent to the type of social action chosen by the actor. Directedness accounts for the person's intent to connect with a specific recipient or a specific group of recipients. Social interactions that address larger groups have a smaller intent to create or intensify interpersonal bonds. A person of public interest that creates a post that is read, liked and commented by thousands of people has smaller intent to create or intensify his social bonds individual followers than a person creating a post for a small group of selected individuals. Directedness of social actions is a measure for the intent of targeting a specific person with that action. As we distribute the action equally over all addressed individuals the directedness δ is $\delta = 1/n$, where n is the number of addressees.

3 TEMPORAL GRAPH PROCESSING

In this paper we rely on the definitions of a dynamic and temporal graphs as explained in [5] and [7]. We use slightly different signs and symbols compared to the original work to avoid ambiguities.

A graph G is a pair (V, E) where V is a finite set of vertices, and E is a finite set of edges of ordered pairs $\{u, v\}$ with $u \in V$ and $v \in V$. A graph G can be called *vertex-dynamic* if the set V varies over time, and *edge-dynamic* if the set E varies over time. Learning networks are both vertex- and edge-dynamic as learners, artifacts, etc. are modeled as vertices and can occur and be removed at any point in time, and edges between these vertices can be added and removed any time. A vertex- and edge-dynamic graph is a dynamic graph G_d.

A temporal graph T is a set of graphs $G_0, G_1, G_2, \ldots, G_t$ where each $G_t = (Vt, Et)$ such that any $G_t \in G_d$ is a snapshot of the dynamic graph at time t. Ordered sets $\tau = G_x, G_{x+1}, G_{x+2}, \ldots, G_{x+n}$ with $\tau \in T$ are selections of time-spans $x \ldots x+n$ from the temporal graph.

3.1 DynamoGraph

As the processing layer of our analytics platform we use the DynamoGraph [14] platform which is a distributed implementation of the aforementioned temporal graph model. DynamoGraph is both a temporal graph storage and processing framework. Its users can use service method calls to manipulate graph data (i.e. add, modify, and remove vertices and edges).

DynamoGraph relies on cloud-based computing infrastructure and is capable of distributing temporal graph data and processing over many compute nodes such that the system can accommodate for a growing amount of data. Moreover, the size of the used compute cluster can be adjusted during runtime. The system provides mechanisms for fault tolerance.

Further DynamoGraph relies on distributed computing concepts and is inspired by Pregel [8]. Pregel-style algorithms execute a fully distributed compute function $c(v, \Gamma)$ over every vertex v in the graph and the global memory Γ, followed by a messaging phase which allows any vertex v to send a message to any other vertex y. Algorithms in Pregel halt when every vertex has voted to halt the algorithm.

DynamoGraph provides several extensions over the original Pregel computing concept: (1) between each compute iteration a global function γ [11], (2) temporal selection and aggregation of vertices in a time-frame τ, (3) aggregation of sets of distributed algorithms that can be executed in a batch. The framework supports the automatic execution of graph algorithm in certain intervals of time and upon certain events. This way it is possible to continuously evaluate metrics and to live monitor temporal graphs.

3.2 Temporal Graph Metrics

The used framework provides a standard set of graph metrics. The following highlights several metrics we propose to be used for PLN analytics.

Centrality measures can be computed to determine how strongly any given vertex is connected to its neighborhood. As static information, this allows us to find strongly connected learners and artifacts. In the temporal use case, one can observe the evolution of centrality over time. On a macro-scale this metric can then be used to observe if a group of learners successfully involves into a certain topic. On a micro-level this allows to draw conclusions about personal learning progress.

A special case of centrality is **betweenness centrality** [3]. Nodes with high betweenness-centrality are naturally found at the edges of structural clusters in a graph and usually mark brokers between these structural clusters. In temporal analysis betweenness centrality is interesting because it can show how learners progress from learning a single topic to a multi-disciplinary learner. In PLN analytics, we can use it to identify and recommend relevant brokers and learning communities.

Through **neighborhood analysis** direct vertex neighbors (and possibly their neighbors) can be analyzed. By comparing the vertex neighborhood from different time-spans, one can quantify learning progress (Is the progress fast or slow? Are there many topic switches in a PLN?). It is assumed that the PLN of an active learner will grow over time.

4 LEARNING NETWORK GRAPH

As mentioned before our goal is to create a global view on social learning networks to identify communities, trends and relevant central brokers to enrich the results in local PLN analytics. As laid out in the previous sections our approach uses methods from temporal graph analysis to derive the relevant graph metrics.

