

UNIVERSITÀ DEGLI STUDI DI UDINE
DIPARTIMENTO DI SCIENZE MATEMATICHE, INFORMATICHE E FISICHE
CORSO DI LAUREA MAGISTRALE IN INFORMATICA

TESI MAGISTRALE

Non-well-founded set based multi-agent epistemic action language

CANDIDATE

Riouak Idriss

SUPERVISOR

Prof. Dovier Agostino

CO-SUPERVISOR

Dr. Fabiano Francesco

Academic Year 2022-2023

INSTITUTE CONTACTS

Dipartimento di Scienze Matematiche, Informatiche e Fisiche

Università degli Studi di Udine

Via delle Scienze, 206

33100 Udine — Italia

+39 0432 558400

<http://www.dimi.uniud.it/>

AUTHOR'S CONTACTS

To my parents,
Hayat and *El Houssine*,
without whom none of this would be possible.

Acknowledgements

I would first like to thank my supervisor Professor *Agostino Dovier* and my co-supervisor Dr. *Francesco Fabiano* for their expertise, ideas, feedback, support and patience. Their office door was always opened whenever I ran into trouble.

I would also like to thank my parents, *El Houssine* and *Hayat*, my brothers, *Bilal*, *Yassine* and *Amine*, my nephews *Joel* and *Noah*, *Marta's family*, and last but not least, *Salvatrice Pollina*, for their support.

A very big thanks to all the members of the AScI and to all my friends.

Finally, I would like to express my gratitude to *Marta*. Her support and encouragement motivated me to complete this chapter of my life.

Abstract

As the research in multi-agent domain continues to grow it is becoming more and more important to investigate the agents' relations in such systems: not only to reason about agents' perception of the world but also about agents' knowledge of her and others' knowledge. This type of study is referred as epistemic reasoning. Epistemic planning *i.e.*, *planning with epistemic reasoning*, is essential in many multi-agent domains.

In a variety of fields, e.g., economy, security, justice and politics, reasoning about others' beliefs could lead to winning strategies or help in changing a group of agents' view of the world.

In this thesis we will formalize the epistemic planning problem where the state description is based on non-well-founded set theory. The introduction of a semantics based on non-well-founded sets would permit us to characterize the planning problem without using Kripke structures, as the state-of-the-art suggests, overcoming problematic aspects such as the huge memory requirements.

Moreover, we present the language $m\mathcal{A}^p$ where states are represented by *possibilities*: structures based on non-well-founded set theory. This representation will allow us to describe the language through set-based operations and also to exploit some of the results from this field, such as the concept of bisimulation, to add important features to the multi-agent epistemic community.

Contents

List of Tables

List of Figures

Introduction to Planning

The *planning* problem consists of finding a sequence of actions, starting from an initial state, to achieve a *goal*. This sequence of actions is called *plan*. The planning problem is one of the oldest and most studied problems of Artificial Intelligence. In literature there are different instances of the planning problem; all of them sharing the same objective: given an initial configuration of the environment, find a finite sequence of permitted actions to reach the desired configuration *i.e.*, the goal, in the same environment. In this section we will briefly introduce the classical and the multi-agent planning problem.

1.1 Basic concepts

Let us review some useful notations and definitions used to describe the planning problem. For this purpose, given its simplicity, we will consider an instance of the well-known *Three-Disk Tower of Hanoi* puzzle. The puzzle consists of few simple elements:

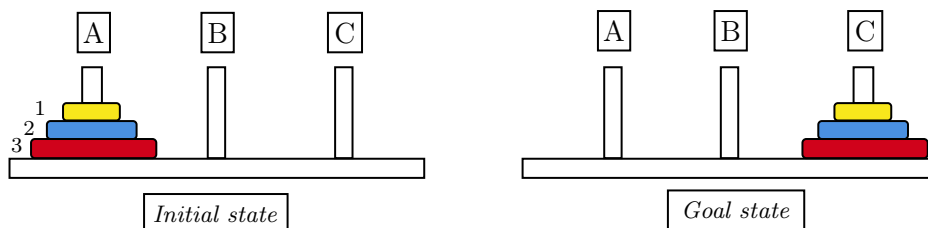


Figure 1.1: Starting and final configurations of the Tower of Hanoi with three disks.

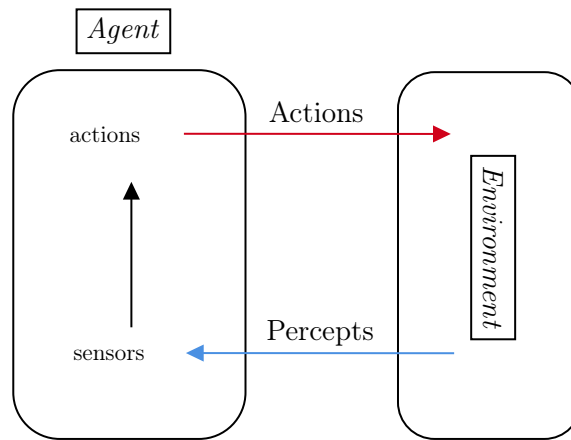


Figure 1.2: Scheme of an agent interacting with the environment through sensors and actuators.

- *Disks* of different size with a hole in their centre;
- *pegs* used to pile the disks;
- an *agent* that can move the disks.

Furthermore there are some rules that regulate this problem:

- Only one disk at time can be moved;
- a disk can be moved only if it is the top disk of a pile; and,
- a disk can never be placed on top of a smaller one.

Definition 1.1 (Agent) *An agent, as defined in [?], is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.*

In planning an agent is the element of the domain that, by performing operations on the states, tries to reach a goal. In Figure ?? is illustrated the idea of agent. In the Tower of Hanoi example, shown in Figure ??, the agent is the entity that moves the disks from one peg to another.

Definition 1.2 (State – Fluent) *A state of the domain is a configuration of the world represented as a conjunction of propositional variables, called fluents.*

Example 1.1 Let us consider the state of the Tower of Hanoi in the left of Figure ?. This one is defined as follow: `clear_one`, `clear_B`, `clear_C`, `one_on_two`, `two_on_three`, `three_on_A`. To

avoid clutters, in all the examples, the fluents that are negative in the state *e.g.*, $\neg\text{clear_A}$ and $\neg\text{one_on_three}$ are omitted.

Definition 1.3 (Action) *An action is an operation made by an agent that can change the environment configuration or her perception of it.*

An action can be performed only if certain conditions are satisfied in the current state. These conditions are called *executability conditions*.

Example 1.2 (Instance of executable and not executable actions) *Let us reconsider the Initial State in Figure ???. An example of action that cannot be executed is $\text{Move}(2, 3, C)$, which the informal semantics is to move the disk 2, currently on 3, in the peg C. The executability conditions related to this action are clear_two , two_on_three , clear_c , two_small_c but since clear_two is false in the current state, the action cannot be executed. On the other hand, an example of executable action is $\text{Move}(1, 2, C)$ since all its executability conditions are satisfied.*

Definition 1.4 (Transition function) *Given a set of states \mathcal{S} and a set of actions \mathcal{A} , a transition function γ is defined as $\gamma : \mathcal{S} \times \mathcal{A} \mapsto 2^{\mathcal{S}}$.*

In other words, the transition function γ is a function that, given a state s and an action a , returns a set of states that can represent the world after the execution of a in s . If a state s does not satisfy the executability conditions of an action a the result of the application of the transition function will be the empty set *i.e.*, $\gamma(s, a) = \emptyset$. In Figure ??? is represented the transition function γ for the Three-Disk Tower of Hanoi puzzle, where a node corresponds to a state and an edge between two nodes represents that it is possible to go from a state to another by applying an executable action.

In Computer Science, a common way to describe the main elements of a problem is through *state-transition system* [?]. These can be divided into two macro categories: *static* and *dynamic*.

Definition 1.5 (Static state-transition system) *A static state-transition system is a triple $\Sigma = \langle \mathcal{S}, \mathcal{A}, \gamma \rangle$ where*

- $\mathcal{S} = \{s_0, s_1, \dots\}$, is a finite or recursively enumerable set of states,
- $\mathcal{A} = \{a_0, a_1, \dots\}$, is a finite or recursively enumerable set of actions, and
- $\gamma : \mathcal{S} \times \mathcal{A} \mapsto 2^{\mathcal{S}}$ is a state transition function.

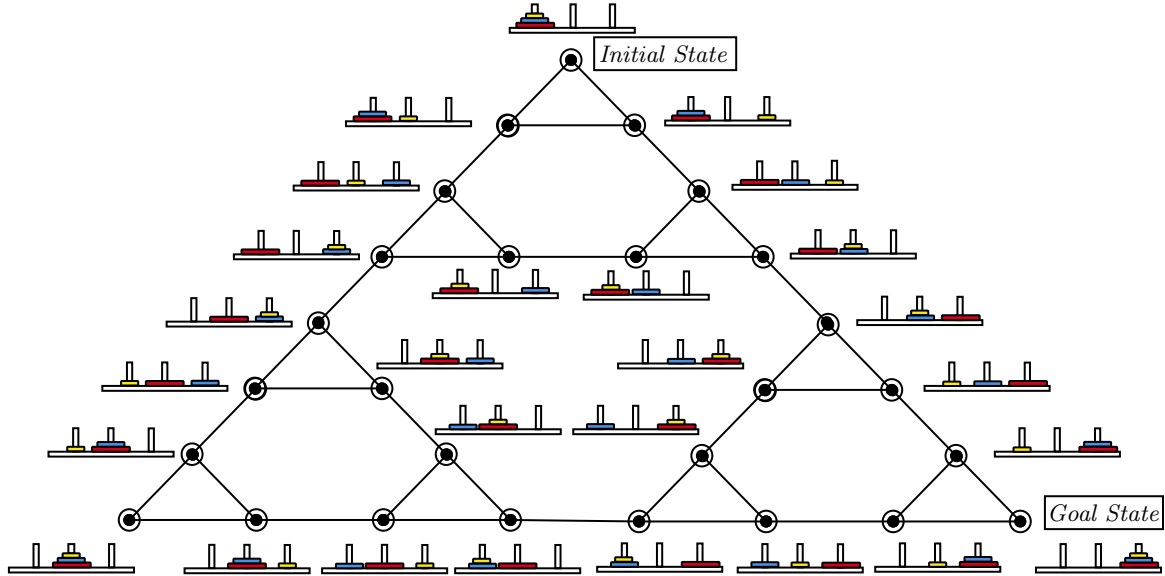


Figure 1.3: Representation of the transition function γ for the Three-Disk Tower of Hanoi Puzzle.

Definition 1.6 (Dynamic state-transition system) A dynamic state-transition system is a quadruple $\Sigma = \langle \mathcal{S}, \mathcal{A}, \mathcal{E}, \gamma \rangle$ where:

- \mathcal{S} and \mathcal{A} are as in the static state-transition system, and
- $\mathcal{E} = \{e_0, e_1, \dots\}$, is a finite or recursively enumerable set of events, and
- $\gamma : \mathcal{S} \times \mathcal{A} \times \mathcal{E} \mapsto 2^{\mathcal{S}}$ is a state transition function.

The main difference between these two state-transition systems is that the actions are controlled by the plan executor (*planner*), instead the *events* are transitions that correspond to internal dynamics of the system and therefore they are not controlled by the planner. The Tower of Hanoi it is a clear example of static state-transition system since there are no occurrences of internal events.

Now we have all the necessary notions to give the definition of *planning problem*.

Definition 1.7 (Planning problem) A planning problem [?] for a state-transition system Σ , is defined as a triple $\mathcal{P} = \langle \Sigma, \mathcal{I}, \mathcal{G} \rangle$ where:

- Σ is a state-transition system;
- $\mathcal{I} = \{i_0, i_1, \dots, i_n\}$, is a finite set of states that represents all the possible initial configurations of the problem; and

- $\mathcal{G} = \{g_0, g_1, \dots, g_n\}$, is a finite set of states, called Goal states, that represents all the possible final configurations of the problem.

The planning problem is to find a solution.

Definition 1.8 (Solution) Given a planning problem $\mathcal{P} = \langle \Sigma, \mathcal{I}, \mathcal{G} \rangle$, a solution of \mathcal{P} is a sequence of actions (a_1, a_2, \dots, a_k) such that

- $\forall s_1 \in \mathcal{I} : \gamma(s_1, a_1) \neq \emptyset$;
- $\forall s_2 \in \gamma(s_1, a_1) : \gamma(s_2, a_2) \neq \emptyset$;
- \vdots
- $\forall s_{k-1} \in \gamma(s_{k-2}, a_{k-2}) : \gamma(s_{k-1}, a_{k-1}) \neq \emptyset$;
- $\emptyset \neq \gamma(s_k, a_k) \subseteq \mathcal{G}$.

The entity that tries to solve a planning problem is called *planner* or *solver*.

1.2 Classical planning milestones

The study of single-agent domain planners dates back to the early days of Artificial Intelligence, and several languages were developed to describe the classical planning problem, such as:

- *Stanford Research Institute Problem Solver* (STRIPS, [?]): the base for most of the languages that express automated planning problem instances.
- Action description language \mathcal{A} [?]: a STRIPS extension that introduces the concept of conditional effects for actions.
- Action description languages \mathcal{B} and \mathcal{C} : are extensions of \mathcal{A} in which is possible to describe actions with indirect effects. This is expressed through statics and dynamics laws, where the first refers to a condition that has to be respected in all the states while the latter represents conditional actions effects.
- *Planning Definition Domain Language* (PDDL, [?]): the de-facto standard language for many planning systems. It could also be viewed as a type of action language.

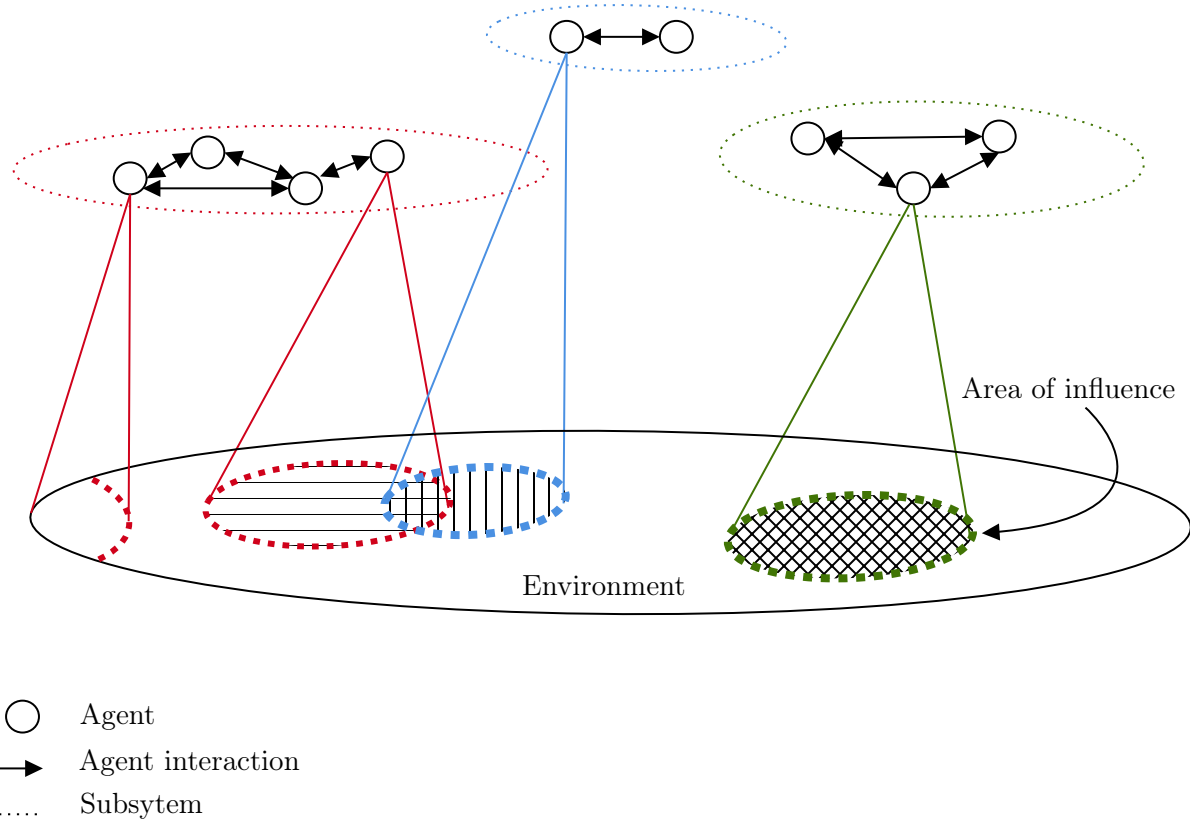


Figure 1.4: Representation of a typical multi-agent structure. The system is composed by subsystems who themselves are compounded by a set of agents that can communicate with each other through communication channels. Every agent can act and change the environment with some actuators. The range of influence of an agent action can be limited or unlimited and act throughout the environment.

