REX – a tool for discovering evolution trends in ontology regions

Victor Christen[1], Anika Groß[1,2], and Michael Hartung[1,2]

[1] Department of Computer Science, Universität Leipzig, Germany
[2] Interdisciplinary Center for Bioinformatics, Universität Leipzig, Germany
mam08bfa@studserv.uni-leipzig.de,
{gross,hartung}@informatik.uni-leipzig.de

Abstract. A large number of life science ontologies has been developed to support different application scenarios such as gene annotation or functional analysis. The continuous accumulation of new insights and knowledge affects specific portions in ontologies and thus leads to their adaptation. Therefore, it is valuable to study which ontology parts have been extensively modified or remained unchanged. Users can monitor the evolution of an ontology to improve its further development or apply the knowledge in their applications. Here we present REX (Region Evolution Explorer) a web-based system for exploring the evolution of ontology parts (regions). REX provides an interactive and user-friendly interface to identify (un)stable regions in large life science ontologies and is available at http://www.izbi.de/rex.
Keywords: ontologies, ontology evolution, graph vizualisation

1 Introduction and Background

In recent years ontologies have become increasingly important for annotating, sharing and analyzing data in the life sciences [1,8]. The heavy usage of ontologies leads to a steady modification of their content [7,9]. In particular, ontologies are adapted to incorporate new knowledge, eliminate initial design errors or achieve changed requirements. Tools like Protégé [16] support the development and change of ontologies. This process is usually distributed since especially large ontologies can not be maintained by single developers, such that collaborative work is performed [3,16]. Typically, the overall development of an ontology is coordinated by a project leader or consortium, and multiple developers contribute knowledge in their field of expertise.

Due to the ontology's size and complexity, the problem arises that coordinators, developers and users want to know whether specific parts (regions) of a large ontology have changed or not. For instance, if a user considers the anatomy part of the NCI Thesaurus (NCIT) [13] for annotating local data such as radiology pictures, she would like to know how this part has evolved recently, i.e., is the part unstable or stable. Unstable regions have been in the focus of development and underlay many modifications. By contrast, a stable region might be already completed or was of low interest during recent ontology development.

Project coordinators are interested in the evolution of different ontology parts (1) to see how work has progressed and (2) to detect potential for future development. Moreover ontology-based algorithms or applications might be affected by ontology changes. For instance, if results of a gene set enrichment analysis [15] are located in a strongly evolving ontology part, it should be re-done based on the newest ontology version to see how results change [2]. By contrast, results located within stable ontology parts are likely to remain unchanged.

Currently, life science ontologies can be accessed through platforms like BioPortal [10] and OBO Foundry [14]. Although it is possible to retrieve different versions of an ontology, such platforms rarely provide information about evolution, i.e., users have the problem to figure out how an ontology has evolved compared to their version in use. Recently, some web tools offer access to information about the evolution of the Gene Ontology (GO). GOChase [12] allows to study the history of individual GO concepts and Park et al. [11] propose graph-based visualization methods to view modified GO terms. In own previous work we designed the OnEX web application [6] for quantitative and concept-based evolution analyses in life science ontologies. Our tool CODEX [5] can be used to determine a diff between two ontology versions covering complex changes (e.g., concept merge or split). For a general overview on ontology and schema evolution including diff computation we refer to [7]. In summary, currently available tools lack the functionality to analyze and compare evolution in different ontology parts especially for large ontologies with several version releases. In own previous work [4] we already proposed an algorithm to detect (un)stable ontology regions for an arbitrary number of ontology versions. However, the algorithm is only applicable offline, i.e., the research community can not make use of it.

We therefore present the novel web application REX (Region Evolution Explorer) based on the region discovery algorithm [4]. REX can be used (1) to determine differently changing regions for periodically updated ontologies, and (2) to interactively explore the change intensity of those regions. REX provides a comparative trend analysis such that users and developers can monitor the long-term evolution for their regions of interest, e.g., to track the work or coordinate future development. REX is online available at `http://www.izbi.de/rex`.

