Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Computer Science

Reversibility of Non-Deterministic Actions

Reversibilita nedeterministických akcí

MASTER'S THESIS

Study Programme:	Open Informatics
Specialization:	Artificial Intelligence
Author:	Bc. Jakub Med
Supervisor:	doc. RNDr. Lukáš Chrpa, Ph.D.
Year:	2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Med** Jméno: **Jakub** Osobní číslo: **483428**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Umělá inteligence**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Reversibilita nedeterministických akcí

Název diplomové práce anglicky:

Reversibility of non-deterministic actions

Pokyny pro vypracování:

Action reversibility deals with the problem of finding a sequence of (other) actions that undoes its effects. Several recent works investigate reversibility of deterministic actions. Action reversibility might be useful in fully observable non-deterministic (FOND) planning where one might want to undo undesirable effects of a non-deterministic action. However, the concept of action reversibility from deterministic planning cannot be straightforwardly adopted to FOND planning as "reverting" actions might also be non-deterministic.

Requirements:

• Investigate the concept of non-deterministic action reversibility and identify interesting classes of non-deterministic action reversibility (e.g., strong reversibility - guaranteed to undo action effects, weak reversibility - has a chance to undo action effects, uniform reversibility - the same policy undoes action effects in a set of states).

• Theoretically investigate relationships of the proposed classes of non-deterministic action reversibility.

• Design and develop methods for solving some classes of non-deterministic action reversibility.

• Evaluate the methods on at least 6 domain benchmarks of FOND planning.

Seznam doporučené literatury:

Michael Morak, Lukás Chrpa, Wolfgang Faber, Daniel Fiser: On the Reversibility of Actions in Planning. KR 2020: 652-661

Lukás Chrpa, Wolfgang Faber, Michael Morak: Universal and Uniform Action Reversibility. KR 2021: 651-654

Thomas Eiter, Esra Erdem, Wolfgang Faber: Undoing the effects of action sequences. J. Appl. Log. 6(3): 380-415 (2008)

Alessandro Cimatti, Marco Pistore, Marco Roveri, Paolo Traverso: Weak, strong, and strong cyclic planning via symbolic model checking. Artif. Intell. 147(1-2): 35-84 (2003)

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. RNDr. Lukáš Chrpa, Ph.D. optimalizace CIIRC

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **14.02.2023** Termín odevzdání diplomové práce: **26.05.2023**

Platnost zadání diplomové práce: **22.09.2024**

doc. RNDr. Lukáš Chrpa, Ph.D.
podpis vedoucí(ho) práce

podpis vedoucí(ho) ústavu/katedry

prof. Mgr. Petr Páta, Ph.D.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.

Datum převzetí zadání

Podpis studenta

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague on ....

Bc. Jakub Med

Acknowledgements

I thank my supervisor doc. RNDr. Lukáš Chrpa, Ph.D. for his valuable feedback, quick responses, periodical consultations, and excellent leadership for my entire work.

I would also like to thank my entire family, which provided me with an opportunity to study and supported me during the whole process, especially in the hard, stressful times.

Bc. Jakub Med

Název práce:
Reversibilita nedeterministických akcí

Autor: Bc. Jakub Med

Studijní program: Otevřená informatika
Obor: Umělá inteligence
Druh práce: Diplomová práce

Vedoucí práce: doc. RNDr. Lukáš Chrpa, Ph.D.
Oddělení průmyslové informatiky, Český institut informatiky, robotiky a kybernetiky, České vysoké učení technické v Praze

Abstrakt: Automatické plánování je podobor počítačových věd zabývající se formální specifikací problému, která je následována algoritmickým rozhodnutím, zda-li je některá cílová konfigurace dosažitelná z dané počáteční konfigurace. Prostředí obsahuje procesy, nazývané akce, které, pokud jsou aplikovány, mění jeho konfiguraci. Nejsou-li tyto procesy deterministické, mluvíme o nedeterminstických akcích nedeterministického prostředí. Jedna z pochopitelných otázek může být, zda-li jsou efekty některé akce zvrátitelné tak, že původní konfigurace prostředí před aplikací akce je opět dosažena. Tato práce představuje a dokazuje několik tvrzení týkajících se nedeterministické akční reversibility kombinací pojmů slabého a silného nedeterministického plánování s reversibilitou deterministický akcí, která již byla široce studována a vyvinutou pro deterministické formalismy. Práce také předkládá několik procesů a algoritmů, sloužících k rozhodnutí reversibility nedeterministické akce. Poté, práce vyhodnocuje navržené metody a algoritmy na množině nedeterministických domén používaných v literatuře. Nakonec je práce uzavřena shrnutím přínosů této práce a empirických poznatků dokazující reversibilitu zkoumaných nedeterministický akcí a diskuzí o možných tématech dalšího výzkumu.
Klíčová slova: reversibilita akcí, automatické plánování, nedeterministické plánování, strategie

Title:
Reversibility of Non-Deterministic Actions

Author: Bc. Jakub Med

Abstract: Automated planning is a sub-field of computer science which concerns a formal problem specification followed by algorithmic reasoning determining whether the goal configuration of the environment can be reached from the initial configuration. The environment contains processes, called actions, which, when applied, change its configuration. If these processes are not deterministic, we call them non-deterministic actions of non-deterministic environment. A legitimate question can be whether the effects of some action can be reversed, such that we end in the same configuration as before the application of the action. This work presents and proves multiple claims concerning non-deterministic action reversibility by combining notions of weak and strong non-deterministic planning with the action reversibility developed, which has been already widely studied and developed for deterministic formalisms. It proposes several processes and algorithms to decide the reversibility of non-deterministic actions. Then, it evaluates proposed methods and algorithms on a set of non-deterministic domains used in the literature. Ultimately, it concludes with a summarization of the contributions of the work and the gathered empirical evidence proving the reversibility of investigated non-deterministic actions and discussion on topics of further research.
Key words: action reversibility, automated planning, non-deterministic planning, policy

Contents

List of Abbreviations

FOND	fully observable non-deterministic
STRIPS	Stanford Research Institute Problem Solver
SAS$^+$	Simplified Action Structures
MDP	Markov Decision Process
BFS	breadth-first search
FD	Fast Downward
PRP	Planner for Relevant Policies
WUU	weak universal uniform action reversibility
SUU	strong universal uniform action reversibility
UI	universal action irreversibility
WU	weak uniform action reversibility
IPC	International Planning Competition

List of Figures, Algorithms and Tables

Introduction

Automated planning is a sub-field of computer science which concerns itself with a formal problem specification followed by algorithmic reasoning to determine whether the goal configuration of the environment can be reached from the initial configuration (Ghallab; Nau; Traverso, 2004; Ghallab; Nau; Traverso, 2016). There are multiple formalisms describing environments of different levels of abstraction possessing different expressiveness (Fikes; Nilsson, 1971; Bäckström; Nebel, 1995; Russell; Norvig, 2010; Helmert, 2006; Cimatti; Pistore; Roveri; Traverso, 2003). Algorithmic reasoning is done in an abstract environment (modelling the world) defined by the formalism. The environment consists (mainly) of the states and the actions. The actions define the possible transitions between the states of the environment through their preconditions and effects. An agent, the entity solving the problem, may use the actions of the environment to change the environment's configuration. One of the current research questions of *automated planning* is to describe when the action effects do not possess inevitable consequences, as the effects can be undone with actions of the environment. This phenomenon is frequently referred to by the literature as the *action reversibility*—the main topic of this work.

This work builds mainly builds on the research conducted on *classical planning* (Russell; Norvig, 2010; Ghallab; Nau; Traverso, 2016) and extends the state-of-the-art work on the *deterministic action reversibility* (Morak; Chrpa; Faber; Fiser, 2020; Chrpa; Faber; Morak, 2021; Faber; Morak; Chrpa, 2021) by adapting introduced notions for the *fully observable non-deterministic planning* (Cimatti; Pistore; Roveri; Traverso, 2003), as well as by proposing new concepts which are able to deal with the stochasticity of actions' effects.

In the first chapter, the necessary terminology and formal definition of *Simplified Action Structures* (Bäckström; Nebel, 1995) and *Fully-Observable Nondeterministic* (Cimatti; Pistore; Roveri; Traverso, 2003) formalisms used in *automated planning* is presented. Up to a difference in the actions of both formalisms, the formalisms are the same. The only difference lies in the stochasticity of actions. Actions of *classical planning* are deterministic, meaning they have only one, completely predictable outcome; whereas actions of *fully observable non-deterministic planning* may have multiple, completely expectable outcomes.

It is followed by a chapter devoted to the introduction of the *action reversibility.* At the beginning of the chapter, an overview of the state-of-the-art literature related to the *action reversibility* is presented. In the rest of the chapter, the *deterministic action reversibility* is outlined and definitions of the *action reversibility* for *fully observable non-deterministic planning*, which are adaptations of definitions of *deterministic action reversibility*, are presented. Together with the definitions, multiple claims and corollaries concerning *non-deterministic action reversibility* are provided. At the end of the second chapter, some complexity results of *deterministic action reversibility* are put forward, followed by a hypothesizing about the complexity of the *non-deterministic action reversibility.*

The third chapter extends existing knowledge, especially algorithms, designed to work with investigated formalism. It puts forward several novel propositions (as well as algorithms) usable for determining the reversibility of non-deterministic actions. It starts with an algorithm deciding a class of *non-deterministic action reversibility.* The algorithm is an adaptation of the deterministic variant of Chrpa; Faber; Morak (2021), and it is invented together with a few lemmas which it utilizes. Then, methods for another class of *non-deterministic action reversibility* are elaborated on, showing that only particular actions may be a part of this class. Therefore, for these actions, more theorems are proven. Furthermore, they are utilized to devise processes for proving the membership of the action to the class of the *non-deterministic action reversibility.* The chapter concludes with a scheme of a process for an incomplete *non-deterministic action reversibility* decision.

Experiments conducted for the purpose of this work are described in the fourth chapter. It firstly explains how the experiments were conducted. Later, an explanation of the domains used in this work is provided. It hypothesizes about the *action reversibility* of the researched domains. Furthermore, the chapter contains two tables showing empirical evidence gathered from the experiments. The first table lays out the overall results of *action reversibility*, whereas the second one summarizes the results regarding *φ-reversibility.* The chapter concludes with a section dedicated to a discussion on insufficiencies and possible improvements of the implementation, as well as on insufficiencies of the presented theory and its consequences. In the end, various applications of the *action reversibility* are put forward.

Finally, the last, concluding, chapter summarises the work, achieved theoretical and experimental results and proposes research topics requiring further investigation.

Chapter 1

Preliminaries

Before we discuss the *action reversibility*, we have to clear the grounds and properly define all notions and notations that are used later in this work.

The clarification is opened with an elaboration on the mathematical basics, namely on the sets, operations over them and sequences. It is followed by a section devoted to propositional logic, which is used in multiple occasions in this work.

1.1 Sets and Relations

This work provide no formal definitions concerning sets and sequences. The reader is referred to the work of Demlová (1999). It provides formal definitions of many notions related to sets and operations over them. One notation, that is ambiguous in the literature and needs clarification, is a power set—the set of all subsets. The work of Demlová (1999) uses $P(\Sigma)$ as a notation for the power set $P(\Sigma)$ of the set Σ and this work adopts it. Later, it defines another mathematical concept — the relation. In the end of the chapter of dedicated to the relation, a partially-ordered and totally-ordered sets are formalised.

This work frequently operates with the mathematical concept of sequence. Unfortunately, in the work of Demlová (1999), the definition of the sequence is missing. Informally, in this work, a sequence is considered a function that maps integers on the set of elements that may be contained in that sequence. They are enclosed in angle braces $\langle$ and $\rangle$ and the sequence of elements a_i of the length n is denoted as $\langle a_1, a_2, \ldots, a_{n-1}, a_n \rangle$. In this case, the element a_1 of the sequence is the first element of the sequence and the element a_n is the last element of the sequence. A *concatenation* of two *sequences* a and b, denoted as $a ^\frown b$, is, informally, a sequence where the elements of both sequences are present, the ordering of both sequences is preserved and new ordering is "added" such that each element of the sequence a is ordered before each element of the sequence b.

In this work, a notion of ordered tuples is also used. We disclaim that there is no difference between an ordered tuple and a sequence; but, this work uses tuples typically for a short sequences — like pairs, triplets and quadruples. Sequences, denoted with angle braces, will be used, once the solutions concepts, typically utilizing long sequences, are introduced.

1.2 Propositional Logic

To avoid any misunderstanding, this section formally presents definitions of propositional logic taken from the work of Demlová (1999). It also points out a few notions of propositional logic, which are used in this work however are not defined in it, with a reference to the work of Demlová (1999).

To be able to compactly represent situations of and conditions on various phenomena of inspected formalisms, we later define a propositional logic bounded to a particular (deterministic or stochastic) planning domain. In order to do so, we need to know what a propositional formula is. The following definition is taken from the work of Demlová (1999).

Definition 1.1. Let Σ be a non-empty set of elementary statements. A finite sequence of elements of the set Σ, of logical connectives $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$ and parentheses is called a **propositional formula** (or shortly a **formula**), if it is formed by the following rules:

1. Every elementary statement $\varphi_1 \in \Sigma$ is a propositional formula.
2. If φ_1, φ_2 are propositional formulae, then so are $(\neg\varphi_1)$, $(\varphi_1 \wedge \varphi_2)$, $(\varphi_1 \vee \varphi_2)$, $(\varphi_1 \Rightarrow \varphi_2)$ and $(\varphi_1 \Leftrightarrow \varphi_2)$.
3. Only sequences that were formed by using finitely many applications of rules 1 and 2, are propositional formulae.

The **set of all propositional formulae**, that we formed from the elementary statements from the set Σ is denoted by $\boldsymbol{\mathcal{P}(\Sigma)}$.

As Demlová (1999) states, formulae formed by using finitely many applications of the rules 1 and 2 in of the definition 1.1 may contain unnecessary parentheses. If it is unequivocally clear what formula is meant even without them, we can omit them and “relax” the structure of the formula.

We skip the definitions of the truth valuation of the formula, of the tautology and of the contradiction, as it is considered a general knowledge. For more detail, the reader is referred to the work of Demlová (1999). In an addition to the notation of Demlová (1999), we add notations representing the contradiction by a symbol $\perp$ and the tautology by a symbol $\top$.

One of the last concepts defined in this section is a semantical consequence. It is a concept describing what has to necessarily hold if a set of assumptions is satisfied. The notion will be frequently utilized when reasoning about the notions of *automated planning*.

Definition 1.2. Let φ be a formula and Φ be a set of formulae. We say that the formula φ is a semantical consequence of the set of formulae Φ, (or that φ semantically follows from the set Φ), denoted as $\Phi \models \varphi$, if and only if φ is true for every truth valuation for which every formula of the set Φ is true.

If the set Φ has only one element φ', then we can write $\varphi' \models \varphi$ instead of $\{\varphi'\} \models \varphi$; and the same thing is meant.

Ultimately, a definition of a semantical equivalence follows. If two formulae are semantically equivalent, then we can substitute one with another without any outcome from the point of view of propositional logic.

Definition 1.3. Let φ and ψ be formulae. We say that the formula φ is semantically equivalent to the formula ψ, denoted as $\varphi \models\mid \psi$, if and only if $\varphi \models \psi$ and $\psi \models \varphi$.

There are many corollaries and propositions about semantical consequence and equivalence. Similarly to sets and relations, this work does not provide them and the reader is referred to the work of Demlová (1999), as a few of them are exploited in the rest of the work.

1.3 State Variables

The work now moves to a part where two formalisms on which we focus in this work are defined simultaneously.

1.3.1 State Variable

The first formalism is the one mostly used in this work. It was developed for the purposes of *automated planning* and aims for dealing with a possibility of expectable non-deterministic behaviour of the environment it describes. The assumptions on the formalism are that the environment is fully observable (meaning the agent has an complete information about the environment which surrounds it) and non-deterministic. The non-determinism of the environment is modeled by stochastic action effects. We have described them as expectable due to a reason that the agent is provided of all possible random behaviours which may occur. Thanks to these two key properties, the field is named *fully observable non-deterministic planning* (Cimatti; Pistore; Roveri; Traverso, 2003).

The second formalism concerns *classical planning*, since *fully observable non-deterministic* (FOND) *planning* can be viewed as an extension of *classical planning* (Russell; Norvig, 2010) where actions may have multiple possible outcomes. In the literature, there are two main formalisms defining the classical planning: *Stanford Research Institute Problem Solver* (STRIPS) formalism (Fikes; Nilsson, 1971) and *Simplified Action Structures* (SAS^+) formalism (Bäckström; Nebel, 1995). Both formalisms have the same expressive ability, therefore one can be substituted with another without any information loss. One can define FOND formalism by extending either STRIPS or SAS^+ formalism to allow stochastic action effects. Such extended STRIPS-based and SAS^+-based FOND formalisms still have the same expressive abilities. For the purpose of this work, SAS^+-based FOND formalism is selected and it is formally defined in the following sections. Due to this reason, SAS^+ formalism will be defined simultaneously as well.

A foundation of both formalisms is a *state variable*. It is a structure with a purpose to capture varying situations of a specific components or parts of the modeled environment. Such varying situations are described by values. Values of the variable together make up a domain of the variable.

Definition 1.4. Let $\boldsymbol{v}$ be a **state variable** (or simply a **variable**). A non-empty set containing all possible objects assignable to the variable v, denoted as $\boldsymbol{dom(v)}$, is called a **domain of the variable** $\boldsymbol{v}$. The object $\boldsymbol{x \in dom(v)}$ assignable to the variable v is called a **value**. An ordered pair of the variable and of the value assigned to it, $\boldsymbol{(v, x)}$, $x \in dom(v)$, is called a **fact**. The **set of all facts** of the of set variables $\mathcal{V}$, denoted as $\boldsymbol{\mathcal{F}(\mathcal{V})}$, is the set $\mathcal{F}(\mathcal{V}) = \{(v, x) \mid v \in \mathcal{V}, x \in dom(v)\}$.

1.3.2 Variable Assignment

A single variable is rarely enough to describe the whole environment. Hence, multiple variables are used to capture all parts of it. Such variables are usually contained in a structure called variable assignment.

Definition 1.5. Let $\mathcal{V}$ be a set of variables. A set of ordered pairs $\Sigma = \{(v, x) \mid v \in \mathcal{V}, x \in \mathit{dom}(v)\}$ is called a **variable assignment over the set of variables** $\mathcal{V}$ if and only if for all $(v, x) \in \Sigma$ it holds that $\forall x' \in \mathit{dom}(v) \setminus \{x\} : (v, x') \notin \Sigma$. The **value of the variable** $\boldsymbol{v}$ **in the variable assignment** $\boldsymbol{\Sigma}$, denoted as $\Sigma[v]$, is equal to x if and only if $(v, x) \in \Sigma$.

According to definition 1.5, a variable assignment may assign values only for a few variables out of all variables available. For a convenience reason, the following definition of a new notation enables to refer to the assigned variables in an easy way.

Definition 1.6. Let Σ be a variable assignment over a set of variables $\mathcal{V}$. A **set of all variables assigned in the variable assignment** $\boldsymbol{\Sigma}$, denoted as $\boldsymbol{vars(\Sigma)}$, is defined as $vars(\Sigma) = \{v \mid (v, x) \in s\}$.

If a variable assignment assigns all possible variables of the set $\mathcal{V}$, then we say that the assignment is complete, meaning that no other variables can be assigned to it (since there is no unassigned variable left). Analogically, if there is some unassigned variable, we speak about partially assigned variable assignment, or simply about a partial variable assignment.

Definition 1.7. A variable assignment Σ over a set of variables $\mathcal{V}$ is called a **complete variable assignment over the set of variables** $\boldsymbol{\mathcal{V}}$ if and only if $vars(\Sigma) = \mathcal{V}$. A variable assignment is called a **partial variable assignment** if and only if it is not a complete variable assignment.