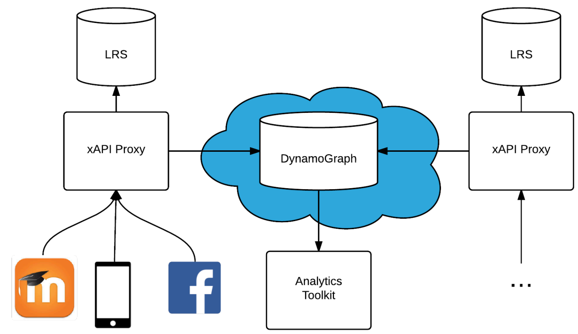

Figure 2: System overview - xAPI proxy to extract graph data

4.1 Prototype Architecture

In practice, this necessitates the transformation of learning data into graph structure. Technically speaking our prototypical implementation is capable of extracting social graph data from xAPI compliant systems. This is achieved using an xAPI proxy layer. The proxy layer sits between an xAPI statement generator and the respective learning record store (LRS) and forwards extracted social interaction data to the DynamoGraph API.

Figure 2 shows the overall architecture of our system. The incoming xAPI statements that are produced by various generators pass through the proxy layer to the respective LRS. As displayed in the figure a multitude of proxy instances populate extracted social interaction data to a single cloud based DynamoGraph instance.

DynamoGraph offers query interfaces to retrieve relevant graph metrics, detected communities and information brokers from the resulting temporal graph. Users of the DynamoGraph API can formalize temporal Pregel algorithms, bundle them in Java JAR files and submit these as queries to the cloud-based distributed cluster. This means that DynamoGraph not only provides distributed and scalable temporal graph storage but also distributed processing over this data.

Given this brief description of DynamoGraph it becomes obvious that the system can be used as the base layer for high-level analytics toolkits. These analytics toolkits are responsible for presenting relevant results to their users. This can be achieved for instance through visualization of the PLNs or through dashboards that provide metrics and guidance about a user's PLN in context with the global social learning network. Example for guidance provided by the system could be "We suggest that you learn about the topic XYZ next, since many people in your current communities are currently learning about it!" or "How about joining community ABC, most users there have similar learning modalities!"

4.2 Learning Network Graph Model

Our temporal graph model consists of typed vertices and typed, weighted and directed edges. The vertex types resulting from the xAPI statement specification are listed in Table 1. We create a vertex for each statement alongside with the vertices for actor, object, and (if present) context. We connect the statement vertex with its properties using a bidirectional statement-property edge type.

Table 1: node types to represent xAPI statements in a social interaction graph. Asterisks (*) denote multiple vertices of a type.

statement property	value	required vertex type(s)
actor	agent	agent
	identified group	group
	anonymous group	agent*
object	activity	activity
	agent	agent
	sub-statement	statement
	statement reference	statement
context	instructor	agent
	team	group, agent*
	statement	statement
	contextActivities	activity*

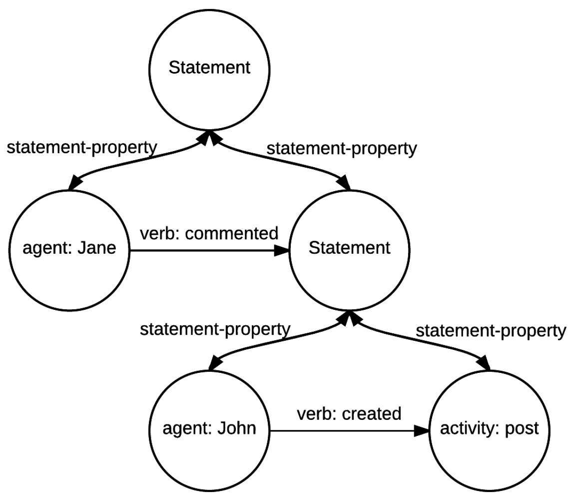

Figure 3: Graph model representing two linked xAPI statements.

Thus, references to a statement are connected to all parts of the statement, i.e. actor, object and context.

Actors and objects are connected by directed edges typed with the verb of the statement. We refer to actions as direct social interaction if an object is an actor or a group of actors.

Figure 3 shows the graph representation of two linked statements. In the displayed example John creates a post and Jane comments on that. Her comment refers to the fact that John created a post and thus connects her with the actor John and the content, the created activity of the type post. The statement vertex mediates the social interaction in the given example. We refer to this type of social interaction as indirect social interaction as Jane does not directly interact with John, but rather with a trace he left. This aligns well with Vygotsky's activity theory [16].

Multiple actions may occur between two vertices. Thus, the graph model is extended to a multi-graph model to preserve the granular information of recorded social actions.

The model of the social learning network itself, however, is represented by a graph. We achieve that by reducing all equally directed edges between two vertices u and v to a single edge $\{u, v\}$ with the weight w_t at a given time t when processing the network graph:

$$w_t(\{u,v\}) = \sum_{\{u,v\} \in E_t} \iota(\{u,v\}) * \delta(\{u,v\})$$

Values for social involvement [ι] and directedness [δ] are assigned to each edge $\{u, v\}$ in the social interaction multi-graph model at creation time.