2 Region Discovery Method

The region discovery method proposed in [4] enables the detection of changing and stable ontology regions. The basic idea is to compute change intensities for regions based on changes between several succeeding versions of an ontology within a specific time interval. The algorithm consists of four main steps: (1) change computation, (2) cost propagation, (3) cost transfer, and (4) region discovery. It first computes differences between two versions to determine changes. It then propagates change costs within the *is-a* hierarchy of the ontology and transfers these costs from the first to the last considered version. Based on computed change intensities we can discover differently evolving ontology regions. First, we briefly describe the method for two input versions O_{old} and O_{new}.

	Change operation	Description	Change costs
Concepts	*addC*	addition of a new concept	1
	delC	deletion of a concept	2
Relationships	*addR*	addition of a new relationship	0.5/0.5
	delR	deletion of a relationship	1.0/1.0
Attributes	*addA*	addition of a new attribute	0.5
	delA	deletion of an attribute	0.5
	chgAttValue	modification / change of an attribute value	0.5

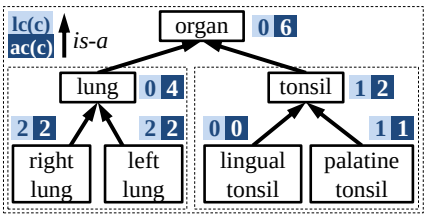
Table 1. Change operations and change cost model used in REX.

In general, ontology content can be added (addition), removed (deletion) or modified (update). Here we distinguish between the basic change operations for ontology concepts, their attributes and relationships between concepts listed in Table 1. Our region discovery method assigns so-called local costs $lc(c)$ to concepts to cover the impact of changes that directly influence a concept c (see change costs in Table 1). For instance, we can assign higher costs to deletions since they might have a higher impact on dependent applications than additions. Note, that the cost model can be adapted according to the application scenario. Additions are registered in the new version while deletions are covered in the old version. Moreover, the assignment depends on which ontology element has changed. Here we assign costs from changes on a concept or its attributes to the concept itself. Costs for relationships are split and assigned to the source and target concept of the relationship, respectively. The local costs are then propagated along *is-a* paths upwards in the ontology hierarchy to obtain aggregated costs. Due to multi-inheritance we may need to split costs during propagation. We therefore determine aggregated costs $ac(c)$ for a concept c as follows:

$$ac(c) = \sum_{c' \in children(c)} \frac{ac(c')}{|parents(c')|} + lc(c)$$

We thus ensure that the root concept(s) of the ontology contain the overall sum of all assigned local costs. Fig. 1 (left) shows an exemplary anatomy ontology with local and aggregated costs. For instance, the aggregated costs of 'organ' ($ac('organ') = 6$) are computed based on the aggregated costs of its children $ac('lung') = 4$ and $ac('tonsil') = 2$ as well as its own local costs $lc('organ') = 0$.

In order to determine (un)stable regions in the new version, we need to transfer aggregated costs from O_{old} into O_{new}. We therefore sum up aggregated



Region Measures

region	*abs_size*	*abs_costs*	*avg_costs*
organ	7	6	0.86
lung	3	4	1.33
tonsil	3	2	0.67

Fig. 1. Example anatomy ontology with regions and local ($lc(c)$) and aggregated ($ac(c)$) concept costs (left). Region measures for example ontology (right).

costs which belong to the same concept in the old/new version. The new ontology version with aggregated costs is used for further processing. The two version method is generalized for multiple released versions $O_1, \ldots, O_n$ by executing it $n-1$ times so that we successively determine aggregated costs (for each version change $O_{i-1} \mapsto O_i$) and transfer them to the newest version O_n. In O_n we can apply different measures to detect ontology regions and their change intensities. For further details about the underlying algorithms we refer to [4].

An ontology region OR consists of an ontology concept (region root rc) and its *is-a* subgraph, i.e. it covers all leaf and inner concept changes within this region. For our example in Fig. 1 we can consider the regions 'lung' and 'tonsil' each consisting of three concepts. Note that the complete ontology can also be regarded as a region defined by the ontology root 'organ'. So far, REX provides a set of measures to describe the change intensity of ontology regions. For each OR one can determine its absolute size ($abs\_size(OR)$) w.r.t. the number of concepts. Absolute change costs of an OR ($abs\_costs(OR)$) are represented by the aggregated costs of its root $ac(rc)$. The average change costs per concept in OR can be computed as the fraction of absolute change costs and the region size: $avg\_costs(OR) = \frac{abs\_costs(OR)}{abs\_size(OR)}$. Applying these measures to our example results in the values displayed in Fig. 1 (right). The 'lung' region changed more intensively ($avg\_costs('lung') \approx 1.33$) compared to 'tonsil' ($avg\_costs('tonsil') \approx 0.67$). The overall change intensity of the ontology is $\frac{6}{7} \approx 0.86$.