Given the preceding definitions, the amount of possible complete assignments over particular set of variables $\mathcal{V}$ is limited. Following definition sets a notation of such a set.

Definition 1.8. Let $\mathcal{V}$ be a set of variables. The **set of all complete variable assignments** over the set of variables $\mathcal{V}$, denoted as $\mathcal{S}(\mathcal{V})$, is the set $\mathcal{S}(\mathcal{V}) = \{\Sigma \mid \Sigma \in P(\{(v, x) \mid v \in \mathcal{V}, x \in \mathit{dom}(v)\}), vars(\Sigma) = \mathcal{V}\}$.

1.4 Actions

1.4.1 Definition

We follow with a definition of an *action*. In FOND planning, each action have its *preconditions* and, as previously outlined, one or more *possible effects*. On the other hand, SAS$^+$ actions do not have multiple possible effects, but exactly one (which can also be empty). In both formalisms, action's preconditions and effects are represented by variable assignments.

Definition 1.9. Let $\mathcal{V}$ be a set of variables. An **action** $\boldsymbol{a}$ over the set of variables $\mathcal{V}$ is an ordered pair $(\mathit{pre}(a), \mathit{eff}(a))$, where

- $pre(a)$ is a variable assignment over the set of variables $\mathcal{V}$, called **a set of preconditions of the action $\boldsymbol{a}$**, and
- $\mathit{eff}(a)$ is a non-empty set of variable assignments over the set variables $\mathcal{V}$, called **a set of possible effects of the action $\boldsymbol{a}$**, where $|\mathit{eff}(a)|$ is number of stochastic outcomes of the action a.

When there is only one possible effect of the action, then we speak about a *deterministic action*, because there is no stochasticity in its behaviour or application. The deterministic FOND action is equivalent to the action of SAS^+ formalism.

Definition 1.10. Let $a = (pre(a), \mathit{eff}(a))$ be a action. The action a is called **deterministic** if and only if $|\mathit{eff}(a)| = 1$.

If an action has a variable in the preconditions or in effects, the variable is relevant for the action. If it does not, the action neither depends on it nor changes it. One can imagine that the set of variables related by the action will be referred to, as it represents a helpful notion.

Definition 1.11. Let a be an action. A **set of variables relevant for the action $\boldsymbol{a}$**, denoted as $\boldsymbol{vars(a)}$, is a set $vars(a) = vars(pre(a)) \cup \bigcup_{e \in \mathit{eff}(a)} vars(e)$.

1.4.2 Determinization

We also define a way of creating multiple deterministic actions from a stochastic FOND action. The deterministic variants of the stochastic actions have same preconditions as the stochastic action, but only one of its possible effects. The resulting actions are called *determinizations*.

Definition 1.12. Let a be an action. A **determinization of an action $\boldsymbol{a}$** with a respect to the effect e, denoted as $\boldsymbol{a_e^d}$, is an action $a_e^d = (pre(a), \{e\})$, where $e \in \mathit{eff}(a)$ is a possible effect of the action a.

1.4.3 Action Applicability

Now, it is necessary to define how preconditions and effects of actions reflects possible changes of the environment. In order to allow action to modify the environment, its preconditions have to be satisfied in a current configuration of the environment. The notion is formally stated in the following definition.

Definition 1.13. Let $\mathcal{V}$ be a set of variables, $a = (pre(a), \mathit{eff}(a))$ be an action over the set of variables $\mathcal{V}$ and Σ be a variable assignment over the set of variables $\mathcal{V}$. We say that the **action $\boldsymbol{a}$ is applicable in the variable assignment $\boldsymbol{\Sigma}$** if and only if $pre(a) \subseteq \Sigma$.

Also, to simplify the notation of the work, a definition of a *set of applicable actions* in a particular variable assignment is utilized.

Definition 1.14. Let Σ be a variable assignment. The **set of applicable actions in the variable assignment $\boldsymbol{\Sigma}$ from the set of actions $\boldsymbol{\mathcal{A}}$**, denoted as $\alpha(\Sigma, \mathcal{A})$, is the set $\alpha(\Sigma, \mathcal{A}) = \{a \mid a \in \mathcal{A}, pre(a) \subseteq \Sigma\}$.

1.4.4 Application of Deterministic Action

When any action has only one effect, we speak about a *deterministic action*. The result of its application in any variable assignment which satisfies its preconditions is unambiguous. The result contains the same facts as the previous assignment, but for each variable mentioned in the effects of the action, the previous value of that variable is replaced with the value from the effects.

Definition 1.15. Let $\mathcal{V}$ be a set of variables, $a = (pre(a), \{e\})$ be a deterministic action over the set of variables $\mathcal{V}$ and Σ be a variable assignment over the set of variables $\mathcal{V}$ in which the action a is applicable. The **application of the deterministic action $\boldsymbol{a}$ in the variable assignment $\boldsymbol{\Sigma}$**, denoted as $\boldsymbol{\gamma(\Sigma, a)}$, is the variable assignment $\gamma(\Sigma, a) = e \cup \{(v, x) \mid (v, x) \in \Sigma, v \notin vars(e)\}$.

Note, if the deterministic action a is not applicable in the variable assignment Σ, then the application of the action a in the variable assignment Σ is undefined.

A variable assignment related to the application of the deterministic action is a set of facts that necessarily *hold after* the application of the action. Intuitively, the set contains all effects of the action and preconditions which were not overridden. Also, the set of facts that hold after the application of a deterministic action is equal to the result of the application of the deterministic action in its preconditions.

Definition 1.16. Let $a = (pre(a), \{e\})$ be a deterministic action. A **set of facts that necessarily hold after the application of the deterministic action $\boldsymbol{a}$**, denoted as $\boldsymbol{ha(a)}$, is the variable assignment $ha(a) = \gamma(pre(a), a)$.

1.4.5 Application of Stochastic Action

As formerly defined, a stochastic action is an action that is not limited in the amount of possible effects, as long as it has at least one. In that case, its application may result in multiple outcomes—one for each possible effect. Each possible outcome then corresponds to an assignment arising as a result of the application of the corresponding determination of the stochastic effect.

Definition 1.17. Let $\mathcal{V}$ be a set of variables, $a = (pre(a), eff(a))$ be an action over the set of variables $\mathcal{V}$ and Σ be a variable assignment over the set of variables $\mathcal{V}$ in which a is applicable. The **application of the action $\boldsymbol{a}$ in the variable assignment $\boldsymbol{\Sigma}$**, denoted as $\boldsymbol{\delta(\Sigma, a)}$, is the set of variable assignments $\delta(\Sigma, a) = \{\gamma(\Sigma, a^d_e) \mid e \in eff(a)\}$.

Equivalently to the definition of γ, if the action a is not applicable in the variable assignment Σ, then the application of the action a in the variable assignment Σ is undefined.

While both stochastic and deterministic action belong to FOND planning, *classical planning* allows only the deterministic variant. This only difference between formalisms is small, yet has wide and significant consequences and allows to model environments which *classical planing* cannot describe.

1.5 Domains and Tasks

1.5.1 Domain

Now, once actions and variables are defined, we have completely described the environment of both SAS$^+$ and FOND formalisms. Possible configurations of the environment are described by complete variable assignments, called states, and the way how the environment changes is modelled with actions. Together they form a model of the environment, called a *domain* of the environment. The domain is a formal description of the real environment and enables automated and algorithmic problem solving of an arbitrary described environment.

Speaking of FOND and SAS$^+$ formalisms, domains, as already noted, differ only in the stochasticity of the allowed actions. For clarity, we provide both formal definitions. When the keyword SAS$^+$ or FOND is missing, the stochastic variant is meant, as it is more general then the other one; unless the context clearly implies the second, deterministic one.

Definition 1.18. A **(FOND) planning domain $\boldsymbol{\mathcal{D}}$** is an ordered pair $(\mathcal{V}, \mathcal{A})$, where

- $\mathcal{V}$ is a finite set of **domain variables**,
- $\mathcal{A}$ is a **set of actions** over the set of variables $\mathcal{V}$.

Definition 1.19. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. The planing domain $\mathcal{D}$ is a **SAS$^+$ planning domain** if and only if all actions in $\mathcal{A}$ are deterministic.

1.5.2 State

When any complete (respectively partial) variable assignment describes the configuration of the environment, it is referred to as a *state* (resp. a *sub-state*) of the *domain* modelling the environment.

Definition 1.20. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. A **state $\boldsymbol{s}$** of the domain $\mathcal{D}$ is a complete variable assignment over the set of variables $\mathcal{V}$. A **sub-state $\boldsymbol{s'}$** of the domain $\mathcal{D}$ is a partial variable assignment over the set of variables $\mathcal{V}$.

As states correspond to possible configurations of modelled environment, they are assumed to be reasonable. Sometimes, this might not be the case. Therefore, in the literature, there are also alternative versions of definitions of planning domains which incorporates the set of states directly in the domain definition as a subset of possible complete variable assignments. Consequently, this allows to forbid some complete variable assignments from being states. For example, to filter out unreasonable configurations. However, in this work, we did not settle for such an approach and we delegate the issue of unreasonable states to the domain engineering.

1.5.3 Formulae over Domains

This work utilizes propositional logic to reason about states and the later defined solution concept. Hence, the following definition of formulae over the elementary statements present in the domain is needed. It describes and restricts a set of formulae serving that purpose.

Definition 1.21. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. A set of all propositional formulae over the domain $\mathcal{D}$, denoted as $\mathcal{P}(\mathcal{D})$, is a set of propositional formulae $\mathcal{P}(\mathcal{F}(\mathcal{V}))$ over the set of facts $\mathcal{F}(\mathcal{V})$, where each fact $(v, x) \in \mathcal{F}(\mathcal{V})$ is considered as an elementary statement.

One of the previous paragraphs claimed that states of the domain describe a particular configuration of the environment. Thus, a formula expressing the exact configuration would be convenient. It is obvious, that the formula has to require all facts (resp. corresponding elementary statements) in a conjunction. The definition 1.22 defines it formally.

Definition 1.22. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and Σ be a variable assignment over the variables $\mathcal{V}$. A formula derived from the variable assignment Σ, denoted as $\Psi(\Sigma)$, is propositional formula over the domain $\mathcal{D}$ such that $\Psi(\Sigma) \in \mathcal{P}(\mathcal{D})$ and $\Psi(\Sigma) = \bigwedge_{f \in s} f$.

1.5.4 Task

In general, *automated planning* concerns of a plan finding—beginning in the initial state to any possible goal state in an abstract space defined by the domain. Such typical problem of *automated planning* is usually called *planning task* or *planning problem*. From the theoretical point of view, the only difference between FOND and SAS$^+$ planning tasks is in the domains.

Definition 1.23. A **(FOND) planning task** $\mathcal{T}$ is an ordered triplet $(\mathcal{D}, s_I, G)$, where

- $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ is a (FOND) planning domain,
- $s_I \in \mathcal{S}(\mathcal{V})$ is an initial state and
- $G \subseteq \mathcal{S}(\mathcal{V})$ is a set of goal states.

Definition 1.24. Let $\mathcal{T} = (\mathcal{D}, s_I, G)$ be a planning task. The planing task $\mathcal{T}$ is a **SAS$^+$ planning task** if and only if all the planning domain $\mathcal{D}$ is the SAS$^+$ planning domain.

1.6 Solution Concepts

Now that task is clear, appropriate structures able to describe the solutions—a ways how to get to some goal state from the initial state—are necessary. These structures will be, in general, referred to as solution concepts.

1.6.1 Plan

If we take into account *classical planning*, then the solution concept is straightforward. As the formalism does not contain any stochasticity, no random phenomenon may occur. Hence, the solution of the task does not have to react to the randomness and it can be a simple sequence of appropriate actions in a right order. Such sequences of actions are called *plans*.

Definition 1.25. Let $\mathcal{D}^d = (\mathcal{V}, \mathcal{A}^d)$ be a SAS$^+$ planning domain. A sequence of actions $\pi = \langle a_1, \dots, a_n \rangle$, $\forall i \in \mathbb{N}, 1 \leq i \leq n : a_i \in \mathcal{A}^d$, is called a **plan** for the planning domain $\mathcal{D}^d$ and the **length of the plan π** is n. The plan π is called **empty** if and only if it is an empty sequence. Otherwise, the plan π is called **non-empty**.

In the literature, this is called a *totally-ordered plan*. Alternative to the *totally-ordered plan* is a *partially-ordered plan* (Nguyen; Kambhampati, 2001), which is represented by a partially-ordered multi-set of actions rather than the totally-ordered sequence of actions.

In order to solve the planning task, we seek a plan which transforms the initial state to some goal state. The transformation is done through consecutive application of actions of the plan. This is called an application of the plan.

If the plan is empty, then, intuitively, the state is not transformed at all.

When there is at least one action to apply, we firstly apply the first of them and then recursively apply the rest of actions on the resulting state.

Similarly to the application of the action, the variable assignment Σ is required to satisfy the preconditions of the first action of the sequence. If this condition is violated, then the application of the non-empty plan is undefined.

Definition 1.26. Let $\mathcal{D}^d = (\mathcal{V}, \mathcal{A}^d)$ be a SAS$^+$ planning domain, Σ be a variable assignment over the variables $\mathcal{V}$, $\pi = \langle a_1, a_2, \ldots, a_n \rangle$ be a non-empty plan for the domain $\mathcal{D}^d$ and $\pi' = \langle\rangle$ be an empty plan for the domain $\mathcal{D}^d$. An **application of the empty plan $\boldsymbol{\pi}$** in the variable assignment Σ, denoted as $\boldsymbol{\gamma(\Sigma, \langle\rangle)}$, is the variable assignment $\gamma(\Sigma, \langle\rangle) = \Sigma$. An **application of the non-empty plan $\boldsymbol{\pi}$** in the variable assignment Σ in which the action a_1 is applicable, denoted as $\boldsymbol{\gamma(\Sigma, \pi)}$, is a variable assignment $\gamma(\Sigma, \pi) = \gamma(\gamma(\Sigma, a_1), \langle a_2, \ldots, a_n \rangle)$.

Now, that we are able to describe multi-step transitions between the states of the domain, the states of the domain with a respect to the planning problem can be split into two disjoint groups—a reachable and an unreachable states.

Intuitively, a reachable state is a state which can be visited by applying some sequence of actions to the initial state of the problem. If there is no sequence of actions achieving the state, then it is unreachable.

Definition 1.27. Let $\mathcal{T}^d = (\mathcal{D}^d, s_I, G)$ be a SAS$^+$ planning task and $s \in \mathcal{S}(\mathcal{V})$ be a state of the planning domain $\mathcal{D}^d$. The state s is **reachable** in the planning task $\mathcal{T}^d$ if and only if exists a plan π for the domain $\mathcal{D}^d$ such that $\gamma(s_I, \pi) = s$. Otherwise, it is **unreachable**.

Focusing on the goal states, if some of them is reachable, the planning task is considered *solvable*, as there is at least one plan (in this case called a goal plan) which achieves some goal state. On the other hand, if none of the goal states is reachable, then the task is *unsolvable*.

Definition 1.28. Let $\mathcal{T}^d = (\mathcal{D}^d, s_I, G)$ be a SAS$^+$ planning task. The task $\mathcal{T}^d$ is **solvable** if and only if some goal state $s_G \in G$ is reachable in the planning task $\mathcal{T}^d$. Otherwise, it is **unsolvable**.

A such plan achieving any of goal states is a desired solution of the problem which is searched. That is why it is called a *solution* or a *goal plan* of the planning task.

Definition 1.29. Let $\mathcal{T}^d = (\mathcal{D}^d, s_I, G)$ be a solvable SAS$^+$ planning task and π be a policy for the planning task $\mathcal{D}^d$. The plan π is called a **goal plan** or a **solution** of the planing task $\mathcal{T}^d$ if and only if holds that $\gamma(s_I, \pi) \subseteq G$.

The problem of finding of any solution of classical planning problem can be considered as the main problem of *classical planning*. It is an active field of research and it has already been widely investigated, yet it still has many unsolved challenges (Richter; Westphal, 2010; Helmert; Sievers; Rovner; Corrêa, 2022; Corrêa; Seipp, 2022; Christen; Eriksson; Pommerening; Helmert, 2022).

1.6.2 Policy

Moving to a second and more important solution concept of this work—a *policy*. In a contrast to *classical planning* and to the plan, FOND planning allows certain degree of stochasticity, which is not in a full control of the agent. Hence, a simple totally or partially-ordered sequence of actions may not be enough to successfully solve the task. When the agent wants to apply any non-deterministic action, he has no guarantee in which environment configuration he will have to act next time. Therefore, a solution concept able to react to the immediate situation is needed. Such concept is more general than the plan itself and is called a policy.

It allows to select a proper action based on the state in which the agent is in a particular situation when it should act. Notice, that the policy does not provide a memory or any similar concept, that would enable to behave conditionally and react to the past. Such concept would be more general then the policy as we defined it and it would allow more advanced strategies. Similarly to *Markov Decision Process* (MDP) (Russell; Norvig, 2010; Mausam; Kolobov, 2012), the agent acts based on the information of the current situation only, as no information about his previous decisions or about the previous configuration of the environment is provided to him.

Definition 1.30. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. A **policy $\boldsymbol{\Pi}$** for the domain $\mathcal{D}$ is a binary relation over the set of states $\mathcal{S}(\mathcal{V})$ and the set of actions $\mathcal{A}$ such that $\Pi \subseteq \{(s, a) \mid s \in \mathcal{S}(\mathcal{V}), a \in \alpha(s, \mathcal{A})\}$. The **set of all states related in the policy $\boldsymbol{\Pi}$**, denoted as $\boldsymbol{\sigma(\Pi)}$, is a set of states $\sigma(\Pi) = \{s \mid (s, a) \in \Pi\}$.

The notion of *policy following* is analogous to the *plan following*. In each step, instead of blindly applying the actions of the plan, we apply an action for which the pair of current state the agent is in and of the action is in the policy.

A special case of the policy following is to follow it for zero steps. It is, as an intuition guides us, defined as doing nothing.

Once the zero-step application is defined, a recursive definition of the *n-step application* may be used. Informally, n-step (where n is a natural number greater than zero) application of the policy Π is defined as a union of the sets of states which are results of the application of related actions in the states that are results of $(n-1)$-step application.

Definition 1.31. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $s \in \mathcal{S}(\mathcal{V})$ be a state of the domain $\mathcal{D}$ and Π be a policy for the planning domain $\mathcal{D}$. A **0-step application of the policy $\boldsymbol{\Pi}$ in the state $\boldsymbol{s}$**, denoted as $\boldsymbol{\delta^0(s, \Pi)}$, is the set states $\delta^0(s, \Pi) = \{s\}$. Let n be a positive integer. An ***n*-step application of the policy $\boldsymbol{\Pi}$ in the state $\boldsymbol{s}$**, denoted as $\boldsymbol{\delta^n(s, \Pi)}$, is the set $\delta^n(s, \Pi) = \bigcup_{s' \in \delta^{n-1}(s, \Pi)} \bigcup_{a \in \{a' \mid (s'', a') \in \Pi, s'' = s'\}} \delta(s', a)$.

As a consequence of the previous definition, there are some states of the domain which are at least in one of the sets of n-step application (given n is natural number and application start in some state). In such a case, these states are called *reachable* by the particular policy.

Notice the discrepancy between the reachability in classical and FOND planning. The reachability in classical planning characterizes the property of the states in domain not bounded to the particular plan, while the reachability in FOND planning (at least as we defined it) is bounded to the particular policy. This is caused by non-determinism of the formalism, as not all states related in the policy must be visit during acting.

