Specification of software project

&  
Implementation timeline

for

Transition-system-based HTN planning solver

HTN planning is a new approach to modelling planning problems. The goal of this project is to implement a HTN solver based on translating the planning problem to a reachability problem in a transition system and using off-the-shelf reachability solver for the actual solving.

2.0

Marian Kazimir

04.02.2021

Content

1. Basic information 1

1.1 Description and focus of the software project 1

1.2 Technology stack 1

1.3 References 1

1.4 Convention of this document 2

2. Brief description of the software project 2

2.1 The purpose of this software project and its parts and aims 2

2.2 Main functions 2

2.3 Motivational example 2

2.4 Application environment 3

2.5 Resctrictions of the software project 3

3. External environment 3

3.1 Users environment, inputs & outputs 3

4. Detailed description of software project 3

4.1 Input parser 3

4.2 Translator to transition system 3

4.3 Backend solver 3

5. Screens 4

5.1 Main screen - console 4

6. Other non-functional requirements 4

6.1 Extensibility & usage requirements 4

7. Restrictions 4

8. Time-line & Milestones 4

9. Notes 4

Revision table

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Changes** | **Version** |
| Marian Kazimir | 04.02.2021 | Minor updates connected to parser | 3.0 |
| Marian Kazimir | 04.12.2020 | Comments processed, extended information, minor mistakes corrected | 2.0 |
| Marian Kazimir | 26.11.2020 | Updates & format changes | 1.0 |
| Marian Kazimir | 10.11.2020 | Initial version | 0.0 |

# Basic information

## Description and focus of the software project

Currently, incremental SAT solvers are most commonly used for planning & scheduling task problems. When applying SAT solving to such problems, the full procedure features four steps: (1) enumerating and instantiating all the possible actions, (2) encoding the instantiated problem into propositional logic, (3) finding a solution with a SAT solver, (4) decoding the found variable assignment back to a valid plan. This SAT-based approach shows generally the best performance. Disadvantage of this metod is that one needs to build and issue a sequence of queries representing plans of fixed lengths. If no plan is found, the same is repeated for incremented length and so on.

The aim so to develop a different approach of solving HTN tasks planning/scheduling using transition system. We will model all possible states in the planning problem just once as a transition system and then existence of a plan is equivalent to a reachability of the goal state in this transition system.

## Technology stack

For all main parts of implementation – parsing, IO operations, etc., Java language will be used. The codebase will be stored under version control (GitHub, later also on school Gitlab). For parser – JavaCC will be used to create parser according to grammar of language used for input files.

## References

<https://www.scitepress.org/Papers/2019/73433/73433.pdf>, Efficient SAT Encodings for Hierarchical Planning - Dominik Schreiber, Damien Pellier, Humbert Fiorino and Tomas Balyo

<http://fmv.jku.at/papers/FazekasBiereScholl-SAT19.pdf>, Incremental Inprocessing in SAT Solving - Katalin Fazekas, Armin Biere , Christoph Scholl

<https://www.aaai.org/ocs/index.php/ICAPS/ICAPS17/paper/viewFile/15580/15097>, Accelerating SAT Based Planning with Incremental SAT Solving - Stephan Gocht and Tomas Balyo

<https://www.researchgate.net/publication/220543188_PDDL21_An_extension_to_PDDL_for_expressing_temporal_planning_domains>, PDDL2.1: An extension to PDDL for expressing temporal planning domains - Maria Fox and Derek Long

[https://www.uni-ulm.de/fileadmin/website\_uni\_ulm/iui.inst.090/Publikationen/2020/Hoeller2020HDDL.pdf](https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.090/Publikationen/2020/Hoeller2020HDDL.pdf%20), HDDL: An Extension to PDDL for Expressing Hierarchical Planning Problems - Daniel Holler, Gregor Behnke, Pascal Bercher, Susanne Biundo, Humbert Fiorino, Damien Pellier and Ron Alford

<http://gki.informatik.uni-freiburg.de/competition/>, IPC 2020 official page - Universität Freiburg

<http://www.cs.toronto.edu/~sheila/publications/soh-bai-mci-ijcai09.pdf>, HTN Planning with Preferences - Shirin Sohrabi, Jorge A. Baier, Sheila A. McIlraith

<http://www2.informatik.uni-freiburg.de/~ki/teaching/ss05/aip/aip01.pdf>, [Prof. Dr. Jussi Rintanen](http://users.rsise.anu.edu.au/~jussi/) - Foundations of Artificial Intelligence

## Convention of this document

The references to web sides are in underlined format followed by the description and optionally author/s.

Example:

* <http://gki.informatik.uni-freiburg.de/competition/>, IPC 2020 official page - Universität Freiburg

# Brief description of the software project

## The purpose of this software project and its parts and aims

The purpose, as stated in section 1.1 is to develop different approach to solving HTN planning problems. The experiment is to simplify problem before giving it to a SAT solver using transition system. This program will take the planning problem in the given format - totally ordered HDDL (more in the environment section) as common input language for hierarchical planning problems, parse it, create inner represetation (states, actions, goal, etc.), translate to transition system, find if the goal is achievable and in case such plan exists, the program will return this plan. Firstly, the approach is to find whatever plan, later there might by some differenet heuristic and optimization to find the best plan possible.