1.3 Multi-agent systems

Now we will focus on those systems where there are two or more agents. As said in [?], "*There is no such things as a single-agent system*". In fact, even the most trivial real-world system is composed by a set of sub-systems that have to *cooperate* and *interact* with each other to successfully complete a task. In Figure ?? is depicted an abstract representation of a typical multi-agent system.

Maybe, at first sight, having more than one agent could seem to be a more efficient way to solve a problem, but most of the time having multiple agents increases the complexity of finding a solution for a planning problem.

Now we will give a synthetic, but sufficient for the purpose of this thesis, categorization of

multi-agent systems. A first classification is given in [?]. The authors distinguish the systems depending on the agents' coordination method, that can be:

- *Centralized*: the agents of the system are coordinated by a master process; or
- *Distributed*: where each agent acts independently.

Moreover, in [?] is provided a categorization based on the cooperation relations between agents, that can be:

- *Cooperative*: all the agents acts in synergy to reach the goal; or
- *Adversary*: the agents have different goals, and they are trying to achieve their own in a competitive way, even penalizing the other agents.

The last categorization done by [?] is about the capability of an agent to share her own information about the world. An agent can be:

- *Privacy limited*: when it is partially o completely not allowed to share information with other agents *e.g.*, for security reasons; or
- *Free*: when an agent can share all her knowledge without limitations.

During this thesis, we will focus on multi-agent systems where an agent can influence, not only the environment but also the beliefs of the other agents. This kind of problems are deeply studied by the *Multi-Agent Epistemic Planning* (MEP) community, but there are some details that need some refinements, such as the state representation or the definition of more efficient equality criteria.

2

Epistemic Reasoning

In this section we will introduce the *epistemic logic* as a framework for reasoning about *knowledge* and *beliefs*. As defined in [?, ?, ?], epistemic logic is used to describe the knowledge and belief that different agents have about the world and about the knowledge/beliefs of each other.

Let \mathcal{AG} be a set of agents and let \mathcal{F} be a set of fluents. We have that a *world* is described by a subset of elements of \mathcal{F} : intuitively, those that are ‘True’ in the world.

For each agent $\text{ag} \in \mathcal{AG}$ we associate a modal operator \mathbf{B}_{ag} that is used to represent the knowledge/belief of that agent. Moreover in epistemic logic are also introduced *group operators*, such as \mathbf{E}_{α} and \mathbf{C}_{α} , that intuitively represent the knowledge/belief of a group of agents α and the common knowledge of α respectively. To be more precise we have that:

Definition 2.1 (fluent formulae) *A fluent formula is a propositional formula built using the propositional variables in \mathcal{F} and the traditional propositional operators \wedge, \vee, \implies and \neg . We will use \top and \perp to indicate True and False, respectively.*

Definition 2.2 (fluent atom and literal) *A fluent atom is a formula composed by just an element $f \in \mathcal{F}$, instead a fluent literal¹ is either a fluent atom $f \in \mathcal{F}$ or its negation $\neg f$.*

Definition 2.3 (belief formula) *A belief formula is defined as follow:*

- *A fluent formula is a belief formula;*
- *let φ be belief formula and $\text{ag} \in \mathcal{AG}$, then $\mathbf{B}_{\text{ag}}\varphi$ is a belief formula;*

¹From now on, we will refer to ‘fluent literal’ as fluent.

- let φ_1, φ_2 and φ_3 be belief formulae, then $\neg\varphi_3$ and $\varphi_1 \text{ op } \varphi_2$ are belief formulae, where $\text{op} \in \{\wedge, \vee, \implies\}$;
- all the formulae of the form $\mathbf{E}_\alpha\varphi$ or $\mathbf{C}_\alpha\varphi$ are belief formulae, where φ is itself a belief formula and $\emptyset \neq \alpha \subseteq \mathcal{AG}$.

Intuitively, a belief formula can represent the state of the world (when is also a fluent formula) or the knowledge/belief of an agent about the world or about the knowledge/belief² of the other agents.

Example 2.1 (Belief formula) *Let us consider the formula $\mathbf{B}_{\text{ag}_1}\mathbf{B}_{\text{ag}_2}\varphi$. This formula expresses that the agent ag_1 believes that the agent ag_2 believes that φ is true, instead, $\mathbf{B}_{\text{ag}_1}\neg\varphi$ expresses that the agent ag_1 believes that φ is false.*

Example 2.2 (Common Knowledge) *Let α be a subset of elements of \mathcal{AG} and let $\text{ag} \in \mathcal{AG}$. The belief formula $\mathbf{C}_\alpha(\neg\mathbf{B}_{\text{ag}}\varphi)$ express that it is common knowledge for the agents in α that ag does not believe that φ is true.*

A classical way of providing semantics for the language of epistemic logic is in terms of Kripke structures [?]:

Definition 2.4 (Kripke structure) *A Kripke structure is a tuple $\langle S, \pi, \mathcal{B}_1, \dots, \mathcal{B}_n \rangle$, such that:*

- S is a set of worlds;
- $\pi : S \mapsto 2^{\mathcal{F}}$ is a function that associates an interpretation of \mathcal{F} to each element of S ;
- for $1 \leq i \leq n$, $\mathcal{B}_i \subseteq S \times S$ is a binary relation over S .

A *pointed Kripke structure* is a pair (M, \mathbf{s}) where M is a Kripke structure as defined above, and $\mathbf{s} \in S$ represents the entry point of the structure. As in [?], we will refer to a pointed Kripke structure (M, \mathbf{s}) as a *state* (see Definition ??).

Following the notation of [?], we will indicate with $M[S]$, $M[\pi]$, and $M[i]$ the components S , π , and \mathcal{B}_i of M , respectively.

Definition 2.5 (entailment w.r.t. a Kripke structure) *Given the belief formulae $\varphi, \varphi_1, \varphi_2$ and a pointed Kripke structure (M, \mathbf{s}) , holds that:*

²During this section we will use belief and knowledge interchangeably. We will investigate the difference between the concept of Knowledge and Belief in Section ??.

- $(M, s) \models \varphi$ if φ is a fluent formula and $M[\pi](s) \models \varphi$;
- $(M, s) \models \mathbf{B}_{\text{agi}}\varphi$ if for each t such that $(s, t) \in M[i]$ it holds that $(M, t) \models \varphi$;
- $(M, s) \models \neg\varphi$ if $(M, s) \not\models \varphi$;
- $(M, s) \models \varphi_1 \vee \varphi_2$ if $(M, s) \models \varphi_1$ or $(M, s) \models \varphi_2$;
- $(M, s) \models \varphi_1 \wedge \varphi_2$ if $(M, s) \models \varphi_1$ and $(M, s) \models \varphi_2$;
- $(M, s) \models \mathbf{E}_\alpha\varphi$ if $(M, s) \models \mathbf{B}_{\text{agi}}\varphi$ for all $\text{agi} \in \alpha$;
- $(M, s) \models \mathbf{C}_\alpha\varphi$ if $(M, s) \models \mathbf{E}_\alpha^k\varphi$ for every $k \geq 0$, where $\mathbf{E}_\alpha^0\varphi = \varphi$ and $\mathbf{E}_\alpha^{k+1}\varphi = \mathbf{E}_\alpha(\mathbf{E}_\alpha^k\varphi)$.

Intuitively, $\mathbf{E}_\alpha\varphi$ means that everyone in the group α knows φ , while $\mathbf{C}_\alpha\varphi$ means that it is commonly known in α that φ is true, which means that all the agents in α knows φ , and all of them knows that everyone knows φ , ad infinitum.

Example 2.3 (Kripke structure) In Figure ?? is represented the pointed Kripke structure (M, v_0) with $\mathcal{F} = \{p\}$ where $M[\pi](v_0) = p$ and $M[\pi](v_1) = \emptyset$.

$$(M, v_0) \equiv (\underbrace{\{v_0, v_1\}}_S, \pi, \underbrace{\{(v_0, v_1), (v_1, v_0)\}}_{B_1}, \underbrace{\{(v_0, v_1), (v_1, v_0), (v_0, v_0), (v_1, v_1)\}}_{B_2}, v_0)$$

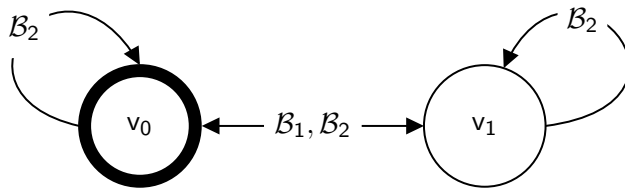


Figure 2.1: Example of a pointed Kripke structure: each circle represents a world in $M[S]$ with its name written inside it. Labelled edges between worlds denote the belief relations of the structure and the bold black circle identifies the real world. The interpretation of the world will be given whenever it is necessary. For example, we write $M[\pi](v_0) = \{p\}$ to denote that p is true in the world v_0 .

An *epistemic state* is characterized by a pointed Kripke structure that describes the state of the worlds and the beliefs of the agents. Every state in $M[S]$ represent a different world and the presence

of multiple worlds identifies uncertainty. To be more precise, given a pointed Kripke structure (M, s) we have that:

- the *actual world* (intuitively, the real world) is represented by the pointed state s ;
- the relation $(s_1, s_2) \in \mathcal{B}_i$ represents that the agent i , even if the actual state is s_1 , believes that s_2 is possible.

2.1 Axioms system and modal logics

Let us denote with $\mathcal{L}_{\mathcal{AG}}^{\mathbf{C}}$ the language of the belief formulae over the sets \mathcal{F} and \mathcal{AG} . Furthermore, we will refer with $\mathcal{L}_{\mathcal{AG}}$ the language over beliefs formulae that does not allow the use of \mathbf{C} . The language \mathcal{L}_1 describes single-agent domains (without common knowledge and belief formulae); $\mathcal{L}_{\mathcal{AG}}$ the language to reason in multi-agent domains without common knowledge, and finally $\mathcal{L}_{\mathcal{AG}}^{\mathbf{C}}$ allows to represent a MEP problem with common knowledge.

Let start analysing the complexity of each one of these languages into the details.

Definition 2.6 (Axioms system [?]) *Let $\mathcal{R}_1, \dots, \mathcal{R}_n$ be n equivalence relations and let φ and ψ be belief formulae. The axioms system is composed by:*

- $A1$ *All the tautologies of propositional calculus*
- $A2$ $(\mathcal{R}_i\varphi \wedge (\mathcal{R}_i(\varphi \implies \psi))) \implies \mathcal{R}_i\psi$ *Distribution Axiom*
 - $R1$ $(\varphi \wedge \varphi \implies \psi) \implies \psi$ *Modus Ponens Rule*
 - $R2$ $\varphi \implies \mathcal{R}_i\varphi$ *Knowledge Generalization Rule*
- $A3$ $\mathcal{R}_i\varphi \implies \varphi$ *Knowledge Axiom*
- $A4$ $\mathcal{R}_i\varphi \implies \mathcal{R}_i\mathcal{R}_i\varphi$ *Positive Introspection Axiom*
- $A5$ $\neg\mathcal{R}_i\varphi \implies \mathcal{R}_i\neg\mathcal{R}_i\varphi$ *Negative Introspection Axiom*
- $A6$ $\neg\mathcal{R}_i(\perp)$ *Consistency Axiom*

In literature, these axioms are known with different names. In particular, we have that the axiom $A2$ is called **K**, $A3$ is called **T**, $A4$ is called **4**, $A5$ is called **5** and the axiom $A6$ is called **D**. Domains that follow distinct combinations of these axioms reason on different modal logics. These logics' names is simply the concatenation of the axioms that they use. For instance, the modal logic that

uses the axioms and the rules $A1$, $A2$, $R1$ and $R2$ is called **K**; this is the logic that models the behaviour in the case of single-agent. Instead **KD45** is the modal logic that combines the axioms **K**, **D**, **4**, **5** and $A1$ with the rules $R1$ and $R2$, while the combination of **KT45** and $A1$ with the two rules $R1$ and $R2$ is called **S5**.

As said in [?], philosophers and logicians have spent years arguing which of these axioms best captures the concept of knowledge and beliefs. The question is still open but the axioms **S5** and **KD45** seems the most appropriate to represent knowledge and belief, respectively. We will focus on **KD45** since our main objective is to model the agents' beliefs.

Now we will give a briefly description of the five axioms that we introduced:

- **K**: express *logical omniscience*. In other words, the agent's knowledge is closed under logical consequence;
- **T**: has been introduced to capture the difference between knowledge and belief. This is the case for the *epistemic reasoning* where agents might believe something that does not reflect the real world;
- **D**: introduced to model the notion of belief and expresses that an agent cannot believe *false*;
- **4**: models the concept of *positive introspection*; this mean that, if $\mathcal{R}_i\varphi$ holds, than it is also true that $\mathcal{R}_i\mathcal{R}_i\varphi$;
- **5**: models the concept of *negative introspection*; this mean that, if $\neg\mathcal{R}_i$ holds, than it is also true that $\mathcal{R}_i\neg\mathcal{R}_i\varphi$.

Now we will investigate the properties of the binary relation \mathcal{R} and how these properties are related with the axioms system above.

Definition 2.7 (Equivalence relations) *Given a set of elements \mathcal{A} , a binary relation $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is an equivalence relation if and only if \mathcal{R} is:*

- Reflexive: $\forall a \in \mathcal{A} : a\mathcal{R}a$;
- Symmetric: $\forall a, b \in \mathcal{A} : (a\mathcal{R}b \implies b\mathcal{R}a)$; and,
- Transitive: $\forall a, b, c \in \mathcal{A} : ((a\mathcal{R}b \wedge b\mathcal{R}c) \implies a\mathcal{R}c)$.

Definition 2.8 (Euclidean property) *Let \mathcal{R} be a binary relation. We say that \mathcal{R} on a set S is Euclidean if, for all $s, t, u \in S$, whenever $(s, t) \in \mathcal{R}$ and $(s, u) \in \mathcal{R}$ then $(t, u) \in \mathcal{R}$.*

Definition 2.9 (Seriality property) Let \mathcal{R} be a binary relation. We say that \mathcal{R} on a set S is Serial if, for all $s \in S$, there is some t such that $(s, t) \in \mathcal{R}$.

Lemma 2.1 Given a binary relation \mathcal{R} , it holds that

- If \mathcal{R} is reflexive and Euclidean, then \mathcal{R} is symmetric and transitive.
- If \mathcal{R} is symmetric and transitive, then \mathcal{R} is Euclidean.

