Interactive Graph Query Language for Multidimensional Data in Collaboration Spotting Visual Analytics Framework

Adam Agocs*, Dimitrios Dardanis*, Jean-Marie Le Goff*, Dimitrios Proios*
*CERN, CH-1211 Geneva 23, Switzerland
E-mail: {Adam.Agocs, Dimitrios.Dardanis, Jean-Marie.Le.Goff, Dimitrios.Proios}@cern.ch

***Abstract*—Human reasoning in visual analytics of data networks relies mainly on the quality of visual perception and the capability of interactively exploring the data from different perspectives. Visual quality strongly depends on networks' size and dimensional complexity while network exploration capability relies upon the intuitiveness and expressiveness of user frontends. The approach taken in this paper aims at addressing the above by decomposing data networks into multiple networks of smaller dimensions and building an interactive graph query language that supports full navigation across the sub-networks. Within sub-networks of reduced dimensionality, structural abstraction and semantic techniques can then be used to enhance visual perception further.**

***Keywords*–Visual analytics; labelled graph; graph query language; visualisation; patents and publications**

I. INTRODUCTION

According to an English idiom, "A picture is worth a thousand words". Visual analytics aims to combine the power of visual perception with high performance computing in order to support human analytical reasoning. Since Wong et al. [1] in 2004, visual analytics has been widely used in various fields, such as biology or national security but also in other fields, such as climate monitoring [2][3] or social networks analysis, the field originally addressed by the Collaboration Spotting project (CS). Multidimensional networks built out of interconnected elements contained in datasets and represented as directed and labelled graphs are a natural means of representing data for visual analytics. These graphs - often referred to as knowledge graphs - comprise labelled nodes and relationships and their data schemas are graphs of labels that correspond to the networks' dimensions.

Graphs as database models and graph query languages defined over these models have been investigated for some 30 years [4]. These models and languages have been used in many applications using a wide spectrum of data (e.g., biology, social network and criminal investigation data), clearly indicating that the combination of visual analytics with graph query languages has become quite popular.

According to Wong et al. [5], one of the biggest challenges in visual analytics is *User-Driven Data Reduction* which calls for "*a flexible mechanism that users can easily control according to their data collection practices and analytical needs*" to reduce the amount of data [6]. This essentially entails an improvement of the visualization clarity and an escalation of data processing performances irrespective of the increasing complexity of the data over the years. To meet this challenge, semantic and structural abstraction techniques, such as clustering, collapsing, extraction and demonstration of relationships among graph entities can be used [7] at the expanse of a loss of information on the network content [8].

Dimension reduction is central to the visualization of data networks since it enables users to increase their insight into the data. The approach taken in the Collaboration Spotting project is to reduce the dimensional complexity of data networks while maintaining the information about their content. It consists in decomposing directed and labelled graphs into multiple directed and weighted graphs of lesser dimensions - named views - and in building an interactive graph query language that supports user-specified views and full navigation across the data networks using these views as a support to the operations of the language. Within a view, structural abstraction techniques can then be used to enhance the visual perception further. The novelty of the approach taken is to combine *Visual graph representation* and *User interactions* [9] at the graph query language level with a view to supporting interactive dimension reduction based on the concept of blueprint where the architectural plan is distributed across different navigable views. In this context, users can select and combine labels according to their semantic understanding of the network models and visualize the corresponding network structures.

Section II gives a short overview on visualisation techniques for visual analytics (focusing on social networks) and on graph query languages fit to data networks. Section III gives a short description of the mathematical background supporting the approach. Section IV, introduces how views are constructed and Section V shows how the basic operations of the query language enable users to conduct their analysis. In Section VI, the use-case that inspired the Collaboration Spotting project and the graphical query language are presented. This paper ends with conclusions and future work in Section VII.

II. RELATED WORK

The related work is twofold since it combines multiple visual analytics techniques with the power of graph query languages. In the last 15 years, a lot of visual analytics articles were published with the aim of showing processes of transformation of multidimensional data into node-link diagrams [9][10].

A lot of articles have been published, especially on the *coordinated multiple views* topic, which introduces a visual analytics paradigm supported by an interactive query language or by a set of operations. These articles can be divided into four different groups:

- OLAP [11] inspired paradigms that are using operations like *slice, roll-up, dice*, etc. The most relevant papers are PivotGraph [12], ScatterDice [13], GraphDice [14], MatrixCube [15] and Orion [16].

 ISBN: 978-1-61208-637-8

- Relational algebra-related solutions such as Cross-filter views [17] which uses *grouping, filtering, projection* and *selection* operations, Polaris [18] that introduces and maps its algebra to SQL and Ploceus [19], which works with first-order logic language.
- Other solutions such as Cross-filter views with hypergraph query language [20], JUNG [21] and Gephi [22] that allow users to use other programming languages (JAVA in these cases).
- Literature on graph query languages is huge [23]–[30]. It covers the use of different graph models reflecting the variety of requirements for applications and languages.

The visual analytics model introduced in this paper promotes a different approach to graph query language. The language operates on a directed, labelled graph that is managed via user interactions treated as query inputs and follows the semantic web query language concepts, SPARQL [31] and Cypher [32][33]. This approach allows users to generate graph patterns and evaluate them directly on the graph. Reducing the complexity of network is not a novel idea [34]. The main differences between existing solutions and the one proposed in this paper are i) the introduction of a proper mathematical model based on labelled graphs; ii) a label-based complexity reduction (views); iii) basic operations to support navigation across views and iv) an intuitive user interface to drive these operations.

