Semantics for Big Data Integration and Analysis

Craig A. Knoblock and Pedro Szekely
University of Southern California
Information Sciences Institute and Department of Computer Science
4676 Admiralty Way
Marina del Rey, CA 90292

Abstract

Much of the focus on big data has been on the problem of processing very large sources. There is an equally hard problem of how to normalize, integrate, and transform the data from many sources into the format required to run large-scale analysis and visualization tools. We have previously developed an approach to semi-automatically mapping diverse sources into a shared domain ontology so that they can be quickly combined. In this paper we describe our approach to building and executing integration and restructuring plans to support analysis and visualization tools on very large and diverse datasets.

Introduction

Developers often spend more time extracting, reformatting, and integrating data to run analysis and visualization programs than they do working on the analyses themselves. Such data reshaping programs are difficult to write because of their complexity, but they are required because each analytic tool expects data in a very specific form and to get the data into that form typically requires a whole series of cleaning, normalization, reformatting, integration, and restructuring operations. The problem is made even more challenging in the context of "big data", where the sources can be large (millions to billions of records, gigabytes to terabytes of size), streaming, and heterogeneous, which means that a developer cannot possibly review all of the data and the data preparation must be run in a scalable way on these large datasets. To achieve the goal of rapidly applying data analytics and visualization tools on new problems requires an approach to rapidly define correct data reshaping plans and then executing them in a scalable way.

Approach

We are addressing the data preparation problem by allowing a developer to interactively define a correct data preparation plan on a small sample of data and then execute such a plan in a parallel, streaming execution environment. To achieve this goal we are building a comprehensive set of data transformation operations that can handle a wide range of real-world data, including structured information as well as semi-structured data, and developing an easy-to-use approach that allows a developer to quickly define correct data reshaping plans that not only transform data, but can easily restructure the output of one tool into the input of another tool.

In previous work, we developed an interactive approach to extracting, modeling, and publishing data in a system called Karma (Tuchinda, Knoblock, and Szekely 2011; Knoblock et al. 2012; Taheriyan et al. 2012; 2013; Wu, Szekely, and Knoblock 2012). The focus on this previous work was on building an approach that allows an end-user to solve their own integration tasks without having to program those tasks. We are building on this core idea of easily defining an integration task, but also addressing the challenge of how to define data transformation tasks and perform these tasks in a scalable way. In the original Karma, the system performs the task interactively on small to moderate size datasets.

In this effort, we are addressing three challenges that are not addressed in our previous work: (1) how to provide the rich set of capabilities required to prepare datasets for both analysis and visualization tools, (2) how to make it easy to define these complex data preparation plans, and (3) how to perform these tasks at scale on massive datasets.

Karma provides tools to semi-automatically build a semantic description (or model) of a data source. This model makes it possible to rapidly map a set of sources (represented in XML, KML, JSON, structured text files, or databases) into a shared domain ontology, which supports the integrated reasoning and processing of data across sources. Once we have modeled the data sources, they are then converted into a variety of formats, including RDF, and published so that various analysis processes can reason over the integrated datasets.

In order to apply Karma to the problem of big data, we plan to start with the same core capabilities to be able to quickly model sources, which allows us to automate many of the required transformations, and then develop new data restructuring capabilities and then execute this restructuring on big datasets. The fact that the system can rapidly build a model of a source means that Karma would enable a user to quickly define a restructuring plan. As shown in Figure 1, the user would define the restructuring plan on a small subset of the data, and then Karma would build the general plan and execute that plan in a distributed environment over the entire dataset.

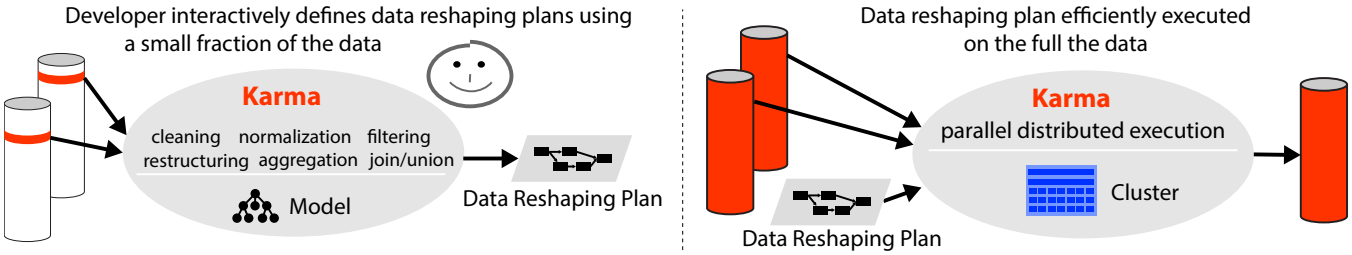

Figure 1: A user would build a data restructuring plan on a sample of the data; Karma would execute the plan at scale