In Table ?? and in Table ?? is described how the properties of the binary relation \mathcal{R} and the axioms of Definition ?? are mutually related.

Property of \mathcal{R}	Axiom
Reflexive	T
Transitive	4
Euclidean	5
Serial	D
Symmetric	T - 5

Table 2.1: Relation between properties of \mathcal{R} and axioms [?].

The encoding of the axioms system in Kripke structures is done as follow: given a Kripke structure $M = \langle S, \pi, \mathcal{R}_1, \dots, \mathcal{R}_n \rangle$, for all $i \in \mathcal{AG}$ and for all $\varphi, \psi \in \mathcal{L}_{\mathcal{AG}}^C$, it holds that:

$$\mathbf{K}: M \models (\mathbf{B}_{\text{ag}_i} \varphi \wedge \mathbf{B}_{\text{ag}_i} (\varphi \implies \psi)) \implies \mathbf{B}_{\text{ag}_i} \psi$$

$$\mathbf{T}: M \models \mathbf{B}_{\text{ag}_i} \varphi \implies \varphi$$

$$\mathbf{4}: M \models \mathbf{B}_{\text{ag}_i} \varphi \implies \mathbf{B}_{\text{ag}_i} \mathbf{B}_{\text{ag}_i} \varphi$$

$$\mathbf{5}: M \models \neg \mathbf{B}_{\text{ag}_i} \varphi \implies \mathbf{B}_{\text{ag}_i} \neg \mathbf{B}_{\text{ag}_i} \varphi$$

$$\mathbf{D}: M \models \neg \mathbf{B}_{\text{ag}_i} \perp.$$

A Kripke structure is said to be a **K**-Kripke (**T**-Kripke, **4**-Kripke, **5**-Kripke, **D**-Kripke, respectively) if it satisfy the property **K**(**T**, **4**, **5**, **D**, respectively). Now we just have to axiomatize the groups operators \mathbf{E}_α and \mathbf{C}_α . Even if the operator \mathbf{C}_α is defined as infinite conjunction, its characterization is made by a finite set of axioms and rules.

MODAL LOGIC	PROPERTIES		
	REFLEXIVE	TRANSITIVE	EUCLIDEAN
K			
KT			✓
K4	✓		
K45	✓	✓	
S4	✓		✓
S5	✓	✓	✓

Table 2.2: Properties of the relation \mathcal{R} with respect the modal logics **K**, **KT**, **K4**, **K45**, **S4** and **S5**.

Definition 2.10 (Axioms for common knowledge)

- $C1 \quad \mathbf{E}_\alpha \varphi \iff \bigwedge_{i \in \alpha} \mathcal{R}_i \varphi$
- $C2 \quad \mathbf{C}_\alpha \varphi \implies \mathbf{E}_\alpha(\varphi \wedge \mathbf{C}_\alpha \varphi)$
 - $RC1 \quad (\varphi \implies \mathbf{E}_\alpha(\psi \wedge \varphi)) \implies (\varphi \implies \mathbf{C}_\alpha \psi)$

From now on, with \mathbf{K}_n (\mathbf{T}_n , $\mathbf{KD45}_n$, $\mathbf{S4}_n$, $\mathbf{S5}_n$) we will indicate the system with n agents and with \mathbf{K}_n^C (\mathbf{T}_n^C , $\mathbf{KD45}_n^C$, $\mathbf{S4}_n^C$, $\mathbf{S5}_n^C$) we indicate the same system with the addition of the axioms above.

2.2 Complexity

In this section, following [?] we recall the decidability results of the validity problem given a formula in \mathcal{L}_{AG}^C .

Definition 2.11 (Validity of a formula) *A formula is valid if and only if it is satisfied under every interpretation.*

The validity problem is *decidable* if and only if there exists an algorithm that, given as input a formula φ , will answer yes or no if the formula is valid or not respectively.

Example 2.4 (Valid formula) *An instance of a valid formula is $p \implies (p \vee q)$. In general, every tautology is valid.*

In particular, we are interested in determining if a formula φ is valid in a certain class of Kripke structures (**K**-Kripke, **T**-Kripke, **4**-Kripke, **5**-Kripke, **D**-Kripke); *i.e.*, if φ is true in every structure of that class.

Now we will consider the satisfiability problem and we will see how it is related to the validity one.

Definition 2.12 (Satisfiability) *A formula is satisfiable if it is possible to find an interpretation (model) that makes the formula true.*

It is easy to see that satisfiability and validity are complementary problems. In fact, a formula φ is valid if and only if $\neg\varphi$ is not satisfiable. Thus we can reduce the validity problem to the satisfiability one.

Before studying this kind of problem, we briefly introduce the *model checking problem* that is a sub-problem of the satisfiability and validity problems.

Definition 2.13 (Model checking w.r.t. Kripke structures) *The model checking problem consists in determining if a formula is true in a given Kripke structure.*

While in Kripke structures with a finite number of states, the model checking problem is straightforward, it is not the same for Kripke structures with an infinite number of states. There are several techniques that exploit abstraction [?] to go from an infinite to a finite number of states or that impose a limit on the maximum number of steps [?], but it is beyond the scope of this thesis to investigate Kripke structures with infinite states, therefore, we will focus only on finite Kripke structures.

Let $M = \langle S, \pi, \mathcal{B}_1, \dots, \mathcal{B}_n \rangle$ be a Kripke structure and let $\|\mathcal{M}\| = |S| + \sum_{i=1}^n |\mathcal{B}_i|$, where $|\mathcal{B}_i|$ represents the number of pairs in \mathcal{B}_i , the complexity of the model checking problem is defined as follows

Proposition 2.1 (Complexity of Model Checking problem w.r.t. Kripke structures)

As said in [?], there is an algorithm that, given a pointed Kripke structure (M, \mathbf{s}) and a formula $\varphi \in \mathcal{L}_{AG}^C$, determines, in time $O(\|\mathcal{M}\| \times |\varphi|)$, whether $(M, \mathbf{s}) \models \varphi$.

This proposition holds under the assumption that, the process of determining whether the value of a propositional variable is *true* or *false* in a state, can be computed in constant time. Notice that the proposition holds regardless of the number of agents and it continues to hold if the problem is modelled with **S5_n**-Kripke or **S4_n**-Kripke structures.

Another important result, used to demonstrate that the validity problem is decidable, is the following

Theorem 2.1 (Consistency of a formula [?]) *If a formula φ is T_n - ($S4_n$, $S5_n$, $KD45_n$)-consistent, then φ is satisfiable in a T_n -Kripke ($S4_n$ -Kripke, $S5_n$ -Kripke, $KD45_n$ -Kripke) structure with at most $2^{|\varphi|}$ states.*

With this last result, that gives us an upper bound on the number of Kripke structures to test, and we can conclude that

Corollary 2.1 *The validity problem for T_n -Kripke, $S4_n$ -Kripke, $S5_n$ -Kripke, $KD45_n$ -Kripke is decidable.*

Now, we will consider the complexity of the satisfiability problem in all the modal logic that we have consider above.

Intuitively the difficulty of determine if a formula is satisfiable is measured by the amount of time and/or space required to do this. Is not the aim of this thesis to study in detail the complexity classes therefore we will recall only their basic notions. The complexity classes that we will take into considerations are: P , NP , $PSPACE$, $EXPTIME$ and $NEXPTIME$. An algorithm, that determine if a formula is satisfiable, belongs to one of this classes if accept or reject the input in a *deterministic polynomial time*, *nondeterministic polynomial time*, *deterministic polynomial space*, *deterministic exponential time* and *nondeterministic exponential time*, respectively. We also know that $P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \subseteq NEXPTIME$, $P \neq EXPTIME$ and that $NP \neq NEXPTIME$.

In Table ?? is pointed out that in case of $S5_n$ and $KD45_n$, passing from a single-agent to multi-agent increases the logic complexity. Furthermore, is possible to see that, the addition of the common knowledge causes a further increase in complexity.

Complexity class	Modal logic
NP-complete	$S5_1, KD45_1$
PSPACE-complete	$K_n, T_n, S4_n$, with $n \geq 1$ $S5_n, KD45_n$, with $n \geq 2$
EXPTIME-complete	K_n^C, T_n^C , with $n \geq 1$ $S4_n^C, S5_n^C, KD45_n^C$, with $n \geq 2$

Table 2.3: Complexity of the satisfiability problem for the modal logics obtained by the composition of the axioms in Definition ?? and the common knowledge operator [?].

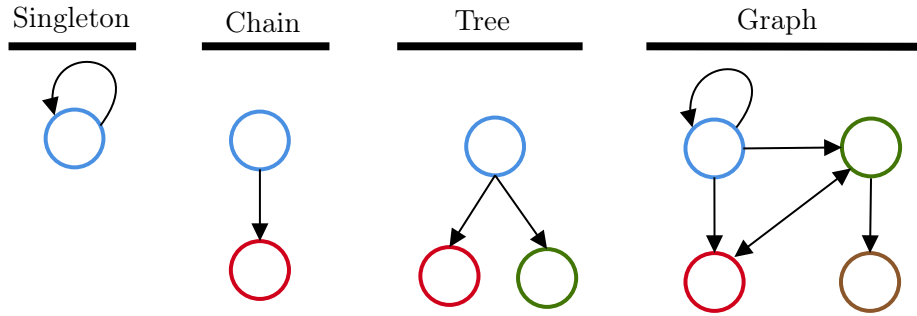


Figure 2.2: Graphical representation, from left to right, of *public announcement* (Singleton), *private announcement* (Chain and Tree) and any other type of *propositional epistemic action* (Graph). The semantics of these actions is better described in the next section.

Until now we have discussed, in a general way, the complexity of determining if a formula is satisfiable in certain modal logics. Now we will briefly consider the problem of determining the existence of a plan considering the complexity of the underlying graph of actions.

Definition 2.14 (Plan existence problem) *The plan existence problem consists of determining, if it exists, a solution for a planning problem P .*

In particular, in [?], the authors distinguish three different types of structures depending on actions:

- *Singletons*: that corresponds to public announcements of propositional facts;
- *Chains* and *Trees*: that corresponds to some kind of private announcements; and,
- *Graphs*: that capture any propositional epistemic actions.

For each one of these classes of structures we will analyse the complexity for the plan existence problem considering the preconditions expressive power. In particular, we will consider

- Non-factual actions (changing only beliefs) with propositional preconditions;
- Factual actions (changing beliefs and fluents) with propositional preconditions; and
- Factual actions with epistemic preconditions.




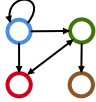
Underlying action structure	Type of actions		
	NON FACTUAL Propositional precondition	FACTUAL Propositional precondition	FACTUAL Epistemic precondition
SINGLETON 	NP-COMPLETE [?]	PSPACE-HARD [?]	PSPACE-HARD [?]
CHAIN 	NP-COMPLETE [?]	OPEN QUESTION	OPEN QUESTION
TREE 	PSPACE-COMPLETE [?]	OPEN QUESTION	OPEN QUESTION
GRAPH 	EXSPACE [?]	NON-ELEMENTARY [?]	UNDECIDABLE [?]

Table 2.4: Complexity of the *plan existence problem* [?].

AXIOMS	SINGLE-AGENT	MULTI-AGENT
K	Undecidable	Undecidable
KT	Undecidable	Undecidable
K4	Undecidable	Undecidable
K45	DECIDABLE	Undecidable
S4	Undecidable	Undecidable
S5	DECIDABLE	Undecidable

Table 2.5: Decidability results in epistemic planning [?].

In Table ?? is summarised the complexity of the plan existence problem depending on the action type and on the underlying action structure. As expected, the complexity of the problem increase as we loose restrictions on the underlying structure.

We will now analyse the decidability of the plan existence problem in an epistemic domain, considering the axioms systems of Section ?. In particular, we will study the single-agent and the multi-agent case.

In Table ?? are summarised the results of the decidability for the plan existence problem. Accordingly with the table above, we can conclude that the plan existence problem into a multi-agent domain is *undecidable*. In fact, as shown in [?], the plan existence problem can be reduced to the *halting* problem that it is well known to be *undecidable*.

In this thesis, as already told, we will focus on the *epistemic planning problem in a multi-agent domain*. In the next section, we will study a language that will allow us to describe some classes of MEP problems that, as shown in this section, are *undecidable*.

Multi-agent Epistemic Planning

Multi-agent planning and epistemic knowledge have recently gained attention from several research communities. Epistemic planners are not only interested in the state of the world but also in the knowledge (or beliefs) of the agents. Such systems, that can reason on the full extent of \mathcal{L}_{AG}^C , base their concept of state on *Kripke structures*. With respect to classical planning this introduces a big overhead: in fact to store all the necessary states, solvers require a high amount of memory. Moreover to perform operations, such as entailment or the application of the transition function, the states have been represented explicitly. That is why, as the research on epistemic reasoning advances [?, ?, ?], it is interesting to analyse alternative representations for the states that could lead to more efficient operations on the search-space.

Now we will introduce what, in our opinion, is the most complete action language for MEP that bases its states on the concept of Kripke structure: the action language $m\mathcal{A}^*$. The introduction of $m\mathcal{A}^*$ is essential since $m\mathcal{A}^\rho$ borrow the syntax and the informal semantics of the actions from it.

3.1 Related works

Over the years different multi-agent languages were designed and implemented and they can be distinguished by the ability to handle nested and common knowledge. In fact, some problems can be expressed through less expressive languages and need less powerful, and usually faster, planners. For example [?, ?] deal with problems where dynamic *common knowledge* and unbounded nested knowledge are respectively not needed. According to what we have said in Section ??, reason about other's knowledge/beliefs introduces a significant increase of complexity. On the other hand, to the best of our knowledge, only few systems [?, ?] can reasoning about epistemic knowledge in

multi-agent domains without these limitations, *i.e.*, using the language $\mathcal{L}_{\mathcal{AG}}^C$.

3.2 The action language $m\mathcal{A}^*$

The $m\mathcal{A}^*$ [?] action language is a generalization of the single-agent action languages, extensively studied in the literature [?, ?], to the case of multi-agent domains for epistemic planning. The language has a declarative, English-like syntax and an event model based semantics which permits to reason about beliefs.

The semantics of $m\mathcal{A}^*$ is based on the assumption that agents are truthful. The language is built over a signature $(\mathcal{AG}, \mathcal{F}, \mathcal{A})$, where \mathcal{AG} is a finite set of agents, \mathcal{F} is a set of fluents, and \mathcal{A} is a set of actions. The following will be used as working example through the thesis:

Example 3.1 (Three Agents and the Coin Box) *Three agents, A, B, and C, are in a room where in the middle there is a box containing a coin. The planning problem is informally described by:*

- **Domain description:**
 - *One needs the key to open the box;*
 - *In order to learn whether the coin lies heads or tails up, an agent can peek into the box, but this require the box to be open;*
 - *If one agent is looking at the box and a second agents peeks into the box, then the first agent will observe this fact and will be able to conclude that the second agent knows the status of the coin. On the other hand, the first agent's knowledge about which face of the coin is up does not change.*
 - *Distracting an agent causes her to not look at the box;*
 - *Signaling an agent causes such agent to look at the box;*
 - *Announcing that the coin lies heads or tails up will make this common knowledge among the agents that are listening.*
- **Initial state:**
 - *The box is locked;*
 - *It is common knowledge that agent A has the key of the box;*

– None of the agents know whether the coin lies heads or tails up;

- **Goal state:**

– Agent A would like to know whether the coin lies heads or tails up. She would also like to let agent B knowing that she knows this fact. However, she would like to keep this information secret from C.

- **Possible plan:** This can be achieved by: i) Distracting C from looking at the box; ii) Signaling B to look at the box if B is not looking at it; iii) Opening the box; and iv) Peeking into the box.

