Automatic Synthesis of Bioconductor Pipelines: A Domain Modeling Challenge

Anna-Lena Lamprecht[1] and Tiziana Margaria[1]

Lero - The Irish Software Research Centre, University of Limerick, Ireland
anna-lena.lamprecht@lero.ie, tiziana.margaria@lero.ie

Poster Abstract

GNU R is a widely used programming language and software environment for statistical data analysis and visualization. Bioconductor [1] is a collection of bioinformatics packages that extends R's standard range of functionality by comprehensive libraries of functions and meta-data predominantly for the analysis of data from high-throughput genomics and molecular biology experiments, and additionally provides several example data sets that are useful for testing, benchmarking and demonstration purposes. Reference manuals and additional manuscripts provided with the packages at the Bioconductor web site describe a variety of data analysis procedures based on the available functionality.

The described bioinformatics procedures or workflows are typically referred to as data analysis *pipelines*, as they typically have a simple, in fact mostly linear, structure. (This is largely due to the fact that the often considerable complexity of the individual analysis steps is encapsulated by a services with a simple interface, and hence hidden from the user at the provided level of abstraction.) Thus, automatic workflow composition functionality like the method described in [6], which is based on a linear-time logic synthesis algorithm, can be easily applied here to generate the complete analysis pipelines automatically.

Making use of the constraint-driven workflow composition functionality of the PROPHETS framework [5], which is based on such a linear-time logic synthesis algorithm, we created a prototype of a corresponding synthesis framework [3]. Focusing on DNA microarray analysis, this prototype comprises around 30 services and provides a framework for user-level construction of analysis pipelines based on appropriately wrapped and integrated Bioconductor functionality. It helps handling the variability that is inherent in microarray data analysis at an user-accessible level and emphasizes the agility of a model-driven and service-oriented approach to workflow design. PROPHETS is the current reference implementation of the loose programming paradigm [4], aiming to simplify workflow development in order to reach application experts without programming background. Working with PROPHETS consists of two major phases:

1. *Domain Modeling*, where PROPHETS is prepared for the application domain by describing services and their interfaces formally, defining service and type taxonomies (simple ontologies with *is-a)* relations only) and specifying domain-specific constraints, and

2. *Workflow Design*, where the actual workflow synthesis takes place in the fashion of an iterative "playing" with the specification, synthesis, and constraint and parameter refinement, until a satisfying solution is found.

Hence, for making new application domains accessible to workflow designers, adequate domain modeling is crucial. For the prototype application we set up the domain model manually. On the component level (i.e. when dealing with the individual services), we could largely reuse available information from the packages' documentation files. Fortunately, consistent terminology is used there for referring to data types and functions, although there is no formal ontology or taxonomy model behind that would explicitly define a controlled vocabulary. On the ontological level (i.e. when defining the service and type taxonomies of the domain model), however, the process was not so straightforward. The package descriptions did not contain any explicit relation annotations, except from the hierarchical information that can be derived from the package structure itself. Hence, we had to define abstract semantic categories for the hierarchical grouping of services and types ourselves.

In contrast to other case studies described in [3], we could furthermore not directly apply terminology from the EMBRACE Data and Methods Ontology (EDAM) [2] here. Although it contains several terms for the field of microarray data processing, they were mostly too generic for the annotation and type-safe composition of services that perform very specific operations like in our case study, so that they had to be refined to an adequate level of detail that matched the individual functionalities. We are now investigating if and how the gap between EDAM and the Bioconductor package documentation regarding domain modeling for synthesis applications can be bridged automatically based on information that is already available. Solving this challenge would enable the automatic synthesis of Bioconductor pipelines at a larger scale, with ultimately thousands of services available for automatic composition into analysis pipelines.

References

1. Gentleman, R.C., Carey, V.J., Bates, D.M., et al.: Bioconductor: open software development for computational biology and bioinformatics. Genome Biology 5(10), R80 (2004)
2. Ison, J., Kalaš, M., Jonassen, I., et al.: EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. Bioinformatics (2013)
3. Lamprecht, A.L.: User-Level Workflow Design - A Bioinformatics Perspective, Lecture Notes in Computer Science, vol. 8311. Springer (2013)
4. Lamprecht, A.L., Naujokat, S., Margaria, T., Steffen, B.: Synthesis-Based Loose Programming. In: 7th Int. Conference on the Quality of Information and Communications Technology (QUATIC 2010). pp. 262–267 (2010)
5. Naujokat, S., Lamprecht, A.L., Steffen, B.: Loose Programming with PROPHETS. In: 15th Int. Conference on Fundamental Approaches to Software Engineering (FASE 2012). LNCS, vol. 7212, pp. 94–98. Springer (2012)
6. Steffen, B., Margaria, T., Freitag, B.: Module Configuration by Minimal Model Construction. Tech. rep., Fakultät für Mathematik und Informatik, U. Passau (1993)