III. BASIC GRAPH AND VIEWS

Let graph G be a directed, labelled graph defined as a four-element tuple $G = (V, E, L, \alpha)$ where V represents a set of nodes and $E \subseteq V \times V$, a set of edges defined as a subset of the Cartesian products of these nodes. L is a set of node labels and $\alpha : V \rightarrow L$ is a mapping function from nodes to the corresponding labels. Figure 1 shows an example of such a graph. We define the reachability graph over graph G as

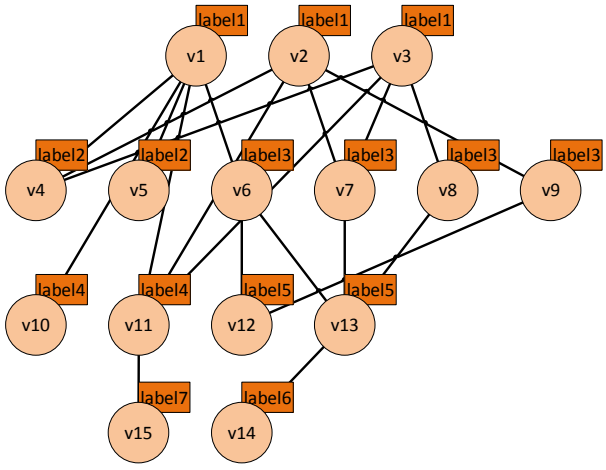


Figure 1. Example of graph G where $V = \{v1, \ldots v15\}$ and $L = \{label1, \ldots label7\}$

$G_{reachability} = (L, E_{reachability})$ where nodes are labels of graph G, $E_{reachability} \subseteq L \times L$ is defined as the Cartesian product of the labels where any two nodes of $G_{reachability}$ are connected if and only if there exists two connected nodes in graph G and their respective labels correspond to the two nodes of graph $G_{reachability}$. Graph $G_{reachability}$ is a description of graph G, it is also called the graph schema of graph G. Graph schema helps users view graph G via different sub-graphs of lesser dimensionality using labels of G as dimensions and facilitates the generation of approximately optimal user-defined graph queries. Let graph $G_{pattern} = (V_{pattern}, E_{pattern})$ be a graph pattern where $V_{pattern} \subseteq L$ and $E_{pattern} \subseteq E_{reachability} \cap V_{pattern} \times V_{pattern}$. To process the answer to a graph query, one needs to find all possible isomorphic subgraphs of G that are homomorphic to a graph pattern $G_{pattern}$ corresponding to the query. This is a graph pattern matching problem, a well-known part of Mathematics [35]. In this case, one defines Graph $G' = (V', E', L, \alpha)$, a subgraph of graph G as a sample matching the graph pattern $G_{pattern}$ if and only if:

- $\forall v' \in V' : \exists v \in V_{pattern}, \alpha(v') = v$,
- $\forall (u', v') \in E' : (\alpha(u'), \alpha(v')) \in E_{pattern}$.

The answer to a graph query is a view containing the set of subgraphs of G matching $G_{pattern}$. To build such a view, one needs first to introduce the graph pairing function $pair$ and the set $Pattern$. Let $G_{pattern_1}$ and $G_{pattern_2}$ be two graph patterns. These graph patterns are paired iff

- $V_{pattern_1} = V_{pattern_2}$ and
- $\exists! a, b \in V_{pattern_1}$:
  path(a,b) $\in E_{pattern_1}$ and path(a,b) $\notin E_{pattern_2}$,
  path(b,a) $\notin E_{pattern_1}$ and path(b,a) $\in E_{pattern_2}$,
  $E_{pattern_1} \setminus$ path(a,b) $= E_{pattern_2} \setminus$ path(b,a).

Where a path is an alternate non-empty sequence of nodes and edges, starting and ending with nodes and requiring that all edges and nodes be distinct from one another. path(a,b) $\in E_{pattern_1}$ indicates that all edges of this path are in set $E_{pattern_1}$. The $pair$ function is defined as

$$pair(G_{pattern}) := \begin{cases} G^{pair}_{pattern} & \text{if } G^{pair}_{pattern} \text{ pair of } G_{pattern} \\ (\emptyset, \emptyset) & \text{else.} \end{cases}$$

And $Pattern$, the set of these pairs is defined as $Pattern := \{(g, g') | g, g' \text{are patterns}, g' = pair(g)\}$.

A view of graph G is defined as a six-element tuple $G_q = (C_q, B_q, E_q, L_q, \epsilon_q, \upsilon_q,)$ where

- $C_q \subset V, L_C := \{\alpha(v) | v \in C_q\}$,
- $B_q \subset V, L_B := \{\alpha(b) | b \in B_q\}$,
- $L_q \subseteq L$ and $L_q = L_C \cup L_B$,
- $E_q := \{(u, v) |$
  $u, v \in C_q, \exists G', G'' \subseteq G,$
  $\exists (G_{pattern}, pair(G_{pattern})),$
  $(G'_{pattern}, pair(G'_{pattern})) \in$ *Patterns* :
  G' *matches to* $G_{pattern}$,
  G'' *matches to* $pair(G'_{pattern})$,
  $\exists b \in B_q$: path$(u, b) \in G'$, path$(b, v) \in G''\}$,
- $\epsilon_q : E_q \rightarrow \mathcal{P}(B_q), \epsilon_q((u, v)) = \{b |$
  $b \in B_q, \exists G', G'' \subseteq G,$
  $\exists (G_{pattern}, pair(G_{pattern})),$
  $(G'_{pattern}, pair(G'_{pattern})) \in$ *Patterns* :
  G' *matches to* $G_{pattern}$,
  G'' *matches to* $pair(G'_{pattern})$,
  path$(u, b) \in G'$, path$(b, v) \in G''\}$,

- $v_q : C_q \rightarrow \mathcal{P}(B_q), v_q(u) = \{b | b \in B_q,$ $\exists G' \subseteq G, \exists (G_{pattern}, pair(G_{pattern})) \in$ *Patterns* : G' *matches to* $G_{pattern}$, path$(u, b) \in G'\}$.

The use of multiple graph patterns for the construction of graph G_q is required since the cardinality of set L_B and set L_C are not necessary equal to 1 (see details in Section IV-A). To ease the reading, graph G_q is noted $G^{L_C}_{L_B}$ to refer directly to the set of labels used in the construction of the view. Also, in practice, we use an aggregation function on edges, respectively on nodes in graph G_q for determining their respective weights instead of the elements in set B_q (for instance, the number of elements). Figure 2 shows an example of a view when the two graph patterns are $G_{pattern} = (\{label1, label3\}, \{(label1, label3)\})$ and pair$(G'_{pattern}) = (\{label1, label3\}, \{(label3, label1)\})$.

