Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Prolog Remote Constraint Logic Programming

Specifications

V.0.1.0

School of Engineering and Architecture of Fribourg
Department of Computer Science
4 March 2024

Author
Noah Godel

Supervisor
Frédéric Bapst

Document version

N° revision	Date	Description
0.1.0	4 March 2024	Initial prototype

Contents

1 Context

The purpose of this project is to develop an innovative solution that facilitates the integration of Operations Research (OR) libraries, such as Google OR Tools and Gecode, as a backend for Constraint Logic Programming (CLP) in Prolog. The primary goal is to create a web API that allows multiple users to access and utilize these OR libraries simultaneously. To achieve scalability and resource optimization, the web API will be deployed on a Kubernetes cluster.

In a previous attempt to bridge the gap between Prolog and the Google OR Tools library, a project was initiated. However, this approach required the local installation of the library, presenting limitations in terms of accessibility. To overcome these challenges, our current initiative focuses on developing a web-based solution that eliminates the need for local installations, giving us an easy-to-use library for CLP in Prolog. This approach not only enhances accessibility but in addition promotes efficient utilization of resources, making it an ideal choice for collaborative and concurrent CLP programming tasks. The proposed architecture leverages the power of Kubernetes to provide a scalable and resilient environment for hosting the web API, making it well-suited for deployment in various operational scenarios.
For the context of this project we will focus on the implementation of an API for the Google OR Tools library. In the future, the same approach could be extended to other OR libraries, such as Gecode.

2 Goals

The primary goals of this project are the following:

- Implement a web API that provides access to the Google OR Tools library for Constraint Logic Programming in Prolog. The API should be designed to handle multiple concurrent requests and provide some scalability in case of increased usage in the future. An authentication mechanism should be implemented to ensure secure access to the API.
- Deploy the web API on a Kubernetes cluster to ensure scalability and resilience. The deployment should be automated with the use of Gitlab CI/CD pipelines.
- Develop a client library for Prolog that allows easy access to the web API. The client library should be usable in a similar way to other CLP libraries in Prolog, such as clpfd.
- Perform a series of tests to ensure the reliability and performance of the web API. The tests should include unit tests and performance tests.

If the planned tasks are completed faster than expected, the addition of a second OR library, such as Gecode, to the web API could be considered. Alternatively the use of more features of the Google OR Tools library could be implemented.

3 Tasks

To achieve the above-mentioned goals the following tasks need to be carried out:

- Analysis of possible architecture for the web API and the client library.
- Implementation of basic endpoints for the web API to have a basic working prototype.
- Implementation of the client library for Prolog.
- Implementation of the remaining endpoints for the web API.
- Implementation of the authentication mechanism for the web API.
- Deployment of the web API on a Kubernetes cluster and Gitlab CI/CD pipeline setup.
- Performance testing of the web API.

4 Planning

See next page.