Proqueries and Praqueries

Gabriel Goren Roig[1,2], Joshua Meyers[3], Emilio Minichiello[3]*, and Ryan Wisnesky[3]

[1] Universidad de Buenos Aires
[2] CONICET
[3] Conexus AI, Inc.

Recent work in categorical database theory [Spi12, SW15, SSVW17, SW17] develops a new data model which we refer to as the *algebraic data model* because it is based on algebraic (i.e. Lawvere) theories, in contrast to both the traditional relational data model [AHV95] and to other approaches to categorical database theory, such as finite limit sketches [RW91] [JR02] or attributed C-sets [PLF22]. The algebraic data model builds on the insight that categories can be interpreted as database schemas, by using objects to represent tables and outgoing arrows to represent columns which refer to other tables (i.e. foreign keys). That is, a database instance on a schema $\mathcal{C}$ is defined as a copresheaf $\mathcal{C} \to \mathbf{Set}$, which populates each table with a set of rows, and given any other schema $\mathcal{D}$, functors $\mathcal{C} \to \mathcal{D}$ serve as the basis for defining data migration operations.

The approach just sketched is enticing, because "every theorem about small categories becomes a theorem about databases" [Spi12]. However it turns out to be too naive to formalize uses of databases beyond simplistic querying. In particular, in the context of data integration some values may be treated up to isomorphism, such as the labelled nulls generated during integration to fill in missing values, while values that appear in the input data, such as the names of people and their salaries, must be treated up to equality. In this sense, categories and copresheaves have too many automorphisms to be used in data integration. This *attribute problem* is solved by defining values to be expressions in an algebraic theory [SSVW17, SW17]. To this end, schemas are defined by equipping categories with an algebraic profunctor into a fixed algebraic theory, called the *typeside*, whose objects represent data types (e.g. Nat, String) and whose morphisms represent data operations (e.g. addition, append). The typeside adds the necessary rigidity for data integration by requiring that constants (such as "Alice"), but not variables (such as Alice's unknown salary n), be preserved by morphisms of instances. Algorithms follow by specialising definitions to finite presentations of schemas, instances, etc. and are implemented in the open-source CQL tool available at categoricaldata.net.

Current research about the algebraic data model has progressed from studying the data transformations associated with *functors* to studying the more structured data transformations associated with *profunctors*, which we call *proqueries*. Proqueries, which subsume traditional conjunctive queries, have been referred to as *bimodules* in [SSVW17], where the theory is developed in the form of a proarrow equipment. Proquery presentations, in turn, were developed in [SW17] under the name of *uberflowers*. Evaluating a proquery on a database instance according to its conjunctive query semantics produces another instance on a different schema, and this process is sufficient for querying. However, it is also possible to *coevaluate* a query, which constitutes the left adjoint to the evaluation functor just described. In fact, the evaluation-coevaluation adjunction is a particular case of the general geometric realisation/nerve adjunction, see [SSVW17, Remark 8.8]. Unlike evaluation, which cannot create labelled nulls in its output, coevaluation can, and thus allows us to perform data integration.

*Speaker.

We are currently studying the exact expressive power of query coevaluation compared to existing data integration techniques. Finally, the connection between proqueries and relational queries gives us a way to implement *query migration*: given an $\mathcal{A}$-shaped proquery $Q : \mathcal{A} \nrightarrow \mathcal{B}$ on schema $\mathcal{B}$, we can migrate Q along a proquery $F : \mathcal{B} \nrightarrow \mathcal{C}$ by composition, i.e. $F \circ Q$ is the desired migrated query. At the level of proquery presentations, composition is implemented by an operation which is called *view unfolding* in the relational literature [KP18]. Kan lifts and extensions of proqueries also have uses in data exchange operations and are under current research.

In this talk we will explain the algebraic data model and present our work on a new generalisation of proqueries which is analogous to *unions* of conjunctive queries. We call this generalisation *praqueries* since it is known that, when the typeside is trivial, they are equivalent to parametric right adjoint functors, also known as prafunctors. We give a composition law for praqueries, hence generalising the proarrow equipment structure in [SSVW17]. Importantly, we also develop the theory of praquery presentations, including an effective algorithm for composition of finite praquery presentations which generalises the view unfolding algorithm. We are able to obtain a correctness proof for this algorithm, which constitutes our main result and the capstone of our talk.

Along the way, we correct some technical errors in the literature, and we provide a proof of correctness for composition of proquery presentations, which to the best of our knowledge was not present in the literature. We expect to settle some further conjectures about praqueries, namely that praqueries are equivalently prafunctors between categories of instances which preserve type-algebras, and the issue of existence of right Kan lifts in the bicategory of praqueries.

References

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of databases*, volume 8. Addison-Wesley Reading, 1995.

[JR02] Michael Johnson and Robert Rosebrugh. Sketch data models, relational schema and data specifications. *Electronic Notes in Theoretical Computer Science*, 61:51–63, 2002.

[KP18] Yannis Katsis and Yannis Papakonstantinou. View-based data integration. In Ling Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*, pages 4452–4461. Springer New York, New York, NY, 2018.

[PLF22] Evan Patterson, Owen Lynch, and James Fairbanks. Categorical Data Structures for Technical Computing. *Compositionality*, 4:5, December 2022.

[RW91] Robert Rosebrugh and RJ Wood. Relational databases and indexed categories. In *Proceedings of the International Category Theory Meeting 1991, CMS Conference Proceedings*, volume 13, pages 391–407, 1991.

[Spi12] David I Spivak. Functorial data migration. *Information and Computation*, 217:31–51, 2012.

[SSVW17] Patrick Schultz, David Spivak, Christina Vasilakopoulou, and Ryan Wisnesky. Algebraic databases. *Theory and Applications of Categories*, 32(16):547–619, 2017.

[SW15] David I Spivak and Ryan Wisnesky. Relational foundations for functorial data migration. In *Proceedings of the 15th Symposium on Database Programming Languages*, pages 21–28, 2015.

[SW17] Patrick Schultz and Ryan Wisnesky. Algebraic data integration. *Journal of Functional Programming*, 27, 2017.