HDDL Parser:
A Realtime Hierarchical Planning Language Validation Toolkit

Mohammad Yousefi, Pascal Bercher

School of Computing, The Australian National University, Canberra, Australia
mohammad.yousefi@anu.edu.au, pascal.bercher@anu.edu.au

Abstract

We present HDDL Parser, an open-source language server providing real-time validation for the Hierarchical Planning Domain Definition Language (HDDL). The toolkit implements the well-known Language Server Protocol (LSP), enabling integration into any IDE, with a provided VS Code client demonstrating seamless real-time error detection and correction feedback. The language server performs a comprehensive analysis including: syntax validation, parameter inconsistencies, undefined entities, duplicate definitions, cyclic hierarchies, contradictory formulae, and type checking. Implemented in Rust, the correctness of the HDDL Parser has been validated against all 33 domains from the hierarchical track of the IPC 2023, and even detected critical errors in one of those domains. By providing automated quality assurance directly within the development environment, this tool significantly reduces debugging time and improves model reliability for hierarchical planning applications.

Code — https://github.com/koala-planner/HDDL-Parser
Demo — https://youtu.be/hRZ21HmcEQU

Introduction

Domain modeling often involves iterative refinement and debugging, where syntax errors, type mismatches, and semantic inconsistencies can significantly slow development. Traditional validation approaches require manual compilation and error checking, creating a disconnect between modeling activities and error detection. This process typically involves passing objects in different formats between various programs, each with specific requirements. We present a comprehensive open-source toolkit, HDDL Parser, that addresses these challenges through real-time validation, seamless IDE integration, and standardized serialization for the Hierarchical Planning Domain Definition Language (HDDL) (Höller et al. 2020). Our toolkit provides both a command-line interface for standalone validation and a Language Server Protocol (LSP) implementation that enables real-time checking within modern IDEs as the domain is being developed. By eliminating the need for manual compilation cycles, developers can identify and resolve errors immediately as they occur when modeling a problem. A screenshot of how this works in practice is provided in Figure 1. Once correctness is verified, the software offers the abstract syntax tree in JSON format that enables planners and other validation mechanisms, in any programming language, to access the parsed tree rather than the raw strings, which further saves time in development. The VSCode client is publicly available in the extension marketplace as "HDDL Parser".

Features

HDDL Parser is designed with performance, usability, and reliability at its core. The rest of this section highlights the key features of this software.

Fast and Reliable

HDDL Parser aims to serve as the backbone of model validation in hierarchical planning, making efficiency and reliability critical to our design decisions.

- **Rust-powered engine**: Built with Rust's memory safety guarantees, our tool can operate reliably over long periods while maintaining minimal CPU and RAM usage.
- **Asynchronous Server**: The language server utilizes an event-driven architecture with high parallelization capabilities. The asynchronicity allows the clients (e.g., VSCode) to continue their operation while the server prepares the result.
- **Thoroughly Tested**: We have tested the command-line version of this software on all hierarchical domains of the IPC 2023 (Alford, Behnke, and Schreiber 2024), and a comprehensive error detection benchmark for hierarchical planning (Sleath and Bercher 2023). In both cases, we have found critical errors that were not previously known. In conjunction with this, we also have 105 unit tests to ensure that each part is working as expected.

Easy to Use

By complying with industry standards, we make sure HDDL Parser can be easily used and integrated into any development environment.

- **Seamless Integration for Users**: Our extension for VSCode, once installed, is automatically triggered when a related file is opened or changed.

