A Roadmap Towards Tuneable Random Ontology Generation

Pietro Galliani

Free University of Bozen-Bolzano, Italy

SCORE 1:0 Bremen

What is this about?

- Generation of **random ontologies** (given feature choices)
- For **testing and benchmarking purposes**
- Using a **theoretically justifiable** model.

What is this about?

- Generation of **random ontologies** (given feature choices)
- For **testing and benchmarking purposes**
- Using a **theoretically justifiable** model.

This is a roadmap: no results, just thoughts.

What is this about?

More and more complex algorithms over ontologies (merging, resolving inconsistencies, evaluating, answering queries, . . .)

Q: How to test/evaluate them?

What is this about?

More and more complex algorithms over ontologies (merging, resolving inconsistencies, evaluating, answering queries, . . .)

Q: How to test/evaluate them?
A: Ontology repositories!

What is this about?

More and more complex algorithms over ontologies (merging, resolving inconsistencies, evaluating, answering queries, . . .)

Q: How to test/evaluate them?
A: ~~Ontology repositories!~~

BAD IDEA!

Problem 1: Not Enough Ontologies

Ontology repositories are big (OntoHub: 21944 ontologies). Surely this is enough?

Not really: tools do not work on arbitrary ontologies, but on ontologies with specific properties (language, size, semantic features, ...).

Also, when using ML techniques over ontologies you often need lots of data (training/testing/validation, ...)

Problem 2: Statistically Dependent Features

Suppose that you want to test the effect of **ontology size** over the performance of your tool.

Obvious Approach: sample random ontologies of different dimensions from OntoHub, run your program over them, write results

Problem 2: Statistically Dependent Features

Suppose that you want to test the effect of **ontology size** over the performance of your tool.

Obvious Approach: ~~sample random ontologies of different dimensions from OntoHub, run your program over them, write results~~

Obvious Problem: What if operator depth (or some other feature) depends on size among your samples?

