5
Case Study Evaluations

This chapter presents three case studies devised to show the value of the framework of exploration operations for the description of exploration strategies in real and well-documented problematic situations. The main goals of the case studies are: 1 – demonstrate the expressive power of the framework to describe complex task solutions in terms of sequence applications of the exploration operations; 2 – demonstrate the usage of the framework as an epistemic tool for devising alternative sequences of steps; 3 – demonstrate possible reuse and adaptation scenarios of explorations.

We selected case studies in different domains, such as, the biological domain, where we describe an exploration case over a cluster of genes; the patent exploration field, where technological trends are analyzed; the scientific publications field, where a publication review task is demonstrated. The criteria to select the cases were the following:

1. *The case should be published as a difficult case in the area.* Since the case is published, we infer that the case is a real problematic situation faced by a community of data users, with reasonable complexity.
2. *The case is difficult to be solved with operators in the state-of-the-art tools*. The rationale for this criterion is that we can use the case studies to compare the expressivity of our model against state-of-the-art tools using the same tasks.

5.1.Case Study 1: Discovering Technological Trends

Patent datasets can be used as a source of information about changes in technological trends either in knowledge fields or in a company R&D strategy. Such information is valuable for the development of competitive intelligence of a company (MUKHERJEA; BAMBA; KANKAR, 2005; SHIH; LIU; HSU, 2010). The following task, raised in (SHIH; LIU; HSU, 2010), has as its main goal to generate a report on technological trends for either a specific company or a patent classification domain. In order to demonstrate the expressivity of our framework

in a complex task, we selected the task discussed in (SHIH; LIU; HSU, 2010), which presents a system that allows patent analysts to trace changes in the activities in technology fields, by analyzing patenting activities on these fields in two different time periods. We published this case study in (NUNES; SCHWABE, 2015). Figure 17 shows a summarized schema of the patent dataset containing only the entities and relations used in the exploration task.

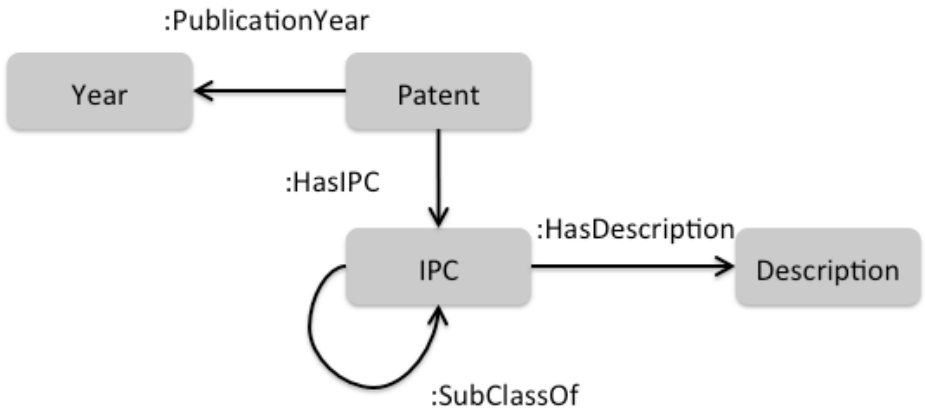

Figure 17 - Patents dataset summarized schema

The changes in the technological landscape that can be identified by analyzing published patents in different time periods are observed by answering four main questions:

- Which industry fields have increased the level of attention throughout given periods?
- Which industry fields have decreased the level of attention throughout given periods?
- Which industry fields started to be addressed throughout given periods?
- Which industry fields stopped to be addressed throughout given periods?

5.1.1.Task Execution

The industry fields are mapped to the patent classifications in the International Patent Classification (IPC) system[9], which organizes a set of patent categories hierarchically. The level of attention of each IPC classification is measured by indicators that consider the age of the patents, the number of

[9] http://web2.wipo.int/ipcpub/#refresh=page

citations, the originality and generality of the patents, and the average age of the cited patents. For more details about the indicators, refer to (SHIH; LIU; HSU, 2010). For illustration purposes, let $lv: P^n \rightarrow \mathbb{Q}$ be a function that maps a set of patent documents *P* into a numeric value in $\mathbb{Q}$ that represents the level of attention that the set is receiving.