Trend Discovery for Regions Using the region discovery method one can determine the most (un)stable regions for a specific time interval. To better monitor region changes over long periods of time and to figure out trends in their evolution, we propose a further method for trend discovery based on sliding windows. The overall procedure `trendDiscovery` looks as follows:

Algorithm 1: trendDiscovery

Input: time interval (t_{start}, t_{end}), ontology O, ontology region of interest $OR \in O$, change costs σ, window size ω, step width Δ
Output: time-based stability values $measuredCosts$

```
1 t ← t_start; measuredCosts ← ∅;
2 while t + ω < t_end do
3     versions ← getReleasedVersions(O, (t − ω, t));
4     latestVersion ← discoverRegions(versions, σ);
5     regionCosts ← getStabilityValuesForRegion(OR, latestVersion);
6     measuredCosts.put((t, regionCosts));
7     t ← t + Δ;
8 return measuredCosts;
```

The algorithm works on an ontology O, a time interval (t_{start}, t_{end}) and an ontology region of interest OR to be monitored. We further use a sliding window of size ω, a step width Δ and change costs σ. In particular, we successively shift the window beginning at $t_{start} - \omega$ over the time interval until we reach its end t_{end}. In each step we first determine the released ontology versions within the

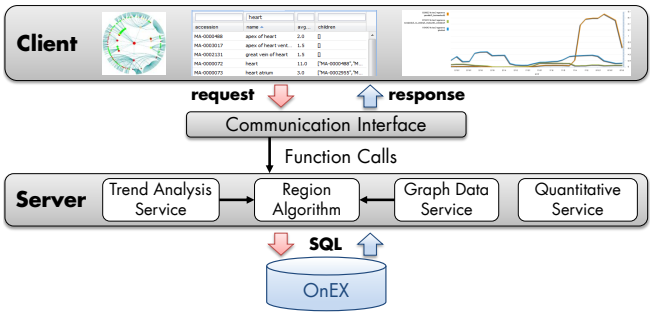


Fig. 2. Three-layered architecture of REX.

window (line 3). We then calculate and save the costs (e.g., *avg_costs*) for OR by calling the region discovery algorithm (`discoverRegions`) for the versions within ω. We thus generate a time-based map (line 6) containing information about the change intensity of OR at specific points in time in the defined window. The results are visualized for end users in the *Trend Analysis* component of REX.

3 Infrastructure and Application

REX is based on a three-layered architecture displayed in Fig. 2. The back-end consists of the OnEX repository [6] which currently provides access to up to 1,000 versions of 16 popular life science ontologies. Note that it supports the import of ontologies in different formats such as OWL and OBO. Users can analyze integrated versions with the offered facilities of REX. The server layer is implemented in Java and realizes different services to access ontology versions in OnEX. Moreover, it provides services to calculate the region measures and to perform trend and quantitative analyses. Every service is encapsulated in its own module, such that it is possible to change the region discovery algorithm independently of the other modules. Results are transformed such that the application can visualize ontologies and changing regions in graphs. Based on the existing services we can create further interfaces like web services for programmatic access. The front-end is a platform-independent web application based on the Google Web Toolkit (GWT)[3] and the graph library InfoVis[4]. In the following we discuss the analysis facilities of REX, namely the *Structural Analysis*, *Trend Analysis* and *Quantitative Change Analysis* in more detail.