The analogous notion for *classical planning* is called a state trajectory—a sequence of states which are visited during plan application in some (mostly in the initial) state.

Definition 1.32. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $s \in \mathcal{S}(\mathcal{V})$ and $s' \in \mathcal{S}(\mathcal{V})$ be a state and Π be a policy for the planning domain $\mathcal{D}$. We say that $\boldsymbol{s'}$ **is reachable from** $\boldsymbol{s}$ **with a policy** $\boldsymbol{\Pi}$ if and only if exists non-negative integer i such that $s' \in \delta^i(s, \Pi)$.

There is another interesting group of states among the reachable states. The states of that group are interested in a way that they are reachable, but, since there is no state-action pair concerning them, no other states are reached from them. Such states are called *terminal*, as we now state formally.

Definition 1.33. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $s \in \mathcal{S}(\mathcal{V})$ and $s' \in \mathcal{S}(\mathcal{V})$ be a state for the domain $\mathcal{D}$ and Π be a policy for the planning domain $\mathcal{D}$. We say that $\boldsymbol{s'}$ **is a terminal state for the policy** $\boldsymbol{\Pi}$ **with respect to the state** $\boldsymbol{s}$ if and only if s' is reachable from s with the policy Π and $s' \notin \sigma(\Pi)$. The **set of all terminal states for the policy** $\boldsymbol{\Pi}$ with respect to the state s is denoted as $\boldsymbol{\tau(\Pi, s)}$.

The work of Cimatti; Pistore; Roveri; Traverso (2003) focusing on the research of FOND formalism introduced *weak*, *strong* and *strong cyclic* planning. Informally, notions of the *weak* planning and *weak* solutions is concerning situations when there is a possibility of reaching some target configuration of the environment. So called *strong* (acyclic) and *strong cyclic* planning and solutions are describing situations when it is guaranteed to reach some target configuration of the environment, no matter what stochastic effects occur. The distinction between *acyclic* and *cyclic strong* solutions lies in the possible cyclicity of the solutions navigating the agent to target configurations. When the agent cannot reach the state in which he has been before, then we speak about *strong (acyclic)* solutions; otherwise, we speak about the *strong cyclic* solutions.

This work adopts the idea, but does not distinguish between *cyclic* and *acyclic* solutions.

Definition 1.34. Let $\mathcal{T} = (\mathcal{D}, s_I, G)$ be a planning task and Π be a policy for the domain $\mathcal{D}$. The policy Π is called a **weak goal policy** for the task $\mathcal{T}$ if and only if $\tau(\Pi, s_I) \cap G \neq \emptyset$.

Definition 1.35. Let $\mathcal{T} = (\mathcal{D}, s_I, G)$ be a planning task and Π be a policy for the domain $\mathcal{D}$. The policy Π is called a **strong goal policy** for the task $\mathcal{T}$ if and only if $\tau(\Pi, s_I) \subseteq G$.

The task of FOND planning is considered solvable, if there is an arbitrary way how to reach any goal state. This is in a coincidence with an existence of weak goal policy of the same task.

Definition 1.36. Let $\mathcal{T} = (\mathcal{D}, s_I, G)$ be a planning task. The **task** $\boldsymbol{\mathcal{T}}$ is called **solvable** if and only if exists a weak goal policy for the task $\mathcal{T}$. Otherwise, it is called **unsolvable**.

A policy can contain multiple state-action pairs which are not necessary for the purpose of the policy. Imagine the situation where there is a policy Π which terminates in some state we want to reach from some beginning[1] state. Such policy can be extended by any pair (s, a) such that s is not reachable from the beginning state by the policy Π. The policy $\Pi \cup (s, a)$ still terminates in the desired state, since the pair (s, a) is, in this case, irrelevant. However, in some situations, such irrelevant actions may cause a problem. For example, when combining policies together. The process of removal of such problematic actions is described in the following definitions.

Definition 1.37. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and Π be a policy for the planning domain $\mathcal{D}$. The **policy Π' for the domain $\mathcal{D}$** is called **the refinement of the policy Π** with respect the set of states Σ, denoted as $\boldsymbol{\rho(\Pi, \Sigma)}$, if and only if $\Pi' = \rho(\Pi, \Sigma) = \{(s', a) \mid (s', a) \in \Pi, s \in \Sigma, s' \text{ is reachable from } s \text{ with the policy } \Pi\}$.

The last notion defined in this section is *implicitly-defined policy.* It will become useful afterwards, as it will simplify notations and explanations when constructing new policies. The following definition provides a recipe how to construct *explicitly-defined* polices as in the definition 1.30, but from a set of pairs of propositional formulae and actions. The (explicitly-defined) policy is constructed in such a way that for each pair of propositional formula and action, each pair of the state, which models the propositional formula, and of the action is added to the policy.

Definition 1.38. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, Φ be a set of pairs (φ, a), where $\varphi \in \mathcal{P}(\mathcal{D})$ is a propositional formula over the domain $\mathcal{D}$ and $a \in \mathcal{A}$ is an action such that $\varphi \models \Psi(pre(a))$, and Π be a policy for the domain $\mathcal{D}$. The set Φ is called an **implicitly-defined policy** for the domain $\mathcal{D}$ and it is equivalent to the (explicitly-defined) policy $\Pi = \bigcup_{(\varphi,a)\in\Phi}\{(s, a) \mid s \in \mathcal{S}(\mathcal{V}), s \models \varphi\}$. We also say that the **set $\boldsymbol{\Phi}$ implicitly defines the policy** Π.

Notice, that the notation $s \models \varphi$ is valid, since the variable assignment is a set of facts and each fact is an elementary statement. The notation $\{\Psi(s)\} \models \varphi$ or $\Psi(s) \models \varphi$ would work the same, but since $s \models \varphi$ works, we have settled with it.

Since, for any propositional formula, there is infinitely many semantically equivalent formulae, it is no surprise of validity of an analogous concept for the implicitly-defined policies. Given an implicitly-defined policy, we can construct a new and equivalent policy, by a substitution of arbitrary propositional formula with another semantically equivalent formula, for example with its conjunctive or disjunctive normal form.

Furthermore, if some formula of an implicitly-defined policy is in a disjunctive normal form, we can split the disjunction into multiple formula-action pairs which can replace the former one.

By other similar tricks, an implicitly-defined policy can contain formulae which are trivial conjunctions of elementary statements only. After all, even the explicitly-defined policy can be viewed as an implicitly-defined one, up to a trivial operation Ψ which transforms states into a formula of conjunction.

[1] We deliberately avoid the word “initial” here to distinguish the state from the initial state of the task, since policies can and will be utilized more widely than being a solution to the planning task only.

Properties of Policy

For the purpose of later proven lemmata and theorems, this part of the work present multiple claims about policies, and especially what happens if they are merged together.

Notice, that due to the definition of the policy, it may happen that for a particular state, multiple actions may be selected. This property have several positive and negative consequences.

One of the positive consequences is that we can simply unify two policies together, without being necessary to deal with a conflicts, if there is some state for which different action is selected. This means that for any two policies the union of them is also a policy.

In either case, the set of all states related in the union of two policies is equal to the union of sets of all states related in both policies.

Lemma 1.1. *Let $\mathcal{D}$ be a planning domain. Then, for any two policies Π_1, Π_2 for the domain $\mathcal{D}$ holds that $\sigma(\Pi_1 \cup \Pi_2) = \sigma(\Pi_1) \cup \sigma(\Pi_2)$.*

Proof. $\sigma(\Pi_1 \cup \Pi_2) = \{s \mid (s,a) \in \Pi_1 \cup \Pi_2\} = \{s \mid (s,a) \in \Pi_1\} \cup \{s \mid (s,a) \in \Pi_2\} = \sigma(\Pi_1) \cup \sigma(\Pi_2)$. □

One of the negative consequences will arise when we have to act according to the policy. If for a particular state the policy contains multiple actions, then it is not clear which of them should be selected by the agent. In our case, we allow a selection of any of them, since definition 1.31 considers all options.

A trivial observation is that there is no state that is related by the policy and also is a terminal state of that policy with respect to any state of the domain. Without any extensive proof, we argue that the observation simply follows from the definitions.

Corollary 1.2. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $s \in \mathcal{S}(\mathcal{V})$ be a state of the domain $\mathcal{D}$. For any policy Π for the planning domain $\mathcal{D}$ it holds that $\sigma(\Pi) \cap \tau(\Pi, s) = \emptyset$.*

Proof. Trivial, from definitions of τ and σ. □

A direct utilization of corollary 1.2 is in a combination with lemma 1.1. These two together proves that any state related in either policy cannot be a terminal state of a general, unified policy and vice versa. This conclusion is trivial, yet will be frequently referred to in this work.

Lemma 1.3. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $s \in \mathcal{S}(\mathcal{V})$ be a state of the domain $\mathcal{D}$. For any two policies Π_1, Π_2 for the planing domain $\mathcal{D}$ it holds that $(\sigma(\Pi_1) \cup \sigma(\Pi_2)) \cap \tau(\Pi_1 \cup \Pi_2, s) = \emptyset$.*

Proof. Trivial, from lemma 1.1 and corollary 1.2. □

Another presented property is also concerning terminal states of the policy of the union. It argues that terminal states of the policy of the union necessarily contains the terminal states of both policies up to the states related in both policies.

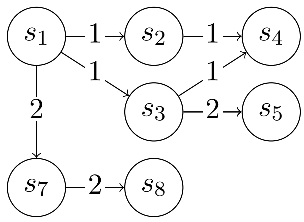

Figure 1.1: Counter-example for the proof of proposition 1.5.

Proposition 1.4. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $s \in \mathcal{S}(\mathcal{V})$ be a state of the domain $\mathcal{D}$. For any two policies Π_1, Π_2 for the planing domain $\mathcal{D}$ holds that $(\tau(\Pi_1, s) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s) \setminus \sigma(\Pi_1)) \subseteq \tau(\Pi_1 \cup \Pi_2, s)$.*

Proof.

Since adding any valid pair to any policy cannot disrupt reachability of any previously reachable state, the set of all reachable states from the state s by the policy $\Pi_1 \cup \Pi_2$ necessarily contains all states reachable from s by the policy Π_1 (resp. Π_2); and therefore, the also all states of $\tau(\Pi_1, s)$ (resp. $\tau(\Pi_2, s)$) as well.

Consider a state $s' \in \tau(\Pi_1, s)$. Because $\sigma(\Pi_1 \cup \Pi_2) = \sigma(\Pi_1) \cup \sigma(\Pi_2)$ and $s' \notin \sigma(\Pi_1)$ (see lemma 1.1 and corollary 1.2), $s' \in \sigma(\Pi_1 \cup \Pi_2)$ if and only if $s' \in \sigma(\Pi_2)$. Since s' is reachable from s by the policy $\Pi_1 \cup \Pi_2$ (see the first paragraph of this proof) and due to the definition of τ, we get that $s' \in \tau(\Pi_1 \cup \Pi_2, s)$ if and only if $s' \notin \sigma(\Pi_2)$. From these claims one can derive that $\tau(\Pi_1, s) \setminus \sigma(\Pi_2) \subseteq \tau(\Pi_1 \cup \Pi_2, s)$. The same claim can be derived for $\tau(\Pi_2, s) \setminus \sigma(\Pi_1)$ symmetrically. Hence, $(\tau(\Pi_1, s) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s) \setminus \sigma(\Pi_1)) \subseteq \tau(\Pi_1 \cup \Pi_2, s)$. □

We have also investigated whether there is an equality between the sets $(\tau(\Pi_1, s) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s) \setminus \sigma(\Pi_1))$ and $\tau(\Pi_1 \cup \Pi_2, s)$. A counter-example against the equality immediately follows in a proof of proposition 1.5.

Proposition 1.5. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $s \in \mathcal{S}(\mathcal{V})$ be a state of the domain $\mathcal{D}$. There are two policies Π_1, Π_2 for the planing domain $\mathcal{D}$ such that $\tau(\Pi_1 \cup \Pi_2, s) \neq (\tau(\Pi_1, s) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s) \setminus \sigma(\Pi_1))$.*

Proof.

Figure 1.1 shows two policies Π_1 and Π_2 on eight states in a directed graph. The states are represented by vertices and state-action pairs of policies by edges. Labels of edges indicate the affiliation to the policy Π_1 or Π_2.

It can be seen that $s_5 \in \tau(\Pi_1 \cup \Pi_2, s_1)$, but neither $s_5 \in \tau(\Pi_1, s_1)$ nor $s_5 \in \tau(\Pi_2, s_1)$. Hence, $s_5 \notin \tau(\Pi_1, s_1) \cup \tau(\Pi_2, s_1) \supseteq (\tau(\Pi_1, s_1) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s_2) \setminus \sigma(\Pi_1))$, and $\tau(\Pi_1 \cup \Pi_2, s_1) \neq (\tau(\Pi_1, s_1) \setminus \sigma(\Pi_2)) \cup (\tau(\Pi_2, s_1) \setminus \sigma(\Pi_1))$.

□

1.7 Abstraction

For a purpose of one latter theorem, a definition of a *projection abstraction* of planning domain's *state space* is put forward. By a state space we basically mean an directed multi-graph, where vertices are the states of the domain and edges are the transitions between them induced by applications of available actions.

Definition 1.39. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. A **state space** of the domain $\mathcal{D}$ is edge-labelled directed multi-graph $\langle \mathcal{S}(\mathcal{V}), \{(s, a, s') \mid s \in \mathcal{S}(\mathcal{V}), a \in \mathcal{A}, s' \in \delta(s, a)\}\rangle$.

A function that puts focus only on some of the variables of the domain is called *projection mapping*. It maps a state of the domain to some sub-state which has the same facts as the mapped state, but only if the variable of that fact is in that focus of the function. Following definition is a formal definition of this idea.

Definition 1.40. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $\mathcal{V}' \subseteq \mathcal{V}$ be a set of variables. We say that $\mathcal{M}_{\mathcal{V}'} : \mathcal{S}(\mathcal{V}) \to \mathcal{S}(\mathcal{V}')$ is a **projection mapping**, where for any $s \in \mathcal{S}(\mathcal{V})$ holds $\mathcal{M}_{\mathcal{V}'}(s) = \{(v, x) \mid (v, x) \in s, v \in \mathcal{V}'\}$.

Finally, an *abstraction* of the state space can be defined. Informally, it is a simpler state space, which focuses only on some variables of the domain, while preserving all previously existing directed edges. It can be seen as an operation on the former state space which merges some states which have the same values of given variables while keeping all of the edges that has been previously present in the multi-graph.

Definition 1.41. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $\mathcal{V}' \subseteq \mathcal{V}$ be a set of variables. A **projection abstraction** for the domain $\mathcal{D}$ with a respect to the set $\mathcal{V}'$ is a edge-labelled directed multi-graph $\langle \{\mathcal{M}_{\mathcal{V}'}(s) \mid s \in \mathcal{S}(\mathcal{V})\}, \{(\mathcal{M}_{\mathcal{V}'}(s), a, \mathcal{M}_{\mathcal{V}'}(s')) \mid s \in \mathcal{S}(\mathcal{V}), a \in \mathcal{A}, s' \in \delta(s, a)\}\rangle$.

Notice, that the *projection abstraction* is a *homomorphism*. Due to this property, it will be utilized to prove a path non-existence, which will result into an important consequence of this work.

Now, that all theoretical necessaries are finally properly defined, we move to the main issue of this work—the *action reversibility*.

Chapter 2

Action Reversibility

In this chapter we elaborate on the *action reversibility*. For *classical planning*, this has been already studied multiple times. One of the first works on the *action reversibility* has been published by Eiter; Erdem; Faber (2008). It studies whether the effects of sequence of actions of any length can be undone. It has been investigated under more expressive action formalisms than FOND formalism is. A notion of so called inverse action, which is the action reversing the effects of another action, has been independently investigated in Koehler; Hoffmann (2000) and Chrpa; McCluskey; Osborne (2012). The work of Daum; Torralba; Hoffmann; Haslum; Weber (2016) on the *action undoability* has shown that it can be decided by *contingent planning*. Further research on the *action reversibility* has been done in the recent years. In the work of Morak; Chrpa; Faber; Fiser (2020) new notions of *reversibility* for *the classical planing*, called *S-reversibility* and *φ-reversibility*, were introduced; together with multiple complexity results concerning the variations of the *action reversibility*. The follow-up work of Chrpa; Faber; Morak (2021) introduces the *uniform* and the *universal action reversibility*. It also proves a theorem describing a special case of tiuniversal uniform reversibility existence when several conditions are met. A further research of Chrpa; Faber; Fiser; Morak (2020) and Faber; Morak; Chrpa (2021) has shown an ability of *answer set programming* and *epistemic logic programs* to determine the action reversibility.

A differentiation of this works from the recent ones lies in used formalism. The recent work on the *action reversibility* focuses on *classical planning* and its STRIPS and SAS$^+$ formalisms. The *action reversibility* researched in this work is examined under the non-determinism of FOND formalism and is heavily inspired by the work of Morak; Chrpa; Faber; Fiser (2020) and Chrpa; Faber; Morak (2021).

2.1 S-reversibility

The *S-reversibility* is a notion introduced by Morak; Chrpa; Faber; Fiser (2020). According to the work, an action is called *S-reversible* if and only if its effects can be undone by some sequence of actions when the action is applied in any state of the set of states S in which it is applicable. Note, that sequences corresponding to two distinct states from S can differ arbitrarily; it does not restrict the sequence in any way; and also for each state from the set S, there can be multiple sequences reversing the effects.

The presented notion describes a one possible approach to the general *action reversibility*.

However, it is not directly transferable to this work, since it cannot handle the stochasticity of action effects. In order to a non-deterministic action be reversible, all of its possible stochastic effects need to be addressed.

To address non-deterministic action effects, we exploit the idea presented in the work of Cimatti; Pistore; Roveri; Traverso (2003). The *weak*, *strong cyclic* and *strong acyclic planning* describes three levels of the solution quality. These levels can be reused when we want to define levels of the quality of the *action reversibility*. Due to that, we combine a work of Cimatti; Pistore; Roveri; Traverso (2003) and of Morak; Chrpa; Faber; Fiser (2020) and introduce *weak S-reversibility* of a non-deterministic action; a combination of the *weak solutions* and of the *S-reversibility*. Informally, we call the action *weakly reversible* when the action's effects may be undone, if the "correct" stochastic action effects happen.

Definition 2.1. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$ and $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$. The action a is called **weakly *S*-reversible** in the domain $\mathcal{D}$ if and only if for each state s from S in which the action a is applicable exists a policy Π for the domain $\mathcal{D}$ such that for each $s' \in \delta(s, a)$ holds $s \in \tau(\Pi, s')$.

An opposite of *weak S-reversibility* is an *action S-irreversibility*. It describes a situation, where there is no possibility of undoing the action's effects. It behaves basically the same as *S-reversibility*, but in an opposite point of view. This work defines multiple classes of the *action irreversibility* as well; often without any clarification. It is considered a self-explanatory, as it invariably mimics the *S-reversibility*.

Definition 2.2. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$ and $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$. The action a is called ***S*-irreversible** in the domain $\mathcal{D}$ if and only if for each state s from S in which the action a is applicable does not exist a policy Π for the domain $\mathcal{D}$ such that for any $s' \in \delta(s, a)$ holds $s \in \tau(\Pi, s')$.

Similarly to the *weak S-reversibility*, we define the *strong S-reversibility*. This work does not distinguish whether the *strong solution* is acyclic or not. We call the action *strongly reversible* if there is no way stochastic action effects may cause that the action effects cannot be undone.

Definition 2.3. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$ and $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$. The action a is called **strongly *S*-reversible** in the domain $\mathcal{D}$ if and only if for each state s from S in which the action a is applicable exists a policy Π for the domain $\mathcal{D}$ such that for each $s' \in \delta(s, a)$ holds $\{s\} = \tau(\Pi, s')$.