The first step is to find the set of IPC classes related to some knowledge area:

1. $S_1 \leftarrow$ P.*Pivot*(:HasIPC)
2. $S_2 \leftarrow S_1$.*Refine*(*matchOne*("semiconductor", "silicon", "led", "insulator", "transistor")

The actions above are an attempt to find all classes related to the field of semiconductors by pivoting from the set of patents *P* to the set of IPCs through the :*HasIpc* relation (step 1) and refining the set of IPCs to those that matche one of the keywords related to the field of interest (step 2). Next, the explorer splits the set of patents into two sets published in different periods by, first, filtering out patents whose IPCs are not in the set of IPCs related with the field of interest using an intersection between the sets (step 3), and then, filtering patents published in the periods of interest (steps 4 and 5):

3. $S_3 \leftarrow$ P.*Refine*(*equalsOne*(:HasIPC, S_2))
4. $S_4 \leftarrow S_3$.*Refine*($2001 \leq$:PublicationYear ≤ 2002)
5. $S_5 \leftarrow S_3$.*Refine*($2003 \leq$:PublicationYear ≤ 2004)

The goal of the next steps is to reorganize the data to answer the questions based on the levels of attention of each IPC:

6. $S_6 \leftarrow S_4$.*Group*(:HasIPC)
7. $S_7 \leftarrow S_5$.*Group*(:HasIPC)
8. $S_8 \leftarrow S_6$.*Map*(2, *lv*)
9. $S_9 \leftarrow S_7$.*Map*(2, *lv*)
10. $S_{10} \leftarrow S_4$.*Diff*(S_5)
11. $S_{11} \leftarrow S_5$.*Diff* (S_4)
12. $S_{12} \leftarrow S_8$.*Refine*(%attentionLevel < S_9[*p*(%attentionLevel)])
13. $S_{13} \leftarrow S_8$.*Refine*(%attentionLevel > S_9[*p*(%attentionLevel)])
14. $S_{14} \leftarrow S_8$.*Refine*(%attentionLevel = S_9[*p*(%attentionLevel)])

Steps 6 and 7 group the sets of patents published within the two different periods by their IPCs. Having the groups of patents per IPC, the explorer applies the function *lv* to extract the level of attention for each IPC (steps 8 and 9). Next, the explorer splits the set of all IPC classifications into classifications that started to gain attention along the periods (step 10), classifications that are no longer addressed from one period to the next (step 11), classifications that have increased the level of attention along the periods (step 12), measured by the function *lv* in steps 8 and 9, classifications that have decreased the level of attention along the periods (step 13), and classifications that remained with the same level of attention.

Since an exploration set is also a relation, we can obtain restricted images using the same notation, such as $S_9[p(\%attentionLevel)]$, in steps 12 and 13, which extracts the level of attention in the set S_9 for the parent IPC of the attention level being refined, given by $p(\%attentionLevel)$. As an example, suppose the set $S_8 = \{<ipc_1, l_1>,...,<ipc_k, l_k>,...,<ipc_n, l_n>\}$ and the set $S_9 = \{<ipc_1, j_1>,...,<ipc_k, j_k>,...,<ipc_n, j_n>\}$, where, the first elements are IPCs and the second elements, $\{l_1...l_n\}$ from S_8 and $\{j_1...j_n\}$ from S_9, are numerical values representing the levels of attention for each IPC. Let the parameter *%attentionLevel* be j_k. Therefore, the access of the level of attention in S_9 for the IPC related to j_k in S_8 is expressed as follows:

$$p(j_k) = ipc_k \text{ and } S_9[p(j_k)] = S_9[ipc_k] = \{l_k\}$$

5.1.2.Alternative Strategies

The sequence of operators for achieving the solution described in the previous subsection is not the only possible solution. One advantage of using the framework is the possibility of devising many possible sequences and analyzing which ones are more appropriate, giving the users profile and the task context, abstracting interface and interaction details.