A Detailed Example

We envision using Karma to create general integration pipelines by demonstrating them to Karma on specific subsets of their data. A user would do this interactively by working in a graphical user interface to gather, combine and process data, working with a small sample so that they can see the results step by step. After demonstrating the desired steps, Karma saves a general script that can be easily modified and can be executed in batch mode on larger datasets.

Figure 2 illustrates the process. Here, the user gathers data from several bioinformatics data sources, KEGG and PharmGKB, and downloads the data into Karma. Because these sources are part of Karma's library, the system automatically imports the data and shows the source model of the data. The model for the KEGG data is simple, as the data just contains attributes of genes. However, the model for the PharmGKB data is more sophisticated as it contains data about pathways, genes, drugs, and diseases. The model shows how the various elements are related using the terminology defined in the ontology for these sources. After the gathering phase, users can easily combine the data in a semantically consistent way. In the example, the user combines the datasets based on the Gene class that is present in the models of both sources. This allows users to easily incorporate attributes from KEGG into the dataset gathered from PharmGKB.

If Karma did not already have a model of a specific source, it would attempt to build one automatically. Karma automatically suggests mappings for each data column to classes and properties in the ontology. These mappings are based on learned models of semantic types that the system encountered previously. For the table of data from PharmGKB, Karma would automatically suggest that the AccessionId column is mapped to the pharmGKBId of Pathway because it has seen similar data before. Whenever suggestions are incorrect or missing, users can click on the black circles to correct them. Most importantly, Karma builds the model that ties the columns together. In this example, Karma would suggest using classes such as Disease, Pathway, Gene, and Drug and suggest the properties to connect these classes. When the suggestions are incorrect, users can change them by selecting the appropriate classes and properties from the ontology (user adjustments are shown in blue). Karma includes a learning module that learns from these adjustments, iteratively improving its accuracy the more it is used with similar data (Taheriyan et al. 2013).

In general, users would build combined datasets to process them using appropriate analysis and visualization tools, as shown in Figure 2. We illustrate how a user would visualize their combined dataset using Cytoscape. First, the user would select Cytoscape from the Karma library of tools. Because this tool would be in the Karma library, it would have a service model that specifies the semantics and syntax of the inputs and outputs. Cytoscape is a network visualization tool, so its model would specify that it needs a dataset containing edges that connect vertices. To invoke Cytoscape on the combined PharmGKB source, the user would need to map the classes in the PharmGKB model to the classes in the Cytoscape model. By leveraging the structure of the ontologies and learning from previous uses of the tool, Karma would propose mappings that users can adjust using an interactive user interface (similarly to how today's Karma semi-automatically generates models of sources). The service model would also specify how to generate the input files and how to invoke the tool.

Users would design and test the process interactively, on a small data sample, and then save a plan that the system would use to perform all the steps in batch. Figure 2 does not illustrate it, but the service models would also include a model of the outputs. This means that the outputs would not be just files, but modeled datasets that researchers can publish and share with others without having to expend additional effort to annotate them with metadata. These datasets would be ready to augment the library of datasets that the whole community can use.

Related Work

Developers today approach data preparation either by laboriously defining and writing programs or by using existing data warehousing tools, neither of which provides the flexibility to cope with diverse and changing data sources, nor the scalability to cope with large, streaming data. We have developed an innovative approach to easily define data integration plans in Karma. It already addresses several data reshaping challenges, and we are now working to add the flexibility and richness required to rapidly apply data analysis and visualization tools to new datasets.