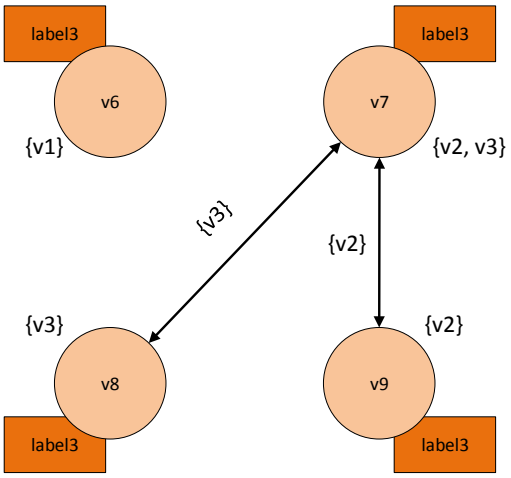


Figure 2. Example of a view where $C_q = \{v6, \ldots, v9\}$, $B_q = \{v1, \ldots, v3\}, L_C = \{label3\}, L_B = \{label1\}$.

IV. Graph creation from user interactions

In this section, we introduce how graph patterns and views can be created as a result of the following user interactions:

- Selection of different nodes in the current view,
- Removal of all nodes with the same label selected in one of the previous views,
- Navigation from one view to another.

Users can modify set L_C and set L_B when performing any of the above interactions. Let $F \subseteq V$ be the set of nodes corresponding to a user selection, we define from F:

1) $L_F := \{l \in L | \exists f \in F : \alpha(f) = l\}$ which contains the labels of nodes in set F and,
2) $F_{|L^*} := \{f \in F | \alpha(f) \in L^*\}$ with $L^* \subseteq L$, a subset of set F, restricted to nodes having their respective labels in set L^*.

In order for set F to operate as a filter, the matched sample definition of Section III has to be restricted by requiring that $\forall v' \in V', \alpha(v') \in L_F \Rightarrow v' \in F$. Example 1 below shows the content of L_F for user selection $F = \{v_4, v_6, v_7, v_{13}\}$ from the graph G depicted in Figure 1.

Example 1.

$$F = \{v_4, v_6, v_7, v_{13}\} \quad (1)$$
$$L_F = \{label_2, label_3, label_5\} \quad (2)$$

A. Graph pattern construction

This section shows how to construct a graph pattern with set L_F containing all the labels of nodes in set F. We exploit the fact that graph patterns are actually only needed when constructing edges in $G^{L_C}_{L_B}$ and their respective weights. A pair of graph patterns are required for each combination of labels in set L_C and set L_B since paths connecting nodes from set L_C and set L_B can have different directions, due to the construction of edges between nodes of C_q and nodes of B_q. Each pattern has to satisfy the following criteria:

- It must be a connected and directed graph,
- It must be minimal,
- Labels from set $L \setminus L_F$ can be used as intermediate nodes in the pattern.

These requirements exactly fit a Steiner Minimal Tree problem [36], known to be NP-complete[37] and for which we use a minimal spanning tree solver as an approximation algorithm. Algorithm 3 describes the full process of pair generation. Figure 4 shows the graph schema of graph G depicted in Figure 1 and the generated patterns pair for $\{v_4, v_6, v_7, V_{13}\}$ as set F, with $L_C = \{label4\}$ and $L_B = \{label1\}$.

Algorithm 3 Pattern generator algorithm

```
1:  function PatternGenerator(F_L, L_B, L_C)
2:      Patterns ← ∅
3:      B ← L_B
4:      while B ≠ ∅ do
5:          from, B ← from ∈ B, B \ {from}
6:          E ← L_C
7:          while E ≠ ∅ do
8:              to, E ← to ∈ E, E \ {to}
9:              Left ← SpanningTree(
                    F_L ∪ {from, to}, from, to)
10:             Right ← SpanningTree(
                    F_L ∪ {from, to}, to, from)
11:             Patterns ← Patterns ∪ {(Left, Right)}
12:         end while
13:     end while
14:     return Patterns
15: end function
```

B. Connecting user interactions and views

Now that graph patterns ($Patterns$) have been created using set F, set L_C and set L_B, one can introduce the gen function $gen : \mathcal{P}(V) \times \mathcal{P}(L) \times \mathcal{P}(L) \rightarrow G^{L_C}_{L_B}$ that generates views from user interactions, ($F \subseteq \mathcal{P}(V)$ and $L_C, L_B \subset \mathcal{P}(L)$) as $gen(F, L_C, L_B) := {}^F G^{L_C}_{L_B} = ({}^F C_q, {}^F B_q, E_q, L_q, v_q, \epsilon_q)$ where

$${}^F C_q := \begin{cases} V \cap F_{|L_C} & \text{if } V \cap F_{|L_C} \neq \emptyset \\ V_{|L_C} & \text{else} \end{cases}$$

are the nodes of graph ${}^F G^{L_C}_{L_B}$ and

$${}^F B_q := \left\{ b \in V'_{|L_B} \;\middle|\; \begin{array}{l} \exists G' = (V', E', L, \alpha) \subseteq G, : \\ G' \text{ matches to } G_{pattern}, \forall v' \in V' : \\ (v' \in F \text{ or } \alpha(v') \notin L_F) \end{array} \right\}$$

are the "interconnection" nodes: The other members of the six-tuple G_q are unchanged since

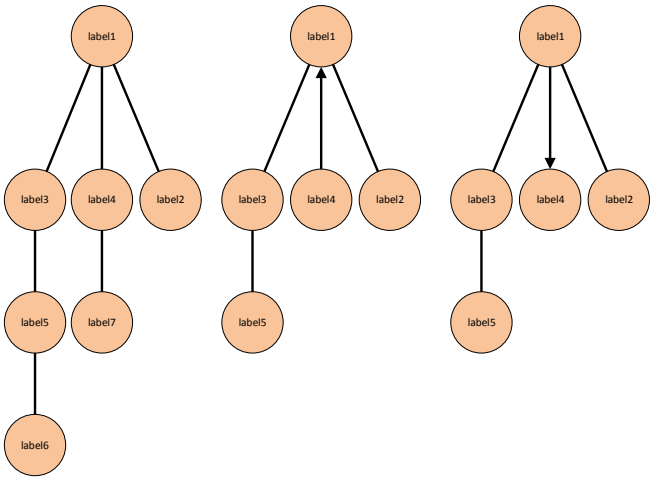


Figure 4. On the left hand-side, the graph schema of graph G; On the middle and on the right hand-side, an example of a graph pattern pair.

