OntoContest

By: Victor Gueorguiev (GRGVIC001) and Andre Lopes (LPSAND004)

# Outline

In the field of computer science, if an individual wanted to find a solution to a problem, they would have to either apply their existing knowledge of algorithms to the problem, or search the internet until they find something that might satisfy what they’re looking for. According to experts and online databases [Magdeburg, U. (2015)], there is currently no means to search for a problem and find a list of solutions, in the form of algorithms.

This problem can very easily be solved with an ontology. However, due to the limitless scope of algorithms, our project, which we’ve named OntoContest, will comprise of problems and solutions of those normally found on programming contest websites, such as UAV Judge Online [de Valladolid, U. (2017)]. We will also be adopting this problem domain for our ontology model, considering the wide applicability of such an ontology.

Once completed, OntoContest will be able to provide computer scientists with an easy way to reason about solutions to a problem, and allow them to better implement said solutions in their own work. It will also allow users to infer properties of said solutions, and in the setting of programming contests will allow users to reason about problems based on how they are used in contests and solved in these contests.

# Related Ontologies

The KISAO (Kinetic Simulation Algorithm Ontology) was constructed to classify simulation algorithms [Zhukova, Anna et al. (2011)]. The purpose of this ontology is to identify the type of algorithm used, as opposed to identifying which one to use for a specific problem, which is what OntoContest will focus on. Our ontology in comparison will classify a larger scope of algorithms, and be more akin to a top-level domain ontology.

# General Implementation

To implement the ontology, use of the following programs were made:

* Protégé: construct and test the ontology.
* Github: Manage versioning and collaborative work.
* HermiT: The default reasoner in Protégé.
* OntoGraf: Visualization of ontology

# The development process

## Methodologies used

The development of OntoContest followed and iterative and collaborative approach. Initial models would be rapidly prototyped, and replaced once a discussion about the impacts of significant changes were held. The development cycles made use of Github to manage merging and version control, but also to act as a central base on which to store the ontology (used for backups). Our repository can be found at: <https://github.com/BrutishGuy/OntoContest>

## Specific workloads

While ideas and changes were discussed, work was still divided between each member:

While Andre focused on the report, instance examples, and research, Victor focused on the logic behind the ontology. Both members additionally contributed to annotations and bugfixes in the ontology.

## Influencing factors

The following factors influenced the OntoContest project:

|  |  |
| --- | --- |
| Factor | Influence |
| According to experts in the field, there are no formally defined definitions for an algorithms knowledge base. | This was the core principle behind the development of OntoContest. It is a unique topic and worthwhile of investigation and contribution. |
| There is a conceptually endless scope to the level of detail in algorithms. | The KISAO project mentioned in the [related ontologies](#_Related_Ontologies) section is a very detailed project, with the limited scope of simulation algorithms. OntoContext has a much broader scope, and in order to limit workload, excessive details have either been omitted or abstracted to terms like “easy, medium or hard”. |
| Existing programming contest websites do not have an ontology implementation. | This was another motive to choose the topic, since there was a readily available domain that could take advantage of OntoContest. |

## Implementation

The project aims to achieve its goals by categorizing algorithms (much like the KISAO project) and identifying traits of those categories (such as complexity, difficulty and language). OntoContest will additionally link these algorithms to solutions for problems (which is what makes it different from the KISAO project), and through this, it will allow inferences to be made on future problems and solutions that are given to it. A full structure of OntoContest can be found in figure 1.

# Limitations of the Ontology framework

The project incurred the following limitations:

* Online programing competition websites generally have a system to display public statistics that depict the percentage of correct answers on a specific problem. An example of this can be found in figure 2. Since OntoContest was supposed to replace these systems, it needed to incorporate this feature, however, to implement this in Protégé would require it to frequently update certain axioms, which is normally done manually, and thus proves to be infeasible in the real world.
* Unfortunately, a system to detect which solutions were faster and slower could not be done, as only one object property assertion could be evaluated as true. i.e. It was not possible to make sure that an instance of an algorithm was both “Slower\_Than”, and “Faster\_Than” a particular algorithm at the same time.

# Appendix:

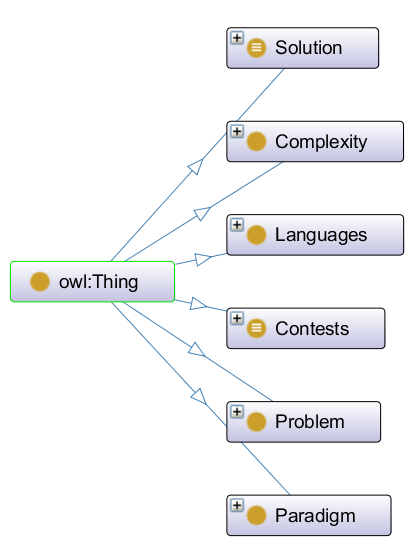


Figure 1: The general structure of the OntoContest Ontology

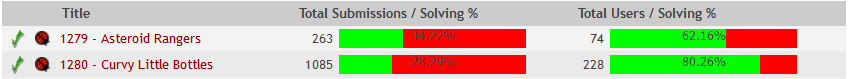


Figure 2: Real-time statistics on correct submissions

# References

[1] de Valladolid, U. (2017). UVa online judge. Retrieved from <https://uva.onlinejudge.org/>

[2] Magdeburg, U., & Bremen, U. (2015). Ontohub beta. Retrieved from <https://ontohub.org/>

[3] Zhukova, A., Waltemath, D., Juty, N., Laibe, C., & Le Novère, N. (2011). Kinetic simulation algorithm ontology.