The most closely related work to this effort is the recent work on GoogleRefine[1] and DataWrangler (Kandel et al. 2011). Both systems provide good support for solving a

[1] http://code.google.com/p/google-refine/

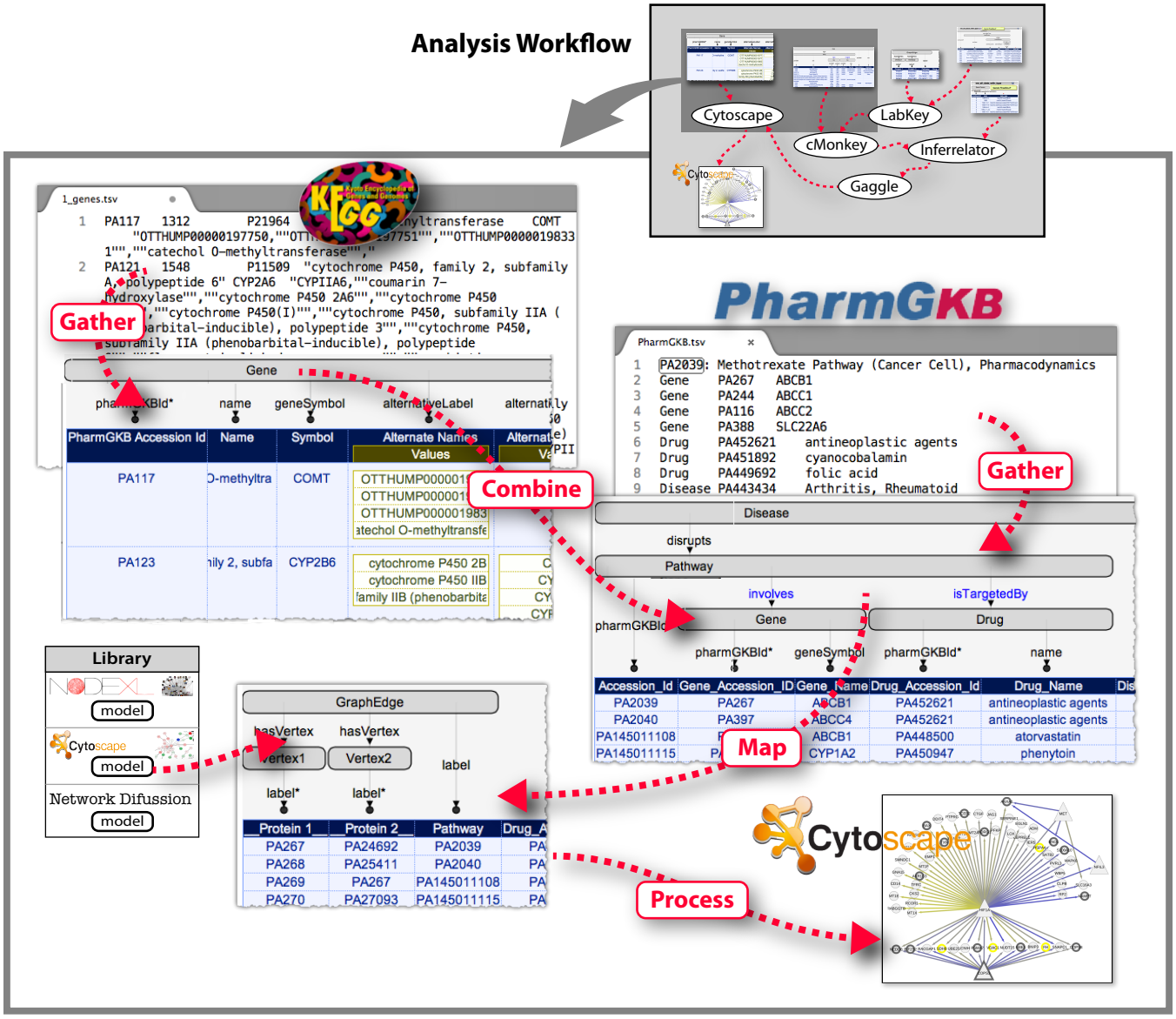

Figure 2: Example of how a user would create a data integration plan that transforms and restructures messy raw data to visualize it using a visualization tool called Cytoscape

small part of the data reshaping problem that we address in Karma. They support structured and semi-structured data, but lack support for hierarchical data (e.g., XML, JSON, RDF). Both are focused on single datasets, lacking support for integrating multiple datasets. Data Wrangler is a browser-based tool with no server component, so it is limited to datasets of a few thousand records. Google Refine allows users to define new operators using Javascript, making it more powerful, but harder to use. It has a server component so it can support large datasets, but has no support for streaming data or parallelization. There is also an extension to Google Refine that makes it possible to publish data in RDF with respect to a domain model (Maali, Cyganiak, and Peristeras 2012), but the mapping to the domain model is defined manually. Neither system has explicit models to capture the input/output requirements of other tools. They provide a set of useful operators, but without the semantic models users receive no guidance on what operations to use to reshape data to satisfy the requirements of the toolkit components they want to invoke.