- labels (set L_q) are not modified and since
- edge definition (set E_q) and weighting functions (υ_q and ϵ_q) only depend on set ${}^{F}C_q$ and set ${}^{F}B_q$.

V. Operations on graphs

User interactions will result in the following graph operations:

- *Selection*: The user selects nodes in a view,
- *Expansion*: The user expands a view by removing in his previous selection, nodes having the same labels,
- *Navigation*: The user navigates from a view to another.

To define these operations one needs first to introduce the concepts of visual equivalence and minimal views since there can be views with nodes of null weight that are hidden to the user and hence non-selectable. Let F_1 and F_2 be two different filters on the same view complying with $F_1 \setminus F_{1|L_C} = F_2 \setminus F_{2|L_C}$. In essence, this means that there is no difference in the sets of nodes with labels contained in $L \setminus L_C$ which technically should be empty. View ${}^{F_1}G^{L_C}_{L_B}$ and view ${}^{F_2}G^{L_C}_{L_B}$ generated using F1 and F2 are said to be visual equivalent if and only if

Definition 1. *(Vis-equivalent)*

$$ {}^{F_1}G^{L_C}_{L_B} \dot{\sim} {}^{F_2}G^{L_C}_{L_B} \Leftrightarrow \begin{array}{l} \forall v \in V_1 \setminus V_2 : \upsilon_q(v) = \emptyset, \\ \forall v' \in V_2 \setminus V_1 : \upsilon_q(v') = \emptyset, \end{array} $$

where V_1 (V_2) represents the nodes of view ${}^{F_1}G^{L_C}_{L_B}$ (${}^{F_2}G^{L_C}_{L_B}$). Intuitively visual equivalence guaranties that nodes that are not common to two views have empty weights. It provides equivalence classification on views. It is easy to prove that for each class of views there is only one which does not have nodes with empty weights. This view is called the minimal view.

A. Selection on graphs

Let F_{select} be the set of user selected nodes within a view. $F_{select} \subseteq V$ and $F_{select} \subseteq V'$ where V' is a set of nodes from the minimal view which is visual-equivalent to graph ${}^{F}G^{L_C}_{L_B}$. The selection operator $\sigma : G_q \times \mathcal{P}(V) \rightarrow G_q$ is defined as

Definition 2. *(Selection)*

$$\sigma({}^{F}G^{L_C}_{L_B}, F_{select}) := gen((F \setminus F_{|L_C}) \cup F_{select}, L_C, L_B),$$

where $F_{|L_C} = \{f | f \in F, \alpha(f) \in L_C\}$. It is to be noted that at view creation the selection operator uses a more general definition of the gen function. Figure 6a and 6b show how the selection operator works. As a result of applying this operator, set L_C and L_B will only contain those nodes that are "related" to this user selection.

B. Expansion on graphs

The expansion operator ξ is in some sense the "inverse" of the selection operator. It is defined as

Definition 3. *(Expansion)*

$$\xi({}^{F}G^{L_C}_{L_B}, L_{C'}) := gen(F \setminus F_{|L_{C'}}, L_{C'}, L_B).$$

The expansion operator changes view when $L_{C'} \neq L_C$ and removes all nodes in set F that are labelled with labels in L_C. Figure 6d and 6e show how the expansion operator works.

C. Navigation through graphs

By selecting a subset of labels from L_C one can build views of graph G with reduced dimensional complexity. Navigation across views is required to enable users to apprehend the full graph G. Therefore the navigation function η goes from view ${}^{F}G^{L_C}_{L_B}$ to a view labelled as $L_{C'}$ and $L_{B'}$ and is defined as:

Definition 4. *(Navigation)*

$$\eta({}^{F}G^{L_C}_{L_B}, L_{C'}, L_{B'}) := gen(F, L_{C'}, L_{B'})$$

Figure 6b and 6c show how the navigation between views works.

D. Navigation history

The navigation history can be represented as a navigation graph G_{nav} where nodes represent navigation states and edges navigation steps between states. $G_{nav} = (N_{nav}, E_{nav})$ complies to

- $N_{nav} \subset \mathcal{P}(V) \times \mathcal{P}(L) \times \mathcal{P}(L)$.
- $E_{nav} \subseteq N_{nav} \times N_{nav} \times \{\sigma, \xi, \eta\}$,

where there is a navigation step between node $n_1 = (F_1, L_{C_1}, L_{B_1})$ to node $n_2 = (F_2, L_{C_2}, L_{B_2})$ if and only if one of the following statements is true:

1) $\sigma(gen(F_1, L_{C_1}, L_{B_1}), F_2 \setminus F_1) = gen(F_2, L_{C_2}, L_{B_2})$, and $L_{C_1} = L_{C_2}, L_{B_1} = L_{B_2}$;
2) $\xi(gen(F_1, L_{C_1}, L_{B_1}), L_{C_2}) = gen(F_2, L_{C_2}, L_{B_2})$, and $F_2 = F_1 \setminus F_{1|L_{C_2}}, L_{B_1} = L_{B_2}$;
3) $\eta(gen(F_1, L_{C_1}, L_{B_1}), L_{C_2}, L_{B_2}) = gen(F_2, L_{C_2}, L_{B_2})$ and $F_1 = F_2$.

In E_{nav}, the third component of an edge is always one of the operations σ, ξ or η. It indicates how the step was processed. The proper size of N_{nav} is $2^n * (3^m - 2^{m+1} + 1)$ where $n = |V|$ and $m = |L|$. A particular navigation history corresponds to a walk in G_{nav}. An example of such a walk is given below.

Example 2 (Walk on graph)**.**

$$(F_0, L_{C_0}, L_{B_0}), \eta, (F_1, L_{C_1}, L_{B_1}), \sigma, \ldots, \xi, (F_f, L_{C_v}, L_{B_b})$$

 ISBN: 978-1-61208-637-8

In practice, a particular set of labels L_{C_0} is used to create an entry view from which all the above mentioned operations can then be performed.

VI. USE-CASE

In the framework of AIDA [38], an FP7 project on Advanced European Infrastructures for Detectors at Accelerators, researchers needed to identify key players from academia and industry for technologies considered as strategic for the particle physics programme. To this end, the Collaboration Spotting project was launched in 2012 with a view to enabling users to search for terms describing particular technologies in titles and abstracts of publications and patents and viewing the organisation, subject category, keywords, city and country landscapes for each of these searches individually. Individual technology searches are represented as nodes in a view named Technogram, used as the user entry view in which edges represent publications and/or patents common to searches.