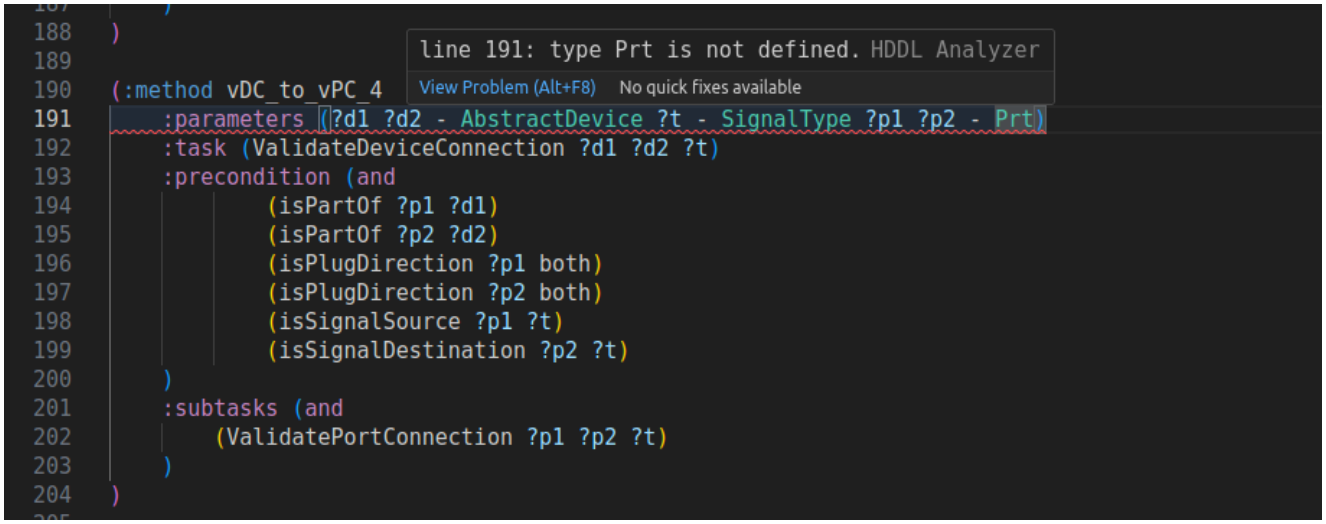

Figure 1: A screenshot of a domain from the IPC2023 benchmarks where we purposefully changed a parameter type from "Port" to "Prt". The VSCode client notifies the server about this change, and the server instantly instructs the client to highlight line 191 as erroneous with a clear error message.

- **Extensibility for Developers**: Based on the industry standard communication protocol, LSP, developers can easily integrate this server into IDEs without going into the internals of our tool.

Real-Time Validation

At the heart of the LSP server is the ability to detect errors as they occur, and reduce development and debugging efforts.

- **Character-by-Character Analysis**: Errors are detected and reported instantly as a user modifies a single character of the planning files.
- **Comprehensive Error Diagnostics**: A wide range of errors–from basic syntax issues to missing primitive refinements for compound tasks–are detected, and appropriate handling approaches to fix them are provided.
- **Immediate Error Visualization**: Syntax and semantic errors are visually marked in the editor as they occur.
- **Syntax Highlighting**: The extension provides personalizable syntax highlighting for HDDL files (e.g., keywords, predicates, etc.).

Standardized Serializer

Our tool makes communication between different parts of the ecosystem easier.

- **Higher Level Abstraction**: Once the validation process is completed, the abstract syntax tree can be exported, allowing other software (such as planners) to skip parsing altogether and further specialize in their area of focus.
- **Interoperability Bridge**: Through JSON message passing, our tool allows software components in different programming languages to communicate and work together seamlessly.

Future Work

Building upon the current capabilities, we plan to extend the toolkit with deserialization functionality that will enable bidirectional communication between the parser and external tools. This will enable a plugin architecture where other software can register themselves as specialized validators, allowing domain-specific checkers and other tools to contribute additional validation layers beyond our core syntax and semantic checking. Furthermore, we aim to expand the error detection capabilities, ultimately creating a more comprehensive and intelligent development environment for modeling hierarchical planning problems.

Acknowledgements

Pascal Bercher is the recipient of an Australian Research Council (ARC) Discovery Early Career Researcher Award (DECRA), project number DE240101245, funded by the Australian Government.

References

Alford, R.; Behnke, G.; and Schreiber, D., eds. 2024. *IPC 2023 – Proceedings of the Hierarchical Task Network (HTN) Track of the 11th International Planning Competition: Planner and Domain Abstracts*.

Höller, D.; Behnke, G.; Bercher, P.; Biundo, S.; Fiorino, H.; Pellier, D.; and Alford, R. 2020. HDDL: An Extension to PDDL for Expressing Hierarchical Planning Problems. In *Proceedings of the 34th Association for the Advancement of Artificial Intelligence (AAAI) conference*, 9883–9891. AAAI Press.

Sleath, K.; and Bercher, P. 2023. Detecting AI Planning Modelling Mistakes – Potential Errors and Benchmark Domains. In *Proceedings of the 20th Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, 448–454. Springer.