Other related work includes the data warehouse tools (e.g., Informatica, IBM DataStage, Oracle Data Integrator, and Talend) that are designed to process very large datasets, but all of these tools require a centralized data repository and require significant manual effort to define the ETL rules to set up this repository (Inmon 2005). There is recent work to to scale these systems to petascale data warehouses, such as the recent system developed by Facebook, called Hive (Thusoo et al. 2010), but like other data warehouse tools, this tool still requires significant effort to define the integrated data warehouse.

Discussion

We believe the approach presented here will dramatically reduce the time to construct data reshaping plans, improve their reliability, and execute them at scale. This will enable developers to rapidly and correctly prepare data for analysis and visualization tools and link the output of one tool to the input of the next, all within the big data environment. This, in turn, will allow developers to focus on the analysis workflows, trying different tools, different parameters, etc. in order to optimize the analyses. Today, there are no tools that attempt to solve this problem. As described above, the closest partial solution to this challenge of transforming and restructuring data considers only static data, and requires either manual programming, tools such as Google Refine with only interactive cleaning, or relatively inflexible data warehouse products. None of these systems provide an end-to-end solution to the problem and for the parts of the problem that they do solve, they do not offer an easy-to-use approach to define data restructuring plans for processing very large or streaming datasets.

One area of future research is how to build the data reshaping plans without access to all of the data. For both very large datasets and streaming data, it is not feasible to consider all of the data to build the data reshaping plans. So the challenge is how to decide which portion of the data to examine and how to decide when the system has seen enough of the data to build correct and reliable data reshaping plans.

Acknowledgements

This research is based upon work supported in part by the National Science Foundation under award number IIS-1117913. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF or any person connected with them.

References

Inmon, W. H. 2005. *Building the data warehouse*. Wiley.

Kandel, S.; Paepcke, A.; Hellerstein, J.; and Heer, J. 2011. Wrangler: interactive visual specification of data transformation scripts. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, 3363–3372.

Knoblock, C. A.; Szekely, P.; Ambite, J. L.; ; Goel, A.; Gupta, S.; Lerman, K.; Muslea, M.; Taheriyan, M.; and Mallick, P. 2012. Semi-automatically mapping structured sources into the semantic web. In *Proceedings of the Extended Semantic Web Conference*.

Maali, F.; Cyganiak, R.; and Peristeras, V. 2012. A publishing pipeline for linked government data. In Simperl, E.; Cimiano, P.; Polleres, A.; Corcho, O.; and Presutti, V., eds., *The Semantic Web: Research and Applications*, volume 7295 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 778–792.

Taheriyan, M.; Knoblock, C. A.; Szekely, P.; and Ambite, J. L. 2012. Rapidly integrating services into the linked data cloud. In *Proceedings of the 11th International Semantic Web Conference (ISWC 2012)*.

Taheriyan, M.; Knoblock, C. A.; Szekely, P.; and Ambite, J. L. 2013. A graph-based approach to learn semantic descriptions of data sources. In *Proceedings of the 12th International Semantic Web Conference (ISWC 2013)*.

Thusoo, A.; Sarma, J. S.; Jain, N.; Shao, Z.; Chakka, P.; Zhang, N.; Antony, S.; Liu, H.; and Murthy, R. 2010. Hive - a petabyte scale data warehouse using hadoop. In *Proceedings of the 26th International Conference on Data Engineering (ICDE)*, 996–1005.

Tuchinda, R.; Knoblock, C. A.; and Szekely, P. 2011. Building mashups by demonstration. *ACM Transactions on the Web (TWEB)* 5(3).

Wu, B.; Szekely, P.; and Knoblock, C. A. 2012. Learning data transformation rules through examples: Preliminary results. In *Ninth International Workshop on Information Integration on the Web (IIWeb 2012)*.