2.2 Uniform Reversibility

One interesting notion of the *action reversibility* is the *uniform action reversibility* presented by Chrpa; Faber; Fiser; Morak (2020). It is a restriction of former definitions of S-reversibility so that there is a common policy for all states of S.

Definition 2.4. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$, $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$ and Π the policy for the domain $\mathcal{D}$. The action a is called **weakly uniformly *S*-reversible** in the domain $\mathcal{D}$ by the policy

Π if and only if for each state s from S in which the action a is applicable holds that for each $s' \in \delta(s, a)$ holds that $s \in \tau(\Pi, s')$.

Definition 2.5. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$, $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$ and Π the policy for the domain $\mathcal{D}$. The action a is called **strongly uniformly *S*-reversible** in the domain $\mathcal{D}$ by the policy Π if and only if for each state s from S in which the action a is applicable holds that for each $s' \in \delta(s, a)$ holds that $\{s\} = \tau(\Pi, s')$.

Analogous notion for the classical planning is defined in the work of Morak; Chrpa; Faber; Fiser (2020) and is called a *reverse plan*. We name the notion concerning non-deterministic planning a *reverse policy*.

Definition 2.6. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and a be an action over the set of variables $\mathcal{V}$. The policy Π for the domain $\mathcal{D}$ is called a **weak (resp. strong) *S*-reverse policy** for the action a if and only if the action a is weakly (resp. strongly) uniformly S-reversible by the policy Π. The notions (concerning later defined φ-reversibility and universal reversibility) of the **weak (resp. strong) φ-reverse policy** and the **weak (resp. strong) universal reverse policy** are defined analogously.

The notion of *uniform S-irreversibility* is defined as well. However, it is some sense the weakest claim among the variants of *action irreversibility* presented in this work and it is not that interesting as the other variants.

Definition 2.7. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, a be an action over the set of variables $\mathcal{V}$, $S \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$. The action a is called **uniformly *S*-irreversible** in the domain $\mathcal{D}$ if and only if does not exist a policy Π for the domain $\mathcal{D}$ such that for each state s from S in which the action a is applicable holds that for each $s' \in \delta(s, a)$ holds that $s \in \tau(\Pi, s')$.

Given former definitions, we can already infer some properties related to the action reversibility.

Obviously, if the action is (uniformly) reversible for all states of the set S, it is also (uniformly) reversible for some states of the set S.

Lemma 2.1. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $S, S' \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$, such that $S' \subseteq S$. If the action $a \in \mathcal{A}$ is weakly (resp. strongly) (uniformly) S-reversible, then the action a is weakly (resp. strongly) (uniformly) S'-reversible. The same holds for the notion of S-irreversibility as well.*

Proof. Trivial, from the definition of the weak and strong S-reversibility and from the definition of the S-irreversibility. □

Alike situation concerns a situation where the are two sets in which some action is S-reversible. If we care about the plain *S-reversibility*, then the action is intuitively reversible in states of both sets. But, if we care about universal S-reversibility, we also have to deal with reverse plans. If the action is uniformly reversible in both sets and reverse plans are the same, then the plan is also a reverse plan of the union. Analogous claims hold for the action S-irreversibility as well.

Lemma 2.2. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action, $S_1 \subseteq \mathcal{S}(\mathcal{V})$ and $S_2 \subseteq \mathcal{S}(\mathcal{V})$ be a set of states of the domain $\mathcal{D}$ and Π be a policy for the domain $\mathcal{D}$. If the action a is weakly (resp. strongly) S_1-reversible and weakly (resp. strongly) S_2-reversible, then the action a is weakly (resp. strongly) $(S_1 \cup S_2)$-reversible. If the policy Π is a weak (resp. strong) S_1-reverse and weak (resp. strong) S_2-reverse policy, then the policy Π is a weak (resp. strong) $(S_1 \cup S_2)$-reverse policy for the action a. The same holds for the notion of S-irreversibility as well.*

Proof. Trivial, from the definition of the weak and strong (uniform) S-reversibility and from the definition of the (uniform) S-irreversibility. □

If we examine a definition of both uniform and regular S-reversibility, we say that the action is considered reversible in each state where it is not applicable. The action a is also considered irreversible in such state. This is formally captured in the corollary 2.3.

Corollary 2.3. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $S \subseteq \{s \mid s \in \mathcal{S}(\mathcal{V}), pre(a) \nsubseteq s\}$ be a set of states of the domain $\mathcal{D}$ and $a \in \mathcal{A}$ be an action. The action a is uniformly S-reversible by any policy Π.*

Proof. Trivial, from definitions 2.4 and 2.5. □

A more interesting observation regarding S-reverse policies is related to corollary 1.2. Once the policy has to revert action effects in some state in which the action is applicable, then that state cannot be related in that policy. The observation is intuitive, yet it has another important consequence we present later in this chapter.

Proposition 2.4. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, Π be a policy for the domain $\mathcal{D}$ and $a \in \mathcal{A}$ be an action. If Π is a weak (resp. strong) S-reverse policy for the action a, then $\{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi) = \emptyset$.*

Proof. Proposition 2.4 will be proven by a contradiction. Assume that $\{s' \mid s' \in S, pre(a) \subseteq s'\} \cap \sigma(\Pi) \neq \emptyset$. Then, there is a state $s \in \{s' \mid s' \in S, pre(a) \subseteq s'\} \cap \sigma(\Pi)$. According to the assumption and since $\{s\} \subseteq S$, we can utilize lemma 2.1. We get that the action a is weakly (resp. strongly) uniformly $\{s\}$-reversible by the policy Π. But, as s is also in $\sigma(\Pi)$, s cannot be a terminal state of the policy Π (see corollary 1.2). Therefore for each $s' \in \delta(s, a)$ holds that $s \notin \tau(\Pi, s')$. This is in a contradiction with the assumption of Π being a weak (resp. strong) S-reverse policy for the action a. □

Another phenomena worth of an examination is whether it is possible to merge a weak or strong S-reverse policy fort some action with another arbitrary policy. It is expected that this procedure will work for the weak reversibility without many limitations, since the weak reversibility is more or less only about path existence. However, one must be cautious about proposition 2.4. Intuitively, if another policy contains a rule for a state from S in which the action a is applicable, then the policy of union of both policies would leave the state once reached and it would not be terminal anymore.

Theorem 2.5. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action, Π_1 be a weak S-reverse policy for the action a and Π_2 be a policy for the domain $\mathcal{D}$. The policy $\Pi_1 \cup \Pi_2$ is a weak S-reverse policy for the action a if and only if $\{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_2) = \emptyset$.*

Proof.

The left-to-right implication will be proven by a contra-position of the implication. Assume, that there is some state $s' \in \{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_2)$. Then, as $\sigma(\Pi_2) \subseteq \sigma(\Pi_1 \cup \Pi_2)$ (see lemma 1.1), $s' \in \{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_1 \cup \Pi_2)$. Ergo (see proposition 2.4), the policy $\Pi_1 \cup \Pi_2$ cannot be a weak S-reversible policy for the action a.

Now, the right-to-left implication will be proven directly. Since addition of any pair to the policy does not break reachability of any state which was previously reachable, all terminal states of the policy Π_1 with a respect to any initial state are reachable by $\Pi_1 \cup \Pi_2$ as well. From the assumption, $\{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_2) = \emptyset$. Together with the fact that $\{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_1) = \emptyset$ (see proposition 2.4), (while utilizing lemma 1.1) we get that $\{s \mid s \in S, pre(a) \subseteq s\} \cap \sigma(\Pi_1 \cup \Pi_2) = \emptyset$. Finally, we conclude that all states of $\{s \mid s \in S, pre(a) \subseteq s\}$ are necessarily terminal with a respect to some starting state for the policy $\Pi_1 \cup \Pi_2$ (see the definition 1.33). Hence, the policy $\Pi_1 \cup \Pi_2$ is a weak $\{s \mid s \in S, pre(a) \subseteq s\}$-reverse policy for the action a and, due to corollary 2.3 and lemma 2.2, Π is S-reverse policy for the action a as well. □

Since every strong S-reverse policy is also a weak S-reverse policy, the same conditions apply when we investigate the analogous property for strong S-reverse policies. However, the conditions of the theorem 2.5 are not enough. A situation that may violate a property of strong reversibility is when the states of an arbitrary policy we want to add coincides with the states of the strong S-reverse policy. Such coincidence may cause that the agent is lead to a different terminal state, because in the incident state a inappropriate action may be selected. As a consequence, the set of terminal states contain a different state other than the target one; and this may violate the condition of strong reversibility.

Theorem 2.6. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action, Π_1 be a strong S-reverse policy for the action a and Π_2 be a policy for the domain $\mathcal{D}$. If $\sigma(\Pi_2) \cap \{s \mid s \in S, pre(a) \subseteq s\} = \emptyset$ and $\sigma(\Pi_1) \cap \sigma(\Pi_2) = \emptyset$, then $\Pi_1 \cup \Pi_2$ is a strong S-reverse policy for the action a.*

Proof.

We have to show that for each state $s \in S$ in which the action a is applicable it holds that for each initial state $s' \in \delta(s, a)$ holds that $\{s\} = \tau(\Pi_1 \cup \Pi_2, s')$.

Since $\sigma(\Pi_2) \cap \{s \mid s \in S, pre(a) \subseteq s\} = \emptyset$, we can utilize the theorem 2.5 as all its conditions are satisfied (each strong S-reverse policy is also a weak S-reverse policy). Therefore, we know that $\Pi_1 \cup \Pi_2$ is a weak S-reverse policy for the action a, which means that for each state $s \in S$ in which the action a is applicable, $\{s\} \subseteq \tau(\Pi_1 \cup \Pi_2, s')$ for each state $s' \in \delta(s', a)$. Hence, what is left to show is that there is no s'' such that $s'' \neq s$ and $s'' \in \tau(\Pi_1 \cup \Pi_2, s')$.

From the assumption of Π_1 being strong S-reverse policy for the action a, we get that $\{s\} = \tau(\Pi_1, s')$. The set of reachable states with a policy Π_1 from the state s' is a subset of $\sigma(\Pi_1) \cup \tau(\Pi_1, s')$. If we add some state-action pair (s''', a''') to the policy Π_1, then policy $\Pi_1 \cup (s''', a''')$ still achieves all states that were achieved by Π_1. Also, if $s''' \in \sigma(\Pi_1) \cup \tau(\Pi_1, s')$, then it reaches states of $\delta(s''', a''')$ as well. If $s''' \notin \sigma(\Pi_1) \cup \tau(\Pi_1, s')$, then the set of reachable states remains the same.

Due to the assumption of $\sigma(\Pi_1) \cap \sigma(\Pi_2) = \emptyset$, for any state-action pair $(s_{\Pi_2}, a_{\Pi_2}) \in \Pi_2$, $s_{\Pi_2} \notin \sigma(\Pi_1)$. Also, as $\sigma(\Pi_2) \cap \{s \mid s \in S, pre(a) \subseteq s\} = \emptyset$, $s_{\Pi_2} \neq s$, and therefore

$s_{\Pi_2} \notin \tau(\Pi_1, s')$. These to facts together proves that the sets of reachable states with either policy Π_1 or $\Pi_1 \cup \Pi_2$ from the state s' are the same; and equal to $\{s\}$.

Ergo, $\Pi_1 \cup \Pi_2$ is a strong S-reverse policy for the action a.

□

If the second policy is not arbitrary, but it is also some reverse policy for which conditions of theorems 2.5 or 2.6 hold symmetrically, then we can unify the policies together as well; and furthermore, we can claim that the action a is uniformly $(S_1 \cup S_2)$-reversible by the policy $\Pi_1 \cup \Pi_2$.

Theorem 2.7. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $S_1, S_2 \subseteq \mathcal{S}(\mathcal{V})$ be a set of states for the domain $\mathcal{D}$ and $a \in \mathcal{A}$ be an action. If for the action a there exists a weak S_1-reverse policy Π_1 and a weak S_2-reverse policy Π_2 such that $\{s \mid s \in S_1, pre(a) \subseteq s\} \cap \sigma(\Pi_2) = \emptyset$ and $\{s \mid s \in S_2, pre(a) \subseteq s\} \cap \sigma(\Pi_1) = \emptyset$, then the policy $\Pi_1 \cup \Pi_2$ is a weak $(S_1 \cup S_2)$-reverse policy for the action a.*

Proof. From theorem 2.5 we get that $\Pi_1 \cup \Pi_2$ is a weak S_1-reverse and weak S_2-reverse policy for the action a. Hence (see lemma 2.2), it is also a weak $(S_1 \cup S_2)$-reverse policy for the action a . □

Theorem 2.8. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $S_1, S_2 \subseteq \mathcal{S}(\mathcal{V})$ and $a \in \mathcal{A}$ be an action. If for the action a exists a strong S_1-reverse policy Π_1 and a strong S_2-reverse policy Π_2 such that $\{s \mid s \in S_1, pre(a) \subseteq s\} \cap \sigma(\Pi_2) = \emptyset$, $\{s \mid s \in S_2, pre(a) \subseteq s\} \cap \sigma(\Pi_1) = \emptyset$ and $\sigma(\Pi_1) \cap \sigma(\Pi_2) = \emptyset$, then the policy $\Pi_1 \cup \Pi_2$ is a strong $(S_1 \cup S_2)$-reverse policy for the action a.*

Proof. From theorem 2.6 we get that $\Pi_1 \cup \Pi_2$ is a strong S_1-reverse and strong S_2-reverse policy for the action a. Hence (see lemma 2.2), it is also a strong $(S_1 \cup S_2)$-reverse policy for the action a. □

The preceding theorems provide instructions on how policies can be merged together to construct policies able to handle multiple states of S-reversibility. It can be argued that a more general policy is more helpful and practical than multiple specialized policies. With the use of preceding theorems, we can build these general policies from smaller ones, even from trivial one-state S-reversibilities (if the conditions hold). If the policy is general as it can be, meaning it works in any state of the domain, then it is the best in the term of multi-usability. Naturally, this level of multi-usability is interesting and wanted. The class of reversibility focusing on this level is defined in the next section.

2.3 Universal Reversibility

The *universal reversibility* is a specific situation of the *S-reversibility*. The specificity lies in the content of the S set. In a case of the *universal reversibility*, we require the S set be equal to the $\mathcal{S}(\mathcal{V})$. This, informally, requires the action to be reversible (uniformly or arbitrary) in all states of the domain.

Note that we define the *universal reversibility* for all possible combinations of so far defined reversibility.

Definition 2.8. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and a be an action over the set of variables $\mathcal{V}$. The action a is called **weakly (resp. strongly) universally (uniformly) reversible** if and only if the action a is weakly (resp. strongly) (uniformly) $(\mathcal{S}(\mathcal{V}))$-reversible.

Definition 2.9. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and a be an action over the set of variables $\mathcal{V}$. The action a is called **universally irreversible** if and only if the action a is $(\mathcal{S}(\mathcal{V}))$-irreversible.

One consequence of proposition 2.4 concerns universal uniform reversibility. The following proposition is presenting a special case of the theorem, when the set S is equal to the set $\mathcal{S}(\mathcal{V})$. Then, the situation of proposition 2.4 can be simplified to the fact that the universal reverse policy can contain only the states in which the action we want to reverse is not applicable.

Lemma 2.9. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $a \in \mathcal{A}$ be an action. If there exists a weak (resp. strong) universal reverse policy Π for the action a, then $\sigma(\Pi) \subseteq \{s \mid s \in \mathcal{S}(\mathcal{V}), pre(a) \nsubseteq s\}$.*

Proof. Trivial, from definitions 2.6, 2.8 and proposition 2.4. □

We consider this lemma of high importance and see it as an essential and strong necessary condition for the universal reverse policy existence.

As the latter experiments show, the amount of universal reverse policies were proven only occasionally (except for one domain, where around 60% of actions was strongly universally uniformly reversible). However, the question whether the actions are weakly or strongly universally uniformly reversible remains mostly unclear. At least for the weak variant, due to an absence of theoretical tools. The strong universal uniform reversibility was decided for a significant part of the tested actions. The more detailed findings can be seen in chapter 4. Therefore, the impact of lemma 2.9 remains unclear as well and the its strictness remains a hypothesis.

2.4 φ-reversibility

An equivalent alternative to the *S-reversibility* may be a *φ-reversibility.* It provides a more compact way of representing *S-reversibility*, but with the use of *propositional logic* and its formulae. It has been indicated, that there is a correspondence between the *S-reversibility* and the *φ-reversibility.* If the *S-reversibility* and the *φ-reversibility* describe the same ability of an action to reverse its effects, then the formula φ is necessarily satisfied by each state (truth assignment) of that set.

Definition 2.10. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, φ be a propositional formula over the domain $\mathcal{D}$ and $a \in \mathcal{A}$ be an action. The action a is called **(weakly, resp. strongly) (uniformly) φ-reversible** if and only if the action a is (weakly, resp. strongly) (uniformly) S-reversible, such that $S = \{s \mid s \in \mathcal{S}(\mathcal{V}), s \models \varphi\}$.

Definition 2.11. Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, φ be a propositional formula over the domain $\mathcal{D}$ and $a \in \mathcal{A}$ be an action. The action a is called **φ-irreversible** if and only if the action a is S-irreversible, such that $S = \{s \mid s \in \mathcal{S}(\mathcal{V}), s \models \varphi\}$.

As argued, the notion of φ-reversibility is analogous to the notion of S-reversibility. As a consequence, one can expect that corollaries, lemmata and theorems of S-reversibility will hold for φ-reversibility as well. The following proposition is an analogy of lemma 2.1. In a contrast to lemma 2.1, this corollary is not that intuitive and obvious as it is for the notion of S-reversibility.

Lemma 2.10. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action, φ and φ' be a propositional formula over the domain $\mathcal{D}$ such that $\varphi' \models \varphi$. If the action a is a weakly (resp. strongly) (uniformly) φ-reversible, then the action a is weakly (resp. strongly) (uniformly) φ'-reversible.*

Proof.

Assume that the action a is weakly (resp. strongly) (uniformly) φ-reversible. By a definition, it means that it is weakly (resp. strongly) (uniformly) S-reversible, such that $S = \{s \mid s \in \mathcal{S}(\mathcal{V}), s \models \varphi\}$.

In order to the action a be weakly (resp. strongly) (uniformly) φ'-reversible, it has to be weakly (resp. strongly) (uniformly) S'-reversible, such that $S' = \{s' \mid s' \in \mathcal{S}(\mathcal{V}), s' \models \varphi'\}$.

Consider a state $s' \in S'$. It holds that $s' \models \varphi'$. Because $\varphi' \models \varphi$, we get $s' \models \varphi$ (due to the transitivity of semantic consequence). But this immediately means that $s' \in S$ for any state $s' \in S'$; in other words, $S' \subseteq S$.

Hence, lemma 2.1 proves the action a being weakly (resp. strongly) (uniformly) φ'-reversible.

□

2.5 Complexity

Several complexity results for the actions of *classical planning* has been proven in Morak; Chrpa; Faber; Fiser (2020). We now present a few of them. According to the work, the problem of the universal action reversibility of the deterministic action, as well as of action φ-reversibility, belong to the set of *PSPACE-complete* problems. Since the existence of a plan for *classical planning* also belongs to the set of *PSPACE-complete* problems (Bylander, 1994), we can then solve reversibility of the action by *classical planning* and vice versa. Also, both problems of the universal uniform reversibility of the deterministic action, as well as of uniform action φ-reversibility, are *PSPACE-hard*. From this it follows that any *classical planning* task can be decided by the (deterministic) universal uniform action reversibility or by the (deterministic) uniform action φ-reversibility. The work then examines a constrained case of the action reversibility such that plans are polynomially bounded. For a various, not presented, complexity results concerning the deterministic action reversibility consult a work of Morak; Chrpa; Faber; Fiser (2020).