As examples of such discussions and adaptations, lets take the steps 1 and 2 of the original solution. In these steps, the user first pivots to a set of IPCs and then applies a keyword refine. These sequences demonstrate a user that may not know very well the IPC taxonomy since keyword refines is non-structured. If only this sequence of operations is available through the interface, more specialized

users may be hindered. A more advanced user may know that all semiconductor-related IPCs is a subclass of the class H01 – BASIC ELECTRIC ELEMENTS. Therefore, s/he may want to use the taxonomy hierarchy to find more precisely the IPCs related to the semiconductors field. The step 2 could be replaced by the following operation:

2. S_2' ← S_1.*Refine*(*equals*(:SubclassOf[%ipc], :H01)

On the other hand, a naïve user may not know anything about the schema. For this type of user, the relations :SubclassOf, :HasIPC, and :PublicationYear may be discovered through previous steps of schema exploration. As an example, the sequence of steps for finding out about IPCs by a naïve user could be the following:

2.1. R ← P.*Pivot*(*:RelationsOf*)

2.2. TypeRel ← R.*Refine*(*equals*(%r, :HasType))

2.3. AllTypes ← TypeRel.*Pivot*(:Image)

2.4. IPCType ← AllTypes.*Refine*(*equals*(:IPC))

2.5. IPCs ← P.*Refine*(*equals*(:HasType, $IPCType_{[1]}$))

2.6. S_2 ← IPCs.*Refine*(*matchOne*("semiconductor", "silicon", "led", "insulator", "transistor")

In step 2.1 the user asks for the relations in the dataset. Next, he selects the relation :HasType, which s/he supposes is a typing relation, and selects the IPC class from the domain set of :HasType. In step 2.3, since the unitary set *TypeRel* contains the identifier of a relation (:HasType), we can use the *:Image* relation to pivot from the *:HasType* relation to its image set, which is the set of all item types in the dataset. In step 2.4 the user selects the item identified as :IPC, which is the type for all patent classifications in the dataset. In step 2.5 the explorer filters out every item *i* of the dataset where :HasType[*i*] ≠ :IPC, i.e., only IPCs are kept in the result set. The expression of the step 2.5 is equivalent to the following expression:

2.5. IPCs ← P.*Refine*(*equals*(:HasType, :IPC)), since $IPCType_{[1]}$ = {:IPC}

Finally, the user issues a keyword refine as an attempt to find semiconductor-related IPCs. From these alternative steps we can observe an

exploration aiming at learning the schema, which is usually the case in exploration tasks.

Although these alternative sequences of operations for schema learning can also be described using our framework, we omit them in the next case studies for conciseness purposes. Henceforth, we assume that the user has already done these steps or s/he already has sufficient knowledge concerning the data schema before starting the task itself.

5.1.3. Generalizing and Reusing Exploration Patterns

The sequence of steps to solve the problem of tracing changes in the technological landscape can be generalized and reused in two ways: it can be applied both to related tasks within the same domain and to a related problem domain, having a similar schema. As an example of reuse of the former, imagine that the same task needs to be executed but the analyst should now analyze the changes in two more recent periods: from 2010 to 2012 and from 2013 to 2014. Therefore, the last five steps of the task execution can be re-evaluated for two different ranges of years:

$\{S_1, S_2, S_3, S_4', S_5', S_6', S_7', S_8', S_9', S_{10}', S_{11}', S_{12}', S_{13}', S_{14}'\} \leftarrow$

$\{S_{1..14}\}.Eval(S_4.2001\$2010, S_4.2002\$2012, S_5.2003\$2013, S_5.2004\$2014)$

The *Eval* function is an intention-oriented function, i.e., its first argument is the functional composition of the task, represented by the range of set indexes $\{S_{1..14}\}$. The remaining arguments are replacements for parameters of some exploration sets of interest. We denote a replacement by the set id followed by the original and the replacement values separated by the replacement operator "$". In the reevaluation above, $S_4.2001\$2010$ represents the replacement of the argument 2001 by 2010 for the operation that generated the set S_4. The result of the *Eval* operator is a set of exploration sets, where, all updated sets and all their dependencies are also reevaluated. In the example above, the result sets from S_4' to S_{14}' are reevaluations of the sets from S_4 to S_{14} for the new arguments.

The second possibility is to reuse solutions in different related domains. For example, we can reuse the composition for discovering technological trends in the scientific publications field to discover changes of interest within research fields. In order to do that, we can replace some schema relations.