## Main functions

* Parse input of HTN domains and problems in totally ordered HDDL format
* Create inner representation in suitable sctructure of the given problem
* Translate to transition system
* Solve the problem using transition system – whether to find the assignment or to return negative answer (no such plan exists)
* Create plan of the problem if the goal is achievable

## Motivational example

As motivational and testing examples we will take tasks from IPC (International Planning Competition) in totally order HDDL format which can be found on IPC websites (<http://gki.informatik.uni-freiburg.de/competition/>, IPC 2020 official page). As input we will take such file, parse it, translate into transition system, solve it and create plan if the goal is achievable.

## Application environment

Application will have no GUI (but with possibility of later extension). The main environment will be just console which will take input file and print the results.

## Resctrictions of the software project

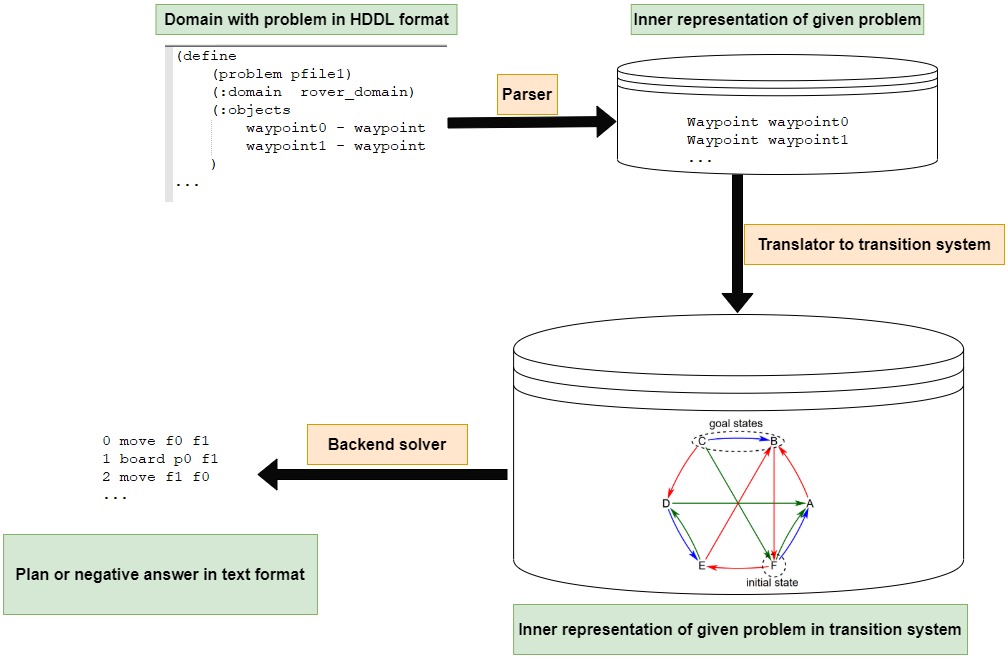
For this purpose the program will be tuned just for Windows OS as console application.. Furthermore, there will not be any check of the input format as only valid unput files are expected.

# External environment

## Users environment, inputs & outputs

The user will submit correct input file of the planning task in the correct given format using application console. For the input we will consider totally ordered HDDL - Hierarchical Domain Definition Language. HDDL is defined as an extension of the STRIPS fragment (language level 1) of the PDDL2.1 definition (Links to detailed info about this formats can be found in articles from 1.3 section). The output will be either negative answer (no such plan exists) or the plan of the given task in form of actions.

# Detailed description of software project



## Input parser

As input format HDDL2.1 was chosen as it is the most common format within this field. Input parser shall parse correct HTN planning tasks in this format and create inner represantion. We expect input in the correct format however the parser will be able to detect syntax errors. The input will take two parameters – paths to:

1. file where domain definition is stored
2. file where problem definition is stored

## Translator to transition system

The translator takes the inner representation of previously parsed HTN planning problem and translates it into the transition system representation.

## Backend solver

The final solver takes the problem in transition system and finds the assignment or states conclusion that such plan does not exist.(Concrete backend solver will be chosen later in progress)

# Screens

## Main screen - console

The aim of this project is not beautiful application with nice GUI. Program will have just one screen (console) which will take path to the input file of domain and problem in that domain and will print the result.

# Other non-functional requirements

## Extensibility & usage requirements

The architecture of this project should allow to change format of input in the future. Formats should be compatible with widely used formats within this field.

# Restrictions

The goal of this project is not development of another SAT solver.

# Time-line & Milestones

|  |  |  |
| --- | --- | --- |
| **Date** | **Milestone** | **presentation method** |
| 20.12.2020 | Complete specs, have the knowledge and plan how and what to do, parser implementation started | Call |
| 20.01.2021 | Parser with inner representation of domain and problem | Call |
| 30.02.2021 | Representation in transition system | Call |
| 29.3.2021 | Fully working program | Call |

# Notes

This specification is more than inspired by these templates

* Software Requirements Specification by Karl E. Wiegers
* SAFE™ Development System Requirements