Unfortunately, so far, there are no complexity results for the non-deterministic actions of the FOND planning. We plan to investigate them formally in the following work. It can argued that the complexity of *non-deterministic action reversibility* may follow the same pattern as the deterministic action reversibility did. This would imply the complexity of non-uniform *non-deterministic action* reversibility be the same as the theoretical complexity of FOND planning; and that is *EXPTIME-complete* problem (Littman, 1997; Rintanen, 2004). The complexity of *uniform non-deterministic action reversibility* would be expected to be at least that hard. Yet, we emphasize that these claims are mere hypotheses and should not be taken as granted in any way.

Chapter 3

Determining Action Reversibility or Irreversibility

In this chapter, we present two algorithms designed for a retrieval of a reverse plan or a reverse policy and to decide the action φ-reversibility in general. The first is taken from the work of Chrpa; Faber; Morak (2021) to demonstrate the idea behind it and the second is novel adaptation of the first one, to deal with non-deterministic actions. In between the algorithms, a brief section elaborating on the action irreversibility if some determinization is not reversible is present. The algorithms are followed by a section dedicated to the weak and strong universal uniform reversibility and their compilations to classical and FOND planning task. The chapter concludes with a scheme describing a process by which the action reversibility was evaluated.

3.1 φ-reversibility of Deterministic Actions

We start with an algorithm of Chrpa; Faber; Morak (2021). It non-deterministically decides the uniform φ-reversibility of a deterministic action in SAS$^+$ formalism. In addition, if there is a φ-reverse plan, it is found and returned together with the formula φ, describing the set of states in which the action is uniformly φ-reversible. The algorithm is presented in this work as the algorithm 1.

It starts with an initialisation of two variable assignments, namely I, S; and with an empty plan π. The set S represents a variable assignment which, at the beginning, necessarily holds after the application of the action, or, later, which holds after a sequential application of selected actions. The set I stores facts that are required in any state of the algorithm such that the applied actions are guaranteed to be applicable. In other words, it contains facts that must be true before the reversed action and its reverse plan is applied in order to the reversed action and each action of the reverse plan to be applicable.

Then, the algorithm goes through all available actions and non-deterministically selects a suitable action which is potentially applicable (meaning there is no variable such that the value in S is different than in preconditions). If such action does not exists, then the action is universally irreversible. Otherwise, the suitable action is non-deterministically selected and denoted as a'. The preconditions of the selected action which are not yet satisfied are added to the set I, selected action is added to the future reverse plan and the set S is updated with facts that holds after the action a' is applied.

Algorithm 1: Uniform φ-reversibility of a deterministic action a.

Input : a set of deterministic actions $\mathcal{A}$, a deterministic action $a \in \mathcal{A}$
Output: a propositional formula φ, an uniform φ-reverse plan π

1 $I \leftarrow pre(a)$;
2 $S \leftarrow ha(a)$;
3 $\pi \leftarrow \langle\rangle$;
4 **while** $I \nsubseteq S$ **do**
5   non-deterministically choose $a' \in \mathcal{A}$ such that $\nexists v \in vars(pre(a')) \cap vars(S) : pre(a')[v] \neq S[v]$;
6   **if** a' does not exist **then**
7   **return** $\bot, \langle\rangle$;
8   **end**
9   $I \leftarrow I \cup \{(v, x) \mid (v, x) \in pre(a'), v \notin vars(S)\}$;
10   $\pi \leftarrow \pi^\frown\langle a' \rangle$;
11   $S \leftarrow ha(a') \cup \{(v, x) \mid (v, x) \in S, v \notin vars(ha(a'))\}$;
12 **end**
13 $\varphi \leftarrow \Psi(S)$;
14 **return** φ, π;

Figure 3.1: Algorithm for decision of uniform φ-reversibility of a deterministic action

This process is repeated until the variable assignment representing a current sub-state S is a superset of the required facts I. In such situation, the reached variable assignment satisfies all necessities of the action which we want to reverse (its preconditions) and of each action in the found reverse plan π (their preconditions not achieved by the preceding actions).

Also note, that if the action is trivially reversible (by an empty plan), then no action is added.

Once the *while*-cycle has ended, a formula φ is constructed. It is evident that application of the action a followed by the reverse plan π in a variable assignment equal to the set of preconditions of the reversed action results in a variable assignment, where exactly facts of S hold. This implies that the action is φ-reversible by the plan π.

Theorem 3.1. *Let $\mathcal{D}^d = (\mathcal{V}, \mathcal{A}^d)$ be a planning domain. The algorithm 1 returns for an action a and the set of actions $\mathcal{A}$ ordered pair (φ, π) such that a is uniformly φ-reversible by the plan π.*

Proof. See Chrpa; Faber; Morak (2021). □

Before we move to the next section, we want to briefly note that in the original paper the pseudo-code is incorrect. The line 8 in the original work should be: $\forall v \in vars(pre(a')) \setminus vars(S) : I[v] \leftarrow pre(a')[v]$.

3.2 Universal Irreversibility

The following lemma is not exactly about universal irreversibility, but it is tightly bounded to it (or at least the part concerning the weak reversibility). It states if any determinization of some action is not weakly (resp. strongly) $\{s\}$-reversible, then the action cannot be weakly (resp. strongly) $\{s\}$-reversible as well.

Theorem 3.2. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action and s be an state of the domain $\mathcal{D}$. If for any determinization a_e^d there is no policy Π by which the determinization a_e^d is weakly (resp. strongly) uniformly $\{s\}$-reversible, then the action a is not weakly (resp. strongly) uniformly $\{s\}$-reversible by any policy.*

Proof. Let a_e^d be an determinization of the action a such that there is no policy Π by which the determinization a_e^d is weakly (resp. strongly) $\{s\}$-reversible. This means that there is no policy which reaches the state s from the state $\gamma(s, a_e^d)$ satisfying the conditions of weak (resp. strong) policies at least for the determinization a_e^d. From the definition of weak (resp. strong) policy and since $\gamma(s, a_e^d) \in \delta(s, a)$, there is at least one state of $\delta(s, a)$ from which we cannot reach the state s which a policy that satisfies the conditions of weak (resp. strong) policies. Hence, the action a is not weakly (resp. strongly) $\{s\}$-reversible by any policy. $\square$

A justification why this section is called $\{s\}$-irreversibility even though the lemma is not exactly about the action irreversibility is that the lemma is an important necessary condition for the action's weak (resp. strong) reversibility. Once it is fulfilled, the action cannot satisfy the conditions. This claim is utilized later, when we decide universal uniform weak (resp. strong) reversibility.

The reason why the S-irreversibility is not an exact opposite of S-reversibility comes from their definitions and it was also outline by corollary 2.3. For any state in which the action is not applicable, the action is considered both $\{s\}$-reversible and $\{s\}$-irreversible. But once the action is applicable in the state, notions of S-reversibility and S-irreversibility becomes mutually exclusive.

As a consequence, we will be able to find any state in which the action is applicable such that the action is $\{s\}$-irreversible in it, we prove the action not being universally weakly reversible. The similar argument hold also for notion of universal strong reversibility, as theorem 3.2 can be utilized.

The second claim this section lays out also utilizes theorem 3.2. It is a one of four important theorems exploited during the gathering of empirical evidence of this work. It is an adaptation of the proposition in the work of Chrpa; Faber; Morak (2021) for a subset of deterministic actions. Informally, if some effect of any determinization cannot be undone, then the action is universally irreversible. Furthermore, this holds also for the abstracted state space where we focus only on variables on which the determinization operates.

Theorem 3.3. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, $a \in \mathcal{A}$ be an action. If for any determinization a_e^d there is no path in the projection abstraction of the domain $\mathcal{D}$ with a respect to the set of variables $vars(a_e^d)$ from the vertex $ha(a_e^d)$ to any vertex s, $pre(a_e^d) \subseteq s$, then the action a is universally irreversible.*

Proof. If for any determinization a_e^d there is no path in the projection abstraction with a respect to the set of variables on which the determinization operates, then, since the projection abstraction is homomorphic, there is no path in the non-abstracted state space as well. Therefore, the determinization a_e^d is universally irreversible, since there is at least one fact which cannot be reversed. Now, the utilization of theorem 3.2 proves that the action a is universally irreversible. $\square$

The advantage of theorem 3.3 is that we can verify the path existence in the abstracted space by *classical planning.* The compilation is straightforward. Stochastic actions are replaced by their determinizations, the variables not in $vars(a_e^d)$ are ignored in both sets of variables and of actions, the initial state and the goal states are set as in the theorem. Then, if any of the compilations is unsolvable, the action a is universally irreversible.

3.3 Weak Uniform φ-reversibility

The algorithm 1 can be adapted to deal with non-deterministic actions.

An interesting situation occurs if the set $\mathcal{A}$ contains deterministic actions only, but the action whose reversibility we want to decide is non-deterministic. According to the definition of the weak uniform action reversibility, a policy reversing all determinizations of the action is needed. Due to the fact that any determinization is a deterministic action and $\mathcal{A}$ contains only deterministic actions as well, we can run algorithm 1 for deciding its weak φ-reversibility. If for any determinization the reversing fails, the determinization is proven universally irreversible. Hence (see theorem 3.2), the action is also universally irreversible. Otherwise, the action may be weakly uniformly φ-reversible.

The problem then lies in deciding whether we can combine formulae and policies together into a general formula φ and a policy Π. A one possible merge is described in the theorem 3.4.

Theorem 3.4. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $a \in \mathcal{A}$ be an action. The action a is weakly uniformly $(\bigwedge_{e \in \mathit{eff}(a)} \varphi_e)$-reversible by the policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$ if and only if each determinization a_e^d of the action a is weakly uniformly φ_e-reversible by Π_e.*

Proof.

Firstly, the right-to-left implication will be proven directly.

From the assumption, each determinization a_e^d is weakly uniformly φ_e-reversible by the policy Π_e. As $(\bigwedge_{e' \in \mathit{eff}(a)} \varphi_{e'}) \models \varphi_e$, lemma 2.10 proves that each determinization a_e^d is weakly uniformly $(\bigwedge_{e' \in \mathit{eff}(a)} \varphi_{e'})$-reversible by the policy Π_e.

If we show that each determinization a_e^d is weakly uniformly $(\bigwedge_{e \in \mathit{eff}(a)} \varphi_e)$-reversible by the policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$, then we will be able to state that the action a is weakly uniformly $\bigwedge_{e \in \mathit{eff}(a)} \varphi_e$-reversible by the policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$. This is due to the reason that, for any $s \in \mathcal{S}(\mathcal{V})$ in which the action a is applicable and $s \models (\bigwedge_{e \in \mathit{eff}(a)} \varphi_e)$, the same policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$ is able to weakly revert action's effects in the state $s' = \gamma(s, a_e^d)$; and due to the fact that $\delta(s, a) = \bigcup_{e' in \mathit{eff}(a)} \gamma(s, a_{e'}^d)$.

Denote the set $\{s \mid s \in \mathcal{S}(\mathcal{V}), s \models \varphi_e\}$ corresponding to a weak uniform φ_e-reversibility of the determinization a_e^d as S_e.

In order to utilize theorem 2.5, we need to show that $\sigma(\bigcup_{e' \in \mathit{eff}(a), e' \neq e} \Pi_{e'}) \cap \{s \mid s \in \bigcap_{e' \in \mathit{eff}(a)} S_{e'}, \mathit{pre}(a) \subseteq s\} = \emptyset$. From an iterative usage of lemma 1.1, we get $\sigma(\bigcup_{e' \in \mathit{eff}(a), e' \neq e} \Pi_{e'}) = \bigcup_{e' \in \mathit{eff}(a), e' \neq e} \sigma(\Pi_{e'})$. For any such $\Pi_{e'}$ holds that $\sigma(\Pi_{e'}) \cap \{s \mid s \in \bigcap_{e' \in \mathit{eff}(a)} S_{e'}, \mathit{pre}(a) \subseteq s\} = \emptyset$, as $\Pi_{e'}$ is a weak $(\bigcap_{e' \in \mathit{eff}(a)} S_{e'})$-reverse policy for the determinization a_e^d (see proposition 2.4). And finally, from the distributivity of the set intersection,

$\sigma(\bigcup_{e' \in \mathit{eff}(a), e' \neq e} \Pi_{e'}) \cap \{s \mid s \in \bigcap_{e' \in \mathit{eff}(a)} S_{e'}, \mathit{pre}(a) \subseteq s\} = \emptyset$, which means that theorem 2.5 can be utilized.

We have shown that each determinization a^d_e is weakly uniformly $(\bigcap_{e' \in \mathit{eff}(a)} S_{e'})$-reversible by the policy $\bigcup_{e' \in \mathit{eff}(a)} \Pi_{e'}$ and together with a claims of the second paragraph of this proof we proved action a being weakly uniformly $(\bigwedge_{e \in \mathit{eff}(a)} \varphi_e)$-reversible by the policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$.

Secondly, the left-to-right implication will directly.

Assume, that the plan Π is a weak φ-reverse plan for the action a. Let $\varphi = \varphi_e$ and $\Pi = \Pi_e$ for each determinization a^d_e of the action a. Clearly, the policy Π is the weak φ-reverse plan for the determinization a^d_e.

□

So far, we were able to exploit the algorithm 1 to determine the weak uniform φ-reversibility of the action a, since the set contained deterministic actions only. However, this will not work when there is at least one non-deterministic action in the set $\mathcal{A}$. Yet, it is not difficult to deal with these actions as well. If a non-deterministic action is selected for the application (as on the line 5 of the algorithm 1), a "suitable" determinization is non-deterministically selected and is used instead of the previously selected stochastic action in the say way as the deterministic action would be used.

With this approach, the non-trivial weak φ-reversibility can be "decided". If there is no combination of formulae that the resulting formula $\bigwedge_{e \in \mathit{eff}(a)} \varphi_e$ is not contradiction and it is not semantic consequence of the states in which the reverted action is inapplicable only, then the non-deterministic action a is universally irreversible (with a respect to the set of non-deterministic actions $\mathcal{A}$).

This algorithm, focusing on the weak uniform φ-reversibility of the action a, is marked as algorithm 2.

Theorem 3.5. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain. The algorithm 2 returns for an action a and a set of actions $\mathcal{A}$ ordered pair (φ, Π) such that the action a is weakly uniformly φ-reversible by the policy Π.*

Proof. The algorithm 2 starts by running algorithm-1-like procedure for determining weak uniform φ-reversibility of all determinizations of the action a. If the algorithm fails to determine weak uniform φ-reversibility of any determinization, it means that the determinization is irreversible. Hence, thanks to theorem 3.2, the action a is irreversible as well and the contraction and empty policy is returned. Otherwise, all formulae φ_e and all policies Π_e proving weak uniform φ_e-reversibility of determinizations a^d_e are calculated and the theorem 3.4 is utilized and φ and Π (proving a weak uniform φ-reversibility of the action a) are returned. □

3.4 Universal Uniform Reversibility

We have partially adapted the theorem of Chrpa; Faber; Morak (2021), designed for SAS^+ formalism, concerning universal uniform reversibility, to work with non-deterministic action. By partially we mean that in a comparison to the claim of Chrpa; Faber; Morak

Algorithm 2: Weak uniform φ-reversibility of an action a.

Input : a set of actions $\mathcal{A}$, a action $a \in \mathcal{A}$

Output: a formula φ, a implicitly-defined weak uniform φ-reverse policy Π

foreach determinization a_e^d of the action a **do**
  $I_e \leftarrow pre(a_e^d)$;
  $S_e \leftarrow ha(a_e^d)$;
  $\Pi_e \leftarrow \emptyset$;
  while $I_e \nsubseteq S_e$ **do**
    non-deterministically choose $a' \in \mathcal{A}$ such that
     $\nexists v \in vars(pre(a')) \cap vars(S_e) : pre(a')[v] \neq S_e[v]$;
    if a' does not exist **then**
      return $\bot, \{\}$;
    end
    non-deterministically choose determinization $(a')_{e'}^d$ of the action a';
    $I_e \leftarrow I_e \cup \{(v, x) \mid (v, x) \in pre((a')_{e'}^d), v \notin vars(S_e)\}$;
    $\Pi_e \leftarrow \Pi_e \cup (\Psi(S_e), a')$;
    $S_e \leftarrow ha((a')_{e'}^d) \cup \{(v, x) \mid (v, x) \in S_e, v \notin vars(ha((a')_{e'}^d))\}$;
  end
  $\varphi_e \leftarrow \Psi(S_e)$;
end
$\varphi \leftarrow \bigwedge_{e \in eff(a)} \varphi_e$;
$\Pi \leftarrow \bigcup_{e \in eff(a)} \Pi_e$;
return φ, Π;

Figure 3.2: Algorithm for decision of weak uniform φ-reversibility of an action

(2021), we were able to prove only one implication of the equivalence, as the latter theorem 3.8 depicts. In general, the theorem of Chrpa; Faber; Morak (2021) states that if the variables in action's preconditions forms a superset of the variables of action's effects, then each reverse plan of the action can contain actions operating with the variables of the preconditions only. This property can then be further utilized to determine universal uniform reversibility reverse plans by planning in the abstracted state space of the domain; allowing to formalize it as *classical planning* task.

Theorem 3.6. *Let $\mathcal{D}^d = (\mathcal{V}, \mathcal{A}^d)$ be a SAS$^+$ planning domain such that $\forall v \in \mathcal{V}$: $|dom(v)| \geq 2$, and $a \in \mathcal{A}^d$ be an action such that $\bigcup_{e \in eff(a)} vars(e) \subseteq vars(pre(a))$. The action a is universally uniformly reversible if and only if there exists a S-reverse plan $\pi = \langle a_1, \ldots, a_n \rangle$ for the action a, such that the set $\{s \mid s \in S, pre(a) \subseteq s\}$ is not empty and $\forall i \in \mathbb{N}, 1 \leq i \leq n : vars(a_i) \subseteq vars(pre(a))$.*

Proof. See Chrpa; Faber; Morak (2021). □

We humbly note that theorem 3.6 incorrectly contains (universal) reverse plan in the original work. We believe the reason was an oversight and the theorem should contain S-reverse plan, such that the set $\{s \mid s \in S, pre(a) \subseteq s\}$ is not empty, because the following proof builds on it.

A deeper investigation shows that the condition of subset of variables holds as well for the non-deterministic universal uniform action reversibility, but only for the strong variant of the research notion. In order to non-deterministic action be strongly universally uniformly reversible, variables assigned in its effects have to be contained in the set of variables of required in the preconditions. Otherwise, the action cannot be strongly universally uniformly reversible.

Lemma 3.7. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain such that $\forall v \in \mathcal{V} : |dom(v)| \geq 2$ and let $a \in \mathcal{A}$ be an action. If $\bigcup_{e \in \mathit{eff}(a)} vars(e) \nsubseteq vars(pre(a))$, then the action a is not strongly uniformly universally reversible by any policy.*

Proof.

The lemma will be proven by a contradiction.

Assume that there is a strong universal reverse policy Π for the action a.

Let $e \in \mathit{eff}(a)$ be an action effect for which exists $v \in vars(e) \setminus vars(pre(a))$. Since $|dom(v)| \geq 2$, there are two distinct states s_1, s_2 that differs in the value of the variable v and the action a is applicable in both of them (since $v \notin vars(pre(a))$). As states s_1 and s_2 differ only in one variable and this variable is in effects, the states $\gamma(s_1, a_e^d)$ and $\gamma(s_2, a_e^d)$ are the same.