For this domain, we have that $\mathcal{AG} = \{A, B, C\}$, while the set of fluents \mathcal{F} consists of:

- **heads:** the coin lies heads up;
- **key(ag):** agent ag has the key of the box;
- **opened:** the box is open; and
- **look(ag):** agent ag is looking at the box.

Let $ag \in \mathcal{AG}$, the set of actions \mathcal{A} comprises:

- **open:** an agent opens the box;
- **peek:** an agent peeks into the box;
- **signal(ag):** an agent signals to agent ag to look at the box;
- **distract(ag):** an agent distracts agent ag;
- **shout_coin:** an agent announces the position of the coin.

3.2.1 Syntax

In [?], the authors distinguish between three types of actions in the following way:

- **Ontic action** (also called *World-altering*): used to modify certain properties (*i.e.*, fluents) of the world, *e.g.*, the action **open** or **distract(ag)** of Example ??.
- **Sensing action:** used by an agent to refine her beliefs about the world, *e.g.*, the action **peek**.

- *Announcement* action: used by an agent to affect the beliefs of other agents. *e.g.*, in Example ?? the action `shout_coin`.

Given an action instance $a \in \mathcal{AI}$, where \mathcal{AI} is the set of all the possible action instances $\mathcal{A} \times \mathcal{AG}$, a fluent literal $f \in \mathcal{F}$, a fluent formula ϕ and a belief formula φ we introduce the syntax adopted in $m\mathcal{A}^*$.

Executability conditions are captured by statements of the form:

$$\mathbf{executable\ a\ if\ \varphi} \tag{3.1}$$

where φ is referred as the *precondition* of a .

For a world-altering actions a , *e.g.*, distracting an agent, we have the following statement that captures the changes made by the action:

$$\mathbf{a\ causes\ f\ if\ \varphi} \tag{3.2}$$

For the sake of readability, if $\varphi = \top$ then the executability condition "if ψ " is omitted.

The statement that captures the execution of a sensing action, such as looking into the box, is the following one:

$$\mathbf{a\ determines\ f} \tag{3.3}$$

Finally announcement actions are expressed as follows:

$$\mathbf{a\ announces\ \phi.} \tag{3.4}$$

Example 3.2 Let $ag, ag_1, ag_2 \in \{A, B, C\}$. The actions of the Example ?? are specified as follow,:

- *Executability conditions:*

executable `open` $\langle ag \rangle$ **if** `key` $\langle ag \rangle$

executable `peek` $\langle ag \rangle$ **if** `opened` \wedge `look` $\langle ag \rangle$

executable `shout_coin` $\langle ag \rangle$ **if** (`B` $_{ag}$ `(heads)` \wedge `heads`) \vee (`B` $_{ag}$ `(¬heads)` \wedge `¬heads`)

executable `signal` $\langle ag_1 \rangle \langle ag_2 \rangle$ **if** `look` $\langle ag_2 \rangle$ \wedge `¬look` $\langle ag_1 \rangle$

executable `distract` $\langle ag_1 \rangle \langle ag_2 \rangle$ **if** `look` $\langle ag_1 \rangle$ \wedge `look` $\langle ag_2 \rangle$

Action type	Full observers	Partial Observers	Oblivious
World-altering	✓		✓
Sensing	✓	✓	✓
Announcement	✓	✓	✓

Table 3.1: Action type and observability relations.

- *Ontic actions:*

$\text{open}\langle\text{ag}\rangle$ **causes** opened
 $\text{signal}(\text{ag}_1)\langle\text{ag}_2\rangle$ **causes** $\text{look}(\text{ag}_1)$
 $\text{distract}(\text{ag}_1)\langle\text{ag}_2\rangle$ **causes** $\neg\text{look}(\text{ag}_1)$

- *Sensing action:*

$\text{peek}\langle\text{ag}\rangle$ **determines** heads

- *Announcement action:*

$\text{shout_coin}\langle\text{ag}\rangle$ **announces** heads

In multi-agent domains the execution of an action might change or not the beliefs of an agent. This because, in such domains, each action instance associates an observability relation to each agent. For example the agent C that becomes oblivious as distracted by the agent A, is not able to see the execution of the action $\text{open}\langle\text{A}\rangle$. On the other hand, watching an agent executing a sensing or an announcement action can change the beliefs of who is watching, *e.g.*, the agent B, who is watching the agent A sensing the status of the coin, will know that A knows the status of the coin without knowing the status herself. In Table ?? are summarized the possible observability relations for each type of action.

More formally, an agent, with respect to an action can be a

- *full observer*: completely aware of the action and its effects;
- *partial observer*: aware of the action but not of its effects; and,
- *oblivious*: completely unaware of the action execution.

Partial observability for world-altering action is not admitted as, whenever an agent is aware of the execution of an ontic action, she must know its effects on the world as well.

The observability relations are captured by the following statements:

$$\text{ag observes } a \text{ if } \varphi \quad (3.5)$$

$$\text{ag aware_of } a \text{ if } \varphi \quad (3.6)$$

The first statement indicates that the agent ag is full observer of the action a if φ holds, instead the second one indicates that the agent is a partial observer of a if the observability condition is true.

Example 3.3 *The observability relations of the actions defined in Example ??:*

$$\begin{array}{ll}
\text{ag}_1 \text{ observes open}\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_2 \text{ observes open}\langle\text{ag}_1\rangle \text{ if look}(\text{ag}_2) & \\
\text{ag}_3 \text{ observes shout_coin}\langle\text{ag}_1\rangle \text{ if } \top & \text{For } \text{ag}_3 \in \mathcal{AG} \\
\text{ag}_1 \text{ observes distract}(\text{ag}_2)\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_2 \text{ observes distract}(\text{ag}_2)\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_1 \text{ observes signal}(\text{ag}_2)\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_2 \text{ observes signal}(\text{ag}_2)\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_1 \text{ observes peek}\langle\text{ag}_1\rangle \text{ if } \top & \\
\text{ag}_2 \text{ aware_of peek}\langle\text{ag}_1\rangle \text{ if look}(\text{ag}_2) &
\end{array}$$

where ag_1 and ag_2 are different agents in \mathcal{AG} .

Next, we will give the definition of frame of reference. Frames are useful as they represent the observability relations in a given state, and we will exploit them to give a more concise definition of update models and of the transition function.

Definition 3.1 (Frame of reference) *Given an state (M, s) and an action a , let us define the following sets*

- $F_D(a, M, s) = \{\text{ag} \in \mathcal{AG} \mid [\text{ag observes a if } \varphi] \in D \wedge (M, s) \models \varphi\}$
- $P_D(a, M, s) = \{\text{ag} \in \mathcal{AG} \mid [\text{ag aware_of a if } \varphi] \in D \wedge (M, s) \models \varphi\}$
- $O_D(a, M, s) = \mathcal{AG} \setminus (F_D(a, M, s) \cup P_D(a, M, s))$

where $F_D(a, M, s)$, $P_D(a, M, s)$ and $O_D(a, M, s)$ represents the sets of agents that are fully observant, partially observant and oblivious of the execution of a in the state (M, s) , respectively. We call the tuple $(F_D(a, M, s), P_D(a, M, s), O_D(a, M, s))$ frame of reference.

Definition 3.2 (Domain in $m\mathcal{A}^*$) A domain in $m\mathcal{A}^*$ is a set of statements of the form $(?) - (?)$.

As underlined in $[?]$, an $m\mathcal{A}^*$ domain cannot contain two statements that specify incompatible effects of an action occurrence such as

$$\text{open}\langle \text{ag} \rangle \text{ causes opened if } \top \qquad \text{open}\langle \text{ag} \rangle \text{ causes } \neg \text{opened if } \top.$$

If a domain, contains incompatible statements, it is said to be *inconsistent*. In this thesis we will focus our attention only on consistent domains.

In $m\mathcal{A}^*$ the initial state is encoded through the following statement

$$\text{initially } \varphi \tag{3.7}$$

where φ is a belief formula. Intuitively, this statement says that φ is true in the initial state. One of the biggest limitation in $m\mathcal{A}^*$ is that the initial state (M, s) must be a finitary **S5**-state.

Definition 3.3 (Initial finitary S5-state $[?]$) Given a Kripke structure M and a state s , we have that (M, s) is an initial **S5**-state iff (M, s) is an initial state and M is a **S5** Kripke structure. In other words, the initial state must contains only formulae of the form:

- | | |
|---|--|
| i) φ ; | <i>All the agents knows that φ holds in the actual world</i> |
| ii) $C(\mathbf{B}_{\text{ag}}\varphi)$; | <i>All the agents knows that ag know φ</i> |
| iii) $C(\mathbf{B}_i\varphi \vee \mathbf{B}_i\neg\varphi)$; and, | <i>All the agents knows that ag know the truth value of φ</i> |
| iv) $C(\neg\mathbf{B}_i\varphi \wedge \neg\mathbf{B}_i\neg\varphi)$. | <i>All the agents knows that ag does not know the truth value of φ.</i> |

Example 3.4 *Representation of the initial state of the Example ??:*

initially $C(\text{key}(A))$
initially $C(\neg\text{key}(B))$
initially $C(\neg\text{key}(C))$
initially $C(\neg\text{opened})$
initially $C(\neg\mathbf{B}_{\text{ag}}(\text{heads}) \wedge \neg\mathbf{B}_{\text{ag}}(\neg\text{heads}))$ *with* $\text{ag} \in \mathcal{AG}$
initially $C(\text{look}(\text{ag}))$ *with* $\text{ag} \in \mathcal{AG}$
initially heads

This set of statements encode exactly the initial state of Example ??.

3.2.2 Update models and transition function

Epistemic planning relies on the notion of *update model*. Update models has been introduced by [?, ?] to describe a transformation of Kripke structures. Intuitively, an update model represents different views of an action occurrence depending on the observability of the agents. Now we will give the definition, as in [?], for each kind of actions of the relative update model.

Definition 3.4 ($\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$ -substitution) *An $\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$ -substitution is a set of substitutions $\{p_1/\varphi_1, \dots, p_k/\varphi_k\}$ where each p_i is a fluent in \mathcal{F} and each $\varphi_i \in \mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$.*

Definition 3.5 (Update Model) *Let \mathcal{AG} be a set of n agents. An update model Ξ is a tuple*

$$\langle \mathcal{E}, \mathcal{R}_1, \dots, \mathcal{R}_n, \text{pre}, \text{sub} \rangle$$

where

- \mathcal{E} is a set, whose elements are called events;
- each \mathcal{R}_i is a binary relation on \mathcal{E} ;
- $\text{pre} : \mathcal{E} \mapsto \mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$ is a function that maps each event $e \in \mathcal{E}$ to a formula in $\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$; and
- $\text{sub} : \mathcal{E} \mapsto \text{SUB}_{\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}}$ is a function that maps each event $e \in \mathcal{E}$ to a substitution in $\text{SUB}_{\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}}$, where $\text{SUB}_{\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}}$ is the set of all the $\mathcal{L}_{\mathcal{AG}}^{\mathcal{C}}$ -substitutions.

An update instance ω is a pair (Ξ, e) where Ξ is an update model and $e \in \mathcal{E}$ is the designated event.

Intuitively, the function pre defines the action preconditions while sub specifies the changes of fluent values after the execution of an action.

Now, for each kind of actions, *i.e.*, ontic, sensing and announcement, we define the relative update model as follows

Definition 3.6 (Update model for ontic actions) *Given an ontic action a with precondition ψ and a frame of reference $\rho = (F, \emptyset, O)$, the update model for a is the following tuple*

$$\langle \mathcal{E}, \mathcal{R}_1, \dots, \mathcal{R}_n, pre, sub \rangle$$

where

- $\mathcal{E} = \{\sigma, \varepsilon\}$;
- $\mathcal{R}_i = \{(\sigma, \sigma), (\varepsilon, \varepsilon)\}$ for $i \in F$ and $\mathcal{R}_i = \{(\sigma, \varepsilon), (\varepsilon, \varepsilon)\}$ for $i \in O$;
- $pre(\sigma) = \psi \wedge pre(\varepsilon) = \top$;
- $sub(\varepsilon) = \emptyset$ and $sub(\sigma) = \{p / (\Psi^+(p, a) \vee (p \wedge \neg \Psi^-(p, a)) | p \in \mathcal{F})\}$ where

$$\Psi^+(p, a) = \bigvee \{\varphi | [a \text{ causes } p \text{ if } \varphi]\}$$

and

$$\Psi^-(p, a) = \bigvee \{\varphi | [a \text{ causes } \neg p \text{ if } \varphi]\}.$$

We indicate with $\omega(a, \rho)$ the update model for a ontic action and with $(\omega(a, \rho), \{\sigma\})$ the relative instance.

We will give now the definition of the update model for sensing actions. For the sake of readability, we indicate with $Sensed(a)$ the fluent that the agent has perceiving.

Definition 3.7 (Update model for sensing actions) *Let a be a sensing action with $Sensed(a) = \{f\}$, ψ be the precondition a and $\rho = (F, P, O)$ be a frame of reference. The update model for a is given by the following tuple*

$$\langle \mathcal{E}, \mathcal{R}_1, \dots, \mathcal{R}_n, pre, sub \rangle$$

where

- $\mathcal{E} = \{\sigma, \tau, \varepsilon\}$;
- For each $i \in \{1, \dots, n\}$, \mathcal{R}_i is define as follows

$$\mathcal{R}_i = \begin{cases} \{(\sigma, \sigma), (\tau, \tau), (\varepsilon, \varepsilon)\} & \text{if } i \in F \\ \{(\sigma, \sigma), (\tau, \tau), (\varepsilon, \varepsilon), (\sigma, \tau), (\tau, \sigma)\} & \text{if } i \in P \\ \{(\sigma, \varepsilon), (\tau, \varepsilon), (\varepsilon, \varepsilon)\} & \text{if } i \in O \end{cases}$$

- The function pre is defined by

$$pre(x) = \begin{cases} \psi \wedge f & \text{if } x = \sigma \\ \psi \wedge \neg f & \text{if } x = \tau \\ \top & \text{if } x = \varepsilon \end{cases}$$

- $\forall x \in \mathcal{E} : sub(x) = \emptyset$.

We indicate with $\omega(a, \rho)$ the update model for a sensing action and with $(\omega(a, \rho), \{\sigma, \tau\})$ the relative instance.

Finally, we can give the last update models that is for announcement actions.

Definition 3.8 (Update model for announcement actions) Let ψ and ρ as above and let a be an announcement action that announces the fluent φ . The update model for a is the following tuple

$$\langle \mathcal{E}, \mathcal{R}_1, \dots, \mathcal{R}_n, pre, sub \rangle$$

where

- $\mathcal{E} = \{\sigma, \tau, \varepsilon\}$;
- For each $i \in \{1, \dots, n\}$, \mathcal{R}_i is define as follows

$$\mathcal{R}_i = \begin{cases} \{(\sigma, \sigma), (\tau, \tau), (\varepsilon, \varepsilon)\} & \text{if } i \in F \\ \{(\sigma, \sigma), (\tau, \tau), (\varepsilon, \varepsilon), (\sigma, \tau), (\tau, \sigma)\} & \text{if } i \in P \\ \{(\sigma, \varepsilon), (\tau, \varepsilon), (\varepsilon, \varepsilon)\} & \text{if } i \in O \end{cases}$$

- The function pre is defined by

$$pre(x) = \begin{cases} \psi \wedge \varphi & \text{if } x = \sigma \\ \psi \wedge \neg\varphi & \text{if } x = \tau \\ \top & \text{if } x = \epsilon \end{cases}$$

- $\forall x \in \mathcal{E} : sub(x) = \emptyset$.

We indicate with $\omega(a, \rho)$ the update model for an announcement action and with $(\omega(a, \rho), \{\sigma\})$ the relative instance.