A. Data

Two different sources are used for searching. The metadata records of publications from Web of Science™ Core Collection [39] developed by Clarivate Analytics (in the past, Thomson Reuters) and the metadata records of patents from PATSTAT developed by the European Patent Office [40]. Although the two sources have a number of labels in common, such as *Organisation*, *City* and *Country* there are others like *Subject Category* and *Keyword* that only belong to publications. The subset of data from the two sources corresponding to the labels of interest for users was used to construct graph G and its schema $G_{reachability}$.

B. Storing data in a graph database (Neo4j)

Graph G is stored in a Neo4j graph database [41], in which individual metadata records are stored as subgraphs of labelled nodes using *Published item, Organisation, Subject Category, Author Keyword, City, Region* and *Country* as labels. Figure 5 represents the reachability graph (graph schema) of this network (Light color nodes represent nodes uploaded by the data administrator and the dark nodes are created by the system itself by using search and authentication modules). Besides these labels, additional labels have been introduced to support user authentication and authorisation (*User*) and technology searches (*Graph* and *Technology*). Searches use full text indices of the Apache Lucene project [42] that have been integrated into the Neo4j database as legacy indices [41].

Statistic of the graph data: Searches on publications and patents metadata records from the 2000 - 2014 period can be performed. The resulting data network contains 45 million nodes and 150 million edges. Its breakdown is given in Table I. and Table II.

As can be noticed, the number of region edges is smaller than the number of country edges due to the use of the 2nd level of Nomenclature of Territorial Units For Statistic [43] created by the European Commission, which covers Europe only.

C. Navigation

The entry point for this use case is individual users. Using the terminology introduced above, the initial values for set F are user IDs.

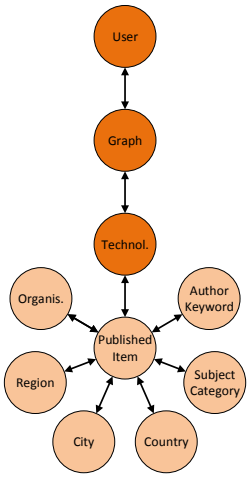


Figure 5. The database schema (reachability graph)

TABLE I. NUMBER OF NODES BY NODE LABELS

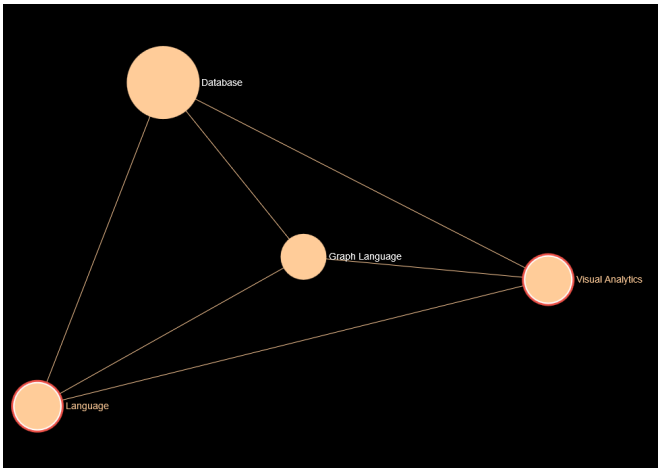
TYPE OF NODES	NUMBER OF NODES
Patents	15.000.442
Publications	20.087.904
Organisations	2.918.060
Author Keywords	8.193.604
Subject Categories	230
Cities	7.741
Regions	946
Countries	128
Total	46.209.055

TABLE II. NUMBER OF EDGES BY NODE LABELS. A PATENT DOES NOT HAVE AUTHOR KEYWORDS OR SUBJECT CATEGORIES PROPERTY

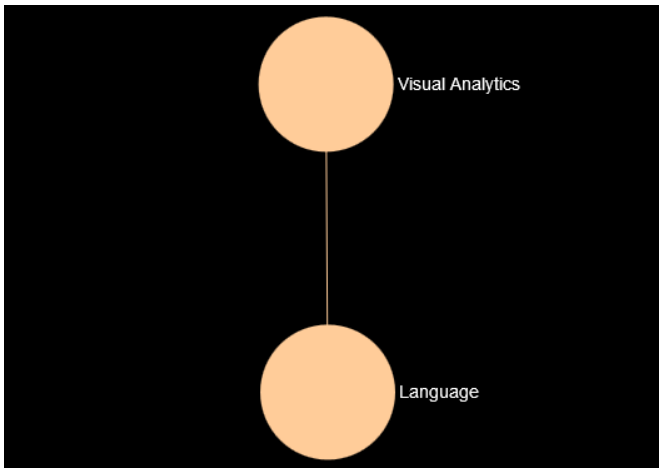
	PATENTS	PUBLICATIONS	TOTAL
Organ.	12.440.903	36.672.677	49.113.580
Author Key.	-	48.941.098	48.941.098
Subject Cat.	-	32.566.806	32.566.806
Cities	3.193.709	8.826.222	12.019.931
Regions	265.421	2.504.441	2.769.862
Count.	3.156.449	8.020.648	11.177.097
Total	19.056.482	137.531.892	156.588.374

Limitations: In the current implementation there is a restriction on the size of L_C and L_B fixed to a single label *Published Item* and the visualization system only supports undirected edges. This calls for the generation of only one graph pattern instead of two making the system faster.