The policy Π being a universal reverse policy for the action a implies that for states s_1 and s_2 hold $\delta(\gamma(s_1, a_e^d), \Pi) = \{s_1\}$ and $\delta(\gamma(s_2, a_e^d), \Pi) = \{s_2\}$. But as $\gamma(s_1, a_e^d) = \gamma(s_2, a_e^d)$, we can substitute one them with another, leaving us with $\{s_1\} = \delta(\gamma(s_1, a_e^d), \Pi) = \{s_2\}$. But, this is in a contradiction with s_1 and s_2 being distinct.

□

The same condition, however, does not hold for the weak universal uniform reversibility. An action can be weakly universally uniformly reversible even though this property is violated.

Figure 3.3 advocates the claim of the previous paragraph. It visualizes the situation, where the a action is weakly universally uniformly reversible by the policy represented by the figure. The situation contains seven states s_i and seven vertices representing some sub-graph which is a strongly connected component. The key thing is that the set of states s_i contains all states in which the action a is applicable. Each sub-graph G_i contains states in which the action is not applicable and where values of variable not in $vars(pre(a))$ are the same as in s_i. The process of application of some determinizations of the action a in each state in which it is applicable is visualized by dashed lines. Dashed circle separates the set of states in which a is applicable from the rest. Let a have two non-empty effects. Evidently, for the effect e_1, $vars(e_1) \subseteq vars(pre(a))$, as the dashed lines of this determinization leads to states, where are value of variable not in $vars(pre(a))$ are the same. For the clarity of the figure, for the effect e_2, only the edge from the state s_1 is shown, even though there is one for state s_i. It can be seen that $vars(e_2) \nsubseteq vars(pre(a))$ and also $vars(pre(a)) \cap vars(e_2) \neq \emptyset$ (as it ends in G_7 and not in s_7). This means that $\bigcup_{e \in \mathit{eff}(a)} vars(e) \nsubseteq vars(pre(a))$. But, since G_i is a strongly connected component and s_1 is reachable from G_1 and G_1 is reachable from G_7, the state s_1 is a terminal state of the visualized policy with respect to the state $\gamma(s_1, a_{e_2}^d)$; and since this holds for all states s_i, the action a is weakly universally uniformly reversible.

The idea behind this counter example is universal. If we want to show universal uniform reversibility, we can add any state-action pair we want as long as the reversed action is not applicable in that state.

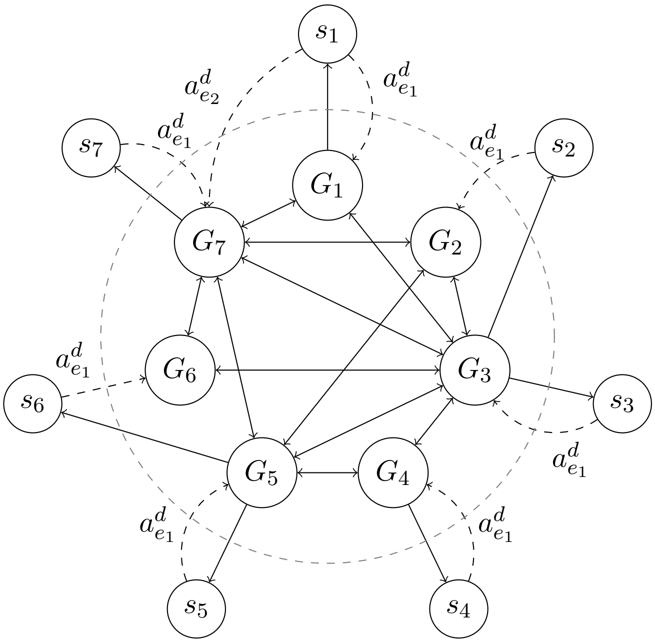

Figure 3.3: Counter-example supporting the argument about the weak universal uniform action reversibility.

The next theorem, which was already referred to in the first paragraph of this section, presents the analogy of theorem 3.6 for FOND formalism.

Theorem 3.8. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain, Φ be an implicitly-defined policy for the domain $\mathcal{D}$ such that $\forall (\varphi', a') \in \Phi : \varphi' \in \mathcal{P}(\mathcal{F}(vars(pre(a))))$ and $a \in \mathcal{A}$ be an action such that $\bigcup_{e \in eff(a)} vars(e) \subseteq vars(pre(a))$. If the policy Π implicitly defined by Φ is a weak (resp. strong) S-reverse policy for the action a such that $\{s \mid s \in S, pre(a) \subseteq s\}$ is not empty and $\forall (s', a') \in \Pi : vars(a') \subseteq vars(pre(a))$, then the policy Π is a weak (resp. strong) universal reverse policy for the action a.*

Proof.

Consider some state $s \in S$ such that $pre(a) \subseteq s$. From the assumption, the policy Π is a weak (resp. strong) $\{s\}$-reverse policy.

To show that Π is a weak (resp. strong) universal reverse policy for the action a, we need to prove that for each state $s' \in \mathcal{S}(\mathcal{V})$ such that $pre(a) \subseteq s'$ other than s the policy Π is also a weak (resp. strong) $\{s'\}$-reverse policy for the action a.

If there is no other state $s' \in \mathcal{S}(\mathcal{V})$ such that $pre(a) \subseteq s'$ and $s \neq s'$, then the policy Π is a weak (resp. strong) universal reverse policy for the action a.

If there is a state $s' \in \mathcal{S}(\mathcal{V})$ such that $pre(a) \subseteq s'$ and $s \neq s'$, then the set $\{v \mid v \in \mathcal{V}, s[v] \neq s'[v]\}$ is non-empty, as s' differs from s in at least one variable. Also, for each variable v in $vars(pre(a))$ hold $s[v] = s'[v]$, because the action is applicable in both states.

The assumption $\bigcup_{e \in eff(a)} vars(e) \subseteq vars(pre(a))$ states that any effect e of the action a operates maximally on variables assigned in the action's a preconditions. For each effect $e \in eff(a)$ it holds that $vars(e) \cap (\mathcal{V} \setminus vars(pre(a))) = \emptyset$. Therefore, values of variables of $\mathcal{V} \setminus vars(pre(a))$ remains unchanged after the application of the action a; no matter which stochastic effect e occurred when applied either to s or s'. Also, values of variables in preconditions are the same in both states. As a consequence, values of these variables is also the same in the states $\gamma(s, a^d_e)$ and $\gamma(s', a^d_e)$.

Due to the reason, that for any state-action pair $(s', a') \in \Pi$ holds $vars(a') \subseteq vars(pre(a))$, neither the actions of the policy Π can influence or change variables of $\mathcal{V} \setminus vars(pre(a))$. Also, as $pre(a') \subseteq vars(a')$, the application of the policy is also independent on variables of $\mathcal{V} \setminus vars(pre(a))$.

From Π being a weak (resp. strong) $\{s\}$-reverse policy we know that $s \in \tau(\Pi, \gamma(s, a^d_e))$ (resp. $\{s\} = \tau(\Pi, \gamma(s, a^d_e)))$. Therefore, either $s = \gamma(s, a^d_e)$, or $s \neq \gamma(s, a^d_e)$ and s can be reached by the policy following.

In the first case, the effect e must have no effect on the variables of $vars(pre(a))$, therefore the state $\gamma(s', a^d_e) = s'$ as well (as these variable have same values in both states s and s').

In the second case, there has to be a state-action pair $(\gamma(s, a^d_e), a') \in \Pi$ which weakly (resp. strongly) leads us towards the state s. The state-action pair corresponds to some formula-action pair (φ', a') in the implicitly-defined policy Φ such that $\gamma(s, a^d_e) \models \varphi'$. As, from the assumption, $\varphi' \in \mathcal{P}(\mathcal{F}(vars(pre(a))))$, the difference in values of variables $\mathcal{V} \setminus vars(pre(a))$ cannot influence the valuation of the formula φ'. Hence, $\gamma(s', a^d_e) \models \varphi'$ as well, and therefore, the policy Π also contains the state-action pair $(\gamma(s', a^d_e), a')$. This means that during the policy following from either state $\gamma(s, a^d_e)$ or $\gamma(s', a^d_e)$, the action a' is applied. It has been already argued that the action a' cannot change the values of the variables in $\mathcal{V} \setminus vars(pre(a))$; hence, the values are still the same even after the application of a'.

For any stochastic effect e' of the action a' and states $\gamma(\gamma(s, a^d_e), (a')^d_{e'})$ and $\gamma(\gamma(s', a^d_e), (a')^d_{e'})$, the same argumentation can be done; leaving us with a conclusion, that the state s' is handled by the policy Π in the same way as s is (no matter what the difference between them is) up to a point when the states s and s' are reached.

Ultimately, the policy Π is able to reverse the changes made by any effect e in the state s', therefore the policy Π is also a weak (resp. strong) $\{s'\}$-reverse policy for the action a; and hence the policy Π is a weak (resp. strong) universal reverse policy for the action a.

$\square$

Note, that the condition of non-empty set $\{s \mid s \in \mathcal{S}(\mathcal{V}), pre(a) \subseteq s\}$ is necessary. It is simply due to corollary 2.3. The proof without this condition would fail at a very beginning. The first paragraph of the proof would still hold, but we would be unable to show that the policy leads the agent from any state $\gamma(s', a^d_e)$ to the state s', because we would not be able to refer to the ability to lead the agent from the state $\gamma(s, a^d_e)$ to the state s.

Finally, once theorem 3.8 is proven, we can exploit it to describe situations which prove desired weak and strong universal uniform action reversibility, in the same fashion as the work of Chrpa; Faber; Morak (2021) did for the deterministic action reversibility.

The following theorem proves that a weak universal uniform reversibility can be shown by a solving of multiple classical planning tasks—one for each determinization. The task is to find a sequence of actions which reverts the effects of the determinization. The problem of weak reversibility is compiled into a classical planning task such that the plan from the variable assignment which necessarily holds after its application to any state where the preconditions of the action are satisfied is found. The actions of the domain of the task cannot contain stochastic actions. Therefore, they are replaced with their determinizations. Furthermore, in order to utilize theorem 3.8, the plan can contain determinizations of actions not touching the variables which are not assigned in preconditions only.

Theorem 3.9. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $a \in \mathcal{A}$ be an action such that $\bigcup_{e \in \mathit{eff}(a)} \mathit{vars}(e) \subseteq \mathit{vars}(\mathit{pre}(a))$. If for each determinization a^d_e of the action a the policy π_e is a goal plan for the SAS$^+$ planning task $\mathcal{T}^d_e = \langle\langle \mathit{vars}(\mathit{pre}(a^d_e)), \{(a')^d_e \mid a' \in A, e \in \mathit{eff}(e), \mathit{vars}((a')^d_e) \subseteq \mathit{vars}(\mathit{pre}(a^d_e))\}\rangle, \mathit{ha}(a^d_e), \{\mathit{pre}(a^d_e)\}\rangle$, then the action a is weakly universally uniformly reversible.*

Proof.

Due to the reason of the plan π_e being a goal plan of the task $\mathcal{T}^d_e$, we get $\gamma(\gamma(\mathit{pre}(a^d_e), a^d_e), \pi_e) = \mathit{pre}(a^d_e)$. Hence, the plan π_e is a $\{\mathit{ha}(a^d_e)\}$-reverse plan for the action a^d_e and the domain $\mathcal{D}^d_e = \langle \mathit{vars}(\mathit{pre}(a^d_e)), \{(a')^d_e \mid a' \in A, e \in \mathit{eff}(a'), \mathit{vars}((a')^d_e) \subseteq \mathit{vars}(\mathit{pre}(a^d_e))\}\rangle$.

Denote the actions of the plan π_e as $\pi_e = \langle a_{e_1}, \ldots, a_{e_n}\rangle$. Now, let $\Pi^d_e = \{(\gamma(\mathit{ha}(a^d_e), \langle a_{e_1}, \ldots, a_{e_{i-1}}\rangle), a_{e_i}) \mid i \in \mathbb{N}, 1 \leq i \leq n\}$. The policy following of the policy Π_e from the state $\mathit{ha}(a^d_e)$ results in the same state as an application of the plan π_e; in the variable assignment $\mathit{pre}(a^a_e)$. This means that Π^d_e at least weak goal policy for the (stochastic) planning task $\mathcal{T}^d_e$. If we substitute all determinizations with their stochastic variants, resulting policy still weakly reaches $\mathit{pre}(a^d_e)$ from $\mathit{ha}(a^d_e)$, hence the policy $\Pi_e = \{(\gamma(\mathit{ha}(a^d_e), \langle a'_{e_1}, \ldots, a_{e_{i-1}}\rangle), a_{e_i}) \mid i \in \mathbb{N}, 1 \leq i \leq n, a'_{e_i} \in \mathcal{A}, a_{e_i}$ is a determinization of $a'_{e_i}\}$ is a weak goal policy for the task $\mathcal{T}_e = \langle\langle \mathit{vars}(\mathit{pre}(a)), \mathcal{A}\rangle, \mathit{ha}(a^d_e), \mathit{pre}(a^d_e)\rangle$.

From Π_e being a weak goal policy for the task $\mathcal{T}_e$ we get that $\mathit{pre}(a^d_e) \in \tau(\Pi_e, \mathit{ha}(a^d_e))$. Also, as $\mathit{ha}(a^d_e) = \gamma(\mathit{pre}(a^d_e), a^d_e)$, we get that the action a^d_e is weakly $\{\mathit{pre}(a^d_e)\}$-reversible by the policy Π_e. We can observe that if we can some variable v with its domain, no matter what value $x \in \mathit{dom}(v)$ it has, the action a^d_e is still $\{\mathit{pre}(a^d_e) \cup (v, x)\}$-reversible by the policy Π_e, and that the conditions of theorem 3.8 are satisfied even for the domain $\langle \mathcal{V}, \mathcal{A}\rangle$ (with the rest of variables present). Hence, utilizing the theorem theorem 3.8 we get the policy Π_e is a universal reverse policy for the action a^d_e and the domain $\langle \mathcal{V}, \mathcal{A}\rangle$.

As this holds for any determinization of a, according to theorem 3.4, we prove that the policy $\bigcup_{e \in \mathit{eff}(a)} \Pi_e$ is a weak universal reverse policy for the action a and the domain $\mathcal{D}$.

$\square$

Quite a similar claim can be stated about the strong universal uniform reversibility, however, classical planning is not enough. It comes from a fact that the strong universal uniform reversibility needs to close all paths not leading to the desired state. But, this is exactly with what the strong planning deals with. The compilation of the reversibility to the FOND planning task is the same as in the weak situation, only the actions are not determinized.

Theorem 3.10. *Let $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ be a planning domain and $a \in \mathcal{A}$ be an action such that $\bigcup_{e \in \mathit{eff}(a)} \mathit{vars}(e) \subseteq \mathit{vars}(\mathit{pre}(a))$. If for each determinization a^d_e exists a strong goal policy Π_e for the FOND planning task $\mathcal{T} = \langle\langle \mathit{vars}(\mathit{pre}(a^d_e)), \{a' \mid a' \in \mathcal{A}, \mathit{vars}(a') \subseteq \mathit{vars}(\mathit{pre}(a^d_e))\}\rangle, \mathit{ha}(a^d_e), \{\mathit{pre}(a^d_e)\}\rangle$ such that the sets $\sigma(\Pi_e)$ are pairwise disjoint, then the action a is strongly universally uniformly reversible.*

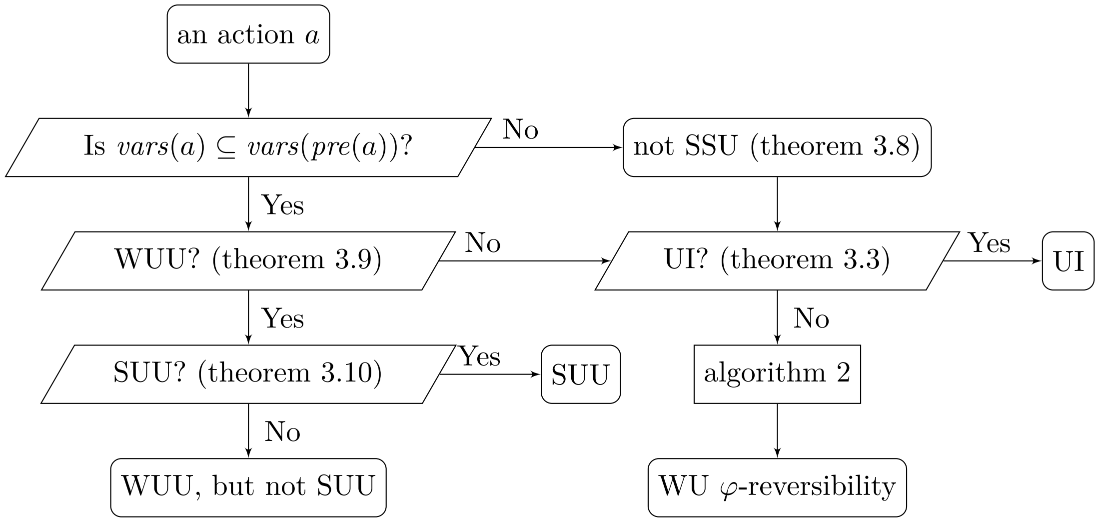

Figure 3.4: Scheme describing the process of deciding of the non-deterministic action reversibility.

Proof.

From Π_e being a strong goal policy for the task $\mathcal{T}$, we know that $\{pre(a_e^d)\} = \tau(\Pi_e, \gamma(pre(a_e^d), a_e^d))$. That means that the action a_e^d is strongly $\{ha(a_e^d)\}$-reversible by the policy Π_e for the domain $\langle vars(pre(a_e^d)), \{a' \mid a' \in \mathcal{A}, vars(a') \subseteq vars(pre(a))\}\rangle$.

As the sets $\sigma(\Pi_e)$ are pairwise disjoint, the conditions of the theorem 2.6 hold. Hence, we can claim that the action a is strongly $\{pre(a_e^d)\}$-reversible by the policy Π_e for the domain $\langle vars(pre(a_e^d)), \{a' \mid a' \in \mathcal{A}, vars(a') \subseteq vars(pre(a_e^d))\}\rangle$.

We can observe that if we add some variable v with its domain, no matter what value $x \in dom(v)$ it can have, the action a_e^d is still $\{pre(a_e^d) \cup (v, x)\}$-reversible by the policy Π_e, and that the conditions of theorem 3.8 are satisfied even for the domain $\langle \mathcal{V}, \{a' \mid a' \in \mathcal{A}, vars(a') \subseteq vars(pre(a))\}\rangle$ (with the rest of variables present). Also, if we expand the set of actions of the domain, the claim still holds. Hence, the policy $\bigcup_{e \in eff(a)} \Pi_e$ is a strong universal reverse policy for the action a and the domain $\mathcal{D} = (\mathcal{V}, \mathcal{A})$.

□

3.5 Decision Scheme

Up to this point, it has not been addressed that the necessary condition on domain sizes of the variables of some theorems in this section does not cause any practical problems. As each domain is non-empty, the only problematic variables are that with trivial domains of only one value. In practice, if the domain contains some such variables, they can be simply removed without any problem, since they could not influence neither domain nor its tasks in any way. If we discard them, the condition is no longer violated and theorems are utilizable.

Firstly, in order to achieve a compact visualization, the scheme utilizes abbreviations not yet used in this work. We believe that they are intuitive even though some of them are ambiguous. The weak (resp. strong) universal uniform action reversibility is abbreviated as WUU (resp. SUU) action reversibility. The universal action irreversibility is represented as UI. The lastly presented abbreviation shows the ambiguity in them. It is the weak uniform φ-reversibility. The ambiguity comes from the letter "U", because it is present in both WU and UI, but in both case it means something different. In one case, the "U" stands for "uniform", and in the second one, it stands for "universal".

The decision process, which is presented in fig. 3.4, begins with a most trivial operation in it. That is a check whether lemma 3.7 can be utilized.

