Specification of software project

&  
Implementation timeline

for

HTN planning solver based on transition system

Attempt to improve planning/scheduling of hierarchical task networks by not solving them directly using incremental SAT but research usage of transition system for this purpose

1.0

Marian Kazimir

10.11.2020

Content

1. Basic information 1

1.1 Description and focus of software project 1

1.2 Used technologies 1

1.3 References 1

1.4 Convention of this document 1

2. Brief description of software project 1

2.1 The purpose of this software project and its parts and aims 1

2.2 Main function 2

2.3 Motivational example of usage 2

2.4 Application environment 2

2.5 Limitations of this project 2

2.6 Documentation 2

3. External interface 2

3.1 GUI, inputs & outputs 2

3.2 Interface with hw 3

3.3 Interface with sw 3

3.4 Communicational interface 3

4. Detailed description of functionality 3

4.1 Funkce 1 3

4.2 Funkce 2 3

4.3 Funkce 3-n 3

5. Screens 4

5.1 Obrazovka 1 4

5.2 Obrazovka 2 4

5.3 Obrazovka 3-n 4

6. Other non functional requirements 4

6.1 Performance requirements 4

6.2 Security requirements 4

6.3 Safety of data requirements 5

6.4 Extensibility & usage requirements 5

7. Other requirements 5

8. Negative definition 5

9. Time-line & Milestones 5

10. Notes 6

Revision table

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Changes** | **Version** |
| Marian Kazimir | 26.11.2020 | Updates & format changes | 1.0 |
| Marian Kazimir | 10.11.2020 | Initial version | 0.0 |

# Basic information

Currently, incremental SAT solvers are used for planning & scheduling tasks. 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. The experiment is to find a way how and if we can improve solving such problems using transition systems.

## Description and focus of the software project

The aim so to try to develop a different approach of solving HTN tasks planning/scheduling using transition system or to come to a conclusion that current approach is better and should be used.

## Technology stack

For all main parts of implementation – parsing, IO operations, etc., Java language will be used.

## References

<<https://www.scitepress.org/Papers/2019/73433/73433.pdf>>

<<http://fmv.jku.at/papers/FazekasBiereScholl-SAT19.pdf>>

<<https://www.aaai.org/ocs/index.php/ICAPS/ICAPS17/paper/viewFile/15580/15097>>

<https://www.researchgate.net/publication/220543188_PDDL21_An_extension_to_PDDL_for_expressing_temporal_planning_domains>

<https://www.uni-ulm.de/fileadmin/website\_uni\_ulm/iui.inst.090/Publikationen/2020/Hoeller2020HDDL.pdf >

< http://gki.informatik.uni-freiburg.de/ipc2020/format.pdf >

## Convention of this document

TBD

# Brief description of the software project

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

Currently, incremental SAT solvers are used for planning & scheduling tasks. This approach includes number of processes. The most expensive one is to find a solution with a SAT solver. The experiment is to find a way how how 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 in totally ordered HDDL format
* Create inner representation in suitable sctructure of the given problem
* Translate to transition systém
* 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

TBD

## Application environment

TBD

## Resctrictions of the software project

We suppos that the input is correct. There will not be any check of the input format.

## Additional documentation

TBD

# External environment

## Users environment, inputs & outputs

The user will submit correct input file of the planning task in the given format. 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 (Fox and Long 2003). The output will be either negative answer (no such plan exists) or the plan of the given task in form of actions.

## Hardware interface

TBD

## Software interface

This program should be compatible in form of inputs/outputs with other programs within this field.

## Communication interface

TBD

# Detailed description of software project

TBD

## Input parser

As input format HDDL was chosen as i tis the most common format within this field. Input parser should parse correct HTN planning tasks in this format and create inner represantion. The goal of the parser is not to determine whether the task is in correct format, so correctness checks are not part of it.

## Translator to transition system

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

## Backed solver

The final solver takes the problem in transition system and finds the assignment or states conclusion that such plan does not exist.

# ~~Screens~~

~~TBD~~

## ~~Obrazovka 1~~

~~TBD~~

## ~~Obrazovka 2~~

~~TBD~~

## ~~Obrazovka 3-n~~

~~TBD~~

# ~~Ostatní (mimofunkční) požadavky~~

## ~~Požadavky na výkon~~

~~<Pokud jsou na produkt kladeny nějaké specifické výkonové požadavky, specifikujte a zdůvodněte je zde. Může jít například o deadlines nebo latenci jednotlivých tasků u real-time systémů.>~~

## ~~Požadavky na bezpečnost využívání aplikace~~

~~<Specifikujte potenciální nebezpečí spojená s využíváním díla. Tato část může obsahovat například potenciální ztrátu dat, fyzické škody v případě řídících systémů nebo zdravotní rizika uživatelů.>~~

## ~~Požadavky na zabezpečení dat~~

~~<Specifikujte požadavky spojené s bezpečností a zabezpečením dat využívaných nebo vytvořených programem. Popište způsob autentizace uživatelů, pokud je v aplikace využívána. Uveďte všechny externí požadavky na bezpečnost, které musí být splněny.>~~

## 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.

# Other requirements

TBD

# 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 |
| 01.02.2021 | Demo version of program | Call |
| 02.02.2020 | Fully working program | Call |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

~~Dodatek A: Vymezení pojmů~~

~~<Definujte ne zcela obvyklé pojmy nutné k pochopení a správné interpretaci této specifikace.>~~

~~Dodatek B: To Be Determined List~~

~~<Uveďte seznam částí specifikace, které nebylo možno rozhodnout a popsat do doby dokončení tohoto dokumentu a budou dospecifikovány později. Při standardním průběhu projektu by tato část měla být nepotřebná.>~~

# Notes

This specification is more than inspired by these templates

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