The AFGen Argumentation Benchmark Generator

Billy D. Spelchan[1] and Gao Yong[1]

University of British Columbia Okanagan, Kelowna, British Columbia, Canada
bspelch@mail.ubc.ca yong.gao@ubc.ca

Abstract

We developed a benchmark generator which provides a more flexible method of generating random argumentation instances with a variety of algorithmic and probability behavior. The design of our generator is based on the random argumentation model proposed in [2] where the probabilistic behavior of the model is analyzed analytically and the hardness of argumentation instances from the model is studied empirically.

1 Introduction

Over the last few years the study of Argumentation Frameworks has been growing. For the development of solvers, it is vital that there be good benchmarks to use when testing the solvers. Regarding the generation of random argumentation benchmarks, the traditional random model of directed graph $D(n, q)$ appears to be a natural choice, where n is the number of vertices and p the edge probability. However, due to a result by Vega [1], the existence of an extension in such a random argumentation framework is almost always guaranteed, making it less useful for testing exact solvers.

In [2] the D(n,p,q) model was proposed with two parameters that control the likelihood of an attack between two vertices being a mutual attack. Due to its theoretically-guaranteed threshold behavior in solution probability and the empirically-observed easy-hard-easy patterns of random argumentation instances from the model, we believe a generator based the model makes a good tool for benchmarking argumentation solvers.

2 The D(n,p,q) Generator

The model D(n, p, q) is a generalization of the traditional random digraph model discussed by Vega [1]. It has three parameters: n being the number of vertices to generate, p being the frequency of attack between the $\binom{n}{2}$ pairs of vertices, and q being the probability of the attack being mutual.

The following result on the phase transition of solution probability is proved in [2] and give us a clear way to set the model parameters to control the probabilistic behavior of random instances generated. See [2] for a detailed account on the difficulties in using Vega's model as a benchmark generator.

The following theorem from [2] characterizes the exact threshold of the phase transition of the D(n,p,m) model to have a non-empty preferred extension. The results also hold for the existence of a stable extension.

Theorem 1. *For any constants* $0 < p < 1$ *and* $0 < q < 1$,

$$\lim_{n\to\infty} \mathbf{P}(D(n,p,q) \text{ has a preferred extension}) = \begin{cases} 1, & \text{if } q > q^* \\ 0, & \text{if } q < q^* \end{cases}$$

where $q^* = q^*(p)$ *is the unique value of* q *between 0 and 1 such that* $\frac{4q}{(1+q)^2} = p$.

3 Benchmark Software

Three small applications are provided. There is a GUI version of the generator, a CLI version of the generator, and a CLI utility for calculating the q* value for a provided p. In addition to the D(n,p,q) generator, the tools have options to generate a modified file by providing two additional parameters: s being the likely hood of a new attack being added, and r being chance of an existing attack being removed. This will allow the generator to also be used within the dynamic track while providing much more flexibility over the difficult and properties of the generated modified benchmark.

The zip file containing the tools, sample benchmarks, and source code is located at

`https://drive.google.com/file/d/1V_CdFlKpzHcDWAQ-Zki1mGm_I07Dz3-H/view?usp=sharing`

The tools were written in Java and packaged as self-executable jar files so are ready to run. Java 8+ is required to run the jar files, which have been tested with both Oracle's Java distribution and the Open JVM distribution. For the GUI version, JavaFX is required which is included in Oracle's distribution but is a separate download for Open JVM.

Further instructions on the usage of the tools is provided in the zip file within the documentation folder.

4 Future Work

After the 2019 competition, we will take a closer look at the dynamic track results and produce a benchmark generator that is better at handling the needs of generating those benchmarks. Related to this project is also our own solver which we hope will be in a future competition.

References

[1] Wenceslas Fernandez de la Vega, "Kernels in random graphs", Discrete Mathematics 82 pp. 213–217, 1990.

[2] Yong Gao, "A Random Model for Argumentation Framework: Phase Transitions, Empirical Hardness, and Heuristics", Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017.