Consider a set of research papers *Rp* and a relation *:ResearchField:* $\boldsymbol{Rp}\times\boldsymbol{T}$, where *T* is a set of research fields. The re-evaluation is carried out as follows:

$$\{S_{1..14}'\} \leftarrow \{S_{1..14}\}.Eval(S_1.P\$Rp,\ S_1.:HasIPC\$:ResearchField,$$
$$S_6.:HasIPC\$:ResearchField,\ S_7.:HasIPC\$:ResearchField)$$

In the reevaluation above, the argument replacements are the set of patents *P* by the set of papers *Rp*, and the relation *:HasIPC* by *:ResearchField* in the steps 1, 6, and 7.

5.2.Case Study 2: Evaluating a scientific paper

Here we choose the case study in the scientific publications field to demonstrate the operations. We also selected the Open Citations (PERONI *et al.*, 2015) dataset for the simulation, which is an RDF dataset of scientific publications. The following task was presented in (DI IORIO *et al.*, 2015):

Consider a reviewer evaluating a scientific paper. In order to do so, the user can take the following strategy:

1. Analyze the age of the citations: the reviewer extracts the years of each citation and calculates, for example, the mean year;
2. Check the lack of citations to relevant publications: The reviewer can extract the keywords of the paper and issue a keyword search for related papers; Rank the articles by the number of incoming citations. Keep the first 20 articles; Differentiate the two sets and verify which ones are not in the bibliography of the paper;
3. Analyze the degree of "self-citations": the reviewer analyzes how self-referential is the paper. A self-citation can be either a citation of previous works of one of the authors or citations from authors of the same research group;
4. Evaluate if the paper fits to the scope of a venue: the reviewer might count the number of citations published in the same venue as an indicator of how adequate the paper is to the targeted venue.

Figure 18 shows a representation of the Open Citations schema slightly adapted for demonstrations purposes.

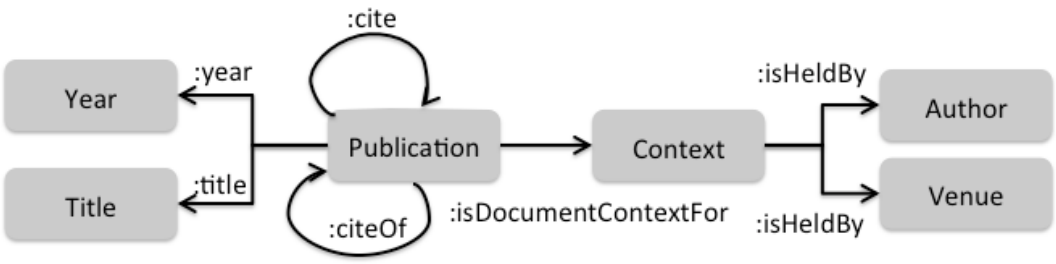

Figure 18 - Open citations summarized schema

The strategy above can be represented as the following sequence of steps in our framework. Let *D* be a dataset of papers, and *p* be a unitary set having the paper under reviewing:

1. $S_1 \leftarrow$ p.*Pivot*(:cite)
2. $S_2 \leftarrow S_1$.*Pivot*(:year)
3. $S_3 \leftarrow S_2$.*Map*(*mean*)

4. $S_4 \leftarrow$ D.*Refine*(matchAll("Semantic Web"))
5. $S_5 \leftarrow S_4$.*Group*(:cite)
6. $S_6 \leftarrow S_5$.*Map*(2, count)
7. $S_7 \leftarrow S_6$.*Rank*(1,c(%item))[0..19]
8. $S_8 \leftarrow S_7$.*diff*(S_1)

9. $S_9 \leftarrow$ p.*Pivot*(:isContextFor:isHeldBy)

10. $S_{10} \leftarrow S_9$.*Refine*(equals(:type, *Author*))
11. $S_{11} \leftarrow S_{10}$.*Pivot*(:isHeldByOf:isContextForOf)
12. $S_{12} \leftarrow S_1$.*Intersect*(S_{11})
13. $S_{13} \leftarrow S_{12}$.*Map*(*count*)

14. $S_{14} \leftarrow$ p.*Pivot*(:isContextFor:isHeldBy)
15. $S_{15} \leftarrow S_{14}$.*Refine*(equals(:type, *Venue*))
16. $S_{16} \leftarrow S_1$.*Refine(equals(*:isContextFor:isHeldBy, S_{15}))
17. $S_{17} \leftarrow S_{16}$.*Map*(*count*)

Step 1 pivots from the set having the paper to the set of citations through the relation *:cite*. Step 2 pivots to the citations' years of publication in order to obtain the average year in step 3.