It is immediate to see that the update model for a sensing action is structurally identical to the update model for an announcement action. The only difference, as specified in [?], is about the set of designated events in the update instance for each type of actions.

Definition 3.9 (Updates through update models) Let M be a Kripke structure and $\Xi = \langle \mathcal{E}, \mathcal{R}_1, \dots, \mathcal{R}_n, pre, sub \rangle$ be an update model. We define $M' = M \otimes \Xi$ as the updated Kripke structure obtained from M through the update model Ξ , where:

1. $M'[S] = \{(s, \xi) \mid s \in M[S] \wedge \xi \in \mathcal{E} \wedge (M, s) \models pre(\xi)\}$;
2. $((s, \xi), (s', \xi')) \in M'[i]$ iff $(s, \xi), (s', \xi') \in M'[S], (s, s') \in M[i] \wedge (\xi, \xi') \in \mathcal{R}_i$;
3. $\forall (s, \xi) \in M'[S] \wedge f \in \mathcal{F}, M'[\pi]((s, \xi)) \models f$ iff $\{f/\varphi\} \in sub(\xi) \wedge (M, s) \models \varphi$.

Finally, we can report the $m\mathcal{A}^*$ transition function Φ_D .

Definition 3.10 (Transition function Φ_D) Let (M, s) be a state and let $\mathbf{a} \in \mathcal{AL}$. The result of executing \mathbf{a} in (M, s) is a set of states, denoted by $\Phi_D(\mathbf{a}, (M, s))$ defined as follow:

- If \mathbf{a} is not executable in (M, s) then $\Phi_D(\mathbf{a}, (M, s)) = \emptyset$
- If \mathbf{a} is executable in (M, s) and (\mathcal{E}, E_d) is the representation of the occurrence of \mathbf{a} in (M, s) then
 - $\Phi_D(\mathbf{a}, (M, s)) = (M, s) \otimes (\mathcal{E}, E_d)$ if \mathbf{a} is an ontic action instance.
 - $\Phi_D(\mathbf{a}, (M, s)) = M[F_D(\mathbf{a}, M, s), f] \otimes (\mathcal{E}, E_d)$ if \mathbf{a} is a sensing action instance that senses f and $(M, s) \models f$;

- $\Phi_D(\mathbf{a}, (M, \mathbf{s})) = M[F_D(\mathbf{a}, M, \mathbf{s}), \phi] \otimes (\mathcal{E}, E_d)$ if \mathbf{a} is an announcement action instance that announces ϕ .

Next, we characterize the *epistemic planning problem* and finally, we will give the definition of *solution* for these kinds of problems.

Definition 3.11 (Epistemic planning problem) *An epistemic planning problem is a triple $(\Sigma, \mathbf{s}_0, \varphi_d)$ where $\Sigma = (S, A, \gamma)$ is a static state-transition system (see Definition ??), $\mathbf{s}_0 \in S$ and φ_d is a formula in \mathcal{L}_{AG}^C .*

A solution (there can be more than one solution) of an epistemic planning problem is a sequence of actions a_1, a_2, \dots, a_n such that $\mathbf{s}_0 \otimes a_1 \otimes a_2 \otimes \dots \otimes a_n \models \varphi_d$.

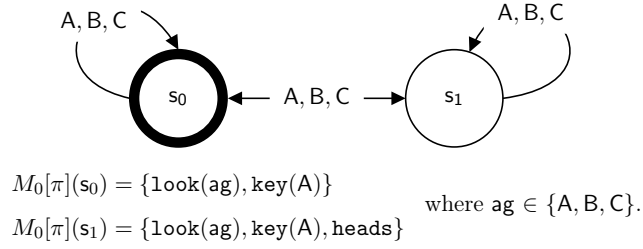
Example 3.5 (Execution of $\Delta_c = \text{distract}(C)\langle A \rangle; \text{open}\langle A \rangle; \text{peek}\langle A \rangle$) *In this example we will show the execution of the action instance sequence $\Delta_c = \text{distract}(C)\langle A \rangle; \text{open}\langle A \rangle; \text{peek}\langle A \rangle$ that leads to the desired goal, in the domain expressed in Example ??. For each state we will represent the relative Kripke structure.*

The observability relations of each action instance in Δ_c is expressed in the following Table:

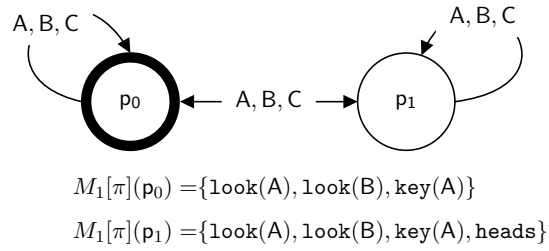
	distract(C) $\langle A \rangle$	open $\langle A \rangle$	peek $\langle A \rangle$
F_D	A, B, C	A, B	A
P_D	-	-	B
O_D	-	C	C

Table 3.2: Observability relations of the actions instances in Δ_c .

As already said, the initial state is represented by the set of statement reported in Example ??. In Figure ?? is represented the pointed **S5**-Kripke structure that encodes the initial state. Intuitively, we have that none of the agents knows if the coin lies tails or heads up, therefore no one can distinguish the states (M_0, \mathbf{s}_0) and (M_0, \mathbf{s}_1) .

Figure 3.1: The initial Kripke structure (M_0, s_0) .

In Figure ?? is represented the Kripke structure (M_0, s_0) after the execution of $\text{distract}(C)\langle A \rangle$. It is immediate to see that (M_0, s_0) and (M_1, p_0) are two structurally identical Kripke structures. The main difference is about the interpretation function π . Before the execution of the ontic action instance $\text{distract}(C)\langle A \rangle$, all the agents were looking the box while after the execution we have that only the agent A and B are looking the box.

Figure 3.2: The Kripke structure (M_1, p_0) , obtained after the execution of $\text{distract}(C)\langle A \rangle$ in (M_0, s_0) (Figure ??).

In the next picture is represented the state obtained after the execution of $\text{open}\langle A \rangle$ in (M_1, p_0) . This state represents the situation in which the agent C is not looking at the box and she does not know that the box was opened by the A and that B was a full observer of that.

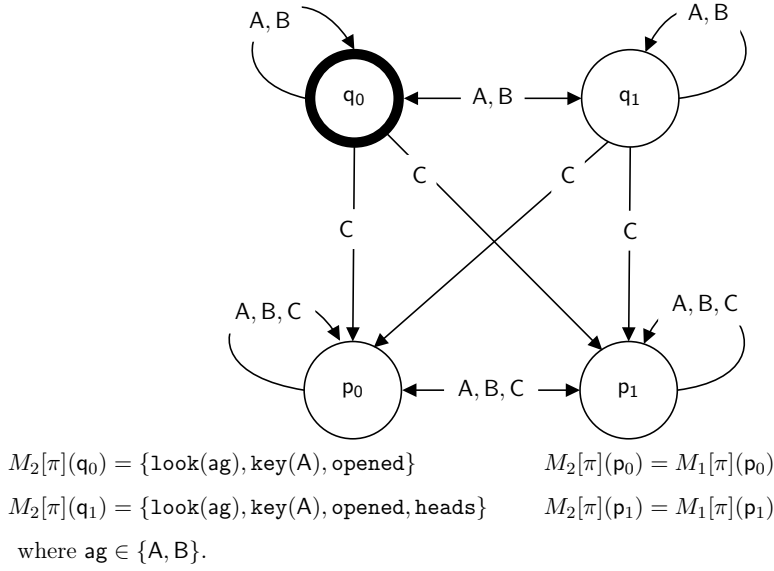


Figure 3.3: The Kripke structure (M_2, q_0) , obtained after the execution of $\text{open}\langle A \rangle$ in (M_1, p_0) (Figure ??).

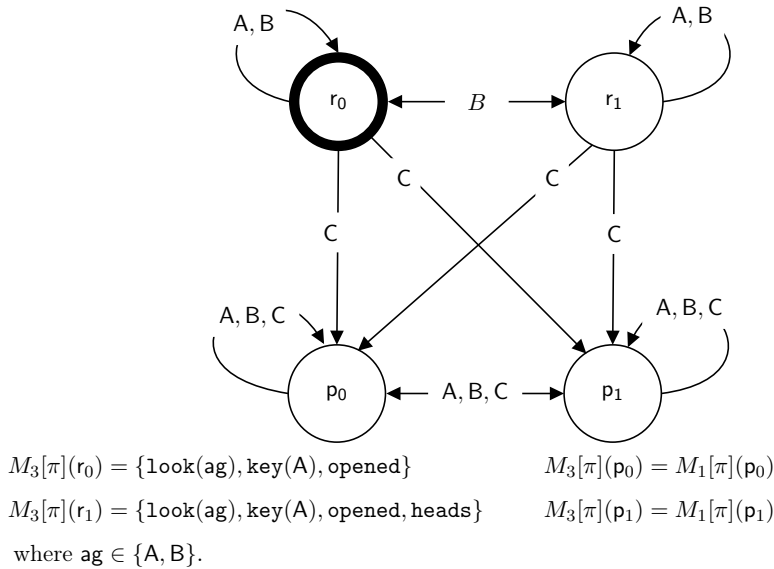


Figure 3.4: The Kripke structure (M_3, r_0) , obtained after the execution of $\text{peek}\langle A \rangle$ in (M_2, q_0) .

Finally, in Figure ?? is represented the situation in which i) A knows that the coin inside the box lies heads up; ii) B believes that A knows the coin status; and iii) C is still thinking to be in the state in which the box is closed and none of the agents know whether the coin lies heads or tails up. It is easy to see, that the desired goal, expressed through the following formulae, holds in (M_3, r_0) :

- $(M_3, r_0) \models \mathbf{B}_A \neg \text{heads} \wedge \mathbf{B}_A (\mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}))$
- $(M_3, r_0) \models \mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}) \wedge (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$
- $(M_3, r_0) \models \mathbf{B}_C (\bigwedge_{\text{ag} \in \{A, B, C\}} (\neg \mathbf{B}_{\text{ag}} \text{heads} \wedge \neg \mathbf{B}_{\text{ag}} \neg \text{heads}))$
- $(M_3, r_0) \models \mathbf{C}_{\{A, B\}} (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$

4

Introduction to Possibilities

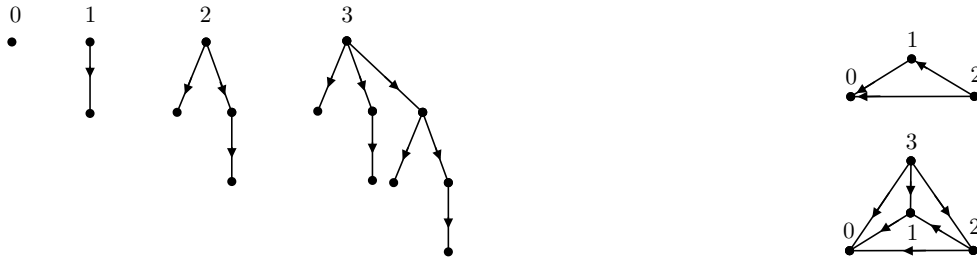
In this section, we will give the necessary notions to introduce the language $m\mathcal{A}^p$: an action description language based on the syntax and semantics of $m\mathcal{A}^*$ with the only difference that $m\mathcal{A}^p$ exploits *possibilities* to represent a state. The concept of possibility was introduced by Gerbrandy and Groeneveld in [?] to represent *multi-agent information change* and it is based on non-well-founded sets theory [?]. An interesting result reported in [?] is that there is a connection between Kripke structures and possibilities, in particular, a possibility corresponds to the whole class of bisimilar Kripke structures.

We will start by giving the basic notions about non-well-founded sets theory. We will then introduce the concepts of possibility and bisimulation.

4.1 Non-well-founded sets theory

The concept of non-well-founded sets was introduced and formalised for the first time by Dimitri Mirimanoff in [?], but for the *Zermelo-Fraenkel-Cantor Foundation Axiom* (**FA**) it was put aside. Sixty years later, Peter Aczel in [?] formalized a unified theory and showed how non-well-founded sets can be used in communication theory. In particular, he investigated the *Anti-foundation Axiom* (**AFA**), that contradicts the **FA**.

The non-well-founded sets theory has been used in different fields, in particular, i) Aczel used the **AFA** to build a model of a Milner's Synchronous Calculus of Communicating Systems (SCCS) [?]; ii) Jon Barwise in [?] used non-well-founded sets to study *mutual information* (common knowledge in distributed systems); and iii) Gerbrandy and Groeneveld in [?] used non-well-founded sets to represent epistemic logic. Now we will briefly introduce the basics of this set theory following [?]



(a) Pictures of von Neumann ordinals where $0 = \emptyset$; $1 = \{\emptyset\}$; $2 = \{\emptyset, \{\emptyset\}\}$; $3 = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}$.

(b) Alternative Pictures of von Neumann ordinals 2 and 3.

Figure 4.1: Well-founded sets represented through graphs [?].

and then we will use this concept to introduce possibilities.

We will start from the *well-founded sets theory*.

Definition 4.1 (Well-founded set) *Let E be a set, E' one of its elements, E'' any element of E' , and so on. A descent is the sequence of steps from E to E' , E' to E'' , etc. ... A set is well-founded (or ordinary) when it only gives rise to finite descents.*

This definition is captured by the *Foundation Axiom*

$$\forall a(\exists x(x \in a)) \implies \exists x(x \in a \wedge \forall y(y \in x \implies \neg(y \in a)))$$

Well-founded set theory states that all the sets in the sense of Definition ?? can be represented in the form of graphs, called *pictures*, (as shown in Figure ??).

To formalize this concept of picture of a set however it is necessary to introduce the concept of *decoration*:

Definition 4.2 (Decoration and picture)

- A decoration of a graph $\mathcal{G}(V, E)$ is a function δ that assigns to each node $n \in V$ a set δ_n in such a way that the elements of δ_n are exactly the sets assigned to successors of n , i.e., $\delta_n = \{\delta_{n'} \mid (n, n') \in E\}$.
- If δ is a decoration of a pointed graph (\mathcal{G}, n) , then (\mathcal{G}, n) is a picture of the set δ_n .

Moreover, in well-founded set theory, it holds the Mostovski's lemma:

Lemma 4.1 (Mostovski's collapsing lemma) *Each well-founded graph¹ is a picture of exactly one set.*

¹A well-founded graph is a graph that doesn't contain an infinite path $n \rightarrow n' \rightarrow n'' \rightarrow \dots$ of successors.

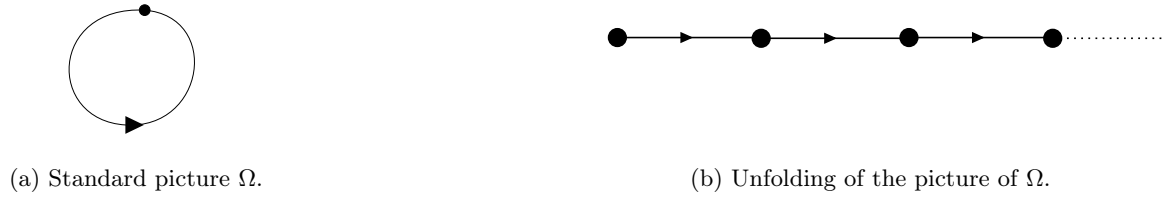


Figure 4.2: Representation of the non-well-founded set $\Omega = \{\Omega\}$.