If it is satisfied, we know that the action is certainly not strongly universally uniformly reversible. In that case, we try the prove universal irreversibility of the action according to theorem 3.3. Otherwise, algorithm 2 is run to, ultimately, decide the weak universal φ-reversibility (up to a incompleteness of our implementation).

If lemma 3.7 is not satisfied, then the action is tested whether theorem 3.9 can be utilized. If it is applicable, then we know that the action is weakly universally uniformly reversible and the analogous test is done for the strong universal uniform reversibility, according to theorem 3.10. If it succeeds, the action is both weakly and strongly universally uniformly reversible; otherwise, it is only weakly universally uniformly reversible and not strongly universally uniformly reversible. If the weak universal uniform reversibility check failed, the action is tested in the same way as if lemma 3.7 was utilized. It is firstly tested for universal irreversibility and then, eventually, algorithm 2 is used to decide the weak universal φ-reversibility.

Chapter 4

Experiments

The chapter presents empirical evidence about investigated phenomena. Based on the claims of previous chapters, we have designed and prepared the experiments for the resolution of any non-deterministic action reversibility for its multiple classes; namely for the weak and strong universal uniform action reversibility, the weak uniform φ-reversibility and the universal uniform action irreversibility. The implementation is discussed in detail in the corresponding section of this chapter. The section dedicated to the implementation is followed by a section enlightening which domains were used for the experiments. The section also hypothesizes and reasons to which class of non-deterministic action reversibility actions of the investigated domains should belong. After that, the empirical evidence gathered from performing the experiments is presented. The evidence mainly shows the incidence rates of the classes of non-deterministic action reversibility. For some of the classes, a deeper analysis and its result is provided. The chapter concludes with a section devoted to the discussion about the results; mainly, whether they met the expectation, what can or should be improved and about their consequences.

All experiments run on a machine with Intel® Core™ i7-7700HQ processor and 32 GB of DDR4 RAM operating on frequency of 2400 MHz. The operating system was Ubuntu 22.04 in WSL 2 of Windows 11.

4.1 Implementation

The evaluation completely follows the scheme depicted in fig. 3.4 of the previous chapter. Firstly, variables of the action are checked against the variables in preconditions. Then, either the universal irreversibility and the weak uniform φ-reversibility is tested or the weak and the strong universal uniform reversibility is tried to prove with a respect to theorems 3.9 and 3.10 (and eventually, depending on the result of the weak universal uniform action reversibility, also the other branch).

4.1.1 Weak Uniform φ-reversibility

We have implemented the algorithm 2 from the scratch. The implementation utilizes our formerly existing framework for *classical planning*; although it had to be extended to be able to handle stochastic action effects. Both framework and the algorithm is implemented in C++ programming language of the standard C++20 [22] and it does not exploit any third-party libraries. The code is compiled with the use of CMake (version 3.22.1) [23] and g++ (version 11.3.0) [14].

The non-determinism of the action selection is handled by the uninformed *breadth-first search* (BFS) (Russell; Norvig, 2010) through the state space utilizing the *closed list*. When an action is selected for an application on the line 5, all “applicable” actions are considered. Similarly, the non-deterministic selection of the determinization of the action is dealt with by an exhaustive iteration through all of them. Consequently, no possible solution is left out. On the other hand, the computational complexity of the algorithm is extensive. As this is an uninformed search, it may happen that it is practically unfeasible to completely explore the state space if it is large enough.

An implementation may be modified to deal with this problem at the cost of completeness of the implementation by limiting the maximal depth of exploration. If the algorithm for any determinization does not find a solution up to that point, it can be terminated without knowing whether there is some “reasonable” formula by which the action is weakly uniformly φ-reversible or not. While experimenting, we set the depth limit to one hundred resulting in no such time-outs while being able to decide weak universal φ-reversibility in a reasonable time. The run-time of the majority of domains was under 10 seconds and the peak time requirement was around 35 minutes for Elevators domain. The peak memory requirements of the calculations were around 8.5 GB of RAM.

Another insufficiency of the implementation of the algorithm 2 is that once we find some solution formula for any determinization, we stop the search for the other ones (as they may be multiple formulae for any particular determinization), instead of exhaustively searching through the rest of the state space for others. The justification of the step-back is again the enormity of the state space and the computational infeasibility. If we would run algorithm 2 not only until the first solution is found, but longer, with this implementation, we would have to possibly explore a large portion of the state space of each examined problem.

Once formulae for all determinizations are available, they are joined together by conjunctions, as algorithm 2 requires. Then, the formula φ is investigated whether it is a contradiction or not. If the contradiction is found, the result of the algorithm is considered negative. Otherwise, there is at least one state in which the action is applicable, the action is weakly reversible in it; hence the result is considered positive. The corresponding columns can be found in later presented figs. 4.1 and 4.2. The negative results are later depicted as $\perp$-reversibility (abbreviated as $\perp$-R), whereas positive results as φ-reversibility (abbreviated as φ-R). time-outs of the algorithm are labelled as T. Other columns of figs. 4.1 and 4.2 are using the same abbreviations as in fig. 3.4.

We then further investigate the formula φ for an amount of variables it assigns. These values necessarily range from the number of variables in preconditions (implying the weak universal uniform reversibility) to overall amount of the variables in the domain (implying the weak φ-reversibility in the only one state). The results of this investigation are shown in fig. 4.4. We also report the depths of the policies for determinizations found by algorithm 2 in fig. 4.3.

4.1.2 Weak and Strong Universal Uniform Reversibility and Universal Uniform Irreversibility

To prove the weak and strong universal uniform action reversibility and the universal uniform irreversibility the implementation utilizes reformulations to classical and FOND planning as they are described by theorems 3.3, 3.9 and 3.10. While the weak universal uniform action reversibility and universal action irreversibility uses Fast Downward (FD) planner (Helmert, 2011; Helmert, 2009) to determine the solvability of the reformulated problem, the strong universal uniform reversibility utilizes Planner for Relevant Policies (PRP) planner (Muise; Sheila A McIlraith; Beck, 2012; Muise; Belle; Sheila A. McIlraith, 2014; Muise; Sheila A. McIlraith; Belle, 2014) instead. The FD planner is set to use LAMA (Richter; Westphal, 2010) search algorithm which is stopped after the first iteration (because it is an iterative anytime search algorithm). To get the reformulations we have modified the translator of PRP which is an extended translator of FD planning system (Helmert, 2009).

4.2 Evaluated Domains

This section briefly describes the domains used in the experiments. The description of each domain is accompanied with an hypothesis about the reversibility of its actions. The work investigates the non-deterministic action reversibility of 11 domains in total and all of them come from a repository of PRP.

4.2.1 Blocks World

A non-deterministic variant of well-known domain of used in classical planning.

The environment that the domain describes contain multiple blocks and arms which manipulates them. Blocks can be placed on the top of another, on the ground (or on table) or they can be picked by one of the arms. The typical task is use arms and associated actions to rearrange the block from some initial configuration such that some conditions are satisfied. For example, placing all blocks on the table or to build one huge tower out of them, where a specific order of the blocks is required.

The stochasticity of the domain comes from an inaccuracy of arms. If the arm tries to pick up the block, it either picks it up or the block falls down to the table. Similarly, if the arm wants to put the hold block on the top on another, it can either succeed, or the block falls down as well. Putting the block down to the table cannot fail and it is deterministic. The arm can also pick up a tower—two blocks on top of each other, but only if they are both on top of some third block. It cannot pick up the tower from the table. Picking a tower can also fail, but if it does, nothing happens. If the arm tries to put hold tower on the top of some block, the tower is either put on the top of that block or it is put down on the table. The arm can put the tower down on the table without any problem as well.

As the agent is able to pick up the fallen block, the actions of picking up and putting down a single block are expected to be strongly reversible no matter where we put the block or from where we pick it up. Even though the action for picking up the tower from table is missing, it can be replaced by a sequence of actions. If the tower is put to a table, we should be able to strongly universally revert this effect, if there is at least one another (third) block, by stacking the blocks one-by-one on the top of that third block and then picking the tower up from it.

4.2.2 Exploding Blocks World

Another variant of the Blocks World domain. It lacks the ability to move towers, but it adds two random effects. The blocks in this domain are either detonated or not. If we want to put down the non-detonated block, it can explode. If it does, either the block on which the the handled block was put or the table is destroyed. A destroyed block can no longer be manipulated with. Similarly, if the table is destroyed, we can no longer put anything else on it. The detonated block can be manipulated in the same way as non-detonated one. It just cannot explode anymore.

Regarding the action reversibility, the same situation occurs as in Blocks World, but the actions of putting down a non-detonated block universally irreversible, as it may explode and irreversibly change the environment.

4.2.3 Tire World

A domain models a situation where there is a car trying to reach some target location. The problem is that every time the car moves from one location to another, there is a possibility of getting a flat tire, resulting in an impassable car. If the spare tire is present in the car, it can be changed. The change can fail, but the spare tire is not lost and nothing happens. If it succeeds, the car can continue in its journey and the spare tire is consumed. There are also locations where exactly one spare tire is present. If the car is there and it has no spare tire, the tire can be loaded without any problem, resulting in no more spare tire in that location.

Our intuition guides us that every movement in this environment is dangerous. If the movement results in a flat tire, we have to fix it. Even if we do, we have a lost the spare one. So these actions should be irreversible. Loading of a spare tire is irreversible as well, because there is at most one spare tire in each location, which is consumed by loading. The same holds for changing it as well.

4.2.4 Elevators

A domain describes a building with multiple elevators. The task of the agent is to collect all coins situated in some floors in front of some elevators. The agent can step in or out of the elevator. The elevators can go up and down without any problem (with or without the agent). If the agent is in front of the elevator on floor where the coin is, he can collect it, leaving no more coin there and having it. In addition, the agent can walk in between the elevators of the same floor. But, if there is a gate out in that floor near the elevator, the agent can accidentally leave the building (and he cannot enter through this gate in again). In that case, he ends up in the front of the first elevator in the first floor after he enters again by the main entrance.

As the actions of the movement of elevators and stepping of the agent in and out are deterministic, these actions should be strongly universally uniformly reversible. Picking up a coin is universally irreversible, because there is no way how to put the coin back. If he moves on the floor, he can end up outdoor. But he can still return to a place which he left no matter what, so the action should be strongly reversible.

4.2.5 Forest

A large domain containing tree well-know sub-problems. The domain consists of Blocks world, Logistics and Grid navigation domain. Due to the cardinality of the actions in the Forest domain, we omit description of actions in other parts and refer the reader International Planning Competition (IPC) webpage [21]. The goal is to traverse through the forest to the other side. The traversal is complicated by the sub-problems distributed over the forest. The agent cannot move away until the task at the position in which he is located is solved. Also, his movement is stochastic. If he wants to move to some desired adjacent position, he can end up in another one.

We expect the movement of the agent to be reversible. Also, once the agent solves the sub-problem, it should no longer be solvable. Hence, the action of marking the sub-problem as solved should be irreversible, as there is no way that the sub-problem reoccurs. Without any justification, the actions of the sub-problems should be strongly reversible for all three sub-domains, due to their nature.

4.2.6 Zeno Travel

In this domain the goal is to transport a set of people located in various cities to their desired ones via planes. The person can start a boarding without any stochasticity. The completion of the boarding is, however, non-deterministic. It either succeeds or does nothing and it has to be repeated. The other actions for getting of the plan, flying of the plane, zooming of the plane to the city and plane refueling follow the same scheme.

As there is no time in the domain and there is an unlimited amount of the fuel at each airport, the actions should be strongly reversible, as long as the planes can return to the airports they flew from.

4.2.7 First Response

In the domain, the goal is to put out all fires and save all injured victims in a city, provided the amount of medical and fire units. In the city, there are locations. These locations can contain injured people, a fire, a hospital, water or nothing. The units can be moved from one location to another. If the fire department unit is in a place where water is, the fire department can pump water into car to be able to put out a fire with it later. The putting the fire out can be done only if the unit has water and it can fail. In that case, water of the unit is lost and nothing else happens. In similar fashion, the medical department unit can load an injured person into the vehicle. There can be an unlimited number of injured people in the vehicle. Any loaded injured person can be unloaded without a problem out the medical vehicle. There are two types of injuries. The person is either dying or is hurt only. If the patient is only hurt, he can be treated on the scene by both medical and fire unit. Both units have the change to fail in providing the help. In that case, it can be tried again. If they succeed, the patient is healthy and saved. Dying patients have to be transported to location with a hospital, where the injured people can be treated without any problems.

We expect the movement of units to be reversible, as well as loading and unloading patients into the medical unit. The treating of the patient should be irreversible, because it irreversibly changes his condition. The same hold for putting out the fire. The loading of water into the fire unit should be reversible as long as there is some fire on which the loaded water can be used to empty it.

4.2.8 Bus Fare

A domain which incorporates multiple non-deterministic effects. The agent starts with 1 coin. His goal is to own a bus fare. The price of the bus fare is 3 coins. If he has 1 coin, he can bet it and either lose it or win 2 more, or he can wash someones car and hope, that he will get a 1 coin for it. If he has 2 coins, he can again bet it and get only 1 coin back if he loses or 3 if he wins, or wash someones car again. In this case, he either receives nothing or he is robbed for 1 coin in a process. If he has 3 coins, he has to buy the fare.

This domain is expected to contain only two variables. The number of coins the agent have and whether he has a fare. There are was to get from 2 coins to 1, and from 1 coin to 2. The action of washing a car should be reversible. On the other hand, once the agent has no coin or 3 coins, he cannot go back to 1 or 2 coins. As both betting actions may leave the agent with one of that amounts, they should be irreversible. Buying a fare is irreversible as well.

4.2.9 Climber

A small domain where the agent is on the top of some building. His task is to get to the ground without being harm. There is a ladder under the build, that can be raised to provide him a safe passage down. He has two options. Either climb down without the ladder and risk dying, or to call for help. The call cannot fail and results in someone raising a ladder for him. The climbing down with raise ladder is also deterministic and completely safe.

As there are no actions to undo any of the available actions in any situation, we claim that these actions should be universally irreversible.

4.2.10 River

This small domain models a situation when the agent wants to cross a river. He begins at a bank. There he has two options. Either to traverse rocks or to swim. If he traverses rocks, then he is no longer on the bank and can either reach the second target bank, die in a process or end up on an island. If he tries to swim, he leaves the bank, and can end up on the second target bank or nothing more happens (yet he still leaves the bank, and therefore cannot repeat the application of this action). When he is on the island, he can do only one thing; and that is to swim to the target bank, with either reaching it or dying in the process.

As there are no actions to undo any of the available actions in any situation, we claim that these actions should be universally irreversible.

Domain	#	$vars(a) \not\subseteq vars(pre(a))$				$vars(a) \subseteq vars(pre(a))$					
		¬ SUU				WUU		¬ WUU			
		UI	⊥-R	WU φ-R	T	¬ SUU	SUU	WU φ-R	⊥-R	UI	T
Blocks World	190	0	0	185	0	0	5	0	0	0	0
Bus Fare	5	0	0	0	0	0	2	0	0	3	0
Climber	3	0	0	0	0	0	0	0	0	3	0
Elevators	41	3	0	30	0	0	8	0	0	0	0
Exploding Bl. W.	75	25	0	45	0	0	0	5	0	0	0
Faults	51	25	0	26	0	0	0	0	0	0	0
First Response	46	6	0	4	0	0	22	8	0	6	0
Forest	148	8	0	117	0	4	0	8	0	11	0
River	3	2	0	0	0	0	0	0	0	1	0
Tire World	52	7	1	0	0	0	0	0	44	0	0
Zeno Travel	740	0	0	96	0	0	504	140	0	0	0
total	1354	76	1	503	0	4	541	161	44	24	0

Table 4.1: Table depicting the results of deciding non-deterministic action reversibility according to fig. 3.4.

4.2.11 Faults

The goal of the domain is to perform multiple indistinguishable operations. Each operation may fail or succeed. If it fails, a counter of faults of increased and the operation has to be repaired and performed again. If the limit number of faults is reached, no more operation can be performed and the task is considered unsolved. In the end, in order to solve the task, the action verifying the completion of the tasks needs to be applied. This action should be irreversible, because the action should be applicable only if the task is not yet complete.

As each fault is counted, the action performing any operation is irreversible. Each operation can fail multiple times, therefore the repair action should be weakly reversible.

4.3 Results

The first thing that we point out is that no action with more than one effect was proven strongly uniformly universally reversible.

For Blocks World domain, all actions were proven reversible. However, only 5 of them are strongly uniformly universally reversible. Other actions violated the condition of theorem 3.8 and were then proven weakly uniformly φ-reversible. This is not surprising, as apparently, the actions are defined in a way that they may modify variables not in preconditions of the action, hence they cannot be strongly uniformly universally reversible. On the other hand, our hypothesis was shown incorrect, as we did not expect that there will be a variable in effects which is not required in preconditions; and hence, the action will not be strongly universally uniformly reversible according to theorem 3.8.

The similar situation occurred in Exploding Blocks World. Neither there we expected that theorem 3.8 will be applicable. But, we have successfully predicted universally irreversible actions of the domain.

The results of non-deterministic action reversibility for small domains we investigated—Bus Fare, Climber and River—follow our hypothesis.

Domain	#	WU φ-R	$\perp$-R	Proven UI	Undecided	T
Blocks World	185	185	0	0	0	0
Elevators	30	30	0	0	0	0
Exploding Blocks World	50	50	0	0	0	0
Faults	26	26	0	0	0	0
First Response	12	12	0	0	0	0
Forest	125	125	0	0	0	0
Tire World	45	0	45	45	0	0
Zeno Travel	236	236	0	0	0	0

Table 4.2: Table presenting results of the completeness of algorithm 2.

In Elevators domain, actions picking up the coin were proven irreversible and the movement of elevators were proven strongly universally uniformly reversible, as expected. On the other hand, other actions were proven not universally uniformly reversible, as they violates the necessary condition $\bigcup_{e \in \mathit{eff}(a)} \mathit{vars}(e) \subseteq \mathit{vars}(\mathit{pre}(a))$.

The actions which perform the operations in Faults domain were proven, consistently with the assumption, as universally irreversible. In contrast, for the action verifying the completion, weak uniform φ-reversibility was proven; indicating that the action may be ill-formed. It can be verified that the action does not require the task to be unsolved before the application. Other actions were proved reversible as expected.

The hypothesis about First Response domain was correct. Some of them were even proven strongly uniformly universally reversible.

The actions of sub-problems were proven reversible, as expected. What is not intuitive is that no action is strongly universally uniformly reversible. This is caused be a fact that in order to any sub-problem action be applicable, the sub-problem has to be initialized and active. The actions performing initialization of the sub-problem were proven irreversible. We have noticed that the action dedicated to the sub-task completion is ill-formed, as it does not require the sub-task not to be done. We have added it and then the action was proven irreversible as expected. The most interesting finding related to this domain is that not all actions for the movement in the grid are reversible, because, for some transitions, there are no action leading back to the same place.

Even though that for Tire World domain the hypothesis was completely correct, it may be surprising that the majority of actions was not decided by theorem 3.3, but by algorithm 2. This is caused by a fact the there is a path in the abstracted domain, which is otherwise inapplicable due to a variable in the precondition which was abstracted out.

Zeno Travel domain resulted in all action being reversible, with a majority of strongly universally uniformly reversible actions. Other actions, namely all action containing some stochasticity plus the actions which start refueling, were proven weakly uniformly φ-reversible only.

Figure 4.2 shows the ability of the incomplete implementation of algorithm 2 to decide action which were given to it. The columns correspond to various situations. The first columns (not counting the domains' column) is nothing more than the number of actions processed by algorithm 2. The second column depict situations where weak universal φ-reversibility was proven. The column labeled as $\perp$-R corresponds to all "failures" of the algorithm. Next to columns distinguish between a situation where some determinization is irreversible and therefore the weak uniform φ-reversibility is proven, and between a situation where algorithm 2 was not able to decide the weak uniform action φ-reversibility. The table concludes with a column of time-outs.