In order to find papers that are relevant to the field but were not cited, in step 4 the user filters the dataset to find papers related to the Semantic Web area using a keyword filter. In steps 5 and 6, the user, first groups semantic web papers by their outgoing citations using the relation *:cite*, and then counts the groups using a *Map* function. In the next step, the user ranks the semantic web papers by their incoming citations count, to measure their relative relevance, and keeps only the first twenty. A set difference is carried out in step 8 to find the relevant papers that were not in the set of citations (S_1) of the paper being reviewed.

Steps 9 to 13 aims at verifying how self-referential is a paper. In order to do so, the user tries to reach the set of papers published by the authors by pivoting to the publication holders (step 9) and obtaining the authors of the reviewing paper (step 10). Then the user pivots back from the authors to the authors' publications (steps 11). The path *:isContextFor:isHeldBy* relates papers and their holders (authors or venues) in the Open Citations dataset. Subsequently, the user counts the intersections with the set of citations (steps 12 and 13). Finally, the user calculates how many citations were published in the same journal as a measure of adequacy of the reviewed paper to the submitted journal (steps 14 to 17).

5.2.1. Alternative Strategies

As an example of alternative strategies, lets take the steps related to the analysis of how self-referential the paper *p* is. The strategy used is checking how many citations the authors of *p* also published. The user could extend this task to compare the citations against the papers published by the same group of researchers. Since, there is no schema relation for research groups, the explorer tries to approximate them by adding the papers published by the co-authors of the authors of *p* in the comparison. The steps 11 to 13 can be replaced by the following sequence:

12. S_{12}' ← S_{11}.*Pivot*(:isContextFor:isHeldBy)
13. S_{13}' ← S_{12}'.*Refine*(equals(:type, *Author*))
14. S_{14}' ← D.*Refine*(*and*(*equals*(:type, *Publication*), *equalsOne*(:isDocumentContextFor:isHeldBy, S_{13}'))
15. S_{15}' ← S_1.*Intersect*(S_{14}')

16. S_{16}' ← S_{15}'.*Map*(*count*)

In alternative step 12, the user pivots from the publications of *p* authors (set S_{11}) to the set of all holders. Next in step 13, the user refines the set of holders in order to keep only those that are authors, thus, excluding the venues. The set S_{13}' includes both the authors of *p* and their co-authors. In the alternative step 14, the user applies a conjunctive filter for all items of the type *Publication* having at least one author in the set S_{13}' (*equalsOne* function). In step 15, the user computes the intersection between the citations in S_1 with the set of publications of both the authors of *p* and their co-authors. The final step is to count the intersection results, which can be found in S_{16}'.

Another interesting case of alternative strategy can be applied to the steps 16 and 17 of the original task. This step aims at refining the citations that were published in the same venue as *p*. Some users more familiarized with browsing actions may prefer to navigate the set of venues of the citations, in order to learn and make sense of them, before refining. Therefore, the steps 16 and 17 could be replaced by the following sequence of steps:

16. S_{16}' ← S_1.*Pivot(*:isDocumentContextFor:isHeldBy*)*
17. S_{17}' ← S_{16}'.*Refine*(equals(:type, *Venue*))
18. S_{18} ← S_{17}'.*Pivot*(isHeldByOf:isDocumentContextForOf)
19. S_{19} ← S_1.*Intersect(*S_{18}*)*
20. S_{20} ← S_{29}.*Map*(*count)*

In the alternative steps 16 and 17, the user decided to pivot from the citations to the set of holders and refine the venues in order to make sense of them. Next, the user pivots back to the set of all publications held by the venues of the citations in step 18, and intersect with the citations. The last step is to count the amount of intersections in step 20.