The chosen approach allows tracing which actions lead to a certain social bonding, as the social network graph model is directly derived from the social interaction multi-graph model. A further benefit of this approach is edge weights can be adjusted post-hoc by refining the assumed values for social involvement for certain types of actions. Finally, we can investigate whether a certain social bond is based on more on frequency, social involvement or directedness. In other words, we can distinguish whether a connection is based on frequent superficial interaction or rather on less frequent, but more intense interaction.

5 RESULTS AND FUTURE WORK

The authors are currently in the process of molding the presented reference architecture in a working scientific prototype. As documented in earlier work [15] the base layer, relevant data import interfaces, and basic temporal graph metrics are available in software. It has been shown that the used data storage and processing architecture is scalable.

First experiments with converting xAPI statements into temporal graph models show promising results. Semi-automatic conversion of scraped xAPI learning data can already be uploaded to the graph processing platform to be visualized and for computing general metrics about this data.

These preliminary results make us keen to continue with this research. We identified the following issues that are to be addressed next:

xAPI Vocabulary: Our current prototype supports a small set of statements and the transformative actions are defined in procedural code of the proxy implementation. We strive to achieve a generic implementation that allows us to define transformations declaratively to ensure extensibility.

Online statement import: As the DynamoGraph API are not yet available as web services, our proxy implementation currently writes the data into a DynamoGraph CSV file which has to be uploaded manually. Hence, we plan to implement a REST API for DynamoGraph to achieve our vision of scalable live learning network analytics.

Model validation: We plan to implement a simple online social learning network application that allows us to collect our own from students without violating their privacy. Although that data naturally lacks the desired massive scale, we can use it to validate important aspects of our model such as effects of direct and indirect social interaction.

Analytics web application: Finally, we want to demonstrate the immediate practical value of our approach by implementing a prototypical learning network analytics toolkit that allows individual learners to review their own metrics, community affiliations, and recommendations.

6 CONCLUSIONS

In this paper, we have presented our early results on temporal analytics of the global learning network to enhance the understanding of personal learning networks (PLNs). We have proposed a reference architecture and have demonstrated its mere technical feasibility through implementing a prototype. We can demonstrate that the prototype is capable of uploading learning networks to the DynamoGraph processing platform. However, up to now experiments were only conducted with artificial learning networks.

We have set the case for ever growing data in the area of PLNs and have argued that analyzing data spanning several learning platforms supporting xAPI will become a crucial part in future learning systems. We have shown that a cloud based solution provides the scalability necessary to implement such a centralized analytics repository.

REFERENCES

[1] Wen Dong, Bruno Lepri, and Alex Sandy Pentland. 2011. Modeling the coevolution of behaviors and social relationships using mobile phone data. In *Proceedings of the 10th International Conference on Mobile and Ubiquitous Multimedia*. ACM, 134–143.

[2] Rebecca Ferguson and Simon Buckingham Shum. 2012. Social learning analytics: five approaches. In *Proceedings of the 2nd international conference on learning analytics and knowledge*. ACM, 23–33.

[3] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

[4] Mark S Granovetter. 1973. The strength of weak ties. *American journal of sociology* 78, 6 (1973), 1360–1380.

[5] Frank Harary and Gopal Gupta. 1997. Dynamic graph models. *Mathematical and Computer Modelling* 25, 7 (1997), 79–87.

[6] Kirsty Kitto, Sebastian Cross, Zak Waters, and Mandy Lupton. 2015. Learning analytics beyond the LMS: the connected learning analytics toolkit. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. ACM, 11–15.

[7] Vassilis Kostakos. 2009. Temporal graphs. *Physica A: Statistical Mechanics and its Applications* 388, 6 (2009), 1007–1023.

[8] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM, 135–146.

[9] Peter J Mucha, Thomas Richardson, Kevin Macon, Mason A Porter, and Jukka-Pekka Onnela. 2010. Community structure in time-dependent, multiscale, and multiplex networks. *science* 328, 5980 (2010), 876–878.

[10] Alex Pentland. 2005. Socially aware, computation and communication. *Computer* 38, 3 (2005), 33–40.

[11] Semih Salihoglu and Jennifer Widom. 2013. Gps: A graph processing system. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*. ACM, 22.

[12] Claude E Shannon and Warren Weaver. 1998. *The mathematical theory of communication*. University of Illinois press.

[13] George Siemens. 2014. Connectivism: A learning theory for the digital age. (2014).

[14] Matthias Steinbauer and Gabriele Anderst-Kotsis. 2016. DynamoGraph: extending the Pregel paradigm for large-scale temporal graph processing. *International Journal of Grid and Utility Computing* 7, 2 (2016), 141–151.

[15] Matthias Steinbauer and Gabriele Kotsis. 2014. Towards Cloud-Based Distributed Scaleable Processing over Large-Scale Temporal Graphs. In *WETICE Conference (WETICE), 2014 IEEE 23rd International*. IEEE, 143–148.

[16] Lev Semenovich Vygotsky. 1980. *Mind in society: The development of higher psychological processes*. Harvard university press.