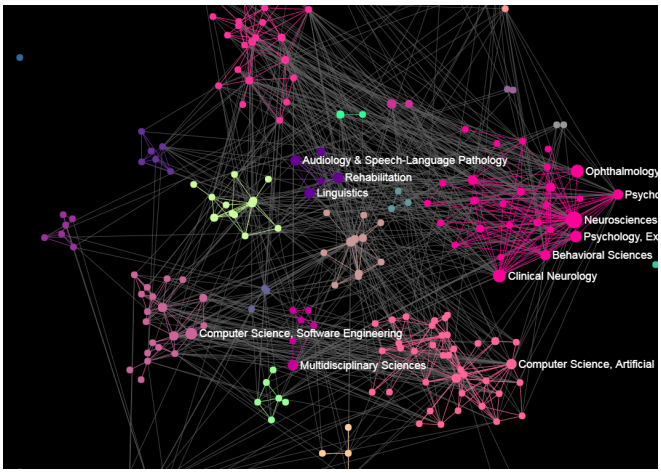
In Figure 6, a series of pictures illustrate how navigation operations work. A user enters the system in a technology view (nodes are labelled with the *Technology* label). In the example, this view contains four "technologies" (obtained as results of searches using Lucene-indices), namely *Database, Language, Graph Language* and *Visual Analytics*. Links between nodes indicate publications and patents common to technology searches. As indicated in the reachability graph of Figure 5, users can access other views via nodes labelled with the *Published Item* label. The user selects two *technology* nodes from Figure 6a, giving F_{select} ={*Visual Analytics*, *Language*}. Figure 6b shows the result of this selection: *Language* and



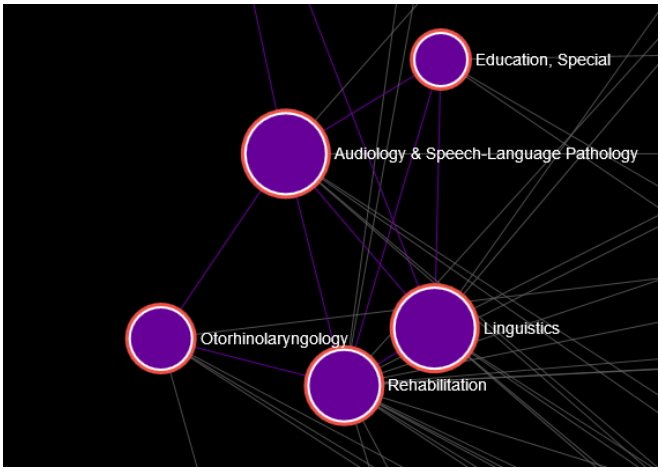
(a) Technology view: ($L_C = \{Technology\}, L_B = \{Published\ Item\}$); Selecting two technologies ($F_{select} = \{Visual\ Analytics, Language\}$)



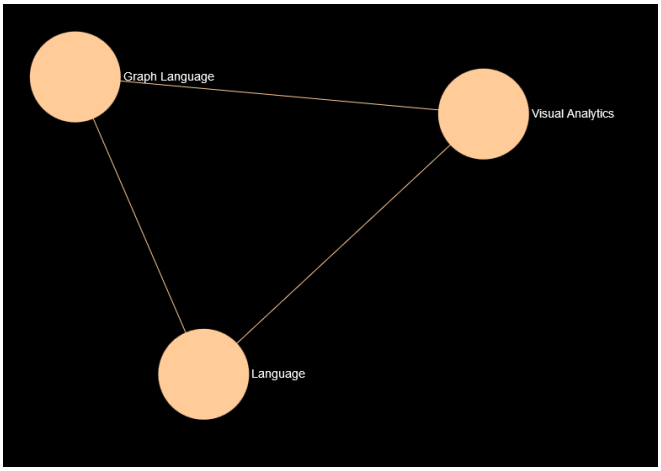
(b) Result of the selection



(c) Navigation to the "Subject Category view"



(d) Selecting a cluster in the Subject Category view; expanding the view and going back to the Technology view



(e) Technology view with $F = \{Language,\ Lingustics \ldots Rehabilitation\}$ filter

Figure 6. Example of operations; navigation, selection and expansion on views

Visual Analytics. Changing set L_C from value $\{Technology\}$ to value $\{Subject\ Category\}$ enables the navigation operation to reach the *Subject Category* view corresponding to the two previously selected *technology* nodes. Figure 6c shows the

view resulting from this operation. It is optained using the $(L_C = \{\textit{Subject Category}\}, L_B = \{\textit{Published Item}\}, F = \{\textit{Language, Visual Analytics}\})$ triplet. In Figure 6d, the user selects a few nodes from the view of Figure 6c (i.e., the F filter was extended with *Linguistic, ...,* values). After selection of the value $\{\textit{Technology}\}$ for L_C, the expansion and navigation operations bring the user back to the *technology* view of Figure 6e. This view shows the *technology* nodes having publications with nodes labelled with the *Subject Category* label corresponding to the last user selection.

VII. Conclusion and Future Work

The current version of Collaboration Spotting running at CERN [44] addresses the implementation of the concepts using patents and publications metadata records. It is a new experimental service that aims to provide the High Energy Physics community (such as HEPTech [45]) with information on Academia & Industry main players active around key technologies, with a view to fostering more inter-disciplinary and inter-sectoral R&D collaborations, and giving the procurement service the opportunity of reaching a wider selection of high-tech companies for bidding purposes. Collaboration Spotting is generic in its concepts and implementation. It can support visual analytics of any kind of data and its backend is implemented using a Neo4j graph database [41]. Conference papers, technical & business news, trademarks & designs and financial data are amongst the data targeted to enrich the information on technologies that one can obtain from publications and patents. The choice of data sources will depend on users' priorities. The tool can be of use to other communities, in particular in dentistry [46] but also to policy makers and investors if data in the knowledge graph is enriched with technical & business news and financial data. Collaboration Spotting also addresses other types of data, such as compatibility and dependency relationships in software and meta-data [47][48] of the LHCb experiment at CERN.

As an interactive graph query language, Collaboration Spotting is intended to provide a fully customisable visual analytics environment. In the current version, data processing supports searches and contextual queries. In the future, labelled & directed relationships and attributes on nodes will be included in the labelled property graph representation of the data network and the processing will be extended to more complex operations directly on the graph resulting from searches and queries with a view to enhancing the visual perception of users.

Acknowledgment

We are very thankful to Richard Forster, Sotiris Fragkiskos, Tim Hertweck, Nikos Kasfikis, Xavier Ouvrard and Eirik Skogstad from CERN for their hard work in implementing the first prototypes of Collaboration Spotting. We are also thankful to Bernard Denis from CERN and Emanuele Leonardi from INFN who provided very valuable feedback when testing the early version of the tool; Thierry Lagrange from CERN for ensuing sufficient funding to support the development.