On the other hand in [?] a *non-well-founded*, or *extraordinary set* in the sense of Mirimanoff, is a set that respects the following definition

Definition 4.3 (Non-well-founded set) *A set is non-well-founded (or extraordinary) when among its descents there are some which are infinite.*

In fact, when the **ZFC(FA)** is substituted by the *Anti-Foundation Axiom (AFA)*, expressed by Aczel in [?] as ‘*Every graph has a unique decoration*’ consequently the following sentences becomes true:

- Every graph is a picture of exactly one set (**AFA** as is formulated in [?]);
- Non-well-founded sets exist given that a non-well-founded pointed graph has to be a picture of a non-well-founded set.

In [?, ?] is pointed out how non-well-founded sets can also be expressed through systems of equations. This concept will help us to formalize the notion of state in our action language. A quick example of this representation can be derived by the set $\Omega = \{\Omega\}$ (Figure ??). We can, in fact, informally define this set by the (singleton) system of equations $x = \{x\}$.

Before defining in detail what a system of equations is, we have to introduce the concept of *atoms* or as in [?] *urelements*:

Definition 4.4 (atoms) *In set theory, atoms (or urelements) are objects that are not sets and have no further set-theoretic structure.*

Systems of equations and their solutions are described more formally, as follows

Definition 4.5 (System of equations [?]) *For each class of atoms² \mathcal{X} a system of equation in \mathcal{X} is a class τ of equations $x = \mathbf{X}$, where $x \in \mathcal{X}$ and $\mathbf{X} \subseteq \mathcal{X}$, such that τ contains exactly one equation $x = \mathbf{X}$ for each $x \in \mathcal{X}$. A solution to a system of equations τ is a function δ that assigns to each $x \in \tau(\mathcal{X})$ ³ a set δ_x such that $\delta_x = \{\delta_y \mid y \in \mathbf{X}\}$, where $x = \mathbf{X}$ is an equation of τ . If δ is the solution*

²Objects that are not sets and have no further set-theoretic structure.

³ $\tau(\mathcal{X})$ denotes the class of atoms \mathcal{X} in which τ is described.

to a system of equations τ , then the set $\{\delta_x \mid x \in \tau(\mathcal{X})\}$ is called the solution set of that system.

Since both graphs and systems of equations are representations for non-well-founded sets, it is natural to investigate their relationships. In particular it is interesting to point out how from a graph $\mathcal{G}(V,E)$ it is possible to construct a system of equation τ and vice-versa. The nodes in \mathcal{G} , in fact, can be the set of atoms $\tau(\mathcal{X})$ and, for each node $v \in V$, an equation is represented by $v = \{v' \mid (v, v') \in E\}$. Since each graph has a unique decoration, each system of equations has a unique solution. This is also true when we consider bisimilar systems of equations. In fact we can collapse them into their minimal representation thanks to the concept of maximum bisimulation [?]. Bisimilar labelled graphs (or Kripke structures) have therefore a unique solution as well since we collapse their representations into the minimal one.

4.2 Bisimulation

Bisimulation is an important notion in modal logic. In particular, bisimulation can be exploited to characterize Kripke models with ‘*the same behaviour*’. This kind of formalism is used in several fields such as, for example, Computer Science, Philosophical Logic and Set Theory.

It is not the aim of this thesis to study in depth this kind of formalism, therefore, we will give only the basics notions.

Definition 4.6 (Bisimulation [?]) *Given two pointed Kripke structures⁴ (M_1, \mathbf{s}_0) and (M_2, \mathbf{t}_0) and a set of propositional variables \mathcal{P} , the relation $R \subseteq S_1 \times S_2$ is a bisimulation if and only if for all $s \in S_1$ and $t \in S_2$, if $(s, t) \in R$ then:*

- **Atom:** $s \in \pi_1(p) \iff t \in \pi_2(p)$, for all $p \in \mathcal{P}$;
- **Forth:** For all s' such that $(s, s') \in R_1$ there exists a t' such that $(t, t') \in R_2$ and $(s', t') \in R$;
- **Back:** For all t' such that $(t, t') \in R_2$ there exists a s' such that $(s, s') \in R_1$ and $(t', s') \in R$;

Two Kripke structures (M_1, \mathbf{s}_0) and (M_2, \mathbf{t}_0) are *bisimilar* if and only if there exists a bisimulation between M_1 and M_2 starting from \mathbf{s}_0 and \mathbf{t}_0 .

Example 4.1 (Bisimilar pointed Kripke structures) *In Figure below are represented two different pointed Kripke structures (M_1, \mathbf{s}_0) (left) and (M_2, \mathbf{t}_0) (right). It is easy to see that these*

⁴ S_1, S_2 are the set of worlds, π_1, π_2 are the interpretation functions, and R_1, R_2 are the accessibility relations of M_1 and M_2 , respectively.

two Kripke structures are structurally different but bisimilar since there exists a relation $\simeq = \{(s_0, t_0), (s_1, t_1), (s_1, t_2)\}$ that is a bisimulation.

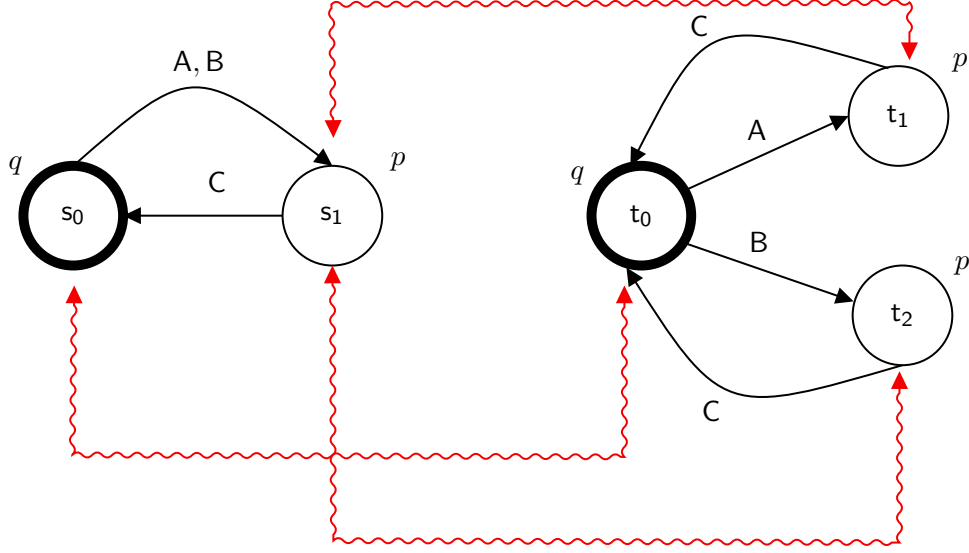


Figure 4.3: Graphical representation of two bisimilar pointed Kripke structures. The wavy edges represent the bisimulation relation \simeq .

On the other hand, if we remove the edge (t_0, t_2) from (M_2, t_0) , we obtain that the two Kripke structures are no more bisimilar.

Moreover, given two pointed Kripke structures (M_1, s_0) and (M_2, t_0) it is easy to see that

Corollary 4.1 (Truth between bisimilar Kripke structures)

$$(M_1, s_0) \simeq (M_2, t_0) \iff \left(\forall \varphi \in \mathcal{L}_{\mathcal{AG}}^C : (M_1, s_0) \models \varphi \iff (M_2, t_0) \models \varphi \right).$$

4.3 Possibilities

Now, following [?], we will introduce the concept of possibility

Definition 4.7 (Possibility [?]) Let \mathcal{AG} be a set of agents and \mathcal{P} a set of propositional variables:

- A possibility u is a function that assigns to each propositional variable $f \in \mathcal{P}$ a truth value $u(f) \in \{0, 1\}$ and to each agent $\mathbf{ag} \in \mathcal{AG}$ an information state $u(\mathbf{ag}) = \sigma$.
- An information state σ is a set of possibilities.

In Section ?? we will use this concept to describe a ‘state’ of the epistemic planning problem. The intuition behind this idea is that a possibility u is a possible interpretation of the world and of the agents’ beliefs; in fact $u(f)$ specifies the truth value of the fluent f in u and $u(A)$ is the set of all the interpretations that the agent A considers possible in u .

Definition 4.8 (Equations for possibilities) *Given a set of agents \mathcal{AG} and a set of propositional variables \mathcal{P} , a system of equations for possibilities in a class of possibilities \mathcal{X} is a set of equations such that for each $x \in \mathcal{X}$ there exists exactly one equation of the form $x(f) = i$, where $i \in \{0, 1\}$, for each $f \in \mathcal{P}$, and of the form $x(\mathbf{ag}) = \mathbf{X}$, where $\mathbf{X} \subseteq \mathcal{X}$, for each $\mathbf{ag} \in \mathcal{AG}$.*

A solution to a system of equations for possibilities is a function δ that assigns to each atom x a possibility δ_x in such a way that if $x(f) = i$ is an equation then $\delta_{x(f)} = i$, and if $x(\mathbf{ag}) = \sigma$ is an equation, then $\delta_{x(\mathbf{ag})} = \{\delta_y \mid y \in \sigma\}$.

Example 4.2 (Example of a possibility) *Instance of a system of equations. We have that r, g and b are possibilities, $\mathcal{F} = \{p, q\}$ and $\mathcal{AG} = \{A, B\}$.*

$$\begin{array}{lll}
 r(p) = 1 & g(p) = 1 & b(p) = 0 \\
 r(q) = 0 & g(q) = 1 & b(q) = 0 \\
 r(A) = \{g\} & g(A) = \{g\} & b(A) = \emptyset \\
 r(B) = \emptyset & g(B) = \{b\} & b(B) = \emptyset
 \end{array}$$

Definition 4.9 (Labelled graph [?]) *Given a set of propositional variable \mathcal{P} and a set of agents \mathcal{AG} , a labelled graph is a triple $\mathcal{G} = \left(G, \left(\frac{A}{A \in \mathcal{AG}} \rightarrow\right), V\right)$ where G is a set of nodes, for each $A \in \mathcal{AG}$, $\frac{A}{A \in \mathcal{AG}} \rightarrow \subseteq G \times G$ is a relation over G , and V is a function that assigns to each $p \in \mathcal{P}$ a subset of G .*

Moreover a possibility can be pictured as a decoration of a labelled graph and therefore as a unique solution to a system of equations for possibilities.

Definition 4.10 (Decoration of a labelled graph) *Given a labelled graph $\mathcal{G} = \left(G, \left(\frac{A}{A \in \mathcal{AG}} \rightarrow\right), V\right)$, the function δ that assigns to each node in V a possibility is a decoration if and only if for each node n it holds that:*

1. $\delta_n(p) = 1 \iff n \in V(p)$;
2. $\delta_n(A) = \{\delta_m \mid n \xrightarrow{A} m\}$.

A possibility represents the solution to the minimal system of equations in which all bisimilar systems of equations are collapsed; that is the possibilities that represent decorations of bisimilar labelled graphs are bisimilar and can be represented by the minimal one. This shows that the class of bisimilar labelled graphs and, therefore, of bisimilar Kripke structures, used by $m\mathcal{A}^*$ as states, can be represented by a single possibility.

The relation between labelled graphs and possibilities is summarised as follows

Proposition 4.1 (Labelled graphs and possibilities [?]) *The relations between labelled graphs and possibilities are summarized as follows:*

- Each possibility can be pictured by a labelled graph;
- Each labelled graph has a unique decoration;
- Two labelled graphs have the same decoration if and only if are bisimilar.

From this proposition we can conclude that a possibility is anything that can be pictured by a labelled graph and therefore by a Kripke structure.

Example 4.3 (Example of labelled graph) *Graphical representation of the unique solution δ of the system of equations of the Example ??.*

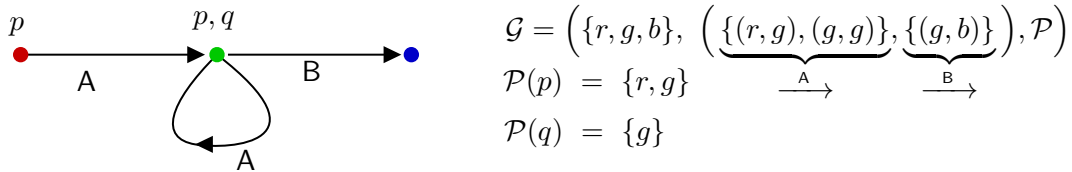


Figure 4.4: Graphical representation of the labelled graph \mathcal{G} where the nodes r, g, b are the red, green and blue ones, respectively.

It is not hard to see that a labelled graph is just a Kripke model and therefore it holds that:

Proposition 4.2 (Relation between Kripke structures and Possibilities)

- Each pointed Kripke structure has exactly one set as its solution;
- Two models are bisimilar if and only if they are picture of the same set; and,
- If (M, s) is a picture of \mathbf{a} , the for each $\varphi \in \mathcal{L}_{AG}^C : (M, s) \models \varphi \iff \mathbf{a} \models \varphi$.

Possibility as a State

The research on multi-agent epistemic domain, both in logic and planning, already comprehends several theoretical studies [?, ?, ?, ?, ?, ?, ?, ?, ?] and also a variety of solvers [?, ?, ?, ?, ?, ?] even if, at the best of our knowledge, only [?, ?] can reason without limitations on domains described by $\mathcal{L}_{\mathcal{AG}}^C$.

Anyway multi-agent epistemic solvers still have to reason on domains where the number of fluents and/or agents is very small. This is to reduce the length of the planning problems solution that otherwise would require an excessive quantity of resources (*i.e.*, time and memory). For these reasons the demand of computational resources needed (for example in respect to classical planning) is one of the central problem in MEP.

To reduce this gap several approaches can be used: i) as in [?, ?, ?, ?] the planning domain can be limited to a less expressive class of problems. On the other hand, when generality is required, ii) heuristics can effectively reduce the resolution process, as shown in [?]. Finally another approach to follow could be iii) to consider alternative representations to Kripke structures; this is what $m\mathcal{A}^p$ tries to do.

Changing the structure is especially important because different state representation can lead to a better use of the resources, and to exploit properties of the new structure to introduce important functionalities; *e.g.*, $m\mathcal{A}^p$ can rely on the concept of *bisimulation* to introduce the notion of *visited states*, being bisimulation an equality criteria for non-well-founded sets.

5.1 State representation

As main contribution of this thesis, we introduce a modified version of $m\mathcal{A}^*$, called $m\mathcal{A}^p$. The difference is in how a state is defined: in $m\mathcal{A}^*$ a state is represented as a Kripke structure while in

$$\begin{cases} w &= \{(\text{ag}, \{w, w'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}\} \\ w' &= \{(\text{ag}, \{w, w'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}, \text{heads}\} \\ v &= \{(A, \{v, v'\}), (B, \{v, v'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A)\} \\ v' &= \{(A, \{v, v'\}), (B, \{v, v'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{heads}\} \end{cases}$$

where $\text{ag} \in \{A, B\}$.

(a) System of equations for possibilities.

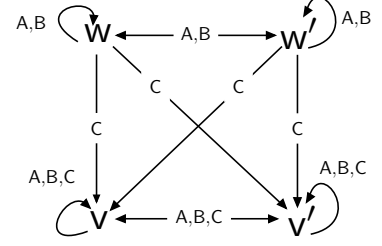
(b) Decoration of the pointed labeled graph (\mathcal{G}, w) .

Figure 5.1: Representation of the possibility w after the execution of the actions $\text{distract}(C)\langle A \rangle; \text{open}\langle A \rangle$ on the initial state of Example ???. The possibility is expanded to its system of equation for clarity.

$m\mathcal{A}^p$ a state is a possibility (Section ??). For simplicity, we have maintained the syntax used in $m\mathcal{A}^*$.

The strict connection of these two structures is highlighted in [?] from the fact that a solution to a system of equations for possibilities (Definition ??) represents a decoration for a labelled graph, which is essentially a Kripke model. In [?], it is also expressed that a ‘‘possibility corresponds with a whole class of bisimilar, but structurally different, Kripke models’’.

