

Concept Instance Sketching and Design for a Biological Database Framework

Alan McCulloch¹
Paul Smale¹

Pieter Demmers²
Russell Smithies¹

Jason Mitchell¹
Craig Miskell¹

David Townley³
Anar Khan¹

Nauman Maqbool¹

¹ AgResearch Ltd

² Crop and Food Research/Nutrigenomics New Zealand

³ CSIRO/Sheep Genomics

Abstract

We have developed a biological resource description framework (BRDF) based on a collection of concept instance sketches expressed using an extended hypergraph notation, and have applied this framework to the data warehousing requirements of a number of research projects in the fields of genomics, genetics, nutrigenomics and molecular biology. The resulting framework and its instances have been implemented in a Postgres database, with a Python object oriented API and web-based interface, and a number of production systems have been built and operated using the framework, and continue to be developed. This paper defines and describes the framework and concept instance sketching technique we have developed, and briefly describes the implementation of the systems based on this framework and our experience with them to date.

1 Introduction

We needed to develop database solutions to support a number of multidisciplinary research programs. These programs were expected to generate diverse and dynamic datasets in both human and agricultural research projects, including gene expression work (such as microarray experiments running on various platforms; quantitative and real time PCR; EST sampling and clustering), proteomics results, metabolite measurement experiments, genotyping experiments, genomic sequencing, rich phenotype data, epidemiological case control diet studies and high throughput food fraction functional assay experiments.

The requirement was met by developing an extensible data framework. We developed an extended hypergraph based concept instance sketching technique to solve the difficult problem of bridging the gap between a large and poorly defined data modelling problem, and the tightly specified entity relationship models typically required by the software engineering design validation and build process.

2 Definition of a Data Framework

In this paper a “data framework” is defined as:

Copyright ©2007, Australian Computer Society, Inc. This paper appeared at the Twenty-Sixth International Conference on Conceptual Modeling - ER 2007 - Tutorials, Posters, Panels and Industrial Contributions, Auckland, New Zealand. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 83, John Grundy, Sven Hartmann, Alberto H. F. Laender, Leszek Maciaszek and John F. Roddick, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

1. A database schema, that we know in advance will be extended and modified over the project lifetime
2. A set of rules for deciding which changes to the schema are permitted according to the framework, and how they are to be implemented. These rules encompass, for example, table naming conventions, and a strict classification systems for tables, so that each addition to the schema must be classifiable into one of the categories supported by the framework

Thus a data framework in this context is an extensible database schema – but with definite limits on the types of extension permitted and how they are implemented.

3 Concept Instance Sketching to Identify Framework Components

As noted by Thalheim (2005), large and complex database schemas can often be seen to consist of “identifiable sub-schemas that are loosely coupled,” and “a component based design allows a developer to derive an understandable schema topography for very large and complex databases.” We used a hypergraph based diagramming technique to sketch instances of objects and the relationships between them, to help reveal common design patterns that can be used to identify and classify the sub-schema components that our framework needs to support. Concept instance sketches convey the structure of relationships *by example*, using pictures isomorphic to instances of objects and relations, rather than by making formal statements in a modelling language. Figure 1 sketches an instance of a regulatory relationship between biological sequences – this could not be a formal modelling statement since the sequence entity appears more than once.



Figure 1: A simple concept instance sketch, representing an instance of a regulatory relationship between biological sequences.

Using a graph consisting of pairs of vertices connected by edges, we are restricted to isomorphic pictures of at most binary relationship instances, e.g. see Figure 2(a). Hypergraphs were used so that ternary and higher relationship instances can be pictured. A hypergraph is a graph in which an *edge* may connect more than two vertices, e.g. see Figure 2(b).

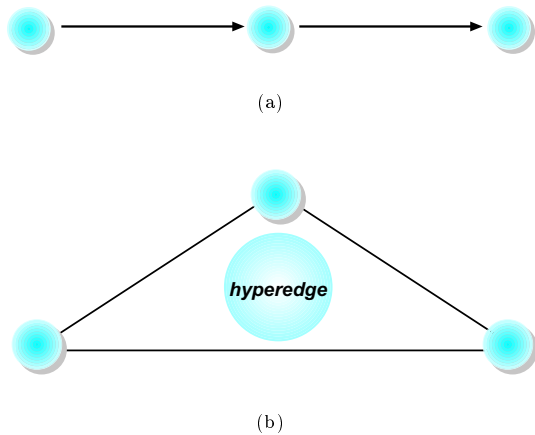


Figure 2: Hypergraph examples: (a) Three vertices connected by two (oriented, i.e. directed) edges in a standard graph. (b) Three vertices connected by a single (unoriented) hyperedge in a hypergraph. We emphasise the existence of the hyperedge by drawing a circle inside the connected vertices.

Furthermore, we needed to depict instances of the typical data-warehouse star-schema design pattern (Útley 2005), with a number of fact tables linked to a given entity. These are treated as unary relations, hence depicted by a dangling edge, as shown in Figure 3.

