HyperGraphDB: A Generalized Graph Database

Borislav Iordanov

Kobrix Software, Inc.
`http://www.kobrix.com`

Abstract. We present HyperGraphDB, a novel graph database based on generalized hypergraphs where hyperedges can contain other hyperedges. This generalization automatically reifies every entity expressed in the database thus removing many of the usual difficulties in dealing with higher-order relationships. An open two-layered architecture of the data organization yields a highly customizable system where specific domain representations can be optimized while remaining within a uniform conceptual framework. HyperGraphDB is an embedded, transactional database designed as a universal data model for highly complex, large scale knowledge representation applications such as found in artificial intelligence, bioinformatics and natural language processing.

Keywords: hypergraph, database, knowledge representation, semantic web, distributed.

1 Introduction

While never reaching widespread industry acceptance, there has been an extensive body of work on graph databases, much of it in the 90s. Various data models were proposed, frequently coupled with a complex object representation as a natural, practical application of graph storage. More recently, several developments have contributed to a renewed interest in graph databases: large-scale networks research, social networks, bioinformatics as well as the semantic web and related standards. Part of that interest is due to the massive amounts of graph-oriented data (e.g. social networks) and part of it to the inherent complexity of the information that needs to be represented (semantics and knowledge management). This body of work has been thoroughly reviewed in [1]. In this paper, we present the implementation of a generalized hypergraph model independently proposed by Harold Boley [4] and Ben Goertzel [5]. The model allows for n-ary hyperedges that can also point to other edges, and we add an extensible type tower [6] to the mix. Two generalizations of graphs related to our work have been the Hypernode model [2] and the GROOVY model [3], both focused specifically on representing objects and object schemas. While hypergraphs have been extensively used as an analytical tool in database research, we are not aware of any other implementation of general hypergraphs as a native database.

The main contributions of HyperGraphDB[1] lies in the power of its structure-rich, reflexive data model, the dynamic schema enforced by an extensible type system, and open storage architecture allowing domain specific optimizations. It must be noted however that the representational flexibility also leads to performance gains when fully exploited. While a hypergraph can be represented as a regular graph, frequently the opposite is also true in a non-trivial way. Many graphs will have repetitive structural patterns stemming from the restrictions of the classical graph model. Such patterns can be abstracted via a hypergraph resentation, leading to much fewer nodes and database operations. For example, flow graphs where edges represent multi-way input-output connections can be stored much more compactly using a hypergraph-based model.

Furthermore, reducing the complexity of a representation is not the only benefit of a hypergraph model. As illustrated in the context of cellular networks, "transformation to a graph representation is usually possible but may imply a loss of information that can lead to wrong interpretations afterward" ([7]). In fact, biological networks are replete with multilateral relationships that find a direct expression as hypergraphs. Another example of how the ability to represent higher-order relations can improve algorithms can be found in [8], where the authors present a learning algorithm for Markov Logic Networks based on what they call "hypergraph lifting" which amounts to working on higher-order relations.

One common criticism of RDF ([9]) stores is the limited expressiveness of binary predicates, a problem solved by HyperGraphDB's n-ary relationships. Two other prominent issues are contextuality (scoping) and reification. A popular solution of the scoping problem has been proposed in the form of *Named Graphs* ([11]). Reification is represented through a standardized reification vocabulary, by transforming an RDF graph into a reified form ([10]). In this transformation a single triplet yields 4 triplets, which is unnatural, breaks algorithms relying on the original representation, and suffers from both time and space inefficiencies. A similar example comes from the Topic Maps standard [12] where reification must be explicitly added as well, albeit without the need to modify the original representation. Those and other considerations from semantic web research disappear or find natural solutions in the model implemented by HyperGraphDB.

The paper is organized as follows: first, we review some variations of hypergraphs and describe the particular one adopted by HyperGraphDB. In subsequent sections, we detail the system's architecture: storage model, typing, indexing, querying and its P2P distribution framework. Lastly, we describe in some detail one particular application in natural language processing.

2 Hypergraphs and the HyperGraphDB Model

The standard mathematical definition of a hypergraph is a family of sets over a universal set of vertices V, or equivalently an undirected graph where an edge

[1] The system is open-source software, freely available at `http://code.google.com/p/hypergraphdb`.While the current implementation is in the Java programming language, we note that the architecture is host-language-agnostic and we make very few references to Java constructs in the exposition below.