Let us define the usage of possibilities as states in a MEP domain where the set of agents is \mathcal{AG} and the set of fluents is \mathcal{F} . A possibility is a function that assigns to each propositional variable $f \in \mathcal{F}$ a truth value $u(f) \in \{0, 1\}$ and to each agent $\text{ag} \in \mathcal{AG}$ a set of possibilities $u(\text{ag})$.

A state in MEP has to encode the truth value of the fluents, as in classical planning, and also the beliefs of the agents about fluents and beliefs themselves. We defined in Section ?? how Kripke structures represent these information. In Figure ?? we use a possibility as a system of equations to encode a state (of the domain in Example ??).

An equation is represented in the form $u = \{(\text{ag}_1, \sigma), (\text{ag}_2, \sigma'), \dots, f, f', \dots\}$ where $\text{ag}_1, \text{ag}_2 \in \mathcal{AG}$, σ, σ' are sets of possibilities and $f, f' \in \mathcal{F}$. When we write (ag, σ) we mean that, in u , the agent ag believes that the possibilities in σ are plausible. On the other hand only if a fluent f is present in the equation this means that the fluent itself is true in u .

It is clear that a possibility correspond to a decoration of a pointed labeled graph and therefore to a unique Kripke model up to bisimulation. The representation through possibilities allows, in our opinion, a more clear and concise view of the state. That is because each state is represented by a single possibility; *e.g.*, Figure ?? is represented by $w = \{(\text{ag}, \{w, w'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}\}$

where $ag \in \{A, B\}$.

5.2 State equality through bisimulation

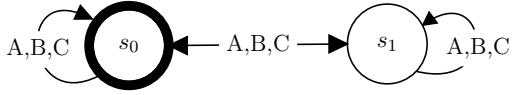
One of the reasons we chose to use possibilities as states is to exploit this new structure to introduce new functionalities to MEP. In fact having the states described as possibilities, which are strongly related to non-well-founded sets, help us to introduce the concept of *visited states*, a core idea in planning.

As said before, a possibility in the sense of decoration, represents all the Kripke structures bisimilar to the decoration itself. This means that with possibilities we can exploit bisimulation to capture the idea of equality between states. In fact, given two bisimilar decorations (or labelled graphs), these, even with structural differences, are represented by the same possibility. On the other hand this is not true when it comes to Kripke structures. This idea is best described through graphical representation, so let us consider the following example

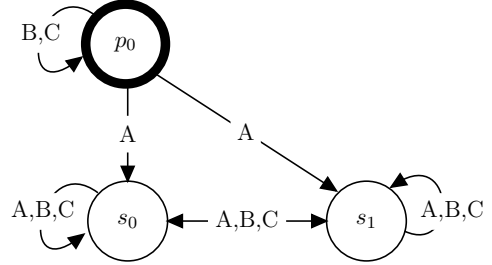
Example 5.1 (Visited states)

*As an example, let us introduce two new actions: **flip** and **tell(ag)**. The first one is an ontic action, where an agent inverts the position of the coin; the observability of this action depends on the fluents *looking*. On the other hand, **tell(ag)** means that an agent announces to **ag** the position in which she thinks the coin lies while all the other agents are oblivious.*

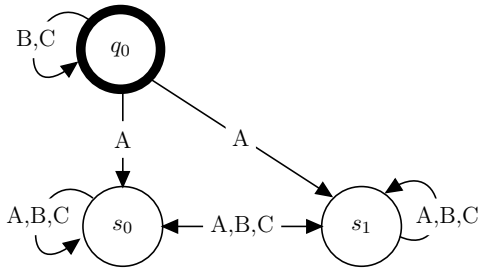
Assuming that the coin lies tails up and given a sequence of action instances $\Delta = \mathbf{peek}(B); \mathbf{distract}(B)\langle C \rangle; \mathbf{flip}(C); \mathbf{tell}(B)\langle C \rangle$ we show in Figure ?? the result of applying Δ in a slightly modified initial state of Example ?? where $C_{\{A,B,C\}}(\mathbf{opened} \wedge \neg \mathbf{look}(A))$.



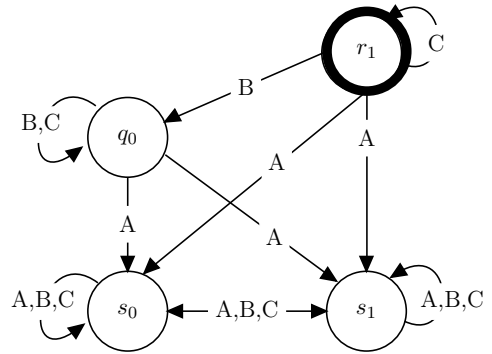
(a) Graphical representation of the initial state through the pointed Kripke structure (M_0, s_0) where $s_0 = \{\text{look}(B), \text{look}(C), \text{opened}\}$ and $s_1 = \{\text{look}(B), \text{look}(C), \text{opened}, \text{heads}\}$.



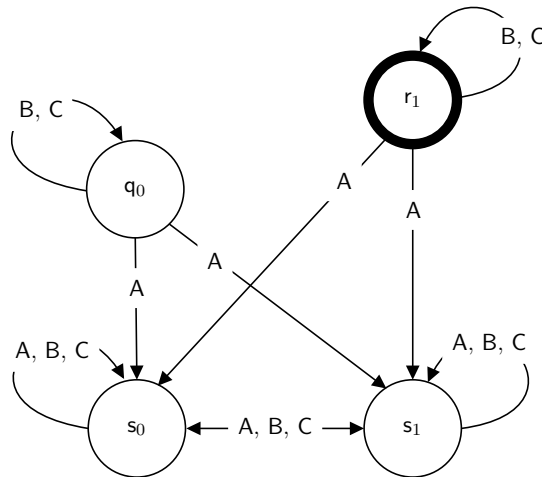
(b) The pointed Kripke structure (M_1, p_0) obtained after the execution of $\text{peek}(B)$ in (M_0, s_0) , where $p_0 = \{\text{look}(B), \text{look}(C), \text{opened}\}$.



(c) The pointed Kripke structure (M_2, q_0) obtained after the execution of $\text{distract}(B)\langle C \rangle$ in (M_1, p_0) , where $q_0 = \{\text{look}(C), \text{opened}\}$.



(d) The pointed Kripke structure (M_3, r_1) obtained after the execution of $\text{flip}(C)$ in (M_2, q_0) , where $r_1 = \{\text{look}(C), \text{opened}, \text{heads}\}$.



(e) The pointed Kripke structure (M_4, r_1) obtained after the execution of $\text{tell}(B)\langle C \rangle$ in (M_3, r_1) .

Figure 5.2: Execution of $\Delta = \text{peek}(B); \text{distract}(B)\langle C \rangle; \text{flip}(C); \text{tell}(B)\langle C \rangle$.

In Figure ??, it is represented a Kripke structure that has structural differences in respect to the one in Figure ?. This means that these two Kripke structures represent two different states in $m\mathcal{A}^*$ even if they are intuitively the same. On the contrary, if we think in term of possibilities, both the Kripke structures of Figure ? are represented by possibility

$$r_1 = \{(A, \{s_0, s_1\}), (B, \{r_1\}), (C, \{r_1\}), \text{look}(C), \text{key}(A), \text{opened}, \text{heads}\}.$$

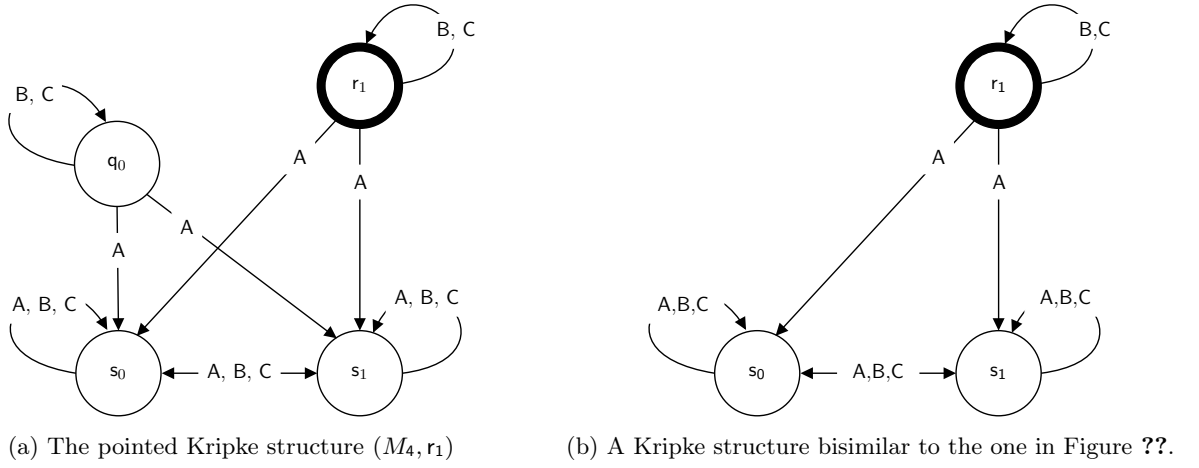


Figure 5.3: Bisimilar Kripke Structures.

Thanks to the use of possibilities we can capture bisimilar decoration while the same were considered different in $m\mathcal{A}^*$ and, therefore, check if a state has already been visited. Next we define the concept of entailment and finally the transition function in $m\mathcal{A}^p$.

We now introduce the concept of entailment w.r.t. possibilities.

Definition 5.1 (Entailment w.r.t. possibilities) *Given the belief formulae $\varphi, \varphi_1, \varphi_2$, a fluent f , an agent ag , a group of agents α , and a possibility u :*

- (i) $u \models f$ if $u(f) = 1$;
- (ii) $u \models \neg\phi$ if $u \not\models \phi$;
- (iii) $u \models \phi_1 \vee \phi_2$ if $u \models \phi_1$ or $u \models \phi_2$;
- (iv) $u \models \phi_1 \wedge \phi_2$ if $u \models \phi_1$ and $u \models \phi_2$.

(v) $u \models \mathbf{B}_{\text{ag}}\varphi$ if for each $v \in u(\text{ag})$, $v \models \varphi$;

(vi) $u \models \neg\varphi$ if $u \not\models \varphi$;

(vii) $u \models \varphi_1 \vee \varphi_2$ if $u \models \varphi_1$ or $u \models \varphi_2$;

(viii) $u \models \varphi_1 \wedge \varphi_2$ if $u \models \varphi_1$ and $u \models \varphi_2$;

(ix) $u \models \mathbf{E}_\alpha\varphi$ if $u \models \mathbf{B}_{\text{ag}}\varphi$ for all $\text{ag} \in \alpha$;

(x) $u \models \mathbf{C}_\alpha\varphi$ if $u \models \mathbf{E}_\alpha^k\varphi$ for every $k \geq 0$, where $\mathbf{E}_\alpha^0\varphi = \varphi$ and $\mathbf{E}_\alpha^{k+1}\varphi = \mathbf{E}_\alpha(\mathbf{E}_\alpha^k\varphi)$.

Now, as we have done in Section ??, we define the observability relations term of sets.

Definition 5.2 (Observability relations) *Given a possibility u and an action a , let us define the following sets*

- $F_D(a, u) = \{\text{ag} \in \mathcal{AG} \mid [\text{ag observes } a \text{ if } \varphi] \in D \wedge u \models \varphi\}$
- $P_D(a, u) = \{\text{ag} \in \mathcal{AG} \mid [\text{ag aware_of } a \text{ if } \varphi] \in D \wedge u \models \varphi\}$
- $O_D(a, u) = \mathcal{AG} \setminus (F_D(a, u) \cup P_D(a, u))$

where $F_D(a, u)$, $P_D(a, u)$ and $O_D(a, u)$ represents the sets of agents that are fully observant, partially observant and oblivious of the execution of a in the state represented by the possibility u , respectively.

5.3 Transition function

Finally we introduce the transition function for $m\mathcal{A}^p$. In defining this transition function we made some assumptions: i) we consider that the given initial state also specifies which world is the pointed one (as said in [?]), this allow us to relax the description of Φ in the case of announcements or sensing actions. Moreover ii) we do not take into consideration the case in which an agent can have *false beliefs*¹ because is still an open question how to deal with them in MEP; we will try to address this problem in future works. The transition function relative to the domain D is $\Phi_D : \mathcal{AI} \times \Sigma \rightarrow \Sigma \cup \{\emptyset\}$ where Σ is the set of all the possibilities.

For the sake of readability, we will indicate with F_D , P_D and O_D the sets $F_D(a, u)$, $P_D(a, u)$, $O_D(a, u)$, respectively., where a is the action of an action instance and u is a possibility.

¹An agent ag has a false belief about φ in state s if $s \models \varphi$ and $s \models \mathbf{B}_{\text{ag}}\neg\varphi$.

Definition 5.3 (Transition function for $m\mathcal{A}^p$) Let $\mathbf{a} \in \mathcal{AI}$, and a possibility \mathbf{u} be given. If \mathbf{a} is not executable in \mathbf{u} then $\Phi_D(\mathbf{a}, \mathbf{u}) = \emptyset$, otherwise if \mathbf{a} is executable in \mathbf{u} :

- $\Phi_D(\mathbf{a}, \mathbf{u}) = \mathbf{v}$ if \mathbf{a} is an ontic action instance and

$$\begin{cases} \mathbf{v}(\mathbf{l}) = \mathbf{u}(\mathbf{l}) & \text{if } \mathbf{l} \neq \text{caused}(\mathbf{a}) \\ \mathbf{v}(\mathbf{l}) = \text{caused}(\mathbf{a})[\mathbf{l}] & \text{if } \mathbf{l} = \text{caused}(\mathbf{a}) \end{cases}$$
 and

$$\begin{cases} \mathbf{v}(\mathbf{ag}) = \mathbf{u}(\mathbf{ag}) & \text{if } \mathbf{ag} \in O_D \\ \mathbf{v}(\mathbf{ag}) = \bigcup_{\mathbf{w} \in \mathbf{u}(\mathbf{ag})} \Phi_D(\mathbf{a}, \mathbf{w}) & \text{if } \mathbf{ag} \in F_D \end{cases}$$
- $\Phi_D(\mathbf{a}, \mathbf{u}) = \mathbf{v}$ if \mathbf{a} sensed the fluent \mathbf{l}

$$\begin{cases} \emptyset & \text{if } \text{sensed}(\mathbf{a})[\mathbf{l}] \neq \mathbf{u}(\mathbf{l}) \\ \mathbf{v}(\mathbf{ag}) = \mathbf{u}(\mathbf{ag}) & \text{if } \mathbf{ag} \in O_D \\ \mathbf{v}(\mathbf{ag}) = \bigcup_{\mathbf{w} \in \mathbf{u}(\mathbf{ag})} \Phi_D(\mathbf{a}, \mathbf{w}) & \text{if } \mathbf{ag} \in F_D \\ \mathbf{v}(\mathbf{ag}) = \bigcup_{\mathbf{w} \in \mathbf{u}(\mathbf{ag})} (\Phi_D(\mathbf{a}, \mathbf{w}) \cup \Phi_D(\neg\mathbf{a}, \mathbf{w})) & \text{if } \mathbf{ag} \in P_D \end{cases}$$
- $\Phi_D(\mathbf{a}, \mathbf{u}) = \mathbf{v}$ if \mathbf{a} announces the fluent formula ϕ

$$\begin{cases} \emptyset & \text{if } \mathbf{u} \not\models \phi \\ \mathbf{v}(\mathbf{ag}) = \mathbf{u}(\mathbf{ag}) & \text{if } \mathbf{ag} \in O_D \\ \mathbf{v}(\mathbf{ag}) = \bigcup_{\mathbf{w} \in \mathbf{u}(\mathbf{ag})} \Phi_D(\mathbf{a}, \mathbf{w}) & \text{if } \mathbf{ag} \in F_D \\ \mathbf{v}(\mathbf{ag}) = \bigcup_{\mathbf{w} \in \mathbf{u}(\mathbf{ag})} (\Phi_D(\mathbf{a}, \mathbf{w}) \cup \Phi_D(\neg\mathbf{a}, \mathbf{w})) & \text{if } \mathbf{ag} \in P_D \end{cases}$$

where:

- $\text{caused}(\mathbf{a})[\mathbf{l}]$ is the fluent modified by the action instance \mathbf{a} ;
- $\text{sensed}(\mathbf{a})[\mathbf{l}]$ represents the truth value of the fluent \mathbf{l} determined by $\mathbf{a} = \mathbf{u}(\mathbf{l})$;
- $\neg\mathbf{a}$ describes the action that senses the opposite of \mathbf{a} ; namely $\text{sensed}(\mathbf{a})[\neg\mathbf{l}]$.