Structural Analysis The structural analysis component represents the evolution of regions in an ontology for a specified time interval as a graph (Fig. 3). The component is divided into a *Browser View* as well as a table to search and filter results (*Table View*). First the user needs to specify the ontology name

[3] Google Web Toolkit: `http://developers.google.com/web-toolkit/`
[4] InfoVis Toolkit: `http://philogb.github.io/jit/`

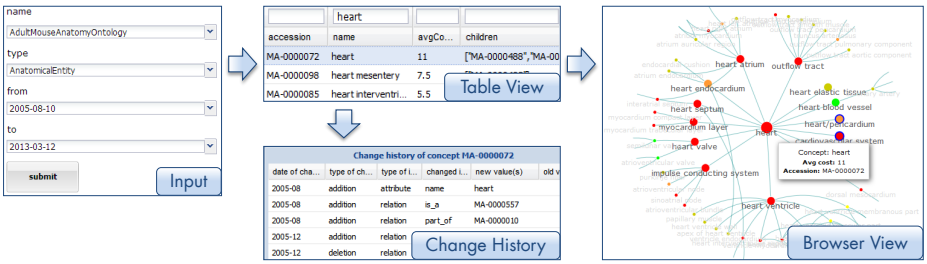


Fig. 3. Structural Analysis component.

and the time period to review in the *Input* form. The system then performs the region discovery algorithms and generates a graph to visualize the results (*Browser View*). Each node in the graph represents an ontology concept, *is-a* relationships are displayed as edges between the nodes. The layout is circular and displays a concept and its near neighborhood, i.e. its descendants and parent nodes (either with or without labels). Users can easily identify interesting sub regions by selecting a concept in the graph (*Browser View*) or in the *Table View*. This concept is then shown as the central node in the *Browser View*. It is possible to navigate in both directions through the ontology. For instance, if one is interested in a specific sub region and its content, one clicks on the node and the graph will display the sub region in more detail. In contrast, one can also navigate to a more general concept (surrounded by blue circles) to see sibling regions of the current one. The colors signal the measured change intensity (*avg_costs*) of a region. Red stays for high change intensity whereby green is used to mark stable regions. Thus, users can easily figure out where (un)stable regions are located. We provide two coloring schemes: (1) interval-based grouping or (2) equal distribution between min/max *avg_costs*. When clicking on a specific concept in the graph one can get further information like the accession number, concept name/label or the measured *avg_costs* in a pop up window.

In general the number of concepts and relationships in an ontology is very high. Thus, it is difficult to recognize interesting regions only by browsing through the graph especially for large ontologies. Moreover, users may be interested in the change intensity of specific regions. The *Table View* therefore allows users to filter and sort ontology regions by their accession number, name and *avg_costs*. In particular, search criteria can be specified in the head of the table to find regions of interest. For instance, one can filter out all regions in the Adult Mouse Anatomy Ontology containing the name 'heart'. Users can simply select their region of interest in the table and move to the *Browser View* for its visualization. To get a more detailed view of occurred changes, users can request the local *Change History* of a selected concept at the bottom of the table.

Quantitative Change Analysis To get information about how many changes occurred in an ontology for a specific time interval REX offers the quantitative change analysis component (Fig. 4 left). Users can generate diagrams to see

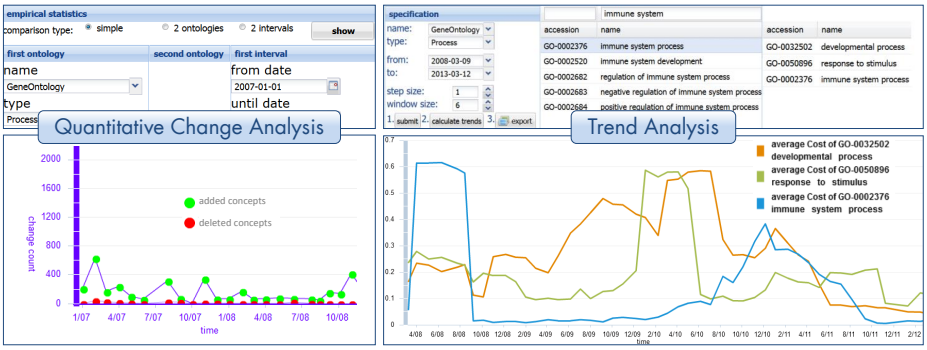


Fig. 4. Trend and Quantitative Change Analysis components.

the differences between released ontology versions in statistical (quantitative) form, i.e., we count and visualize how many changes ($addC$, $delC$, $addR$, $delR$) occurred. In particular, users can display the number of changes in one ontology for a specific time interval, e.g., GO Biological Processes in 2011. Moreover, one can compare the evolution of two different ontologies for a specified time interval or compare two different time intervals for the same ontology. Users can thus identify interesting ontologies and time periods for later region analyses.