Although further analyzes should be carried out, we believe that the sequence of steps employed may reveal some characteristics of the user profile. For example, the alternative sequence to count citations published in the same venue as the reviewing paper may reveal a user that has more familiarity or prefer browsing operations. Another possible case is the lack of knowledge with regards to publications venues of a specific area, which forced him/her to explore this area

of the dataset. We are not judging here which sequence is better, since there are many variables involved. The main goal, though, is to emphasize the utility of the framework as an epistemic tool for investigations of alternative exploration strategies.

5.2.2.Generalization and Reuse

Considering an inexperienced reviewer carrying out the same task, s/he can miss the self-citation analysis step. In this case, at least two actions can be taken. 1 – the system can identify in its database that other users have carried out the same task and their solution has a greater number of sessions and steps, thus, the system suggests additional actions; 2 – before finishing the task, the user issues a query for existing previous compositions for the same task and verifies that other reviewers have considered further steps. The user, therefore, decides to incorporate the self-citation analysis to his solution.

The main goal of the reuse in this case is the transference of knowledge not only with regards to the results of the tasks but also concerning the resolution processes. Therefore, new users can draw upon the experience of previous users to aid their task resolution strategies.

5.3. Case Study 3: Summarizing Gene Clusters

The technique of representing gene expression events as microarrays brought new possibilities of computations and analysis of gene co-expressions and expression conditions. The microarray data is organized in the form of a matrix where each row represents a gene and each column represents different environmental conditions (EISEN *et al.*, 1998). The problem in making sense of microarrays is the lack of information about the genes involved in the processes, where only the gene identifiers are present. Gathering data concerning the genes in the cluster usually requires some degree of exploration of one or many datasets. Therefore, in this context, gene clusters can be considered starting points of a sequence of future exploration actions targeting at making sense and explaining the functions and relations between the genes within the clusters.

In the context of exploration, a common operation over microarray datasets is to create clusters of genes based on similarity degrees (KANKAR *et al.*, 2002). Such operation is useful to discover, for example, which genes are expressed in the same time slice of a specific biological process or tissue type. Gene clusters

generated solely based on microarray data usually needs to be cross-referenced with known biological facts, theories, and results originated from large amounts of research materials.

The task consists in crossing the gene identifiers of the cluster with a bibliographical dataset in order to find terms that better describe the genes in the cluster. The strategy employed is the following:

1. A gene can have many identifiers. Therefore, the user tries to obtain all identifiers for each gene in the cluster;
2. Once having a more complete set of identifiers, the user tries to query a bibliographic dataset in order to find all publications that mention the gene identifiers;
3. From publications, find the terms that better describe the genes;
4. Rank the terms using specific ranking criteria designed to extract different information from the cluster.

Three distinct and interrelated datasets are used in the execution of the task described in (KANKAR *et al.*, 2002). The datasets are:

- *M*: the cluster of genes achieved through microarray clustering;
- *PubMed*[10]: a database of citations and abstracts from where the summary of the cluster is extracted;
- *G*: a general dataset of genes and gene descriptions, such as the datasets of NCBI[11];

Consider the schema of Figure 19 for the execution of the task.