It can be seen that no action reached the depth limit of the implementation of BFS. Also, the algorithm left no action left undecided, meaning that incompleteness of the algorithm did not manifested.

Another table, labeled as fig. 4.3, presents depths of weak φ-reverse policies (resp. lengths of reverse plans) produced by determinizations of the actions inspected by the algorithm 2. Depths of policies are separated by commas and if the policy for the determinization does not exist, it is marked in the table as dash.

Lengths of reverse plans are in the most cases really short. In Zeno Travel domain, the policies are deeper than usual. The deepest policy was found in Elevators domain. It corresponds to a situation where the agent moved around the floor but accidentally leaved the building out.

Domain	Depths	#
Blocks World	1	85
	1, 1	80
	1, 2	20
Elevators	1	27
	7, 1	3
Exploding Blocks World	1	50
Faults	1	26
First Response	1	12
Forest	1	117
	1, 1	8
Tire World	-	1
	-, 1, 1	44
Zeno Travel	3	200
	5	36

Table 4.3: Table of number of facts of formulae found by algorithm 2 if it was not a contradiction.

The last table, labeled as fig. 4.4, puts forward how complex the formulae produced by algorithm 2 are. Due to a nature of the algorithm, it produces formulae which are conjunctions of facts only and the minimal amount of facts in the conjunction are the facts in preconditions. These fact can be removed from a formula without any influence on the set of states in which the action is proven weakly uniformly reversible (see). Hence, the presented amount (in the column labeled “minimal $|\varphi|$”) of facts is containing only constraining facts only (an analogy of corollary 2.3). This means that if the value is zero, the action is weakly universally uniformly reversible (however, this did not happened).

The values differ from 1 to 8, where values 1 and 2 are most common. It can be said that the values are not high, even when compared to the total number of variables. The highest amount of conditions was found in Faults domain, while Faults being the domain with second highest amount of variables. A high number of conditions is also found in Zeno Travel domain. The amount is high even in relative numbers.

Domain	$\|\mathcal{V}\|$	reduced $\|\varphi\|$	$\|pre(a)\|$	#
Blocks World	11	1	3	20
		2	1	5
			2	20
			3	80
		3	2	20
			3	40
Elevators	22	1	1	15
			2	12
		4	1	1
		5	1	1
		6	1	1
Exploding Blocks World	22	2	4	5
		3	3	5
			4	40
Faults	26	1	6	1
		7	2	5
		8	2	20
First Response	14	1	1	5
			2	4
		4	1	3
Forest	52	1	1	6
			2	12
			3	15
		2	1	3
			2	8
			3	40
		3	1	3
			2	6
			3	32
Zeno Travel	12	2	2	96
		4	3	48
		5	2	92

Table 4.4: Table of depths of policies found for determinizations of the inspected action by algorithm 2.

4.4 Discussion

The first thing we want to point out is the insufficiency of the algorithm 2. Firstly, the algorithm is implemented as uninformed BFS. For example, it can be reimplemented with best-first search which utilizes some guidance. One possible guidance may be to heuristically explore nodes corresponding to less complex formulae first. Secondly, the implementation lacks a complete exploration of state-space by a settling with first found solution. Due to this inadequacy, the implementation is not complete. This was justified by an intractability of the complete variant if no further improvements are added. Another idea not yet formalized may improve the performance of the search significantly. The key of the idea is to close not only expanded node, but also any other node which corresponds to a sub-state that is a superset of the sub-state of the expanded node. Intuitively, such “super-states” are more specific than the state of expanded node; and hence, each solution found from any “super-state” is necessarily contained in a solution found from the general state.

In a contrast to the *deterministic action reversibility*, for a non-deterministic variant, for many equivalences in the deterministic formalism, only one implication was proven in the non-deterministic. We hope that for many of them, the second implication can be proven as well. It would caused a significant improvement in the deciding of the non-deterministic action reversibility. In this moment, multiple unnecessary calculations could be performed. Hopefully, it would also outline more ways for deciding non-deterministic action reversibility, especially for the strong uniform non-universal variant, as this is a blind spot in our solution.

In 8 out of 11 domains, some non-trivial deterministic reversibility was proven; and in 9 out of 11, at least one *universally irreversible* action was found. Hence, it can be argued that the *action reversibility* and *action irreversibility* is a common phenomena in multiple domains. The phenomena may be leveraged for example for a *dead-end* detection. If the planner applies an action in some state in which the action is strongly reversible, then the planner is guaranteed that it can move back to the former state. This means that the novel state cannot be a *dead-end*, unless the former one was a *dead-end* as well. Analogously, if the action is irreversible in the former state, then the novel state is a *dead-end* if the action is not a part of the path to the goal state from the former one.

Another application of the action reversibility is in domain engineering, as this chapter has already pointed out. If the engineer designs the domain, he can verify its action with the action reversibility. If the result is different then he would expect, he has an indication that something may be wrong in the domain description.

The last application we present in this work is concerning plan optimization, either as an in-processing or a post-processing. When the (sub-optimal) plan is constructed, it can be then further optimized. The literature has shown that sub-optimal plans often contain set of actions which consists of the reversed action and its *reverse plan*, called an *action cycle* (Med; Chrpa, 2022). Such sets may be then removed while the resulting plan remains being the goal plan. If the information about the action reversibility is available in advance (as it is invariant on the task), these sets may be found without any non-trivial calculation. This idea represents post-processing. The usage of the action reversibility in plan in-processing is similar. If the planner is traversing through the state-space, it keep a track of the applied actions. In that situation, the information about the action reversibility can be used in a similar manner—the sequence of applied actions can be (probably when some conditions hold) filtered, resulting into a simpler sequence; and ultimately, into a simpler plan.

Conclusion

The main contribution of this work is that it has extended the notions of the *deterministic action reversibility*, which has been studied in several works, to *fully observable non-deterministic planning* capable of describing of stochastic environments. Furthermore, the work presented various theoretical findings and pointed out derivable consequences of novel definitions.

A combination of the state-of-the-art concept of the *weak* and *strong solutions* in the non-deterministic environments and of the deterministic action reversibility has been put forward and named the *weak (respectively strong) action reversibility*; allowing to leverage these definitions to design algorithms to determine a *weak (resp. strong) action reversibility* of the non-deterministic actions of FOND planning.

Several claims concerning the *weak* and *strong action reversibility* have been proposed and proven. With the use of these theorems, an algorithm deciding the *weak uniform φ-reversibility* has been described. A part of proven claims concerned the *weak* and *strong universal uniform action reversibility.* It has been shown that in order to an action be *strongly universal uniformly reversible*, it has to satisfy a non-trivial condition. Thanks to an counter-example shown by figure 3.3, the analogous situation concerning the *weak action reversibility* was proven unnecessary. With the use of another claims related to the *weak* and *strong universal uniform action reversibility*, a way how to prove the *weak* and *strong universal uniform action reversibility* was described. Finally, the process designed for determination of the *non-deterministic action reversibility* was described and this process was then used to evaluate the action reversibility of actions of multiple FOND domains.

An experiments were performed to gather an empirical evidence deciding *non-deterministic action reversibility.* We have described 11 domains of non-deterministic FOND planning and put forward hypotheses about tiaction reversibility of actions present in the domains.

Later, the empirical evidence gathered from experiments was compared to the hypotheses that were previously put forward. Hypotheses were proven partially. Even though that they were frequently correct, in some cases, the evidence and the hypothesis were in conflict. The chapter dedicated to the experiments discussed a reason of the conflict; showing that often the conflict may come from the possibly "incorrect" definition of the action in the domain.

We have provided four tables depicting the empirical evidence. The first one shows a overall results gather with a respect to the scheme presented in the end of third chapter. According to the table, in 8 out of 11 domains, all actions were non-trivially *weakly reversible*, proving that the *action reversibility* are frequently present in the domains. Analogously, at least one *universally irreversible* action is present in 9 out of the 11 researched domains. A *strongly universally uniformly reversible* action are present in 5 out of the 11 domains.

The rest three tables are dedicated to the results of algorithm 2. The first of them shows that the implementation of algorithm 2, even though being incomplete, decided all action on which it was used. The second proves that the *weak uniform φ-reverse policies* are other shallow, thank to a fact that the *reverse plans* for the *determinizations* are often not long. The last table is devoted to show that formulae which are found by the algorithm algorithm 2 are not complex, because they frequently contain only a few facts of the domain.

In the future work, a deeper theoretical results may be provided, as, hopefully, more corollaries and theorems concerning the *weak* and the *strong action reversibility* and *action irreversibility* can be derived. One particular aim may be to explore computational complexity of the *non-deterministic action reversibility.* Also, it was discussed that the presented algorithm 2, which decides *weak uniform action φ-reversibility*, may be further improved to be more efficient, for example by utilizing a heuristic search. Regarding the *strong action reversibility*, there is currently no algorithm presented. So, the proposal of new algorithms would also be beneficial.

Bibliography

1. BÄCKSTRÖM, Christer; NEBEL, Bernhard. Complexity Results for SAS+ Planning. *Comput. Intell.* 1995, vol. 11, pp. 625–656. Available from DOI: 10.1111/j.1467-8640.1995.tb00052.x.
2. BYLANDER, Tom. The Computational Complexity of Propositional STRIPS Planning. *Artificial Intelligence*. 1994, vol. 69, no. 1-2, pp. 165–204. Available from DOI: 10.1016/0004-3702(94)90081-7.
3. CHRISTEN, Remo; ERIKSSON, Salomé; POMMERENING, Florian; HELMERT, Malte. Detecting Unsolvability Based on Separating Functions. In: KUMAR, Akshat; THIÉBAUX, Sylvie; VARAKANTHAM, Pradeep; YEOH, William (eds.). *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. AAAI Press, 2022, pp. 44–52. Available also from: https://ojs.aaai.org/index.php/ICAPS/article/view/19784.
4. CHRPA, Lukás; FABER, Wolfgang; FISER, Daniel; MORAK, Michael. Determining Action Reversibility in STRIPS Using Answer Set Programming. In: DODARO, Carmine; ELDER, George Aristidis; FABER, Wolfgang; FANDINNO, Jorge; GEBSER, Martin; HECHER, Markus; LEBLANC, Emily; MORAK, Michael; ZANGARI, Jessica (eds.). *International Conference on Logic Programming 2020 Workshop Proceedings co-located with 36th International Conference on Logic Programming (ICLP 2020), Rende, Italy, September 18-19, 2020*. CEUR-WS.org, 2020, vol. 2678. CEUR Workshop Proceedings. Available also from: http://ceur-ws.org/Vol-2678/paper2.pdf.
5. CHRPA, Lukás; FABER, Wolfgang; MORAK, Michael. Universal and Uniform Action Reversibility. In: BIENVENU, Meghyn; LAKEMEYER, Gerhard; ERDEM, Esra (eds.). *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*. 2021, pp. 651–654. Available from DOI: 10.24963/kr.2021/63.
6. CHRPA, Lukás; MCCLUSKEY, Thomas Leo; OSBORNE, Hugh. Determining Redundant Actions in Sequential Plans. In: *IEEE 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012, Athens, Greece, November 7-9, 2012*. IEEE Computer Society, 2012, pp. 484–491. Available from DOI: 10.1109/ICTAI.2012.72.
7. CIMATTI, Alessandro; PISTORE, Marco; ROVERI, Marco; TRAVERSO, Paolo. Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence*. 2003, vol. 147, no. 1-2, pp. 35–84. Available from DOI: 10.1016/S0004-3702(02)00374-0.

8. CORRÊA, Augusto B.; SEIPP, Jendrik. Best-First Width Search for Lifted Classical Planning. In: KUMAR, Akshat; THIÉBAUX, Sylvie; VARAKANTHAM, Pradeep; YEOH, William (eds.). *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. AAAI Press, 2022, pp. 11–15. Available also from: `https://ojs.aaai.org/index.php/ICAPS/article/view/19780`.
9. DAUM, Jeanette; TORRALBA, Álvaro; HOFFMANN, Jörg; HASLUM, Patrik; WEBER, Ingo. Practical Undoability Checking via Contingent Planning. In: COLES, Amanda Jane; COLES, Andrew; EDELKAMP, Stefan; MAGAZZENI, Daniele; SANNER, Scott (eds.). *Proceedings of the Twenty-Sixth International Conference on Automated Planning and Scheduling, ICAPS 2016, London, UK, June 12-17, 2016*. AAAI Press, 2016, pp. 106–114. Available also from: `http://www.aaai.org/ocs/index.php/ICAPS/ICAPS16/paper/view/13091`.
10. DEMLOVÁ, Marie. *Mathematical Logic*. Vydavatelství ČVUT, 1999. ISBN 80-01-02011-8.
11. EITER, Thomas; ERDEM, Esra; FABER, Wolfgang. Undoing the effects of action sequences. *Journal of Applied Logic*. 2008, vol. 6, no. 3, pp. 380–415. Available from DOI: `10.1016/j.jal.2007.05.002`.
12. FABER, Wolfgang; MORAK, Michael; CHRPA, Lukás. Determining Action Reversibility in STRIPS Using Answer Set and Epistemic Logic Programming. *Theory and Practice of Logic Programming*. 2021, vol. 21, no. 5, pp. 646–662. Available from DOI: `10.1017/S1471068421000429`.
13. FIKES, Richard; NILSSON, Nils J. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*. 1971, vol. 2, no. 3/4, pp. 189–208. Available from DOI: `10.1016/0004-3702(71)90010-5`.
14. FREE SOFTWARE FOUNDATION, INC. *GNU Compiler Collection (g++)* [online]. 2023. Version 11.3.0 [visited on 2023-05-21]. Available from: `https://gcc.gnu.org/`.
15. GHALLAB, Malik; NAU, Dana S.; TRAVERSO, Paolo. *Automated planning - theory and practice*. Elsevier, 2004. ISBN 978-1-55860-856-6.
16. GHALLAB, Malik; NAU, Dana S.; TRAVERSO, Paolo. *Automated Planning and Acting*. Cambridge University Press, 2016. ISBN 978-1-107-03727-4. Available also from: `http://www.cambridge.org/de/academic/subjects/computer-science/artificial-intelligence-and-natural-language-processing/automated-planning-and-acting?format=HB`.
17. HELMERT, Malte. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*. 2006, vol. 26, pp. 191–246. Available from DOI: `10.1613/jair.1705`.
18. HELMERT, Malte. Concise finite-domain representations for PDDL planning tasks. *Artifical Intelligence*. 2009, vol. 173, no. 5-6, pp. 503–535. Available from DOI: `10.1016/j.artint.2008.10.013`.
19. HELMERT, Malte. The Fast Downward Planning System. *CoRR*. 2011, vol. abs/1109.6051. Available from arXiv: `1109.6051`.
20. HELMERT, Malte; SIEVERS, Silvan; ROVNER, Alexander; CORRÊA, Augusto B. On the Complexity of Heuristic Synthesis for Satisficing Classical Planning: Potential Heuristics and Beyond. In: KUMAR, Akshat; THIÉBAUX, Sylvie; VARAKANTHAM, Pradeep; YEOH, William (eds.). *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. AAAI Press, 2022, pp. 124–133. Available also from: `https://ojs.aaai.org/index.php/ICAPS/article/view/19793`.

21. ICAPS INC. *International Conference on Automated Planning and Scheduling (ICAPS) Competitions* [online]. 2023. [visited on 2023-05-21]. Available from: `https://www.icaps-conference.org/competitions/`.

22. ISO. *ISO/IEC 14882:2020: Programming Languages - C++* [online]. 2020. C++20 [visited on 2023-05-21]. Available from: `https://www.iso.org/standard/79358.html`.

23. KITWARE, INC. *CMake* [online]. 2023. Version 3.22.1 [visited on 2023-05-21]. Available from: `https://cmake.org/`.

24. KOEHLER, Jana; HOFFMANN, Jörg. On Reasonable and Forced Goal Orderings and their Use in an Agenda-Driven Planning Algorithm. *Journal of Artificial Intelligence Research*. 2000, vol. 12, pp. 338–386. Available from DOI: `10.1613/jair.715`.

25. LITTMAN, Michael L. Probabilistic Propositional Planning: Representations and Complexity. In: KUIPERS, Benjamin; WEBBER, Bonnie L. (eds.). *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Applications of Artificial Intelligence Conference, AAAI 97, IAAI 97, July 27-31, 1997, Providence, Rhode Island, USA*. AAAI Press / The MIT Press, 1997, pp. 748–754. Available also from: `http://www.aaai.org/Library/AAAI/1997/aaai97-116.php`.

26. MAUSAM; KOLOBOV, Andrey. *Planning with Markov Decision Processes: An AI Perspective*. Morgan & Claypool Publishers, 2012. Synthesis Lectures on Artificial Intelligence and Machine Learning. Available from DOI: `10.2200/S00426ED1V01Y201206AIM017`.

27. MED, Jakub; CHRPA, Lukás. On Speeding Up Methods for Identifying Redundant Actions in Plans. In: KUMAR, Akshat; THIÉBAUX, Sylvie; VARAKANTHAM, Pradeep; YEOH, William (eds.). *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. AAAI Press, 2022, pp. 252–260. Available also from: `https://ojs.aaai.org/index.php/ICAPS/article/view/19808`.

28. MORAK, Michael; CHRPA, Lukás; FABER, Wolfgang; FISER, Daniel. On the Reversibility of Actions in Planning. In: CALVANESE, Diego; ERDEM, Esra; THIELSCHER, Michael (eds.). *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*. 2020, pp. 652–661. Available from DOI: `10.24963/kr.2020/65`.

29. MUISE, Christian; BELLE, Vaishak; MCILRAITH, Sheila A. Computing Contingent Plans via Fully Observable Non-Deterministic Planning. In: *The 28th AAAI Conference on Artificial Intelligence*. 2014. Available also from: `http://www.haz.ca/papers/muise-aaai-14.pdf`.

30. MUISE, Christian; MCILRAITH, Sheila A; BECK, J Christopher. Improved Non-deterministic Planning by Exploiting State Relevance. In: *The 22nd International Conference on Automated Planning and Scheduling (ICAPS)*. 2012.

31. MUISE, Christian; MCILRAITH, Sheila A.; BELLE, Vaishak. Non-Deterministic Planning With Conditional Effects. In: *The 24th International Conference on Automated Planning and Scheduling*. 2014. Available also from: `http://www.haz.ca/papers/muise-icaps-14.pdf`.

32. NGUYEN, XuanLong; KAMBHAMPATI, Subbarao. Reviving Partial Order Planning. In: NEBEL, Bernhard (ed.). *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*. Morgan Kaufmann, 2001, pp. 459–466.

33. RICHTER, Silvia; WESTPHAL, Matthias. The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research*. 2010, vol. 39, pp. 127–177. Available from DOI: 10.1613/jair.2972.

34. RINTANEN, Jussi. Complexity of Planning with Partial Observability. In: ZILBERSTEIN, Shlomo; KOEHLER, Jana; KOENIG, Sven (eds.). *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004), June 3-7 2004, Whistler, British Columbia, Canada*. AAAI, 2004, pp. 345–354. Available also from: http://www.aaai.org/Library/ICAPS/2004/icaps04-041.php.

35. RUSSELL, Stuart J.; NORVIG, Peter. *Artificial Intelligence - A Modern Approach, Third International Edition*. Pearson Education, 2010. ISBN 978-0-13-207148-2. Available also from: http://vig.pearsoned.com/store/product/1,1207,store-12521%5C_isbn-0136042597,00.html.

Appendices

A Source Code

A source code of both framework and implementation of algorithm 2 written in C++ programming language is submitted. It is accompanied with a modified version of Planner for Relevant Policies (PRP) translator used for compilation of planning problems.