Trend Analysis The trend analysis component can be used to study and compare the long-term evolution of selected regions (Fig. 4 right). Users first need to specify the ontology, the time interval (first and last version) and the window size and step width (number of versions). Next they are able to select regions of their interest either by searching the respective accession number / concept name or by choosing from top-level concepts of the ontology. REX executes the proposed `trendDiscovery` algorithm to measure the avg_costs for the selected regions at different points in time. The results are converted into a line chart which displays the trend of the measured avg_costs for each region over time. Users are thus able to compare the change intensity for different regions of interest within one diagram. Of course, the interpretation of trend results is up to the user and depends on the application scenario. Some regions may be of high research interest and are thus continuously adapted (constantly high avg_costs). Other regions have been adapted heavily in the past and become stable after a while. By contrast, some long-term stable regions might have just been of low interest in the past and need future development.

4 Conclusion and Future Work

REX provides interactive access to information about the evolution of life science ontologies. Users can explore (un)stable ontology regions by different workflows. The knowledge about changing ontology regions can be used to support ontology-based algorithms and analysis. Furthermore, the development of large life science

ontologies can be monitored with REX, i.e., developers and project coordinators can inform themselves about ongoing work in different ontology parts. For future, work we plan to extend REX such that users are able to perform region analysis on their individual ontologies and can apply different cost models. We further like to build a web service interface such that algorithms can directly access the region analysis algorithms.

Acknowledgment

This work is funded by the European Social Fund and the Free State of Saxony (E-Science Network Sachsen).

References

1. Bodenreider, O., Stevens, R.: Bio-ontologies: current trends and future directions. Briefings in Bioinformatics 7(3) (2006)
2. Groß, A., Hartung, M., Prüfer, K., et al.: Impact of ontology evolution on functional analyses. Bioinformatics 28(20) (2012)
3. Groza, T., Tudorache, T., Dumontier, M.: Commentary: State of the art and open challenges in community-driven knowledge curation. Journal of Biomedical Informatics 46(1) (2013)
4. Hartung, M., Gross, A., Kirsten, T., Rahm, E.: Discovering Evolving Regions in Life Science Ontologies. In: Data Integration in the Life Sciences (2010)
5. Hartung, M., Gross, A., Rahm, E.: CODEX: exploration of semantic changes between ontology versions. Bioinformatics 28(6) (2012)
6. Hartung, M., Kirsten, T., Gross, A., Rahm, E.: OnEX: Exploring changes in life science ontologies. BMC bioinformatics 10(1) (2009)
7. Hartung, M., Terwilliger, J.F., Rahm, E.: Recent Advances in Schema and Ontology Evolution. In: Schema Matching and Mapping. Springer (2011)
8. Lambrix, P., Tan, H., Jakoniene, V., Strömbäck, L.: Biological ontologies. In: Semantic Web. Springer (2007)
9. Malone, J., Stevens, R.: Measuring the level of activity in community built bio-ontologies. Journal of Biomedical Informatics 46(1) (2013)
10. Noy, N.F., Shah, N.H., Whetzel, P., et al.: BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37(suppl 2) (2009)
11. Park, J.C., Kim, T., Park, J.: Monitoring the evolutionary aspect of the gene ontology to enhance predictability and usability. BMC Bioinformatics 9 (2008)
12. Park, Y.R., Park, C.H., Kim, J.H.: GOChase: correcting errors from Gene Ontology-based annotations for gene products. Bioinformatics 21(6) (2005)
13. Sioutos, N., Coronado, S.d., Haber, M.W., et al.: NCI Thesaurus: a semantic model integrating cancer-related clinical and molecular information. Journal of Biomedical Informatics 40(1) (2007)
14. Smith, B., Ashburner, M., Rosse, C., et al.: The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nature Biotechnology 25(11) (2007)
15. Subramanian, A., Tamayo, P., Mootha, V.K., et al.: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. PNAS 102(43) (2005)
16. Tudorache, T., Noy, N.F., Tu, S., Musen, M.A.: Supporting collaborative ontology development in Protégé. In: The Semantic Web-ISWC 2008. Springer (2008)