References

[1] P. C. Wong and J. Thomas, "Visual analytics," IEEE Comput. Graph. Appl., vol. 24, no. 5, pp. 20–21, Sep. 2004, ISSN 0272-1716, doi: 10.1109/MCG.2004.39.

[2] J. B. Kollat, P. M. Reed, and R. M. Maxwell, "Many-objective groundwater monitoring network design using bias-aware ensemble Kalman filtering, evolutionary optimization, and visual analytics," Water Resour. Res., vol. 47, no. 2, pp. 1:1–1:18, 2011, ISSN 1944-7973, doi: 10.1029/2010WR009194,, w02529.

[3] A. Scharl, A. Hubmann-Haidvogel, A. Weichselbraun, H. P. Lang, and M. Sabou, "Media watch on climate change – visual analytics for aggregating and managing environmental knowledge from online sources," in 2013 46th Hawaii Int. Conf. System Sciences, Jan 2013, pp. 955–964, ISSN 1530-1605, doi: 10.1109/HICSS.2013.398.

[4] P. T. Wood, "Query languages for graph databases," SIGMOD Rec., vol. 41, no. 1, pp. 50–60, Apr. 2012, ISSN 0163-5808, doi: 10.1145/2206869.2206879.

[5] P. C. Wong, H. W. Shen, C. R. Johnson, C. Chen, and R. B. Ross, "The top 10 challenges in extreme-scale visual analytics," IEEE Comput. Graph. Appl., vol. 32, no. 4, pp. 63–67, July 2012, ISSN 0272-1716, doi: 10.1109/MCG.2012.87.

[6] E. Namey, G. Guest, L. Thairu, and L. Johnson, "Data reduction techniques for large qualitative data sets," in Handbook for team-based qualitative research, vol. 2, pp. 137–161, ISBN 978-0-7591-1373-2.

[7] T. A. Davis and Y. Hu, "The university of Florida sparse matrix collection," ACM Trans. Math. Softw., vol. 38, no. 1, pp. 1:1–1:25, Dec. 2011, ISSN 0098-3500, doi: 10.1145/2049662.2049663.

[8] Z. Shen, K.-L. Ma, and T. Eliassi-Rad, "Visual analysis of large heterogeneous social networks by semantic and structural abstraction," IEEE Trans. Vis. Comput. Graphics, vol. 12, no. 6, pp. 1427–1439, Nov 2006, ISSN 1077-2626, doi: 10.1109/TVCG.2006.107.

[9] T. von Landesberger et al., "Visual analysis of large graphs: state-of-the-art and future research challenges," Comput. Graph. Forum, vol. 30, no. 6, pp. 1719–1749, 2011, ISSN 1467-8659, doi: 10.1111/j.1467-8659.2011.01898.x.

[10] J. Kehrer and H. Hauser, "Visualization and visual analysis of multi-faceted scientific data: A survey," IEEE Trans. Vis. Comput. Graphics, vol. 19, no. 3, pp. 495–513, March 2013, ISSN 1077-2626, doi: 10.1109/TVCG.2012.110.

[11] J. Gray et al., "Data Cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," Data Min. Knowl. Discov., vol. 1, no. 1, pp. 29–53, Mar 1997, ISSN 1573-756X, doi: 10.1023/A:1009726021843.

[12] M. Wattenberg, "Visual exploration of multivariate graphs," in Proc. SIGCHI Conf. Human Factors in Computing Systems, ser. CHI '06. New York, NY, USA: ACM, 2006, pp. 811–819, ISBN 1-59593-372-7, doi: 10.1145/1124772.1124891.

[13] N. Elmqvist, P. Dragicevic, and J. D. Fekete, "Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation," IEEE Trans. Vis. Comput. Graphics, vol. 14, no. 6, pp. 1539–1148, Nov 2008, ISSN 1077-2626, doi: 10.1109/TVCG.2008.153.

[14] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, and J. D. Fekete, "Graphdice: A system for exploring multivariate social networks," in Proc. 12th Eurographics / IEEE - VGTC Conf. Vis., ser. EuroVis'10. Chichester, UK: The Eurographs Association and John Wiley & Sons, Ltd., 2010, pp. 863–872, doi: 10.1111/j.1467-8659.2009.01687.x.

[15] B. Bach, E. Pietriga, and J.-D. Fekete, "Visualizing dynamic networks with matrix cubes," in Proc. SIGCHI Conf. Human Factors in Computing Systems, ser. CHI '14. New York, NY, USA: ACM, 2014, pp. 877–886, ISBN 978-1-4503-2473-1, doi: 10.1145/2556288.2557010.

[16] J. Heer and A. Perer, "Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks," Inf. Vis., vol. 13, no. 2, pp. 111–133, 2014, doi: 10.1177/1473871612462152.

[17] C. Weaver, "Cross-filtered views for multidimensional visual analysis," IEEE Trans. Vis. Comput. Graphics, vol. 16, no. 2, pp. 192–204, March 2010, ISSN 1077-2626, doi: 10.1109/TVCG.2009.94.

[18] C. Stolte, D. Tang, and P. Hanrahan, "Polaris: a system for query, analysis, and visualization of multidimensional relational databases," IEEE Trans. Vis. Comput. Graphics, vol. 8, no. 1, pp. 52–65, Jan 2002, ISSN 1077-2626, doi: 10.1109/2945.981851.

[19] Z. Liu, S. B. Navathe, and J. T. Stasko, "Network-based visual analysis of tabular data," in 2011 IEEE Conf. Visual Ana-

lytics Science and Technology (VAST), Oct 2011, pp. 41–50, doi: 10.1109/VAST.2011.6102440.

[20] R. Shadoan and C. Weaver, "Visual analysis of higher-order conjunctive relationships in multidimensional data using a hypergraph query system," IEEE Trans. Vis. Comput. Graphics, vol. 19, no. 12, pp. 2070–2079, Dec 2013, ISSN 1077-2626, doi: 10.1109/TVCG.2013.220.

[21] J. O'Madadhain, D. Fisher, S. White, and Y. Boey, "The JUNG (Java universal network/graph) framework," University of California, Irvine, California, 2003. [Online]. Available: http://jung.sourceforge.net/index.html 2018.04.03

[22] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: an open source software for exploring and manipulating networks," 3rd Int. AAAI Conf. Weblogs and Social Media (ICWSM), vol. 8, pp. 361–362, 2009.