[10] https://www.ncbi.nlm.nih.gov/pubmed/
[11] https://www.ncbi.nlm.nih.gov/

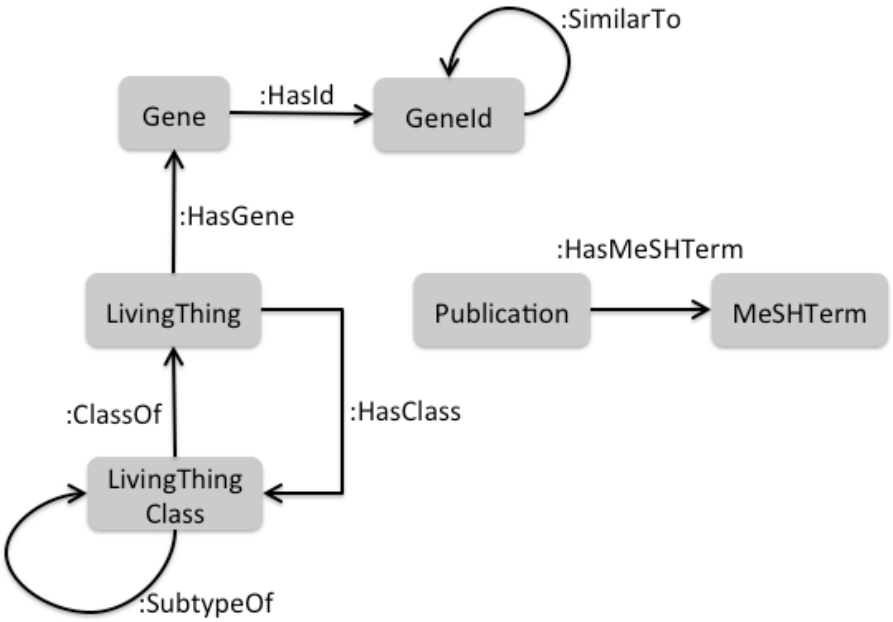

Figure 19 - Schema of a gene dataset

The following steps describe the summarization of *M* based on the publications in *PubMed*.

1. $S_1 \leftarrow$ M.*Pivot*(:GeneId)
2. $S_2 \leftarrow S_1$.*Pivot*(:SimilarTo)
3. $S_3 \leftarrow S_1$.*Unite*(S_2)
4. $S_4 \leftarrow$ PubMed.*Refine(matchOne(*S_3*))*
5. $S_5 \leftarrow S_4$.*Pivot(*:HasMeshTerms*)*

6. $S_6 \leftarrow$ G.*Refine(equals(:Type, "LivingThingClass"))*
7. $S_7 \leftarrow S_6$.*Pivot*(:SubtypeOf^n)
8. $S_8 \leftarrow S_7$.*Pivot*(:ClassOf:HasGene)
9. $S_9 \leftarrow \{S_{1..5}\}$.*Eval*($S_1$.M$S_8$)[5]
10. $S_{10} \leftarrow S_9$.*Rank*(1, *freq*(%term))[0..100]
11. $S_{11} \leftarrow S_5$.*Diff*(S_{10})

12. $S_{12} \leftarrow S_9$.*Rank*(1, *r1*(*normFreq*(%term))
13. $S_{13} \leftarrow S_9$.*Rank*(1, *r2*(*normFreq*(%term))
14. $S_{14} \leftarrow S_9$.*Rank*(1, *r3*(*normFreq*(%term))

The explorer is aware that many distinct identifiers can mention the same gene. Therefore, s/he pivots to the set of gene identifiers, and then, to the set of

similar identifiers in the base (steps 1 and 2). A union is carried out in step 3 to obtain a complete list of gene identifiers. In step 4, the explorer uses this extended set of identifiers to retrieve publications in *PubMed* using a keyword refine with the *matchOne* filter.

The results of the keyword search (step 4) are a set of documents that contains the gene identifiers, the article titles, the abstracts, the authors, and the MeSH keywords. MeSH (Medical Subject Headings) is a controlled vocabulary for indexing medical articles. After that, in step 5 the explorer extracts the MeSH terms related with each gene by the relation *:HasMeshTerm*.

In step 4, the explorer has a list of terms that is supposed to give an overview of the subjects that the cluster *M* concerns with. Nevertheless, among those terms there are also general terms, such as, “gene”, “DNA”, “sequence”, “animal”, etc., which are useless for particularly describing the genes in *M*. Therefore, the explorer decided to employ a strategy to eliminate the words that are associated with genes in general in steps 6 to 11.

In order to filter out the stop words, the explorer decided to use a large set of genes from different classes of organisms, such as Eukaryotes and Prokaryotes, and eliminate the words that have high frequency among the classes. Since the classes are disjoint, high frequency terms appearing on both classes should be too general for describing the gene cluster.

In step 6, the explorer queries the dataset for all items of the type “LivingThingClass”. In step 7, the explorer tries to obtain the root classes of organisms, through a relation path of size *n* of the relation :SubtypeOf^{n}, where *n* is the distance to the root. The root classes should be the most disjoint possible.

In step 8 s/he pivots to the set of genes of the organisms within the classes through the relation path :ClassOf:HasGene. Step 9 extracts the MeSH terms from the genes found in step 8 by reevaluating the same composition used to extract the MeSH terms for the genes of the cluster *M*, which is a case of reuse within the task. Step 10 ranks the set of terms by the frequency of documents using the *freq(term)* function, thus, finding the most popular terms. Step 11 finally eliminates the stop words from the set of terms related to the genes of cluster *M*.