Note that sensing and announcement actions generate the empty set when the action effects do not respect the fluents truth values of the possibility where the action is executed. That is because epistemic action (*i.e.*, announcement or sensing) cannot change the state of the world but only the beliefs of the agent.

As an example, we present the same sequence of action that we have done with Kripke structure, in Example ??, using possibilities as states². For readability, we will expand each possibility to its respective system of equations.

Example 5.2 (Coin in the box with possibilities)

The observability relations and the definition of the initial state are the same of the Example ??.

Let start the example with the possibility that represent the initial state.

$$\begin{cases} \mathbf{u} &= \{(\mathbf{ag}, \{\mathbf{u}, \mathbf{u}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A})\} \\ \mathbf{u}' &= \{(\mathbf{ag}, \{\mathbf{u}, \mathbf{u}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A}), \mathbf{heads}\} \end{cases}$$

where $\mathbf{ag} \in \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$.

Figure 5.4: The initial possibility \mathbf{u} .

$$\begin{cases} \mathbf{v} &= \{(\mathbf{ag}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{A}), \mathbf{look}(\mathbf{B}), \mathbf{key}(\mathbf{A})\} \\ \mathbf{v}' &= \{(\mathbf{ag}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{A}), \mathbf{look}(\mathbf{B}), \mathbf{key}(\mathbf{A}), \mathbf{heads}\} \end{cases}$$

where $\mathbf{ag} \in \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$

Figure 5.5: Possibility \mathbf{v} , obtained after the execution of $\mathbf{distract}(\mathbf{C})\langle\mathbf{A}\rangle$ in \mathbf{u} , *i.e.*, $\Phi_D(\mathbf{distract}(\mathbf{C})\langle\mathbf{A}\rangle, \mathbf{u})$.

$$\begin{cases} \mathbf{w} &= \{(\mathbf{ag}, \{\mathbf{w}, \mathbf{w}'\}), (\mathbf{C}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A}), \mathbf{opened}\} \\ \mathbf{w}' &= \{(\mathbf{ag}, \{\mathbf{w}, \mathbf{w}'\}), (\mathbf{C}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A}), \mathbf{opened}, \mathbf{heads}\} \end{cases}$$

where $\mathbf{ag} \in \{\mathbf{A}, \mathbf{B}\}$ and \mathbf{v}, \mathbf{v}' , are the possibilities represented in Figure ??.

Figure 5.6: Possibility \mathbf{w} , obtained after the execution of $\mathbf{open}\langle\mathbf{A}\rangle$ in \mathbf{v} , *i.e.*, $\Phi_D(\mathbf{open}\langle\mathbf{A}\rangle, \mathbf{v})$.

$$\begin{cases} \mathbf{z} &= \{(\mathbf{A}, \{\mathbf{z}\}), (\mathbf{B}, \{\mathbf{z}, \mathbf{z}'\}), (\mathbf{C}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A}), \mathbf{opened}\} \\ \mathbf{z}' &= \{(\mathbf{A}, \{\mathbf{z}'\}), (\mathbf{B}, \{\mathbf{z}, \mathbf{z}'\}), (\mathbf{C}, \{\mathbf{v}, \mathbf{v}'\}), \mathbf{look}(\mathbf{ag}), \mathbf{key}(\mathbf{A}), \mathbf{opened}, \mathbf{heads}\} \end{cases}$$

where $\mathbf{ag} \in \{\mathbf{A}, \mathbf{B}\}$ and the possibilities \mathbf{v}, \mathbf{v}' are represented in Figure ??.

Figure 5.7: Possibility \mathbf{z} , obtained after the execution of $\mathbf{peek}\langle\mathbf{A}\rangle$ in \mathbf{w} , *i.e.*, $\Phi_D(\mathbf{peek}\langle\mathbf{A}\rangle, \mathbf{w})$.

²In Appendix A can be found the simultaneous execution of the ‘*coin in the box*’ example with both the states representation.

As for Kripke structures, we have that, the desired goal, holds in the possibility z :

- $z \models \mathbf{B}_A \neg \text{heads} \wedge \mathbf{B}_A \left(\mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}) \right)$
- $z \models \mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}) \wedge (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$
- $z \models \mathbf{B}_C \left(\bigwedge_{ag \in \{A, B, C\}} (\neg \mathbf{B}_{ag} \text{heads} \wedge \neg \mathbf{B}_{ag} \neg \text{heads}) \right)$
- $z \models \mathbf{C}_{\{A, B\}} (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$

6

Conclusions

In this thesis, we have investigated an alternative to Kripke structures as representation for multi-agent epistemic planning states. In fact, inspired by [?] we have exploited non-well-founded set theory to define the new action language $m\mathcal{A}^\rho$: an action language for MEP based on possibilities. The semantics of $m\mathcal{A}^\rho$ relies on the notion of possibility that is used to encode the world status and the agents' knowledge or beliefs.

Moreover, we have exploited possibilities to define a stronger concept of equality between states collapsing, all the bisimilar Kripke structures into the same possibility. Thanks to that, recognizing an already visited state can be done by simply checking the structural composition of the state itself. Finally, given that in $m\mathcal{A}^\rho$ the state representation is more compact, it is possible to reduce the search-space dimension.

Furthermore, we have investigated and analysed the multi-agent epistemic problem, studying the complexity of the more relevant problems, such as

- the explicit state representation through pointed Kripke structures. As we have seen, a single epistemic state is represented with a huge (the size can be exponential in the number of agents and fluents) Kripke structure;
- the *satisfiability* and *validity* problem of epistemic formulae, that are EXPSPACE-COMPLETE problems; and,
- the UNDECIDABILITY of the plane existence problem in the case where $|\mathcal{AG}| \geq 2$.

6.1 Future works

As an immediate continuation of this thesis and of the work that we have done in [?], we will implement a planner that is parametric with respect to the state representation since we want to consider other alternatives to Kripke structures and possibilities such as *OBDDs* [?, ?].

On the other hand, as further developments to this thesis we plan to

- study the expressiveness of the *distributed knowledge* operator [?];
- try to exploit more set-based operations: especially for the entailment of group operators \mathbf{E}_α and \mathbf{C}_α ;
- formalize the concept of *non-consistent* belief for $m\mathcal{A}^\rho$;
- adding the concept of static laws. As now $m\mathcal{A}^*$ and $m\mathcal{A}^\rho$ deals only with dynamic laws;
- investigate more thoroughly the connection between Kripke structures and non-well-founded sets;
- consider other heuristics to reduce the search space as in [?];
- consider symbolic representations of the state.

Appendix

7.1 $m\mathcal{A}^*$ and $m\mathcal{A}^p$ comparison

For a more clear comparison we show the execution, on both $m\mathcal{A}^*$ and $m\mathcal{A}^p$, of the action instances sequence $\Delta_c = \text{distract}(C)\langle A \rangle; \text{open}\langle A \rangle; \text{peek}\langle A \rangle$, that leads to the desired goal, in the domain expressed in Example ???. With this we want to give a graphical explanation of both the transition functions and state-space defined by the two languages. Each state in $m\mathcal{A}^*$ will be represented by a Kripke structure while in $m\mathcal{A}^p$ will be a possibility (expanded to its respective system of equations for clarity).

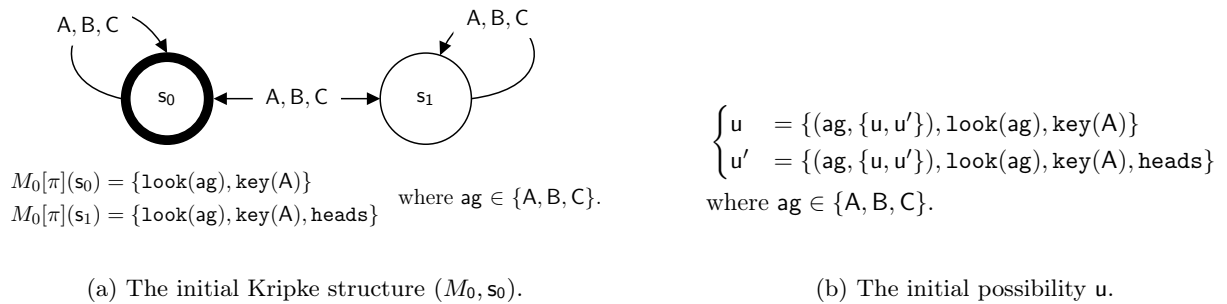
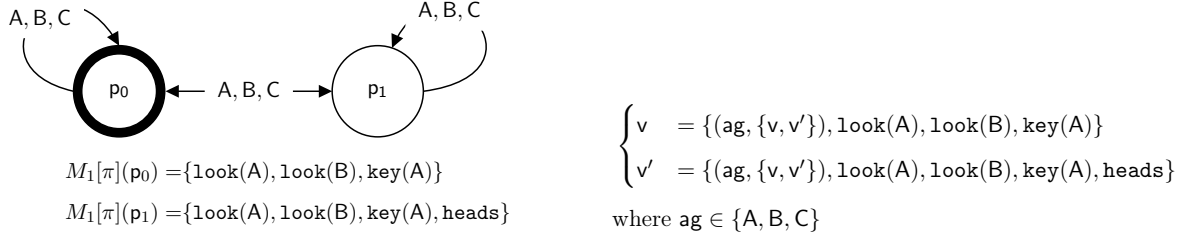


Figure 7.1: The initial state.



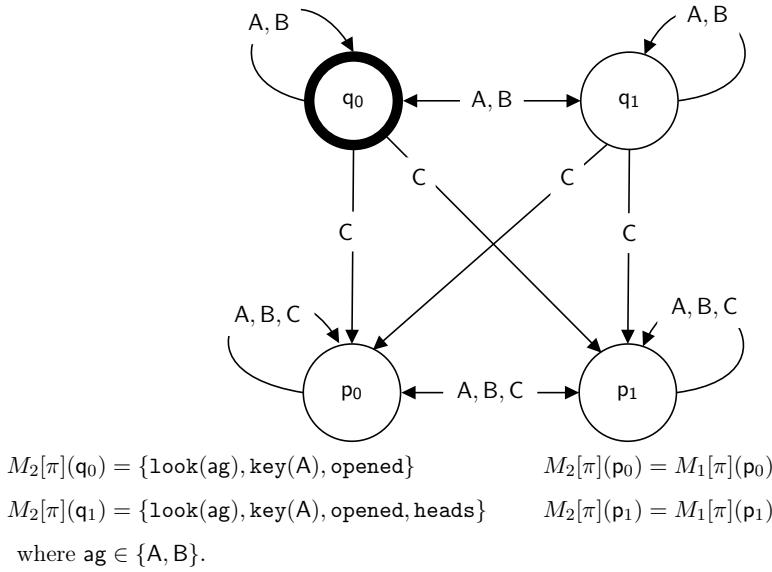
$$\begin{cases} v = \{(\text{ag}, \{v, v'\}), \text{look}(A), \text{look}(B), \text{key}(A)\} \\ v' = \{(\text{ag}, \{v, v'\}), \text{look}(A), \text{look}(B), \text{key}(A), \text{heads}\} \end{cases}$$

where $\text{ag} \in \{A, B, C\}$

(a) The Kripke structure (M_1, p_0) , obtained after the execution of $\text{distract}(C)\langle A \rangle$ in (M_0, s_0) .

(b) Possibility v , obtained after the execution of $\text{distract}(C)\langle A \rangle$ in u .

Figure 7.2: Execution of $\text{distract}(C)\langle A \rangle$.



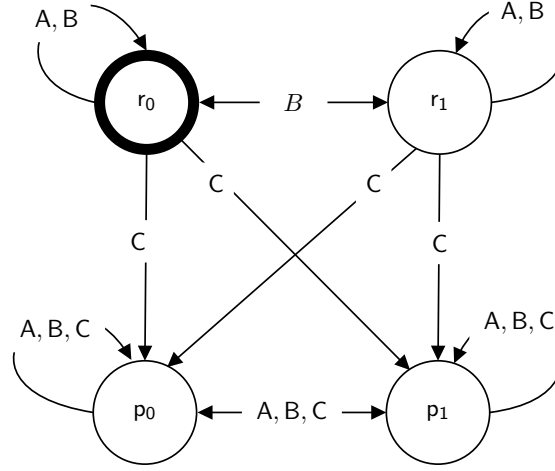
(a) The Kripke structure (M_2, q_0) , obtained after the execution of $\text{open}\langle A \rangle$ in (M_1, p_0) .

$$\begin{cases} w = \{(\text{ag}, \{w, w'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}\} \\ w' = \{(\text{ag}, \{w, w'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}, \text{heads}\} \end{cases}$$

where $\text{ag} \in \{A, B\}$ and v, v' are the possibilities represented in Figure ??.

(b) Possibility w , obtained after the execution of $\text{open}\langle A \rangle$ in v .

Figure 7.3: Execution of $\text{open}\langle A \rangle$.



$$\begin{aligned}
 M_3[\pi](r_0) &= \{\text{look}(\text{ag}), \text{key}(A), \text{opened}\} & M_3[\pi](p_0) &= M_1[\pi](p_0) \\
 M_3[\pi](r_1) &= \{\text{look}(\text{ag}), \text{key}(A), \text{opened}, \text{heads}\} & M_3[\pi](p_1) &= M_1[\pi](p_1)
 \end{aligned}$$

where $\text{ag} \in \{A, B\}$.

(a) The Kripke structure (M_3, q_0) , obtained after the execution of $\text{peek}(A)$ in (M_2, q_0) .

$$\begin{cases}
 z &= \{(A, \{z\}), (B, \{z, z'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}\} \\
 z' &= \{(A, \{z'\}), (B, \{z, z'\}), (C, \{v, v'\}), \text{look}(\text{ag}), \text{key}(A), \text{opened}, \text{heads}\}
 \end{cases}$$

where $\text{ag} \in \{A, B\}$ and the possibilities v, v' are represented in Figure ??.

(b) Possibility z , obtained after the execution of $\text{peek}(A)$ in w .

Figure 7.4: Execution of $\text{peek}(A)$.

Finally the goal that both the state in Figure ?? entail is expressed with the following formulae:

- $\mathbf{B}_A \neg \text{heads} \wedge \mathbf{B}_A (\mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}))$
- $\mathbf{B}_B (\mathbf{B}_A \text{heads} \vee \mathbf{B}_A \neg \text{heads}) \wedge (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$
- $\mathbf{B}_C (\bigwedge_{\text{ag} \in \{A, B, C\}} (\neg \mathbf{B}_{\text{ag}} \text{heads} \wedge \neg \mathbf{B}_{\text{ag}} \neg \text{heads}))$
- $\mathbf{C}_{\{A, B\}} (\neg \mathbf{B}_B \text{heads} \wedge \neg \mathbf{B}_B \neg \text{heads})$