[23] J. Hidders and J. Paredaens, "Goal, a graph-based object and association language," in Adv. Database Systems: Implementations and Applications, J. Paredaens and L. Tenenbaum, Eds. Vienna: Springer Vienna, 1994, pp. 247–265, ISBN 978-3-7091-2704-9, doi: 10.1007/978-3-7091-2704-9_13.

[24] J. Paredaens, D. V. Gucht, J. V. den Bussche, and M. Gyssens, "A graph-oriented object database model," IEEE Trans. Knowl. Data Eng., vol. 6, pp. 572–586, 08 1994, ISSN 1041-4347, doi: 10.1109/69.298174.

[25] J. Hidders, "Typing graph-manipulation operations," in Database Theory — ICDT 2003, D. Calvanese, M. Lenzerini, and R. Motwani, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 394–409, ISBN 978-3-540-36285-2, doi: 10.1007/3-540-36285-1_26.

[26] R. H. Güting, "GraphDB: modeling and querying graphs in databases," in Proc. 20th Int. Conf. Very Large Data Bases, ser. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 297–308, ISBN 1-55860-153-8.

[27] H. S. Kunii, "DBMS with graph data model for knowledge handling," in Proc. 1987 Fall Joint Computer Conf. Exploring Technology: Today and Tomorrow, ser. ACM '87. Los Alamitos, CA, USA: IEEE Computer Society Press, 1987, pp. 138–142, ISBN 0-8186-0811-0.

[28] P. Barceló Baeza, "Querying graph databases," in Proc. 32nd Symp. Principles of Database Systems, ser. PODS '13. New York, NY, USA: ACM, 2013, pp. 175–188, ISBN 978-1-4503-2066-5, doi: 10.1145/2463664.2465216.

[29] J. Paredaens, P. Peelman, and L. Tanca, "G-Log: a graph-based query language," IEEE Trans. Knowl. Data Eng., vol. 7, no. 3, pp. 436–453, Jun 1995, ISSN 1041-4347, doi: 10.1109/69.390249.

[30] J. Yang, S. Zhang, and W. Jin, "DELTA: indexing and querying multi-labeled graphs," in Proc. 20th ACM Int. Conf. Information and Knowledge Management, ser. CIKM '11. New York, NY, USA: ACM, 2011, pp. 1765–1774, ISBN 978-1-4503-0717-8, doi: 10.1145/2063576.2063832.

[31] S. Harris, A. Seaborne, and E. Prudhommeaux. (2013) Sparql 1.1 query language. [Online]. Available: https://www.w3.org/TR/sparql11-query/ 2018.04.03

[32] J. Webber, "A programmatic introduction to Neo4J," in Proc. of the 3rd Annual Conference on Systems, Programming, and Applications: Software for Humanity, ser. SPLASH '12. New York, NY, USA: ACM, 2012, pp. 217–218, ISBN 978-1-4503-1563-0, doi: 10.1145/2384716.2384777.

[33] Neo Technology. openCypher project. [Online]. Available: http://www.opencypher.org 2018.04.03

[34] R. M. Martins et al., "Multidimensional projections for visual analysis of social networks," Journal of Computer Science and Technology, vol. 27, no. 4, pp. 791–810, Jul 2012, ISSN 1860-4749, doi: 10.1007/s11390-012-1265-5.

[35] B. Gallagher, "Matching structure and semantics: A survey on graph-based pattern matching," AAAI Fall Symp. Capturing and Using Patterns for Evidence Detection, vol. 6, pp. 45–53, 2006.

[36] F. K. Hwang, D. S. Richards, and P. Winter, "Introduction," in The Steiner Tree Problem, ser. Annals of Discrete Mathematics. Elsevier, 1992, vol. 53, pp. 3–19, ISSN 0167-5060, doi: 10.1016/S0167-5060(08)70192-4.

[37] M. R. Garey, R. L. Graham, and D. S. Johnson, "The complexity of computing Steiner minimal trees," SIAM J. Appl. Math., vol. 32, no. 4, pp. 835–859, 1977, ISSN 0036-1399, doi: 10.1137/0132072.

[38] AIDA Collaboration. AIDA - website. [Online]. Available: http://aida2020.web.cern.ch/content/aida 2018.04.03

[39] Clarivate Analytics. Web of Science™. [Online]. Available: http://webofknowledge.com 2018.04.03

[40] European Patent Office. PATSTAT - worldwide patent statistical database. [Online]. Available: http://www.epo.org/searching-for-patents/business/patstat.html 2018.04.03

[41] Neo4j, The Neo4j manual, November 2015, Release 2.3.2. [Online]. Available: http://neo4j.com/docs/stable/index.html 2018.04.03

[42] Lucene™/Solr™ Committers, Apache Lucene™ Documentation. [Online]. Available: https://lucene.apache.org/core/documentation.html 2018.04.03

[43] European Commision. NUTS - Nomenclature of territorial units for statistics. [Online]. Available: http://ec.europa.eu/eurostat/web/nuts/overview 2018.04.03

[44] Collspotting Developerment Team. Collspotting. [Online]. Available: http://collspotting.web.cern.ch 2018.04.03

[45] HEPTech Collaboration. HEPTech - website. [Online]. Available: http://heptech.web.cern.ch 2018.04.03

[46] E. Leonardi, A. Agocs, S. Fragkiskos, N. Kasfikis, J. Le Goff, M. Cristalli, V. Luzzi, and A. Polimeni, "Collaboration spotting for dental science," Minerva Stomatologica, vol. 63, no. 9, pp. 295–306, Sep 2014.

[47] M. Cattaneo, M. Clemencic, and I. Shapoval, "LHCb software and conditions database cross-compatibility tracking system: A graph-theory approach," in 2012 IEEE Nuclear Science Symp. and Medical Imaging Conf. Record (NSS/MIC), Oct 2012, pp. 990–996, ISSN 1082-3654, doi: 10.1109/NSSMIC.2012.6551255.

[48] I. Shapoval, M. Clemencic, and M. Cattaneo, "ARIADNE: a tracking system for relationships in LHCb metadata," J. Phys. Conf. Ser., vol. 513, no. 4, pp. 1:1–1:7, 2014, 042039.