Once having a set of MeSH terms that can be used for describing the genes of cluster *M*, the next goal of the explorer in step 12 is to discover the most popular terms to get an overview of the functions of the genes in *M*. In order to

extract the most popular terms, considering only the frequency of documents as a ranking criteria can bias the results due to the skew on the number of publications addressing each gene. Well-studied genes tend to have greater number of publications, which can impact in the ranking position of the MeSH terms. Therefore, the explorer decided to use the *mean* of the normalized term frequency (*normFreq*) as ranking criteria *r1*. The formula of the normalized term frequency is defined in (KANKAR *et al.*, 2002).

Step 13 ranks the MeSH terms that have a high total frequency among all genes. However, there are also terms that are associated with most of the genes but have moderate-to-low frequency. Such terms is expected to have moderate mean and low variance. Therefore, the ranking criteria is defined by *r2= mean/standardDeviation* of the normalized frequency of the term.

The last ranking criteria presented in (KANKAR *et al.*, 2002) ranks the terms that appear most in a subgroup of the genes in *M*. Such terms is expected to have high variance and moderate-to-low total frequency. Hence, the ranking criteria is defined by *r3 = variance/mean*.

5.4.Alternative Strategy

As an alternative strategy, consider the steps 6 to 11 of the original task. These steps aims at finding and eliminating general terms that apply to most of the genes, hence, being too abstract to describe the functions of the genes in *M*. In order to do that, the user navigates through the living things classification hierarchy to find genes that appears in the most abstract classes. Instead of navigating through the living things classification taxonomy, a more skilled user could simply specify the classes that s/he knows to be abstract enough for the task. In order to that, the steps 6 to 8 could be replaced by the following step:

```
G.Refine(
        and(
            equals(:Type, Gene),
            containsAll(:HasGene-1:HasClass, [“Eukaryota”,
            “Bacteria”, “Virus”])
        )
    )
```

5.4.1.Generalization and Reuse

As a reuse scenario for this task, we can think of the same explorer having to make sense of a different cluster of genes. In order to that, s/he could reevaluate the exploration trail for a different set of genes, thus, replacing the cluster *M* by the new cluster. Let *C* be a new cluster of genes. The reevaluation is represented as:

$$\{S_{1..14}'\} \leftarrow \{S_{1..14}\}.Eval(S_1.M\$C)$$

In the steps 12, 13, and 14 of the original task, the user applies different ranking strategies on the set of MeSH terms to make sense of the genes in the cluster. However, (KANKAR *et al.*, 2002) also describes a score function that is a combination of the three score functions applied. The user could, not only reevaluate the exploration for a new cluster *C*, but also apply a new ranking using the combined score function *r4*:

1. $\{S_{1..14}'\} \leftarrow \{S_{1..14}\}.Eval(S_1.M\$C)$
2. $S_{15} \leftarrow S_{14}'.Rank(1, r4(normFreq(\%term))$

This reuse case demonstrates the possibility of adapting and improving previous explorations. We expect that repeated auditing and adaptations of functional compositions could result in accurate and efficient data flow machines (DEELMAN *et al.*, 2009) that would benefit the whole community of data consumers.

5.5.Conclusions

This chapter presented three case studies, where, the main goal is to demonstrate the potential of our framework both to describe exploration tasks relevant to communities of users, and to support adaptations and reuse of previously discovered exploration paths. Thereby, we selected well-documented exploration problems. The conclusions drawn from the case studies can be summarized as follows:

- The framework is useful for formally describing relatively complex exploration tasks of different domains of knowledge;
- The framework leverages analyzes of the task resolution process abstracting interface and interaction details. Therefore, the framework can be used as an epistemic tool for design decisions, where the

designers could use it to devise alternative exploration paths and analyze which sequences is mostly indicated given the task execution context and user profile.

- Once the solution strategy employed can be formally represented, they can be shared and reused either in different scenario within the same domain or within different domains. Reuse in different domains will require adaptations of the schema used in the task, which can be achieved by parameterizations. Moreover, by representing exploration strategies formally, it is possible to audit the strategies for results validation purposes, which is of a great value for validating scientific results, for example.