TU WIEN Informatics

Titel der Arbeit

Optionaler Untertitel der Arbeit

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Logic and Computation

eingereicht von

Pretitle Forename Surname, Posttitle
Matrikelnummer 0123456

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Pretitle Forename Surname, Posttitle
Mitwirkung: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Wien, 1. Jänner 2001

Forename Surname	Forename Surname

TU WIEN Informatics

Title of the Thesis

Optional Subtitle of the Thesis

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Logic and Computation

by

Pretitle Forename Surname, Posttitle
Registration Number 0123456

to the Faculty of Informatics

at the TU Wien

Advisor: Pretitle Forename Surname, Posttitle
Assistance: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Vienna, 1st January, 2001

Forename Surname

Forename Surname

Erklärung zur Verfassung der Arbeit

Pretitle Forename Surname, Posttitle

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. Jänner 2001

Forename Surname

Danksagung

Ihr Text hier.

Acknowledgements

Enter your text here.

Kurzfassung

Ihr Text hier.

Abstract

Enter your text here.

Contents

CHAPTER 1

Introduction

Intro here

CHAPTER 2

The Evolog Language

2.1 Syntax

Every valid ASP-Core2 program is a valid Evolog program. In addition, Evolog programs may contain *action rules* and *module literals*.

Definition 2.1.1 (Action Rule)**.** Action Rules are ASP rules that have a body as defined by the ASP-Core2 standard [**?**] and an *action head*, where an action head is of the following form:

$$h : @a[i_1, \ldots, i_n] = v_r$$

where

- the head atom h is an ASP atom of form $p(t_1, \ldots, t_n)$ with p and $t_1 \ldots, t_n$ being a predicate symbol and a list of terms, respectively.
- the function symbol a is the name of an action function, i.e. an identifier starting with a lower-case letter
- action input terms t_1 through t_n are a list of terms
- result variable r_v is a variable.

Action result variables must not occur in the rule body.

Definition 2.1.2 (Module Literals)**.** TBD

2.2 Semantics

2.2.1 Action Rules

Desiderata

For every Evolog Program P and answer set A, the following must be clearly defined:

- $D1$: Which actions were executed by the program?
- $D2$: For every individual action *act*, what led to the action being executed, i.e. of which rule body is *act* a consequence?

Combining $D1$ and $D2$ it follows that

- $D3$: for actions that depend on other actions, it is clearly visible in which sequence they were executed, i.e. the respective execution sequence can be unambiguously reconstructed using the answer set and program('s dependency graph).

Furthermore,

- $D4$: all state changes effected on the outside world by execution of P are reflected in each answer set (as results of actions).

This is an example, make into a proper definition

Definition 2.2.1 (Expansion of action rules)**.** Semantically, every action rule is equivalent to its *expansion*:

$$file1\_open(OP\_RES) : @fileInputStream[PATH] = OP\_RES :- file1(PATH).$$

The expansion of $r1$ is:

$$action\_result(r1, fileInputStream, PATH, fileInputStream(PATH)) :- file1(PATH).file1\_open$$

Consequently, it is ensured that for each ground instance of an action rule R_a that fires, there is exactly one *action_result* instance in every answer set. We call this atom a *witness of action act*. Requirement $D1$ is fulfilled through the existence of action witnesses. Furthermore, inspection of a program (or its dependency graph) and all action witnesses in an answer set yields the information demanded in $D2$.

Definition 2.2.2 (Applicability of action rules)**.** In order to guarantee $D1$ and $D4$, for every (ground) action rule R_a that fires, it must hold that the corresponding *witness atom* is part of *every answer set*. Implementations may further restrict this in order to ensure static verifiability of the condition (e.g. by restricting action rules to the stratified part, i.e. common base program of a program).

Definition 2.2.3 (Rule Identifier)**.** Given a non-ground Evolog rule R, $id(R)$ denotes a (program-wide) unique identifier of R.

Definition 2.2.4 (Action function)**.** An action function f_{act} maps a rule id r, a tuple S, and a list of input terms $t_1, \ldots, t_n$ to a result term t_{res}.

- Identifier r references the rule (within a program) that is the *action source* (i.e. that fires in order to trigger the action)
- State S is a ground susbtitution for all body variables of the action source rule, i.e. it encodes the state of the world on which the action operates.

In accordance with Definition 2.2.4, an action witness $action\_result(r_1, fileInputStream, PATH, OP\_RES)$ then reads as "Function $fileInputStream$, with action source r_1, applied to input $PATH$, given world state ($PATH$), gives result OP_RES".

this should be done nicer. Actually, a function is applied to its input, given a rule body holds, that's exactly the elegance, we wanna put this into mathspeak.

Definition 2.2.5 (Interpretations of Evolog programs)**.** An Evolog interpretation I of a program P is a tuple (F, H) consisting of a *Frame* F and a herbrand interpretation H. The frame F defines the action functions associated with rules in P.

List of Figures

List of Tables

List of Algorithms

Bibliography

[Tur36] Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. *J. of Math*, 58:345–363, 1936.