`armstrong-hypergraph` Design Decisions

Henriette Harmse

HenrietteHarmse.com

Abstract. This document details the design decisions made in implementing the `armstrong-hypergraph` library for use in computing Armstrong relations/ABoxes for schemas/TBoxes with functional dependencies.

1 Purpose

The intended purpose of `armstrong-hypergraph` is to be usable by the implementation of the Armstrong ABox plugin for Protégé. The specific functionality required is to calculate the minimal transversals (minimal hitting sets) of a hypergraph.

2 Tool Decisions

2.1 Java

Java has been chosen as the implementation language of `armstrong-hypergraph` due to Protégé 5.2 and related libraries like the OWL API being written in Java. Version 8 of Java will be used since both Protégé and the `owlapi-distribution-4.2.8.jar` (used in Protégé) supports Java 8.

2.2 Graph/Hypergraph Tool Considerations

Frankly I would prefer to use an existing library rather than write it myself. An existing well-used library is likely to be more robust than anything I may write merely because they have resolved design issues I still need to discover. However, even though there are a number of Java libraries that implement graph algorithms, these are all limited to graphs having edges consisting of two vertices at most.

The following libraries/tools were considered for implementing hypergraphs:

JUNG The underlying graph implementation is based on that of the Google Guava library, which does not support hypergraphs.

Hypergraph This applet seems to be focussed at visualization of hypergraphs rather than providing an API.

JGraphT Does not support hypergraphs.

Guava Does not support hypergraphs.

It is possible that Guava or JUNG in future may support hypergraphs, in which case it may be sensible to discontiunue this library in favour of mainstream libraries.

3 Implemetation Details

3.1 Minimal Hypergraph Transversal/Minimal Hitting Set

In this document we will refer to minimal hitting sets (MHS). For an initial implementation of MHS the algorithm of Berge has been used [1–3].

4 To Do

1. Based on the findings of Gainer-Dewar et al. [2] the MMCS algorithm may provide beter performance than the Berge algorithm.
2. Currently `armstrong-hypergraph` is hard-coded to use `java.util.HashSet` as a set implementation. With Java 8 it possible to refactor the code such that the set implementation is generic and to instantiate the generic set without using reflection. See Generic instantiation trick in Java 8.

References

1. C. Berge, *Hypergraphs: Combinatorics of Finite Sets*, North-Holland, 1989.
2. Andrew Gainer-Dewar and Paola Vera-Licona, *The Minimal Hitting Set Generation Problem: Algorithms and Computation*, January 2016.
3. Matthias Hagen, *Lower Bounds for Three Algorithms for Transversal Hypergraph Generation.*, Discrete Applied Mathematics **157** (2009), no. 7, 1460–1469.