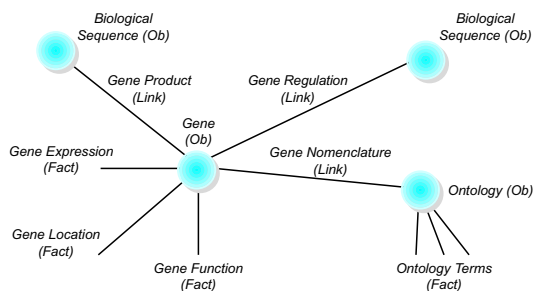


Figure 3: Extended hypergraph notation, including dangling edges for fact tables.

A further point to note about our sketches is that a hyperedge may also appear in another context as a vertex. This is because most relation instances in the database are also objects in their own right and can appear as terms in some other relationship. We developed around 25 concept instance sketches of parts of the model.

4 Results

The results of concept instance sketching helped to classify the types of table to be supported by the framework by revealing common design patterns across different data domains, as listed in Table 1. Any new tables added to the schema must be assignable to one of the resulting framework table classes, and table naming is determined by class. These framework rules help control and limit the complexity of the schema as it extends. The schema currently consists of some 85 tables.

5 Concept instance example – gene expression study and observations

Figure 4(a) shows a gene expression study instance sketched as a hyperedge connecting one or more samples used in the experiment, the list of lab resources used, and a protocol. In Figure 4(b), gene expression studies appear in another context, as a vertex connected via observation links with microarray spots, where an expression observation is a relation between a study and a spot. Each distinct hyperedge, including dangling edges, is realised as a table in the schema. Thus the diagram below would be realised as 13 database tables. (Note that multiple instances only require a single table.)

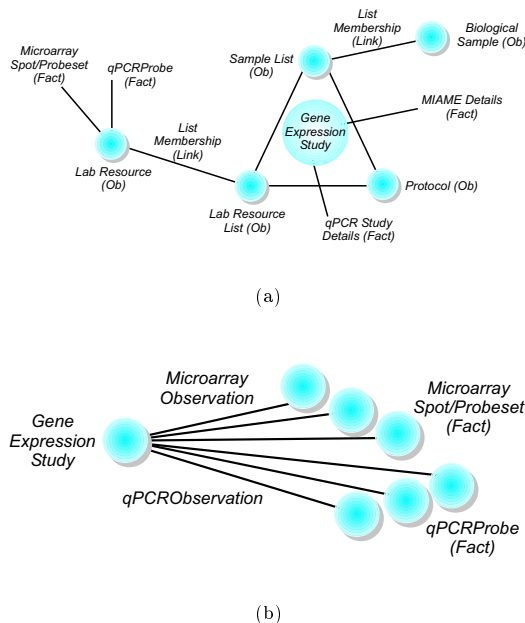


Figure 4: An advanced concept instance sketch example: (a) An instance of a gene expression study, represented by a hypergraph. (b) An instance of a gene expression study in another guise, as a vertex in a unary relation with two fact tables which record qPCR and microarray spot fact dimensions.

The BRDF framework models other types of experiment, including *in silico* experiments such as BLAST searches, using the same study, observation and fact table design patterns.

6 Summary and Future Work

We have developed a simple concept instance sketching technique which we have found very useful in solving a difficult data modelling problem. The framework has enabled us to quickly roll out warehouses to a number of projects, and to extend the schema for individual projects as required, while maintaining control of complexity. We are currently running five production instances of this framework for five distinct science programs.

However we acknowledge that this technique is incomplete and informal. Because our technique is not a modelling language, the opportunity for formalism is probably limited. Nevertheless one line of future work would be to define the technique more formally than we have done. Furthermore we have already encountered areas where the technique needs extending – for example some vertices of a hyperedge may be

Table 1: Classification of database tables, derived from concept instance sketching.

Class	Description	Count
Fact tables	One data dimension per fact table. Additional fact tables added as needed	29
Link tables	Associate 2 entities (examples: predicate link to record is-a and other subject-object relations; list membership link to record membership of lists)	16
Study tables	Examples include gene expression study, genotype study, <i>in silico</i> study (e.g. BLAST run). Ternary or higher master relation between a protocol, and usually a sequence or sample or biological database, and a lab resource list.	5
Observation tables	An observation is a relation between a study and the experimental units observed. Gene expression observation (a study connected to a spot or probe set); genotype observation (a study connected to a genetic test); <i>in silico</i> observation (a study connected to a BLAST hit sequence)	6
Function tables	Associate 3 or more entities. For example, sequencing function associates a sequence, a sample and a list of lab resources	5
Ob tables	Basic entities in the database – biological subjects, samples, sequences, protocols etc.	20
Framework	Tables to support the API and runtime, including a data dictionary	4

optional, but we have not attempted to develop any diagramming convention to indicate this.

One of our main focuses for future development is in development of the database browser user interface, mainly in the direction of providing single integrated views of data and images; and in developing additional data extract reports and extending current reports.

Acknowledgements

This research was supported by SheepGenomics which is an initiative of Australian Wool Innovation Limited and Meat & Livestock Australia together with 11 leading research organisations in Australia and New Zealand. This is SheepGenomics publication number 85. We also acknowledge the support of Nutrigenomics New Zealand.

References

- Thalheim, B. (2005), 'Component development and construction for database design', *Data and Knowledge Engineering* **54**, 77–95.
- Utley, C. (2005), 'Designing the Star Schema Database', <http://www.ciobriefings.com/whitepapers/StarSchema.asp>.