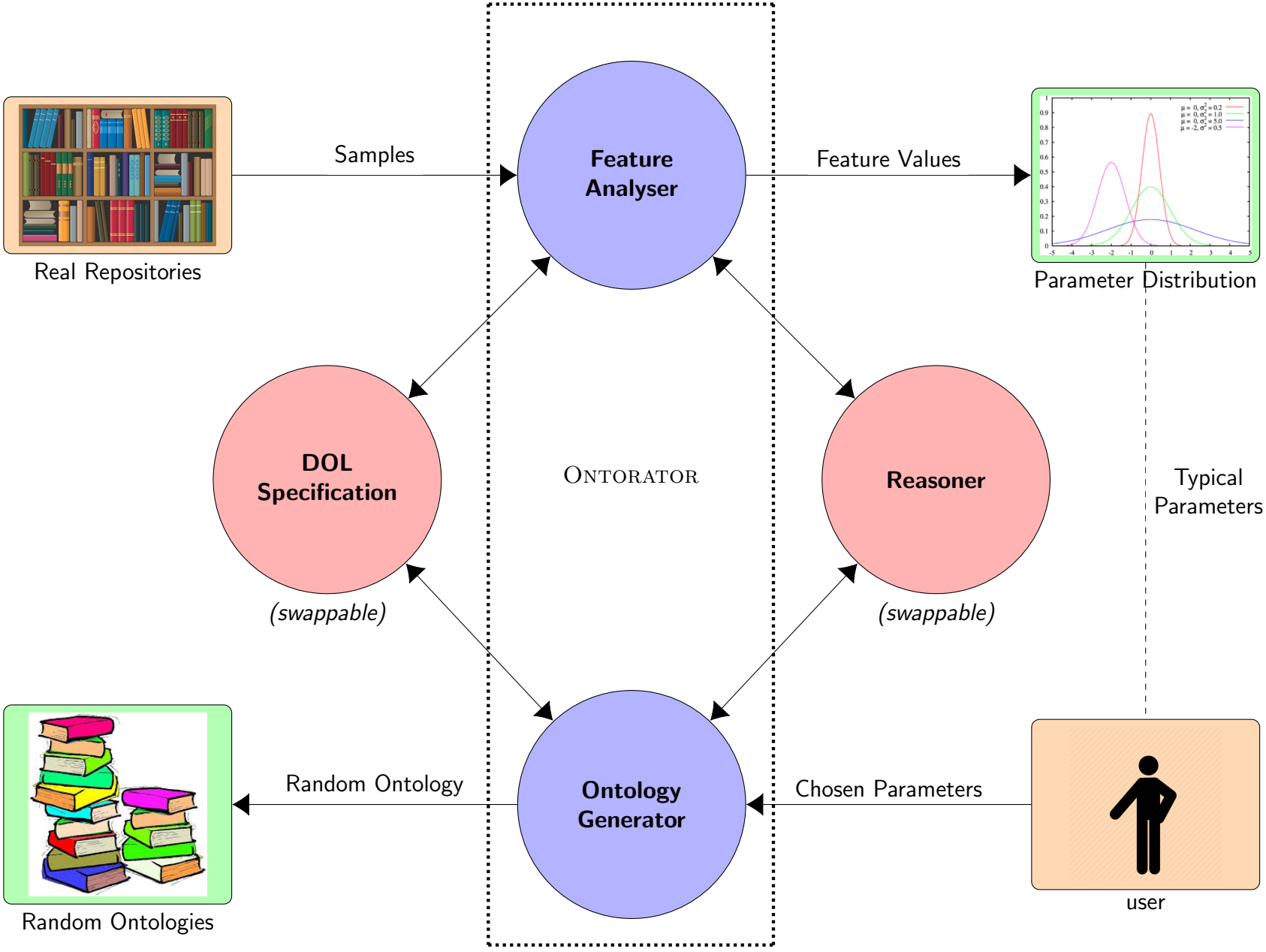
Problem 3: Weird Ontologies Needed

Some tools (e.g. ontology repair, evaluation, . . .) need to run on "bad" ontologies.

However, such ontologies are not generally published in repositories (but see BOG @ JOWO 2018 @ FOIS 2018).

Induce badness (e.g., adding arbitrary axioms)? Sure, but no real control about features / realism.

What We Propose



Main Points

- Identification of **features** from real ontology repositories via **network analysis** and semi-automated methods (no arbitrary choices);
- **Typical Parameters** (and parameter distributions) provided to user, who is however **free** to change them;
- **Language-Agnostic:** reasoner and DOL language specifications are external modules, can be changed.
- **Generative Probabilistic Model** for generating random ontology via MCMC-like sampling.

The Competition: LUBM and UOBM

Guo, Y., Pan, Z., and Heflin, J. (2005). LUBM: a benchmark for OWL knowledge base systems.; Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y., and Liu, S. (2006). Towards a complete OWL ontology benchmark.

- Fixed languages (OWL-Lite or OWL DL);
- Mostly fixed TBox Structure;
- Not very tuneable (can change some size parameters, number of individuals, little else)

The Competition: OTAGen

Ongenae, F., Verstichel, S., De Turck, F., Dhaene, T., Dhoedt, B., and Demeester, P. (2008). OTAGen: A tunable ontology generator for benchmarking ontology-based agent collaboration.

- Truly random ontology generation (not variations on a fixed one);
- Fairly tuneable;
- Generation algorithm $\rightarrow$ choice of available parameters, not vice versa;
- No attempt at statistical verisimilitude (ontologies generated are not "typical").

The Competition: Mips Benchmark

Zhang, Y., Ouyang, D., and Ye, Y. (2015). An automatic way of generating incoherent terminologies with parameters.

- Generates **inconsistent ontologies**;
- Tuneable, tests shows parameters are relevant for complexity analysis;
- Special purpose, no statistical verisimilitude.

Conclusion

There is a need for a

1. Tuneable;
2. General-Purpose;
3. Language-agnostic

generator of random ontologies suitable for the testing and benchmarking of algorithms and tools, based on **generative probabilistic models**.

Aside from practical utility, such a tool (and its model) would be a step towards the integration of statistical and symbolic reasoning.